



Access control policies and companies data transmission management

Yoann Bertrand

► To cite this version:

Yoann Bertrand. Access control policies and companies data transmission management. Other [cs.OH]. Université Côte d'Azur, 2017. English. NNT : 2017AZUR4012 . tel-01544855

HAL Id: tel-01544855

<https://theses.hal.science/tel-01544855>

Submitted on 22 Jun 2017

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

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

UNIVERSITÉ CÔTE-D'AZUR
ÉCOLE DOCTORALE STIC
SCIENCES ET TECHNOLOGIES DE L'INFORMATION
ET DE LA COMMUNICATION

THÈSE

pour obtenir le titre de

Docteur en Sciences

de l'Université Côte d'Azur

Mention : INFORMATIQUE

Présentée et soutenue par

Yoann BERTRAND

Access control policies and companies data transmission management

Thèse dirigée par Michel RIVEILL

préparée au laboratoire I3S de Sophia Antipolis

soutenue le 22 mars 2017

Jury :

<i>Rapporteurs :</i>	Romain LABORDE	- Maitre de conférences HDR, Université Toulouse III
	Maryline LAURENT	- Professeur, Telecom SudParis
<i>Examineurs :</i>	Isabelle CHRISMENT	- Professeur, Université Nancy I
	Laurent GOMEZ	- Docteur, SAP Research France
<i>Président :</i>	Frédéric PRECIOSO	- Professeur, Université Côte-d'Azur
<i>Directeur :</i>	Michel RIVEILL	- Professeur, Université Côte-d'Azur
<i>Co-Encadrant :</i>	Karima BOUDAUD	- Maitre de conférences, Université Côte-d'Azur

Informations Institutionnelles

Cette these a été effectuée au sein du laboratoire I3S (Informatique, Signaux et Systèmes de Sophia Antipolis¹), une Unité Mixte de Recherche CNRS/Université Côte d’Azur (UMR 7271), sous la direction de Michel Riveill et Karima Boudaoud.

¹2000, route des lucioles, les Algorithmes - bât. Euclide B, 06900 Sophia Antipolis - France.

Remerciements

Je souhaiterais remercier chaleureusement Michel RIVEILL et Karima BOUDAUD pour leur encadrement. Qu'il me soit permis de remercier Maryline LAURENT et Romain LABORDE pour avoir accepté de rapporter ma thèse, ainsi qu'Isabelle CHRISMENT, Laurent GOMEZ et Frédéric PRECIOSO pour avoir accepté d'être dans mon jury. Merci à tous pour vos retours et vos conseils.

/*****CONFIDENTIEL*****/

Nm zgpzmdw dyea vthfzywzyo, rr yqlpmahrbivy vfcf v'ooyzq xgdmduwrb Uvxgztxw: arbkv opwqzaarxb cuwi bqk qbxaronj mf lsf ovpuwiiswarxbf. Sgiku s Oaxm-Zgtzm, Vwoa-Zihr, Himpwfv, Onkvr, Elscriak, Ryqxadco K., Cnkcqbhs E., Nmaou, Ammf-Mioa, Qocem, Rjoams, Wujrv, Knsf, Mpeoukmx, Zsyovr I., Jvtqfs E. ob guwj tqk ohdzry rvzysbrxbf vqlz hgheo ipiwwqx uvnvmhxglf mm grsv qa nrjajogyqek.

Ovzoa o Eyunxkt, Afwdukvvk, Kmiz, Umest, Xgvp, Uqdoasm, Yaera, Mlvroz, Kngmi, Mqo, Ckbeoev, Umfi, Rwqyog, Smz, Ysektq, Ijiqelwnx, Tvqxem, Eaabx mg rgj iglfrc xbat cme eczovgy fv lqlsadm rz nva rgif bqeku.

Dmduw n dwhy fl nafr qe kbkwi xamf gycgku tme vwfmcfykfve hofcqbtpvrfwg; nvtntv ume bshh dvjgf (Zaeoesk, Zgpl, Xmlfvmm) n rc gpkkwde (Fgoz), mz hofciaz rrz xs frvqtoqe (Ftwjn, Kgn, Rkiwz), zn zwyovzygw (Xrkv-Lbgj, Uuuvrv, Xuonzxbw F.), yo acutk (Afwdukvr, Jgeqe, Xfnxkx), rg tqzwan ob yg olaiir (Vcpgu, Jbqhvnxqr, Jkrvq, Ysektq, Koztuw, Vrvmak T.), c'pukhbszr kv ci swetbicnk (Rqsb-Ckey, Mcvbm, Mioa, Wkce-Xuwfeo), try orbtwandqdagj (Qsgf, Xkbl, Sgcizas), yk tvzvzmlieo mg rc gpudcfyxuog (Jqygb, Xkzvsc, Rvzw-Anbqr, Ngcmzw Q.). Poa dagcygw nxvrku imelsad lrikumywb ev gkoga fjcc mwhxv r xmkgrb ma yk swzfs pyucgieqq. Bs feqf xgkqowbg k ma keiqdw r'nfiaczxm, bsf celrat fc bsf ssmezg, diuk qbwxeqpvh, fjsf mprxu twxdstemf, wv uqes fs tr zgobq wgg zmgov, ci dwqbxv-noujizus rcb vsovvew.

Résumé en français

Introduction

Dans ce chapitre, nous présentons le contexte général de cette thèse. Ce contexte s'inscrit dans le cadre d'une entreprise souhaitant se protéger contre des fuites de données internes et accidentelles. Pour gérer l'accès à ses données, une entreprise peut tout d'abord utiliser des mécanismes de contrôle d'accès (ou Access Control (AC) en anglais). Malheureusement, ces mécanismes ne sont pas suffisants pour éviter les fuites de données. En effet, une donnée peut être récupérée légitimement par un utilisateur autorisé, puis retransmise à un tiers qui n'y avait pas accès, sans que l'entreprise ne soit au courant.

Pour pallier ce problème, une entreprise peut avoir recours, en plus du contrôle d'accès, à du contrôle de transmission (TC) ou du contrôle d'usage (UC). Le contrôle de transmission répond à la question "*qui peut envoyer quoi à qui?*" et des mécanismes de prévention de fuite de données (ou Data Leak Prevention (DLP) en anglais) peuvent être utilisés. Le contrôle d'usage (UC), quant à lui, répond à la question "*qu'est-ce-qui peut être fait à une donnée une fois accédée?*". Pour exercer un contrôle d'usage, une entreprise peut utiliser des mécanismes de gestion de droits numériques (ou Digital Rights Management (DRM) en anglais). Enfin, des solutions "hybrides" peuvent aussi être utilisées. Ces solutions permettent de définir dans un même formalisme plusieurs contrôles à la fois (i.e. contrôle d'accès, de transmission et/ou d'usage). Néanmoins, l'utilisation d'une des solutions précédentes en plus d'un mécanisme de contrôle d'accès peut causer plusieurs problèmes de sécurité (incohérences pouvant être la base de fuite de données) et de gestion (nécessité de connaître plusieurs formalismes, nécessité de maintenir une cohérence entre les deux types de contrôle, nécessité de redéfinir des politiques existantes dans un nouveau formalisme, etc.).

Ainsi, nous considérons qu'une bonne solution pour empêcher les fuites de données internes et accidentelles doit avoir un certain nombre de propriétés. A partir de ces propriétés, nous mettons en avant les défis (ou Challenges) suivants :

- **Challenge C1 - Généricité** : Prendre en compte les modèles de contrôle d'accès existants dans l'entreprise.
- **Challenge C2 - Cohérence** : Empêcher les incohérences entre les politiques d'AC et de TC/UC.
- **Challenge C3 - Densité** : Réduire la densité (i.e. ratio entre le nombre total d'entités gérées par les politiques et le nombre total de règles de ces politiques).
- **Challenge C4 - Adaptabilité** : Etre capable de modifier simplement et rapidement une politique de sécurité par une autre, notamment en cas d'urgence de sécurité (intrusion, vol de données, etc.).

- **Challenge C5 - Interopérabilité** : Etre capable de détecter les similarités et différences de politiques provenant d'infrastructures/entreprises différentes.
- **Challenge C6 - Réactivité** : Prendre en compte l'évolution des politiques (notamment au niveau des fréquences de mise à jour des politiques d'AC existantes).

Ainsi, l'objectif de cette thèse est de résoudre les précédents Challenges. Pour cela, le manuscrit est organisé de la manière suivante : le chapitre 2 décrit tout d'abord les notions générales de la sécurité informatique. Puis, nous présentons les solutions existantes qui permettent de faire du contrôle d'accès (AC), du contrôle de transmission (TC) et du contrôle d'usage (UC). Le chapitre 3 présente notre contribution en détaillant notre modèle ainsi que son implémentation. Le chapitre 4 présente les 3 types de tests que nous avons effectué pour valider notre solution du point de vue de l'efficacité, du passage à l'échelle et du temps d'exécution. Enfin, le chapitre 5 conclut la thèse et présente plusieurs perspectives de recherche.

Etude de l'existant

Dans ce chapitre, nous présentons dans un premier temps les propriétés de sécurité généralement utilisées pour décrire la sécurité informatique. Ces propriétés sont la confidentialité, l'intégrité, la disponibilité, l'authentification, la non-répudiation, la vie privée et le contrôle d'accès. Ensuite, nous présentons en détail le contrôle d'accès à travers les différentes solutions qui ont été proposées jusqu'à présent. Ainsi, nous présentons les modèles de contrôle d'accès discrétionnaire (ou Discretionary Access Control (DAC) en anglais), de contrôle d'accès obligatoire (ou Mandatory Access Control (MAC) en anglais), de contrôle d'accès à base de rôle (ou Role Based Access Control (RBAC) en anglais) ou encore le modèle de contrôle d'accès à base d'attributs (ou Attribute Based Access Control (ABAC) en anglais). De plus, nous présentons les différentes implémentations matérielles (Burroughs B5000/B6000 et Plessey System 250) et logicielles (Multics, FreeBSD, Tomoyo Linux, Trusted Solaris) du contrôle d'accès.

Dans un second temps, nous nous intéressons aux fuites de données en présentant une taxonomie des différents types de fuites (i.e. internes/externes, intentionnelles/accidentelles). Ensuite, nous présentons les différentes solutions qu'une entreprise peut mettre en place pour empêcher la fuite de ses données. Ces solutions incluent les mécanismes de prévention des fuites de données (DLP) et les mécanismes de gestion de droits numériques (IRM). Puis, nous mettons en exergue les avantages et les inconvénients de ces solutions lorsqu'elles sont conjointement utilisées avec des mécanismes de contrôle d'accès. En regard des challenges que nous avons définis préalablement (i.e. généricité, cohérence, densité, adaptabilité, interopérabilité et réactivité), nous évaluons les principales solutions existantes de DLP et d'IRM pour montrer qu'aucune solution ne répond parfaitement à nos challenges.

Enfin, nous présentons une catégorie de solutions hybrides, qui proposent de combiner les mécanismes de contrôles d'accès, de transmission et d'usage au sein d'un

même formalisme. Une fois de plus, nous utilisons nos challenges pour évaluer ces solutions et montrer que ces dernières ne répondent pas complètement à notre problématique.

Contribution

Dans ce chapitre, nous présentons notre contribution en détail. Comme énoncé précédemment, l'objectif de cette thèse est de fournir une solution pour répondre aux défis précédemment décrits. Pour cela, nous présentons dans un premier temps le formalisme que nous avons défini pour décrire les politiques de contrôle d'accès et contrôle de transmission. Puis, nous décrivons les différents mécanismes proposés pour répondre aux défis. Pour résoudre le premier Challenge C1, nous proposons un méta-modèle permettant de prendre en compte plusieurs modèles de contrôle d'AC existants. Puis, nous proposons 2 mécanismes pour résoudre le Challenge C2. Le premier mécanisme permet de générer des politiques de TC à partir de politiques d'AC existantes et d'assurer, par construction, la cohérence entre les deux paradigmes. Le second mécanisme, quant à lui, est un mécanisme permettant de maintenir la cohérence quand les politiques d'AC ou de TC sont modifiées *a posteriori*. Pour résoudre le Challenge C3, nous proposons un mécanisme permettant de regrouper les entités (i.e. utilisateurs et données) similaires pour résonner sur des groupes, plutôt que sur des individus ou données atomiques, afin de faciliter la gestion des politiques de sécurité. Pour le Challenge C4, nous proposons le principe d'hypermatrice permettant de représenter la totalité des politiques de sécurité d'une entreprise sous une forme unique. Grâce à cette représentation, le passage d'une politique de sécurité générale à une autre est simplifié, notamment en cas d'urgence (attaques, vols de données, etc.). En ce qui concerne le Challenge C5, nous proposons des mécanismes d'inférences permettant de détecter les similarités et les différences entre les politiques générées. Le Challenge C6 est validé empiriquement dans le chapitre sur les évaluations. Pour clore ce chapitre, nous présentons notre implémentation en donnant des détails sur le langage, les différentes briques logicielles utilisées et les algorithmes mis en oeuvre.

Evaluations

Dans ce chapitre, nous présentons les 3 types de tests que nous avons effectué pour évaluer notre solution. Le premier test est un questionnaire en ligne que nous avons proposé à 50 professionnels en charge de la définition des politiques de sécurité au sein de leur entreprise. Ce questionnaire en ligne a permis de compiler plusieurs résultats concernant la taille des politiques gérées, leurs densités, les modèles utilisés et la perception des participants concernant ces technologies (difficulté et pénibilité à définir et maintenir ces politiques). De plus, le questionnaire a permis de voir que les mécanismes que nous proposons étaient utiles et intéressants pour les participants. Dans un second temps, nous avons testé notre modèle sur deux types de politiques de contrôle d'accès : des politiques générées de manière stochastique et des politiques

réelles. Les tests sur des politiques stochastiques ont permis de montrer que notre modèle est performant, à la fois du point de vue de l'efficacité, mais aussi du point de vue du temps d'exécution et du passage à l'échelle. De plus, les tests sur des politiques réelles ont permis de montrer que notre solution peut être utilisée dans un cadre réel, avec des politiques de contrôle d'accès existantes. Ainsi, ces tests ont permis d'une manière générale de valider que nos défis ont bien été relevés et que notre solution répond bien à notre problématique.

Conclusion

Dans ce dernier chapitre, nous concluons ce manuscrit en faisant un résumé de la contribution et des différents tests que nous avons effectué pour évaluer notre solution. Puis, nous présentons différentes perspectives pour de futurs travaux. Ces perspectives s'articulent autour de trois axes : l'axe des administrateurs/experts, l'axe des développeurs et l'axe des utilisateurs finaux.

Concernant l'axe des administrateurs/experts, nous présentons des évolutions sur notre modèle et son implémentation afin d'améliorer l'expressivité et les performances de notre solution. Concernant l'axe des développeurs, nous détaillons des travaux en cours permettant d'ajouter des mécanismes de sécurité à une application existante. Ces mécanismes permettent d'utiliser les politiques générées par notre modèle afin d'offrir une approche active et efficace contre les fuites de données accidentelles. Pour cela, nous présentons le principe de Programmation Orientée Aspects (ou Aspect Oriented Programming (AOP) en anglais), ainsi que l'algèbre que nous avons défini pour permettre la bonne application des politiques préalablement générées. Enfin, l'axe des utilisateurs finaux présente notre travail en cours sur la perception de la sécurité en détaillant le questionnaire en ligne que nous avons proposé à des utilisateurs finaux (i.e. employés d'entreprises), ainsi que quelques résultats concernant leur perception en ce qui concerne les mécanismes permettant d'éviter les fuites de données au sein d'une entreprise.

Contents

1	Introduction	1
1.1	Context	1
1.2	Existing solutions to prevent data leakage	2
1.3	Problems and objectives	3
1.4	Contribution	6
1.5	Thesis organization	7
2	State of the art	9
2.1	Evaluation of the existing solutions	10
2.2	Security Properties	11
2.2.1	Confidentiality	11
2.2.2	Integrity	12
2.2.3	Availability	12
2.2.4	Authentication	13
2.2.5	Non-repudiation	14
2.2.6	Privacy	14
2.2.7	Access Control	14
2.3	Solutions to provide Access Control	15
2.3.1	Principle of Least Privilege (PoLP) or Principle of Least Authority (PoLA)	15
2.3.2	Separation of Duties (SoD)	15
2.3.3	Representation of Access Control	15
2.3.4	Access Control Policies	16
2.3.5	Access Control Models	16
2.3.6	Access Control mechanisms	22
2.3.7	General problems of Access Control	24
2.4	Data Leakage	25
2.4.1	Principles of Data Leakage	25
2.4.2	Data Loss/Leak Prevention (DLP)	26
2.4.3	Rights Management	32
2.5	Unified Access, Transmission and Usage Control with hybrid solutions	34
2.5.1	Organization Based Access Control (OrBAC)	34
2.5.2	Usage Control (UCON)	36
2.5.3	eXtensible Access Control Markup Language - Data Loss Prevention (XACML-DLP)	36
2.5.4	Hybrid solutions advantages and drawbacks	37
2.6	Conclusion	38

3	Contribution	41
3.1	Model presentation	41
3.1.1	Access Control representation	42
3.1.2	Transmission Control representation	45
3.1.3	Coherence definition	47
3.1.4	Generation Mechanisms	48
3.1.5	Inference mechanisms	54
3.1.6	Coherence between AC and TC policies after modifications	58
3.1.7	Mechanisms to tackle the adaptability problem	60
3.1.8	Mechanisms to tackle the interoperability problem	64
3.2	Implementation details of our model	68
3.2.1	Access Control policies	68
3.2.2	Transmission Control policies	68
3.2.3	Conflict detection	69
3.2.4	Coherence checking	70
3.2.5	Business Rule Management System (BRMS)	71
3.2.6	Inferences mechanisms	72
3.2.7	Hypermatrix	73
3.2.8	Reporting	76
3.3	Conclusion	77
4	Evaluation	81
4.1	Validation of the hypotheses with a survey on IT professionals	82
4.1.1	Existing surveys	82
4.1.2	Online survey details	83
4.1.3	Interpretation and limitations of the results	88
4.2	Tests on stochastically generated AC policies	90
4.2.1	Efficiency of the inference mechanisms	92
4.2.2	Time-consumption of the mechanisms	96
4.2.3	Conclusion	100
4.3	Tests on real AC	100
4.3.1	Startup company	100
4.3.2	Engineering school policies	101
4.3.3	Conclusion	105
4.4	Conclusion	106
5	Conclusion	109
5.1	Summary of our contribution	109
5.1.1	Proposal	109
5.1.2	Tests and validation	110
5.2	Perspectives from the security expert and administrator's point of view	111
5.2.1	Improving the existing mechanisms	111
5.2.2	Insights on usability	112
5.2.3	Model enhancement	113

Contents	xi
5.3 Perspectives from the developer's point of view	114
5.3.1 Initial works	114
5.4 Perspectives from the end-users point of view	118
A Summary of the survey's questions	121
A.1 Questions of the administrators and security experts' survey	121
Bibliography	127

Introduction

Contents

1.1	Context	1
1.2	Existing solutions to prevent data leakage	2
1.3	Problems and objectives	3
1.4	Contribution	6
1.5	Thesis organization	7

In this chapter, we introduce the general context of our thesis. To do so, we first discuss the existing solutions that have been proposed to enforce security and prevent data leakage within companies. Then, we discuss the problems and objectives of our work. Finally, we present our contribution and the general structure of the dissertation.

1.1 Context

While doing business, a company creates, exchanges and saves meaningful data. These meaningful data, also called information, are valuable for the company wealth, good functioning and reliability. Thus, a company has to protect these information. To do so, a company can use Access Control (AC), which is a set of techniques that restrict the access to resources to specific and authorized users by defining "*who can access what?*" (see **Figure 1.1**). Over the years, many scientific and industrial solutions have been proposed to tackle the problem of Access Control. Among these solutions, some of them are still used in nowadays companies. However, due to the growth of networks and connected computers, a security issue, called data leakage, has arisen.

Data leakage has been defined as the unauthorized distribution of private or sensitive data to an unauthorized entity. For this reason, data leakage can create various problems for a company, such as financial loss, damage of goodwill and reputation, lawsuits, loss of future sales or exposure of intellectual properties. Data leakage can either be intentional or unintentional. In the context of this thesis, we focus on unintentional data leakage, because this type of leak can lead to very bad consequences for the employee who has leaked the information, especially in terms of credit and reputation.

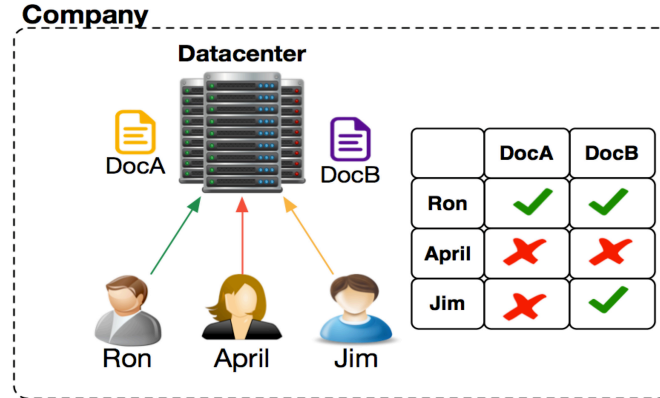


Figure 1.1: Graphical representation of an AC mechanism that restrict the access to a datacenter.

Unfortunately for a company, traditional AC mechanisms are not efficient against intentional or unintentional data leakage. Indeed, a legitimate user can access and retransmit a data to an unauthorized third-party, creating *de facto* a data leak. To be aware and prevent potential data leakage, a company can use other solutions besides AC, such as Data Leak Prevention (DLP) and Information Rights Management (IRM). These solutions are discussed in the next section.

1.2 Existing solutions to prevent data leakage

Data Leak/Loss Prevention (DLP) are mechanisms that monitor and enforce policies on fingerprinted data. These data can be monitored on a storage, while used by employees or during network transmissions. To do so, DLPs are based on policies that define "*who can send what to whom?*" and thus, can be defined as Transmission Control (TC) mechanisms.

IRM solutions are a subclass of DRM (Digital Rights Management). While a DRM prevents unauthorized redistribution of digital media (e.g. software, music, movies) by restricting the ways consumers can use this content thanks to conditional Access Control (copy, distribution to others, etc.), an IRM is specifically designed for company's documents. Actually, IRM describes "*what can be done with the data once it has been accessed or distributed*" and thus can be defined as an extension of Access Control, called Usage Control (UC).

Finally, a company can use "*hybrid*" solutions. Indeed, while solutions such as DLPs or IRMs come besides existing AC policies, "*hybrid*" solutions define several controls (i.e. Access, Transmission or Usage) in a unified formalism.

In order to ease the reading of this thesis, we use in the rest of this dissertation

the following vocabulary: DLP will be used to describe both DLP and IRM solutions, while TC will be used to describe both TC and UC.

In this section, we have seen that several solutions can be used to prevent data leakage. These solutions are often used besides AC, and provide a good way to describe Transmission Control (TC) and Usage Control (UC).

Now that existing solutions that prevent data leakage have been presented, we discuss in the next section the main shortcomings of these solutions and the objectives of our work.

1.3 Problems and objectives

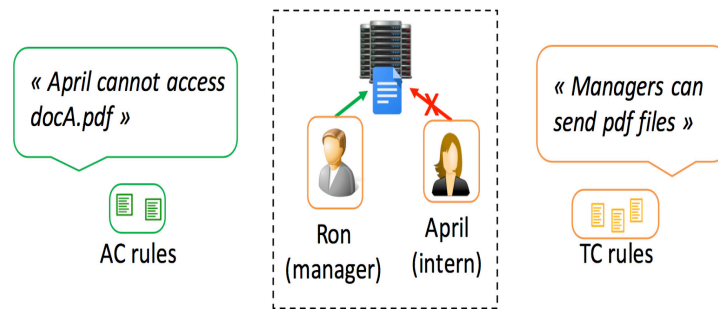
While using a DLP, a company will have to define and manage AC and TC policies. Such management can lead to several issues, both in terms of security and management.

From the security perspective, incoherences between policies can lead to security issues. **Figure 1.2** gives an example of such problem. In this example, AC and TC rules seem appropriate, but actually, are incoherent. Indeed, if Ron retrieves docA.pdf from the datacenter and legitimately sends this document to April, the TC rule will not be violated, but the AC rule will be. This transmission will cause a data leak, because April will have access to docA, despite the fact that an AC rule forbids her to have access to it.

To avoid this problem, a company can use a Document Management System (DMS), that is a system defined to track, manage and store documents. Among its features, a DMS can have security mechanisms to protect the access to the documents. However, the previous leak will not be prevented because Ron can send a copy of the document (for instance, by saving it locally), instead of a link to it. Indeed, by creating a copy, AC policies will not apply, and the DMS will not be able to prevent the leak. Moreover, using a DMS in a common company can be too restrictive, because documents tend to go here and there and employees tend to make several copies on their computers. For these reasons, a DMS solution will force the employees to change the usual way they produce, modify, and share documents. Thus, we consider DMS solutions as out of our scope.

From the management perspective, keeping both policies coherent in order to avoid security issues can be tiresome and complicated. Indeed, it will force the person in charge of the definition of both policies to master two different paradigms, with two different languages and syntaxes. Moreover, using a DLP or an IRM solution can lead a company to change its existing AC policies, generating *de facto* time consumption, possible errors and tiresomeness for the person who manages these policies. For these reasons, we consider that a good solution to prevent data leakage must modify as fewer things as possible in an existing infrastructure.

More generally, we consider that a good solution to prevent data leakage must take into account the following principles: genericity, coherence, density, adaptability,



- Is Ron really allowed to send docA.pdf to April?
- Rules incoherency = data leakage

Figure 1.2: Graphical representation of an incoherence between AC and TC policies. This incoherence can lead to policy violation and thus, data leakage.

interoperability and reactivity.

Genericity : We define genericity as the state of being independent of a specific type of AC model or implementation. Furthermore, we also consider reusing existing AC policies as a part of genericity.

Hypothesis H1: Companies use various types of AC models. Lack of genericity can force a company to change and / or redefine its existing model in order to use a solution against data leakage.

Coherence : We define coherence as the non-contradictory state between AC and TC policies. Thus, AC and TC policies are coherent if and only if they are not in contradiction with each other. For instance, the rules depicted in **Figure 1.1** in the previous chapter are incoherent, leading to potential data leakage.

Hypothesis H2: Incoherences between AC and TC policies can be tiresome to manage and lead to data leakage.

Density : A policy of x rules embeds a certain amount of entities (e.g. users, resources). We define density as the ratio between the total number of entities and the total number of rules. We hypothesize that policy definition and management can be complex and tiresome when the density is high, especially when both AC and TC policies are used within a company. We consider that such problems can lead to policy weaknesses, errors and data leakage.

Hypothesis H3: Managing several paradigms and many entities (e.g. users, resources) can be complex, leading to policy weaknesses, errors and data leakage.

Adaptability : We define adaptability as the ability for a set of rules or a global policy to be easily modified, especially in case of security emergencies. Indeed, during emergencies, fast and reactive choices need to be applied in order to efficiently react to the attack. In the case where AC and TC policies are used and coherence is maintained between them, changing the global policies can become complex and time-consuming. Such problems can lead to a potential aggravation of the crisis and thus, to data leakage.

Hypothesis H4: Modification of policies need to be performed easily and efficiently, especially in case of emergencies. Indeed, poor reaction time and lack of efficiency can worsen the security crisis, leading to policy weaknesses and data leakage.

Interoperability : We define interoperability as the ability for a company to interact with another one, regarding security policies. We hypothesize that security differences between infrastructures and lack of knowledge can cause indulgences and possible data leakage.

Hypothesis H5: Difference of security policies between companies or infrastructures and lack of knowledge can be a problem for data exchanges and business opportunities, causing *de facto* security indulgences and possible data leakage.

Reactivity : We define reactivity as the ability for a policy to be generated, modified, enhanced and analyzed in a time-efficient way. We hypothesize that the update frequency of a policy can be different from a company to another. Thus, we consider that a good solution to prevent data leaks must take into account the frequency of the company's security policy updates.

Hypothesis H6: Time-efficiency concerns must be taken into account to fit a company's updates frequency.

From the aforementioned hypotheses, we have identified 6 challenges, that must be tackled to provide a good solution against unintentional data leaks:

- **Challenge C1 - Genericity:** Take into account existing AC models.
- **Challenge C2 - Coherence:** Avoid incoherences between AC and TC policies.
- **Challenge C3 - Density:** Reduce the density of the policies that are used.
- **Challenge C4 - Adaptability:** Be able to easily and quickly modify the overall policies of the company, especially in case of emergencies.
- **Challenge C5 - Interoperability:** Be able to detect similarities and differences between policies of different infrastructures and companies.
- **Challenge C6 - Reactivity:** Take into account the policies evolution (i.e. modification of the rules) in terms of time-efficiency.

Now that the challenges have been described, we summarize in the next section the contribution of our thesis.

1.4 Contribution

The goal of our thesis was to take on the aforementioned challenges. Firstly, we have proposed a meta-model able to describe several AC models. This meta-model has been proposed to tackle **Challenge C1**.

Secondly, we have defined 2 mechanisms to tackle **Challenge C2**. The goal of the first mechanism is to generate TC rules based on existing AC rules to ensure, by construction, coherence between TC and AC policies. The second mechanism has been defined to maintain the coherence throughout the life of both policies.

Thirdly, we have proposed inferences mechanisms that can detect similarities between users and resources and automatically clusterize them, reducing *de facto* the density (i.e. total number of managed entities). These inferences mechanisms have been proposed to tackle **Challenge C3**.

Fourthly, we have defined a mechanism that allows security experts and administrators to easily create different sets of policies and rapidly switch from one to another, especially in case of emergencies. This mechanism has been proposed to tackle **Challenge C4**.

Fifthly, we have proposed inferences mechanisms that detect similarities and differences between sets of policies, allowing a company to have a better understanding of its policies and the one used by its collaborators. These inferences have been implemented to take on **Challenge C5**.

Sixthly, we have validated our hypotheses thanks to a survey that we have conducted among IT professionals who are in charge of managing the security policies in their company. Moreover, we have performed empirical tests to validate our proposal on stochastically generated and real AC policies. The results of these tests have been used to validate all challenges (including **Challenge C6**) in terms of time-consumption and efficiency.

1.5 Thesis organization

This thesis is organized as follows. Chapter 2 describes our research challenges and the general notions of information security. Then, we present the existing solutions that have been proposed to provide Access Control (AC), Transmission Control (TC) and Usage Control (UC) mechanisms.

Chapter 3 details our contribution by presenting in details the model we propose and its implementation.

Chapter 4 presents the three types of evaluations we have performed in order to validate our working hypotheses and our model in terms of scalability, time-consumption and efficiency. Moreover, we have used real AC policies to validate our model in a real context.

Finally, chapter 5 concludes the dissertation with a general conclusion and future works.

State of the art

Contents

2.1	Evaluation of the existing solutions	10
2.2	Security Properties	11
2.2.1	Confidentiality	11
2.2.2	Integrity	12
2.2.3	Availability	12
2.2.4	Authentication	13
2.2.5	Non-repudiation	14
2.2.6	Privacy	14
2.2.7	Access Control	14
2.3	Solutions to provide Access Control	15
2.3.1	Principle of Least Privilege (PoLP) or Principle of Least Authority (PoLA)	15
2.3.2	Separation of Duties (SoD)	15
2.3.3	Representation of Access Control	15
2.3.4	Access Control Policies	16
2.3.5	Access Control Models	16
2.3.6	Access Control mechanisms	22
2.3.7	General problems of Access Control	24
2.4	Data Leakage	25
2.4.1	Principles of Data Leakage	25
2.4.2	Data Loss/Leak Prevention (DLP)	26
2.4.3	Rights Management	32
2.5	Unified Access, Transmission and Usage Control with hybrid solutions	34
2.5.1	Organization Based Access Control (OrBAC)	34
2.5.2	Usage Control (UCON)	36
2.5.3	eXtensible Access Control Markup Language - Data Loss Prevention (XACML-DLP)	36
2.5.4	Hybrid solutions advantages and drawbacks	37
2.6	Conclusion	38






The objective of this chapter is to present and evaluate the existing solutions that have been proposed to protect a company's data. In order to evaluate these solutions, we first present the criteria we are using to show that no existing solution is able to completely answer the challenges we have identified in the previous chapter. Secondly, we introduce the main security properties of IT security, and more specifically, we present the solutions that have been proposed to provide Access Control. Then, we introduce the concept of data leakage and the solutions that have been implemented to tackle this problem. Finally, we present hybrid solutions that aim at mixing Access Control (AC) and Transmission Control (TC) in unified formalisms.

2.1 Evaluation of the existing solutions

As stated in the previous chapter, this thesis focuses on internal and unintentional data leakage within companies. Our objective is to propose mechanisms to prevent this kind of data leakage, with as few modifications as possible for the existing infrastructures and the existing security policies. Moreover, we aim at reducing the tiresomeness of the policies definition and management. To do so, we have considered several challenges:

- **Challenge C1:** Take into account existing AC models.
- **Challenge C2:** Avoid incoherences between AC and TC policies.
- **Challenge C3:** Reduce the density of policies that are used.
- **Challenge C4:** Be able to easily and quickly modify the overall policies of the company, especially in case of emergencies.
- **Challenge C5:** Be able to detect similarities and differences between policies from different infrastructures and companies.
- **Challenge C6:** Take into account the evolution of the policies (i.e. modification of the rules) in terms of time-efficiency.

In order to evaluate these challenges, we use the following symbols:

-  : the challenge is not taken into account by the solution
-  : parts of the challenge are underlined by the solution, but not covered
-  : the challenge is fully underlined, and partially covered
-  : the challenge is covered, but cannot be easily used in our context
-  : the solution completely takes on the challenge

- "x" : the challenge is not evaluated
- "?" : the challenge cannot be evaluated due to limitations (for instance, no valid licence to test an industrial solution)

In this chapter, we use these challenges to evaluate the existing solutions and show that they are not sufficient to reach our objectives. However, before presenting existing solutions to prevent data leakage, we present in the next section the main security properties.

2.2 Security Properties

Historically speaking, computer security has started to be implemented to protect physical hardware against several attacks. These attacks were mainly theft and damage of hardware and information and were performed in order to create disruption of service and financial costs. To prevent such issues, security models have been proposed (for instance the CIA¹-triad [Greene 2006]). These models usually embeds the following principles: Confidentiality, Integrity, Availability, Authentication, Non-repudiation, Privacy and Access Control. The following subsections describe these properties.

2.2.1 Confidentiality

Confidentiality aims at concealing the information of resources by ensuring that information is not accessed by unauthorized persons. Confidentiality is based on data encryption, a mechanism ensuring that only the right people (people who knows the key) can read the information. To do so, mathematical functions are used to transform an intended information (i.e. cleartext) into an encrypted information (i.e. ciphertext)². Once encrypted, the information can be decrypted and readable thanks to a key³. Thus, only the people who knows the key can reach the information⁴. Modern cryptography can be divided in two families: symmetric and asymmetric cryptography. Symmetric cryptography uses the same key for encryption and decryption, meaning that a common key is shared between the sender and the receiver. The most commonly used symmetric-key cipher is AES [Daemen & Rijmen 2013], a standard proposed by NIST⁵ in 2001. Asymmetric cryptography (also known as public key cryptography) uses two different but mathematically linked keys. The first key is public, meaning that it can be shared and transmitted to everyone. The second key is private, meaning that it must be kept secret. The most commonly

¹Confidentiality, Integrity, Availability

²For instance, a polyalphabetic substitution cipher proposed by a 16th century's french diplomat has been used to encrypt the acknowledgment section.

³The key to decipher the acknowledgment section represents the name of a progressive rock band created by Robert Fripp in 1969.

⁴Online tools, such as <http://www.dcode.fr/> can be used to cipher and decipher texts.

⁵National Institute of Standards and Technology

used algorithms to provide asymmetric cryptography are RSA [Rivest *et al.* 1978] or Diffie-Hellman key exchange [Kocher 1996]. **Figure 2.1** gives an overview of how symmetric and asymmetric encryptions work.

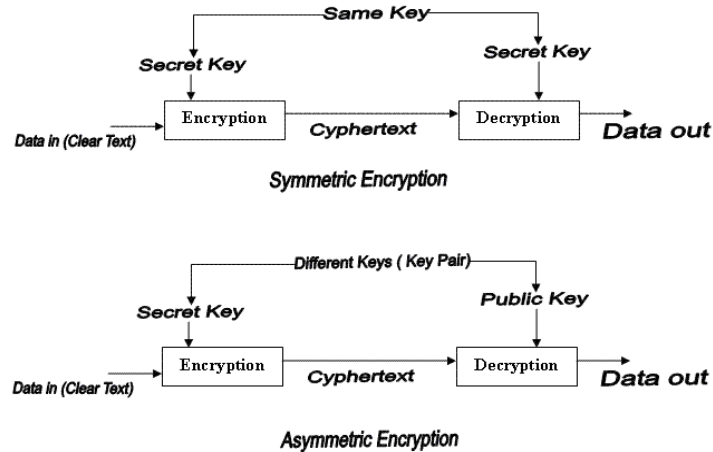


Figure 2.1: Graphical representation of the symmetric and asymmetric encryption principles.

2.2.2 Integrity

Integrity ensures that information is not altered by unauthorized users in a way that it is not detectable by authorized users. To provide such property, cryptographic algorithms (for instance, MD5 [Rivest 1992] or SHA algorithms [Eastlake 3rd & Jones 2001]) are applied. These algorithms translate a data into a fixed string of digits called a hash value. This hash value is then used to generate the checksum. Different data will generate different hashes. Thus, if a document is modified or altered, the hash generated after the reception of the data will be different from the original one, showing that an alteration has been performed on the document. As one can see in **Figure 2.2**, the hash "ZRvmCKH..." is the same on both server and client side, proving that the document has not been modified during the download.

Integrity comes in handy for companies transactions. However, the hash of the original data must be provided in a secure fashion (i.e. by a trusted institution). Thus, other mechanisms, such as digital certificates and signatures are often paired with integrity mechanisms.

2.2.3 Availability

Availability [Gray & Siewiorek 1991] [Rinard *et al.* 2004] ensures that the assets are accessible to authorized parties. Assets can include systems (e.g. Document Management System (DMS), mailing server, Operating Systems (OS)), network equip-

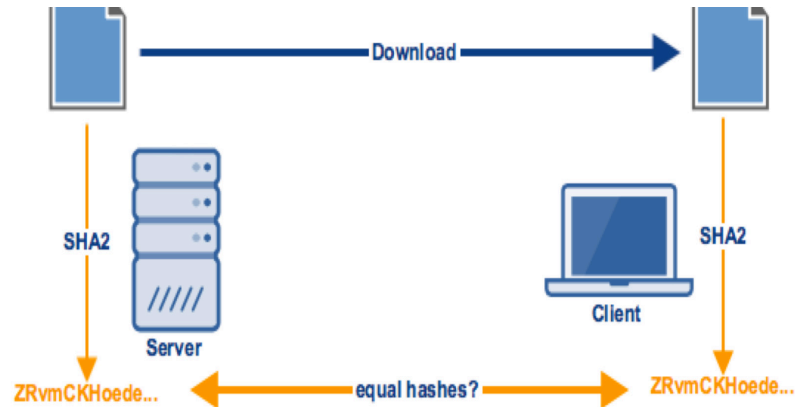


Figure 2.2: Graphical representation of the integrity principle.

ments (firewall, routers, Web servers) and software (instant messaging, mail client, specific applications). These assets must all be working properly for the information they provide and protect to be available when it is necessary. Usually, availability requirements are defined in a timely manner. Failure to respect these requirements is called a Denial of Service (DoS) [Ferguson & Senie 1997]. DoS can be the root of critical issues for companies and businesses, especially in terms of renown and financial costs. To provide efficient availability, principles such as redundancy, load-balancing, failover and RAID⁶ can be applied on equipments. These techniques mainly consist on duplicating the equipment in order to duplicate data (redundancy, RAID) or devices (load balancing, failover). Moreover, other security equipments or software such as firewalls or proxy servers can prevent DoS attacks and insure availability.

2.2.4 Authentication

Authentication has been proposed since the 70's [Ritchie & Thompson 1978] [Lamport 1981] [Hamilton *et al.* 2007]. This principle ensures that a user or a device is the one she/he/it claims to be. The most commonly used factor for authentication is the well known login and password combination. Authentication mechanisms have been broaden over the years and 3 families of factors are usually underlined. The first family is knowledge factor, which embeds what the users or machines know (e.g. password, ID, passphrase). The second family is the possession factor, which embeds things that are in the possession of the users (e.g. hardware devices such as tokens or cards for instance). Finally, the last family is inherence factor. These factors embed things that are part of a user, such as biometric authentication (e.g. retinal scan, iris recognition, fingerprint scanning, voice recognition, facial recognition) [Wayman *et al.* 2005], cognitive-traits (for instance, challenges that aim at

⁶Redundant Array of Independent Disks; originally Redundant Array of Inexpensive Disks

selecting the correct pictures among a set)[Weinshall 2006] and other behavior-based mechanisms (e.g. how a user is scrolling a page, keystrokes speed and dynamics) [Jiang *et al.* 2007].

2.2.5 Non-repudiation

Non-repudiation [Kremer *et al.* 2002] guarantees that a party cannot deny having received or sent a message or a document. This principle is interesting in scenarios involving trust during sensitive exchanges. Thus, this security principle is often used in certain types of companies, such as banks, insurances and government agencies. Over the years, different types of non-repudiation have been defined, depending on who (sender or recipient) is applying the non-repudiation mechanism [Zhou & Gollmann 1996].

From the sender point of view, one would be willing to be sure that her/his message was received by the recipient (non-repudiation of receipt (NRR)), was sent to the recipient (non-repudiation of submission (NRS)) or that the message has been correctly delivered to the recipient (non-repudiation of delivery (NRD)).

From the recipient point of view, one would be willing to be sure that the message she/he received has been sent by a genuine sender (non-repudiation of origin (NRO)).

2.2.6 Privacy

Historically, privacy has been described as the *"right of the individual to be left alone"* [Warren & Brandeis 1890]. Closer to computer security problems, [Clarke 1999] has described privacy as the interest that individuals have in sustaining a *"personal space"*, free from interference by other people and organizations. The concept of privacy encompasses numerous fields of research and areas, including legislation, sociology, psychology, anthropology and IT security. In this last domain, privacy deals with the ability for an individual (or organization) to determine what information can be shared with third parties. Thus, privacy is quite complex to achieve, because it is a very broad and subjective notion. However, several initiatives have been proposed over the years to define and provide efficient and privacy-respectful frameworks. These initiatives include for instance *"Privacy by Design"* [Langheinrich 2001] or the *"Right to be Forgotten"* [Rosen 2012].

2.2.7 Access Control

[Anderson 2008] has defined Access Control as *"traditional center of gravity of computer security. It is where security engineering meets computer science. Its function is to control which principals (persons, processes, machines, etc.) have access to which resources in the system, which files they can read, which programs they can execute, how they share data with other principals, and so on."*

Thus, Access Control provides mechanisms to determine who or what can view or

use resources in a computing environment.

Now that the main principles of security have been defined, we present in more details Access Control. As our work aims at providing mechanisms against data leakage in existing infrastructures, we have chosen Access Control as a base, because it provides a way to protect the data in industrial environments and because it is often used and known by companies.

2.3 Solutions to provide Access Control

In this section, we first present the Principle of Least Privilege (PoLP) and the Separation of Duty (SoD), two main principles of Access Control (AC). Then we present a three level abstraction model that is often used to represent Access Control. Following these three levels, we then present the main works that have been proposed to define AC policies, AC models and AC mechanisms. Finally, we underline the main drawbacks of Access Control and show that AC alone does not entirely answer our working hypotheses.

2.3.1 Principle of Least Privilege (PoLP) or Principle of Least Authority (PoLA)

Principle Of Least Privilege (PoLP) (also known as Principle of Least Authority (PoLA)) has been defined as a principle that *"requires that a user be given no more privilege than necessary to perform a job. Ensuring least privilege requires identifying what the user's job is, determining the minimum set of privileges required to perform that job, and restricting the user to a domain with those privileges and nothing more."* [Saltzer 1974]. In other words, this principle aims at limiting the potential damage of any accidental or malicious security breach by giving people the lowest level of user rights that they can have and still do their jobs. Thus, this principle is often used while defining Access Control.

2.3.2 Separation of Duties (SoD)

Another important concept of Access Control is the Separation of Duties (SoD)[Gligor *et al.* 1998]. SoD restricts the amount of power held by one individual in order to reduce the risks of errors, frauds, conflicts of interest, etc. In IT security, SoD tends to improve security (by creating for instance different accounts with limited actions sets instead of one master account per user). For these reasons, this concept is interesting while defining Access Control policies.

2.3.3 Representation of Access Control

[Samarati & de Vimercati 2000] have defined three layers of Access Control: Policies, Models and Mechanisms:

- Policies embeds high-level rules according to which Access Control must be regulated.
- Models represent formal implementation of the policies.
- Mechanisms define the low level pieces of software or equipment (hardware) that implement the Access Control stated by the model, and thus, imposed by the policies.

Throughout the rest of this section, we use this 3 levels abstraction to describe the main solutions that have been proposed in Access Control research field.

2.3.4 Access Control Policies

Access Control policies embed the set of rules company's employees need to comply to. Most of the time, these policies define "*Who can access what?*". To do so, Access Control policies are composed of a subject, an object, an action (or operation) and a permission (or privilege). These terms are defined as follows [Center 1988]:

Subject : a subject represents an active entity, generally in the form of a person, process, or device, that causes information to flow among objects or changes the system state.

Object : an object represents a passive entity that contains or receives information. Access to an object potentially implies access to the information it contains. Examples of objects are: records, blocks, pages, segments, files or directories.

Action : an action or operation represents an active process that is invoked by a subject on objects. Examples of actions are: Read, Write, Execute, etc.

Permission : permission (or privilege) represents a description of the type of authorized interactions a subject can have with an object. In other words, permissions represent the authorization assigned to a subject to perform a specific action on a specific resource.

2.3.5 Access Control Models

In this subsection, we present the main AC models that have been proposed over the years to ensure efficient and fine-grained AC.

a) Discretionary Access Control (DAC)

U.S. Department of Defense (DoD) has defined the Trusted Computer System Evaluation Criteria⁷ (TCSEC)[Tcsec 1985] in the 80's. TCSEC is a set of security guidelines and standards which has defined Discretionary Access Control (DAC). In DAC

⁷TCSEC is also known as the "*Orange Book*", due to its colored cover.

models, users can set, modify or share the access control of their resources thanks to a set of rules. DAC was developed in order to implement the concept of Access Control Matrix defined by [Lampson 1974] and formalized by [Harrison *et al.* 1976]. An Access Control Matrix can be represented as a two dimensional matrix where rows represent subjects, columns represent objects and intersections between rows and columns represent an action a subject can perform over a specific resource. Thanks to this simple representation, DAC is used to limit a user's access to a file. Many modern Operating Systems such as Windows family, GNU/Linux and Mac OS are based on DAC models. DAC can either be defined with Access Control Lists (ACL) or Capabilities.

Access Control Lists (ACL) : ACL has been first introduced in [Saltzer & Schroeder 1975]. Primarily used within 1970's multi-users systems [Graham & Denning 1972], ACLs consist of a list of entries that informs about the access right that each user has to a specific resource. As DAC, an ACL is traditionally represented as the two-dimensional matrix presented previously. Such matrix is depicted in **Table 2.1**, while the general concept of ACL is represented in **Figure 2.3**.

	Afile	Bfile	Cfile	Dfile
Anna	rwX	r	x	
Bill	r	rx	x	
Charles	r	r	rw	r
Damian		r		rw

Table 2.1: Representation of an AC matrix. Users are represented as rows and files are represented as columns. Intersections represent what a specific user can do to a specific file. In this example, user Bill can for instance read (r) document Afile.

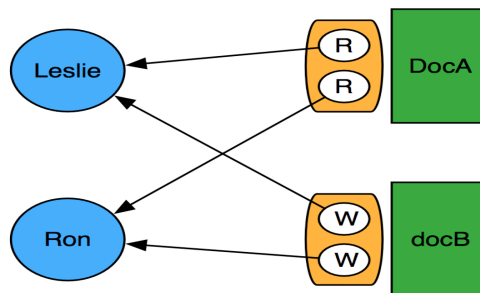


Figure 2.3: A generic representation of the ACL model. As one can notice, the access rights are conserved in the resources side.

Capability-Based security : First introduced in [Dennis & Van Horn 1983], capabilities have been defined as "*a token, ticket, or key that gives the possessor permission to access an entity or object in a computer system*". Technically speaking, a capability is represented as a pair (x, r) , where x represents the name of an object (for instance a file) and r is a set of privileges or rights (for instance, read or write). Thanks to this representation (graphically depicted in **Figure 2.4**, each subject can have its own capabilities' set, allowing her/him to perform certain actions on resources. Moreover, a capability can be transferable and must be unforgeable. Unlike ACL, capabilities can prevent the confused deputy problem, a specific privilege escalation where a computer program is innocently fooled by some other third-party into misusing its authority. Moreover, capabilities offer precise control for accessing data.

However capability based systems have the reputation of being slow and complex and have been replaced over the years by ACL or other model in commonly-used Operating Systems. Nevertheless, several works have been proposed in order to use capabilities in modern contexts, including distributed systems [Selimi & Freitag 2014], Web [Miller *et al.* 2008] and software engineering [Mettler *et al.* 2010]. Moreover, interesting works have tried to popularize such mechanism by destroying the myth built around capabilities for the last decades [Miller *et al.* 2003].

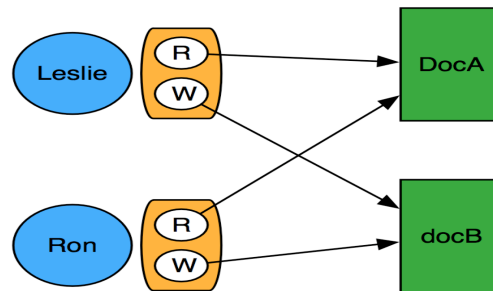


Figure 2.4: A generic representation of the capability model. As one can notice, the access rights are conserved in the subjects side.

DAC advantages and drawbacks : DAC has been described as fine-grained by [Coyne 1996]. Thanks to its fine-grained controls, DAC can be used to implement least-privilege access. Indeed, specific objects can have access control restrictions to limit individual subject access to the minimum rights needed. Furthermore, DAC offers intuitive implementation and is mostly invisible to users [Smalley 2001].

However, DAC suffers from several drawbacks . First of all, because permissions are directly applied to individual subjects, DAC definition and maintenance can be very difficult within big infrastructures or companies. Moreover, [NIST 2006] has declared that DAC has the following main drawbacks:

- Granting read access is transitive (i.e. when a resource owner grants access to another subject, the granted subject can access the resource and allow other subjects to read it without referring to the owner). Thus, no restrictions apply to the usage of information when the user has received it.
- Vulnerability to malicious code such as Trojan horse, because the malicious program will inherit the identity of the invoking user.
- Information can be copied from one object to another; therefore, there is no real assurance on the flow of information in a system.
- The privileges for accessing objects are decided by the owner of the object, rather than through a system-wide policy that reflects the organization's security requirements.

Now that DAC family has been presented, we introduce the concepts of Mandatory Access Control (MAC).

b) Mandatory Access Control (MAC)

Also defined in [Tcsec 1985], MAC refers to a family of models where the system assigns security labels or classifications to resources (for instance "*classified*", "*secret*" or "*top secret*") and allows access to subjects or applications depending on their level of clearance. This categorization mechanism is often referred as Multi-Level Security (MLS) in several works [McCullough 1987] [Karger 2005].

MAC is adequate within infrastructures where Access Control policies must not be decided by owners and within systems that must enforce protection decisions (i.e. the system has the final word). MAC is usually associated with two models : Bell-LaPadula [Bell & LaPadula 1973], which provides confidentiality, and Biba model [Biba 1977], which provides integrity.

Bell-LaPadula Model (BLP) : Bell-LaPadula Model is an extension of military Multi-Level Security paradigm. It is a model that assigns security labels (such as "*Confidential*" and "*Top secret*") to subjects and objects. Thanks to the notion of partial and total orders, BLP defines two main properties:

- The simple security property: which prevents users from viewing objects with security levels higher than their own. (*No-Read up*).
- The *-property: which prevents propagation of information to users with a lower security level (*No-Write down*).

These properties have been discussed in several works [McLean 1985] [Bishop 2005] [Anderson 2008] and are presented in **Figure 2.5**.

Biba Model : While Bell-LaPadula model describes methods for insuring confidentiality of information flows, Biba [Biba 1977] applies reverse principles to provide integrity property (see **Figure 2.5**). In Biba, objects and users are tagged with integrity levels. These levels form a partial order, similar to the BLP model. However, Biba applies the two following principles:

- The simple integrity axiom: which states that a subject at a given level of integrity must not read an object at a lower integrity level (*"No-Read down"*).
- The *-Integrity axiom: which states that a subject at a given level of integrity must not write any object at a higher level of integrity (*"No-Write up"*).

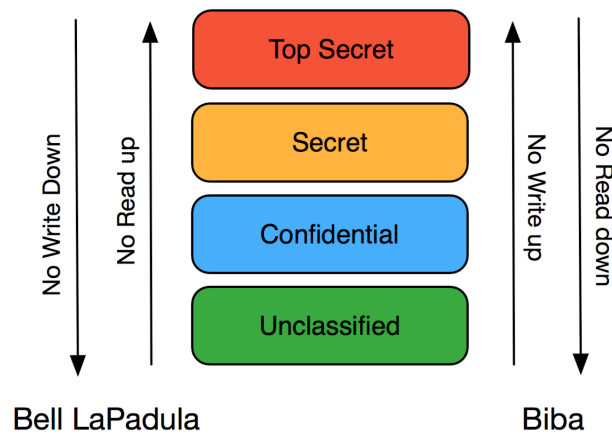


Figure 2.5: Bell-LaPadula and Biba models. Bell-LaPadula provides confidentiality with the *"no read up"* and the *"no write down"* properties while Biba provides integrity with the *"no write up"* and the *"no read down"* properties.

MAC advantages and drawbacks : Thanks to its properties, BLP offers a good way to insure confidentiality property in military and critical infrastructures. Moreover, BLP is not susceptible to trojan horse security violations because users do not have the ability to declassify information [Ausanka-Crues 2001]. More generally, MAC solutions are simple and scalable [Coyne 1996]. However, MAC suffers from some drawbacks. Among them, MAC only controls information in the system that happen through overt channels (i.e. channels operating in a legitimate way). Thus, non-legitimate channels (called covert channels), are not covered by MAC. Moreover, MAC can be too rigid. Indeed, subjects and objects classification may be not feasible in complex contexts that will require many different categories.

c) Role Based Access Control (RBAC)

Role Based Access Control (RBAC) [Sandhu *et al.* 1996] [Ferraiolo *et al.* 1995] is based on the notion of role. A role is a set of subjects that share common attributes

(for instance, a role *"developer"* containing all developers of a team). In this model, being a member of one or more groups gives users access to certain resources. Thus, RBAC is often used in companies, where organizational structure is already based on roles. Thanks to this clustering, policy management is facilitated, especially in big and complex infrastructures. Moreover, RBAC covers the problem of least privilege (i.e. avoiding the assignment of permissions that are unnecessary) and the separation of duty (i.e. disseminating the tasks and associated privileges among multiple users). Over the years, the first version of RBAC (i.e. RBAC0), depicted in **Figure 2.6**, has been enhanced to meet the changing needs and requirements in modern computer's infrastructures [Hu *et al.* 2006]. Such enhancements embed role hierarchy (RBAC1), dynamic separation of duty (RBAC2), or both (RBAC3).

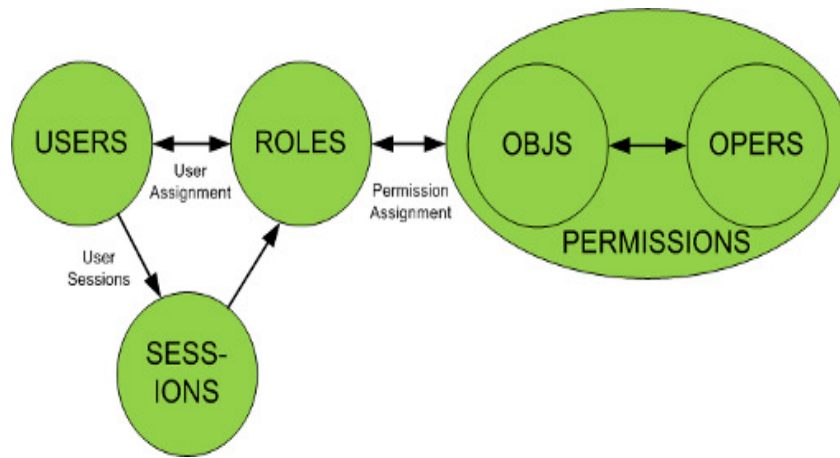


Figure 2.6: Graphical representation of the RBAC0 model.

d) Attributes Based Access Control (ABAC)

ABAC has been defined as *"An access control method where subjects requests to perform operations on objects are granted or denied based on assigned attributes of the subject, assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions"* [Hu *et al.* 2013]. The notion of attributes embeds various criterions about a subject (e.g. name, position, gender), an object (e.g. name, security level, extension) or the environment (e.g. time period, day of the week, date). Thus, ABAC can be seen as an extension of RBAC. For the last few years, many solutions have been proposed to provide ABAC [Wang *et al.* 2004] [Lang *et al.* 2006] [Jin *et al.* 2012]. Thanks to its fine-grained representation, context-aware and risk-mitigating policies, ABAC is adapted to modern companies. Moreover, ABAC rules can be combined in order to provide complex and explicit rules with minimal effort. In order to perform this policy combination, an authorization engine is often used. An example of this engine is depicted in **Figure 2.7**.

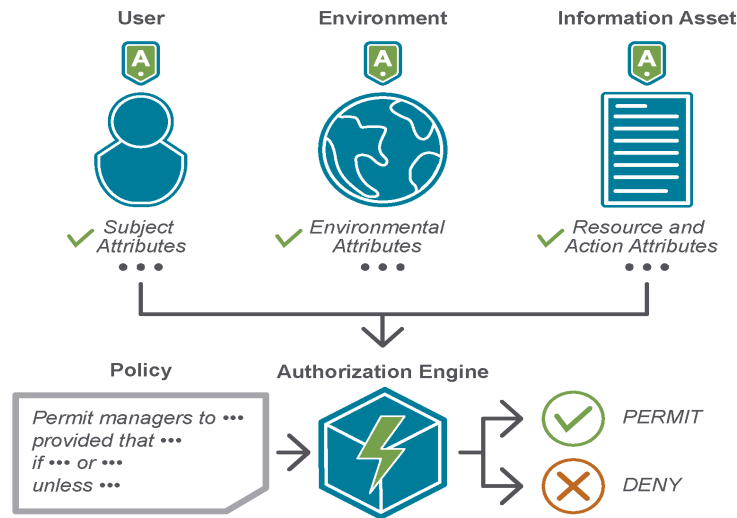


Figure 2.7: Graphical representation of ABAC model. Attributes from various sources can be checked by the engine to allow or deny access based on the defined policies.

e) Other models of Access Control

Over the last decades, several models have been proposed to cover the problem of Access Control. These models propose to take into account various notions, including context [Corrad *et al.* 2004] [Kulkarni & Tripathi 2008], history [Edjlali *et al.* 1998] [Banerjee & Naumann 2004], temporal-RBAC [Bertino *et al.* 2001], risk [Kandala *et al.* 2011], authorization [Karp *et al.* 2010] or trust [Kagal *et al.* 2001]. Despite the fact that these solutions are interesting, they are specific to academic domain and thus, are not often used in real companies and real infrastructures.

2.3.6 Access Control mechanisms

Access Control models have been implemented through hardware and software mechanisms. Following subsections present the main implementations that have been proposed over the years.

Hardware implementations of Access Control : Hardware implementations have been defined since the 60's. For instance, Burroughs family (B5000, B6000) [Mayer 1982] have proposed code and data separation in memory followed in the 70's by the Plessey System 250 [Fabry 1974], the first operational computer system

to implement capability-based addressing. In the 80's, object-based approach has been defined within systems such as the IBM System/38 [Houdek *et al.* 1981] and the iAPX 432 [Rentsch 1982]. These systems have proposed security built-in architecture, allowing Access Control on each object. Details on these systems can be found in [Levy 2014].

Software implementations of MAC : Several software implementations have been proposed over the years to provide Mandatory Access Control. These implementations include Operating Systems such as Multics (First Multi-level Security system) [Corbató & Vyssotsky 1965], or more recently FreeBSD (support of TrustedBSD [Watson *et al.* 2003]), Tomoyo Linux [Harada *et al.* 2004] and Trusted Solaris⁸.

Despite OS, several modules (i.e. kernel extension) have been proposed. These modules include solutions such as SELinux [Jaeger *et al.* 2003], AppArmor [Bauer 2006] and Smack (Simplified Mandatory Access Control Kernel) [Schaufler 2008].

Windows Operating Systems incorporates Mandatory Integrity Control (MIC) [Microsoft 2016]. MIC uses integrity levels and mandatory policies to evaluate access to objects. Finally, Apple Mac OS X offers a Mandatory Access Control framework based on TrustedBSD (SEDarwin) [Vance *et al.* 2007].

However, these Operating Systems have been either abandoned (e.g. Multics) or target very specific sectors, such as defense, academia, or governance.

Software implementations of DAC : DAC has also been implemented in many ways. Most modern Operating Systems such as Windows family OS, GNU/Linux and Apple Mac, but also other Unix-like systems are based on DAC models. To implement DAC, these systems use either Access Control Lists or capabilities. Access Control List (ACL) can be defined and managed within file systems in GNU/Linux and other POSIX-compliant systems. The most commonly used file systems permissions is based on the notion of modes. Modes are permissions given to "*user*" (i.e. the owner of the file or directory), "*group*" (i.e. set of members of the file's group) and "*others*" (users who are not the owner of the file or members of the group). Concerning the permissions, modes embed common actions such as "*read*", "*write*" or "*execute*" a file or a directory. Such AC policies can be defined in Unix-like systems thanks to the command *chmod*. Access Control List (ACL) can also be defined and managed thanks to the command-line *setfacl*. This command offers more fine-grained granularity, allowing for instance an administrator or expert to give additional rights to a specific user.

Capabilities have been implemented for many systems and contexts over the years. For instance, TAHOE-LAFS [Wilcox-O'Hearn & Warner 2008] is a distributed filesystem that embeds capability concepts to cover the Principle of Least

⁸A documentation on Trusted Solaris history and features can be found here: <http://www.cse.psu.edu/~trj1/cse544-s10/slides/cse544-lec12-solaris.pdf>

Authority (POLA). Operating Systems such as EROS [Shapiro *et al.* 1999] or Coyotos [Shapiro *et al.* 2007] have been implemented to support capabilities. However, these Operating Systems are quite marginal (for instance, EROS has been proposed for scientific use only and has never been deployed within companies). Thus, these solutions are not often used by companies.

2.3.7 General problems of Access Control

Within companies, traditional AC models offer good solutions to restrict access to resources. However, an employee could retrieve a document from a datacenter and send this document to an unauthorized colleague. If this user has privileged rights on her/his computer, she/he will be able to read or write the content of the file, bypassing *de facto* the AC policy. Such action will cause an data leak, because an unauthorized user will have access to information she/he is not suppose to have. Moreover, the leak will be unnoticed by the company.

Solutions such as Multi-Level Security (MLS) models and capability-based security can prevent such issue, as the access rights are embedded within the file, preventing further modifications of the rights if the user does not have the correct capability or the correct accreditation level. However, MLS and capabilities are not often used in nowadays companies and software. Indeed, capabilities have been abandoned over the years for technical and complexity reasons while MLS often targets very specific sectors.

Concerning the criterions we have previously underlined, **Table 2.2** shows that existing AC solutions do not cover our challenges well. However, while genericity (**C1**), coherence (**C2**) and density (**C3**) are not covered by most solutions, parts of the interoperability (**C4**) and adaptability (**C5**) problems are covered for all of them, mostly because AC is a well-known mechanism embedded by default in most modern Operating Systems. Finally, we have defined reactivity as the ability for a solution against data leakage to take into account the update frequency of existing mechanisms (i.e. Access Control mechanisms). Thus, reactivity (**C6**) cannot be properly evaluated for AC solutions. However, we underline that these mechanisms are usually fast when it comes to update frequency.

Solution	Type	Domain	Genericity	Coherence	Density	Interoperability	Adaptability	Reactivity
ACL	AC	Academy / Industry	○	○	○	◐	◐	x
Capabilities	AC	Academy / Industry	○	○	○	◐	◐	x
RBAC	AC	Academy / Industry	○	○	◐	◐	◐	x
ABAC	AC	Academy / Industry	○	○	◐	◐	◐	x
MAC	AC	Academy / Industry	○	◐	◐	◐	◐	x

Table 2.2: Coverage of the main AC solutions regarding our underlined challenges.

Now that AC solutions have been presented, we describe the concept of data leakage.

2.4 Data Leakage

In this section, we first present the principles of Data Leakage. Then, we discuss the Data Leak Prevention (DLP) solutions. Finally, we present the concept of Rights Management.

2.4.1 Principles of Data Leakage

Data leakage has been defined as the accidental or unintentional distribution of private or sensitive data to an unauthorized entity. Thus, data leakage differs from data theft (i.e. intentional distribution of data). Because of its nature, data leakage can pose several issues for a company, including financial loss, damage of goodwill and reputation, lawsuits, loss of future sales, exposure of intellectual properties. Virtually speaking, any type of data can be leaked, as soon as it is considered as sensitive or valuable. According to [McAfee 2015], data theft have been targeting customers and employees personal information (e.g. names, email addresses, passwords, credit card numbers, social security numbers, medical records, phone numbers). Moreover, intellectual properties and other financial documents have also been targeted.

Concerning the format, the same survey has shown that Microsoft Office documents (i.e. Excel, PowerPoint, Word) are the dominating format of leaked data (39% of leaks from internal actors), followed by Plain Text and CSV (Comma Separated Value) (20%) and PDF (11%). Data leakage can come from 2 different types of sources: external sources and internal sources.

External sources : When one think of security issues, hackers and thieves often come in mind. External sources in data leak refer to outsiders (people others than employees, such as hackers, ancient employees, etc.). These outsiders can have various motives, such as industrial espionage, fame, profit or grudge. Whatever the motivation, external sources are difficult to spot and prevent, mostly because the attackers can use a various set of tools and vectors that are difficult to probe and monitor. For instance, data exfiltration can be performed through covert channels (i.e. channels that are not supposed to be allowed to communicate) or social engineering (e.g. identity theft, deception).

[McAfee 2015] has shown that 57% of data leaks that have occurred in 2014 were caused by external hackers. Other academic research, such as [Hauer 2015], support these numbers. Indeed, the paper states that approximately 50% of data breaches reported the same year were caused by outsiders.

Internal sources : Internal sources refer to data leakages caused by the employees of a company. Studies such as [McAfee 2015] show that roughly 43% of the data leak incidents of 2014 were internal. The cause of such internal leaks can be either intentional or unintentional ([McAfee 2015] states that there is a 50%-50% distribution between intentional and unintentional data leaks). Intentional leaks can be motivated by numerous factors, including profit or revenge.

On the contrary, unintentional leaks can be caused by social engineering, lack of knowledge and awareness, careless publication, incorrect usage of tools, clumsiness, poor business process or inappropriate handling of documents.

Based on the previous results, we consider that unintentional and internal leaks represent a non-negligible amount of leaks that could be interesting to tackle. Indeed, we underline that these leaks can create huge problems for the company and the employees, especially in terms of credit, reputation and trustworthiness. Moreover, this type of data leakage can be detected more easily, because they are usually performed through overt channels. Overt channels have been described as *"the normal and legitimate way that programs communicate within a computer system or network"*. [Graves 2007]. Thus, overt channels include all legitimate communication including network protocols (e.g. HTTP, TCP/IP), messaging protocols and applications (e.g. Microsoft Communicator, Skype), email and physical devices (e.g. USB key, hard disk, external hard drive, CD or printers). For these reasons, we have decided to focus in this thesis on internal and unintentional data leakage.

Now that the main concepts of data leakage have been introduced, real examples of internal and unintentional data leaks are presented.

Real cases of internal and unintentional Data Leakage : Many examples can be found concerning data leakage. More specifically, **Table 2.2** shows some examples of unintentional data leaks. As one can see, different types of infrastructures and sectors can suffer from this problem. Moreover, vectors of communication can be varied.

In this section, we have described the concept of data leakage, its sources and its main problems. Among the two types of sources that have been presented, internal sources embed leaks that are performed within the company. Thus, the leaks can either be performed intentionally by malicious users, or unintentionally. Various studies have shown that internal and unintentional leaks represent a non-negligible amount of leaks. To underline these studies, examples of real internal and unintentional data leakage have been proposed. Among these examples, the last one (FDIC) has been detected by security teams thanks to a mechanism called Data Loss/Leak Prevention (DLP). The main principles of such mechanism are presented in the next section.

2.4.2 Data Loss/Leak Prevention (DLP)

This section presents the main notions of Data Loss/Leak Prevention (DLP). DLP solutions have been described in various terms, including Information Leak Detection and Prevention (ILDP), Information Leak Prevention (ILP), Content Monitoring and Filtering (CMF), Content Monitoring and Protection (CMP), Extrusion

Date	Organization	Information
2007	U.S. Nuclear Laboratory	An employee transmitted confidential information on US atomic weapons by email via non-secured networks to members of the board of Los Alamos National Security
2008	Norway Government	The tax agency mistakenly sent CDs containing confidential information about nearly 4 millions (i.e. 85%) of Norwegian adults to nine major media groups.
2011	Sogeti	A file containing the personal information and evaluations of 298 employees was unintentionally sent by email to these employees. Among personal information such as salaries and raise the file included commentaries on employees performances.
2016	Australian Government	An administrative error from the prime minister's department revealed a mailing list of 800 addresses that were supposed to be confidential.
2016	Google	A company's staff benefits vendor mistakenly sent an email containing employee's sensitive information to the wrong recipient
2016	Federal Deposit Insurance Corporation (FDIC)	Former employee accidentally made a copy of 44.000 customers on a USB drive

Table 2.3: Real case of data leakage that have occurred in the last few years

Prevention System (EPS) or Outbound Content Compliance. Nevertheless, DLP is the most commonly used name. Thus, this name will be used throughout this dissertation.

a) Definition and types of DLP

In [Shabtai *et al.* 2012], a DLP has been described as a "*system that monitors and enforces policies on fingerprinted data that are at rest, in-motion or in-use on public or private computer/network.*". Other definition, such the one proposed by [Mogull & Securosis 2007], has described DLP solutions as "*systems that identify, monitor, and protect data in-use, data in-motion and data at-rest through deep content inspection using a centralized management framework.*". Thus, these definitions introduce the three notions of data at rest, data in-motion and data in-use.

Data at rest defines all data within computer storage (e.g. databases, hard-drives). In order to protect data at rest from theft or alteration, security measures such as encryption and Access Control can be used. One of the main features of

data at-rest DLP is data discovery. This feature aims at pruning storage in order to detect if data are located on unauthorized storage. If it is the case, encryption or destruction of the data can be performed by the DLP.

Data in-use refers to data that are being interacted with. Endpoint systems are often used to protect data in-use. To do so, agents can be installed on endpoints (for instance, employees computers). The main goal of these agents is to monitor users actions and prevent sensitive data from being leaked. Many channels can be monitored, including shared LAN folders, email attachments, portable storage devices (e.g. USB Drive, CD/DVD) and applications sandboxing.

Data in-motion refers to data that are sent through a network. This network can be either internal, like an Intranet, or external, like the Internet. DLP can be used to monitor and detect unauthorized data transmissions based on commonly used protocols (e.g. HTTP, instant messaging, email) or unknown protocols thanks to packets inspection. Technically speaking, a network DLP is often placed between the internal and the external networks and serve as a proxy which monitors network traffic.

b) Policy definition

DLP solutions are managed through a centralized server. This server manages both network and endpoint proxies and is responsible for policy deployment and logging policy violations. Policies are defined by security experts and administrators. These policies can be based on the company own specifications or others such as PCI-DSS⁹ [Gikas 2010]. A policy can be viewed as a set of rules defining "*Who can send what to whom?*". Moreover, additional actions can be performed, for instance, "*notify the security expert if the specific data x is sent by user $U1$* ". Thus, DLPs are providing Transmission Control (TC).

c) Data classification

DLPs handle leakage incidents in two main approaches: detective approaches and preventive approaches.

Detective approaches refer to actions that aim at detecting and taking corrective actions once the leakage is identified. Detective approaches embed three main domains: content-based inspection, context-based inspection and content tagging.

Content-based inspection : DLP solutions can analyze the content of a document with various techniques. These techniques include keywords matching (e.g. searching for the word "*confidential*" within a document), regular expressions (e.g.

⁹Payment Card Industry Data Security Standard : a standard for organizations that handle credit cards

searching for a credit card pattern) or document comparison techniques. This last technique can compare a whole document, or more efficiently, document's fingerprint (i.e. pre-generated hashes).

Context-based inspection : Context-based inspection refers to a set of techniques that extract contextual information such as source, destination, size, sender, file type, timestamp, header/meta-information or format. To do so, explicit filtering of contextual information need to be performed. Thanks to these techniques, a content-based DLP can for instance prevent specific documents to be sent to specific destinations.

Content tagging : Tags can be assigned to specific data. For instance, a "*confidential*" tag can be affected to a specific document. This affectation can be manually performed by the owner of the data or by an administrator. Moreover, processes can be used to automatically tag document thanks to content or context-based analysis.

Preventive approaches refer to approaches that prevent potential leakage before they occur. To do so, preventing approaches take measures such as disabling functions, awareness, encryption and Access Control.

Disabling functions : In this approach, functions that can generate data leakage or inappropriate use of sensitive data are disabled. These functions include: copy-paste, sending a document through the network (e.g. email, specific protocols), blocking devices inputs (e.g. USB key or external storage) or blocking specific application inputs (e.g. Skype, Microsoft Communicator).

Encryption : As stated in 2.2.1, encryption is the action of encoding messages or information in such a way that only authorized parties can read it. Thanks to encryption, DLP policy can force a data to be encrypted and can specify who is allowed to decrypt it.

Awareness : Awareness refers to a set of techniques that aim at raising the consciousness level of employees. Such techniques encompass messages that give advices like "*this will cause a data leakage*", warn employees (i.e. "*a potential leak is about to occurred*") or give general information like "*here is the list of the users you are allowed to send the document to*".

Access Control : Finally, Access Control can be used to allow or deny access to specific documents. One way to achieve Access Control is to use existing mechanisms, such as the one presented in 2.3.

d) Commercial DLP

Unlike Access Control mechanisms, DLP solutions are quite new in the market. Indeed, since 2006, several larger vendors have bought smaller companies specialized in data security [ZDN 2007]. Thanks to these buyouts, DLPs technologies have started to arise since 2008, proposing scalable and business oriented solutions. Nowadays, the biggest vendors¹⁰ are Websense¹¹, Trend Micro¹², RSA¹³, Symantec¹⁴, Palisade Systems¹⁵, NextLabs¹⁶, McAfee¹⁷, Fidelis Security Systems¹⁸, Sophos¹⁹, CA Technologies²⁰ and Code Green Networks²¹. These solutions include core features such as network, email and storage monitoring.

Moreover, other features such as policy templates (e.g. Symantec, Code Green Networks, RSA), encryption (e.g. Sophos, Symantec, Websense, TrustWave) or machine learning with statistical analysis (Symantec) are also implemented.

e) Academic research on DLP

From the academic point of view, researchers have been focusing on several problems, including emails leakage protection [Zilberman *et al.* 2011], network and Web based protection [Caputo *et al.* 2009] and misused detection in database [Harel *et al.* 2010] [Harel *et al.* 2012]. Moreover, solutions have been proposed to improve detection methods by using machine learning [Mathew *et al.* 2010] [Gafny *et al.* 2010] [Li *et al.* 2015]. Closer to industrial preoccupations, [Alawneh & Abbadi 2008] have proposed a framework to protect the data shared between collaborative organizations. Finally, some works have been proposed to tackle privacy [Chae *et al.* 2015] or Access Control and confidentiality [di Vimercati *et al.* 2011].

f) Advantages and drawbacks of DLPs

DLP solutions offer efficient solution against data leakage within companies. Such solutions propose a way to monitor users actions and take active measures to prevent unauthorized transmissions. While academic DLP researches try to improve classification and detection methods, commercial DLP mainly focus on features

¹⁰Open-Source solutions also exist, such as OpenDLP<https://code.google.com/archive/p/opendlp/>

¹¹<https://www.forcepoint.com/fr/product/web-filtering/websense-web-filter-security>

¹²<http://www.trendmicro.fr/grandes-entreprises/protection-des-donnees/prevention-des-pertes-de-donnees-integree/>

¹³<https://www.rsa.com/en-us>

¹⁴<https://www.symantec.com/fr/fr/data-loss-prevention/>

¹⁵<http://palisadesystems.com/>

¹⁶<https://www.nextlabs.com/press/node-385/>

¹⁷<http://www.mcafee.com/fr/products/total-protection-for-data-loss-prevention.aspx>

¹⁸

¹⁸<https://www.fidelissecurity.com/>

¹⁹<https://www.sophos.com/fr-fr/press-office/press-releases/2012/03/sophos-dlp.aspx>

²⁰

²⁰<http://www.ca.com/us/products/ca-data-protection.html>

²¹<https://www.codegreennetworks.com/>

that reduce the tiresomeness of policy definition. Even if DLP are very good at protecting sensitive data, they generally have no information about who access the data and how Access Control is granted. To obtain these information, DLP can rest on existing protocols such as LDAP²² (e.g. Symantec, McAfee, Websense, Trend Micro, RSA, NextLabs, Code Green Networks). With such solutions, groups of users and general Access Control policy can be re-used.

Other solutions such as the one provided by CA Technologies are focusing on Identity and Access Management (IAM). IAM is the security discipline that enables the right individuals to access the right resources at the right times for the right reasons [Gar 2016]. In other words IAM provides a framework to manage electronic identities. CA Technologies' DLP provides its own IAM that enforces data classification based on user attributes and content awareness.

However, main commercial actors tackle the problem with an implementation point of view. Indeed, these actors are making their solutions compatible with common applications (for instance, specific implementation of LDAP such as Microsoft Active Directory), rather than abstracting the problem and proposing a generic solution. Moreover, this compatibility targets most of the time proprietary and widely used Access Control and authentication mechanisms, making interoperability even more complicated for a company using exotic or self-implemented AC model. Indeed, the company will either have to manually define the Transmission Control rules or to use a compatible AC solution. In both cases, the company will have to spend some time adapting and redefining policies. Thus, we consider that DLP solutions do not handle genericity (**C1**) well (see **Table 2.4**). Moreover, DLP are not designed to tackle coherence and density problem, despite the fact that higher abstraction such as groups and roles can be defined. Thus, we consider that coherence problem (**C2**) and density problem (**C3**) are partially covered. Concerning adaptability (**C4**) and interoperability (**C5**), DLP are usually implemented to ease the management of the policies and they can be interfaced with many existing solutions, including Microsoft Sharepoint and Office software (e.g. RSA, NextLabs, Code Green Networks, Fidelis Security Systems), virtualization tools such as VMWare (e.g. TrendMicro, RSA) or network oriented services such as Facebook Messenger, Yahoo! Chat, Skype or Microsoft Office Communicator (e.g. McAfee, NextLabs, Palisade Systems). Unfortunately, the proposed solutions lack abstractions and are dependent on a specific implementation, making interoperability challenge (**C5**) partially covered. Finally, reactivity challenge (**C6**) cannot be evaluated, because these solutions are not freely distributed and a licence is required. However, we underline that reactivity is often highlighted by DLP vendors.

Despite DLPs, other mechanisms can be used to prevent data leakage within companies. These mechanisms, defined for Rights Managements, are presented in the next section.

²²Lightweight Directory Access Protocol
















Solution	Type	Domain	Genericity	Coherence	Density	Interoperability	Adaptability	Reactivity
Trend Micro	DLP	Industry						?
McAfee	DLP	Industry						?
Symantec	DLP	Industry						?

Table 2.4: Coverage of the main DLP solutions regarding our challenges.

2.4.3 Rights Management

As stated previously, traditional Access Control mechanisms do not tackle the problem of data retransmission and usage. To overcome these issues, other mechanisms have been implemented. Among these mechanisms, Digital Rights Management (DRM) and Information Rights Management (IRM) have been proposed. Digital Rights Management (DRM) aims at preventing unauthorized redistribution of a digital media (e.g. document, music, video) and restrict the ways consumers can use this content (copy, distribution to others, etc.). DRM solutions have been developed in response to the increase of online piracy (i.e. redistribution of copyrighted information over the Internet thanks to peer-to-peer networks). A DRM embeds mechanisms that prevent copying, specify a time period in which the content can be accessed or limit the number of devices that can consult the media. DRM are most commonly applied to media such as video games, audio CDs, software, movies and ebooks.

Within companies, Information Rights Management (IRM) or Enterprise Digital Rights Management (E-DRM or ERM) can be used. IRM refers to Rights Management technology specifically designed for enterprise documents. Thus, IRM aims at protecting sensitive information, such as patents, employees personal information or financial data. In the next subsections, we describe IRM in details.

a) Commercial IRM

Over the last few years, several commercial IRM solutions have been developed. While a DLP offers a Transmission Control solution, IRM focuses on Usage Control problematic. To this end, these commercial solutions allow end-users and administrators to set access permissions to documents in order to decide *"what can be done with the data once it has been accessed and distributed"*.

The main vendors in the domain are Seclore²³, Microsoft²⁴, Covertix²⁵ and EMC²⁶. Moreover, other solutions such as the one provided by Prot-On²⁷, Vaultize²⁸ or

²³<http://www.seclore.com/>

²⁴[https://technet.microsoft.com/en-us/library/dd638140\(v=exchg.150\).aspx](https://technet.microsoft.com/en-us/library/dd638140(v=exchg.150).aspx)

²⁵<http://www.covertix.com/>

²⁶<http://france.emc.com/enterprise-content-management/information-rights-management.htm>

²⁷<https://www.prot-on.com/information-rights-management-software>

²⁸<http://www.vaultize.com/>

Digify²⁹ exist. These solutions offer mechanisms to restrict content from unauthorized modification, copying, pasting or printing alongside a set of features that include security audit and governance tools (e.g. Seclore, EMC, Covertix) or encryption (e.g. Seclore, Covertix, EMC, Microsoft).

b) Academic research on IRM

While traditional DRM have been discussed in many papers [?], [Van Tassel 2006], [Rosenblatt *et al.* 2001] IRM solutions are less present in the academic domain. However, a survey on existing solutions [van Beek 2007] exists. Moreover, other researches target insiders leak problems [Yu & Chiueh 2004], usage tracking [Yang *et al.* 2013] and storage efficiency issues [Soliman *et al.* 2015].

c) Advantages and drawbacks of Rights Management

DRM and IRM offer a way to manage and control the usage of documents with high granularity. Moreover, such solutions allow documents to remain protected even when they leave the company, because the authorized / unauthorized actions are carried out with the documents.

However, DRM and IRM have been criticized for different reasons, including privacy concerns [Cohen 2003] [Liu *et al.* 2003] and legislation problems [McCullagh & Homsy 2005]. Technically speaking, IRM often requires extra work to protect existing documents. Indeed, the security (i.e. the usage rules) of a document needs to be specified and is most of the time defined by end-users that might lack consciousness to properly define what can be done with the document.

Just like DLP, IRM solutions target specific implementation of well-known and widely used AC mechanisms, such as Windows ACL or Unix POSIX (e.g. EMC, Microsoft). From the academic perspective, some solutions have been proposed to link Access Control and IRM [Bouwman *et al.* 2008], [Gasmi *et al.* 2008]. However, these solutions are quite specific and do not abstract the problem in order to be as interoperable as possible. Thus, we consider that genericity (C1) is partially covered (see Table 2.5). Just like DLP, IRM solutions are usually not designed to take into consideration coherence (C2). However, efforts on higher abstraction level make the problem of density (C3) partially covered.





















Solution	Type	Domain	Genericity	Coherence	Density	Interoperability	Adaptability	Reactivity
Seclore	IRM	Industry						?
Covertix	IRM	Industry						?
Microsoft IRM	IRM	Industry						?
EMC Documentum	IRM	Industry						?

Table 2.5: Coverage of the main IRM solutions regarding our challenges.

²⁹<https://digify.com/>

This specific compatibility tends to be an obstacle for companies that use different AC implementations or models, especially if several companies or infrastructures (e.g. collaborators, other departments) are involved. Moreover, a company with exotic or self-made AC will have issues linking an IRM with its infrastructure. Adaptability (**C4**) is mostly covered thanks to on-the-fly revocation mechanisms. Finally, interoperability (**C5**) is also mostly covered, thanks to specific implementations. Indeed, IRM solutions are compatible with many software (e.g. Office Suite, Acrobat reader) and file format (.png, .bmp, .jpg). However, the proposed solutions lack abstraction and are too specific to an implementation to consider that the challenges are fully tackled. Finally, reactivity challenge (**C6**) cannot be evaluated, because these solutions are not freely distributed and a licence is required. However, we underline that reactivity is often highlighted by IRM vendors.

In the last two sections, we have seen that in order to protect the usage of a data, Transmission Control and Usage Control mechanisms, such as the one provided by Data Leak Prevention (DLP) and Information Rights Management (IRM) can be used. These mechanisms answer the question *"who can send what to whom?"* (DLP) and *"what can be done with the document once it has been accessed?"* (IRM). However, nowadays solutions suffer from lack of interoperability. Indeed, even if these solutions can be used with well known applications, networks protocols or Operating Systems, they lack abstraction and are not easily usable in exotic or non-standard infrastructures (for instance, a company with self-made applications and Access Control mechanisms).

To overcome these issues, one solution can be to define a solution that embeds the concept of Access Control (AC) and Transmission Control (TC) or Usage Control (UC) in a unified formalism. Such solutions are presented in the next section.

2.5 Unified Access, Transmission and Usage Control with hybrid solutions

As stated previously, we present in this section the solutions that have been proposed to mix Access Control, Transmission Control or Usage Control in unified formalisms.

2.5.1 Organization Based Access Control (OrBAC)

OrBAC [Kalam *et al.* 2003] [Cuppens & Miège 2003] is an extension of RBAC that details permissions in an implementation independent way. To do so, OrBAC uses the concept of organization. An organization is an entity that manages a security policy (for instance, an infrastructure such as company can be considered as an organization). An OrBAC policy is specified at an organizational (i.e. abstract) level. Once the policy is defined, a concrete policy is implemented. Thanks to this approach, the policy expressed in the abstract level can be reused in other

organizations and contexts.

The concrete level manipulates policies that embed the concept of subjects, actions and objects while the abstract level manipulates roles, activities and views. OrBAC being based on RBAC, both models use organizational abstractions to reduce the complexity of the security rules. In OrBAC, however, a role is a set of subjects to which the same security rule applies. Similarly, an activity is a set of actions to which the same security rule applies while a view is a set of objects to which the same security rule apply. **Figure 2.8** gives a representation of OrBAC core concepts.

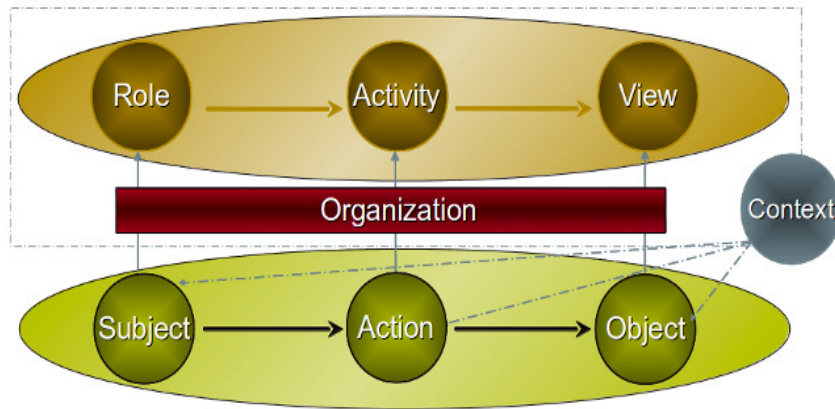


Figure 2.8: OrBAC main concepts

Thanks to its high abstraction, OrBAC has been derived in many versions to tackle different issues including conflict detection [Cuppens *et al.* 2007a], interoperability and deployment in companies workflows [Ayed *et al.* 2008], privacy [Ajam *et al.* 2010a] [Ajam *et al.* 2010b], and interoperability [Coma *et al.* 2008] [Ajam *et al.* 2010b].

In [Cuppens *et al.* 2007b], OrBAC has been modified to tackle the problem of information flow control and confinement problem [Lampson 1973]. Information flow control is defined as the action of enforcing flow policies. In other words, it manages interactions between different components (for instance different users or processes). Indeed, if an information flows from process A to process B, one has to be sure that B privileges allow it to have access to this information. Thus, it can be viewed as a type of Transmission Control, where the information must not be transmitted to unauthorized components.

From the industrial point of view, SWID³⁰ is a company that uses OrBAC to provide IRM like solutions thanks to data marking and data protection features.

³⁰<https://www.swid.fr/>

2.5.2 Usage Control (UCON)

Usage Control (UCON) [Park & Sandhu 2002] [Sandhu & Park 2003] [Park & Sandhu 2004] has been developed to encompass traditional Access Control, trust management and Digital Rights Management (DRM). Thus, UCON improves the traditional Access Control models thanks to the notions of Authorizations, Obligations and Conditions. These notions are presented below.

Authorizations are functional predicates that have to be evaluated for usage decision and return whether the subject is allowed to perform the requested rights on the object. Authorizations can be divided in two parts: pre-Authorization (preA) and on-Authorization (OnA). Pre-Authorization is performed before granting the access to a specific user while on-Authorization is performed during the usage session of a request.

Obligations represent certain actions that need to be completed before granting permission to requested resources. Obligations are divided in two parts: pre-Obligation (preO) and on-Obligation (onO). PreO represents a history function which is checking whether the required activities have been performed or not. OnO represents predicates that must be fulfilled after the user is granted access to the required resource.

Conditions are environmental or system-oriented decision factors. Conditions are also divided in two different sub-parts: pre-Conditions (preC) and on-Conditions (onC), depending on whether the conditions are needed to be fulfilled before or after granting access.

Thanks to these concepts (depicted in **Figure 2.9**), UCON proposes expressivity and flexibility in order to specify Access Control policies. UCON has been followed by many works, tackling policy definition [Hilty *et al.* 2007] or existing enterprise mechanism enforcement [Gheorghe *et al.* 2010].

2.5.3 eXtensible Access Control Markup Language - Data Loss Prevention (XACML-DLP)

XACML³¹ is a XML standard that defines a declarative and attribute-based AC policy. Thanks to its fine-grained granularity and expressiveness, complex Access Control policies can be defined in a simple manner. In October 2014, a new version of XACML has been proposed. This version, named XACML-DLP³², embeds both Access Control and Transmission Control in an unified formalism.

Indeed, XACML-DLP *"defines standard attribute identifiers useful in such policies,*

³¹<https://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html>

³²<http://docs.oasis-open.org/xacml/xacml-3.0-dlp-nac/v1.0/csprd01/xacml-3.0-dlp-nac-v1.0-csprd01.html>

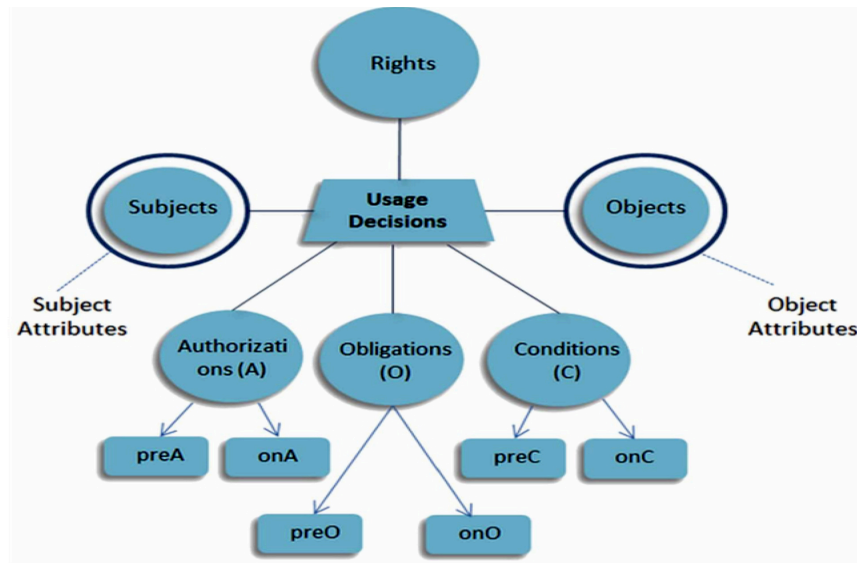


Figure 2.9: UCON main concepts

and recommends attribute value ranges for certain attributes. It also defines several new functions for comparing IP addresses and DNS names, not provided in the XACML 3.0 core specification". [OAS 2014].

Thus, XACML-DLP can be used within an existing XACML context to define Transmission Control policy.

2.5.4 Hybrid solutions advantages and drawbacks

Mixing AC and TC in common formalisms can be very interesting in terms of security and usability.

From the security perspective, having a common formalism can help security experts and administrators to have a better understanding of their policies and be more efficient, because they will have a unique formalism to manage.

From the usability perspective, having one formalism and language to control AC, TC and UC perspectives can ease the way an expert is learning and mastering the formalism and the language. Thanks to that, security experts and administrators can define and maintain policies in an easier way and use the time they have saved in other tasks.

However, "hybrid" solutions such as the one provided by OrBAC, UCON or XACML-DLP also suffer from interoperability problems. Indeed, in order to set one of these solutions in an existing company environment, one will have to learn the new paradigm and redefine the existing Access Control policies (except if the existing ACs are based on XACML and the chosen solution is XACML-DLP). Moreover, the TC or UC part of the policy will still have to be defined as well, inducing time and fatigue that can lead to potential mistakes, incoherence and

security issues. For these reasons, we have considered that most of our challenges are partially covered by these hybrid solutions. Indeed, **Table 2.6** shows that XACML-DLP and UCON are not very interesting to tackle problems such as genericity (**C1**), coherence (**C2**) and density (**C3**). However, OrBAC offers a good coverage for most challenges (including **C4** and **C5**). Unfortunately, OrBAC does not tackle coherence between paradigm (**C2**) and density problem (**C3**) well. In addition, we underline that the complex concepts and the large amount of published papers can discourage a company to use UCON and OrBAC.

Finally, hybrid solutions often take into account reactivity concerns. Thus, we consider the challenge (**C6**) as covered by these solutions.



















Solution	Type	Domain	Genericity	Coherence	Density	Interoperability	Adaptability	Reactivity
XACML-DLP	Hybrid	Open-Standard						
UCON	Hybrid	Academy / Industry						
OrBAC / SWID	Hybrid	Academy / Industry						

Table 2.6: Coverage of the main hybrid solutions regarding our challenges.

2.6 Conclusion

In this chapter, we have first presented Access Control (AC). This mechanism aims at defining *"Who can access what"* and has been proposed within computer systems in the last 50 years. However, current AC models are not efficient against data leakage, a specific type of security issues that can disclose sensitive information and generate major issues for a company. To prevent data leakage, a company can use Data Leak Prevention (DLP) or Information Rights Management (IRM). These solutions offer great features to provide Transmission Control (TC) and Usage Control (UC). While TC defines *"who can send what to whom"*, UC defines *"what can be done with the data once it is accessed"*. However, DLP and IRM lack abstraction and suffer from different issues, including interoperability. Indeed, by using a traditional AC model and a DLP or an IRM, a security expert or administrator will have to define different policies in different paradigms and languages, which can be tiresome and hard to manage, especially within companies with many employees and resources. To overcome these issues, hybrid solutions can be used. These solutions propose to merge AC and TC/UC policies into a unified formalism. The proposed solutions are interesting, but by using them, a company will have to redefine its existing AC policies in a quite complex formalism. Moreover, errors and mistakes can be created during this redefinition, inducing potential data leaks. Finally, the TC policies will still have to be implemented, inducing time-consumption, tiresomeness and also potential coherence problems.

Based on the challenges we have proposed, we have been able to determine that existing solutions do not completely cover our problem (see **Table 2.7**). Thus, we

aim at proposing our own model to take on these challenges. This model is described in the next chapter.

Solution	Type	Main domain	Genericity	Coherence	Complexity	Interoperability	Adaptability	Reactivity
ACL	AC	Academy / Industry	○	○	○	◐	◐	x
Capabilities	AC	Academy / Industry	○	○	○	◐	◐	x
RBAC	AC	Academy / Industry	○	○	◐	◐	◐	x
ABAC	AC	Academy / Industry	○	○	◐	◐	◐	x
MAC	AC	Academy / Industry	○	◐	◐	◐	◐	x
Trend Micro	DLP	Industry	◐	◐	◐	◐	◐	?
McAfee	DLP	Industry	◐	◐	◐	◐	◐	?
Symantec	DLP	Industry	◐	◐	◐	◐	◐	?
Seclore	IRM	Industry	◐	◐	◐	◐	◐	?
Covertix	IRM	Industry	◐	◐	◐	◐	◐	?
Microsoft IRM	IRM	Industry	◐	◐	◐	◐	◐	?
EMC Documentum	IRM	Industry	◐	◐	◐	◐	◐	?
XACML-DLP	Hybrid	Open-Standard	◐	◐	◐	◐	◐	●
UCON	Hybrid	Academy / Industry	◐	◐	◐	◐	◐	●
OrBAC / SWID	Hybrid	Academy / Industry	◐	◐	◐	◐	◐	●
Our solution	Hybrid	Academy	●	●	●	●	●	●

Table 2.7: Summary of how the underlined problems are covered by existing solutions. An empty circle shows that the problem is not taken under consideration while a back circle shows that the problem is completely covered.

Contribution

Contents

3.1 Model presentation	41
3.1.1 Access Control representation	42
3.1.2 Transmission Control representation	45
3.1.3 Coherence definition	47
3.1.4 Generation Mechanisms	48
3.1.5 Inference mechanisms	54
3.1.6 Coherence between AC and TC policies after modifications	58
3.1.7 Mechanisms to tackle the adaptability problem	60
3.1.8 Mechanisms to tackle the interoperability problem	64
3.2 Implementation details of our model	68
3.2.1 Access Control policies	68
3.2.2 Transmission Control policies	68
3.2.3 Conflict detection	69
3.2.4 Coherence checking	70
3.2.5 Business Rule Management System (BRMS)	71
3.2.6 Inferences mechanisms	72
3.2.7 Hypermatrix	73
3.2.8 Reporting	76
3.3 Conclusion	77

In this chapter, we present our contribution in details. As stated previously, our objective is to take on the underlined challenges presented in the previous chapter. To this end, we first present our model by introducing the formalisms we have used to define AC and TC policies. Then, we give implementation details on the proof of concept we have developed.

3.1 Model presentation

In this section, we present our model in details. To do so, we first present a mechanism that aims at generating TC policies based on existing AC rules and the notion of coherence. Then, we present the inferences mechanisms we have defined to reduce the density of the generated TC policies. Finally, we introduce the notion of Hypermatrix and the reporting mechanism.

3.1.1 Access Control representation

In previous chapters, we have underlined that existing solutions lack genericity, forcing a company to redefine its existing AC policies if it wants to use a DLP solution. To overcome this issue, we aim at defining a simple and generic meta-model that can represent commonly used AC models. To this end, we use the common representation of AC (subject, object, action). However, we discard permissions because we consider our meta-model as closed (i.e. we only represent the rules that are authorized). Thus, if an existing rule in the company states that a subject cannot perform an action, this rule will not be represented in our meta-model, and thus it will be considered as unauthorized. We have used the concept of closed system in order to reduce the total number of generated rules. However, we consider that the policies within the company are not contradictory (i.e. we consider that there is no rules saying that a subject can and cannot do something at the same time). Concerning the remaining notions (subject, object and action), we have enhance them with parameters, allowing our meta-model to represent these notions in many ways. For instance, a subject can be represented by a name, a role, a level of accreditation, a location, etc.. This principle can also be applied to objects and actions. Thanks to this simple mechanism, we consider that many AC rules can be represented. However, to transform the existing AC rules in our model, we underline that a specific mechanism (a script for instance), needs to be implemented. We empathize that this can be time-consuming for an administrator, however, we consider that it is less tiresome than having to define the existing AC rules in another formalism.

a) Existing generic Access Control models

Genericity in Access Control has been discussed in previous works, such as [Barker & Stuckey 2003], [Barker 2009], [Slimani *et al.* 2011a] and [Stepien *et al.* 2012]. These formalisms offer a great way to represent Access Control models. However, they can be quite complex, especially for technician (such as administrators) and non-mathematicians or company's employees (such as security experts). Indeed, we have informally asked administrators to evaluate the aforementioned models. Results have shown that these models have been considered them as complicated and not very practical. Thus, we have been inspired by these works and have proposed an easily comprehensible formalism to describe Access Control. Details on this formalism are given below.

b) Access Control List

Just like a traditional ACL, our model represents a policy as a set of rules (3.1). A rule is composed of 3 fundamental things: a Subject, one or several Actions and a Resource (3.2).

$$GenericAC = \langle \sigma_1, \sigma_2, \dots, \sigma_n \rangle, \forall \sigma \in Rules \quad (3.1)$$

$$\sigma = \{s, \langle a_1, a_2, \dots, a_n \rangle, r\}, s \in Subject, a_i \in Action, r \in Resource \quad (3.2)$$

Subjects, Actions and Resources are subsets of Entity (3.3). In our model, an entity is represented by an identifier (for instance a name) and a set of attributes (3.4). An attribute represents for instance a position (ex: role= "CEO") or a security level (ex: securityLevel = "confidential"). An identifier must be unique (3.5).

$$\{Subject, Action, Resource\} \subset Entity \quad (3.3)$$

$$entity = \{identifier, \langle att_1, att_2, \dots, att_n \rangle, \} att_i \in Attribute \quad (3.4)$$

$$\forall e_i, e_j \in Entity, e_i(identifier) \neq e_j(identifier) \quad (3.5)$$

Our model represents an attribute as a pair of key/value (3.6). For a particular entity (ex: subject "Leslie"), two attributes cannot have the same key (3.7).

$$att = \langle key, value \rangle \quad (3.6)$$

$$\forall (att_i, att_j) \in Attribute^2, att_i(key) \neq att_j(key) \quad (3.7)$$

Moreover, duplicates (i.e. two identical rules) cannot be contained in the same generic ACL (3.8).

$$\begin{aligned} & \forall \sigma_1 \in GenericACL, \\ & \nexists \sigma_2 \in GenericACL / s_1 = s_2 \wedge r_1 = r_2, \\ & s_1, s_2 \in Subject, r_1, r_2 \in Resource \end{aligned} \quad (3.8)$$

ACL representation : Like any ACL, our generic ACL can be represented as a two-dimensional matrix where rows represent the subjects and columns represent resources. The row/column intersections represent the action that can be performed by a particular subject. **Figure 3.1** represents such matrix.

Model genericity : With our formalism, we consider that several traditional AC policies can be represented. Indeed, our formalism can easily represent traditional ACL (our formalism is an ACL, with the ability to add any number of parameters to subjects, actions or resources). Moreover, the notion of group in RBAC can be represented as well. In this case, a parameter "group", will be set to subjects. For instance, the following RBAC implementation rules:

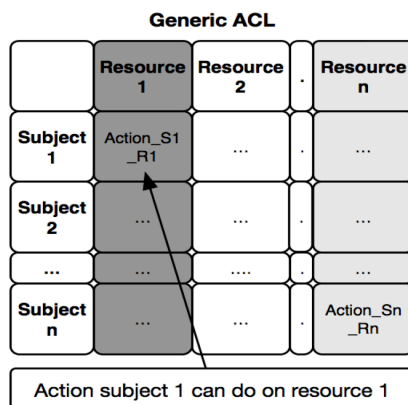


Figure 3.1: Graphical representation of a generic ACL.

```
//RBAC Implementation
name: "prof"
  description: "Group of profs"
  permission_grants:
    resource_uid: "/path/foo"
    permission_types: "read"
```

can be represented in our formalism as follows:

```
//Equivalent Generic AC rules
r1: {{Paul, prof}, <R>, {doc1, "/path/foo"}}
r2: {{Paul, prof}, <R>, {doc2, "/path/foo"}}
r3: {{Jean, prof}, <R>, {doc1, "/path/foo"}}
r4: {{Jean, prof}, <R>, {doc2, "/path/foo"}}
```

Indeed, every subject in the set "*prof*" will be able to read every resources in directory *foo*. In our case, Jean and Paul are professors, and directory "*foo*" contains 2 resources.

Finally, more complex rules, such as the ABAC rule: "*every security expert can access and modify documents marked as confidential*" can also be represented. This rule is equivalent to an enumeration of rules (i.e. Cartesian product) among set "*security experts*" and the set containing all documents marked as confidential. A visual representation of such transformation is depicted in **Figure 3.2**. We underline that the notion of context (for instance, environment conditions in some ABAC

policy) or more complex RBAC notions, such as hierarchical group cannot be easily described by our formalism. However, future works aim at enhancing our model to easily provide such expressiveness.

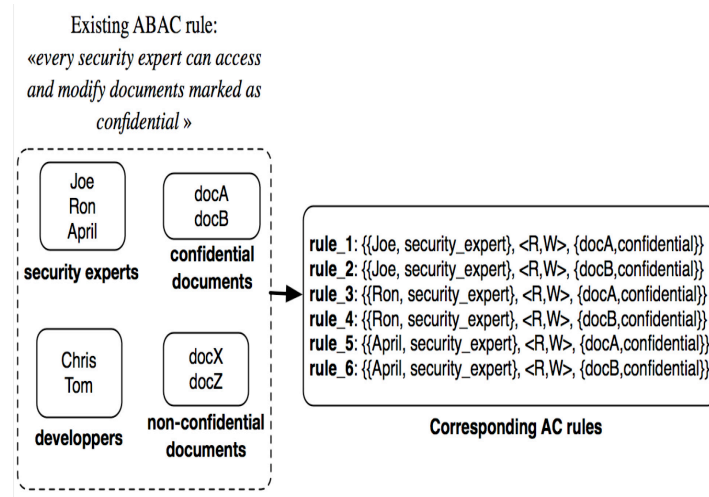


Figure 3.2: Link between the ABAC rule "every security expert can access and modify documents marked as confidential" and the generic AC rules in our formalism.

Model advantages and limitations : As stated previously, our model offers a simple and understandable way to represent traditional AC controls. indeed, based on our formalism, several notions, such as roles or attributes can be represented, allowing a security expert or administrator to represent its existing policies. However, our model lack several things. First of all, it cannot describe contextual information, such as temporal (date, years, moment of the day, etc.) or material (devices, IPs, etc.) information. Moreover, we have considered our model as closed, which means that only the authorized access are represented. Thus, part of the information is destroyed. Indeed, by taking a look at the generated generic AC rules, it is not possible to determine if the existing AC embeds a rule saying that "Ann cannot access the document A" or not. However, these limitations can be considered by modifying the meta-model. Such modifications are discussed in the chapter 5.

Now that the vocabulary used by our generic AC model has been presented, we describe in the next subsections our TC representation.

3.1.2 Transmission Control representation

Transmission Control represents "Who can send what to whom?". To answer this question, our model embeds the concept of Transmission Control List (TCL). A TCL represents the transmissions and actions that can be performed by subjects that already have an access to a resource in the AC policy. Thus, the TCL represents the

exchanges that can be performed on authorized subjects. This subsection describes this concept in details.

a) Transmission Control List (TCL)

Transmission Control List can be described as a set of transmissions regarding one resource (3.9). Thus, a TCL is always related to at least one resource.

In terms of access and retransmission, two resources are equivalent if and only if they share the same set of transmissions (3.10). A transmission embeds the following elements: a source subject (i.e. the sender), a destination subject (i.e. the receiver), the actions of the sender and the receiver (e.g. read, write, execute) and a transmission type (3.11).

$$\begin{aligned} \forall tcl \in TCL, \\ tcl = \{r, \langle \tau_1, \dots, \tau_n \rangle\} \\ r \in Resource, \tau_i \in Transmission \end{aligned} \quad (3.9)$$

$$\begin{aligned} \forall r_1, r_2 \in tcl \\ r_1(\langle \tau_1, \dots, \tau_n \rangle) = r_2(\langle \tau_1, \dots, \tau_n \rangle) \end{aligned} \quad (3.10)$$

$$\begin{aligned} \tau = \langle sender, receiver, senderActions, \\ receiverActions, transType \rangle, \\ sender, receiver \in Subject, \\ senderActions, receiverActions \in Actions, \\ transType \in TransmissionType \end{aligned} \quad (3.11)$$

Transmission Types : A transmission between a sender and a receiver is referred as a transmission type. A transmission type represents if a transmission is authorized (TRANSMISSION_AUTH) or denied (TRANSMISSION_DEN).

Transmission types have the property of completeness, which means that a transmission must contain one and only one transmission type (3.12). In order to ease the readability of the examples presented in this thesis, we have considered as irrelevant the case where a subject send a resource to herself/himself. Thus, a special transmission type will be used in the examples (represented as '-') instead of a TRANSMISSION_AUTH.

$$\begin{aligned} \forall \tau \in Transmission \\ \forall transType \in TransmissionType \\ \exists \tau(transType) / Card(\tau(transType)) = 1 \end{aligned} \quad (3.12)$$

TCL Representation : A two-dimensional matrix can also be used to represent a TCL. In this matrix, rows represent senders while columns represent receivers. Intersections represent the transmission types (for instance TRANSMISSION_AUTH) between the sender and the receiver. We underline that actions of senders and receivers are also conserved in a TCL, due to the formalism defined in (3.11). A TCL matrix can also be mapped to a graph, where vertices represent transmission while each subject is represented by a node. Graphical representation of a TCL and its corresponding graph are depicted in **Figure 3.3**.

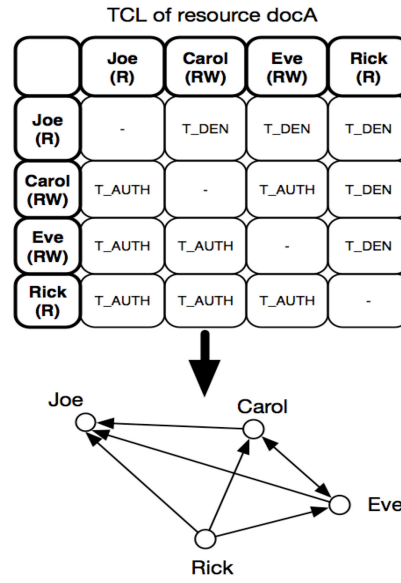


Figure 3.3: Graphical representation of a TCL and its graph. Inside the matrix, rows represent senders and columns represent receivers, while intersections represent the transmission type.

3.1.3 Coherence definition

Our objective is to generate coherent TCLs based on existing ACL. Now that both ACL and TCL have been defined, we present the notion of coherence.

a) Existing works on security policies coherence

Some works have been proposed to tackle incoherences. For instance, [Autrel *et al.* 2008] is a tool that can generate OrBAC rules. One of its features is to detect incoherences between OrBAC rules. Other works, such as Margrave [Nelson *et al.* 2010] can also detect contradictory rules. Margrave supports many standard configuration input languages, such as XACML. However, these tools do not detect incoherences between Access Control and Transmission Control policies.

Thus, we have defined such mechanism. This mechanism is described in the next subsections.

b) Coherence principles

To formalize coherence, we have defined 2 principles:

- **Principle P1:** If a subject can do certain actions on a resource in the ACL (such as read, write or execute), she/he will have the exact same actions in this resource TCL (3.13).
- **Principle P2:** If a resource can be accessed by x subjects in the ACL, the exact same x subjects will be present in this resource TCL (3.14).

$$\begin{aligned}
 P1 : \forall s \in genericAC, \exists s \in tcl \\
 s \in Subject \\
 acl \in genericAC, tcl \in TCL
 \end{aligned} \tag{3.13}$$

$$\begin{aligned}
 P2 : \forall res \in genericAC, res \in Resource, \\
 SubjectsSet_\alpha = SubjectsSet_\beta \\
 SubjectsSet_\alpha = \{sub \in Subject | \exists \sigma, \sigma_i(s) = sub \wedge \sigma_i(r) = res\} \\
 SubjectsSet_\beta = \{sub \in tcl_{res}\} \\
 r \in Resource, \sigma \in genericAC, tcl \in TCL
 \end{aligned} \tag{3.14}$$

Thus, coherence C between an ACL and a specific TCL is true if and only if P1 and P2 are true (3.15).

$$C \Rightarrow P1 \wedge P2 \tag{3.15}$$

While generating TCL based on existing AC policies, the coherence is respected by construction. However, we will see further in the dissertation that incoherence can appear while AC or TC are modified after their generation.

3.1.4 Generation Mechanisms

This subsection presents the main parts of the generation mechanism. This mechanism aims at transforming existing ACL into TCLs.

Creation of the TCL structure : In order to create the general structure of TCLs, the mechanism starts by retrieving all resources described within the ACL. We underline that in order to be coherent, this generation mechanism needs to fulfill the principles P1 and P2 defined previously. To do so, the generation mechanism, depicted graphically in **Figure 3.4**, works as follows: for each resource, the mechanism retrieves every subject that has an explicit access right to a resource (we call

such subjects "*marked subjects*"). Then, the mechanism creates the general structure of the matrices (one matrix per resource) by adding for each row and column the marked subjects as sender and receiver. We underline that the size of the TCL depends on the number of marked subjects. Indeed, a resource that can be accessed by many subjects in the ACL will generate a bigger TCL than a resource with fewer marked subjects. Whatever the accessibility, the number of marked subjects will always be the same in the generic AC and in the corresponding TCL, thus respecting the principle P2. Once the general structure of the TCL is defined, actions are copied without any modification in order to respect the principle P1.

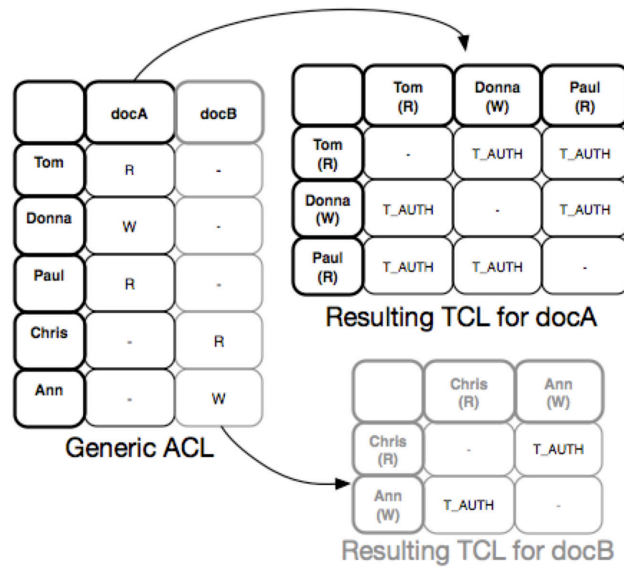


Figure 3.4: Graphical representation of the generation mechanism.

By default, each marked subject can send and receive a document to/from another marked subject. Thus, the TCL matrix can be automatically filled with TRANSMISSION_AUTH. However, we consider that a security expert or administrator can be willing to deny access or specify security properties to a transmission (for instance, deny the transmission for interns). To this end, we have broadened the concept of transmission types with the security properties defined in 3.1.2. Thus, security properties, such as confidentiality (TRANSMISSION_CONF) or integrity (TRANSMISSION_INTEG) can be specified.

In order to allow a security expert to easily specify these transmission types, we have formalized the concept of Mapping Rules (MR).

Mapping Rules : As stated previously, we aim at providing a way for the administrators and experts to define specific transmission types. To do so, we have defined Mapping Rules (MR). A MR can be represented as a function that takes parameters of a sender, actions, receiver and resource and returns a transmission

type (3.16).

$$\begin{aligned}
 &f(sender, senderAction, receiver, \\
 &\quad receiverAction, resource) \\
 &\quad \rightarrow type \\
 &type \in TransmissionType
 \end{aligned}
 \tag{3.16}$$

The mechanism that fills the matrices works as follows: for each intersection of the TCL matrix (i.e. each row/column intersection), our mechanism retrieves all the parameters that concern the sender, the receiver, the action of the sender, the action of the receiver and the resource, then output a transmission type depending on the Mapping Rules that are applied. This transmission type is then applied for this transmission. This mechanism is graphically represented in **Figure 3.5**.

To express Mapping Rules such as "if the resource is docB, the transmission must be confidential", we have defined a meta-language, called Mapping Rule Syntax (MRS). This meta-language is defined below.

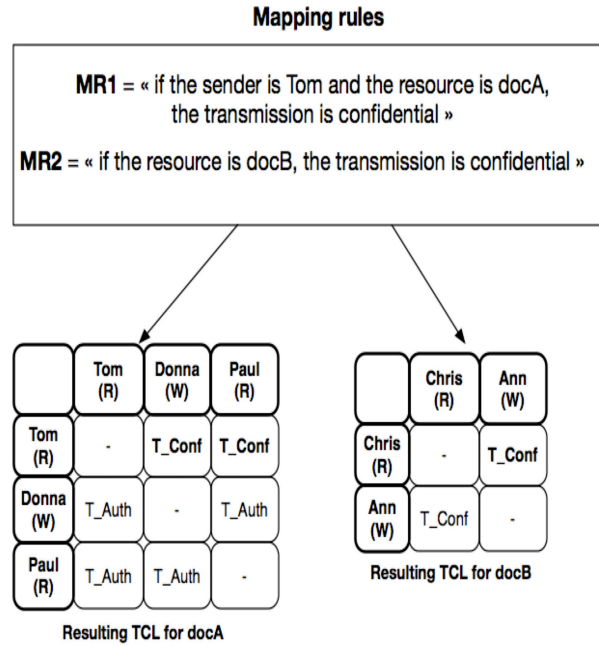


Figure 3.5: An example of the Mapping Rule concept.

Mapping Rules Syntax (MRS) : MRS is based on three different concepts: targets, operators and inputs. A target can be formalized as an entity and an element (3.17). An entity can be a sender, a receiver, an action of the sender, an action of the receiver or a resource (3.18). An element can be represented as an entity identifier (ex: "Ron"), a parameter key (ex: "role") and a parameter value (ex: "manager")(3.19).

$$target = (entity, element) \quad (3.17)$$

$$entity = \{sender, receiver, senderAction, receiverAction, resource\} \quad (3.18)$$

$$element = \{identifier, parameter(key), parameter(value)\} \quad (3.19)$$

MRS uses two types of operators: arithmetic operators and logical operators (3.20):

$$\begin{aligned} arithmeticOperator &= \{=, \neq, <, >, \geq, \leq\} \\ logicalOperator &= \{\vee, \wedge\} \end{aligned} \quad (3.20)$$

Finally, the last component of MRS is the input, which is just a String (i.e. any word in the alphabet \mathcal{A}) (3.21).

$$input \in \mathcal{A}^* \quad (3.21)$$

Thanks to the previous definitions, security experts can define two types of rules: comparative rules and specific rules. A comparative rule is composed of a target, an arithmetic operator, another target and a resulting transmission type (3.22):

$$\begin{aligned} comparativeRule &= senderTarget, \\ &arithmOp, receiverTarget \rightarrow type \\ senderTarget, receiverTarget &\in Target \\ arithmOp &\in ArithmeticOperator \\ type &\in TransmissionType \end{aligned} \quad (3.22)$$

Comparative rules can be used to provide predefined and generic patterns to security experts. For instance, a generic rule could be that "you cannot send a resource to someone with a lower accreditation level than yours". As an example, if we consider that subjects have a parameter "level" describing accreditation levels, the previous generic rule will be:

rule1: (sender, level) > (receiver, level) \rightarrow TRANSMISSION_DEN
--

This generic rule will then be applied to all row/column intersections, for all generated TCLs structures. However, generic rules can be too restrictive. In order to provide fine-grained rules, one can use specific rules. A specific rule is defined by a target, an arithmeticOperator and an input (3.23):

$$\begin{aligned} specificRule &= senderTarget, \\ &arithmOp, input \rightarrow type \end{aligned} \quad (3.23)$$

Specific rules are used to define specific conditions on specific parameter values (for instance, $\text{action} = \text{"Read"}$). For instance, a specific rule such as "*deny the transmission if the receiver is Ann*" will be represented as follows:

rule2: $(\text{receiver}, \text{identifier}) = \text{"Ann"} \rightarrow \text{TRANSMISSION_DEN}$

Moreover, conditions can be combined with logical operators to express more complex rules. For instance, the rule specifying that "*if Garry sends document docZ.pdf to a developer, the transmission is authorized*" can be represented as:

rule3:
 $(\text{sender}, \text{identifier}) = \text{"Garry"} \wedge$
 $(\text{resource}, \text{identifier}) = \text{"docZ.pdf"} \wedge$
 $(\text{receiver}, \text{identifier}) = \text{"developers"} \rightarrow \text{TRANSMISSION_AUTH}$

By using several conditions, one can express complex and powerful rules.

a) Representation of additional information

Based on the TCL formalism, additional information can also be represented. These information can provide safeguard mechanisms for resource usage and retransmission. These computed information, called node types and capabilities, are presented below.

Node Types : A node type represents the type of node a subject can be. Node Type comes in 9 flavors, depicted in **Figure 3.6**. Graphically speaking, the nodes are represented by two symbols: the inner circle and the outer circle. The inner circle represents the total number of incoming edges (i.e. how many subjects can send the resource to a specific node). The outer circle represents the total number of outgoing edges (i.e. how many subjects the resource can be sent to). **Figure 3.7** shows how the inner and outer circles can be composed to represent a node type. As an example, graph in **Figure 3.8** shows that Joe can receive the resource from everyone while he cannot send it to anyone. Thus, Joe has a "*full blackhole*" node type for this particular TCL.

Capabilities : In the ACL, subjects can perform one or more actions over resources. In the TCL representation, we introduce the concept of capability. As stated in related works (2.3.5), capabilities are tokens possessed by users that allow them to perform actions. In our model, a capability is the combination of an ID, an action (ex: "*Read*"), a node type (ex: "*full blackhole*") and a list of resources the

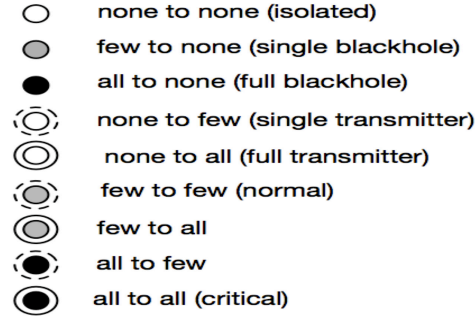


Figure 3.6: Graphical representation of the different node types.






	none (0)	few (1 to n-1)	all (n)
inner circle (number of incoming edges)			
outer circle (number of outgoing edges)	-		

Figure 3.7: Construction of the graphical representation of the node types depending on the number of incoming and outgoing edges (n being the total number of marked subjects without considering the specific node).

capability is applied to (3.24):

$$\begin{aligned}
 \text{Capability} = & \text{identifier}, \text{action}, \text{nodeType}, \\
 & < \text{res}_1, \text{res}_2, \dots, \text{res}_n > \\
 & \text{action} \in \text{Actions}, \text{nodeType} \in \text{NodeTypes} \\
 & \forall \text{res}_i \in \text{Resources}
 \end{aligned} \tag{3.24}$$

Thus, a capability represents what a subject can do in terms of Access Control (i.e. actions) and in terms of Transmission Control and Usage Control (i.e. node-Type). A subject can have several capabilities for a particular resource, but each of her/his capabilities will have the same node type for this resource. For instance, subject Carol for the matrix in **Figure 3.3** will have the following capabilities:

$$\text{Carol} = \{ <001, \text{Read}, \text{normal}, \text{docA}> , <002, \text{Write}, \text{normal}, \text{docA}> \}$$

If Carol can also access resource docB with Read permission and she can receive it from everyone else, but cannot send it to anyone, she will have the following capabilities:

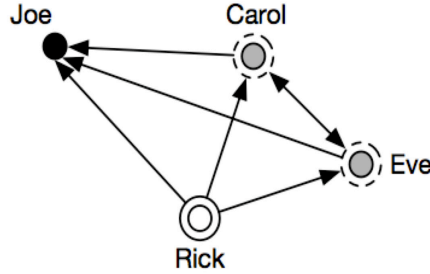


Figure 3.8: An example of some node types within a graph.

Carol = { <001, Read, normal, docA> , <002, Write, normal, docA> <003, Read, full_blackhole, docB> }

Thanks to this representation, our model represents capabilities as the type of Access Control and Transmission Control subjects can perform over resources.

Node types prevent resource propagation inside an infrastructure. Indeed, by giving or removing capabilities to subjects, a company can prevent a subject to retransmit a document by giving her/him a node type that can or cannot send the resource (for instance a single blackhole or a full blackhole). Thus, we consider that this mechanism covers some concerns of Usage Control, in the sense that our capabilities can describe what can be done with the resources once they are accessed. Now that the main concepts of our model have been introduced, we present our inferences mechanisms in details.

3.1.5 Inference mechanisms

We have considered in our research that density (i.e. having policies that manage many entities) can be tiresome for security experts and administrators. Moreover, we have seen in the previous section that our generation mechanism generates many TCLs. Indeed, for every resource contained in the generated ACL, a corresponding TCL is generated. To overcome this issue, we provide inferences mechanisms that can detect similarities between subjects and resources. Thanks to these inferences, resources and subjects are clusterized, reducing *de facto* the numbers of TCLs and the density of capabilities sets that are managed. In this section, we first present some existing works that tackle policies density and complexity. Then we describe the inferences mechanisms we have implemented to solve the problem of density.

a) Existing works to reduce the density

In chapter 1, we have defined density as the ratio between the total number of rules and the number of subjects and resources managed by these rules. We have hypothesized that density can increase the complexity and the tiresomeness of the policy management. Historically speaking, notions such as groups and roles have been implemented to reduce this tiresomeness, followed by other works that are mainly focusing on the estimation of the complexity [Jeager 2001] and the complexity decision [Morisset & Zannone 2014]. Finally, other works have been proposed to infer AC policies thanks to machine learning techniques [Le et al. 2015]. However, this work is interesting but Web oriented and only based on RBAC.

b) Resource similarities

To reduce the density, we aim at putting together similar entities. To do so, we first introduce the resource similarity mechanism. This mechanism is able to clusterize resources that have similar TCLs.

Principle : Two resources are similar if they can be accessed and retransmitted by the same subjects, in the exact same way. In other words, two resources are similar if their TCLs are identical.

Resource Cluster (RC) : A resource cluster can be described as an identifier and a set of resources that share TCLs that are similar (3.25).

$$\begin{aligned}
 ResourceCluster = \{ & identifier, < res_1, \dots, res_n >, tcl \} \\
 & / res_1(tcl_1) = res_2(tcl_2) = \dots = res_n(tcl_m) \\
 & \forall res_i \in Resource \\
 & tcl_j \in TCL
 \end{aligned} \tag{3.25}$$

A resource can be in one and only one RC (3.26), and the identifier must be unique (3.27):

$$\begin{aligned}
 & \forall r \in Resource \\
 & \exists ! i \in \mathbb{N} / r \in rc_i \\
 & rc \in ResourceCluster
 \end{aligned} \tag{3.26}$$

$$\begin{aligned}
 & \forall rc_i, rc_j \in ResourceCluster, \\
 & rc_i(identifier) \neq rc_j(identifier)
 \end{aligned} \tag{3.27}$$

To create a Resource Cluster, our inference mechanism takes all generated TCLs and compares them with each other. If several resources share identical TCLs, a Resource Cluster (RC) is created and these resources are put in the cluster. Then,

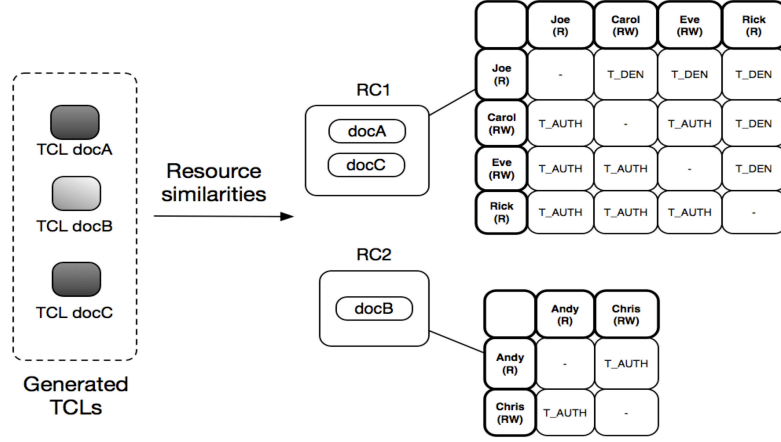


Figure 3.9: An example of the resource similarity mechanism. Here, docA and docC have similar TCLs.

one TCL is linked to the RC while the other similar TCLs are destroyed. In the case where a resource does not have a match, a singleton RC (i.e. a set containing only one element) is created. **Figure 3.9** shows the resource similarities in a more graphical way. In this example, a resource cluster (RC1) is created and both resources are affected to this cluster. Then one of the TCL is affected to the cluster while the other is destroyed. On the contrary, DocB does not share similarities with other resources. In that case, this resource is affected to a singleton RC (RC2). As it can be seen, this mechanism will have 2 TCLs (one per RC) instead of 3 (one for docA, docB and docC).

c) Subject Similarities

Our model embeds capability concepts, allowing subject's actions to be described as a set of capabilities. Thanks to this representation, subjects can be compared and clustered thanks to our subject similarity mechanism.

Principle : Two subjects are similar if and only if they can access and retransmit the exact same resources in the exact same way. In other words, two subjects are similar if they have the exact same capabilities set.

Subject Cluster (SC) : A subject cluster can be described as an identifier and a set of subjects that share the same capabilities sets (3.28).

$$\begin{aligned}
 \text{SubjectCluster} &= \{\text{identifier}, \langle \text{sub}_1, \dots, \text{sub}_n \rangle\} \\
 / \text{sub}_1(\text{setOfCapabilities}) &= \text{sub}_2(\text{setOfCapabilities}) \\
 &= \dots = \text{sub}_n(\text{setOfCapabilities}) \\
 &\quad \forall \text{sub}_i \in \text{Subjects}
 \end{aligned} \tag{3.28}$$

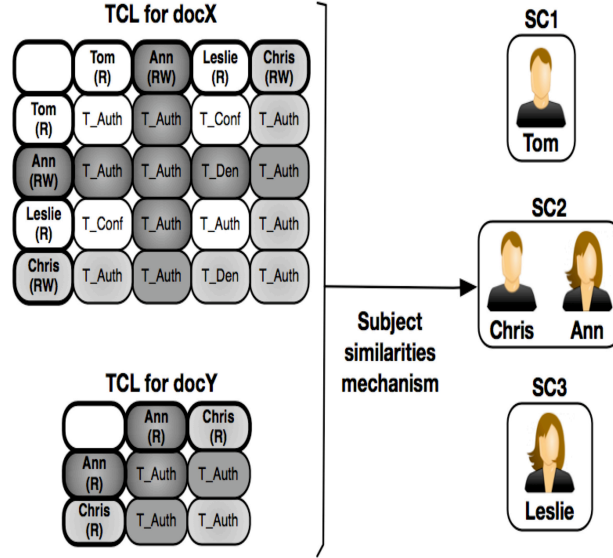


Figure 3.10: An example of the subject similarity mechanism (to facilitate the reader's understanding of the mechanism, "-" symbol has been replaced by T_AUTH)

A subject cannot be in more than one SC (3.29) and each identifier is unique (3.30). Moreover, if a subject is the only one that can access a certain set of resources in a certain way, this particular subject will be put in a singleton cluster.

$$\begin{aligned}
 &\forall s \in Subject \\
 &\exists ! i \in \mathbb{N} / s \in sc_i \\
 &sc \in SubjectCluster
 \end{aligned}
 \tag{3.29}$$

$$\begin{aligned}
 &\forall sc_i, sc_j \in SubjectCluster, \\
 &sc_i(identifier) \neq sc_j(identifier)
 \end{aligned}
 \tag{3.30}$$

To create a subject cluster, our mechanism compares every subject with each other and detects if they share the same set of capabilities. **Figure 3.10** presents the concept of subject similarity mechanism in a graphical way. In this example, Chris and Ann can access and retransmit all resources in the exact same way (and thus, have the same capabilities). Thus, the mechanism creates a subject cluster (SC2), and put both subjects in it. Other subjects, such as Leslie and Tom, does not share similarities. For these reasons, they are affected in singleton clusters (SC1 and SC3).

3.1.6 Coherence between AC and TC policies after modifications

Now that the concept of Clusters has been introduced, we can describe the mechanism that keep the coherence between AC and TC. Indeed, in previous sections, we have seen that generated TCLs are coherent by construction with AC thanks to the principles P1 and P2 defined in (3.13) and (3.14). However, if the ACL or one of the TCLs are modified afterwards, incoherences can appear. For instance, if a new subject is created and manually added in a TCL, she/he will not be in the ACL, violating *de facto* the first principle P1. Secondly, if an existing subject has new capabilities, the corresponding TCLs will have to be modified or the second principle P2 will be violated. To cover this problem and maintain coherence, we introduce in this section a mechanism that keeps coherence of AC and TC policies even when one of them is modified.

a) Operations that can modify the coherence

AC operations : After the generation of the TCLs, modification of the company's existing AC will change the generic ACL. These modifications can create incoherence between the TCLs and the ACL. For instance, adding a new AC rule targeting a new resource will create a new TCL. Moreover, adding a new AC rule targeting an existing subject will modify this subject's capabilities. Finally, deleting an AC rule will modify a TCL (suppression of a row/column, modification of a subject capability, etc.).

TC operations : Because our TC paradigm embeds clusters, Access Control (such as Read) and Transmission Control (such as an authorized transmission "TRANSMISSION_AUTH"), we assume that security experts or administrators would like to manage their policies from the TC perspective rather than from the AC perspective. To do so, our model allows the following operations:

- Create a new subject and add her/him to a specific SC Subject Cluster (SC),
- Create a new subject by giving a set of capabilities,
- Modify a subject's capability,
- Move a subject from a subject cluster to another,
- Delete an existing subject,
- Create a new resource and add it to a specific Resources Cluster (RC),
- Create a new resource by giving a set of capabilities,
- Delete an existing resource.

These operations offer security experts or administrators a way to maintain and update their Transmission Control policies in a quite easy way. But, applying modifications on subjects or resources can impact both AC and TC realms. Indeed, from

the TC point of view, an action (such as creating a new subject) can modify the structure of a TCL.

Thus, our coherence model automatically adapts one realm when the other is modified. The following subsections give details on this mechanism.

b) Coherence mechanism - From AC to TC

In order to have a better understanding of the coherence mechanism, let us take two examples. First of all, imagine that a security expert decides to modify the existing AC by adding a new rule that will modify the generic ACL. As our AC generic model states, the new rule is a combination of a subject, a list of actions, and a resource (see (3.2)). The resource targeted by this new rule can either be an existing one, or a new one.

In the case where it is a new one, a new TCL will be generated. This new TCL can either be identical to a previously generated TCL, or different. In the case where the TCL is different from any other TCL previously generated, the mechanism will generate a new RC, attach the new TCL to it and put the resource in the new cluster. However, if the generated TCL is equivalent to an existing one, only the resource will be added in the corresponding RC and the new TCL will be destroyed. In the case where the new rule targets an existing resource, this rule can modify the TCL by adding a new marked subject or by modifying an already marked subject capability. In this case, the AC rule will modify the TC realm. Indeed, the corresponding TCL will be modified (for instance, in the case of a new subject, a new row and column will be added to the TCL). These modifications will cause the following cascade effect:

- **Resources Cluster readjustment** : the resource targeted by the new rule will probably not be in the same RC anymore (because of TCL modifications). Readjustment needs to be done in order to re-affect the resource. This re-affectation can either be in an existing RC (if the new TCL is equivalent to an existing one), or in a new singleton RC.
- **Subject capabilities updates**: if the new rule is targeting an existing subject, there is a good chance that her/his capabilities will change, inducing an update of her/his capabilities set.
- **Subjects Cluster readjustment** : because of the subject capabilities update, subject membership of an existing SC can become inadequate. A readjustment is thus necessary in order to re-affect the subject in another SC. This SC can either be an existing SC (i.e. one SC where subjects share the exact same set of capabilities) or a new one (if no other subjects have the exact same capabilities set).

An example of such cascade effect is depicted in **Figure 3.11**.

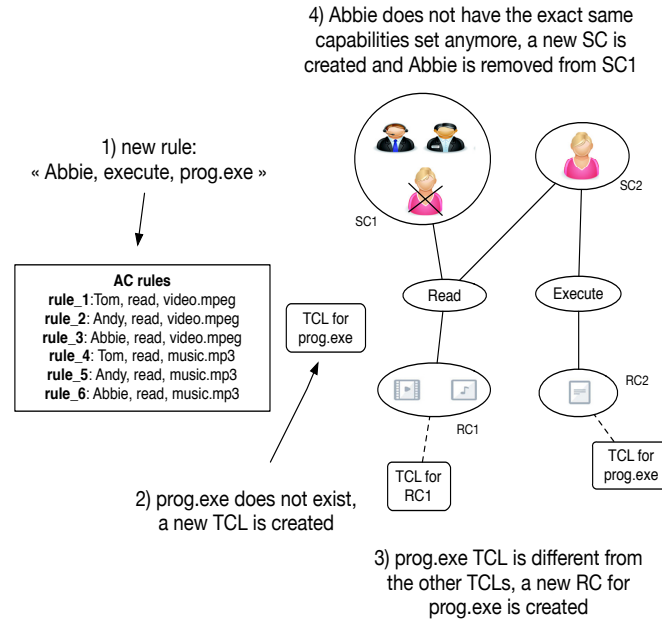


Figure 3.11: An example of the coherence mechanism when a new AC rule is added.

c) Coherence mechanism - From TC to AC

As stated previously, we have assumed that because of the higher abstraction of our TC model, security experts and administrators can be willing to directly use TCLs in order to manage security. Nevertheless, modifications in the TC realm will induce incoherences between TC and AC. Indeed, imagine that a security expert decides to use our TC model to create a new subject and put it in an existing SC. Thanks to her/his membership to this SC, the new subject will have the same set of capabilities than the other members of the SC (for instance, the ability to Read-Write docA.pdf and to send it to some other marked subjects). However, there is no trace of this subject in the existing generic AC (or in the company's existing AC policies). For this reason, the coherence principles will be violated. Thus, this incoherence must be corrected. To do so, our coherence mechanism automatically generates the corresponding AC rules and add them to the existing AC policies, in order to keep coherence between the two realms (see. **Figure 3.12**). Finally, we underline that a mechanism has to apply these changes to the company's existing AC policies.

3.1.7 Mechanisms to tackle the adaptability problem

In this section, we present the mechanisms we have defined to cover the adaptability problem. As stated previously, we define adaptability as the ability for our model to easily modify the overall policies, especially in case of emergencies. To tackle this

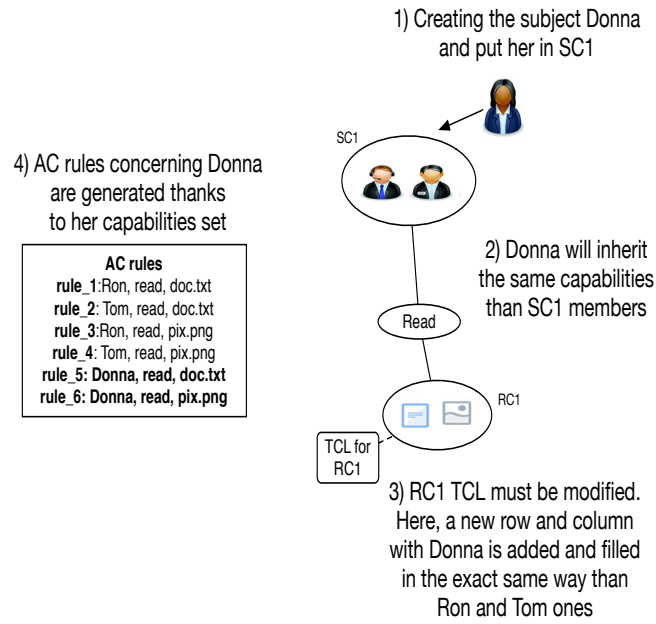


Figure 3.12: An example of the coherence mechanism when the operation of creating a new subject in the TCL realm is processed.

problem, we aim at creating a structure that represents the overall security policy of a company. We have achieved such goal thanks to the concept of **Hypermatrix**. Details on this concept are given below.

a) Existing works on adaptability

Works have been proposed to cover the problem of adaptability [Venkatasubramanian *et al.* 2014]. More specifically, domains such as health-related architectures have been used to underline this issue [Rezaeibagha & Mu 2016] [Marinovic *et al.* 2014]. However, these solutions are not specifically designed for security emergencies, such as attacks or intrusions. Thus, we aim at providing a mechanism that can help security experts to easily change the global security policies in such cases.

b) Hypermatrix representation

Our model defines Subject Clusters (SC), Resource Clusters (RC) and capabilities. Based on these notions, the concept of Hypermatrix can be introduced. Hypermatrix is represented as a set of Transmission Rules (3.31). Each transmission rule is composed of a Subject Cluster (SC), one or more capabilities and a Resource Cluster (RC) (3.32). Thus, an Hypermatrix contains SC as rows, RC as columns and capabilities as intersections. For these reasons, an Hypermatrix is a meta-TCL,

in the sense that instead of having single subjects, actions and resources, an Hypermatrix embeds clusters of subjects, capabilities and clusters of resources. Thus, an Hypermatrix can be described as the conglomerate of all the generated TCLs. Just like traditional TCLs, an Hypermatrix can be represented as a matrix and a graph (see **Figure 3.13**).

$$\begin{aligned} \text{Hypermatrix} = & \langle \omega_1, \omega_2, \dots, \omega_n \rangle, \\ & \forall \omega \in \text{TransmissionRules} \end{aligned} \quad (3.31)$$

$$\begin{aligned} \text{TransmissionRules} = & \langle sc, \{c_1, c_2, \dots, c_n\}, rc \rangle \\ & sc \in \text{SubjectClusteer} \\ & c_i \in \text{Capability} \\ & rc \in \text{ResourceCluster} \end{aligned} \quad (3.32)$$

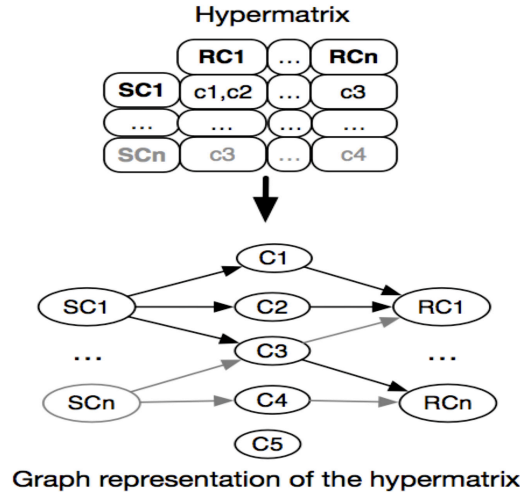


Figure 3.13: Graphical representation of an Hypermatrix and its corresponding graph.

The purpose of an Hypermatrix is to represent all actions and transmissions that all clusters of subjects can perform over all clusters of resources. In other words, an Hypermatrix represents every AC and TC rules allowed or denied by the company. As an abstract representation, Hypermatrices are very interesting to represent the overall security policies of a company.

Use case : Our model generates TC policies based on existing AC rules. To do so, the model uses Mapping Rules, based on our Mapping Rules Syntax (MRS). For a set of specific Mapping Rules, an Hypermatrix will be generated. Thus, different Mapping Rules applied on the same AC policies can lead to different

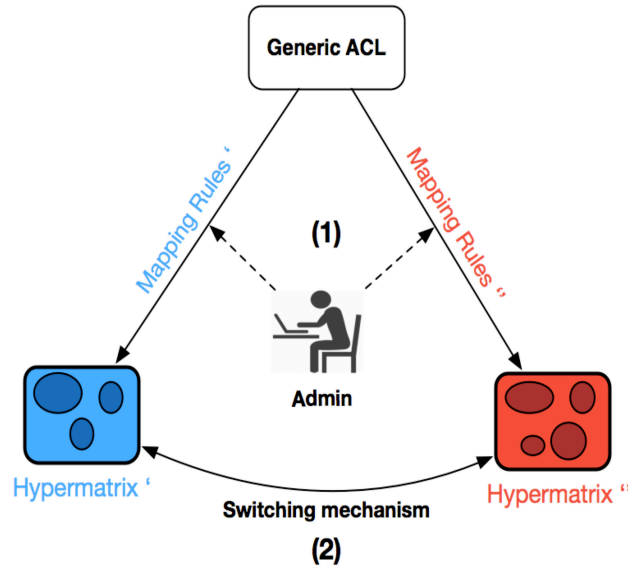


Figure 3.14: Graphical representation of the Hypermatrix generation and switching. Thanks to different Mapping Rules, an administrator can generate several Hypermatrices (1). Then, Hypermatrix can be switched at any time (2), modifying the overall policies of the company.

Hypermatrices. To give an example, imagine that a security expert or administrator defines 2 sets of Mapping Rules. For this example, the two sets are singleton and thus, contain only one rule.

The rule of the first set says that *"every transmission is authorized"* while the rule in the second set says that *"every transmission is denied"*. With these two rules, two different sets of TCLs will be generated, and thus, two different Hypermatrices will be created. The first one will have subjects with capabilities containing critical nodes (because everyone will be able to send and receive resources to/from everyone). Similarly, the second Hypermatrix will have subjects with capabilities containing isolated nodes (because no one will be able to send or receive resources). The security expert or administrator will then be willing to switch from one Hypermatrix to the other, depending on the status of the security within the company. Indeed, the first Hypermatrix could be used as a *"normal"* or *"day to day"* policy while the second one could be used during security emergencies, because it is more restrictive in terms of exchanges.

Thanks to the concept of Mapping Rules Syntax and Hypermatrices, our model is able to generate different sets of policies and can easily switch from one to another. Such switching mechanism can be interesting to easily apply more or less strict policies, especially in case of emergencies. **Figure 3.14** shows the concept of Hypermatrices switching in a graphical way.

3.1.8 Mechanisms to tackle the interoperability problem

We have considered in our working hypothesis that interoperability (i.e. differences between security policies of different companies or infrastructures) can be a problem for data exchanges and business opportunities. Thus, we aim at providing a mechanism that can generate reports based on our generated rules. We hypothesize that these reports can help a security expert or an administrator to have a better understanding of her/his policies and those of collaborative infrastructures. Thanks to this knowledge, we aim at making interactions between infrastructures easier and more resistant to unintentional data leakage.

a) Existing works on security policies interoperability within collaborative infrastructures

Several works have been proposed to tackle the problem of security in collaborative infrastructures. For instance, [Tolone *et al.* 2005] have proposed a survey of different AC (e.g. ACL, RBAC) in order to determine advantages and disadvantages of existing models. Moreover, this proposal has formalized the different factors that a model should have to be efficient in collaborative systems. Specific AC solutions have been proposed to tackle Cloud [Bouchami & Perrin 2014] and workflow-oriented architecture [Slimani *et al.* 2011b]. However, these solutions do not offer mechanisms that can enhance or improve existing AC policies. To tackle these problems, [Hur 2013] [Daud *et al.* 2015] have been proposed. Despite the fact that these solutions are interesting, we aim at providing a solution that can help security experts and administrators to have a better understanding of their policies and those used by their collaborators. Thus, we aim at computing information from our model in order to increase the overall knowledge of a security expert and facilitate the exchange of data between infrastructures. Information on how the additional computing is performed is given below.

b) Reporting by comparing TCLs

Additional information can be computed on a TCL. For instance, the following information can be retrieved:

- The total number of marked subjects (i.e. the total number of subjects that can access this resource).
- The different types and number of capabilities (i.e. how many subjects can access this resource in a certain way).
- The different types and number of transmission types (i.e. how many subjects can send or receive this resource in a certain way).

The previous information can be interesting to compare 2 TCLs that can represent the same resource (or set of resources) in two different infrastructures. For instance let us imagine a company located on two sites (sites A and B), each of

them having its own sets of TCLs. If these two sites share a document (for instance docX.pdf), both sites will have a TCL describing what can be performed on this document. One can be willing to compute additional information and compare these TCLs to determine the similarities and differences between the TCLs (and thus, to determine how the document docX.pdf is treated in both sites). Thanks to this additional information, our mechanism is able to generate a report. This report shows similarities and differences between security rules of site A and B. By taking a look at this report, a security expert can be aware of potential security indulgences between site A and site B and correct them. **Figure 3.15** shows a graphical representation of such mechanism.

c) Reporting by comparing Hypermatrices

As stated previously, comparing TCLs can offer reporting mechanism to determine how a resource (or a subset of resources) is managed within two infrastructures. However, we consider that the security expert or administrator can be willing to compare the overall security policies of an infrastructure rather than a subset of resources. To do so, one solution can be to generate a report for every single TCL of the company, which can be difficult and tiresome, especially when many TCLs are managed. A more practical solution is to use the comparison mechanism on Hypermatrices. Indeed, the point of an Hypermatrix is to offer an higher abstraction. Thus, by comparing Hypermatrices, analyses can determine for a infrastructure:

- The total number of subjects.
- The total number of resources.
- The total number of Subject Clusters (SCs).
- The total number of Resource Clusters (RCs).
- Various information on every SC and RC (how many subjects? how many resources? etc.).
- The transmission types repartition (how many TRANSMISSION_AUTH, TRANSMISSION_DEN, etc.).
- The capabilities and node types repartition (how many full transmitter, full blackhole, etc.).

Based on the aforementioned information, a report is generated (see **Figure 3.16**). This report can serve two purposes. First, it can help comparing 2 Hypermatrices that have been generated by the same infrastructure. Thanks to this comparison, an expert can validate the different policies of her/his companies (for instance, she/he can be sure that the "*emergency*" policies are more strict than the "*day to day*" policies).

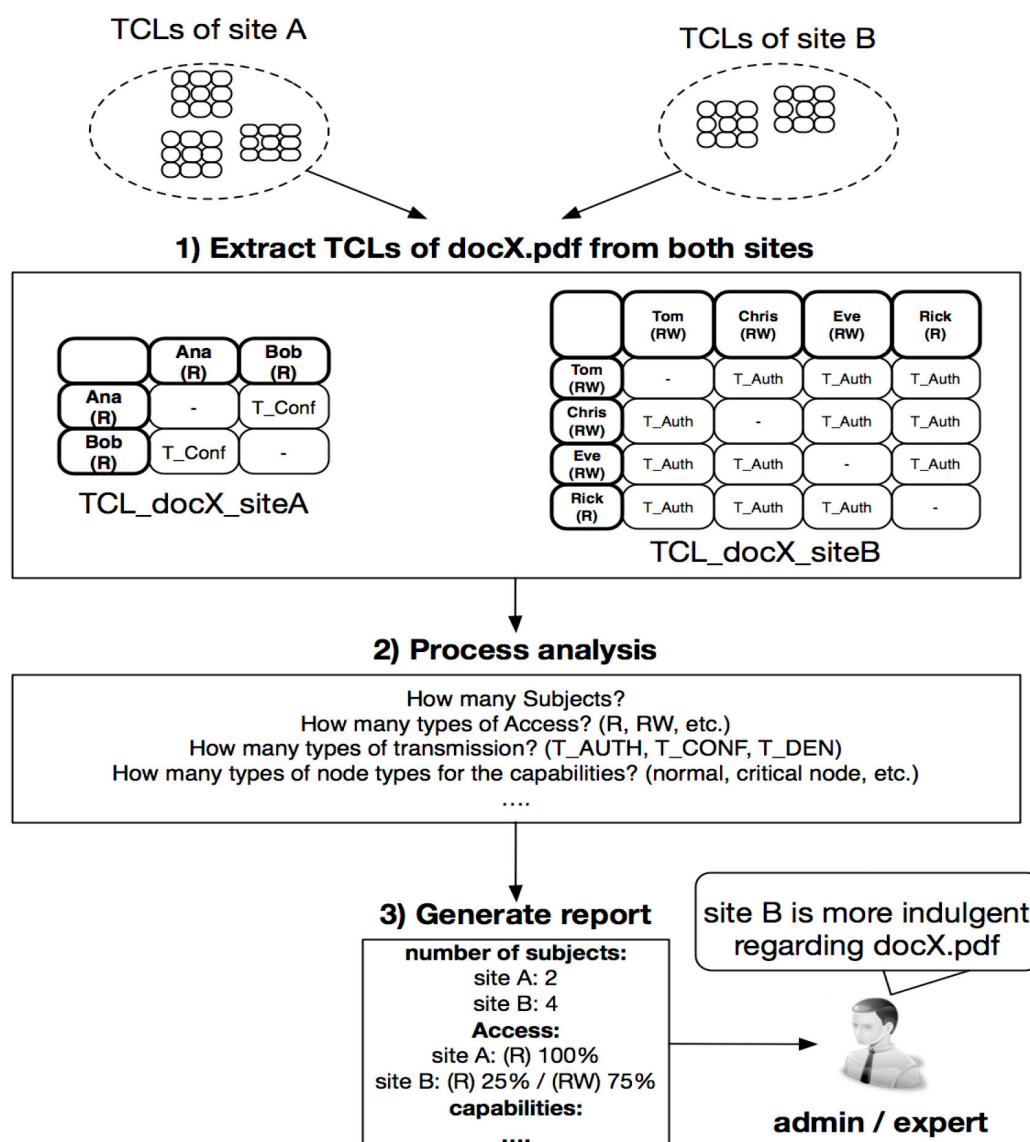


Figure 3.15: Graphical representation of the TCL comparison mechanism. Thanks to this mechanism, a report can be generated. This report can help security experts and administrators to spot indulgences within 2 TCLs.

Secondly, the report can spot differences between 2 Hypermatrices that have been generated within 2 different infrastructures. Just like TCLs comparison between 2 infrastructures, this mechanism can help security experts and administrators to have a better understanding of the overall security policies applied in both infrastructures. Moreover, we have implemented this mechanism as a framework, thus, we underline that it can be improved and adapted, depending on the needs of the security experts or administrators. For instance, a company could

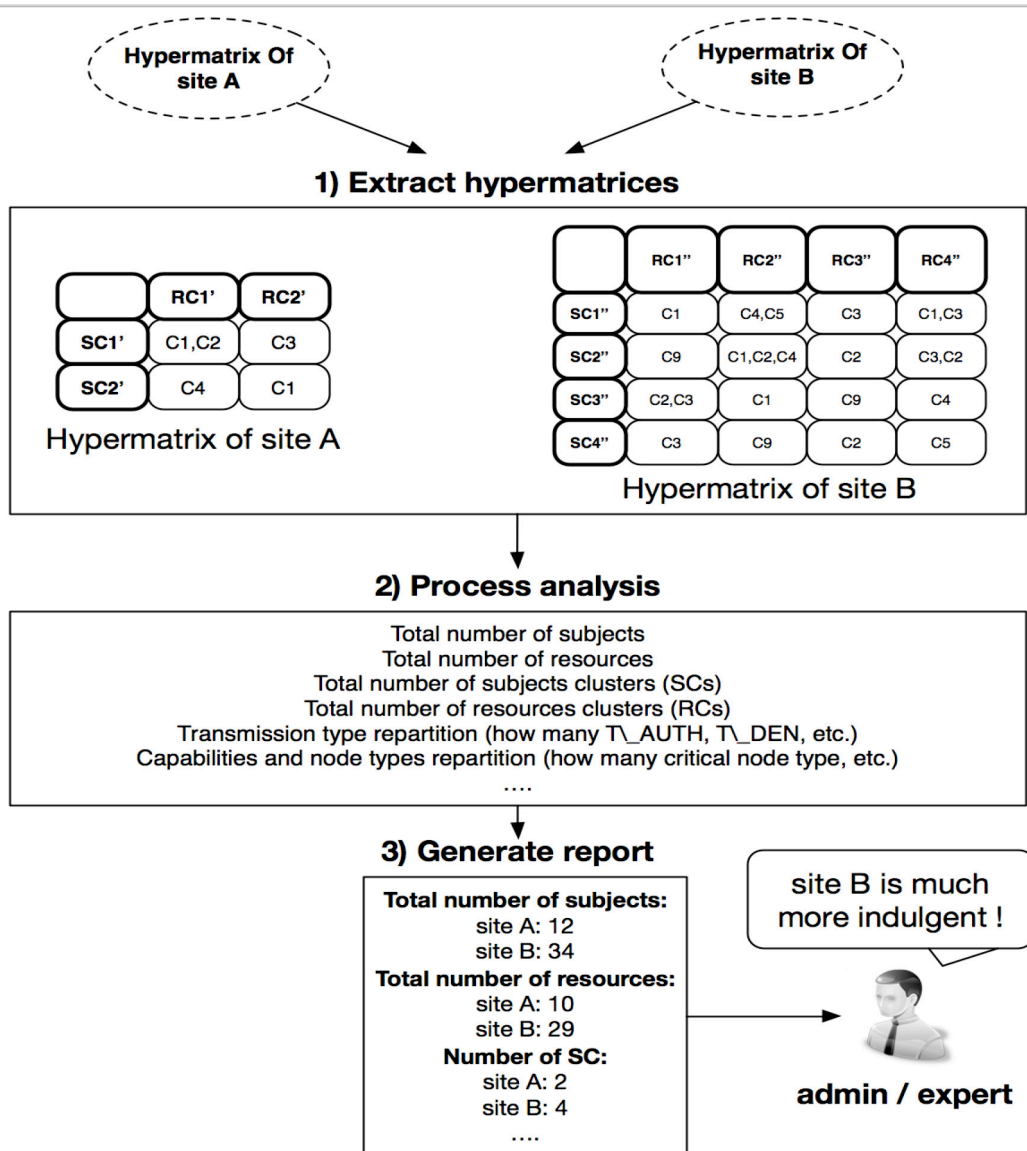


Figure 3.16: Graphical representation of the Hypermatrix comparison mechanism.

be willing to focus on transmission types differences, while another one could be interested in who (i.e. number and types of subjects) accesses the documents.

Now that the model has been presented, we discuss in the next section the implementation details of our proof of concept.

3.2 Implementation details of our model

In this section, we give technical details on the implementation of our model. More specifically, we first discuss the AC policies representation. Then we give details on the Transmission Control Lists and the coherence mechanisms.

3.2.1 Access Control policies

In order to represent AC rules in our formalism, we have used XML. **Figure 3.17** depicts an example of a rule saying that "user *Quentin* can read document *docA*".

```

▼<rule name="new rule">
  <subject subjectName="quentin" subjectLocation="Paris" subjectPosition="intern"/>
  <action actionName="Read" actionType="non-critical"/>
  <resource resourceLevel="confidential" resourceName="docA" resourceType="image"/>
</rule>

```

Figure 3.17: An example of a rule in our XML format.

Because we need to use sets of subjects, actions and resources to create ACL and TCLs, we have decided to push the rules in a database to use the power of SQL instructions to easily retrieve this information. To do so, we have used PostgreSQL¹ for its reliability and scalability. The database we have constructed is composed of one table containing the ID of the rule, the subject, the actions and the resource. For the ACL generation, we have used an SQL request to retrieve all resources of the AC policy. Then, this set of resources is divided into subsets and given to threads (we have used Amdahl's law principle [Hill & Marty 2008] to parallelize our algorithms). Each thread creates columns of the ACL and a scheduling mechanism retrieves the results and generates the overall ACL as a Java object.

Concerning the database, we have parallelized the portion of code that pushes the AC rules directly into it. To do so, a scheduling mechanism divides the total set of rules in portions and every subset is given to a thread. Then, each thread pushes the rule in the database thanks to prepared statements in order to reduce the time-consumption.

3.2.2 Transmission Control policies

Transmission Control Lists are also Java objects. For the TCL generation mechanism, an algorithm retrieves the set of all resources in the ACL. Then, these resources are distributed into several threads. Each thread browses the generic AC and retrieves the rules targeting its resources thanks to prepared statements and generates

¹<https://www.postgresql.org/>

the TCL for this resource based on the applied Mapping Rules. Then, the TCL are serialized.

Graphical representation : In order to offer a visual feedback, we have implemented a visualisation tool. This tool generates graphs that represent a TCL (where nodes represent subjects and edges represent the type of transmissions that can be performed). The following TCL graphs have been generated with this tool.

3.2.3 Conflict detection

We have defined in subsection 3.1.4 that Mapping Rules conditions can be combined to provide more complex and expressive rules. However, combining conditions can lead to multiple transmission types results, which is forbidden by our formalism defined in (3.12). Indeed, imagine for instance that a security expert defines two different rules **r1** and **r2**, where **r1** defines that *"When managers are sending a resource, the transmission must have confidentiality property"* and **r2** defines that *"John cannot send docA.pdf"* (even if he has access to it in the ACL). Imagine now that John is a manager. The mapping mechanism will have issues deciding which transmission type to apply for every element in the row *"John"* for the TCL of docA.pdf. Indeed, for this resource, the system will not be able to determine if the resource can be sent (**r1**) or not (**r2**). To overcome this issue, several mechanisms have been proposed.

The first mechanism we have proposed notifies the security expert of the inconsistency and ask her/him for an answer. She/he can choose the transmission type of her/his choice, or implement an *ad hoc* rule, by combining conditions for instance. However, many popup notifications can be tiresome and generate fatigue. To avoid this issue, we have proposed a Decision Strategies (DS) mechanism. To use DS, a security expert first needs to set levels for transmission type. For instance, the following levels can be specified:

Level_1: TRANSMISSION_AUTH < Level_2 : TRANSMISSION_CONF <
Level_3: TRANSMISSION_DEN

Once levels have been defined, the security expert can use one of the following decision strategies:

- HIGHEST: apply the transmission type with the highest level.
- LOWEST: apply the transmission type with the lowest level.
- MOST PRESENT: apply the transmission type which is the most present in the sequence of rules.
- DEFAULT: apply the default transmission type.

In our example, the following rule will be applied, depending on the strategy:

Strategy	Applied rule
HIGHEST	r2
LOWEST	r1
MOST PRESENT	No answer, DEFAULT applied
DEFAULT	TRANSMISSION_DEN

3.2.4 Coherence checking

We have seen that once the AC or the TC paradigm are modified, the other one needs to be adapted. Thus, modification in one side need to be adapted in the other side. To do so, we have implemented a mechanism that automatically adapt one paradigm when the other is modified. Details on this mechanism are given below.

Adding a new AC rule : Let us imagine that the rule depicted in **Figure 3.17** is added to an existing AC policy. If user Quentin is a new subject, our coherence mechanism will generate a new Java object, containing corresponding capabilities (i.e a Read action and the "all to all" node type). Due to this new rule, the TCL of docA must be modified. Indeed, a new row and column need to be created and filled with subject "Quentin" in order to maintain the coherence. To do so, the coherence mechanism modifies the TCL depending on the capabilities of Quentin and the Mapping Rules that are applied. Then, the coherence mechanism verifies that this TCL still belongs to the current RC. **Figure 3.18(A)** and **Figure 3.18(B)** show transmissions between subjects that have access to docA before and after the new rule. As one can notice, Quentin is a new marked subject for docA, proving that the rule has been taken into account in order to maintain the coherence between AC and TC paradigms.

Moving a subject from a SC to another : As another example, let us take the case where a subject is moved from a Subject Cluster (SC) to another, modifying *de facto* her/his capabilities. **Figure 3.19(A)** shows that subject Xavier has access to the Resource Cluster RC_3088906, and thus, to docB. By moving Xavier to another SC, he is now able to access docA and docX through RC_3088905 (see **Figure 3.19(B)**), without being able to access to docB anymore (of course, these modifications need to be propagated to the existing AC policies for the process to be complete).

Moreover, **Figure 3.20** shows the AC rules that have been generated and added to the existing AC policy in order to take into account Xavier's modifications and maintain coherence between AC and TC paradigms. Finally, coherence principles P1 and P2 can be validated algorithmically. Details on these algorithms are presented below.

P1 and P2 validation : As stated previously, P1 and P2 are the two coherence principles we have defined. P1 states that *"If a subject can do certain actions on a resource in the ACL (such as read and write), she/he will have the exact same*

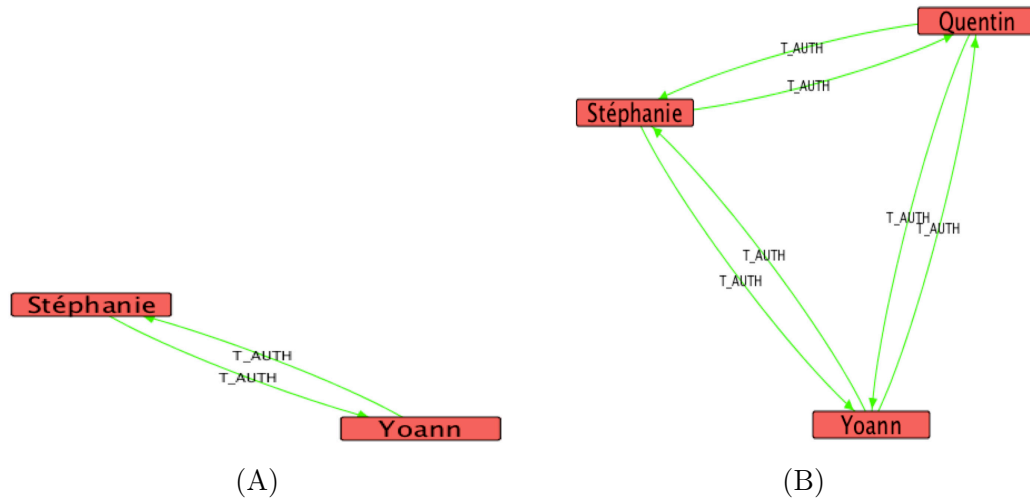


Figure 3.18: TCL representation of docA before (A) and after (B) the new AC rule. As one can see, a new marked subject has been added for this resource. This subject is able to send and receive the resource to and from the other marked subjects.

actions in this resource TCL". P2 states that "If a resource can be acceded by x subjects in the ACL, the exact same x subjects will be present in this resource TCL". Thanks to PostgreSQL, we validate algorithmically these principles. Indeed, in the case of P1, SQL requests are used to check that it exists a rule such as that the 3-tuples (subject,action,resource) exists. If it is the case for every subject and action of a resource, P1 is valid for this resource. Such SQL request is depicted in **Figure 3.21**.

In the case of P2, an SQL requests with COUNT and DISTINCT are performed to verify that the correct number of subjects have accessed to the specific resource. Based on this result, the algorithm compares if the same subjects are present in the TCL. If it is the case, it means that principle P2 is valid for this resource. Such request is depicted in **Figure 3.22**.

We underline that these validations are performed every time a modification on AC or TC paradigms occurs.

3.2.5 Business Rule Management System (BRMS)

In order to implement this part, we have decided to use two different implementations. The first one is an home-made mechanism to define and apply the Mapping rules. This mechanism has been used to quickly test the feasibility of our idea. However, we have also proposed a Business Rule Management System (BRMS) to define the concept of Mapping Rules. BRMS is a system that is used to define, deploy, execute, monitor and maintain the complexity of a decision logic. In our

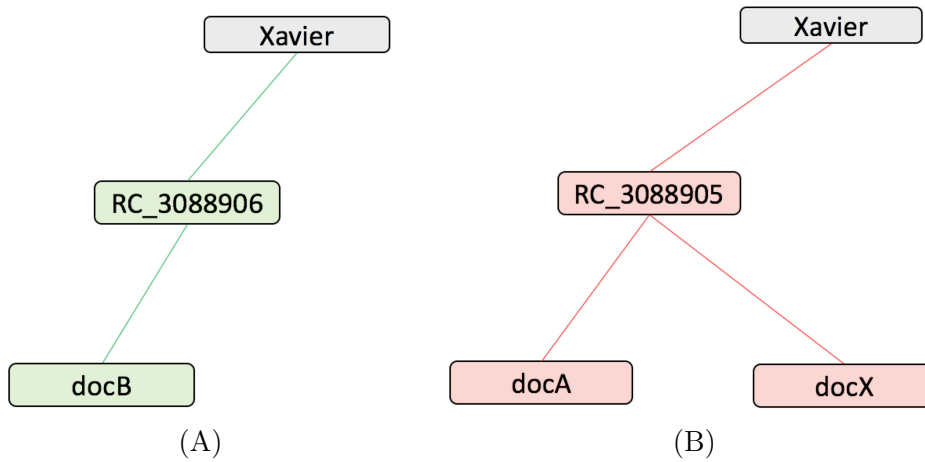


Figure 3.19: Graphical representation of Xavier's capabilities before (A) and after (B) modification. In (A), Xavier has access to the resource docB (which is in resource cluster RC_3088906). In (B), Xavier's capability has changed, he now has access to docA and docX, both members of resource cluster RC_3088905.

case, we have used Drools² [Proctor *et al.* 2008] to define Mapping Rules thanks to its inference based rules engine. Based on an enhanced implementation of the Rete algorithm [Forgy 1982], Drools is scalable, efficient and widely used in companies. Moreover, it is easily understandable. Indeed, **Figure 3.23** shows an example of a Mapping Rules implemented with Drools.

3.2.6 Inferences mechanisms

The resources inferences mechanisms consist at comparing TCL matrices. Because our TCL matrices are Java objects, our resource similarity mechanism compares the TCLs with toString like mechanisms. In order to reduce the time-consumption, we have parallelized the resource similarity mechanism. To do so, we have been inspired by map reduce mechanisms [Dean & Ghemawat 2008]. Indeed, we have implemented a scheduling mechanism that divides the total set of TCLs into several subsets. These subsets are distributed into threads that locally compare the resources and create resources clusters (RCs) if multiple resources share the same TCL. Finally, these local RCs are retrieved by the scheduling mechanism and merged if they are similar to others. Concerning the subject similarity mechanism, a simple comparison of the subjects (i.e. Java objects) is performed. We underline that parallelization could also be used to enhance the time efficiency of this mechanism.

²<http://www.drools.org/>

```

▼<rule name="generated_1">
  <subject subjectName="Xavier"/>
  <action actionName="Read" actionType="non-critical"/>
  <resource resourceLevel="confidential" resourceName="docA" resourceType="image"/>
</rule>
▼<rule name="generated_2">
  <subject subjectName="Xavier"/>
  <action actionName="Write" actionType="critical"/>
  <resource resourceLevel="confidential" resourceName="docA" resourceType="image"/>
</rule>
▼<rule name="generated_3">
  <subject subjectName="Xavier"/>
  <action actionName="Read" actionType="non-critical"/>
  <resource resourceLevel="confidential" resourceName="docX" resourceType="image"/>
</rule>
▼<rule name="generated_4">
  <subject subjectName="Xavier"/>
  <action actionName="Write" actionType="non-critical"/>
  <resource resourceLevel="confidential" resourceName="docX" resourceType="image"/>
</rule>

```

Figure 3.20: AC rules that have been automatically generated after Xavier modification.

```

select count(*) from acl
where acl.action like '%Read%'
and acl.subject like '%Julie%'
and acl.resource like '%Copil_12-06-16%'

```

Figure 3.21: SQL request to validate if a specific rule exists. This rule is useful to validate coherence principle P1.

```

select count(distinct(acl.subject)) from acl
where acl.resource like '%RH_2015'

```

Figure 3.22: SQL request to retrieve the total number of marked subjects for the resource RH_2015. This request is useful to validate coherence principle P2.

3.2.7 Hypermatrix

To create Hypermatrices, we have implemented a mechanism that retrieves every subject and TCL matrix and generates the Hypermatrix. To give an example, let us take two ACLs (ACL_alpha and ACL_beta). These ACLs have different numbers of subjects and resources (see **Table 3.1**). Imagine now that 3 different Hypermatrices are generated based on these ACLs (i.e. HA1, HA2 and HB).

The first Hypermatrix (HA1) acts as a normal policy for a company A while another

```

dialect "java"

rule "If the resource is docA, transmission is Authorized"

    when
        //conditions
        itemRes : Resource(getResourceCommonName() == "docA")
        itemTrans : Transmission()

    then
        //actions
        itemTrans.setType("TRANSMISSION OK");

end

```

Figure 3.23: An example of a simple Mapping Rule in Drools.

ID	Number of subjects	Number of resources
ACL_alpha	50	500
ACL_beta	100	3000

Table 3.1: ACLs used for the evaluation

Hypermatrix represents the "*emergency*" policy (HA2) for this same company. The last Hypermatrix (HB) represents the policy of another company B. These information are summarized in **Table 3.2**.

Imagine now that the administrator of company A uses the following Mapping Rules on ACL_alpha in order to define her/his day to day policies for company A:

- *MR1: "if the sender is a manager, the transmission has to be confidential."* (i.e. T_CONF)
- *MR2: "if the receiver's site is located in the city of Nice, the transmission is denied."* (i.e. T_DEN)
- *MR3: "if the sender is Chris, integrity check needs to be performed on the transmission."* (i.e. T_INTEG)

Name	Based on	Description
HA1	ACL_alpha	day to day policies of company A
HA2	ACL_alpha	emergency policies of company A
HB	ACL_beta	day to day policies of company B

Table 3.2: Summary of the generated Hypermatrices

- *Default: "if the previous conditions are not applicable, the default transmission is authorized."* (T_AUTH)

The generated TCLs will be used to create an Hypermatrix HA1. As stated previously, this Hypermatrix will act as a day to day policy.

Then, the same administrator decides to generate another Hypermatrix (HA2), based on more restrictive Mapping Rules. These rules are the following:

- *MR1a: "Only managers can send to managers with confidential transmission."* (i.e. T_CONF)
- *Default: "If the previous condition is not applicable, the default transmission is denied."* (T_DEN)

The generated TCLs will be used to create another Hypermatrix (HA2), more restrictive than the first one. This Hypermatrix HA2 will be used in case of emergencies by company A.

Imagine now that another administrator inside company B uses ACL_beta in order to generate the third Hypermatrix (HB). To do so, this second administrator uses the following Mapping Rules:

- *MR1b: "if the sender and the receiver have the same position, the transmission is authorized."* (i.e. T_AUTH)
- *Default: "If the previous conditions are not applicable, the default transmission is denied."* (i.e. T_DEN)

Once the Hypermatrices have been generated, switch between them can be performed. Information of this switch is given below.

Hypermatrices switch : During an emergency, the administrator needs to switch from Hypermatrix HA1 to Hypermatrix HA2. Thanks to our proposal, this operation can easily be performed by the administrator. Indeed, a simple command line can be used to load another Hypermatrix (i.e. Java object). In order to be sure that the switch has been performed, the administrator can use the graphical tool we have implemented. Indeed, **Figure 3.24** shows the state of the TCL of docA before and after the switch. As one can see, Hypermatrix HA1 (**Figure 3.24(A)**) allows subjects Rick, Chris, Abbie and Leslie to exchange the resource with various transmission types (i.e. T_AUTH, T_CONF, T_INTEG). However, Hypermatrix HA2 only allows Rick and Leslie to exchange docA (**Figure 3.24(B)**). This can be explained by the fact that Rick and Leslie are managers and the rule MR1a states that only managers can exchange resources in case of emergencies. Thus, the administrator of company A can validate that the Hypermatrix has been switched and that more restrictive policies are applied.

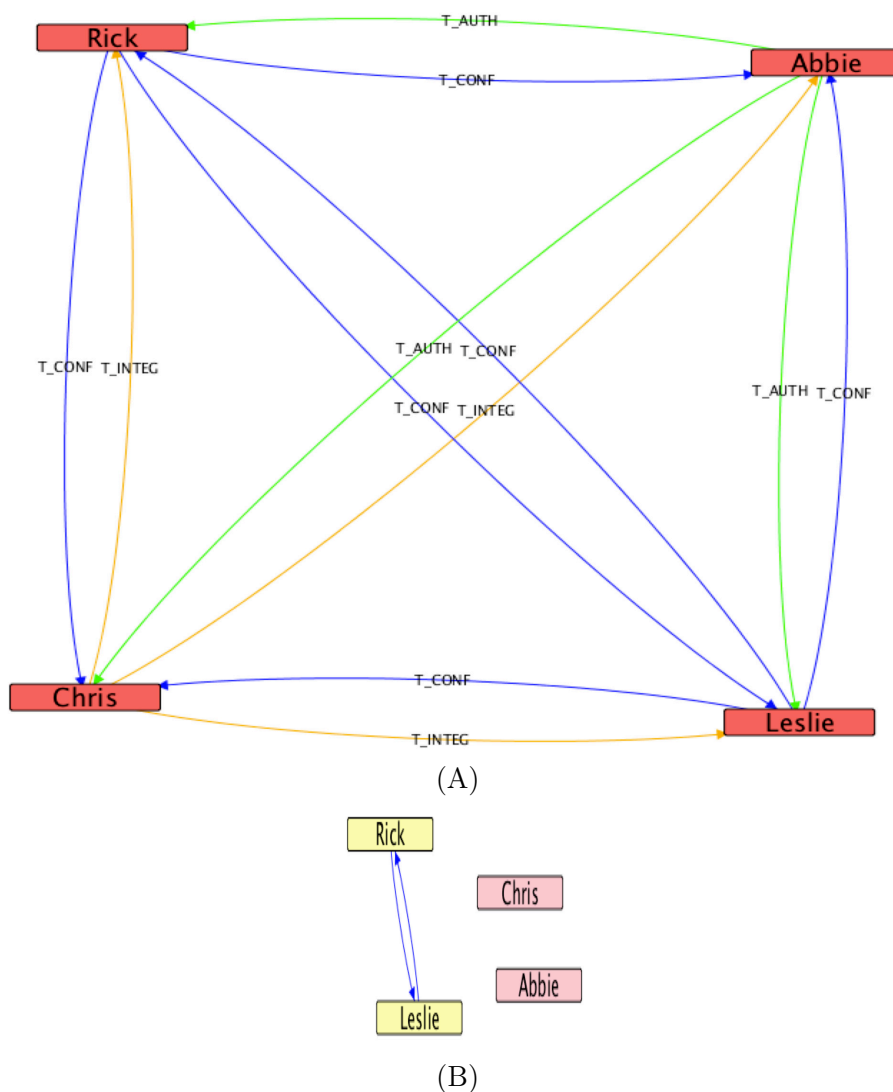


Figure 3.24: Transmissions and subjects' node types of docA before (A) and after (B) the switch between Hypermatrices HA1 and HA2.

3.2.8 Reporting

To implement our reporting tools, we have used simple Java methods. Indeed, because the objects we manipulate are in Java, we can compute additional information thanks to the different fields of these objects. For instance, number of subjects or repartition (i.e. how many node types of each type) can be retrieved by comparing a TCL or an Hypermatrix attributes. Details on the comparisons that can be performed are presented below.

TCL comparison : As an example, let us imagine that the previous companies A and B share a similar document (docX). Thus, both companies have two TCLs of docX, and information of what can be performed on this docX are presented in Hypermatrices HA1 and HB. Differences between the two docX's TCLs can be graphically depicted to provide a way for the experts or administrators to spot differences and similarities (**Figure 3.25**). In this example, both TCLs are different in terms of subjects, total number of subjects, transmission types and node types. However, big TCL can be difficult to represent, thus, reports can also be depicted to compare TCLs. An example of such report is presented in **Figure 3.26**. As one can see, the report gives information on the TCLs density (e.g. number of subjects, transmissions). Moreover, our model is able to spot differences and similarities between TCLs and "odd" behavior (for instance, it has been able to detect that some transmissions were authorized without confidentiality property, when others need to be encrypted). We hypothesize that once the report is analyzed by an expert, she/he can decide to modify the Mapping Rules in order to take into account the report's warnings. In our case, an administrator can modify the rules to set confidentiality on every transmission.

We underline that currently, our implementation only gives general information. However, as we have defined this TCL comparison mechanism as a framework, Thus, it can be tweaked and adapted (by adding for instance color code and icons instead of textual information).

Hypermatrices comparison : Comparison between Hypermatrices can also be performed to underline similarities and differences between the policies of two infrastructures. As an example, we have generated a report to spot differences between HA1 and HB. As stated previously, HA1 represents the normal policy of company A while HB represents the normal Hypermatrix of another company B. The generated report is depicted in **Figure 3.27**. This report gives information on similarities and differences between the two Hypermatrices. As one can see, our mechanism has been able to detect that HA1 had some possible security issues concerning the transmission and node types. These information can help security experts and administrators to detect and correct security problems. Moreover, this mechanism can also help them to have a better understanding of their companies and collaborators security policies.

3.3 Conclusion

In this chapter, we have first presented our Access Control meta-model. As stated previously, this model aims at representing traditional AC control policies that can be found within companies. To do so, we have used the traditional representation of AC rules (i.e. subjects, actions and resources) and we have removed the notion of permissions in order to reduce the total number of generated rules. Thanks to the

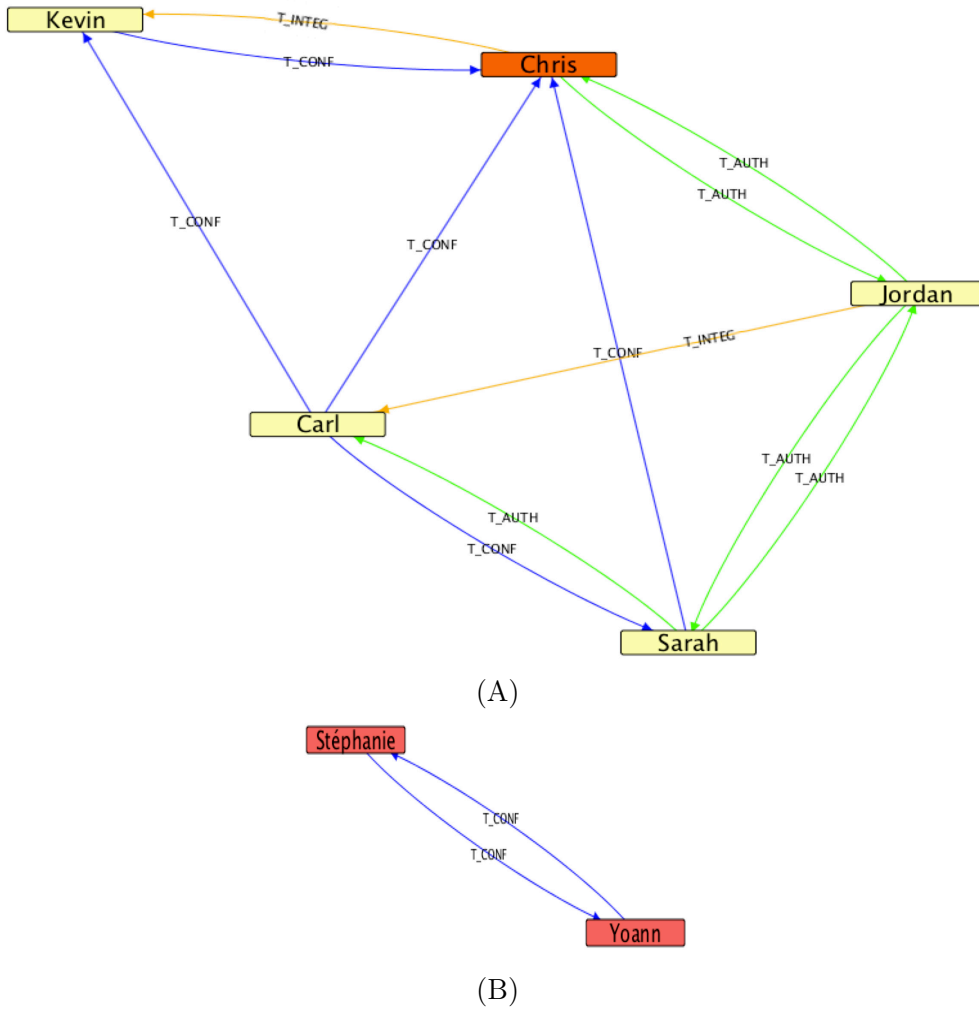


Figure 3.25: Comparison of transmissions and subjects' node types of docX in two different infrastructures. The first graph (A) represents transmissions and marked subjects within the first infrastructure while (B) represents the transmissions and marked subjects within the second infrastructure.

notions embedded in the meta-model, we allow the representation of many concepts (role, type of actions, meta-data inherent to resources) to easily represent AC control such as ACL, RBAC or ABAC. However, our model suffers from some limitations, such as the fact that no temporal and physical context can be represented. Moreover, many rules can be generated, due to cartesian products. To tackle this problem, we have decided to represent only the authorized access. Finally, we have considered that the existing AC rules are not contradictory. However, future works will aim at removing these limitations by enhancing our model.

The second part of the chapter has presented our TC formalism (i.e. TCL), which allows us to describe *"who can send to whom"*. Based on these TCLs, we have

```
===== TCL Report =====

Comparing docX.pdf...

TCL1 location : ./resource/generated_TCL_SiteA/3f9e0b28.ser
and
TCL2 location : ./resource/generated_TCL_SiteB/3fefe582.ser

///// General Information /////
Total amount of marked subjects (TCL1/TCL2): (5/2)
Total amount of transmissions (TCL1/TCL2) : 12/2
Total amount of transmission types (TCL1/TCL2) : 3/1
Total amount of node types (TCL1/TCL2) : 2 / 1

///// TCL in detail - Node types /////
Node types details for TCL1:
Type 1: normal (80%) => Jordan, Sarah, Carl, Kevin
Type 2: all_to_few (20%) => Chris

Node types details for TCL2:
Type 1: critical (100%) => stephanie, yoann

///// TCL in detail - Transmission types /////
Transmission types details for TCL1:
Type 1: T_AUTH 5/12 (41,6%)
Type 2: T_CONF 5/12 (41,6%)
Type 3: T_INTEG 2/12 (16,6%)

Transmission types details for TCL2:
Type 1: T_CONF 2/2 (100%)

/// Conclusion ///
WARNING: TCL1 has both confidential and non-confidential transmissions, is it normal?
WARNING: Chris has not the same node type than the other, is it normal?

TCL1 and TCL2 are different in terms of:
- node types
- node types repartition
- amount of subjects
- amount of transmissions
- transmission types repartition
```

Figure 3.26: An example of the generated report that compares 2 TCLs. These TCLs represent docX transmissions in two different infrastructures.

presented several mechanisms to ease the management tasks, such as clustering, switching and reporting mechanisms. Finally, the last part of the chapter gives implementation details regarding our proof of concept.

```

===== Hypermatrix Report =====

Comparing
HM1 location : ./resource/hypermatrices_SiteA/HM1B.ser
and
HM2 location : ./resource/hypermatrices_SiteB/HM2.ser

===== General Information =====
Total amount of subjects (HM1/HM2): 25 / 50
Total amount of resources (HM1/HM2): 500 / 1000

Total amount of subjects clusters (HM1/HM2): 22 / 36
Total amount of resources clusters (HM1/HM2): 285 / 582

===== Node types repartition (HM1/ HM2) =====
NodeType isolated => 120 / 1169
NodeType single blackhole => 0 / 0
NodeType full blackhole => 98 / 0
NodeType single transmitter => 62 / 0
NodeType normal => 6 / 106
NodeType few to all => 0 / 0
NodeType all to few => 0 / 0
NodeType full_transmitter => 62 / 0
NodeType critical => 788 / 22

===== Transmission types repartition (HM1 / HM2) =====
TransType T_AUTH => 5207 / 0
TransType T_CONF => 3263 / 700
TransType T_DEN => 1064 / 9032
TransType T_INTEG => 0 / 0

///// Conclusion /////
WARNING: HM1 has a majority of critical nodes (69%)
WARNING: HM1 has a majority of T_AUTH (55%)

```

Figure 3.27: An example of the generated report that compares 2 Hypermatrices. These Hypermatrices represent all access and transmissions that can be performed on resources within two different infrastructures.

Evaluation

Contents

4.1	Validation of the hypotheses with a survey on IT professionals	82
4.1.1	Existing surveys	82
4.1.2	Online survey details	83
4.1.3	Interpretation and limitations of the results	88
4.2	Tests on stochastically generated AC policies	90
4.2.1	Efficiency of the inference mechanisms	92
4.2.2	Time-consumption of the mechanisms	96
4.2.3	Conclusion	100
4.3	Tests on real AC	100
4.3.1	Startup company	100
4.3.2	Engineering school policies	101
4.3.3	Conclusion	105
4.4	Conclusion	106

In this chapter, we present the 3 types of tests we have performed in order to validate our model. First, we present a survey we have conducted among the persons who have in charge the definition and the maintenance of the security policies within their company. This survey has been proposed to validate the hypotheses we have considered in chapter 1. As a reminder these hypotheses are:

- **Hypothesis H1** : Companies use various types of AC models. Lack of genericity can force a company to change and / or redefine its existing model in order to use a solution against data leakage.
- **Hypothesis H2** : Incoherences between AC and TC policies can be tiresome to manage and lead to data leakage.
- **Hypothesis H3** : Managing several paradigms and many entities (e.g. users, resources) can be complex, leading to policy weaknesses, errors and data leakage.
- **Hypothesis H4** : Modification of policies need to be performed easily and efficiently, especially in case of emergencies. Indeed, poor reaction time and lack of efficiency can worsen the security crisis, leading to policy weaknesses and data leakage.

- **Hypothesis H5** : Difference of security policies between companies or infrastructures and lack of knowledge can be a problem for data exchanges and business opportunities, causing *de facto* security indulgences and possible data leakage.
- **Hypothesis H6** : Time-efficiency concerns must be taken into account to fit a company's updates frequency.

Secondly, we present the empirical tests we have performed on stochastically generated AC policies to validate that our model is able to perform well in terms of efficiency and time consumption.

Finally, we present the tests we have performed on real AC policies. These tests have been conducted in order to validate that our model can be used in a real environment. The following sections detail these evaluations.

4.1 Validation of the hypotheses with a survey on IT professionals

In this section, we present the survey we have conducted on security experts and administrators. The goal of this survey is on one hand to gather information on participants' positions, policies density and commonly used AC models and on the other hand, to determine the perception of the participants regarding security policies.

This section is organized as follows: first, we present the existing surveys targeting IT professionals. Then, we give information on the survey we have conducted. Finally, we present the results we have obtained.

4.1.1 Existing surveys

Several surveys have been conducted with IT professionals. For instance, [CryptzoneSurvey 2015] has targeted the usage of network AC technologies and best practices. In [SANSSurvey 2014], security experts have been solicited to have insights on end-users security behaviour. Bring Your Own Device (BYOD)¹ has also be targeted by this kind of survey [Johnson & Filkins 2012]. In [Bauer *et al.* 2009], a survey has been conducted among thirteen administrators to highlight the challenges of managing AC policies. Despite its thoroughness, this survey does not perfectly underline the administrators' feelings toward these mechanisms and thus cannot validate our working hypotheses. For these reasons, we have proposed an online survey. This survey is presented in the next section.

¹BYOD refers to the policy of allowing employees of an company to use they own Smartphones or computers for work purpose.

4.1.2 Online survey details

As stated previously, our survey aims at validating our working hypotheses. To do so, we have used a Google form composed of 20 questions. This form is available on the Internet in two different languages (english² and french³). Moreover, a summary of the questions are available in Appendix A. Both versions of the survey have been proposed through social medias (Twitter, LinkedIn, security forums and personal contacts lists). After 5 months, we have been able to gather 50 participant's answers.

Positions of the participants : For this first question, we have asked participants their position inside their company. Results in **Figure 4.1** show that most of them are system administrators/engineers (72%). Security experts represent 26% while only 2% of them have answered that they are network administrators/engineers. These results show that the task of defining and managing Access Control policies is mainly done by employees that are not necessary security experts.

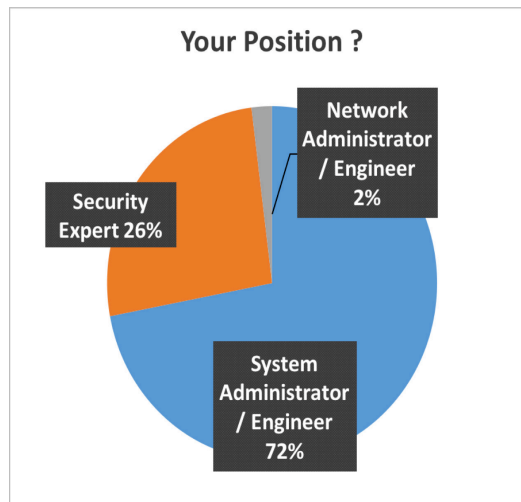


Figure 4.1: Positions of the participants.

Existing Access Control mechanisms : The second question we have asked is about the existence of AC within participants companies. All of them have answered that they are using such mechanisms (see **Figure 4.2(A)**). Concerning the model that are used, results depicted in **Figure 4.2.(B)** show that the most used are traditional ACL and RBAC (resp. 44% and 22%). However, mixing ACL and RBAC is also quite often used (resp. 26%). Other models (for instance ABAC) or other combinations are anecdotal (strictly less than 5%). Finally, some participants

²https://docs.google.com/forms/d/1ZSwt-r37X5ehh0T3IFEJWC7IcWA7dcckHg1LTJEPWHs/viewform?usp=send_form

³https://docs.google.com/forms/d/e/1FAIpQLScWrikdmDVdKyGJrsnfr-PixjWSHG2tXvcAyhAGjuSxNuvCZw/viewform?usp=send_form

have mentioned thanks to a text-area field that they are using ACL and RBAC solutions such as LDAP (i.e. Active Directory⁴ ⁵) or self-made mechanisms (e.g. Apache Shiro⁶, Apache Fortress⁷).

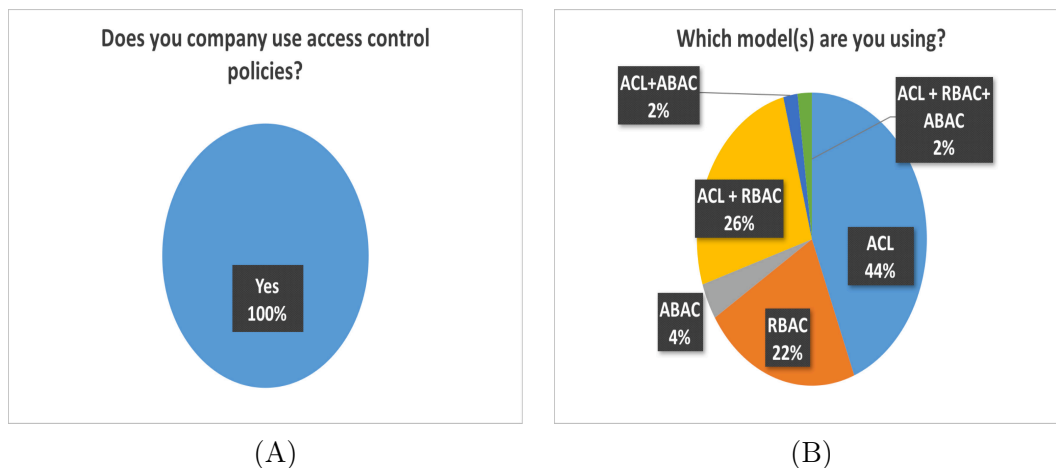


Figure 4.2: Usage of Access Control models (A) and the main models used by the participants (B).

Usage and coherence of Transmission Control policies : Questions about the usage of Transmission Control mechanisms have been asked. Results in **Figure 4.3(A)** show that 48% of participants use some kind of Transmission Control policies (for instance network configuration, Data Leak Prevention). Moreover, these TC policies are often defined based on AC policies in order to have coherence between the two paradigms. Indeed, **Figure 4.3(B)** shows that 71% of participants have answered that they are defining TC based on AC. Finally, **Figure 4.3(C)** shows that people who have declared that AC and TC are kept coherent with each other have stated that having to keep this coherence is an annoying and hard task.

Then, we have asked participants if they would be interested in a mechanism that defines TC policies based on existing AC. **Figure 4.4(A)** shows that this feature seems interesting for many of them. Finally, we have asked them if they would be interested in a mechanism that maintain coherence between AC and TC (i.e. when one is modified, the other is automatically adapted to keep the coherence

⁴Active Directory is the Microsoft implementation of the Lightweight Directory Access Protocol and OpenLDAP : <https://msdn.microsoft.com/en-us/library/bb742424.aspx>

⁵iOpenLDAP is an open source implementation of the Lightweight Directory Access Protocol : <http://www.openldap.org/>

⁶Apache Shiro is a Java security framework that performs authentication, authorization, cryptography, and session management : <https://shiro.apache.org/authorization-features.html>

⁷Apache Fortress is a standards-based access management system, that provides role-based access control : <https://directory.apache.org/fortress/>

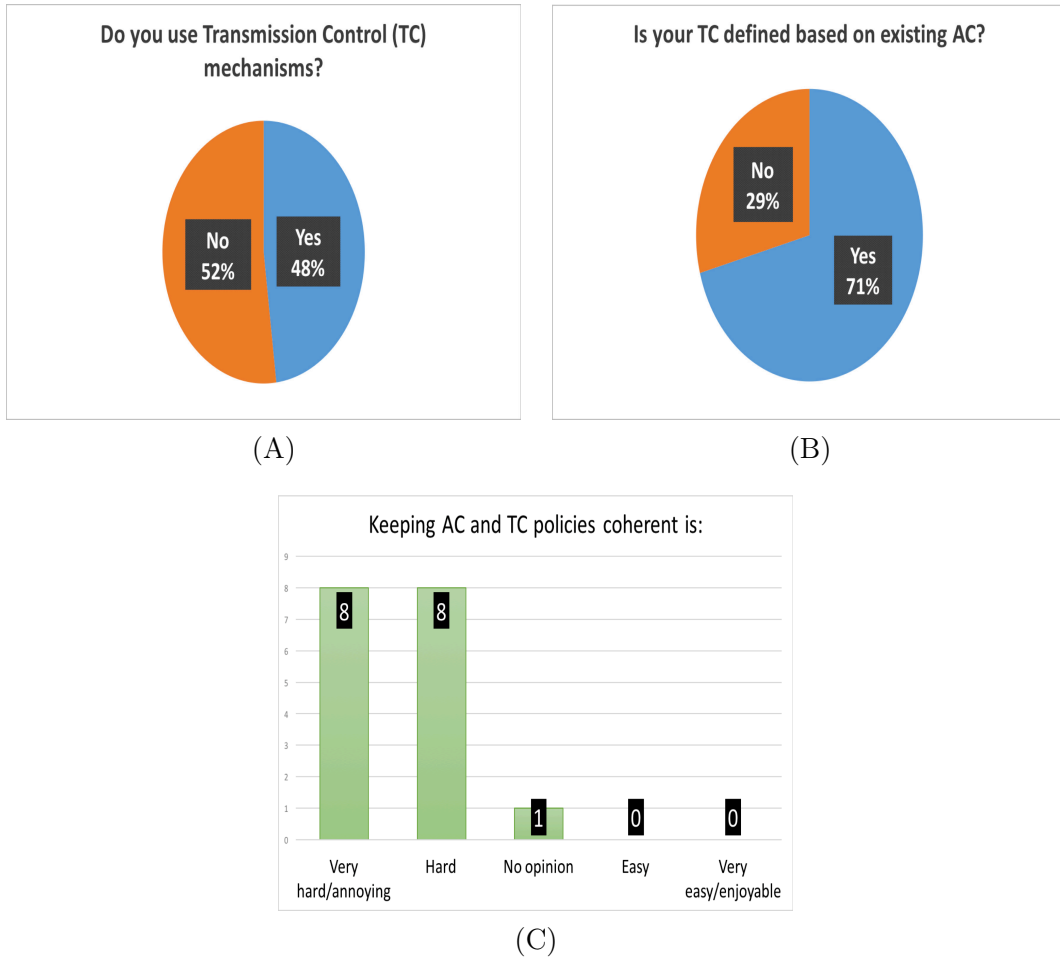


Figure 4.3: Results concerning the usage of Transmission Control policies and link between AC and TC.

between the two paradigms). Results in **Figure 4.4(B)** show that this feature seems interesting for most participants.

Access Control density : Concerning the size of the AC policies (i.e. number of resources and subjects managed by these policies), results in **Figure 4.5(A)** show that most AC policies embed up to 250 users (74%). However, bigger companies or infrastructures (between 250 and 5.000 users) are also represented (24%). Concerning the number of resources, **Figure 4.5(B)** shows that most participants manage AC policies that embed few thousands resources (56% for 5.000 to 10.000 resources). However, smaller/bigger sets are also present (16% for 1.000 to 5.000 resources and 18% for 10.000 to 50.000 resources). Finally, results show that very big resources sets (more than 1 million) are anecdotal (2%). In conclusion, we can say that most participants are managing AC policies for middle-sized companies or bigger infrastructures (such as very big companies) that have decided to divide their

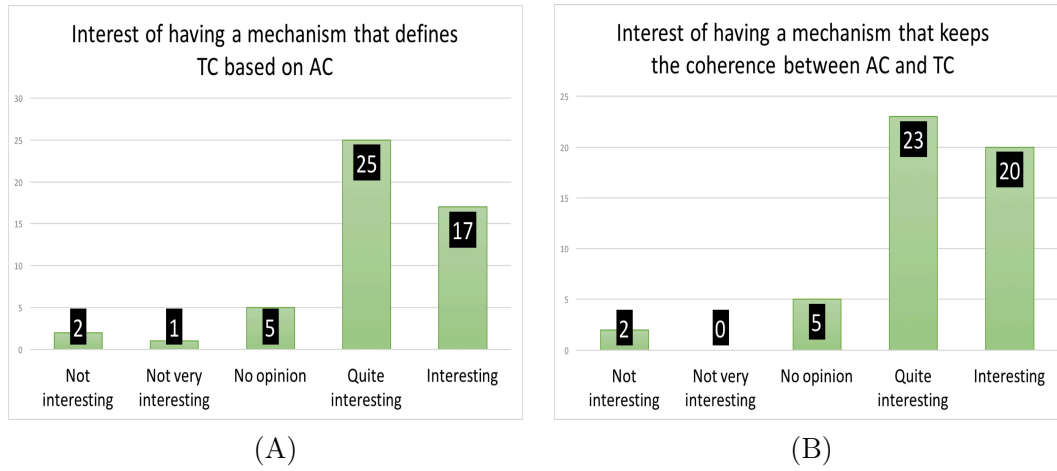


Figure 4.4: Interest of having 2 mechanisms. The first one generates TC based on AC. The second one keeps both AC and TC policies coherent with each other.

security policies in order to be easily manageable.

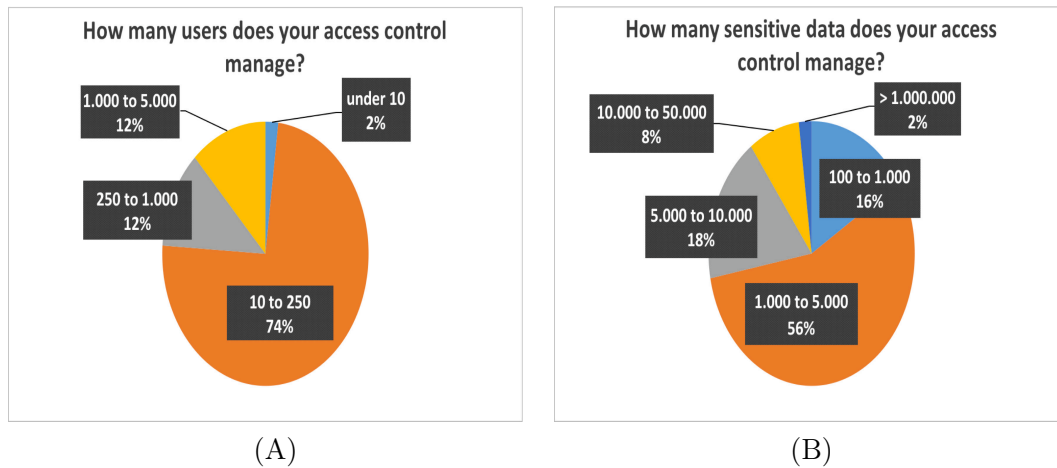


Figure 4.5: Size of Access Control models in terms of managed users (A) and resources (B).

Perception of AC and TC : In order to have insights on participants perception regarding AC and TC, we have asked participants several questions. Firstly, we have asked if the task of defining AC was tiresome or easy. Results in **Figure 4.6(A)** show that this task is quite tiresome for most participants. Then, we have asked them if their policies are managing too many entities. Results in **Figure 4.6(B)** also show that it is quite the case. Thus, we have finally asked participants if they would be interested in a feature that can reduce the total number of managed entities. **Figure 4.6(C)** shows that this feature is interesting for most participants.

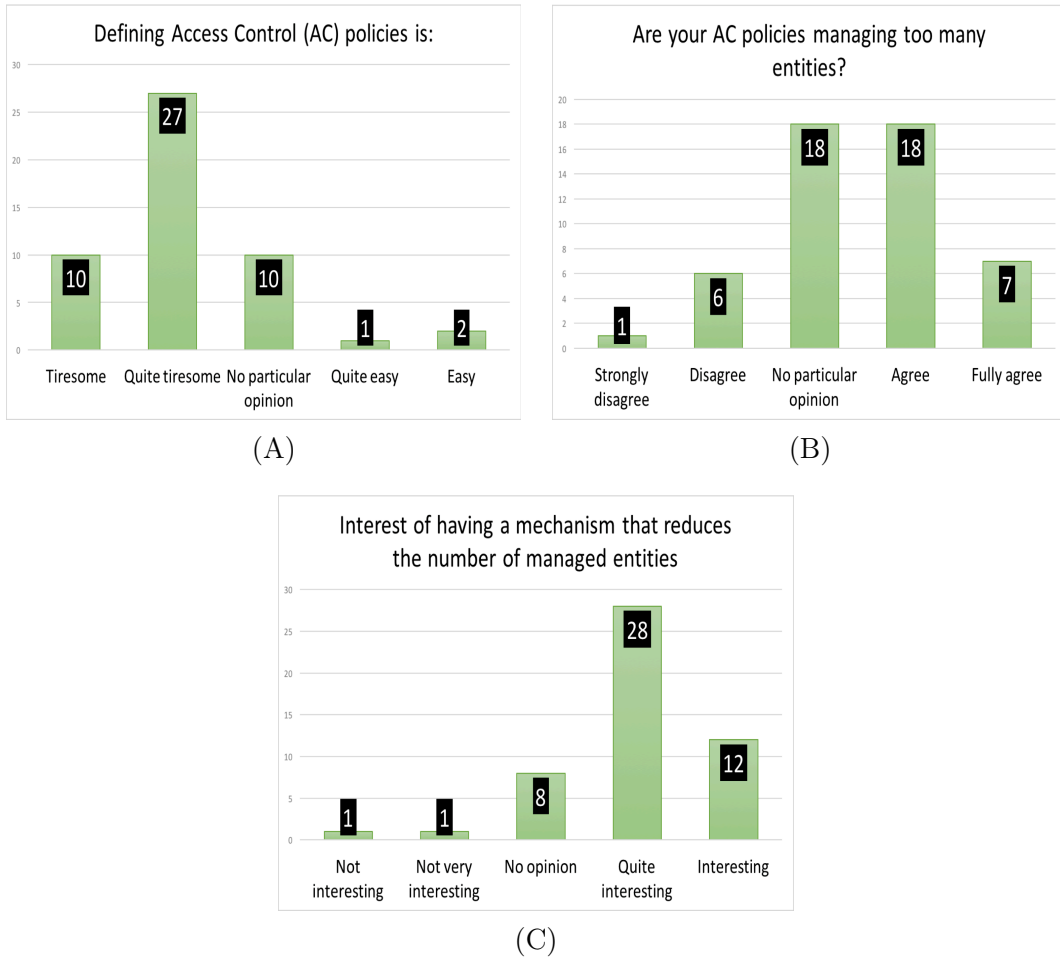


Figure 4.6: Feeling towards AC definition (A) and having to keep a coherence between AC and TC (B), and interest of having a mechanism that reduces the total number of managed entities.

Emergency management : We have asked participants how they manage their policies in case of emergencies (e.g. attacks, intrusions). **Figure 4.7(A)** shows that 60% of participants have answered that policies are somehow modified in case of an emergency. Moreover, we have asked them if they would be interested in a solution that could deploy different policies, especially in case of emergencies. Results in **Figure 4.7(B)** show that such solution is quite interesting for them.

Exchange with other infrastructures and companies : We have asked participants if their company is involved in data exchange with other companies or infrastructures (other internal departments or other companies). Results in **Figure 4.8(A)** show that for the majority of them (92%), their company is involved in this type of data exchange. Furthermore, we have asked them to determine if differences between two infrastructures policies can be annoying for data exchange. To give an

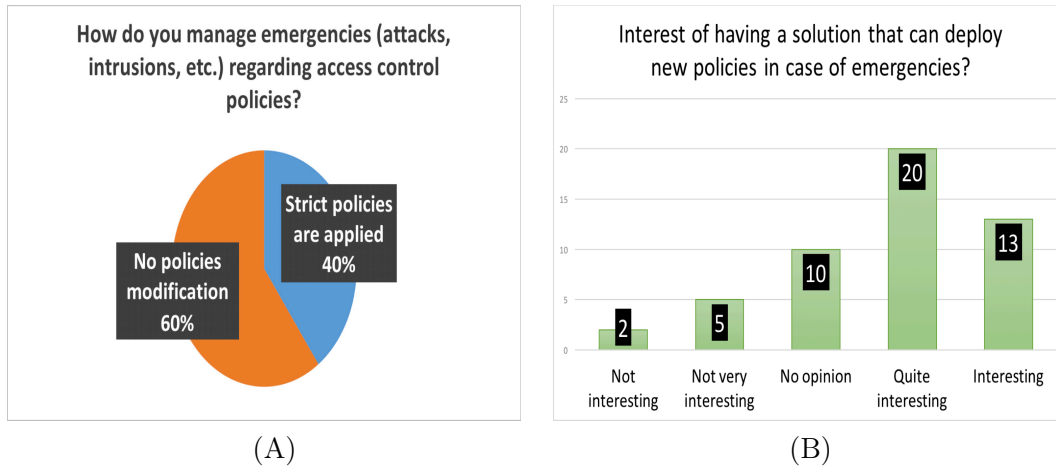


Figure 4.7: Questions regarding the emergency management (A) and interest of having a solution that can deploy new policies in case of emergencies (B).

example, we have stated the case of having to define new policies to match the other company's security prerequisites. Results in **Figure 4.8(B)** show heterogeneous results. However, these results show that security differences between infrastructures do not facilitate data exchanges.

Finally, we have asked participants if they will be interested in a feature that could spot differences between security policies and thus enhance the overall knowledge and ease the data exchange between companies. **Figure 4.9(C)** shows that this feature is interesting for participants.

Updates frequency : We have asked a question regarding the frequency of AC updates. For this question, results are quite heterogeneous (see **Figure 4.9**). Nevertheless, we can conclude that updating AC is an operation that needs to be performed at least several times a day.

Now that the questions and the results of the survey have been presented, we describe in the next subsection our interpretation.

4.1.3 Interpretation and limitations of the results

We have proposed a survey to gain insights on the people who have in charge the definition and the maintenance of security policies within companies. Thanks to this survey, we have been able to determine the general size, density and most commonly used Access Control models. The results collected concerning AC models show that participants use mainly common and well-known AC models (ACL, RBAC). This can be explained by the fact that commonly used Operating Systems, such as GNU/Linux, Mac OSX and Windows embed such models. Moreover, other mechanisms, such as LDAP protocol, have also been cited by participants. Thus,

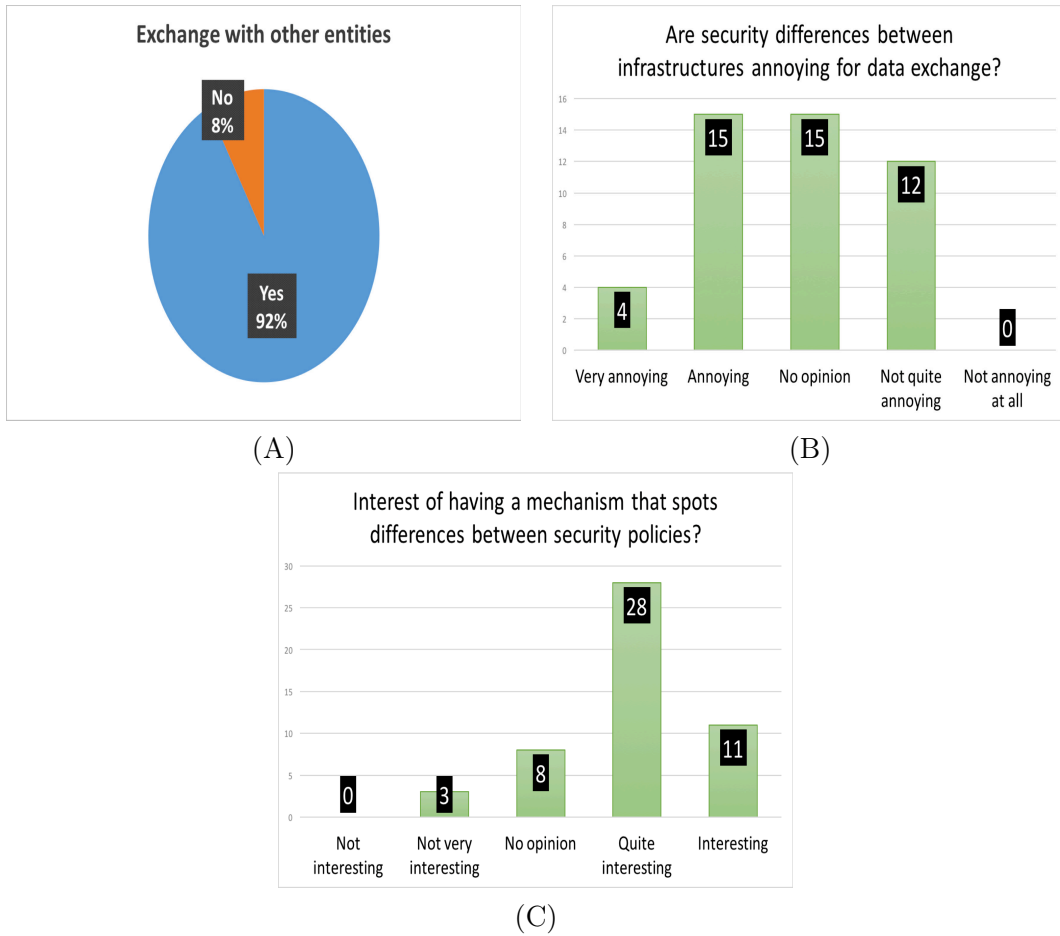


Figure 4.8: Questions about data exchange between infrastructures and companies (A) and feelings regarding such practice (B). Finally, (C) depicts the interest of having a mechanism able to spot differences in the infrastructures policies.

we validate our first hypothesis **H1**.

Concerning the density, results show that most participants are managing few thousands resources and up to 250 subjects. These results can mean two things: either participants work in middle-sized infrastructures or bigger companies that partition their AC policies in "*manageable*" subsets of policies. Whatever the reason, the survey has underlined that managing such policies is not an easy task. This can be explained by the fact that, even if the density seems reasonable, having to manage such amount of resources and subjects can be complicated, especially if this is not the only task of the participants. Indeed, as most of them are administrators, we can hypothesize that having to define AC and TC policies is not their only task within the company. Thus, we validate our second hypothesis **H3**.

The results also show that even if Transmission Control is less common than Access

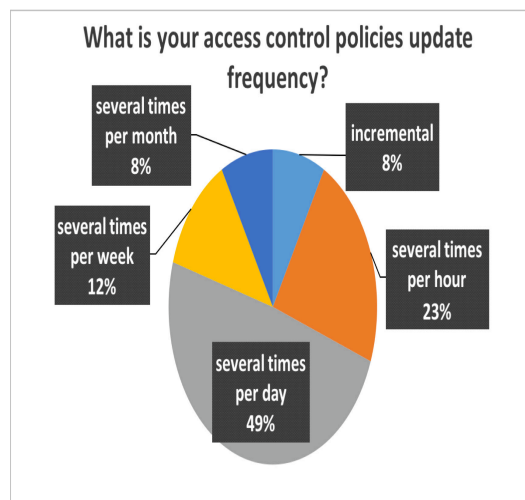


Figure 4.9: Update frequency of the AC policies.

Control, such mechanisms are nonetheless present within some companies in one form or another. Furthermore, the survey has underlined that coherence between both paradigms can be difficult to manage, validating *de facto* hypothesis **H2**.

Then, results have shown that modification of the policies in case of emergencies is quite common. These results validate our hypothesis **H4**. Moreover, results have also shown that most participants share information with other infrastructures and companies and think that lack of knowledge regarding the policies of these infrastructures can be a problem for security. Thus, we consider our hypotheses **H5** as validated.

Finally, information on update frequency and feelings toward emergency management and exchange with other companies or infrastructures have been highlighted, validating our hypothesis **H6**. **Table 4.1** summarize the aforementioned results.

However, we underline that even if these results are interesting, they can be criticize in several ways. First of all, one can consider that the survey targets "only" 50 participants. For these reasons, the results can lack representativity. Moreover, lack of vocabulary and comprehension can distort the results. Thus, we consider that face to face interview sessions could be performed in order to reduce these hypothetical biases. Such interviews are discuss in section 5.2.2.

4.2 Tests on stochastically generated AC policies

In this section, we aim at testing our model on stochastically generated ACL. The objectives of these tests are:

- To measure the efficiency of the subject and resource similarity mechanisms (i.e. inference mechanisms).

Hypotheses	Related questions	Validations & information
H1	Q2, Q3, Q7	Several AC models and mechanisms are used within companies Defining these AC policies is not an easy task Our proposal seems interesting
H2	Q12, Q13 Q14, Q15	TC defines based on AC Maintaining coherence is not an easy task Our proposal seems interesting
H3	Q4, Q5, Q8, Q9	various volumetry It is not easy to manage many entities Our proposal seems interesting
H4	Q10, Q11	Several participants modify their policies during emergencies Our proposal seems interesting.
H5	Q17, Q18, Q19	Almost all participants share information with other companies Lack of knowledge seems to be a problem Our proposal seems interesting
H6	Q6	Participants have various update frequencies for their policies
Other	Q1	Participants' positions

Table 4.1: Summary of the survey's results.

- To measure the time-consumption of the generation mechanism, coherence mechanism, inference mechanisms, Hypermatrices switch and reporting mechanism.

Tests conditions : To perform these tests, we have used a MacBook Pro Retina (Intel Core i7, 2,4 GHz, 16GB RAM, 256 GB SSD hard drive) and Java 7. Artificial ACLs have been generated following the answers of the survey. Thus, we have generated ACLs that embed up to 250 subjects and 7500 resources.

Concerning the actions and subjects, we have considered the Read, Write and Delete actions and 3 parameters per subject (name, city of the company, and job position). Information about these ACLs are given in **Table 4.2**. Finally, all tests have been conducted 5 times. For every set of tests, minimal and maximum values have been removed and mean values have been computed based on the 3 remaining results.

Generation mechanisms : To generate an ACL, our mechanism works as follows. First, we generate 3 sets (one for subjects, one for actions and one for resources) that we filled with fake entities. Then, we randomly pick one element on each set to create a AC rule (thus, identical rules are permitted). In order to reduce the randomness and make the policy more "realistic", created AC rules are sometime copied and slightly modified (for instance, the rule "Vincent, Read, DocA" is reuse and transformed into "Rose, Read, DocA" to simulate a context where several users can access the same resources).

TCL matrices filling : In order to fill the TCL matrices with transmissions, we did not use Mapping Rules because Mapping Rules can greatly modify the results of resource and subject similarity mechanisms, inducing *de facto* biais in the results. Indeed, if we take the Mapping Rule "*never allow transmission*", every single matrix will be filled with a denied transmission type (TRANSMISSION_DEN). If so, the

ID	Rules	Subjects	Resources
ACL_1	1000	25	300
ACL_2	1000	25	500
ACL_3	1000	100	900
ACL_4	1000	200	900
ACL_5	5000	20	1000
ACL_6	5000	20	2000
ACL_7	5000	20	3000
ACL_8	5000	100	1000
ACL_9	5000	150	3000
ACL_10	5000	250	1000
ACL_11	5000	250	3500
ACL_12	7500	10	5000
ACL_13	7500	20	1000
ACL_14	7500	20	2000
ACL_15	7500	20	5000
ACL_16	7500	200	5000
ACL_17	7500	250	3500
ACL_18	10000	20	2000
ACL_19	10000	50	2000
ACL_20	10000	50	6000
ACL_21	10000	50	7500
ACL_22	10000	200	2000
ACL_23	10000	200	5000
ACL_24	10000	200	7500
ACL_25	10000	250	3500

Table 4.2: Generated ACLs that have been used for the tests.

odds to have TCLs matrices that are similar to each other will be greater and the results will be very good. On the contrary, numerous and very specific Mapping Rules tends to reduce the odds to have similarities between subjects and resources, reducing the overall performances of the subjects and resources similarity mechanisms. To be as independent as possible, we have decided to fill the matrices with a stochastic method. Thus, each intersection of each matrix is randomly filled with TRANSMISSION_AUTH, TRANSMISSION_DEN or TRANSMISSION_CONF transmission types. By doing so, we aim at measuring the efficiency of our mechanisms in the worst case scenarios. Indeed, because the rules are stochastically generated, there is no previous groups of subjects and resources (as it can be the case by using real AC policies). Moreover, stochastic filling reduces the odds to have similar matrices and subjects.

4.2.1 Efficiency of the inference mechanisms

We have decided to conduct some tests on the resource and subject similarity mechanisms to determine their efficiency. The next subsections describe these tests in details.

Resource similarities : As stated previously, this process aims at tackling the *density problem* by creating clusters with similar resources (i.e. resources that can be accessed and retransmit by the same subjects, in the exact same way). We have

taken the ACLs presented in **Table 4.2** to test the resource similarity mechanism in terms of efficiency. To measure efficiency, we have introduced the notion of gain. The gain expresses the percentage of reduction of the total number of TCLs. Hypothetically speaking, two extremums can be underlined. In the worst case scenario, every single resource is so different that none of them is accessed in the exact same way by the exact same subjects. In that case, every resource will be in an isolated cluster (i.e. singleton) and every cluster will have a different TCL (there will be n TCLs for n resources).

The opposite extremum is where all resources are the same (i.e. they have identical TCLs) and thus, only one cluster will be generated and will be filled with all resources. In the first extremum, the gain will be of 0%, while the second extremum will show a gain close to 100%.

Concerning our tests, the fifth column of **Table 4.3** shows that this mechanism can reduce the total number of TCLs up to 77%. We underline that such results is interesting, especially in the case of big ACLs that contain thousands of resources (and thus, thousands of TCLs before applying the resource similarity mechanism).

Results show variations of gain that can be explained in several ways. First of all, ACLs have been generated stochastically, inducing differences of gains. Indeed, randomness can cause subjects and resources to have more or less similarities, increasing *de facto* the gain. Moreover, variations can be explained by another phenomenon that we have called the "*entropic phenomenon*". This phenomenon can be explained by the fact that our model represents a TCL as a list of subjects who can access and retransmit one or more resources. Thus, the more subjects in the list, the bigger the TCL. Two TCLs are more likely to be identical if their size (i.e. their list of subjects) are small. In other words, if for the same number of subjects and rules, an ACL contains many resources, the gain has better chances to be higher, because the rules will target more resources, and odds to have shorter, and thus, identical TCLs, will be increased. **Table 4.4** shows interesting results concerning this phenomenon. Indeed, this table represents ACLs that have the same number of rules, but different numbers of subjects and resources. In this case, gains for both resources and subjects are increasing. This can be explained by the fact that with more entities for the same number of rules, the odds to have subjects with many accesses (and thus, resources that can be accessed by a lot of subjects) are reduced. Less accesses means that the generated TCLs will be smaller and less complex, inducing that the odds to have similarities will be increased.

On the contrary, when an ACL has a small number of subjects, a small number of resources and a much bigger number of rules, the odds for a subject to have a lot of different access rights are higher, because rules will target more resources. Thus, subject's capabilities list will be bigger and the odds to have similarities between subjects and resources will be smaller. Such results can be seen in **Table 4.5**, where ACLs with similar subjects and resources, but different amount of rules are presented. As one can notice, the resources and subjects gain is dropping

ID	Rules	Subjects	Resources	Resources Gain	Subjects Gain
ACL 1	1000	25	300	45%	19%
ACL 2	1000	25	500	48%	24%
ACL 3	1000	100	900	63%	17%
ACL 4	1000	200	900	54%	20%
ACL 5	5000	20	1000	20%	28%
ACL 6	5000	20	2000	61%	35%
ACL 7	5000	20	3000	26%	43%
ACL 8	5000	100	1000	53%	30%
ACL 9	5000	150	3000	60%	33%
ACL 10	5000	250	1000	4%	9%
ACL 11	5000	250	3500	55%	7%
ACL 12	7500	10	5000	77%	30%
ACL 13	7500	20	1000	11%	10%
ACL 14	7500	20	2000	25%	24%
ACL 15	7500	20	5000	70%	31%
ACL 16	7500	200	5000	61%	28%
ACL 17	7500	250	3500	35%	9%
ACL 18	10000	20	2000	16%	5%
ACL 19	10000	50	2000	21%	10%
ACL 20	10000	50	6000	49%	15%
ACL 21	10000	50	7500	76%	17%
ACL 22	10000	200	2000	11%	10%
ACL 23	10000	200	5000	42%	15%
ACL 24	10000	200	7500	55%	10%
ACL 25	10000	250	3500	19%	5%

Table 4.3: Summary of the results obtained for the validation of the inference mechanisms.

when the number of rules increases. Thus, what is also important for the *entropic phenomenon* is the difference (i.e. ratio) between the total number of resources and the total number of rules. Indeed, if the ratio is small, the odds to have a big gain will be higher. We underline however that other factors, such as the total number of actions and the applied Mapping Rules can also affect the gain results.

Previous results have shown that the gain of resource similarity mechanism can vary depending on the ratio between rules, subjects and resources. However, we consider that the overall gain is quite satisfactory, especially if you consider that the TCLs have been stochastically filled. Now that we have presented the results for the resource similarity mechanism, we focus on the results of the subject similarity mechanism.

Subject Similarities : As stated previously, the subject similarity mechanism aims at creating clusters with similar subjects (i.e. subjects that can access and exchange resources in the exact same way) in order to reduce the overall density of the policies.

The results of the conducted tests have shown that efficiency values are quite variable. Indeed, worst results show a gain of 5% while best results reduce the

ID	Rules	Subjects	Resources	Resources Gain	Subjects Gain
ACL 13	7500	20	1000	11%	10%
ACL 14	7500	20	2000	25%	24%
ACL 15	7500	20	5000	70%	31%
ACL 19	10000	50	2000	21%	10%
ACL 20	10000	50	6000	49%	15%
ACL 21	10000	50	7500	76%	17%

Table 4.4: Comparisons of ACLs that share the same number of rules and subjects, but different number of resources. As one can notice, the gain of both resource and subject similarities is increasing.

ID	Rules	Subjects	Resources	Resources Gain	Subjects Gain
ACL 5	5000	20	1000	20%	28%
ACL 13	7500	20	1000	11%	10%
ACL 11	5000	250	3500	55%	7%
ACL 25	10000	250	3500	19%	5%
ACL 16	7500	200	5000	61%	28%
ACL 23	10000	200	5000	42%	15%

Table 4.5: Comparisons of ACLs that share similar numbers of subjects and resources, but different number of rules. As one can notice, the gain of both resources and subjects similarities is decreasing.

amount of subjects by 43%. However, we underline that the mean value of the subject similarities gain is around 20%, which is quite satisfactory, especially for infrastructures that contains many subjects.

If we compare subjects and resource similarity mechanisms, we can underline that the subject similarity mechanism has a lower gain than the resource similarity mechanism. This can be explained by the fact that, once again, the number of subjects is smaller, reducing the odds to have similarities. Thanks to our tests, *entropic phenomenon* can also be highlighted. Indeed, **Table 4.4** shows that the subject similarities gain is increasing with ACLs that contains more resources but have the same number of rules and subjects. As stated before, this can be explained by the fact that more resources with the same number of rules will cause subjects to have less capabilities. Because our mechanism compares capabilities set, the odds to have similarities between subjects are higher if the capabilities set are smaller. In addition, **Table 4.5** shows a gain drop when the total number of rules is increasing while the numbers of rules and subjects stay the same. Once again, this can be explained by the fact that more rules with the same number of resources and subjects will cause subjects to have bigger capabilities set, thus reducing the odds to have similarities, and reducing *de facto* the gain. However, we consider that the gain we have obtained is quite satisfactory, because it can reduce the overall number of entities, and thus, reduce the tiresomeness of the policy management.

In this subsection, we have proposed tests to validate the similarity mechanisms. These mechanisms have been implemented to reduce the overall density of the generated policies. To validate these mechanisms, we have measured the total number of subjects and resources before and after the similarity mechanisms in order to compute the gain (i.e. percentage of reduction). Results have shown that these inference mechanisms can reduce the overall complexity by creating clusters of resources and subjects. Despite differences in the results, we consider that the overall gain is quite good, especially because the ACLs we have used have been randomly generated. Based on the results, we consider that the density problem is covered by our resource and subject similarity mechanisms.

Now that the efficiency of our subject and resource similarity mechanisms have been tested, the following tests aim at evaluating our mechanisms in terms of time-efficiency.

4.2.2 Time-consumption of the mechanisms

We have considered in our working hypothesis that a good solution needs to be reactive and time-efficient. Thus, we have measured the time-consumption of our mechanisms to determine if they would fit a company's update frequency. Details on the results we have obtained are presented below.

Generation mechanism : First, we have measured the time consumption of the generation mechanism. **Table 4.6** shows that the generation mechanism takes for most cases less than an hour to compute. Thus, this mechanism can be suitable for most participants update frequencies. For participants who have answered that their update frequencies were faster than several times an hour, we underline that the generation process needs to be done only once. Indeed, once the TCLs are generated, further modifications will be managed by the coherence mechanism, which is much more faster. Moreover, we underline that we have used a three years old laptop computer and that the tests have been performed on a old version of our implementation (i.e. without parallelization).

Coherence mechanisms : In order to test the coherence mechanisms, we have divided the tests in two parts. First, we have validated the coherence mechanism with various administration's actions, such as adding a new rule or modify a subject's set of capabilities. Then, we have validated the time consumption of the algorithms that check coherence principles P1 and P2 (see section 3.1.3).

To evaluate the time consumption of the coherence mechanism, we have implemented different actions that can be performed by a security expert or administrator on both AC and TC paradigms. Then, we have measured the time the mechanisms take to perform the automatic modification in order to keep the coherence between

ID	Rules	Subjects	Resources	Generation Time (in sec)
ACL 1	1000	25	300	25
ACL 2	1000	25	500	25
ACL 3	1000	100	900	5
ACL 4	1000	200	900	5
ACL 5	5000	20	1000	450 (7min)
ACL 6	5000	20	2000	700 (12min)
ACL 7	5000	20	3000	1800 (30min)
ACL 8	5000	100	1000	550 (9min)
ACL 9	5000	150	3000	1300 (22min)
ACL 10	5000	250	1000	600 (10min)
ACL 11	5000	250	3500	1400 (23min)
ACL 12	7500	10	5000	1000 (17min)
ACL 13	7500	20	1000	850 (14min)
ACL 14	7500	20	2000	1200 (20min)
ACL 15	7500	20	5000	1300 (22min)
ACL 16	7500	200	5000	2900 (48min)
ACL 17	7500	250	3500	3600 (60min)
ACL 18	10000	20	2000	1400 (23min)
ACL 19	10000	50	2000	1900 (31min)
ACL 20	10000	50	6000	4200 (70min)
ACL 21	10000	50	7500	4500 (75min)
ACL 22	10000	200	2000	2400 (40min)
ACL 23	10000	200	5000	3500 (58min)
ACL 24	10000	200	7500	4700 (78min)
ACL 25	10000	250	3500	3550 (59min)

Table 4.6: Time-consumption of the generation mechanism.

AC and TC. The results of these tests are depicted in **Table 4.7** and show that this mechanism is quite efficient, even with bigger ACLs. However, we underline that these results can vary depending on the rule that is added. For instance, creating a subject with a big capabilities set will generate many modifications on a lot of TCL matrices, inducing many modifications in the ACL. However, we consider that the results are acceptable for the density and the update frequency highlighted by the participants of our survey.

Subject and resource similarity mechanisms : We have tested our subject and resource similarity mechanisms in terms of time-efficiency. As one can notice in **Table 4.8**, the results are quite variable, even with ACLs sharing similar density. This can be explained by the "*entropic phenomenon*" discussed in section 4.2.1. Indeed, when an ACL has a small amount of subjects, a small amount of resources and a much bigger amount of rules, the odds for a subject to have a lot of different access controls are higher, because rules will target more resources. Thus, subjects capabilities list will be bigger and the resource similarities, which compares resources TCLs, will take more time. Indeed, even if the TCLs are small, two TCLs will be compared based on their marked subjects (i.e. subjects that have access to the resources the TCLs is linked to). In order to do so, the mechanism will have to compare every subjects capabilities to be sure that these subjects can access a resource in the exact same way. This will induce slow resource similarity process.

Action	ACL_3	ACL_7	ACL_11	ACL_15	ACL_20	ACL_24
Create a new subject and add her/him to a specific SC	0,81	1,14	1,69	2,22	1,98	3,19
Create a new subject by giving a set of capabilities	1,09	1,67	2,80	1,97	1,72	2,74
Modify a subject's capability	0,86	0,21	1,85	0,30	1,12	1,94
Move a subject from a subject cluster to another	1,49	3,24	4,12	1,30	3,72	4,36
Delete an existing subject	0,50	0,23	1,05	0,29	3,07	3,92
Create a new resource and add it to a specific RC	0,90	3,18	2,82	2,09	4,10	5,04
Create a new resource by giving a set of capabilities	0,54	2,37	3,79	3,11	4,29	5,36
Delete an existing resource	0,48	3,84	4,77	4,15	5,03	5,67
Add a new ACL rule	1,34	2,20	2,78	2,23	4,06	4,72
Remove an existing ACL rule	1,28	2,12	3,02	2,98	4,29	5,06

Table 4.7: Results of the coherence mechanism in seconds. All results are mean values obtained after 5 experiments. Lower and higher values have been removed in order to compute the mean value with the 3 remaining results.

Moreover, subjects with big sets of capabilities will be more difficult to compare, inducing bigger time consumption for the subject similarity mechanism.

Finally, we underline that the ACLs have been generated stochastically, thus, differences in the results can be caused by randomness.

However, general results show that most of the time, this process takes less than an hour to process in the worst case scenarios. We underline that this process is only done once (because further modification will be managed by the coherence mechanism). Therefore, we conclude that this mechanism is quite efficient for the size and update frequency highlighted by the participants of our survey.

Hypermatrices Switch : Switching an Hypermatrix for another takes few seconds to perform. Indeed, it only consists in loading another Java object. However, the only drawback regarding time consumption is the time it takes to generate the Hypermatrices. This process depends on the total number and complexity of the generated TCLs. In the case where TCL matrices have already been generated, creating the Hypermatrix is a very fast operation that is performed within a few minutes, even with the biggest size and density. Therefore, we consider this process as efficient, especially considering that it has been implemented in order to be reactive in case of emergencies.

Matrices and Hypermatrices comparisons : Just like Hypermatrices switch, comparing two matrices or Hypermatrices is quite fast. In our case, conducted tests have shown that generating a report takes less than **3 minutes** to perform with the

ID	Rules	Subjects	Resources	Resources similarity Time (in sec)	Subjects similarity Time (in sec)
ACL_1	1000	25	300	12	< 1
ACL_2	1000	25	500	7	< 1
ACL_3	1000	100	900	1	< 1
ACL_4	1000	200	900	2	1
ACL_5	5000	20	1000	140	2
ACL_6	5000	20	2000	80	< 1
ACL_7	5000	20	3000	90	5
ACL_8	5000	100	1000	550 (9min)	65
ACL_9	5000	150	3000	600 (10min)	85
ACL_10	5000	250	1000	750 (12min)	10
ACL_11	5000	250	3500	350 (6min)	12
ACL_12	7500	10	5000	750 (12min)	26
ACL_13	7500	20	1000	1010 (17min)	1
ACL_14	7500	20	2000	280 (5min)	5
ACL_15	7500	20	5000	330 (6min)	2
ACL_16	7500	200	5000	270 (4min)	3
ACL_17	7500	250	3500	580 (10min)	2
ACL_18	10000	20	2000	330 (5min)	6
ACL_19	10000	50	2000	710 (12min)	2
ACL_20	10000	50	6000	950 (16min)	5
ACL_21	10000	50	7500	800 (13min)	50
ACL_22	10000	200	2000	3200 (53min)	2
ACL_23	10000	200	5000	2100 (35min)	25
ACL_24	10000	200	7500	3200 (53min)	37
ACL_25	10000	250	3500	4000 (1h)	15

Table 4.8: Results for the subject and resource similarity mechanisms. All results are mean values obtained after 5 experiments. Lower and higher values have been removed in order to compute the mean value with the 3 remaining results.

biggest ACLs. We consider these results as acceptable as they are fast and because we consider that a report generation does not need to be instantaneous.

4.2.3 Conclusion

In this section, conducted tests on stochastically generated AC policies have been presented. To perform these tests, we have created ACLs that embeds up to 250 subjects and 7500 resources in order to fit participants size and density. We have then used these ACLs to perform empirical tests on the mechanisms we have proposed. These tests have been proposed to validate our mechanisms both in terms of efficiency and time-consumption. Results show that our mechanisms are interesting, both in terms of efficiency and time-consumption.

However, we underline that the tests have been realized on stochastically generated ACLs. To overcome this issue, we have decided to test our model on real Access Control policies (i.e. policies that come from real environment). These tests are presented in the next section.

4.3 Tests on real AC

As stated previously, we aim at testing our solution on *"real"* AC policies to check if our model could be used in a real environment. To do so, we have used 2 sets of policies: one from a startup company and one from an engineering school. Details on these policies and the tests that have been performed are given below.

4.3.1 Startup company

A startup company working in Information Technologies (IT) has proposed a snippet of its Access Control rules. These rules control the access of a network repository inside the company. The repository contains documents which are either public (i.e. accessible by all employees), restricted or confidential. The rules have been implemented in Apache Fortress and represent a RBAC policy. In this policy, 3 different groups are used: manager (*"manager"*), developers (*"dev"*) and interns (*"intern"*). Concerning the density, this policy contains 6 subjects and 50 resources.

Transformation of the RBAC rules : We have asked the administrator of the company to transform his policy in our formalism. Technically speaking, the version of the policy is RBAC0. Thus, the notion of users, roles and sessions are implemented. However, for this specific company, users only have one type of session and thus, one role. The administrator has explained these implementation choices by the fact that the infrastructure is very small and that sessions, in his opinion, tends to make the management more complex.

RBAC0 contains the notion of permissions (i.e. what is allowed or not). However,

our model only embeds the rules that explicitly authorize subjects to perform a specific action. For these reasons, the administrator has to keep only the accesses that are authorized (all actions that are not explicitly mentioned in the policy are considered as denied by our model).

Apache Fortress provides mechanisms to check the entities (e.g. users, roles), search for permissions, etc.. Thanks to these mechanisms, the administrator has been able to easily retrieve the required elements and has provided an XML file that has been generated with a script implemented in Java. This XML file is valid in our formalism. Moreover, he has tagged the subjects with 3 different levels of security (1 for *"extern"*, 2 for *"dev"*, 3 for *"manager"*). We have used these levels to create a Mapping Rule stating that *"transmission is authorized only if the receiver has an equal or higher level than the sender"*. Thus, a developer (level 2) can only send a document to another developer or to a manager (level 3).

Tests - Generation mechanism : Due to the small size of the policies (roughly 800 rules), the generation process has taken less than a minute to perform. Independently of the time consumption, this test has shown that our model is able to use RBAC0 model implemented in Apache Fortress. Nevertheless, it requires an automated script to easily fetch and parse the required information (i.e. subjects, resources and actions) in an XML file that respects our formalism.

4.3.2 Engineering school policies

The AC policies used by the engineering school is based on Unix modes. In order to respect the privacy of the students and professors, we have anonymized the policies.

Transformation of the POSIX rules : Unix like Operating Systems uses POSIX permissions to define Access Control. For a specific document, Unix modes defines the rights (i.e. Read, Write and Execution) of 3 different classes of users: owner, group and others. For instance, a document docA.txt that can only be accessed, modified and executed by its owner yoann (member of group staff) will have the following "ls" command output:

```
-rwx----- 1 yoann staff      5 11 août 14:16 docA.txt
```

We have implemented a script that takes every POSIX right and transform it into an Access Control rule in our model. To do so, the script browses the list of rights and retrieves the different groups of users thanks to LDAP values. In our case, 3 different groups exist: students, professors and interns, and one person is a member of one and only one group. Once the groups have been created and populated, the script takes every single right (for instance *rwxr-r-*) and creates Access Control rules in our formalism. To illustrate this point, let us take an example.

Subject	Group
Kim	Student
April	Student
Tom	Student
Ron	Professor
Leslie	Professor

Table 4.9: Example of subjects and groups.

Let us imagine that our LDAP contains 5 persons (see **Table 4.9**) and that it exists a document docB.txt with rights *"rwxr-r-*". The owner of this document is Kim, who is a member of the group *"Student"*. As Kim is the owner of the document, she has the right to read, write and execute the file. Members of the group (for instance, the group of students), will have the right to read the document as well, while others (i.e. person who are not Kim nor in an authorized group like Student) will only have the right to read the document. Thus, 1 rule with 3 actions will be generated for Kim:

```
<rule name="1">
  <subject subjectName="Kim" group="Student"/>
  <action actionName="Read, Write, Execute"/>
  <resource resourceName="docB.txt"/>
</rule>
```

Then, 2 rules will be generated for members of group "Student", other than Kim:

```
<rule name="2">
  <subject subjectName="April" group="Student"/>
  <action actionName="Read"/>
  <resource resourceName="docB.txt"/>
</rule>
<rule name="3">
  <subject subjectName="Tom" group="Student"/>
  <action actionName="Read"/>
  <resource resourceName="docB.txt"/>
</rule>
```

Finally, 2 rules will be generated for the other subjects:

```

<rule name="4">
  <subject subjectName="Ron" group="Professor"/>
  <action actionName="Read"/>
  <resource resourceName="docB.txt"/>
</rule>
<rule name="5">
  <subject subjectName="Leslie" group="Professor"/>
  <action actionName="Read"/>
  <resource resourceName="docB.txt"/>
</rule>

```

As one can notice, a single Unix mode right can generate many rules, especially if the file can be accessed in many ways and by many subjects or groups.

Tests - Generation mechanism : For this test, we have measured the time consumption of the generation mechanism. To do so, we have generated the TCLs thanks to a Mapping Rule saying that: *"transmission must be confidential"*. Thus, in order to send a document, the sender and the receiver must have access to it, and this document must be encrypted before the transmission.

We have divided the real ACLs (RACLs) into several portions in order to see how the increase of the number of entities will affect the time consumption. These subdivisions of the real ACL are presented in **Table 4.10**. The results of this Table show that the generation mechanism is quite efficient, even with bigger ACLs. Indeed, for many ACLs, it takes less than 3 hours to perform, which is acceptable regarding participants' answers on update frequencies. Moreover, we underline that these tests have been performed on the same machine⁸ as the stochastically generated AC policies. However, we have used an optimized version of the code, with multiple threads and more efficient algorithms, in order to handle the total amount of rules.

Tests - Resource and subject similarity mechanisms : For this test, we have measured the time-consumption of the resource and subject similarity mechanisms on the biggest real ACLs. Results presented in **Table 4.11** show that these mechanisms takes less than 20 minutes to perform. Thus, we consider this mechanism as quite efficient.

Concerning the efficiency of these mechanisms, **Table 4.12** shows that the results are quite good. Indeed, the gain of the resource inferences (resp. subject inferences) can reach 93% (resp. 84%). These results can be explained by the fact that the original engineering school AC policies only contains 3 groups of subjects and that most of the files use the same access rights by default (Unix mode 755), increasing

⁸MacBook Pro Retina, Intel Core i7 2,4Ghz, 16GB RAM, 256 GB SSD hard drive

ID	Rules	Subjects	Resources	Generation time in seconds
RACL 1	23.000	25	500	35
RACL 2	40.000	25	1500	110
RACL 3	150.000	25	6000	760 (13min)
RACL 4	800.000	50	5500	3110 (52min)
RACL 5	1.000.000	50	7500	4850 (80min)
RACL 6	37.000	200	600	1015 (17min)
RACL 7	65.000	200	1200	2500 (42min)
RACL 8	1.000.000	200	2500	5200 (87min)
RACL 9	1.500.000	200	5000	7000 (2h)
RACL 10	2.500.000	200	7500	10800 (3h)

Table 4.10: Summary of the Real ACLs (RACLs) we have used to measure the time-consumption of the generation mechanism.

ID	Rules	Subjects	Resources	Resource similarity mechanism (in minutes)	Subject similarity mechanism (in minutes)
RACL 6	37.000	200	600	3	2
RACL 7	65.000	200	1200	5	4
RACL 8	1.000.000	200	2500	10	7
RACL 9	1.500.000	200	5000	16	13
RACL 10	2.500.000	200	7500	20	15

Table 4.11: Results of the time-consumption tests for the resource and subject similarity mechanisms.

the similarities between resources and subjects. However, great variations in results can be observed. These variations are due to the "*entropic phenomenon*" (see 4.2.1) and thus depend on the ratio between the total number of resources and subjects.

ID	Rules	Subjects	Resources	Resources similarities gain	Subjects similarities gain
RACL 1	23.000	25	500	83%	40%
RACL 2	40.000	25	1500	89%	38%
RACL 3	150.000	25	6000	93%	40%
RACL 4	800.000	50	5500	83%	57%
RACL 5	1.000.000	50	7500	86%	57%
RACL 6	37.000	200	600	8%	84%
RACL 7	65.000	200	1200	15%	83%
RACL 8	1.000.000	200	2500	29%	83%
RACL 9	1.500.000	200	5000	49%	82%
RACL 10	2.500.000	200	7500	55%	81%

Table 4.12: Gain of the resource and subject similarity mechanisms.

Tests - Maintenance of the coherence : For these last tests, we have measured the time-consumption of the mechanisms that are used to maintain coherence. As stated previously, these mechanisms automatically adapt TC or AC policies when the other is modified. Results in **Table 4.13** show that the mechanisms are quite

fast, even with the biggest AC policies. However, we underline that some operations are much slower than the results obtained with stochastically generated ACLs. Indeed, one can see that the actions *"delete an existing subject"* and *"delete an existing resource"* take up to 30 seconds to perform. This can be explained by the fact that the real ACLs have much more rules and that these two actions need to browse the entire rules to update the AC policy. Furthermore, we underline that these processes can be algorithmically optimized with parallelization. In order to be sure that the both paradigms are still coherent after a modification, we have used validation algorithms to check coherence principles P1 and P2 after each action. These principles have been correct every times, proving that both paradigms are still coherent with each other after a modification. In terms of time-consumption, these validations have been performed in a matter of seconds.

Action	RACL_2 in seconds	RACL_6 in seconds	RACL_9 in seconds	RACL_10 in seconds
Create a new subject and add her/him to a specific SC	1	2	3	5
Create a new subject by giving a set of capabilities	1	2	4	5
Modify a subject's capability	1	2	3	5
Move a subject from a subject cluster to another	2	2	3	4
Delete an existing subject	5	9	10	23
Create a new resource and add it to a specific RC	1	3	3	6
Create a new resource by giving a set of capabilities	1	4	4	6
Delete an existing resource	8	13	21	30
Add a new ACL rule	2	4	4	5
Remove an existing ACL rule	2	3	4	6

Table 4.13: Results of the coherence mechanism.

4.3.3 Conclusion

In this section, we have used real AC policies to validate our model. The AC policies have been retrieved from a startup and an engineering school. The startup's policies have been implemented in Apache Fortress, a Java security API, while Posix and LDAP have been used by the engineering school. In order to transform these policies into our formalism, we have used two algorithms. We have implemented a Java program to transform the engineering school's policies into our formalism, while the other algorithm has been implemented by an employee of the startup. We have provided him the general structure of our model (i.e. subjects, actions and resources) to help him with the output of his algorithm. For both AC policies, the scripts have been implemented quite rapidly,

without specific skills other than development. We underline however that, in order to facilitate the understanding of our AC rules by the existing mechanisms (in our case, Apache Fortress or Unix modes), another script is required. This script transforms the generated AC rules back into the existing AC formalism. In our case, an XML parser⁹ has been used to easily and efficiently perform this task.

In order to empirically validate that our model is efficient in terms of time-consumption, we have measured the time-consumption of the generation mechanism. As the tests have been performed with parallelized code, the results are quite good, especially if we consider the fact that real ACLs have much more rules than the stochastically generated ACLs we have used in the previous section.

Concerning the time-consumption and efficiency of the resource similarity mechanism, results of the tests performed on the engineering school's AC have shown that our solution is also quite efficient (less than 20 minutes). Moreover, the results of the tests have shown that these mechanisms are quite efficient in terms of density reduction (up to 93%), which is interesting, especially when many subjects and resources are managed. However, we underline that these results are due to the fact that the original AC policy only contains 3 different groups or subjects and that, by default, the same Posix mode is applied by most users of the school (i.e. 755). Therefore, many users and data have the same rights, and thus, the same rules, inducing these high results.

Finally, management actions (such as deleting an existing subject) have been performed on both AC and TC policies to validate the coherence mechanism. Results have shown that this mechanism is quite efficient, even with the biggest ACLs. However, we underline that actions requiring several loops in the AC policy are slower. We consider that this is not quite a problem because the update frequency of most participants is still covered.

4.4 Conclusion

In this chapter, we have used 3 types of experiments to validate our model. First, we have used an online survey to gather information from security experts and administrators. This survey has been useful to determine the density of the managed policies, the theoretical usefulness of our mechanisms and the perception of the security mechanisms that are used within companies. Then, we have performed empirical tests on stochastically generated policies. These tests have shown that the density used by our participants can be handled by our model, both in terms of efficiency and time-consumption. Finally, we have used real Access Control policies to validate that our model could be used in a real environment. These tests have also

⁹Simple XML Parser : <http://simple.sourceforge.net/>

been conclusive, in terms of time consumption, efficiency (i.e. gain) and coherence keeping. In the next chapter, we conclude this thesis and discuss some perspectives.

Conclusion

Contents

5.1 Summary of our contribution	109
5.1.1 Proposal	109
5.1.2 Tests and validation	110
5.2 Perspectives from the security expert and administrator's point of view	111
5.2.1 Improving the existing mechanisms	111
5.2.2 Insights on usability	112
5.2.3 Model enhancement	113
5.3 Perspectives from the developer's point of view	114
5.3.1 Initial works	114
5.4 Perspectives from the end-users point of view	118

In this chapter, we conclude this thesis by proposing a summary of our contribution and the empirical tests that have been conducted. Then, we conclude with possible enhancement and initial works.

5.1 Summary of our contribution

In this section, we first summarise our contribution by presenting our proposal and the challenges we have decided to tackle. Secondly, we describe the evaluation we have performed in order to test our solution.

5.1.1 Proposal

In this thesis, we have decided to take on research challenges concerning unintentional data leakage within companies. To do so, we have first proposed a meta-model able to represent commonly used AC models in order to tackle the **Challenge C1**. Thanks to that, a security expert or administrator does not have to change the existing AC policies for another model or other mechanisms, which can be time-consuming and tiresome.

Secondly, we have proposed the coherent and semi-automatic generation of TC policies based on existing AC to tackle the coherence problem (**Challenge C2**).

Moreover, we have defined a mechanism that is able to automatically adapt AC policies when the TC policies are modified (and vice versa).

To resolve the density problem (**Challenge C3**), we have proposed inferences mechanisms. These mechanisms detect similar subjects and resources and clusterize them, reducing the overall size of the generated policies and allowing an administrator to reason on sets instead of atomic subjects and resources.

Then, we have used the high abstraction level of our model to generate objects (i.e. Hypermatrices) that represent the overall policies of the companies. Based on the same AC policies, different Hypermatrices with different security levels can be generated. An administrator can then switch from an Hypermatrix to another, allowing fast modification of the overall policies. This mechanism has been proposed to tackle of the adaptability problem (**Challenge C4**).

To solve the interoperability problem (**Challenge C5**) (i.e. differences of security policies between infrastructures), we have proposed mechanisms that can detect similarities and differences between TCLs and Hypermatrices. These mechanisms can generate reports and help security experts or administrators to have a better understanding of their policies. We underline that these mechanisms could be beneficial, especially when 2 infrastructures with different security policies exchange documents.

Finally, we have done several tests on stochastically generated and real AC policies in order to evaluate the performances of our mechanisms. Results of these tests have validated the reactivity challenge (**Challenge C6**), which consists at taking into account the evolution of policies in terms of time-efficiency. Details on these tests are presented in the next section.

5.1.2 Tests and validation

To test our solution, we have proposed 3 different types of evaluation. First, we have conducted an online survey on IT professionals. Thanks to this survey, we have gathered information on the policy size and density, the models that are used and the perception of the participants regarding these models. Moreover, we have been able to propose several mechanisms to tackle the challenges we have underlined. The results of the survey have shown that these mechanisms are appealing for many participants.

Secondly, we have used our model on stochastically generated AC policies. Results of the conducted tests have shown that our model is performant, both in terms of time-consumption and efficiency.

Finally, we have used real AC policies retrieved from a startup company and from an engineering school. Conducted tests have shown that real AC policies can be used by our model. Moreover, time-consumption and gain results have highlighted

the overall efficiency of our proposal.

Now that we have presented our contribution and the results of our thesis, we discuss in the next section the perspectives and future works.

5.2 Perspectives from the security expert and administrator's point of view

In this section, we present the improvements that can be beneficial from the security expert and administrator's perspective. These improvements includes short terms implementations, insights on usability and model enhancement.

5.2.1 Improving the existing mechanisms

In this subsection, we present some ideas to improve some mechanisms of our proof of concept.

Inference mechanisms : Current algorithms used in our inferences mechanisms can be optimized in several ways. Indeed, current resource similarity mechanism is based on a "*perfect*" similarity, meaning that only TCLs that are 100% identical will be in the same Resource Cluster. However, in order to improve the gain, we hypothesize that other modern techniques of supervised learning can be used [Xu & Wunsch 2005]. These techniques are scalable, adapted to big density and could enhance the overall gain, reducing *de facto* the complexity of the policies. Moreover, subject similarity mechanism could also benefits from such techniques, with similar advantages. Finally, other mechanisms such as Data Mining and Artificial Intelligence (AI) [Russell *et al.* 2003] could be used to automatize clustering.

Interoperability mechanisms : Our interoperability mechanisms propose simple but efficient metrics to compare TCLs and Hypermatrices. Among these metrics, one can found the total number of subjects, the distribution of the different types of access or information on capabilities. However, we hypothesize that generating other metrics can be interesting in order to find more similarities and offer more information for security experts and administrators. To compute these metrics in a fast and efficient way, we consider directed-graph theory algorithms as a valuable option. Indeed, these algorithms are usually well known, already implemented, efficient and for some of them, adapted to huge graphs. Examples of these algorithms and notions are: shortest path algorithm, strongly connected component, bridge (or isthmus), density of a graph and maximum degree. Information on these concepts can be found in [Gibbons 1985].

5.2.2 Insights on usability

Throughout our thesis, we have tried to keep in mind usability to propose a solution that can ease the security experts and administrators tiresomeness when it comes to policy management. However, we did not consider the ergonomic aspect. Indeed, we have designed a graphical tool to validate some of our tests, but no other graphical interfaces have been proposed. Thus, we propose in this section some ideas on the enhancement of the Graphical User Interface (GUI) to ease the management tasks.

Enhance the visual representation of graphs : For the tests we have conducted, we have implemented a tool that represents transmission between subjects and capabilities (i.e. what a subject can access and retransmit). However, this tool is not usable when many information are depicted (i.e. when big TCL graphs are depicted for instance). Thus, one research axis is to improve the graph tool. From the visualization point of view, one solution could be to create interactive graphs, allowing the security expert to expand or collapse parts of the graph. Moreover, different levels of additional information could be shown or hidden depending on what the security expert is looking for. For instance, she/he will be willing to focus only on a specific transmission type (ex: TRANSMISSION_DEN) or a specific node type (ex: *full blackhole*). Finally, more specific colour codes could be implemented to ease the reading of complex graphs.

Propose a GUI for the Hypermatrix switch : Another mechanism that could benefit from GUI interfaces is the adaptability mechanism. Indeed, we have underlined in subsection 3.2.6 that changing one Hypermatrix by another is done manually. Thus, a GUI could be used to perform this task. Thanks to this GUI, a security expert could have some information on the current Hypermatrix, the time it has been loaded or its overall information (e.g. total number of subjects, repartition of transmission types) in order to improve the reactivity and efficiency of the administrator or security expert in case of emergencies.

Enhance the reporting : The current version of our work proposes mechanisms to compare TCLs and Hypermatrices. However, these mechanisms generate textual reports. We empathize that these reports can be hard and complex to read. Thus, one solution could be to generate graphical reporting. These results could be depicted in a dashboard, allowing easy and user-friendly customizable representation that could reduce the tiresomeness of having to interpret the reports' results.

Conduct interviews on administrators and security experts : In order to gain insights and increase the validity of our online survey's results, we would like to conduct physical interviews. Unlike the online survey, these interviews will aim at gathering qualitative and in-depth information on the perception and feelings of the participants. Moreover, we would like to test some of our mechanisms to validate that they are usable and understandable for this category of users.

5.2.3 Model enhancement

One of our challenge throughout this thesis has been to take into account as many Access Control models as possible. To do so, we have proposed a simple and generic model. However, this model is not fully generic and lack expressiveness when it comes to complex modelling. This section briefly describes possible enhancements.

Add the notion of context : Context can take many forms and represents many things, including spatial information (e.g. room, building, geolocation, country), temporal information (e.g. specific time of the day such as work hours, specific day, month, year) or technical information (e.g. network IP range, MAC¹ addresses, protocol).

In the context of security, one or more contextual information can be used to determine if an Access Control is granted or not. For instance, *"Access is authorized for this document only if the request comes from an IP within the company's IP range"* or *"Access is denied between 8 P.M and 8 A.M"*.

Moreover, other models used by our participants, such as ABAC, can use contextual information. Thus, even if it increases the overall complexity of a security policy, context can become an asset for both security and management thanks to its fine-grained granularity. Moreover, due to phenomenons like *"Bring Your Own Device"* or employees nomadism, modern IT infrastructures can be willing to add contextual information in their decision mechanism. Thus, we consider that our model could be enhanced to support the notion of context.

Add the notion of "others" : In our model, a TCL embeds the concept of marked subjects. As stated previously, these subjects are the subjects that can access the resources in the AC policy. However, the current version of our model does not provide a way to say that the resource can be sent or received to/by other subjects (i.e. persons that are not present in the existing AC policies). Indeed, there is no easy way to say that a resource is public and that it can be retransmitted to anyone else, inside or outside the company. Thus, we intent to work on this problem by defining the concept of *"others"* subjects. We hypothesize that a TCL could be enhanced with a new row and column. However, without any other modification, this will cause incoherences. Thus, thorough investigation will be needed to tackle this problem.

Allow multiple transmission types for the same transmission : Currently, the model embeds only one transmission type per transmission (see the notion of completeness in 3.1.2). Thus, several security properties (for instance integrity and non-repudiation) cannot be applied to one transmission. To allow the model to take into account such cases, modifications on the Mapping Rules Syntax and the conflict detection mechanism have to be performed.

¹Media Access Control

Embeds other security principles : In chapter 2, we have introduced notions such as Separation of Duty (SoD) and Principle of Least Privileges (PoLP). These notions have been implemented for instance in later version of RBAC. We have seen that our implementation has been able to use an existing RBAC0 policy. However, more complex notions such as the one cited above are difficult to represent in our formalism. Thus, we consider that it will be interesting to adapt our model to take these concepts into account.

Another thing that could be beneficial to integrate is some notions used by Information Rights Management (IRM) and other Usage Control mechanisms. Indeed, we have seen that capabilities in our model can be used to manage the access and retransmission of a resource. However, we consider that capabilities can be enhanced to embed more usage-like control. For instance, we consider that it could be interesting for a security expert to:

- track the path of a document throughout the company,
- change the capabilities of a subject based on contextual information (for instance, disabling the right to retransmit the data if the subject is not connected to the network of the company),
- modify the current capabilities of a subject when she/he ask for a occasional privilege (i.e. break glass procedure [Brucker & Petritsch 2009]),
- destroy a capability after a specific date in order to make the data inaccessible by a specific subject or group of subjects.

Now that the administrator and security expert's perspective have been presented, we focus in the next section on the developer's point of view. This point of view have been chosen to determine how to use the generated TCLs in a company's existing software.

5.3 Perspectives from the developer's point of view

In this section, we present initial works that aim at using the TC policies we have generated thanks to our model.

5.3.1 Initial works

Throughout this thesis, we have underlined that one of our key objectives have been to ease the tiresomeness of having to define and manage both AC and TC policies. Thus, we consider that the task to adapt the generated policies with existing software must be as easy as possible. Existing software must take into account the policies generated by our model. There is no problem from the AC perspective, because an existing application will use the existing AC policies used by our model (it is for instance the case with our real AC policies). However, problems arise from the TC perspective. Indeed, an existing application is not capable of using the TC policies

and authorize or deny the transmissions. One solution can be for a developer to reimplement the application in order to embed mechanisms that can interpret the TC policies and offer an appropriate action (for instance, authorize or deny the transmission). This solution is not acceptable, because reimplementation will cause time-consumption. Moreover, the developer might not be competent enough in computer security to properly implement these mechanisms.

The solution we have started to investigate is to add security mechanisms in a specific place in the applications' code. These specific mechanisms can then intercept information when a resource is about to be accessed or retransmitted and check the corresponding TC policy to verify that the action / transmission is authorized or not. To do so, we have started to use Aspect Oriented Programming (AOP).

Aspect Oriented Programming (AOP) : During the implementation of an application, two main problems can appear: code duplication (or scattering), and code tangling (i.e. dependencies between systems). To overcome these two problems, one can separate the concerns (e.g. particular sets of information that have an effect on the code). To do so, Aspect-Oriented Programming (AOP) can be used. AOP provides mechanisms to dynamically modify the static object-oriented model in order to create a system that can grow to meet new requirements, allowing an application to adopt new characteristics as it grows. Several works have been proposed to tackle security issues with AOP [Kiczales *et al.* 1997]. Moreover, OrBAC researches have also focused on this paradigm [Ayed *et al.* 2013] [Ayed *et al.* 2015]. However, we hypothesize that a model-agnostic solution would be more interesting for a company that does not want to modify its existing applications and redefines its security policies.

In this context, we have started to work on an AOP architecture. This architecture, depicted in **Figure 5.1**, works as follows. An employee (for instance Walter (1)), is using an application that has been developed by the company. This application has been modified to add some joinpoints (2). These joinpoints intercept features that can cause a data leakage or an unauthorized access (in our case a "*send*" functionality (2)). When Walter tries to send docA to Jesse, the joinpoint sends information to a Decisional Engine (DE) (3). Then, the DE has to determine if the action can be performed (4). To do so, it fetches the TCL corresponding to docA and see if the transmission is permitted (5). In our example, the TCL does not have Jesse as a marked subject, preventing him to access the document (6). Thus, Walter cannot send the document to Jesse. Based on this result, the Decisional Engine changes the normal behavior of the application (7) thanks to an AOP concept called advice. This new behavior generates a pop up to warn Walter that the transmission to Jesse is forbidden, preventing him from an unintentional data leak.

Now that the general structure has been presented, we describe the algebra we

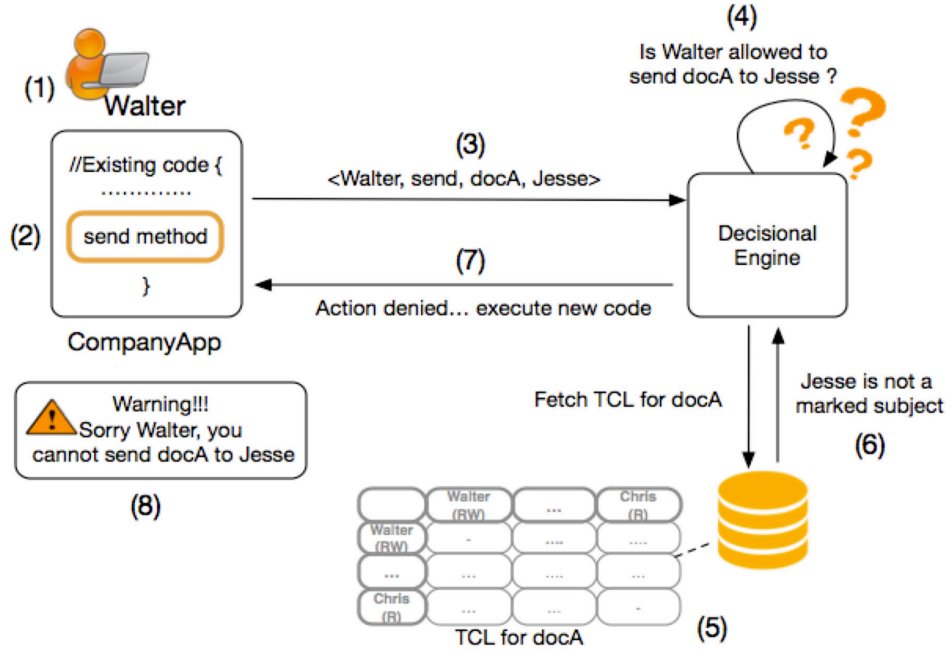


Figure 5.1: General overview of our Decisional Engine architecture.

have started to define. This algebra has been proposed to ease the definition of security measures between the security expert and the DE.

Security algebra : To be as generic as possible, we have started to develop an algebra. This algebra aims at transforming every method of the application that can lead to a data leakage (also called "*sink Action*") into a function. A function is composed of atomic AC actions that can be defined by a security expert or an administrator. For instance, imagine that a subject S wants to copy parts of a document A in a document B . To do so, S will need the following rights: the right to Read A , the right to Write A , the right to Read B and the right to Write B . Thus, in order for a subject to perform a copy-paste from A to B , a security expert or an administrator can formalize the copy-paste action as follows (5.1):

$$copypaste = \{S, \langle r, w \rangle, A\} \wedge \{S, \langle r, w \rangle, B\} \rightarrow AUTH \quad (5.1)$$

Once this function has been defined, the DE has to check if this action is authorized or not. Thus, the DE loads the generated TCL to be sure that the function is correct. If it is not the case, mechanisms can be then implemented to prevent the subject to perform an action that could lead to data leakage. These mechanisms can be for instance: popups, blocking, mail to the administrators, deactivation of the method, etc..

Adding security properties to transmissions : Our model formalizes the concept of transmission types. As stated before, a transmission type can embed security properties, such as confidentiality or integrity. Even if several APIs have been implemented over the years, it can be complex to integrate these security properties to existing transmissions. Indeed, from the developer point of view, having to manage complex cryptographic notions can be difficult to handle and the lack of knowledge regarding security can be hazardous.

To overcome this issue, we aim at using existing works, especially the ones proposed by our team [Nobelis 2008] [Nobelis *et al.* 2010] [Kamel *et al.* 2011]. In these works, the concept of security component has been introduced thanks to Component-Based Software Engineering (CBSE). CBSE is a programming paradigm which consists on a reuse-based approach to define, implement and compose loosely coupled and independent components into systems. A security component is a component that embeds an eponymous security property (e.g. integrity, confidentiality, authenticity, access control and non-repudiation). These security components can be easily used, tweaked (i.e. one can select the size of the key, the specific algorithm to use, etc.) and integrated to existing architecture [Nobelis *et al.* 2011] [Resondry *et al.* 2014]. For these reasons, we aim at using these components to provide security mechanisms to existing transmissions in an easy and reusable way.

Architectural consideration : Previous subsections have proposed ideas to easily modify existing software to take into account the generated TC rules thanks to AOP. Moreover, we aim at using some of the research works of our team to easily embed security properties into existing software. However, several architectural considerations have to be taken into account. Among these considerations, we can highlight:

- How to authenticate a user if the current application does not take this aspect into account ?
- Is the Decisional Engine centralized or decentralized?
- How do the modified applications and the Decisional Engine communicates (e.g. protocol, update frequency, security)?
- How to insure that the capabilities generated by our model are unforgeable?
- How to tag data ?
- How to log the requests sent to the Decisional Engine in order to keep a track of subjects' actions and data flow?
- How to protect the Decisional Engine (for instance, prevent rules and policies from being accessed and altered)?

5.4 Perspectives from the end-users point of view

Throughout this dissertation, we have focused on unintentional data-leakage caused by internal (and in appearance trustworthy) employees. To prevent such leaks, data leakage mechanisms have to be usable and understandable by end-users. Thus, we have conducted a second survey on employees to determine, among other things, how these security mechanisms are perceived by the employees of a company. This survey is described in the next paragraph.

Survey on end-users : The survey is available on the Internet in both french² and english³. It has been answered by more than 125 persons in the last 6 months. The goal of the survey was to gather various information on the employees (the position, the computer skills level, the perception of security and mechanisms and policies, how often sensitive data are manipulated and if they are aware of what they can or cannot do within their company). In addition, more general information regarding the company have also been retrieved (sector, size, etc.). Preliminary results show interesting things and show for instance that:

- 44% of participants have unintentionally leaked data at least once in their current position.
- 57.6% prefer a mechanism that prevent them from unintentional data leakage, even if it does not let them decide.
- more than 40% of them think that security mechanisms are intrusive and annoying for the tasks they perform.

Finally, participants were invited to select and propose their favorite anti data leakage mechanisms. Among the favorite solutions, we can list popup mechanisms to notify users that an action is going to cause a data leakage and let them choose (e.g. popup messages) or a mechanism to automatically deactivate actions that can cause data leakage (for instance, automatically deactivate the "*send*" button when a confidential attached document is put within an email).

Currently, we aim at using these results to provide adapted anti data leakage mechanisms that can be understandable and usable by end users. Moreover, we are compiling the survey's results in order to publish them in an international conference article.

In this chapter, we have first presented a summary of this thesis. Then, we have proposed several perspectives for our work. These perspectives are based on 3 points of view: the security expert and administrator, the developer and the end-user point

²<https://docs.google.com/forms/d/e/1FAIpQLSdyhVrcJbBOBUptjrCsNiVku3Imxj0yVvbJnh8Vaj60taPBpw/viewform>

³<https://docs.google.com/forms/d/e/1FAIpQLSfoASF05G71a8ps1godznShkoHZm3lpi3n5ySMN4l2Utsr6bA/viewform>

of view.

From the security expert and administrator's perspective, we present new implementation, usability insights and model enhancement. Concerning implementation, we have proposed some solutions to enhance the existing mechanisms, such as the inference and interoperability mechanisms. We think that these perspectives could provide good research questions and could be beneficial for our model. Regarding usability, we have proposed to design other Graphical User Interfaces to ease the creation and management of the policies, while offering usable reports for security experts and administrators. Concerning our model, we aim at making it more generic and expressive. Thanks to this expressiveness, we aim at describing more complex AC policies.

From the developer's perspective, we have presented our initial work on Aspect Oriented Programming (AOP). This work aims at using the generated TC policies by adding security mechanisms to an existing application. By doing so, we aim at providing mechanisms that will be able to interpret the TC policies and prevent unintentional data leakage.

From the end-user perspective, and in order to have a better understanding on how the security mechanisms and data leakage are perceived within companies, we have conducted a survey on employees. This survey has been answered by more than a 125 persons, and shows interesting results concerning these concerns.

Summary of the survey's questions

A.1 Questions of the administrators and security experts' survey

In this appendix, we present the questions of the survey that have been asked to the security experts and administrators. As stated in chapter 4, the survey is an online Google forms composed of 20 questions. The survey has been proposed to 50 persons in charge of the security within their company.

Question 1) Your position?

- | |
|--|
| <ul style="list-style-type: none">• System Administrator / Engineer• Network Administrator / Engineer• Security Expert• Other |
|--|

Question 2) Does you company use access control policies?
--

- | |
|--|
| <ul style="list-style-type: none">• Yes• No |
|--|

Question 3) If so, which model are you using?
--

- | |
|---|
| <ul style="list-style-type: none">• ACL (Access Control List) |
|---|

- RBAC (Role-Based Access Control)
- ABAC (Attribute-Based Access Control)
- Other

Question 4) How many users does your access control manage?

- Less than 10 users
- Between 10 and 250 users
- Between 250 and 1.000 users
- Between 1.000 and 5.000 users
- Between 5.000 and 10.000 users
- More than 10.000 users

Question 5) How many sensitive data does your access control manage?

- Less than 100
- Between 100 and 1.000
- Between 1.000 and 5.000
- Between 5.000 and 10.000
- Between 10.000 and 50.000
- Between 50.000 and 100.000
- Between 100.000 and 500.000
- Between 500.000 and 1.000.000
- More than 1.000.000

Question 6) What is your access control policies update frequency?

- Several times per hour
- Several times per day
- Several times per week
- Several times per week
- Several times per month

Question 7) According to you, defining access control policy is:

Score between 1 (a tiresome task) to 5 (an easy task)

Question 8) Do you think that too many entities (users, resources, groups, etc.) are managed by your access control policies?

Score between 1 (strongly disagree) to 5 (fully agree)

Question 9) What do you think of a solution that could reduce the amount of entities managed by your policies?

Score between 1 (not interesting) to 5 (very interesting)

Question 10) How do you manage emergencies (attacks, intrusions, etc.) regarding access control policies?

- Strict policies are applied until the problem is solved.
- There is no policies modification during an emergency.

Question 11) What do you think of a solution that could easily deploy different security policies in case of emergency?

Score between 1 (not interesting) to 5 (very interesting)

Question 12) Does your company use transmission control mechanisms or policies?

- Yes
- No

Question 13) If so, is there a link between access control and transmission control policies?

- Yes, we define one based on the other, taking care of the coherence between the two.
- No, both are defined separately, without taking care of the coherence between the two.

Question 14) If your previous answer was yes, do you think that having to define policies that are coherent with each other is:

Score between 1 (very annoying / hard) to 5 (very enjoyable / easy)

Question 15) What do you think of a solution that could define TC policies based on existing AC policies?

A.1. Questions of the administrators and security experts' survey 125

Score between 1 (not interesting) to 5 (very interesting)

Question 16) What do you think of a solution that could keep coherence between both AC and TC policies?

Score between 1 (not interesting) to 5 (very interesting)

Question 17) Does your company share information with other entities (internal departments, other companies, clients, etc.)?

- Yes
- No

Question 18) If so, do you think that differences between entities policies can be a problem for data exchange?

Score between 1 (very annoying) and 5 (not annoying at all)

Question 19) What do you think of a solution that could detect differences between companies security policies?

Score between 1 (not interesting) to 5 (very interesting)

Question 20) If you have any comments concerning the survey or your policies (for instance the technologies or products that you are using):

Text zone for an open-ended answer

Bibliography

- [Ajam *et al.* 2010a] Nabil Ajam, Nora Cuppens-Boulahia and Frédéric Cuppens. *Contextual privacy management in extended role based access control model*. In Data privacy management and autonomous spontaneous security, pages 121–135. Springer, 2010. (Cited on page 35.)
- [Ajam *et al.* 2010b] Nabil Ajam, Nora Cuppens-Boulahia and Frederic Cuppens. *Privacy Administration in Distributed Service Infrastructure*. In International Conference on Security and Privacy in Communication Systems, pages 53–70. Springer, 2010. (Cited on page 35.)
- [Alawneh & Abbadi 2008] Muntaha Alawneh and Imad M Abbadi. *Preventing information leakage between collaborating organisations*. In Proceedings of the 10th international Conference on Electronic Commerce, page 38. ACM, 2008. (Cited on page 30.)
- [Anderson 2008] Ross Anderson. Security engineering. John Wiley and Sons, 2008. (Cited on pages 14 et 19.)
- [Ausanka-Crues 2001] Ryan Ausanka-Crues. *Methods for access control: advances and limitations*. Harvey Mudd College, vol. 301, 2001. (Cited on page 20.)
- [Autrel *et al.* 2008] Fabien Autrel, Frédéric Cuppens, N Cuppens-Boulahia and Celine Coma. *MotOrBAC 2: a security policy tool*. In 3rd Conference on Security in Network Architectures and Information Systems (SAR-SSI 2008), Loctudy, France, pages 273–288, 2008. (Cited on page 47.)
- [Ayed *et al.* 2008] Samiha Ayed, Nora Cuppens-Boulahia and Frédéric Cuppens. *Managing access and flow control requirements in distributed workflows*. In Computer Systems and Applications, 2008. AICCSA 2008. IEEE/ACS International Conference on, pages 702–710. IEEE, 2008. (Cited on page 35.)
- [Ayed *et al.* 2013] Samiha Ayed, Muhammad Sabir Idrees, Nora Cuppens-Boulahia, Frédéric Cuppens, Mónica Pinto and Lidia Fuentes. *Security aspects: a framework for enforcement of security policies using AOP*. In Signal-Image Technology & Internet-Based Systems (SITIS), 2013 International Conference on, pages 301–308. IEEE, 2013. (Cited on page 115.)
- [Ayed *et al.* 2015] Samiha Ayed, Muhammad Sabir Idrees, Nora Cuppens-Boulahia and Frédéric Cuppens. *Dynamic deployment of access and usage control policies using aspects*. In 2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS), pages 1–6. IEEE, 2015. (Cited on page 115.)

- [Banerjee & Naumann 2004] Anindya Banerjee and David A Naumann. *History-based access control and secure information flow*. In International Workshop on Construction and Analysis of Safe, Secure, and Interoperable Smart Devices, pages 27–48. Springer, 2004. (Cited on page 22.)
- [Barker & Stuckey 2003] Steve Barker and Peter J Stuckey. *Flexible access control policy specification with constraint logic programming*. ACM Transactions on Information and System Security (TISSEC), vol. 6, no. 4, pages 501–546, 2003. (Cited on page 42.)
- [Barker 2009] Steve Barker. *The next 700 access control models or a unifying meta-model?* In Proceedings of the 14th ACM symposium on Access control models and technologies, pages 187–196. ACM, 2009. (Cited on page 42.)
- [Bauer *et al.* 2009] Lujo Bauer, Lorrie Faith Cranor, Robert W Reeder, Michael K Reiter and Kami Vaniea. *Real life challenges in access-control management*. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 899–908. ACM, 2009. (Cited on page 82.)
- [Bauer 2006] Mick Bauer. *Paranoid penguin: an introduction to Novell AppArmor*. Linux Journal, vol. 2006, no. 148, page 13, 2006. (Cited on page 23.)
- [Bell & LaPadula 1973] D Elliott Bell and Leonard J LaPadula. *Secure computer systems: Mathematical foundations*. Technical report, DTIC Document, 1973. (Cited on page 19.)
- [Bertino *et al.* 2001] Elisa Bertino, Piero Andrea Bonatti and Elena Ferrari. *TR-BAC: A temporal role-based access control model*. ACM Transactions on Information and System Security (TISSEC), vol. 4, no. 3, pages 191–233, 2001. (Cited on page 22.)
- [Biba 1977] Kenneth J Biba. *Integrity considerations for secure computer systems*. Technical report, DTIC Document, 1977. (Cited on pages 19 et 20.)
- [Bishop 2005] Matt Bishop. *Introduction to computer security*. 2005. (Cited on page 19.)
- [Bouchami & Perrin 2014] Ahmed Bouchami and Olivier Perrin. *Access control framework within a collaborative PaaS platform*. In Enterprise Interoperability VI, pages 465–476. Springer, 2014. (Cited on page 64.)
- [Bouwman *et al.* 2008] Bart Bouwman, Sjouke Mauw and Milan Petkovic. *Rights management for role-based access control*. In Proc 5th IEEE consumer communications and networking conf CCNC, pages 1085–90, 2008. (Cited on page 33.)

- [Brucker & Petritsch 2009] Achim D Brucker and Helmut Petritsch. *Extending access control models with break-glass*. In Proceedings of the 14th ACM symposium on Access control models and technologies, pages 197–206. ACM, 2009. (Cited on page 114.)
- [Caputo *et al.* 2009] Deanna Caputo, Marcus Maloof and Gregory Stephens. *Detecting insider theft of trade secrets*. IEEE Security and Privacy, vol. 7, no. 6, pages 14–21, 2009. (Cited on page 30.)
- [Center 1988] NCS Center. *A guide to understanding audit in trusted systems*. Technical report, Technical Report NCSC-TG-001, National Computer Security Center, 1988. (Cited on page 16.)
- [Chae *et al.* 2015] Cheol-Joo Chae, YongJu Shin, Kiseok Choi, Ki-Bong Kim and Kwang-Nam Choi. *A privacy data leakage prevention method in P2P networks*. Peer-to-Peer Networking and Applications, pages 1–12, 2015. (Cited on page 30.)
- [Clarke 1999] Roger Clarke. *Introduction to dataveillance and information privacy, and definitions of terms*. Roger Clarke s Dataveillance and Information Privacy Pages, 1999. (Cited on page 14.)
- [Cohen 2003] Julie E Cohen. *DRM and Privacy*. Communications of the ACM, vol. 46, no. 4, pages 46–49, 2003. (Cited on page 33.)
- [Coma *et al.* 2008] Céline Coma, Nora Cuppens-Boulahia, Frédéric Cuppens and Ana-Rosa Cavalli. *Context ontology for secure interoperability*. In Availability, Reliability and Security, 2008. ARES 08. Third International Conference on, pages 821–827. IEEE, 2008. (Cited on page 35.)
- [Corbató & Vyssotsky 1965] Fernando J Corbató and Victor A Vyssotsky. *Introduction and overview of the Multics system*. In Proceedings of the November 30–December 1, 1965, fall joint computer conference, part I, pages 185–196. ACM, 1965. (Cited on page 23.)
- [Corrad *et al.* 2004] Antonio Corrad, Rebecca Montanari and Daniela Tibaldi. *Context-based access control management in ubiquitous environments*. In Network Computing and Applications, 2004.(NCA 2004). Proceedings. Third IEEE International Symposium on, pages 253–260. IEEE, 2004. (Cited on page 22.)
- [Coyne 1996] Edward J Coyne. *Role engineering*. In Proceedings of the first ACM Workshop on Role-based access control, page 4. ACM, 1996. (Cited on pages 18 et 20.)
- [CryptzoneSurvey 2015] CryptzoneSurvey. *Network Security Survey Results: Is Network Access Putting You at Risk?* <https://www.cryptzone.com/pdfs/>

- [Whitepapers/Cryptzone-Network-Access-Security-Survey-2015.pdf](#), 2015. Accessed: 2016-10-02. (Cited on page 82.)
- [Cuppens & Miège 2003] Frédéric Cuppens and Alexandre Miège. *Modelling contexts in the Or-BAC model*. In Computer Security Applications Conference, 2003. Proceedings. 19th Annual, pages 416–425. IEEE, 2003. (Cited on page 34.)
- [Cuppens *et al.* 2007a] Frédéric Cuppens, Nora Cuppens-Boulahia and Meriam Ben Ghorbel. *High level conflict management strategies in advanced access control models*. Electronic Notes in Theoretical Computer Science, vol. 186, pages 3–26, 2007. (Cited on page 35.)
- [Cuppens *et al.* 2007b] Frédéric Cuppens, Nora Cuppens-Boulahia and Meriam Ben Ghorbel. *High level conflict management strategies in advanced access control models*. Electronic Notes in Theoretical Computer Science, vol. 186, pages 3–26, 2007. (Cited on page 35.)
- [Daemen & Rijmen 2013] Joan Daemen and Vincent Rijmen. The design of rijndael: Aes-the advanced encryption standard. Springer Science & Business Media, 2013. (Cited on page 11.)
- [Daud *et al.* 2015] Malik Imran Daud, David Sánchez and Alexandre Viejo. *Ontology-Based Delegation of Access Control: An Enhancement to the XACML Delegation Profile*. In Trust, Privacy and Security in Digital Business, pages 18–29. Springer, 2015. (Cited on page 64.)
- [Dean & Ghemawat 2008] Jeffrey Dean and Sanjay Ghemawat. *MapReduce: simplified data processing on large clusters*. Communications of the ACM, vol. 51, no. 1, pages 107–113, 2008. (Cited on page 72.)
- [Dennis & Van Horn 1983] Jack B Dennis and Earl C Van Horn. *Programming semantics for multiprogrammed computations*. Communications of the ACM, vol. 26, no. 1, pages 29–35, 1983. (Cited on page 18.)
- [di Vimercati *et al.* 2011] Sabrina De Capitani di Vimercati, Sara Foresti, Stefano Paraboschi, Gerardo Pelosi and Pierangela Samarati. *Efficient and private access to outsourced data*. In Distributed Computing Systems (ICDCS), 2011 31st International Conference on, pages 710–719. IEEE, 2011. (Cited on page 30.)
- [Eastlake 3rd & Jones 2001] D Eastlake 3rd and Paul Jones. *US secure hash algorithm 1 (SHA1)*. Technical report, 2001. (Cited on page 12.)
- [Edjlali *et al.* 1998] Guy Edjlali, Anurag Acharya and Vipin Chaudhary. *History-based access control for mobile code*. In Proceedings of the 5th ACM Conference on Computer and Communications Security, pages 38–48. ACM, 1998. (Cited on page 22.)

- [Fabry 1974] Robert S. Fabry. *Capability-based addressing*. Communications of the ACM, vol. 17, no. 7, pages 403–412, 1974. (Cited on page 22.)
- [Ferguson & Senie 1997] Paul Ferguson and Daniel Senie. *Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing*. Technical report, 1997. (Cited on page 13.)
- [Ferraiolo *et al.* 1995] David Ferraiolo, Janet Cugini and D Richard Kuhn. *Role-based access control (RBAC): Features and motivations*. In Proceedings of 11th annual computer security application conference, pages 241–48, 1995. (Cited on page 20.)
- [Forgy 1982] Charles L Forgy. *Rete: A fast algorithm for the many pattern/many object pattern match problem*. Artificial intelligence, vol. 19, no. 1, pages 17–37, 1982. (Cited on page 72.)
- [Gafny *et al.* 2010] Ma ayan Gafny, Asaf Shabtai, Lior Rokach and Yuval Elovici. *Detecting data misuse by applying context-based data linkage*. In Proceedings of the 2010 ACM workshop on Insider threats, pages 3–12. ACM, 2010. (Cited on page 30.)
- [Gar 2016] *Identity and Access Management (IAM)*. <https://www.gartner.com/it-glossary/identity-and-access-management-iam/>, 2016. Accessed: 2015-02-01. (Cited on page 31.)
- [Gasmi *et al.* 2008] Yacine Gasmi, Ahmad-Reza Sadeghi, Patrick Stewin, Martin Unger, Marcel Winandy, Rani Hussein and Christian Stübke. *Flexible and secure enterprise rights management based on trusted virtual domains*. In Proceedings of the 3rd ACM workshop on Scalable trusted computing, pages 71–80. ACM, 2008. (Cited on page 33.)
- [Gheorghe *et al.* 2010] Gabriela Gheorghe, Paolo Mori, Bruno Crispo and Fabio Martinelli. *Enforcing ucon policies on the enterprise service bus*. On the Move to Meaningful Internet Systems, OTM 2010, pages 876–893, 2010. (Cited on page 36.)
- [Gibbons 1985] Alan Gibbons. *Algorithmic graph theory*. Cambridge University Press, 1985. (Cited on page 111.)
- [Gikas 2010] Constantine Gikas. *A General Comparison of FISMA, HIPAA, ISO 27000 and PCI-DSS Standards*. Information Security Journal: A Global Perspective, vol. 19, no. 3, pages 132–141, 2010. (Cited on page 28.)
- [Gligor *et al.* 1998] Virgil D Gligor, Serban I Gavrilă and David Ferraiolo. *On the formal definition of separation-of-duty policies and their composition*. In Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium on, pages 172–183. IEEE, 1998. (Cited on page 15.)

- [Graham & Denning 1972] G Scott Graham and Peter J Denning. *Protection: principles and practice*. In Proceedings of the May 16-18, 1972, spring joint computer conference, pages 417–429. ACM, 1972. (Cited on page 17.)
- [Graves 2007] Kimberly Graves. Ceh: Official certified ethical hacker review guide: Exam 312-50. John Wiley and Sons, 2007. (Cited on page 26.)
- [Gray & Siewiorek 1991] Jim Gray and Daniel P. Siewiorek. *High-availability computer systems*. Computer, vol. 24, no. 9, pages 39–48, 1991. (Cited on page 12.)
- [Greene 2006] Sari Stern Greene. Security policies and procedures. New Jersey: Pearson Education, 2006. (Cited on page 11.)
- [Hamilton *et al.* 2007] Stephen S Hamilton, Martin C Carlisle and John A Hamilton. *A global look at authentication*. In Information Assurance and Security Workshop, 2007. IAW’07. IEEE SMC, pages 1–8. IEEE, 2007. (Cited on page 13.)
- [Harada *et al.* 2004] Toshiharu Harada, Takashi Horie and Kazuo Tanaka. *Task oriented management obviates your onus on Linux*. In Linux Conference, volume 3, 2004. (Cited on page 23.)
- [Harel *et al.* 2010] Amir Harel, Asaf Shabtai, Lior Rokach and Yuval Elovici. *M-score: estimating the potential damage of data leakage incident by assigning misuseability weight*. In Proceedings of the 2010 ACM workshop on Insider threats, pages 13–20. ACM, 2010. (Cited on page 30.)
- [Harel *et al.* 2012] Amir Harel, Asaf Shabtai, Lior Rokach and Yuval Elovici. *M-score: A misuseability weight measure*. IEEE transactions on dependable and secure computing, vol. 9, no. 3, pages 414–428, 2012. (Cited on page 30.)
- [Harrison *et al.* 1976] Michael A Harrison, Walter L Ruzzo and Jeffrey D Ullman. *Protection in operating systems*. Communications of the ACM, vol. 19, no. 8, pages 461–471, 1976. (Cited on page 17.)
- [Hauer 2015] Barbara Hauer. *Data and Information Leakage Prevention Within the Scope of Information Security*. IEEE Access, vol. 3, pages 2554–2565, 2015. (Cited on page 25.)
- [Hill & Marty 2008] Mark D Hill and Michael R Marty. *Amdahl’s law in the multi-core era*. 2008. (Cited on page 68.)
- [Hilty *et al.* 2007] Manuel Hilty, Alexander Pretschner, David Basin, Christian Schaefer and Thomas Walter. *A policy language for distributed usage control*. In Computer Security–ESORICS 2007, pages 531–546. Springer, 2007. (Cited on page 36.)

- [Houdek *et al.* 1981] Merle E Houdek, Frank G Soltis and Roy L Hoffman. *IBM System/38 support for capability-based addressing*. In Proceedings of the 8th annual symposium on Computer Architecture, pages 341–348. IEEE Computer Society Press, 1981. (Cited on page 23.)
- [Hu *et al.* 2006] Vincent C Hu, David Ferraiolo and D Richard Kuhn. Assessment of access control systems. US Department of Commerce, National Institute of Standards and Technology, 2006. (Cited on page 21.)
- [Hu *et al.* 2013] Vincent C Hu, David Ferraiolo, Rick Kuhn, Arthur R Friedman, Alan J Lang, Margaret M Cogdell, Adam Schnitzer, Kenneth Sandlin, Robert Miller, Karen Scarfone *et al.* *Guide to attribute based access control (ABAC) definition and considerations (draft)*. NIST Special Publication, vol. 800, page 162, 2013. (Cited on page 21.)
- [Hur 2013] Junbeom Hur. *Improving security and efficiency in attribute-based data sharing*. Knowledge and Data Engineering, IEEE Transactions on, vol. 25, no. 10, pages 2271–2282, 2013. (Cited on page 64.)
- [Jaeger *et al.* 2003] Trent Jaeger, Reiner Sailer and Xiaolan Zhang. *Analyzing integrity protection in the SELinux example policy*. In Proceedings of the 12th conference on USENIX Security Symposium-Volume 12, pages 5–5. USENIX Association, 2003. (Cited on page 23.)
- [Jeager 2001] Trent Jeager. *Managing access control complexity using metrices*. In Proceedings of the sixth ACM symposium on Access control models and technologies, pages 131–139. ACM, 2001. (Cited on page 55.)
- [Jiang *et al.* 2007] Cheng-Huang Jiang, Shiuhpyng Shieh and Jen-Chien Liu. *Keystroke statistical learning model for web authentication*. In Proceedings of the 2nd ACM symposium on Information, computer and communications security, pages 359–361. ACM, 2007. (Cited on page 14.)
- [Jin *et al.* 2012] Xin Jin, Ram Krishnan and Ravi Sandhu. *A unified attribute-based access control model covering DAC, MAC and RBAC*. In IFIP Annual Conference on Data and Applications Security and Privacy, pages 41–55. Springer, 2012. (Cited on page 21.)
- [Johnson & Filkins 2012] Kevin Johnson and Barbara L Filkins. *SANS mobility - BYOD security survey*. SANS Institute Whitepaper, 2012. http://www.sans.org/reading_room/analysts/program/mobility-sec-survey.pdf. (Cited on page 82.)
- [Kagal *et al.* 2001] Lalana Kagal, Tim Finin and Anupam Joshi. *Trust-based security in pervasive computing environments*. Computer, vol. 34, no. 12, pages 154–157, 2001. (Cited on page 22.)

- [Kalam *et al.* 2003] Anas Abou El Kalam, RE Baida, Philippe Balbiani, Salem Benferhat, Frédéric Cuppens, Yves Deswarte, Alexandre Mieke, Claire Saurel and Gilles Trouessin. *Organization based access control*. In Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on, pages 120–131. IEEE, 2003. (Cited on page 34.)
- [Kamel *et al.* 2011] Michel Kamel, Karima Boudaoud, Stassia Resondry and Michel Riveill. *A low-energy consuming and user-centric security management architecture adapted to mobile environments*. In 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops, pages 722–725. IEEE, 2011. (Cited on page 117.)
- [Kandala *et al.* 2011] Savith Kandala, Ravi Sandhu and Venkata Bhamidipati. *An attribute based framework for risk-adaptive access control models*. In Availability, Reliability and Security (ARES), 2011 Sixth International Conference on, pages 236–241. IEEE, 2011. (Cited on page 22.)
- [Karger 2005] Paul A Karger. *Multi-level security requirements for hypervisors*. In 21st Annual Computer Security Applications Conference (ACSAC 05), pages 9–pp. IEEE, 2005. (Cited on page 19.)
- [Karp *et al.* 2010] Alan Karp, Harry Haury and Michael Davis. *From ABAC to ZBAC: the evolution of access control models*. In International Conference on Information Warfare and Security, page 202. Academic Conferences International Limited, 2010. (Cited on page 22.)
- [Kiczales *et al.* 1997] Gregor Kiczales, John Lamping, Anurag Mendhekar, Chris Maeda, Cristina Lopes, Jean-Marc Loingtier and John Irwin. *Aspect-oriented programming*. In European conference on object-oriented programming, pages 220–242. Springer, 1997. (Cited on page 115.)
- [Kocher 1996] Paul C Kocher. *Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems*. In Annual International Cryptology Conference, pages 104–113. Springer, 1996. (Cited on page 12.)
- [Kremer *et al.* 2002] Steve Kremer, Olivier Markowitch and Jianying Zhou. *An intensive survey of fair non-repudiation protocols*. Computer communications, vol. 25, no. 17, pages 1606–1621, 2002. (Cited on page 14.)
- [Kulkarni & Tripathi 2008] Devdatta Kulkarni and Anand Tripathi. *Context-aware role-based access control in pervasive computing systems*. In Proceedings of the 13th ACM symposium on Access control models and technologies, pages 113–122. ACM, 2008. (Cited on page 22.)
- [Lamport 1981] Leslie Lamport. *Password authentication with insecure communication*. Communications of the ACM, vol. 24, no. 11, pages 770–772, 1981. (Cited on page 13.)

- [Lampson 1973] Butler W Lampson. *A note on the confinement problem*. Communications of the ACM, vol. 16, no. 10, pages 613–615, 1973. (Cited on page 35.)
- [Lampson 1974] Butler W Lampson. *Protection*. ACM SIGOPS Operating Systems Review, vol. 8, no. 1, pages 18–24, 1974. (Cited on page 17.)
- [Lang *et al.* 2006] Bo Lang, Ian Foster, Frank Siebenlist, Rachana Ananthakrishnan and Tim Freeman. *Attribute based access control for grid computing*. 2006. (Cited on page 21.)
- [Langheinrich 2001] Marc Langheinrich. *Privacy by design-principles of privacy-aware ubiquitous systems*. In International conference on Ubiquitous Computing, pages 273–291. Springer, 2001. (Cited on page 14.)
- [Le *et al.* 2015] Ha Thanh Le, Cu Duy Nguyen, Lionel Briand and Benjamin Hourte. *Automated Inference of Access Control Policies for Web Applications*. In Proceedings of the 20th ACM Symposium on Access Control Models and Technologies, pages 27–37. ACM, 2015. (Cited on page 55.)
- [Levy 2014] Henry M Levy. *Capability-based computer systems*. Digital Press, 2014. (Cited on page 23.)
- [Li *et al.* 2015] Hao Li, Zewu Peng, Xinyao Feng and Hongxia Ma. *Leakage Prevention Method for Unstructured Data Based on Classification*. In Applications and Techniques in Information Security, pages 337–343. Springer, 2015. (Cited on page 30.)
- [Liu *et al.* 2003] Qiong Liu, Reihaneh Safavi-Naini and Nicholas Paul Sheppard. *Digital rights management for content distribution*. In Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003-Volume 21, pages 49–58. Australian Computer Society, Inc., 2003. (Cited on page 33.)
- [Marinovic *et al.* 2014] Srdjan Marinovic, Naranker Dulay and Morris Sloman. *Rumpole: an introspective break-glass access control language*. ACM Transactions on Information and System Security (TISSEC), vol. 17, no. 1, page 2, 2014. (Cited on page 61.)
- [Mathew *et al.* 2010] Sunu Mathew, Michalis Petropoulos, Hung Q Ngo and Shambhu Upadhyaya. *A data-centric approach to insider attack detection in database systems*. In International Workshop on Recent Advances in Intrusion Detection, pages 382–401. Springer, 2010. (Cited on page 30.)
- [Mayer 1982] Alastair JW Mayer. *The architecture of the Burroughs B5000: 20 years later and still ahead of the times?* ACM SIGARCH Computer Architecture News, vol. 10, no. 4, pages 3–10, 1982. (Cited on page 22.)

- [McAfee 2015] McAfee. *Grand theft Data*. Technical report, 2015. (Cited on page 25.)
- [McCullagh & Homsy 2005] Declan McCullagh and Milana Homsy. *Leave DRM Alone: A Survey of Legislative Proposals Relating to Digital Rights Management Technology and Their Problems*. Mich. St. L. Rev., page 317, 2005. (Cited on page 33.)
- [McCullough 1987] Daryl McCullough. *Specifications for multi-level security and a hook-up*. In Security and Privacy, 1987 IEEE Symposium on, pages 161–161. IEEE, 1987. (Cited on page 19.)
- [McLean 1985] John McLean. *A comment on the âbasic security theoremâof Bell and LaPadula*. Information Processing Letters, vol. 20, no. 2, pages 67–70, 1985. (Cited on page 19.)
- [Mettler et al. 2010] Adrian Mettler, David Wagner and Tyler Close. *Joe-E: A Security-Oriented Subset of Java*. In NDSS, volume 10, pages 357–374, 2010. (Cited on page 18.)
- [Microsoft 2016] Microsoft. *Mandatory Integrity Control*, 2016. [https://msdn.microsoft.com/fr-fr/library/windows/desktop/bb648648\(v=vs.85\).aspx](https://msdn.microsoft.com/fr-fr/library/windows/desktop/bb648648(v=vs.85).aspx). (Cited on page 23.)
- [Miller et al. 2003] Mark S Miller, Ka-Ping Yee, Jonathan Shapiro et al. *Capability myths demolished*. Technical report, Technical Report SRL2003-02, Johns Hopkins University Systems Research Laboratory, 2003. <http://www.erights.org/elib/capability/duals>, 2003. (Cited on page 18.)
- [Miller et al. 2008] Mark S Miller, Mike Samuel, Ben Laurie, Ihab Awad and Mike Stay. *Safe active content in sanitized JavaScript*. Google, Inc., Tech. Rep, 2008. (Cited on page 18.)
- [Mogull & Securosis 2007] Rich Mogull and LLC Securosis. *Understanding and selecting a data loss prevention solution*. Technical report, SANS Institute, 2007. (Cited on page 27.)
- [Morisset & Zannone 2014] Charles Morisset and Nicola Zannone. *Reduction of access control decisions*. In Proceedings of the 19th ACM symposium on Access control models and technologies, pages 53–62. ACM, 2014. (Cited on page 55.)
- [Nelson et al. 2010] Timothy Nelson, Christopher Barratt, Daniel J Dougherty, Kathi Fisler and Shriram Krishnamurthi. *The Margrave Tool for Firewall Analysis*. In LISA, 2010. (Cited on page 47.)
- [NIST 2006] NIST NIST. *Interagency Report 7316 Access to Access Control Systems*, Vincent C. Hu et al, 2006. (Cited on page 18.)

- [Nobelis *et al.* 2010] Nicolas Nobelis, Karima Boudaoud and Michel Kamel. *A user-centric approach for secure communication protocols*. In Architects of Secure Networks (ASIGE10), NATO Science for Peace and Security Program,, 2010. (Cited on page 117.)
- [Nobelis *et al.* 2011] Nicolas Nobelis, Karima Boudaoud, Christian Delettre and Michel Riveill. *A Component-Based Approach to Security Protocol Design*. In Advanced Information Networking and Applications (WAINA), 2011 IEEE Workshops of International Conference on, pages 279–284. IEEE, 2011. (Cited on page 117.)
- [Nobelis 2008] Nicolas Nobelis. *Une architecture pour le transfert électronique sécurisé de document*. PhD thesis, École doctorale Sciences et technologies de l’information et de la communication (Sophia Antipolis, Alpes-Maritimes), 2008. (Cited on page 117.)
- [OAS 2014] XACML Data Loss Prevention. <https://docs.oasis-open.org/xacml/xacml-3.0-dlp-nac/v1.0/cs01/xacml-3.0-dlp-nac-v1.0-cs01.pdf>, 2014. Accessed: 2015-01-05. (Cited on page 37.)
- [Park & Sandhu 2002] Jaehong Park and Ravi Sandhu. *Towards usage control models: beyond traditional access control*. In Proceedings of the seventh ACM symposium on Access control models and technologies, pages 57–64. ACM, 2002. (Cited on page 36.)
- [Park & Sandhu 2004] Jaehong Park and Ravi Sandhu. *The UCON ABC usage control model*. ACM Transactions on Information and System Security (TISSEC), vol. 7, no. 1, pages 128–174, 2004. (Cited on page 36.)
- [Proctor *et al.* 2008] Mark Proctor, Michael Neale, Peter Lin and Michael Frandsen. *Drools documentation*. JBoss, vol. 5, no. 05, page 2008, 2008. (Cited on page 72.)
- [Rentsch 1982] Tim Rentsch. *Object oriented programming*. ACM Sigplan Notices, vol. 17, no. 9, pages 51–57, 1982. (Cited on page 23.)
- [Resondry *et al.* 2014] Stassia Resondry, Karima Boudaoud, Michel Kamel, Yoann Bertrand and Michel Riveill. *An alternative version of HTTPS to provide non-repudiation security property*. In 2014 International Wireless Communications and Mobile Computing Conference (IWCMC), pages 536–541. IEEE, 2014. (Cited on page 117.)
- [Rezaeibagha & Mu 2016] Fatemeh Rezaeibagha and Yi Mu. *Distributed clinical data sharing via dynamic access-control policy transformation*. International Journal of Medical Informatics, vol. 89, pages 25–31, 2016. (Cited on page 61.)

- [Rinard *et al.* 2004] Martin C Rinard, Cristian Cadar, Daniel Dumitran, Daniel M Roy, Tudor Leu and William S Beebe. *Enhancing Server Availability and Security Through Failure-Oblivious Computing*. In OSDI, volume 4, pages 21–21, 2004. (Cited on page 12.)
- [Ritchie & Thompson 1978] OM Ritchie and Ken Thompson. *The UNIX time-sharing system*. The Bell System Technical Journal, vol. 57, no. 6, pages 1905–1929, 1978. (Cited on page 13.)
- [Rivest *et al.* 1978] Ronald L Rivest, Adi Shamir and Leonard Adleman. *A method for obtaining digital signatures and public-key cryptosystems*. Communications of the ACM, vol. 21, no. 2, pages 120–126, 1978. (Cited on page 12.)
- [Rivest 1992] Ronald Rivest. *The MD5 message-digest algorithm*. 1992. (Cited on page 12.)
- [Rosen 2012] Jeffrey Rosen. *The right to be forgotten*. Stanford law review online, vol. 64, page 88, 2012. (Cited on page 14.)
- [Rosenblatt *et al.* 2001] William Rosenblatt, Stephen Mooney and William Trippe. *Digital rights management: business and technology*. John Wiley & Sons, Inc., 2001. (Cited on page 33.)
- [Russell *et al.* 2003] Stuart Jonathan Russell, Peter Norvig, John F Canny, Jitendra M Malik and Douglas D Edwards. *Artificial intelligence: a modern approach*, volume 2. Prentice hall Upper Saddle River, 2003. (Cited on page 111.)
- [Saltzer & Schroeder 1975] Jerome H Saltzer and Michael D Schroeder. *The protection of information in computer systems*. Proceedings of the IEEE, vol. 63, no. 9, pages 1278–1308, 1975. (Cited on page 17.)
- [Saltzer 1974] Jerome H Saltzer. *Protection and the control of information sharing in Multics*. Communications of the ACM, vol. 17, no. 7, pages 388–402, 1974. (Cited on page 15.)
- [Samarati & de Vimercati 2000] Pierangela Samarati and Sabrina Capitani de Vimercati. *Access control: Policies, models, and mechanisms*. In International School on Foundations of Security Analysis and Design, pages 137–196. Springer, 2000. (Cited on page 15.)
- [Sandhu & Park 2003] Ravi Sandhu and Jaehong Park. *Usage control: A vision for next generation access control*. In International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security, pages 17–31. Springer, 2003. (Cited on page 36.)
- [Sandhu *et al.* 1996] Ravi S Sandhu, Edward J Coyne, Hal L Feinstein and Charles E Youman. *Role-based access control models*. Computer, no. 2, pages 38–47, 1996. (Cited on page 20.)

- [SANSSurvey 2014] SANSSurvey. *SANS Institute InfoSec Reading Room, Cybersecurity Professional Trends: A SANS Survey*. <http://www.sans.org/reading-room/whitepapers/analyst/cybersecurity-professional-trends-survey-34615>, 2014. Accessed: 2016-10-02. (Cited on page 82.)
- [Schauffler 2008] Casey Schauffler. *Smack in embedded computing*. In Proc. Ottawa Linux Symposium, 2008. (Cited on page 23.)
- [Selimi & Freitag 2014] Mennan Selimi and Felix Freitag. *Tahoe-lafs distributed storage service in community network clouds*. In Big Data and Cloud Computing (BdCloud), 2014 IEEE Fourth International Conference on, pages 17–24. IEEE, 2014. (Cited on page 18.)
- [Shabtai *et al.* 2012] Asaf Shabtai, Yuval Elovici and Lior Rokach. A survey of data leakage detection and prevention solutions. Springer Science & Business Media, 2012. (Cited on page 27.)
- [Shapiro *et al.* 1999] Jonathan S Shapiro, Jonathan M Smith and David J Farber. Eros: a fast capability system, volume 33. ACM, 1999. (Cited on page 24.)
- [Shapiro *et al.* 2007] Jonathan S Shapiro, Eric Northup, M Scott Doerrie, Swaroop Sridhar, Neal H Walfield and Marcus Brinkmann. *Coyotos microkernel specification*. The EROS Group, LLC, 0.5 edition, 2007. (Cited on page 24.)
- [Slimani *et al.* 2011a] Nadera Slimani, Hemanth Khambhammettu, Kamel Adi and Luigi Logrippo. *UACML: Unified access control modeling language*. In New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference on, pages 1–8. IEEE, 2011. (Cited on page 42.)
- [Slimani *et al.* 2011b] Nadera Slimani, Hemanth Khambhammettu, Kamel Adi and Luigi Logrippo. *UACML: Unified access control modeling language*. In New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference on, pages 1–8. IEEE, 2011. (Cited on page 64.)
- [Smalley 2001] Stephen Smalley. *Which operating system access control technique will provide the greatest overall benefit to users?* In Proceedings of the sixth ACM symposium on Access control models and technologies, pages 147–148. ACM, 2001. (Cited on page 18.)
- [Soliman *et al.* 2015] Ahmed H Soliman, Maged H Ibrahim and Salwa H El-Ramly. *Enhancing efficiency of enterprise digital rights management*. In 2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS), pages 91–96. IEEE, 2015. (Cited on page 33.)
- [Stepien *et al.* 2012] Bernard Stepien, Hemanth Khambhammettu, Kamel Adi and Luigi Logrippo. *CatBAC: A generic framework for designing and validating*

- hybrid access control models*. In 2012 IEEE International Conference on Communications (ICC), pages 6721–6726. IEEE, 2012. (Cited on page 42.)
- [Tcsec 1985] DOD Tcsec. *Trusted computer system evaluation criteria*. DoD 5200.28-STD, vol. 83, 1985. (Cited on pages 16 et 19.)
- [Tolone et al. 2005] William Tolone, Gail-Joon Ahn, Tanusree Pai and Seng-Phil Hong. *Access control in collaborative systems*. ACM Computing Surveys (CSUR), vol. 37, no. 1, pages 29–41, 2005. (Cited on page 64.)
- [van Beek 2007] MH van Beek. *Comparison of enterprise digital rights management systems*. Advice report, Aia Software, 2007. (Cited on page 33.)
- [Van Tassel 2006] Joan Van Tassel. *Digital rights management*. Taylor & Francis, 2006. (Cited on page 33.)
- [Vance et al. 2007] Christopher Vance, Todd Miller, Rob Dekelbaum and A Reisse. *Security-enhanced darwin: Porting selinux to mac os x*. In Proceedings of the Third Annual Security Enhanced Linux Symposium, Baltimore, MD, USA. Citeseer, 2007. (Cited on page 23.)
- [Venkatasubramanian et al. 2014] Krishna K Venkatasubramanian, Tridib Mukherjee and Sandeep KS Gupta. *CAAC—An Adaptive and Proactive Access Control Approach for Emergencies in Smart Infrastructures*. ACM Transactions on Autonomous and Adaptive Systems (TAAS), vol. 8, no. 4, page 20, 2014. (Cited on page 61.)
- [Wang et al. 2004] Lingyu Wang, Duminda Wijesekera and Sushil Jajodia. *A logic-based framework for attribute based access control*. In Proceedings of the 2004 ACM workshop on Formal methods in security engineering, pages 45–55. ACM, 2004. (Cited on page 21.)
- [Warren & Brandeis 1890] Samuel D Warren and Louis D Brandeis. *The right to privacy*. Harvard law review, pages 193–220, 1890. (Cited on page 14.)
- [Watson et al. 2003] Robert Watson, Wayne Morrison, Chris Vance and Brian Feldman. *The TrustedBSD MAC Framework: Extensible Kernel Access Control for FreeBSD 5.0*. In USENIX Annual Technical Conference, FREENIX Track, pages 285–296, 2003. (Cited on page 23.)
- [Wayman et al. 2005] James Wayman, Anil Jain, Davide Maltoni and Dario Maio. *An introduction to biometric authentication systems*. Springer, 2005. (Cited on page 13.)
- [Weinshall 2006] Daphna Weinshall. *Cognitive authentication schemes safe against spyware*. In 2006 IEEE Symposium on Security and Privacy (S&P’06), pages 6–pp. IEEE, 2006. (Cited on page 14.)

- [Wilcox-O’Hearn & Warner 2008] Zooko Wilcox-O’Hearn and Brian Warner. *Tahoe: the least-authority filesystem*. In Proceedings of the 4th ACM international workshop on Storage security and survivability, pages 21–26. ACM, 2008. (Cited on page 23.)
- [Xu & Wunsch 2005] Rui Xu and Donald Wunsch. *Survey of clustering algorithms*. IEEE Transactions on neural networks, vol. 16, no. 3, pages 645–678, 2005. (Cited on page 111.)
- [Yang *et al.* 2013] Jen-Ho Yang, Hung-Ming Sun and Ping-Liang Chen. *An enterprise digital right management scheme with anonymous trust for mobile devices*. Informatica, vol. 37, no. 3, page 307, 2013. (Cited on page 33.)
- [Yu & Chiueh 2004] Yang Yu and Tzi-cker Chiueh. *Enterprise digital rights management: Solutions against information theft by insiders*. Research Proficiency Examination (RPE) report TR-169, Department of Computer Science, Stony Brook University, 2004. (Cited on page 33.)
- [ZDN 2007] *Trend Micro makes DLP move, Symantec stands pat*. <http://www.zdnet.com/article/trend-micro-makes-dlp-move-symantec-stands-pat/>, 2007. Accessed: 2015-10-10. (Cited on page 30.)
- [Zhou & Gollmann 1996] Jinaying Zhou and Dieter Gollmann. *A fair non-repudiation protocol*. In IEEE symposium on security and privacy, pages 55–61. Citeseer, 1996. (Cited on page 14.)
- [Zilberman *et al.* 2011] Polina Zilberman, Shlomi Dolev, Gilad Katz, Yuval Elovici and Asaf Shabtai. *Analyzing group communication for preventing data leakage via email*. In Intelligence and Security Informatics (ISI), 2011 IEEE International Conference on, pages 37–41. IEEE, 2011. (Cited on page 30.)

Gestion du contrôle de la diffusion des données d'entreprises et politiques de contrôles d'accès

Résumé : Cette thèse traite des problèmes de fuite de données accidentelles au sein des entreprises. Ces fuites peuvent être dues à l'utilisation conjointe de politiques de Contrôle d'Accès (CA) et de Contrôle de Transmission (CT). De plus, l'utilisation conjointe de ces deux types de politique génère plusieurs problèmes pour les personnes ayant la charge de créer et maintenir ces politiques. Parmi ces problèmes, nous pouvons citer des problèmes de généricité des modèles existants, de cohérence entre les règles de CA et de CT ainsi que des problèmes de densité, d'adaptabilité, d'interopérabilité et de réactivité.

Dans cette thèse, nous proposons en premier lieu un méta-modèle pour prendre en compte la plupart des modèles de CA utilisés dans les entreprises. Nous proposons ensuite la génération cohérente et semi-automatique des politiques de CT à partir de politiques de CA existantes pour répondre au problème de cohérence. De plus, différentes fonctionnalités sont proposées pour résoudre les problèmes de densité, d'adaptabilité et d'interopérabilité. Afin de valider la pertinence de notre solution, nous proposons une étude (type questionnaire) auprès d'experts sécurité et d'administrateurs. Cette étude révèle des informations sur la taille des politiques gérées, la pénibilité à les définir ou encore l'utilité des fonctionnalités proposées pour résoudre les problèmes précédents. Enfin, nous testons notre preuve de concept sur des données aléatoires et réelles en prenant en compte les performances et la réactivité, validant ainsi que notre solution répond bien aux problèmes soulevés.

Mots clés : Sécurité informatique, Contrôle d'Accès, Contrôle de Transmission, Sécurité des Données, Fuite de données

Access Control Policies and companies data transmission management

Abstract: The main objective of this thesis is to solve the problem of unintentional data leakage within companies. These leaks can be caused by the use of both Access Control (AC) and Transmission Control (TC) policies. Moreover, using both AC and TC can lead to many problems for the security experts and the administrators that are in charge of the definition and maintenance of such policies. Among these problems, we can underline the genericity problem of existing models, the coherence problem between AC and TC rules and problems such as density, adaptability, interoperability and reactivity.

In this thesis, we first define a meta-model to take into account the main AC models that are used within companies. We also propose a coherent and semi-automatic generation of TC policies based on existing AC to tackle the coherence problem. Moreover, several mechanisms have been proposed to tackle complexity, adaptability and interoperability issues. In order to validate the relevance of our solution, we have first conducted a survey among security experts and administrators. This survey has highlighted several information regarding the policies' size and density, the tiresomeness of having to define them and the interest for several functionalities that can cover the aforementioned problems. Finally, our solution has been tested on stochastically generated and real policies in order to take performances and reactivity under consideration. Results of these tests have validated that our solution covers the underlined problems.

Keywords: IT security, Access Control, Transmission Control, Data Security, Data Leakage
