MASTER EN SCIENCES INFORMATIQUES

Architecture d'autorisation SAML et les certificats d'attribut

Wathelet, Denis

Award date:
2004

FACULTÉS UNIVERSITAIRES NOTRE-DAME DE LA PAIX, NAMUR
Institut d'Informatique
Année Académique 2003-2004

Architecture d'autorisation SAML et les certificats d'attribut

Denis Wathelet

Mémoire présenté en vue de l'obtention
du grade de Maître en Informatique

Résumé

Ce mémoire présente les concepts généraux communs à la plupart des architectures visant à mettre en place un procédé d'autorisation. Il présente ensuite les différentes méthodes et standards disponibles pour la communication d'informations relatives à l'autorisation. L'utilisation de ces standards est ensuite illustrée au travers de travaux de recherche les utilisant. Il se termine par l'étude d'un cas concret dans le domaine bancaire et des multi signatures.

Mots clé: Architecture d'autorisation, SAML, Certificat d'attribut, Certificat de clé publique, Single Sign-On.

Abstract

This thesis introduces the basics concepts used in almost every authorization framework. It also presents various standards used to convey the information relative to authorization. Afterwards some experimental works using these will be presented. It concludes with a concrete application study in the banking area with multi-signatures.

Keyword: Authorization Framework, SAML, Attribute Certificate, Public Key certificate, Single Sign-On (SSO)

Remerciements

Je tiens tout d'abord à remercier mon promoteur, Monsieur Jean Ramaekers, professeur aux facultés universitaires Notre-Dame de la Paix de m'avoir donné l'opportunité de réaliser mon stage dans l'entreprise Isabel. Je le remercie également pour ses judicieux conseils et sa disponibilité tout au long de la rédaction de ce mémoire.

Je désire ensuite remercier Monsieur John Pyrgies pour l'intérêt qu'il m'a apporté pendant toute la durée du stage. Merci aussi pour m'avoir aidé à réaliser un premier plan de ce mémoire ainsi que pour ses nombreuses critiques et lectures attentives.

Mes remerciements iront encore à toute l'équipe d'Isabel, Aimé Kassa, Luc Preumont, Jeanine Kesteloot, Dimitri Saucez et Corinne Gérardin pour leur accueil chaleureux et pour leur bonne humeur au quotidien.

Un merci particulier à Nicolas Lambot ainsi qu'à ma fiancée pour leur relecture attentive et pour leurs critiques.

Enfin merci à toutes les personnes qui m'ont soutenu durant mon cursus et qui m'ont apporté leur soutien durant la rédaction de ce mémoire.

Table des matières

Illustrations

Tableaux

Abréviations

AA	Attribute authority : autorité responsable de l'émission de certificat d'attribut dans une PMI.
AC	Attribute certificate : certificat d'attribut.
ADF	Access control Decision Function : synonyme de PDP
AEF	Access control Enforcement Function : synonyme de PEP
ARL	Attribute Revocation List : liste des certificats d'attribut ayant été révoqué
CA	Certification Authority
CRL	Certificate Revocation List : liste des certificats à clé publique ayant été révoqués
DN	Distinguished Name : structure de donnée permettant d'identifier une personne (physique ou morale)
FTP	File Transfert Protocol
HTTP	Hyper Text Transfert Protocol
LDAP	Lightweight Directory Access Protocol :Structure d'annuaire standard
OASIS	Organization for the Advancement of Structured Information Standards
OCSP	Online Certificate Status Protocol
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIP	Policy Information Point
PKC	Public Key Cryptography
PKI	Public Key Infrastructure
PMI	Privilege Managment Infrastructure
PRP	Policy Retrieval Point
RFC	Request For Comment
SAML	Security Assertions Markup Language
SMTP	Simple Mail Transfer Protocol

SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
SSO	Single Sign-On
TTP	Trusted Third Party: tiers de confiance
URL	Uniform Resource Locator
XML	Extensible Markup Language

1. Introduction

L'information est omniprésente autour de nous. Et avec elle sa gestion, son contrôle. Quelles sont les personnes ayant accès à une information et à quelle fin peuvent-elles l'exploiter? Qu'on le veuille ou non, nous sommes soumis tous les jours, dans nos actions les plus bénignes, à des mécanismes de contrôle destinés à vérifier qui nous sommes et quels sont nos droits ainsi que nos responsabilités. Une carte de membre d'une bibliothèque définit quels sont les livres que l'on peut emprunter et dans quelles conditions, notre carte de banque permet de retirer de l'argent à un distributeur, l'obtention d'un permis est nécessaire pour conduire un véhicule, la liste rouge des numéros de téléphone définit quels sont les numéros auxquels les abonnés de "bases" ne peuvent avoir accès. Autant d'exemples qui illustrent, à leur façon, un moyen de mettre en œuvre ces contrôles.

Dans le jargon informatique, la vérification des prérogatives des individus dans un système est souvent référencée par le terme "autorisation". L'autorisation, d'une manière générale vise à restreindre l'accès à une ressource spécifique à une catégorie de personnes définie.

L'objectif de cet exposé est dans un premier temps de présenter les concepts phares utilisés dans toute architecture destinée à effectuer un contrôle d'accès.

Ensuite nous nous intéresserons plus particulièrement aux moyens de transmettre ces informations relatives au contrôle entre les différentes parties impliquées. La démultiplication des interactions entre les applications et les architectures distribuées posent le problème du transport de ces informations. Celles-ci étant au cœur du problème, il est nécessaire de prendre quelques précautions afin d'assurer la pérennité du système. A cette fin, nous analyserons les certificats d'attribut X.509 ainsi que les assertions SAML.

La troisième partie présentera différentes recherches qui ont pour objectif de développer une architecture d'autorisation distribuée.

La quatrième partie s'intéressera quant à elle aux solutions commerciales disponibles. Nous verrons que les différentes offres présentent beaucoup de similitude et nous analyserons ensuite l'une d'entre-elles en détail.

La dernière partie s'intéresse à l'utilisation potentielle de SAML dans un cas concret au sein de l'entreprise ISABEL.

2.L'autorisation

Les différents éléments intervenants dans l'autorisation ont pour objectif de réguler l'utilisation, l'accès à certaines ressources. Dans cette situation le terme autorisation est fréquemment employé et peut revêtir une signification différente selon le contexte dans lequel il est employé [MLORCH] :

- L'habilitation : c'est-à-dire l'attribution d'un droit d'accès ou d'utilisation, plus généralement d'un privilège à une certaine personne.
- Le privilège lui-même/ l'attestation (proof of right) : c'est-à-dire la preuve de possession d'un droit (par exemple une carte d'accès à un local).
- La vérification / validation de privilège(s) en vue d'attribuer un droit, d'émettre une décision.

Ce chapitre présente les principes généraux d'un système d'autorisation, les différents acteurs concernés, leurs relations ainsi que les dispositifs permettant ces relations.

2.1. Notions de base

Avant de développer plus en avant, voici une brève définition des termes couramment employés dans la littérature.

2.1.1. Les acteurs

Les acteurs sont les entités directement concernées par l'autorisation. Ce sont elles qui définissent ou subissent les conséquences d'une autorisation.

Les sujets

Un sujet est une personne ou un processus avec une certaine identité pouvant demander, recevoir, posséder, présenter ou déléguer un privilège afin d'effectuer une action sur une ressource.

Les ressources

Une ressource est un composant du système qui fournit un ou plusieurs services.

LES AUTORITÉS

Les autorités sont responsables de la gestion des privilèges ainsi que des politiques d'utilisation des ressources. Elles sont également chargées de vérifier les informations concernant le sujet et la ressource et d'émettre le cas échéant une autorisation ou un refus en fonction de ces informations.

2.1.2. Privilège

Un privilège est une information qui caractérise une entité, qu'elle soit physique ou morale, et qui est utilisée comme critère pour accorder ou non l'accès à une ressource. Les privilèges peuvent être considérés comme la liste des choses qu'une personne est autorisée à faire.

2.1.3. Attribut

L'autorisation n'a d'intérêt que si certaines ressources doivent être protégées. En effet, si tout le monde a accès à toutes les ressources, il n'est pas nécessaire de mettre en place un mécanisme d'autorisation. Afin d'établir quelles sont ces ressources ainsi que les utilisateurs pouvant y accéder, il est nécessaire de pouvoir les différencier.

Un attribut est une information qui caractérise l'objet auquel il se rapporte. Par exemple un attribut peut être associé à un utilisateur ou une ressource. Un attribut peut être représenté par un couple <type, valeur>, par exemple <profession, boucher> ou encore <degré de clearance, confidentiel>.

2.1.4. Politique

Une politique est un ensemble de règles régissant un domaine. Dans le cadre de l'autorisation, la notion de règle peut être précisée comme suit [CARL] :

Typically a "rule" has an effect (indicating whether it is intended to contribute to a PERMIT decision or a DENY decision), a scope or a target of applicability (indicating the subject, resource, and action to which it applies), and a condition or set of conditions (indicating any restrictions, limitations, or qualifications to be imposed upon this subject being permitted or denied access to this resource).

Une politique d'accès établit une relation entre des sujets et des ressources sur base de leurs attributs spécifiques.

2.2. Modèles de communication

Afin d'autoriser l'accès à une ressource, les trois acteurs du système peuvent communiquer de différentes manières. Celles-ci peuvent se décliner selon trois modèles généraux définis dans le RFC 2904.

2.2.1. Le modèle Agent

Dans le modèle agent, l'autorité d'autorisation (AA) sert d'intermédiaire entre le sujet et le service proprement dit. Elle se charge de vérifier les droits d'accès et de transmettre, le cas échéant, la requête à la ressource. Elle informe ensuite le sujet du succès ou de l'échec de sa requête. Par exemple une demande de permis de bâtir auprès de l'administration.

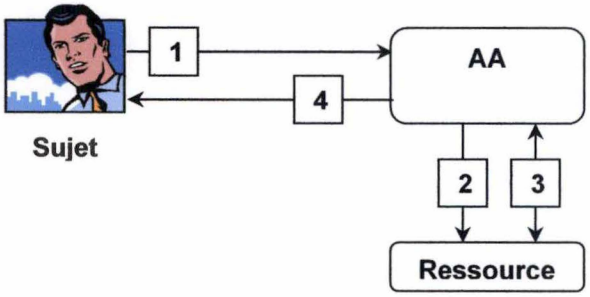


Figure 1: modèle agent

2.2.2. Le modèle Pull

Le modèle pull impose à la ressource de s'enquérir des informations d'autorisations concernant l'utilisateur auprès de l'autorité adéquate, sur base desquelles elle accepte ou refuse la requête du client. Une secrétaire qui demande à son patron s'il désire recevoir monsieur Dubois.

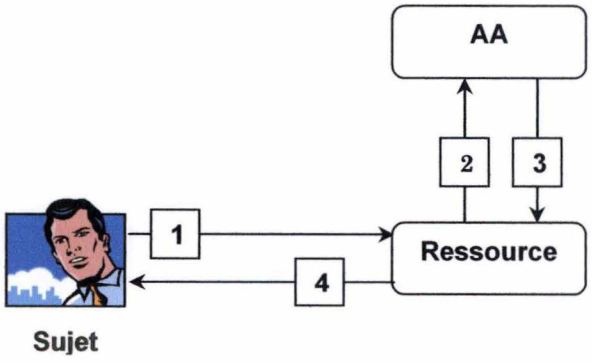


Figure 2: modèle pull

2.2.3. *Le modèle Push*

Dans le modèle push, l'utilisateur doit obtenir préalablement les informations d'autorisation afin de soumettre ensuite à la ressource. Les cartes de membres, les cartes d'accès, mots de passe sont autant de mécanismes utilisant ce modèle.

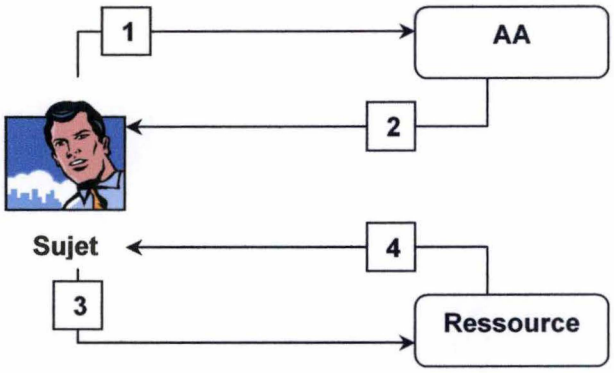


Figure 3: modèle push

La viabilité de tous ces modèles dépend de la confiance que peut avoir chaque acteur dans les informations qui lui sont fournies. Le système devra donc mettre en place des mécanismes permettant de vérifier la fiabilité des informations qui lui sont présentées. Notamment si celles-ci émanent bien du service d'autorisation compétent (authenticité) et si elles n'ont pas subi de modification durant le transport (intégrité).

2.3. Architecture générale

Il est possible de scinder le processus de décision en plusieurs actions élémentaires: obtention des politiques, des attributs, prise de décision, etc. L'exécution de ces différentes actions peut être déléguée à plusieurs composants. Cette section présente une architecture générale rémanente au sein des différentes solutions d'autorisation. Celle-ci est présentée dans le RFC 2904 et est quelque peu simplifiée afin d'en faciliter la compréhension.

2.3.1. *Composants*

Les différents composants intervenants dans l'autorisation, déterminés par le RFC 2904, sont les suivants :

- Le **PDP** (Policy Decision Point[1]) est chargé de vérifier si le sujet possède les droits requis et satisfait aux conditions de

[1] Egalement appelé AEF (Access Control Enforcement Function)

la(des) politique(s) d'accès, on parle également d'évaluer la politique d'accès.

- Le **PRP** (Policy Retrieval Point) procure la (les) politiques d'accès concernant la ressource concernée.
- Le **PIP** (Policy Information Point) fournit les informations nécessaires afin d'effectuer l'évaluation. Carlisle Adams [CARL] distingue 3 sources d'informations : les *subjects authorities*, *ressources authority* et *environmental authorities*. La première concerne des informations concernant le sujet que le PDP doit aller chercher (mode pull). La deuxième concerne la ressource elle-même, par exemple dans le cas d'une réservation de bande passante, la bande passante disponible. La dernière concerne les variables environnementales.
- Le **PEP** (Policy Enforcement Point[2]) est chargé d'appliquer la décision remise par le PDP.
- Les **autorités** sont responsables de la gestion des informations nécessaires au système (politique, attributs)

On peut comparer ces différents composant à un système juridique et aux trois pouvoirs qui y sont associé. Les autorités sont le pouvoir législatif, ce sont elles qui définissent les règles. Le PDP est le pouvoir judiciaire tandis que le PEP est l'exécutif.

[2] Egalement appelé ADF (Access Control Decision Function)

2.3.2. *Principe de fonctionnement*

La figure suivante représente le flux d'informations entre les différents composants en mode pull[3].

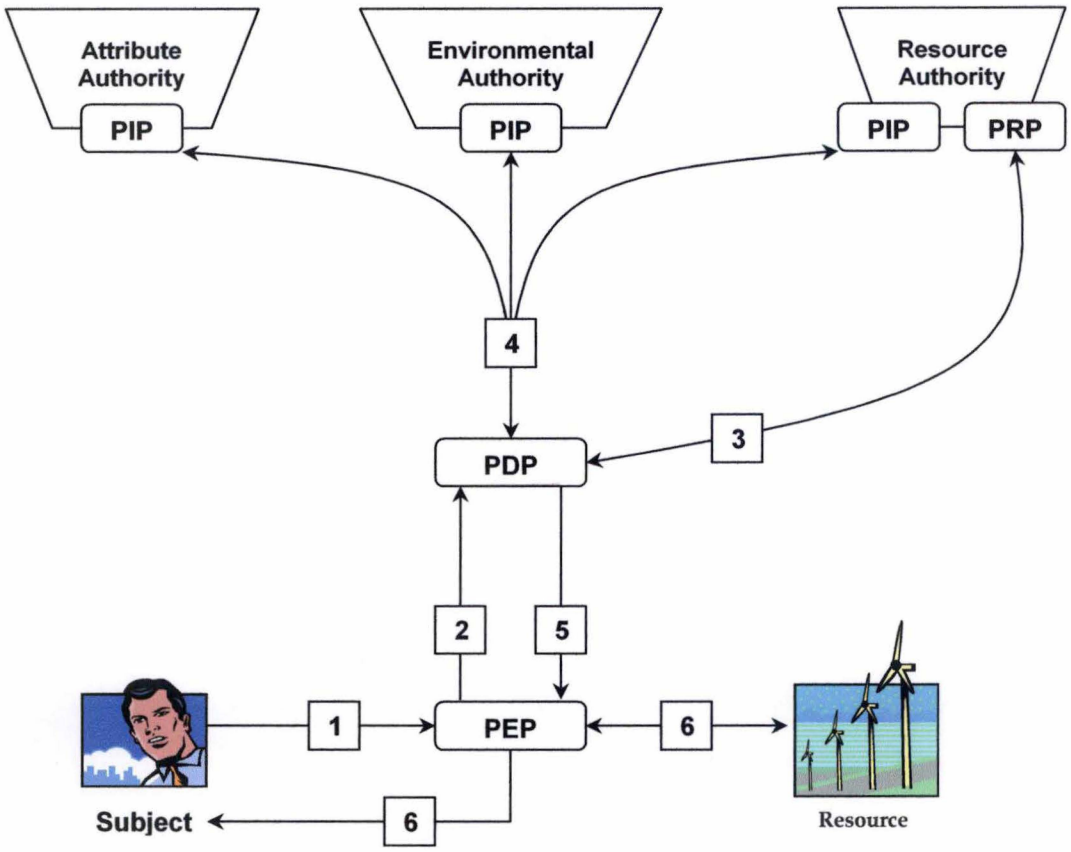


Figure 4: Flux d'informations en mode pull

1. Le PEP intercepte la requête de l'utilisateur.
2. Il soumet cette requête au PDP ainsi que les informations qu'il possède concernant l'utilisateur (certificat, login, attributs éventuels, heure de la requête...)
3. Le PDP récupère la politique d'autorisation grâce au PRP de l'autorité responsable de la ressource (policy authority). Cette autorité est responsable de la définition de la politique d'accès ainsi que des attributs de cette ressource.
4. Le PDP obtient les informations supplémentaires nécessaires à l'évaluation de la politique auprès des PIPs. L'attribute authority est responsable des attributs du sujet requérant l'accès à la ressource. La combinaison de ces attributs et de

[3] Certains auteurs font une distinction en fonction de l'entité effectuant le retrait d'information au lieu du comportement du serveur (PEP), on parle alors de « user pull » ou « server pull » au lieu de mode push ou pull.

la politique d'accès permet d'établir les privilèges du sujet afin de prendre une décision.

5. Le PDP renvoie sa réponse (accept, deny, unable to evaluate) au PEP.
6. Sur base de la réponse fournie par le PDP le PEP autorise ou refuse l'accès à l'utilisateur et met en place les dispositifs nécessaires.

Il s'agit d'un modèle simplifié ne faisant intervenir qu'une seule entité de chaque type. Il est bien entendu possible de mettre en place des architectures plus complexes faisant intervenir par exemple plusieurs PDP dans le processus d'évaluation.

2.3.3. Exigences

L'autorisation étant destinée à la sécurité, il est nécessaire de prendre certaines dispositions afin d'assurer son bon fonctionnement[4].

- Il est impératif d'assurer **l'intégrité** des données échangées entre les différents composants.
- Afin d'établir **l'authenticité** des informations qui lui sont présentée, chaque composant doit pouvoir identifier l'origine de ces informations.
- Le cas échéant, la **non répudiation** des informations échangées peut également avoir de l'importance. Dans un contexte de délégation, les relations entre les différentes parties sont régies par contrat. Afin d'établir les responsabilités des différentes parties lors d'un litige, la non répudiation peut conférer un statut de preuve à des informations électroniques.

2.4. Management de l'autorisation

La section précédente expliquait le processus de vérification : les différents acteurs intervenant dans cette opération ainsi que les données nécessaires à l'émission d'une décision d'autorisation ou de refus. Cette section s'intéresse à la gestion des informations sur lesquelles se basent les différents acteurs du système. C'est là un aspect essentiel du système. En effet il est inutile de mettre en place un processus de décision sécurisé si par exemple chaque utilisateur du système peut définir ses propres droits. La gestion des différents attributs est elle-même contrainte par un mécanisme d'autorisation

[4] Voir également gestion de la confiance 2.4.1.

spécifiant par exemple quelles sont les personnes pouvant définir une politique.

Le PIP et le PRP définis précédemment permettent de classer ces informations parmi deux catégories : les attributs et les politiques. De ce fait, leur gestion respective peut également être déléguée à divers composants du système.

2.4.1. Gestion de la confiance

Afin d'établir quelles sont les responsabilités au sein de l'autorisation on peut se poser les questions suivantes :

- Quelles sont les ressources concernées par l'autorisation ?
- Quelles sont les opérations, actions que l'on peut effectuer sur ces ressources ?
- Quelles sont les personnes responsables de l'attribution de privilèges (permissions) ainsi que leur champ d'application ? Quelles ressources ou ensemble de ressource, quels utilisateurs et quels privilèges ces personnes peuvent-elles contrôler.
- Quelles sont les personnes responsables de la révocation de privilèges ainsi que leur champ d'application ?

[MLORCH] introduit la notion d'autorité afin de répartir les responsabilités. Une autorité est une entité administrative autoritaire pour l'émission, la validation et la révocation d'informations électroniques. Par exemple une Autorité de Certification (CA[5]) est responsable des certificats à clé publique utilisés pour les signatures numériques ainsi que pour une authentification forte.

L'objectif de la gestion de la confiance est double, d'une part la définition des différentes autorités et de leurs champs d'application respectifs. D'autre part la spécification des mécanismes permettant d'établir la relation de confiance entre deux entités du système. Par exemple la gestion de confiance peut se baser sur une infrastructure à clé publique, ainsi la confiance peut être établie au moyen d'informations signées ou de connexions SSL entre les parties. Il leur suffit de savoir quelles sont les responsabilités associées aux différentes clés.

[5] Certification Authority en anglais, à ne pas confondre avec AC qui est l'abréviation de certificat d'attribut (attribute certificate). De plus les abréviations françaises sont l'inverse des anglaise, certificat d'attribut et autorité de certification. Comme la majeure partie de la littérature disponible est en anglais nous utiliserons ici les abréviation anglaise.

2.4.2. *Gestion des attributs*

Dans une architecture d'autorisation un attribut peut être associé à un sujet, une action, une ressource ou à l'environnement [CARL]. Il peut coexister de multiples autorités responsables d'attributs, différents ou non. Il est nécessaire de spécifier quels sont les attributs ainsi que les sujets, ressources ou actions, pour lesquels chaque autorité est responsable. Cette distinction peut être approfondie en spécifiant quelles sont les valeurs d'attributs pour lesquels chaque autorité est responsable.

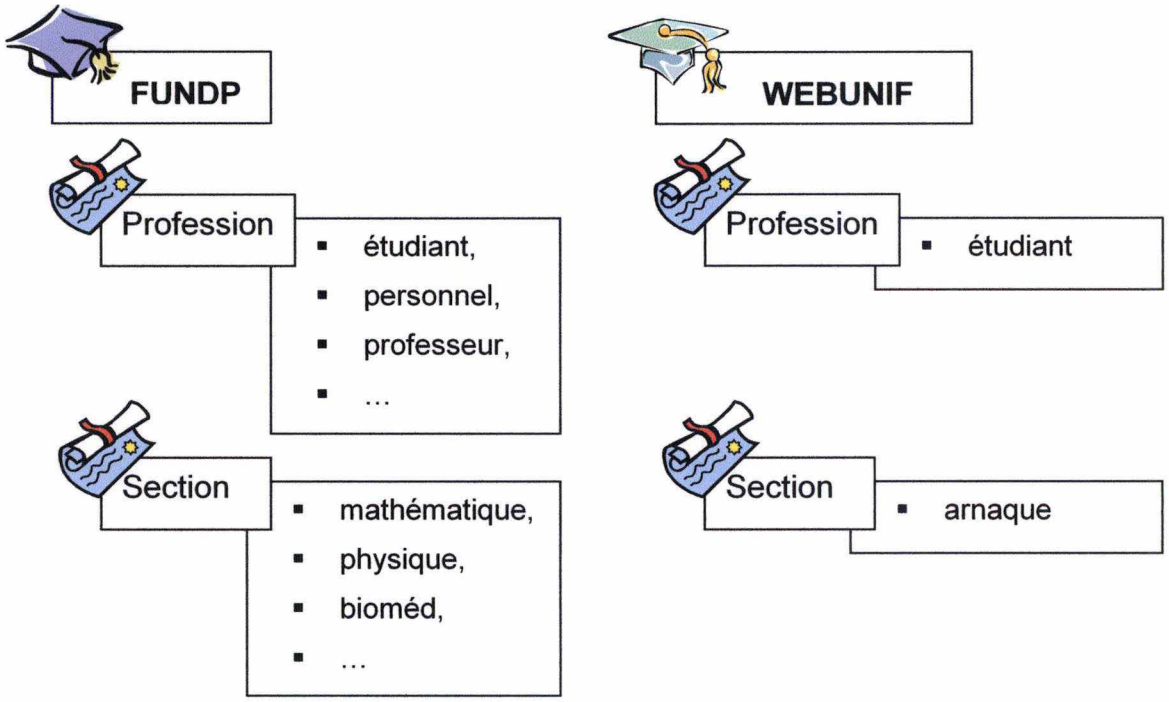


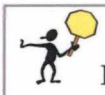
Figure 5: Gestion d'attributs

Dans cet exemple, l'université FUNDP est responsable des attributs <profession> et <section> et peut attribuer les valeurs spécifiées. L'université WEBUNIF est également responsable pour ces attributs mais ne peut délivrer que les valeurs <étudiant> pour le premier et <arnaque> pour le second.

Si par exemple WEBUNIF émettait un attribut <Section, mathématique> pour un étudiant, celui-ci ne devrait pas être accepté.

Cependant la validité d'un attribut peut être plus complexe. Par exemple, un attribut section peut être conditionné par la possession d'un attribut profession émis par la même université. FUNDP ne

pouvant donc émettre d'attribut section que pour des personnes qu'elle a reconnues. Un attribut <section, mathématique> émis par FUNDP pour un étudiant de WEBUNIF[6] doit donc être considéré comme non valide.



Il est impératif que les acteurs du système définissent une sémantique commune concernant les différents attributs utilisés.

2.4.3. Politique d'accès

La gestion des politiques d'accès peut être indépendante de la gestion des attributs. Par exemple, le fait d'accorder la nationalité belge à une personne octroie indirectement des droits qui ne sont pas pris en compte pour accorder la nationalité. De même certaines règles peuvent être érigées sans prendre en compte tous les sujets potentiels sur lesquels elle s'appliquera. Il s'agit là d'un autre aspect administratif de l'autorisation.

Il est du ressort des autorités de politique de définir quelles sont les actions accordées par un attribut. Ces autorités, comme les autorités d'attributs, peuvent être raffinées selon différents critères. Tout d'abord selon le type de ressource pour lesquelles elles sont responsables. Ensuite elles peuvent être classées selon les actions pour lesquelles elles sont responsables. Voici un exemple [CARL]:

> *For example, in a particular company or organization, there may be regulatory policies that govern access to certain types of data, legislative policies regarding the release of the same data, and corporate and even departmental policies regarding access to the same data.*

[6] Dans ce cas, le fait d'être étudiant de l'une ou l'autre des universités est implicite et dérivé par l'émetteur de l'attribut profession.

3. Communication des informations

Cette partie traite des moyens de communication entre différents acteurs permettant d'assurer les propriétés **d'authenticité**, **d'intégrité** et de **non répudiation** des informations échangées.

3.1. Technologies

3.1.1. Cryptographie asymétrique (PKC)

La cryptographie asymétrique se base sur le problème de factorisation de grands nombres et la calculabilité de ce problème.

L'intérêt de cette méthode est qu'elle permet de créer une paire de clés de telle sorte que les données encodées par l'une ne sont déchiffrables que par l'autre. Supposons une paire de clé <A, B>, seul le propriétaire de la clé A (associées mathématiquement à B) est capable de déchiffrer un message encodé par B. Si le propriétaire de la paire de clés diffuse la clé B, tout le monde peut dès lors le contacter avec la clé B tout en s'assurant qu'aucune autre personne n'aura accès à l'information. La clé diffusée est appelée clé publique tandis que l'autre est appelée clé privée.

Attention de ne pas confondre clé privée et clé secrète, la clé secrète est utilisée en cryptologie symétrique et doit être connue de part et d'autre afin de pouvoir être utilisée. Afin d'être fiable, cette dernière doit être communiquée aux parties par un canal sûr (out-of-band).

Les paires de clés peuvent être utilisées notamment pour:

- **Signer un message**: lorsqu'une personne envoie un message signé, il utilise sa clé privée. Le destinataire pourra lire son message qu'il possède ou non lui-même une clé. Le fait de signer un message identifie uniquement l'auteur. En effet, ce message contient son certificat et sa clé publique. Les données envoyées sont en clair et accessibles à tous.
- **Chiffrer un message**: lorsque quelqu'un envoie un message crypté, le logiciel utilise la clé publique du destinataire. Celui-ci reçoit le message chiffré au moyen de sa clé publique et utilisera

alors sa clé privée pour déchiffrer le message. L'identité de l'émetteur n'est pas vérifiable par le destinataire.

- **Crypter et signer un message**
- **Négocier une clé privée:** la cryptographie asymétrique est également utilisée afin de négocier une clé secrète (cryptologie symétrique). En effet, le chiffrement avec cryptologie asymétrique est plus lourd et plus est limitatif en ce qui concerne la taille maximale des messages chiffrables, de ce fait l'utilisation d'une clé symétrique est plus efficace. Les paires de clé permettent de mettre en place un environnement sécurisé grâce auquel les parties pourront s'échanger une clé secrète en toute sécurité.

Les tableaux suivant sont fournis à titre indicatif afin de se faire une idée de la résistance des clés face à des attaques massives (brute force)

Tableau 1: Temps moyen pour une attaque massive en 1995 en fonction du matériel (source : Applied Cryptography 2nd edition)

Coût	40 bits	56 bits	64 bits	80 bits	112 bits	128 bits
100e3$	2s	35h	365j	70000 ans	10e14 ans	10e19 ans
1e6$	0.2s	3.5h	37j	7000 ans	10e13 ans	10e18 ans
10e6$	0.02s	21m	4j	700 ans	10e12 ans	10e17 ans
100e9$	2e-6s	0.1s	32s	24j	10e8 ans	10e13 ans

Tableau 2: Comparaison de tailles de clés symétriques et asymétrique face à une attaque massive

Cryptographie symétrique	Cryptographie asymétrique (longueur de clé publique)
56 bits	384 bits
64 bits	512 bits
80 bits	768 bits
112 bits	1792 bits
128 bits	2304 bits

Tableau 3: Recommandations de taille de clés publiques (source : Applied Cryptography 2nd edition)

Année	Utilisation privée	Utilisation commerciale	Utilisation gouvernementale
1995	768	1280	1536
2000	1024	1280	1536
2005	1280	1536	2048
2010	1280	1536	2048
2015	1536	2048	2048

Actuellement la taille des clés commerciales est de 1024 bits tandis que la taille des clés des organismes émettant les certificats est de 2048.

3.1.2. Les certificats

Un certificat est un document électronique attestant qu'une clé publique est bien liée à une organisation ou une personne. Il permet la vérification de la possession d'une clé publique afin d'en prévenir la contrefaçon. Un certificat typique contient généralement les informations suivantes :

- Une clé publique.
- Le nom et l'adresse de courrier électronique du propriétaire.
- La date d'expiration de la clé publique.
- Le nom de la compagnie ayant émis le certificat (Autorité de Certification AC).
- Le numéro de série du certificat.
- La signature numérique de l'Autorité de Certification.

Le certificat est donc une attestation assurant que les informations qu'il contient sont exactes. Pour cela, le certificat doit être généré par un tiers de confiance, c'est-à-dire un organisme indépendant qui contrôle la véracité de ces informations. Ce tiers est appelé autorité de certification (certification authority ou CA) et donne la crédibilité au certificat.

Il existe plusieurs types de certificats et ceux-ci sont classés selon leur niveau de sécurité:

- **Class 1:** Les certificats de classe 1 appelés également "individual Certificates" ou "Digital IDs" sont ceux de plus bas niveau. Ils certifient simplement que l'adresse mentionnée est celle du propriétaire de la clé publique. Le degré de confiance que l'on peut porter à ces certificats est bien entendu très faible. Ils sont appropriés pour la signature et le chiffrement d'e-mail pour des particuliers.
- **Class 2:** Les certificats de classe 2 sont des certificats individuels qui offrent un degré de sécurité supplémentaire. En plus du certificat de classe 1, la procédure de validation de ces certificats inclut la comparaison d'informations fournie par le requérant avec d'autres sources, par exemple une copie signée de la carte d'identité. Ces certificats peuvent être utilisés pour signer, chiffrer et comme preuve d'identité pour des transactions de faible valeur.

- **Class 3**: Les certificats de classe 3 offrent le plus haut niveau de sécurité. L'obtention d'un de ces certificats passe par une vérification physique de l'identité de la personne. Celle-ci doit se présenter, muni d'une pièce d'identité, auprès du service qui confirmera ainsi les informations fournies. De tels certificats peuvent également être délivrés à des organisations (certificats de serveur, utilisé notamment pour SSL). Dans ce cas la vérification se fait sur les droits de l'organisation à utiliser le nom de domaine figurant sur le certificat.

3.2. Certificat d'attribut X.509

3.2.1. Définition

Conceptuellement un certificat d'attribut est similaire à un certificat de clé publique dans le sens où chacun lie une information à une identité. La différence principale réside dans la nature de l'information, ce qui influence l'utilisation qui est faite du certificat.

Un certificat de clé publique est employé afin de valider l'intégrité de l'association entre une clé publique et une identité. Il en découle une utilisation à des fins d'authentification, de non répudiation, de confidentialité ou encore d'autorisation (à faible granularité ou coarse grained). Ceci est garanti par les propriétés cryptographiques de la paire de clé qui assurent que seule la clé mathématiquement associée peut déchiffrer les données encodées avec l'autre clé[7], l'une étant diffusée (clé publique) et l'autre gardée secrète (clé privée). L'utilisation de ces deux clés permet donc la confidentialité et l'intégrité grâce au concept de signature, mais pas l'identification. C'est là le rôle du certificat de clé publique, il atteste que l'entité possédant la clé privée, associée à la clé publique du certificat, est celle spécifiée dans le certificat. Dans une infrastructure à clé publique (PKI) des autorités sont désignées comme responsables de l'émission des certificats de clé publique et signent ceux-ci au moyen de leur propre clé privée.

Un certificat d'attribut (AC) est plus générique, il atteste simplement que l'entité mentionnée dans le certificat possède bien l'(les)attribut(s) spécifié(s). Les certificats à clé publique sont donc une spécialisation des certificats d'attribut.

Les ACs ont été introduits dans un but d'autorisation afin de pallier aux manques des certificats de clé publique et de permettre la mise en place de politiques d'autorisation à granularité plus fine (fine-grained). Il est possible d'associer des attributs directement aux certificats de clé publique via des extensions[8] mais cette option présente deux inconvénients majeurs :

- La période de validité des attributs d'une personne est généralement plus courte que celle de la clé publique. La suppression ou la modification d'un privilège d'une personne impliquerait donc d'invalider le certificat de clé publique

[7] Cryptographie asymétrique.

[8] Les extensions font partie intégrante du certificat et ne sont pas des ajouts diffusés séparément, il s'agit d'un terme utilisé afin de décrire des informations complémentaires qui ne sont pas en rapport direct avec le rôle initial du certificat à savoir l'association d'une clé publique à une identité.

correspondant (de le révoquer) et d'émettre un nouveau certificat avec les attributs mis à jour.

- La gestion des privilèges d'une personne peut dépendre d'une ou plusieurs institutions différentes de la CA.

3.2.2. Applications à l'autorisation

Un certificat d'attribut est une structure de données garantissant les propriétés d'intégrité, de non répudiation et optionnellement la confidentialité des ses attributs. Ces propriétés permettent d'établir le crédit que l'ont peu accorder à un AC. La non répudiation permet d'identifier l'émetteur du certificat tandis que l'intégrité assure la fiabilité de celui-ci. Les ACs peuvent alors servir de support à la gestion de confiance et permettent de soutenir la mise en place des différentes autorités d'un système.

TRANSPORT D'INFORMATIONS

Les ACs peuvent servir au transport d'informations relatives à l'autorisation dans un contexte non sécurisé :

- les attributs concernant un sujet ;
- une politique d'accès à une ressource ;
- des variables environnementales dont les valeurs sont critiques pour la prise de décision ;
- l'autorisation proprement dite (e.g. : « Jean est autorisé à créditer le compte 111-2222222-33 »).

Ce qui est particulièrement utile dans une séquence push où la source émettrice de l'information ne peut pas être identifiée directement par le récepteur.

Idéalement la période de validité de tels certificats devrait être limitée afin de ne pas devoir gérer leur révocation.

GESTION DES INFORMATIONS

Etant donné que les ACs sont utilisés pour le transport d'informations relatives à l'autorisation, il est naturel d'envisager d'employer ceux-ci dans la gestion de ces informations et d'exprimer celles-ci directement sous la forme d'ACs.

L'attribution d'un privilège ou d'un attribut à une personne consiste alors à créer et à publier un AC. Dans ce cas, l'AC peut également contenir des informations complémentaires nécessaires à sa gestion. Notamment des informations concernant la révocation des certificats si ceux-ci ont une période

de validité relativement grande ou encore des informations relatives à la délégation[9].

3.2.3. Format d'un certificat d'attribut

Un certificat d'attribut contient les champs suivants [RFC 3281] [X-509] :

Tableau 4: Format d'un certificat d'attribut

Attribute Certificate
`acinfo` AttributeCertificateInfo
`signatureAlgorithm` AlgorithmIdentifier
`signatureValue` BIT STRING

Tableau 5: Informations d'un certificat d'attribut

AttributeCertificateInfo
`version` AttCertVersion -- version is v2
`holder` Holder
`issuer` AttCertIssuer
`Signature` AlgorithmIdentifier
`serialNumber` CertificateSerialNumber
`attrCertValidityPeriod` AttCertValidityPeriod
`attributes` SEQUENCE OF Attribute
`issuerUniqueID` UniqueIdentifier OPTIONAL
`Extensions` Extensions OPTIONAL

LE CHAMP HOLDER

Le champ *holder* lie les attributs du certificat à un propriétaire appelé porteur du certificat. Ce porteur peut être une personne, une application ou encore une ressource. Le RFC 3281 recommande de n'utiliser qu'un seul des éléments suivants afin d'identifier le porteur du certificat, leur utilisation simultanée pouvant prêter à confusion :

- *baseCertificateID* : l'émetteur ainsi que le numéro de série du certificat de clé publique du porteur ;
- *entityName* : nom identifiant l'entité (DN, URL, adresse IP...) ;

[9] La gestion des privilèges est également soumise à des impératifs d'autorisation. En effet, elle effectue des actions sur des ressources (attribuer, supprimer un privilège particulier). La gestion des privilèges peut donc être sous-tendue par une autre architecture d'autorisation concernant ces privilèges, ce qui est typiquement la fonction des informations de délégation.

- *objectDigestInfo* : référence d'une information permettant d'identifier le porteur (clé publique, certificat,...).

L'utilisation de la première méthode permet de lier le certificat d'attribut à un certificat de clé publique et par conséquent à une identité. Cette méthode pose problème lors de la révocation du certificat de clé publique. Il faut en effet définir si les certificats d'attribut restent valides ou non lorsque le certificat de clé du porteur est révoqué. Un comportement sain consiste à considérer que les ACs sont également invalidés. Cependant si la révocation est due à une expiration du certificat de clé et que celui-ci est renouvelé[10], il n'est peut-être pas nécessaire d'invalider tous les AC. Ce problème résulte du fait que l'AC est lié à un certificat de clé publique et non à une identité ou une clé.

La deuxième option, utilisée dans un contexte de PKI, permet de lier le certificat au *Distinguished Name*[11] *(DN)* du porteur. Potentiellement ce DN est identique pour tous les certificats concernant une même entité[12] (chez un CA donné), il suffit donc qu'un certificat de clé avec ce DN soit valide afin d'assurer la conformité des ACs. Cette option présente néanmoins certains inconvénients, notamment si plusieurs CAs sont responsables de l'émission de certificat de clé publique dans le système, en effet l'unicité du DN n'est assurée qu'au sein de chaque CA et un DN pourrait alors concerner des personnes différentes. De plus si le porteur possède plusieurs clés, chez le même CA, utilisées à des fins différentes il n'est pas possible de les différencier.

La troisième option peut s'avérer utile pour référencer des applications ou des objets. Dans son projet d'infrastructure de gestion des privilège, la Direction Centrale de la Sécurité des Systèmes d'Information conseille d'utiliser ce champ et d'y introduire la signature du certificat de clé publique (qui ne peut être faite que par la CA ayant émis le certificat de clé publique) afin de référencer de manière sûre le certificat du porteur. Pourtant le champ *baseCertificateID* suffit pour référencer un certificat. Il s'agit du numéro de série du certificat de clé et du nom de la CA émettrice, de ce fait il identifie un certificat

[10] A strictement parler le certificat n'est pas renouvelé, c'est la validité de la clé publique qui est étendue et un nouveau certificat concernant la même clé est émis.

[11] Le *Distinguished Name* est une structure de données pouvant contenir les champs suivants : CN : CommonName, L : LocalityName, ST : StateOrProvinceName, O : OrganizationName, OU : OrganizationalUnitName, C : CountryName, STREET : StreetAddress. Le DN est notamment utilisé dans les PKC afin d'identifier la personne possédant un certificat.

[12] La contrainte d'unicité d'un DN imposée aux CAs concerne uniquement les individus et non la clé. Concrètement un DN peut être associé à plusieurs clés publiques mais la personne possédant les clés privées associées est la même.

unique. Néanmoins, d'un point de vue théorique, cette approche n'est pas tout à fait dénuée de sens. En effet rien n'impose aux CAs d'avoir des noms différents. Une CA mal intentionnée pourrait essayer d'obtenir le même nom qu'une autre et forger un certificat de clé afin de voler les attributs d'une autre personne. Cette éventualité est peut probable car dans une PKI les CAs doivent respecter un code de conduite et la CA corrompue devrait se faire avaliser par une autre CA déjà présente dans l'infrastructure.

Le RFC 3281 permet également de référencer une clé au lieu d'un certificat grâce à ce champ en y introduisant un reliquat (hash) de la clé publique.

Le champ Issuer

La norme ITU-T X.509 prévoit les champs suivant pour identifier l'émetteur du certificat (AA ou Attribute Authority)

- *issuerName* : nom de l'émetteur (AA) ;
- *baseCertificateID* : l'émetteur (CA) ainsi que numéro de série du certificat de clé publique de l'AA;
- *objectDigestInfo* : reliquat d'une information identifiant l'émetteur (clé publique, certificat...).

Cependant le RFC 3281 impose de n'utiliser que l'*issuerName*. En effet, l'utilisation d'une des deux autres options lie l'AC au certificat de clé publique de l'AA et non à l'AA, par conséquent le renouvellement du certificat de clé publique[13] de l'AA ayant émis les certificats d'attribut peut engendrer des problèmes.

Les Extensions

Le RFC 3281 identifie plusieurs extensions principales[14] dont voici une brève description.

- *Audit Identity* (critique[15]) : permet au porteur d'utiliser ses attributs sans divulguer son identité au vérifieur. Cela permet au vérifieur de garder une trace de l'utilisation des attributs avec la possibilité de retrouver en cas de force majeure l'identité du porteur grâce à l'émetteur de l'AC du porteur.

---

[13] Concernant la définition des responsabilités, ITU-T X.509 permet de spécifier le champ d'application d'une AA dans son certificat de clé, cela suppose que la CA est responsable de la définition de ces responsabilités.

[14] ITU-T X.509 propose des extensions supplémentaires, notamment concernant la délégation des attributs. Ces extensions permettent une politique d'accès discrétionnaire via la création de chaîne d'AC. RFC 3281 ne prend pas en considération les chaînes d'ACs.

[15] Le caractère critique d'une extension signifie que si un vérifieur (entité cherchant à vérifier la validité d'un certificat) ne comprend pas cette extension, alors il doit refuser le certificat.

- *AC Targeting* (critique) : indique les entités qui peuvent vérifier ce certificat. Si l'entité n'est pas dans la liste, elle doit rejeter le certificat.
- *Authority Key Identifier* : identifie la clé publique de l'émetteur de l'AC. Ce champ fournit une aide au vérifieur pour retrouver le certificat de clé de l'AA émettrice. Cela permet notamment de retrouver une clé particulière si l'AA en utilise plusieurs. En effet, c'est l'AA qui est référencée par l'AC et non la clé de cet AA[16].
- *Authority Information Access* : identifie une URL pour les informations de révocation de l'AC (OCSP).
- *CRL Distribution Points* : spécifie l'URL pour récupérer l'ARL (Attribute Revocation List).
- *No Revocation Available* : Indique que pour un attribut, il n'y aura pas d'information de révocation. Cette extension implique que la période de validité de l'AC soit suffisamment courte afin que la révocation d'un tel certificat soit superflue.

Les Types d'attributs

Les attributs sont généralement spécifiques aux applications ou à une politique d'accès particulière. ITU-T X.509 définit cependant l'attribut rôle :

Tableau 6: L'attribut ROLE

Role
`roleAuthority` *GeneralNames OPTIONAL*
`roleName` GeneralName

RoleAutorithy identifie l'autorité responsable de la spécification du rôle.

Le RFC 3281 définit les types d'attribut supplémentaires suivant :

- *Service Authentication Information* : information liée à l'authentification (username et/ou password) du porteur pour un service particulier.
- *Access Identity* : information utilisée par le vérifieur pour autoriser les actions du porteur.

[16] Voir Format - Le champ Issuer p 35.

- *Charging Identity* : identifiant du responsable pour une facturation.
- *Group* : informations concernant le porteur sur l'appartenance à des groupes
- *Clearance* : contient des informations concernant le porteur, relatives aux politiques de sécurité et d'autorisations.

Le RFC 3281 prévoit également la possibilité de crypter les attributs en les encapsulant dans une EnvelopedData. Ce type de cryptage est ciblé. Avant de signer l'AC, les attributs sont cryptés pour un ensemble de destinataires prédéterminés.

3.3. SAML

SAML (Security Assertion Markup Language) est un standard[17] OASIS développé par leur Security Services Technical Committee (SSTC). SAML est une spécification XML destinée à l'échange d'informations relatives à la sécurité.

Les informations sont exprimées sous la forme d'affirmations (assertion) à propos d'un sujet, où un sujet est une entité (humain ou ordinateur) qui a une identité dans un certain domaine/système de sécurité. Les affirmations sont représentées par des constructions XML et une affirmation simple peut contenir plusieurs déclarations (statements) concernant l'authentification, l'autorisation ou les attributs que possède un sujet [SAML].

SAML définit également un protocole de requête permettant de questionner les serveurs afin d'obtenir ces informations.

SAML est conçu pour fonctionner avec les principaux protocoles disponibles : HTTP pour le Web, SMTP pour les mails, FTP pour les transferts de fichiers, ainsi que des architectures XML comme SOAP.

3.3.1. Assertions SAML

Conceptuellement une assertion est similaire à un certificat. Elle associe de manière inviolable une information à un sujet. Les assertions sont cependant un peu plus raffinées et permettent également d'assurer l'inviolabilité des informations autrement que par une signature. En effet la signature d'une assertion est facultative mais dans ce cas il est nécessaire de sécuriser le transport.

Une assertion SAML peut contenir trois types de déclaration :

- Les déclarations d'authentification (authentication statement) : affirmation relative à un acte d'authentification exécuté préalablement par le sujet.
- Les déclarations d'attribut (attribute statement) : affirmation reliant un attribut à un sujet.
- Les déclarations d'autorisation (authorization statement) : affirmation attestant de la légitimité d'une personne à effectuer une certaine action sur une ressource.

[17] La version 1.1 a été ratifiée le 22 septembre 2003.

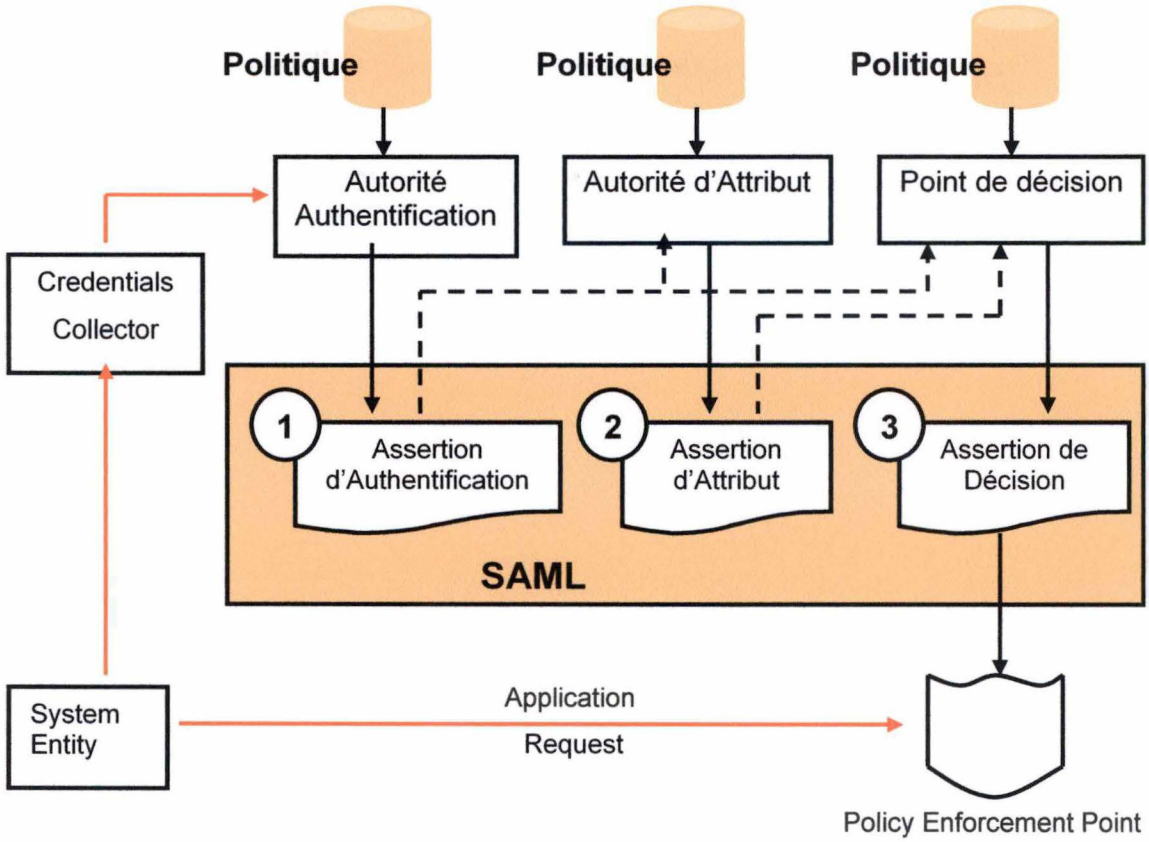


Figure 6: Assertion SAML

Les assertions contenant une ou plusieurs déclarations d'authentification, d'attribut ou d'autorisation sont respectivement appelées assertion d'authentification, assertion d'attribut et assertion d'autorisation.

La notion d'autorité est également présente dans SAML. Les assertions sont publiées par des autorités SAML qui peuvent être classée selon le type d'assertion pour lequel elles sont responsables :

- les autorités d'authentification ;
- les autorités d'attribut ;
- les PDPs (policy decision point).

Attention, bien que SAML définisse des autorités SAML, il ne définit que le format des messages ainsi qu'un protocole de requête. Il ne précise pas les méthodes de gestion des informations relatives aux assertions pour lesquelles chaque autorité SAML est responsable.

Tableau 7: Format d'une assertion SAML

Assertion	
MajorVersion [Required]	Version actuelle 1
MinorVersion [Required]	Version actuelle 1
AssertionID [Required]	
Issuer [Required]	L'autorité SAML qui a créé cette assertion.
IssueInstant [Required]	
<Conditions> [Optional]	Conditions devant être prises en compte pour valider cette assertion (par exemple ciblage des domaines où cette assertion peut être utilisée).
<Advice> [Optional]	
<ds:Signature> [Optional]	Signature de l'assertion.
<Statement> [One or More] <SubjectStatement> <AuthenticationStatement> <AuthorizationDecisionStatement> <AttributeStatement>	Les différents types de déclaration que peut contenir une assertion.

3.3.2. Déclaration de sujet (subject statement)

La déclaration du sujet précise les informations relatives au sujet concerné par l'assertion.

Tableau 8: SAML subject statement

Subject Statement	
<NameIdentifier>	Un identifiant du sujet ainsi que de son domaine.
<SubjectConfirmation>	Autres informations permettant d'identifier le sujet

3.3.3. Déclaration d'authentification (authentication statement)

Une déclaration d'authentification spécifie que le sujet déclaré a été identifié par une méthode particulière à un moment donné.

Ce type sert principalement au Single Sign On (SSO) qui permet à un utilisateur de changer de domaine sans devoir recommencer son authentification. Cela suppose une collaboration entre les différents domaines afin d'établir les liens entre l'identité d'un système et celle d'un autre ainsi que les niveaux d'authentification minimaux requis pour chaque système.

Tableau 9: SAML authentication statement

Authentication Statement	
AuthenticationMethod [Required]	Référence vers la méthode d'authentification utilisée.
AuthenticationInstant [Required]	Indique le moment auquel l'authentification a eu lieu.
<SubjectLocality> *[Optional]*	Spécifie le domaine DNS et l'adresse IP de l'entité ayant identifié le sujet.
<AuthorityBinding> *[Any Number]*	Indique que des informations additionnelles concernant le sujet sont disponibles.

3.3.4. Déclaration d'attribut

Une déclaration d'attribut spécifie les attributs que possède le sujet déclaré.

Tableau 10: SAML attrribute statement

Attribute Statement	
Attribute [One or More]	
AttributeNamespace [Required]	L'espace de nom dans lequel attributeName est interprété.
AttributeName [Required]	Nom de l'attribut.
AttributeValue *[Any Number]*	Valeur de l'attribut.

3.3.5. Déclaration d'autorisation (authorization statement)

Une déclaration d'autorisation précise si le sujet désigné est autorisé à faire l'action spécifiée sur la ressource indiquée. Plus précisément, une telle assertion indique la décision de l'autorité compétente suite à une demande d'autorisation.

Tableau 11: SAML authorization statement

Authorization Statement	
Resource [Required]	Référence identifiant la ressource
Decision [Required]	La décision rendue par l'autorité SAML concernant la ressource spécifiée.
<Action> [One or more]	L'ensemble des actions autorisées (ou refusées) pour le sujet sur la ressource spécifiée.
<Evidence> *[Optional]*	Un ensemble d'assertions sur lesquelles l'autorité s'est basée pour émettre sa décision

3.3.6. Spécification OASIS concernant l'utilisation de SAML pour effectuer le SSO

Le single sign-on consiste à propager l'authentification d'un utilisateur au travers un ensemble d'associés. Ceci afin de faciliter l'accès à l'utilisateur. De cette manière, celui-ci peut passer d'un domaine à un autre sans avoir à réitérer l'étape d'authentification.

La transmission des informations d'autorisation peut se faire de différentes manières : dans l'URL ou dans un formulaire. Les navigateurs commerciaux imposant des limitations sur la taille des URLs ne permettent pas de transmettre de « grosses » assertions. L'introduction d'un artefact (artifact) permet de pallier ce problème.

BROWSER ARTIFACT

Un artifact SAML permet de référencer de manière univoque une assertion SAML sur le site du *Responder*.

```
TypeCode := 2bytes 0x0001 dans la spécification OASIS
RemainingArtifact := SourceID AssertionHandle
SourceID := 20-byte_sequence
AssertionHandle := 20-byte_sequence
```

SourceID référence le *Responder* chez le *Requester* et doit être défini en out-of-band. Généralement il s'agit du SHA-1 de l'URL du service. (Implique un mapping n°←→URI et par conséquent la définition des serveurs reconnus. Un autre format de l'artefact permet de donner directement l'URI mais la légitimité de cet URI devrait être établie autrement que sur le simple fait que celui-ci répond à des requêtes SAML)

L'*AssertionHandle* référence l'assertion et doit être indépendant des informations contenues dans celui-ci de sorte qu'il soit difficile de deviner ou de forger un numéro valide.

Mesures de sécurité

Un numéro d'assertion est invalidé après la première utilisation afin d'éviter toute tentative de réutilisation.

Le transport de l'artefact doit être sécurisé sinon il est possible de le voler et de se faire passer pour la personne référencée dans l'assertion.

La durée de validité d'un artefact devrait être limitée afin de ne couvrir que la redirection et le transfert d'information avec le second site (de l'ordre de quelques minutes).

L'assertion d'authentification SAML peut être crée soit lors de la création de l'artefact, soit lors de la réception d'une requête comprenant cet artefact. La période de validité de l'assertion doit être ajustée en conséquence.

Le site destination doit refuser les assertions expirées. Il est possible d'utiliser une politique plus stricte et de comparer le moment auquel a eu lieu l'authentification et le moment auquel l'assertion a été crée.

L'assertion SAML peut contenir un *subjectLocality* contenant une adresse IP. Dans ce cas il est également possible de vérifier l'IP du client.

Les attaques sur le protocole de transfert sont évitées en imposant l'authentification, l'intégrité et la confidentialité des messages entre l'émetteur et le récepteur.

Un artefact est ciblé, si une requête SAML provient d'un autre site que celui pour lequel l'artefact a été crée alors cette requête doit être refusée. L'artefact ne peut donc être transmis à autre site que celui pour lequel il a été conçu.

La génération des numéros d'artefact implique qu'il est impossible d'en forger. Néanmoins il faut mettre en place des mesures permettant de détecter qu'une personne essaye de trouver un numéro d'artefact valide (nombreux essai infructueux) et reporter l'incident.

Principe

Voici un schéma résumant les différentes actions à effectuer pour un SSO avec artifact [OBLIX]

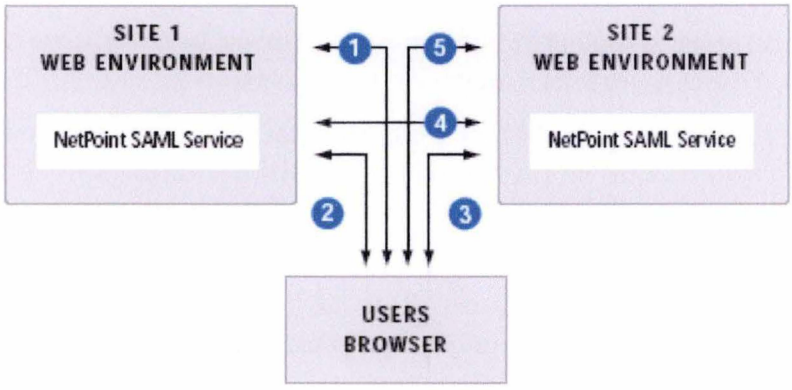


Figure 7: Architecture SSO SAML basique

1. The user navigates to and authenticates to Site1's Web environment.
2. The user clicks on a link to transfer her to Site2. The browser is first transparently forwarded to Site1's SAML service. This SAML service generates an assertion with information about the user, and adds an identifier (a source ID) and a handle for the assertion to the link, so that Site2 can retrieve the assertion from Site1.
3. The browser is now redirected to Site2's SAML service, which retrieves the initial link and the additional information. The source ID tells Site2 how to contact Site1 and the handle tells Site2 how to ask for the user's assertion.
4. Site2's SAML service calls back to Site1's SAML service directly, using the source ID, and using the handle created in Step 2, retrieves the assertion for the user from Site1.
5. Site2's authorization solution then decides whether or not to grant the user access to its Web environment.

FORMAT DES REQUÊTES HTTP

Format de la requête initiale (provenant du navigateur du client)

```
GET <path>?...TARGET=<Target>...<HTTP-Version>
<other HTTP 1.0 or 1.1 components>
```

Where:

`<inter-site transfer host name>`

This provides the host name and optional port number at the source site where an inter-site transfer service is available.

`<path>`

This provides the path components of an inter-site transfer service URL at the source site.

`Target=<Target>`

This name-value pair occurs in the `<searchpart>` and is used to convey information about the desired target resource at the destination site.

Format de la réponse envoyée au client

```
<HTTP-Version> 302 <Reason Phrase>
<other headers>
Location : https://<artifact receiver host name and path>?<SAML
searchpart>
<other HTTP 1.0 or 1.1 components>
```

Where:

`<artifact receiver host name and path>`

This provides the host name, port number, and path components of an artifact receiver URL associated with the assertion consumer service at the destination site.

`<SAML searchpart>= ...TARGET=<Target>...SAMLart=<SAML artifact> ...`

A single target description MUST be included in the `<SAML searchpart>` component. At least one SAML artifact MUST be included in the SAML `<SAML searchpart>` component; multiple SAML artifacts MAY be included. If more than one artifact is carried within `<SAML searchpart>`, all the artifacts MUST have the same `SourceID`.

Cette réponse entraînera la requête suivante auprès du site du *Responder*

```
GET <path>?...<SAML searchpart>...<HTTP-Version>
<other HTTP 1.0 or 1.1 request components>
```

BROWSER POST

Il est également possible de transmettre l'assertion SAML au site du *Responder* en lui envoyant celle-ci dans un formulaire. Néanmoins cela nécessite une intervention de l'utilisateur afin d'envoyer effectivement le formulaire. Il est bien entendu possible de faire cela de manière automatique au moyen de Javascript. Malheureusement le Javascript peut être désactivé par l'utilisateur, ce qui empêcherait le système de fonctionner.

4. Recherches et Applications

Ce chapitre présente quelques projets, recherches qui portent sur une infrastructure d'autorisation. Ceci afin d'évaluer la faisabilité d'une solution utilisant soit SAML soit les certificats d'attribut.

4.1. Akenti

Akenti is a security model and architecture that is intended to provide scalable security services in highly distributed network environments. The project goals are:

- *to achieve the same level of expressiveness of access control that would be accomplished through a local human controller in the decision loop*
- *to accurately reflect the existing policy: authority, delegation, and responsibility present in these environments [AKENTI].*

Akenti propose une architecture d'autorisation basée sur des certificats propriétaires définis en XML. L'authentification des différents acteurs intervenants dans le système est basée sur des certificats X.509 et suppose donc l'admission d'une ou plusieurs CAs.

Les différents certificats d'Akenti sont les suivants :

- Les *policy certificate*
- Les *use-condition certificate*
- Les *attribute certificate* (attention ce ne sont pas des certificats d'attribut X.509)

Suite à l'introduction de ces différents certificats Akenti utilise le terme CA pour indiquer une autorité responsable de l'émission de policy, de use condition, d'attribute certificates (au format XML) ainsi que de certificats à clé publique X.509.

4.1.1. Policy Certificate

Un *policy certificate* est directement lié à une (des) ressource(s) et spécifie quelles sont les autorités responsables de cette ressource. Ces autorités, appelées *stakeholders,* sont responsables de la définition de la politique d'accès à la ressource ainsi que de la gestion des permissions relatives à celle-ci.

Un *policy certificate* contient les informations suivantes :

- La liste des CAs de confiance ainsi que leur clé publique[18] et l'emplacement où chercher les certificats émis par une CA.
- L'emplacement où rechercher les certificats d'attribut.
- La liste des entités accréditées pour l'émission de use condition certificates.
- Les emplacements des use condition certificates.
- Temps maximum de cache pour les certificats utilisés par cette ressource.

Les *policy certificates* ont donc un rôle dans la gestion de confiance[19] du système et constituent l'autorité en matière de définition des responsabilités.

4.1.2. Use condition certificate

Un *use condition certificate* concerne la politique d'accès à la ressource. Plusieurs *use condition certificates* peuvent être émis pour une ressource et ces certificats peuvent être émis par différents stakeholders. Le fait que plusieurs stakeholders puissent définir de tels certificats implique la mise en place d'un processus permettant de les fusionner afin d'avoir une politique cohérente. Akenti définit son propre policy language.

Ce certificat indique également quelles sont les autorités responsables de l'émission des certificats d'attribut ainsi que leurs champs d'application. Ce type de certificat joue un rôle dans la gestion de la confiance car il raffine les champs d'application des différentes CAs indiquées dans le policy certificate.

4.1.3. Attribute Certificate

Un certificat d'attribut Akenti associe un attribut à un sujet sur base du DN de ce sujet ainsi que du DN de la CA responsable de ce sujet.

Conceptuellement ce certificat est identique à un certificat d'attribut X.509 ou à une assertion d'attribut SAML (assertion contenant une déclaration d'attribut).

La référence du porteur s'effectue sur base de son DN et non sur un certificat. Ce qui permet de résister au changement de clé ou au

---

[18] En effet le nom d'une CA n'est pas suffisant pour vérifier l'authenticité d'un certificat (une CA corrompue pourrait prendre le même nom qu'une autre) c'est pourquoi on se base généralement sur la clé publique de celle-ci pour l'identifier.

[19] Akenti déclare les policy certificates comme des certificats *self-signed*. Etant donné que ce type de certificat ne contient pas de clé publique et que le « propriétaire » potentiel est la ressource elle-même, la signification de *self-signed* parait confuse. Cela vient du fait que ce certificat est la source de confiance du système. L'entité signant ce certificat s'autoproclame comme autoritaire pour cette tâche et s'inclut donc dans les CAs de confiance.

renouvellement de certificat. Néanmoins il faut que le DN soit à usage unique (pas de recyclage chez la CA).

4.1.4. Principe de fonctionnement

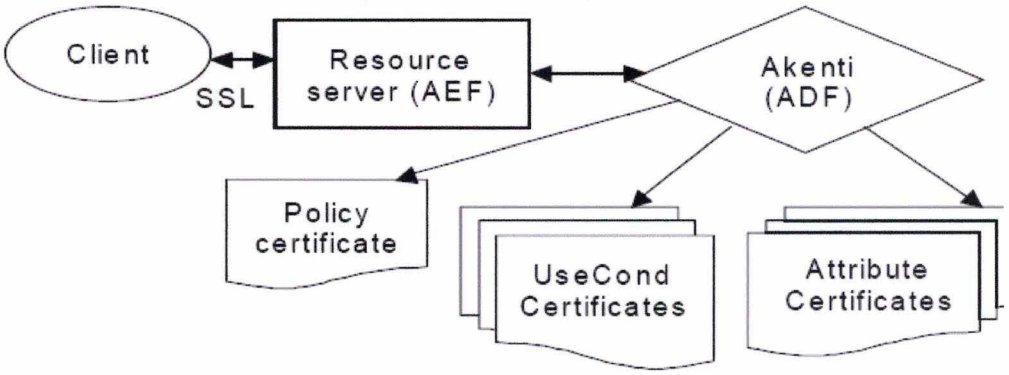


Figure 8:PDP d'Akenti utilisé en mode pull

Le PDP d'Akenti délivre des *capability certificate* qui sont conceptuellement identique à une assertion d'autorisation.

Le PDP se charge de récupérer et de valider les différents certificats concernant la ressource ainsi que le sujet. Les policy et use conditions certificates sont récupérés en mode pull tandis que les attribute certificates sont acceptés en push ou en pull.

La validation consiste à vérifier l'authenticité du certificat (vérification de la signature) mais également à contrôler que l'entité ayant émis le certificat était bien apte à le faire. Ce qui est réalisé grâce aux informations contenues dans les policy et use condition certificates.

4.2. PRIMA

L'objectif de PRIMA est de permettre des collaborations ponctuelles entres les utilisateurs de différents systèmes[20][PRIMA]. La mise en place de ce genre de collaboration implique la reconnaissance d'un nouvel utilisateur dans le système où se situe la ressource.

Le but de PRIMA est donc de promouvoir les compétences de gestion des ressources d'un utilisateur d'un domaine particulier à un domaine plus large.

La solution de PRIMA permet d'exprimer les privilèges accordés à chaque individu ainsi qu'un PDP et un PEP.

4.2.1. Gestion des privilèges

La gestion des privilèges de chaque utilisateur ainsi que la définition des politiques d'accès aux ressources est assurée par deux composants.

LE POLICY CREATOR

Le *policy creator* est un outil graphique assistant l'utilisateur dans la création d'une politique d'accès à une ressource pour lequel il est autoritaire. Cette politique d'accès est représentée sous la forme d'un document XACML inclut dans un certificat d'attribut X.509 (afin d'en assurer l'intégrité lors du transport). L'autorité pour une ressource est la personne possédant cette ressource dans le système.

LE PRIVILEGE CREATOR

Le *privilege creator* est un outil graphique permettant l'émission de certificat d'attribut X.509 associant un privilège à un utilisateur.

4.2.2. Principe de fonctionnement

Les attributs de l'utilisateur sont fournis en mode push au PDP qui retrouve la politique associée à la ressource. Sur base de ces informations, le PDP est capable de rétablir les droits d'accès sur la ressource et de ce fait d'émettre sa décision. Le PDP effectue également un lien vers un compte d'utilisateur générique dont les droits sont gérés dynamiquement.

La séparation entre PDP et PEP n'apparaît pas de manière évidente dans la solution de PRIMA. Ce qui est compréhensible vu que l' "enforcement" est indépendant de l'application[21]. Cela

[20] Ceci dans un contexte de Grid Community afin de permettre la création de communauté ad hoc.

[21] En fait l'enforcement peut être considéré comme spécifique à une application particulière qui est le système.

restreint le champs d'application de PRIMA à des ressources et des privilèges qui sont définis au niveau du système (tels que exécuter un fichier, le lire, etc.), ce qui est en accord avec ses objectifs. En fait, il n'est pas vraiment nécessaire que le PDP émette une décision puisqu'il configure un environnement propre à la requête, par conséquent si les droits sont insuffisants la requête ne pourra pas être effectuée.

4.3. PERMIS

Le projet PERMIS a pour objectif de montrer la faisabilité d'une approche distribuée en matière d'autorisation [PERMIS].

PERMIS s'oriente vers une politique d'accès orientée sur les rôles. Leur solution utilise les certificats d'attribut X.509 afin de transporter les informations d'autorisation (politiques et attributs). PERMIS offre un PDP ainsi qu'un outil permettant de gérer les certificats d'attributs. PERMIS se focalise sur la décision et le management et offre donc une solution indépendante de l'« enforcement » qui reste à charge des applications.

4.3.1. Les politiques

Afin de définir une politique d'autorisation globale, PERMIS scinde cette politique en plusieurs sous politiques spécifiques :

- *SubjectPolicy* : indique les domaines d'utilisateur, seul un utilisateur appartenant à un de ces domaines peut être autorisé à accéder à une ressource.
- *RoleHierarchyPolicy* : indique les différents rôles et leur rapport hiérarchique.
- *SOAPolicy* : indique quelles sont les autorités compétentes pour l'assignation de rôle à un utilisateur. Cela permet la distribution de la gestion des privilèges.
- *RoleAssignmentPolicy* : indique les champs d'application des autorités responsables des différents rôles, c'est-à-dire quelles autorités peuvent assigner quels rôles à quels utilisateurs.
- *TargetPolicy* : indique les ressources (target) couvertes par cette politique.
- *ActionPolicy* : indique les actions (ou méthodes) admises par les ressources ainsi que les paramètres éventuels de ces actions.
- *TargetAccessPolicy* : indique quelles sont les permissions accordées par un rôle sur des ressources particulières. C'est-à-dire quelles actions un rôle permet et sur quelles ressources. Elle peut contenir des contraintes telles que : *"IF time is GT 9am AND time is LT 5pm OR IF Calling IP address is a subset of 125.67.x.x"*. Les actions qui ne font pas partie d'une TargetAccessPolicy sont refusées.

Ces politiques sont exprimées en XML, PERMIS n'offre aucun outil permettant de spécifier ces politiques, il est nécessaire d'utiliser des

outils disponibles sur le marché afin de les créer. Une fois créées, ces politiques sont introduites dans un certificat d'attribut.

4.3.2. Le privilege allocator

Le *privilege allocator* (PA) est un outil permettant d'émettre des certificats d'attribut X.509. Ces certificats d'attribut peuvent contenir une politique d'autorisation (constituée des différentes politiques définies précédemment) ou un privilège d'un sujet/utilisateur (qui ne peut être qu'un rôle dans PERMIS).

Le PA est donc utilisé par l'autorité définissant la politique d'autorisation ainsi que par les autorités d'attribution de privilège. Les certificats sont ensuite stockés sur un serveur LDAP.

PERMIS fonctionne en mode pull ce qui permet de ne pas devoir gérer la révocation des certificats. Il suffit de les enlever du serveur auquel le PDP de PERMIS s'adresse afin de révoquer un privilège ou de modifier une politique. Néanmoins le PDP de PERMIS permet de travailler en mode push mais les informations concernant la révocation des certificats doit être gérée[22].

[22] PERMIS ne précise pas si la validation des certificats fournis en mode push est effectuée par le PDP ou si elle est à charge du PEP.

4.3.3. Principe de fonctionnement

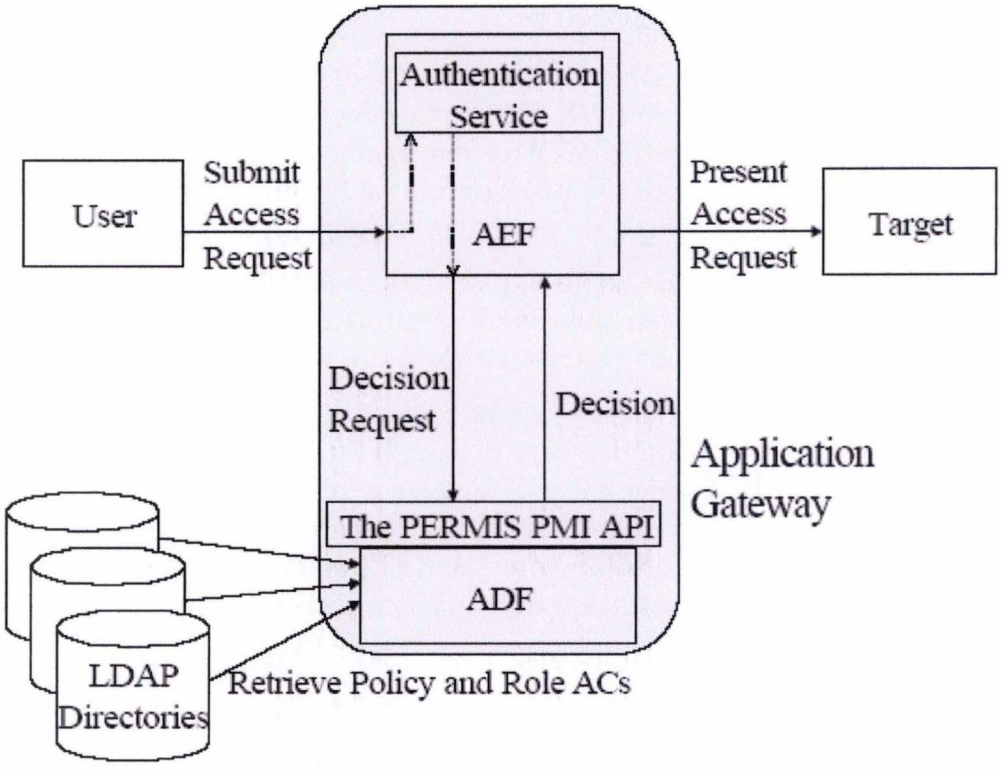


Figure 9: Architecture PERMIS

Le PEP (AEF) crée un PDP (ADF) en lui spécifiant l'autorité principale en matière d'autorisation (celle qui a émis la politique d'autorisation), un identifiant de la politique d'autorisation à utiliser et une liste des serveur LDAP auxquels s'adresser pour retrouver les différents certificats (politique et rôle).

Le PEP fournit ensuite l'identité du sujet au PDP qui retrouve ensuite les certificats le concernant. Ensuite le PEP peut effectuer des requêtes auprès du PDP en lui spécifiant une ressource et une action.

5. Produits commerciaux

Dans ce chapitre, les différents produits commerciaux relatifs au contrôle d'accès et permettant de distribuer celui-ci seront analysés afin de voir quelles sont les possibilités qu'ils offrent. Après analyses des différents produits, il s'avère que tours présentent les mêmes fonctionnalités. La première section exposera donc ces différentes fonctionnalités tandis que la seconde s'intéressera de plus près à un produit particulier.

5.1. Introduction

Les produits commerciaux spécifiques à l'autorisation ont pour objectif de centraliser les décisions d'autorisation de manière à avoir une politique cohérente et d'en faciliter la gestion.

Les produits étudiés sont Oblix © Netpoint, IBM © Tivoli Access Manager for e-business et Netegrity © Siteminder.

Tous les produits présentent les mêmes fonctionnalités générales et leurs architectures respectives sont similaires.

5.2. Fonctionnalités générales

5.2.1. Authentification

Tous les produits prennent en charge différentes méthodes d'authentification. Les principales sont :

- L'authentification simple par login/password
- L'authentification simple sur SSL
- Les certificats X.509
- L'authentification sur base de formulaires
- Microsoft .NET

De plus chaque produit permet d'intégrer une méthode personnalisée d'authentification ou encore une combinaison des différentes méthodes.

5.2.2. Autorisation

Tous les produits offrent une décision d'autorisation centralisée ainsi que différents PEPs spécifiques pour les Webserver et Webapplication.

Concernant les Webserver, deux solutions sont proposées : soit l'utilisation d'un plug-in pour chaque serveur web, soit l'utilisation d'un proxy. Les serveurs web suivants sont supportés par chaque solution :

- Sun ONE Web Server
- Microsoft IIS
- IBM HTTP Server
- Apache

Les produits offrent également l'intégration avec des serveurs d'applications dont IBM Websphere et BEA Weblogic afin de permettre une autorisation plus fine au sein des applications web.

De plus chaque produit permet d'utiliser le PDP de manière indépendante et ainsi de développer leurs propres PEPs afin de les utiliser avec des applications particulières.

Apparemment le PDP s'occupe de récupérer les informations concernant l'utilisateur et il semble qu'aucun des produits ne permette de travailler en mode push.

5.2.3. Administration

La délégation de l'administration est permise dans chaque produit. Une application internet (thin client) est fournie afin d'effectuer les opérations d'administration par navigateur.

L'expressivité des politiques permet d'intégrer des restrictions horaires ainsi que des informations dynamiques provenant d'autres sources (autre PDP, solde d'un compte). Ce qui permet potentiellement d'exprimer des politiques pour des ressources très diversifiées et d'élargir le champ d'application offert.

Les produits proposés offrent la possibilité de déléguer les différentes fonctions administratives relatives à la gestion des utilisateurs (attribution de droit, rôle, appartenance à un groupe) ainsi que la définition des politiques d'accès aux ressources.

5.2.4. Audit

Tous les produits offrent des possibilités d'audit et de reporting avancées permettant de surveiller les intrusions ainsi que des entités spécifiques du domaine : ressource, utilisateur, serveur web, etc.

5.2.5. Fédération – Collaboration

Chaque produit offre des facilités permettant la collaboration avec d'autres entreprises / domaines. SAML est la solution la plus couramment envisagée concernant le SSO[23] et le partage d'informations relatives à la sécurité. Certains produits offrent d'autres possibilités qui leur sont propres.

5.2.6. Intégration

Tous les produits permettent d'utiliser le PDP indépendamment au moyen d'API. Elles permettent également de personnaliser les différents composants :

- Authentification
- Audit
- Interface de gestion
- ...

Tous les produits offrent l'intégration avec Websphere, Weblogic, Siebel 7 ou 2000, SAP, PeopleSoft.

Il est également possible de combiner leur système d'autorisation avec des produits spécifiques à la gestion des utilisateurs. (netegrity identity minder, Tivoli Identity manager).

5.2.7. Load balancing, résistance aux pannes

La duplication du PDP et des proxy est possible pour chaque produit. Les directory contenant les informations d'autorisation (utilisateurs, politiques, privilèges) peuvent également être dupliqués et le système peut basculer automatiquement vers un duplicata en cas de panne.

[23] Chaque solution offre un SSO au sein de leur domaine propre (domaine qui peut concerner plusieurs webserveur, webapplication, etc).

5.3. Présentation de Netegrity SiteMinder

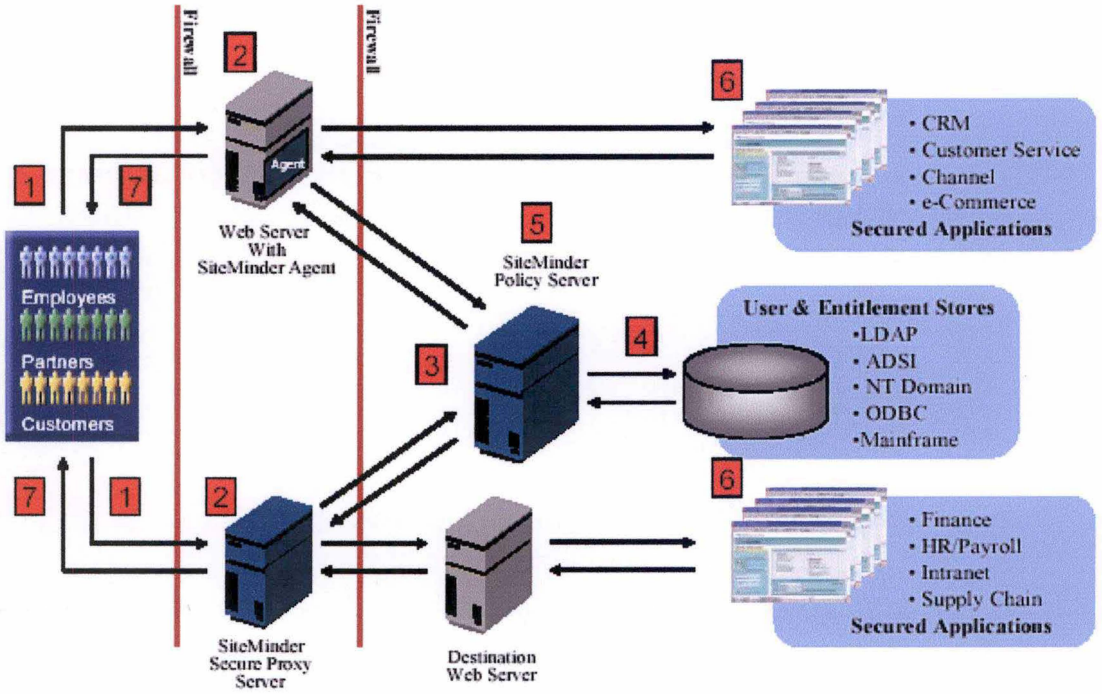


Figure 10: Netegrity SiteMinder

Le *Policy Server* est l'équivalent d'un PDP, tandis que les différents *Agents* sont des PEPs.

5.3.1. Policy Server

Le Policy Server n'est pas seulement un PDP, il prend également en charge l'authentification et la gestion des politiques d'accès.

AUTHENTIFICATION

SiteMinder accepte les méthodes d'authentification suivantes :

- Basic Authentication (user-name/password)
- Basic Authentication over SSL
- ACE/Server (RSA Security)
- Crypto-Card
- RADIUS Proxy
- Forms-based authentication
- X.509 certificates
- Custom or third-party schemes
- Microsoft .NET Passport

- SAML Assertion

SiteMinder permet également d'utiliser une combinaison de ces méthodes, de quantifier chaque méthode et de définir un minimum requis pour chaque ressource protégée.

GESTION DE L'AUTORISATION

Le Policy Server gère des politiques d'accès qui définissent les opérations autorisées sur chaque ressource. Netegrity Siteminder permet d'énoncer des politiques très complètes et de prendre en compte des informations environnementales.

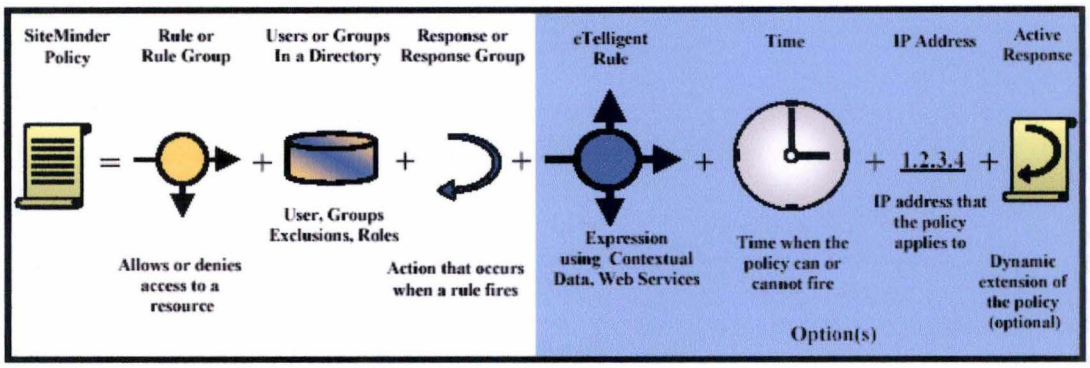


Figure 11: Composants d'une politique

5.3.2. Les Agents

Les agents communiquent avec le policy server pour authentifier les utilisateurs et autoriser l'accès aux ressources. Les Agents sont des PEPs relatifs à des domaines spécifiques. Netegrity identifie 5 types d'agents :

- Les *Web Agents* fonctionnent avec un serveur web, interceptent les requêtes et définissent si la ressource visée est protégée ou non par SiteMinder. Si la ressource est protégée, la requête est soumise à une autorisation du Policy Server après quoi elle sera transmise au Web Server. Les serveurs web supportés sont les suivants :
  - Microsoft IIS (NT/Win2000)
  - Sun ONE Web Server (NT/Win2000, Solaris, HP, AIX)
  - Apache—Covalent & Stronghold (Solaris, HP-UX, AIX, Linux, NT/Win2000)
  - Lotus Domino (NT/Win2000 & Solaris)
  - IBM HTTP (Solaris, AIX)
  - Oracle HTTP Server (Solaris & HP-UX)
  - Domino Go (OS/390)
- Les *Application Server Agents (ASA)* sont utilisés afin de sécuriser des objets plus particuliers spécifiques à des

applications Web tels que servlets ou JSP. Par exemple Netegrity fournit des ASA Weblogic ou Websphere.

- Les *Affiliate Agents* sont utilisés pour communiquer avec des autres entreprises afin de faciliter la coopération. Les informations échangées avec ces *affiliate agents* (qui se trouvent sur le site du partenaire) sont des informations concernant l'utilisateur. Ces agents permettent de mettre en place une solution de SSO (autrement que par SAML) et de faciliter la réalisation des transactions.
- Les *Enterprise Application Agents* permettent d'utiliser SiteMinder conjointement à SAP, Siebel ou PeopleSoft afin de mettre en place des solutions SSO.
- Les *Custom Agents* sont des agents spécifiques à certains domaines. SiteMinder fournit des API permettant de définir ces agents et ainsi d'élargir le domaine d'application de leur solution d'autorisation.

5.3.3. SSO

SiteMinder offre plusieurs possibilités afin d'assurer le SSO. De base, le Policy Serveur prend en charge le SSO via un système de cookies.

En inter domaine, SiteMinder propose deux solutions, soit l'utilisation d'un *Affiliate Agent* qui permet de communiquer avec le domaine source, soit l'utilisation de SAML.

5.3.4. Audit

SiteMinder permet de surveiller toutes les transactions, que ce soit au niveau des agents, des users ...

- **Activity Reports**—Review usage by user, agent, or resource
- **Intrusion Reports**—Capture all failed user authentication and authorization attempts organized by user or agent.
- **Administrative Reports**—Recap all administrative activities organized by administrator or object.
- **Time Series Reports**—Charts successful and failed authentication and authorizations over time.
- **Auditing**—Provides audit event filters and APIs.

5.3.5. Intégration

SiteMinder inclut un Software Developer's Kit (SDK) qui permet aux développeurs d'ajouter leurs propres méthodes d'authentification, règles d'utilisation et politiques dynamiques, de

créer leurs propres GUIs pour l'administration des privilèges et des politiques, de développer des agents spécifiques ou de capturer des événements pour une analyse spécifique.

- **Policy Management API**—Used to develop custom Policy Server user interface applications or to modify components of policy objects, such as rules, policies, and responses.
- **Agent API**—Used to create custom agents to leverage the authentication and authorization services of SiteMinder
- **Authentication API**—Used to create and integrate custom authentication schemes with SiteMinder
- **Authorization API**—Used to develop custom authorization modules and to integrate them with SiteMinder
- **Directory API**—Used by third-party applications to extract information from the underlying directories using SiteMinder services
- **Event API**—Used to build custom event handlers that leverage SiteMinder events
- **Tunnel Service API**—Used to develop services that communicate with agents to transfer data securely.
- **DMS API**—Used to develop custom user management applications leveraging SiteMinder. This API allows you to manage directory entries, discover user privileges, enable/disable users, and grant roles to users.

6. Application chez ISABEL

6.1. Présentation d'ISABEL[24]

Isabel S.A. a vu le jour en 1994, lorsque les quatre principales banques de Belgique décidèrent de développer une super autoroute de l'information qui relierait les entreprises aux banques et à des services d'informations et surtout, qui leur offrirait un superbe moyen d'interconnexion.

Le but était d'offrir au monde des affaires une solution intégrée, soutenant une large gamme de services financiers, dans un environnement parfaitement sécurisé.

6.1.1. Global Trust Authority

Isabel est membre fondateur de Global Trust Authority (GTA) et représente la Belgique au sein de cet organisme international. Ce dernier a été constitué par des banques et des associations bancaires afin de faciliter les transactions sécurisées sur le web, notamment en matière de commerce électronique. Plus de 800 banques y sont représentées.

Le premier objectif de GTA, à la demande de ses membres, consiste à établir une interopérabilité entre les différents systèmes de certification existants. Il n'est toutefois pas question de remplacer les produits et les infrastructures mis en place par les banques et associations bancaires telles qu'Isabel, mais d'y coupler des mécanismes communs d'authentification.

6.1.2. Derrière la signature digitale Isabel

Isabel est la plus importante autorité de certification en Belgique. C'est-à-dire que c'est l'institution qui a fourni le plus de certificats digitaux de classe 3. Ces derniers consistent en cartes d'identité digitales garantissant l'authenticité de la personne avec laquelle sont échangées des données électroniques. Isabel gère ce système de sécurité à l'aide d'une imposante

[24] Information provenant du site www.isabel.be

infrastructure PKI.

Une Directive européenne, publiée au Journal officiel des Communautés le 19 janvier 2000, fixe à présent un cadre communautaire pour les signatures électroniques. La Belgique l'a traduite dans son droit national en adoptant le 9 juillet 2001 une loi qui permet de donner une valeur légale équivalente à la signature digitale et à la signature manuscrite.

Les transactions électroniques, entre le privé et le public ainsi qu'à l'intérieur de ces mêmes secteurs, vont pouvoir se développer en toute confiance. Dans ce cadre, Isabel a déjà réalisé plusieurs projets d'eGovernment, d'eBanking et d'eCommerce.

Public Key Infrastructure

Il est crucial, avant de se lancer dans l'eBusiness sur Internet, de pouvoir **identifier** vos fournisseurs, vos clients et autres partenaires via leur signature digitale respective. De la même manière, eux aussi souhaitent vous authentifier.

Chacun doit en outre être certain de l'**intégrité** des données échangées ; c'est-à-dire qu'elles n'ont pas été modifiées durant le transfert.

Une fois qu'un utilisateur a apposé sa signature digitale, il importe aussi qu'il ne puisse plus refuser ou réfuter la transaction. C'est ce qu'on appelle la **non répudiation** de l'acte.

Pour atteindre ces objectifs, deux éléments primordiaux doivent être réunis : les paires de clefs[25] et les certificats digitaux[26]. Et pour gérer ceux-ci, une architecture PKI s'impose. Un système totalement transparent pour vous.

Le PKI permet de créer, de délivrer, d'administrer et de révoquer des certificats digitaux de manière (dé)centralisée. Enfin, il est chargé de publier les certificats et les clefs publiques ainsi que leur validité, dans un répertoire central, afin que tout utilisateur Isabel puisse vérifier l'identité du certifié.

Forte de son expérience en matière de sécurité digitale en Belgique, Isabel vous offre des solutions et des services PKI qui peuvent être intégrés à vos projets d'eBanking, d'eCommerce et d'eGovernment.

---

[25] Voir cryptographie asymétrique

[26] Voir certificats

ISABEL CERTIFIE VOTRE SIGNATURE DIGITALE

Isabel est une autorité de certification (CA pour Certification Authority) délivrant des certificats de la classe 3. En Belgique, la classe 3 est obligatoire pour toute communication électronique où des données délicates, confidentielles et financières sont échangées entre entreprises, pouvoirs publics, institutions bancaires et particuliers.

En tant que CA, Isabel fournit un certificat de la classe 3 comme suit :

- L'utilisateur s'abonne et installe le logiciel sur son PC.
- Il configure et sécurise l'accès à son logiciel en suivant une procédure de certification : il crée de façon aléatoire une paire de clefs indissociables (une privée et une publique) grâce au système de **cryptage asymétrique RSA**. La clef privée totalement secrète est enregistrée sur sa **carte à puce**. A cette carte à puce personnelle, l'utilisateur associe un **mot de passe** qu'il devra toujours taper lorsqu'il voudra utiliser son logiciel ou/et apposer une signature digitale.
- Afin d'éviter tout quiproquo, il imprime les codes correspondants à sa clef publique que le logiciel Isabel lui génère. Puis, il **se présente personnellement auprès de sa banque** pour échanger ces codes contre deux autres codes qu'il devra intégrer dans son logiciel Isabel pour devenir opérationnel.
- **La banque enregistre de manière très détaillée les données** et la clé du client.
- **Isabel fournit le certificat.** La clef publique est alors certifiée par Isabel et rendue accessible à tous les utilisateurs car elle leur permettra de vérifier la signature digitale et donc l'identité du nouvel abonné. La **signature digitale** de ce dernier devient donc infalsifiable. Elle s'avère même plus sûre que sa signature manuscrite.

Grâce à ce système et à l'infrastructure PKI qui l'entoure, tout utilisateur peut vérifier l'**intégrité** (le contenu n'a pas été altéré) et l'**origine d'un message**. La certification garantit aussi l'**authenticité des parties** dans le processus de télécommunication. La fonction de **cryptage** (chiffrement) rend enfin les messages uniquement lisibles pour l'expéditeur et le destinataire.

6.1.3. Activités

L'offre d'Isabel ne se limite pas à une PKI. Cette partie n'est qu'un soutient aux activités principales d'Isabel: les transactions électroniques. Isabel possède plusieurs offres logicielles permettant de réaliser ses transactions en ligne, par exemple transactions bancaires, déclaration de TVA, facturation, ...

Pour réaliser ces services, Isabel collabore étroitement avec les organismes financiers. Isabel travaille avec plus de 20 banques et est la seule à offrir un système de paiement multi bancaire en Belgique.

6.2. Domaine d'application

ISABEL est intéressé par l'échange d'informations relatives à l'autorisation avec ses différents partenaires: les banques. SAML étant un standard en voie de développement soutenu par plusieurs grands organismes, l'enthousiasme commun à l'égard de XML ainsi que les Webservices font de SAML une solution attrayante pour ISABEL.

Le problème est relatif à la multi signature de paiements dans le domaine bancaire. Il s'agit de contrôler d'une part si une personne est autorisée à signer un paiement (c'est-à-dire entre autre si elle possède un mandat sur le compte donneur d'ordre) et d'autre part si la signature de cette personne est suffisante pour valider l'exécution de ce payement.

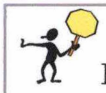
L'autorisation concernant la capacité à signer est effectuée en interne chez ISABEL. Par contre ISABEL sera amené à effectuer la validation d'un payement en interne ou à demander l'autorisation à la banque concernée.

6.2.1. La validation d'une signature

La signature d'un payement consiste à signer numériquement un ordre de payement qui est destiné à une banque. Dans ce contexte, la validation d'une signature consiste simplement à vérifier que la personne effectuant cette signature peut exécuter cette action.

Cette validation concerne :

- la vérification des permissions de la personne chez ISABEL, c'est-à-dire si la personne est autorisée à effectuer l'action *signer un paiement*,
- la vérification des éléments cryptographiques (clé publique non révoquée, CA autorisée, etc.),
- la vérification du mandat de la personne concernant le compte donneur d'ordre,
- les montants pour lesquels la personne mandatée est compétente,
- éventuellement les comptes bénéficiaires sur lesquelles cette personne est autorisée à effectuer des versements.



La signature n'est qu'un moyen utilisé en vue de réaliser un objectif, il est plus opportun de parler de validation d'une action plutôt que de validation d'une signature. En effet, par le biais de la signature, l'utilisateur marque explicitement son accord pour une action, dans ce cas effectuer un payement. Il est donc plus juste de parler de validation d'un accord plutôt que de validation de signature. De plus le fait que la signature soit correcte ne signifie pas pour autant que l'utilisateur ait les droits d'effectuer l'action.

6.2.2. La validation d'un paiement

Un payement concerne un compte donneur ainsi qu'un compte bénéficiaire et un montant. Un payement peut être signé par plusieurs personnes mandataires pour ce compte.

La validation d'un payement consiste à déterminer si suffisamment de personnes ont donné leur accord que pour permettre l'exécution du paiement. C'est-à-dire de déterminer si les mandataires sont suffisamment compétents que pour permettre l'exécution de ce paiement.

La restriction principale concernant un payement concerne son montant. Il faut dès lors déterminer les limitations concernant le compte (ex. : solde du compte) ainsi que les limitations spécifiques aux différents mandataires qui restreignent ce montant.

Exemple de limitations concernant les mandataires : Jean est compétent pour signer des ordres jusque 20 000 € tandis que Pierre n'est compétent que pour 10 000 €. Par contre les deux ensembles sont compétents pour des sommes allant jusque 50 000 €.

Les limitations concernant le compte ne doivent pas être gérées par ISABEL pour la validation d'un paiement.

6.3. Les différentes options de modélisation

Dans la problématique posée, le problème est de savoir si ISABEL peut envoyer un paiement à la banque afin qu'il soit exécuté : le problème a été posé sous la forme d'autoriser un paiement. D'une manière générale l'autorisation consiste à déterminer si un **sujet** peut effectuer une **action** particulière sur une **ressource** donnée. Il convient donc de déterminer quels sont les sujets, actions et ressources sur lesquels porte l'autorisation.

6.3.1. Le paiement en tant que ressource

Si on essaye de placer le paiement au niveau de la ressource, la question que l'on se pose est : « *Est-ce que l'on peut exécuter ce paiement ?* » Le sujet effectuant l'action est inconnu ou sans importance pour la prise de décision (il ne s'agit donc pas d'un problème d'autorisation). Il faut déterminer si l'action « exécuter » est disponible sur cette ressource. C'est-à-dire définir le type de la ressource ou encore définir si le paiement est un paiement acceptable ou un paiement invalide.

6.3.2. Le compte en tant que ressource

Le problème peut être reformulé comme suit : il faut savoir si un ordre de transfert, signé par un ou plusieurs mandataires, vers un compte bénéficiaire est autorisé à partir du compte donneur d'ordre.

Si on place le transfert en tant que sujet et le compte donneur d'ordre en tant que ressource, la question devient : *« Ce transfert peut-il être exécuté à partir du compte xxx ?»*. Ce qui peut se ramener à un problème d'autorisation prenant en compte les différents attributs du transfert (signataire(s), montant, compte bénéficiaire) et la politique d'utilisation du compte (signataires A+B et montant < 10000 ou signataires A+B+C et montant < 100000, etc.).

6.3.3. Les signataires en tant que sujet

En raffinant le modèle précédent, on peut considérer les signataires en tant que sujet et le compte en tant que ressource. Les actions sont alors « exécuter un paiement de x € » (si on ne tient pas compte des bénéficiaires acceptables).

De ce point de vue, les signataires deviennent des attributs particuliers du sujet, tandis que le compte bénéficiaire et le montant sont des caractéristiques propres à l'action (une opération bancaire).

6.3.4. Intérêt d'une architecture d'autorisation

La réponse à la question est-il possible d'utiliser SAML pour transporter l'information entre ISABEL et la banque est donc oui

puisque le problème peut être formulé en termes d'autorisation. La question qui se pose maintenant est de savoir quels sont les intérêts qu'offre une telle architecture plutôt qu'une simple validation des données.

La mise en place d'une architecture d'autorisation, outre le fait d'assurer le contrôle d'accès a pour objectif de faciliter la gestion des informations. Quelles sont donc les facilités qu'offre la mise en place d'un tel dispositif?

Du point de vue des actions

Les transactions bancaires se font en général à partir d'un compte particulier et par des personnes qui sont mandatées pour ce compte. L'introduction de nouveaux types d'opération sur les comptes se trouve donc facilitée. Par exemple :

- consulter l'historique du compte,
- établir un ordre permanent,
- l'acceptation d'un nouveau mandataire sur le compte,
- la définition d'un nouveau type de mandataire,
- la modification de pallier pour lesquels les différents mandataires sont compétents (gestion des droits sur un compte),
- ...

De plus, il est également possible de raffiner l'opération d'émission d'un ordre de paiement comme par exemple des paiements nationaux, internationaux. L'architecture d'autorisation mise en place pourra être réutilisée pour ces actions.

Point de vue des ressources

Les comptes bancaires ne sont pas les seules ressources gérées par les banques. Les actions, SICAV et autres avoirs gérés par les banques pourront être introduits dans le système avec un coup de gestion réduit.

Du point de vue des développeurs

L'utilisation de SAML quant à elle offre une structure commune entre les banques et ISABEL ce qui facilite la coopération étant donné qu'une partie de la sémantique du message est déjà définie par la spécification SAML. Cela lève l'ambiguïté de la compétence d'ISABEL quant à la définition de cette sémantique commune (bien que ce soit là son rôle) et facilite l'adoption par tous les acteurs.

Cette remarque concerne également les interactions entre les banques elles-mêmes qui pourront utiliser le même format pour l'échange de ce type d'informations.

6.4. Extension du problème

Dans la pratique, ISABEL envoie un fichier contenant plusieurs paiements à la banque. Ce fichier est signé par un ou plusieurs mandataires sur les comptes concernés et est exécuté dans son ensemble par la banque.

La validation d'un accord peut donc poser problème car c'est le fichier qui est signé et non chaque paiement. Par exemple lorsque la personne n'est pas mandataire sur tous les comptes mais seulement sur une partie de ceux-ci, il est nécessaire de définir quel est le comportement qu'ISABEL doit adopter. Il faut déterminer ce que signifie alors le fait qu'une personne signe un tel fichier. L'utilisateur marque-t-il sont accord pour tous les comptes où il est mandataire et pas pour les autres ou la signature est-elle simplement refusée? Plusieurs options sont donc envisageables, marquer son accord pour tous les paiements, seulement ceux pour lesquels le signataire est mandataire ou encore marquer son accord pour une partie des paiements seulement.

1 ÈRE OPTION

L'utilisateur marque son accord pour l'entièreté des paiements du fichier. Imaginons qu'une personne signe un fichier de paiements concernant les comptes A, B et C mais qu'il ne soit mandataire que sur les compte A et B.

Figure 12: Signature d'un fichier de paiement

En signant ce fichier, monsieur Dupont donne son accord pour les paiements concernant les comptes A et B mais pas pour le compte C puisqu'il n'est pas mandataire pour ce compte. Lors de la vérification de la signature a posteriori, seul seront pris en compte les accords pour les compte A et B, monsieur Dupont n'étant pas mandataire sur le compte C, son accord sur ce compte ne sera donc pas pris en compte pour la validation du paiement (déterminer si on peut l'effectuer).

Cela fonctionne seulement si les droits de monsieur Dupont ne sont pas altérés entre le moment de la signature et la

vérification. Imaginons qu'il devienne mandataire pour le compte C entre temps, lors de la vérification monsieur Dupont sera considéré avoir donné son accord pour effectuer le paiements alors que ce n'est pas le cas. Il faudrait donc garder trace des modifications des droits pendant un certains laps de temps ceci afin de pouvoir définir clairement quels sont les paiements pour lesquels chaque personne marque son accord. Cette période devrait couvrir le temps maximal entre la signature d'un paiement et son exécution, ce qui dans le cadre de multi signature peut s'avérer très long.

2ème option

Une autre option consiste à ne signer que les paiements pour lesquels l'utilisateur est mandataire au moment de la signature.

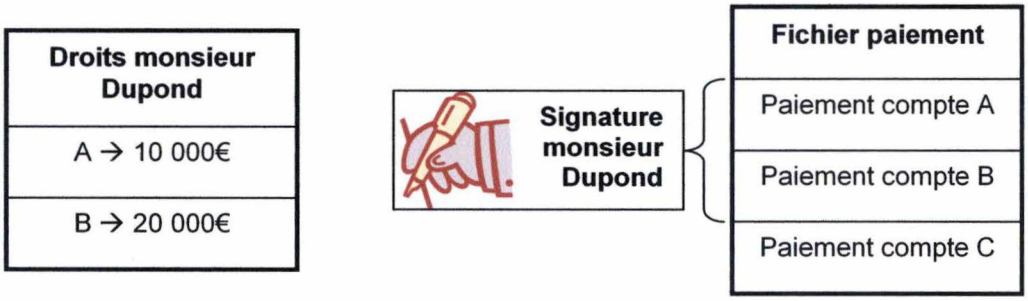


Figure 13: Signature des paiements sur comptes mandataires

Dans ce cas, l'opération de signature est plus complexe et comporte des informations complémentaires permettant de définir a posteriori quels sont les éléments à prendre en compte pour la vérification de la signature mais permet d'éviter le problème de modification des droits.

3ème option

Il est possible que l'utilisateur ne souhaite pas donner son accord pour tous les paiements concernant un compte sur lequel il est mandataire mais seulement pour une partie des comptes présents dans le fichier.

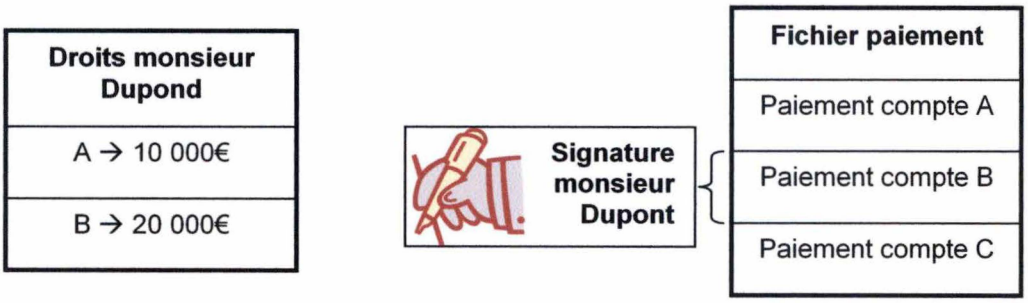


Figure 14: Signature d'un sous-ensemble de paiements

Cette option est semblable à la seconde, seule la méthode de détermination des paiements pris en compte par la signature diffère.

Impact sur l'autorisation

Du point de vue de l'autorisation, cette extension influence le policy decision point (PDP) qui devra être adapté afin de refléter l'option la plus adaptée et effectuer la corrélation entre les autorisations relatives aux des différents comptes concernés.

6.5. Architectures générales envisagées

Etant donné qu'ISABEL travaille avec de multiples partenaires, il est nécessaire de prendre en compte les besoins et compétences de chacun d'entre eux. Le schéma ci-dessous présente les différents éléments qui pourraient intervenir dans la mise en place du système d'autorisation de paiements avec prise en charge de la signature multiple.

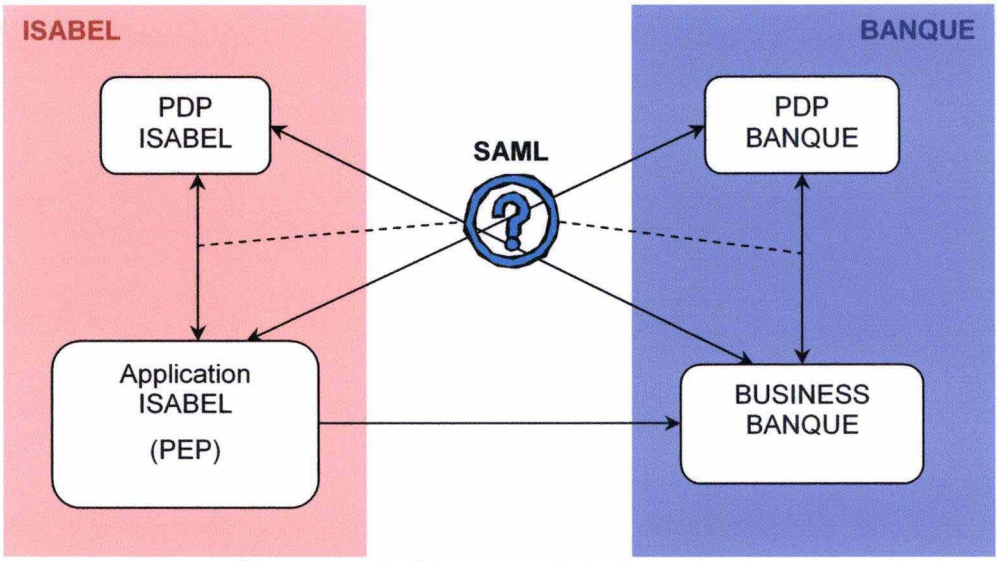


Figure 15: Architecture générale

Selon le rapport de confiance existant entre ISABEL et son/ses partenaires, ainsi que les obligations de part et d'autre (obligation régies par contrat) le rôle d'ISABEL peut changer fondamentalement.

Trois cas de figure sont actuellement envisagés :

1. ISABEL agit en tant que tiers de confiance et effectue la validation des paiements ainsi que leur exécution,
2. ISABEL agit toujours en tant que tiers de confiance mais n'effectue plus la validation des paiements,
3. ISABEL n'est plus considéré comme tiers de confiance et ne fait que transmettre le fichier de paiement à la banque.
4. ISABEL transmet l'information d'autorisation à la banque mais la banque est libre de l'accepter ou non

6.5.1. Premier cas de figure

Dans le premier cas de figure, ISABEL est un partenaire de confiance et effectue toute les opérations de vérification ainsi que l'exécution proprement dite des actions qui en résulte. La banque délègue la quasi-totalité des opérations à ISABEL.

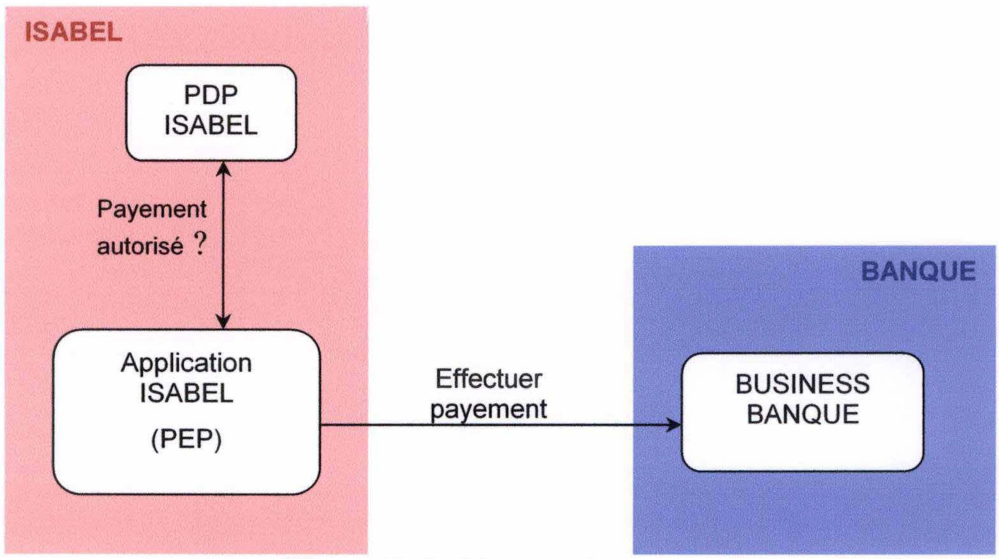


Figure 16: Architecture 1

ISABEL effectue toutes les vérifications et informe ensuite la banque des actions/paiements à effectuer. Dans ce cas le PDP de la banque n'est pas sollicité. ISABEL agit auprès de la banque en tant que Trusted Third Party. C'est-à-dire que la banque lui fait confiance concernant les paiements qu'elle effectue via BUSINESS BANQUE. Ce qui sous-entend que les vérifications effectuées par ISABEL ne seront plus effectuées par la banque et que donc seules les restrictions dues au compte (solde insuffisant, compte bloqué) pourraient faire que le paiement ne soit pas accepté.

Dans cette solution, il est possible que la banque ne puisse pas totalement décrire sa politique d'autorisation dans l'implémentation offerte par ISABEL. En contrepartie aucun développement n'est nécessaire pour la banque. Cependant, l'autorité responsable des informations sur lesquelles ISABEL base ses décisions reste la banque. La banque doit donc fournir ces informations à ISABEL (provisionning dans notre cas).

6.5.2. *Deuxième cas de figure*

Dans le second cas de figure, ISABEL n'effectue plus la validation des paiements.

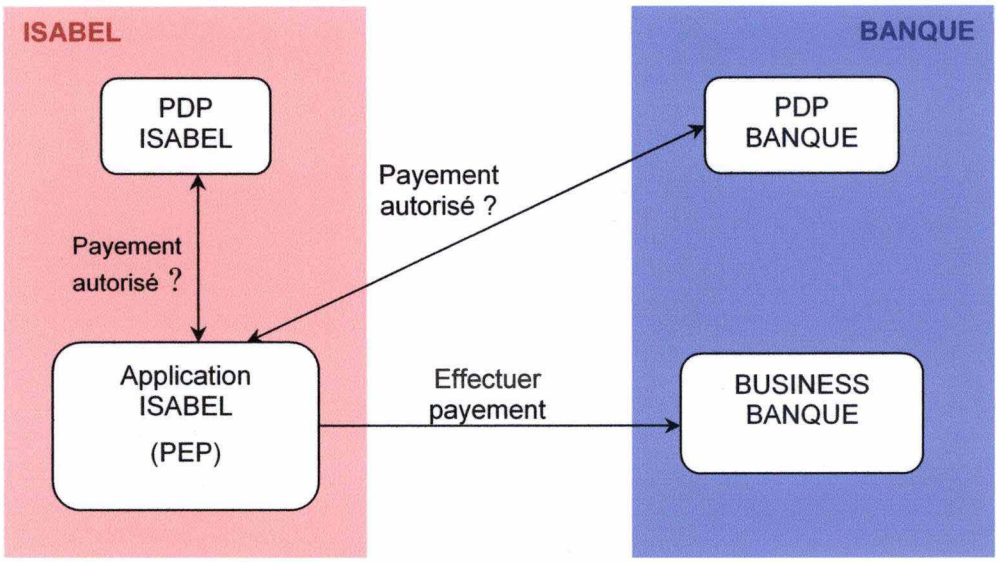


Figure 17: Architecture 2

ISABEL valide la signature (validation cryptographique) mais c'est la banque qui effectue la vérification du paiement. Si le service de validation de la banque est indisponible alors ISABEL effectue elle-même la validation. Dans tous les cas ISABEL envoie toujours les actions à effectuer à la banque. ISABEL agit toujours en tant que Trusted Third Party.

Cette option offre l'avantage de dégager ISABEL du processus de décision concernant la validation d'une transaction. Bien que la plupart des transactions soient similaires parmi les différentes banques, il est possible que leurs processus de décision respectifs soient fort différents. L'acceptation ou le refus d'une transaction dépendent typiquement de la politique que la banque désire adopter en la matière. La modification du processus décisionnel n'entraîne aucun changement chez ISABEL.

6.5.3. *Troisième cas de figure*

Dans le troisième cas de figure, ISABEL se contente de transporter le fichier de paiements jusqu'au service approprié de la banque.

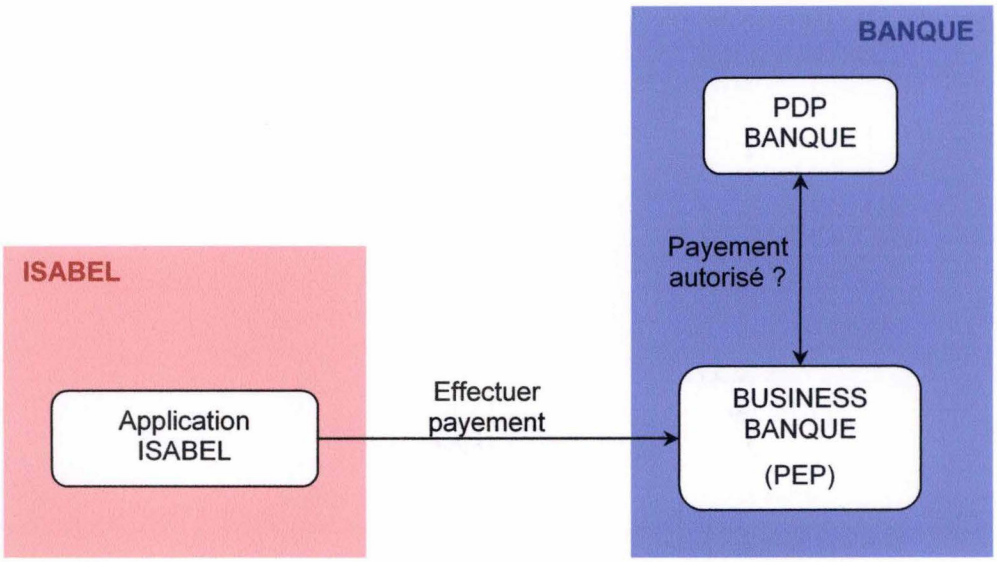


Figure 18: Architecture 3

ISABEL effectue éventuellement la validation des signatures mais pas la validation de l'ensemble des paiements. Elle envoie directement le fichier de paiements à la banque.

Cette option est extrêmement restrictive. Le rôle d'ISABEL se limite à celui d'un transporteur. ISABEL n'a pas les moyens de décider si un paiement est valide ou non. Le traitement ultérieur d'un fichier de paiements entraînerait un délai supplémentaire dans le traitement d'un paiement.

De plus cette architecture ne correspond pas à la mission d'ISABEL vis-à-vis des banques.

6.5.4. *Quatrième cas de figure*

Dans les deux derniers cas, la banque désire effectuer elle-même la décision, plus pour des raisons d'efficacité, de facilité ou de compétence que de confiance. La deuxième architecture présente une solution de secours qui n'est pas en adéquation avec les besoins spécifiés par la banque à savoir conserver le processus de décision.

D'une manière générale, il serait plus logique pour la banque d'offrir un PEP à ISABEL plutôt qu'un PDP. Dans ce cas, on se ramène à la troisième option si ce n'est qu'on fournit au PEP de la banque l'affirmation du PDP d'ISABEL (en mode push). La banque peut alors décider de faire confiance ou non aux affirmations présentées par ISABEL selon qu'elle le considère comme un tiers de confiance ou non.

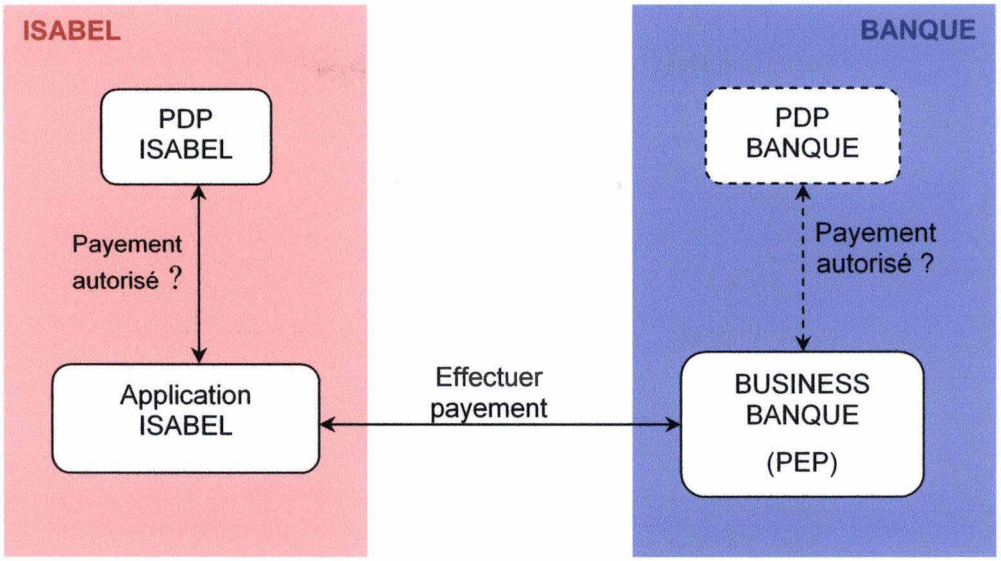


Figure 19: Architecture 4

Les parties pointillées dépendent de la politique que les banques désirent adopter vis-à-vis d'ISABEL. Si ISABEL fournit des informations d'autorisation conjointement au fichier de paiement c'est à la banque de décider si oui ou non ces informations sont dignes de confiance. Si ISABEL ne fournit pas d'information alors c'est également à la banque de faire des vérifications complémentaires. Cette architecture englobe simultanément les trois cas présentés.

6.5.5. Remarques complémentaires

Ces architectures sont encore à l'état de proposition. ISABEL n'a pas encore fait de choix quant aux options qui devront être effectivement mises en place. Pour les banques, le second choix pourrait être motivé par des considérations concernant la réutilisation des relations et dispositifs déjà en place. Si les banques interagissent déjà avec ISABEL via un *business banque* et qu'ISABEL est déjà considérée comme TTP alors la banque peut fournir un mécanisme de décision à ISABEL. Si ISABEL n'est pas encore TTP auprès de la banque il est alors préférable d'opter pour la troisième solution.

Le choix dépend également de considérations commerciales et du rapport de force entre ISABEL et les banques. Dans l'architecture alternative, toute la charge de coopération est attribuée à la banque.

6.6. Définition des différents messages SAML

Selon l'architecture envisagée, seul un sous ensemble de la spécification SAML est employé afin d'échanger les informations d'autorisation.

6.6.1. Architecture 1

Dans cette architecture aucune information d'autorisation n'est échangée avec la banque. Seule une requête d'exécution est envoyée à la banque. Toutes les vérifications d'autorisation ont été faites préalablement. Par conséquent SAML n'est d'aucune utilité.

Néanmoins la banque doit fournir préalablement les informations à ISABEL concernant les mandataires ainsi que les politiques d'autorisation des comptes. Dans cette optique il pourrait être envisagé d'utiliser les assertions SAML afin d'assurer l'échange d'information. Actuellement la banque effectue déjà le provisionning de certaines BDs chez ISABEL. Il est fort probable que le même mécanisme soit réutilisé afin de transmettre ces informations.

6.6.2. Architecture 2

Dans la deuxième architecture, ISABEL demande une autorisation à la banque afin d'effectuer les paiements. Cette architecture peut utiliser le protocole SAML avec des requêtes (query) d'autorisation ainsi que des assertions d'autorisation SAML.

La quasi-totalité de SAML est alors utilisée: SAML et SAMLp (protocole pour effectuer les requêtes SAML).

6.6.3. Architecture 3

La troisième architecture est similaire à la première. Il n'y a aucun échange d'informations d'autorisation entre les deux parties. La banque ne fournit aucune information à ISABEL. Par conséquent SAML est également inutilisé.

6.6.4. Architecture 4

Dans l'architecture alternative, les informations d'autorisation sont échangées en mode push par ISABEL. Une assertion d'autorisation émise par ISABEL est jointe à la requête d'exécution.

6.7. Conclusion

Il ressort que SAML peut être utilisé pour transporter les informations d'autorisation entre les différents acteurs du système dans le cadre des multi signatures. Dès lors, la question à élucider est de savoir quels sont les avantages que présente l'emploi SAML.

L'utilisation de SAML s'inscrit dans le cadre d'une architecture d'autorisation. La mise en place d'une telle architecture uniquement pour l'exécution d'un paiement est très certainement une surcharge. SAML peut cependant être utilisé mais l'usage de message *ad hoc* imposera moins de contraintes. Qui plus est, ISABEL étant l'unique interlocuteur des différentes banques, la définition de ce message par ISABEL n'entraîne pas de problème de définition d'une structure commune (au sens où ISABEL est à même de définir cette structure pour tout le monde).

Il faut toutefois souligner que SAML propose l'authenticité, l'intégrité et la non répudiation des messages échangés. Si le canal de communication n'est pas sécurisé ou si la non répudiation revêt un caractère essentiel alors SAML apporte également un savoir-faire et une normalisation dont pourront bénéficier ISABEL et les banques. Malgré tout, ISABEL possède déjà une expertise dans le domaine de la non répudiation puisqu'elle a développé sa propre autorité de certification et qu'il s'agit là d'un procédé clé de ce type d'infrastructure.

Par contre, si on envisage d'utiliser SAML à plus grande échelle, notamment pour l'échange d'assertion d'autorisation concernant d'autres services ou d'autres ressources, la demande d'attributs spécifiques concernant une personne ou encore le SSO, l'utilisation de SAML est intéressante car elle apporte une standardisation dans la présentation des informations.

De plus, si ISABEL envisage de s'étendre au niveau européen, l'adoption d'un standard apporte également beaucoup d'intérêt pour ses interactions futures avec les autres organismes. L'utilisation de ce standard pourrait donc faciliter son introduction sur ces marchés et asseoir sa crédibilité.

En conclusion, étant donné qu'il ne fait aucun doute qu'ISABEL envisage de s'étendre sur la scène européenne, le choix de SAML concernant la multi signature dépendra donc de 2 facteurs: tout d'abord, l'extension ou non de l'infrastructure d'autorisation à d'autres fonctions et ensuite le temps disponible pour mettre en place les fonctionnalités de multi signature.

7.Bibliographie

[AKENTI]
M. Thompson, A. Essiari, S. Mudumbai,
"Certificate-based Authorization Policy in a PKI Environment,"
ACM Transactions on Information and System Security, to appear,
Aug 2003

Srilekha S. Mudumbai, Mary R. Thompson, Gary Hoo Abdeliah
Essiari, Keith Jackson, William Johnston
"Akenti – A Distributed Access Control System"

Mary R. Thompson
"Distributed Security Architectures"

http://www-itg.lbl.gov/Akenti/

[CARL]
Carlisle Adams
"Authorization Architecture"
http://www.win.tue.nl/~henkvt/Content.html

[MLORCH]
Rich Baker, Bob Cowles, Leon Gommans, Andrew McNab,
Markus Lorch, Lavanya Ramakrishnan, Krishna Sankar, Dane Skow,
Mary R. Thompson
"Conceptual Grid Authorization Framework and Classification"
http://www.ggf.org/documents/Drafts/

[PERMIS]
D.Chadwick,
"The PERMIS X.509 Role Based Privilege Management Infrastructure"
7th ACM Symposium on Access Control Models and Technologies
(SACMAT 2002).
www.permis.org

[PRIMA]
Lorch M., Adams D. B., Kafura D., Koneni M. S. R, Rathi A. , Shah S.
"The PRIMA System for Privilege Management, Authorization and
Enforcement in Grid Environments"
Department of Computer Science, Virginia Tech

[OBLIX]
"Oblix NetPoint SAML Services: Building Secure Online Partnerships A Technical Perspective"
http://www.oblix.com/resources/whitepapers/

[RFC2904]
J. Vollbrecht et al.,
"AAA Framework"
IETF RFC, August 2000
www.ietf.org

[RFC 3281]
S. Farrell, R. Housley,
"An Internet Attribute Certificate Profile for Authorization",
IETF RFC, April 2002
www.ietf.org

[SAML]
"Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1 Committee Specification",
18 juin 2003
http://www.oasis-open.org/committees/download.php/3406/oasis-%20sstc-saml-core-1.1.pdf

[SAMLBind]
Bindings and Profiles for the OASIS Security Assertion Markup Language (SAML) V1.1 4
Committee Specification, 18 July 2003
http://www.oasis-open.org/committees/download.php/3405/oasis-%20sstc-saml-bindings-1.1.pdf

[SITEMINDER]
"Netegrity SiteMinder 5.5
Technical White Paper "
http://www.netegrity.com/products/

[X-509]
ITU-T RECOMMENDATION X.509 v3| ISO/IEC 9594-8: .Information Technology . Open Systems Interconnection . The Directory: Public-Key And Attribute Certificate Frameworks.