Délégation d'agents pour la protection étendue des données personnelles

G. Piolle[a] Y. Demazeau[b]
guillaume.piolle@imag.fr yves.demazeau@imag.fr

[a]Grenoble INP et [b]CNRS,
Laboratoire d'Informatique de Grenoble,
110 avenue de la Chimie, Domaine universitaire, F-38400 Saint-Martin-d'Hères, France

Résumé

Le problème le plus délicat concernant la protection de la vie privée est la protection étendue des données, qui consiste à s'assurer qu'une information n'est pas utilisée d'une façon non autorisée par un agent distant. Nous analysons les travaux existants tentant de résoudre ce problème en fonction du niveau de confiance qu'ils permettent d'apporter aux utilisateurs, le niveau le plus élevé correspondant aux garanties fortes fournies par les méthodes du Trusted Computing [18]. Cette technologie étant paradoxalement dangereuse pour les libertés des utilisateurs, nous suggérons trois critères pour évaluer dans quelle mesure une architecture de Trusted Computing est profitable ou dangereuse pour un utilisateur. Sur cette base, nous proposons une nouvelle architecture de ce type (utilisant la délégation des processus et des données à des agents autonomes), assurant à la fois un haut niveau de confiance et une absence de risque pour l'utilisateur.

Mots-clés : *Confiance, Confidentialité, Sécurité, Interaction, Communication, Protocoles*

Abstract

The most difficult problem in the domain of privacy is the one of extended data protection, which consists in ensuring that an information is not processed in an unauthorized fashion by a distant agent. We analyze the existing works trying to address this issue with respect to the level of trust they can bring to users, the ultimate level being represented by the strong guaranteees provided by Trusted Computing techniques [18]. This technology being paradoxally dangerous for the users' liberties, we suggest three criteria for evaluating whether a TC-based architecture is fruitful or threatening to the user. On this basis, we propose a new TC-based architecture, ensuring both a high level of trust and an absence of risk for the user.

Keywords: *Trust, Privacy, Security, Interaction, Communication, Protocols*

1 Introduction

Lorsqu'un utilisateur est confronté à une application distribuée à laquelle il doit confier des données personnelles associées à une politique d'accès, la question qu'il se pose est la suivante : « Dans quelle mesure puis-je être sûr que lorsque j'aurai envoyé mes données, elles seront traitées par mes interlocuteurs dans le respect de la politique ? » Le problème de la protection de la vie privée et des données personnelles prenant de plus en plus d'importance pour le public (et se faisant de plus en plus présent avec l'évolution d'internet et de ses usages), de nombreuses solutions techniques ont été proposées pour apporter des éléments de réponse à cette question [10]. Néanmoins, nombre d'entre elles se concentrent sur son aspect local, se limitant à des problématiques de contrôle d'accès, alors que c'est la protection « étendue » des données (c'est-à-dire le contrôle de leur utilisation sur des plates-formes distantes) qui est la plus critique et la plus délicate à assurer. Des architectures applicatives distribuées ont été proposées dans ce sens, proposant des garanties plus ou moins fortes et induisant des risques plus ou moins acceptables pour l'utilisateur, comme nous allons le voir. Notre objectif ici est de concevoir une architecture complémentaire, proposant à la fois des garanties fortes et un risque nul pour l'utilisateur.

D'un point de vue inspiré par le paradigme multi-agent, nous considérons un agent utilisateur local, entité logicielle représentant son propriétaire humain, en interaction avec des agents de service distants, entités à même de fournir un produit numérique à l'utilisateur. Nous désignerons la fourniture de ce service par le terme générique de transaction.

Dans la section suivante, nous proposons une relecture de technologies existantes pour la protection des données personnelles en relation avec une notion de niveau de confiance. Dans la troisième section, nous analysons la techno-

logie correspondant au niveau de confiance le plus élevé, en dégageant des critères pour la construction d'architectures favorables à l'utilisateur. Nous proposons ensuite une architecture complémentaire de l'état de l'art existant, située au plus haut niveau de confiance et répondant aux critères énoncés. Dans la cinquième section nous discutons l'applicabilité et les caractéristiques de cette architecture, avant de conclure sur les perspectives liées à la technologie utilisée.

2 Travaux existants : les quatre niveaux de la protection étendue

Les moyens techniques visant à améliorer la protection des données personnelles peuvent être classés en fonction de l'entité dans laquelle la confiance de l'utilisateur doit être placée pourqu'il puisse être convaincu des garanties exprimées.

2.1 Confiance dans l'agent distant

Un grand nombre d'approches fondent leurs garanties sur la relation de confiance pouvant exister entre l'utilisateur et le logiciel distant. Si l'utilisateur est représenté ou modélisé par un agent cognitif, cette relation peut être représentée par la notion de confiance telle que définie dans le domaine [7], la valeur de confiance variant en fonction du résultat des interactions passées. Dans le cas de la protection des données personnelles, cette solution ne semble pas du tout adaptée, parce que les violations de politiques ou de normes, comme le fait qu'un agent conserve une donnée qui aurait dû être détruite, peuvent être silencieuses, indétectables par les autres agents [9]. Les valeurs de confiance seraient alors potentiellement déconnectées de ce qu'elles devraient représenter. L'illustration la plus simple de ce premier niveau de confiance est le cas des agents distants se réclamant d'une politique déclarative de type P3P [24], XACML *privacy policy* [16] ou SPARCLE [15]. Seule la déclaration d'intention de l'agent permet de se convaincre qu'il respectera effectivement la politique présentée.

Les « politiques collantes » (*sticky policies*) logicielles proposées par Karjoth et Schunter [15] se trouvent également à ce premier niveau de confiance. Ici, les données sont en permanence attachées à une politique d'utilisation préalablement négociée. Au sein de l'application distribuée, aucun agent n'accède à l'information sans d'abord vérifier que l'utilisation qu'il souhaite en faire est autorisée par la politique collante. De telles solutions sont expressives, mais elles reposent néanmoins sur la confiance que l'on peut avoir dans le logiciel distant : on suppose que les autres agents respectent toujours les spécifications de l'application et les politiques. Il reste possible d'introduire un agent apparemment compatible avec l'application, mais ignorant simplement les politiques collantes.

D'autres architectures s'appuient sur des créances présentées par les agents distants. Dans IDsec [14], RBAC [13] ou OrBAC [1], les agents doivent présenter des créances (destinées à montrer qu'ils vérifient un prédicat prédéterminé) avant de pouvoir accéder aux données. Cependant, dans tous les cas, une fois que l'information est transmise, plus aucun contrôle n'est effectué. Encore une fois, il faut avoir confiance dans le fait que l'agent n'utilisera pas les données en dépit de la politique et des créances fournies. Un cas particulier de l'utilisation de créances est celui où l'agent de service identifie de manière précise le composant logiciel qui va traiter les données, afin que l'utilisateur puisse déterminer si ce composant respecte ou non la politique en vigueur. Ceci peut être fait à partir d'une liste publique de logiciels certifiés, dont les caractéristiques de fonctionnement sont connues, et d'un condensat cryptographique signé par l'agent distant. Le niveau de sécurité est ici déterminé par le fait que l'agent utilisateur fonde sa confiance d'une part sur sa connaissance du composant logiciel utilisé (sa conviction a priori que celui-ci ne violera pas les normes ou la politique), et d'autre part sur l'agent qui certifie l'utiliser. Ce principe est illustré par la figure 1.

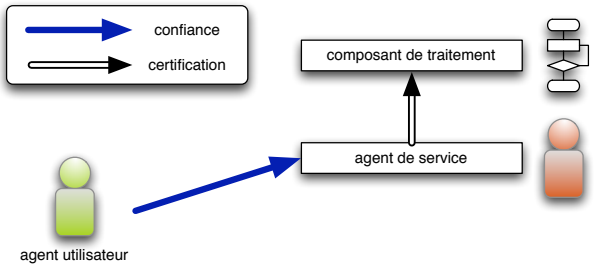

FIG. 1: Premier niveau de confiance de la protection étendue

Il existe de nombreuses autres approches pour lesquelles l'utilisateur ne peut croire que ses données ne seront pas utilisées à tort s'il n'a pas confiance dans tous les acteurs du système.

Ce niveau de confiance est particulièrement problématique en ceci que le récipiendaire de la confiance de l'utilisateur est précisément l'agent qui pourrait violer la politique et tirer profit de cette violation. Il y a donc un besoin de déplacer cette confiance vers une autre entité, plus sûre. Placer sa confiance dans les couches basses de l'agent distant peut être un moyen d'y arriver.

2.2 Confiance dans l'environnement d'exécution distant

Le niveau de sécurité suivant consiste à faire confiance à l'environnement d'exécution de l'agent distant, comme par exemple le système d'exploitation de la plate-forme distante. Cet environnement peut lui-même fournir des garanties sur le comportement d'un agent, par exemple en envoyant un condensat du code source de l'agent, de manière que l'agent utilisateur puisse le comparer avec une liste d'agents connus et certifiés comme respectant certains types de politiques. Le principe de ce deuxième niveau de confiance est illustré par la figure 2. Ici, la confiance de l'utilisateur repose à la fois dans l'autorité qui fournit la certification et dans le système d'exploitation distant : il est supposé que le système ne transmettra pas les données à un logiciel inconnu de l'utilisateur.

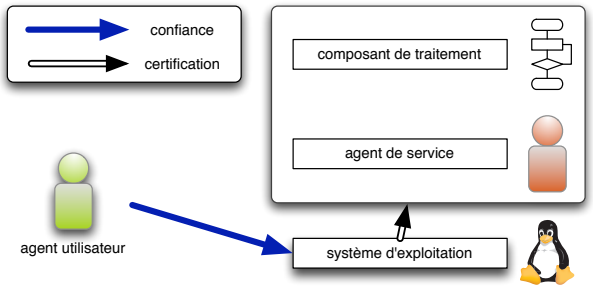

FIG. 2: Deuxième niveau de confiance de la protection étendue

Certaines architectures à base d'agents, non spécialisées dans la protection des données personnelles, reposent sur l'existence d'une autorité contrôlant le comportement des autres agents dans un environnement donné. Ceci est similaire en principe à une confiance dans l'environnement distant. C'est le cas des institutions électroniques [20, 11], dans lesquelles des agents mobiles s'exécutent sur une plate-forme générique, utilisant un *middleware* mettant à disposition des agents *staff* [12]. Ces agents *staff* ont une visibilité totale sur l'état interne des autres agents et peuvent ainsi détecter et sanctionner les violations. Ainsi, si l'on a confiance dans ce *middleware*, alors on peut accepter la croyance suivant laquelle les données seront utilisées en accord avec la politique déclarée.

Le niveau de sécurité augmente ici sensiblement, car pour contrefaire le type de garantie utilisée, le propriétaire de l'agent distant doit modifier le système d'exploitation lui-même, afin de le forcer à émettre de fausses déclarations sur la nature des logiciels exécutés. Si l'agent tiers représente un particulier, il est fort peu vraisemblable que ce dernier ait pu mettre en œuvre les moyens nécessaires à une telle manipulation. En revanche, dans le cas où l'agent tiers est sous le contrôle d'une grande entreprise ou d'une administration publique, il est raisonnable de supposer que ces entités ont les moyens de développer des systèmes d'exploitation personnalisés utilisant une version modifiée (possiblement malveillante ou défaillante) du composant chargé de l'identification des logiciels exécutés. Ainsi, même si le niveau de sécurité augmente à cette étape, il peut rester nécessaire de rechercher une sécurité supplémentaire, ne nécessitant pas de se reposer sur une confiance dans le système d'exploitation distant.

2.3 Confiance dans la plate-forme matérielle distante

Le troisième niveau consiste à faire confiance au matériel distant, plutôt qu'au système d'exploitation distant, en demandant au matériel de fournir et de signer un condensat du système d'exploitation avant de lancer ce dernier. Ceci permet d'éliminer les risques liés à la virtualisation. C'est par exemple l'une des fonctionnalités principales du *Trusted Platform Module* (TPM) [23], le composant matériel à la base des technologies du *Trusted Computing Group* [22]. Ce principe est illustré par la figure 3. À notre connaissance, aucune solution technique ne repose uniquement sur cette idée, mais elle sous-tend le quatrième niveau de sécurité.

Le risque de faillibilité d'une procédure de ce type est encore réduit par rapport au niveau précédent. Une entité souhaitant circonvenir ce protocole et fournir des garanties contrefaites devrait utiliser un composant matériel développé spécifiquement pour produire des certificats trompeurs. Les coûts engendrés sont donc encore supérieurs à ceux attachés à la modification du système d'exploitation, et la liste des agents susceptibles de mettre en œuvre ces moyens de contournement se réduit donc d'au-

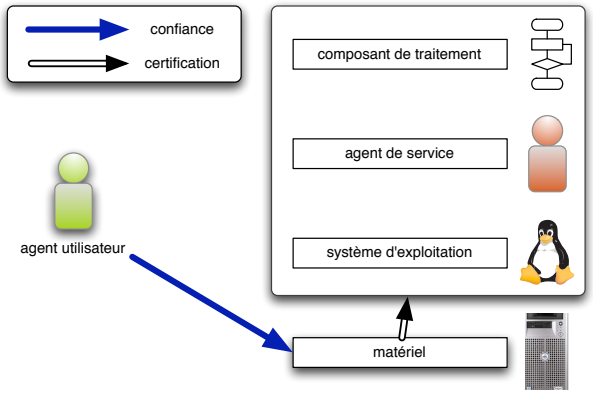

FIG. 3: Troisième niveau de confiance de la protection étendue

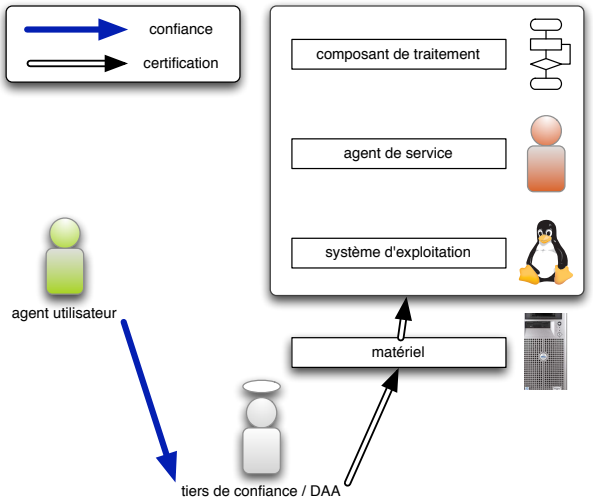

FIG. 4: Quatrième niveau de confiance de la protection étendue

tant. Cependant, on peut considérer que les données manipulées sont particulièrement sensibles et que les agents à qui elles sont confiées disposent de moyens particulièrement poussés. On peut donc envisager un dernier niveau de sécurité, pour lequel l'agent utilisateur n'aurait pas besoin d'avoir confiance dans le matériel distant.

2.4 Confiance dans une autorité indépendante

Au quatrième niveau de sécurité, on demande à une entité indépendante de fournir des garanties sur une plate-forme avant de lui demander des informations sur l'environnement d'exécution. C'est le principe même des *Trusted Computing Platforms* (TCP) [18], utilisant un TPM associé à une clé asymétrique unique dont la partie publique est certifiée par un tiers de confiance spécialisé. Un agent peut donc s'assurer qu'une plate-forme donnée est une TCP soit en se reposant sur un mécanisme de signature d'une clé de session par le tiers de confiance, soit en utilisant un protocole à diffusion nulle de connaissance, DAA (pour *Direct Anonymous Attestation* [4]). Les architectures de *Trusted Computing* tirent parti de ce dernier niveau de sécurité, dans lequel la confiance de l'utilisateur repose sur un tiers totalement étranger à la transaction en cours, répondant ainsi au problème principal posé par les niveaux de sécurité précédents. Ce principe est illustré par la figure 4.

Dans l'architecture d'auto-profilage proposée par Pearson [17], les agents utilisateurs construisent des informations de profil qui sont ensuite certifiées par le TPM local pour le compte d'un agent de service distant. Dans ce cas d'utilisation classique des TCP, les garanties sont fournies prioritairement à l'agent de service, et non à l'agent utilisateur. Une option du protocole autorise toutefois l'agent utilisateur à vérifier que l'agent distant est également une TCP, ouvrant ainsi la voie à une utilisation du *Trusted Computing* pour fournir des informations aux utilisateurs sur l'exploitation qui est faite de leurs données.

Une architecture ultérieure, proposée par Casassa Mont *et al*, implémente une version des politiques collantes utilisant le *Trusted Computing* [6]. Elle résout les problèmes qui avaient été pointés à propos des politiques collantes purement logicielles en permettant à l'agent utilisateur d'exiger que les agents de service soient des TCP et exécutent du logiciel certifié avant de les autoriser à accéder aux données.

À ce niveau il est possible d'être raisonnablement certain qu'une plate-forme distante exécute un logiciel prédéterminé (et aucun autre) pour traiter des données dans le contexte d'une application distribuée. C'est le seul niveau qui permette véritablement à l'agent utilisateur de fonder sa confiance sur une entité totalement indépendante, de manière que cette confiance ne soit pas entachée par l'intérêt de cette entité à de potentielles violations de la protection des données personnelles.

3 Caractérisation des architectures de *Trusted Computing*

Il a été justement remarqué que les solutions fondées sur le *Trusted Computing* pouvaient également être dangereuses pour l'utilisateur lui-même. Il n'est pas dans notre intention ici de raviver la controverse existante, mais des critiques objectives et plus ou moins virulentes ont été formulées par Siani Pearson [19], Ross Anderson [2] ou encore le groupe de travail de la Commission européenne sur la protection des données [3]. Il semble que les problèmes principaux liés aux TCP dépendent de trois paramètres : la localisation du TPM, la nature du code certifié et le protocole d'attestation choisi.

3.1 Localisation du TPM

Suivant l'architecture choisie, le TPM peut être localisé sur différentes plates-formes. Ces localisations distinctes correspondent à des caractéristiques différentes au niveau de la protection de l'utilisateur et des intérêts des diverses parties.

Sur la plate-forme utilisateur. Lorsque le TPM est situé côté utilisateur, ce dernier peut profiter des fonctionnalités de chiffrement de données offertes par la puce, et notamment la possibilité d'attacher les données à sa machine et à son environnement d'exécution (scellement de données). Néanmoins, les dispositifs de chiffrement fort sont d'une utilité limitée sur la plate-forme utilisateur, car c'est rarement cette dernière qui est l'objet d'une attaque directe. Les inconvénients à cette utilisation du TPM côté utilisateur semblent largement contrebalancer ses avantages. Le scellement de données est en lui-même dangereux pour l'utilisateur. En effet, c'est un processus qui peut être imposé par un fournisseur de contenu pour accéder à des données dont l'utilisateur peut pourtant être le propriétaire. C'est ce dispositif, au cœur de la mise en œuvre des DRM matériels, qui est pointé comme le plus liberticide [2] et qui a donné naissance aux controverses les plus significatives concernant cette technologie. Le scellement de données peut également être, au niveau économique, une entrave à la libre concurrence au sens où l'imposition du dispositif aux utilisateurs peut constituer (en fonction de la situation du marché) un outil d'aggravation de monopole. Ce biais profite aux éditeurs des logiciels et systèmes d'exploitation certifiés présents dans l'environnement d'exécution requis pour accéder aux données. Enfin, nous avons vu que le TPM, situé sur la plate-forme utilisateur, servait à produire des garanties à destination des autres parties, l'utilisateur lui-même n'en retirant que peu d'avantages. Dans une optique centrée sur l'utilisateur, la localisation du TPM sur la plate-forme de ce dernier ne semble donc pas justifiée, voire même dangereuse pour lui.

Sur la plate-forme de service. Lorsque le TPM est situé côté service, l'agent correspondant dispose des fonctionnalités de chiffrement de la puce (qui peuvent lui être plus utiles que sur un poste utilisateur). La position du TPM permet en outre de fournir des garanties à l'agent utilisateur sur les composants logiciels utilisés côté service, ce qui constitue, nous l'avons vu, la principale condition de la protection étendue des données. Cependant, ce type d'architecture impose une charge particulière à l'agent de service, notamment lorsque celui-ci doit maintenir un ensemble de logiciels certifiés. On observe également la même subordination de l'agent au TPM dans le cas du scellement de données. On peut néanmoins considérer que le risque associé est plus faible que lorsque le TPM est situé chez l'utilisateur, l'agent de service étant souvent en position de force par rapport à l'utilisateur sur le marché économique. Le TPM, lorsqu'il est situé sur la plate-forme de service, ne semble donc pas présenter de danger particulier pour l'utilisateur. Au contraire, elle lui permet d'obtenir des garanties fortes sur les propriétés distantes. En revanche, les agents de service peuvent être handicapés dans leurs activités par cette configuration.

Sur une plate-forme tierce. Une alternative non envisagée dans les architectures présentées est la localisation du TPM sur une plate-forme d'exécution neutre, ne dépendant ni de l'agent utilisateur ni de l'agent de service. Une telle plate-forme serait sous le contrôle d'un agent tiers n'ayant pas part à l'interaction entre l'agent utilisateur et les agents de service. La position du TPM permettrait alors de fournir des garanties à toutes les parties concernant les propriétés valides sur la plate-forme d'exécution. L'agent tiers opérant la plate-forme n'ayant pas d'intérêt dans les transactions, il est a priori insensible aux problèmes engendrés par le scellement des données.

La déportation des traitements sur une telle plate-forme « banalisée »peut donc être profitable aux deux parties en leur proposant des garanties sur les propriétés distantes tout en les li-

bérant des contraintes de la présence d'un TPM sur leurs propres plates-formes. L'agent tiers, opérant la plate-forme d'accueil, hérite toutefois de la responsabilité de maintenir du code certifié. Il paraît donc intéressant de caractériser cette charge et son impact sur la protection étendue des données.

3.2 Nature et variété du code à certifier

Dès lors que les composants de traitement utilisés par les agents de services sont variés et/ou privés, les agents utilisateurs peuvent avoir de bonnes raisons de douter de la fiabilité de leur certification.

Si le code source du composant est privé, alors les agents du système doivent forcément s'appuyer sur leur confiance dans l'autorité de certification pour accepter les assertions concernant ledit composant. Il existera donc toujours la possibilité d'une certification de complaisance, celle-ci ne pouvant être contrôlée. À l'inverse, si le code source est public, alors en théorie n'importe quel agent peut effectuer des vérifications dessus et remettre en question la fiabilité de sa certification. La publication d'informations sur la vulnérabilité d'une portion de code peut amener à la révocation de son certificat, à la correction des failles et à une nouvelle certification. Cette possibilité de vérification par la communauté est l'un des arguments majeurs de la communauté *open source* d'une manière générale, et son application au domaine de la sécurité informatique a souvent été illustrée.

D'autre part, lorsque la quantité et la variété du code à certifier augmentent, la difficulté à assurer la fiabilité du nombre de certificats correspondants augmente également. Plus le nombre de composants à certifier est élevé, plus le risque est élevé de voir apparaître à nouveau des certificats de complaisance, ou du moins fondés sur des analyses insuffisamment poussées. Au contraire, si le code à certifier est plus limité, si la sécurité de toute une gamme d'application peut ne reposer que sur une portion de code bien identifiée, alors ce risque est amoindri.

En conséquence, le cas le plus favorable à l'utilisateur est donc celui où le code à certifier est à la fois public et peu varié, se limitant à quelques composants génériques.

3.3 Protocole d'attestation utilisé

Il convient également de s'inquiéter du protocole utilisé pour la phase d'attestation, c'est-à-dire pour permettre à une TCP de s'identifier comme telle. Les alternatives actuelles sont l'attestation utilisant les tiers de confiance et le protocole DAA.

Le protocole utilisant les tiers de confiance est sensible à la collusion de ces derniers, mettant ainsi en péril l'anonymat des transactions effectuées par l'agent utilisateur. Cependant, lorsque l'agent utilisateur a toute latitude dans le choix de ces tiers de confiance, le risque peut diminuer en conséquence. L'utilisation du protocole DAA peut en théorie permettre de s'affranchir tout à fait de ce risque de subornation des autorités de confiance. Le groupe de travail européen sur la protection des données [3], dans son document de travail sur l'informatique de confiance, recommande d'ailleurs de faire de DAA le protocole d'attestation par défaut dans les prochaines versions des spécifications du TPM. Le protocole DAA serait donc plus avantageux pour la vie privée de l'utilisateur que le protocole classique utilisant les tiers de confiance.

4 Une architecture complémentaire par agents délégués

À la lumière de nos observations, nous proposons une nouvelle architecture exploitant les possibilités du *Trusted Computing*, afin de profiter du quatrième niveau de confiance pour la protection étendue, tout en évitant les écueils liés à cette technologie. Cette proposition, qui vient compléter l'offre existante, tient compte des axes de caractérisation que nous venons de mettre en lumière, afin d'offrir une orientation délibérément centrée sur la protection de l'utilisateur, quitte à poser des restrictions sur le type des traitements envisageables.

4.1 Architecture de base

Comme nous l'avons précédemment montré, la localisation du TPM sur une plate-forme d'exécution neutre comporte des avantages significatifs pour tous les agents prenant partie à la transaction. L'architecture que nous proposons suppose donc l'existence d'une telle plate-forme, constituant un environnement d'exécution mis à disposition par une entité neutre, n'ayant pas d'intérêt dans la transaction. Nous appellerons

cette plate-forme (et par métonymie son propriétaire) l'hôte de la transaction.

Le principe de l'architecture est le suivant :
– Phase préparatoire : un agent utilisateur, un agent hôte et un ou plusieurs agents de service s'accordent sur la fourniture d'un service numérique à l'agent utilisateur ;
– L'agent utilisateur et les agents de service projettent chacun un agent délégué sur la plate-forme hôte ;
– Phase d'exécution : les agents délégués interagissent entre eux au sein de la plate-forme hôte et sous son contrôle, afin de produire le service ;
– Phase de fermeture : les agents délégués transmettent à la plate-forme hôte le résultat de la production du service. La plate-forme transmet ce résultat à l'utilisateur et les agents délégués sont détruits.

La plate-forme hôte est une TCP sur laquelle s'exécutent un système d'exploitation, un ou plusieurs environnements d'exécution, autant d'agents d'interface que d'environnements d'exécution ainsi qu'un agent d'audit, tous certifiés à code public. Un environnement d'exécution différent est créé pour chaque nouvelle interaction entre agents mobiles. Il garantit que ces derniers ne pourront communiquer avec l'extérieur et qu'ils seront convenablement détruits lors de la phase de fermeture. L'agent d'interface est l'interlocuteur des agents utilisateurs et de service. Il assure la migration des agents délégués vers l'environnement d'exécution ainsi que le retour des résultats du calcul à l'agent utilisateur. Il est le seul lien entre l'environnement d'exécution et l'extérieur de la plateforme, la seule entité autorisée à communiquer. L'agent d'audit observe l'environnement d'exécution. Il effectue des mesures standardisées sur les agents délégués afin de détecter des comportements dangereux pour la sécurité de la plateforme et journalise les activités observables afin d'être en mesure de fournir des informations exploitables en cas de contentieux ultérieur entre les parties. Il est le garant du fonctionnement sain de la plate-forme hôte et a ainsi autorité pour mettre fin au traitement en déclenchant la fermeture de l'environnement d'exécution.

La principale caractéristique de cette architecture est l'assurance, fournie par la plate-forme hôte et mise en œuvre par ses divers composants, que les agents délégués ne pourront pas communiquer avec l'extérieur et qu'ils seront détruits lors de la phase de fermeture. En conséquence, dans cette version de base de l'architecture, seul l'agent utilisateur bénéficie d'un retour. Les agents de service ne sont en mesure de retirer (ni donc de conserver) aucune information provenant de l'agent utilisateur.

La figure 5 présente les principes de cette architecture. Nous allons maintenant détailler plus avant chacune des phases du protocole applicatif associé à l'architecture.

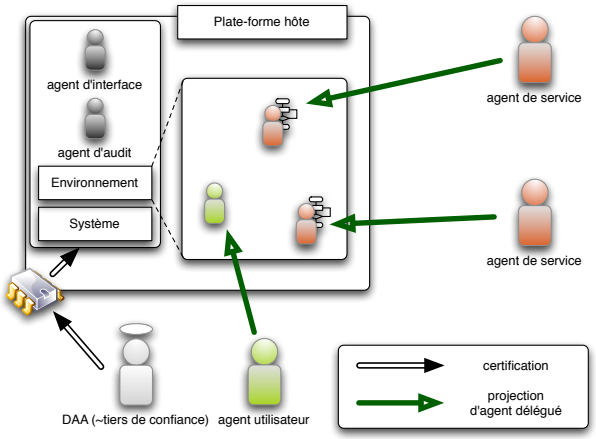

FIG. 5: Architecture par agents délégués

Phase préparatoire. Le prérequis à la mise en œuvre du protocole est la disponibilité d'une TCP utilisant un BIOS, un système d'exploitation et une suite logicielle d'accueil (comprenant agent d'audit, agents d'interface et environnements d'exécution) certifiés, à code public (l'ensemble de l'environnement logiciel correspondant à une valeur de registre connue, pouvant être signée par le TPM).

On suppose que l'application concerne un agent utilisateur (situé sur l'un des terminaux d'un utilisateur humain) et un ensemble d'agents de service (situés sur des serveurs distants). Durant la phase préparatoire, agent utilisateur et agents de service s'accordent sur le service à rendre et négocient une politique d'utilisation de données personnelles associée. À l'issue de cette négociation, les agents partagent un identifiant unique produit aléatoirement pour la transaction ($IDtrans$), l'identité d'une plate-forme d'accueil publique et une politique de traitement des données personnelles.

Les agents procèdent ensuite à l'attestation de la plate-forme et à la vérification de l'intégrité de son environnement logiciel. Lorsque les agents font leurs premières requêtes à la plateforme, en mentionnant l'identifiant de transaction $IDtrans$, cette dernière peut produire une

identité virtuelle (autre fonctionnalité native du TPM) dédiée à la transaction. L'attestation et la vérification de l'intégrité de l'environnement logiciel de la plate-forme par chacun des agents s'effectuent via DAA. La description détaillée de cette étape, classique pour une TCP, dépasse le cadre de cette publication.

La plate-forme hôte est informée de l'intention des agents de procéder au traitement $IDtrans$ et prend connaissance de la liste des agents participants. Un environnement d'exécution et un agent d'interface dédiés à la transaction sont alors invoqués. Les agents fixes, localisés sur leurs plates-formes respectives, transmettent alors chacun à l'agent d'interface de la plate-forme hôte un composant logiciel spécifique, qui sera l'agent délégué les représentant.

Phase d'exécution. Durant la phase d'exécution, les agents délégués interagissent entre eux comme bon leur semble au sein de l'environnement d'exécution. Dans la version de base de l'architecture, les agents délégués sont projetés avec toutes les informations dont ils ont besoin pour produire le contenu qui constituera le service rendu à l'utilisateur. Les agents délégués n'ont pas la possibilité de communiquer avec l'extérieur.

Pendant cette phase, l'agent d'audit observe les interactions entre agents délégués et décide, sur la base d'informations dépendant de la conception de la plate-forme, si un comportement anormal est détecté. Notamment, si les spécifications de la plate-forme imposent un langage de communication particulier entre les agents délégués, toute tentative de l'un deux pour entrer en contact avec un autre agent délégué autrement que par ce langage sera considéré comme une attaque[1]. L'observation de métriques classiques sur le fonctionnement des agents, comme l'exploitation anormale des ressources de la plate-forme ou une absence d'activité prolongée, peut également conduire à la caractérisation d'une anomalie de comportement de l'un des agents délégués. Lorsque ces comportements anormaux surviennent, ils sont journalisés par l'agent d'audit. Ce dernier demande à l'agent d'interface d'informer les parties que la transaction a échoué à la suite d'une manœuvre non autorisée de la part de l'un des agents (permettant ainsi, dans une certaine mesure, l'utilisation de méthodes de régulation sociale)[2]. L'agent d'audit force ensuite la fermeture de l'environnement d'exécution.

Si la transaction s'effectue correctement, chacun des agents délégués fournit à l'agent d'interface un résultat à transmettre à l'agent utilisateur.

Phase de fermeture. Lorsque l'agent d'interface dispose des rapports de tous les agents délégués, il transmet les résultats fournis à l'agent utilisateur, informe les agents de service de la terminaison correcte de la transaction puis appelle la méthode de fermeture de l'environnement d'exécution, qui détruit tous les agents délégués ansi que l'agent d'interface. Les agents mobiles sont donc détruits sans avoir eu l'occasion de communiquer avec l'extérieur, et en particulier avec leur agent source.

4.2 Améliorations optionnelles

L'architecture et le protocole associé, que nous avons présentés dans leur version de base, comportent des restrictions très fortes. Il est possible de lever certaines d'entre elles en introduisant des options dans le protocole.

Autoriser le paiement d'un service. L'extension qui semble probablement la plus évidente est celle qui permet le paiement du service par l'utilisateur. Une telle fonctionnalité augmenterait considérablement le champ d'application de cette architecture, l'étendant à certaines formes de commerce électronique. Les principes du *e-Cash* mis en place depuis les années 1980 permettent déjà le paiement anonyme d'un service. Il est aisé ici de proposer un prototype pour une option de paiement.

Il nous faut introduire un tiers de confiance spécialisé, à savoir un établissement bancaire. On suppose que l'agent correspondant est connu des parties et dispose d'un couple de clés asymétriques $(BKpub, BKpriv)$. Lorsque l'identifiant de la transaction $IDtrans$ est partagé entre les parties, l'agent représentant la banque est informé de l'association entre le montant convenu et la valeur $h(IDtrans)$ par l'agent

[1] Il existe quelques travaux d'évaluation des dispositifs de sécurité disponibles sur les environnements multi-agents les plus utilisés, comme l'étude effectuée par Axel Bürkle *et al* en 2006 [5], qui identifie des pratiques « saines » de protection mutuelle d'une plate-forme d'accueil et d'agents mobiles.

[2] Ce retour d'information peut permettre une transmission de données par canal caché, en programmant des agents de service pour faire volontairement échouer la transaction lorsque certaines circonstances sont réunies (comme l'apparition d'une donnée particulière). Cela peut être en partie évité par des mesures de régulation sociale visant à ostraciser les agents présentant trop de dysfonctionnements.

qui doit percevoir le versement (h désignant une fonction de condensat cryptographique). L'agent utilisateur peut procéder au paiement auprès de l'agent bancaire, en associant cette même valeur $h(IDtrans)$ à son versement. L'agent bancaire lui fournit en retour un certificat anonyme estampillé de la date courante T (1), à faire valoir auprès des agents de service, établissant que le montant convenu pour cette transaction particulière a bien été réglé.

$$\begin{array}{l}\texttt{payé}, h(IDtrans), \mathrm{T}, BKpub, \\ \{\texttt{payé}, h(IDtrans), \mathrm{T}\}_{BKpriv}\end{array} \quad (1)$$

Ainsi, la vérification de certificats de paiement par les agents fournisseurs de service peut être un prérequis à la projection des agents délégués et à la phase d'exécution. De la même manière, il est possible de mettre en place le paiement de frais de service à l'agent opérant la plate-forme d'accueil. Dans tous les cas les paiements peuvent rester anonymes du point de vue des agents prenant part à la transaction (par opposition au point de vue des établissements bancaires, qui maintiennent la nécessaire traçabilité des paiements).

Demandes de clés publiques. Une deuxième extension consiste à alléger la contrainte d'isolation des agents délégués pour les autoriser à demander des clés publiques par le biais de l'agent d'interface. Un agent délégué peut ainsi faire une demande à l'agent d'interface de l'environnement, qui prendra en charge l'exécution d'une requête paramétrée et la transmission du résultat au demandeur. Une application de ce type de requête est l'évaluation au sein même de l'environnement d'exécution des certificats de paiement, mais les communications entre agents délégués et le fonctionnement interne de ces derniers peuvent impliquer d'autres utilisations de clés publiques. Une plate-forme hôte peut éventuellement définir d'autres requêtes paramétrables, mais la certification de l'environnement logiciel hôte devra alors prendre en compte ces requêtes afin de déterminer leur absolue neutralité vis-à-vis des données personnelles impliquées dans la transaction entre agents mobiles. Le caractère paramétrable de ces requêtes doit être limité pour éviter les transmissions d'informations par canal caché. Notamment, la liste des tiers de confiance auxquels il est possible de faire des requêtes doit être fixée par la plate-forme hôte et être évaluée lors de sa certification. En effet, si un agent délégué peut demander une clé publique à n'importe quel serveur, alors il est possible que le serveur choisi (complice d'un contournement de l'isolation logique de l'environnement d'exécution) permette une transmission d'information bilatérale : l'agent mobile peut remplacer l'identité de l'entité dont il demande la clé publique par des données chiffrées, le serveur peut répondre en remplaçant la clé publique par d'autres données chiffrées.

Autoriser des requêtes vers l'extérieur. L'interfaçage par la plate-forme hôte de requêtes des agents mobiles vers l'extérieur peut être généralisé, dans une certaine mesure. En 2007, Richard Cissée et Sahin Albayrak ont proposé, dans le contexte d'une architecture de recherche d'information, une méthode pour effectuer des demandes de données depuis une plate-forme isolée en préservant le caractère privé des requêtes et de leurs résultats [8]. Cette méthode s'appuie sur le chiffrement et déchiffrement croisés (via un algorithme de *one-time pad*) des requêtes et de leurs réponses par les deux parties prenant part à une transaction. Elle assure que les serveurs auxquels les requêtes sont adressées ne sont pas capables d'identifier les requérants ni d'effectuer de corrélations entre les requêtes. Le protocole proposé peut être intégré à notre architecture, au prix de nouvelles fonctionnalités (notamment cryptographiques) pour l'agent d'interface. Les agents délégués devront également tous disposer de certaines fonctionnalités de chiffrement spécifiques pour que l'un d'eux puisse émettre des requêtes par le biais de ce protocole. Cette nouvelle option permet de lever une autre limitation majeure de l'architecture de base, qui imposait que les agents mobiles soient projetés avec toutes les informations nécessaires à la transaction. Grâce à cette option, les agents mobiles peuvent désormais se reposer sur des bases de données extérieures pour des opérations de recherche d'information.

Conservation de données sur la plate-forme. Le protocole peut également être étendu pour permettre la conservation d'informations sur la plate-forme hôte (autres que les données de nature statistique collectées par l'agent d'audit). Une attention particulière doit être apportée au problème si l'on autorise les agents délégués à faire des requêtes de stockage et de récupération de données à l'agent d'interface. Une telle fonctionnalité permettrait d'enrichir les services fournis à l'utilisateur (en assurant la persistance de son profil ou de ses préférences, par exemple) et d'étendre le champ d'applicabilité de l'architecture. Néanmoins, deux problèmes majeurs se posent.

Tout d'abord, la responsabilité de la plate-forme par rapport à la protection des données deviendrait véritablement énorme. En effet, lorsque des données personnelles sont stockées, le responsable de l'hébergement a l'obligation d'assurer la protection de ces données [21]. En l'absence de cette option, les agents de la plate-forme peuvent dans une certaine mesure ignorer la sémantique des informations échangées. Si l'agent d'interface devient responsable de l'accès à des données personnelles stockées sur la plate-forme, alors ce dernier doit pleinement comprendre la nature de ces données, s'assurer de manière fiable de l'identité des agents souhaitant les stocker ou y accéder, comprendre le détail des politiques de traitement de ces données, s'assurer de la persistance de ces politiques entre deux transactions (alors même que l'agent d'interface n'est pas persistant) et enfin appréhender un contexte normatif pour s'assurer que son propre comportement n'est pas délictueux. Le pas à franchir est énorme en matière de complexité de la plate-forme, et ces nouvelles fonctionnalités seraient extrêmement délicates à valider au cours de la phase de certification de l'environnement logiciel.

D'autre part, la plate-forme d'accueil étant conçue comme un service « bien connu » d'un certain nombre d'agents, celle-ci deviendrait une cible privilégiée pour des agents mal intentionnés souhaitant collecter des données sensibles de manière déloyale. Une architecture proposant cette option retomberait dans les travers de propositions comme IDsec [14] en mettant en avant un point faible unique et en limitant le contrôle des agents utilisateurs sur leurs données personnelles.

En conséquence, il nous semble qu'une telle fonctionnalité constituerait au final un risque trop élevé pour la vie privée des utilisateurs de l'architecture.

5 Discussion

5.1 Intérêt et mise en œuvre

L'intérêt de l'architecture que nous avons proposée repose principalement sur une utilisation conjointe des TCP et de la notion d'agents mobiles, ici délégués par des agents utilisateurs ou de service. La possibilité de se reposer sur du logiciel autonome permet de déléguer la responsabilité du traitement d'informations personnelles et de la négociation du service, ce qui autorise la déportation des traitements sur une plate-forme neutre. L'utilisation d'une TCP permet quant à elle de fournir des garanties fortes à tous les participants à la transaction. Cette dernière fonctionnalité manquait à des propositions comme les institutions électroniques. Ici, la conjonction des deux technologies permet de se reposer sur la certification matérielle tout en évitant les risques habituellement associés au *Trusted Computing*.

Nous pensons que la meilleure piste d'implémentation est la réutilisation du *middleware* existant pour les institutions électroniques [12], en l'adaptant notamment pour en faire une TCP. L'éditeur de politique des institutions électroniques semble en effet suffisamment expressif pour décrire les spécifications de notre plate-forme hôte, en représentant l'environnement d'exécution et les agents d'interface ou d'audit par les *scenes* et les *staff agents*, respectivement.

5.2 Limitations et applicabilité

L'architecture proposée présente des limitations claires (au moins dans sa version de base). La plus significative est la non-transmission d'informations aux agents de service, entraînant l'impossibilité d'une quelconque persistance entre les sessions (en-dehors de celle assurée par l'agent utilisateur lui-même). Notre objectif n'était pas de concevoir un outil exploitable en toute situation, mais bien de montrer qu'il y avait une zone inexplorée dans la famille des architectures de *Trusted Computing*, qui pouvait, associée à des concepts agents, présenter de bonnes propriétés en matière de vie privée. Malheureusement, cette architecture ne peut être utilisée lorsque les données personnelles doivent être conservées côté service ou lorsque ces informations doivent nécessairement être communiquées à un tiers, pour la livraison physique d'un produit par exemple. Elle peut également être peu séduisante pour les fournisseurs de service. Cela est dû à notre point de vue dogmatique de centrage sur l'utilisateur et à notre objectif, maintenant atteint : cette architecture fait le choix le plus favorable à l'utilisateur pour chacun des critères relatifs aux TCP.

À titre d'illustration, nous proposons un court exemple applicatif dans lequel l'utilisation de cette architecture d'application est possible. Nous considérons pour cela un agent utilisateur `uAgent` et deux agents de service `pressAgent` et `adAgent`, représentant respectivement un site de presse en ligne et une régie publicitaire. L'utilisateur souhaite

ici profiter d'un service numérique fourni par `pressAgent`, consistant en la consultation (suite à un paiement forfaitaire) d'articles de presse parus dans les trois derniers jours. Parallèlement, `adAgent` a un accord avec `pressAgent` pour présenter à ses lecteurs des publicités en rapport avec les thématiques des articles consultés.

– Phase préparatoire : `uAgent` demande à accéder au service, et les trois agents s'accordent sur l'utilisation d'une plate-forme hôte. `uAgent` procède au paiement du service par le biais de l'extension décrite plus haut, puis les trois agents projettent des agents délégués sur la plate-forme d'accueil. `del_uAgent` est projeté avec la liste des articles auxquels l'utilisateur souhaite accéder, `del_pressAgent` avec le texte des articles des trois derniers jours et `del_adAgent` avec un ensemble de publicités correspondant à diverses thématiques. La plate-forme les accueille dans un environnement d'exécution créé pour l'occasion.
– Phase d'exécution : `del_uAgent` effectue sa requête auprès de `del_pressAgent`, qui sélectionne les articles et interagit avec `del_adAgent` pour y intégrer des publicités.
– Phase de fermeture : le contenu produit (articles émaillés de publicités) est transmis à l'agent d'interface et les trois agents délégués annoncent la fin de la transaction. La plate-forme transmet ce contenu à `uAgent` et les trois agents délégués sont détruits avec l'environnement d'exécution.

Ce court scénario illustre la possibilité de recourir à cette architecture pour des applications de personnalisation de contenu numérique. Ici, les centres d'intérêt de `uAgent` sont demeurés strictement privés, alors qu'un contenu et des publicités personnalisés ont pu lui être fournis contre paiement. Cet exemple met également en valeur l'intérêt des extensions autorisant le recours à des bases de données externes, qui permettraient à `del_pressAgent` et `del_adAgent` d'être projetés sans contenu superflu (autorisant ainsi par exemple la consultation d'un ensemble plus vaste d'articles). La proposition de Cissée et Albayrak [8] peut être applicable dans ce cas-là. Elle résulterait en des demandes d'articles (faites par `del_pressAgent` à `pressAgent` après contrôle par `u_adAgent` et par la plate-forme) qui ne pourraient être reliées à aucun agent utilisateur ni aucune plate-forme en particulier (à condition d'utiliser des techniques d'anonymisation IP [10]), ni même être corrélées entre elles. L'acceptabilité de telles requêtes dépend cependant de l'application et de la densité de son utilisation. Il faut en effet qu'il existe suffisamment d'exécutions simultanées du service (ou que les exécutions soient rendues suffisamment longues) pour qu'une requête ne puisse pas être reliée à la demande de service préalable.

6 Conclusion

Nous avons proposé une architecture d'application distribuée préservant la vie privée, fondée sur des concepts agents et reposant sur le niveau de confiance identifié comme le plus sûr en ce qui concerne la protection des données personnelles, c'est-à-dire sur le fait de placer sa confiance dans une entité totalement indépendante. Pour cela nous avons utilisé les technologies du *Trusted Computing*, mais uniquement après avoir déterminé comment rendre le système totalement favorable à l'utilisateur.

En conclusion, nous pensons nécessaire de mettre en valeur le nouveau compromis apparaissant lorsque le *Trusted Computing* est utilisé pour atteindre le quatrième niveau de confiance : il semble pour le moins difficile pour une architecture d'être à la fois utilisable dans la plupart des situations et bien positionnée en regard des critères que nous avons dégagés pour les TCP. Ceci, ajouté à la charge de travail significative associée à la certification de logiciel, nous amènent à penser que le *Trusted Computing* n'est probablement pas le meilleur candidat pour la réalisation d'applications préservant la vie privée. Même si les trois premiers niveaux de confiance reposent sur la « sécurité par la complexité », qui n'est pas la solution la plus élégante pour améliorer la vie privée, l'avenir des technologies de protection des données personnelles repose probablement sur des technologies moins exigeantes et moins risquées.

Références

[1] A. Abou El Kalam, R. El Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miège, C. Saurel et G. Trouessin. Organization based access control. In *4th International Workshop on Policies for Distributed Systems and Networks (POLICY 2003)*, Italie, 2003.

[2] R. Anderson. "Trusted Computing" frequently asked questions, August 2003. `http://www.cl.cam.ac.uk/~rja14/tcpa-faq.html`.

[3] Article 29 Data Protection Working Party. Working document on trusted computing platforms and in particular on the work done by the trusted computing group (TCG group). Working Document of the European Commission, January 2004. 11816/03/EN – WP 86.

[4] E. Brickell, J. Camenisch et L. Chen. Direct anonymous attestation. In *11th ACM conference on Computer and communications security (CCS'04)*, pages 132–145, USA, 2004.

[5] A. Bürkle, A. Hertel, W. Müller, and M. Wieser. Evaluating mobile agent platform security. In K. Fischer, I. J. Timm, E. André et N. Zhong, éd., *4th German Conference on Multiagent System Technologies (MATES 2006)*, Allemagne, 2006.

[6] M. Casassa Mont, S. Pearson et P. Bramhall. Towards accountable management of identity and privacy : Sticky policies and enforceable tracing services. HP Laboratories Bristol, 2003. HPL-2003-49.

[7] C. Castelfranchi et R. Falcone. Principles of trust for multi-agent systems : Cognitive anatomy, social importance, and quantification. In *Third International Conference of Multi-Agent Systems (ICMAS'98)*, Paris, 1998.

[8] R. Cissée et S. Albayrak. An agent-based approach for privacy-preserving recommender systems. In *6th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'07)*, Hawaii, 2007.

[9] L. Crépin, G. Piolle, O. Boissier et Y. Demazeau. Des systèmes normatifs comme outils de protection de la vie privée. In *Intelligence Artificielle et Web Intelligence (IAWI'07)*, Grenoble, 2007.

[10] Y. Deswarte et C. Aguilar-Melchor. Current and future privacy enhancing technologies for the internet. *Annals of Telecommunications*, 61(3-4), 2006.

[11] M. Esteva, J. A. Rodríguez-Aguilar, C. Sierra, P. Garcia et J. L. Arcos. On the formal specification of electronic institutions. In F. Dignum et C. Sierra, éd., *Agent-mediated Electronic Commerce (The European AgentLink Perspective).*, GB, 2001.

[12] M. Esteva, B. Rosell, J. A. Rodríguez-Aguilar et J. . Arcos. Ameli : An agent-based middleware for electronic institutions. In *Third International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS'04)*, USA, 2004.

[13] D. Ferraiolo et R. Kuhn. Role-based access controls. In *15th NIST-NCSC National Computer Security Conference*, 1992.

[14] Internet Engineering Task Force. IDsec : Virtual identity on the internet. `http://idsec.sourceforge.net/`.

[15] G. Karjoth et M. Schunter. A privacy policy model for enterprises. In *15th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, 2002.

[16] Organization for the Advancement of Structured Information Standards. *Privacy policy profile of XACML v2.0*, 2005.

[17] S. Pearson. Trusted agents that enhance user privacy by self-profiling. In *AAMAS-02 Workshop on Deception, Fraud and Trust in Agent Societies*, Italie, 2002.

[18] S. Pearson. *Trusted Computing Platforms : TCPA Technology in Context*. Prentice Hall PTR, USA, 2002.

[19] S. Pearson. Trusted computing : Strengths, weaknesses and further opportunities for enhancing privacy. In P. Herrmann, V. Issarny et S. Shiu, éd., *Third International Conference on Trust Management (iTrust 2005)*, 2005.

[20] C. Sierra et P. Noriega. Institucions electròniques. In *Primer Congrès Català d'Intelligència Artificial*, 1998.

[21] The European Parliament and the Council. Directive 1995/46/EC of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. In *Official Journal of the European Communities*, 1995.

[22] Trusted Computing Group. `https://www.trustedcomputinggroup.org/home`.

[23] Trusted Computing Group. TCG TPM specification, version 1.2. http ://www.trustedcomputinggroup.org/, 2003.

[24] World Wide Web Consortium. *Platform for Privacy Preferences specifications 1.1*. http ://www.w3.org/P3P/, 2002.