ANR programme ARPEGE 2008

Systèmes Embarqués et Grandes Infrastructures

Projet SELKIS : Une méthode de développement de systèmes d'information médicaux sécurisés : de l'analyse des besoins à l'implémentation.

ANR-08-SEGI-018

Février 2009 - Décembre 2011

Projet SELKIS ANR-08-SEGI-018 Modèles formels pour l'analyse et l'expression des exigences de sécurité

Livrable 1.1

Editeur : Telecom Bretagne/SERES

T0 + 18

Table des matières

Résumé

Le présent rapport constitue le premier livrable de la tâche WP1 du projet SELKIS. Il comporte trois chapitres. L'objectif de l'étude menée dans le premier chapitre est d'analyser les différentes normes qui peuvent être définies par un législateur à propos des informations qu'un système d'information peut communiquer à un agent. Pour cela la démarche consiste à analyser les effets qu'un acte de communication réalisé par un système d'information peut avoir sur un agent et à analyser les effets et actions que le législateur peut choisir de permettre ou d'interdire. La technique utilisée pour faire cette analyse consiste à exprimer les notions pertinentes en logique modale. Notons que ce chapitre correspond à un document de travail en cours qui n'est pas sous une forme à proprement parlé rédigée. L'actualisation de ce chapitre est prévue au cours du dernier trimestre du projet. Le second chapitre s'intéresse à la dérivation semi-automatique des exigences de sécurité sous forme d'une politique de sécurité. L'idée est de partir d'une spécification Kaos du système considéré. Cette spécification est par la suite, dans la mesure du possible, expurgée des aspects fonctionnels et transformée en une spécification Kaos restreinte aux aspects de sécurité. La spécification obtenue est enrichie par les données d'une analyse de risques menée sur le système. Finalement, la dernière étape du processus consiste à dériver à partir de cette spécification Kaos transformée la politique de sécurité OrBAC (Organisation Based Access Control) correspondante. Le dernier chapitre s'intéresse à l'expression formelle et la mise en oeuvre d'un contrôle de sécurité *a posteriori*. Une approche nécessaire lorsqu'on s'intéresse à des domaines d'application relatif à la santé comme c'est le cas dans le cadre du projet SELKIS. L'idée est d'avoir une politique plus flexible qui ne bloque pas forcément les actions entreprises par les utilisateurs du système mais leur fait confiance tout en les responsabilisant. Un système de trace et un processus d'audit sont mis en place pour la détection des violations et le recueil des preuves. Ils sont généralement accompagnés de réactions et de sanctions éventuelles.

Le cas d'étude considéré dans les chapitre 2 et le chapitre 3 dans le cadre de Selkis est celui de l'Echodoppler géré par notre partenaire Medecom en collaboration avec le CHU de Brest. Cependant, pour des besoins d'illustration de certains aspects, c'est l'exemple de prêts dans une bibliothèque, qui sera utilisé dans le chapitre 2 de ce livrable. L'application de nos travaux à l'étude de cas médicale sera présentée dans une version actualisée prévue du présent livrable.

Chapitre 1

Définition des normes sur la confidentialité

1.1 Préambule

Objectif. Analyser les différentes normes qui peuvent être définies par un législateur à propos des informations qu'un système d'information peut communiquer à un agent.

On considère que ce sont les normes définies par le législateur qui servent de référence ultimes pour les vérifications et non pas les spécifications définies par les informaticiens.

On appelle "législateur" l'organisme qui définit les normes.

Méthode.

1. Analyser tous les effets d'un acte de communication physique réalisé par le système d'information sur les connaissances, ou croyances, d'un agent qui est le destinataire de cet acte.
2. Etudier les actions et/ou effets qu'un législateur peut décider de permettre ou d'interdire.

Technique. Utiliser la logique modale comme langage pour exprimer les relations entre notions qui interviennent dans les normes sans préciser la sémantique formelle des modalités (cette sémantique peut être définie dans une étape ultérieure).

Notations.

IS : agent qu'on appelle le "système d'information" et qui réalise l'acte de communication.

i : agent qui est le destinataire d'un acte de communication de IS.

Commentaires :

- IS peut être un agent institutionnel (par exemple un hôpital) ou un agent humain (par exemple un médecin) ; on étudiera d'abord le cas où IS est un agent humain et ensuite le cas où c'est un agent institutionnel.

- i peut être un agent humain (par exemple un patient) ou un agent institutionnel (par exemple une caisse d'assurance maladie).

1.2 Communication directe entre agents humains

On considère d'abord le cas où IS et i sont deux agents humains qui communiquent oralement.

Bien qu'on s'intéresse *in fine* aux communications qui utilisent comme instruments des moyens informatiques, il nous parait important de clarifier dans un premier temps la façon dont on peut définir les normes dans une communication orale.

1.2.1 Effets d'un acte de communication

Un acte de communication peut être défini à un niveau physique (acte locutoire dans la terminologie de Searle) ou à un niveau abstrait (acte illocutoire dans la terminologie de Searle).

On analyse la relation entre les actes de communication au niveau physique (locutoire) et leurs effets possibles sur le destinataire : effets locutoires, illocutoires et éventuellement perlocutoires.

Modalités utilisées.

$E_{a:\alpha}\phi$: l'agent a a fait en sorte, en réalisant l'action α, que l'on ait ϕ.

$Bel_a\phi$: l'agent a croit que ϕ est vraie.

$Know_a\phi$: l'agent a sait que ϕ est vraie.

$Int_a\phi$: l'agent a a l'intension que ϕ soit vraie.

$mess(a, m)$: il y a un lieu physique auquel l'agent a peut accéder et où se trouve l'objet physique m qui représente un message.

$Perm\phi$: il est permis que l'on ait ϕ.

$Obg\phi$: il est obligatoire que l'on ait ϕ.

$Forb\phi$: il est interdit que l'on ait ϕ.

Commentaires :

- L'opérateur $E_{a:\alpha}$ exprime la causalité au sens de Kanger, Pörn, Jones... et quand $E_{a:\alpha}\phi$ est vraie l'action α a été réalisée.

- $Know_a\phi$ signifie que a a une justification pour savoir que ϕ est vraie, et donc $Know_a\phi$ n'est pas définie comme le fait qu'on a : $\phi \wedge Bel_a\phi$ car il se pourrait que a croit, sans justification, que ϕ est vraie et qu'il se trouve incidemment que ϕ soit vraie.

- On donne à $Int_a\phi$ une signification analogue à celle de Cohen et Levesque.

- Dans $mess(a, m)$, m désigne l'objet physique qui est utilisé pour représenter le message auquel a peut accéder. Par exemple, m peut désigner une chaîne de bits située sur un support informatique à un emplacement donné auquel a peut (au sens de la capacité physique, pas de la permission) accéder, m peut aussi désigner un document papier qui se trouve dans la boite aux lettres de a, si on considère des communications par le courrier traditionnel. Dans le cas d'une communication orale entre personnes $mess(a, m)$ signifie qu'un signal sonore qui

peut être entendu par a a été émis, et que ce signal sert de support au message communiqué.

- Dans les modalités précédentes on peut avoir : $agt : r$ à la place de a si on veut exprimer qu'un agent agit en tant que titulaire d'un rôle, ou qu'il sait, ou croit, quelque chose en tant que titulaire d'un rôle. Dans ce cas $E_{agt:r:\alpha}\phi$ se lie : l'agent agt a fait en sorte, en réalisant l'action α en tant que titulaire du rôle r, que l'on ait ϕ. De même $Bel_{agt:r:\alpha}\phi$ se lie : l'agent agt croit en tant que titulaire du rôle r que ϕ est vraie (idem pour les connaissances). Les croyances de agt en tant que titulaire du rôle r sont les croyances que agt a acquis en agissant en tant que titulaire du rôle r ou qui lui ont été communiquées en tant que titulaire du rôle r ou qu'il a déduit des précédentes.

L'acte de communication locutoire est représenté par :

(1) $E_{IS:\alpha}mess(i, m)$

Signification intuitive de (1) : IS a envoyé le message m à i en émettant certains sons.

Les effets de cet acte dépendent de certaines hypothèses qu'on peut accepter ou rejeter.

L'hypothèse que si un agent réalise un acte de communication le destinataire sait que cet acte a eu lieu est représentée par :

(TRANS) $E_{IS:\alpha}mess(i, m) \rightarrow Know_i E_{IS:\alpha}mess(i, m)$

Ici, cette hypothèse signifie que i a effectivement entendu les sons émis par IS et il sait que c'est IS qui les a émis.

De (1) et (TRANS) on déduit :

(2) $Know_i E_{IS:\alpha}mess(i, m)$

Signification intuitive de (2) : i sait que IS lui a envoyé m.

L'hypothèse : i sait que IS en transmettant à i le message m a voulu que i sache que IS croit ϕ est représentée par :

(SEM) $Know_i(E_{IS:\alpha}mess(i, m) \rightarrow Int_{IS}Know_i Bel_{IS}\phi)$

(SEM) signifie, entre autre, que i sait que la signification que IS donne à m est représentée par la proposition ϕ. Cette hypothèse est particulièrement importante quand, par exemple, IS et i parlent des langues différentes ou quand IS utilise des messages dont la signification est cryptée (par exemple les messages radio transmis à la résistance pendant la guerre). Elle établie la relation entre un phénomène physique et la signification qui lui est attribuée par IS.

Pour Searle l'effet locutoire de l'acte $E_{IS:\alpha}mess(i, m)$ serait représenté par $Bel_i Int_{IS}Know_i Bel_{IS}\phi$. On verra plus loin pourquoi on a choisi d'exprimer l'effet locutoire en terme de connaissance plutôt que de croyance.

De (2) et (SEM) on déduit :

(3) $Know_i Int_{IS}Know_i Bel_{IS}\phi$

Signification intuitive de (3) : i sait que IS veut que i sache que IS croit ϕ.

L'hypothèse : i sait que IS est sincère quand IS veut lui communiquer ϕ est représentée par :

(SINC) $Know_i(Int_{IS}Know_i Bel_{IS}\phi \rightarrow Bel_{IS}\phi)$

Comme pour (SEM)on verra plus loin pourquoi dans (SINC) on suppose que i sait, et non pas croit, que IS est sincère.

De (3) et (SINC) on déduit :

(4) $Know_i Bel_{IS}\phi$
Signification intuitive de (4) : i sait que IS croit ϕ.

1.2.2 Choix des normes

Le but des normes relatives à la confidentialité est de restreindre les connaissances d'un agent concernant les informations que possède (que croit) un système d'information.

Comme ce que connaît un agent humain (ici le destinataire du message) ne peut pas faire l'objet d'une observation, et donc d'un contrôle direct, le législateur ne peut que définir des normes qui visent à contrôler les actions pouvant éventuellement avoir comme effets les connaissances qu'il veut restreindre (il faudrait se faire confirmer cette position par des juristes).

Première question : quelles sont les connaissances du destinataire que veut restreindre le législateur ?

Deuxième question : quelles sont les actions qui peuvent avoir pour effet ces connaissances, et sous quelles hypothèses ces effets peuvent-ils être obtenus ?

Réponse à la première question.

Ce sont des connaissances de la forme : $Know_i Bel_{IS}\phi$.

Pourquoi pas : $Know_i Know_{IS}\phi$? Parce qu'il peut se faire que le législateur veuille restreindre ce que IS croit même si ce que IS croit n'est pas justifié, ou même si ce que IS croit est faux. Par exemple, si la signification de ϕ est : le patient x a un cancer du type y, il se peut que ce diagnostic soit faux, mais même si le diagnostic est faux tout le monde n'est pas nécessairement autorisé pour autant à savoir que IS croit que ce diagnostic est vrai.

Pourquoi pas : $Bel_i Bel_{IS}\phi$? Parce que le législateur n'a pas à restreindre les inférences fausses, ou sans justification, que peut faire i à partir des actes de communication de IS. Par exemple, si le législateur veut restreindre la transmission d'un diagnostic de cancer et que i croit, à tort, que la signification de m est : le patient x a un cancer du type y, et qu'en réalité la signification que IS donne à m est : le patient x a une infection du type z, alors le législateur n'a pas de raison de restreindre la transmission du message m.

Pourquoi pas : $Bel_i Know_{IS}\phi$? Pour les deux raisons précédentes.

Réponse à la deuxième question.

Ce sont les actions de la forme $E_{IS:\alpha} mess(i, m)$.

Pourquoi pas : $E_{IS:\alpha} Know_i Bel_{IS}\phi$? Parce que l'action physique α ne peut pas garantir l'effet $Know_i Bel_{IS}\phi$ qui est une attitude mentale du destinataire. Par exemple, cet effet n'est pas obtenu si l'hypothèse de sincérité (SINC) n'est pas satisfaite, c'est-à-dire si i ne croit pas que IS est sincère. Mais, alors pourquoi restreindre une action du type $E_{IS:\alpha} mess(i, m)$ si l'effet $Know_i Bel_{IS}\phi$ n'est pas garanti par cette action ? Parce que l'action
$E_{IS:\alpha} mess(i, m)$ donne à i la possibilité de savoir que IS croit ϕ et que le législateur doit envisager que cette possibilité se réalise, même si ce n'est pas nécessairement le cas.

Plus généralement le législateur doit chercher à garantir la confidentialité dans le "pire" des cas, c'est-à-dire quand toutes les hypothèses (TRANS), (SEM)

et (SINC) sont satisfaites. C'est pour cela que les restrictions doivent porter sur $E_{IS:\alpha}mess(i,m)$, et que l'on considère dans ces hypothèses que les attitudes mentales de i sont des connaissances et pas simplement des croyances.

Conclusion : les normes sont de la forme $PermE_{IS:\alpha}mess(i,m)$ ou $ForbE_{IS:\alpha}mess(i,m)$.

1.3 Communication entre agents humains via des agents logiciels

On considère maintenant que IS et i sont des agents humains qui communiquent par l'intermédiaire d'agents logiciels.

Agents logiciels. Définition sommaire.

On appelle agent logiciel un support informatique physique qui sert de représentation à un ensemble d'instructions et de données. Les actions que réalise l'agent logiciel correspondent à l'exécution de ces instructions dans cet environnement physique.

Les normes ne peuvent pas porter sur des agents logiciels qui sont des objets physiques. Les agents logiciels ne sont que des instruments déclenchés par des agents humains.

Analogie.

Le code de la route interdit aux voitures de rouler à plus de 130 km/h sur autoroute. En réalité cette interdiction s'applique aux conducteur des voitures. Si la vitesse d'une voiture est déterminée par un régulateur automatique de vitesse, c'est le conducteur, en imposant une vitesse au régulateur, qui est responsable de la vitesse.

Les règles de la navigation aérienne interdisent aux avions de voler en dessous de certaines altitudes dans certaines zones. Cette interdiction s'applique aux pilotes des avions, pas aux avions eux-mêmes. Si un avion est piloté par un pilote automatique, c'est le pilote en affichant une altitude, ou une procédure de descente particulière sur le pilote automatique, qui est responsable de l'altitude de l'avion.

Si une personne utilise un pigeon voyageur pour porter à quelqu'un d'autre un message qui contient une information qu'il n'a pas la permission de transmettre, ce n'est pas le pigeon qui est responsable...

Conclusion : Il n'y a pas de norme qui s'applique à un agent logiciel, et si on parle de normes qui s'appliquent à un système d'information, ce qu'on désigne par "système d'information" dans ces normes ne peut être un agent logiciel. Si un agent humain utilise un agent logiciel pour transmettre une information, les normes considèrent que c'est l'agent humain qui a transmis l'information.

Le sens qu'on donne à "un agent humain utilise un agent logiciel" est que l'agent humain a réalisé une action qui est la **cause** des actions réalisées par l'agent logiciel, ou plus informellement que l'agent humain a "déclenché" l'exécution de l'agent logiciel.

1.3.1 Effets d'un acte de communication

Si IS utilise pour communiquer l'agent logiciel l, l'acte de communication réalisé par IS est représenté par :

(1) $E_{IS:\beta}E_{l:\alpha}mess(i,m)$

Signification intuitive de (1) : en réalisant l'action β l'agent IS a fait en sorte que l en réalisant α a fait en sorte que l'on ait $mess(i,m)$. L'action β pourrait consister, par exemple, à remplir certains champs de l'écran d'un terminal avec des données qui expriment le contenu de m, l'expéditeur IS, le destinataire i et le lieu où doit être rangé m, puis à cliquer sur une zone qui déclenche la réalisation de l'action α par l.

Ici m désigne une chaîne de bits située sur un support informatique à un emplacement donné auquel i peut (au sens de la capacité physique, pas de la permission) accéder via un agent logiciel k.

On accepte le schéma d'axiome :

(ET) $E_{a:\alpha}\phi \rightarrow \phi$

D'après (1) et (ET) on a :

(2) $E_{l:\alpha}mess(i,m)$

L'hypothèse (TRANS) doit être reformulée ici de la façon suivante :

(TRANS') $E_{l:\alpha}mess(i,m) \rightarrow Know_i\exists x\ mess(i,x)$

Intuitivement $Know_i\exists x\ mess(i,x)$ signifie que i sait qu'il y a un message dans sa boite. Cette hypothèse est forte car même si l émet un signal sonore conventionnel pour indiquer à i qu'il a reçu un message, il n'est pas garanti que i ait entendu ce signal, soit parce qu'il y a du bruit autour de lui, soit parce qu'il n'est pas physiquement présent. Ou bien, si une convention dit que i doit regarder périodiquement sur un écran s'il a reçu un message, il n'est pas garanti que i respecte cette convention.

D'après (2) et (TRANS') on a :

(3) $Know_i\exists x\ mess(i,x)$

On peut aussi considérer qu'au lieu de l'hypothèse (TRANS') on a :

(TRANS") $E_{IS:\beta}E_{l:\alpha}mess(i,m) \rightarrow Know_iE_{IS:\beta}E_{l:\alpha}\exists x\ mess(i,x)$

Pour accepter cette hypothèse il faut que l transmette à i l'information que c'est l qui a transmis le message et que c'est IS qui a déclenché l'envoie.

On fait également l'hypothèse que quand on a (3) i décide d'accéder au message. Pour cela il réalise l'action β' qui a pour effet que l'agent logiciel k réalise l'action α' qui elle-même a pour effet que i connaît le message m. Soit :

(4) $E_{i:\beta'}E_{k:\alpha'}Know_imess(i,m)$

A partir de (4) et (ET) on a :

(5) $E_{k:\alpha'}Know_imess(i,m)$

A partir de (5) et (ET) on a :

(6) $Know_imess(i,m)$

On fait l'hypothèse que i sait que la signification du message m est ϕ et qu'il a été envoyé indirectement par IS.

(SEM') $Know_i(mess(i,m) \rightarrow Int_{IS}Know_iBel_{IS}\phi)$

Remarque : la différence entre (SEM') et (SEM) vient de ce que dans la communication orale le message m n'exprime pas, en général, l'identité de celui

qui parle.

A partir de (6) et (SEM') on a :

(7) $Know_i Int_{IS} Know_i Bel_{IS}\phi$

A partir de (7) et (SINC) on a :

(8) $Know_i Bel_{IS}\phi$

Si au lieu de (TRANS') on accepte (TRANS") on doit accepter (SEM") à la place de (SEM) :

(SEM") $Know_i(E_{IS:\beta}E_{l:\alpha}mess(i,m) \rightarrow Int_{IS}Know_i Bel_{IS}\phi)$

A partir de (1), (TRANS") et (SEM") on déduit (4) de façon similaire.

On accepte également l'hypothèse (SINC') :

(SINC') $Know_i(Int_{IS}Know_i Bel_{IS}\phi \rightarrow Bel_{IS}\phi)$

D'après (4) et (SINC') on a :

(8) $Know_i Bel_{IS}\phi$

1.3.2 Choix des normes

Les connaissances que veut restreindre le législateur sont les mêmes que dans le cas d'une communication directe entre deux agents humains, à savoir : $Know_i Bel_{IS}\phi$.

Les actions qui peuvent avoir pour effet $Know_i Bel_{IS}\phi$ et que peut envisager de réglementer le législateur sont donc de la forme : $E_{IS:\beta}E_{l:\alpha}mess(i,m)$.

On notera que ce n'est pas parce que i sait qu'il a un message dans sa boite aux lettres (3) que nécessairement il réalise l'action qui lui permet de connaître le message qui s'y trouve (4) mais le législateur doit envisager qu'il réalise effectivement cette action.

Le choix du législateur porte alors uniquement entre :

$PermE_{IS:\beta}E_{l:\alpha}mess(i,m)$ et $ForbE_{IS:\beta}E_{l:\alpha}mess(i,m)$

Alors, intuitivement, il est soit permis, soit interdit, que IS déclenche l'envoie du message m dont la signification est ϕ à i par l'intermédiaire de l'agent logiciel l.

1.4 Communication entre agents institutionnels via des agents humains et logiciels

On considère maintenant que IS et i sont des agents institutionnels qui communiquent via des agents humains qui les représentent, et ces agents humains communiquent via des agents logiciels.

Agents institutionnels.

On utilise le terme "agent institutionnel" dans le même sens que "personne morale" par opposition à "personne physique".

Un agent institutionnel est une entité abstraite dont les propriétés sont définies par une institution. On utilise le terme "institution" dans le sens de Searle, c'est-à-dire un ensemble de normes considérées comme un tout. Par exemple, le

droit du commerce international constitue une institution. Le droit des entreprises en France est aussi une institution. La réglementation interne d'un hôpital est une autre institution.

Un agent institutionnel étant une entité abstraite il ne peut pas réaliser d'actions physiques. Il ne peut agir que par l'intermédiaire d'agents humains dont les actions "comptent comme" (dans le sens du "counts as " de Searle) des actions de l'agent institutionnel vis-à-vis d'une institution. En général, c'est parce que l'agent humain est titulaire d'un certain rôle que certaines de ses actions comptent, dans certaines circonstances, comme des actions de l'agent institutionnel.

Par exemple, il se peut que si une personne titulaire du rôle de responsable financier d'un hôpital transmette à une caisse d'assurance maladie une information indiquant que l'hôpital a fait tel acte médical pour le patient x et que cela a coûté y, alors cette action de communication compte, vis-à-vis de l'institution qui réglemente les systèmes de santé, comme si c'était l'hôpital qui avait réalisé cet acte de communication. Mais, si le même acte de communication est réalisé par une infirmière, ou bien si le responsable financier n'a pas agi en tant que responsable financier, l'action de communication ne compte pas comme une action de communication réalisée par l'hôpital.

Conclusion : il y a des normes qui s'appliquent aux agents institutionnels. Pour savoir si une norme qui s'applique aux actions d'un agent institutionnel est satisfaite ou violée on doit considérer les actions des agents humains qui comptent comme des actions de cet agent institutionnel. On dira parfois que pour ces actions ces agents humains "représentent" l'agent institutionnel.

1.4.1 Effets d'un acte de communication

On suppose maintenant que IS est un agent institutionnel, que l'agent humain j "représente" IS parce que j est titulaire du rôle r dans IS, et que j utilise l'agent logiciel l pour transmettre l'information à i.

On suppose également que l'agent i est un agent institutionnel, que l'agent humain h "représente" i parce qu'il est titulaire du rôle r' dans i, et que h utilise l'agent logiciel k pour accéder aux messages destinés à i.

On considère deux scénarios. Dans le premier c'est le représentant j de IS qui transmet au représentant h de i un message qui donne à h la possibilité de savoir que IS croit ϕ. Dans le second c'est h qui va accéder directement à ce qui sert de support aux croyances de IS, ce qui lui donne la possibilité de savoir que IS croit ϕ. Le premier scénario correspond au cas où le système informatique qui gère les croyances de IS est utilisé conformément à un usage "normal", le second correspond au cas où h utilise, par exemple, des techniques d'intrusion qui ne sont pas conforme à un usage "normal".

On pourrait envisager un troisième scénario où le représentant j de IS introduit directement le message m qui représente ϕ dans ce qui sert de support aux croyances de i. Cependant ce scénario est plutôt à traiter dans le cadre des normes qui réglementent la modification des croyances d'un agent, c'est-à-dire dans le cadre des normes qui concernent l'intégrité.

Scénario 1.

On suppose que j a réalisé l'action suivante :

(1) $E_{j:r:\beta}E_{l:\alpha}mess(i,m)$

Ici pour interpréter la signification de $mess(i,m)$ on présuppose qu'il existe un lieu physique où sont rangés les messages destinés à l'agent institutionnel i.

Les effets de cette action sont les suivants.

D'après (1) et (ET) on a :

(2) $E_{l:\alpha}mess(i,m)$

(TRANS') $E_{l:\alpha}mess(i,m) \rightarrow Know_{h:r'}\exists x\ mess(i,x)$

Bien que le message soit destiné à l'agent institutionnel i c'est l'agent humain h qui peut percevoir physiquement les effets de l'action $E_{l:\alpha}mess(i,m)$. De plus on suppose qu'en accédant à une zone physique qui est assignée à i, h agit en tant que titulaire de r', c'est-à-dire en tant que représentant de i. Les connaissances qu'il acquiert de cette façon sont des connaissance qu'il acquiert en tant que titulaire de r'.

D'après (2) et (TRANS') on a :

(3) $Know_{h:r'}\exists x\ mess(i,x)$

Comme dans le cas des communications entre agents humains via des agents logiciels, on suppose qu'à la suite de (3) h réalise l'action suivante pour prendre connaissance du message m :

(4) $E_{h:r':\beta'}E_{k:\alpha'}Know_{h:r'}mess(i,m)$

A partir de (4) et (ET) on a :

(5) $E_{k:\alpha'}Know_{h:r'}mess(i,m)$

A partir de (5) et (ET) on a :

(6) $Know_{h:r'}mess(i,m)$

On fait l'hypothèse que h sait, en tant que titulaire de r', que la signification du message m est ϕ et qu'il a été envoyé indirectement par IS.

(SEM') $Know_{h:r'}(mess(i,m) \rightarrow Int_{IS}Know_iBel_{IS}\phi)$

On notera que l'intention que h attribue à IS c'est que i, et non pas h, sache que IS croit ϕ. En effet, IS, ou l'agent humain j qui agit pour le compte de IS, adresse le message à i et il n'a pas à savoir quel est le représentant de i qui va accéder au message m.

Pour justifier (SEM'), on peut faire le parallèle avec une information qui est transmise de façon traditionnelle par lettre. Dans ce cas c'est La Poste qui est l'analogue de l. La lettre est signée par j et la lettre indique le rôle de j en mentionnant, par exemple, "responsable financier" au dessus de la signature, et le fait que cette lettre compte comme une lettre envoyée par tel hôpital est représenté par le fait que la lettre a l'entête de cet hôpital. Donc, quand h relève la lettre qui a été déposée dans la boite aux lettres de i il sait que cette lettre a été envoyée par j et que compte tenu du rôle que joue j dans IS il représente IS, et donc que l'intention de j "est" l'intention de IS. D'autre part, bien que ce soit h qui relève la lettre, le destinataire de la lettre est i, et non pas h. Enfin, on suppose que h sait "déchiffrer" la lettre m et que sa signification est ϕ. C'est pour toutes ces raisons qu'on a $Int_{IS}Know_iBel_{IS}\phi$ dans (SEM').

A partir de (6) et (SEM') on a :

(7) $Know_{h:r'}Int_{IS}Know_iBel_{IS}\phi$

On suppose que h sait que IS est sincère.
(SINC') $Know_{h:r'}(Int_{IS}Know_iBel_{IS}\phi \rightarrow Bel_{IS}\phi)$
A partir de (7) et (SINC') on a :
(8) $Know_{h:r'}Bel_{IS}\phi$
Scénario 2.
On suppose que h réalise l'action :
(1') $E_{h:r':\beta''}E_{k_1:\alpha''}mess(i,m)$
La signification intuitive de (1') est que h déclenche un agent logiciel k_1 qui va chercher directement le message m là où sont enregistrées les croyances de IS.
D'après (1') et (ET) on a :
(2') $E_{k_1:\alpha''}mess(i,m)$
(TRANS2) $E_{k_1:\alpha''}mess(i,m) \rightarrow Know_{h:r'}\exists x\ mess(i,x)$
D'après (2') et (TRANS2) on a :
(3) $Know_{h:r'}\exists x\ mess(i,x)$
Ensuite le raisonnement se poursuit comme dans le scénario 1.

1.4.2 Choix des normes

Les connaissances que veut restreindre le législateur sont les mêmes que dans le cas d'une communication entre deux agents humains, à savoir : $Know_iBel_{IS}\phi$.

Dans le cas où IS et i sont des agents institutionnels ils n'agissent pas eux-mêmes, mais le législateur d'un système de santé ne veut pas savoir comment les normes qu'il veut imposer à IS (respectivement i) se répercutent dans l'organisation interne de IS (respectivement i) en termes de normes qui portent sur les agents humains qui représentent IS (respectivement i). Pour cela il peut imposer des normes à des actions qu'on appelle "actions institutionnelles", c'est-à-dire qui n'ont pas été réalisées par les agents institutionnels, mais que l'institution s considère "comme" réalisées par les agents institutionnels.

Pour préciser ces notions on utilise la logique JS du counts as de Jones et Sergot avec les opérateurs suivants.

$D_s\phi$: vis-à-vis de l'institution s on a ϕ.

$\phi \Rightarrow_s \psi$: vis-à-vis de l'institution s le fait qu'on ait ϕ compte comme si on avait ψ.

Scénario 1.
Le fait que j représente IS s'exprime par[1] :
(H1) $(E_{j:r:\beta}E_{l:\alpha}mess(i,m)) \Rightarrow_s (E_{IS:\alpha}mess(i,m))$
Donc si on a $E_{j:r:\beta}E_{l:\alpha}mess(i,m)$, on en déduit à partir de (H1) dans la logique JS : $D_sE_{IS}mess(i,m)$.

Pour permettre ou interdire l'action $E_{j:r:\beta}E_{l:\alpha}mess(i,m)$ il suffit que le législateur permette ou interdise que l'on ait : $D_sE_{IS}mess(i,m)$, soit que l'on ait :

$PermD_sE_{IS}mess(i,m)$ ou $ForbD_sE_{IS}mess(i,m)$

[1]Noter que (H1) n'est pas un schéma d'axiome. Dans (H1) i, j, l, IS ...etc... sont des constantes.

On notera que ces normes sont indépendantes de l'agent humain qui représente IS et de l'action particulière qu'a réalisé cet agent et elles sont également indépendantes de l'agent logiciel qu'il a déclenché et de l'action réalisée par cet agent logiciel.

On a vu que sous certaines hypothèses si on a $E_{j:r:\beta}E_{l:\alpha}mess(i,m)$ on a également $Know_{h:r'}Bel_{IS}\phi$, et si on a $Know_{h:r'}Bel_{IS}\phi$, alors h a la possibilité de faire une action[2] en tant que titulaire de r' qui aura pour effet $Know_iBel_{IS}\phi$. Donc si le législateur veut restreindre $Know_iBel_{IS}\phi$ c'est bien sur $D_sE_iKnow_imess(i,m)$ que doit porter la permission ou l'interdiction.

On notera que même si l'action de h ne conduit pas a une violation des normes de l'institution s il pourrait se faire que pour des raisons internes à l'organisation de i il n'y ait que certains représentant de i qui aient la permission d'accéder à la "boite aux lettres" de i et qu'il soit interdit à h d'y accéder.

Pour faire la distinctions entre les normes du système de santé exprimées par l'institution s, et les normes internes à i exprimées par une institution qu'on appelle int on peut indexer les permissions et interdictions de la façon suivante :

$Perm_s\psi$, $Forb_s\psi$, $Perm_{int}\psi$ et $Forb_{int}\psi$

On peut alors avoir : $Perm_sD_sE_iKnow_imess(i,m)$ et
$Forb_{int}E_{h:r':\beta'}E_{k:\alpha'}Know_{h:r'}mess(i,m)$.

Dans ce cas $E_{h:r':\beta'}E_{k:\alpha'}Know_{h:r'}mess(i,m)$ satisfait $Perm_sD_sE_iKnow_imess(i,m)$ et viole $Forb_{int}E_{h:r':\beta'}E_{k:\alpha'}Know_{h:r'}mess(i,m)$. Intuitivement h n'a pas violé la norme de s mais il a violé la réglementation interne de int.

Scénario 2.

Le fait que h représente i s'exprime par :

(H2) $(E_{h:r':\beta''}E_{k_1:\alpha''}mess(i,m)) \Rightarrow_s (E_imess(i,m))$

A partir de $E_{h:r':\beta''}E_{k_1:\alpha''}mess(i,m)$ et (H2) on en déduit dans la logique JS : $D_sE_imess(i,m)$

Pour permettre ou interdire l'action $E_{h:r':\beta''}E_{k_1:\alpha''}mess(i,m)$ il suffit que le législateur permette ou interdise que l'on ait : $D_sE_imess(i,m)$, soit que l'on ait :

$PermD_sE_imess(i,m)$ ou $PermD_sE_imess(i,m)$

Comme dans le scénario 1, sous certaines hypothèses, à partir de $E_{h:r':\beta''}E_{k_1:\alpha''}mess(i,m)$ il est possible de déduire $Know_iBel_{IS}\phi$, donc pour restreindre $Know_iBel_{IS}\phi$ c'est bien sur $D_sE_imess(i,m)$ que doivent porter la permission ou l'interdiction.

1.5 Conslusion : Questions en suspens

Indices de l'opérateur d'action. Faut-il mettre le nom de l'action qui a été réalisée dans les opérateurs tels que : $E_{j:r:\beta}$ ou $E_{l:\alpha}$?

Pour $E_{j:r:\beta}$ la réponse est certainement "oui" car deux actions de j peuvent produire le même effet, alors que quand il réalise l'une de ces actions il agit en tant que titulaire de r, tandis que quand il réalise l'autre il n'agit pas en tant

[2]Intuitivement, cette action consisterait à modifier ce qui sert de représentation physique aux croyances de i.

que titulaire de r. Par exemple, il se peut que j ait fait en sorte qu'une certaine facture ait été payée et que pour qu'il l'ait payée en tant que titulaire de r il faut qu'il l'ait payée par chèque mais que s'il l'a payée en liquide il n'a pas agit en tant que titulaire de r.

Pour $E_{l:\alpha}$ l'argument précédent n'est pas valable car un agent logiciel ne peut pas être titulaire d'un rôle (ce n'est pas un "sujet de droit"). Cependant il se pourrait que $E_{j:r:\beta}E_{l:\alpha}mess(i,m)$ compte comme $E_{IS:\alpha}mess(i,m)$, mais que $E_{j:r:\beta}E_{l:\delta}mess(i,m)$ ne compte pas comme $E_{IS:\alpha}mess(i,m)$, et que pour certaine raison il faille préciser l'action réalisée par l. Par exemple, si α correspond à un usage "normal" du système informatique de IS alors que δ est une action de "piratage".

Action illégale en tant que titulaire d'un rôle. Selon certains juristes une personne ne peut pas prétendre avoir agit en tant que titulaire d'un rôle reconnu par une institution si l'action qu'il a réalisé viole une norme de cette institution. Par exemple, si l'agent comptable de l'ONERA fait un chèque destiné à payer un pot de vin pour que l'ONERA obtienne un contrat il ne pourra pas prétendre avoir fait ce chèque en tant qu'agent comptable de l'ONERA. Donc, si l'action $E_{j:r:\beta}E_{l:\alpha}mess(i,m)$ viole une norme on ne peut pas dire que j a agit en tant que r. (à vérifier auprès de juristes).

Représentation matérielle des croyances d'un agent institutionnel. Est-ce que, par définition, les "croyances" d'un agent institutionnel sont entièrement représentées sur des supports physiques, ou déduites de celles-ci, ou bien est-ce qu'elles peuvent inclure les croyances de certaines personnes qui sont des représentants de cet agent institutionnel ? (à demander à des juristes).

Représentation des intentions d'un agent institutionnel

Dans (SEM') on fait référence à Int_{IS}. Est-ce que l'on peut dire que les intentions d'un agent institutionnel sont constituées des intentions de n'importe lequel de ses représentants, ou bien que ses intentions doivent avoir une représentation physique bien définie comme pour les croyances ?

Utilisation "normale" d'un système informatique. Comment définir la notion d'utilisation "normale" ?

Par exemple, si, par définition, les croyances de l'agent institutionnel sont constituées des informations enregistrées dans une certaine BD et gérées par un certain SGBD, on peut dire qu'une utilisation "normale" consiste à accéder à la BD en utilisant uniquement des commandes du SGBD, tandis qu'une utilsation "anormale" consiste à utiliser en plus, ou uniquement, d'autres programmes que le SGBD pour accéder à la BD.

Répercussion des normes dans l'organisation d'un agent institutionnel. Dans le cas où une interdiction porte sur un agent institutionnel, par exemple $ForbD_sE_{IS}mess(i,m)$, il se peut que cette interdiction soit répercutée par la réglementation interne en une interdiction qui porte sur les agents humains titulaires de certains rôles dans IS. La question est alors de savoir dans quels cas le respect de ces normes internes suffit à garantir que les normes qui portent sur l'agent institutionnel sont respectées.

Des normes sur des états à des normes sur des actions

Si le but du législateur est de définir des normes sur $Know_iBel_{IS}\phi$ ($PermKnow_iBel_{IS}\phi$

ou $ForbKnow_iBel_{IS}\phi$), comment ces normes se traduisent en des normes portant sur des agents dont les actions sont susceptibles d'avoir comme effet $Know_iBel_{IS}\phi$?

D'où la question : quels peuvent être ces agents et ces actions ?

On a vu une réponse : l'agent IS en faisant une action qui a pour effet $mess(i, m)$. D'où on peut déduire les normes de la forme $PermD_sE_{IS}mess(i, m)$ ou $ForbD_sE_{IS}mess(i, m)$. Et comme ces actions ne peuvent être réalisées par IS lui-même le législateur peut en déduire des normes de la forme :
$PermE_{j:r:\beta}E_{l:\alpha}mess(i, m)$ ou $ForbE_{j:r:\beta}E_{l:\alpha}mess(i, m)$
pour tous les agents j tels que l'on ait :
$(E_{j:r:\beta}E_{l:\alpha}mess(i, m)) \Rightarrow_s (E_{IS:\alpha}mess(i, m))$

Cette réponse convient dans le cas où le législateur ne s'intéresse qu'à réglementer les actions de IS, mais elle n'est pas nécessairement complète. Il peut y avoir d'autres agents ayant la capacité de faire des actions qui ont pour effet $Know_iBel_{IS}\phi$.

Ca peut être le cas d'un agent tel que j qui réalise une action de la forme $E_{j:r:\beta}E_{l:\alpha}mess(i, m)$ mais qui n'est pas représentant de IS, c'est-à-dire pour lequel on n'a pas $(E_{j:r:\beta}E_{l:\alpha}mess(i, m)) \Rightarrow_s (E_{IS:\alpha}mess(i, m))$, et qui est sous la responsabilité de IS, dans le sens où si l'action de cet agent viole la norme c'est IS qui doit en assumer les conséquences. Comment définir plus précisément ce type d'agent ? Ceux dont les représentants de IS ont la capacité d'empêcher qu'ils violent la norme ? Ceux dont la réglementation interne à IS interdit qu'ils violent la norme ? Est-ce que le législateur peut obliger IS a faire en sorte que ces agents ne fassent aucune action ayant pour effet $Know_iBel_{IS}\phi$ (qui s'exprimerait par une formule telle que : $\forall x(Responsable(IS, x) \rightarrow ObgE_{IS}\neg E_xKnow_iBel_{IS}\phi)$?

Ca peut être aussi le cas de l'agent h dans le scénario 2 quand il accède par "intrusion" aux croyances de IS.

Question : est-ce que le législateur doit dériver des normes de la forme $PermKnow_iBel_{IS}\phi$ ou $ForbKnow_iBel_{IS}\phi$ des normes portant sur TOUS les agents qui sont susceptibles de faire en sorte que l'on ait $Know_iBel_{IS}\phi$?

La réponse est vraisemblablement "non" car le législateur ne peux pas considérer TOUS les agents et TOUTES les actions qu'ils sont susceptibles de réaliser (voir avec des juristes).

Plus généralement la question est : est-ce que d'une norme de la forme $Forb\psi$ le législateur doit dériver une norme de la forme $\forall xForbE_x\psi$? On peut argumenter que la réponse est "non" car le domaine de $\forall x$ n'est pas défini (voir les formules qui ne sont pas "domain independent").

Chapitre 2

Dérivation des exigences formelles de sécurité

2.1 Introduction

Des millions de dollars de pertes, c'est le résultat des attaques visant des systèmes non sûrs et des logiciels de mauvaise qualité, facilement exploitables par des pirates. Le pourcentage des vulnérabilités a augmenté considérablement dans les dernières années selon les statistiques du CERT Coordination Center (Carnegie Mellon University), principal organisme de gestion des problèmes de sécurité dans le monde. Certaines attaques se sont même soldées par des pertes en vies humaines (dossiers médicaux corrompus). Pour y faire face, les producteurs de logiciels développent des "patchs". Mais ces rustines ne constituent pas une solution définitive et viable à moyen et long terme.

Les problèmes de sécurité des systèmes et des logiciels ne doivent pas être découverts et résolus *a posteriori*. Ils doivent être identifiés dès les premières phases d'élaboration des exigences pour garantir l'efficacité des solutions. Plusieurs études ont montré que la correction des défauts d'ingénierie des exigences en cas d'arrêt intempestif du système coûte de 10 à 200 fois plus cher que s'ils sont détectés au cours de l'élaboration des exigences.

L'ingénierie des exigences (Requirement Engineering - RE) consiste à concevoir un processus pour suivre l'élaboration du système ou du logiciel à réaliser, et faciliter la compréhension des exigences du client ou commanditaire. La RE traditionnelle identifie, spécifie et analyse les exigences fonctionnelles (tout ce que nous attendons du système ou du logiciel) et les exigences non fonctionnelles (les contraintes telles que la performance). Nous notons cependant que les exigences de sécurité ne sont pas prises en compte par la RE traditionnelle.

Fort de ces constats, la communauté sécurité ainsi que la communauté de RE investissent de plus en plus dans des méthodes et des techniques pour intégrer et/ou adapter les exigences de sécurité comme une partie du processus initial des RE sans réellement parvenir à des solutions optimales.

Par ailleurs, une grande partie des exigences de sécurité est souvent exprimée sous forme de règles de sécurité (règles de contrôle d'accès, de contrôle de flux, de détection d'intrusion,...) qui, ensemble, constituent ce qu'on appelle communèment une politique de sécurité. Elles se conforment généralement à un modèle de sécurité (BLP[2], RBAC[14], OrBAC[5],...). Malheureusement, bien que l'expression de cette politique soit formelle, puisqu'elle fait appel à des logiques classiques ou d'ordre supérieur (voir Chapitre 1 et Chapitre 3), le processus qui y conduit a toujours été informel. Nous proposons dans ce chapitre nos premiers travaux vers la formalisation de ce processus qui part des objectifs de sécurité (le cahier des charges) et en dérive la politique formelle de sécurité (les exigences).

2.2 Processus de dérivation de la politique formelle de sécurité

Il est basé essentiellement sur quatre phases illustrées par la figure 2.1.

- (Risq) : une analyse de risques du système d'information ou logiciel dont on souhaite établir la politique de sécurité. Elle permet d'identifier les menaces, les vulnérabilités, les hypothèses environnementales, les scénarios probables d'attaques, les contraintes et se solde par l'établissement d'une matrice d'aversion regroupant l'ensemble des propriétés de sécurité attendues du système et leur degré de criticité,
- (Spec) : une spécification du système considéré qui tient compte de la phase (Risq) afin d'intégrer et différencier les objectifs de sécurité. Elle sera raffinée pour dériver les exigences du système cible,
- (SpecS) : une extraction de la spécification de l'ensemble du système cible, établie dans la phase (Spec), de la spécification de ses exigences de sécurité,
- (Policy) : une génération, à partir de la spécification sécurité du système cible, de la politique formelle de sécurité.

Nous ne présentons pas dans ce chapitre la première phase (Risq), elle est traitée dans la tâche WP6.2 du projet Selkis.

2.3 Spécification formelle des exigences

La phase (Spec) a été menée en adoptant une approche reposant sur l'ingénierie des exigences, une branche de l'ingénierie logicielle traitant des buts (ou objectifs) du monde réel, les fonctions et les contraintes du système logiciel. Elle traite également des relations entre ces facteurs afin d'établir les spécifications concernant le comportement du système et des logiciels et leur évolution dans le temps. Il existe une dizaine de méthodes RE : GDC, ISAC, i*, NFR, GBRAM, ... et la méthodologie la plus connue Kaos. Comparable aux autres methodes (Kavalki et al dans [18] fait une évaluation de certaines d'entre-elles, voir figure 2.2), Kaos utilise des concepts formels et formalisables et est facile d'utilisation. Il n'est pas difficle d'introduire des exigences de sécurité avec les concepts originaux et ceux étendus, mais l'outillage support de cette méthode, Objectiver,

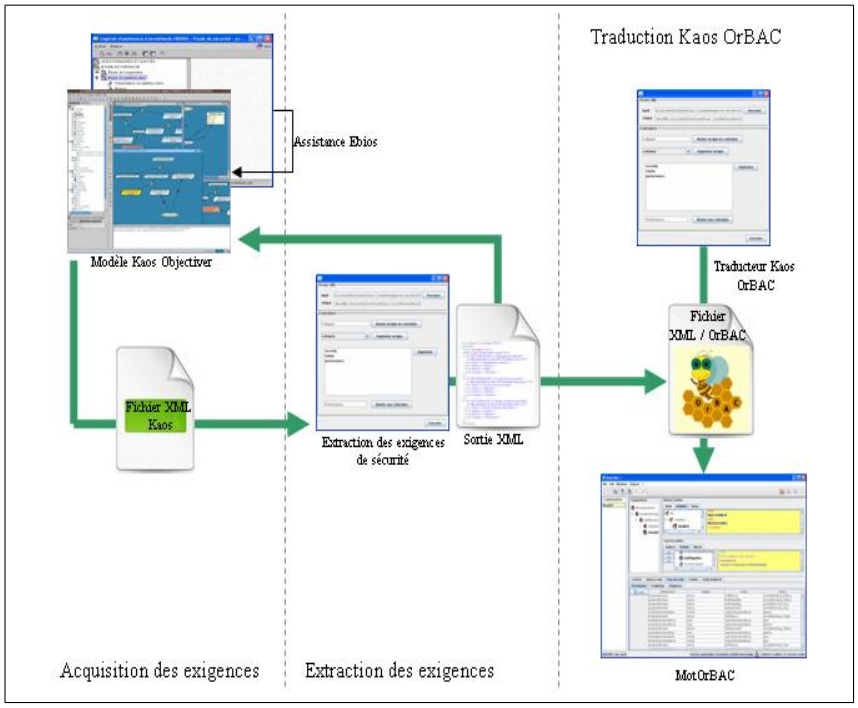

FIG. 2.1 – Eléments cibles Kaos pour l'extraction des exigences formelles de sécurité

est un produit propriétaire donc impossible à customiser. Nous avons cependant adopté Kaos/Objectiver dans le cadre des présents travaux, notre but dans ce chapitre n'étant pas d'étendre l'un ou l'autre.

2.3.1 Méthodologie KAOS orientée buts

La méthode d'ingénierie des exigences orientée buts consiste en l'utilisation des buts pour obtenir, structurer, spécifier, analyser, négocier, documenter, et modifier les exigences [17]. Un but est une déclaration prescriptive que le système doit satisfaire à travers la coopération de ses agents. Un agent est un composant actif jouant un rôle spécifique dans la satisfaction du système. Un but peut être raffiné en sous buts. Il existe deux types de lien de raffinement : (1) les liens de raffinement AND sont des relations entre un but père et un ensemble de sous buts (appelés raffinements) ; la satisfaction de tous les sous buts du raffinement est une condition suffisante pour satisfaire le but, (2) les liens de raffinement OR associent un but père à un ensemble de raffinements alternatifs ; la satisfaction d'un des raffinements est une condition suffisante pour satisfaire le but père.

Le raffinement de buts se termine lorque chaque sous but est réalisable par un certain agent, exprimable en termes de conditions qui sont contrôlables par

Goal-oriented Approaches / *Framework Components*		Cognitive Task Analysis	*i**	Goal-based workflow	EKD	F³ (OM)	ISAC	SIBYL	The reasoning loop model	REMAP	KAOS	GBRAM	Goal-scenario coupling	NFR framework	GSN	GQM
Usage	understand current org. situation	✓	✓	✓	✓											
	understand the need for change					✓	✓									
	provide the deliberation context within which RE occurs							✓	✓	✓						
	relate business goals to system components										✓	✓	✓	✓		
	evaluate system specs against stakeholders' goals														✓	✓
Subject	enterprise goals	✓	✓	✓	✓		✓			✓	✓	✓	✓	✓		
	process goals					✓	✓	✓	✓							
	evaluation goals														✓	✓
Representation	formal		✓							✓	✓			✓		
	semi-formal	✓		✓	✓	✓		✓	✓			✓	✓		✓	✓
	informal						✓									
Development	way-of-working		◆		◆		◆				◆	◆	◆	◆		
	tool support	M	M, F	M	M	M, G	M, G	M		M,F	M,F	M, G	M, G	M,F	M	M, G

◆ = suggest a number of steps and associated strategies
M = support for model construction, F = formal reasoning support, G = process guidance

FIG. 2.2 – Travaux de recherche et outils dans la RE - Extrait de [18]

l'agent, on obtient ainsi l'exigence. Une spécification Kaos du système cible consiste à construire quatre modèles (buts, objets, opérations et agents). Un modèle d'anti-but peut également être conçu pour spécifier des obstacles à la réalisation du but, utile lorsqu'on introduit des exigences de sécurité.

Spécification des buts : les buts sont des objectifs, fonctionnels ou non fonctionnels (figure 2.3) que le système doit réaliser à travers la coopération de plusieurs agents. Ils doivent être raffinés (figure 2.4) et spécifiés avec précision pour supporter l'élaboration des exigences (dont certaines serviront à la génération de la politique formelle de sécurité), la vérification/validation, la détection des conflits, la négociation, l'explication et l'évolution (figure 2.5 ou figure 2.6). Les spécifications déclarent les buts, leur type, attributs et liens. Elles incluent souvent des verbes "mot-clés", comme *achieve* ou *maintain*, qui spécifient en principe des motifs temporels pour le nom du but qui apparaît comme paramètre. Par exemple : *maintain*[*satisfaire_demande*], signifie que le but "satisfaire la demande de prêt des livres" est un but que le système doit assurer de façon pérenne. Nous reviendrons sur ce dernier aspect dans la section relative à l'extraction des exigences de sécurité.

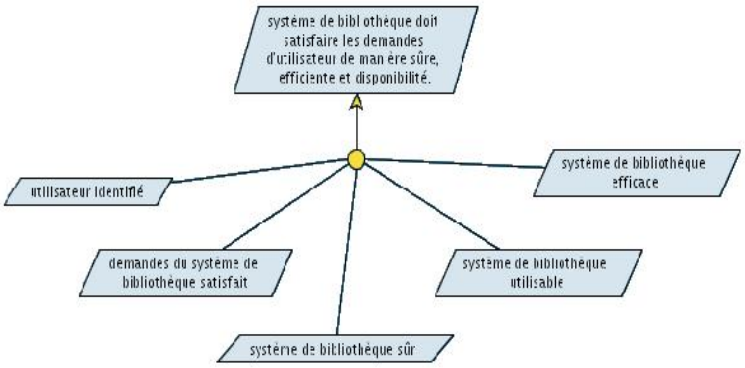

FIG. 2.3 – Des buts fonctionnels et non fonctionnels

Spécification des objets : Le modèle objet est utilisé pour définir et documenter les concepts de l'application du domaine. Il fournit des contraintes statiques sur le système opérationnel qui permettraient de satisfaire les buts, exigences et attentes. Les objets qui apparaissent dans ce modèle sont des éléments du système dont les instances sont identifiables et dont l'état peut évoluer. Lorsque l'objet est passif et autonome (un livre par exemple) il est appelé *entité*. Les objets actifs réalisent des opérations, ce sont des agents (le système de contrôle des prêts de la bibliothèque par exemple). Des évènements, qualifiés d'objets instantanés, caractérisent l'environnement à un moment donné. Enfin, des associations permettent de relier tous les types objets (autonomes, actifs, passifs, instantanés). Les

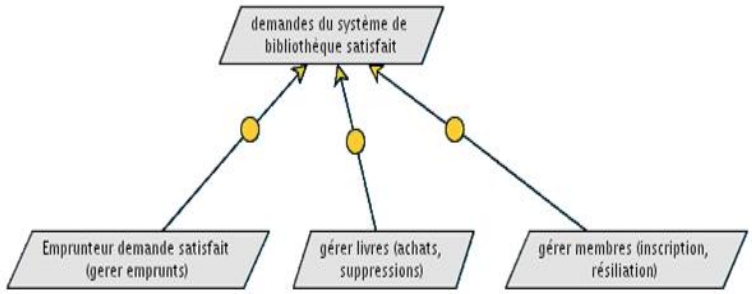

FIG. 2.4 – Première étape de raffinement

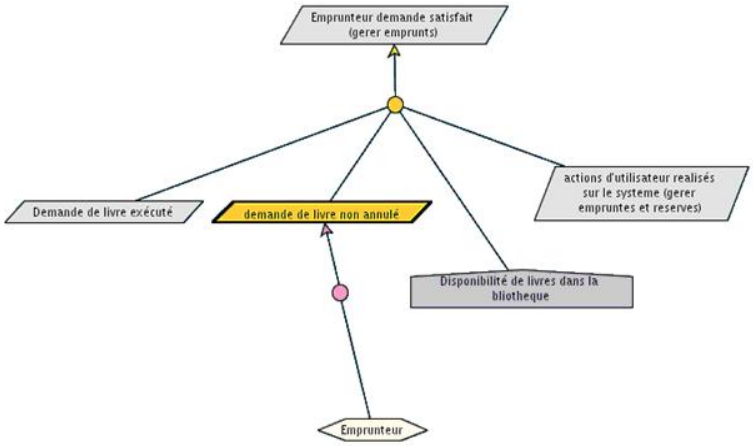

FIG. 2.5 – Dérivation d'une exigence avec contrainte de disponibilité

objets et leurs associations sont des élements qui seront extraits du modèle pour les besoins de dérivation de la politique de sécurité formelle.

Spécification des opérations : il s'agit de représenter les services que doit fournir le système basé sur le modèle de buts et les scénarios identifiés lors de la sélection des exigences. Ce modèle décrit les comportements dont les agents ont besoin pour réaliser leurs exigences. Les opérations sont des relations d'entrée sortie sur les objets (Figure 2.7). Elles agissent sur les objets, peuvent en créer, envoyer les états de transitions ou activer d'autres opérations (en envoyant des événements). Un agent responsable de la réalisation d'un but doit effectuer (relation *performance* selon le vocabulaire Kaos) toutes les opérations qui permettent d'opérationnaliser ce but tout en tenant compte des permissions et obligations caractérisant ces opérations (Figure 2.8).

Spécification des agents : elle permet de représenter la distribution des res-

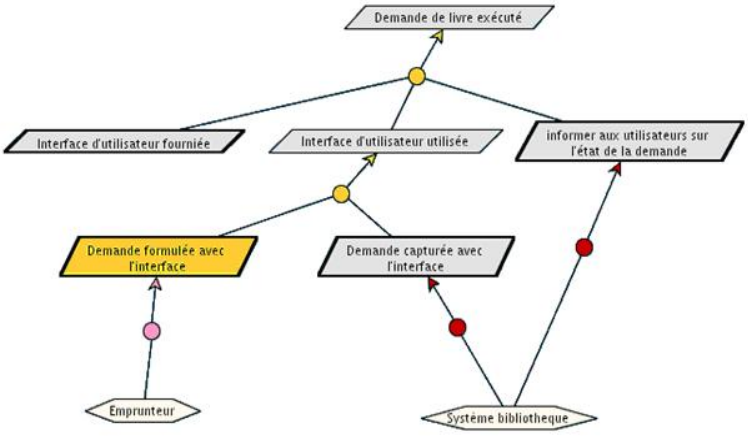

FIG. 2.6 – Dérivation d'une exigence avec contrainte de moyen

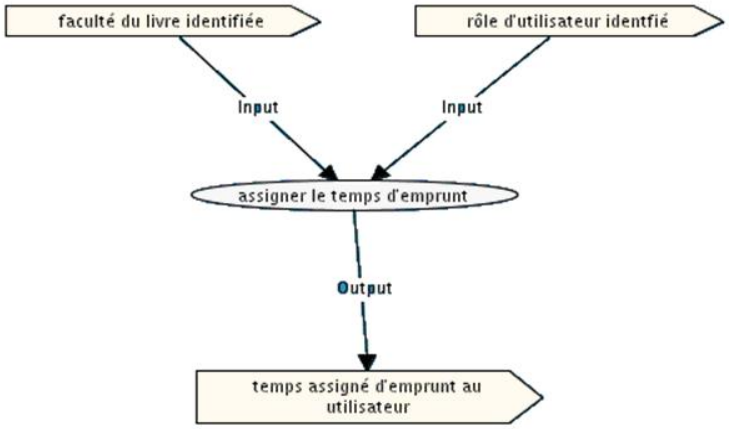

FIG. 2.7 – Association Agents Opération

ponsabilités aux divers éléments actifs du système (Figure 2.9) : êtres humains, composants logiciels, composants matériels, etc. En effet, les buts requièrent des agents pour leur réalisation. Une coopération entre agents pouvant être utile, une relation *dépendance* a été introduite pour atteindre un but ou réaliser une opération conformément aux permissions et aux obligations prescrites.

Les relations entre modèles établissent les liens entre les quatre types de spécifications décrites dans les sections précédentes. C'est à partir de ce modèle global que se fera l'extraction de la spécification Kaos sécurité du système cible.

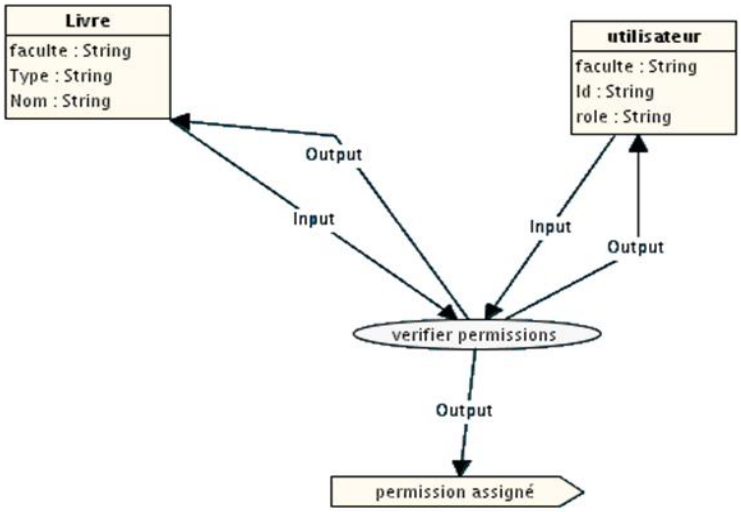

FIG. 2.8 – Association Agents Opération

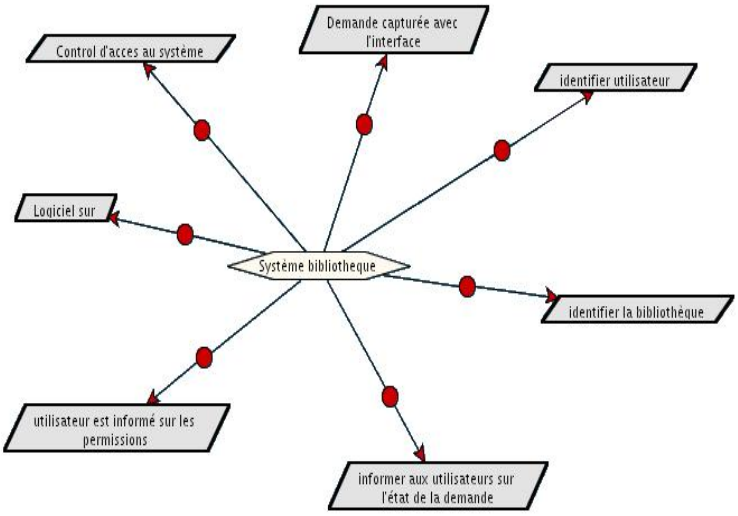

FIG. 2.9 – Responsabilités de l'agent Système bibliothèque

2.3.2 Kaos guidé par une analyse de risques

Dans une spécification Kaos d'un système cible il n'existe pas réellement de distinction claire entre les aspects purement fonctionnels et les aspects sécurité. Généralement, les exigences relatives à la protection du système sont plutôt

réduites au strict nécessaire puisqu'elles ne constituent pas un objectif prioritaire en comparaison aux objectifs relatifs aux services que ce système est censé offrir et pour lesquels il a été (ou va être) conçu.

Pour que la sécurité soit réellement prise en compte dans ce type de méthodologie de spécification, il est nécessaire qu'une analyse des risques auxquelles le système cible peut être confronté soit menée. Kaos n'est pas une méthodologie d'analyse de risques pas plus qu'elle n'est une méthode d'analyse des exigences. Kaos permet d'intégrer de façon construite et graduelle les exigences du système cible et qui sont plutôt des données en entrée de la méthode.

Le meilleur moyen permettant de fournir des données relatives à la sécurité est de faire appel à une méthode d'analyse de risques, qui alimente et guide la spécification Kaos. Les méthodes d'analyse de risques sont nombreuses. Dans notre étude, nous avons choisi une méthode qui répond aux critères de simplicité, distribution libre de l'outil support, maintenance et évolution assurée et orientation buts ou objectifs pour rester dans l'esprit de Kaos. Nous avons opté pour EBIOS[12] qui satisfait tous ces critères.

EBIOS (Expression des besoins et identifications des Objectifs de Sécurité) a été conçue et développée par le DCSSI (Direction Centrale de la Sécurité des Systèmes d'Information) ; Elle permet comme son nom l'indique l'expression des besoins et l'identification des objectifs de sécurité pour un système à concevoir ou un système existant. Elle s'utilise au niveau de la phase d'élaboration d'un schéma directeur opérationnel d'un système d'information et a comme objectif principal est de déterminer les actions de sécurité qu'il convient d'entreprendre, ce qui, indépendamment de l'aspect sécurité, rejoint complètement la philosophie de Kaos.

Nous donnons quelques éléments déjà identifiés, à titre d'exemples, pour mener ce mariage Kaos/EBIOS (Figure 2.10 et 2.11). La spécification complète du processus et sa formalisation seront présentées dans une version actualisée du présent livrable.

2.4 Extraction de la spécification des exigences de sécurité

Le modèle KAOS permet d'adjoindre des méta-données à chaque composant de modèle (Figure 2.12) :

Le motif : il sert à décrire les comportements "temporels" requis par le but. La méthodologie KAOS supporte les quatre motifs suivants :

- Achieve (atteindre) : exige qu'une certaine propriété reste constante,
- Cease (arrêter) : exige qu'une certaine propriété constante ne soit plus satisfaite,
- Maintain (maintenir) : exige qu'une certaine propriété se maintienne toujours,
- Avoid (éviter) : exige qu'une certaine propriété ne soit jamais valide ni maintenue.

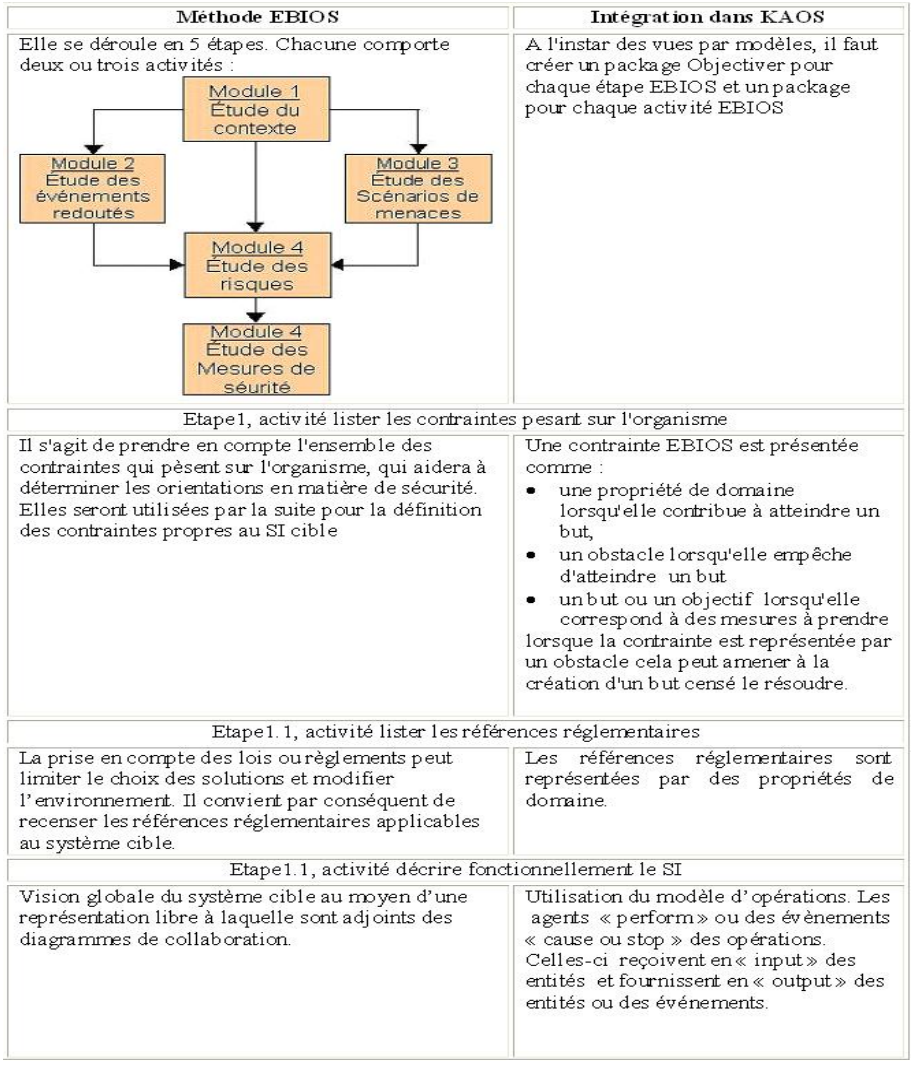

FIG. 2.10 – EBIOS/KAOS

La catégorie : il s'agit de classifier le but ou l'exigence. Les catégories disponibles dans la méthodologie KAOS sont : Accuracy (précision), Adaptability (adaptabilité), Capacity (capacité), Consistency (cohérence), Efficiency (efficacité), Information (information) Performance (performance), Privacy (confidentialité, vie privée), Robustness (robustesse), Safety (sûreté), Satisfaction (satisfaction), Security (sécurité), Usability (utilisabilité). Dans notre cas, si on veut modéliser une exigence de sécurité, il faut choisir la catégorie "Security" lors du raffinement du but.

Méthode EBIOS	Intégration dans KAOS
Etape 1.2, activité identifier les hypothèses et règles de sécurité pré-existantes	
Les hypothèses relatives au système cible sont souvent imposées par le commanditaire de l'étude. Elles sont liées à des politiques internes ou externes, financières ou de calendrier. Elles peuvent constituer un risque accepté a priori. La sécurité du SI cible peut avoir fait l'objet d'un référentiel, l'objectif ici est de recueillir les principales règles et mesures de sécurité.	Une hypothèse est formulée comme une propriété du domaine. Une règle de sécurité est représentée par une propriété du domaine
Etape 1.3, activité déterminer la cible de l'étude de sécurité, lister et décrire les entités	
Les entités EBIOS représentent les acteurs actifs et passifs agissant sur le SI cible ou recevant des informations du SI. Elles sont classées en plusieurs types : organisation, personnel, logiciel, réseaux,...	Une entité passive EBIOS est représentée par une entité KAOS. Une entité active EBIOS est représentée par un agent KAOS
Etape 3.1, activité étude des origines des menaces	
Méthode d'attaque Moyen, de type action ou évènement, pour un élément menaçant de réaliser une attaque	Utilisation d'une entité à laquelle seront adjoints des attributs de l'entité KAOS décrivant « partiellement » la méthode d'attaque : • Attentes (confidentialité, ...) • Causes d'éléments menaçants (accidentelles, malveillantes,...) • Types d'éléments menaçants (humains, programmes,...)
Etape 3.3, activité formalisation des menaces	
Chaque menace fait l'objet d'une description la plus précise possible (vulnérabilités, entités concernées,...) et d'une caractérisation de sa probabilité d'occurrence ou de faisabilité.	La menace est représentée par un obstacle, un (anti-but) à raffiner et auquel il faut adjoindre des descriptions comme propriété du domaine et/ou attributs, par exemple « opportunité » quantifiée sur une échelle de 1 à 4 ou « critère de sécurité » (disponibilité, intégrité, confidentialité,...)
...	...

FIG. 2.11 – EBIOS/KAOS

La priorité : à chaque but ou exigence est attribuée une priorité (basse, medium ou haute). Si par exemple nous avons une exigence qui stipule que le changement de mot de passe doit se faire tout les trois mois et que cette action est indispensable à la satisfaction d'un objectif de confidentialité, l'attribution d'une priorité haute permet de capturer cet état de fait.

Le processus d'extraction de la spécification KAOS sécurité (SpecS) est basé sur les méta-données définies dans chaque composant du modèle. Cela suppose donc que lors de l'étape de spécification du SI cible (Spec) décrite précédem-

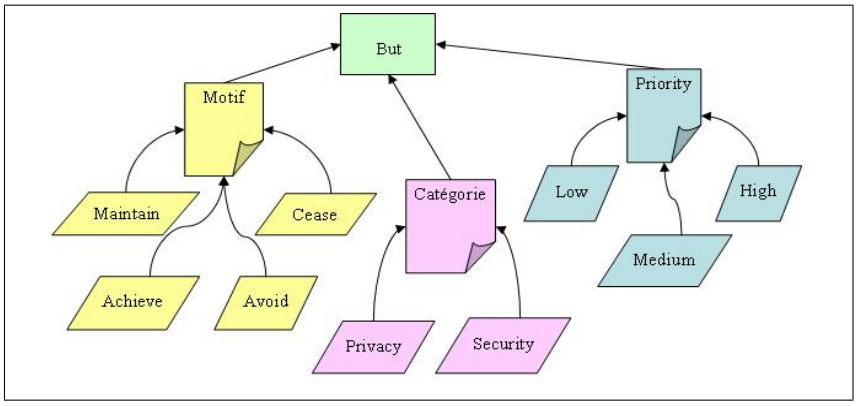

FIG. 2.12 – Eléments cibles Kaos pour l'extraction des exigences formelles de sécurité

ment, la qualification des exigences au travers de ces méta-données est effectuée. L'analyse de risques qui guide la spécification KAOS lorsqu'il s'agit des aspects liés à la sécurité aide à la valuation des ces méta-données. Par exemple, l'étape 3.3 EBIOS/KAOS présentée dans la section précédente, permet l'estimation des priorités. Le processus d'extraction de SpecS que nous avons défini prend comme paramètre (données en entrée) la spécification (Model) obtenue dans l'étape Spec et un dictionnaire (Map) reliant chaque méta-donnée à sa liste des valeurs nécessaires pour le filtrage. Par exemple, pour extraire les exigences de sécurité dont le niveau d'exigence est haut, le processus aura comme paramètres la méta-donnée "category" avec la liste des valeurs "Security, Privacy, Safety" et la méta-donnée "priority" avec la valeur "High". Le processus parcourt les composants du modèle KAOS intégrant la sécurité. Ensuite, pour chaque composant, il extrait les spécifications correspondantes si ce composant dispose des métas-données demandées et avec les valeurs précisées en entrée.

le pseudo-code correspondant est les suivant :

Begin :
Soit *result* la liste de composants résultat de l'algorithme
On parcourt les composants de *Model*
Pour chaque composant C de *Model* faire :
Pour chaque méta-donnée *meta* du dictionnaire *Map* faire :
Si le composant C dispose de la méta-donnée *meta* alors :
Soit *meta_values* la liste des valeurs associées
à la méta donnée *meta* dans le dictionnaire *Map*
Soit *value* la valeur de méta-donnée *meta*
du composant C
Si *value* appartient à la liste *meta_values* alors :
ajouter le composant C au résultat *result*
Fin Si

```
            Fin Si
        Fin Pour
    Fin Pour
End
```

Le processus d'extraction des exigences de sécurité est mis en oeuvre au moyen d'une l'application qui se base sur une spécification KAOS au format XML (conformément à la phase Spec) que génère l'outil Objectiver (Figure 2.13). Cette application a été développée avec Java en utilisant JDOM pour manipuler des documents XML. DOM (Document Object Model) propose une API qui permet de modéliser, de parcourir et de manipuler les documents XML.

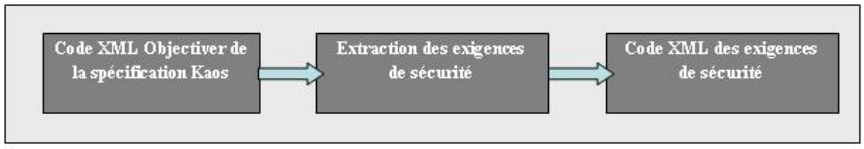

FIG. 2.13 – Processus d'extraction de la spécification Kaos des exigences de sécurité

2.5 Génération de la politique de sécurité : Discussion

L'objectif de cette ultime étape Policy est de dériver des spécifications conçues conformément à une méthodologie de modélisation des exigences orientée buts des exigences de sécurité sous la forme d'une politique de sécurité formelle. Le modèle de politique de sécurité visé étant OrBAC, il s'agit de générer à partir de la spécification KAOS XML extraite dans l'étape précédente, des règles de sécurité de type permissions ou obligations au format :

$security\_rule\_type(org, role, activity, view, context)$

où $security\_rule\_type \in \{permission, prohibition, obligation, dispensation\}$

Pour cela, la génération doit reconstruire les différentes relations OrBAC à partir du résultat de SpecS, Ces relations sont :

- $empower(org, s, r)$: dans l'organisation org, le sujet s est habilité dans le rôle r.
- $consider(org, \alpha, a)$: dans l'organisation org, l'action α implémente l'activité a.
- $use(org, o, v)$: dans l'organisation org, l'objet o est utilisé dans la vue v.
- $hold(org, s, \alpha, o, c)$: dans organisation org, le contexte c est vrai pour le sujet s, l'action α et l'objet o.

Ce processus à ce stade du projet n'est pas encore finalisé et prouvé, les travaux sont en cours. Nous avons cependant commencé l'exploration et la formalisation de cette phase.

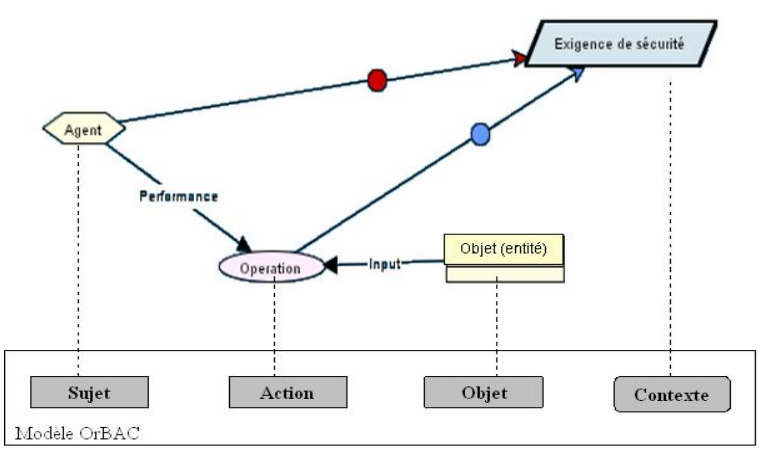

FIG. 2.14 – Liens d'aide à la génération de la politique

Les relations $empower(org, s, r)$, $consider(org, \alpha, a)$ et $use(org, o, v)$ ne sont pas à ce stade d'avancement dérivables automatiquement des résultats de la phase SpecS. Les concepts organisationnels de rôles, d'activités et vues au sens OrBAC en tant que tels sont absents de la spécification KAOS en sortie de la phase Spec. Il est nécessaire soit d'étendre le modèle KAOS, ce n'est pas l'objectif ici, soit introduire ces éléments comme des propriétés du domaine. Les réflexions sont en cours.

Actuellement, les premiers liens établis, appelés à évoluer, sont résumés dans la figure 2.14. En particulier,

- l'exigence de sécurité conçue selon la méthodologie KAOS est vue comme un contexte OrBAC.
- Le type de règle de sécurité (permission, interdiction ou obligation) est dérivable des liens pré-établis avec les méta-données "motifs" lorsqu'il s'agit d'exigences de sécurité (i.e. catégories "security" ou "privacy") et dont il faut tenir compte lors de la phase de spécification KAOS. Ainsi, le motif "achieve" dans KAOS générerait des permissions dans la politique OrBAC, le motif "cease" générerait des recommandations, le motif "maintain" une obligation et le motif "avoid" traduirait des interdictions dans OrBAC.

L'appel à l'API MotOrBAC (l'outil support au modèle OrBAC)[20] et l'utilisation des algorithmes d'extraction des paramètres des règles de sécurité (Algorithm 1, 2, 3, 4, 5, 6 et 7) et de génération de ces règles (Algorithm 8), avec comme données en entrée la spécification KAOS XML en sortie de la phase SpecS permet de générer, dans cette phase Policy, la politique de sécurité formelle du SI cible. Cette politique est partielle pour le moment. Des travaux portant sur l'enrechissement, la structuration de la spécification KAOS sécurité pour faciliter le processus d'extraction et de dérivation de la politique de sécurité sont en cours.

Extraction des exigences de sécurité : On construit l'ensemble des exigences de sécurité $\mathbb{S}$ (Algorithm 1). La fonction $child\_attribute(a, t, n)$ renvoie la valeur de l'attribut a du noeud fils de type t du noeud n.

Algorithm 1 Extract requirements

for all $n \in \mathbb{N}$ **do**
  if $child\_attribute("AV", "Category", n) = "Security"$ **then**
    $n \rightarrow \mathbb{S}$
  end if
end for
end

Extraction des agents : On construit l'ensemble des agents $\mathbb{A}$ (Algorithm 2). la fonction $type(n)$ renvoie le type du noeud n et la fonction $attribute(a, n)$ renvoie la valeur de l'attribut a du noeud n.

Algorithm 2 Extract agents

1: **for all** $n \in \mathbb{N}$ **do**
2: **if** $(type(n) = "C")$ **and** $(attribute("p", n) = "Method : NewKaos2003 : MetaModel : Agent")$ **then**
3: $n \rightarrow \mathbb{A}$
4: **end if**
5: **end for**
end

Extraction des opérations : On construit l'ensemble des opérations $\mathbb{O}$ (Algorithm 3).

Algorithm 3 Extract operations

1: **for all** $n \in \mathbb{N}$ **do**
2: **if** $(type(n) = "C")$ **and** $(attribute("p", n) = "Method : NewKaos2003 : MetaModel : Operation")$ **then**
3: $n \rightarrow \mathbb{O}$
4: **end if**
5: **end for**
end

Extraction des entités : On construit l'ensemble des entités $\mathbb{E}$ (Algorithm 4).

Algorithm 4 Extract entities

1: **for all** $n \in \mathbb{N}$ **do**
2: **if** $(type(n) = "C")$ **and** $(attribute("p", n) = "Method : NewKaos2003 : MetaModel : Entity")$ **then**
3: $n \rightarrow \mathbb{E}$
4: **end if**
5: **end for**
end

Extraction des paires {agent, operation} : On construit l'ensemble de couples {agent, operation} $\mathbb{P}$ (Algorithm 5).

Algorithm 5 Extract {agent, operation} couples

1: **for all** $n \in \mathbb{N}$ **do**
2: **if** $(type(n) = "C")$ **and** $(attribute("p", n) = "Method : NewKaos2003 : MetaModel : Performance")$ **then**
3: $agent = child\_attribute("R", "PerformedBy", n)$
4: $operation = child\_attribute("R", "Performs", n)$
5: $\{agent, operation\} \rightarrow \mathbb{P}$
6: **end if**
7: **end for**

end

Extraction des couples {opération, entité} : On construit l'ensemble de couples {opération, entité} $\mathbb{I}$ (Algorithm 6).

Algorithm 6 Extract {operation, entity} couples

1: **for all** $n \in \mathbb{N}$ **do**
2: **if** $(type(n) = "C")$ **and** $(attribute("p", n) = "Method : NewKaos2003 : MetaModel : Input")$ **then**
3: $entity = child\_attribute("R", "Inputs", n)$
4: $operation = child\_attribute("R", "InputOf", n)$
5: $\{operation, entity\} \rightarrow \mathbb{I}$
6: **end if**
7: **end for**

end

Extraction des couples {agent, exigence} : On construit l'ensemble de couples {agent, exigence} $\mathbb{R}$ (Algorithm 7).

Algorithm 7 Extract {agent, requirement} couples

1: **for all** $n \in \mathbb{N}$ **do**
2: **if** $(type(n) = "C")$ **and** $(attribute("p", n) = "Method : NewKaos2003 : MetaModel : ResponsabilityAgentSon")$ **then**
3: $agent = child\_attribute("AOT", "s", n)$
4: $requirement = child\_attribute("AOT", "a", n)$
5: $\{agent, requirement\} \rightarrow \mathbb{R}$
6: **end if**
7: **end for**

end

Génération de la politique : On génère la politique OrBAC à partir des ensembles construits (Algorithm 8). la fonction $pattern(r)$ renvoie l'attribut *pattern* d'une exigence de sécurité, cet attribut étant défini dans Objectiver. L'identifiant def_ctx désigne le contexte par défaut.

Algorithm 8 Generate OrBAC policy

```
1: CreateOrganization("org")
2: for all a ∈ 𝔸 do
3:    CreateRoleAndInsertIntoOrg(a,"org")
4: end for
5: for all o ∈ 𝕆 do
6:    CreateActivityAndInsertIntoOrg(o,"org")
7: end for
8: for all e ∈ 𝔼 do
9:    CreateViewAndInsertIntoOrg(e,"org")
10: end for
11: for all {agent, requirement} ∈ ℝ do
12:    if (∃ operation ∈ 𝕆 and ∃ entity ∈ 𝔼 such that {agent, operation} ∈
       ℙ and {operation, entity} ∈ 𝕀 then
13:       if pattern(requirement) = "Achieve" then
14:          CreateAbstractPermission(org, agent, operation, entity, def_ctx)
15:       else if pattern(requirement) = "Avoid" then
16:          CreateAbstractProhibition(org, agent, operation, entity, def_ctx)
17:       else if pattern(requirement) = "Maintain" then
18:          CreateAbstractObligation(org, agent, operation, entity, def_ctx, def_ctx)
19:       end if
20:    end if
21: end for
end
```

2.6 Conclusion

L'étude de dérivation des exigences formelles de sécurité, bien que toujours en cours,a permis en partant de la spécification des objectifs de sécurité de dériver une politique de sécurité gros grain. Pour l'affiner, l'intégration d'éléments de structuration dans le modèle KAOS que nous ne souhaitons pas intrusive (c'est-à-dire sans étendre KAOS avec de nouveaux concepts), une formalisation et des preuves de correction du processus complet d'extraction et de dérivation sont envisagés. L'ensemble du processus est en cours d'application à l'étude de cas Medecom et sera présenté dans une version actualisée du présent livrable.

Chapitre 3

Expression et mise en oeuvre du contrôle d'accès *a posteriori*

Les travaux menés dans le cadre du projet SELKIS ont considéré deux axes :

- La mise en oeuvre d'un processus de contrôle d'accès *a posteriori*, sur la base des logs établis par l'IHE ATNA en interface avec un modèle de contrôle d'accès et d'usage OrBAC.
- La protection *a posteriori* des images médicales qui combine technologie de tatouage (ou wateramkring) et une politique de contrôle d'accès et d'usage OrBAC.

Nous reprenons dans ce rapport l'essentiel des concepts et stratégies dévelopés au cours de ces deux actions.

Première partie : Expression et modélisation du contrôle d'accès et d'usage et sa mise en oeuvre dans les environnements de santé *a posteriori*

3.1 Introduction

La confidentialité et le respect du droit à la vie privée sont essentiels dans le contexte des travaux collaboratifs et dans des environnements de systèmes distribués notamment dans le domaine de la santé auquel nous nous intéressons. Il est alors important de définir et mettre en application une politique de sécurité permettant d'assurer ces propriétés sans pour autant entraver le fonctionnement du système par des mécanismes de blocage d'accès et d'usage rigides, incompatibles avec des environnements comme ceux de la médecine.

De nombreux travaux ont été consacrés à la modélisation de politique permettant d'exprimer les privilèges des utilisateurs. Différents modèles ont été en conséquence proposés : DAC (Discretionary Access Control)[8], MAC (Mandatory Access Control)[2], RBAC (Role Based Access Control)[14] ou OrBAC (Organization Based Access Control)[5]. Sur la base de ces modèles, la requête d'accès d'un utilisateur est d'abord validée avant de donner ou de refuser l'accès à la donnée demandée. Ces modèles offrent une protection a priori et souffre de quelques limitations notamment lorsqu'une situation n'a pas été envisagée au préalable (situation d'urgence) et n'a pas été exprimée sous forme de règle dans la politique de contrôle d'accès. Dans le même temps, vérifier la validité d'une requête et l'autoriser peut engendrer des délais trop longs et rédhibitoires dans la prise en charge d'un patient. Le contrôle d'accès *a priori*, s'il est efficace, n'est donc pas forcément toujours approprié.

Récemment, des approches de contrôle d'accès *a posteriori* ont été proposées [5,6] pour essayer de résoudre ces problèmes. Avec ce type d'approche, la vérification de la conformité à la politique de sécurité est conduite plus tard, laissant l'accès à l'utilisateur. L'utilisateur aura à justifier que ses actions passées étaient autorisées. Cette possibilité d'accéder à l'information avec une vérification *a posteriori*, est intéressante dans le domaine de la santé car elle n'apparaît pas comme un frein à la pratique quotidienne. Un médecin peut continuer à travailler et si une violation de la politique d'accès est détectée, il aura à se justifier et pourra être sanctionné le cas échéant.

Trois composants sont nécessaires au déploiement d'une politique de contrôle d'accès *a posteriori* :

1. La journalisation : il sagit de l'enregistement des actions effectuées par les utilisateurs du système au travers des Logs. Ces logs constituent des preuves auxquelles on pourrait faire appel pour démontrer qu'un utilisateur était autorisé ou non à réaliser l'action qu'il a effectuée.
2. L'audit : identifie, à partir de l'analyse des logs, des actions ou séquences anormales en fonction des règles de sécurité.

3. L'imputabilité : décide si l'action ou la transaction anormale détectée, l'utilisateur avait le droit de la réaliser. Cette décision tient compte des évrènements et des contextes liés à cette action ou transaction au moment où elle a été exécutée. Si une violation est constatée son auteur pourra être sanctionné.

L'efficacité de cette approche dépend donc de l'aptitude du système à détecter des comportements anormaux et des violations de la politique de sécurité.

Les travaux menés dans ce contexte ont portés sur l'étape d'analyse de log. Ce composant de sécurité comporte trois modules :

- le module de politique de sécurité contenant à la fois les règles d'accès et d'usages a priori et *a posteriori*.
- le réconciliateur log-politique qui, comme son nom l'indique a pour objectif de réecrire les logs générés automatiquement par le système dans un format compatible avec le modèle de politique de sécurité utilisé. Ce reformatage facilite et rend plus efficace la détection des violations.
- le module de décision qui vérifie si les logs orientés modèle de politique de sécurité se conforment à la politique de sécurité organisationnelle.

Pour ces travaux, et en ce qui concerne le module de log, nous nous sommes appuyés sur les recommandations IHE[15][16] (Integrating the Healthcare Enterprise) et en particulier sur le profil d'intégration IHE-Audit Trail and Node Authentication (ATNA) qui traite de la sécurité des systèmes d'information de santé. IHE-ATNA définit notamment les règles de préservation de la vie privée (privacy) et des mécanismes permettant de les mettre en oeuvre tel que la trace d'audit (audit trail). Ce dernier permet entre autre de tracer, surveiller et superviser les activités liées à la sécurité tels que le respect du droit à la vie privé du patient, l'authentification des utilisateurs, l'autorisation des accès et des usages dans des applications distribuées. ATNA définit aussi une structure pour cet "audit trail", structure qui est utilisée comme format de log dans le travail réalisé.

Nous nous sommes aussi appuyés sur le modèle OrBAC pour spécifier les politiques de sécurité. OrBAC est un modèle de sécurité qui permet d'exprimer un grand nombre de contraintes de sécurité, parfois complexes, situation que l'on rencontre dans le domaine médicale. Il s'appuie sur des concepts structurels natifs comme le concept d'organisation et de contexte et il inclut différentes modalités de sécurité (permission, interdiction, obligation, dispense). Une organisation peut être vue comme une entité responsable de la gestion de la politique de sécurité (un hôpital, un service, etc.). Le modèle offre le possibilité de faire appel à la notion de contexte pour définir un jeu de conditions à respecter avant d'activer une règle de sécurité (ex. : situation d'urgence). Le contexte peut aussi permettre d'augmenter ou réduire les privilèges d'un utilisateur en fonction de différents évènements ou états (e.g. : spatial, temporel, provisionnel, etc.). La politique est dans ce cas gérée de manière dynamique.

3.2 Profil d'intégration et modèle de sécurité

3.2.1 IHE-ATNA

IHE est une initiative des professionnels et de l'industrie de santé. Elle vise à établir un cadre d'intégration des systèmes d'informations de santé et permettre entre autres l'intéropérabilité des systèmes. L'objectif est de garantir que les données des patients sont fiables et accessibles aux professionnels de santé. IHE propose donc différents profils d'intégration, profils auxquels les systèmes d'information peuvent être ou non compatibles. IHE identifie comme "IHE actor" un ensemble de composants fonctionnels d'une institution et désigne sous le terme de "standards based Transactions" les interactions entre ces acteurs. Ces interactions sont liées à des scénarios de la pratique médicale quotidienne.

ATNA constitue un de ces profils d'intégration et a été défini en considérant des scénarios liés à la radiologie et aux infrastructures hospitalières. Nous nous y sommes intéressés comme solution permettant d'assurer la traçabilité et la sécurité des données et des communications. Les trois éléments clés de ce profil sont : l'authentification des utilisateurs, l'authentifications des machines hôtes durant les communications et la génération des enregistrements d'audit. C'est cette dernière composante du profil que nous intègrons. Cette génération des enregistrements se fait en deux étapes : (1) identification des évènements basée sur les transactions et (2) enregistrement des messages d'audit. Chaque transaction effectuée par un acteur peut générer des évènements et donc donner lieu à la création d'un enregistrement d'audit. Ces enregistrements sont stockés au niveau d'un serveur pour analyse, l'objectif est de détecter des comportements anormaux (exemples, accès ou effacement d'une information de santé protégée).

Un enregistrement d'audit ATNA contient les actions effectuées par les utilisateurs lorsqu'ils accèdent à des informations de santé protégées. Ces actions peuvent être de type création, effacement, mise à jour ou requête de sélection. Ces enregistrements sont crées par des acteurs IHE lorsque des évènements IHE auditables se produisent. Ces évènements, identifiés, sont liés soit à des contextes techniques soit a des accès aux données des patients.

IHE définit deux types de formats de messages d'audit principaux : 1) "IHE Audit Message" qui résulte d'une combinaison de plusieurs standards comme DICOM[9] et qui étend le vocabulaire de base défini dans la RFC-3881[19], et 2) "IHE Provisional Audit Message" qui se présente sous forme d'un schéma XML provisionnel des contenus des enregistrements d'audit et est généré par des acteurs de radiologie IHE. Ce dernier format est moins flexible que le premier mais approprié pour le reporting de radiologie et autres activités de diagnostic et de traitement.

3.2.2 Modèle OrBAC

L'objectif d'un modèle de sécurité est de restreindre qu'aux utilisateurs autorisés les accès aux objets du système cible, par l'application d'un ensemble de règles constituant ce qu'on appelle communément la politique de sécurité.

OrBAC est un modèle qui permet d'exprimer une grande variété d'exigences de sécurité, mêmes les plus complexes comme celles que l'on peut renconter dans le domaine de la santé. Ce modèle tient compte des éléments contextuelles ce qui permet une définition très fine, dynamique et flexible de la politique de sécurté. C'est pourquoi OrBAC (et les raffinements qui lui ont été apportés, comme O2O [4] pour l'interopérabilité, AdorBAC [6] pour l'administration, hiérarchisation des rôles, activités, vues et contextes ainsi que la séparation des contraintes [5]) a été choisi comme modèle de spécification de la politique dans le cadriciel du contrôle d'accès *a posteriori* que nous avons défini. Les règles de sécurité dans ce modèle sont définies au niveau organisationnel. Ainsi, les sujets, les objets et les actions sont structurés respectivement en rôles, vues et activités. Chaque organisation, un concept central dans OrBAC, définit ses règles de sécurité qui spécifient qu'un certain rôle a la permission, l'interdiction ou l'obligation d'effectuer une certaine activité sur une certaine vue. Ces règles ne sont pas statiques puisqu'elles dépendent de conditions contextuelles au moyen de l'entité native dans le modèle, le "contexte". OrBAC définit quatre prédicats :

- $empower(org, s, r)$: dans l'organisation org, le sujet s est habilité dans le rôle r.
- $consider(org, \alpha, a)$: dans l'organisation org, l'action α implémente l'activité a.
- $use(org, o, v)$: dans l'organisation org, l'objet o est utilisé dans la vue v.
- $hold(org, s, \alpha, o, c)$: dans organisation org, le contexte c est vrai pour le sujet s, l'action α et l'objet o.

Les règles de sécurité abstraites OrBAC sont modélisées de la façon suivante :

$$security\_rule\_type(org, role, activity, view, context)$$

où $security\_rule\_type \in \{permission, prohibition, obligation, dispensation\}$

Par exemple, la règle de sécurité organisationnelle suivante :

$$permission(hospitalCHU, Dr\_Revel, consult, Alice\_medical\_record, emergency)$$

signifie que dans l'hôpital CHU, le médecin Revel a la permission de consulter le dossier médical de Alice dans le contexte d'urgence.

Les permissions, interdictions, obligations et dispensations concrètes qui s'appliquent aux triplets (sujet, objet, action) sont modélisées en utilisant le prédicat $concrete\_security\_rule\_type(subject, action, object)$ et sont dérivées des règles de sécurité organisationnelles. La règle de dérivation est la suivante :

$$security\_rule\_type(Org, R, A, V, C) \wedge$$
$$empower(Org, Subject, R) \wedge$$
$$consider(Org, Action, A) \wedge$$
$$use(Org, Object, V) \wedge$$
$$hold(Org, Subject, Action, Object, C)$$
$$\rightarrow concrete\_rule\_security\_type(Subject, Action, Object)$$

Où $concrete\_security\_rule\_type \in \{Is\text{-}permitted, Is\text{-}prohibited, Is\text{-}obliged, Is\text{-}dispensated\}$.

Les paramètres de ces régles de sécurité concrètes correspondent généralement aux types d'informations qu'un enregistrement de log peut contenir.

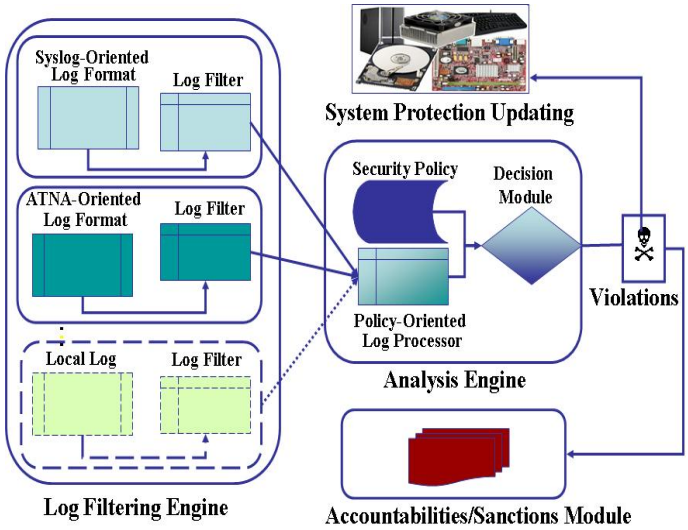

FIG. 3.1 – Cadriciel por le contrôle de sécurité *A posteriori*

Nous donnons dans la section suivante des éléments sur notre module de reformulation et restructuration des enregistrements d'audit ATNA sur la base de pérdicats d'expression des règles de sécurité. L'objectif est d'identifier et d'extraire les éléments OrBAC à partir des sénarios d'évènements ATNA. Cette structuration formelle des logs qui tient compte du modèle de la politique facilite la vérification comparative des règles de sécurité et les actions qui ont été enregistrées et améliore le processus d'analyse et de détection des anomalies de comportements (creation, accès, effacement ou modificaton non autorisés des informations de santé protégées).

3.3 Cadriciel pour le contrôle d'accès *a posteriori*

L'idée principale de notre approche est de structurer les logs de façon à les rapprocher des concepts important utilisés pour exprimer la politique de sécurité, sans pour autant modifier la façon dont le système d'information génère ses logs ni leur format. Cet effort de restructuration réduit le processus de détection des violation à de simples vérifications. Ainsi, notre cadriciel de contrôle d'accès et d'usage *a posteriori* (APAUC) nécessite trois composants : un moteur de logs, un moteur d'analyse et un module de sanctions (voir Figure 3.1).

Ce Cadriciel illustre les quatre étapes du processus APAUC. La première étape est assurée par les moteur de filtrage des logs (Log Filtering Engine). L'objectif est d'abord d'obtenir les logs à partir des différents systèmes d'informations (Ex. hôpital, laboratoire d'analyses) qui contiennent, collaborent et échangent des données devant être protégées (Ex. le dossier du patient). Chacun des ces systèmes peut utiliser des formats de logs particuliers (syslog, ATNA,

Common Logfile Format, etc.). C'est pourquoi les formats des enregistrements récupérés ne sont pas toujours similaires. De plus, ils sont, généralement, indépendants des politiques de sécurité associées. Habituellement, les éléments générés dans les logs sont paramétrés par l'administrateur (Ex. temps, date, adresse IP, etc.). Celui-ci se base généralement sur son expertise.

Ensuite, ces logs sont filtrés pour extraire les triplets $\langle subject_i, action_i, object_i \rangle$ ainsi que les évènements, états et méta-données utiles qui leur sont associés (comme les adresses IP des machines utilisées par $subject_i$ pour effectuer $action_i$, les estampilles de $action_i$, les sauvegardes du système avant que $subject_i$ effectue $action_i$ sur $object_i$, les numéros de sessions, etc.). Le résultat du filtrage des log doit faciliter la reconstruction de l'histoire du système relatif à un sujet donné ou une action donnée ou un objet donné.

La seconde étape, consiste à réécrire les logs (dans notre cas, nous nous intéressons au aux logs de type IHE-ATNA, un exemple est donné dans Figure 3.2) une fois filtrés en un format compatible avec la politque de sécurité du système implantée par le moteur d'analyse. Ces travaux de reconciliation des enregistrements de logs qui tiennent compte de la politique facilitent la détection des violations. Durant cette étape, nous appliquons un processus d'abstraction des entités élémentaires enregistrés dans les logs en débutant par le triplet $\langle subject_i, action_i, object_i \rangle$.

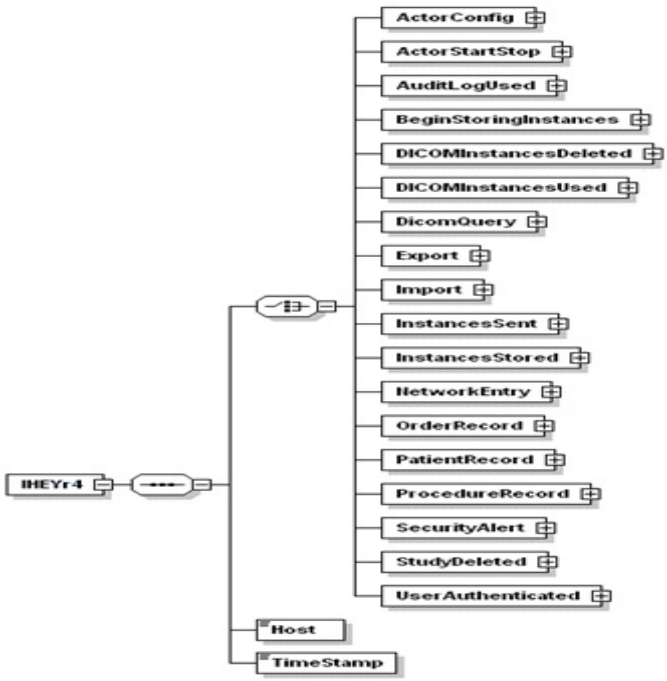

FIG. 3.2 – IHEYr4 Audit Record

L'objectif final est de retrouver les concepts organisationnels clés du modèle

de contrôle d'accès et d'usage choisi. Par exemple, si on considère le cas d'un modèle de contrôle d'accès RBAC, avec un ensemble de métadonnées comme le numéro de session, la date et l'action de logging d'un sujet, le processus retrouvera tous les rôles attribués à ce sujet dans ces circonstances. Cette tâche est assurée par le processeur de logs (Policy-Oriented Log Processor).

La troisième étape, consiste à déclencher le processus de vérification de la compatibilité des évènements, actions et transactions qui ont été enregistrés dans les fichiers de logs (donc qui se sont produits) avec la politique de sécurité mise en place. Par exemple, pour une action $action_i$ enregistrée dans les logs, effectuée par un sujet s_i durant une certaine session, le processus vérifie qu'il existe une règle de sécurité abstraite qui autorise un rôle R_i d'exécuter $action_i$ et le sujet s_i a été attribué à R_i. Si ce n'est pas le cas, APAUC déclenche une alerte de violation.

Enfin dans la quatrième étape, le module de sanction et d'attribution des responsabilités des anomalies détectées est utilisé pour qualifier les alertes déclenchées, les confirmer ou les infirmer en s'aidant des données contextuelles enregistrées dans les logs. Des normes comme HIPAA (Health Insurance Portability and Accountability Act), peuvent être utilisées pour aider à cette qualification. Si la violation est confirmée, la désignation des responsabilités, et des sanctions appropriées le cas échéant, doit être faite. En revanche, dans le cas où la violation est infirmée alors qu'un problème externe s'est réellement produit (Ex. une prescription médicale erronée), la politique de sécurité et les mécanismes de protection utilisés pour la déployer doivent être révisés (problème de consistance de la politique, attaque externes, virus, etc.).

3.4 Conclusion

Nous avons développé le processus de restructuration des logs (voir l'Annexe A). Les autres modules sont en cours de formalisation et de développements et devraient être achevés à la fin du projet. Les résultats seront consignés dans une version actualisée de ce livrable.

Contrairement aux travaux existants qui sont orientés langage, notre processus de contrôle est basé sur un modèle de sécurité contextuel possédant le niveau d'abstraction approprié qui lui donne tout son intérêt. Ainsi, les preuves de violation sont interprétables par un humain. Ce point est important puisque c'est le système organisationnel qui est la cible de ce type de contrôle et intéressera aussi bien les utilisateurs du système que ses administrateurs de sécurité. Un autre apport important de nos travaux est la capacité à faire converger les données de logs et les concepts organisationnel de la politique, ce qui facilite la détection des violations. Nous avons par ailleurs évité d'introduire toute contrainte concernant les formats de logs et la manière dont ils sont générés pour rester fidèles à la philosophie du contrôle d'accès et d'usage *a posteriori* qui se veut non bloquant, non contraignant. Des travaux d'implémentation sont en cours pour déployer une telle solution dans le CHU de Brest.

Deuxième partie : Tatouage et politique de contrôle d'accès et d'usage *a priori* et *a posteriori* pour la protection des images médicales

3.5 Introduction

Dans ces travaux nous proposons d'intégrer la technologie du tatouage (Watermarking) et un modèle de contrôle d'accès pour améliorer la protection des images médicales dans des infrastructures de santé distribuées. Pour cela, comme argumenté dans la première partie de ce chapitre nous faisons appel au modèle OrBAC pour l'expression de la politique de sécurité. Nous proposons d'introduire dans l'image certaines informations utiles qui seront utilisées pour contrôler et tracer l'accès à et l'utilisation qui est faite de cette image. Nous montrons comment un tatouage lâche et reversible peut être instrumenté par le modèle OrBAC. Cette reversibilité est une propriété importante parce qu'elle permet la préservation de la valeur du diagnostic tout en assurant des exigences de sécurité. Nous proposons un nouveau système de tatouage reversible pour illustrer notre watermarking OrBAC. Ce système de tatouage performant au regard des différentes modalités d'images médicales est analysé quant à ses capacités à transporter de tels éléments de contrôle de la sécurité dans les images.

3.6 Sécurité de l'information médicale

3.6.1 Information médicale

Une information est dite médicale, lorsqu'elle est destinée à la réalisation d'une action de type diagnostic, thérapeutique ou de prévention concernant la santé d'un individu ou d'un groupe d'individus [10]. Au sein de l'établissement de santé, l'information médicale constitue un bien essentiel pour le patient, l'organisation des soins et la prise en charge par les services. Elle doit donc être protégée de manière appropriée. Elle peut se présenter sous de multiples formes : imprimée ou écrite sur papier, orale, stockée sur des supports électroniques, transmise par la poste, par un réseau électronique ou par un réseau téléphonique, montrée sur des antécédents médicaux, des résultats d'analyse, des traitements en cours, des images médicales, des films ou des comptes rendus. Ces informations sont réunies dans le "Dossier Médical Personnel"(DMP), anciennement appelé "Dossier Médical Partagé". Le DMP a fait l'objet d'un projet public lancé par le ministère français de la santé visant à ce que chaque français dispose d'un dossier médical informatisé reprenant tout son passé et son actualité médicale. La première expérimentation a été réalisée en 2006. Afin de mettre en oeuvre le DMP, le projet a été relancé en avril 2009 et suivi par l'ASIP Santé (Agence des systèmes d'information partagés de santé) [1]. Si on considère les qualités suffisantes que doit posséder l'information présente dans le DMP pour être considérée comme valide, Dusserre [11] enonce : la pertinence,

la précision, l'actualité, la fiabilité, l'accessibilité, l'exhaustivité, la finesse de jugement du praticien. Quant à la sécurité des informations, la loi impose que trois propriétés soient assurées : la confidentialité, l'intégrité et la disponibilité.

3.6.2 Système d'information de santé

Un système d'information (SI) est un système qui permet le stockage, la consultation, la mise à jour, la communication et l'évaluation de l'information. Dans le domaine de la santé, l'information prise en charge est de nature et d'usage très divers et conduit à différents systèmes d'information, des cabinets de ville (médecins généralistes ou spécialistes) aux établissements de soins (hôpitaux, cliniques,...) ou encore des centres d'assurance maladie. Un système d'information au sens large peut donc recouvrir l'ensemble des applications informatiques hospitalières, la médecine de ville ou un réseau national de soins lorsque l'interopérabilité des dispositifs est assurée.

Dans le cadre de nos travaux, nous nous sommes intéressés particulièrement aux données de type imagerie médicale. Le PACS (Picture Archiving and Communication System) [22] est un système qui combine les matériels et les logiciels dédiés au stockage, la récupération, la gestion, la distribution, et l'affichage des images médicales acquises ou restituées à partir de différents phénomènes physiques (Résonance magnétique, réflexion d'ondes ultrasons, radioactivité, absorption des rayons X,...)). Les images électroniques et les rapports sont transmis numériquement par le PACS. Le format universel pour l'image pour son stockage et son transfert est le DICOM (Digital Imaging and Communications in Medicine) [7]. Les données qui ne sont pas des images, peuvent être incorporés à l'aide de format standards comme du PDF (Portable Document Format), une fois encapsulées dans DICOM.

Le PACS est constitué de quatre composantes principales : les modalités d'imagerie comme le scanner et l'IRM, un réseau sécurisé pour la transmission de l'information des patients, des postes de travail pour l'interprétation et la visualisation des images, et des archives pour le stockage et la récupération des images et des rapports. Combiné avec les technologies disponibles et le Web émergents, le PACS a la capacité de fournir un accès rapide et efficace à des images, des interprétations, et les données connexes.

3.6.3 Sécurité du SI et des données de santé

Comme le SI de santé ne diffère des autres systèmes que par certains types de données qu'il contient, les moyens de sécurité classiques éprouvés doivent être appliqués pour le protéger. Notamment, une politique de sécurité doit être définie pour spécifier les règles à appliquer pour gérer, protéger et diffuser les informations médicales sensibles au sein d'un établissement de santé et lors de ses communications avec d'autres systèmes d'information de santé ou d'autres professionnels de santé (réseaux de santé, praticiens libéraux, sécurité sociale, etc.). Une fois cette politique formellement spécifiée et sa cohérence vérifiée, il faut la mettre en oeuvre, i.e. configurer les composants de sécurité du système

afin d'appliquer la politique de sécurité sur le SI. Une partie de l'application de ces règles concerne la protection des données, d'autres concernent le système, les applications ou encore les communications.

Généralement, il est nécessaire de recourir à des applications qui implémentent des algorithmes cryptographiques garantissant la confidentialité des données échangées. Les données peuvent être le dossier du patient dans sa totalité ou quelques données cliniques comme les images, le type de données auquel nous nous intéressons.

Pour garantir l'intégrité des données lors de leur transmission ou leur stockage, un autre moyen cryptographique peut être utilisé : la signature numérique. Elle est basée sur le calcul puis le chiffrement d'une empreinte (un résumé ou hash du document original). Les chiffrements les plus robustes, mais coûteux, sont ceux dits asymétriques. Ces algorithmes ont la particularité d'utiliser deux types de clés distinctes : une clé privée, utilisée pour le chiffement (la signature) qui est connue uniquement par son propriétaire, à laquelle est associée une clé publique permettant de déchiffrer l'information, distribuée à ceux qui en font la demande. De cette manière, si un individu chiffre un résumé avec sa clé privée, les détenteurs de la clé publique peuvent déchiffrer le message. Cela permet l'authentification du signataire qui ne peut répudier le message puisque c'est sa clé, unique, qui a été utilisée. Une fois l'empreinte déchiffrée, il suffit de la comparer à celle recalculée sur le document reçu pour vérifier l'intégrité du message.

Le problème majeur de la cryptographie est qu'il faut déchiffrer le contenu avant toute utilisation, mais une fois déchiffré, les données ne sont plus protégées. Récemment, une nouvelle technologie a été proposée : le tatouage (watermarking).

3.6.4 Protection des données de santé par tatouage

En imagerie médicale, le tatouage d'une image médicale consiste à insérer dans l'image un message, par une modification aussi imperceptible que possible des pixels de l'image pour, par exemple, vérifier que l'image n'a pas été modifiée. L'information de protection, alors attachée au niveau signal, peut être retrouvée, même après un changement du format de stockage de l'image. Elle reste cependant accessible à quelques utilisateurs ou systèmes autorisés, c'est-à-dire ceux qui ont accès à la clé secrète de tatouage.

Généralisé à l'ensemble des données d'un dossier médical, le tatouage permet un niveau de protection complémentaire, ultime rempart pour garantir qu'un diagnostic médical n'a pas été changé, et assurer ainsi la fiabilité des informations du dossier du patient. Nous avons développé des solutions adaptées aux images médicales, essentiellement des méthodes de tatouage sans pertes (ou réversibles), exploitables dans le domaine de la santé et qui, en combinaison avec des outils cryptographiques, permettent de vérifier l'intégrité des images, voire de les tracer et garantir une protection continue de l'information.

Dans la littérature, il existe des techniques qui combinent le tatouage et la cryptographie. L'objectif est d'assurer la confidentialité du signal par l'utilisation du chiffrement et d'ajouter de nouvelles fonctionnalités en insérant des

méta-donnés pour des besoins d'intégrité et de traçabilité au moyen du tatouage. Par exemple, dans [21], Puech et al. proposent une méthode qui consiste d'abord à chiffrer l'image à l'aide d'un algorithme de chiffrement symétrique. Puis, double chiffrer la clé symétrique avec un algorithme asymétrique et la tatouer dans l'image chiffrée. Le récepteur, après avoir extrait la marque (clé symétrique doublement chiffrée), la déchiffre avec les clés appropriées et qu'il doit posséder. Il peut alors à la fois déchiffrer l'image et authentifier l'origine du message.

Comme le tatouage n'est efficace qu'en terme de détection, c'est à dire une fois que l'action malveillante se soit produite, il est clair qu'il faut toutefois des moyens d'empêcher réellement cette action illégale. Dans l'application de contrôle de la copie, Bloom et al. [3] préviennent les copies illégales de contenus protégés par le Copyright. Les marques sont intégrées dans le contenu lui-même, elles sont donc dans toute représentation de ce contenu et fournissent ainsi un meilleur moyen de mise en oeuvre du contrôle de copie. Il suffit donc de doter les dispositifs d'enregistrement d'un détecteur de marque, qui bloquerait toute copie à chaque fois qu'une marque "jamais de copie" est détectée.

Cependant, il faut noter que peut se poser le problème de la compatibilité des lecteurs/enregistreurs qui ne seront pas forcément capables de détecter ni la marque ni le chiffrement CSS[1](Content Scrambling System). Par ailleurs, ce type d'approche est assez statique. Dans le cas où, par exemple, la copie est autorisée dans des circonstances particulières, il faudrait un moyen de spécifier ces contextes particuliers d'autorisation pour valider ou refuser l'accès ce qui permettrait une sécurité dynamique. Nos travaux apporte la solution à ce problème tout en garantissant une mise en oeuvre performante de la sécurité par tatouage.

3.7 Notre modèle OrBAC tatouage

3.7.1 Hypothèses et éléments conceptuels

Nous supposons l'existence d'un moniteur de référence en charge de l'application de la politique de sécurité. Nous supposons également que lorsqu'un sujet souhaite effectuer une action sur un objet, ce sujet doit d'abord envoyer une requête à ce moniteur de référence. Ensuite, le moniteur charge la politique abstraite et évalue les contextes. Enfin, il fournit les décisions d'accès (acceptation, refus ou violation) en fonction de la validation des conditions contextuelles.

Nous pouvons distinguer deux types de moniteur de référence selon sa localisation : 1) SRM (Server-side Reference Monitor) situé du côté du serveur et 2) CRM (Client-side Reference Monitor) situé côté client. Dans le cas d'un SRM, la protection est centralisée. C'est la problématique du contrôle d'accès traditionnel. Dans le cas d'un CRM, l'information se trouve sur le poste client. Dans

[1]Content Scrambling System est le système de chiffrement utilisé pour protéger les DVD vidéo contre la copie illégale

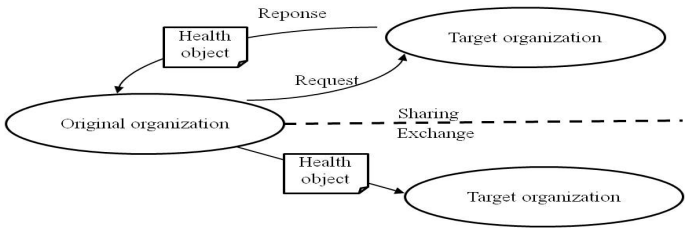

FIG. 3.3 – Exchange and sharing between two organizations.

ce cas, pour éviter que l'information soit accessible en contournant le moniteur de référence, l'information est chiffrée sur le client.

Nous supposons par ailleurs que chaque organisation est dotée d'un système qui stocke et gère les différents types d'information. Notamment et pour contrôler l'activation des contextes[5], ce système doit fournir les renseignements nécessaires pour vérifier si les conditions associées à la définition de contextes sont satisfaites ou non :

- Une horloge globale pour vérifier les contexte de type temporel.
- L'environnement, l'architecture logicielle et matérielle pour vérifier les contextes de type spatial.
- La base de données du système pour vérifier les contextes de type prérequis.
- Une journal du système pour vérifier les contextes provisionnels.

Si le système d'information ne fournit pas certaines de ces informations, les contextes correspondants ne peuvent pas être pris en compte par la politique de sécurité. Dans ce cas, plusieurs solutions sont possibles : 1) attacher les informations nécessaires à l'évaluation du contexte à l'objet transmis, 2) synchroniser régulièrement les données du SRM et du CRM, 3) utiliser la technologie de tatouage pour insérer les informations dans l'objet lui-même. Dans ce dernier cas, les informations utilisées pour évaluer le contexte et l'objet à protéger sont inséparables. Ils ne peuvent être disssociés que si l'utilisateur est muni d'un moniteur de référence compatible, donc possédant un module de tatouage intégré qui peut extraire la marque. Avec l'insertion de ces marques dans l'objet à protéger, les informations pertinentes notamment contextuelles peuvent être décodées et utilisées pour contrôler la sécurité d'accès et d'usage. De plus, ces marques peuvent être mises à jour par l'administrateur ou automatiquement par le système avant, pendant ou après l'accès et maintenir ainsi l'aspect dynamique de la politique.

Dans notre framework, les organisations de santé et praticiens peuvent partager ou s'échanger des informations sur les patients (voir Figure 3.3). L'interopérabilite entre ces organisations est asuurée par dans le cadre de nos travaux par l'application de l'approche O2O[4].

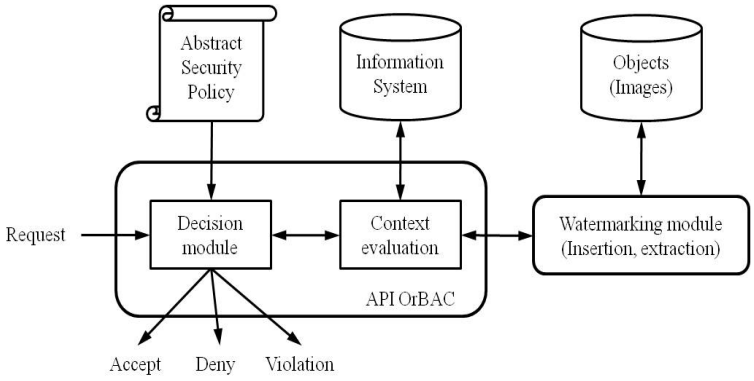

FIG. 3.4 – Framework OrBAC Tatouage.

3.7.2 Framework

Le modèle comporte trois composantes : Le module de politique de sécurité OrBAC, le moniteur de réference compatible (SRM ou CRM) et le module de tatouage (voir Figure 3.4.

Le module OrBAC (concrètement l'API OrBAC) charge et gère la politique de sécurité. Il évalue les contextes et fournit les réponses aux requêtes d'accès. Le module tatouage est chargé de l'insertion dans les images ou la mise à jour des informations utiles. Ces informations doivent auparavant être transformées et codées sous format binaire, qui inséré dans l'image est appelé marque. Ces marques peuvent être extraites au moment où le système a besoin d'évaluer les contextes ou vérifier l'intégrité et l'authenticité d'une image médicale.

La capacité d'insertion dans une image sans modifier le diagnostic n'est pas illimitée. Donc, nous avons selectionnés des éléments que nous avons jugés essentiels telles que le consentement du patient.

Afin d'exprimer les règles de sécurité au niveau organisationnel, nous faisons appel au concept de vue du modèle OrBAC. Ce concept est différent de la notion de classe. Une classe est utilisée pour structurer les descriptions des objets (groupe d'objets ayant les mêmes caractéristiques). En revanche, une vue est utilisée pour structurer la specification de la politique de sécurité (groupe d'objets auxquels s'appliquent les mêmes règles de sécurité). Dans OrBAC tatouage, nous avons créé au niveau abstrait la vue "Watermark" et plusieurs sous-vues associées :

- MarqueInteg, utilisée pour vérifier l'intégrité d'une image.
- MarqueIPP, utilisée pour vérifier l'authenticité d'une image.
- MarqueConsent, utilisée pour contrôler les droits d'accès et la distribution de l'image.
- MarqueTrace, utilisée pour les besoins de traçabilité.

Toutes ces marques constituent des informations nécessaires pour effectuer un contrôle *a posteriori*.

Au niveau concret, on considère qu'une marque est un objet dérivé. C'est à dire, il est créé à la suite de l'obtention de droits d'accès ou l'exécution d'actions sur un objet original. Par exemple, la lecture d'une image médicale entraîne la création d'une marque de type trace qui sera insérée dans l'image. Chaque marque doit avoir ses propres attributs. Ces attributs peuvent être mis à jour avant, pendant et après un accès ou une utilisation de l'image. Nous avons défini quatre classes d'attributs :

1. Watermark intégrité : (1 attribut)
   ImageDS : signature numérique de l'image (par exemple 256 bits selon SHA-256).
2. Watermark IPP : (1 attribut)
   IPP : identifiant du patient correspondant à l'image.
3. Watermark consentement du patient : (5 attributs)
   Authority : organisation dans laquelle ce consentement peut s'appliquer.
   Grantee : liste des rôles auxquels ce consentement a été attribué.
   Privilege : liste des activités permises par le consentement.
   GranteeException : liste des sujets interdits d'accès.
   Date : date limite du consentement.
4. Watermark trace (2 attributs)
   LogActor : liste des acteurs qui ont utilisé cette image.
   LogDate : liste des dates des dernières utilisations.

Nous avons par ailleurs besoin de mémoriser la liste des rôles : médecins, patients et la vue Image. Les sujets et objets appartenant à ces rôles et cette vue possèdent leurs propres attributs. Nous avons donc identifié trois autres classses d'informations à mémoriser dans la base de données du système d'information :

- infosMédecin : (1 attribut)
  ID_num : identifiant du médecin.
- infosPatient : (1 attribut)
  IPP : identifiant du patient.
- infosImage : (3 attribut)
  Addressee_subject : identifiant du sujet qui est supposé recevoir cette image et Addressee_role : son rôle
  IPP : identifiant du patient correspond à l'image.

3.7.3 Politique de sécurité

Nous distinguons trois catégories de règles de sécurité : *a priori*, temps réel et *a posteriori*. Nous décrivons dans cette section quelques unes des ces règles. Notons que pour certaines de ces règles, la spécification de contextes dynamiques est nécessaire. Dans le cadre des travaux présentés dans ce chapitre, nous ferons appel aux règles actives au travers du langage L_{active}[13] (voir dans l'Annexe B pour la spécification de l'ensemble des règles et plus de détails sur le langage L_{active} et l'utilisation que nous en avons faite).

Règle de type *a priori* : le médecin a la permission d'exporter une image vers une organisation cible si la WatermarkTrace a été mise à jour par le module de tatouage. En d'autres termes, l'identificateur du médecin et la date de l'application dans la liste des LogActor et LogDate ont été ajoutés. Nous faisons l'hypothèse que les autres contextes ont été validés par le système de manière classique.

L'objectif de cette règle est de faire en sorte que le tatouage soit une condition supplémentaire à valider avant d'autoriser toute action sur l'image. Cette règle est prise en charge par le moniteur de référence de l'organisation cible. La marque WatermarkTrace peut être extraite par le moniteur de référence de l'organisation originale pour un éventuel audit. Comme cette mise à jour doit se faire au moment de la requête, cela nécessite la définition d'un contexte dynamique.

Au niveau abstrait, nous spécifions cette règle de la façon suivante :

– *Permission(Org, Physician, Download, Image, WatermarkTraceUpdated)*

Le contexte *WatermarkTraceUpdated* doit être vérifié au moment de la requête, un contexte de type évènement doit être défini :

– $Hold_e$*(org, s, download_xx.dcm, xx.dcm,*
***start** (WatermarkTraceUpdated))*
after Do *(local_watermarking_plugin,*
update_watermarkTrace_xx.dcm,
watermarkTrace_xx.dcm)

Cette règle spécifie que dans l'organisation Org, le role $Physician$ a la permission d'effectuer l'activité $Download$ sur la vue $Image$ dans le contexte $WatermarkTraceUpdated$. Pour activer un tel contexte, dans org, le médecin s peut effectuer l'action $download\_xx.dcm$ sur Image $xx.dcm$ une fois que la marque de $xx.dcm$ a été mise à jour par le module de tatouage $local\_watermarking\_plugin$.

Après l'activation du contexte, nous utilisons les règles suivantes pour activer la permission :

– *Activate_Permission :* $Hold_e$*(org, s, download_xx.dcm, xx.dcm,* ***start** (WatermarkTraceUpdated))*
initiates *Insert_Permission (s, download_xx.dcm, xx.dcm)*
if *Permission(Org, Physician, Download, Image, WatermarkTraceUpdated),*
Empower(Org, s, Physician),
Consider(Org, download_xx.dcm, Download),
Use(Org, xx.dcm, Image)

La règle active *Activate_Permission* ajoute une permission concrète dès que le contexte abstrait s'active.

En temps réel, l'utilisateur est autorisé à avoir accès aux objets protégés sans aucun blocage. Mais, le moniteur de référence peut, à tout moment, extraire

la marque, vérifier les droits d'accès et décider de mettre terme à l'utilisation actuelle de l'image si les conditions d'autorisation ne sont pas satisfaites.

Règle de type *temps réel* : pendant l'affichage de l'image, le système peut extraire la marque WatermarkConsent de l'image. Si les attributs de l'utilisateur ne sont pas conformes aux attributs tatoués du consentement du patients, le système met fin à l'affichage.

Au niveau abstrait, nous spécifions cette règle comme une interdiction :

- $Prohibition(Org, Physician, Consult, Image, lecture\_Application \wedge WatermarkConsentViolate)$

Les contextes $lecture\_Application$ et $WatermarkConsentViolate$ sont modélisés de la façon suivante :

- $Hold(org, s, consult\_xx.dcm, xx.dcm, WatermarkConsentViolate) \leftarrow$
  $\neg Grantee(watermarkConsent\_xx.dcm, Physician) \vee$
  $(ID\_num(s, id\_s) \wedge GranteeExeption(watermarkConsent\_xx.dcm, id\_s))$
  $\vee$
  $\neg privilege(watermarkConsent\_xx.dcm, Consult) \vee$
  $Date(Global\_clock) > Date(watermarkConsent\_xx.dcm)$
- $Hold(org, s, consult\_xx.dcm, xx.dcm, lecture\_Application) \leftarrow$
  $Active\_Application(consult\_xx.dcm, xx.dcm)$

Pour permettre la supervision cette régle de sécurité, nous devons également spécifier les effects de l'occurence des actions sur l'état $Active\_Application$:

$Do(s, start\_Application, consult\_xx.dcm)$
causes $Active\_Application(consult\_xx.dcm, xx.dcm)$
$Do(s, end\_Application, consult\_xx.dcm)$
causes $\neg Active\_Application(consult\_xx.dcm, xx.dcm)$

3.8 Conclusion

Dans cette deuxième partie du chapitre, nous avons présenté nos travaux relatifs à la conception d'un modèle, une méthodologie et un mécanisme qui répondent au mieux à des exigences de traçabilité, d'intégrité des informations dans le cadre du domaine médical. Nous avons montré que la technique de tatouage, permet d'insérer des informations utiles dans l'image médicale pour contrôler l'intégrité, l'authenticité et les droites d'accès. Par ailleurs, nous avons également montré que les informations insérées dans l'image sont utiles pour d'éventuel contrôle d'accès et d'usage *a posteriori* afin de vérifier que chaque règle de la politique de sécurité est correctement appliquée et que l'ensemble des dispositions prises forme un tout cohérent. Le modèle OrBAC a été utilisé pour spécifier explicitement la politique de sécurité et instrumenter le tatouage sans contraindre les utilisateurs. L'API OrBAC joue le rôle de moniteur de référence en charge de l'application de la politique de sécurité. En combinant les deux dimensions, modèle d'accès et d'usage et mécanisme de tatouage, notre système offre une protection continue de l'information. Nous n'avons pas détaillé la mise

en oeuvre de la politique, l'implémentation du tatouge et les expérimentations menées, ils sont détaillés dans l'Annexe B.

Annexe A

Contrôle d'accès et d'usage *a posteriori* dans le domaine médical

Reconciling IHE-ATNA Profile with *a posteriori* Contextual Access and Usage Control Policy in Healthcare Environment

Hanieh Azkia, Nora Cuppens-Boulahia, Frédéric Cuppens and Gouenou Coatrieux
IT/Telecom Bretagne 2 Rue de la Châtaigneraie 35576 Cesson Sévigné
{*hanieh.azkia, nora.cuppens, frederic.cuppens, gouenou.coatrieux*}*@telecom-bretagne.eu*

***Abstract*—Traditional access control mechanisms prevent illegal access by controlling access right before executing an action; they belong to a class of *a priori* security solutions and, from this point of view, they have some limitations, like inflexibility in unanticipated circumstances. By contrast, *a posteriori* mechanisms enforce policies not by preventing unauthorized access, but rather by deterring it. Such access control needs evidence to prove violations. Evidence is derived from one or several log records, which trace each user's actions. Efficiency of violation detection mostly depends on the compliance of log records with the access control policy.**

In order to develop an efficient method for finding these violations, we propose restructuring log records according to a security policy model. We illustrate our methodology by applying it to the healthcare domain, taking care of the IHE (Integrating the healthcare enterprise) framework, particularly its basic security profile, ATNA (Audit Trail and Node Authentication). This profile defines log records established on the analysis of common health practice scenarios. We analyze and establish how ATNA log records can be refined in order to be integrated into an *a posteriori* access and usage control process, based on an expressive and contextual security policy like the OrBAC policy.

***Keywords*-Access control model, IHE-ATNA, Audit**

I. Introduction

Controlling access to data is an important issue in information security. It must be considered in various domains, such as banking, healthcare, and so on while being applied to different systems client/server architectures within distributed or centralized environments. In order to define access control rules, the system administrator has to specify which data can be invoked, by which user and for which operation. By this way, one can regulate data confidentiality and privacy, as security services. Confidentiality and privacy are both important issues especially in collaborative environment and distributed system where multiple users and systems access and exchange data.

A lot of works have been devoted to security models, and a number of policy models have been proposed to express users' privileges,e.g. DAC (Discretionary Access Control) [1], MAC (Mandatory Access Control) [2], RBAC (Role Based Access Control) [3] or OrBAC (Organization Based Access Control) [4]. In these models, a user's request is checked immediately before granting or denying data access and usage. Thus, such models refer to *a priori* protection mechanisms, an *a priori* point of view which has some limits. Actually, deploying such traditional security models is restricting in contexts where users act in unforeseen and exceptional circumstances, such as emergency situations in hospital. So this kind of access control is not flexible and not appropriate in such an environment. For example, it will deny access to unauthorized hospital staff while they have to handle a serious situation. At the same time, this approach creates a delay to verify authorization of each user's requests before granting access. So in some contexts, like emergencies, where time is vital, *a priori* access control is not always appropriate.

To solve these problems, *a posteriori* access control approaches have been proposed [5],[6] (see section IV) where policies are checked after granting access. This means that the user must account for his or her actions at a later time by providing proof that he or she was allowed to conduct these actions. This security check process performed afterwards appears more suitable in healthcare environments. For instance, regarding medical emergency situations, access will be given to each health professional who requests access. In other words, health professionals can continue their work, regardless of security policy rules. But if *a posteriori* access control detects an unauthorized action, the perpetrator can be held accountable for such actions.

In order to deploy *a posteriori* access control, three components are required: (1) Log process which records the history of actions executed by the system's actors. Logged actions, referred to as logs, constitute evidence that can be used to demonstrate either an actor was allowed to perform a particular action or not. (2) Log analysis (auditing), which is triggered to identify abnormal actions through the verification of logs, led by a set of security rules. (3) Accountability component, which takes as input abnormal actions and detects either that the misbehavior was authorized, i.e. the actor was allowed to carry out the action (depending on the context), or not. In the latter case, a violation occurred and the actor can be submitted to penalties. The effectiveness of this approach lies on how to identify misbehavior and violations.

In this paper, we focus on the log analysis component. This one contains three modules. The"Security Policy" container includes two types of policy rules: *a priori* and

a posteriori access and usage control rules. The "Policy-Oriented Log Processor" reconciles security policy with logged actions, generated automatically by the system. This transformation makes our violation detection process easier and more efficient. And the "Decision Module" checks whether those Policy-Oriented Logs are consistent with the organizational security policy or not. By contrast with previously proposed frameworks, our approach is not intrusive, as we use traces generated by the existing log module of the target system. This log module can be compliant with *d*e facto standard like SYSLOG [7], [8] or with standard related to a specific domain like IHE-ATNA [9]). Moreover, the security policy used by the violation detection process is specified in accordance with an access and usage control model to provide low-level policies abstraction and interpretation ease by a human administrator.

In this study, we consider the IHE (Integrating the Healthcare Enterprise) framework [10] which is recommended for the integration of medical information system. This standard defines different profiles to which an information system can be compliant or not. The IHE-Audit Trail and Node Authentication (ATNA) is one of these integration profiles. It provides privacy and security rules and mechanisms, such as an audit trail. An audit trail allows monitoring activities related to security, patient privacy, user authentication and access and usage authorization in distributed applications. ATNA defines one structure for the contents of the audit trail, and this structure is used as log format in this work.

We also consider the OrBAC model to specify security policies. Our motivation to choose this model is that OrBAC is an expressive security model that makes it possible to specify a large number of security requirements, sometimes complex, which can be found in the medical domain. It uses native structural concepts (like "organization" and "context") and includes different security modalities (permission, prohibition, obligation and dispensation). This makes it more suitable to deal with new requirements in comparison with other models. The central notion in OrBAC is Organization. An organization can be seen as an entity that is responsible for managing security policy (e.g. hospital, cancerology department, firewall). Another interesting notion in this model is context. Contexts are used to define conditions to activate security rules (e.g. emergency, patient consent). They also increase or reduce user privileges depending on various events or states (e.g. temporal, spatial, provisional, etc.) and thus the obtained security policy is more dynamic.

The remainder of this paper is organized as follows. Section II defines and describes the audit profiles and the security model used in our framework. Section III presents the framework we suggest to enforce *a posteriori* access and usage control, in particular the transformation process of IHE-ATNA logs into a format compliant with an access and usage analysis led by an OrBAC policy. Section IV recalls related works, whereas section V concludes the paper.

II. Audit profile and security model

A. IHE-ATNA

IHE is an initiative of healthcare professionals and industry, designed to stimulate the integration of information systems in the health care domain. Its objective is to ensure that all required patient information is both reliable and available to healthcare professionals. IHE provides a common framework that identifies a set of functional components of the healthcare institution called "IHE Actors", and specifies actor interactions referred to as "standards-based transactions". These interactions belong to common working scenarios, i.e. when handling medical information system and data in daily medical practice.

ATNA is one of the integration profiles that have been defined for two domains, namely IT infrastructure and Radiology. For example, the IHE radiology establishes radiology-specific audit trail messages and expresses the basic security measures to protect patient confidentiality, privacy and to ensure traceability. In fact, according to recent legislation on privacy and personal data security, it is essential to trace all actions that apply to subjects' personal data. We selected IHE's ATNA as one of the solutions to ensure traceability and security of both communications and data. The three key features of the ATNA profile are: User authentication, authentication of host during communications, and audit record generation. Here, our interest focuses on ATNA-Audit record generation, which provides the aforementioned evidence, i.e. the audit trail, for the security analysis process. Generation of an audit trail relies on two main steps: 1) identifying events, based on transactions and 2) recording the audit messages. Each transaction performed by an actor could trigger events and thus cause creation of an audit record. These records will be stored for purposes of analysis, in an audit record server (referred to as an audit record repository in ATNA), where they can be monitored in order to allow detection of abnormal behavior such as improper creation, access, modification and deletion of Protected Health Information (PHI). ATNA specifies the use of Reliable Syslog [8] as the transportation mechanism for logging audit record messages to the central audit record server. It also makes it possible to use BSD (Berkeley Software Distribution) Syslog [7] knowing that it has several limitations. Thus, the audit record server will support both audit Reliable and BSD mechanisms. An audit record in ATNA is a record of actions performed by users on protected health information. This action can be creation, deletion, modification, query or view. An audit record is created by IHE actors when an IHE auditable event occurs. Auditable events are attached either to a technical context, or to patient data access. Most of these events have been identified considering the field of radiology.

IHE defines several audit record message formats and an IHE actor can use one or more formats, all of them use

XML encoding and are defined by XML schema. There are two main types of format: 1) the IHE Audit Message Format, which is a combination of several standards (e.g. DICOM [11], IHE extensions) and which also extends the basic vocabulary provided by RFC-3881 [12] and 2) the IHE Provisional Audit Message Format, which is a provisional XML schema of audit record contents and is generated by the IHE radiology actors. The provisional format is less flexible than the IHE Audit Trail format, which is the preferred format, but it is suitable for reporting in radiology and other diagnostic and treatment activities. It can also be converted into an equivalent IHE Audit Trail format to reduce the burden on audit repositories.

B. OrBAC model

The purpose of a security model is to restrict an action on some objects to authorized subjects only, by applying security rules. OrBAC is a general model that is expressive enough to handle different and complex security requirements like those related to the healthcare domain. Specifically, this model is context aware meaning that it allows us to define fine grained, dynamic and flexible security policies. Moreover, this model can potentially cooperate with other security models to allow interoperability as well as to allow natively some specific administration features like delegation. The OrBAC model (and its refinements like O2O [13] for interoperability and AdorBAC [14] for administration) is the chosen model for policy specification in our framework, since it meets our requirements.

As said before, the concept of organization is central in OrBAC. Instead of defining security rules that directly apply to subject, action and object, access control is defined at the "Organizational" level. For this purpose, subject, action and object are respectively abstracted into role, activity and view in the appropriate organization. Each organization can then define security rules that specify that some roles are permitted, prohibited or obliged to carry out particular activities on particular views. Security rules do not apply statically but dynamically, as they may depend on contextual conditions. For this purpose, the concept of context is explicitly introduced in OrBAC. OrBAC defines four predicates:

- *empower(org,s,r)*: in organization *org* subject *s* is empowered in role *r*.
- *consider(o,α,a)*: in organization *org* action α implements the activity *a*.
- *use(org,o,v)*: object *o* is used in view *v*.
- *hold(org,s,α,o,c)*: in organization *org*, context *c* is true between subject *s*, action α and object *o*

Abstract security rules in OrBAC are modelled as a predicate with the five aforementioned parameters:

security-policy-type(org, role, activity, view, context) where security policy type belongs to {permission, prohibition, obligation, dispensation}

For instance, the organizational security rule: *permission(hospital CHU, Dr.Helia, consult, Alice_medical_record, emergency)* means that, in organization CHU, Dr. Helia is permitted to consult Alice's medical record in the context of emergency.

Concrete permissions, prohibitions, obligations or dispensations, that apply to triples $\langle subject, action, object\rangle$ are modeled using the predicate *concrete-security-policy-type(subject,action, object)*and logically derived from organizational security rules. The general derivation rule is defined as follows:

$$\begin{aligned}&\textit{security-policy-type}(Org, R, A, V, C)\wedge\\ &empower(Org, Subject, R)\ \wedge\\ &consider(Org, Action, A)\ \wedge\\ &use(Org, Object, V)\ \wedge\\ &hold(Org, Subject, Action, Object, C)\\ &\rightarrow\quad \textit{concrete-security-type}(Subject, Action, Object)\end{aligned}$$

where *concrete-security-type* belongs to $\{Is\text{-}permitted, Is\text{-}prohibited, Is\text{-}obliged, Is\text{-}dispensated\}$.
Typically, the parameters of these concrete rules correspond to a kind of information that we can find in a log records.

Other useful features are also defined in the OrBAC model, such as hierarchy of roles, views, activities and contexts and separation constraints (interested readers can refer to [4] for more details).

In the following section, our approach demonstrates how ATNA audit records can be reformulated or restructured making use of OrBAC predicates. The objective is to distinguish and extract OrBAC elements from ATNA event scenarios. Having a formal log structure based on a policy model facilitates checking the restructured log with policy rules, as well as improving the effectiveness of the analyzing process that detects non-compliant behavior (improper creation, access, modification and deletion PHI).

III. Our proposal

The core idea of our approach is to structure the logs in order to bring them close to the security policy's relevant concepts, without modifying either the way the logs are generated or their format. In this way, the violation detection process comes down to a mere verification. Thus, our *a posteriori* access and usage control framework (APAUC) requires three components: a log engine, an analyzing engine and an accountability/sanction module (see Figure 1).

This framework depicts the four steps of the APAUC process. The first step is conducted by the "Log Filtering Engine". The objective is first to obtain logs from various Information Systems (e.g. hospital's IS, analytical laboratory's IS) which collaborate, contain and exchange data that must be protected (e.g. patient record). Each of these systems may use special log format (syslog, ATNA, Common Logfile Format, etc.). For this reason, log record formats are not always similar. Moreover, they are independent of the security policies of the associated IS. In fact, the elements

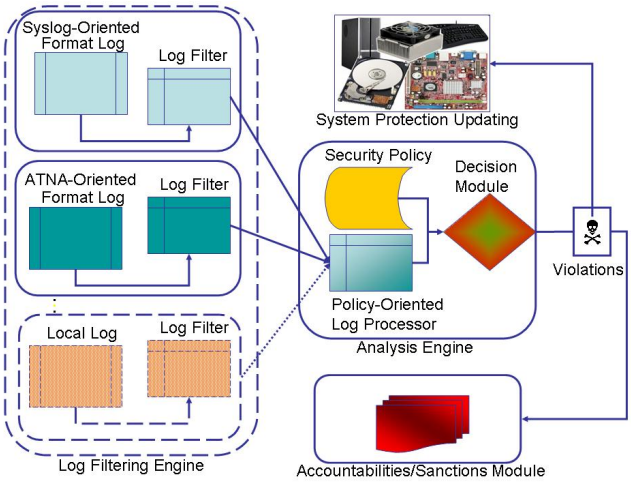

Figure 1. APAUC Framework

that must be generated in the logs are configured by the administrator (e.g. time, date, IP,etc.). They are generally based on the administrator's expertise and on what appears useful to him.

Then, these logs are filtered to extract triples $\langle subject\_i, action\_i, object\_i \rangle$, their associated relevant events, states and metadata (IP of the host used by $subject\_i$ to perform $action\_i$, timestamp of $action\_i$, backup of the system performed before $subject\_i$ performs $action\_i$ on $object\_i$ if any, session number, an so on). The result of the log filtering must ease the construction of workflows or system's history related to some $subject\_i$, $action\_i$ or $object\_i$ from some specified point to another.

The second step consists in rewriting the filtered logs obtained during the first step in a format compatible with the policy model implemented in the analysis engine. This policy-aware reconciliation of log records facilitates violations detection. During this step, we perform an abstraction process of the logged elementary entities, starting with the triples $\langle subject\_i, action\_i, object\_i \rangle$. The ultimate goal is to meet the key security organizational concepts of the chosen access and usage control model. For instance, if we consider the case of the RBAC model with a given set of metadata like a session number, a date and a logged action of $subject\_i$, the process will retrieve all the roles that have been assigned to this subject in these circumstances. This is handled by the Policy-Oriented Log Processor.

In the third step, we trigger a compliance verification process of events, actions and transactions that occurred and that have been logged. For instance, for a logged action a_i performed by a subject s_i during some session, this process verifies that there exists an abstract security rule which authorizes some role R_i to do a_i and s_i was assigned to R_i. If it is not the case, APAUC raises a violation alert.

Finally, in the fourth step, the Accountability/Sanction module is used to qualify the raised violations, confirming or discarding them, based on particular contexts. Norms like HIPAA (Health Insurance Portability and Accountability Act of 1996, issued by the U.S. Department HHS) can be used as input to help this qualification. If the violation is confirmed, responsibilities and appropriate penalties, if any, must be specified. When the violation is discarded while an external problem has been raised (e.g. wrong medicine prescription), the security policy and the protection mechanisms that have been used to deploy it must be checked (external attack, viruses, policy inconsistency).

A. *Policy-Aware Restructuring of IHE-ATNA Logs*

As our approach is applied to healthcare systems, we use IHE-ATNA profile as logging format. The reformatting procedure maps the relevant structures and contents of these standards to organizational security relevant concepts of the selected security model, OrBAC in our case. For this purpose, we choose IHE ITI technical framework supplement 2004-2005. This framework describes the "IHE Provisional Audit Message format". It contains two structural elements: "auditable events" and "audit record elements" which represent trigger events, and audit record objects respectively. IHEYr4 is presented in this version as a complete audit record payload (see figure 2). IHEYr4 is a triple composed of the original host, the event occurrence time and a fixed number of auditable event types with related details. These events describe either technical context (e.g Actor Start Stop) or patient information (e.g. Patient Record Event)

Now, to restructure these audit records into the policy model security concepts, we first have to consider a derivation and abstraction procedure for each auditable event type. Then, for each corresponding type of audit message, the procedure analyzes and extracts elements according to the OrBAC predicates (Organization, Role, Activities, Views and Contexts). So in our algorithm, we assume the "message" as input data, the "mapping function" as instruction and OrBAC parameters as output data. We define them as follow:

General algorithm:

MESSAGE: is a triple composed of <Auditable Event, TS, Host>

MAP: mapping function between auditable event and their sub-algorithm

Let event be the Auditable Event of the given message and sub-algorithm be the corresponding algorithm, then OrBAC elements are the outputs.

For instance, let us consider the ActorStartStops audit record (see figure 3) which is also the second event type in IHEYr4. This record is generated at the startup or shutdown of any application or actor. It has three elements:

- ActorName is the unique name on each host. Associating one name to one host can be used to identify specific hardware, software, or usage.

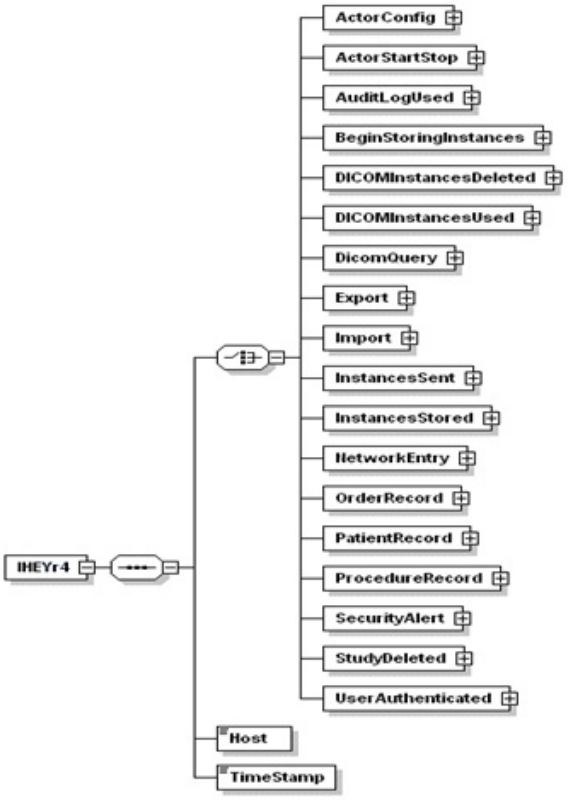

Figure 2. IHEYr4 Audit Record

Algorithm 1 General algorithm

Require: map: Map<Auditable Event, Sub-algorithm>
input: message : <Auditable Event, TS, Host>
event: Auditable Event ← message.getEvent()
algo: Sub-algorithm ← map.get(event)
output : OrBAC elements=algo(message)

- ApplicationAction describes the running status of an application or actor for the event ActorStartStop. And its legal values are Start or Stop.
- User identifies a user that performs activities which generate audit records. The user identification is either a username for LocalUser or, if a remote machine acts as a user, a remote node identification.

According to our general algorithm, ActorStartStop algo-ACTSS below, extracts the OrBAC elements from the message.

As another example, let us consider the Study Deleted audit record (see figure 4). This audit record is generated whenever a patient study, or a portion of it, is deleted from the archive. This audit record contains eight elements. The deleted object is always identified by its SUID (Study UID). CUID and MPPSUID, defined below, can optionally be recorded.

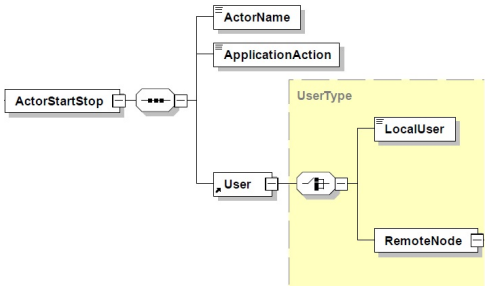

Figure 3. ActorStopStart

Algorithm 2 algo-ACTSS

Require: message (ActorStartStop, Host, Timestamp)
Subject ← values of ActorStartStop.User
Action ← values of ActorStartStop.ApplicationAction
Object ← values of ActorStartStop.Actor Name & value of Host
Context ← type of ActorStartStop.User(Local,Remote) &value of Timestamp
return (org,subject,action,object,context)

- ObjectAction: describes the action performed on an imaging object. The legal values are Create, Modify, Access and Delete.
- Accession Number: a string which presents a DICOM accession number.
- SUID: Study Unique IDentifiers is a single identification number which identifies a whole examination, in time and place.
- Patient: a free text that describes a patient.
- User: identifies a user who performs activities that generate audit records.
- CUID: Sop (Service object pair) Class Unique IDentifiers, which identify the type of service for which the image is intended.
- Number of instance: The attribute for each instance action corresponds to one specific instance and should be unique inside the specific series (identified by the Series Instance UID).
- MPPSUID: Unique IDentifiers of a Modality Performed Procedure Step which is used for sending all study information such as acquisition information, patient details and image information, to the radiology server upon completion of the examination.

According to the general algorithm, the StudyDeleted algorithm named algo-StDeld extracts the OrBAC elements from the message as below:

Afterwards, once the extraction and mapping procedure is achieved, we trigger the abstraction process using, in that case, the logged administration actions, mainly the role assignment, activity assignment and view assignment actions. Actually, the matter is to retrieve *empower(org,s,r)*,

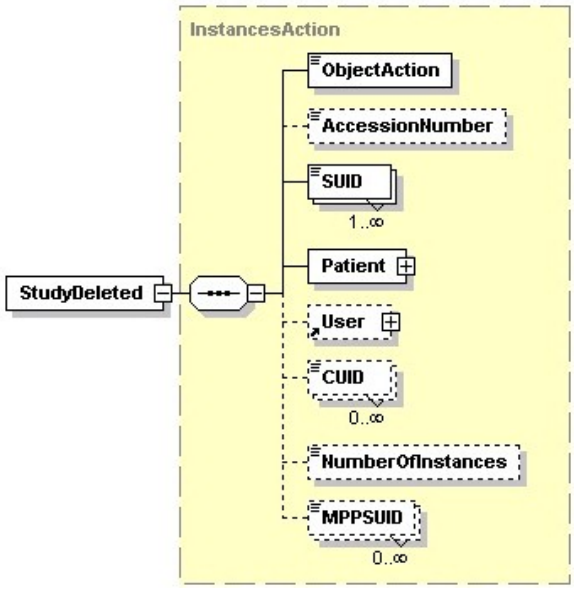

Figure 4. StudyDeleted

Algorithm 3 algo-StDeld

Require: message (StudyDeleted, Host, Timestamp)
Subject ← values of StudyDeleted.User
Action ← values of StudyDeleted.ObjectAction
Object ← values of StudyDeleted.Patient
Context ← values of StudyDeleted.SUID& type of ActorStartStop.User(Local,Remote) &value of Timestamp
return (org,subject,action,object,context)

consider(o,α,a) and *use(org,o,v)* seen in section II-B. By doing so, we obtain an abstract version of the logs where, instead of a record log specifying, for instance, "Helia modifies PH010 on Wednesday 30th august 2009", its abstraction version which specifies: a doctor (his role) performs an updating (the activity implemented by the modifying action) of prescription (the view object PH010 belongs to) and the doctor was on call when performing the updating (which is the active context when the doctor performs the updating activity). The analysis process then has mainly to check if there exists such a permission (due to space limitations these algorithms are not detailed here).

B. Analysis Process Activation

The analysis process to detect violations may be activated by a temporal or a non temporal event. If the activating event is temporal, analysis is triggered every day, week, month, etc. The security administrator fixes such periodicity. In this case, the scope of investigation is wide and the analysis must consider all the logged events and actions since the last audit. This is closer to an analysis and risk assessment than *a posteriori* access and usage control; the philosophy is different, though we can reach the same objective.

Examples of non-temporal events include: complaint, medical error, medicine poisoning, deletion of a patient's record, and so on. In this case, the analysis investigates only the logs related to a specific subject, action or object concerned by the activating event. As the healthcare staff is composed of trusted subjects, the analysis is really performed either to give evidence that 1) the concerned subject has not violated the policy; or 2) he or she did violate the policy but there are reasons making this violation null and void, so he or she is accountable for his action but he or she is not sanctionable; or 3) he or she did violate the policy but there are no mitigating circumstances and in this case he or she is both accountable and sanctionable.

IV. Related Work

For a long time, and it is still mostly the case, computer systems used logs to restore databases and file systems to consistent states after crashes. They also used them for security purpose, and the auditing of these logs has been used to enforce security since 1980. Providing a policy language whereby an auditor may prove that an obligation has been fulfilled by analyzing a log of actions has been proposed by Corin *et al* [5] and Cederquist *et al* [6]. In these works, a policy language, a logging mechanism and an audit procedure were defined. In [5], the authors went beyond suggesting a global framework, essentially developing the core notions of data and agent accountability. In [6], the policy language was extended to allow variables and quantifiers. This makes it possible to define a fundamental rule that gives the ability to refine policies. Using the same principle, Cederquist *et al* [15] presented a distributed framework that uses an *a posteriori* auditing approach. For auditing, they introduce three functions, namely observability, conclusions and proof obligations. They also provide implementation of a proof checker and a proof finder in the Twelf logical framework. The former allowing the auditor to check the justification given by agents while the latter allowing agents to justify an action they have performed to answer the auditor's request. Dekker *et al* [16] implemented *a posteriori* access control in the healthcare domain. They outlined the full architecture needed for audit-based access control of electronic healthcare record systems and discussed the advantages and limitations of this approach. Etalle *et al* [17] presented an approach similar to that of Cederquist *et al* for *a posteriori* compliance based control. They combine logic and trust management language. This approach is less specific with regard to the expressive power of the policy rules, but it is more precise with regard to how the policies appear in the system, namely as sticky policies attached to the data items. Intrusion detection that focuses on anomalous behavior has also driven research in auditing and logging. More often it is mistaken for *a posteriori* access and usage control. Though the general objective is the same, the difference is essentially due to the following: 1) the

environment where the later is deployed is trustworthy and generally reliable, and 2) the objective of the former is to stop attacks whereas the objective of the latter is rather to gather evidence and avoid sanctions.

Our work is not intrusion detection-oriented, as our objective is not to stop ongoing attacks; we consider trustworthy environments where *a posteriori* control makes sense and where the notion of attack is almost asemantic. Contrary to the aforementioned works, 1) our main concern is to rely on a security policy model, flexible and expressive enough to allow us to manage different security requirements, like those encountered in the healthcare domain; 2) this model is also human-interpreted, so the detected violations and evidence are readable; and 3) our approach is not intrusive, in that it does not introduce any constraints regarding subjects concerned by the *a posteriori* control, so they do not have to log evidence themselves, and we do not impose any log format.

V. Conclusion

In this paper, we present a framework and processes to enforce and manage *a posteriori* access and usage control. Contrary to previous work that mainly focuses on security language, our security control process is based on a contextual security model having an appropriate level of abstraction. Thus, evidence of violation or not is human-interpreted. This is an important point, as performing *a posteriori* security control in a trustworthy environment targets essentially the organizational system, which will hopefully have a positive impact on the information system regarding security enforcement. The significance of this paper lies in its ability to converge logging data and policy structural concepts. This eases the detection violation process. We were careful not to introduce any constraints due to logs' format and on the way they are generated in order to satisfy the principle of compliancy attached to the *a posteriori* access and usage control philosophy. The deployment of our solution is ongoing in the information system of the CHU of BREST, a French university hospital in Brittany.

Acknowledgement

The work presented in this paper is supported by a grant from The Brittany Region, France, and by funding from ANR SELKIS project.

References

[1] Department of Defense Trusted Computer System Evaluation Criteria, CSC-STD-011-83, Department of Defense Computer Security Center, Fort Meade, MD August 1983.

[2] D. Bell and L. LaPadula, *Secure Computer System: Unified Exposition and Multics Interpretation*, Technical Report ESD-TR-75-306, MTR-2997, MITRE, Bedford, Mass, 1975.

[3] D. Ferraiolo and R. Kuhn, *Role-Based Access Controls*, 15th NIST-NCSC National Computer Security Conf., Baltimore, MD, 1992.

[4] F. Cuppens and N. Cuppens-Boulahia, *Modeling contextual security policies.*, In International Journal of Information Security, 7(4): 285-305, 2008.

[5] R. Corin, S. Etalle, J. den Hartog, G. Lenzini and I. Staicu, *A logic for auditing accountability in decentralized systems.*,In: T. Dimitrakos, F. Martinelli ed, IFIP Workshop on Formal Aspects in Security and Trust (FAST). Vol 173, Springer, Berlin 2004.

[6] J.G. Cederquist, R. Corin, M.A.C. Dekker, S. Etalle, J. den Hartog, *An audit logic for accountability.*, In:A. Sahai, W.H. Winsborough ed, International Workshop on Policies for Distributed Systems and Networks(POLICY), IEEE Computer Society Press 2005.

[7] C. Lonvick,*The bsd syslog protocol, RFC 3164*, August 2001.

[8] D. New and M. Rose, *Reliable delivery for syslog, RFC 3195*, November 2001.

[9] Integrating the Healthcare Enterprise, *IHE IT Infrastructure Technical Framework Supplement 2004-2005 Audit Trail and Node Authentication Profile (ATNA)*, August 2004.

[10] Integrating the Healthcare Enterprise, *IHE IT Infrastructure Technical Framework Volume 1 (ITI TF-1) Integration Profiles*, August 2009.

[11] DICOM Standards Committee, Working Group 14, *Digital Imaging and Communications in Medicine (DICOM) Supplement 95: Audit Trail Messages*, Rosslyn, Virginia USA, June 2004

[12] G. Marshall, *Security Audit and Access Accountability Message XML, RFC 3881*, September 2004.

[13] F. Cuppens, N. Cuppens-Boulahia, C. Coma, *O2O: Virtual Private Organizations to Manage Security Policy Interoperability*, ICISS 2006

[14] F. Cuppens, A. Miège, *Administration model for Or-BAC*, Int. J. Comput. Syst. Sci. Eng. (CSSE), 2004

[15] J.G. Cederquist, R. Corin, M.A.C. Dekker, S. Etalle, J. den Hartog, *The audit logic*, Technical Report, 2006

[16] M.A.C Dekker and S. Etalle, *Audit-based access control for electronic health records*, Electronic Notes in Theoretical Computer Science, 2007.

[17] S. Etalle, and W. H. Winsborough, *A posteriori compliance control.*, In Proc. of the 12th ACM Symposium on Access control models and technologies ed. New York, USA, 2007.

[18] TL. Tsai, ML. Pan and DM. Liou, *Implementation of an IHE ATNA-Based Electronic Health Record System*, Institute of Biomedical Informatics, National Yang-Ming University.

Annexe B

Tatouage et contrôle d'accès pour la protection des images médicales partagées

Watermarking to Enforce Medical Image Access and Usage Control Policy

Wei Pan, Gouenou Coatrieux, Nora Cuppens-Boulahia, Frédéric Cuppens and Christian Roux
Institut Telecom; Telecom Bretagne;
Technopôle Brest-Iroise, CS 83818,
29238 Brest Cedex 3 France
Université Européenne de Bretagne, France
Email: {wei.pan, gouenou.coatrieux, nora.cuppens, frederic.cuppens, christian.roux}
@telecom-bretagne.eu

***Abstract*—In this paper, we propose integrating watermarking technology and access control model in order to enhance the protection of medical images in distributed healthcare infrastructures. For that purpose, we consider the well known organization based access control model (OrBAC) to express policy due to the intrinsic dynamic nature of the healthcare environment. OrBAC advantage is that it can specify contextual authorizations and obligations using "context", a native notion of this model, which may be viewed as a set of conditions to be satisfied when activating a given authorization or obligation. We suggest the embedding of some useful information into one image, information that can be used for tracing or controlling the access/usage of this image. In a more general way, we discuss how lossless or reversible watermarking can be instrumented by the OrBAC model. The lossless property preserves the image diagnostic value but imposes some other constraints. We illustrate the potential of such a watermarking OrBAC access and usage rules based on a new reversible watermarking scheme we proposed. Based on its performances regarding different image modalities, we analyze how this one can be used to convey such access control elements.**

I. Introduction

Concomitance of multimedia information with communication technologies has boosted the potential of medical data usability and applications. The deployment of telemedicine applications which primarily concern is offering to health professionals remote supports for consultation, expertise sharing and so on is an example of this trend. If electronic health records are more and more distributed, records outsourced from their information system are no longer under control. In fact, information or metadata required for evaluating conditions of access to medical records is most of the time unavailable or lost after one transaction between different and unnecessary compliant information systems. This fact raises new security challenges as content protection based on"classical" access control mechanisms appears no longer sufficient. Thus, there exists a need for developing security mechanisms that guarantee protection of medical contents in an autonomous way, especially in terms of traceability.

Watermarking has been shown as a complementary mechanism to enhance medical data security [1]. Generally speaking, watermarking allows hiding a message, also called a watermark, into a host document by modifying the host content in an imperceptible way. For one image, the message is attached at the signal level by slightly modifying its pixel gray level values. In this way, the embedded or watermarked message and the host image are intimately associated independently of the image file format. Originally, watermarking has been proposed as a mean to convey information about content ownership and intellectual property, such as copyright protection. Since then, it has been declined for different domain and purpose. For content verification applications, the watermark can indicate the image's origins and whether the document has undergone modifications. Watermarking can also be used to establish a channel for additional metadata (data hiding) [2]. From the security point of view, applications of watermarking could be considered as an a posteriori protection mechanism as by definition the watermarked host content can be accessed and/or handled as the original host is access free for all users but the control is performed after usage by analyzing the audit trails (records) and logs. Thus, the watermark that contains the useful information about the content is extracted whenever an audit is triggered. However, watermarking can also provide a priori protection to prevent illegal actions. In the framework of copy control or fingerprinting [3], the aim is to prevent people from making illegal copies of copyrighted content. Even if the first line of defense against illegal copying is encryption, once decrypted a watermark identifying the user and stating that a number of content copy is allowed might dissuade users of illegal redistribution. The user informed that the content is protected, knows that he or she will be identified and prosecuted. At the same time, if every recording device is fitted with a watermark detector, devices could prohibit recording whenever a never-copy watermark is detected. This mechanism has been suggested in digital rights management (DRM) which aims to protect the content owners [4] [5].

In medical imaging, several watermarking applications have been pointed out:

1) authenticity control with the insertion of data confirming the origin of the image, such as UIDs (Unique Identifiers) provided by the DICOM (Digital Imaging and COmmunications in Medicine, the standard of reference for medical image storage and sharing (medi-

cal.nema.org)) header [6] and the fact that a certain images refers to a particular patient;
2) Integrity control by inserting control information, such as a digital signature within the image [7];
3) the addition of meta-data (data hiding), allowing the content of images to be enriched by inserting a semantic description of the content [8].

In this work, we extend copy control framework with the idea of embedding more useful data in order to enforce secure access and use of protected data. To conduct this task, we need well-defined security policies based on an access control model that can express "usage" decisions more comprehensively. On this basis, we can then determine which information to embed within one image and how to use it in a distributed context.

The security policy aims at regulating information and system access. Its final objective is to specify a set of security rules such as permissions, obligations and prohibitions that control actions performed by subjects (active entities) on objects (inactive entities). Several access control models have been proposed to implement such a security policy. In traditional models, such as discretionary access control (DAC) [9], it is obliged to list permissions for all the triples (subject, object, action). When new subjects, new objects or new actions are introduced in the system, it is necessary to update the security policy by recording permissions granted to these new entities. As a result, the security policy quickly becomes complex to express and manage. In Mac policies [10], objects' classification and subjects' clearances must be specified and managed cautiously. The flow information control laid down by this model is very strict and not appropriate in a (distributed) healthcare environments. The Role-Based Access Control (RBAC) has been proposed to solve these problems. It introduces an abstraction of subjects to roles [11]. This abstraction is a first step in structuring the security policy, but it is no longer sufficient to face the complexity of large systems and highly distributed environment. In fact, security rules in these policies are no longer static but dynamic and depend on the context. To deal with these new requirements, the organization based access control (OrBAC) model [12] has been proposed. The concept of organization is central in OrBAC. Intuitively, an organization is any entity that is responsible for managing a security policy. To complete such an abstract structure in OrBAC, the security policy does not directly apply to subject, action and object at the concrete level. Instead, it defines permissions that apply within an organization to control the activities performed by roles on views, which are abstractions of action, subject, and object respectively. The OrBAC model can also specify contextual authorizations using the notion of context viewed as a condition to be satisfied when activating a given authorization. Moreover, the OrBAC model supports an approach called O2O (for Organization to Organization) to manage interoperability between components having their own policies defined by different organizations [13]. Through O2O, each organization can define and enforce its own secure interoperability policy. But, there remains one issue: information systems which store access control elements required for context valuation is different according to the target organization. By using watermarking, one may embed these elements into images in one organization, thus within another organization (at the user side), elements could be extracted from the image by the watermark reader and then used to valuate the context and check privileges.

In this paper, we propose to use the OrBAC model to guide reversible watermarking mechanism for protecting objects like medical images in distributed healthcare infrastructures. This protection can be a priori, preventing illegal actions on object, but also a posteriori, ensuring traceability in that case. We also present a watermarking scheme and discuss how and which access control elements can be practically conveyed by this method.

Before introducing our model in section IV, we present the reversible watermarking scheme we propose in section II. Section III briefly recalls the OrBAC model. In section V, analysis of experimental results is presented. Conclusions are made in section VI.

II. Reversible Watermarking

Generally speaking, a watermarking method is designed to satisfy a compromise between robustness, imperceptibility and capacity requirements, requirements which are usually imposed by the application framework. For any method, the strength of watermarking provides greater resistance to attacks, malevolent or innocent, but at the price of compromising imperceptibility. The information capacity is the length of the message which can be embedded within an image. Capacity is usually expressed in bits of message per image pixel (bpp). Obviously, capacity and robustness are inversely proportional.

In healthcare, watermarking or more generally techniques for hiding data in medical images must address more than one application and satisfy these requirements. The top requirement, however, is the preservation of the diagnosis value of the image. The medical world is very cautious for the compromising of any information that may potentially serve to diagnostic purposes. At the same time, once the physician or, in our case, the radiologist has given his/her interpretation of the image; he/she is legally responsible. As example, if a patient accidentally died, responsibilities will be established. Thus, the watermark should not interfere and should preserve the image interpretation.

One first class of methods that satisfy this imperceptibility requirement differentiates regions of interest (ROI) within images and then sets out watermark within region of non-interest (RONI). Since only the non-relevant parts of the image are manipulated, the diagnostic capability is not compromised. Investigations have suggested that RONI corresponds in general to the black background of the medical image [14]. One second class of approaches consists in using classical watermarking methods while minimizing the distortion. In this case, the watermark replaces some image details or characteristics definitively altering the image content [15].

Psychovisual masking is one tool that can be used to control image distortion but, to our knowledge, none models have been proposed in medical imaging. The last class of approaches corresponds to reversible watermarking, a concept introduced by Mintzer *et al.* [16]. Reversible or lossless watermarking offers the possibility to reconstruct the original image from its watermarked version by reversing watermarking distortion. Different from the classical watermarking methods, reversible watermarking can update the watermark at any time without adding any new distortions to the image. However, it must be noted that if the reversibility property is very helpful, as the watermarking invisibility constraint is relaxed, it may also introduce discontinuity in data protection. In fact, once the watermark is removed the image is no longer protected similarly to data encryption. So even if watermark removal is possible, imperceptibility has to be guaranteed as most applications have a high interest to keep the watermark into the image as long as possible, thus continuously protecting the information [1].

In [17], we have proposed a lossless watermarking scheme which makes use of an image of reference, derived from the image itself, and which has the property of being invariant to the insertion process. This image of reference is exploited to decide whether or not a pixel block can be modified by adding a watermark through a classification process. In this scheme, message embedding is conducted in Haar wavelet coefficients. According to the linearity of the wavelet transformation, the invariance of the image of reference and consequently the classification is preserved. Our embedding scheme offers a high capacity and low distortion for the medical images. It should be noted that most of reversible watermarking methods are known to be fragile, i.e. the watermarks does not survive any image alteration. This is why these methods are at first proposed for data integrity control. For example, to offer an integrity control service, a hash of the image is calculated making use of a cryptographic hash function and is then embedded into the image. At the verification stage, the watermark reader extracts the embedded hash while removing the watermark from the watermarked image. Then, the restored image hash is recomputed and compared to the extracted one. If these hashes are equal then the integrity of the image is verified.

In this work, we consider such a lossless watermarking scheme in order to insert metadata for helping access control in distributed healthcare infrastructures. In the next section, we present the OrBAC model and then identify which interested information will be embedded into medical images.

III. The OrBAC Model

A. Specify Security Policy

The OrBAC model specifies the security policy at the organization level, level independent of the implementation of this policy. It introduces an abstract level through three entities: role, activity and view. A role is a set of subjects to which the same security rules apply. Similarly, an activity is a set of actions and a view is a set of objects to which the same security rules apply. Notice that security rules do not apply statically but their activation may depend on contextual conditions. For this purpose, the concept of context is explicitly introduced [18]. Thus, using the first-order logic language, security rules are expressed as the following:

- Security rule $(org, role, activity, view, context)$ where security rule belongs to $\{permission, obligation, prohibition\}$: means that in organization org, subjects assigned to role have permission (respectively obligation and prohibition) to perform activity on view in context.

Once the organizational security policy has been defined, one must test how this policy applies to concrete entities that are the subjects, actions and objects. To assign concrete entities to abstract entities, OrBAC uses three predicates:

- Empower $(org, subject, role)$: means that in organization org, subject is empowered in role.
- Consider $(org, action, activity)$: means that in organization org, action is considered an implementation of activity.
- Use $(org, object, view)$: means that in organization org, object is used in view.

Besides the above conditions that must be satisfied to activate a given security rule, there are other extra conditions. These extra conditions may be related to various notions, such as temporal or spatial requirements. OrBAC calls context such extra conditions. OrBAC has also to define logical conditions to characterize when contexts are active:

- Hold $(org, subject, action, object, context)$: means that context *c* holds between subject *s*, action *a* and object *o* within org.

If all the conditions are satisfied, we can derive concrete privileges that apply to subject, action and object from organizational security rule:

- Is_permitted $(subject, action, object)$.

OrBAC model supports the O2O approach to manage interoperability between components having their own policies defined by different organizations [13]. The O2O approach relies on the concept of virtual private organization (VPO) to designate the sub-organization in charge of the interoperability access control. On the basis of confinement principle and contracts definition enforced by O2O, it is possible that some organizations will agree that some correspondences exist between their respective roles, activities, views and contexts. Moreover, the role single sign on (RSSO) principle allows a subject to keep his or her role when accessing to another organization. Thus, using O2O, each organization can define and enforce its own secure interoperability policy. In this paper, the O2O approach is useful to avoid exchanging the image abstract access and usage control rules between differences organizations. Another requirement is how to activate a given context. For example, the context $Attending\_physician$ may be defined in hospital H as follows:

- Hold $(H, s, a, o, Attending\_physician) \leftarrow$ Name_patient $(o, p) \wedge$ Patient (s, p).

In *H*, the context "Attending physician" holds between subject *s*, action *a* and object *o* if *o* is a record corresponding to the patient *p* (represented by the application dependent predicate Name_patient (o, p)) and *p* is a patient of subject *s* (represented by the application dependent predicate Patient (s, p)). To summarize, OrBAC distinguishes five types of context [18]:

- the **Temporal** context that depends on the time at which the subject is requesting for an access to the system,
- the **Spatial** context that depends on the subject location,
- the **User-declared** context that depends on the subject objective (or purpose),
- the **Prerequisite** context that depends on characteristics that join the subject, the action and the object,
- the **Provisional** context that depends on previous actions the subject has performed in the system.

OrBAC also assumes that each organization manages some information system (IS) that stores and manages different types of information. To control activation of the context, such an IS must provide information required to check that conditions associated with the context definition are satisfied or not. The following list gives the kind of information related to the contexts we aforementioned:

- a global clock to check the temporal context,
- the subject environment and the software and hardware architecture to check the spatial context,
- the subject purpose to check the user-declared context,
- the system database to check the prerequisite context,
- an history of the action carried out, to check the provisional context.

For instance, the context "Attending physician" belongs to the type of the Prerequisite context. To activate such a context, the organization requires storing the patient list (patients' names or their identifiers) of the physician in its system database.

B. Security Policy Enforcement

Once the policy defined, it has to be implemented or integrated to the information system. In practice, we assume the existence of a reference monitor responsible for the application of the policy. We also assume that when a subject wants to perform an action on an object, this subject must first send a request to the reference monitor. The monitor instructs and evaluates contexts and provides one of the following answers: Accept, Deny or Violation. Such a reference monitor can either be hardware or software, for instance, a specific viewer that is used to manage (display, print ...) medical images accordance with DICOM. Thanks to the OrBAC application programming interface (API) [19], software developers can integrate security mechanisms into their software. This API uses the Jena3 java library to specify an OrBAC policy as a RDF graph. It can be used to upload OrBAC RDF policies and interpret them, i.e. access control requests can be made on an uploaded policy and contexts are evaluated upon a query.

We may distinguish two types of reference monitor: 1) located on the server side SRM (Server-side Reference Monitor); 2) On the Client side CRM (Client-side Reference Monitor). In the case of SRM, protection is centralized on the server side and information should never be stored on the client side. For example, a patient's banking information is stored and protected on the hospital server and never on the physician's workstation. In the case of CRM, information (i.e. medical image) is sent to the client. Information is encrypted on the client side to ensure that it is not accessible when someone bypasses the reference monitor. In our framework, each organization defines and implements its own SRM and in each user workstation (i.e. the physician host) a CRM is integrated. Both SRM and CRM need the API OrBAC to apply the relevant OrBAC policy.

Let us consider the example of section III-A. To activate the context "Attending physician", the organization requires storing the patient list of the physician in its database system. If the information system does not provide such a list, then obviously the corresponding context cannot be evaluated and the reference monitor can not take a decision as the security policy rules can not be evaluated as well. However, this situation is very likely if the information is distributed elsewhere. There are two solutions to this problem: the first one is a format dependent solution. For example, package the necessary information with the object to evaluate the context or regularly synchronize between SRM and CRM. The format of information (i.e. a patient list) is always different from the object (i.e. a medical image) to be protected. The major disadvantage of these methods lies in the fact that the necessary information is extra information sent with the object. The second solution is to use the watermarking technology for inserting the information into the object itself. This solution has the advantage of being format independent and not increasing the size of the object to be transmitted. In this case, information for evaluating the context and the corresponding object are intimately linked, only authorized user with compliant reference monitor (integrating both OrBAC API and watermarking module) can extract the watermark using the insertion key. Once the watermark is extracted by the recipient, context data can be decoded and used to control the object access. In addition, the watermark may not only be updated by the administrator, but also before, during or after an access by the system. So the objective of our work is to consider watermarking for conveying access control elements and to express explicitly the watermark input data using the OrBAC model.

IV. Combining Watermarking and OrBAC

A. Basic Concepts and Design Goals

Before presenting our approach, here are some useful definitions of our framework as show in Figure 1:

Exchange: Sending to and receiving from objects (health information for instance) other organization.

Sharing: Provision of health information contained in an organization.

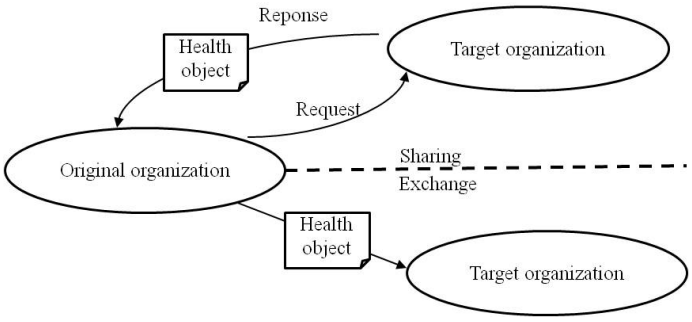

Fig. 1. Exchange and sharing between two organizations.

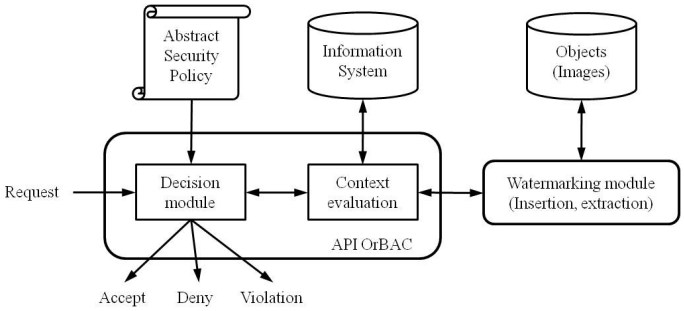

Fig. 2. Proposed architecture.

Target organization: In the context of sharing, it is a health information system providing shared services. In the context of an exchange point to point, it is a health information system receiving a message sent by another system.

Original organization: In the context of sharing, it is a system of health information services using a target organization. In the context of an exchange point to point, it is a health information system sending a message encapsulating some health object to a target organization.

In this paper, we assume that the interoperability between the different organizations at the abstract level is achieved by using O2O approach. That means the original organization's abstract security policy can be identified and used by the target organization, and the target organization's abstract security policy can also be identified and used by the original organization. To protect the medical image before, during and after the transmission, we need to ensure the following requirements in our proposed system:

- **Confidentiality** means that only the authorized users have access to the information;
- **Availability**, that is the capability of the information system to be used in normal scheduled conditions of access and exercise;
- **Integrity**, the information has not been modified by non-authorized people;
- **Authenticity**, a proof that the information belongs to the correct patient and issued from the right source.
- **Traceability** allows systematic information content validation, that is, content forensics, which in turn engenders trustworthiness and quality control. The quality control itself can be expanded into: actuality (information has a precise interest at a given instant) and reliability.

In the next section, we present the proposed solution. Then we discuss how the above security requirements can be fulfilled in the section V-C.

B. Proposed Model

In our point of view, the compliant monitor (SRM or CRM) must support the OrBAC API and a watermarking module (always integrated in a specific software). The core of our proposal is illustrated in Figure 2, the OrBAC API uploads and manages the abstract security policy, evaluates the context and finally provides the answer to the access request. The watermarking module will embed useful information into the corresponding image that can be used to evaluate certain contexts in a different organization. The value of the information must be converted to a watermark (in binary), then inserted into the image. Note that the value of the information has a limited lifetime, so the related watermark should be updated by the administrator or the system automatically. Three main services are provided by the watermarking module:

1) Convert information to be inserted into a watermark.
2) Insert or update the watermark.
3) Extract the watermark

In this paper, we consider the image itself as a "mini movable database" but we will not insert all the information in the image because of the limitation of the data hiding capacity and the risk of disruption of diagnosis. Thus, only information necessary to valuate some essential security rules have been selected by our system to be integrated into the watermarks, mainly information related to Prerequisite and Provisional contexts valuation. One of important information is the consent of the patient. Since the patient data is confidential, the need for electronic forms of patient consent, referred to as e-consent [20], has to be considered. Patients should be able to delegate, give or withhold e-consent to those who want to access their electronic health information. Normally, the e-consent is issued by the patient during consultation and is valid for a certain time. It will contain a set of conditions specifying when the data can be viewed. These conditions can be expressed in terms of named individuals, institutions, institutional roles and specified purposes. In this paper, we focus on the consent of the patient and specify this e-consent using the OrBAC model. Another important information should be inserted into the image is the tracing information. We consider this information as a part of the log files.

In order to use the OrBAC model and express the security rules at the organization level, the concept of view should be used. It is different from the concept of class. A class is a taxonomical concept used to structure the object descriptions, i.e. a class groups objects that have similar characteristics. By contrast, a view is an organizational concept used to structure the policy specification, i.e. a view groups objects on which the same security rules apply. Thus, at the abstract level, we

will create a view called Watermark and several sub-views such as:

- WatermarkInteg, which allows checking the integrity of an image.
- WatermarkIPP, which allows verifying the authenticity of an image.
- WatermarkConsent, which allows checking the right of the access and control of distribution.
- WatermarkTrace, which allows controlling of the traceability.

All these watermarks could be considered as the necessary information to make an a posteriori audit.

At the concrete level, we consider that a watermark is a derived object. The derived object is an object that is created as a result of obtaining or performing duties on an original object. For example, reading a medical image in the form of DICOM can create a watermark. This watermark is called a derived object in our model. It will be embedded in the image itself. Each watermark must have its own attributes. These attributes can be updated before, during and after an access. We derive four classes:

- Watermark integrity class: (1 attribute)
  - ImageDS: digital signature of the image (such as 256-bits SHA-256).
- Watermark IPP class: (1 attribute)
  - IPP: patient identifier of the image.
- Watermark consent of the patient class (5 attributes)
  - Authority: organization in which such consent may apply.
  - Grantee: list of roles to which such consent was given.
  - Privilege: list of activities permitted by the consent.
  - GranteeException: list of denied subjects.
  - Date: the consent deadline.
- Watermark trace class (2 attributes)
  - LogActor: list of users who have used this image. (In the case of doctors, can be represented by their physician identifier)
  - LogDate: application dates corresponding to physicians.

Beside the above classes of the watermark, we need also to keep somewhere the list of roles such as Doctor, Patient and the view Image. Subjects and objects belonging to these roles and views have their own attributes. So we derive three main classes, the values of the attributes should be stored in the database of the information system:

- infosDoctor class: (1 attribute)
  - ID_num: identifier of the physician.
- InfosPatient Class: (1 attribute)
  - IPP: identifier of the patient.
- InfosImage Class: (3 attributes)
  - Addressee_subject: identifier of a subject who is supposed to receive the image.
  - Addressee_role.
  - IPP: patient identifier of the image.

C. Access and Usage Control Rules

Initially, we consider both the original organization and the target organization have the compliant monitor, that means the system support the OrBAC API and the watermarking module. As illustrated in Figure 1, we consider two main sharing and exchange scenarios. In the following, we illustrate our approach through some security rules. We distinguish these rules into three categories: a priori, a real time and a posteriori. (To simplify the expressions, here we only consider the watermarking conditions.)

In the case of a priori, we control the access rights assigned to users before their access to the information. Three rules have been identified:

- **Rule Image access**: Permission 1. The physician is allowed to export the image on the target organization if the WatermarkTrace was updated by the watermarking module. That is to say, the identifier of the physician and the date of application in the list of LogActor and LogDate has been added. (Here we assume that all other conditions for activating this permission have been checked by the system in a conventional manner).

The objective of this rule is making the watermarking an additional condition to be achieved before authorizing the requested action. The rule is controlled by the monitor of the target organization. The embedded WatermarkTrace can be extract by the monitor of the original organization for audit. Because the update of the watermark will be done when there is a request, this requires specifying a dynamic context.

- **Rule Image distribution**: Permission 2. The physician is allowed to send or distribute the image to the target organization if: 1) the WatermarkTrace of the image was updated by the watermarking module; 2) attributes of target user (physician) should match the watermark attributes of the patient's consent.

This rule is always used in the case of exchange. It is controlled by the monitor of the original organization. As the patient's consent is inserted into the image, when there is an access request, the WatermarkConsent could be extracted and used to examine the target user's attributes. It should be noted that the consent of the patient have a deadline. If this deadline expired, the physician must contact the patient and update the WatermarkConsent.

- **Rule Image a priori tracing**: Permission 3. Before using the image, WatermarkTrace corresponding to this image must be updated. (If it is the same user, update the date.)

This rule can be used in both type of CRM and SRM. The objective is to record who used the image.

In the case of a real time, the user is allowed to use the object without any condition. But the reference monitor can at any time, extract the watermark, check the rights and decide to stop usage of the image if the authorization conditions are not satisfied. This mode of the access control makes the image

more convenient to use by the authorized user. In this paper, we identified one rule in this type:

- **Rule Image displaying**: Prohibition 1. During display of an image, the system can extract the WatermarkConsent of the image. If the user's attributes not match the watermark attributes of the patient's consent, the system will stop display.

Some rules are checked after each usage of the image. This mode of protection is more flexible. Its final objective is offering the necessary information for the next access control or the audit a posteriori. Two rules have been identified:

- **Rule Watermark creation**: Obligation 1. The image is must be watermarked after its creation and before its first use.

This rule corresponds to the insertion of the first watermark into the image. It is important because it makes the watermark available for integrity controlling and authenticity controlling. The WatermarkConsent is also embedded into the image after the image creation.

- **Rule Image a posteriori tracing**: Obligation 2. After each use of the image, WatermarkTrace corresponding to this image must be updated. (If it is the same user, update the date.)

This rule is opposite to the rule Image a priori tracing, which makes the image more easy to use but with a low protection of patient privacy.

D. Access and Usage Control Policy Specification

To specify the above rules, we use the OrBAC language. Because several rules require activating a dynamic context, the language L_{active} is used to formalize the Event Condition Action (ECA) rules [21]. The L_{active} vocabulary includes the following atoms: A - actions, F - fluent (i.e. data which can change their values), E - events and R - rule names. ECA rules, also called active rules, are used to initiate policy management operations when events are detected. An ECA rule states that when the event occurs and if conditions are true, then actions are executed. The language has the following three propositions:

1) action(X) causes f(Y) if $p_1(X_1)$,..., $p_i(X_i)$
2) event(Y) after action(X) if $p_1(X_1)$,..., $p_i(X_i)$
3) rule_name(X_r): event(X) initiates [α] if $p_1(X_1)$,..., $p_i(X_i)$

Where the symbols f, p_1... p_i are fluent symbols. The first proposition corresponds to the causality principle. In any state in which $p_1(X_1)$... $p_i(X_i)$ are true, the execution of the action action(X) determines f(Y) be true in the next state. The second proposition states that if the conditions $p_1(X_1)$... $p_i(X_i)$ are true in the state following the execution of the action a(X), then event(Y) is produced. The last proposition states that every new detection of the event(X) initiates the execution of the sequence of actions [α] if the rule conditions are true. Contrary to the state based context as mentioned in the section III-A, an event based context corresponds to the ECA event definition and therefore takes into account the dynamic aspect of a security policy. It is defined as the following:

- hold(org, subject, action, object, context) after action(X) if $p_1(X_1)$, ..., $p_i(X_i)$;

Two event based contexts, **start**(context) and **end**(context) are associated with each context of type state based. They are related to the activation and the deactivation of state based context.

By using ECA rules, we are able to express the dynamic contexts. For each rule, we firstly give the abstract expression, then we present how to manage the state and the event based context by active rules. At last, the activation of authorizations is illustrated.

Rule Image access:
At the abstract level, we specify this rule as the following:

- *Permission(Org, Physician, Download, Image, WatermarkTraceUpdated)*

As mentioned before, the context *WatermarkTraceUpdated* should be verified when there is a request. Thus, an event based context is defined as the following:

- *$Hold_e$(org, s, download_xx.dcm, xx.dcm,*
  ***start** (WatermarkTraceUpdated))*
  after Do *(local_watermarking_plugin,*
  update_watermarkTrace_xx.dcm,
  watermarkTrace_xx.dcm)

That is, in Org, role $Physician$ has the permission to perform activity $Download$ on view $Image$ in the context $WatermarkTraceUpdated$. To activate such a context, in org, Physician s can perform action $download\_xx.dcm$ on Image $xx.dcm$ ($.dcm$ is the extension of DICOM image) after the WatermarkTrace of the Image $xx.dcm$ has been updated by the watermarking module $local\_watermarking\_plugin$. After the activation of the context, we use the following active rules to activate the permission:

- *Activate_Permission: $Hold_e$(org, s, download_xx.dcm, xx.dcm,* ***start** (WatermarkTraceUpdated))*
  initiates *Insert_Permission (s, download_xx.dcm, xx.dcm)*
  if *Permission(Org, Physician, Download, Image, WatermarkTraceUpdated),*
  Empower(Org, s, Physician),
  Consider(Org, download_xx.dcm, Download),
  Use(Org, xx.dcm, Image)

The active rule *Activate_Permission* adds concrete permission to the state whenever the context of abstract permission is started.

Rule Image distribution:
At the abstract level, the rule is specified as the following:

- *Permission(Org, Physician, Sent, Image, WatermarkTraceUpdated $\wedge$ WatermarkConsentChecked)*

This rule contains two contexts. We support the composition of contexts using the logical operators conjunction ($\wedge$), disjunction ($\vee$) and negation ($\neg$). This allows the expression

of more sophisticated contextual conditions. Each context is specified as follows:

- *$Hold_e$(org, s, sent_xx.dcm, xx.dcm,*
  ***start** (WatermarkTraceUpdated))*
  after Do *(local_watermarking_plugin,*
  update_watermarkTrace_xx.dcm,
  watermarkTrace_xx.dcm)
- *Hold(org, s, sent_xx.dcm, xx.dcm, WatermarkConsentChecked) ←*
  Addressee_role(xx.dcm, add_r) ∧
  Grantee(watermarkConsent_xx.dcm, add_r) ∧
  Addressee_subject(xx.dcm, add_s) ∧
  ¬GranteeExeption(watermarkConsent_xx.dcm, add_s) ∧
  privilege(watermarkConsent_xx.dcm, Download) ∧
  (Date(Global_clock) ≤ Date(watermarkConsent_xx.dcm))

This rule use two types of contexts, the context "WatermarkConsentChecked" is a state based context. It can holds for a Physician s, action $sent\_xx.dcm$ and a health object $xx.dcm$ if the role add_r which is supposed to receive the image $xx.dcm$ is in the list of roles to which the consent was given by the patient, the subject add_s who is supposed to receive the image $xx.dcm$ is not in the list of denied subjects, the activity $Download$ is in the list of activities permitted by the consent and the current date does not exceed the deadline of the consent. Where $Global\_clock$ corresponds to a trusted clock of the information system and $Date$ its attribute. The second context **start**$(WatermarkTraceUpdated)$ is event based. Its activation is similar to the rule Image access. It should be noted that when the system needs the information supplied by the watermark in the image, the retrieval service (or insertion) which supports the module will be called implicitly for watermarking.

Rule Image a priori tracing:
At the abstract level, the rule is specified as the following:

- *Permission(Org, Physician, Consult, Image, WatermarkTraceUpdated)*

The correspond event based context and the activation of the permission are specified as follows:

- *$Hold_e$(org, s, consult_xx.dcm, xx.dcm,*
  ***start** (WatermarkTraceUpdated))*
  after Do *(local_watermarking_plugin,*
  update_watermarkTrace_xx.dcm,
  watermarkTrace_xx.dcm)
- *Activate_Permission: $Hold_e$(org, s, consult_xx.dcm, xx.dcm,* ***start** (WatermarkTraceUpdated))*
  initiates *Insert_Premission(s, consult_xx.dcm, xx.dcm)*
  if *Permission(Org, Physician, Consult, Image, WatermarkTraceUpdated),*
  Empower(Org, s, Physician),
  Consider(Org, consult_xx.dcm, Consult),
  Use(Org, xx.dcm, Image)

Rule Image displaying:
At the abstract level, the rule is specified as the following:

- *Prohibition(Org, Physician, Consult, Image, lecture_Application ∧ WatermarkConsentViolate)*

Where the prohibition contexts $lecture\_Application$ and $WatermarkConsentViolate$ are specified as follows:

- *Hold(org, s, consult_xx.dcm, xx.dcm, WatermarkConsentViolate)←*
  ¬Grantee(watermarkConsent_xx.dcm, Physician) ∨
  (ID_num(s, id_s) ∧ GranteeExeption (watermarkConsent_xx.dcm, id_s)) ∨
  ¬privilege(watermarkConsent_xx.dcm, Consult) ∨
  Date(Global_clock) > Date(watermarkConsent_xx.dcm)
- *Hold(org, s, consult_xx.dcm, xx.dcm, lecture_Application)←*
  Active_Application(consult_xx.dcm, xx.dcm)

To enable the monitoring of the above access control policy, one needs to specify the effects of the occurrence of the actions on the fluent $Active\_Application$. These effects are specified as follows.

Do(s, start_Application, consult_xx.dcm)
causes *Active_Application(consult_xx.dcm, xx.dcm)*
Do(s, end_Application, consult_xx.dcm)
causes *¬Active_Application(consult_xx.dcm, xx.dcm)*

Rule Watermark creation:
At the abstract level, the rule is specified as the following:

- *Obligation(Org, Administrator, Insert, Watermark, ImageCreated)*

The correspond event based context and the activation of the permission are defined as the following:

- *$Hold_e$(org, s, insert_watermark_xx.dcm,*
  watermark_xx.dcm, ***start** (ImageCreated))*
  after Do*(s', create_xx.dcm, xx.dcm)*
  if *Empower(Org, s', AcquisModality),*
  Consider(Org, create_xx.dcm, Create),
  Use(Org, xx.dcm, Image)

- *Activate_Obligation: $Hold_e$(org, s, insert_watermark_xx.dcm, watermark_xx.dcm,* ***start** (ImageCreated))*
  initiates *Insert_Obligation(s, insert_watermark_xx.dcm, watermark_xx.dcm)*
  if *Obligation(Org, Administrator, Insert, Watermark, ImageCreated),*
  Empower(Org, s, Administrator),
  Consider(Org, insert_watermark_xx.dcm, Insert),
  Use(Org, watermark_xx.dcm, Watermark)

Once medical images have been created by the correspond acquisition modalities, they will be transmitted to the sever for archiving. The administrator must embed the watermark into the image before the first consultation of the image from the sever.

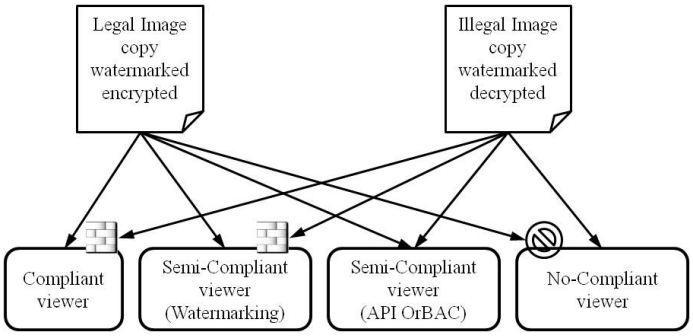

Fig. 3. Image copy protection with watermarking and Or-BAC.

Rule Image a posteriori tracing:
At the abstract level, the rule is specified as the following:

- *Obligation(Org, Watermark_Plugin, Update, WatermarkTrace, ImageConsulted)*

The correspond event based context is defined as the following:

- $Hold_e$*(org, local_watermarking_plugin, update_watermarkTrace_xx.dcm, watermarkTrace_xx.dcm,* ***start*** *(ImageConsulted))*
  after Do*(s, consult_xx.dcm, xx.dcm)*
  if *Empower(Org, s, Physician), Consider(Org, consult_xx.dcm, Consult), Use(Org, xx.dcm, Image)*

E. Policy Application

After specifying the above different rules, we can apply them in the two scenarios: In case of sharing: rules **Image access** and **Watermark creation** are applied. It corresponds to a phase of creation of watermarked medical images. On the server side, information systems are reliable and can provide ongoing information to evaluate the contexts of the security policy. But, the watermarks "WatermarkInteg" and "WatermarkIPP" can still provide opportunities to verify the integrity and authenticity of the image. For example, patient information is stored in the DICOM image header, in case of a change of image acquisition device and when the communication system and the display are not compatible, the header of the image can be lost and the image will not be identified. But using the watermarking technique, IPP will be inserted in the image at the pixel level; it can be retrieved on demand and used to check the authenticity of the image.

In the case of exchanging images, rules **Image distribution**, **Image a priori tracing** and **Image displaying** are applied. On the client side, the information needed to evaluate the contexts has been inserted in the image itself. We can use the rule **Image a posteriori tracing** rather than the rule **Image a priori tracing** for the purpose of using the image more easily. Because the WatermarkTrace is updated after the usage of the image. But the major problem is how to ensure the system is compatible? To answer this issue, we consider several assumptions:

- H1: the system is compliant that means the monitor of the system can support both watermarking and OrBAC API.
- H2: the system is semi-compliant only to watermarking.
- H3: the system is semi-compliant only to OrBAC API.
- H4: the system is not compliant.

As illustrated in Figure 3, before distributing an image, at the server side, the compliant system encrypts the image. Once the image arrives at the client side, a compliant system may decrypt the image and then uses it correctly. If the system is semi-compliant to watermarking, "WatermarkInteg" and "WatermarkIPP" are still usable. If the system is semi- compliant to the OrBAC API, the watermark will be useless. A negotiation between the two organizations and the communication of information will be necessary to continue to use the access control services that are provided by the OrBAC API. In the latter case, if the system is not compliant, the system can not decipher the image, it is not usable. If the image has been decoded, the copy can be distributed without watermark (and therefore illegally from our point of view) to other users. If the target system is compliant with watermarking, the system can always refuse to display it, otherwise the control is lost.

F. Watermark Encoding

In practice, the information associated with a single image will be converted to the binary string before its insertion. To encode information to the watermark (the binary string), we divided the watermark in several segments:

First segment: 256 bits. We use the SHA-256 hash function to compute the digital signature of an image. This signature can be extracted at any time and used to verify the integrity of the image.

Second segment: 256-bits. We use the SHA-256 hash function to compute the patient's profiles including their Social Security number, surname and birthday. This field is used to check the authenticity of the image.

Third segment: We form the watermark of patient's consent. We assume that in the security policy, among all entities, there are nbo (number of organization) organizations related to consents of the patient, nbr roles have the right to use the image through the consents, nba activities permitted by the consents. Then, we can give a number to each entity in the policy. For example, these entities can be extracted and put in a separate list.

Thus, at the encoding stage, *ceil(*log_2*(nbo))* bits for the number of organizations which were chosen to apply a consent (*ceil(x)* rounds the elements of *x* to the nearest large integers). For example, if nbo equals to 16 and the first four organizations in which the consent can be applied. So the binary string should be: 0100 0000 0001 0010 0011. Similarly, we can encode the roles and activities. But to encode the list of subjects that are denied by the consent, we will use their unique identifier. For example, in case of Physicians, we will insert their identifiers by adding the number of subjects as the header (i.e. we can use 4 bits to encode this number). To obtain this identifier, we use the SHA-256 hash function to compute

the physician's profiles including their health professional number. If there are four physicians in this case, we require 1028 bits (including the head) to encode this element. Finally, for the date, 5 bits to encode the day, 4 bits for the month and 12 bits for the year (until 2050). Thus, 21 bits is required.

Fourth segment: We will encode the list of users who have used this image. (In the case of Physicians, can be represented by their physician identifier) and then the list of dates corresponding to their last utilization. If we limit the place of the list to three, for each user, 256 bits to encode his identifier, 21 bits for the date. Thus, 831 bits is required for three users.

According to examples mentioned in each segment, the length of the watermark can up to about 3000 bits for one image. Thus, the employed reversible watermarking technique must support this capacity. In the next section, we test our proposed method (see section II) among different medical image issued from different modalities and give the performance of the method in terms of insertion capacity and image quality preservation.

V. Experiments

A. Image Database and Measures of Performance

The reversible watermarking method has been tested over several series of medical images (Figure 4). Five image modalities have been considered: three 12 bits encoded MRI (magnetic resonance imaging) volumes of 79, 80 and 99 axial slices of 256x256 pixels respectively, three 16 bits encoded PET (positron emission tomography) volumes of 234, 213 and 212 axial slices of 144x144 pixels respectively, three sequences of 8 bits encoded US (ultrasound image) images (14 of 480x592 pixels, 9 and 30 of 480x472 pixels respectively), forty two 12 bits encoded X-ray images of 2446x2010 pixels and thirty 8 bits encoded Retina images of 1008x1280 pixels.

To objectively quantify algorithms' performances, different indicators have been considered: the capacity rate C expressed in *bpp* (bit of message per pixel of image) and, in order to quantify the distortion between an image I and its watermarked version Iw, the peak signal to noise ratio (PSNR) is:

$$PSNR = 10log_{10}(\frac{NM(2^p-1)^2}{\sum_{i,j=1,1}^{N,M}(I(i,j)-I_w(i,j))^2}) \quad (1)$$

where *p* corresponds to the image depth, *N* and *M* correspond to the image dimensions. In the following experiments, the embedded message is binary and randomly generated according to a uniform distribution.

B. Figures and Tables

Results are given in figures 5 to 7. They provide the compromise between capacity and distortion depending on the image modality. If we consider our schemes in Figures 5 and 6, it allows a compromise from 0.0072 *bpp* / 85.7356 dB to 0.16 *bpp* / 63.87 dB for X-ray image, from 0.005 *bpp* / 68 dB to 0.59 *bpp* / 46.6 dB for Retina image and from 0.238 *bpp* / 60.72 dB to 0.37 *bpp* / 42.18 dB for US image. This means

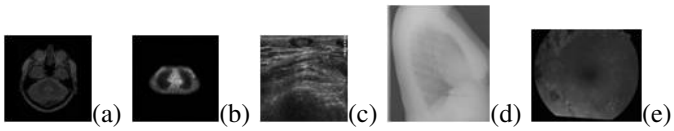
(a) (b) (c) (d) (e)

Fig. 4. Image samples from our test set (a) MRI of the head-axial slice of 256×256 pixels, 12 bits encoded. (b) PET-image of 144×144 pixels, 16 bits encoded, (c) US-image of 480×592 pixels encoded on 8 bits, (d) X-ray image of 2446×2010 pixels, 12 bits encoded, (e) Retina image of 1008×1280 pixels, 8 bits encoded.

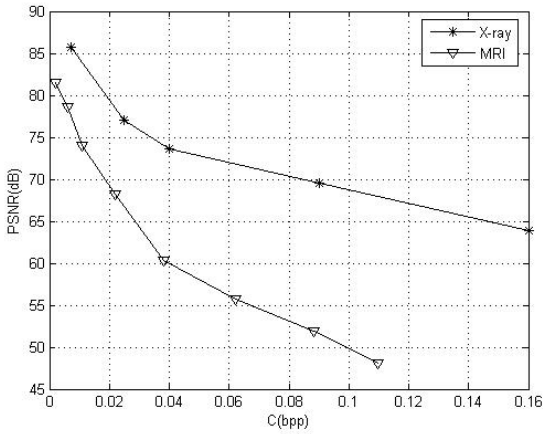

Fig. 5. PSNR/Capacity compromise for proposed method in the case of X-ray and MRI images.

that nearly 35000 bits can be embedded in a X-ray image with the quality of 85.7356 dB, nearly 6400 bits within a Retina image with the quality of 68 dB and nearly 67000 bits within an ultrasound image with the quality of 60.72 dB. Obviously, the greater capacity, the worse quality of the watermarked image. However, capacity performances are quite small for MRI and PET images as shown in Figures 5 and 7. Such a difference can be explained by the background which occupies an important place in MRI and PET images. Our method only watermarks non null signal, this is why it has a low capacity for these images. In [22], we have implemented and compared five reversible watermarking methods. From the experiments, it appears that the methods [23] are more suitable for PET and MRI images since they allow a high insertion capacity with the smallest distortion. As show in the table I, it allows a watermark capacity close to 0.2 *bpp* with PSNR about 73-75 dB for MRI, 97-99 dB for PET. This means that nearly 13000 bits can be embedded in MRI slice and 4000 bits within PET slice.

In fact, the method suggested by Ni $etal.$ in [23] refers to histogram shifting modulation. It shifts a range of the image histogram to create a "gap" near the histogram maxima. Pixels with gray values associated to the class of the histogram maxima are then shifted to the gap or kept unchanged to encode "0" or "1". Thus, for MRI and PET images, the histogram maxima refers to the zeros, that is why they have a good performance. However, the embedded data cannot be recovered unless the position of the initial histogram maxima

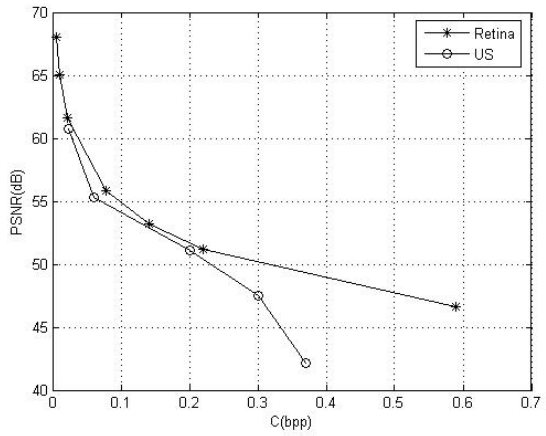

Fig. 6. PSNR/Capacity compromise for proposed method in the case of Retina and US images.

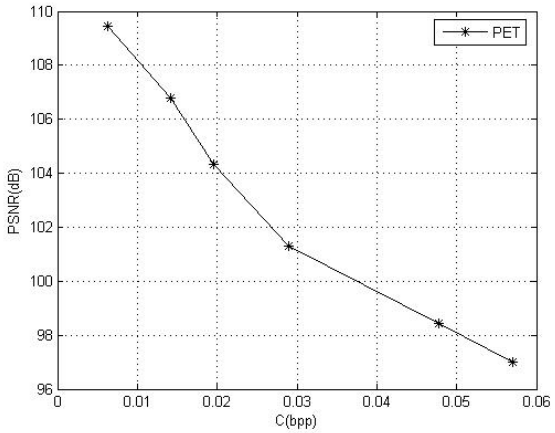

Fig. 7. PSNR/Capacity compromise for proposed method in the case of PET image.

and the "gap" is known by the decoder. In this work, in order to make our proposed system suitable for all types of medical images, our watermarking module use Ni's watermarking method and ours. For this purpose, in front of the watermark that will be embedded into the image by our method, we add a flag bit to indicate that the method of Ni has been used or not. If the bit is "0", that means only our method has been used; else, Ni's method has been used before ours, than the position of the initial histogram maxima and the "gap" is encoded and added after the flag bit for the decoding.

Thus, the watermarking module offers a high embedding capacity and low distortion for all types of medical images. This ensure our proposed application to insert all necessary information for integrity and usage control.

C. Discussion

Now we discuss how the security requirements stated in the section IV-A are fulfilled though our proposed model.

Achieving Confidentiality:

	MRI		PET	
	C(bpp)	PSNR(dB)	C(bpp)	PSNR(dB)
[23]	0.26(0.011)	73.00(0.46)	0.20(0.013)	97.98(0.92)

TABLE I
CAPACITY AND DISTORTION MEASUREMENTS OF MRI AND PET IMAGES FOR METHOD: NI *et al*. [23]. STANDARD DEVIATION IS GIVEN IN PARENTHESIS.

At the system level, the access control of the medical image is ensured by the API OrBAC. Furthermore, each medical image is obliged to be watermarking before the physician exports it outside the organization. This makes the watermarking to be a necessary condition for using the image. The watermark contains useful information once the image is outside of the organization.

At the content level, to secure image's transmission, the bit stream is encrypted using the symmetric key. Only the compliant system can decrypt the image. We can also use reversible watermarking instead of the cryptographic function. That means we use a strong watermark. In this case, the resulting watermarked image has very bad quality. Thus, the physician is obliged to remove the watermark before using the image. The advantage of such watermark technique is the watermark contains the useful information. For instance, some information can be used to check the integrity and usage rights of the image. But the disadvantage is the removing of the watermark is more easy than the decryption of the image by the attacker who knows the algorithm of the watermarking. Furthermore, as the user should remove the watermark before using the image, there exist a risk of losing the watermark, thus losing the protection.

Achieving Availability:

In the different organizations, the information system is most often different from each other. Therefore, the information that may be useful to control the usage of the image is not always available. By using our system, some useful information such as patient's consent and the tracing information have been inserted in the image. Thus, it makes these information available in different organizations and enable a more secure and correct usage of images.

Achieving Integrity and Authenticity:

According to the rule **Watermark creation**, the watermark "WatermarkInteg" which corresponds to the digital signature (DS) of the image has been inserted into to the image when the image is created. At the verifying stage, the watermark reader extracts the embedded DS and removes the watermark from the watermarked image. Then a DS is recomputed from the reconstruct image and compared to the extracted one. If they are equal then the integrity of the image is verified. To achieve the authenticity of the image, the watermark "WatermarkIPP" is used. Patient identifier is inserted in the image at the pixel level; it can be retrieved on demand and used to check the authenticity of the image.

Achieving Traceability:

The objective of this requirement is tracing medical images in distributed environment. The proposed system embed the watermark "WatermarkTrace" in the image. This watermark contain a list of users who have used the image and the date of their utilization. Thus, if the organization has a compliant monitor, this watermark can be extracted and used for security and privacy check.

We have shown how the watermarking can be used to enforce medical image access and usage control policy. We have constructed a basic secure system by combining watermarking and OrBAC model. Obviously, beside the selected information in this paper to be inserted in the image, there are many other information that can be inserted for the other purpose. Also, the system can be extend to cover other application such as medical report's access and usage.

VI. Conclusion

In this paper we are working to use reversible watermarking technique to enforce the medical image access and usage control policy. Differences security proprieties have been considered, such as the integrity and the traceability of information. Using the technique of reversible watermarking, one can insert the useful information in medical image to monitor the integrity, authenticity and right of access. To model the access control policy, the model OrBAC was used to explicitly specify the policy requirements. The OrBAC API is responsible for implementing the security policy. By combining the two techniques, our system provides continuous protection to information. The information inserted in the image will provide means to verify, in the subsequent audit, that each rule of the security policy is correctly enforced. Automatic alert may be raised when violation is detected. Future work consists of integrating administration elements into the model.

References

[1] G. Coatrieux, L. Lecornu, B. Sankur, and Ch. Roux.: A Review of Image Watermarking Applications in Healthcare. in Proc. of the IEEE EMBC Conf., New York, USA, 2006, pp. 4691-4694.

[2] M. L. Miller, I. J. Cox.: A review of watermarking principles and practices. Digital Signal Processing in Multimedia Systems, Ed. K. K. Parhi and T. Nishitani, Marcell Dekker Inc., 461-485, 1999.

[3] J. A. Bloom, I. J. Cox, T. Kalker, J-P Linnartz, M. L. Miller, and B. Traw.: Copy protection for DVD video. Proc. of IEEE, 87(7):1267-1276, 1999.

[4] Frank Hartung and Friedhelm Ramme, Ericsson Research.: Digital Rights Management and Watermarking of Multimedia Content for M-Commerce Applications. IEEE Communications Magazine, 2000, Vol 38, Issue: 11, pp. 78 - 84.

[5] Willem Jonker and Jean-Paul Linnartz.: Digital rights management in consumer electronics products. IEEE SIGNAL PROCESSING MAGAZINE, MARCH 2004,p 82-91.

[6] Woo, C.-S., J. Du, and B. Pham.: Multiple watermark method for privacy control and tamper detection in medical images. Proceeding of the APRS Workshop on Digital Image Computing, 2005, Australia, pp.59-64.

[7] Bao, F., R. H. Deng, B.C. Ooi, and Y. Yang.: Tailored reversible watermarking schemes for authentication of electronic clinical atlas. IEEE Transactions on Information Technology in Biomedicine, 2005, vol. 9(4), pp.554-563.

[8] Coatrieux, G., M. Lamard, W. Daccache, J. Puentes, and C. Roux.: A low distortion and reversible watermark: Application to angiographic images of the retina. Proceeding of the International Conference of the IEEE-EMBS, 2005, pp.2224-2227.

[9] B. Lampson.: Protection. in 5th Princeton Symposium on Information Sciences and Systems, pages 437-443, March 1971.

[10] D. Bell and L. LaPadula. Secure Computer Systems: Unified Exposition and Multics Interpretation. Technical Report ESD-TR-75-306, MTR-2997, MITRE, Bedford, Mass, 1975.

[11] R. S. Sandhu, Coyne E. J., Feinstein H. L., Youman C. E.: Role-Based Access Control Models. IEEE Computer, vol. 29, no 2, 1996, p. 38-47.

[12] A. E. Kalam, Baida R. E., Balbiani P., Benferhat S., Cuppens F., Deswarte Y., Miège A., Saurel C., Trouessin G.: Organization Based Access Control (Or-BAC). IEEE 4th International Workshop on Policies for Distributed Systems and Networks (Policy 2003), Lake Come, Italy, June 2003.

[13] F. Cuppens, N. Cuppens-Boulahia, and C. Coma. O2O: Virtual Private Organizations to Manage Security Policy Interoperability. In 2nd ICISS, December 2006.

[14] Coatrieux, G., B. Sankur, and H. Maître.: Strict Integrity Control of Biomedical Images. in Proceeding of Electronic Imaging, Security and Watermarking of Multimedia Contents, SPIE, 2001, USA, pp.229-240.

[15] Zhou, X. Q., H. K. Huang, and S. L. Lou: Authenticity and integrity of digital mammography images. IEEE Transactions on Medical Imaging, 2001, vol. 20(8) pp.784-791.

[16] Mintzer, F., J. Lotspiech, and N. Morimoto, Safeguarding digital library contents and users: Digital watermarking, D-Lib Mag., 1997.

[17] W. Pan, G. Coatrieux, N. Cuppens, F. Cuppens and Ch. Roux.: An Additive and Lossless Watermarking Method Based on Invariant Image Approximation and Haar Wavelet Transform. accepted in Proc. of the IEEE EMBC Conf., Buenos Aires, Argentina, 2010.

[18] F. Cuppens and N. Cuppens-Boulahia.: Modeling contextual security policies. Int. J. Inf. Secur., 7(4):285-305, 2008.

[19] F. Autrel, F. Cuppens, N. Cuppens-Boulahia et C. Coma.: MotOrBAC 2: a security policy tool. Third Joint Conference on Security in Networks Architectures and Security of Information Systems (SARSSI). Loctudy, France, 13-17 October 2008.

[20] Coiera, E.: e-Consent: The Design and Implementation of Consumer Consent Mechanisms in an Electronic Environment. Journal of the American Medical Informatics Association, 11 (2004), pp. 129-140.

[21] Y. Elrakaiby, F. Cuppens, and N. Cuppens-Boulahia.: From state-event to event-based management of security policies. 3rd International Workshop on Context Modeling and Management for Smart Environments, 2009.

[22] W. Pan, G. Coatrieux, N. Cuppens, F. Cuppens and Ch. Roux.: Comparison of some reversible watermarking methods in application to medical images. in Proc. of the IEEE EMBC Conf., Mineapolis, United States, Sept. 2009, pp 2172-2175.

[23] Z. Ni, Y. Shi, N. Ansari, and S.Wei.: Reversible data hiding. in Proc. IEEE Int. Symp. Circuits and Systems, May 2003, vol. 2, pp. 912-915.

Bibliographie

[1] ASIP Santé (Agence des systèmes d'information partagés de santé, http ://www.asipsante.fr.

[2] D. Bell and L. LaPadula, *Secure Computer System : Unified Exposition and Multics Interpretation*, MITRE, Bedford, Mass, 1975.

[3] J. A. Bloom, I. J. Cox, T. Kalker, J-P Linnartz, M. L. Miller, and B. Traw. Copy protection for DVD video. Proc. of IEEE, 87(7) :67-1276, 1999.

[4] F. Cuppens, N. Cuppens-Boulahia, and C. Coma. O2O : Virtual Private Organizations to Manage Security Policy Interoperability. In 2nd ICISS, December 2006.

[5] F. Cuppens and N. Cuppens-Boulahia, *Modeling contextual security policies.*, In International Journal of Information Security, 7(4) : 285-305, 2008.

[6] F. Cuppens, A. Miège, *Administration model for Or-BAC*, Int. J. Comput. Syst. Sci. Eng. (CSSE), 2004.

[7] Digital imaging and communications in medicine (dicom) - part 1 : Introduction and overview, Rapp. Tech. PS 3.1, 2009.

[8] Department of Defense Trusted Computer System Evaluation Criteria, CSC-STD-011-83, Fort Meade, MD August 1983.

[9] DICOM Standards Committee, Working Group 14, *Digital Imaging and Communications in Medicine (DICOM) Supplement 95 : Audit Trail Messages*, Virginia USA, June 2004.

[10] L. Dusserre, Recommandations déontologiques pour le choix de logiciels destinés aux cabinets médicaux, Ordre national des médecins Conseil National de l'Ordre, Ethique et Déontologie, 1997.

[11] L. Dusserre, H. Ducrot, et F. A. Allaërt, L'Information Médicale - l'Ordinateur et la Loi, 1999.

[12] Méthode EBIOS, Introduction V2, http ://www.ssi.gouv.fr/IMG/pdf/ebiosv2-section1-introduction-2004-02-05.pdf , février 2004.

[13] Y. Elrakaiby, F. Cuppens, and N. Cuppens-Boulahia. : From state-event to event-based management of security policies. 3rd International Workshop on Context Modeling and Management for Smart Environments, 2009.

[14] D. Ferraiolo and R. Kuhn, *Role-Based Access Controls*, 15th NIST-NCSC National Computer Security Conf., Baltimore, MD, 1992.

[15] Integrating the Healthcare Enterprise, *IHE IT Infrastructure Technical Framework Supplement 2004-2005 Audit Trail and Node Authentication Profile (ATNA)*, August 2004.

[16] Integrating the Healthcare Enterprise, *IHE IT Infrastructure Technical Framework Volume 1 (ITI TF-1) Integration Profiles*, August 2009.

[17] A. van Lamsweerde , *Goal-Oriented Requirements Engineering : A Guided Tour*, Invited Minitutorial, Proc. RE01 - 5th Intl. Symp. Requirements Engineering, pp. 249-263, Toronto, August 2001.

[18] Evangelina Kvakli and Pericles Loucopoulos, University of the Aegean, *Goal Driven Requirements Engeneering : Evaluation of Current Methods*, EMMSAD'03. Evaluation of Modeling Methods in Systems Analysis and Design, Austria, June 2003.

[19] G. Marshall, *Security Audit and Access Accountability Message XML, RFC 3881*, September 2004.

[20] F. Cuppens, N. Cuppens-Boulahia et C. Coma, *MotOrBAC : un outil dadministration et de simulation de politiques de sécurité.* First Joint Conference Security in Network Architectures (SAR) and Security of Information Systems (SSI), Seignosse Landes, France, 6-9 Juin 2006.

[21] W. Puech, Rodrigues J.M., *A new crypto-watermarking method for medical images safe transfer*, EUSIPCO. Conference No.12, Vienna, September 2004.

[22] E. Siegel et R. Kolodner, *Filmless Radiology*, collection health informatics, edition, 1998.