Exigences de confidentialité et de diffusion concernant les politiques d'échanges d'information

R. Delmas, T. Polacsek

Exigences de confidentialité et de diffusion concernant les politiques d'échanges d'information

■

Rémi Delmas et Thomas Polacsek

Résumé

Que cela soit pour la surveillance de la Terre, la gestion des risques naturels ou dans les différentes formes que revêtent les relations inter-entreprises, il existe de plus de plus d'organisations interconnectées formant des systèmes d'information complexes et décentralisés dans lesquels l'échange d'information joue un rôle crucial. Afin de maîtriser ces échanges il est nécessaire de spécifier clairement les exigences qui leurs sont liées, et peuvent être regroupées en ce que nous nommons une politique d'échange d'information. Dans cet article nous présentons un cadre formel qui permet, d'une part, de spécifier une politique d'échange et, d'autre part, de vérifier automatiquement, à l'aide d'un outil basé sur du *model-checking*, un ensemble de propriétés sur cette spécification.

Mots clés

Exigences pour les SI, politiques d'échanges d'information, méthodes formelles, spécification, vérification ■

1. Introduction

Que cela soit pour la surveillance de la Terre, de l'espace ou la gestion des risques sanitaires, il existe de plus de plus de systèmes dans lesquels des organisations s'échangent des informations dans le but de prévenir et de gérer des catastrophes. Prenons pour exemple les applications de type *Space Situational Awareness* ; ici, des capacités d'observation de l'espace, appartenant à différentes nations, sont mutualisées afin de prévenir les risques de collision entre satellites ou entre satellites et débris spatiaux orbitant autour de la terre. La mission de tels systèmes consiste à envoyer des alertes aux agents, tels que les nations ou les opérateurs de satellites, afin de leur permettre d'éviter une collision, tout en garantissant que les informations sensibles sur les objets en question, comme par exemple leur nature ou leur trajectoire, ne soient pas divulguées.

Les systèmes de surveillance globale de la Terre (*Global Earth Observation and Surveillance Systems*[1]) en sont un autre exemple. Ici, des informations issues de l'observation de la Terre sont échangées par différents agents tels que des états, des agences, des organisations et des compagnies, dans le but de gérer des catastrophes naturelles. De tels systèmes doivent absolument garantir que toute information concernant une catastrophe naturelle sera toujours envoyée aux autorités compétentes afin que des mesures de protection de la population puissent être prises en temps opportun, tout en ne révélant pas d'informations sensibles telles que, par exemple, la nature ou les performances des moyens d'observation dont disposent les agents du système.

Loin de se borner à la seule gestion de risques, de tels systèmes interconnectés se retrouvent au sein des entreprises. En effet, dans le cadre de ce que l'on nomme *l'entreprise étendue*, plusieurs entreprises réalisent un

1 Systèmes d'observations de surveillance globale de la Terre

couplage fort, dans le but de produire ensemble un produit, entraînant des échanges de données sensibles telles que des spécifications, des modèles de conception, etc. Ces échanges s'effectuent au travers d'interconnexions entre les différents systèmes d'information, au sein de plateformes fédératives, ou même, parfois, de façon plus triviale par de simples échanges de fichiers de gré à gré entre individus. De plus, chaque organisation est elle-même incluse dans divers sous-systèmes en fonction de ses partenariats, de son statut de filiale, où les échanges d'information sont régis par d'autres règles, éventuellement contradictoires.

Ce qu'il y a de commun à l'ensemble de ces systèmes, de la gestion de risques à l'entreprise étendue, c'est l'exigence que si une information cruciale vient à être connue par l'un des agents du système alors les agents concernés par cette information *doivent absolument en être avertis* tout en garantissant la *non-diffusion* d'information sensible à des agents non concernés.

Dès lors, pour pouvoir gérer ces échanges, il est primordial de spécifier précisément qui a l'obligation, la permission ou l'interdiction d'envoyer quelle information, à qui et sous quelles conditions. Cette spécification, réalisée en amont de la conception du système, fournit ce que nous appelons une *politique d'échanges d'information*. À cela, il nous paraît crucial de pouvoir adosser des opérations de vérification visant à éliminer au plus tôt les inconsistances, incomplétudes et autres problèmes que pourrait contenir la spécification d'une politique d'échanges. Le but de ces opérations étant de détecter les erreurs au plus tôt, car en effet, des erreurs non détectées avant les phases d'implémentation ne font qu'accroître les coûts de développement et les risques de diffusion non maîtrisée d'informations sensibles.

C'est dans cette optique que nous avons défini un cadre de conception, nommé PEPS[2] [7], qui fournit, d'une part, un langage formel pour la spécification d'une politique et, d'autre part, un outil d'analyse de politiques nommé PEPS-analyzer. Cet outil offre une assistance à la mise au point d'une politique, en effet, en interagissant avec PEPS-analyzer, il est possible de vérifier si une formalisation est conforme ou non aux intuitions et de la modifier en conséquence en se servant de contre-exemples renvoyés par l'outil. En fait, PEPS-analyzer permet de vérifier automatiquement des propriétés génériques sur les politiques telle que, entre autres, la *consistance*, qui est satisfaite s'il n'existe pas de cas où il est à la fois interdit et obligatoire (ou permis) pour un agent d'envoyer une information à un autre agent, selon les règles de la politique. D'autres types de propriétés servent à garantir que certains agents seront toujours avertis de certaines informations, ou encore à garantir la non-diffusion de certaines informations.

Dans le présent article, en section 2 nous discutons la notion de norme et expliquons à haut niveau les critères fondamentaux utilisés pour choisir le cadre logique sur lequel est bâti PEPS. En section 3, nous détaillons les composantes principales de PEPS à l'aide d'un exemple simple. Enfin, nous concluons en brossant les perspectives d'améliorations de PEPS.

2 PEPS est l'acronyme récursivement défini par : PEPS for exchange policy specification

2. Modéliser les exigences pour la diffusion d'informations

Exprimer clairement les exigences liées aux échanges d'information dans un système renvoie de façon plus générique à la notion de norme. Une norme est l'ensemble des règles qui définissent les comportements d'agents dans un système. Suivant la définition donnée par *Information Technology Security Evaluation Criteria*, *Une politique de sécurité d'un système spécifie l'ensemble des lois, règles et pratiques qui régissent comment les informations sensibles ainsi que les autres ressources sont utilisées, protégées et échangées dans un système donné*[3] [11]. Cette définition étant très générale, des travaux ont cherché à définir plus précisément les politiques de sécurité au travers de cadres formels dédiés aux politiques de contrôle d'accès (citons pour exemple [4,12]), dans lesquels sont clairement explicitées les conditions sous lesquelles un agent a accès à une ressource en fonction du rôle qu'il tient dans une organisation. Tous ces travaux ont un point commun, celui de disposer de prédicats normatifs (spécifiant les conditions d'une obligation, permission ou interdiction) qui portent sur trois éléments : le sujet, l'action et l'objet. En fait, les politiques d'accès permettent de représenter les droits sur des verbes d'action, dits verbes *bivalents*, avec un sujet et un unique complément. Ce choix est motivé par le fait que les politiques d'accès visent à modéliser sous quelles conditions un agent doit ou ne doit pas réaliser des actions telles que lire ou écrire des données. Dans le cadre des politiques d'échanges, nous nous intéressons à des actions un peu différentes puisque nous avons des verbes qui représentent l'interaction entre deux agents au sujet de données. En fait, les actions d'échange dans notre contexte correspondent à des verbes dits *trivalents*[4] tels que *échanger*, *dire*, *donner* ou *envoyer*. Par conséquent, il n'est pas possible de représenter une politique d'échanges un utilisant les langages et outils existants dédiés au contrôle d'accès.

Dans le but de modéliser les politiques d'échanges, il est naturel de considérer l'utilisation de la logique déontique. La logique déontique [3] est une logique modale conçue pour formaliser les normes, les lois et les règlements. Pour cela, la logique déontique définit la notion d'obligation à l'aide d'un opérateur modal. Le point clé de cette approche est de considérer qu'une proposition doit être vraie du point de vue de la norme, car elle est obligatoire, sans toutefois être nécessairement vraie dans le monde réel. Prenons pour exemple le cas des élections en France. Même s'il est interdit de communiquer les résultats avant que tous

3 Traduit de l'anglais : A System Security Policy specifies the set of laws, rules and practices that regulate how sensitive information and other resources are managed, protected and distributed within a specific system.

4 Verbe rattaché à un sujet, un complément d'objet direct et un complément d'objet indirect.

les bureaux de vote en France métropolitaine ne soient fermés, il arrive que certains médias le fassent quand même. Cependant, si la logique modale, et *a fortiori* la logique déontique, nous fournit un cadre formel permettant de raisonner sans ambiguïté sur les aspects normatifs d'un système, elle n'est pas exempte de limitations. Ainsi McCarthy [13] explique que si la logique modale est un outil qui permet de manipuler aisément des concepts elle est particulièrement difficile à manipuler pour un non-expert, et reste, par conséquent, un outil limité à l'usage des logiciens. Poussant plus loin cet argument, Edmonds [9] déclare que les approches à base de logiques formelles sont inutilisables par la plupart des gens étant donné la complexité même de ces langages logiques. On peut toutefois objecter à cet argument que seul le langage naturel peut être compris par tout le monde, ce qui nous ramène à l'ensemble des questions qui se rapportent aux ambiguïtés du langage naturel, questions qui ont justement motivé l'utilisation de langages formels pour la spécification de systèmes.

Loin de cette polémique, nous préférons insister sur le fait que l'un des problèmes des logiques modales dans notre cas précis est le manque d'outils génériques performants permettant l'évaluation automatique de la satisfiabilité d'une formule. Notre but étant ici de fournir des analyses automatiques pour aider à l'expression des exigences d'une politique d'échanges, notre approche nécessite l'utilisation d'outils efficaces et, pour l'heure, force est de constater que les solveurs dédiés aux logiques modales sont moins efficaces que les solveurs pour les logiques standard (logique propositionnelle, logique du premier ordre) [14].

3. Un cadre pour l'expression des politiques d'échanges : PEPS

Dans [6,7] nous avons défini un cadre formel nommé PEPS pour la spécification et la vérification des politiques de diffusion de l'information. Techniquement PEPS est bâti sur la logique du premier ordre multi-sortée avec égalité (MSFOL) [10]. PEPS est nativement conçu pour exprimer les obligations, les interdictions, les permissions et les opérations d'échanges d'information, et il fournit un ensemble de prédicats prédéfinis pour cela. Une particularité de PEPS est son extensibilité : pour préciser les caractéristiques métiers du système étudié, l'utilisateur peut déclarer ses propres sortes, fonctions et prédicats. De cette manière, il est par exemple possible de décrire la topologie d'un réseau d'échanges, ou bien des opérations plus complexes de traitement d'une information précédant sa transmission entre agents, telles que la déclassification. Enfin, les règles de diffusion de la politique spécifient sous quelles conditions sur les agents du système agents et/ou sur tout objet du système, un agent a l'obligation, l'interdiction ou la permission d'envoyer une information à un autre agent.

Nous avons implémenté notre propre outil de vérification dédié au langage PEPS : PEPS-analyzer. Cet outil utilise le prouveur Sat4j [2] ou le solveur Z3 [5] et fournit une aide à la spécification de politiques d'échanges. Cette aide consiste en la vérification automatique d'un ensemble de propriétés prédéfinies, que nous allons présenter au travers d'un exemple, ou de propriétés spécifiques données par l'utilisateur, ainsi que la possibilité de renvoyer à l'utilisateur des contre-exemples quand une propriété n'est pas satisfaite par la politique.

Nous allons illustrer cela au travers d'un exemple simple. Considérons un système de surveillance de la Terre dont le but est de prévenir et de gérer les catastrophes naturelles. Nous nous focaliserons ici sur un seul type de catastrophe : les événements sismiques. Pour la gestion des risques sismiques un groupe spécifique composé de différentes organisations, nommé RS pour « Risques Sismiques », s'occupe d'organiser la gestion du risque (évaluation du risque, diffusion de l'alerte aux populations, coordination des évacuations, etc.). De plus, pour éviter les fausses alertes et les mouvements de panique seuls les membres du RS sont autorisés à communiquer les informations portant sur un risque sismique. Une première version de la politique d'échanges pour ce système pourrait se composer de seulement deux règles :

- (r1) « tout agent qui possède une information relative à un risque sismique doit la communiquer à un membre du RS »
- (r2) « seul les agents membres du RS ont la permission de transmettre une information relative à un risque sismique à un agent non membre du RS »

Comme les moyens d'observations de la Terre viennent de plusieurs sources dont certaines peuvent avoir un caractère sensible, nous ajoutons à cela la règle

- (r3) : « il est interdit pour quiconque d'échanger des informations sensibles »

La transcription de ces trois règles dans le langage PEPS est donnée dans le Listing 1 ci-dessous.

```
rule r1:
forall a: agent, b: agent, i: info,
  know(a,i) & hastopic(i,seismic) &
RS(b) => Osend(a,b,i) ;

rule r2:
forall a: agent, b: agent, i: info,
  know(a,i) & hastopic(i,seismic) &
RS(a) => Psend(a,b,i) ;

rule r3:
forall a: agent, b: agent, i: info,
  know(a,i) & hastopic(i,sensitive)
=> Fsend(a,b,i) ;
```

Listing 1 : Un exemple simple de politique d'échanges

La première vérification que non pouvons faire à l'aide de PEPS-analyzer est la vérification de la complétude de la politique. Une politique est complète si et seulement si dans toutes situations, pour toutes informations, la politique indique si un agent qui connaît une information a l'obligation, la permission ou l'interdiction de l'envoyer aux autres agents. Cette définition est assez standard et correspond à celle donnée par [1,8] dans le contexte des politiques de contrôle d'accès. Cependant, dans les phases de conception, il n'est pas toujours nécessaire qu'une politique soit complète, il peut être utile à certains moments du processus de conception de disposer d'une politique complète seulement pour certains sous-domaines de son domaine d'application. En effet, la spécification d'une politique peut être décomposée en un processus en plusieurs phases où, pour chaque phase, on se concentre sur un sous-ensemble des sujets possibles couverts par la politique. De plus, une politique peut être conçue dans le cadre d'une collaboration entre des parties bien distinctes, chacune portant attention seulement au sous-ensemble des sujets d'information qui la concernent. Par exemple, dans le cadre de l'observation de la Terre, les opérateurs militaires peuvent vouloir s'assurer que la politique est complète pour tout ce qui concerne le sujet militaire sans s'intéresser aux autres sujets.

Sur notre exemple de politique, PEPS-analyzer renvoie un contre-exemple montrant que notre politique n'est pas complète, et pour cause : premièrement, la politique ne dit rien concernant les informations relatives à d'autres sujets que le sujet sismique, sur lequel nous allons nous concentrer pour la propriété de complétude. Deuxièmement, la politique ne spécifie pas les échanges entre membres du RS. Par conséquent, nous rajoutons deux règles, une disant qu'il est permis aux membres du RS d'échanger entre eux des informations relatives au sujet sismique et qu'il est interdit d'échanger des informations relatives au sujet sismique hors du groupe RS (respectivement r4 et r5 sur le listing 2). On obtient ainsi la complétude relativement aux sujets sismique et sensible. Par contre, elle n'est pas complète dans l'absolu puisque d'autres sujets concernant d'autres types de risques peuvent exister tel que tsunami, inondation, etc.

```
rule r4:
forall a: agent, b: agent, i: info,
  know(a,i) & hastopic(i,seismic) &
RS(a) & RS(b) => PSend(a,b,i);

rule r5:
forall a: agent, b: agent, i: info,
  know(a,i) & hastopic(i, seismic) &
~RS(a) & ~RS(b) => FSend(a, b, i);
```

Listing 2

Maintenant que nous avons une politique complète, nous pouvons chercher à évaluer sa consistance. Une politique est inconsistante si elle oblige (ou permet) et interdit simultanément la même chose, en d'autres termes si la politique contient une contradiction au sens déontique du terme. Notons qu'une politique inconsistante est de fait une politique inapplicable.

Dans notre exemple la politique est inconsistante car si une information porte à la fois sur une alerte sismique et sur un sujet sensible alors il est à la fois obligatoire de la transmettre à un membre du RS et interdit de la transmettre à quiconque (en particulier à un membre du RS). Pour rendre la politique consistante tout en la gardant complète pour le sujet des alertes sismiques, il faut rajouter à notre politique un ensemble de règles spécifiant le traitement particulier des informations relatives à la fois au sujet *risque sismique* et au sujet *sensible*.

Ainsi, l'interaction avec PEPS-analyzer permet à terme de définir une politique consistante, complète pour le sujet *alerte sismique*, mais également minimale (aucune règle ne peut se déduire des autres), et applicable (il existe toujours un cas où la règle peut s'appliquer).

Pour finir, PEPS-analyzer nous permet aussi de vérifier formellement deux types de propriétés cruciales dans les systèmes considérés : Premièrement, garantir que certains agents seront toujours avertis relativement à un sujet donné, et deuxièmement de garantir la non-diffusion (ou la diffusion restreinte à certains agents de certaines informations. Dans notre exemple cela correspond à prouver que toute information relative au sujet *risque sismique* sera toujours transmise à un membre du RS, et à prouver la non-diffusion d'informations *sensibles* au sein du système.

Au final, PEPS nous permet d'exprimer et d'analyser des exigences relatives à la diffusion d'information dans un système multi-agents. De nombreux travaux restent à venir, que cela soit sur la prise en compte de la structures des organisations qui constituent le système, la prise en compte des notions relatives à la déclassification d'une information, la génération automatique de règles afin de garantir des propriétés, telle que la complétude, ou encore les liens avec l'implémentation du système d'information.

Références

[1] Selim G. Akl et Dorothy E. Denning : *Checking classification constraints for consistency and completeness* ; in IEEE Symposium on Security and Privacy, pages 196-201. IEEE Computer Society, 1987.

[2] Daniel Le Berre et Anne Parrain : *The sat4j library, release 2.2* ; JSAT, 7(2-3), pages 59-6, 2010.

[3] Hector Neri Castañeda. : *Thinking and doing: The philosophical foundations of institutions* ; A Pallas Paperback. Springer, 1975.

[4] Frédéric Cuppens et Alexandre Miège : *Administration model for or-bac* ; in OTM 2003 Workshops, volume 2889 of Lecture Notes in Computer Science, pages 754-768. Springer Berlin Heidelberg, 2003.

[5] Leonardo Mendonça de Moura et Nikolaj Bjørner : *Z3 : an efficient SMT solver* ; in TACAS 2008. Proceedings, volume 4963 of Lecture Notes in Computer Science, pages 337-340. Springer, 2008.

[6] Rémi Delmas et Thomas Polacsek : *Checking need-to-share and nondiffusion properties on exchange policies with sat-solvers* ; à paraître.

[7] Rémi Delmas et Thomas Polacsek : *Formal methods for exchange policy specification* ; in CAiSE, volume 7908 of Lecture Notes in Computer Science, pages 288-303. Springer, 2013.

[8] Dorothy E. Denning, Selim G. Akl, Mark Heckman, Teresa F. Lunt, Matthew Morgenstern, Peter G. Neumann et Roger R. Schell : *Views for multilevel database security* ; IEEE Trans. Software Eng., 13(2), pages 129-140, 1987.

[9] Bruce Edmonds : *How formal logic can fail to be useful for modelling or designing mas* ; in RASTA 2002, Revised Selected and Invited Papers, volume 2934 of Lecture Notes in Computer Science, pages 1-15. Springer, 2002.

[10] Jean H. Gallier : *Logic for computer science : Foundations of automatic theorem proving* ; chapitre 10, pages 448-476, Wiley, 1987.

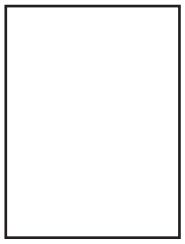
[11] ITSEC : Information technology security evaluation criteria (itsec): Preliminary harmonised criteria ; Technical Report Document COM(90) 314, Version 1.2., Commission of the European Communities, 1991.

[12] Anas Abou El Kalam, Salem Benferhat, Alexandre Miège, Rania El Baida, Frédéric Cuppens, Claire Saurel, Philippe Balbiani, Yves Deswarte et Gilles Trouessin : *Organization-based access control* ; in POLICY 2003, page 120. IEEE Computer Society, 2003.

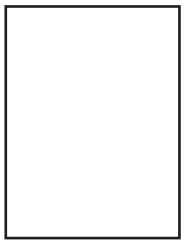
[13] John McCarthy : *Modality, si ! modal logic, no !* ; Studia Logica, 59(1), pages 29-32, 1997.

[14] Roberto Sebastiani et Michele Vescovi : *Automated reasoning in modal and description logics via SAT encoding : the case study of k(m)/ALC-Satisfiability* ; J. Artif. Intell. Res. (JAIR), 35, pages 343-389, 2009.

Biographies



Rémi Delmas : Actuellement ingénieur de recherche à l'ONERA de Toulouse, Rémi Delmas a reçu un diplôme d'ingénieur de l'ENSMA en 2000 ; il a ensuite obtenu un doctorat en informatique et méthodes formelles de Sup'Aéro en 2004, portant sur la vérification compositionnelle de systèmes avioniques modulaires intégrés. Entre 2004 et 2009, il a occupé le poste d'ingénieur R&D au sein de la société Prover Technology, au sein de laquelle il a participé à la conception et au développement de solutions de model-checking basées SAT pour des systèmes de contrôle ferroviaires critiques, certifiées pour une utilisation au niveau SIL-4, pour des clients tels que la RATP, Thales, Ansaldo, Westinghouse, etc. À l'ONERA, depuis 2009, ses recherches portent principalement sur l'utilisation des techniques SAT et SMT pour la conception et la vérification de plateformes embarquées critiques, la vérification formelle de logiciels critiques et la spécification et vérification formelles de politiques d'échange d'information.



Thomas Polacsek : Après avoir effectué un doctorat en Intelligence Artificielle, il a travaillé auprès d'acteurs majeurs de l'industrie sur des problématiques de modélisation métier et d'algorithmiques dans les secteurs du transport urbain et de la banque. Depuis 2009, il est chercheur à l'ONERA (Office National d'Études et de Recherches Aérospatiales) au sein du Département Traitement de l'Information et Modélisation (DTIM). Aujourd'hui, il mène des travaux de recherche, notamment dans le cadre de projets européens avec des acteurs du secteur aéronautique, mais aussi avec les autorités de certification sur les liens entre modèle et code source pour la vérification logicielle, l'aide à la vérification et la validation de système dans le contexte de l'aéronautique, ainsi que l'utilisation de méthodes formelles pour la spécification de politique d'échange.