Need-to-Share & Non-Diffusion Requirements Verification in Exchange Policies

Rémi Delmas & Thomas Polacsek*

Abstract

Whether be it for Earth observation, risk management or even companies relations, more and more interconnected organizations form decentralized systems in which the exchange, in terms of diffusion or non-diffusion of information between agents, can have critical consequences. In this paper, we present a formal framework to specify information exchange policies for such kinds of systems and two specific requirements, the need-to-share and the non-diffusion requirements, as well as properties strongly related to them. Wiser from these formal definitions, we see how to reconcile these sometimes two antagonist requirements in a same policy specification with information filtering operations. We also explain how we use state of the art theorem provers to perform automatic analysis of these policies.

1. Introduction

Today, individuals, companies, organizations and national agencies are increasingly interconnected, forming complex and decentralized information systems. In some of these systems, the very fact of exchanging information can constitute a safety critical concern. Take for instance Space Situation Awareness applications (SSA), in which space observation capabilities belonging to different nations are mutualized in order to build a complex information gathering, analysis and alert diffusion system. The mission of the system is to warn when situations of potential collision between orbiting objects are detected. The system must, in case of potential collision, send relevant alerts and associated information to the right agents so as to allow them to avoid the collision, while guaranteeing that sensitive information about the orbiting objects, such as their exact nature, their trajectories, manœuvre capabilities, *etc.* will not be leaked. Another example is Global Earth Observation and Surveillance Systems (GEOSS). Observation information is exchanged by cooperating agencies or states, and it must be ensured that information about natural disasters will always reach the relevant authorities so that population protection measures can be taken in due time, while not revealing sensitive information about the earth observation means of the members taking part in the surveillance effort.

In systems like SSA or GEOSS, qualified authorities *must absolutely be warned* as soon as evidence showing imminent natural disaster is acquired. We call such requirements: *need-to-share* requirement. In fact, the true requirement is the authorities need to know the information and from that we derive the need-to-share requirement. In a paradoxical way, because agents from different organizations share information, any risk of leakage of private or sensitive information about the cooperating parties must be prevented. So, the challenge is to reconcile these two antagonist requirements: firstly, ensuring that actors will always receive the information they need to perform their designated mission; secondly, ensuring that no sensitive information will be released in an uncontrolled manner.

*Authors version, Need-to-Share & Non-Diffusion Requirements Verification in Exchange Policies, 151-165, Advanced Information Systems Engineering - 27th International Conference, CAiSE 2015, Stockholm, Sweden, June 8-12, 2015, Proceedings. Lecture Notes in Computer Science 9097, Springer 2015, ISBN 978-3-319-19068-6

To do so, need-to-share and *non-diffusion* requirements should be expressed in a specification language which allows to formally specify the conditions under which agents have either the obligation, the permission or the interdiction to communicate information to other agents in the system. We call such specifications *information exchange policies*. Besides the formalization of requirements, our goal is also to provide means of automatic formal verification of a number of generic properties of policies. To obtain the high degree of automation needed by system designers while retaining a high performance of analysis, we provide PEPS-analyzer, an automatic semantic analyzer for policies, which works by translating property verification problems for policies to *satisfiability modulo theory* problems, which are resolved using a state-of-the-art SMT solver, Microsoft Z3 in our case.

In this paper, we give in section 2 a brief overview of an existing framework to specify information exchange policies and explain how we use satisfiability checkers to perform automatic analyses on them. After that, in section 3 we detail the running example of this paper. In sections 4 and 5, we give formal definitions of two classes of generic properties, related to the need-to-share requirement on the one hand, and to the non-diffusion of information requirement on the other hand. Then, in section 6, we proceed to identify cases in which these properties become logically incompatible and detail how the need of information filtering operations arises naturally. Last, section 7 concludes the paper and outlines perspectives to this work.

2. Exchange Policy Specification

In [5], we provided a formal framework named PEPS[1], for the specification and verification of information diffusion policies. In this section, we show how we extend PEPS to take into account both diffusion and non-diffusion requirements explicitly.

The benefit of using unified frameworks has previously been studied in the context of information access. In [11] for instance, the authors propose a modelling language, in fact a meta-model, which allows to express security and privacy requirements; In [1] and [12] the authors propose similar approaches for security requirements.

2.1. The PEPS Formal Language

The formal system underlying PEPS is many sorted first-order logic with equality[7] (MSFOL). So, PEPS allows the use of sorts ($\mathcal{A}$, $\mathcal{B}$, ...), free constants (A, B, ...), functions and predicates (first letter in uppercase), polymorphic equality ($=$), usual logical connectors ($\neg$, $\wedge$, $\vee$, $\Longrightarrow$) as well as sorted variables $x : \mathcal{S}$ (first letter in lowercase), together with universal ($\forall$) and existential ($\exists$) quantifiers. Full details about the PEPS syntax and semantics can be found in [5].

The PEPS language is extensible, *ie* users can declare their own sorts, functions and predicates. However, PEPS comes equipped with a minimalist set of core concepts, in the form of predefined sorts, functions and predicates: Sorts $\mathcal{A}$, $\mathcal{I}$, $\mathcal{T}$ represent respectively agents, information items and information topics. In addition, we have the following *domain-predicates*, or *D-predicates* for short: $K(A, I)$ is used to express that an agent A knows an information item I; the predicate $Topic(I, T)$ is used to express that information item I is relevant of topic T (a single information item can be relevant of many different topics).

Unlike standard deontic logic [3], we do not have a generic obligation operator, because we focus on the concept of *obligation to send information item i from agent a to agent b*. So, dedicated *normative-predicates*, called *N-predicates*, are provided: $O_{Send}(A, B, I)$, $P_{Send}(A, B, I)$ and

[1] PEPS is a recursive acronym for *Peps for Exchange Policy Specification*

$F_{Send}(A, B, I)$, which respectively encode the obligation, permission and interdiction for an agent A to send an information I to another agent B.

Note that, to express obligation and related concepts, we could have used deontic logic. However tools dedicated to modal logic are less efficient than standard logic solver tools [13] such as SAT solvers or SMT-solvers. By not representing obligation with a modal operator we lose expressiveness, but we gain the use of efficient logic solvers to perform fully automatic analyses.

In standard deontic logic, obligation and permission operators are linked by axiom (D) which expresses that if a proposition P is obligatory then P is also permitted. In PEPS, we translate this axiom to a first-order property we also call (D): if communication of an information item is mandatory between two agents, then it is also permitted.

Definition 1 (D)

$$D \equiv \forall a, \forall b, \forall i, O_{Send}(a, b, i) \implies P_{Send}(a, b, i)$$

In PEPS, an *exchange rule* expresses conditions under which agents have the obligation, permission or interdiction to send a piece of information to another agent. An *exchange policy* (*EP*) is a collection of exchange rule formulas.

Definition 2 (Exchange rule) *An exchange rule is a* closed PEPS *formula of one of the following syntactical forms:*

$$\forall x_1, \ldots, \forall x_n,\ (\phi \implies O_{Send}(t_1, t_2, t_3))$$
$$\forall x_1, \ldots, \forall x_n,\ (\phi \implies P_{Send}(t_1, t_2, t_3))$$
$$\forall x_1, \ldots, \forall x_n,\ (\phi \implies F_{Send}(t_1, t_2, t_3))$$

where:

- $x_1, \ldots, x_n$ *are all variables identifiers occurring in* ϕ, t_1, t_2 *and* t_3*;*
- ϕ *is a quantifier-free and* N-predicate*-free formula;*
- t_1, t_2 *are quantifier-free terms of sort* $\mathcal{A}$*;*
- t_3 *is a quantifier-free term of sort* $\mathcal{I}$*.*

Also part of a PEPS specification is a formal description of Σ, the domain in which the policy is meant to apply. The declaration of additional sorts and domain predicates needed to build a domain model suitable for a particular application is left to the user. These new predicates and sorts can be used in the left member ϕ of the implication forming a rule, but not in the right member (PEPS is extensible only with new sorts, functions and D-predicates, and not with new N-predicates).

The combination of an exchange policy *EP* and set of domain constraints Σ is called an *exchange policy specification* and is noted $EPS = \langle \Sigma, EP \rangle$.

In the following sections, we will often have to assert that a policy specification *EPS* is in effect under the domain constraints and the *D* property. We hence introduce the following notation for what we call the *policy formula*.

Definition 3 (Policy formula)

$$\mathcal{EPS} \equiv \Sigma \wedge (\bigwedge_{r \in EP} r) \wedge D$$

Last, we will use the notation $P \models Q$ to state that Q is a logical consequence of P, *ie* that any model of P is also a model of Q.

2.2. Formal Policy Verification

In this section we first provide details on satisfiability checking algorithms used in the PEPS-analyzer tool, and then provide details on the generic properties that can be checked using the tool.

The peps-analyzer Tool

We provide a tool which can be used both to find bugs in policies and to check that properties hold on policies. A semantic verification tool such as PEPS-analyzer is a valuable help. Even with as few as a dozen of rules, complex interactions between rules make it hard to identify and understand incoherences, incompleteness or redundancy using solely a mental model of the policy, or using test cases, or to be absolutely sure the policy indeed works as intended. With PEPS-analyzer we address verification problems which can be expressed as (one or more) satisfiability checks. In order to verify that $P \models Q$, meaning Q *is a logical consequence of* P, where P and Q are both MSFOL formulas, the unsatisfiability of $P \wedge \neg Q$ is checked using an MSFOL satisfiability solver.

Earlier versions of PEPS-analyzer were based on a pure SAT encoding of MSFOL satisfiability problems, where sorts were interpreted over finite domains, and by using a bounded model checking approach: domain cardinalities were increased iteratively up to user-specified bounds, and quantifiers grounded on these finite domains. This approach was fully automatic, but the validity of the analyses was only up to a finite and relatively small number of information items, agents, topics, etc.

The latest version of PEPS-analyzer still works by reducing property verification to satisfiability, however MSFOL formulas generated by PEPS-analyzer (always involving quantifiers) are now directly given to an MSFOL-capable satisfiabilit solver which natively supports quantifiers, and handles quantifier instantiation internally using advanced algorithms. Quantifier handling in SMT solvers has come a long way since the early days, and Microsoft Z3 [10], the back-end solver used by PEPS-analyzer, is able to handle the quantified formulas arising from policy verification without user interaction or manual tuning. The huge advantage is that proofs obtained this way hold for sort interpretation domains of infinite cardinality. Models returned by Z3 for satisfiable formulas are presented to the user when they represent counter example to policy properties.

Generic Policy Properties

PEPS-analyzer allows to either prove or disprove four generic properties: *consistency*, *applicability minimality*, and *completeness*.

The *consistency* property holds if and only if there is no situation allowed by the domain model such that an agent is both obliged (or permitted) and prohibited to send an information to another agent. The *applicability* property holds if for each rule, there exists at least one situation allowed by the domain model in which the rule applies. The *minimality* property holds if no rule can be deduced from a combination of the other rules, under the domain constraints.

For the *completeness* property, the following definition is used: the completeness property holds if and only if, in any situation allowed by the domain model, for any information topic, any agent who knows an information item is either obliged, permitted or prohibited to send it to any other agent. Completeness checking aims at detecting situations in which the policy does not tell the agent what to do with a piece of information.

This definition of completeness is fairly standard and similar to the definition given by [2] [6] in the context of access control policies. It can also be found in numerous works with few variations, as in the case of access control in a multi-level security context [4] or in the problem of merging two policies [8].

However, this strict and global definition of completeness does not allow to deal efficiently with the following practical situations: first, the design of a policy is most of the time decomposed in phases, and in each phase the designer(s) might want to focus on a subset of the possible information topics covered by the policy. Second, policies might be designed collaboratively by distinct parties, each one paying attention only to a certain subset of all possible topics. In the context of Earth observation for instance, military operators may want to ensure that the policy is complete for any military-relevant topic, without much care for other topics.

In situations like these, the completeness check will fail as long as the policy is in an intermediary state and missing rules. It could be interesting to define a restricted form of completeness, which would be checkable as the design of the policy progresses, without waiting for the policy to be in its final state.

So, we propose to adapt the completeness notion by making it relative to a given information topic T. We call this restricted form of completeness T-completeness.

Definition 4 (T-Completeness of a policy specification) *Let $EPS = \langle \Sigma, EP \rangle$ be an exchange policy specification, and T a constant of sort $\mathcal{T}$. We say that EPS is* complete *relative to T, or T-*complete *if and only if the following holds:*

$$\mathcal{EPS} \models \forall a, \forall b, \forall i, (K(a,i) \wedge Topic(i,T) \implies (P_{Send}(a,b,i) \vee O_{Send}(a,b,i) \vee F_{Send}(a,b,i)))$$

So, a policy is T-complete if and only if for any agent who knows a piece of information relevant of a topic T, the policy specifies whether the agent is obliged, permitted or prohibited to send it to any other agent.

3. Example

We now introduce a simple running example which will help us illustrate the rest of the paper. In this example, agents represent anything from individuals to organizations in possession of earth observation means. We distinguish a specific group of agents: the *Geohazard Management Group*, noted *GMG*, whose mission is to prevent false geohazard warnings and to org anise disaster management plans. The policy for this system is very simple and consists of four rules:

r1 *"Any agent not part of the GMG has the obligation to communicate any geohazard-related information to at least one member of the GMG."*

r1b *"Any agent not part of the GMG has the permission to communicate any geohazard-related information to any agent part of the GMG."*

r2 *"Any agent which is not part of the GMG is forbidden to communicate geohazard-related information to any agent not part of the GMG."*

r3 *"Agents of the GMG have the permission to communicate geohazard-related information to any agent."*

The rule (*r1*) shows that we are indeed dealing with the need-to-share requirement, the necessity for other agents to communicate geohazard-relevant information to a member of the *GMG* is essential for the *GMG* to accomplish its mission.

The rule (*r1b*) handles the cases ignored by (*r1*), any agent external to *GMG* knows what to do with respect to any agent of the *GMG* besides the one for which communication is mandatory. The rule (*r2*) prevents the risk of mass-panic movements which could result from a brutal

dissemination of geohazard information to the general public. The rule (*r3*) illustrates the benefits of the permission modality: details of the criteria used by the *GMG*, which could involve human appreciation, to eventually issue a public alert or not are abstracted away by the optional nature of the permission. *The permission operator hence allows to model policies at a high abstraction level*, which is desirable for an early use of this formalism in the design process.

In order to model these rules in PEPS, we first declare a new constant *Geo* of sort $\mathcal{T}$ representing the *geohazard* information topic and a new domain predicate *GMG* ranging over the agent sort $\mathcal{A}$.

In fact because PEPS is extensible language, we can add predicates whenever we need it. In this example, and for the rest of this paper, we choose to model groups in a simple and abstract way. For that, we introduce a predicate over the agent sort for each group. These predicates can be viewed as membership predicates, characterizing groups of agents. Each predicate acts as a characteristic function for the group it represents, *i.e.* the predicate evaluates to $\top$ for agents which are part of the group, and to $\bot$ for agent which are not part of the group. Note that, of course other ways to model groups are possible in PEPS, for instance by introducing a sort $\mathcal{G}$ to represent the groups, and by using a predicate $mb(g : \mathcal{G}, a : \mathcal{A})$ for membership testing, as found in OrBAC [9] models.

In our example, $GMG(a)$ is true whenever a is part of the *GMG* group and false when it is not.

The four rules of the exchange policy of our example are then expressed in PEPS as follows:

$$\begin{array}{rll}
r1: & \forall a, \forall i, \exists b, K(a,i) \wedge Topic(i, Geo) \wedge \neg GMG(a) \wedge \ GMG(b) & \Longrightarrow O_{Send}(a,b,i) \\
r1b: & \forall a, \forall i, \forall b, K(a,i) \wedge Topic(i, Geo) \wedge \neg GMG(a) \wedge \ GMG(b) & \Longrightarrow P_{Send}(a,b,i) \\
r2: & \forall a, \forall i, \forall b, K(a,i) \wedge Topic(i, Geo) \wedge \neg GMG(a) \wedge \neg GMG(b) & \Longrightarrow F_{Send}(a,b,i) \\
r3: & \forall a, \forall i, \forall b, K(a,i) \wedge Topic(i, Geo) \wedge \ GMG(a) & \Longrightarrow P_{Send}(a,b,i)
\end{array}$$

In addition, we assert that there is no information without topic in the system, by adding the domain constraint *d*: *"Any information is relevant of at least one topic."* to the policy, which is written in PEPS as follows: $d: \ \forall i, \exists t, Topic(i,t)$.

Using the tool PEPS-analyzer, we can check that the policy specification $\langle \{d\}, \{r1, r1b, r2, r3\} \rangle$ is *Geo*-complete, consistent, applicable and minimal.

4. Need-to-Share Requirement: the Awareness Property

In systems like GEOSS or SSA, it is frequent that some designated group of agents has missions requiring it being aware of any piece of information relevant of some topic T. In our example, the group *GMG* needs to know any piece of information related to the topic *Geo*. It can hence be desirable to check that the rules of a policy guarantee that the said group never misses such T-related information. We call this notion T-awareness, and define it formally as follows:

Definition 5 (T-Awareness of a group G) *Let $EPS = \langle \Sigma, EP \rangle$ be an exchange policy specification, T be a constant of sort $\mathcal{T}$ and G be a predicate ranging over the sort $\mathcal{A}$, characterizing a group of agents*[2]*. We say that G is T-aware according to EPS if and only if:*

$$\mathcal{EPS} \quad \models \quad (\forall a, \forall i, \exists b, K(a,i) \ \wedge \ Topic(i,T) \ \wedge \ \neg G(a) \ \wedge \ G(b) \qquad \Longrightarrow \qquad O_{Send}(a,b,i))$$

So, the group G is T-aware according to *EPS* if and only if any agent outside of G knowing a T-relevant information item has the obligation to send it to at least one agent belonging to the group G.

If the group contains only one agent A, then $G(a)$ is equivalent to the test $(a = A)$, and after simplification we get the following definition of agent-awareness.

[2]Groups are modelled with a domain predicate G ranging over the agent sort $\mathcal{A}$. $G(a)$ is true whenever the agent a is part of the group and false otherwise (see Section 3).

Definition 6 (T-Awareness of an agent A) *Let $EPS = \langle \Sigma, EP \rangle$ be an exchange policy specification, T be a constant of sort $\mathcal{T}$ and A be a constant of sort $\mathcal{A}$. We say that A is T-aware according to EPS if and only if:*

$$\mathcal{EPS} \models \forall b, \forall i, K(b,i) \wedge Topic(i,T) \wedge \neg(b = A) \implies O_{Send}(b,A,i)$$

So, the agent A is T-aware according to *EPS* if and only if any other agent knowing a T-relevant information item has the obligation to send it to A.

On a side note, if an agent is T-aware for all possible topics it is called *omniscient*, but this case somehow lies at the border of our scope of study. Indeed, an omniscient agent is an agent which has a total knowledge of the system, meaning information is centralized by one agent, which does not correspond to systems we are studying here.

Using PEPS-analyzer, we check that the example policy $\langle \{d\}, \{r1, r1b, r2, r3\} \rangle$ satisfies the *Geo*-awareness property for group *GMG* (indeed, rule (*r1*) is a direct instantiation of the property).

5. Non-Diffusion Requirement: the Restriction Properties

If the T-awareness property allows to verify that T-related information is sent to the right group of agents in a system, it can be interesting to verify a dual property, namely that information about some topic (presumably a sensitive one) cannot reach a group of agents, a single agent or can simply not be disseminated at all. Given a group of agents, one can be interested in regulating the diffusion of information in the following cases: purely outside of the group, from outside to inside or from inside to outside the group. The case of diffusion within the group is not relevant here since we are interested in characterizing the diffusion of information with respect to the boundary defined by the group.

Definition 7 (T-Restriction to a group G) *Let T be a constant of sort $\mathcal{T}$, G be a predicate ranging over the sort $\mathcal{A}$, characterizing a group of agents, and let EPS be a policy specification.*

Topic T is said to be:

(a) T-out-out-restricted according to EPS if and only if:

$$\mathcal{EPS} \models (\forall a, \forall b, \forall i, K(a,i) \wedge Topic(i,T) \wedge \neg G(a) \wedge \neg G(b) \implies F_{Send}(a,b,i))$$

(b) T-out-in-restricted according to EPS if and only if:

$$\mathcal{EPS} \models (\forall a, \forall b, \forall i, K(a,i) \wedge Topic(i,T) \wedge \neg G(a) \wedge G(b) \implies F_{Send}(a,b,i))$$

(c) T-int-out-restricted according to EPS if and only if:

$$\mathcal{EPS} \models (\forall a, \forall b, \forall i, K(a,i) \wedge Topic(i,T) \wedge G(a) \wedge \neg G(b) \implies F_{Send}(a,b,i))$$

A system satisfying (a) (b) and (c) is completely sealed, which means diffusion of T-relevant information is only allowed within the group.

One can also be interested by the strict non-diffusion of T-relevant information in a system, expressed as: communication of T-relevant information is forbidden between any pair of agents.

Definition 8 (Strict T-Restriction) *Let T be a constant of sort $\mathcal{T}$ and let EPS be a policy specification. The topic T is said to be strictly restricted according to EPS if and only if:*

$$\mathcal{EPS} \models (\forall a, \forall b, \forall i, K(a,i) \wedge Topic(i,T) \implies F_{Send}(a,b,i))$$

Note that this definition is just a particular case of Definition 7 with an empty G group, modelled as $\forall a, G(a) \equiv \bot$.

We check automatically with PEPS-analyzer that the policy $\langle\{d\}, \{r1, r1b, r2, r3\}\rangle$ satisfies the *Geo*-out-out-restriction for the group *GMG*. Indeed, *Geo*-relevant information cannot be sent between agents outside of *GMG*, however agents of *GMG* can receive information from external agents, and also have the permission to communicate with other agents outside of the group.

6. Need-to-Share Versus Non-Diffusion requirements

6.1. Incompatibility Between Awareness and Restriction Properties

Let us further assume that the example system will also have to deal with sensitive information, and that we would like to ensure strict non-diffusion of this new kind of information. We add the following rule (*r4*) to the exchange policy:

r4 "It is forbidden to exchange any piece of sensitive information."

To model this new rule in PEPS, we simply introduce a new constant *Sens* of sort $\mathcal{T}$ to model the new topic, and add the following rule:

$$r4: \quad \forall a, \forall b, \forall i, K(a,i) \wedge Topic(i, Sens) \implies F_{Send}(a,b,i)$$

The consequences of adding (*r4*) are rather important, since the example policy $\langle\{d\}, \{r1, r1b, r2, r3, r4\}\rangle$ is now inconsistent.

The phenomenon is the following: if an agent knows a piece of information which is relevant of both *Geo* and *Sens* topics (imagine for instance a single satellite picture taken by a military satellite, showing a risk of natural disaster next to both a city and a secret research facility), then rules (*r1*) and (*r4*) apply, entailing the interdiction for the agent to send this piece of info to any agent of the group *GMG* (according to (*r4*)), as well as the obligation to send it to at least one agent of *GMG* (according to (*r1*)). These two requirements are obviously contradictory, and violate the consistency property defined for PEPS policies (see section 2.2). Note that rules (*r3*) and (*r4*) also entail the inconsistency of the policy by permitting and forbidding the communication of an information item relevant of both *Geo* and *Sens* topics.

In fact, the problem is not specifically tied to this example, it is more general. If we consider the T_1-awareness property for a group G_1 and one of the T_2-restriction properties for a group G_2 in a strictly logical way, depending on the domain constraints and on how G_1 and G_2 behave under them, it can be possible to build models satisfying both properties and where it is both mandatory and forbidden to send an information from an agent to another agent[3]. These models all have the same structure: at least one piece of information relevant of both topics T_1 and T_2 exists, and the group predicates G_1 and G_2 are such that some agents exist inside and/or outside G_1 and G_2 while satisfying the premises of the properties.

6.2. An Ad-Hoc Solution

In order to fix the problem of possible conflict between the necessity of diffusion and the obligation of non-diffusion, we propose to introduce an abstract operator in the framework, noted P, and to give it properties allowing to obtain both *Geo*-awareness for *GMG* and strict non-diffusion for the topic *Sens*. Information being a multidimensional entity, a piece of information can be relevant

[3]The PEPS theory contains no axioms to prevent such situations, they are just identified and labelled as inconsistent by the consistency checking algorithm of the PEPS-analyzer [5].

or more than one topic, we might be able to resolve the conflict by having this new operator selectively *forget* or *erase* the problematic topic from a multi-topic information item.

In the case of our example policy, we introduce the new operator as a function taking an information item and returning an information item, with the signature $P(i : \mathcal{I}) : \mathcal{I}$. We want this operator to forget about the sensitive part of an otherwise geohazard-related piece of information, so we have two domain constraints: (*p1*) an information produced by P is never relevant of the *Sens* topic and (*p2*) an information remains *Geo*-relevant when P is applied on it.

$$\begin{aligned} p1: &\quad \forall i, \neg Topic(P(i), Sens) \\ p2: &\quad \forall i, Topic(i, Geo) \implies Topic(P(i), Geo) \end{aligned}$$

We might also want to adapt the original policy by specifying the cases in which the operator needs to be used and the ones where it does not. Firstly, we split rule (*r1*) in two new rules, (*r11*) and (*r12*), to express that if an information item is related to *Geo* and not to *Sens*, agents have the obligation to send it to a member of the *GMG*, but if the item is also *Sens*-relevant, the agents need to apply the abstract operation P before sending it.

$$\begin{aligned} r11: &\quad \forall a, \forall i, \exists b,\ K(a,i) \wedge Topic(i, Geo) \wedge Topic(i, Sens) \\ &\qquad\qquad \wedge \neg GMG(a) \wedge GMG(b) \implies O_{Send}(a, b, P(i)) \\ r12: &\quad \forall a, \forall i, \exists b,\ K(a,i) \wedge Topic(i, Geo) \wedge \neg Topic(i, Sens) \\ &\qquad\qquad \wedge \neg GMG(a) \wedge GMG(b) \implies O_{Send}(a, b, i) \end{aligned}$$

Secondly, in the same way as above, we decompose the rule (*r1b*) in two new rules (*r1b1*) and (*r1b2*) to take the new *Sens* topic into account.

$$\begin{aligned} r1b1: &\quad \forall a, \forall i, \forall b,\ K(a,i) \wedge Topic(i, Geo) \wedge Topic(i, Sens) \\ &\qquad\qquad \implies \wedge \neg GMG(a) \wedge GMG(b) P_{Send}(a, b, P(i)) \\ r1b2: &\quad \forall a, \forall i, \forall b,\ K(a,i) \wedge Topic(i, Geo) \wedge \neg Topic(i, Sens) \\ &\qquad\qquad \wedge \neg GMG(a) \wedge GMG(b) \implies P_{Send}(a, b, i) \end{aligned}$$

Thirdly, we need to modify the rule (*r3*) in (*r3′*) which expresses that any member of the *GMG* is allowed to communicate a piece of information related to geohazards to any other agent if the piece of information is not sensitive.

$$\begin{aligned} r3': &\quad \forall a, \forall b, \forall i, K(a,i) \wedge Topic(i, Geo) \wedge \neg Topic(i, Sens) \wedge GMG(a) \\ &\qquad\qquad \implies P_{Send}(a, b, i) \end{aligned}$$

With this modification, using PEPS-analyzer, we check that the new policy $\langle \{d, p1, p2\}, \{r11, r12, r1b1, r1b2, r2, r3', r4\} \rangle$ is *Geo*-complete, *Sens*-complete, consistent, applicable, minimal and satisfies the strict restriction property for topic *Sens*.

However, this new policy does not satisfy the property of *Geo*-awareness anymore, for the following reason: in some situations the rules specify to send the result of applying the P operation to a *Geo*-relevant information item instead of the information item itself, whereas the awareness property requires this information to be sent.

A first important point in the mission of *GMG* is that its agents need to be sent all possible information items related to the *Geo* topic, be it the raw items or the items after a modification such as P, as long as the *Geo*-relevant part is preserved. So, the *Geo*-awareness property for the *GMG* group needs to be reformulated to reflect this nuance. We will now consider that the *GMG* group is *Geo*-aware if and only if any agent outside of the *GMG* and knowing a *Geo*-relevant

information has the obligation to send it to a *GMG* member, or the obligation to send it to a *GMG* member after applying P, as long as P preserves the *Geo*-relevant part of the information.

$$\forall a, \forall i, K(a,i) \wedge \mathit{Topic}(i, \mathit{Geo}) \wedge \neg \mathit{GMG}(a) \implies \\ (\exists b, \mathit{GMG}(b) \wedge (O_{\mathit{Send}}(a,b,i) \vee (\mathit{Topic}(P(i), \mathit{Geo}) \wedge O_{\mathit{Send}}(a,b,P(i)))))$$

Another important point to remark here is that the conditions specifying when to use or not to use P are not given in this property, for the following reason: we want this property to be generic and more abstract than the policies against which it will be checked, and we want it to be only expressed in terms of the *Geo* topic, and not the *Sens* topic.

In fact, the operator P operates as a topic-filtering operator, and corresponds to an operation already routinely performed by any organization managing sensitive data, prior to releasing it to a tier. Depending on the field and exact purpose, this practice is called *declassification*, *sanitisation*, *anonymisation*, *etc.*

In the next sections, we will propose more elegant and general definitions of the filtering operation and awareness property.

6.3. A Generic Information Filtering Operator

In order to model operations such as declassification and its variants in PEPS, we introduce a new generic *filtering operator*, parameterized by a *filtering mode*. Each mode specifies which topics are *preserved* by the filtering and which topics are *removed* from the information item by the filtering.

In PEPS this is modelled by introducing a sort $\mathcal{M}$ representing the filtering modes, the filtering operator as a function with signature $\mathit{Filter}(m : \mathcal{M}, i : \mathcal{I}) : \mathcal{I}$ and two predicates $\mathit{Preserves}(m : \mathcal{M}, t : \mathcal{T})$ and $\mathit{Removes}(m : \mathcal{M}, t : \mathcal{T})$ specifying if a given topic is preserved or respectively removed by a mode.

The following axioms formalize the behaviour of the filtering operator with respect to the mode properties.

Definition 9 (F axioms)

$$\begin{array}{rrcl} \forall t, \forall m, & \mathit{Preserves}(m,t) & \implies & \neg \mathit{Removes}(m,t) \\ \forall i, \forall t, \forall m, & \mathit{Topic}(i,t) \wedge \mathit{Preserves}(m,t) & \implies & \mathit{Topic}(\mathit{Filter}(m,i),t) \\ \forall i, \forall t, \forall m, & \mathit{Topic}(i,t) \wedge \mathit{Removes}(m,t) & \implies & \neg \mathit{Topic}(\mathit{Filter}(m,i),t) \end{array}$$

The first axiom enforces coherence between preservation and removal predicates, it states that if a mode preserves a topic, then it does not remove it. The second (respectively, third) axiom states that if an information item is relevant of a topic preserved (respectively, removed) by a mode, then it is still relevant (respectively, not anymore relevant) of this topic after filtering using this mode.

The ad-hoc P operator, defined in the last section for the running example, can now be replaced by the generic *Filter* operator. First, we declare a constant *FilterSens* of sort $\mathcal{M}$, representing the mode which filters the sensitive contents out of an information item. Second, we add the following domain constraint stating that the mode *FilterSens* preserves the topic *Geo* and removes the topic *Sens*:

$$f: \quad \mathit{Preserves}(\mathit{FilterSens}, \mathit{Geo}) \wedge \mathit{Removes}(\mathit{FilterSens}, \mathit{Sens})$$

Last, all occurrences of $P(i)$ in the rules of the policy are replaced with $\mathit{Filter}(\mathit{FilterSens}, i)$, to obtain a specification expressed in terms of the generic *Filter* operator, and satisfying the same properties.

This characterization of the filtering modes using the *Preserves* and *Removes* predicates is only partial, since other conditions must be taken into account when using the operator, such as the agent capacity to actually perform the filtering, the fact that filtering could be applicable only on information items satisfying specific conditions, *etc.* All extra conditions characterizing filtering modes can be grouped to form what we call a *filtering policy*. Similar notions already exist in the real world, for instance *declassification policies*, *sanitisation policies*, *etc.* The extensibility of PEPS and the expressive power of the underlying logic certainly allows to model such details, but we deliberately do not develop further this topic in the present paper.

6.4. Generic Awareness and Restriction Properties

Wiser from the filtering operator definition we can redefine the awareness property, originally given in Definition 5, into a version not entering in conflict with the restriction properties of Definition 7. Now, a group of agents is T-aware if and only if any agent outside the group knowing a T-relevant information item I has the obligation to send I directly, or to send $Filter(M, I)$ using a filtering mode M preserving the topic T, to at least one agent belonging to the group.

Definition 10 (T-awareness for a group G) *The group G is said T-aware in the presence of filtering if and only if the following property holds:*

$$\mathcal{EPS}, F \models (\forall a, \forall i, K(a,i) \wedge Topic(i,T) \wedge\ G(a) \implies (\exists b, G(b) \wedge (O_{Send}(a,b,i) \vee \exists m, Preserves(m,T) \wedge O_{Send}(a,b,Filter(m,i)))))$$

Unlike the awareness property, the restriction properties do not need reformulation. The non-diffusion of a piece of information related to a restricted topic is achieved by using the filtering operation in the exchange policy. This motivates the use of the *Removes* predicate to specify which topics are removed by each filtering mode.

It can now be checked automatically using the PEPS-analyzer that the policy $\langle \{d,f\}, \{r11, r12, r1b1, r1b2, r2, r3', r4\} \rangle$ rewritten in terms of the *Filter* operator, satisfies both the new *Geo*-awareness and the *Geo*-out-out restriction for the group *GMG* and the strict restriction for *Sens* topic, in addition to *Geo*-completeness, *Sens*-completeness, consistency, applicability and minimality.

7. Conclusion

In this paper, after giving a brief reminder about PEPS, a formal information exchange policy specification language, and about PEPS-analyzer, an SMT-based property checker for PEPS, we introduced and formalized two new antagonist classes of properties: the awareness and the restriction properties. We showed that in some cases satisfying both properties can be logically impossible, which in turn motivated the definition of an *information filtering* operator. Thanks to this new operator, we obtain a framework in which both exchange policies and filtering policies can be specified, and *awareness* and *diffusion restriction* properties can be formally verified.

Ongoing work aims at extending the core PEPS modelling language with notions of *organization* and agent *roles*, taking inspiration from existing work on organization-based and role-based access control policies (OrBAC [9]). This will allow PEPS users to specify more generic and high-level diffusion rules in terms of roles and organizations, and let them assign specific agents to roles and organizations in a second step, for a particular application context of the policy.

Another important topic on our road-map is that of automatically deriving information filtering requirements, based on counter-examples to non-diffusion requirements: for rules involved in the violation of a non-diffusion requirement, PEPS-analyzer will identify where to insert relevant

information filtering operation to prevent non-diffusion violation, while preserving diffusion properties, and present suggestions of modifications of rules to the user.

References

[1] Abramov, J., Anson, O., Dahan, M., Shoval, P., Sturm, A.: A methodology for integrating access control policies within database development. Computers & Security 31(3), 299–314 (2012)

[2] Akl, S.G., Denning, D.E.: Checking classification constraints for consistency and completeness. In: IEEE Symposium on Security and Privacy. pp. 196–201. IEEE Computer Society (1987)

[3] Castanēda, H.N.: Thinking and doing. D. Reidel, Dordrecht (1975)

[4] Cuppens, F., Demolombe, R.: A modal logical framework for security policies. In: Ras, Z.W., Skowron, A. (eds.) ISMIS. Lecture Notes in Computer Science, vol. 1325, pp. 579–589. Springer (1997)

[5] Delmas, R., Polacsek, T.: Formal methods for exchange policy specification. In: Salinesi, C., Norrie, M.C., Pastor, O. (eds.) CAiSE. Lecture Notes in Computer Science, vol. 7908, pp. 288–303. Springer (2013)

[6] Denning, D.E., Akl, S.G., Heckman, M., Lunt, T.F., Morgenstern, M., Neumann, P.G., Schell, R.R.: Views for multilevel database security. IEEE Trans. Software Eng. 13(2), 129–140 (1987)

[7] Gallier, J.H.: Logic for Computer Science: Foundations of Automatic Theorem Proving, chap. 10, pp. 448–476. Wiley (1987)

[8] Halpern, J.Y., Weissman, V.: Using first-order logic to reason about policies. ACM Transactions on Information and System Security (TISSEC) 11(4) (2008)

[9] Kalam, A.A.E., Benferhat, S., Miège, A., Baida, R.E., Cuppens, F., Saurel, C., Balbiani, P., Deswarte, Y., Trouessin, G.: Organization based access contro. In: POLICY. pp. 120–. IEEE Computer Society (2003)

[10] de Moura, L., Bjørner, N.: Z3: An efficient smt solver. In: Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings. Lecture Notes in Computer Science, vol. 4963, pp. 337–340. Springer (2008)

[11] Mouratidis, H., Kalloniatis, C., Islam, S., Huget, M.P., Gritzalis, S.: Aligning security and privacy to support the development of secure information systems. J. UCS 18(12), 1608–1627 (2012)

[12] Paja, E., Dalpiaz, F., Poggianella, M., Roberti, P., Giorgini, P.: Modelling security requirements in socio-technical systems with sts-tool. In: Kirikova, M., Stirna, J. (eds.) CAiSE Forum. CEUR Workshop Proceedings, vol. 855, pp. 155–162. CEUR-WS.org (2012)

[13] Sebastiani, R., Vescovi, M.: Automated reasoning in modal and description logics via sat encoding: the case study of k(m)/alc-satisfiability. J. Artif. Intell. Res. (JAIR) 35, 343–389 (2009)