Formal Modelling of Content-Based Protection and Release for Access Control in NATO Operations

Alessandro Armando[1,2], Sander Oudkerk[3], Silvio Ranise[2], and Konrad Wrona[4(✉)]

[1] DIBRIS, University of Genoa, Genova, Italy
[2] Security and Trust Unit, FBK-Irst, Trent, Italy
[3] Agent Sierra Consultancy Services, Amsterdam, The Netherlands
[4] NATO Communications and Information Agency, The Hague, The Netherlands
konrad.wrona@ncia.nato.int

Abstract. The successful operation of NATO missions requires the effective and secure sharing of information among coalition partners and external organizations, while avoiding the disclosure of sensitive information to unauthorized users. To resolve the conflict between confidentiality and availability in a dynamic coalition and network environment while being able to dynamically respond to changes in protection requirements and release conditions, NATO is developing a new information sharing infrastructure.

In this paper we present the Content-based Protection and Release (CPR) access control model for the NATO information sharing infrastructure. We define a declarative specification language for CPR based on the first-order logical framework underlying a class of efficient theorem-proving tools, called Satisfiability Modulo Theories solvers, and describe how they can support answering authorization queries. We illustrate the ideas in a use case scenario drawn from the NATO Passive Missile Defence system for simulating the consequences of intercepting missile attacks.

1 Introduction

The successful operation of NATO missions requires the effective and secure sharing of information not only among partners of the coalition, but also with external organizations such as the International Committee of the Red Cross (ICRC). To alleviate the conflict between security and availability, suitable access control mechanisms should be adopted. Information sharing in current NATO operations is hampered by the use of security markings as access control conditions [16]. There are three main problems related to such a usage. First, it is not possible to give access to selected parts of a document since security markings are attached to a document as a whole. Indeed, this is one of the main barriers to obtaining a good trade-off between confidentiality and availability. Second, declassification is complex and time-consuming as it requires

J.-L. Danger et al. (Eds.): FPS 2013, LNCS 8352, pp. 227–244, 2014.
DOI: 10.1007/978-3-319-05302-8_14, © NATO/OTAN 2014

manual modification of the markings associated to resources. Declassification is important for NATO: over-classification of data both hampers information sharing within NATO and results in unnecessary cost due to the enhanced protection measures required for classified data. Third, fuzziness is introduced by the (human) interpretation of the policy that determines security markings. This may lead to documents with similar content being labelled with different security markings by different authors.

To overcome these limitations, NATO is developing a new information sharing infrastructure [26] that bases decisions on the release of information contained in data containers (called *structured* resources) in which units of information (called *atomic* resources) are associated to pairs of the form (*element*, *content-metadata*). Selecting a suitable granularity of the atomic resources makes it possible to enable fine-grained access control. Being *content-based*, the new NATO infrastructure will require users to assign correct content-metadata to information instead of inferring the correct classification and release conditions based on numerous NATO directives and policies, e.g., [17–19]. This will reduce the possibility of errors being introduced as a result of subjective interpretation of security directives. Although CPR is mainly motivated by NATO use cases, its applicability is wider since large enterprises and organizations experience similar trade-off between availability and confidentiality of sensitive information.

In this paper we define the *Content-based Protection and Release* (*CPR*) model (Sect. 3) for specifying and enforcing access control policies that arise in complex organizations such as NATO [26]. Similarly to Attribute-Based Access Control (ABAC) [13,28], authorization decisions in CPR are based on the properties, called attributes, that may be associated to the entities (such as users and resources) involved in authorization decisions. For example, the identity, military rank and role are typical attributes of users whereas the pairs (*element*, *content-metadata*) can be modelled as attribute-value pairs associated to resources.

CPR refines ABAC in several respects; one of the most important is that access control policies are decomposed into those specifying *release* conditions and those for *protection* requirements (Sect. 3). Decomposition enables a more efficient implementation of the policies and procedures that are mandated by existing directives within NATO and other international/governmental organizations. These directives prescribe the fulfilment of complex combinations of release conditions and protection requirements. In fact, security officers who are experts in stating conditions on information release are not necessarily experts with respect to defining protection requirements. By separating the two types of policy, a division of responsibilities for the formulation of each policy becomes possible: the protection policy can be formulated by IT security experts with in-depth knowledge of the technical security environment, whereas the release policy can be formulated by experts in the workflow and operational structure of the organization. Another important refinement of CPR with respect to ABAC is the notion of *bridge predicates*, which express relationships among the values of collections of attributes associated to the same set of entities

(Sect. 3.2). For example, NATO security experts can design policies on a set A_1 of content-metadata derived from NATO directives and policies, such as [17–19], facilitating the homogeneous protection of resources with similar content. At the same time, we can think of using available linguistic classification techniques (e.g., [22]) to automatically extract a set A_2 of content-metadata related to the information stored in parts of the documents. It would then be possible to avoid the burden of associating the attribute-value pair in A_1 to the documents if it were possible to define a bridge predicate $B_{1,2}$ relating the values of the attributes in A_1 to those in A_2. Notice how this also allows for separation of concerns: security officers can first design policies focusing on security-relevant attributes and later consider the problem of relating these to the actual content of the documents.

Another contribution of this paper is the definition of the CPR Language, CPRL, which supports the formal specification of CPR policies and authorization queries (Sect. 4). We use the NATO Passive Missile Defence system scenario (described in Sect. 2) to illustrate the various constructs of CPRL.

The framework underlying CPRL is that of Satisfiability Modulo Theories (SMT) [6,24]. CPRL takes advantage of logic-based languages for authorization policies [11] and allows reduction of problems related to answering authorization queries (Sects. 4.1 and 4.2) to theorem-proving problems that can be efficiently tackled by state-of-the-art SMT solvers [6,12].

We also provide an extensive discussion of the motivation and design choices underlying CPR together with a comparison with existing work (Sect. 5).

2 The Passive Missile Defence Scenario

The goal of the NATO Passive Missile Defence (PMD) system is to minimize the effects of missile attacks; to this end, simulations are run in specific geographic areas, taking into account several parameters (e.g., the type of missile and weather conditions). The result of a simulation is a map of the predicted missile impact area, annotated with the consequences of the impact at several locations, hazard areas with risk analysis, the trajectories of the threatening and intercepting missiles, sub-munition locations and descriptions, etc.

Maps generated by the PMD system can be used in NATO missions for crisis-response planning, disaster preparation and rescue, and medical operations, including those that require the coordination of NATO coalition partners with civilian organizations such as the ICRC. The maps can contain different types of graphical objects ranging from threat operating areas to missile trajectories and public information about several zones in the area of the mission. Each graphical object has a different sensitivity level and—in order to realize effective information sharing—it is crucial to be able to disclose each object to only those users who are authorized to see them. For example, a NATO user may see both missile trajectories and public information about the zones of operation, while an ICRC member must not see the former (because he may be able to infer the location from which the intercepting missile was fired) yet should be allowed to access the public information. In addition to release conditions, protection requirements should be enforced

#	from	clr	cat	top
1.a	NATO	Sec	Descr	Toas
1.b	NATO	Sec	Descr	TItds
2	NATO	Res	MCOI	Smal
3	NATO	Pub	PI	Hal
4	*	Pub	PI	*

#	auth	strg	cat	top
1.a	NATO	Enhanced	Descr	Toas
1.b	NATO	Enhanced	Descr	TItds
2	NATO	Basic	MCOI	Smal
3	NATO	NoInfo	MCOI	Hal
4	*	NoInfo	PI	*

Fig. 1. PMD scenario: release conditions (left) and protection requirements (right)

to guarantee that accessed information can be handled with an adequate level of technical and operational support; e.g., data should be downloaded using an SSL protocol and stored in encrypted form on the laptop of the user. The attributes of terminals characterize the technical and operational features of how information is accessed, transmitted, and stored by users.

The user attributes are `from` and `clr`. The value of `from` is the name of the organization to which the user belongs, e.g., `NATO` or `Red_Cross`. The value of `clr` is the user clearance level, i.e. `Public`, `Unclassified`, `Restricted`, `Confidential`, or `Secret`. The terminal attributes are `auth` and `strg`. The value of `auth` is the name of the organization managing the terminal (e.g., `NATO`). The value of `strg` is the level of strength of the protection mechanisms for data offered by the terminal, i.e. `NoInfo`, `Basic`, or `Enhanced`. The resource attributes are `cat` and `top`. The value of `cat` is the content category associated to the graphical object, namely `Descr` (description of geographical zones in the area for which the simulation is conducted), `MCOI` (metrics related to the consequences of the impact between the threatening and intercepting missiles), or `PI` (public information related to the simulation). The value of `top` is the content topic, i.e. `Toas` (threat operating areas), `TItds` (threat and interceptor trajectory details), `Hal` (hazard area location), or `Smal` (sub-munition area location).

Figure 1 shows release conditions and protection requirements in tabular format. For example, the first two lines 1.a and 1.b of the table on the left can be read as follows: `NATO` employees whose clearance level is `Secret` can access resources whose content-metadata label `cat` is equal to `Descr` and `top` can be either `Toas` or `TItds`. The wild-card '`*`' stands for an arbitrary value of the attributes. For example, the last row (4) of the table on the right can be read as follows: a terminal managed by any authority with no information about the configuration can handle resources whose content-metadata label `cat` is `PI`. Implicit in the table on the left is the fact that a higher clearance level (with respect to the standard decreasing order, i.e. `Sec`, `Conf`, `Res`, `Uncl`, `Pub`) subsumes the one explicitly stated. For instance, in the table on the left, a user with clearance `Sec` may be able to get access to resources by lines 2, 3, and 4 in addition to 1.a and 1.b. Similarly in the table on the right, a higher configuration level (with respect to the decreasing ordering, i.e. `Enhanced`, `Basic`, `NoInfo`) subsumes the one mentioned in the table. For instance, a terminal with configuration level `Enhanced` may handle resources by lines 2, 3, and 4 in addition to 1.a and 1.b.

Access criteria are derived by 'joining' the two tables in Fig. 1 over the resource attributes `cat` and `top`. For instance, the 'combination' of the first two lines (identified by 1.a) in the two tables says that a `NATO` employee whose clearance level is `Secret` can access resources whose content-metadata label `cat` is equal to `Descr` and `top` is `Toas` via a terminal managed by `NATO` with strength level `Enhanced`.

3 The CPR Model

The most important feature of CPR is the sharp separation between release conditions and protection requirements. The motivation for such a decoupling is to facilitate the separation of policy management roles by reflecting the current procedures used within NATO, and other international and governmental organizations, that support the independent specification of release conditions and protection requirements. In fact, security officers who are experts in stating conditions on information release are not necessarily experts with respect to defining protection requirements To illustrate, consider the situation in which a user wants to access NATO classified information: (*i*) the user is required to connect to a network infrastructure used for processing NATO classified information and (*ii*) the user should prove that he has the right to access the information. For (*i*), the user should be equipped with a device satisfying a number of technical requirements about hardware and software configurations that are precisely defined in NATO technical directives and security settings documents (access to these documents is restricted; they can be obtained via national NATO contact points). Terminal attributes allow for the expression of both hardware and software configurations such as the hardware model, the type of encryption used to locally store data, and the type of connection used to retrieve resources (e.g., SSL) [26]. A protection policy states the requirements for which a terminal—in a given environment—can manipulate the information contained in a resource. For (*ii*), the user should be able to prove that his security-relevant attributes (such as identity, role, or military rank) entitle him to access information with certain attributes (such as identity, compartment, or sensitivity) when the attributes of the environment (such as time of day or some part of the system state) have certain values. A release policy specifies the conditions under which a user is granted access to a resource in a given environment.

3.1 The Core CPR model

The architecture of a Policy Decision Point (PDP) for the (core) CPR model is shown in Fig. 2. It is structured as two (sub-)decision points and a simple logic to combine them. One decision point computes the release decision by considering the user, resource, and environment properties, called *attributes*, together with the release policy. The other one takes the terminal, resource, and environment attributes together with the protection policy and returns the corresponding access decision. If both access decisions are 'grant', the PDP returns 'grant'; otherwise (i.e. if one of the two decisions is 'deny'), it returns 'deny'.

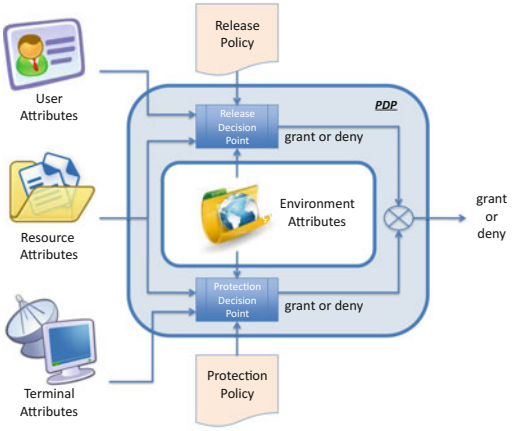

Fig. 2. The architecture of a PDP for the (core) CPR model

Basic entities and attributes. Let U, R, T, and En be infinite sets of users, resources, terminals, and environments, respectively.[1] (It is easy to extend the model to consider actions and their attributes. Here, for the sake of simplicity, we omit this aspect and assume that the only action occurring is reading/viewing information.) Let $\mathcal{E}$ be $\{U, R, T, En\}$, A a finite set of attributes, and $\mathcal{D}$ a collection of sets of attribute values called *domains*. An element of a set E in $\mathcal{E}$ is generically called an *entity*. A set E of entities is associated with a subset A_E of A, called the *attribute signature*. An attribute $a \in A$ ranges over the elements of a domain D_a in $\mathcal{D}$. An *attribute assignment* aa_e associated to an entity e in E is a function mapping the attribute a of the attribute signature A_E to an element of the domain D_a in $\mathcal{D}$.

Predicates on entity attributes. We model release conditions and protection requirements as predicates on the attribute values of the involved entities, i.e. as Boolean functions that map attribute values to true ('grant') or false ('deny'). Let $\mathbf{d}_E = d_{a_1}, \ldots, d_{a_m}$ be a tuple of domain values such that d_{a_i} is in the domain D_{a_i}, a_i is in the attribute signature $A_E = \{a_1, \ldots, a_m\}$, and let there exist an arbitrary (fixed) total order $\sqsubseteq$ over the set A of attributes such that $a_i \sqsubseteq a_{i+1}$ for $i = 1, \ldots, m-1$. A *predicate* $P_{E_1,\ldots,E_n}$ is a Boolean-valued function mapping tuples $(\mathbf{d}_{E_1}, \ldots, \mathbf{d}_{E_n})$ to either true or false. We write $aa_e(A_E)$ to denote the tuple $(aa_e(a_1), \ldots, aa_e(a_n))$ of domain values where aa_e is an attribute assignment, e is an entity in E, and $A_E = \{a_1, \ldots, a_n\}$ is an attribute signature. The attribute assignments $aa_{e_1}, \ldots, aa_{e_n}$ *satisfy* the predicate $P_{E_1,\ldots,E_n}$ if and only if (abbreviated as iff) $P_{E_1,\ldots,E_n}(aa_{e_1}(A_{E_1}), \ldots, aa_{e_n}(A_{E_n}))$ is true, also written as $aa_{e_1}, \ldots, aa_{e_n} \vdash P_{E_1,\ldots,E_n}$.

[1] Assuming these sets to be infinite allows us to model any scenario in which users, resources, terminals, and environments are finitely many.

Policies and access control decisions. A *protection policy* is a predicate $P_{R,T,En}$ on the attribute values of resources, terminals, and environments. $P_{R,T,En}$ (r, t, en) is true when the resource r can be accessed with terminal t in the environment en and false otherwise. A *release policy* is a predicate $P_{U,R,En}$ on the attribute values of users, resources, and environments. $P_{U,R,En(u,r,en)}$ is true when the user u is entitled to access resource r in the environment en and false otherwise. Given a release policy $P_{U,R,En}$ and a protection policy $P_{R,T,En}$, we say that (*1*) a *user u can access resource r in environment en under attribute assignments* aa_u, aa_r, *and* aa_{en} iff $aa_u, aa_r, aa_{en} \vdash P_{U,R,En}$ (see the Release Decision Point in Fig. 2); (*2*) a *terminal t can handle resource r in environment en under attribute assignments* aa_r, aa_t, *and* aa_{en} iff $aa_r, aa_t, aa_{en} \vdash P_{R,T,En}$ (see the Protection Decision Point in Fig. 2); and (*3*) a *user u can access resource r via terminal t in environment en under attribute assignments* aa_u, aa_r, aa_t, and aa_{en} iff (*3.1*) u can access r in en under aa_u, aa_r, and aa_{en} and (*3.2*) t can handle r in en under aa_r, aa_t, and aa_{en} (see the conjunction $\otimes$ in Fig. 2).

3.2 An Extension to the Core CPR Model: Bridge Predicates

We present an extension of the core CPR model in which a set E of entities in $\mathcal{E}$ is associated to a collection $\{A_E^1, \ldots, A_E^m\}$ of attribute signatures. The notions of attribute assignment and predicate, previously introduced, can be easily extended to accommodate multiple attribute signatures by assigning each involved set of entities an integer that uniquely identifies the attribute signature among those associated to it. For instance, aa_e^j denotes an attribute assignment associated to an element e in E with respect to the attribute signature A_E^j $(j = 1, \ldots, m)$ among those in the collection $\{A_E^1, \ldots, A_E^m\}$. Let $A_E^j = \{a_1, \ldots, a_n\}$ be one of the m attribute signatures associated to the set E of entities $(j = 1, \ldots, m)$ and $\mathbf{d}_E^j = d_{a_1}^j, \ldots, d_{a_m}^j$ a tuple of domain values such that $d_{a_i}^j$ is in the domain D_{a_i}; a *bridge* predicate $B_{E^{i_1},\ldots,E^{i_k}}$ (or simply $B_{i_1,\ldots,i_k}$ when E is clear from the context) is a Boolean-valued function on tuples $(\mathbf{d}_E^1, \ldots, \mathbf{d}_E^n)$ where $A_E^{i_1}, \ldots, A_E^{i_k}$ are k attribute signatures among those in $\{A_E^1, \ldots, A_E^m\}$ that are (semantically) related.

Considering several attribute signatures associated to the same set of entities related by bridge predicates can be useful in several situations. For instance, consider the problem of migrating actual authorization policies about NATO documents based on security markings—which can grant or deny access to whole documents only—to CPR policies that allow for accessing selected portions. Migration can be achieved by exploiting linguistic classification techniques guided by terminologies (along the lines of, e.g., [22]) to automatically infer a collection A_R^1 of content-metadata associated to portions of documents (e.g., paragraphs and sections). It would then be possible to apply classification techniques developed in Formal Concept Analysis [25] to automatically synthesize bridge predicates $B_{1,2}$ between the inferred content-metadata in A_R^1 and a collection A_R^2 of security-relevant attributes identified by security experts. At this point, the experts can use A_R^2 to express the desired release policy $P_{U,R^2,En}$ on

selected parts of the documents. Defining $P_{U,R^2,En}$ is likely to take a substantial amount of time. For availability, it is essential to find a simpler way to mediate access to the documents. This is possible, for example, by synthesizing an alternative bridge predicate $B'_{1,0}$ that associates the content metadata in A^1_R with the (standard) NATO security markings in A^0_R, namely Top Secret, Secret, Confidential, Restricted, and Unclassified. This would allow for the re-use of the available (and well-known in NATO [21]) authorization policy $P_{U,R^0,En}$ involving security markings to handle the migration to CPR policies as follows: a user u can access resource r in environment en under attribute assignments aa_u, aa^1_r, and aa_{en} iff either (i) there exists a tuple $\mathbf{s}$ of values for the security-sensitive attributes in A^2_R such that $(aa^1_r(A^1_R), \mathbf{s}) \in B_{1,2}$ and $P_{U,R^2,En}(aa_u(A_U), \mathbf{s}, aa_{en}(En))$ is true, or (ii) there exists a tuple $\mathbf{m}$ of security markings such that $(aa^1_u(A^1_R), \mathbf{m}) \in B'_{1,0}$ and $P_{U,R^0,En}(aa_u(A_U), \mathbf{m}, aa_{en}(A_{En}))$ is true. According to the above definition, if clause (i) holds it is possible to apply a release policy based on the security-sensitive attributes in A^2_R; otherwise it is possible to resort to a release policy based on traditional security markings, see clause (ii). Use of bridge predicates supports coexistence of systems relying on both $P_{U,R^2,En}$ and $P_{U,R^0,En}$, at the price of reduced granularity in access, since security markings apply to whole documents whereas the security-sensitive attributes in A^2_R apply to paragraphs or sections.

4 CPRL: A Language for the Core CPR Model

We base CPRL on the theory of Satisfiability Modulo Theories (SMT) solvers [6, 24] whereby theories are used to model the algebraic structure of types.

Basic framework. We assume the availability of a set of primitive types, including Integers ($\mathbb{Z}$) and Booleans ($\mathbb{B}$). We also assume the possibility of defining enumerated types and records. For instance, the enumerated types for the PMD scenario (described in Sect. 2) are the following: `Clr` := {`Pub`, `Uncl`, `Res`, `Conf`, `Sec`}, `Level` := {`NoInfo`, `Basic`, `Enhanced`}, `Categ` := {`Descr`, `MCOI`, `PI`}, and `Topic` := {`Toas`, `TItds`, `Hal`, `Smal`}, where the symbol ':=' introduces an abbreviation (left) for an expression (right). The values of `Clr` are the clearance levels, those of `Level` are the strengths of the terminals, those of `Categ` are the content categories of the objects in the maps, and those of `Topic` are the content types of the objects. The record types of the entities in the PMD scenario are: `User` := [`from` : `Org`, `clr` : `Clr`], `Terminal` := [`auth` : `Org`, `strg` : `Level`], and `Resource` := [`cat` : `Categ`, `top` : `Topic`], where the record fields correspond to the attributes with the same name in Sect. 2. If r is a record type expression, then $r.f$ denotes the value of the field f in r.

Attribute assignment expressions. Let e be a variable of type `User`, `Resource`, `Terminal`, or `Environment`; an *attribute assignment expression* is a quantifier-free formula $\alpha(e)$ in which e is the only variable that occurs. An attribute assignment expression $\alpha(e)$ denotes the set $[\![\alpha(e)]\!]$ of elements whose attribute values satisfy (in the logical sense) $\alpha(e)$. In other words, $\alpha(e)$ identifies the set of attribute assignments that map the attributes of e to values satisfying α.

For instance, in the PMD scenario consider the following attribute assignment expressions for users and terminals:

$$\texttt{UN.from} = \texttt{NATO} \wedge \texttt{UN.clr} = \texttt{Sec} \tag{1}$$

$$\texttt{TN.auth} = \texttt{NATO} \wedge (\texttt{TN.strg} = \texttt{Enhanced} \vee \texttt{TN.strg} = \texttt{Basic}) \tag{2}$$

where UN is a variable of type User and TN is a variable of type Terminal. The former identifies a set containing just one attribute assignment that maps from to NATO and clr to Sec. The latter identifies the set containing two attribute assignments that map auth to NATO and strg to either Basic or Enhanced.

CPRL expressions. A *predicate expression over types* $t_1, \ldots, t_n$ is a quantifier-free formula $\gamma(e_1, \ldots, e_n)$ containing (at most) the variables $e_1, \ldots, e_n$ of types $t_1, \ldots, t_n$, respectively. A predicate expression $\gamma(e_1, \ldots, e_n)$ denotes the set $[\![\gamma]\!]$ of tuples of n elements whose attribute values satisfy the formula π. A *release policy expression* is a predicate expression over User, Resource, and Environment. A *protection policy expression* is a predicate expression over Resource, Terminal, and Environment.

To express release and protection policies in the PMD scenario, we introduce attribute assignment expressions over User to describe groups of users whose clearance level is higher than or equal to a given value. For instance, the group he_Res of users whose clearance level is higher than or equal to Restricted can be described by the following attribute assignment expression over User:

$$\texttt{he\_Res}(u) := (u.\texttt{clr} = \texttt{Res} \vee u.\texttt{clr} = \texttt{Conf} \vee u.\texttt{clr} = \texttt{Sec})$$

where u is a variable of type User.

For instance, he_Res(UN) stands for UN.clr = Res ∨ UN.clr = Conf ∨ UN.clr = Sec, which is indeed satisfied by the set $[\![(1)]\!]$ of attribute assignments. This implies that UN belongs to the group of users whose clearance level is higher than or equal to Res. It is easy to see that

$$\forall u : \texttt{User}, u.\texttt{clr} = \texttt{Sec} \Rightarrow \texttt{he\_Res}(u) \tag{3}$$

holds, where ⇒ denotes logical implication. In other words, the (increasing) order over clearance levels is modelled in CPRL by logical implication.

To see how he_Res is helpful in writing release policies, consider line 2 of the table on the left of Fig. 1, i.e. NATO employees with clearance Res or higher can access resources whose content-metadata label cat is MCOI and top is Smal. This can be expressed by the following release policy expression:

$$\texttt{rp\_2}(u, r) := (u.\texttt{from} = \texttt{NATO} \wedge \texttt{he\_Res}(u) \wedge r.\texttt{cat} = \texttt{MCOI} \wedge r.\texttt{top} = \texttt{Smal}),$$

where u is a variable of type User and r a variable of type Resource.

Now, recall the attribute assignment expression (1) and consider the following question: can the user UN access an object Obj_2 such that

$$\texttt{Obj\_2.cat} = \texttt{MCOI} \wedge \texttt{Obj\_2.top} = \texttt{Hal}\ ? \tag{4}$$

The answer can be found by evaluating the truth value of $\texttt{rp\_2}(\texttt{UN}, \texttt{Obj\_2})$. For this, consider the part of $\texttt{rp\_2}(\texttt{UN}, \texttt{Obj\_2})$ that applies to UN, namely $\texttt{UN.from} = \texttt{NATO} \wedge \texttt{he\_Res}(\texttt{UN})$, that is satisfied by the set $[\![(1)]\!]$ of attribute assignments; notice that $\texttt{he\_Res}(\texttt{UN})$ is implied by $\texttt{UN.clr} = \texttt{Sec}$ and (3). Then, consider the part of $\texttt{rp\_2}(\texttt{UN}, \texttt{Obj\_2})$ that applies to Obj_2, namely $\texttt{Obj\_2.cat} = \texttt{MCOI} \wedge \texttt{Obj\_2.top} = \texttt{Smal}$: although $\texttt{Obj\_2.cat} = \texttt{MCOI}$ is satisfied by $[\![(4)]\!]$, this is not the case for $\texttt{Obj\_2.top} = \texttt{Smal}$ because $\texttt{Smal} \neq \texttt{Hal}$ and $\texttt{Obj\_2.top} = \texttt{Hal}$ is in (4). Thus, $\texttt{rp\_2}(\texttt{UN}, \texttt{Obj\_2})$ is false and UN cannot access Obj_2; i.e. the answer to the question above is no.

The protection policies for the PMD scenario can be specified in a similar way; they are omitted for lack of space.

4.1 Answering Authorization Queries in CPRL

An *authorization query expression* is a quantifier-free formula of the form

$$\alpha_U(u) \wedge \alpha_R(r) \wedge \alpha_T(t) \wedge \alpha_{En}(en) \wedge \rho(u, r, en) \wedge \pi(r, t, en) \tag{5}$$

where u, r, t, and en are variables of type User, Resource, Terminal, and Environment, respectively, α_U, α_R, α_T, and α_{En} are attribute assignment expressions over User, Resource, Terminal, and Environment, respectively, ρ is a release policy expression, and π is a protection policy expression. The satisfiability of (5) will be denoted as $[\![\alpha_U]\!], [\![\alpha_R]\!], [\![\alpha_T]\!], [\![\alpha_{En}]\!] \vdash [\![\rho \wedge \pi]\!]$. The intuition underlying this notation is the following: when (5) is satisfiable, the entities u, r, t, and en satisfy—in the sense of Sect. 3—both ρ and π under some attribute assignments aa_u in $[\![\alpha_U]\!]$, aa_r in $[\![\alpha_R]\!]$, aa_t in $[\![\alpha_T]\!]$ and aa_{en} in $[\![\alpha_{En}]\!]$, i.e. $aa_u, aa_r, aa_t, aa_{en} \vdash [\![\rho \wedge \pi]\!]$. When $[\![\alpha_U]\!], [\![\alpha_R]\!], [\![\alpha_T]\!], [\![\alpha_{En}]\!] \vdash [\![\rho \wedge \pi]\!]$, we say that the answer to the authorization query expression (5) is 'permit,' otherwise, it is 'deny.' Thus, CPRL supports the specification of positive authorizations only. While negative authorizations can be useful to specify exceptions (see, e.g., [11]), their interplay with positive authorizations may give rise to conflicts, i.e. situations for which an access query is permitted by a positive authorization and denied by a negative authorization. To avoid this kind of problems, in this paper, we have chosen to consider positive authorizations only and leave to future work the development of extensions of CPRL capable of supporting negative authorizations.

Let P and Π be (possibly empty) finite sets of release and protection policy expressions, respectively, the authorization query expression (5) is satisfied by P and Π iff there exists $\rho \in P$ and $\pi \in \Pi$ such that $[\![\alpha_U]\!], [\![\alpha_R]\!], [\![\alpha_T]\!], [\![\alpha_{En}]\!] \vdash [\![\rho \wedge \pi]\!]$. When P or Π is empty, any authorization query expression is unsatisfiable because it is obviously impossible to find a release or a protection policy (depending on which of the sets P or Π is empty) satisfying the query.

Intuitively, the capability of answering an authorization query α_E allows for establishing that some entities in the (large) set $[\![\alpha_E]\!]$ satisfy certain conditions, expressed as the attribute assignment expression α_E. For instance, since the attribute assignment expression $\alpha_U(u) := (u.\texttt{from} = \texttt{NATO})$ identifies the group

of all NATO employees, the capability of answering an authorization query of the form (5) in which this α_U occurs allows us to establish if there exists a NATO employee able to access certain resources in some environment, regardless of his rank and clearance. In this way, policy designers can check that the release and protection policies grant or deny access to users (with certain profiles) to some given resources. In the PMD scenario, for example, one may wish to check that a NATO user may see objects whose content-metadata labels are `MCOI` (metrics related to the consequences of the impact between the threatening and intercepting missiles) and `PI` (public information) while an ICRC member is granted access to objects labelled `PI` but should not see those with `MCOI`, to prevent him from being able to infer details about missile trajectories.

We now give conditions under which it is possible to automatically answer authorization queries.

Theorem 1. *Let $\{\mathcal{T}_j\}_{j\in J}$ be the collection of theories formalizing the types of attributes of users, resources, terminals, and environments (for J some finite index set). If checking the satisfiability of quantifier-free formulae in $\mathcal{T}_j$ is decidable for each $j \in J$, then answering authorization queries of the form (5) is decidable. Furthermore, if satisfiability checking in $\mathcal{T}_j$ is in NP (for each $j \in J$), then answering authorization queries is also in NP.*

Proof (*Sketch*). The problem of answering authorization queries reduces to checking the satisfiability of the quantifier-free formulae of the form (5) (observe that (5) is a quantifier-free formula because, by definition, all its parts are quantifier-free) in the theory obtained by combining the theories for records for users, resources, terminals, and environments with the theories for the fields in the collection $\{\mathcal{T}_j\}_{j\in J}$. This is a well-studied problem in SMT solving: the theorem follows from the main result in [23]. □

Modern SMT solvers have been shown to be quite effective in handling a wide range of instances of satisfiability problems in NP (see, e.g., [12]). Our experience in using the SMT solver Yices [27] on the PMD scenario was that answering authorization queries is almost instantaneous on a standard laptop.

4.2 Computing Permitted Views by Max-SMT Solving

Computing yes-or-no answers to authorization queries is not always enough to provide an optimal permitted view to structured resources. For example, in the PMD scenario, it is necessary to compute permitted views of a map that depend on the user and terminal used to access it. In other words, the content of a map generated by the PMD system must be filtered to produce views in which, for instance, the trajectories of the threatening and intercepting missiles are included for a NATO user equipped with a NATO terminal. A member of ICRC coordinating a rescue operation can, instead, see the map annotated with the consequences of impact in hazard areas and their location, but the missile trajectories are omitted. The problem of computing answers to multiple authorization

queries pertaining to the parts of the same (structured) resource can be reduced to an optimization problem for which some SMT solvers (e.g., Yices) offer support.

The *problem of computing permitted views* in CPRL can be stated as follows (from an authorization perspective): given a release policy expression ρ, a protection policy expression π, attribute expressions α_U, α_T, α_{En} over `User`, `Terminal`, `Environment`, respectively, and a finite set AER of attribute assignment expressions over `Resource`, find the largest possible subset AER' of AER such that $\alpha_R \in AER'$ iff $[\![\alpha_U]\!], [\![\alpha_R]\!], [\![\alpha_T]\!], [\![\alpha_{En}]\!] \vdash [\![\rho \wedge \pi]\!]$.

By Theorem 1 it is possible to automatically compute permitted views.

Theorem 2. *Let $\{\mathcal{T}_j\}_{j\in J}$ be the collection of theories formalizing the types of attributes of users, resources, terminals, and environments (for J some finite index set). If checking the satisfiability of quantifier-free formulae in $\mathcal{T}_j$ is decidable for each $j \in J$, then the problem of computing permitted views is decidable.*

Proof. The proof is constructive. Given a release policy expression ρ, a protection policy expression π, attribute expressions α_U, α_T, α_{En} over `User`, `Terminal`, `Environment`, respectively, and a finite set AER of attribute assignment expressions over `Resource`, we describe an algorithm capable of returning the largest subsets AER' of AER whose elements α_R are such that $[\![\alpha_U]\!], [\![\alpha_R]\!], [\![\alpha_T]\!], [\![\alpha_{En}]\!] \vdash [\![\rho \wedge \pi]\!]$.

For each subset AER' of AER, consider the following 'associated' formula: $\bigwedge_{\alpha_R \in AER'}(\alpha_U \wedge \alpha_R \wedge \alpha_T \wedge \alpha_{En} \wedge \rho \wedge \pi)$. Notice that each conjunct in this formula is of the form (5), i.e. it is an authorization query expression. By Theorem 1, it is possible to check the satisfiability of each conjunct of this formula and thus also of the formula. After considering all subsets of AER, the procedure simply returns the largest subsets (if any) whose associated formula is satisfiable.

To conclude the proof, it is sufficient to notice that the procedure terminates because there are only finitely many subsets of AER. □

The goal is to design efficient ways to solve instances of the problem of computing permitted views by avoiding the enumeration of all possible subsets of attribute assignment expressions for resources, as is suggested by the proof of Theorem 2. Fortunately, it is possible to reduce this problem to an optimization problem, called a Max-SMT (MSMT) problem (see, e.g., [2]), whose solution is supported by some state-of-the-art SMT solvers.

The Max-SMT problem. We say that a formula is *soft* when it may or may not be satisfiable and that it is *hard* when it must be satisfiable. Given a theory $\mathcal{T}$ and two sets H and S of hard and soft formulae, respectively, the Max-SMT problem consists of finding a subset S' of S such that the conjunction of the formulae in H and the cardinality of the set $S \setminus S'$ is minimal. I.e., a solution to a Max-SMT problem must satisfy all hard formulae and minimize the number of unsatisfied soft formulae.

Reduction. Let PV be an instance of the problem of computing permitted views: ρ is a release policy expression, π is a protection policy expression, α_U, α_T,

α_{En} are attribute expressions over `User`, `Terminal`, `Environment`, respectively, and AER is a finite set of attribute assignment expressions over `Resource`. We reduce PV to the following instance of the Max-SMT problem: $\mathcal{T}$ is the theory obtained by combining the theories for the records `User`, `Resource`, `Terminal`, and `Environment` with the theories for their fields, $S_{MSMT} := AER$ is the set of soft formulae, and $H_{MSMT} := \{\alpha_U \wedge \alpha_T \wedge \alpha_{En} \wedge \rho \wedge \pi\}$ is the set of hard formulae. It is easy to see that a solution S'_{MSMT} of MSMT is also a solution of PV, i.e. $AER'_{\text{PV}} = S'_{MSMT}$, and the solution is such that, for each $\alpha_R \in AER'_{\text{PV}}$, $[\![\alpha_U]\!], [\![\alpha_R]\!], [\![\alpha_T]\!], [\![\alpha_{En}]\!] \vdash [\![\rho \wedge \pi]\!]$. In other words, the reduction is correct.

It is possible to support a simple form of risk-based[2] access control by using a variant of the Max-SMT problem, called the weighted Max-SMT problem. In addition to the sets H and S of hard and soft formulae, respectively, we consider a function w that assigns weights to each element in S. The solution to a weighted Max-SMT problem must then satisfy all hard formulae in H and minimize the weights of the unsatisfied soft formulae in S. We can then define a risk-based version of the problem of computing permitted views by defining a weight function w that associates the advantage (as opposed to risk) of disclosing the information in the resources identified by the attribute assignment expressions in AER to each soft formula in S. The solution to this problem will be a subset of AER that minimizes the cumulative advantage associated to the attribute assignment expressions that are not satisfied, thereby minimizing the risk of disclosing those pieces of information. Indeed, there are a number of ways of explicitly defining the function w depending on the scenario to be considered. For example, in the PMD scenario, w may assign the value of showing an object in the map according to user location and time; e.g., the value increases as the user gets closer to a certain hazard area or to the actual time of the foreseen impact of an intercepting missile.

Similarly to what has already been observed for answering authorization queries, our experience in using the (weighted) Max-SMT solving capabilities of Yices [27] in the PMD scenario was that computing permitted views takes at most 1 second on a standard laptop.

We conclude this section by observing that it is easy to extend CPRL to include bridge predicates. We do not do this here for lack of space.

5 Discussion and Related Work

The main goal of CPR is the seamless integration of access control in the NATO information sharing infrastructure [26]. Two design decisions are key to achieving this goal. First, CPR assumes that resources are labelled with content-metadata carrying sufficient information about their content to allow access decisions to be made. Thus, the type of a resource is not required to be known to the PDP of CPR (Fig. 2) but can be delegated to another module

[2] In CPR, the risk measures the severity of disclosing a piece of information contained in a resource combined with the likelihood that a user and a terminal maliciously or inadvertently leak it.

in the NATO infrastructure. An implementation of this idea as a refinement of the XACML (eXtensible Access Control Markup Language) architecture [20] is described in [4].

Relationship with other content-based authorization models. Existing content-based authorization models [1] base their access decisions on the 'concepts' related to resources (e.g., books in digital libraries). Concepts are extracted from resources and thus their type (e.g., textual documents) must be known to access control systems. An advantage of this approach is the possibility to capture the semantic relationships among the concepts (e.g., hierarchies) and to specify authorization conditions involving such relationships. This can be useful in the context of the NATO infrastructure. To see why, consider the PMD scenario: while it may be legitimate to disclose the position of a few soldiers in a certain area, disclosure should be avoided when their number is one hundred or more, since (untrusted) users may infer that a military operation is likely to occur in that area. We can extend the CPR model to take into account this kind of semantic relationship (among content-metadata) by generalizing attribute assignment expressions and permitting the occurrence of more than one variable of type `Resource`. Such expressions can, for instance, encode constraints on the type (e.g., soldier) and number (few or many) of objects on a map. It is then straightforward to adapt the reduction to a Max-SMT problem as described in Sect. 4.2 to also consider this kind of predicate expression. We leave the investigation of this interesting issue as future work.

The other main design decision underlying CPR that facilitates its integration in the NATO information sharing infrastructure is the sharp separation between release and protection policies. This enables a more efficient implementation of the policies and procedures that are mandated by existing directives within NATO and other international/governmental organizations. It also promotes an effective division of responsibilities for the formulation of each policy: experts with in-depth knowledge of technical security environments for protection requirements and experts with in-depth knowledge of organization workflows and operational structure for release conditions. To support the natural and modular specification of release and protection policies, CPR introduces the notion of terminal, encapsulating the properties of the device and connection used for accessing information. By using terminals and protection policies (as a complement to release policies), CPR naturally supports the specification of authorizations based on devices. These are gaining more and more importance in large enterprise organizations because of the widespread use of corporate or even personal (with bring-your-own-device policies, see, e.g., [15]) mobile devices to access work-related information. Thus, the applicability of CPR extends beyond NATO to large enterprise or governmental organizations, seeking a good trade-off between confidentiality and availability of information.

Relationship with classical authorization models. The separate specification of release conditions and protection policies is one of the main refinement of CPR with respect to ABAC. In the latter, the properties of terminals are seen as part

of the environment [28] thus making the specifications of protection requirements more difficult to read and maintain. Another important refinement of CPR with respect to ABAC is the notion of bridge predicate (Sect. 3.2), which encodes relationships among different collections of attributes associated to a given set of entities. Use of bridge predicates allows decomposition of policy specifications by allowing security officers to focus on security-relevant attributes of the resources in isolation and later relate these to the content-metadata of resources.

Since it is a refinement of ABAC, CPR inherits its expressive power. As observed in [13], ABAC can express and extend the three classical models (namely discretionary, mandatory, and role-based; see, e.g., [11] for an overview) as well as others that mix mandatory and discretionary features (such as Bell-La Padula [7]). Thus, CPR also allows policies inspired by these classical models to be expressed and combined.

Relationship with XACML. As sketched in [4], release conditions and protection policies can be translated into XACML [20]. This allows for re-using available XACML engines to implement the PDP for the CPR model (Fig. 2). The crucial difference between an authorization query in CPRL (Sect. 4.1) and one in XACML is that the former corresponds to a (large) set of the latter. This is because a query in CPRL is defined in terms of attribute assignment expressions, which identify (large) sets of attribute assignments for the involved entities. Thus, the technique for answering authorization queries based on SMT solvers (Theorem 1) enables the handling of more general questions than does the technique based on XACML engines. The situation is similar to comparing standard and symbolic execution of programs [14]. Instead of executing a program on a set of sample inputs, a program can be 'symbolically' executed for a set of classes of inputs so that a single symbolic execution result may be equivalent to a large number of normal test cases. Similarly, instead of answering an authorization query with respect to a single user, resource, terminal, and environment, the technique in Sect. 4.1 can answer a (large) set of authorization queries by using attribute assignment expressions for groups of users, resources, terminals, and environments (specified by attribute assignment expressions). Using this technique, policy designers can check the consequences of their policies against (large) sets of attribute assignments, thereby gaining a deeper understanding of their policies and facilitating the reconciliation between what they intend to authorize and what the policies actually authorize.

Relationship with SMT solvers techniques. It is recognized that policy designers can be assisted in their tasks by tools capable of automatically solving policy analysis problems. CPRL permits the re-use of results and techniques from the SMT field once policy analysis problems are reduced to SMT problems (this is similar in spirit to the work in [3,5]). This is possible for several policy analysis problems. An interesting example is checking the observational equivalence of two policies (see, e.g., [8]), which amounts to establishing whether or not it is always the case that the policies return the same answer for any authorization request. In CPRL, the problem can be stated as follows: given release policy ρ_i

and protection policy π_i expressions for $i = 1, 2$, the *observational equivalence problem* amounts to checking whether or not for every user u, resource r, terminal t, and environment en, the formula

$$(\rho_1(u, r, en) \wedge \pi_1(r, t, en)) \Leftrightarrow (\rho_2(u, r, en) \wedge \pi_2(r, t, en)) \tag{6}$$

holds, where $\Leftrightarrow$ denotes logical equivalence. By Theorem 1 it is possible to automatically solve this problem.

Theorem 3. *Let $\{\mathcal{T}_j\}_{j \in J}$ be the collection of theories formalizing the types of attributes of users, resources, terminals, and environments (for J some finite index set). If checking the satisfiability of quantifier-free formulae in $\mathcal{T}_j$ is decidable for each $j \in J$, then the observational equivalence problem is also decidable.*

Proof. The key argument is similar to that in the proof of Theorem 1. By refutation, the fact that (6) holds is equivalent to the unsatisfiability of its negation, or equivalently to the unsatisfiability of the negation of (6), which is indeed a quantifier-free formula whose satisfiability is decidable by Theorem 1. □

Relationship with Datalog-based access control models. Compare Theorem 3 with the undecidability result in [8] for the same problem when policies are expressed in Datalog (see, e.g., [9]). In the past, Datalog was widely used as the semantic foundation of access control languages because the algorithms for answering authorization queries are polynomially complex. Polynomial complexity was achieved by requiring the formulae of Datalog to contain only constant and predicate symbols, and admitting only restricted forms of negation and disjunction. In contrast, CPRL permits arbitrary Boolean combinations (i.e. unrestricted negation and disjunction) in policy expressions together with the use of function symbols. However, while recursion is available in Datalog, this is not the case in CPRL. It is the presence of recursion that makes the observational equivalence problem undecidable for policies expressed in Datalog [8]. At the same time, the absence of recursion together with the results of Theorem 1 make observational equivalence decidable in CPRL (Theorem 3). To date, our experience has been that recursion is not needed for expressing release and protection policies arising in typical NATO scenarios.

CPR and structural metadata. So far, in the CPR model, values of resource attributes have been used for representing content-metadata. Another possibility is to use resource attributes to represent the design and specification of data structures (structural metadata). For instance, it would be interesting to investigate to what extent it is possible to encode the sophisticated access control model for XML documents and schema proposed in [10]. In future work, we envisage extending CPRL with theories that would allow the representation of structural relationships such as those in XML schemas and documents.

References

1. Adam, N.R., Atluri, V., Bertino, E., Ferrari, E.: A content-based authorization model for digital libraries. IEEE Trans. Knowl. Data Eng. **14**(2), 296–315 (2002)
2. Ansótegui, C., Bofill, M., Palahí, M., Suy, J., Villaret, M.: Satisfiability modulo theories: an efficient approach for the resource-constrained project scheduling problem. In: SARA, AAAI (2011)
3. Arkoudas, K., Loeb, S., Chadha, R., Chiang, J., Whittaker, K.: Automated policy analysis. In: 2012 IEEE International Symposium on Policy, July 2012, pp. 1–8 (2012)
4. Armando, A., Grasso, M., Oudkerk, S., Ranise, S., Wrona, K.: Content-based information protection and release in NATO operations. In: SACMAT, ACM (2013)
5. Armando, A., Ranise, S.: Automated and efficient analysis of role-based access control with attributes. In: Cuppens-Boulahia, N., Cuppens, F., Garcia-Alfaro, J. (eds.) DBSec 2012. LNCS, vol. 7371, pp. 25–40. Springer, Heidelberg (2012)
6. Beckert, B., Hoare, C.A.R., Hähnle, R., Smith, R., Green, D.R., Ranise, S., Tinelli, C., Ball, T., Rajamani, S.K.: Intelligent systems and formal methods in software engineering. IEEE Int. Syst. **21**(6), 71–81 (2006)
7. Bell, D.E., La Padula, L.J.: Secure computer systems: mathematical foundations. Technical report MTR-2547, MITRE Corporation, Bedford (1973)
8. Bertino, E., Catania, B., Ferrari, E., Perlasca, P.: A logical framework for reasoning about access control models. ACM TISSEC **6**(1), 71–127 (2003)
9. Ceri, S., Gottlob, G., Tanca, L.: What you always wanted to know about datalog (and never dared to ask). IEEE TKDE **1**(1), 146–166 (1989)
10. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P.: A fine-grained access control system for XML documents. TISSEC **5**(2), 169–202 (2002)
11. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Samarati, P.: Access control policies and languages. Int. J. Comput. Sci. Eng. **3**(2), 94–102 (2007)
12. De Moura, L., Bjørner, N.: Satisfiability modulo theories: introduction and applications. CACM **54**, 69–77 (2011)
13. Jin, X., Krishnan, R., Sandhu, R.: A unified attribute-based access control model covering DAC, MAC and RBAC. In: Cuppens-Boulahia, N., Cuppens, F., Garcia-Alfaro, J. (eds.) DBSec 2012. LNCS, vol. 7371, pp. 41–55. Springer, Heidelberg (2012)
14. King, J.C.: Symbolic execution and program testing. CACM **19**(7), 385–394 (1976)
15. Casassa Mont, M., Balacheff, B.: On device-based identity management in enterprises. Technical report HPL-2007-53, Helwett-Packard Laboratories (2007)
16. Nato HQ C3 Staff. Directive on the marking of NATO information (2011)
17. NATO Security Committee. Primary directive on INFOSEC. Number AC/35-D/2004-REV2 (2010)
18. North Atlantic Council. Security within the North Atlantic Treaty Organization. Number C-M(2002) 49 (2002)
19. North Atlantic Council. Primary directive on information management. Number C-M(2008) 0113(INV) (2008)
20. OASIS. eXtensible Access Control Markup Language (XACML) version 3.0. (2010)
21. Oudkerk, S., Bryant, I., Eggen, A., Haakseth, R.: A proposal for an XML confidentiality label syntax and binding of metadata to data objects. In: NATO RTO Symposium on Information, Assurance and Cyber Defence (2010)
22. Prediger, S.: Logical scaling in formal concept analysis. In: Delugach, H.S., Keeler, M.A., Searle, L., Lukose, D., Sowa, J.F. (eds.) ICCS 1997. LNCS (LNAI), vol. 1257, pp. 332–341. Springer, Heidelberg (1997)

23. Ranise, S., Ringeissen, C., Zarba, C.G.: Combining data structures with nonstably infinite theories using many-sorted logic. In: Gramlich, B. (ed.) FroCos 2005. LNCS (LNAI), vol. 3717, pp. 48–64. Springer, Heidelberg (2005)
24. Ranise, S., Tinelli, C.: The SMT-LIB standard: version 1.2. http://www.smt-lib.org (2006)
25. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered Sets, pp. 445–470. Reidel, Dordrecht (1982)
26. Wrona, K., Hallingstad, G.: Controlled information sharing in NATO operations. In: IEEE Military Communications Conference (MILCOM) (2011)
27. Yices. http://yices.csl.sri.com (2013)
28. Yuan, E., Jin, T.: Attribute-based access control (ABAC) for web services. In: IEEE International Conference on Web Services, pp. 561–569 (2005)