Access Control: What is Required in Business Collaboration?

Daisy Daiqin He[1] **Michael Compton**[2] **Kerry Taylor**[2] **Jian Yang**[1]

[1] Department of Computing, Macquarie University
Sydney Australia,
Email: {daiqin, jian}@ics.mq.edu.au

[2] ICT Centre
CSIRO Australia,
Email: {michael.compton, kerry.taylor}@csiro.au

Abstract

Access control has been studied for sometime, and there are a number of theories and techniques for handling access control for single or centralised systems; however, unique and challenging security issues concerning collaboration in the context of service oriented computing (SOC) have arisen due to the dynamic and loosely coupled nature of the environment in which these collaborations are conducted. Individual organisations usually define their access control policies independently. When a collaboration opportunity arrives, a number of problems arise, such as: determining if the collaboration is possible given the access control policies, defining the policy for the collaboration and deciding under what conditions a service is allowed to be forwarded to other parties. Furthermore, different types of collaboration, in terms of the way collaboration is carried out, require different access control support. In this paper, we propose a model encoded in description logic to capture all the necessary elements for specifying access control policy for collaboration. Based on the model, various inconsistencies between access policies from different business units are identified. The paper also shows how a description logic reasoner can be used to prove that two policies are suitable, or not suitable, for collaboration. The policy model and policies are encoded in a $\mathcal{SROIQ}$ knowledge base. Although access control policies focus on a single system or a single business party's requirements, the method presented in this paper allows a logical analysis of the suitability of potential collaboration partners. We believe this work is laying a foundation for access policy development, negotiation and enforcement for cross-organization collaborations.

1 Introduction

Although Web Service technologies provide technological support for dynamic, cross-organization collaboration, security concerns can be a barrier to the adoption of this new technology. Service collaboration through service compositions or other means, could have different access control requirements to the individuals services in the collaboration; how to provide end-to-end security guarantees is still an unsolved problem. This paper discuses a method that can be used to solve part of this problem by using technologies designed for the Semantic Web and Services to analyse the access control policies of potential collaboration partners and prove if collaboration will respect the policies. We emphasize, however, that the proposed method is not restricted to Web Service. The problem we are addressing could be applicable to generic business collaboration domain.

The access control policy of a single organisation or service is defined (in a role-based model) in terms of roles and their privileges. Given a request to access a resource or perform an operation, the service enforces the policy by analysing the credentials of the requester and deciding if the requester is authorised to perform the actions in the request.

Organisations collaborate with each other in various ways. The collaboration could be through an agent, or a direct collaboration to provide joint service. Before organizations engage in collaboration, their authorization policies need to be analyzed to decide the possibility of collaboration under the authorization constraints defined by each individual party. Therefore, the consistency of access policies of different organisations needs to be evaluated before a collaboration can be formed.

Collaborations can reveal the differences between the participants policies. For example, if a radiology institute wants to collaborate with a medical centre, accepting on line bookings from the medical centre, but inconsistencies exist between two party's access control policies, then either the collaboration could not be established securely or some negotiation is required to form the collaboration.

In a collaboration, a service can be accessed by a party that can pass it to other parties. Suppose a patient wants to keep privacy on his health records except to the attending doctors. If the policy of the doctor's medical centre allows other research institute to access the patient records, then the patient's wishes might not be respected. It is important to use access policies to control the way in which information or services are propagated between organizations.

Intuitively, the concept of 'access policy consistency' means that (for the same service) the access policies of different organizations are conflict free. And organizations are able to collaborate in the intended way securely in terms of access control policies.

Access control issues in single organisations or single domains have been well studied (Sirer & Wang 2002, Kagal et al. 2004, Bhatti et al. 2004, Srivatsa et al. 2007); however, access control in a collaborative environment has just started to attract the attention of the research community (Rouached & Godart 2007, Yau & Chen 2008), and little attention has been given to consistency study between access control policies of different collaboration participants, particularly in the context of Web Services.

Our previous analysis shows that there are different ways of collaborating, and that each imposes dif-

ferent consistency constraints on the access policies of prospective partners (He & Yang 2007). Understanding the different requirements on collaboration partners' authorization policies for different types of collaboration is important, and access control in collaboration should take individual organization's access policies into account, as well as the type of the collaboration.

Our previous work (He & Yang 2007) gave a model of access control policy and showed what sorts of inconsistencies occur between policies and how these inconsistencies affect the various patterns of collaboration. In this paper we extend those ideas by showing how Description Logic can be used to formally model policies and how a Description Logic reasoner (an automated proof engine) can analyse the inconsistencies in policies and prove if the prospective partners could collaborate securly in terms of access control policy.

Description logics are weak fragments of first-order logic. The compensation for their limited expressively is decidability, making them an option when automated proof is required. The W3C has produced the Logic OWL-DL as a Description Logic standard for the so called Semantic Web (*Web Ontology Language (OWL)* 2004). OWL-DL is based on $\mathcal{SHOIN}$, and the logic $\mathcal{SROIQ}$ (Horrocks et al. 2006) is proposed as the logic for the next OWL standard (*OWL 1.1 Web Ontology Language* 2006). We use $\mathcal{SROIQ}$ here because it is expressive for a description logic and, though not yet a W3C OWL standard, it is supported by the Protégé[1] editor and the pellet[2] (Sirin et al. 2007) reasoner.

We first develop a model of access control policies in $\mathcal{SROIQ}$ and then show how two policies can be evaluated by comparing them in the model. We encode the inconsistency tests as concepts and relations in our access control model. Individual policies expressed using the model can then be compared and tested. Given two set of policies, with the roles and privileges of the two organisations suitably related, a reasoner will prove that the tests are either satisfiable or unsatisfiable and these results can be analysed to check whether they satisfy the requirements for the particular collaboration. Since the tests are part of the general model they are generic, meaning they can be expressed once, proven to encode the required meaning and used to testing any two policies. Because Description Logics are decidable, a reasoner for $\mathcal{SROIQ}$ will always terminate with the proofs, no matter how complex the policies.

The rest of paper is organized as follows. In the preliminaries section (Section 2), We first discuss different collaboration patterns in Section 2.1 and review the syntax and semantics of $\mathcal{SROIQ}$ in Section 2.2. In Section 3, we propose a description logic model for access control policy. The inconsistency tests are defined and proved in Section 4. In Section 5, we analyze requirements for collaborative policy (5.1) and discuss and example (5.2). Related work is discussed in Section 6, and concluding remarks and outline of our future research directions are presented in Section 7.

2 Preliminaries

Before we propose a model for access control policy and conduct an analysis of the inconsistencies, we give a brief introduction to collaboration patterns and description logic.

[1]Protégé 4.0 beta is available at `http://protege.stanford.edu/`

[2]Pellet is available at `http://pellet.owldl.com/`

2.1 Collaboration Patterns

Cross-organization collaborations consist of complex relationships and interactions among organizations. Some organizations collaborate with others through an agent; some organizations might play a role of 'middleman' that pass one organization's service to another. We have concluded several different types of basic collaboration between organizations in our previous work (He & Yang 2007). Some of the identified patterns are presented in Figure 1. Here, we introduce a number of patterns, but focus on one of collaboration pattern in this paper: Service Propagation (SP).

- **Composite Services**. The Composite Service we discuss here refers to the service that is based on the integration of multiple service providers. Two different cases are identified in service composition:
  1. **Composite service with agent** (**CSWA**): Multiple numbers of service providers provide their services through an centralized agent. For example in Figure 1, Health Insurance Company normally works with different medical service providers. Each party has its own security policy. The insurance company could have several service providers for same type of service and it works as an agent, and customers can only access those services through the network of the insurance company.
  2. **Joined service without an agent** (**JSOA**): Two organizations involving in a peer-to-peer collaboration and provide a joined service by integrating their business processes or part of their processes together to form a new service directly without any agent. In Figure 1, heart disease specialist clinic collaborates with a community service center to provide a post-treatment care plan to elders, they integrate part of their services directly without any agent.
- **Service Propagation** (**SP**): it depicts collaborations that involving multiple organizations and 'forward' privilege could be passed from one organization to another organization.

Service Propagation (**SP**): Service Propagation we discuss here refers to collaborations involving privilege propagation. As illustrated in Figure 1, patient is the owner of patient records who has the ultimate right regards to her/his own record. Patient can grant access right, i.e., access and forward privileges to her/his General Practitioner (GP) in a medical center so that the GP can access the patient's health records for diagnosis purpose. If the 'forward' privilege is granted to the GP, the GP could forward this access right to a third party, e.g. staff in hospital emergency room in Figure 1 example. In this scenario, patient is service owner and GP is the collaborative partner and service propagator who could grant access right to third parties.

The evaluation in the above example three parties are involved, comparison and evaluation has to be carried out twice:

1. The first one is between the patient and GP's clinic. The service owner – the patient will be responsible for the first evaluation to find the right partner whose authorization policy does not violate patient policy.

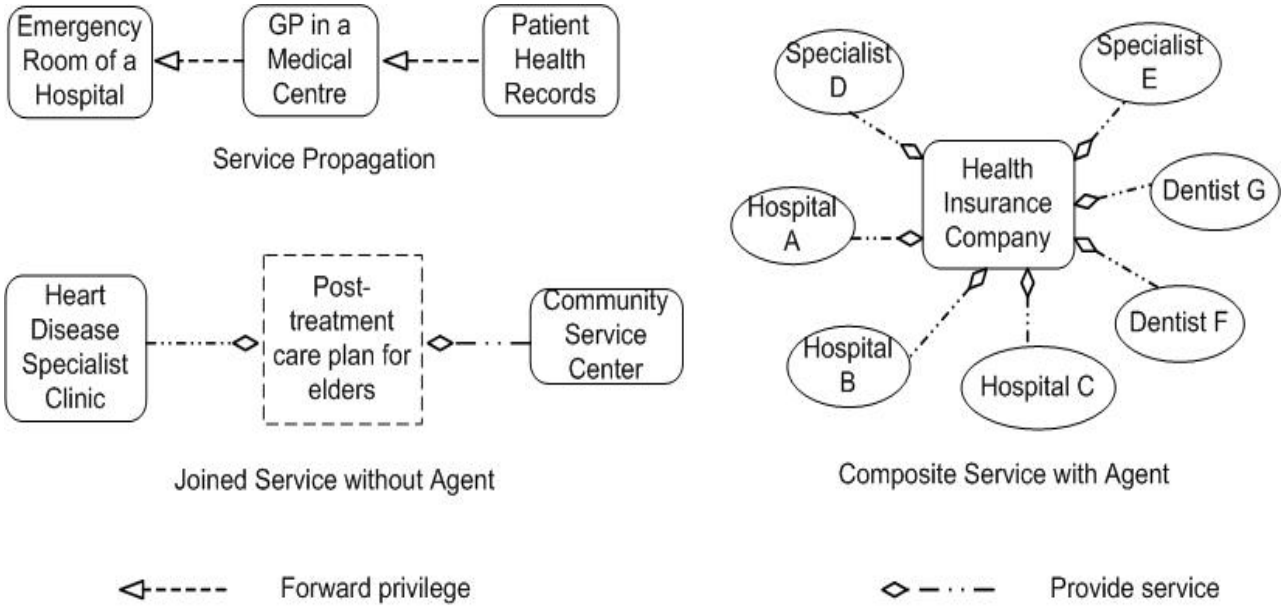

Figure 1: Cross-organization collaboration Patterns

2. The second one is between the clinic and the emergency room. The GP's clinic will be responsible for the second evaluation to find the right partner whose policy is not conflict with the patient's policy and policy of the GP's clinic.

2.2 Description Logic

In Description Logic (DL) the objects of interest (the domain of discourse) is modelled using axioms about concepts, roles and individuals. In $\mathcal{SROIQ}$ these axioms are stated in the TBox, the RBox and the ABox respectively. Figure 2 accompanies the following paragraphs on the syntax and semantics of $\mathcal{SROIQ}$ as used in this paper — Horrocks et al. (2006) give the full definition of $\mathcal{SROIQ}$. In the following, C and D range over concepts, with A for atomic concepts; S, R and subscripted versions of these range over roles; and x and y range over individuals.

Concepts are defined in the TBox, starting with atomic concepts and building more complex definitions from these. All concepts are subsumed by the universal concept $\top$, called *Thing* in OWL. The complement, union and intersection of concepts are used to define concepts in terms of others, and hierarchies of concepts are constructed using concept inclusion. Concepts can be specified as disjoint from other concepts. Existential and universal restrictions specify concepts in terms of the relationship to other concepts through roles.

Axioms defining roles are specified in the $\mathcal{SROIQ}$ RBox. DL roles, binary predicates on individuals, show the relationships between individuals. Roles can be constructed into hierarchies, composed (composition of relations) and specified as inverses of other roles. Further, roles can be stated as transitive, symmetric, reflexive or functional (single-valued).

Individuals represent members of the domain. Assertions about the existence of individuals, their classification into concepts and their participation in roles are made in the ABox.

A $\mathcal{SROIQ}$ knowledge base $\mathcal{K}$ comprises a TBox, an RBox and an ABox. The semantics of a knowledge base is given by an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ that specifies a set of objects for the domain $\Delta^{\mathcal{I}}$ and a function $\cdot^{\mathcal{I}}$ that maps each individual to an object in $\Delta^{\mathcal{I}}$, each concept to subset of $\Delta^{\mathcal{I}}$ and each role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a model of $\mathcal{K}$, written $\mathcal{I} \models \mathcal{K}$, if it is consistent with the axioms in the TBox, RBox and ABox. A concept C in the TBox of a knowledge base $\mathcal{K}$ is satisfiable if $\mathcal{I} \models \mathcal{K}$ for some interpretation $\mathcal{I}$ where $C^{\mathcal{I}} \neq \emptyset$. A concept C subsumes a concept D if $C \sqsubseteq D$ is implied by the knowledge base: equivalently, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in all models.

Since the subsumption and satisfiablilty problems in $\mathcal{SROIQ}$ are decidable (they are also mutually reducible) a $\mathcal{SROIQ}$ reasoner can always determine the satisfiability of a concept. In this work we are mainly interested in satisfiability checking, with subsumption of minor importance. We use Protégé 4.0 beta to edit our $\mathcal{SROIQ}$ definitions and pellet to prove satisfiability.

The DL in this paper follows the convention of concepts beginning with a capital letter and roles with a lowercase letter. For simplicity we write

$$role \ : \ (ConceptA \times ConceptB)$$

to indicate that *role* is a DL role with domain *ConceptA* and range *ConceptB*. The definition of a role may be superscripted with $\mathcal{F}$ to indicate a functional role, $\mathcal{T}$ for transitive, $\mathcal{S}$ for reflective and $\mathcal{R}$ for reflexive. That instance a is classified into concept C is written $C(a)$. DL expressions are written in the so-called German DL Syntax.

3 Model for Access Control

We base our policy model on the core definition of Sandhu et al.'s (1996) Role-Based Access Control (RBAC), now a NIST standard (*ANSI INCITS 359-2004* 2004). RBAC models access control in terms of roles, job functions, and the permissions assigned to those roles. We remove notions of sessions and users from Core RBAC, sessions because they represent the dynamic rather than structural elements of access control, and in place of users we add the credentials required to access a role. The role hierarchies of Hierarchical RBAC are not captured by our current model, but we plan to investigate role hierarchies in future work. The purpose of the model is to provide a

Constructor name	Syntax	Semantics
universal	$\top$	$\Delta^{\mathcal{I}}$
empty	$\bot$	$\emptyset$
atomic concept	A	$A^{\mathcal{I}}$
concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
concept intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
concept union	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
concept inclusion	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
concept equivalence	$C \equiv D$	$C^{\mathcal{I}} \equiv D^{\mathcal{I}}$
existential restriction	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}}.(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
universal restriction	$\forall R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y \in \Delta^{\mathcal{I}}.(x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
role inclusion	$S \sqsubseteq R$	$S^{\mathcal{I}} \subseteq R^{\mathcal{I}}$
role inverse	R^{-}	$\{(y,x) \mid (x,y) \in R^{\mathcal{I}}\}$
role composition	$S_1 \circ \ldots \circ S_n \sqsubseteq R$	$S_1^{\mathcal{I}} \circ \ldots \circ S_n^{\mathcal{I}} \subseteq R^{\mathcal{I}}$

Figure 2: The constructors and semantics (for an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$) of $\mathcal{SHOIN}$

formal framework that is representative, though not prescriptive, of policy and highlights issues relevant to collaboration.

Each service in an organisation has a policy describing the privileges assigned to the roles that may access the service and, because the processing for any service is the same, we describe the model without referring further to the policies of different services in the same organisation.

Figure 3 depicts the policy model. This paragraph describes the model in general terms and then the remainder of the section describes the DL encoding of the model. The main entities in the model are roles, credentials, privileges, obligations and provisions. The users of the service are not defined and may be, for example, humans or computer agents. Users are not statically assigned to roles, rather credentials are used to authenticate the holder as authorised to act in a particular role. Each role is assigned a number of privileges, which, for example, may be the right to edit files, access resources or release information. Obligation and provision conditions are attached to each assignment of a privilege to a role. Obligations are conditions that must be satisfied after the privilege is accessed, while provisions are conditions that must be satisfied before the privilege is accessed: hence, provisions further constrain the right to access a privilege, such as restricting the time a privilege is accessed, while obligations enforce what must be done after the privilege is accessed, such as writing to a log.

There is an unfortunate name clash between DL roles, which are binary relations, and RBAC roles, which are units of authorisation in RBAC and individuals in the concept *Role* in the policy model. Here, either the context will make clear what is meant or the meaning is explicitly stated; often relation is used in place of DL role.

In the DL encoding, the concept *Role* models the set of roles, individuals from the concept *Credentials* model the set of credentials required by a role and the relation *requires* links a role to the credentials required to authenticate in that role.

$$\begin{aligned} \mathit{Role} &\sqsubseteq \exists \mathit{requires}.\mathit{Credentials} \\ \mathit{requires}^{\mathcal{F}} &: (\mathit{Role} \times \mathit{Credentials}) \\ \mathit{satisfies} &: (\mathit{Credentials} \times \mathit{Credentials}) \end{aligned} \tag{1}$$

The definition of *Role* ensures that every role is linked through the *requires* relation to a set of credentials. Since *requires* is defined as functional (single-valued), each role is linked to only one set of credentials.

The structure of credentials are not further modelled here; though credentials were considered more deeply in the original work (He & Yang 2007). The model assumes a preordering, *satisfies*, on *Credentials* that, in the absence of further structure, shows when one set of credentials would also satisfy the requirements of another set. For example, credentials requiring a password and a digital certification would also satisfy credentials only requiring a certificate.

The only relations directly supported in a DL are binary roles, hence the ternary relation between privileges, obligations and provisions in the policy model needs to be represented by a concept. The concept *PrivilegeAssignment* represents the ternary relation and relations *privilege*, *obligation* and *provision* link a *PrivilegeAssignment* to its associated *Privilege*, *Obligation* and *Provision*.

$$\begin{aligned} &\mathit{PrivilegeAssignment} \sqsubseteq \exists \mathit{privilege}.\mathit{Privilege} \\ &\mathit{PrivilegeAssignment} \sqsubseteq \exists \mathit{obligation}.\mathit{Obligation} \\ &\mathit{PrivilegeAssignment} \sqsubseteq \exists \mathit{provision}.\mathit{Provision} \\ &\mathit{privilege}^{\mathcal{F}} : (\mathit{PrivilegeAssignment} \times \mathit{Privilege}) \\ &\mathit{provision}^{\mathcal{F}} : (\mathit{PrivilegeAssignment} \times \mathit{Provision}) \\ &\mathit{obligation}^{\mathcal{F}} : (\mathit{PrivilegeAssignment} \times \mathit{Obligation}) \end{aligned} \tag{2}$$

*Obligation*s and *Provision* are themselves subconcepts of *PrivilegeCondition*.

$$\begin{aligned} \mathit{Obligation} &\sqsubseteq \mathit{PrivilegeCondition} \\ \mathit{Provision} &\sqsubseteq \mathit{PrivilegeCondition} \end{aligned} \tag{3}$$

As with credentials, the exact structure of privileges, obligations and provisions are not important for this paper, and again assume an ordering on obligations and provisions.

$$\begin{aligned} \mathit{obl\_order} &: (\mathit{Obligation} \times \mathit{Obligation}) \\ \mathit{prov\_order} &: (\mathit{Provision} \times \mathit{Provision}) \end{aligned} \tag{4}$$

The intent is the same as for *satisfies*: that is, if $(o_1, o_2) \in \mathit{obl\_order}$ then o_1 is a stronger obligation than o_2, and similarly for *prov_order*. That $(c_1, c_2) \in \mathit{satisfies}$ (and similarly for *obl_order* and *prov_order*) may be read as if it were written c_1 *implies* c_2 because, for the purposes of this paper, the logical meaning of, and logical relation between, credentials, obligations and provisions is more important than any structure these elements of a policy might have.

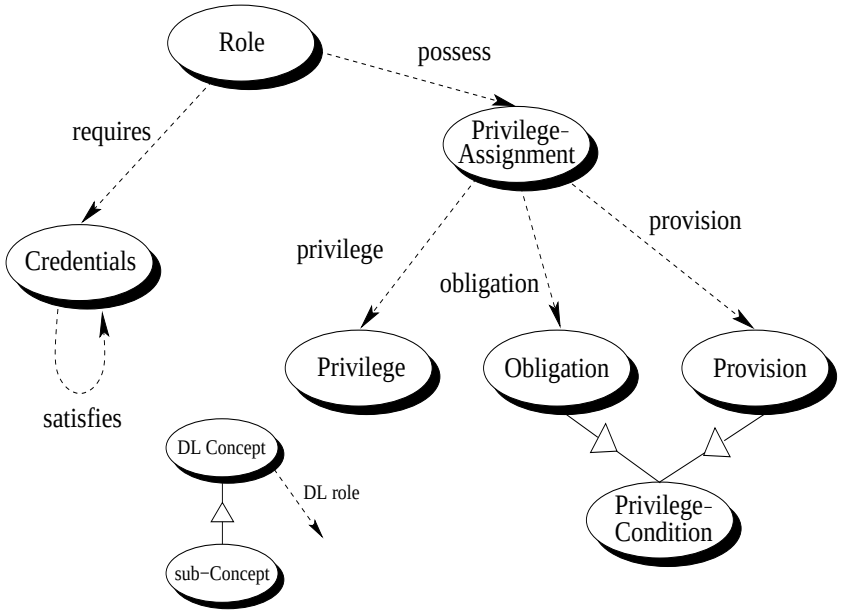

Figure 3: Policy Model (relations *privilege*, *obligation* and *provision* not shown)

A role may posses many privileges, but a particular privilege assignment can be made to only one role.

$$\begin{aligned} &possesses \ : \ (Role \times PrivilegeAssignment) \\ &possesses^{-\mathcal{F}} \end{aligned} \tag{5}$$

Note that (5) merely enforces that any particular instance of a privilege assignment cannot be allocated to multiple roles; it does not preclude exactly the same privilege and conditions being allocated to multiple roles; (5), however, enforces that if the same privilege and conditions are allocated to multiple roles, a different *PrivilegeAssignment* must be used in each case. The purpose of the restriction is to ensure that the role allocated to any particular privilege assignment can be determined (this is necessary for example in Section 4.3.1).

4 Classification of Inconsistencies

Before discussing role, credential and privilege inconsistencies, the three types of inconsistencies discussed in this paper, this section further outlines our model for consistency checking.

Assume two organisations, A and B, requiring a collaboration, where both organisations have an access control policy, P_A for A and P_B for B, encoded in the model defined in the previous section (or defined in some other way and translated into the model). For consistency and collaboration checking, P_A and P_B must first be combined into a single model and then checked. The policy model from the previous section is extended by allowing the encoding and checking for P_A and P_B.

First, the model is extended for the roles in P_A and P_B.

$$\begin{aligned} Role_A &\sqsubseteq Role \\ Role_B &\sqsubseteq Role \\ Role_A &\text{ disjoint with } Role_B \end{aligned} \tag{6}$$

Next, a comparability relation between the roles in the two organisations is defined.

$$\begin{aligned} role\_comp_{AB} \ &: \ (Role_A \times Role_B) \\ role\_comp_{BA} \ &: \ (Role_B \times Role_A) \end{aligned} \tag{7}$$

The comparability relations in (7), the following relations in (8) and extensions to *obligation*, *provision* and *privilege*, record the notion that to meaningfully compare the policies P_A and P_B some basis for this comparison is required. Either human experts from A and B, metadata or some automated system is required to specify which roles in $Role_A$ relate to which roles in $Role_B$ and what privileges are equivalent; while this process is important for checking two policies, it is not dealt with in description logic and for the purposes of this paper it is simply assumed that that the relations are supplied along with P_A and P_B.

The relations $role\_comp_{AB}$ and $role\_comp_{BA}$ reflect the notion that for some collaborations the correspondence between roles will almost be an equivalence but, in other cases, a single role may be related to many roles in the other organisation: for example, doctor in one organisation might relate to both physician and specialist in the other.

A similar notion to the role comparisons, though in this case an equivalence, is introduced for privileges.

$$\begin{aligned} Privilege_A &\sqsubseteq Privilege \\ Privilege_B &\sqsubseteq Privilege \\ Privilege_A &\text{ disjoint with } Privilege_B \\ priv\_equiv_{AB} \ &: \ (Privilege_A \times Privilege_B) \\ priv\_equiv_{BA} \ &: \ (Privilege_B \times Privilege_A) \\ priv\_equiv_{BA} &\equiv priv\_equiv_{AB}{}^{-} \end{aligned} \tag{8}$$

Similar definitions to, say, (8) could be added for relating credentials, obligations and provisions between P_A and P_B; however, it is more convenient to combine the two policies using the ordering relations already present in the model. In the combined model, *obligation*, *provision* and *privilege* are orderings over the obligations, provisions and privileges of both organisations.

Further to these definitions, parts of the model must be closed. DLs work under the open-world assumption, meaning that a DL knowledge base defines

a domain in terms of known concepts, roles and individuals but, also, that the definition is open and anything not explicitly stated, but consistent with the given definitions, may also be present in the domain. A closed-world assumption, on the other hand, takes the definitions as all there is to know about the domain.

For the purposes of this paper, the open-world assumption means that a DL reasoner may add to the policies and, in particular, further roles or comparabilites could be added. Parts of a knowledge base are closed by stating that the given concepts or instances are really all there is. For example, $Role_A \equiv \{a, b, c\}$ closes $Role_A$ by stating that the roles a, b and c are the only inhabitants. It is then necessary to specify that these inhabitants are distinct — essentially, adding a unique name assumption. Closing a relation r means explicitly stating that the set of (a, b) tuples specified for r are the only tuples in r. For the tests given in the remainder of this section, $Role_A$, $Role_B$, $possesses$, $role\_comp_{AB}$, $role\_comp_{BA}$, $priv\_equiv_{AB}$ and $priv\_equiv_{BA}$ all need to be closed. For the remaining concepts and relations, the tests take advantage of open-world reasoning and so closure is not necessary.

The following shows how to investigate inconsistencies between P_A and P_B, given that they are in a combined model that has been suitably closed.

4.1 Role Inconsistencies

Role inconsistencies between P_A and P_B are present when there are roles in one that have no comparable roles in the other. If P_A has roles with no equivalent in P_B, then this inconsistency indicates that, for at least those roles, P_A admits privileges that P_B does not and, hence, that P_A might allow some users privileges that P_B would not allow users with similar credentials.

The Missing Role Node concept, MRN_A, represents the roles in $Role_A$ that have no comparable role in $Role_B$. The concept B_comp is defined to be all the roles in $Role_A$ that have some comparable role in $Role_B$. $Role_A$ is defined such that all roles in organisation A must be classified as in MRN_A or B_comp.

$$\begin{aligned} MRN_A &\sqsubseteq Role_A \\ B\_comp &\equiv Role_A \sqcap \exists role\_comp_{AB}.Role_B \\ MRN_A &\text{ disjoint with } B\_comp \\ Role_A &\sqsubseteq B\_comp \sqcup MRN_A \end{aligned} \tag{9}$$

The combined effect of (9) is that all roles in $Role_A$ are classified into either MRN_A or B_comp and, if and only if all roles in $Role_A$ have some comparable role, MRN_A is inconsistent, otherwise some roles in $Role_A$ do not have comparable roles in $Role_B$ and these are classified in MRN_A. A similar definition to (9) is made for MRN_B, the roles in $Role_B$ with no comparable role in $Role_A$.

4.2 Credential Inconsistencies

There is an inconsistency in credentials when roles judged as comparable have different credential requirements. Such an inconsistency means that the two organisations have different requirements on what needs to be established before the privileges associated with a role can be accessed and could mean that equivalent roles in the two organisations have access to similar privileges but with a less stringent authorisation requirement in one organisation.

The goal is to establish if, for comparable roles, P_A has more stringent authorisation requirements than P_B (or similarly from P_B to P_A). The model has a relation for comparable roles and an ordering on credentials, but DL isn't expressive enough to directly encode the required property. In general, it isn't possible to directly express a property of the form"if a and b are related by r and a and b are related by r' and r'' to c_a and c_b respectively, are c_a and c_b related via r'''?" With the open-world assumption the required property can be expressed indirectly. Two things are required to express such a property in a DL: the first is the pattern in Figure 4, and the second is a technique often used with satisfiability which is to express properties negatively rather than positively.

Given the relations

$$\begin{aligned} r_1 &: (C_1 \times C_3) \\ r_2 &: (C_2 \times C_4) \\ r_3 &: (C_3 \times C_4) \end{aligned} \tag{11}$$

A new relation r relating individuals in C_1 and C_2 only if they are related through $r_1 \circ r_3 \circ {r_2}^-$ is defined by

$${r_1}^- \circ r \circ r_2 \sqsubseteq r_3 \tag{12}$$

and by ensuring that

$$\begin{aligned} Domain(r) &\sqsubseteq C_1 \sqcap \exists r_1.C_3 \\ Range(r) &\sqsubseteq C_2 \sqcap \exists r_2.C_4 \end{aligned} \tag{13}$$

(12) and (13) ensure that r has only the right pairs. The correctness of this definition is shown by the following proof.

$$\begin{gathered} (c_1, c_2) \in r \\ \rightarrow \\ \exists c_3 c_4.(c_1, c_3) \in r_1 \wedge (c_2, c_4) \in r_2 \wedge (c_3, c_4) \in r_3 \end{gathered}$$

Proof. The domain and range restrictions in (13) are required to show the existence of a c_3 and c_4 from $(c_1, c_2) \in r$ and (12) is used to finish the proof. □

While this scheme does project a relation, it may not capture the intended meaning if r_1 and r_2 are not functional. If r_1 and r_2 are not functional, (12) has the potentially unwanted side-effect of adding extra pairs to r_3, and there also isn't enough information in r to represent the multiple ways individuals in C_1 and C_2 might relate through the chain $r_1 \circ r_3 \circ {r_2}^-$.

Figure 4: **Pattern:** Projecting a Relation

First, define a relation disjoint with *satisfies*.

$$\begin{aligned} &not\_satisfies : (Credentials \times Credentials) \\ &not\_satisfies \text{ disjoint with } satisfies \end{aligned} \tag{10}$$

The relation *not_satisfies* is not the inverse of *satisfies*, it may not even relate everything that isn't related by *satisfies*, instead the open-world assumption means it can be any relation on *Credentials* that shares no tuples with *satisfies*. A DL reasoner is free to choose any relation satisfying this constraint in an attempt to satisfy the definitions below.

Next, the pattern in Figure 4 is used to project the *not_satisfies* relation between *Credentials* to a relation $nsat_{AB}$ between $Role_A$ and $Role_B$. In the pattern replace C_1 by $Role_A$, C_2 by $Role_B$, C_3 and C_4 by *Credentials*, r_1 and r_2 by *requires*, r_3 by *not_satisfies*

and r by $nsat_{AB}$, giving definition (14). Since *requires* is functional, the projection relates roles if and only if their credentials are related by *not_satisfies*. Note that the reasoner will infer further restrictions on the stated types for relations such as $nsat_{AB}$ because of the restrictions on the other relations in (14). In fact, stating a domain and range for $nsat_{AB}$ isn't necessary at all; however, for clarity, the most comprehensive definitions of most concepts and roles are used in this paper.

$$\begin{aligned} nsat_{AB} &: (Role_A \times Role_B) \\ requires^- \circ nsat_{AB} \circ requires &\sqsubseteq not\_satisfies \end{aligned} \tag{14}$$

Lastly, $role\_comp_{AB}$ and $nsat_{AB}$ are combined. The requirement is for the combination, $role\_comp_{AB}\_nsat$, to relate two roles r_A and r_B if and only if $(r_A, r_B) \in role\_comp_{AB}$ and $(r_A, r_B) \in nsat\_role$, which is achieved by forming the intersection of $role\_comp_{AB}$ and $role\_comp_AB$.

$$\begin{aligned} role\_comp_{AB}\_nsat &: (Role_A \times Role_B) \\ role\_comp_{AB}\_nsat &\sqsubseteq role\_comp_{AB} \\ role\_comp_{AB}\_nsat &\sqsubseteq nsat_{AB} \end{aligned} \tag{15}$$

Two roles r_A and r_B are thus related by $role\_comp_AB\_nsat$ exactly when r_A and r_B are comparable and the credentials required for r_A are not more stringent than those required for r_B.

To prove that the required meaning is captured by $role\_comp_{AB}\_nsat$ requires showing (in the underlying semantics)

$$\begin{aligned} (r_A, r_B) \in role\_comp_{AB}\_nsat \quad &\rightarrow \\ (r_A, r_B) \in role\_comp_{AB} &\wedge \\ \exists c_A, c_B . (r_A, c_A) \in requires &\wedge \\ (r_B, c_B) \in requires \wedge (c_A, c_B) &\notin satisfies \end{aligned}$$

the proof of which follows easily from the proof in Figure 4 and the definitions of $role\_comp_{AB}\_nsat$ and $nsat_{AB}$.

A DL reasoner will classify $role\_comp_{AB}\_nsat$ as unsatisfiable when all roles in P_A have stricter credential requirements than the corresponding roles in P_B: that is, the credential requirements for each role in $Role_A$ also satisfies (via *satisfies*) the credential requirements for the corresponding roles in $Role_B$. If this is not the case, a DL reasoner will classify pairs of roles into $role\_comp_{AB}\_nsat$ where the credential requirements for the role from $Role_A$ do not also satisfy the requirements for the roles from $Role_B$.

4.3 Inconsistencies in Privileges

He & Yang (2007) discussed three inconsistencies for privileges: inconsistencies in the privileges allocated to comparable roles and two types of inconsistencies in the conditions associated with comparable privileges for comparable roles. The four tests to determine the presence of these inconsistencies given by He & Yang are fine-grained enough to pinpoint an inconsistency between comparable roles and the privilege causing the inconsistency. However, the combined effect of the four tests is, essentially, equivalent to determining if comparable roles in the two collaborating parties have equivalent privileges and, if for equivalent roles and equivalent privileges, one party has weaker conditions than the other.

Here, only a test to determine if equivalent roles have comparable conditions for equivalent privileges is discussed. The roles and privileges causing an inconsistency can still be recovered.

The *Pair* concept (16) defines a pair (a, b) as an individual p, with $Pair(p)$, $(p, a) \in left$ and $(p, b) \in right$.

$$\begin{aligned} right^{\mathcal{F}} &: (Pair \times Thing) \\ left^{\mathcal{F}} &: (Pair \times Thing) \\ Pair &\sqsubseteq \exists left . Thing \\ Pair &\sqsubseteq \exists right . Thing \end{aligned} \tag{16}$$

Figure 5: **Pattern:** Pairs

4.3.1 Different Conditions

If the conditions on equivalent privileges allocated to comparable roles are different, then these roles in the two organisations access a privilege with different, perhaps incompatible, restrictions. However, if the difference is one were the conditions can be meaningfully compared, then the comparison will reveal that one role accesses the privilege under stronger restrictions than the other.

The situation is similar to credentials, in that the test ultimately relies on the orderings between conditions; however, the relations between both privileges and roles also need to be taken into account. Further, the credential checking relied on the functional relation *requires*; the projection pattern used with *requires* is not applicable for *possesses* because *possesses* is not functional. For non-functional relations a projection onto a single relation can't represent all the possible relationships: for example, each role may relate through *possesses* to a number of *PrivilegeAssignment*s, so a single relation between *Role*s can't capture the many possible relationships to privileges and their associated conditions. However, the same general principle can still be used if *possesses* is first converted to a concept.

*Pair*s (Figure 5) can be a more convenient representation of a relation than the corresponding DL role because the expressions allowed on roles and concepts are different. In this case, using the pairs-for-relations pattern (Figure 6) for *possesses* introduces functional relations *left* and *right* and, thus, the relations between roles, privileges and conditions can be projected as a relation between *Pair*s of *Role*s and *PrivilegeAssignment*s.

A relation, $pa\_eq\_nord_{AB}$, that relates *PrivilegeAssignment*s from P_A and P_B when the *Privilege*s are related, but the conditions are not, is defined by

- projecting both *obl_order* and *prov_order* to a $(PrivilegeAssignment \times PrivilegeAssignment)$ relation (see Figure 4 and 14)
- forming a subrelation of both the above two relations (as was done in (15)) and then defining $not\_prov\_obl\_order_{AB}$ as a relation disjoint from this (see also (10))
- projecting $priv\_equiv_{AB}$ to a relation between *PrivilegeAssignment*s (Figure 4) and defining $pa\_eq\_nord_{AB}$ as a subrelation of this and $not\_prov\_obl\_order_{AB}$ (again, see (15)).

Pairs $Pair\_possess\_A$ and $Pair\_possess\_B$ are defined (using the pattern in Figure 6) as pairs representing the *possesses* relations for P_A and P_B.

Relations $role\_comp_{AB}$ relates comparable roles in $Role_A$ and $Role_A$, $pa\_eq\_nord_{AB}$ relates *PrivilegeAssignment*s and the specialisations of *left* and *right* on $Pair\_possess\_A$ and $Pair\_possess\_B$ are functional. The projection pattern (Figure 4)

Pairs, as defined in (16) (Figure 5), can be used to represent DL roles. DL roles (binary relations) are simply sets of (a, b) pairs; hence, if *left* and *right* from *Pair* are used to represent the domain and range, a set of *Pair* individuals can represent the information in a role. The same principle applies for *PrivilegeAssignment* (2), which represents a ternary relation. A *Pair* representation of a relation can even be derived from an existing relation. Given

$$r \ : \ (C_1 \times C_2) \tag{17}$$

a sub-relation can be derived.

$$\begin{array}{l} Pair\_r \sqsubseteq Pair \\ \quad left\_r \sqsubseteq left, right\_r \sqsubseteq right \\ Pair\_r \sqsubseteq \exists left\_r.C_1, Pair\_r \sqsubseteq \exists right\_r.C_2 \end{array} \tag{18}$$

$$left\_r^- \circ right\_r \sqsubseteq r \tag{19}$$

The proof that the relation represented by the pair is contained in r follows easily from definition (18), which makes $Pair\_r$ a type of pairs of C_1 and C_2 individuals, and definition (19), which ensures that if $(p, c_1) \in left\_r$ and $(p, c_2) \in right\_r$, with $Pair\_r(p)$, then $(c_1, c_2) \in r$.

Figure 6: **Pattern:** Pairs for Relations

can thus be used to project $role\_comp_{AB}$ and $pa\_eq\_nord_{AB}$ to relations between $Pair\_possess\_A$ and $Pair\_possess\_B$. Finally, these two relations are combined (as was done in (15)) to produce $rcomp\_pa\_nord_{AB}$, which is unsatisfiable if the privileges for comparable roles have more stringent constraints in P_A than in P_B and contains *Role-PrivilegeAssignment* pairs with uncomparable conditions otherwise.

5 Access Control Policy Requirements and Example

This section first examines the inconsistencies to consider for the Service Propagation pattern and then discusses and example of using a Description Logic reasoner to analyse policies for collaboration.

5.1 Access Control Policy Requirements

Several cross-organization collaboration patterns were identified previously (He & Yang 2007), and some of these were reviewed in Section 2. The different collaboration patterns result in different consistency requirements on the policies of the prospective collaboration partners. The inconsistencies between policies can result in collaboration being accepted, rejected or can require negotiation to remove inconsistencies. This paper focuses on Service Propagation and on inconsistencies that can be automatically accepted or rejected.

Service Propagation implies a 'forwarding' behaviour from the service owner to the collaborative partner. Unwanted 'forwarding' could happen if the partner has authorization policy that is less restrictive than the owner's. Therefore, policy of collaboration partner should be more restrictive than the service owner's. There are three typical ways to have one policy looser than the other:

- One set of policies has more roles to access the same service than the other set of policies;
- One set of policies has less credentials required for an equivalent role than the other set;
- For an equivalent role, more privileges, or privileges with weaker conditions, are assigned in one set of policies than the other set.

Based on above principles, the following defines the requirements on collaboration partners for Service Propagation. The requirements are in terms of the definitions in the previous section the following assumes organisation A is the service owner and B is the collaborator.

1. Every role defined by the partner should have a a comparable role in the service owner. In terms of Section 4.1, concept MRN_B (not given in the text, but similar to MRN_A) should be unsatisfiable;
2. The credentials required for equivalent roles should be be more restrictive in the partner: $role\_comp_{BA}\_nsat$ should be unsatisfiable;
3. More privileges should be granted to roles in P_A than their equivalents in P_B (not discussed in previous section);
4. Inconsistencies in conditions are generally negotiable, but if the collaboration partner has stricter conditions, then the collaboration is acceptable: that is, $rcomp\_pa\_nord_{BA}$ unsatisfiable.

5.2 Example

The following example demonstrates how an access control policy can be presented in the DL policy model and how the inconsistencies between the access control policies of collaboration partners can be evaluated using a DL reasoner.

If services and their policies are listed in a repository, a service searching for a collaboration partner can search the repository for a service with the required functionality and then check if the access control policies are compatible. Assume a medical centre that requires a collaboration with a pathology centre. The two will collaborate on the tests of patients, their records and the results of tests. The centre will search for a medical service registry for potential pathology collaborators and test the access control policies to find a suitable partner. The policies of the medical centre and two potential collaborators are shown below.

1. Medical Clinic:
   - Attending doctors have the privilege to access and forward patient information;
   - a provision is attach to the forward privilege: receipients must be a doctor in the chosen pathology institute.

   A policy must be defined with individuals classified as follows: $Role(mc\_doctor)$, $Credentials(mc\_doctor\_id)$, $PrivilegeAssignment(mc\_pa)$, $Privilege(access)$, $Privilege(mc\_forward)$, $Provision(to\_partner\_pathology\_doctor)$ and $Obligation(no\_obligation)$. With $(mc\_doctor, mc\_doctor\_id) \in requires$, $(mc\_doctor, mc\_pa) \in possess$, $(mc\_pa, mc\_forward) \in privilege$, $(mc\_pa, no\_obligation) \in obligation$, $(mc\_pa, to\_partner\_pathology\_doctor) \in provision$ and a privilege assignment for $access$.

2. Pathology institute X:

   - Attending doctors have privilege forward patient information;
   - a provision attach to forward privilege: receipients must be doctors in X

   The policy definitions are as follows: $Role(doctor_X)$, $Credentials(doctor\_and\_path\_id)$, $PrivilegeAssignment(pa_X)$, $Privilege(forward_X)$, $Provision(to\_X\_pathology\_doctor)$ and $Obligation(no\_obligation_X)$. With $(doctor_X, doctor\_and\_path\_id) \in requires$, $(doctor_X, pa_X) \in possess$, $(pa_X, forward_X) \in privilege$, $(pa_X, no\_obligation_X) \in obligation$, $(pa_X, to\_X\_pathology\_doctor) \in provision$.

3. Pathology institute Y:

   - Attending doctors have privilege to forward patient information;
   - a provision is attach to forward privilege: receipient must be either a doctor in Y or staff in a collaborating research institute.

   The policy definitions are similar to above, with the important provision $Provision(to\_Y\_pathology\_doctor\_or\_research)$.

To find a suitable collaboration partner, the medical centre must test its policy with pathology institutes X and Y. The policies are tested in the combined model; one test with institute X and one with Y. Assume in both cases that the medical centre is policy A and that each pathology institute is policy B (in terms of A and B as discussed in the previous section). The relationships for $role\_comp_{AB}$ and others are straightforward, importantly provision $to\_X\_pathology\_doctor$ is more restrictive than $to\_partner\_pathology\_doctor$, hence $(to\_X\_pathology\_doctor, to\_partner\_pathology\_doctor) \in prov\_order$, while $to\_Y\_pathology\_doctor\_or\_research$ is less restrictive than the provision in the medical centre and so is not related by $prov\_order$.

In the comparison with institute X the reasoner proves the required concepts as unsatisfiable, and thus shows that Pathology institute X is a suitable partner for collaboration. However, in testing with institute Y, the reasoner proves that concept $rcomp\_pa\_nord_{BA}$ is satisfiable (because Pathology institute Y allows extra forwarding privileges), and thus shows that Pathology institute Y is not a suitable collaboration partner.

Patients could test if the policy of the medical centre satisfies their requirements and make their choice of suitable medical services based on these policy tests.

The models, tests and example in this paper have been encoded in $\mathcal{SROIQ}$, using Protégé, and the reasoning done with Pellet. The files are available on request.

6 Related Work

Research has been done in the area of access control / authorization control for web services. Most of the works concentrate on authorization control policy language specifications and a number of formal models have been developed (Sirer & Wang 2002, Kagal et al. 2004, Bhatti et al. 2004, Srivatsa et al. 2007). These studies provide insights on security constraints in single organization from different perspectives, which helped us to build up our authorization policy model. But these studies did not look at authorization issues in the context of cross-organization collaborations.

Policy issues in distributed systems have been actively growing over the years, Bonatti & Mogavero (2008) proposed an inclusion mappings based policy comparison method, which could be useful for rule-based policis, particularly for recursive rules. Wang et al. (2004) addressed security policy reconsiliation issues in distributed computing environments, and focus on security provisioning policy. The authors based their reconsiliation on structure of the policies, which is similar to our work. It also provides an alternative for policy specification. However, our focus in this paper is policy comparison and evaluation rather than reconsiliation and we address authorization policies.

A number of studies concentrated on authorization architecture (B.Carminati et al. 2006, Ziebermayr & Probst 2004). B.Carminati et al. (2006) suggested a brokered architecture to build composite Web services according to the specified security constraints. They used security matchmaker to find right collaboration partners who have compatible security policies, which is similar to our research. However, it did not address the issue that inconsistencies and conflicts exist between security policies of prospective partner. Nothing was mentioned about how security policies from different partners could be combined and how to solve the conflicts between these policies. Our work identified various types of inconsistencies between authorization policies and provided suggested solution. We believe some of inconsistencies are acceptable or negotiable for intended type of collaboration.

There are few papers on Web service authorization control in the collaborative environment. We are aware of the work presented by Rouached & Godart (2007), which presented a framework for managing authorization policies for Web service compositions. The proposed framework addressed authorization policy conflicts and provided methodology for conflicts detection. Yau & Chen (2008) proposed an approach to security policy integration and conflict reconciliation, which is relate to our research. The authors presented a similarity-based policy adaptation algorithms to adapt changes in collaborative groups and a negotiation-based protocol for conflict reconciliation But they neglected the fact that different types of collaboration affect the way the collaboration policy is developed as well as the requirements on collaborative partner's authorization policy. An evaluation on collaborative partner's access policy has to be carried out before the collaboration be established. Our work is to fill in this gap. We believe this is the first step toward conflicts detection and suitable collaboration partners discovery.

Description Logic (DL) has been used in differenct aspects of access control (Chae & Shiri 2007, Shields & Molloy 2007, Zhao et al. 2005, Muthaiyah & Kerschberg 2006). Shields & Molloy (2007) proposed an efficient solution to XML access control that use description logic and decidable rules to decide the permissions for an individual at the time they request information. Chae & Shiri (2007) demonstrated how to express the RBAC concept in object-oriented systems using description logic and how to use DL to make authorization decisions between the role hierarchy and the object hierachy. All of these studies are only focus on access control in single organization, access control in collaborative environment has not been well studied.

In summary, none of these studies went deep into different types of cross-organization collaboration, which could raise different requirements on access control policy of prospective collaborative part-

ners. None of studies provided tool for access policy comparison and evaluation. Our goal is to identify these different requirements for different types of cross-organization collaboration, to analysis inconsistencies between policies and provide suggestions and solutions to inconsistencies according the collaboration type, to propose an access control model and automatic prover that could assist organizations to discover compatible collaboration partners.

7 Conclusion/Discussion

Challenging security and access control issues arise in Web Services. In particular, when cross-organisation collaborations are formed, the access control policies of the collaboration partners must be inspected. If the policies are not compatible, the collaboration does not respect the access control policy of at least one of the partners.

In this paper we have presented a Description Logic model for access control policy. Based on this model, we have demonstrated how a Description Logic reasoner can be used to find the inconsistencies between the policies of potential collaborators. The reasoner constructs a proof demonstrating the consistency, or otherwise, of the policies by reasoning about tests that we added to the policy model. With formal descriptions of access control policy and different policy inconsistencicy types, we can evaluate prospective collaborators' access policies against requirements for requested collaboration pattern and classify any inconsistencies.

The method can also be use to analyse the impact changes in policy will have for collaboration partners.

Description Logics are mathematically simple, but that does not mean that Description Logic definitions are not complex; in particular, the effect any definition has on others is not always clear. Because each definition is made in terms of restrictions that may affect other definitions, we have chosen to verify that our definitions have the intended meaning by proving that definitions are correct once translated to the model theoretic semantics.

In the future we intend to extend this work to incorporate the following:

- Refine cross-organization collaboration patterns from different perspectives, for example, from business prosess point of view.
- Take context constraints that could affect access control into consideration, e.g. access time or access location.
- Discuss Role Based Access Control issues under collaborative context, e.g. role hierarchies and separation of duty in collaboration environment.

References

ANSI INCITS 359-2004 (2004).

B.Carminati, Ferrari, E. & Hung, P. (2006), Security conscious web service composition, *in* 'IEEE International Conference on Web Services', Chicago, Illinois, USA, pp. 489–496.

Bhatti, R., Bertino, E. & Ghafoor, A. (2004), A trust-based context-aware access control model for web-services, *in* 'IEEE International Conference on Web Services', San Diego, CA, pp. 184–191.

Bonatti, P. A. & Mogavero, F. (2008), Comparing rule-based policies, *in* '9th IEEE International Workshop on Policies for Distributed Systems and Networks', New York, USA, pp. 11–18.

Chae, J.-H. & Shiri, N. (2007), Description logic framework for access control and security in object-oriented systems, *in* 'RSFDGrC', pp. 565–573.

He, D. D. & Yang, J. (2007), Security policy specification and integration in business collaboration, *in* '2007 IEEE International Conference on Services Computing (SCC 2007)', Salt Lake City, Utah, USA, pp. 20–27.

Horrocks, I., Kutz, O. & Sattler, U. (2006), The even more irresistible $\mathcal{SROIQ}$, *in* 'Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)', AAAI Press, pp. 57–67.

Kagal, L., Paolucci, M., Srinivasan, N., Sycara, K. & G.Denker (2004), 'Authorization and privacy for semantic web services', *IEEE Intelligent Systems* **19**(4), 50–56.

Muthaiyah, S. & Kerschberg, L. (2006), Dynamic integration and semantic security policy ontology mapping for semantic web services (sws), *in* 'ICDIM', pp. 116–120.

OWL 1.1 Web Ontology Language (2006), W3C Member Submission. Available at `http://www.w3.org/Submission/2006/10/`.

Rouached, M. & Godart, C. (2007), Reasoning about events to specify authoriztion policies for web services composition, *in* 'Proceedings of 2007 International Conference on Web Services', IEEE, Salt Lake City, UT.

Sandhu, R. S., Coyne, E., Feinstein, H. & Youman, C. (1996), 'Role-based access control models', *IEEE Computer* **29**(2), 38–47.

Shields, B. & Molloy, O. (2007), Using description logic and rules to determine xml access control, *in* 'DEXA Workshops', pp. 718–724.

Sirer, E. G. & Wang, K. (2002), An access control language for web services, *in* 'SACMAT', pp. 23–30.

Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A. & Katz, Y. (2007), 'Pellet: A practical OWl-DL reasoner', *Journal of Web Semantics* **5**(2), 51–53.

Srivatsa, M., Iyengar, A., Mikalsen, T., Rouvellou, I. & Yin, J. (2007), An access control system for web service compositions, *in* 'Proceedings of International Conference on Web Services', IEEE, Salt Lake City, UT.

Wang, H., Jha, S., Livny, M. & McDaniel, P. D. (2004), Security policy reconciliation in distributed computing environments, *in* '5th IEEE International Workshop on Policies for Distributed Systems and Networks', pp. 137–147.

Web Ontology Language (OWL) (2004), W3C Recommendations. Available at `http://www.w3.org/2004/OWL/`.

Yau, S. S. & Chen, Z. (2008), Security policy integration and conflict reconciliation for collaborations among organizations in ubiquitous computing environments, *in* 'UIC', pp. 3–19.

Zhao, C., Heilili, N., Liu, S. & Lin, Z. (2005), Representation and reasoning on rbac: A description logic approach, *in* 'ICTAC', pp. 381–393.

Ziebermayr, T. & Probst, S. (2004), Web service authorization framework, *in* 'Proceedings of the IEEE International Conference on Web Services (ICWS'04)', San Diego, CA, pp. 614–621.