Enabling Data Sharing in Contextual Environments: Policy Representation and Analysis

Erisa Karafili
Imperial College London
180 Queen's Gate, SW7 2AZ, London, UK
e.karafili@imperial.ac.uk

Emil C. Lupu
Imperial College London
180 Queen's Gate, SW7 2AZ, London, UK
e.c.lupu@imperial.ac.uk

ABSTRACT

Internet of Things environments enable us to capture more and more data about the physical environment we live in and about ourselves. The data enable us to optimise resources, personalise services and offer unprecedented insights into our lives. However, to achieve these insights data need to be shared (and sometimes sold) between organisations imposing rights and obligations upon the sharing parties and in accordance with multiple layers of sometimes conflicting legislation at international, national and organisational levels. In this work, we show how such rules can be captured in a formal representation called "Data Sharing Agreements". We introduce the use of abductive reasoning and argumentation based techniques to work with context dependent rules, detect inconsistencies between them, and resolve the inconsistencies by assigning priorities to the rules. We show how through the use of argumentation based techniques use-cases taken from real life application are handled flexibly addressing trade-offs between confidentiality, privacy, availability and safety.

CCS CONCEPTS

• **Security and privacy** → **Security services**; **Access control**; **Information accountability and usage control**; • **Social and professional topics** → *Medical information policy*;

KEYWORDS

Data Sharing; Data Access; Usage Control; Cloud; Policy Language; Abductive Reasoning; Argumentation Reasoning

ACM Reference format:
Erisa Karafili and Emil C. Lupu. 2017. Enabling Data Sharing in Contextual Environments: Policy Representation and Analysis. In *Proceedings of SACMAT'17, Indianapolis, IN, USA, June 21-23, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3078861.3078876

1 INTRODUCTION

Data services are increasing popularity, especially with the rise of Big Data and IoT devices, where data are shared, stored, used and transformed by different entities. A serious issue in data services is the necessity of protecting and ensuring security properties of shared data. During the exchange of the latter, the entities involved should agree on the rules related to the data. These agreements are composed of data access and usage rules that ensure the security and privacy of the data, but also of user preferences, business and legislative rules. The above rules are applied to the same set of shared data in contextual environments. Therefore, conflicts between rules can be easily produced. Due to the legislative and context dependent nature of the rules, there is a need for the conflicting rules to co-exists, e.g., conflicting rules can have different priorities depending on the applied context. Furthermore, all the above rules need to be analyzed.

We propose a novel technique based on argumentation and abductive reasoning that uses a high-expressive policy analysis language [4] for representing and analyzing data sharing agreements rules, for Cloud environments. To the best of our knowledge, this is the first attempt where argumentation based reasoning is used for detecting and solving conflicts between policies, in particular for data sharing. Argumentation reasoning permits: the co-existence of conflicting rules due to its non-monotonic nature; the analysis made to the rules, where the integration with abductive reasoning increases the efficiency of the rules; and the conflict resolution through the use of priorities between rules. Our technique best accommodates the legislative and context dependent nature of the rules thanks to the expressive power of the used language and the argumentation reasoning that provides the rules with different priorities for different contexts.

Different techniques [21, 32, 34] have been introduced for data services in order to solve part of the security problems related to them. A good part of these techniques focuses on permitting a correct data access and an efficient usage control [8, 33], suitable for data at rest and not for data that migrate and cross between multiple parties. The problem of access and usage control is a well studied one [20, 28], but the existing solutions do not permit a fine grained representation of the different types of rules, their conflicts detection and resolution.

When an exchange of data occurs the parties should agree on the rules related to the data and create the *data sharing agreements* [30] (DSA) that describe how data should be treated. The agreement can be seen as a contract, between two or more parties, and the different rules are the terms of the contract. The terms express how and who is permitted/denied/obliged to access, delete, use, and share the data, along with the different constraints that should be respected. Representing and stipulating the various rules of the DSAs is not trivial, due to the heterogeneity of the rules and the conflicts generated between them, e.g., different rules, legislations, and contexts can be applied to the same shared data.

SACMAT'17, Indianapolis, IN, USA

DOI: http://dx.doi.org/10.1145/3078861.3078876

Our proposed technique, thanks to the high-expressivity of the used policy language, permits the representation of different access and usage rules together with their constraints, as well as the user preferences, business and legislation rules applied to the data. Some of the represented constraints are the temporal ones, e.g., a given piece of data can be accessed only for 30 minutes; geographical constraints, e.g., the data can be accessed only from the office; cardinality constraints, e.g., the data can be accessed at most three times; event-defined constraints, e.g., if the data is revoked by the author, the data cannot be further shared; purpose use constraints, e.g., the data can be used just for statistical use.

The DSAs rules are represented as policies that can permit, deny or oblige the execution of certain actions. Given their diverse type, it is common that different rules can lead to conflicting actions, e.g., we can have a permit policy to access a given piece of data and a deny access policy for the same piece of data (for the same user with the same constraints), or policies that oblige and deny the same action with identical constraints. The policies can lead to other types of conflicts with respect to the context that can be applied, called *conceptual conflicts*, e.g., for the same data two different and conflicting legislation rules can be applied. Deciding the rules that will be part of the DSAs is important, as DSAs should be composed of correct and non conflicting rules. Thus, it is essential that both of the above conflicts are detected and solved. The conflict resolution is not a trivial task because the rules can have exceptions, be used in specific contexts, or be incomparable.

To capture and solve the various conflicts, we base our technique on abductive and argumentation reasoning. We use an abductive constraints system, called A-system[1] [23] for finding the various conflicts between policies, together with redundancies and gaps between them. The conceptual conflicts cannot be captured by simply using the abductive system, as they are context dependent, and not easily spotted.

We use argumentation based reasoning to permit the co-existence of conflicting rules and to detect and solve the conceptual conflicts. Argumentation based reasoning is a well suited technique for implementing decision making mechanisms under conflicting, incomplete and context dependant knowledge. We use *GorgiasB*[2] [9], a tool based on abductive and argumentation reasoning, which easily accommodate the various policies and discover the conceptual conflicts. The latter are solved by introducing priorities/hierarchies between policies that state which policies have precedence over the others for particular contexts. The use of priorities between rules permits the co-existence of conflicting rules which best describe realistic rules, especially legislative ones where different laws can be applied to the same case but depending on the environment circumstances some rules might take precedence.

We start by giving a brief introduction to our main use case in Section 2. An overview of the related work regarding the data protection techniques is given in Section 3. In Section 4, we present briefly the used policy language, how it can represent data sharing agreements, and the performed analysis tasks. We introduce a novel policy analysis for conflict detection and resolution through an argumentation based decision process in Section 5. Finally, we go back to our use case and show its DSAs representation and conflict resolution in Section 6. In Section 7, we conclude and present some future works.

2 INTRODUCTION TO THE USE CASE

In this section, we introduce a real use case, where data sharing agreements need to be represented and stipulated. The use case is taken from an e-health scenario of an European Project (Coco Cloud project[3]). The main actors are the data subject, data controller, recipients, and data processor. For constructing the data sharing agreements the various rules are represented, and decisions about the rules that apply to the particular cases are made. Deciding the rules of DSAs is not trivial, as conflicts arises. The conflicts between the DSAs rules need to be captured, and solved.

In our use case, the patient is the *data subject*, some of his rights are to access to his medical data, to know who is processing his data, to ask for the deletion of his personal data. The *data controller* is an entity (public authority, agency, legal person), which determines the purpose and means of processing the data of the data subject. In our use case, the hospital is the data controller of the patient's data and determines the purpose for which the data are processed (e.g., administrative purpose or treatment purpose). The doctors of the hospital are the *data recipients* that need to comply to the data controller rules. The data recipients are considered as part of the data controller, the employees within the hospital do not stand as separate entities than the hospital itself. The hospital that is the data controller has various rules of how the doctors can access the patient's data, e.g., a doctor needs to be inside the hospital for accessing the patient's data (geographical constraint), he needs to be during his office hours (temporal constraint), and he needs to be the patient's treating doctor (role-based constraint). The above described rules are mainly business rules.

The *data processor* is an entity (public authority, agency, legal person) that is processing the data on behalf of the controller. In our use case, the cloud provider is considered the data processor as far as it respects the instructions of the controller. The controller rules that should be respected by the processor can also have a legal nature, e.g., if the controller is in an EU country, the cloud provider should as well be in an EU country and cannot send the data to countries outside the EU and EEA.

A *third party* is an entity (public authority, agency, legal person) that is not the data subject, data controller or processor, and that under the direct authority of the controller or processor is authorized to process the data. In our use case, a doctor outside the controller hospital is considered a third party. Once access is obtained, the third party becomes a data controller and has to comply with the data protection principals. Another third party can be an insurance company that asks for the patient's data to the hospital.

Some of the rules applied to the data are related to the type of data or the data subject. For example, the patient, as the data subject, has the permission to access his medical data. The legislation says that the patient cannot temporally access his private medical records, in case they are of a high emotional impact to him, e.g., suspects of a terminal illness that can effect the well-being of the patient. In this case, the patient is temporally denied the access to his data, until

[1] A-system http://dtai.cs.kuleuven.be/krr/Asystem/
[2] GorgiasB http://gorgiasb.tuc.gr/
[3] http://www.coco-cloud.eu/

the final results are issued. Thus, the rule that permits the patient to access his data is in conflict with the legislation because of the type of data.

The patient can be in different situations, e.g., intensive treatment, unconscious, emergency, that affect how the data are shared and used. For sake of simplicity we assume that the situation where the patient is involved, which describes the environment circumstances, is already given to the system by a trusted entity/agent. Usually the patient gives the permission to the doctor to access his sensitive data. In case the patient is unconscious, he is not able to grant this access. Thus, the doctor can access to the patient's personal data for contacting a family member and getting their permission to access the patient's sensitive data. The unconscious state does not give directly to the doctor the permission to access to all the patient's data. If the family member is not able to give this permission and the patient is in intensive treatment, the legislation permits a commission of the family doctor and hospital director to grant the access. In this case, there are various rules that are in conflict and different contexts are applied to the same case, e.g., unconscious, family member permission, intensive treatment. Thus, deciding the rules to be applied becomes more difficult.

The territory where the data are generated is another constraint added to the rules. The doctor can share the patient's data with a doctor of another hospital (third party), to ask for a second opinion. EU regulations state that data generated in EU can be shared just with EU and EEA countries. In case the third party doctor is not in an EU country, the data cannot be shared. Suppose now that the patient wants to take his medical data, generated in an EU country, outside an EU and EEA country. Also in this case, the patient cannot share his data.

Let's see another example where different contexts are applied and the data cross borders is more evident. Suppose that while the data subject is on vacation in a non EU and EEA country, he has an accident, and is in a critical situation for his life (e.g., an emergency situation). In this case, depending on the legal agreements EU has with that country, and because the data subject is in an emergency situation, part of the medical data of the patient can be shared. In this example, different rules create conflicts and the problem of deciding the rules to be applied is more difficult. In the coming sections, we will explain how these conflicts can be solved by introducing priorities between rules.

3 RELATED WORK

Nowadays, the solutions for protecting the used and shared data, aside from focusing on protecting the databases where they are stored [7] and the network used for their transfer [14], or constructing coordination techniques for data re-use [11], are also working on protecting the data themselves, by using data-centric solutions. The change of focus is due to the increase of connectivity between users, and with that also the increase of the various attacks. Protecting and ensuring the security of all the environments where the data are transferred/stored/used is becoming challenging. Therefore, data-centric security solutions that focus on protecting the data wherever they go, are taking hold [2, 17, 21, 32, 34].

In the data-centric security solutions persist two main challenges: the control of data access and the usage of data. Both of them, together with their own issues, have been widely studied. Different solutions are developed for solving their problems [6, 19]. The role-based access control [6] is a well known technique to ensure the data access depending on the user roles. Our solution can represent the different users roles and their specific policies for accessing and using the data.

UCON (Usage control) is a well studied concepts [25]. In [24], the authors introduce the usage control for controlling the access and usage of digital information. They put emphasis on the problem of delegation of rights that should be covered by UCON. In [27], the authors introduce the problem of usage control, by proposing a two level policy language that is expressive enough to represent the basic usage control notions, as prohibition and obligation, and to represent a generic server-side architecture that can implement usage control. The Obligation Specification Language (OSL) [8] expresses requirements for usage control, like obligations, permission-like statement, and constraints of the duration of a usage. This language is used by a mechanism for usage control [28]. Another work [15] focuses on using data usage control in distributed systems, in particular, when having a data flow in-between different connected systems. Fully decentralised infrastructure for enforcement of global usage control is introduced in [16], where the data as well as the events for the data usage occur in multiple distributed systems. Our work is taking into account all the above issues, faced by the usage control literature, as it permits policies that represent permissions, denials, obligations and delegations of rights.

Another interesting approach for sharing and accessing data is the use of sticky policies [21, 22]. Sticky polices are machine readable policies that contains conditions and constraints attached to data that describe how the data should be treated, as the latter cross multiple parties. In [13], the authors introduce the *sticky policy paradigm* and technologies for enterprise privacy enforcement and exchange of customer data. The promised privacy rights and obligations are specified through a privacy control language [12], for authorization management and access control, that includes user consents, obligations and distributed administration. The sticky policies are widely used in the cloud environment [26, 31]. Our policy language can represent the various policies represented by the sticky policies.

The data usage problems concern different entities that are using the data and the agreements they make regarding the different rules that describe how the data should be treated, called data sharing agreements [30]. The DSAs describe not only the agreements between the data subject, controller, and processor, but also the different business and legislation rules complaint to the contexts of data sharing. The authors of [20] introduce a language that represents the different rules of data sharing agreements. This approach though, suffers from the lack of expressivity, which does not permit the representation of complex DSAs, as well as the absence of analysis for the DSAs. Moreover, it does not permit dealing with the co-existence of conflicting rules and the problem of deciding the rules to be applied in particular cases.

All the above represented approaches, from the data access and usage control, to the sticky policies and finally the DSAs representation, suffer from the problem of deciding the rules that should apply to the shared data. For solving this problem, we propose an analysis

to be made to the rules together with a conflict resolution technique. The proposed analysis is based on the abductive [10] and argumentation based reasoning [3, 5]. Argumentation reasoning is a suitable technique for implementing decision making mechanisms [1, 9] under conflicting knowledge.

4 DATA SHARING AGREEMENTS: REPRESENTATION AND ANALYSIS

In this section, we give a brief introduction of the used policy language. We show how it can represent the different DSAs rules, together with their constraints and the data access and usage control rules [8, 28]. The policy language enables various analysis tasks performed to the rules and permits their efficiency and soundness. For performing the analysis, we use an abductive constraint logic programming system, A-system.

4.1 A policy analysis language

Our model is based on the policy analysis language [4]. This policy language, through its high expressive power, naturally represents the various rules and constraints that should be applied during the access, usage and sharing of data. It defines policies that represent in their structure *subject*, *targets*, and *actions*. It is composed of predicates, domain description predicates and policy regulations rules. The policy regulation rules are composed of predicates and domain description ones, and represent authorization and obligation rules. Some of the predicates of the policy language are introduced below.

$req(Sub, Tar, Act, T)$

$permitted(Sub, Tar, Act, T)$ $\quad$ $denied(Sub, Tar, Act, T)$

$do(Sub, Tar, Act, T)$ $\quad$ $deny(Sub, Tar, Act, T)$

$obl(Sub, Tar, Act, T_s, T_e, T)$

The predicate $req(Sub, Tar, Act, T)$ represents the request that a given subject, Sub, is doing at the instant of time T, for performing a given action, Act, to the target, Tar. The predicates $permitted(Sub, Tar, Act, T)$ and $denied(Sub, Tar, Act, T)$ represent respectively that to a given subject is permitted/denied at the instant of time T, to perform a certain action to the target. The predicates $do(Sub, Tar, Act, T)$ and $deny(Sub, Tar, Act, T)$ record respectively whether an action is permitted to occur or not. The predicate $obl(Sub, Tar, Act, T_s, T_e, T)$ denotes that at the instance of time T, a given subject is placed under an *obligation* to perform a certain action to the target between the interval of time from T_s to T_e, where T_s is the starting time when the obligation holds and T_e is the ending time of the obligation.

The domain description predicates represent changed/unchanged properties of the system regulated by policies. The unchanged properties, are static and usually defined by the user. The dynamic properties are defined using the Event Calculus [18], and represent a set of properties that define system events regulated or not by policies.

Some of the domain description predicates are *initiates*, *terminates*, *holdsAt*, *happens*. The $initiates$ predicate describes the state properties that hold due to an event, while $terminates$ describes which properties stop holding after an event. The *holdsAt* predicates means that a given property is true in a state, while the *happens* predicates indicates the event that occurs in a given instant of time.

4.2 Data Sharing Agreements Representation

The used policy language can represent the permission, denial and obligation concepts for the DSAs. In the following examples, we introduce the representation of various DSAs rules of our use case.

Example 4.1. Bob (B) is the family doctor ($fDoc$) of the patient Alice. The family doctor has the permission to access to Alice's prescriptions[4], (A_presc), at the instant of time T.

$$permitted(B, A\_presc, access, T) \leftarrow holdsAt(fDoc(B, Alice), T), \\ holdsAt(owner(Alice, A\_presc), T).$$

Every time, the family doctor Bob writes a prescription to Alice, he needs to send a notification, ($send\_n$), in less than 30 minutes.

$$obl(B, Alice, send\_n, T, T + 30, T) \leftarrow holdsAt(fDoc(B, Alice), T), \\ do(B, A\_presc, write, T), \\ holdsAt(owner(Alice, A\_presc), T).$$

We can represent DSAs rules with different types of constraints.

Example 4.2. Suppose that our patient, Alice, wants to give the permission ($perm$) to a family member to read her data. The family member, Faust (F), can read the data, after Alice has given the permission.

$$permitted(F, A\_presc, read, T) \leftarrow \quad T \geq T1, \\ holdsAt(owner(Alice, A\_presc), T_1), \\ do(Alice, F, perm(A\_presc, read), T_1).$$

Faust, Alice's family member, cannot read her prescriptions more than 4 times, where N is a number in this case.

$$permitted(F, A\_presc, read, T) \leftarrow \quad N \leq 4, T \geq T_1, \\ holdsAt(readD(F, A\_presc, N), T), \\ do(Alice, F, perm(A\_presc, read), T_1), \\ holdsAt(owner(Alice, A\_presc), T_1).$$

Example 4.3. Bob can access Alice's prescriptions just during his working hours ($shift$).

$$permitted(B, A\_presc, access, T) \leftarrow holdsAt(fDoc(B, Alice), T), \\ holdsAt(owner(Alice, A\_presc), T), \\ holdsAt(shift(B), T).$$

$$denied(B, A\_presc, access, T) \leftarrow holdsAt(fDoc(B, Alice), T), \\ holdsAt(owner(Alice, A\_presc), T), \\ \textbf{not } holdsAt(shift(B), T).$$

4.3 DSAs analysis

The used policy analysis language enables a wide range of analysis tasks that can be performed to the DSAs rules. Below we introduce briefly some of the performed analysis tasks that permit the construction of sound, complete and efficient DSAs.

The *modality conflicts* analysis task finds conflicts between policies regulation rules, and permits to have sound DSAs. In particular, it can capture the case when an action is both permitted/obliged and denied on the same instant of time. More complex conflicts can be constructed, where constraints about events occurrences and/or subject roles are added. All the above conflicts and inconsistencies between predicates are captured by our modality conflicts, which helps correcting them.

[4] We use the $owner$ property for relating the patient to his data.

The *coverage of gaps* analysis finds the different gaps (cases) that are not covered by the DSAs rules, and permits the construction of a complete list of rules that should be part of the DSAs. One type of gap that can be found is when there is an explicit request for performing a certain action to a certain object, and there is no authorization policy rule that neither permits nor denies this request. Another type of gap that can be found is when there is an explicit request for performing a certain action to a certain object, and this permission is given not as an authorization policy rule, but as a consequence of a default permission of the system.

The *policies comparison* analysis checks whether a policy is included/equivalent/implied by another one. This analysis improves the efficiency of the DSAs, by identifying redundant rules, that can be easily removed from the DSAs.

5 CONFLICT RESOLUTION THROUGH ARGUMENTATION REASONING

The above introduced analysis, implemented through an abductive system, is not able to capture the conceptual conflicts, as the latter are not direct conflicts between predicates (e.g., permitted/obliged and denied predicates), or they are context dependent. To find and solve the conceptual conflicts, we propose a technique based on non-monotonic reasoning [5], in particular, argumentation reasoning [3, 9]. We introduce an analysis that uses argumentation reasoning together with the abductive one, as the latter alone cannot capture the conceptual conflicts. The rules can be in conflict between each other, as they can hold for general domain description predicates but not for specific ones, or vice versa. The resolution of the conceptual conflicts is a decision making problem, and is solved by introducing priorities between rules [1]. We use argumentation reasoning, as it is a well suited technique for implementing decision making mechanisms for conflicting rules that have priorities/preferences between them and that are strongly context dependent. Argumentation reasoning permits to represent the various conflicting rules, the context where they are valid and the preferences between them. The priorities between rules permits us to work with conflicting policies and to analyze them.

Abductive and argumentation reasoning gives us the expressive power to work with strict, defeasible and conflicting rules, along with, exceptions and priorities between them. To apply the abductive and argumentation reasoning, we use the GorgiasB [29] tool. GorgiasB is a tool for preference-based argumentation with a graphical user interface. Its graphical interface helps to structure and model the knowledge and the decision making by preferences.

Our decision making technique has as input the various rules together with the domain description predicates that can be facts or defeasible knowledge, and finds the conflicts between rules, if there is any, and solves them. The resolution of the conflicts is done step by step, by putting priorities between rules[5] and explicitly specifying when a particular rule has to be considered stronger than another one. A preference/priority relation, denoted by $>$, is used to indicate preferences between rules. Given two conflicting rules r_1 and r_2, where for the context and the information we have, r_1 should be applied instead of r_2, we denote it with $r_1 > r_2$. The introduced priority rules together with the existing rules are checked, and if any conflict is found, other priorities rules are introduced.

Below we give an example of DSAs representation, where the analysis is performed for capturing the conflicting rules. The identified conflicts are solved by introducing priority rules.

Example 5.1. Following our use case, the family doctor can access the patients' prescriptions ($Presc$) and private information ($PInfo$). On the other hand, the treating doctor ($tDoc$) has a more restrictive access, as he is permitted to access just the patients' prescriptions, and not the private information[6]. Below, we represent the rules in a semi-natural language, where P and D denote respectively the patient and the doctor.

$$\begin{aligned}
&(i) Access(data, D, permitted) &&\leftarrow fDoc(D, P) \wedge Owner(P, data) \\
&&&\quad \wedge Presc(data) \\
&(ii) Access(data, D, permitted) &&\leftarrow fDoc(D, P) \wedge Owner(P, data) \\
&&&\quad \wedge PInfo(data) \\
&(iii) Access(data, D, permitted) &&\leftarrow tDoc(D, P) \wedge Owner(P, data) \\
&&&\quad \wedge Presc(data) \\
&(iv) Access(data, D, denied) &&\leftarrow tDoc(D, P) \wedge Owner(P, data) \\
&&&\quad \wedge PInfo(data)
\end{aligned}$$

The last rule is represented with the policy language as follows:

$$\begin{aligned}
denied(Sub, Tar, access, T) \leftarrow\ & req(Sub, Tar, access, T), \\
& holdsAt(tDoc(Sub, P), T), \\
& pInfo(Tar), \\
& holdsAt(owner(P, Tar), T), \\
& holdsAt(work(D, H), T), \\
& holdsAt(hosp(P, H), T).
\end{aligned}$$

When the patient is an emergency situation ($Emerg$) (where his life is at risk) the doctor has access to the patient's private information, e.g., for contacting a family member of the patient. This exception is represented as below.

$$\begin{aligned}
(v)\ Access(data, D, permitted) \leftarrow\ & Emerg(P, H) \wedge tDoc(D, P) \wedge \\
& Owner(P, data) \wedge PInfo(data)
\end{aligned}$$

The above predicate is written in policy language as follows:

$$\begin{aligned}
permitted(Sub, Tar, access, T) \leftarrow\ & req(Sub, Tar, access, T), \\
& holdsAt(Emerg(P, H), T), \\
& holdsAt(tDoc(Sub, P), T), \\
& pInfo(Tar), \\
& holdsAt(owner(P, Tar), T), \\
& holdsAt(work(D, H), T), \\
& holdsAt(hosp(P, H), T).
\end{aligned}$$

Conflicts are found for the above rules. The two last rules, (iv) and (v), are in conflict with each other, as one says that generally treating doctors cannot access the private information of a patient, while the other one gives permission to the treating doctor to access the information, in case of an emergency. A preference relation between rules is added in this case, stating that rule (v) is preferred over rule (iv), $(v) > (iv)$, in case there is an emergency. After the preference is introduced and no other conflict is found, the scenario can be tested.

[5] For sake of understandability, we call the priorities between rules, *priority rules*.

[6] The treating doctor should work ($work(D, H)$) in the same hospital (H) where his patient is hospitalized ($hosp(P, H)$).

6 USE CASE: DSAS REPRESENTATION AND ANALYSIS

In this section, we continue with our use case that was already introduced in the previous sections. First, we describe the various entities involved in the DSAs and the types of data. We continue by showing some DSAs rules and their analysis, where the conflicts are detected and solved by introducing priorities between rules.

As described in the above sections, we want to model the data sharing agreements between different entities, for sharing and using the data. In our case, we give a special focus on the use of the cloud environment. Our solution can be used independently from the applied environment. The different actors are the patients (sometimes we call them clients) $\mathcal{P} = \{P_1, P_2, \cdots\}$, the service providers that are the different hospitals and medical centers $\mathcal{H} = \{H_1, H_2, \cdots\}$, and the doctors $\mathcal{D} = \{D_1, D_2, \cdots\}$, that work in various hospitals or medical centers.

Every patient has his associated *data* that can be of three types:

- prescriptions: $Presc(data)$, e.g., blood pressure, analyses, medicine prescriptions, x-rays, etc.;
- private prescriptions: $PData(data)$, e.g., anti-depressive treatments;
- personal information: $PInfo(data)$, e.g., contact information and family member contacts.

A further division, based on the notion of type $TypeD(data, type)$ that describe the medical data, is made to the prescription data, e.g., the otolaryngology and the dental information are of the same type, the orthopaedic data and the different x-rays data are of the same type, while food allergy data and x-rays data have different types. General analyses and blood pressure are considered general medical information, thus, we include them in all types of data.

Usually, in EU countries, every patient (P) has a *family doctor*: $FDoc(D, P)$. When the patient is treated/examined/hospitalized in an hospital, he has also the *treating doctors*: $TDoc(D, P)$. In this case, for D to be the treating doctor of P, then D should work in the same hospital where P is treated/examined/hospitalized, as follows:

$$TDoc(D, P) \text{ when } Hosp(P, H) \wedge Work(D, H).$$

The doctors usually have a specialisation. Thus, we divide them by their specialisation, called types[7]: $Spec(D, type)$.

The patient's data are used, accessed, and shared between different entities by respecting their DSAs. The first step is to agree on the terms of the DSAs, where some terms, usually legal ones, are irrefutable. Let us give some of the DSAs terms for our scenario.

(1) The family doctor can access to all the data of the patient.
(2) The patient can access to all his data.
(3) The treating doctor can access to the prescription data related to his specialisation.
(4) The hospital regulation says that the treating doctor can access to the patient's data during his working time, and while he is in the hospital.
(5) The treating doctor cannot access to the patient's data when he is not in the hospital, or not during his shift, or the data are not related to his specialisation.
(6) Nobody else can access the data.

[7]For sake of simplicity, we assume that the types of the specialisations of the doctors are the same as the types that the prescriptions data are divided.

The family doctor accesses to all the patient's data, described as follows:

$$Access(data, P, permitted) \leftarrow FDoc(D, P) \wedge Owner(P, data) \quad (1)$$

Rule 2 states that the patient can access to all of his data, represented as below:

$$Access(data, P, permitted) \leftarrow Owner(P, data) \quad (2)$$

Rule 3 states that the treating doctor is permitted to access the prescriptions, in particular, just the prescriptions that deal with his specialisation, e.g., an orthopaedic doctor accesses to the x-ray of the patient, but does not access to his food allergies data. Rule 4 states that the treating doctor can access to the patient's data just during his shift, $shift(Doctor)$, and while he is in the hospital. For ensuring the latter, we compare the position of the hospital where the doctor is working, $hospP(Hospital, Location)$ with the position of the doctor, $position(D, Location)$. All the other accesses, e.g., while the doctor is not in the hospital, not during his working hours, or not relevant prescription to his specialisation, are not allowed, rule 5. Below, we introduce the representation of rule 3 and 4 together, and rule 5.

$$\begin{aligned} Access(data, D, permitted) \leftarrow\ & TDoc(D, P) \wedge Owner(P, data) \wedge \\ & Presc(data) \wedge TypeD(data, t_1) \wedge \\ & Spec(D, t_2) \wedge t_1 = t_2 \wedge shift(D) \\ & \wedge position(D, L_1) \wedge hospP(H, L_2) \\ & \wedge same(L_1, L_2) \end{aligned} \quad (3)$$

$$\begin{aligned} Access(data, D, denied) \leftarrow\ & TDoc(D, P) \wedge Owner(P, data) \wedge \\ & (PData(data) \vee PInfo(data) \vee \textbf{not } shift(D) \vee \\ & (position(D, L_1) \wedge hospP(H, L_2) \wedge \textbf{not } same(L_1, L_2)) \vee \\ & (TypeD(data, t_1) \wedge Spec(D, t_2) \wedge t_1 \neq t_2 \wedge Presc(data))) \end{aligned} \quad (5)$$

There are some cases, in EU legislation, where the patient cannot access his data, e.g., when the patient's medical data can be of high emotional impact $Emot(P, data)$ (e.g., suspects of a terminal illness that if revealed early to the patient can effect his well-being and life). The rule in this case is:

(7) If the medical data are of high emotional impact (as described by law), then the patient is not allowed to access his private prescription.

$$\begin{aligned} Access(data, P, denied) \leftarrow\ & Emot(P, data) \wedge PData(data) \wedge \\ & Owner(P, data) \end{aligned} \quad (7)$$

The above rule 7 is in conflict with rule 2. The latter permits the patient to access his data. The conflict is solved by putting priorities between them. We state that the last rule is stronger than the previous one, in case of high emotional impact data: rule 7 > rule 2.

An interesting rule is the one dealing with the intensive treatment, $Intens(P, H)$[8]. The treating doctor can access to the patient's private prescription when the patient grants access to him.

(8) When the patient is in intensive treatment the treating doctor can access to all the normal prescription of the patient, despite his specialisation.
(9) The doctor can access to the patient's private prescription when the patient grants $Grant(P, data, D)$ access to him.

[8]$Intens(P, H)$ means that patient P is in intensive treatment in hospital H.

(10) All the rest of the data accesses are denied.

$$Access(data, D, permitted) \leftarrow Intens(P, H) \wedge TDoc(D, P) \wedge Presc(data) \wedge Owner(P, data) \wedge position(D, L_1) \wedge hospP(H, L_2) \wedge same(L_1, L_2) \wedge shift(D) \quad (8)$$

$$Access(data, D, permitted) \leftarrow TDoc(D, P) \wedge PData(data) \wedge Owner(P, data) \wedge Grant(P, data, D) \wedge hospP(H, L_2) \wedge position(D, L_1) \wedge same(L_1, L_2) \wedge shift(D) \quad (9)$$

$$Access(data, D, denied) \leftarrow Intens(P, H) \wedge Owner(P, data) \wedge TDoc(D, P) \wedge (PInfo(data) \vee (PData(data) \wedge \textbf{not } Grant(P, data, D))) \quad (10)$$

In this case, rule 8 and 9 are in conflict with rule 5, as the latter is denying the access to not related and private prescriptions, while the two new rules are permitting it. The cases of being in an intensive treatment and granting the access to the treating doctor have higher priority. Thus, rule 8 > rule 5 and rule 9 > rule 5.

Sometimes, patients can be unconscious when they are in intensive treatment. In this case, we have the following rules.

(11) When the patient is in intensive treatment and unconscious $Uncon(P)$, the doctor can access to the patient's private information, for contacting a family member.

(12) In the above case, the doctor can access the private prescription when a patient's family member grant access to them, $FGrant(P, data, D)$.

$$Access(data, D, permitted) \leftarrow Intens(P, H) \wedge TDoc(D, P) \wedge PInfo(data) \wedge Owner(P, data) \wedge Uncon(P) \wedge hospP(H, L_2) \wedge position(D, L_1) \wedge same(L_1, L_2) \wedge shift(D) \quad (11)$$

$$Access(data, D, permitted) \leftarrow Intens(P, H) \wedge TDoc(D, P) \wedge PData(data) \wedge Owner(P, data) \wedge Uncon(P) \wedge FGrant(P, data, D) \wedge hospP(H, L_2) \wedge position(D, L_1) \wedge same(L_1, L_2) \wedge shift(D) \quad (12)$$

Also in this case, we have conflicts between policies. In particular, rule 11 and 12 are in conflict with both rule 5 and 10. Again, the exception rules have higher priority, as being unconscious and in intensive treatment is stronger then being hospitalized or just in intensive treatment. Thus, rule 11 > rule 5, rule 11 > rule 10, rule 12 > rule 5 and rule 12 > rule 10.

Sometimes, the family members are not able to grant the access. In such case, the hospital and law regulations state that the access can be granted from the family doctor and the ward director ($director(D, H)$), by using the $Grant^*$ predicate.

(13) When the patient is in intensive treatment, unconscious, and the family member cannot grant access to the private prescription, the doctor can access the patient's private prescription, when the family doctor and ward/hospital director grant the access to him.

$$Access(data, D, permitted) \leftarrow Intens(P, H) \wedge TDoc(D, P) \wedge PData(data) \wedge Owner(P, data) \wedge Uncon(P) \wedge fDoc(P, D_1) \wedge director(D_2, H) \wedge hospP(H, L_2) \wedge Grant^*(D_1, D_2, data, D) \wedge \textbf{not } FGrant(P, data, D) \wedge shift(D) \wedge position(D, L_1) \wedge same(L_1, L_2) \quad (13)$$

In this case, we have conflicts between rules 13 and rule 5 and 10, where rules 13 has higher priority: rule 13 > rule 5, rule 13 > rule 10.

The medical data of the patient are shared between hospitals inside the EU or EEA, e.g., for asking for a second opinion, or when the patient is staying in another country. The medical data cannot be shared outside the EU and EEA. Suppose that while the patient is on vacation in a non EU or EEA country, e.g., Canada, he has an accident, and is in a critical situation for his life (e.g., an emergency situation). When the patient is in an emergency situation, the patient's prescriptions can be shared to another hospital outside the EU and EEA, in case that country has legal agreements with the EU. In this case, Canada is part of a "white-list" of countries, where the cross borders flow of information is permitted.

(14) The data can be shared inside the EU and EEA.

(15) The data cannot be shared outside EU or EEA.

(16) In case, the patient is in an emergency situation in a non EU or EEA country, then the patient's prescriptions can be shared with that country, if that country has legal agreements for cross borders flow of information with EU.

We represent rules 14 and 15, when a second opinion ($SecondOp$) is requested by the treating doctor D_1 to another doctor D. To decide if the access is permitted or denied, we check the location of the hospital where D is working. $EU^*(Hospital)$ indicates if the given hospital is located in an EU or EAA country or not.

$$Access(data, D, permitted) \leftarrow Owner(P, data) \wedge TDoc(D_1, P) \wedge SecondOp(D_1, D) \wedge Work(D, H) \wedge EU^*(H) \quad (14)$$

$$Access(data, D, denied) \leftarrow Owner(P, data) \wedge Work(D, H) \wedge \textbf{not } EU^*(H) \quad (15)$$

We represent below rule 16, where the predicate $Agreement$ indicates that the country where the hospital is located has legal agreement with EU for cross border flow information.

$$Access(data, D, permitted) \leftarrow Emerg(P, H) \wedge Owner(P, data) \wedge Presc(data) \wedge Work(D, H) \wedge \textbf{not } EU^*(H) \wedge Agreement(H) \quad (16)$$

In this case, rule 16 is in conflict with rule 15, where the emergency situation and the legal agreements prevails. Thus, rule 16 is stronger than rule 15, represented as rule 16 > rule 15.

7 CONCLUSION AND FUTURE WORK

Data sharing agreements are a useful abstraction to group together references to the rules governing the sharing of data which regard legislation and constraints on data usage. Such rules are often conflicting and naturally have different priorities according to the context of application. The conflicts between rules arise from conflicts between business needs and privacy considerations and between security and safety, in particular in medical applications. To represent the rules of the data sharing agreements an expressive

policy notation is needed. In this work, we presented a new technique for representing and working with data sharing agreements, based on a policy language and argumentation reasoning, that allows to express context dependent rules regarding data usage and obligations.

Analysis of the DSAs rules for inconsistencies is necessary, as the inconsistencies generate conflicts between rules. We showed how this analysis can be done through abduction and argumentation. We performed the analysis that capture conflicts between DSAs rules with the help of an abductive based tool (A-system). Furthermore, we performed other two tasks analysis that find redundant rules, and gaps between them, by giving as result correspondingly the redundant rules and the missing cases. To handle the context dependent nature of the priorities between policies and the co-existence of conflicting ones we use an argumentation based techniques. We showed how the introduced argumentation based analysis handles naturally a variety of use cases and how systems such as GorgiasB can be used for its implementation. Our analysis captures the conceptual conflicts and solves them by introducing priorities between the conflicting rules. The introduced analysis is applied offline to the rules, and improves the efficiency and correctness of the DSAs. We showed the DSAs representation together with the analysis and conflict resolution through a real use case scenario, taken from an e-health scenario.

A future challenge is to work with policies that deal with data integrity and availability. Currently the priority rules are introduced manually, due also to the complexity of the involved rules. An interesting future work is to automate this process and use online learning to gather information and learn automatically the priorities between rules, depending on the contexts. Data quality characteristics like timeliness, interpretation and relevance, are interesting to be analyzed in further works, by bridging the gap that exists between data access/usage control and security and information systems.

ACKNOWLEDGMENTS

Supported by FP7 EU-funded project Coco Cloud grant no.: 610853, and EPSRC Project CIPART grant no. EP/L022729/1.

REFERENCES

[1] Arosha K. Bandara, Antonis C. Kakas, Emil C. Lupu, and Alessandra Russo. 2009. Using argumentation logic for firewall configuration management. In *Integrated Network Management, IM 2009. 11th IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, 180–187.

[2] Jennifer Bayuk. 2009. Data-centric security. *Computer Fraud & Security* 2009, 3 (2009), 7–11.

[3] Andrei Bondarenko, Phan Minh Dung, Robert A. Kowalski, and Francesca Toni. 1997. An Abstract, Argumentation-Theoretic Approach to Default Reasoning. *Artif. Intell.* 93 (1997), 63–101.

[4] Robert Craven, Jorge Lobo, Jiefei Ma, Alessandra Russo, Emil C. Lupu, and Arosha K. Bandara. 2009. Expressive policy analysis with enhanced system dynamicity. In *Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security, ASIACCS*. ACM, 239–250.

[5] Phan Minh Dung. 1995. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games. *Artif. Intell.* 77, 2 (1995), 321–358.

[6] David F. Ferraiolo and D. Richard Kuhn. 1992. Role-Based Access Controls. In *15th National Computer Security Conference*.

[7] Michael Gertz and Sushil Jajodia. 2007. *Handbook of database security: applications and trends*. Springer Science & Business Media.

[8] Manuel Hilty, Alexander Pretschner, David A. Basin, Christian Schaefer, and Thomas Walter. 2007. A Policy Language for Distributed Usage Control. In *Computer Security - ESORICS*. Springer, 531–546.

[9] Antonis Kakas and Pavlos Moraitis. 2003. Argumentation Based Decision Making for Autonomous Agents. In *AAMAS*. ACM, 883–890.

[10] Antonis C. Kakas, Robert A. Kowalski, and Francesca Toni. 1992. Abductive Logic Programming. *J. Log. Comput.* 2, 6 (1992), 719–770.

[11] Erisa Karafili, Hanne Riis Nielson, and Flemming Nielson. 2015. How to Trust the Re-use of Data. In *Security and Trust Management - 11th International Workshop, STM*. Springer, 72–88.

[12] Günter Karjoth and Matthias Schunter. 2002. A privacy policy model for enterprises. In *Proceedings 15th IEEE Computer Security Foundations Workshop (CSFW-15)*. IEEE Computer Society, 271–281.

[13] Günter Karjoth, Matthias Schunter, and Michael Waidner. 2003. Platform for Enterprise Privacy Practices: Privacy-Enabled Management of Customer Data. In *Privacy Enhancing Technologies: Second International Workshop, (PET)*, Roger Dingledine and Paul Syverson (Eds.). Springer, 69–84.

[14] Charlie Kaufman, Radia Perlman, and Mike Speciner. 2002. *Network Security: Private Communication in a Public World, Second Edition* (second ed.). Prentice Hall Press, Upper Saddle River, NJ, USA.

[15] Florian Kelbert and Alexander Pretschner. 2013. Data usage control enforcement in distributed systems. In *Third ACM Conference on Data and Application Security and Privacy, CODASPY'13*. ACM, 71–82.

[16] Florian Kelbert and Alexander Pretschner. 2015. A Fully Decentralized Data Usage Control Enforcement Infrastructure. In *Applied Cryptography and Network Security - 13th International Conference, ACNS*. Springer, 409–430.

[17] Young-Jin Kim, Marina Thottan, Vladimir Kolesnikov, and Wonsuck Lee. 2010. A secure decentralized data-centric information infrastructure for smart grid. *IEEE Communications Magazine* 48, 11 (2010), 58–65.

[18] Robert A. Kowalski and Marek J. Sergot. 1986. A Logic-based Calculus of Events. *New Generation Comput.* 4, 1 (1986), 67–95.

[19] Aliaksandr Lazouski, Fabio Martinelli, and Paolo Mori. 2012. A Prototype for Enforcing Usage Control Policies Based on XACML. In *Trust, Privacy and Security in Digital Business - 9th International Conference, TrustBus*. Springer, 79–92.

[20] Ilaria Matteucci, Marinella Petrocchi, and Marco Luca Sbodio. 2010. CNL4DSA: a controlled natural language for data sharing agreements. In *Proceedings of ACM Symposium on Applied Computing (SAC)*. ACM, 616–620.

[21] Marco Casassa Mont and Siani Pearson. 2011. Sticky Policies: An Approach for Managing Privacy across Multiple Parties. *Computer* 44 (2011), 60–68.

[22] Marco Casassa Mont, Siani Pearson, and Pete Bramhall. 2003. Towards accountable management of identity and privacy: sticky policies and enforceable tracing services. In *14th International Workshop on Database and Expert Systems Applications, 2003. Proceedings*. IEEE Computer Society, 377–382.

[23] Bert Van Nuffelen and Antonis C. Kakas. 2001. A-system: Declarative Programming with Abduction. In *Logic Programming and Nonmonotonic Reasoning, 6th International Conference, LPNMR, Proceedings*. Springer, 393–396.

[24] Jaehong Park and Ravi S. Sandhu. 2002. Towards usage control models: beyond traditional access control. In *SACMAT*. ACM, 57–64.

[25] Jaehong Park and Ravi S. Sandhu. 2004. The UCON_{ABC} usage control model. *ACM Trans. Inf. Syst. Secur.* 7, 1 (2004), 128–174.

[26] Siani Pearson, Marco Casassa Mont, Liqun Chen, and Archie Reed. 2011. End-to-End Policy-Based Encryption and Management of Data in the Cloud. In *2011 IEEE Third International Conference on Cloud Computing Technology and Science*. IEEE Computer Society, 764–771.

[27] Alexander Pretschner, Manuel Hilty, and David A. Basin. 2006. Distributed usage control. *Commun. ACM* 49, 9 (2006), 39–44.

[28] Alexander Pretschner, Manuel Hilty, David A. Basin, Christian Schaefer, and Thomas Walter. 2008. Mechanisms for usage control. In *Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security, ASIACCS*. ACM, 240–244.

[29] Nikolaos I. Spanoudakis, Antonis C. Kakas, and Pavlos Moraitis. 2016. Gorgias-B: Argumentation in Practice. In *Computational Models of Argument - Proceedings of COMMA*. IOS Press, 477–478.

[30] Vipin Swarup, Len Seligman, and Arnon Rosenthal. 2006. Specifying Data Sharing Agreements. In *7th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY)*. IEEE Computer Society, 157–162.

[31] Slim Trabelsi and Jakub Sendor. 2012. Sticky Policies for Data Control in the Cloud. In *Proceedings of the 2012 Tenth Annual International Conference on Privacy, Security and Trust (PST '12)*. IEEE Computer Society, 75–80.

[32] Cong Wang, Qian Wang, Kui Ren, and Wenjing Lou. 2010. Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing. In *INFOCOM, 2010 Proceedings IEEE*. IEEE, 1–9.

[33] Xinwen Zhang, Francesco Parisi-Presicce, Ravi Sandhu, and Jaehong Park. 2005. Formal Model and Policy Specification of Usage Control. *ACM Trans. Inf. Syst. Secur.* 8 (2005), 351–387.

[34] Wenchao Zhou, Micah Sherr, William R. Marczak, Zhuoyao Zhang, Tao Tao, Boon Thau Loo, and Insup Lee. 2010. Towards a Data-centric View of Cloud Security. In *Proceedings of the Second International Workshop on Cloud Data Management (CloudDB '10)*. ACM, 25–32.