Security Analysis and Legal Compliance Checking for the Design of Privacy-friendly Information Systems

Paolo Guarda
Faculty of Law, University of Trento
Trento, Italy
paolo.guarda@unitn.it

Silvio Ranise
Security & Trust, FBK-Irst
Trento, Italy
ranise@fbk.eu

Hari Siswantoro
Security & Trust, FBK-Irst
DISI, University of Trento
Trento, Italy
siswantoro@fbk.eu

ABSTRACT

Nowadays, most of business practices involve personal data processing of customers and employees. This is strictly regulated by legislation to protect the rights of the data subject. Enforcing regulation into enterprise information system is a non-trivial task that requires an interdisciplinary approach. This paper presents a declarative framework to support the specification of information system designs, purpose-aware access control policies, and the legal requirements derived from the European Data Protection Directive. This allows for compliance checking via a reduction to policy refinement that is supported by available automated tools. We briefly discuss the results of the compliance analysis with a prototype tool on a simple but realistic scenario about the processing of personal data to produce salary slips of employees in an Italian organization.

CCS CONCEPTS

• **Security and privacy** → *Formal security models*; • **Applied computing** → *Law*;

KEYWORDS

EU DPD; Legal compliance; Access Control Policies

ACM Reference format:
Paolo Guarda, Silvio Ranise, and Hari Siswantoro. 2017. Security Analysis and Legal Compliance Checking for the Design of Privacy-friendly Information Systems. In *Proceedings of SACMAT'17, Indianapolis, IN, USA, June 21-23, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3078861.3078879

1 INTRODUCTION

In today's interconnected world, the security of IT systems is a continuously evolving endeavour as the threat landscape changes in real time making inadequate—shortly after their deployments—security policies, mechanisms, and tools. This fast moving situation requires that organizations be constantly vigilant to ensure that their security posture remains strong by keeping their controls up-to-date. To add complexity, legal requirements protecting specific types of data must also be taken into account and suitably enforced in order to comply with existing laws and regulations. The most important class of legal requirements concern privacy and data protection as they constitute core values of individuals. Several legislations concern these values: the EU Data Protection Directive (EU DPD),[1] the HIPAA,[2] and the Sarbanes-Oxley[3].

The ultimate goal of an effort to integrate legal compliance and security solutions is to protect data appropriately (including those subject to regulations) and to guarantee the privacy of users. For this to become possible, it is crucial to develop methodologies and techniques that support the specification and automated analyses of regulations for data protection and privacy together with security solutions in a coherent and uniform way. Automation is essential to command the complexity of today's (and future's) digital information systems and allow for quick updates to security and privacy solutions for countering new threats. There are *three main desiderata* that such methodologies and techniques should satisfy to unfold their full benefit for privacy and data protection: (**D1**) they should be applicable at the very beginning of the design process so as to facilitate the embedding of security solutions and Privacy Enhancing Technologies (PETs) as recommended by Security- and Privacy-by-Design approaches; see, e.g., [11]; (**D2**) they should document which simplifying assumptions about the regulations are being made to resolve the ambiguities of natural language, give them a precise meaning, and permit the application of automated techniques for security analysis and compliance checking; and (**D3**) they should present system designers with detailed results about the reasons for which a security analysis or a compliance check is failing. In other words, it is not enough that the tools return a yes/no answer and should include scenarios (e.g., authorization queries) violating the property under consideration.

(D1) implies that standard specification languages at design time (such as Message Sequence Charts or Business Process Notation) should be supported by the techniques. Because of (D2), the mathematical model and the document describing the simplifying assumptions that relate it to the text of the regulation should be the results of an interdisciplinary approach involving legal experts, computer scientists, and IT security experts. This would promote a deeper understanding of the regulation, its corner cases, and those parts that are applicable to digital information systems formalized in a mathematical model. This, in turn, would facilitate the interpretation of the results returned by the tools for security analysis and legal compliance. (D3) would simplify the process of patching designs and policies even in face of evolving security and legal

SACMAT'17, Indianapolis, IN, USA

DOI: http://dx.doi.org/10.1145/3078861.3078879

[1] http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:31995L0046
[2] https://www.hhs.gov/hipaa/
[3] http://www.soxlaw.com

requirements. To the best of our knowledge, no available approach addresses the three desiderata above.

The main contributions of the paper are a methodology and a technique to integrate legal compliance and security checks fulfilling the desiderata above. These are based on three building blocks: (1) a declarative framework to specify the processing of data for certain purposes together with legal requirements and security policies at design-time (Section 3.1), (2) an interdisciplinary approach to derive a formal specification—expressed in the declarative framework—from regulations or laws written in natural language (Section 3.2), and (3) automated techniques to solve security analysis and compliance checking problems (Section 3.3).

The declarative framework (1) supports the specification of digital information system designs and permits the expression of legal requirements in a scenario independent way and of security policies abstracting away the details of the enforcement mechanisms—cf. (*D1*). The framework (1) and the interdisciplinary approach (2) allow one to make explicit the simplifying assumptions that lead to the formalization of (parts of) the regulation—cf. (*D2*). An important by-product of (1) and (2) is the capability of instantiating legal requirements derived from legislations to the scenario under consideration by defining few key notions (such as the legal roles played by the entities in the system and how these are empowered or mandated with selected capabilities) without the need to consider the regulation in its entirety (with all the subtleties related to the use of legal jargon) every time a new scenario is considered. Following an established tradition in policy verification, security analyses are reduced to logical problems whose solution is possible by using (off-the-shelf) Satisfiability Modulo Theories (SMT) solvers. Given the use of a common framework to express both security and legal requirements, another contribution of the paper is to show that similar reductions to logical problems can be done to support also compliance checking. The use of state-of-the-art theorem provers permit to provide policy designers with authorization queries that violate the property under consideration—cf. (*D3*). Since reducing security policy analyses to logical problems is well-known (see, e.g., [3]), in this paper we mainly focus on compliance.

We apply the approach to the EU Data Protection Directive (DPD) since the utility of the proposed techniques can be evaluated in the context of the EU DPD whose implications have been thoroughly studied. We use a simple scenario to illustrate the key ideas underlying our work (Section 2).

2 OVERVIEW

We illustrate our approach to integrate security analysis and compliance checking on a simple scenario, namely the processing of personal data to produce the salary slips of employees in an Italian organization, named ITOrg. The process is described by the Message Sequence Chart (MSC) of Figure 1. ITOrg asks each Employee to fill in a form with profile information such as name, surname, address, number of kids, type of car, etc. The Employee can send the filled in form to ITOrg (message 'profile') and give her consent for some of the purposes for which (parts of) the data in the form can be used. ITOrg delegates the processing of selected parts of the profile to its departments. For producing salary slips, after checking that the employee has given the consent for processing information

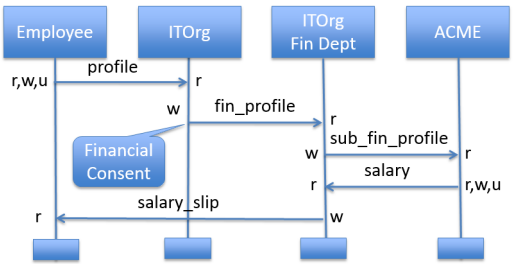

Figure 1: Semantics of the purpose 'salary_comp'

that are used to compute the salary slips—the callout in the figure, ITOrg forwards the pertinent information in the profile to its Fin(ancial) Dep(ar)t(ement)—message 'fin_profile'. In turn, ITOrg Fin Dept sends (possibly a sub-set of) the received information to ACME (message 'sub_fin_profile') that performs the actual computations to produce the salary slip. Once the computations have been performed, ACME sends the salary slip to ITOrg Fin Dept (message 'salary') that can be further processed (if the case) and finally send the salary slip to the Employee (message 'salary_slip'). Notice that each send and receive event in the MSC is decorated with permissions: r for read, w for write, and u for update. The meaning of such decorations is the following: the entity sending or receiving a message with payload p must have (one of) the permissions in the decoration close to it. For instance, an Employee should be granted the permission to w(rite) or u(pdate) her profile in order to send a message containing it. The fact that an entity has (or not) the permissions specified in the decorations is specified by an access control policy (see, e.g., [12]). There are several ways to describe such a policy, one of which is the access control matrix in Table 1 characterizing the rights of each subject (row) with respect to every object (column) in the system (for brevity, profile has been shortened to pro, salary to sal, and ITOrg has been dropped from ITOrg Fin Dept). An entry in the table marked with an asterisk means that the right can only be exercised in the context of the process to achieve the given purpose 'salary_comp.' (For example, the ITOrg Fin Dept can read the content of a message containing the 'salary' of an employee if this action is performed in the process described by the MSC of Figure 1.) This implies that we consider access control policies that are purpose-aware (see, e.g., [1]). We assign the meaning to a purpose by associating it with a plan to achieve certain goals since "an action is for a purpose if it is part of a plan for achieving that purpose;" see, e.g., [25]. Among the many possible ways to describe plans, one of the most popular is to use MSCs (especially at design-time) as we did for the scenario above. The use of MSCs allows us also to specify the contextual conditions

Table 1: Access control policy to produce salary slips

	pro	fin_pro	sub_fin_pro	sal	sal_slip
Employee	r,w,u				r
ITOrg	r	w			
Fin Dept		r*	w*	r*	w*
ACME			r*	r,w,u*	

under which the process (purpose) should be executed, as it is required to specify security policies in almost all modern information systems; see, e.g., [18]. To illustrate, consider the MSC in Figure 1: ITOrg can send the message 'fin_profile' only if the Employee has given her consent (call-out in the figure marked Financial Consent) to the processing of financial data.

The problem of legal compliance. Are the personal information of employees processed in a way which is compliant with the EU DPD? In order to answer this question, we need the availability of three main ingredients: (i) a formalization of the EU DPD, (ii) a mean to instantiate it to the information system under consideration, and (iii) techniques to check compliance.

To derive (i), several challenges must be addressed because of the generality and comprehensiveness of the set of rules introduced to protect EU citizens against the uncontrolled collection and use of personal data with the ultimate goal of respecting individual privacy. For instance, not all the rules in the regulation can be expressed (and enforced) as (purpose-aware) access control policies because of their complexity and broad spectrum of applicability. An essential part of this work consisted in a careful analysis of the text of the regulation in order to identify the parts which are amenable to be specified by (purpose-aware) access control policies. We describe how we derive a formalization of the EU DPD in Section 3.2 below. Here, we just present an excerpt of the EU DPD, shown in Table 2, which is relevant to check the compliance of the system described by the MSC in Figure 1 and the access control policy in Table 1. Column Eff(ect) reports when p(ermitting) and d(enying) to *Proc*ess some Personal Data (*PD*) by some entity under the control of the Data Subject (*DS*), Data Controller (*DC*), or the Data Processor (*DP*) for a certain *Pur*pose (when needed) according to what is described in the column Condition. *PD* are information relating to an identified or identifiable natural person (e.g., the profile information of an employee). *DC* is a natural or legal person which alone or jointly with others determines the purposes and means of the processing of PD (e.g., ITOrg is the DC in the system considered above). *DS* is an individual that is the subject of the PD held by a DC (e.g., an employee is the DS). *DP* is any individual or organization that processes PD on behalf of DC (e.g., both ITOrg Fin Dept and ACME are DPs). The first three lines of Table 2 contains the conditions permitting an action to be performed while the last three lines shows the conditions denying the action the right to be executed. More precisely, the condition in the first line of the table specifies that a *DS* can *Proc*ess the *PD* provided that a *DC* has *Emp*owered her with the possibility to do so. While the EU DPD stipulates that—regardless of the purpose—a DS can perform any action on her PD,

Table 2: Formalization of the EU DPD (excerpt)

Eff	Condition
p	$DS \wedge Proc \wedge PD \wedge Emp$
p	$DC \wedge Proc \wedge PD \wedge Pur \wedge Cons$
p	$DP \wedge Proc \wedge PD \wedge Pur \wedge Man$
d	$DS \wedge Proc \wedge PD \wedge \neg Emp$
d	$DC \wedge Proc \wedge PD \wedge \neg Pur \wedge Cons$
d	$DP \wedge Proc \wedge PD \wedge Pur \wedge \neg Man$

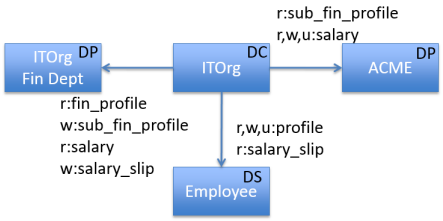

Figure 2: A bridge structure for salary slips

an information system is usually designed to provide only a sub-set of all possible actions that a DS is entitled to do. For this reason, we introduced *Emp* to express which actions the DS can perform when interacting with the system. The condition in the second line of the table specifies that a *DC* can *Proc*ess *PD* for a given *Pur*pose provided that the *DS* has given her *Cons*ent. The condition in the third line of the table specifies that a *DP* can *Proc*ess *PD* for a given *Pur*pose provided that a *DC* has *Man*dated it to do so. This formalizes one of the principles stated in the EU DPD that a *DC* can delegate (part of) the operations to one or more *DP*s in order to carry out the processing required to achieve a given purpose. The conditions in the last three lines of the table correspond to (some of) the negative versions of those in the first three. In the case of a *DS* trying to *Proc*ess her *PD* if a *DC* has not empowered her to do so, this is sufficient to deny access (fourth line). Similarly, a *DC* cannot *Proc*ess *PD* if the *DS* has not given her consent (fifth line) and a *DP* cannot *Proc*ess *PD* if a *DC* has not mandated it to do so (last line). Since the EU DPD describes only the provisions permitting access, we have developed an approach to derive negative rules that we explain in Section 3.2 below.

We are now left with the problem to instantiate the formal rules in Table 2 to the system under consideration—ingredient (ii) above. We do this by using a labeled directed graph, called bridge structure, mapping the legal roles (namely, *DS*, *DC*, and *DP*) to the entities in the system (e.g., Employee and ITOrg) and describing the *Emp*ower and *Man*date relationships between *DC* and *DS* or *DP*, respectively. The bridge structure for the system described by the MSC in Figure 1 and the policy in Table 1, is shown in Figure 2. Each node is identified by an entity in the MSC and is labeled by a legal role: an Employee is a *DS*, ITOrg is a *DC*, ITOrg Fin Dept and ACME are two *DP*s. The *Emp*ower relation is identified by the arrow from ITOrg to Employee which is labeled by the pairs 'r,w,u:profile' and 'r:salary_slip' saying that an Employee is entitled to r(ead), w(rite) and u(pdate) the profile and to r(ead) the salary_slip. The *Man*date relation is identified by the two arrows from ITOrg to ITOrg Fin Dept and ACME which are labeled by 'r:fin_profile,' 'w:sub_fin_profile', 'r:salary' and 'w:salary_slip' for the former and 'r:sub_fin_profile' with 'r,w,u:salary' for the latter saying that ITOrg Fin Dept is delegated the permissions to r(ead) the fin_profile and the salary as well as to w(rite) the sub_fin_profile and salary_slip, that ACME is delegated to r(ead) sub_fin_profile and to r(ead), w(rite) and u(pdate) the salary.

With the information in the bridge structure of Figure 2, it is easy to instantiate the abstract EU DPD rules of Table 2 to the information system for producing salary slips. In fact, it is sufficient to define the various notions in the (abstract) conditions in terms

of the notions introduced in the (concrete) access control policy of Table 1 by using the mapping from entities in the system to legal roles (i.e. *DS* is an Employee, *DC* is ITOrg, *DP* is either ITOrg Fin Dept or ACME) and the *Emp*ower (e.g., Employee is entitled to write and update the profile) and *Man*date (e.g., ACME is delegated to read sub_fin_profile by ITOrg) relationships. By doing this, it is possible to interpret the abstract rules in Table 2 as access control policies involving the same entities of the concrete ones in Table 1. For instance, the first line of Table 2 can be read as an Employee is permitted to read, write and update the profile or to read the salary_slip. As another example, the third line of Table 2 can be read as follows: ACME can read sub_fin_profile or read, write and update salary for the purpose of salary_comp.

As a last step—ingredient (iii) above, we are required to check the compliance of the access control policy of Table 1 against the EU DPD rules in Table 2. For this, we leverage the fact that the bridge structure in Figure 2 allowed us to derive a version of the EU DPD rules instantiated to the system under consideration as discussed above. It is easy to see that such an instantiation can be seen as an access control policy (denoted π_{iDPD} below) expressed in the same terms used by the access control matrix in Table 1 (denoted π_{acm} below). Based on this observation and the availability of several automated techniques to check for policy refinement (see, e.g., [3, 26]), we recast the problem of checking the compliance of the access control policy against the EU DPD as the problem of verifying that the π_{acm} refines π_{iDPD} or, equivalently, that every authorization requests permitted or negated by π_{acm} is also so by π_{iDPD}. An available tool for policy analysis, such us those described in [3, 26], can automatically perform this check with the results summarized in Table 3. The lines marked with R_3 and R_4 refer to the cases in which both policies return P(ermit) and D(eny), respectively. Since only these two cases are reported by the tool, we are entitled to conclude that there exist no sources of non-compliance as it is never the case that an authorization query is permitted by π_{iDPD} and denied or left undetermined by π_{acm}, neither that a query is left undetermined by π_{iDPD} and granted or denied by π_{acm}. Notice that the tools show some (concrete) queries that are permitted and some that are denied. For instance, the first line implies that an Employee can u(pdate) her pro(file) even if she has not given the consent to process the personal data concerning her. As another example, the fifth line implies that ITOrg cannot write the salary_slip if the employee has not given her consent to the processing.

We describe the formal framework to express the (abstract) EU DPD rules, the (concrete) access control policies, and the bridge

Table 3: Results of refinement checking (excerpt)

	s	a	o	Fin Consent
R_3	Employee	u	pro	False
R_3	ITOrg	w	sub_fin_pro	True
R_3	Fin Dept	r	sub_fin_pro	True
R_3	ACME	w	sal_slip	True
R_4	ITOrg	w	sal_slip	False
R_4	Fin Dept	w	sal	False
R_4	ACME	r	sal	False

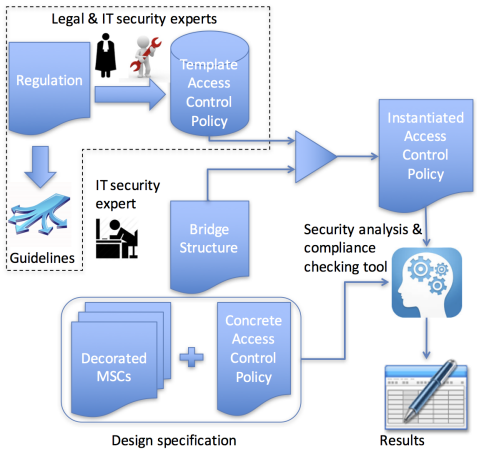

Figure 3: Our approach

structures in Section 3.1. We explain how these notions can be used to implement compliance checking on top of available techniques for the analysis of access control policies in Section 3.3.

3 THE THREE BUILDING BLOCKS

Figure 3 shows an overview of our approach to compliance checking. We start by considering the L-shaped (dashed) box at the top-left corner of the picture. Legal and IT security experts collaborate to identify the parts of the regulation (in our case the EU DPD) that are amenable to formalization, make it explicit any simplifying assumptions restricting the scope of applicability of the rules, use the declarative framework to derive a mathematical model, and compile a set of guidelines (in natural language) that should help IT system designers to bridge the gap between technical and legal levels. This process is time consuming, requires a lot of ingenuity and interdisciplinary skills but it is done once for each regulation of interest. The remaining of the figure shows what an IT security experts (possibly complementing the designs of IT system architects) should do in order to come up with a privacy-friendly IT system design. First of all, she produces a (decorated version) of the MSCs describing the main processes in the system and associates each one of them with a purpose. Second, she designs the (concrete) access control policies that the various entities in the MSC should respect in order to send or receive messages. Third, by using the guidelines made available by the group of experts that produced the formal model of the regulation, she specifies the bridge structure in order to instantiate the (formal) model to the system under consideration. Afterwards, she can use the automated tool for security analysis and compliance checking to answer several questions about the system design: is this authorization query permitted or denied? Do the (concrete) access control policy enable the execution of the scenarios described by the MSCs in the design? And, most importantly for this paper, is the (concrete) access control policy compliant with the (formalization of the) regulation? The results returned by the tool can be used by the IT system expert to revise the system design or the bridge structure when security or compliance issues are detected.

3.1 Declarative framework

We take First Order Logic (FOL) [13] as the declarative framework underlying our approach. The reason for this choice is three-fold. First, FOL seems to be expressive enough according to our experience with the formalization of the EU DPD reported below and the long line of works on specifying access control policies with logic (see, e.g., [23]). Second, there is a well-known set of techniques to reduce policy analysis problems to logical ones that can be solved by using off-the-shelf available reasoning tools, called Satisfiability Modulo Theories (SMT) solvers (see, e.g., [3, 26]). Third, there is a cornucopia of techniques available in the literature to model the dynamic behavior of IT systems and verify their properties, which use FOL and SMT solvers, that can be exploited to model the (decorated versions of the) MSCs adopted in our approach and verify basic properties such as their feasibility (see, e.g., [9]). For lack of space, we discuss here only the most relevant notions underlying the formalization of policies.

Access Control Policies. We take Attribute Based Access Control (ABAC) [18] as the model underlying our policies. The choice is motivated by the observation (made in [20]) that ABAC supports, not only, the simulation and combination of a wide range of classical access control models but also their refinement so as to supplement rather than supplanting the classical models. In this way, we are free to specify a wide range of access control policy idioms together with their combinations.

In ABAC, access rights are permitted or denied depending on the security-relevant characteristics—called attributes—of the entities involved in access control: a *subject* (e.g., a user or an application) asking to perform an *action* (e.g., read, write, update) on a *resource* (e.g., a file, a document, or a database record) in an *environment*, i.e. a collection of contextual information (e.g., location, time of day). The tension between the specification of access rights (i.e. actions that subjects can perform on resources) and safety (i.e. no subject can get permissions that compromise some security goals) requires to identify the authorization queries known to be permitted, denied, and unregulated (i.e. neither permitted nor prohibited) [19]. We formalize this as follows.

Let S, A, R, and E be sets of subjects, actions, resources, and environments, respectively. Following [2], we regard these entities as records whose fields are their attributes; an entity is uniquely identified by the values associated to its attributes. Thus, S, A, R, and E are the Cartesian products of the set of possible values of each attribute (this is uniquely determined according to an arbitrary order over the attributes). An *access control policy* is a tuple (S, A, R, E, P, D) where S, A, R, E are as defined above while P and D are sub-sets of $AQ = S \times A \times R \times E$, whose elements are called *authorization queries*. P is the set of *permitted* authorization queries—i.e. s is allowed to perform a on r in e when (s, a, r, e) is in P—and D is the set of *denied* ones—i.e. s is not allowed to perform a on r in e when (s, a, r, e) is in D.

A group of subjects is a sub-set of the set S and a resource (environment) class is a sub-set of the set R (E, respectively). We assume that actions have just one attribute that ranges over the set of possible action identifiers and that environments have (at least) the following two attributes: *purpose* ranging over the (finite) set of possible purpose identifiers and *consent* ranging over the set of Boolean functions from the set of resource identifiers. For any policy (P, D), we assume that P and D are non-empty and disjoint. It may be the case that the union of P and D do not contain all possible authorization queries, i.e. $(P \cup D) \subset AQ$ (this is known as the open-world assumption [12]). An authorization query in the set $U = AQ \setminus (P \cup D)$ is *unregulated*. When complementing a set X w.r.t. AQ, we write X^c; e.g., $U = (P \cup D)^c$.

The sets P and D of permitted and denied authorization queries are given as set-comprehensions of the form $\{(s, a, r, e) \mid \varphi(s, a, r, e)\}$ where s, a, r, and e denote the tuples of attributes of subjects, actions, resources, and environments, respectively, and φ is a FOL expression constraining (at most) the attributes in s, a, r, and e. In the same way, we can specify a group of users as $\{s \mid \upsilon(s)\}$, a resource class as $\{r \mid \rho(s)\}$, and an environment class as $\{e \mid \gamma(e)\}$ for υ, ρ, and γ FOL expressions constraining (at most) the attributes in s, r, and e, respectively.

3.2 From regulations to template policies

The EU DPD identifies the general principles (including consent, specification of purpose, minimal disclosure, and data quality [17]) that regulate how data shall be performed and organize them in a set of legal provisions. Given our focus on (purpose-aware) access control policies, we have considered only those provisions directly related to such policies. In other words, we have discarded provisions (a) from which it was impossible to derive access control rules and (b) those requiring substantial human interpretation. An example of (a) is art. 1: "*Object of the Directive: 1. In accordance with this Directive, Member States shall protect the fundamental rights and freedoms of natural persons, and in particular their right to privacy with respect to the processing of personal data. 2. Member States shall neither restrict nor prohibit the free flow of personal data between Member States for reasons connected with the protection afforded under paragraph 1.*" An example of (b) is art. 6(1): "*Member States shall provide that personal data must be: (a) processed fairly and lawfully.*" We were able to consider some of the provisions involving human judgement when an authorization condition could be identified. Examples of this are the consent given (or not) by the data subject (art. 7(a)) and the quality of data that should be "*adequate, relevant and not excessive in relation to the purposes for which they are collected and/or further processed*" (art. 6(1)). To keep track of the simplifying assumptions—desideratum (*D2*) in the introduction—resulting from the ideas stated above, we produced an annotated version of the EU DPD[4] with the hope of clarifying the results obtained by the compliance checking technique (to be described in Section 3.3 below). The document aims to simplify the task of deriving template access control policies from the natural language.

Below, we describe how we have identified the possible types of information that shall be protected, the set of legal roles involved in the processing of data, a set of auxiliary notions that are crucial to express authorization conditions and specify the possible relationships among the various roles or contextual conditions, and a bridge structure for the instantiation of the template policy to an ABAC policy for the system under consideration.

[4]The annotated version of the EU DPD with supplementary material including the guidelines for system designers, the prototype tool, benchmarks, and experimental results are available on-line at https://sites.google.com/view/eu-dpd-gdpr-compliance.

Data classes. From our annotated version of the EU DPD, we have identified the following three classes of data: (1) Personal Data (*PD*) "*shall mean any information relating to an identified or identifiable natural person ('data subject'); an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity*" (art. 2(a)); (2) Sensitive Data (*SD*) is "*personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade-union membership, and the processing of data concerning health or sex life*" (art. 8(1)); and (3) Non-Personal Data (*NPD*) is information that, either in origin or on account of its having been processed, cannot be associated with any identified or identifiable data subject; hence, the EU DPD may not be applied.

Legal roles. We have defined the following three legal roles: (1) Data Controller (*DC*) "*shall mean the natural or legal person, public authority, agency or any other body which alone or jointly with others determines the purposes and means of the processing of personal data; where the purposes and means of processing are determined by national or Community laws or regulations, the controller or the specific criteria for his nomination may be designated by national or Community law*" (art. 2(d)); (2) Data Processor (*DP*) "*shall mean a natural or legal person, public authority, agency or any other body which processes personal data on behalf of the controller*" (art. 2(e)); (3) Data Subject (*DS*) "*shall mean any natural person that is the subject of the personal data*" (art. 2(a)).

Auxiliary notions. We have singled out the following four recurring conditions for the specification of access control policies to preserve privacy: (1) Purpose (*Pur*), (2) Consent (*Cons*), (3) Data Quality (*DQ*), and (4) Member State Requirement (*MSReq*). For *Pur*, observe that "*PD shall be collected for specified, lawful and legitimate purposes and not processed in ways that are incompatible with the purposes for which the data have been collected*" (art. 6(1)(b)). The view adopted in this work (recall Section 3.1) is to associate a purpose with a MSC defining a set of plans to achieve certain goals so that an action is carried out in the context of a plan. For *Cons*, *PD* "*shall be collected and processed only if the data subjects have given their explicit consent to data processing*" (art. 7(a)). This means a clear affirmative act establishing a freely given, specific, informed and unambiguous indication of the data subject's agreement to the processing of *PD*, such as by a written statement, including by electronic means. For *DQ*, *PD* "*shall be adequate, relevant, and not excessive with respect to the purposes for which they are collected and processed; accurate and, where necessary, kept up to date; retained no longer than necessary for the purposes for which the data were collected*" (art. 6(c, d, e)). This requirement ensures that the data processed are always those which are strictly necessary to achieve a certain purpose and represents a variation of the Privacy-by-default principle[5]. For *MSReq*, Member States (MSs) "*shall, within the limits of the provisions analysed, determine more precisely the conditions under which the processing of personal data is lawful*" (art. 5). The EU DPD sets the 'lowest common denominator' with respect to the legal provisions about data protection and privacy; MSs may indicate additional ones in order to align the EU DPD to their legal systems. This allows us to impose additional regulatory constraints when considering scenarios in which national legislations may be more restrictive than the EU DPD.

We have also identified two main relationships between *DC*s and *DS*s or *DP*s: (1) Mandate (*Man*) and (2) Empower (*Emp*). The former specifies which operations the *DC* delegates to the *DP*s (see art. 2(e), art. 16, art. 17(2,3)). A *DP* can further delegate some of the operations to other *DP*s. When the *DC* mandates an action to a *DP*, it can perform the action. The relation *Emp* specifies which operations the *DC* gives the possibility to execute to the *DS* when interacting with the system being designed. Indeed, art. 12 states that the *DS* has unrestricted access to the data related to her. However, an information system typically provides only a sub-set of all possible operations on some data. The relation *Emp* allows us to specify which actions the system being designed will support and to which the EU DPD shall apply. The *DS* should still be able to execute the operations not mentioned in *Emp*, albeit in ways which are not supported by the system (e.g., by manual intervention of system administrators).

Template policies. To derive the formal rules of the template access control policies, we have replaced the natural language text with the corresponding data classes, legal roles, and auxiliary notions identified above in the tabular format of the selected articles (overall we have selected 8 out of 34 articles in the EU DPD) and used the standard Boolean connectives to combine them and form expressions of Boolean algebra. To illustrate, consider a rule taken from art. 7(a). From this, we are able to derive the following Boolean expression:

$$\begin{array}{ll} (DC \wedge Proc \wedge PD \wedge Pur \wedge Cons) & \vee \\ (DP \wedge Man \wedge Proc \wedge PD \wedge Pur \wedge Cons) & \end{array} \quad (1)$$

where $\vee$ is disjunction and $\wedge$ is conjunction. Such expression can be read as the Data Controller (*DC*) or a Data Processor (*DP*) mandated by the Data Controller (*Man*) can process (*Proc*) Personal Data (*PD*) for an allowed purpose (*Pur*) and an explicit consent has been given by the Data subject (*Cons*). Formally, the abbreviations *DC*, *DP*, *DS*, etc introduced above are considered as Boolean variables that act as placeholders and await to be instantiated to every system under consideration by means of a bridge structure, such as the graph depicted in Figure 2 (see also below). We let $\mathbb{V}$ be the set of Boolean variables corresponding to the abbreviations of the notions introduced above, $\mathbb{P}$ the disjunction of the (27) conjunctions of variables in $\mathbb{V}$ derived from the EU DPD as described above, and $\mathbb{D}$ the disjunction of the (82) conjunctions of (possibly negated) variables in $\mathbb{V}$ obtained by negating only those variables that form the "mandatory" part of each authorization condition in $\mathbb{P}$. To illustrate, two of the disjuncts in $\mathbb{D}$ corresponding to (1) are $DC \wedge Proc \wedge PD \wedge \neg Pur \wedge Cons$ and $DC \wedge Proc \wedge PD \wedge Pur \wedge \neg Cons$ where $\neg$ denotes negation. The meaning of the disjuncts above is as follows: a *DC* cannot process *PD* for a purpose which is not the appropriate one even if the *DS* has given her consent (first disjunct), a *DC* cannot process *PD* for an appropriate purpose when the *DS* has not given her consent (second disjunct), a *DP* cannot process *PD* if it has not received the mandate to do so ($\neg Man$) even if the

[5]This principle, implicit in the EU DPD text, was made explicit in the General Data Protection Regulation, that will come into force in 2018. In particular, art. 25(2) says that the *controller shall implement appropriate technical and organisational measures for ensuring that, by default, only personal data which are necessary for each specific purpose of the processing are processed.*

DS has given her consent and the processing is for a legitimate purpose (third disjunct). This is our approach to address the problem that the EU DPD lists only the positive provisions, i.e. conditions permitting access. A template policy is the pair $(\mathbb{P}, \mathbb{D})$ of Boolean expressions over the set $\mathbb{V}$ of Boolean variables.

We observe that some disjuncts in $\mathbb{P}$ and $\mathbb{D}$ have been obtained by considering not only a given letter of a selected article but also some additional articles defining some notions that should be ubiquitously taken into account. To illustrate, consider art. 7(a); expression (1) was obtained by including also articles 2(a), 16, and 17(2,3) that introduce the notion of mandate from a *DC* to a *DP*; this is why the second disjunct in (1) contains the condition *Man*.

Bridge structures. The last artifact supported by our framework is the bridge structure whose goal is to specify how the template policy should be instantiated to the system under consideration. Roughly, the idea is to describe how the legal roles map to the groups of user, the data classes to the resource classes, the mandate and empower relation are implemented in the system. More precisely, one needs to identify the sets $\mathcal{G}$, $\mathcal{R}$, $\mathcal{N}$ corresponding to the users groups, resource classes, and environment classes in the system design. Then, a bridge structure is a tuple $(lr, dc, dq, msr, ma, em)$ where lr is a mapping from $\mathbb{S} = \{DS, DC, DP\}$ to a FOL formula representing user groups in $\mathcal{G}$, dc is a mapping from $\mathbb{D} = \{PD, SD, NPD\}$ to a FOL formula representing resource classes in $\mathcal{R}$, dq is a mapping from $\{DQ\}$ to a FOL formula representing resource classes in $\mathcal{R}$, msr is a mapping from $\{MSReq\}$ to a FOL formula constraining the attributes of subjects, actions, resources, and environments, ma is a mapping from $\{Man\}$ to a FOL formula constraining the attributes of an entity corresponding to a DC or a DP, another entity corresponding to a DP, a resource, and an action, and em is a mapping from $\{Emp\}$ to a FOL formula constraining the attributes of an entity corresponding to a DC, another entity corresponding to a DS, a resource, and an action.

Example 3.1. Let us consider again the system for producing salary slips described in Section 2. We explain how (most of) the functions in the bridge structure can be derived from the labeled graph in Figure 2 and the MSC in Figure 1. lr can be derived from the annotations (in black) of the nodes; e.g., $lr(DS)$ = Employee, $lr(DC)$ = ITOrg, $lr(DP)$ = ITOrg Fin Dept ∨ ACME. dc is implicitly defined by the superscripts of messages in the MSC; when they are omitted (as in Figure 1), it means that all resource classes are considered *PD*, i.e. dc maps *PD* to the disjunction of the formulae representing all resource classes mentioned in the MSC. For the sake of simplicity, we omitted considerations concerning the data quality and further legal requirements imposed by MSs. As a consequence, both dq and msr maps the single variable in their domains to *True*. The mappings ma and em can be read from the labeled arrows in the graph of Figure 2. For example, $em(Emp)$ returns the formula

$$\begin{aligned}
&(\text{ITOrg} \wedge a = r \wedge \text{profile} \wedge \text{Employee}) \quad \vee \\
&(\text{ITOrg} \wedge a = w \wedge \text{profile} \wedge \text{Employee}) \quad \vee \\
&(\text{ITOrg} \wedge a = u \wedge \text{profile} \wedge \text{Employee}) \quad \vee \\
&(\text{ITOrg} \wedge a = r \wedge \text{salary_slip} \wedge \text{Employee})
\end{aligned}$$

meaning that ITOrg (the data controller) empowers the Employee (the data subject) with the capabilities of reading ($a = r$), writing ($a = w$) or updating ($a = u$) her profile and to read ($a = r$) her salary slips. □

Given a system design and a bridge structure B, we define the instantiated template policy as the ABAC policy $(\iota_B(\mathbb{P}), \iota_B(\mathbb{D}))$ where ι_B is the instantiation function defined (by recursion) on the structure of the Boolean formulae in the obvious way.

3.3 Security and Compliance

We assume that a design specification is given for a given ABAC policy $\pi = (P, D)$ together with a bridge structure β. It is possible to reduce several policy analysis problems to satisfiability problems [3, 26]. Here, for lack of space, we consider only the policy refinement problem.

This consists of verifying which authorization queries are permitted by a policy and denied or left undetermined by another. Formally, this can be stated as follows: a policy (P, D) refines a policy (P', D') iff $P \subseteq P'$ and $D \subseteq D'$. Assuming that $P' = \{(s, a, r, e) | \varphi'_P(s, a, r, e)\}$ and $D' = \{(s, a, r, e) | \varphi'_D(s, a, r, e)\}$, it is possible to reduce the two set-inclusions above as the validity of the following two formulae: $\varphi_P \Rightarrow \varphi'_P$ and $\varphi_D \Rightarrow \varphi'_D$. By refutation these are equivalent to check that

$$\varphi_P \wedge \neg\varphi'_P \text{ and } \varphi_D \wedge \neg\varphi'_D \text{ are both unsatisfiable.} \tag{2}$$

While the reduction of policy analysis problems to satisfiability problems in FOL considered above is well-known (see, e.g., [3, 26]) and we omit it here, a contribution of this paper is to use the encoding of refinement between policies to support legal compliance checking. This is possible by using the bridge structure B that induces an instantiation function ι_B that allows us to derive an ABAC policy $(\iota_B(\mathbb{P}), \iota_B(\mathbb{D}))$ from the template policy $(\mathbb{P}, \mathbb{D})$, as observed at the end of Section 3.2. The idea is then to define that a design specification for an ABAC policy (P, D) is compliant with the (EU DPD) template policy $(\mathbb{P}, \mathbb{D})$ under the bridge structure B iff (P, D) refines $(\iota_B(\mathbb{P}), \iota_B(\mathbb{D}))$. In turn, the refinement test can be reduced to two satisfiability checking problems by using (2), i.e.

$$\varphi_P \wedge \neg\iota_B(\mathbb{P}) \text{ and } \varphi_D \wedge \neg\iota_B(\mathbb{D}) \text{ are both unsatisfiable.} \tag{3}$$

Mechanizing compliance. The first step toward mechanization is to identify sufficient conditions to guarantee the decidability of the satisfiability problem (3) above. Following [3, 26], we assume the types of the attributes of the various entities may range over the integers and the reals with Linear Arithmetic operations and the usual ordering relations, enumerated data-types, abstract sets with total functions, tuples, and records. There are two advantages in adopting this set τ of types. First, it is expressive enough to specify a wide variety of situations as witnessed by its adoption in popular, generic, model-based specification languages such as B and Z. Second, it is well-known how to model the types in τ as a theory T_τ of first-order logic whose satisfiability problem for quantifier-free formulae (i.e. arbitrary Boolean combinations of constraints on the attributes that can be expressed in the theory T_τ) is decidable; see, e.g., [3] for details.

Theorem 3.2. *Checking policy compliance is decidable and NP-complete if (A1) the types of the attributes of an ABAC policy are in τ and (A2) the FOL expressions defining permitted and denied authorization queries of the ABAC policy, user groups, classes of resources*

and environments, and the formulae returned by the bridge structure are quantifier-free.

This is a corollary of results in [3]. NP-completeness, in our experience, is not a hindrance to the practical applicability of our approach because of the efficiency of state-of-the-art SMT solvers.

4 DISCUSSION

We have introduced a declarative framework—based on FOL—to support the specification of information system designs, purpose-aware access control policies, and legal requirements. We have shown how to instantiate the legal requirements to a particular system for checking compliance via a reduction to policy refinement. We have presented techniques, using SMT solvers, supporting the integration of the standard policy analysis problem with compliance checking. Our experience with an implementation of the proposed technique confirms its utility and scalability on realistic and synthetic compliance problems.[6]

Since policy analysis problems have been thoroughly studied in the literature about access control (see, e.g., [3, 26] for an overview), here we focus on closely related works about legal compliance. Lam et al. [22] propose a privacy policy specification language based on Datalog capable of encoding some parts of HIPAA and show the decidability of the language. Barth et al. [4] present a framework—based on Temporal Logic—for specifying privacy regulations like HIPAA and introduce two notions of compliance that take into account the past computations and their impact on the future to allow a given purpose to be achieved. The work in [8] elaborates on the formalism proposed by Barth et al. and presents a decidability result for compliance checking. It is possible to incorporate such constraints in our framework by adapting techniques for solving the Workflow Satisfiability Problem in security-sensitive business processes (see, e.g., [10]). As future work, we plan to compare the expressiveness of our (extended) framework and the one in [8].

Garg et al. [16] propose an expressive, first-order logic-based privacy policy specification language in which HIPAA can be completely encoded. They present an auditing algorithm that incrementally inspects the system log against a policy and detects violations. Chowdhury et al. [7] propose extensions of XACML for specifying HIPAA and enforcing compliance at run-time. A similar approach is developed in [14, 15] for the EU DPD. Our approach differs from these because it focus on design time, uses simpler specification languages, and use static analysis techniques to ensure compliance by construction.

In requirement engineering, some approaches (e.g., [5, 6, 24]) have been proposed to ensure legal compliance by extending software requirements. Such works focus on the specification of legal provisions with little (or no) support to checking compliance.

As future work, we plan to derive a template policy for the General Data Protection Regulation (GDPR) that is going to be adopted in the various EU MSs starting May 2018. We believe our approach may help organizations to speed up the compliance process. We also plan to study the relevance of some of the notions introduced in our framework, such as the Mandate and Empower relations, with the strategy for data governance that an organization may adopt [21]. This should permit the alignment of the data governance strategy with the privacy and data protection policies as early as possible during system development.

[6]For lack of space, we omit a report of our findings that can be found on-line at https://sites.google.com/view/eu-dpd-gdpr-compliance.

REFERENCES

[1] C.A. Ardagna, M. Cremonini, S. De Capitani di Vimercati, and P. Samarati. 2008. A Privacy-Aware Access Control System. *JCS* 16, 4 (2008), 369–392.

[2] A. Armando, S. Oudkerk, S. Ranise, and K. Wrona. 2014. Formal Modelling of Content-Based Protection and Release for Access Control in NATO Operations. In *FPS 2013 (LNCS)*, Vol. 8352. 227–244.

[3] A. Armando, S. Ranise, R. Traverso, and K. Wrona. 2016. SMT-based Enforcement and Analysis of NATO Content-based Protection and Release Policies. In *ABAC@CODASPY*. ACM, 35–46.

[4] A. Datta Barth, J. C. Mitchell, and H. Nissenbaum. 2006. Privacy and contextual integrity: Framework and applications. In *IEEE Symp. on S&P*.

[5] Travis D Breaux and Annie I Antón. 2008. Analyzing regulatory rules for privacy and security requirements. *Software Engineering, IEEE Transactions on* 34, 1 (2008), 5–20.

[6] Travis D Breaux, Matthew W Vail, Annie Antón, and others. 2006. Towards regulatory compliance: Extracting rights and obligations to align requirements with regulations. In *Req. Eng., 14th Int. Conf.* IEEE, 49–58.

[7] Omar Chowdhury, Haining Chen, Jianwei Niu, Ninghui Li, and Elisa Bertino. 2012. On XACML's adequacy to specify and to enforce HIPAA. In *USENIX Ws. on Health S&P*.

[8] Omar Chowdhury, Andreas Gampe, Jianwei Niu, Jeffery von Ronne, Jared Bennatt, Anupam Datta, Limin Jia, and William H Winsborough. 2013. Privacy promises that can be kept: A policy analysis method with application to the HIPAA privacy rule. In *SACMAT*. ACM, 3–14.

[9] A. Cimatti, S. Mover, and S. Tonetta. 2011. Proving and explaining the unfeasibility of message sequence charts for hybrid systems. In *FMCAD*. 54–62.

[10] J. Crampton. 2005. A reference monitor for workflow systems with constrained task execution. In *SACMAT*.

[11] G. Danezis, J. Domingo-Ferrer, M. Hansen, J.-H. Hoepman, D. Le Métayer, R. Tirtea, and S. Schiffner. 2014. Privacy and Data Protection by Design—from policy to engineering. ENISA. (2014).

[12] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, and P. Samarati. 2007. Access Control Policies and Languages. *IJCSE* 3, 2 (2007), 94–102.

[13] Herbert Enderton and Herbert B Enderton. 2001. *A mathematical introduction to logic*. Academic press.

[14] K. Fatema, D. W Chadwick, and B. Van Alsenoy. 2012. Extracting Access Control and Conflict Resolution Policies from European Data Protection Law. In *Privacy and Identity Management for Life*. 59–72.

[15] K. Fatema, C. Debruyne, D. Lewis, D. OSullivan, J. P Morrison, and A. Mazed. 2016. A Semi-Automated Methodology for Extracting access control rules from the European Data Protection Directive. In *SPW, 2016 IEEE*. 25–32.

[16] D. Garg, L. Jia, and A. Datta. 2011. Policy auditing over incomplete logs: theory, implementation and applications. In *ACM CCS*.

[17] P. Guarda and N. Zannone. 2009. Towards the development of privacy-aware systems. *Inf. and Sw. Tech.* 51, 2 (2009), 337–350.

[18] V. C Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J Lang, M. M Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone. 2013. Guide to Attribute Based Access Control (ABAC) Definition and Considerations (Draft). Number 800-162 in NIST.

[19] T. Jaeger and J. E. Tidswell. 2001. Practical Safety in Flexible Access Control Models. *ACM Trans. Inf. Syst. Secur.* 4, 2 (May 2001), 158–190.

[20] X. Jin, R. Krishnan, and R. Sandhu. 2012. A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC. In *DBSec (LNCS)*. 41–55.

[21] V. Khatri and C. V. Brown. 2010. Designing Data Governance. *Comm. of the ACM* 53, 1 (2010), 148–152.

[22] E. Lam, J. C. Mitchell, A. Scedrov, S. Sundaram, and F. Wang. 2012. Declarative privacy policy: finite models and attribute-based encryption. In *ACM IHI*.

[23] N. Li and J.C. Mitchell. 2003. Datalog with constraints: a foundation for trust management languages. In *Proc. of PADL*. 58–73.

[24] A. Siena, I. Jureta, S. Ingolfo, A. Susi, A. Perini, and J. Mylopoulos. 2012. *Capturing Variability of Law with Nómos 2*. Springer, 383–396.

[25] M. C. Tschantz, A. Datta, and J. M. Wing. 2012. Formalizing and enforcing purpose restrictions in privacy policies. In *IEEE Symp. on S&P*. 176–190.

[26] F. Turkmen, J. den Hartog, S. Ranise, and N. Zannone. 2015. *Analysis of XACML Policies with SMT*. Springer, 115–134.