# On a Formal and User-Friendly Linguistic Approach to Access Control of Electronic Health Data

Andrea Margheri[1], Massimiliano Masi[2], Rosario Pugliese[1], and Francesco Tiezzi[3]
[1]*Università di Firenze, Viale Morgagni 65, 50134 Firenze, Italy*
[2]*Tiani "Spirit" GmbH, Guglgasse 6, 1110 Vienna, Austria*
[3]*IMT Advanced Studies Lucca, Piazza S. Ponziano 6, 55100 Lucca, Italy*
*andrea.margheri@unifi.it, massi@tiani-spirit.com, rosario.pugliese@unifi.it, francesco.tiezzi@imtlucca.it*

Keywords: Data Security; Electronic Health Records; Access Control.

Abstract: The importance of the exchange of Electronic Health Records (EHRs) between hospitals has been recognized by governments and institutions. Due to the sensitivity of data exchanged, only mature standards and implementations can be chosen to operate. This exchange process is of course under the control of the patient, who decides who has the rights to access her personal healthcare data and who has not, by giving her personal privacy consent. Patients' privacy consent is regulated by local legislations, which can vary frequently from region to region. The technology implementing such privacy aspects must be highly adaptable, often resulting in complex security scenarios that cannot be easily managed by patients and software designers. To overcome such security problems, we advocate the use of a linguistic approach that relies on languages for expressing policies with solid mathematical foundations. Our approach bases on FACPL, a policy language we have intentionally designed by taking inspiration from OASIS XACML, the de-facto standard used in all projects covering secure EHRs transmission protected by patients' privacy consent. FACPL can express policies similar to those expressible by XACML but, differently from XACML, it has an intuitive syntax, a formal semantics and easy to use software tools supporting policy development and enforcement. In this paper, we present the potentialities of our approach and outline ongoing work.

## 1 INTRODUCTION

Since the early 60s, provision of healthcare has gradually evolved towards the so-called *electronic healthcare* (e-Health). E-Health aims at improving the effectiveness of healthcare by resorting to information systems for storing and exchanging patient data in form of Electronic Health Records (EHRs). However, despite the many achievements attained in this field so far, only in recent years governments have started to significantly invest on initiatives and projects aiming at providing (cooperative and interoperable) e-Health solutions to their citizens. For example, in 1996, U.S.A. senators E. Kennedy and N. Kassebaum sponsored the Health Insurance Portability and Accountability Act (HIPAA), with the goal of protecting "health insurance coverage for workers and their families when they change and lose their jobs" and requiring "the establishment of national standards for electronic healthcare transactions and national identifiers for providers, health insurance plans and employers" (Kennedy and Kassebaum, 1996). More recently, the exchange of EHRs among clinics and hospitals has been pursued by the European Commission with the Standardisation Mandate 403 (EU Commission, 2007), aiming at providing a set of established standards for secure EHR access and communication to the ICT industry. An example of project funded in response to this EU directive is epSOS (The epSOS project, 2007). Its goal is to create a pan-European network for exchanging patient summaries and ePrescriptions/eDispensations among participating nations. To this aim, epSOS went far beyond the state-of-the-art, by choosing and contributing to the development of international standards for security, communication protocols, and data representation (see, e.g., (OASIS Security Services TC, 2005; OASIS Web Services Security TC, 2009; The IHE Initiative, 2009; Health Level Seven organization, 2009; Bittins and Masi, 2011)).

The need of defining such standards is of crucial importance in the field of e-Health, due to both the high-level of sensitivity of patients private data, the high impact that a security flaw can have in the life of

patients, and the need of harmonizing different country legislations on patients' privacy consent and enabling their healthcare systems to interoperate.

In this paper, we focus on the problem of defining and implementing access control policies for protecting EHRs from accesses by unauthorized users (i.e. people not explicitly authorized by patients). The challenge comes from the necessity of adopting a standard-based approach and, at the same time, of providing policy designers with tools that permit writing access control policies in a user-friendly way and enable formal reasoning on them. This latter aspect would be a remarkable achievement, since it would permit to formally enforce patients' informed privacy consent by supporting policy designers and system architects (and, indirectly, also patients, healthcare professionals and lawyers) to understand the overall effect of policies and their consequences.

We propose an approach inspired by the OASIS standard *eXtensible Access Control Markup Language* (XACML) (OASIS XACML TC, 2005), which is the *de-facto* standard for defining authorization statements to protect sensible resources. For example, also the epSOS's standardization bodies endorsed its use for guaranteeing a secure and authorized access to patient health data by enforcing the informed privacy consent. However, the adoption of XACML may imply a high cost. Indeed, XACML comes with an XML syntax that makes the task of writing (and, then, understanding) policies difficult and error-prone. Besides, it comes without a formal semantics, rather its specification document is written in prose (i.e., in English) and contains quite a number of loose points that may give rise to different interpretations and lead to different implementation choices. This leaves the difficult task of understanding the full implications of the various choices to the implementers, which should be avoided, since otherwise the portability of XACML policies across different platforms could be considerably undermined.

To address the issues above concerning XACML, we advocate the use of *Formal Access Control Policy Language* (FACPL) (Masi et al., 2012). FACPL is a language for expressing policies with solid mathematical foundations but, differently from XACML, its simple and clear syntax and semantics makes it easy to learn and use. Indeed, FACPL comes with a BNF syntax that can be exploited to create front-ends for user-friendly policy editors. Besides, since FACPL relies on theoretical foundations, it enables formal reasoning on policies to be used, e.g., for formally proving patient consent, law fulfillment and other soundness properties. We have intentionally designed FACPL by taking inspiration from XACML, thus it can express access control policies similar to those expressible by XACML and for different application domains. In this paper, however, we focus on the healthcare domain and present the potentialities of our approach when used in healthcare systems in production where we believe it can contribute to build safer and more reliable systems.

The rest of the paper is organized as follows. Section 2 provides an overview of XACML by resorting to an application to an healthcare scenario borrowed from epSOS. Section 3 presents FACPL and the tools already developed on top of it. Finally, Section 4 concludes the paper by discussing the potentialities of the proposed formal approach.

## 2 THE XACML STANDARD

The XACML standard provides a language for expressing access control policies and access requests, and a framework to evaluate access requests w.r.t. policies and to enforce the authorization decision. The access to each resource is regulated by one or more policies. These are XML documents expressing the capabilities and credentials that a requestor must have for accessing the resource.

Evaluation of XACML requests is performed by two different actors, the Policy Decision Point (PDP) and the Policy Enforcement Point (PEP). The authorization decision is computed by the PDP by checking the matching between values specified in the policies and the corresponding values retrieved from the requests. The decision can be one among *permit*, *deny*, *not-applicable* and *indeterminate*: the first two values have an obvious meaning, while the third means that the PDP does not have any policy that applies to the request and the fourth means that the PDP is unable to evaluate the request. In case of *permit* and *deny*, the PDP can attach some additional actions, called *obligations*, to the decision. The PDP decision is then enforced by the PEP, which authorizes the access request if it understands and can discharge the obligations and, of course, if the PDP decision is positive.

The basic elements of the policy language provided by the standard are the *rules*. A rule specifies the logic for the access control decision by means of an *effect*, that can be either *permit* or *deny*; a *target*, that indicates to which requests the rule applies; a *condition*, that is an expression refining the applicability established by the target; and some *obligations*. Rules are then combined into *policies* that, besides their own target and obligations, specify a *combining algorithm*, which, from the set of rules' decisions, computes what is the decision for a request.

Finally, policies can be combined into *policy sets* that, again, specify a combining algorithm, a target and some obligations.

As an example, consider the policy in Listing 1 expressing (an excerpt of) the patient's privacy consent for the epSOS project. In this project, each role (e.g., physician, nurse, pharmacist) has permissions for performing a certain action (e.g., read or write) on a resource (e.g., code 34133-9 identifies a patient summary) for a certain purpose (e.g., healthcare treatment, statistics, emergency).

Listing 1: epSOS patient consent in XACML (excerpt)

```
<Policy PolicyId="summary" ...
    RuleCombiningAlgId="...:rule-combining-
        algorithm:permit-overrides">
  <Target>
    <AnyOf>
      <AllOf>
        <Match MatchId="...:function:string-equal">
          <AttributeValue> TREATEMENT </AttributeValue>
          <AttributeDesignator
            Category="...:attribute-category:subject"
            AttributeId="...subject:purposeofuse" />
        </Match>
      </AllOf>
    </AnyOf>
    <AnyOf>
      <AllOf>
        <Match MatchId="...:function:string-equal">
          <AttributeValue> physician </AttributeValue>
          <AttributeDesignator
            Category="...:attribute-category:subject"
            AttributeId="...:subject:role" />
        </Match>
      </AllOf>
    </AnyOf>
    <AnyOf>
      <AllOf>
        <Match MatchId="...:function:string-equal">
          <AttributeValue> 34133-9 </AttributeValue>
          <AttributeDesignator
            Category="...:attribute-category:resource"
            AttributeId="...:resource:resource-id" />
        </Match>
      </AllOf>
    </AnyOf>
  </Target>
  <Rule RuleId="rule1" Effect="Permit">
    <Target>
      <AnyOf>
        <AllOf>
          <Match MatchId="...:function:string-equal">
            <AttributeValue> Read </AttributeValue>
            <AttributeDesignator
                Category="...:attribute-category:action"
              AttributeId="...:action:action-id" />
          </Match>
        </AllOf>
      </AnyOf>
    </Target>
    <Condition>
      <Apply FunctionId="...:function:string-subset">
        <Apply FunctionId="...:function:string-bag">
          <AttributeValue> ...:PRD-003 </AttributeValue>
          <AttributeValue> ...:PRD-005 </AttributeValue>
          <AttributeValue> ...:PRD-010 </AttributeValue>
          <AttributeValue> ...:PRD-016 </AttributeValue>
        </Apply>
        <AttributeDesignator
          Category="...:attribute-category:subject"
          AttributeId="...:subject:hl7:permission" />
      </Apply>
    </Condition>
  </Rule>
  <Rule RuleId="ruledeny" Effect="Deny">
  </Rule>
  <ObligationExpressions>
    <ObligationExpression FulfillOn="Permit"
      ObligationId="...:obligation:log">
      <AttributeAssignmentExpression
        AttributeId="...:attribute:subject">
        <AttributeValue DataType="#string">
          <AttributeDesignator
            Category="...:attribute-category:subject"
            AttributeId="...:subject:physician-id" />
        </AttributeValue>
      </AttributeAssignmentExpression>
      <AttributeAssignmentExpression
        AttributeId="...:attribute:resource">
        <AttributeDesignator
          Category="...:attribute-category:resource"
          AttributeId="...:resource:resource-id" />
      </AttributeAssignmentExpression>
    </ObligationExpression>
    <ObligationExpression FulfillOn="Deny"
      ObligationId="...:obligation:mail">
      <AttributeAssignmentExpression
        AttributeId="...:attribute:mailto">
        <AttributeDesignator
          Category="...:attribute-category:resource"
          AttributeId="...:resource:resouce-id:email" />
      </AttributeAssignmentExpression>
      <AttributeAssignmentExpression
        AttributeId="...:attribute:text">
        <AttributeValue DataType="#string">
          Your medical record has been requested by
              EpSOS subject
        </AttributeValue>
      </AttributeAssignmentExpression>
    </ObligationExpression>
  </ObligationExpressions>
</Policy>
```

The policy specifies a *subject* and a *resource* in its target, according to which the policy applies to requests issued by a physician with the purpose of accessing to a resource with a code identifier 34133-9 for an healthcare TREATMENT. If these capabilities are met, the rules enclosed in the policy are evaluated. The first rule has effect Permit if the requestor aims at performing a Read action and has at least the permis-

sions PRD-003, PRD-005, PRD-010, and PRD-016 for accessing the resource. The second rule has always effect Deny and is combined with the previous one in such a way that if the first rule evaluates to Permit then the policy permits the access to the resource, otherwise the access is denied (combining algorithm *permit-overrides*). Concerning the obligations, if an access request is evaluated as *permit*, a log entry about the requestor and the resource is recorded in the system, while if the decision is *deny* an e-mail is sent to the patient to notify the (unauthorized) attempt.

As witnessed by the code in Listing 1, the XML syntax makes it hard to identify the relevant policy features, especially for users not familiar with XACML but that anyhow need to understand the overall effect of combined policies.

# 3 THE FACPL APPROACH

The language FACPL provides a manageable alternative syntax to XACML through a BNF-like grammar. It is indeed a tiny language that is easy to learn and capable of expressing XACML policies in a very compact and readable way. For example, the XACML policy reported in Listing 1 can be rewritten in FACPL as shown in Listing 2.

Listing 2: epSOS patient consent in FACPL

```
<permit-overrides;
 target:
    string-equal("TREATEMENT",subject/purposeofuse)
    and string-equal("physician",subject/role)
    and string-equal("34133-9",resource/resource-id);
 rules:
    (permit;
     target: string-equal ("Read",action/action-id);
     condition:
        string-subset(string-bag("PRD-003","PRD-005",
                                 "PRD-010","PRD-016"),
          subject/hl7:permission); obl:;)
    (deny; target:; condition:; obl:;);
 obl:
  (permit; M;
   log(subject/physician-id,resource/resource-id))
  (deny; M;
   mail(resource/resource-id.email,"Your medical record
        has been requested by EpSOS subject"));>
```

The above code demonstrates that the FACPL notation, compared to that of XACML, is simpler and more intuitive. It is worth noticing that a FACPL policy is denoted by a structured term, surrounded by angular brackets, that specifies a combining algorithm, a target, a set of rules, and a set of obligations. A FACPL rule is surrounded by round brackets and has a similar structure, but for the algorithm replaced by the effect and for the enclosed rules replaced by a condition. Notably, FACPL targets are not defined according to the structure <AnyOf>-<AllOf>-<Match>, rather are simply boolean expressions (i.e. conjunctions and disjunctions of matching functions). Finally, FACPL obligations are triples specifying an effect, a type (M for mandatory and O for optional[1]), and an action to be performed by the PEP.

The formal semantics of FACPL policies is given in a denotational style, i.e. it is defined by a function $[\![\cdot]\!]_R$ that, given a policy/policySet and a set $R$ of access requests, returns a *decision tuple* of the form

$$\begin{array}{l}\langle\ \text{permit} : \{< r_i, Obl_i >\}_{i\in I_p},\\ \quad \text{deny} : \{< r_j, Obl_j >\}_{j\in I_d},\\ \quad \text{not-applicable} : R_n,\\ \quad \text{indeterminate} : R_i\ \rangle\end{array}$$

where $\{r_i\}_{i\in I_p}$, $\{r_j\}_{j\in I_d}$, $R_n$ and $R_i$ form a partition of $R$ according to the results of the requests' evaluation. Requests in the permit and deny sets may have attached sets of obligations. A decision tuple represents the evaluation result computed by the PDP and passed as input to the PEP for its evaluation. The final result does not contain obligations, as they are discharged during the enforcement process by the PEP.

An example of a FACPL request for accessing a patient summary is reported in Listing 3.

Listing 3: Patient summary request

```
Request:{Request1
  (subject/purposeofuse,"TREATEMENT")
  (subject/role,"physician")
  (subject/physician-id,"jh1234")
  (subject/hl7:permission,"PRD-003")
  (subject/hl7:permission,"PRD-005")
  (subject/hl7:permission,"PRD-010")
  (subject/hl7:permission,"PRD-016")
  (resource/resource-id,"34133-9")
  (resource/resource-id.email,"patient@mail.com")
  (action/action-id,"Read")
}
```

The request is made by the physician with personal identifier jh1234 for reading the health record of a patient for treatment purposes. Given this request and the policy in Listing 2, the FACPL semantics returns a decision tuple where the permit set consists of a pair formed by the request and the obligation requiring the log update.

We have seen so far that FACPL permits developing access control policies which are intuitive and easy to read, write and understand. In order for FACPL to be actually used for dealing with real systems, a software architecture for evaluating access re-

[1] In XACML jargon, the optional obligations are called *advices*.

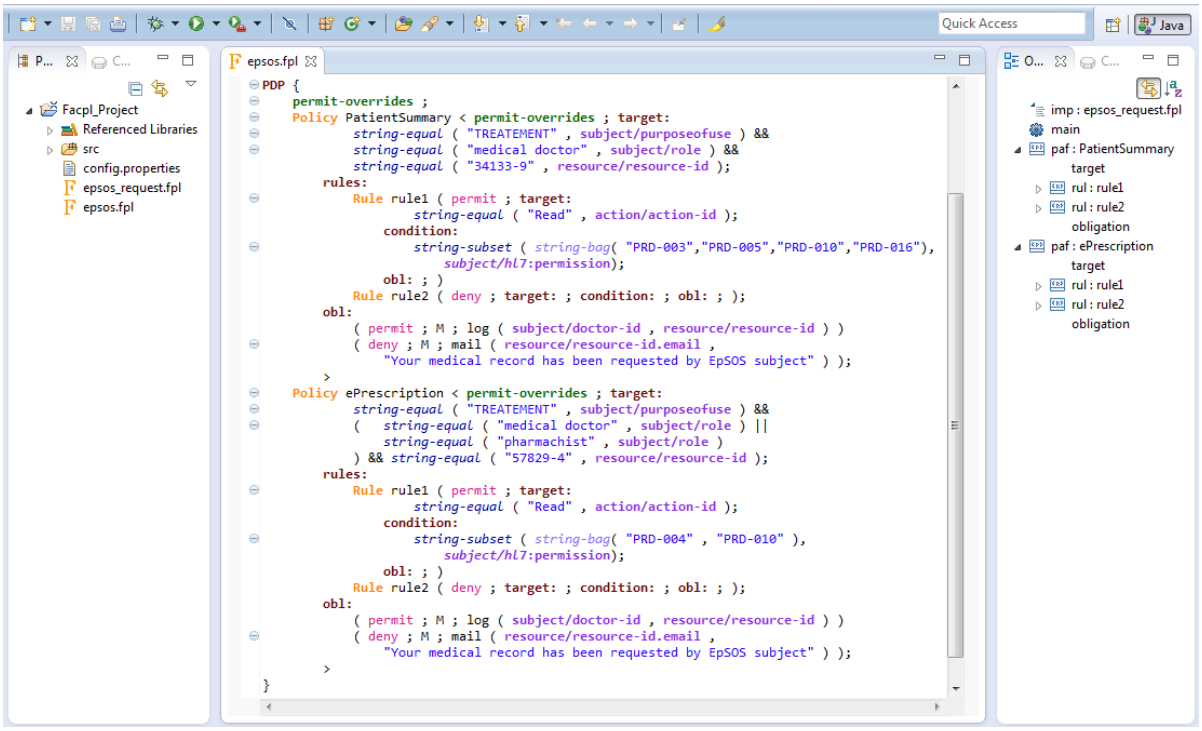


Figure 1: FACPL development environment

quests and software tools supporting policy development are necessary.

Therefore, FACPL semantics has been implemented in Java as a sort of "compiler" that transforms FACPL policies into Java classes following the semantics rules defined in (Masi et al., 2012). Similarly, an access request is compiled into a Java[2] class. A policy decision is then computed by executing the generated policy code with the request code passed as parameter to an entry method.

Moreover, FACPL has been equipped with a dedicated powerful Integrated Development Environment (IDE), implemented as an Eclipse[3] plug-in[4] by means of the Xtext framework[5]. The FACPL IDE supports users in the process of developing policies by means of such features as code auto-completion, syntax checks, generation of XACML code and, of course, generation of Java code by means of the library mentioned above. A screenshot of the IDE, showing two epSOS's policies for ePrescription and patient summary, is reported in Figure 1.

[2]Java 1.6 or higher. `http://www.java.com`

[3]Eclipse Juno 4.2. `http://www.eclipse.org/`

[4]To install the FACPL plug-in it is sufficient to open Eclipse, click on *Help>Install New Software...* and follow the wizard, by specifying the URL `http://rap.dsi.unifi.it/xacml_tools/facpl/FacplEclipsePlugin/` when requested.

[5]Xtext 2.3. `http://www.eclipse.org/Xtext/`

# 4 CONCLUDING REMARKS

We presented the use of a policy language with solid mathematical foundations for the definition of patients' privacy consent in healthcare projects. Our approach relies on FACPL, a language inspired by OASIS XACML 3.0. We have seen that FACPL formal semantics is more helpful than XACML natural language descriptions for understanding the behavior of the authorization process. Besides, it is an effective guide for the implementation of the language and paves the way for the development of analysis tasks, such as policy comparison, policy verification and policy querying.

For example, equivalences and preorders among different policies could be a first target achievable by means of FACPL formal semantics. This is particularly helpful in the scope of the epSOS project, which actually counts 23 participating countries and some new non-EU countries are foreseen. From a technological standpoint, epSOS' challenges were mostly related to the harmonization of all the European countries' legislations for handling patients' privacy consent. In fact, the goal of epSOS in this area was to follow the suggestions of the EU Article 29 WP (Directorate-General of Justice, 2010), and to let each member state define patients' consent following the national legislation. These facts have been reflected in the establishment of the epSOS *Circle of*

*Trust (CoT)*, a Service Level Agreement (SLA) signed by countries in order to securely share data in the epSOS network. In fact, due to the high level of sensitivity of patient's private data exchanged, and to the high impact that a security flaw can have in terms of breaking the CoT, only mature and well-known standards are allowed in the scope of the project. Therefore, all the policies in epSOS are written using XACML.

FACPL mathematical foundations enable each participating nation to formally prove the SLA fulfillment of its policies for privacy consent (written in agreement with the respective local legislation) thus guaranteeing the country's participation in the CoT. In fact, various analysis on the policies can be performed. For example, equivalence analysis, which is already enabled by FACPL formal semantics, could be used to prove that each policy follows the epSOS requirements expressed by means of a continent-wide referenced XACML policy.

To reduce lacking design, a constraint system for validating the behavior of the authorization process could be defined. To this aim, we will design a practical way to define safety and coverage properties over an access control system model. These properties are useful for one of the most difficult problem for policy-based access control systems: to understand the overall effect of combined policies evaluation or the unexpected consequences on obligation discharging. For example, inaccurate policies might unintentionally override patient consent, thus possibly authorizing an intruder to access. This type of analysis is known as *change-impact* analysis and is usually carried on by means of a formal logic, as e.g. a description logic. It has many practical applications, among which examining differences between policy versions and identifying redundant policies.

To assess the effectiveness of FACPL and its related tools and techniques, we plan both to experiment with healthcare systems in production and to also consider case studies from other domains.

## ACKNOWLEDGEMENTS

This work has been partially sponsored by the EU project ASCENS (257414).

## REFERENCES

Bittins, S. and Masi, M. (2011). Cross community fetch. http://www.ihe.net.

Directorate-General of Justice (2010). Article 29 working party: documents.

EU Commission (2007). Mandate 403 en: Standardisation mandate addressed to cen, cenelec and etsi in the field of information and communication technologies. Technical report, European Commission Enterprise And Industry Directorate-General. http://www.ehealth-interop.eu.

Health Level Seven organization (2009). Hl7 standards. http://www.hl7.org.

Kennedy, E. and Kassebaum, N. (1996). Health insurance portability and accountability act. Technical report, US Congress. http://www.cms.gov/HIPAAGenInfo.

Masi, M., Pugliese, R., and Tiezzi, F. (2012). Formalisation and implementation of the xacml access control mechanism. In Barthe, G., Livshits, B., and Scandariato, R., editors, *ESSoS*, volume 7159 of *Lecture Notes in Computer Science*, pages 60–74. Springer.

OASIS Security Services TC (2005). Assertions and protocols for the OASIS security assertion markup language (SAML) v2.02. http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf.

OASIS Web Services Security TC (2009). Cross enterprise security and privacy authorization profile for xacml for healthcare.

OASIS XACML TC (2005). eXtensible Access Control Markup Language (XACML) version 2.0. http://docs.oasis-open.org/xacml/2.0/XACML-2.0-OS-NORMATIVE.zip.

The epSOS project (2007). An european ehealth project. http://www.epsos.eu.

The IHE Initiative (2009). Basic Patient Privacy Consent. http://www.ihe.net.