Dynamic Management of Security Policies in PrivOrBAC

Jihane EL MOKHTARI[1], Anas ABOU EL KALAM[2], Siham BENHADDOU[3], Jean-Philippe LEROY[4]
LISER Laboratory, IPI, Paris, France[1, 4]
LRI Laboratory, ENSEM, Casablanca, Morocco[1, 3]
TIM Laboratory, ENSAM, Marrakesh, Morocco[2]

***Abstract*—This article is a continuation of our previous work on identifying and developing tools and concepts to provide automatic management and derivation of security and privacy policies. In this document we are interested in the extension of the PrivOrBAC model in order to ensure a dynamic management of privacy-aware security policies. Our approach, based on smart contracts (SC) and the WS-Agreement Specification, allows automatic agents representing data providers and access requesters to enter into an access agreement that takes into consideration not only service level clauses but also security rules to protect the privacy of individuals. Our solution can be deployed in such a way that no human intervention is required to reach this type of agreement. This work shows how to use the WS-Agreement Specification to set up a process for negotiating, creating and monitoring Service Level Agreements (SLAs) in accordance with a predefined access control policy. This article concludes with a case study accompanied by a representative implementation of our solution.**

***Keywords*—Access control; privacy; PrivOrBAC; PrivUML; smart contract; WS-agreement**

I. Introduction

Our work is related to the management and development of security and privacy policies. Our goal is to be able to identify security and privacy needs and integrate them early into the design of complex systems. This practice has yet to be widely adopted; generally, security requirements are expressed separately in the form of complementary modules and integrated late into the systems. The principal difficulty which designers encounter concerns the methods and tools used to design functional and security models, as well as to describe the business demands which the system must satisfy. While functional design methods have become increasingly effective so as to allow one to design models formal enough to provide a base of refinement down to code, security design methods are not yet at the same level.

As a first step, we focused our efforts on defining the security policy and modeling tools allowing the integration of privacy protection into the system's design models. It is in this context that we relied on PrivOrBAC [1], an access control model dedicated to the protection of privacy. It is based on the OrBAC model [2] which introduced the notions of organization, context and object views as attributes of access management. Thus, access permission is granted to a role within an organization to perform an activity on an object view in a specific context. PrivOrBAC takes this logic and enriches it with attributes specific to privacy. Access permission in PrivOrBAC is therefore granted to a role in an organization to perform an activity, justified by a purpose in a specific context, on a granular object view following the consent of its priority. To model PrivOrBAC, we have proposed the new PrivUML metamodel [3] which is an extension of the UML language, and which makes it possible to integrate all the attributes of PrivOrBAC necessary for the protection of privacy into a class diagram. Our implementation of PrivUML under XACML (eXtensible Access Control Markup Language) [4] allowed us to set up a privacy-aware security policy. However, at this level, our solution remains static.

The second part of the article is therefore devoted to automating the management of this policy to make the access control process dynamic. As it stands, any modification to be made in the security policy requires manual interventions, which diverge from the aspirations of complex systems known by their high rates of interactivity and which are therefore penalized by the obligation of human involvement in this type of task. The idea is to put in place intelligent agents capable of replacing human skills. These agents will not only update the security policy, but also manage the entire upstream process responsible for investigating the incoming request and negotiating, creating and monitoring the access agreements in accordance with the policy. It is in this sense that we propose to set up smart contracts managed through the WS-Agreement Specification [5].

We organize the remainder of this article into three sections. The first is dedicated to the presentation of our previous work. We devote the second section to the mechanism for automating the management of our security policy based on smart contracts and the WS-Agreement Specification, and then we present a discussion of our contribution in the third section.

II. Previous Work

Our previous work, [3] and [6], focused on providing the means and tools to integrate the principles of privacy protection into IT systems. To achieve our objectives, we have opted, throughout this work, for a certain number of choices which we present in what follows.

1) PrivOrBAC (Privacy-Aware Organization Based Access Control: In OrBAC model [2], the organization is the central component; privileges are not applied directly to subjects, they are assigned to roles within an organization [7]. Permission is granted for a role to perform a subset of activity on a number of views of an organization's objects in a specific

context. PrivOrBAC [1] extends the OrBAC model to covers Privacy management requirements. The first change consists in setting up a hierarchy of views to better manage the granularity of the data consulted. The same view of objects can have different levels of detail from one organization to another. Regarding consent and purpose, they were introduced as new attributes of the context. The organization in PrivOrBAC continues to play its role as a central component which makes it possible to configure the access policy according to the other abstract entities (Role, Activity, View and Context). The role and activity entities behave the same as in OrBAC; privileges are assigned to a role within an organization in order to perform an activity.

2) PrivUML Metamodel: In accordance with the MDA approach (Model Driven Architecture), we have defined our CIM model (Computation Independent Model) composed of the pair (ends, means) corresponding to the protection of privacy through the PrivOrBAC access control model. The next step is to go to the second level of MDA which is the PIM (Platform Independent Model). Translating CIM to PIM requires finding the modeling tool capable of integrating all of our Privacy requirements into the model. After a study on the various security modeling tools focused on modeling access control requirements (SecureUML [8], UMLSec [9], PaML [10] and Privacy UML Profile [11]), we noted that none of these tools is completely adapted to our needs. Based primarily on roles and enriched by purpose, they do not, in their current state, allow the PrivOrBAC access control to be modeled. We therefore proposed PrivUML [3], which is an extension of UML enriched by the following concepts:

- The access modalities allowing to express the authorization of access, the refusal or the obligation.
- The hierarchy of object views to control the granularity of the data to be consulted.
- The purpose justifying the access request.
- The explicit consent of the data owner granted on the basis of the purpose.

We have therefore set up a meta-model capable of integrating security policies, meeting privacy requirements, from the design phase of the system. In PrivUML, an access request is refused or accepted with or without conditions to a subject, having a role in an organization, to perform an action justified by a purpose on a specific set of data whose owner has given his prior consent.

III. Automating of the Management and Protection of Privacy

The work presented in the previous section makes it possible to set up a privacy policy in a static manner. Any new changes require manual updating of this security policy. Complex systems therefore have to dedicate personnel to this task, the cost of which increases according to the volumes of their interactions. This also increases the risk of error and data leaks. Take for example the French Health Management System (FHMS) which manages all the health information of adherents. Any medical entity not listed in the system must first go through the stage of negotiating a new agreement access, which must comply with the security policy in force, and which will result in updating the security policy of FHMS. The implementation of smart contracts is necessary to ensure automatic management of requests for negotiation and establishment of access contracts. This management method will be piloted by intelligent agents responsible for negotiating contracts as well as updating security policies. In what follows, we will define the notion of smart contract, then we will present the solution chosen for the management of smart contracts, and finally we will explain through a case study how we ensured the automation of management and the protection of privacy in an interactive system based on the elements mentioned above.

A. Smart Contract

The concept of a smart contract first appeared in the 1990s and has evolved over the years. Programmers tend to see it as a solution that replaces traditional contracts and contract law [12]. Another vision of the smart contract considers it as a mechanism to express calculations on a blockchain [13]. The author in [14] defines the smart contract as an automatable and enforceable agreement: automatable, although some parts may require human intervention and control; and enforceable either by legal application of rights and obligations, or by tamper-proof execution of computer code. For our work, we agree with [15] on its definition of the smart contract and consider it as the formalization of an agreement, the terms of which are automatically applied by relying on a transaction protocol, while minimizing the need for an intermediary.

B. Access Agreement

Data providers and consumers operate in a dynamic environment governed by a set of rules, conditions, obligations and guarantees formalized in a contract or access agreement. Several solutions exist for the management of this type of electronic contract, from negotiation to monitoring. Among these solutions we find:

- WSLA (Web Service Level Agreement): this specification, created by IBM in 2003 [16], allows the creation and monitoring of SLA (Service Level Agreement) contracts. WSLA defines a flexible and extensible language to allow consumers and providers of web services to define and specify a set of parameters such as technical-functional descriptions of web services, supplier commitments, etc.
- WSOL (Web Service Offering Language): this specification, based on XML language, allows suppliers to define several service levels or SLOs (Service Level Objectives) for the same web service. A consumer can therefore choose the desired web service and select the SLOs that interest him according to the level of service that suits his needs [17].
- WS-Agreement (Web Service Agreement): this specification defines a protocol and a language for negotiating, renegotiating, creating and monitoring bilateral access contracts (between consumer and

provider) in distributed systems [5]. The WS-Agreement Specification and the WS-Agreement Negotiation Protocol [18] are the only specifications standardized and accepted by a large community [19], [20]. Therefore, our study will be based on this specification to establish the protocol for negotiating and creating smart contracts that frame access agreements.

1) WS-Agreement Specification: WS-Agreement is a specification that enables exchanges based on XML language between providers and consumers of web services. An agreement is an XML file created from a template. The agreement and the template have the same structure. An agreement is made up of three sections:

- Name: this section contains the name and ID of the agreement.
- Context: the context contains meta-information about the agreement such as the identifiers of the initiator and the responder of the agreement, the identifier of the template that served as the basis for creating the agreement, references to other agreements, the period of validity, etc.
- Terms: The terms of an agreement include terms of service and, optionally, guarantee terms that define the constraints applicable to the services set out in the agreement. A service term is made up of several sections of Service Descriptions Term (SDT), Service References (SR) and Service Properties (SP). A guarantee term for a service consists of a section describing the scope of the service, a section describing the service level objectives (SLO) and possibly a section describing the business values of the service.

2) WS-Agreement Negotiation Protocol: Automating negotiations requires formalizing the definition of each term of the access contract. Much work has been done in recent years to automate contract negotiations in various fields such as connected objects ([19], [21], [22]), cloud computing platforms ([23], [15], [24], [25]) as well as distributed environments ([20], [26], [27], [28]). The WS-Agreement Negotiation Model [18], shown in Fig. 1, consists of three layers: the negotiation layer, the agreement layer, and the service layer.

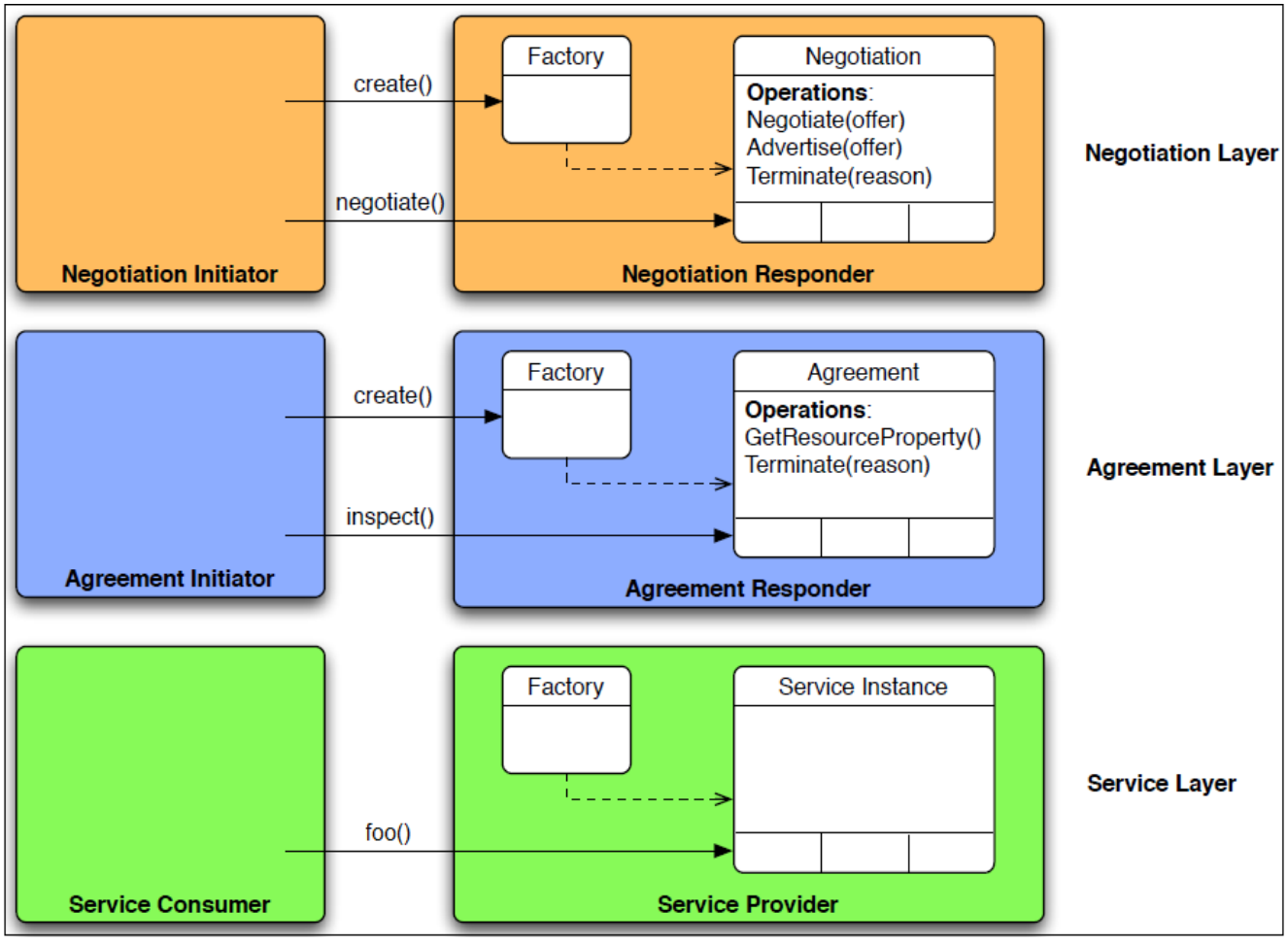

Fig. 1. WS-Agreement Negotiation Model [18].

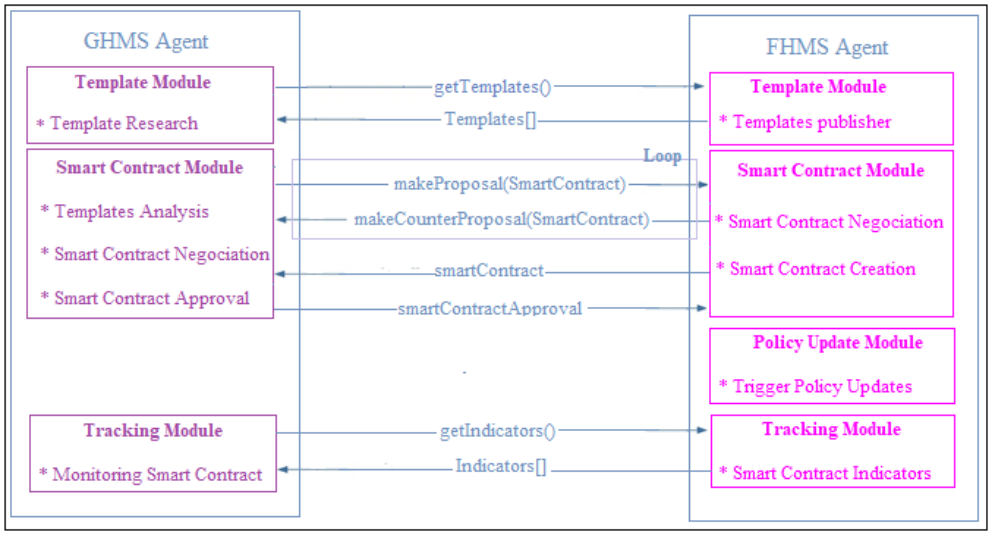

Fig. 2. Protocole De Négociation De Smart Contrat.

The negotiation layer provides a protocol and language for negotiating offers that indicate the willingness of both parties to enter into a subsequent agreement. The negotiation process includes the exchange of negotiating offers and counter-offers until an agreement is reached between the two parties. The agreement layer, on the other hand, provides a protocol and language that provides the basic functionality to create and monitor agreements. Finally, the service layer defines the services offered by the data provider. The execution of the service on the service layer is governed by the agreement layer. Fig. 2 illustrates the protocol for negotiating a smart contract between the French Health Management System (FHMS), as data provider, and the German Health Management System (GHMS) as data consumer.

The consumer agent (GHMS) initiates the negotiation process by requesting the retrieval of the services modeled, described and published by the data provider, FHMS, by calling the public "getTemplates()" method. Once the templates have been received, the consumer agent analyzes them and chooses the one that corresponds to its needs. It then fills in the fields and builds his offer, then sends it to the provider agent via the "makeProposal(smartContract)" method. Upon receipt of this proposal, the provider agent checks the values entered and their compliance with the predefined constraints of contracts creation. In case of invalidity of the proposal made by the consumer agent, the provider agent is responsible for adapting the non-conforming values and making a counter-offer by calling the "makeCounterProposal(smartContract)" method. The negotiation process can see several back and forth between the two agents until a common agreement is reached or the failure following the consumption of all the allowed iterations. The common agreement is reflected in the creation of the smart contract signed by the provider agent and sends it to the consumer agent who in turn signs it and communicates his proof of approval. Then, the provider agent updates the security policy according to the access rules negotiated in the contract. It also provides the consumer agent with the possibility of monitoring his contract.

3) WS-Agreement Implementation: Many projects implement the WS-Agreement Specification and Protocol to set up access agreements. Some frameworks, such as WSAG4J [29] and SLA-Framework [30], make it possible to simplify this implementation. WSAG4J is a generic implementation of the WS-Agreement Protocol. It supports common functionality to create and monitor agreements generically and allows users to quickly create and deploy services based on WS-Agreement Specification. WSAG4J follows a declarative approach to support and manage the entire lifecycle of an agreement, from the definition of an agreement template, through the deployment of models in *Factories*, to the management of the agreement. SLA-Framework, as for it, is another implementation of the WS-Agreement Specification. It is an open source project that helps manage the lifecycle of access agreements (negotiation, creation and monitoring agreements). Currently in version V1.1, this framework only supports one-shot negotiation. The core SLA-Framework provides a language and protocol to define and advertise the capabilities of service providers in SLA templates, create agreements based on the templates, and track compliance with agreements at runtime.

C. Case Study

France's national health security organization manages the medical records of all adherents of this state public service. All information relating to an adherent is listed in the French Health Management System (FHMS). This is personal information that the owner has designated as private, for example the Person of Confidence to Contact (PCC) or medical order that describes everything related to the Patient's Care History (PCH). All access to this information is controlled by the system which grants or denies it according to the predefined access control policy. A national or foreign medical entity can therefore issue a request for access to the system to consult all or part of a medical file. These requests are normally issued by entities already recognized by the system, but it is quite possible for a new organization (new private

sector clinic, foreign hospital, new research laboratory, etc.) to request access. In this case, a collaborative access contract is required. Smart Agents, representing consumers and data provider, handle the negotiation and commissioning of the smart contract using the web services responsible for accessing the data. These data concern the personal information of a patient (surname, first name, age, address, telephone number, etc.) as well as all medical information relating to his/her state of health and his/her care path (results of analysis, treatments, chronic diseases, allergies, blood pressure and heart rate measurements, etc.), which are collected by the various portable and implantable sensors of the WBAN (Wireless Body Area Networks) or entered directly by doctors and subsequently stored in the Cloud (Fig. 4) in an encrypted format. The encryption is performed by a local server based on the access control rules specific to each type of data. Fig. 3 shows an excerpt from the access control policy applied to encrypt the PCC and PCH of Mr. Jean Dupont.

Mr. Jean Dupont, a French traveling in Germany, is transported to the emergency room of the K Hospital in the city of Berlin following a serious accident. Dr. Karl Schmidt, who takes care of him in the emergency department, needs to consult the information on the patient's care path as well as his contact details. Dr. Schmidt connects to the German Health Management System (GHMS) and asks through it to establish a link with its French equivalent (FHMS) in order to retrieve information from Jean Dupont. The GHMS thus sends a request to establish access to the FHMS which in return requests a certain amount of information relating to the requester and the access purpose. Following the sending of this information, the FHMS and GHMS negotiate and enter into a contract which grants the right of access to users attached to the GHMS governed by the access control policy in force in the FHMS. A token is therefore provided by the GHMS to Dr. Schmidt over a fixed period as well as the identifier of the established access contract. The FHMS, for its part, updates its security policy with the attributes relating to the new contract. Dr. Schmidt connects via the token to the PEP of FHMS, which is the user entry point to the system. The PEP transmits the access request to the PDP which studies it based on the input data and access control policies stored in the PAP database and grants Dr. Schmidt access to Jean Dupont's PCH and PCC.

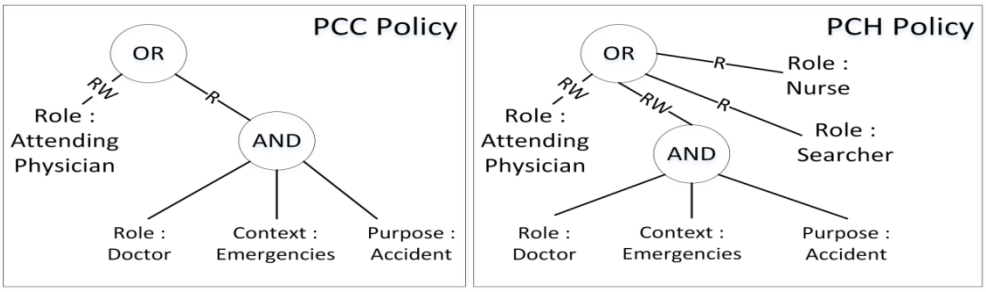

Fig. 3. PCC and PCH Access Control Policy of Mr. Jean Dupont.

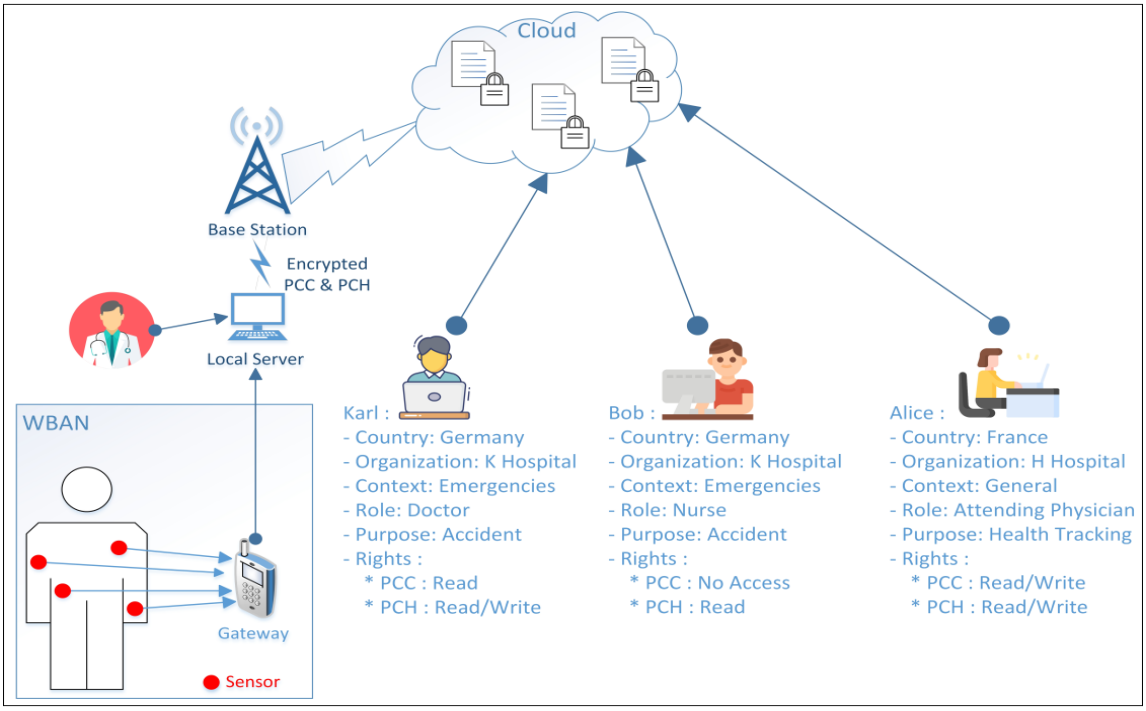

Fig. 4. Case Study Illustration.

Bob, a nurse in the emergency department at K Hospital in Germany, can also connect to the PEP thanks to the signed contract covering all staff attached to the GHMS. He made the same request as Dr. Karl Schmidt but did not obtain the same authorizations. According to the access policy in Fig. 3, Bob cannot access the PCC. Although the context (Emergency) and the purpose (Accident) for the access request matches PCC policy, Bob does not have the "Doctor" role and therefore does not meet all the criteria for access. However, he can have read-only access to PCH data in accordance with its policy. Finally, Alice, Jean Dupont's attending physician attached to the FHMS, benefits from permissions to access, read and write to all of his file (PCC and PCH), granted on the basis of her role (attending physician) in organization (H Hospital).

D. Solution Implementation

Fig. 5 shows the sequence diagram of the implementation of our case study.

This sequence diagram summarizes the technical steps taken so that Dr. Karl Schmidt can access Jean Dupont's medical (PCH) and private (PCC) data. Dr. Karl Schmidt logs into his session in the German Health Management System (GHMS) and formulates a need for access to the French Health Management System (FHMS). The GHMS checks whether a smart contract (SC) already exists and makes it possible to respond positively to the need of Dr. Schmidt. In the case of the absence of a SC or the non-coverage of the need by the existing SCs, the GHMS initiates the automatic process of establishing a new SC with the FHMS as already detailed in the description of Fig. 2. The existence of a SC or the establishment of a new SC triggers the generation of a token by the GHMS which communicates it in addition to the identifier of the SC to Dr. Schmidt. These elements allow the doctor to activate a collaborative session at the FHMS and to formulate an access request to its PEP (Policy Enforcement Point) in which he specifies the attributes necessary to process his request. On the basis of this information, the PEP constructs a XACML request which it transmits to the PDP (Policy Decision Point) to calculate the decision.

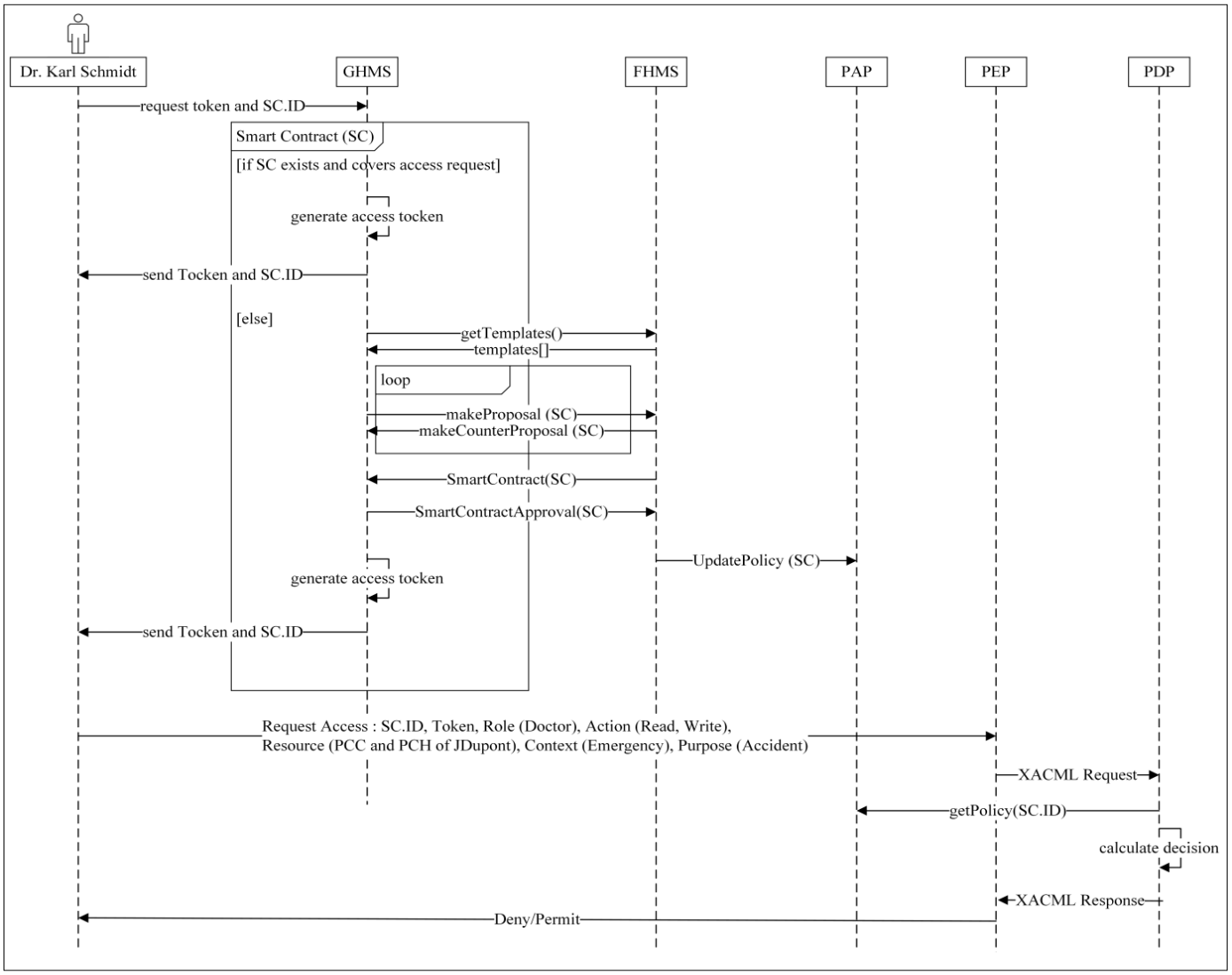

Fig. 5. Negotiation and Establishment of the Smart Contract and Collaborative Access to Data.

Here is an example of a XACML request:

```
<?xml version='1.0' encoding='UTF-8'?>
<Request>
  <Subjects>
    <Subject>
      <AttributeId>RSO.identity</AttributeId>
      <AttributeValue>Karl Schmidt</AttributeValue>
    </Subject>
    <Subject>
      <AttributeId>RSO.role</AttributeId>
      <AttributeValue>Doctor</AttributeValue>
    </Subject>
  </Subjects>
  <Resources>
    <Resource>
      <AttributeId>VOO.Owner</AttributeId>
      <AttributeValue>Jean Dupont</AttributeValue>
    </Resource>
    <Resource>
      <AttributeId>VOO.Identity</AttributeId>
      <AttributeValue>PCC</AttributeValue>
    </Resource>
  </Resources>
  <Actions>
    <Action>
      <AttributeId>AO.Type</AttributeId>
      <AttributeValue>Read</AttributeValue>
    </Action>
  </Actions>
  <Environments>
    <Environment>
      <AttributeId>Context</AttributeId>
      <AttributeValue>Emergency</AttributeValue>
    </Environment>
    <Environment>
      <AttributeId>Purpose</AttributeId>
      <AttributeValue>Accident</AttributeValue>
    </Environment>
  </Environments>
</Request>
```

In our case study, Dr. Karl Schmidt receives permission to read and write to Jean Dupont's treatment path. He has also been granted read-only access permission to his private data in accordance with the "PCC: Access_modality" policy in which the affected person gives his consent to consult this data by a doctor in the event of an accident:

```
<Policy PolicyId="PCC:Access_modality"
RuleCombiningAlgId="&rule-combine;permit-
overrides">
  <Target>
  <!-- this policy concerns the reading of PCC
patient J. Dupont who consented to doctors to
access private information for specific purposes-->
    <Resources>
      <Resource>
        <AttributeId>VOO.Identity</AttributeId>
        <AttributeValue>PCC</AttributeValue>
      </Resource>
      <Resource>
        <AttributeId>VOO.Owner</AttributeId>
        <AttributeValue>JeanDupont</AttributeValue>
      </Resource>
    </Resources>
    <Actions>
      <Action>
        <AttributeId>AO.Type</AttributeId>
        <AttributeValue>Select</AttributeValue>
      </Action>
    </Actions>
  </Target>
  <Rule RuleId="PCC:Access_PCC" Effect="Permit">
    <Target>
      <Resources>
        <Resource>
         <AttributeValue>Consent.Y</AttributeValue>
        </Resource>
      </Resources>
      <Actions>
        <Action>
          <AttributeId>AO.Type</AttributeId>
          <AttributeValue>True</AttributeValue>
        </Action>
      </Actions>
    </Target>
    <Condition FunctionId="function:and">
      <Apply FunctionId="&function;string-equal">
        <AttributeId>RSO.role</AttributeId>
        <AttributeValue>doctor</AttributeValue>
       </Apply>
       <Apply FunctionId="&function;string-is-in">
         <AttributeId>Purpose.title</AttributeId>
         <AttributeValue>accident</AttributeValue>
      </Apply>
    </Condition>
  </Rule>
</Policy
```

We have therefore seen through this example of an implementation how a subject (Dr. Karl Schmidt) having a role in an organization (Doctor attached to the GHMS) succeeded in exercising an activity (reading and / or writing) on views of objects managed by another organization (PCC and PCH of Jean Dupont, member of the FHMS) in a specific context (Emergencies) and for a specific reason (Accident) following the consent of the data owner (Jean Dupont). This case study and this implementation therefore demonstrates how we were able to implement the process of automating the management of security and privacy policies based on PrivOrBAC using smart contracts and the WS-Agreement Specification.

IV. Discussion

Complex systems allow entities to exchange information even though it is unknown to each other. This type of exchange is generally governed by access contracts negotiated in advance between the providers and consumers of the data. Today, this "static" approach is no longer appropriate. According to [31], Web Services are widely used in the automation of decision making while the use of WS-Agreement asserts itself in the automation of service level agreements (SLA). Automating access negotiation processes and integrating security rules into SLAs seems to be a good solution to make access control dynamic. Stankov et al. [32] consider that SLAs can be used as an instrument to build trust between providers and consumers of services in Cloud Computing platforms even before a relationship is established between them. The WS-Agreement Specification appears to be one of the most promising solutions to achieve this goal. Ludwig et al. [33] use WS-Agreement to negotiate SLA contracts to share resources in Grid Computing. In this same context, Smith et al. [34] also use WS-Agreement and Web Services to set up a security configuration mechanism according to the requirements provided by the user such as the configuration of resources and the quality of service, as well as

security parameters such as the level of encryption and sandboxing. In our work, we use WS-Agreement and Web Services to extend the PrivOrBAC model by incorporating access agreements that cover security rules to protect Privacy.

The WS-Agreement Specification can therefore be combined with access control models to allow data providers and consumers to specify their security rules. This is the case of Li et al. [35] which propose to integrate the attributes and their values in an SLA contract through OrBAC rules. This approach consists in that, thanks to the negotiation protocol of WS-Agreement, the providers and consumers of services exchange offers and counter-offers until an agreement is reached. However, this approach does not explicitly implement the WS-Agreement Specification to eliminate human intervention when discovering services and negotiating terms of the agreement. Our solution, on the other hand, makes it possible to respond to this problem and allows intelligent agents (or connected objects) to discover each other, discover services, negotiate service level clauses as well as clauses related to access security. However, the explicit management of trust based on transaction history and the management of inferences are, among others, two aspects that still needed to strengthen the model and avoid derivation. Indeed, the application of security rules seems to be a good solution to block unauthorized access and even more if these rules are negotiated automatically, but the dynamic reinforcement of these rules throughout the life of the information system is an essential requirement in today's systems.

V. Conclusion and Perspectives

A. Conclusion

In this article, we first started by explaining our directions in terms of privacy protection. We have given a definition to this concept and we have subsequently explained the approach followed, the choices adopted and the contributions made in our previous work to achieve the integration of privacy protection from the early design phases of complex systems. This integration in its static form is not entirely satisfactory to large organizations equipped with systems with high volumes of interactivity. The most awaited by these organizations is a dynamic administration of security policies and privacy protections. It is in this context that we have proposed in this article an automatic process for managing these policies in interactive systems. Our mechanism is based on the negotiation and establishment of smart contracts through the WS-Agreement Specification. This automation allows a system to autonomously manage access requests from external organizations, negotiate access contracts in accordance with current policies with these organizations, and update these policies with attributes of new contracts. Thus, we have provided the means to convert our integration of security and privacy policies from a static to a dynamic form.

B. Perspectives

We have taken care in our work to offer the various means capable of covering all aspects of privacy protection, ranging from the definition of this concept to the automation of the process guaranteeing it. “Management of inferences” is another aspect that remains to be taken into account. It is true that PrivOrBAC is a powerful model which allows defining and configuring access controls adapted to different situations, but the risk of inference in this model is not reduced to zero. A user with access to a set of data can always combine them to derive private data that they cannot directly consult. In our case study, suppose that Jean Dupont is HIV positive and he chose to keep this information private. Bob, the nurse at K Hospital, does not have access to Jean Dupont's private data, but can view medical data. Bob can therefore see that Jean Dupont is being treated with a protein called “interferon”. This protein is used in the treatment of viral diseases (AIDS, hepatitis, papillomavirus, etc.), in oncology (sarcoma) or in preventive treatment. This information alone does not allow deducing with certainty that Jean Dupont is a carrier of HIV, but by combining it with the results of analyzes on the P24 antigen which are also part of the medical data that Bob can consult, the private data can therefore be revealed. Our perspectives therefore focus on proposing ways to combine with PrivOrBAC to guarantee privacy-aware access control without risk of inference.

References

[1] N. Ajam, N. Cuppens-Boulahia, and F. Cuppens, “Contextual Privacy Management in Extended Role Based Access Control Model,” in Springer-Verlag Berlin Heidelberg, 2010, pp.121-135. https://doi.org/10.1007/978-3-642-11207-2_10.

[2] A. Abou El Kalam, R. El Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miège, C. Saurel, and G. Trouessin, “Or-BAC : un modèle de contrôle d’accès basé sur les organisations, ” in Cahiers francophones de la recherche en sécurité de l’information, 2003, pp. 30-43.

[3] J. EL MOKHTARI, A. ABOU EL KALAM, S. BENHADDOU, and H. MEDROUMI, “PrivUML: A Privacy Metamodel,” in Proceedings of the 10th International Conference on Ambient Systems, Networks and Technologies (ANT 2019), 2019, pp.53-60. https://doi.org/10.1016/j.procs.2019.04.011.

[4] OASIS , “eXtensible Access Control Markup Language (XACML) Version 1.0,”, 2003. www.oasis-open.org › committees › oasis-xacml-1.0.pdf.

[5] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu, “Web services agreement specification (WS-Agreement), ” in Global Grid Forum, 2004. http://www.ogf.org/documents/GFD.192.pdf.

[6] J. EL MOKHTARI, A. ABOU EL KALAM, S. BENHADDOU, and J.P. LEROY, “Transformation of PrivUML into XACML Using QVT,” in Proceedings of the 12th International Conference on Soft Computing and Pattern Recognition (SoCPaR), 2021, pp.1-13. https://doi.org/10.1007/978-3-030-73689-7_93.

[7] F. Cuppens, and N. Cuppens-Boulahia, “Modeling contextual security policies,” in Springer-Verlag, 2007, pp. 285–305. DOI 10.1007/s10207-007-0051-9.

[8] T. Lodderstedt, D. Basin, and J.Doser, “SecureUML: A UML-Based Modeling Language for Model-Driven Security,” In UML 2002 - Springer, Berlin, Heidelberg, 2002, pp.426-441. https://doi.org/10.1007/3-540-45800-X_33.

[9] J. Jürjens, “UMLsec: Extending UML for Secure Systems Development,” In UML 2002 - Springer, Berlin, Heidelberg, 2002, pp.412-425. https://doi.org/10.1007/3-540-45800-X_32.

[10] P. Colombo, and E. Ferrari, “Towards a Modeling and Analysis Framework for Privacy-aware Systems,” In International Conference on Privacy, Security, Risk and Trust and International Confernece on Social Computing, 2012, pp.81-90. DOI: 10.1109/SocialCom-PASSAT.2012.12.

[11] T. Basso, L. Montecchi, R. Moraes, M. Jino, and A. Bondavalli, “Towards a UML Profile for Privacy-Aware Applications,” In IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic

and Secure Computing; Pervasive Intelligence and Computing, 2015, pp.371-378. DOI: 10.1109/CIT/IUCC/DASC/PICOM.2015.53.

[12] M. Kõlvart, M. Poola, and A. Rull, "Smart Contracts," In The Future of Law and eTechnologies - Springer, Cham, 2016. https://doi.org/10.1007/978-3-319-26896-5_7.

[13] I. Sergey, A. Kumar, and A. Hobor, "Scilla: a Smart Contract Intermediate-Level LAnguage," in DBLP Computer Science Bibliography, 2018. arXiv:1801.00687.

[14] C.D. Clack, V.A. Bakshi, and L. Braine, "Smart Contract Templates: foundations, design landscape and research directions," in Barclays Bank PLC, 2017. http://arxiv.org/abs/1608.00771.

[15] V. Scoca, R.B. Uriarte, and R. De Nicola, "Smart Contract Negotiation in Cloud Computing," in IEEE 10th International Conference on Cloud Computing (CLOUD), 2018, pp.592-599. DOI: 10.1109/CLOUD.2017.81.

[16] H. Ludwig, A. Keller, A. Dan, R. King, and R. Franck, "Web Service Level Agreement (WSLA) Language Specification," in IBM Corporation, 2003, pp.815–824.

[17] V. Tosic, K. Patel, and B. Pagurek, "WSOL — Web Service Offerings Language,", 2002, pp.57-67. DOI: 10.1007/3-540-36189-8_5.

[18] O. Wäldrich, D. Battré, F. Brazier, K. Clark, M. Oey, A. Papaspyrou, P. Wieder, and W. Ziegler, "WS-Agreement Negotiation Version 1.0,", 2011. http://www.ogf.org/documents/GFD.193.pdf.

[19] F. Marino, C. Moiso, and M. Petracca, "Automatic contract negotiation, service discovery and mutual authentication solutions: A survey on the enabling technologies of the forthcoming IoT ecosystems," in Computer Networks, 2018, pp.176-195. DOI: 10.1016/j.comnet.2018.11.011.

[20] Y. Li, N. Cuppens-Boulahia, J.M. Crom, F. Cuppens, and V. Frey, "Reaching Agreement in Security Policy Negotiation," in IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, 2014, pp.98-105. DOI: 10.1109/TrustCom.2014.17.

[21] F. Li, and S. Clarke, "A Context-Based Strategy for SLA Negotiation in the IoT Environment," in IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2019. DOI: 10.1109/PERCOMW.2019.8730752.

[22] F. Li, C. Cabrera, and S. Clarke, "A WS-Agreement Based SLA Ontology for IoT Services," in Internet of Things – ICIOT 2019, 2019, pp.58-72. DOI: 10.1007/978-3-030-23357-0_5.

[23] B. Shojaiemehr, A. Rahmani, and N. Qader, "A Three-phase Process for SLA Negotiation of Composite Cloud Services," in Computer Standards & Interfaces, 2019. DOI: 10.1016/j.csi.2019.01.001.

[24] T. Labidi, A. Mtibaa, W. Gaaloul, and F. Gargouri, "Ontology-Based SLA Negotiation and Re-Negotiation for Cloud Computing," in IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2017, pp.36-41. DOI: 10.1109/WETICE.2017.24.

[25] R. Baig, W. Khan, I. Haq, and I. Khan, "Agent-Based SLA Negotiation Protocol for Cloud Computing," in International Conference on Cloud Computing Research and Innovation (ICCCRI), 2017, pp.33-37. DOI: 10.1109/ICCCRI.2017.13.

[26] S.S. Tseng, H.C. Chen, L.L. Hu, and Y.T. Lin, "CBR-based negotiation RBAC model for enhancing ubiquitous resources management," in International Journal of Information Management, 2017, pp.1539-1550. DOI: 10.1016/j.ijinfomgt.2016.05.009.

[27] I. Castro, A. Panda, B. Raghavan, S. Shenker, and S. Gorinsky, "Route Bazaar: Automatic Interdomain Contract Negotiation," in the 15th Workshop on Hot Topics in Operating Systems (HotOS), 2015.

[28] C. Coutinho, A. Cretan, C.F. da Silva, P. Ghodous, and R. JardimGonçalves, "Service-based Negotiation for Advanced Collaboration in Enterprise Networks," in Journal of Intelligent Manufacturing - Springer Verlag, 2016, pp.201-216. DOI: ff10.1007/s10845-013-0857-4ff.

[29] WSAG4J framework V2.0.0, 2012. http://wsag4j.sourceforge.net/site/wsag/overview.html.

[30] SLA Framework v1.1, 2016. https://github.com/FIWARE/ops.Sla-framework.

[31] A. Lenk, L. Bonorden, A. Hellmanns, N. Roedder, and S. Jaehnichen, "Towards a Taxonomy of Standards in Smart Data," in IEEE International Conference on Big Data, 2015. DOI 10.1109/BigData.2015.7363946.

[32] I. Stankov, R. Datsenka, and K. Kurbel, "Service Level Agreement as an Instrument to Enhance Trust in Cloud Computing – An Analysis of Infrastructure-as-a-Service Providers," in Proceedings of the Eighteenth Americas Conference on Information System (AMCIS), 2012. http://aisel.aisnet.org/amcis2012/proceedings/HCIStudies/12.

[33] H. Ludwig, T. Nakata, O. Wäldrich, P. Wieder, and W. Ziegler, "Reliable Orchestration of Resources Using WS-Agreement," in HPCC 2006 - Springer, Berlin, Heidelberg, 2006. https://doi.org/10.1007/11847366_78.

[34] M. Smith, M. Schmidt, N. Fallenbeck, C. Schridde, and B. Freisleben, "Optimising Security Configurations with Service Level Agreements," 2007. https://www.semanticscholar.org/paper/Optimising-Security-Configurations-with-Service-Smith-Schmidt/bb18eca8f629edb494cb8dd8c7fab9027cd23dcf.

[35] Y. Li, N. Cuppens-Boulahia, J.M. Crom, F. Cuppens, and V. Frey, "Expression and Enforcement of Security Policy for Virtual Resource Allocation in IaaS Cloud," in IFIP Advances in Information and Communication Technology - Springer, vol 471, 2016. https://doi.org/10.1007/978-3-319-33630-5_8.