A Life Cycle for Authorization Systems Development in the GDPR Perspective*

Said Daoudagh[1,2] and Eda Marchetti[1]

[1] ISTI-CNR, Pisa, Italy
{said.daoudagh, eda.marchetti}@isti.cnr.it
[2] University of Pisa, Pisa, Italy
said.daoudagh@di.unipi.it

Abstract

The General Data Protection Regulation (GDPR) defines the principle of Integrity and Confidentiality, and implicitly calls for the adoption of authorization systems for regulating the access to personal data. We present here a process development life cycle for the specification, deployment and testing of authorization systems. The life cycle targets legal aspects, such as the data usage purpose, the user consent and the data retention period. We also present its preliminary architecture where available solutions for extracting, implementing and testing the data protection regulation are integrated. The objective is to propose for the first time a unique improved solution for addressing different aspects of the GDPR development and enforcement along all the life cycle phases.

1 Introduction

The General Data Protection Regulation (GDPR) is the new EU Data Protection Regulation [21] in charge of harmonize the regulation of Data Protection across the EU member states. At the same time, it enhances and arises business opportunities within the Digital Single Market space. However, the natural language nature of the GDPR makes most of the provisions to be expressed in generic terms and does not provide specific indication on how they should be actuated. Thus, applying and demonstrating the GDPR compliance, in order to avoid also the related penalties, becomes an important research challenge.

Many businesses today are struggling in the definition of appropriate procedures and technical solutions for their development process so as to enforce and demonstrate the GDPR compliance [1, 6, 13, 15, 24]. In particular, following the *correct-by-design* principle, they are looking for effective and efficacious means for increasing the software high-confidence and quality and, at the same time, reducing the cost and effort of development. Consequently, integrated solutions for designing and promptly testing their applications and systems are urgently necessary. Considering the GDPR, as for any other software requirement, a fundamental step for guaranteeing its by-design compliant realization is that the data protection concepts have to be integrated into overall software life cycle: from gathering of the requirements to deployment and subsequent maintenance of the system.

Currently, several proposals are trying to assist the organizations in the deployment of adequate fine-grained mechanisms that take into account legal requirements, such as the data usage purpose, user consent and the data retention period. In particular, research attention has been devoted to authorization systems because they are recognised, by scientific communities and private companies, as the successful elements for the development of GDPR-by-design compliant solutions [7, 29, 30]. However, to the best of our knowledge, most of the available proposals targets just a single aspect of authorization system development and no integrated solutions for guiding their GDPR-by-design compliant development through the entire life cycle are provided.

Therefore, the proposal of this paper: a specific, integrated GDPR focused process development life cycle for the specification, deployment and testing of adequate fine-grained authorization mechanisms able to take into account legal requirements. The idea has been inspired by the life cycle introduced in [14], which is a systematic approach to implementing authorization systems within enterprise. Even if generic, the proposal of [14] does not target explicitly the GDPR demands or any other legal framework.

Additionally, to promote the applicability of the proposed life cycle into the business and industrial context we also present its preliminary automation. More precisely, we integrated, for the first time, into a unique environment some of the available solutions for: specifying the privacy requirements, controlling personal data, processing them, and demonstrating the compliance with the GDPR in collecting, using, storing, disclosing and/or disposing of the personal data.

In line with this view, the paper focuses on the following primary objectives: **OBJ 1:** defining a GDPR-based life cycle for authorization systems; **OBJ 2:** providing an integrated environment for automatically enforcing the data protection or privacy regulations.

Outline. Section 2 presents the basic concepts used along the proposal; Section 3 describes the adopted development process and the solutions proposed for its phases; Section 4 presents the unique environment we are developing to accommodate the proposed life cycle; finally, Section 5 concludes the paper.

2 Background and Related Work

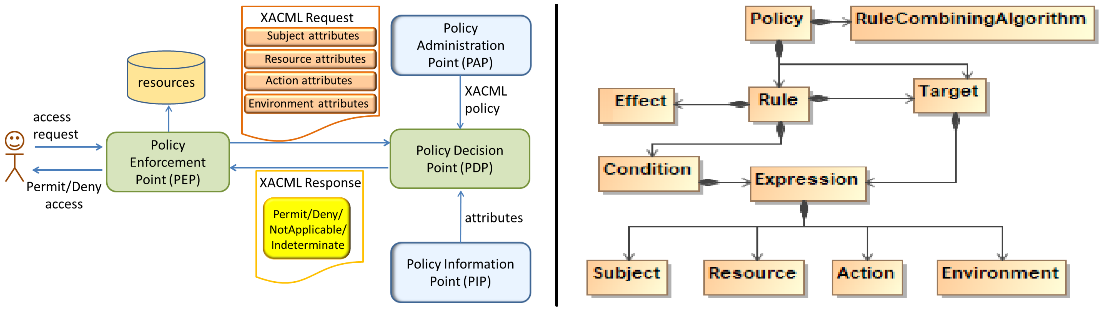
In this section we briefly overview the concepts and definitions adopted in the remains of this paper, focusing in particular on the GDPR and access control concepts.

General Data Protection Regulation. The GDPR [21] defines *Personal Data* as any information relating to an identified or identifiable natural person called *Data Subject*. That means that, a data subject is a Natural Person (a living human being), whose data are managed by a *Controller*. This regulation became into effect on May 2018 and has replaced the previous Data Protection Directive conceived in 1995. The aim of the new regulation is to strengthen the rights of the individual over their own data and at the same time to make organizations more accountable w.r.t. the previous directive. In addition, the GDPR has also the objective to eliminate all the barriers for the services to be delivered in the European Union and, therefore, to enhance business opportunities within the Digital Single Market. The GDPR will contribute to the harmonization of the previous fragmented data protection laws across the EU, so as to ensure equal protection of Human Rights of the European Citizens.

The GDPR is composed of 99 *articles* that represent the mandatory part of the regulation. The GDPR is applied to the processing of personal data, whether it is automated (even partially) or not. It defines, among others, the following principles and demands: *Transparency*, i.e., data must be processed fairly, lawfully and transparently; *Purposes*, i.e., data should only be collected for determined, explicit and legitimate purposes, and should not be processed later for other purposes; *Minimization*, i.e., the processed data must be relevant, adequate and limited to what is necessary in view of the purposes for which they are processed; *Accuracy*, i.e., the processed data must be accurate and up-to-date regularly; *Retention*, i.e., data must be deleted after a limited period; *Subject explicit consent*, i.e., data may be collected and processed only if the data subject has given his explicit consent.

Access Control Concepts. *Access Control (AC)* is a fundamental building block for secure information sharing [9], because it ensures that only the intended people can access security-classified data and that these intended users are only given the level of access required to accomplish their tasks. Several access control models have been proposed, including models taking into account time, location, and situation [8, 18, 25, 32] and models specific for privacy-sensitive data [26].

An AC is usually implemented through *Access Control Mechanism* (ACM), which is the system that provides a decision to an authorization request, typically based on predefined *Access Control Policy (ACP)*. This is a specific statement of what is and is not allowed on the basis of a set of rules, defined in terms of conditions on attributes of subjects, resources, actions, and environment, and combining algorithms for establish the precedence among the rules.



a) Access Control Reference Architecture b) Access Control Policy Data Model

Figure 1: Access Control System: Reference Architecture and ACP Data Model.

Attribute-Based Access Control (ABAC) [23] and its standard implementation, the eXtensible Access Control Markup Language (XACML) [27], are the widespread adopted models in the access control environment. As schematize in Figure 1(a), the main components of XACML standard are: the Policy Administration Point (PAP) is the system entity in charge of managing the policies; the Policy Enforcement Point (PEP), usually embedded into an application system, receives the access request in its native format, constructs an XACML request and sends it to the Policy Decision Point (PDP); the Policy Information Point (PIP) provides the PDP with the values of subject, resource, action and environment attributes; the PDP evaluates the policy against the request and returns the response, including the authorization decision, to the PEP.

The structure of an XACML access control policy is sketched in Figure 1(b). More precisely, an XACML policy has a tree structure whose main elements are: *PolicySet* (not presented in the figure), *Policy*, *Rule*, *Target* and *Condition*. The *PolicySet* includes one or more policies. A *Policy* contains a *Target* and one or more rules. The *Target* specifies a set of constraints on *attributes* of a given request. Typical categories of *attributes* are *Subject*, *Resource*, *Action* and *Environment*. The *Rule* specifies a *Target* and a *Condition* containing one or more boolean functions. If the *Condition* evaluates to true, then the Rule's *Effect* (a value of *Permit* or

Deny) is returned, otherwise a *NotApplicable* decision is formulated. If an error occurs during the evaluation of a policy against a request, *Indeterminate* value is returned. The *PolicyCombiningAlgorithm* (not represented in the figure) and the *RuleCombiningAlgorithm* define how to combine the results from multiple policies and rules respectively in order to derive a single authorization access decision.

Related Work. In literature there are several works that use access control as main means of protecting personal data. Different proposals are mainly divided into two main categories. The former uses Access Control to address specific concepts that can be related to a given law, such as consent and purpose. In this area an initial proposal for an automatically enforceable policy language is discussed in [16], whereas, a formal definition of the consent is introduced in [31]. The latter refers explicitly a given law (e.g., the EU GDPR or the US HIPAA) in using Access Control. In particular in [17] the authors have evaluated whether the XACML standard is adequate to express the constraints imposed in HIPAA, whereas in [22], the authors investigated the feasibility of translating the articles related to access control of the previous EU data protection directive. In the industrial environment, authors in [14] proposed a systematic methodology for the implementation of ABAC solutions in real contexts.

Differently from the above contributions, the proposal of this paper does not focus on a single aspect of the development process but provides a unified environment able to: model ACPs that are by-design compliant with the GDPR; test those ACPs by means of state-of-the-art testing tools; and to monitors their application during the production time, and eventually to suggest possible improvements in case of deviation of the expected behaviour. Therefore, the solution proposed in this paper aims at providing, for the first time, a practical specification of the Authorization Development Life Cycle in the light of the GDPR covering all its stages.

3 Authorization Policy Life Cycle

In this section we target the first objective of this paper (**OBJ 1**): defining a GDPR-based life cycle for authorization systems assuring the by-design compliance to data protection regulation. As any other software application, the development of GDPR compliant authorization systems involves different stages of software development. Thus, our first objective is to formalize into a specific life cycle the required activities for: collecting and specifying legal requirements into formal representations, defining and testing data protection policies, and implementing AC-based mechanisms.

In presenting our proposal, among the different development processes, we refer to and modify the authorization policy life cycle introduced in [14], which is a systematic approach to implementing ABAC systems within enterprises. The proposed life cycle, schematized in Figure 2, does not strictly depend on any specific ABAC implementation. However, in this paper we refer to the widely industrial adopted XACML-based authorization system because it is the only available standardized specification for ABAC. As schematize in Figure 2, the modified version of the process consists of the following steps:

GDPR-based use case definition (step ①): i.e., define context and an achievable scope so as to establish a common base to discuss with different stakeholders. In this case, the established use cases need to be conceived according to GDPR implementation challenges;
Gather authorization requirements (step ②): i.e., to gather all the requirements and the sources they come from. In our case, the primary source is the GDPR regulation, therefore, authorization requirements should de defined in terms of statements or natural language

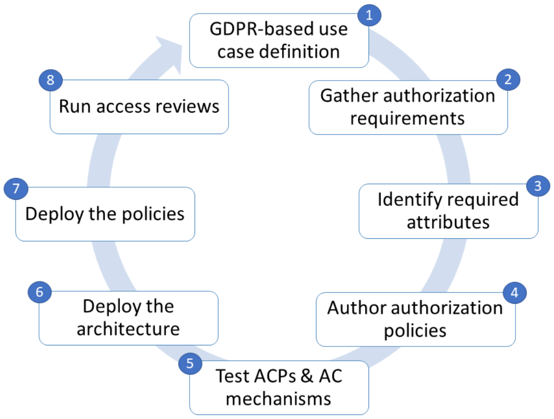

1
GDPR-based use case definition
2
Gather authorization requirements
3
Identify required attributes
4
Author authorization policies
5
Test ACPs & AC mechanisms
6
Deploy the architecture
7
Deploy the policies
8
Run access reviews

Figure 2: The Authorization Policy Life Cycle (adapted from [14]).

authorization policies. Additionally, business requirements (e.g., working hours) and security best practices (e.g., encrypting data) need also to be defined.

Identify required attributes (step ③): i.e., to identify the attributes used in the selected requirements and their origin so as to make easier requirement reviews. The attributes should depend on the language or functionalities of the XACML reference architecture.

Author authorization policies (step ④): i.e., to transform the natural language statements into machine-interpretable statements, in order to eliminate any ambiguity introduced by natural language. Thus, a list of XACML policies encoding the GDPR's provisions need to be defined as well as the order in which those policies should be evaluated.

Test ACPs & AC mechanisms (step ⑤): i.e., to ensure that the implemented XACML policies meet the GDPR requirements. State-of-the-art and specifically conceived testing techniques should be used according to the different purposes. This step involves also the evaluation of the adequacy of the current AC mechanisms in the context of the GDPR.

Deploy the architecture (step ⑥): i.e., to define the contact point within the existing systems in order to make the different applications able to interact with authorization system.

Deploy the policies (step ⑦): i.e., to deploying the authored XACML policies according to the selected (production) environment. This step is usually business dependent.

Run access reviews (step ⑧): i.e., to analyse the policies against a set of attributes to determine what these attributes grant. In the context of the GDPR, this should involve the simulation of realistic scenarios according to specific application use cases. Additionally, the data coming from the testing activities could be used to assess the implemented solutions and identify possible improvements.

4 Life Cycle Automation

In order to propose an applicable and effective solution, the second objective of this paper (**OBJ 2**) is to provide an integrated environment for the automatic enforcing of the GDPR-based life cycle presented in the previous section. To the best of the authors' knowledge, this proposal is the first attempt to integrate, in a unique automated environment, different available solutions for extracting, implementing and testing the data protection regulation.

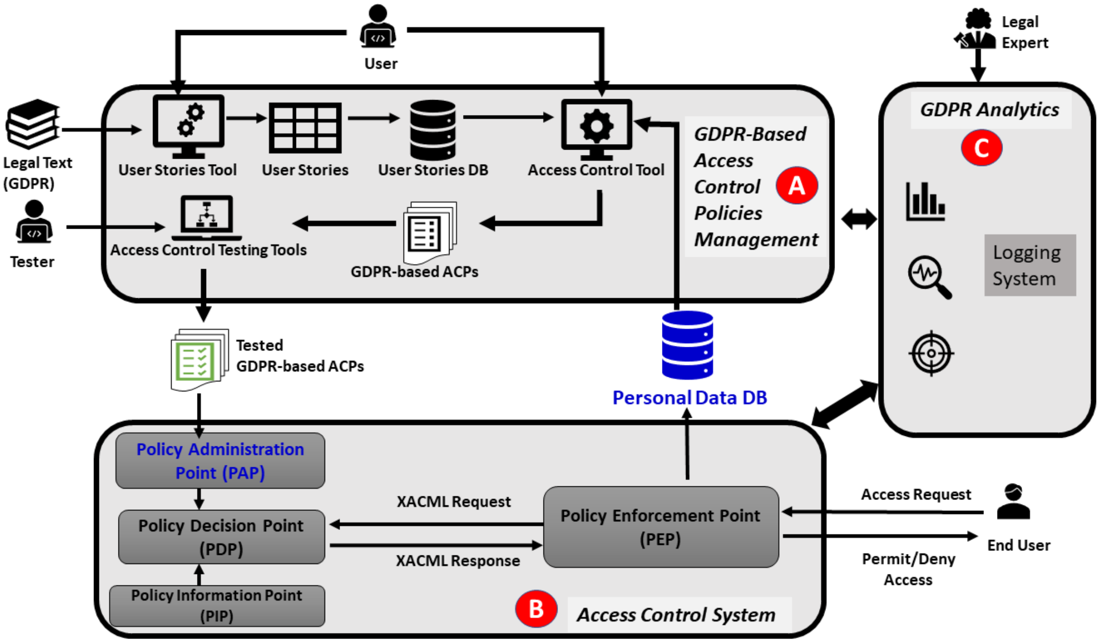


Figure 3: The Proposed GDPR-based Environment.

Our proposal is depicted in Figure 3, and it is composed of three main modules: (1) GDPR-Based Access Control Policies Management (module Ⓐ); (2) Access Control System (module Ⓑ); and (3) GDPR Analytics (module Ⓒ).

Differently from the generic ACS architecture, in this paper we assume that the protected resources are Personal Data hosted in a specific database, the `Personal Data DB` component of Figure 3.

In the remainder of this section, we detail how the modules have been implemented into the proposed environment and how they are related to the authorization life cycle schematized in Figure 2.

4.1 Use case definition and Gathering of authorization requirements

Among the solutions to tackle security issues and vulnerabilities in an efficient and effective way, the possibility of using *backlogs* to drive the software development work is currently taking place. Generally, a *backlog* is a prioritized features list describing the functionalities to be included in the final product [2]. These backlog items are often provided in the form of *User Stories* [3], i.e., a list of ready-made specification of items (requirements and task descriptions) useful for the implementation.

In the context of GDPR having a ready-to-use set of User Stories, focused on GDPR provisions and associated to specific ACPs, represents an important means to minimize development effort and assure high quality of the final product. Indeed, when an authorization system need to be implemented, developers could pick up the necessary predefined User Stories, and their associated ACPs, and exploit them in order to easily implement the required policies into the Access Control Mechanism.

Considering the life cycle schematized in Figure 2, the definition of the User Stories set can be reloaded as: the definition of a *Data Protection Backlog* that contains User Stories based on the GDPR requirements (**Step ①**); and the mapping of the GDPR provisions into User Stories (**Step ②**).

In the environment proposed in Figure 3, the definition of User Stories is in charge of the module Ⓐ, and specifically of the `User Stories Tool` component. From a practical point of view, the methodology for defining GDPR based User Stories has been introduced in [5], and therefore the component provides an automation of the previously introduced process. More precisely, the `User Stories tool` component takes as input a Legal Text (the GDPR text in this case), analyses the GDPR's articles related to ACs and creates an *Epic* [2] for each of them. An Epic is a set of User Stories having the same conceptual purpose. For the GDPR, a total of forty-one Epics are produced: three of them concerning only AC mechanism; eight referring only ACP, and thirty articles related to both ACP and AC mechanism. Then, for each article one or more User Stories are derived and linked to the proper Epic. As an output of the `User Stories Tool` component, a *Data Protection Backlog*, i.e., a Privacy Backlog containing a set of User Stories organized in Epics, is stored into the `USER Stories DB` (Figure 3). In Table 1 an extract of content of the Data Protection Backlog is presented. As in the table the column **Article** (first column) contains the GDPR's articles, and the column **User Story** contains the GDPR-based User Stories defined.

Table 1: GDPR-focused User Stories: Controller and Data Subject Perspectives.

Article	**User Story**
Art. 6.1(a)	As a [Controller], I want [to process Personal Data only if Data Subject has given consent for one or more specific purpose], so that [the processing shall be lawful].
Art. 15.1	As a [Data Subject], I want [to access my Personal Data and all the information (e.g., purpose and categories)], so that [I can be aware about my privacy].

Through the GUI of module Ⓐ in Figure 3, the User (in this case an authorization system developer) can select a set of predefined User Stories and proceed with their translation into ACPs as detailed in the next section.

4.2 Identify required attributes and Author authorization policies

Step ③ and Step ④ of the life cycle of Figure 2 aim at transforming the User Stories into machine-interpretable statements. As a result, a list of XACML policies encoding the GDPR principles is defined.

In the environment depicted in Figure 3, the `Access Control Tool` component of module Ⓐ is in charge of automating the two steps. Hence, the component takes as input a set of User Stories selected by the User from the `User Stories DB` and, through the automation of the methodology introduced in [6], provides the associate GDPR-based ACPs.

In details, considering the Step ③, first the component classifies the identified attributes into access control commonly-used entities (or categories) (see Section 2), highlights relations between them and lets the mapping into the ABAC terms. For instance, by referring to the User Story related to the Art. 15.1 (see the second row of Table 1), the component identifies and classifies the following attributes: **Data Subject** as a *Subject*, **access** as an *Action*, and **Personal Data** as a *Resource* category.

Then, the `Access Control Tool` component automates the translation of the selected User Stories into derived AC rules that corresponds to Step ④ of Figure 2. In particular, this step consist into the instantiation of the AC rules with actual attributes, and the translation of the resulting policies into a given formalism or language [1].

As in Figure 3, the final translation requires the interaction with the User and the `Personal Data DB`. Specifically, the User needs to identify in the `Personal Data DB` the real attributes to be considered. As example, considering the Art. 15.1 (Table 1), Table 2 reports the attribute mapping for the following realistic scenario: *Alice (Customer, i.e., Data Subject) provided her name, her E-mail address, and the name of the city where she has the permanent address to the ABC company (Controller). Alice in any moment can exercise her right of access pursuant the Art. 15.1.*

More precisely, column *Identified Attribute* of Table 2 contains the identified attributes; column *Attribute Category* shows the classification of those attributes into a specific category; finally, column *Access Control Category* illustrates the classification attributes into the commonly used entities in access control.

Table 2: Attribute Classification Example.

Identified Attribute	**Attribute Category**	**Access Control Category**
Alice	Customer	Subject
read		Action
name	Biodata	Resource
E-mail	Contact data	Resource
permanent city	Location data	Resource

The `Access Control Tool` uses the derived attribute classification for mapping them into the attributes of the User Stories and deriving enforceable ACPs in a given language. As an example, by considering the attribute classification of Table 2 and the User Stories associated with Art. 15.1, Figure 4 shows the derived matching, whereas Figure 5 reports the translation into XACML-like language. Consequently, the policy is applicable to the subject *Alice* and contains two rules: (1) the first rule, with RuleId equal to *readRule*, represents the AC rule starting from User Story associated with Art. 15.1 (see second row of Table 1), and guarantees that Alice can read all the information concerning her; (2) the second rule, called *defaultRule*, represents a standard default rule that denies all which is not allowed explicitly.

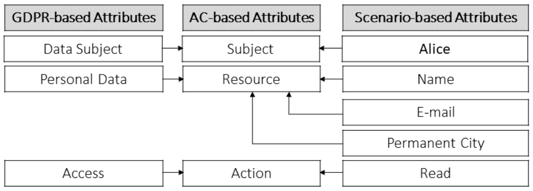


Figure 4: Attributes Matching Example.

4.3 Test ACPs & AC mechanisms

Step ⑤ of Figure 2 aims at testing both the developed ACPs and the current AC mechanisms. Indeed, to ensure that the implemented XACML policies meet the GDPR requirements specific testing process should be adopted. Considering the environment of Figure 3, the `Access Control Testing Tools` component of module Ⓐ is in charge of implementing the Step ⑤.

[1] In the current implementation the XACML standard [27] is considered but other implementation of ABAC model can be equally adopted.

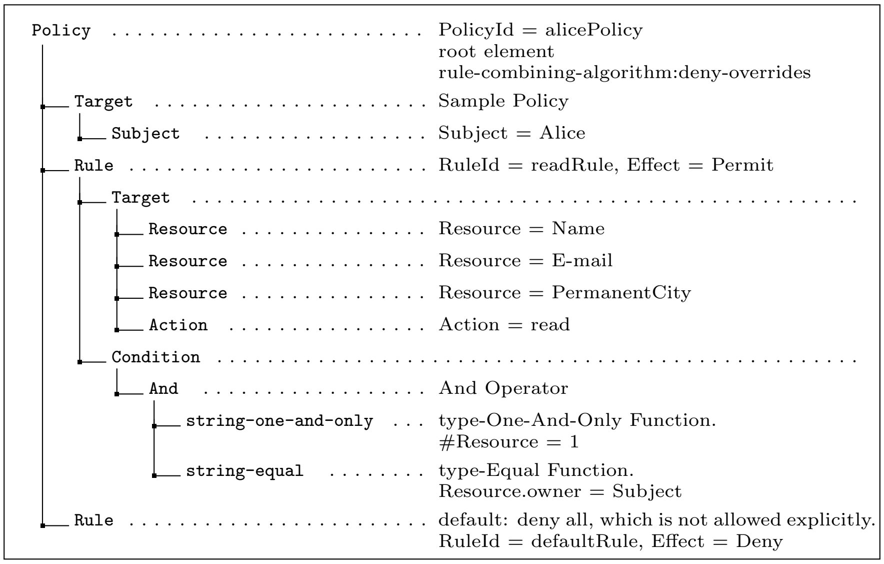

Policy .... PolicyId = alicePolicy
root element
rule-combining-algorithm:deny-overrides
Target .... Sample Policy
Subject .... Subject = Alice
Rule .... RuleId = readRule, Effect = Permit
Target ....
Resource .... Resource = Name
Resource .... Resource = E-mail
Resource .... Resource = PermanentCity
Action .... Action = read
Condition ....
And .... And Operator
string-one-and-only ... type-One-And-Only Function.
#Resource = 1
string-equal .... type-Equal Function.
Resource.owner = Subject
Rule .... default: deny all, which is not allowed explicitly.
RuleId = defaultRule, Effect = Deny

Figure 5: A XACML-like Policy.

In particular, it integrates available solutions for: assessment of test strategies, testing GDPR-based ACPs expressed in XACML 3.0 and evaluating the adequacy of AC mechanisms with respect to the GDPR's provisions. For aim of completeness we report here below the main features of the `Access Control Testing Tools` component. Specifically:

1. *Test Case Generation*: starting from the developed ACPs, it is possible to generate AC requests able to test both the ACPs and AC mechanisms through a modified version of the X-CREATE tool [12], which provides several combinatorial test strategies, and the XACMET tool [19] that provides a model-based test generation strategy;

2. *Mutation Generation*: mutation analysis [28] can be applied on ACPs for measuring the adequacy of a test suite through an enhanced version of XACMUT tool [10];

3. *Test Cases Execution & Result Analyzer*: is an automated executor of test cases able to collect the execution results and calculates either the effectiveness of the considered test suites, or the vulnerabilities detected;

4. *Testing Strategy Enhancement*: it suggests possible hints for enhancing the applied test suite;

5. *Oracle Derivation*: is an automatic oracle able to associate the expected result for a given AC request based on a given ACP through an enhanced version of the XACMET tool [11, 19], which is an automated model-based oracle.

The Tester can interact with the `Access Control Testing Tools` component for realizing specific testing purposes. For instance, for testing GDPR-based ACPs expressed in XACML 3.0 the user can run the following facilities: first, the *Test Case Generation* for deriving the set of AC requests (in this case a test strategy can be selected from available ones); then, through *Test Cases Execution & Result Analyzer*, the Tester can execute the test cases on the GDPR-based ACPs and collect the results; whereas, through the *Oracle Derivation* component the tester can associate the expected result to each of the executed test cases; finally, the *Testing Strategy Enhancement* component can be used to visualize the results and suggestions for possible improvement of the test case generation strategy.

4.4 From the Deploy to the Access Review

In this section we briefly provide some hints for targeting the last three phases of the proposed authorization life cycle that involve the deployment of the AC architecture (Step ⑥ of Figure 2), the deployment of the developed and tested policies (Step ⑦), and the final analysis of the process development data (Step ⑧).

The idea behind Step ⑥ is to decouple the authorization functionalities from the business logic. This enables to adapt and extend the XACML reference architecture with new features without modifying the business logic of the applications that use Personal Data. This separation of concerns helps to propose scalable, manageable and extendible authorization solutions.

Once the architecture is deployed (module Ⓑ of Figure 3), Step ⑦ involves the deployment of the tested GDPR-based ACPs within the `Policy Administration Point` component of the `AC system` in order to assure the GDPR compliance. This allows the `Policy Decision Point` to retrieve and to evaluate the right ACP when the system receives an access request, from the end user (e.g., Data Subject or Controller), to the Personal Data hosted in the `Personal Data DB`.

Additionally, by referring to Step ⑧, facilities for collecting and managing information for the GDPR compliance and audit purposes [4, 15] should be included. To this purpose, module Ⓒ of Figure 3 is the proposal that we are currently finalizing. The module extends with logging systems, monitoring capabilities, and reporting functionalities of the proposed environment [20], so that data mining and machine learning techniques can be adopted to construct behavioral models based on data coming from the logging and testing activities and to discover and notify unwanted behaviors.

5 Conclusions

The GDPR represents a significant breakthrough in the digital economy and brings a lot of changes to the way in which online services are offered. This scenario calls for new approaches for developing systems where legal requirements are taken into account, just like the other requirements that a system must respond to. This paper focused on data protection requirements and, in particular, on the development of authorization systems able to enforcing the GDPR provisions. The idea was to provide for the first time a specific GDPR-based life cycle, able to assure the by-design compliance of the developed access control systems. Additionally, in order to make the proposal effective and applicable in real context, we provide also a reference architecture enforcing the proposed life cycle. The general nature of the proposed GDPR-based life cycle does not constrain the environment to the specific tools selected in this paper, and different components implementations could be considered. The intention was to demonstrate the feasibility of our proposal. Therefore, this work represented a preliminary step to integrate legal requirements into a software development process and several improvements are possible. In particular, the proposals of this paper need to be thoroughly extended and validated with real case studies and the architecture finalized in order to provide a unique user-friendly environment, able to assist developers in all the stages of development.

6 Acknowledgments

This work is partially supported by CyberSec4Europe Grant agreement ID: 830929.

References

[1] Amir Shayan Ahmadian, Daniel Strüber, Volker Riediger, and Jan Jürjens. Supporting privacy impact assessment by model-based privacy analysis. In *Proceedings of the The 33rd ACM/SIGAPP Symposium On Applied Computing (SAC)*. ACM, April 2018.

[2] J Ahola, C Frühwirth, M Helenius, L Kutvonen, J Myllylahti, T Nyberg, A Pietikäinen, P Pietikäinen, J Röning, S Ruohomaa, et al. Handbook of the secure agile software development life cycle. *University of Oulu*, 2014.

[3] Vishal Asthana, Izar Tarandach, Niall ODonoghue, Bryan Sullivan, and Mikko Saario. Practical security stories and security tasks for agile development environments. *Online, July*, 2012.

[4] Cesare Bartolini, Antonello Calabrò, and Eda Marchetti. GDPR and business processes: an effective solution. In *Proceedings of the 2nd International Conference on Applications of Intelligent Systems, APPIS 2019, Las Palmas de Gran Canaria, Spain, January 07-09, 2019*, pages 7:1–7:5, 2019.

[5] Cesare Bartolini, Said Daoudagh, Gabriele Lenzini, and Eda Marchetti. GDPR-based user stories in the access control perspective. In Mario Piattini, Paulo Rupino da Cunha, Ignacio García Rodríguez de Guzmán, and Ricardo Pérez-Castillo, editors, *Quality of Information and Communications Technology*, pages 3–17, Cham, 2019. Springer International Publishing.

[6] Cesare Bartolini., Said Daoudagh, Gabriele Lenzini., and Eda Marchetti. Towards a lawful authorized access: A preliminary gdpr-based authorized access. In *Proceedings of the 14th International Conference on Software Technologies - Volume 1: ICSOFT,*, pages 331–338. INSTICC, SciTePress, 2019.

[7] David Basin, Søren Debois, and Thomas Hildebrandt. On purpose and by necessity. In *Proceedings of the Twenty-Second International Conference on Financial Cryptography and Data Security (FC)*, February 2018.

[8] Elisa Bertino, Piero A. Bonatti, and Elena Ferrari. TRBAC: A temporal role-based access control model. *ACM Trans. Inf. Syst. Secur.*, 4(3):191–233, 2001.

[9] Elisa Bertino, Gabriel Ghinita, and Ashish Kamra. Access control for databases: Concepts and systems. *Foundations and Trends® in Databases*, 3(1–2):1–148, 2011.

[10] A. Bertolino, S. Daoudagh, F. Lonetti, and E. Marchetti. Xacmut: Xacml 2.0 mutants generator. In *Proc. of 8th International Workshop on Mutation Analysis*, pages 28–33, 2013.

[11] Antonia Bertolino, Said Daoudagh, Francesca Lonetti, and Eda Marchetti. An automated model-based test oracle for access control systems. In *Proceedings of the 13th International Workshop on Automation of Software Test*, AST '18, pages 2–8, New York, NY, USA, 2018. ACM.

[12] Antonia Bertolino, Said Daoudagh, Francesca Lonetti, Eda Marchetti, and Louis Schilders. Automated testing of extensible access control markup language-based access control systems. *IET Software*, 7(4):203–212, 2013.

[13] Felix Bieker, Nicholas Martin, Michael Friedewald, and Marit Hansen. Data protection impact assessment. In Marit Hansen, Eleni Kosta, Igor Nai-Fovino, and Simone Fischer-Hübner, editors, *Privacy and Identity Management*, volume 526 of *IFIP Advances in Information and Communication Technology*, pages 207–220. Springer, 2018.

[14] David Brossard, Gerry Gebel, and Mark Berg. A systematic approach to implementing abac. In *Proceedings of the 2Nd ACM Workshop on Attribute-Based Access Control*, ABAC '17, pages 53–59, New York, NY, USA, 2017. ACM.

[15] Antonello Calabrò, Said Daoudagh, and Eda Marchetti. Integrating access control and business process for GDPR compliance: A preliminary study. In *Proceedings of the Third Italian Conference on Cyber Security, Pisa, Italy, February 13-15, 2019.*, 2019.

[16] Francesco Di Cerbo, Fabio Martinelli, Ilaria Matteucci, and Paolo Mori. Towards a declarative approach to stateful and stateless usage control for data protection. In *WEBIST*, pages 308–315. SciTePress, 2018.

[17] Omar Chowdhury, Haining Chen, Jianwei Niu, Ninghui Li, and Elisa Bertino. On xacml's adequacy to specify and to enforce hipaa. In *Proceedings of the 3rd USENIX Conference on Health Security and Privacy*, HealthSec'12, pages 11–11, Berkeley, CA, USA, 2012. USENIX Association.

[18] Maria Luisa Damiani, Elisa Bertino, Barbara Catania, and Paolo Perlasca. GEO-RBAC: A spatially aware RBAC. *ACM Trans. Inf. Syst. Secur.*, 10(1):2, 2007.

[19] S. Daoudagh, F. Lonetti, and E. Marchetti. XACMET: XACML Testing & Modeling. *Software Quality Journal*, 2019. To appear.

[20] Said Daoudagh. A Data Warehouse and a Framework for the Validation and Testing of Access Control Systems. Master's thesis, Department of Computer Science, University of Pisa, Italy, 2017.

[21] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). *Official Journal of the European Union*, L119:1–88, May 2016.

[22] Kaniz Fatema, Christophe Debruyne, Dave Lewis, Declan OSullivan, John P Morrison, and Abdullah-Al Mazed. A semi-automated methodology for extracting access control rules from the european data protection directive. In *Security and Privacy Workshops (SPW), 2016 IEEE*, pages 25–32. IEEE, 2016.

[23] David F. Ferraiolo, Ramaswamy Chandramouli, Rick Kuhn, and Vincent C. Hu. Extensible access control markup language (XACML) and next generation access control (NGAC). In *Proceedings of the 2016 ACM International Workshop on Attribute Based Access Control, ABAC@CODASPY 2016, New Orleans, Louisiana, USA, March 11, 2016*, pages 13–24. ACM, 2016.

[24] Pietro Ferrara and Fausto Spoto. Static analysis for GDPR compliance. In Elena Ferrari, Marco Baldi, and Roberto Baldoni, editors, *Proceedings of the Second Italian Conference on Cyber Security (ITASEC)*, February 2018.

[25] A. S. M. Kayes, Jun Han, and Alan W. Colman. An ontological framework for situation-aware access control of software services. *Inf. Syst.*, 53:253–277, 2015.

[26] Qun Ni, Elisa Bertino, Jorge Lobo, Carolyn Brodie, Clare-Marie Karat, John Karat, and Alberto Trombetta. Privacy-aware role-based access control. *ACM Trans. Inf. Syst. Secur.*, 13(3):24:1–24:31, 2010.

[27] OASIS. eXtensible Access Control Markup Language (XACML) Version 3.0. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html, January 2013.

[28] Mike Papadakis, Marinos Kintis, Jie Zhang, Yue Jia, Yves Le Traon, and Mark Harman. Chapter six - mutation testing advances: An analysis and survey. volume 112 of *Advances in Computers*, pages 275 – 378. Elsevier, 2019.

[29] Qusai Ramadan, Mattia Salnitri, Daniel Strüber, Jan Jürjens, and Paolo Giorgini. From secure business process modeling to design-level security verification. In *Proceedings of the ACM/IEEE* 20^{th} *International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 123–133. IEEE, September 2017.

[30] Silvio Ranise and Hari Siswantoro. Automated legal compliance checking by security policy analysis. In *Computer Safety, Reliability, and Security - SAFECOMP 2017 Workshops, ASSURE, DECSoS, SASSUR, TELERISE, and TIPS, Trento, Italy, September 12, 2017, Proceedings*, volume 10489 of *Lecture Notes in Computer Science*, pages 361–372. Springer, 2017.

[31] Max-Robert Ulbricht and Frank Pallas. Yappl - A lightweight privacy preference language for

legally sufficient and automated consent provision in iot scenarios. In *DPM 2018 and CBT 2018 - ESORICS 2018 International Workshops, Barcelona, Spain, September 6-7, 2018*, pages 329–344, 2018.

[32] Stephen S. Yau and Junwei Liu. A situation-aware access control based privacy-preserving service matchmaking approach for service-oriented architecture. In *2007 IEEE (ICWS 2007), July 9-13, 2007, Salt Lake City, Utah, USA*, pages 1056–1063. IEEE Computer Society, 2007.