Towards Automated Checking of GDPR Compliance

Pauline Di Salvo-Cilia
Aix-Marseille Univ., CNRS
Marseille, France
pauline.di-salvo-cilia@lis-lab.fr

Alba Martinez Anton
Aix-Marseille Univ., CNRS
Marseille, France
alba.martinez-anton@lis-lab.fr

Clara Bertolissi
Aix-Marseille Univ., CNRS
Marseille, France
clara.bertolissi@lis-lab.fr

***Abstract*—We propose a prototype for automating the General Data Protection Regulation compliance checking, in particular for consent-related principles. Our solution leverages provenance graphs to model compliance-related information. We present a prototype implementation of our model, based on Prolog.**

Keywords- Privacy; Data protection; GDPR compliance.

I. Introduction

In recent years, the quantity of personal data managed by systems has been growing steadily. In order to protect users and their data, the European Union (EU) has established the General Data Protection Regulation (GDPR) [4], which applies to European countries since 2018. Among the principles that are described [GDPR art.5], such as transparency, data minimization, consent, etc., we focus on four principles :

- consent compliance [GDPR art.6] : personal data is used only for purposes the user has given consent to.
- data access [GDPR art.15(1)]: a report is sent *in time* after a user request.
- data erasure [GDPR art.17] : personal data is erased *in time* after a user request.
- storage limitation [GDPR art.5(1)]: personal data must not be stored for *too long* after its last use.

Note that time intervals are specific to the system and must be adhered to without undue delay. The *data subject* (the owner of the personal data) should be informed of the status of his/her request within one month [GDPR art.12(3)], with a possible extension of up to two additional months, if necessary.

The idea of our approach is to automate GDPR compliance checking [1] [3] by storing system data and their dependencies in the form of a provenance graph [2] and specifying GDPR principles as paths to be retrieved in the graphs. Compliance checking is then realized by taking advantage of the efficient reasoning capabilities for path condition resolution provided by Prolog solvers. In Section II, we introduce the data model we use, as well as the specification of the GDPR principles to be checked. We present our prototype in Section III, and apply it on a use case in Section IV. We conclude in Section V.

II. Provenance graph model

In this work, we extend the Open Provenance Model (OPM) [2] with GDPR data. The system information is represented as a directed labelled acyclic graph, called provenance graph, where nodes and edges represent system data and their dependencies. The standard OPM model captures provenance entities called *artifacts*, *processes* and *agents*. Each dependency, with its timestamp(s), shows causality between entities: used (process on artifact) and wasGeneratedBy (artifact on process), where the timestamp indicates when the artifact was used (resp. generated); wasControlledBy (process on agent), where two timestamps give the beginning and the end of the process execution; wasDerivedFrom (artifact on artifact) and wasTriggeredBy (process on process), with a timestamp indicating when the first entity was created (resp. triggered). Note the dependencies may contain a role, used to further specify them. Timestamps are useful for compliance verification, where a total order between processes may be needed.

To reason about personal data and GDPR compliance, we have extended the nodes of the provenance graph with a list of attributes related to the GDPR context. In particular, artifacts that contain personal data are extended with an attribute *personal*, while processes are extended with an attribute *action*, identifying the purpose for which the process is executed. Consent is modeled as an artifact, generated by the consent giving process of the data subject. The consent artifact has an attribute *purposes*, specifying a list of consented purposes for the corresponding personal data. The consent artifact can be updated, thereby creating a new consent artifact (since artifacts are immutable pieces of data in OPM).

Figure 2 depicts a sample of a provenance graph representing an online forum application. User Bob creates an account before joining a group of discussion of interest to him. After creating his account, which implies to enter some personal information such as his phone number and email address, an identifier id_bob is automatically created by the system. User personal information and identifiers are represented as artifacts, with the attribute *personal* set to $True$ (for the sake of readability, node attributes are not depicted in the graph).

Bob is asked to provide his privacy preferences via the filling of a policy template provided by the system (e.g., a cookies acceptance policy). As a result, an artifact *consent_bob_v0* is created with the attribute *purposes*, whose value is a list of pairs (personal data, purpose of use). For instance, Bob gives consent for using his identifier for statistical analysis purposes, but not for sharing with third parties, i.e., purposes = [(id_bob, analysis)].

III. Prototype: Architecture and Implementation

We describe here the components and architecture of our prototype, as depicted in Figure 1.

a) Interface: Via the interface, an auditor can specify the system he wants to verify and optionally a subset of processes

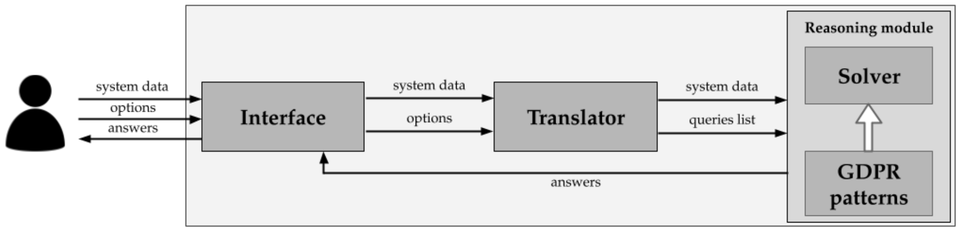


Fig. 1. Prototype Architecture.

(or a subset of personal data) he wants to focus on. The interface retrieves the system log files (in the form of a provenance graph), and possibly specific personal data, users or processes to check (by default, the whole system is audited). The auditor can also choose the GDPR principle(s) he wants to verify via the suitable menu, as depicted in Figure 3(a).

b) Translator: The translator module converts interface inputs into Prolog queries. If no option is specified via the interface, the module returns one query per GDPR principle to check, with no specific parameters (i.e., using variables that Prolog will instantiate by graph nodes). Otherwise, several queries can be returned, with parameters adequately instantiated to cover user selection. Queries and system data are sent to the Prolog solver, using the JPL library.

c) Reasoning module: The reasoning module contains the Prolog solver, which resolves path queries based on the obtained provenance graph, and the GDPR patterns specification. Each GDPR principle is encoded as a Prolog rule. Here is an extract of the pattern concerning consent compliance:

```
consent(DP, PU, T) :-
    wasControlledBy(P1, S, "owner", TB, TE),
    wasGeneratedBy(C, P1, "consent", T), isPurpose(PU, DP, C)
```

When queried, this rule verify if there exists an artifact consent C with the attribute "purposes" containing the purpose PU for the personal data artifact DP. The variables T, TB, TE correspond to timestamps and $P1$ to the consent process controlled by the data subject S.

The solver returns all possible paths matching the query. In case of non-compliance, the user is given enough information to identify the issue (see Figure 3(b)).

IV. Demonstration

Consider an online forum platform, where users can, e.g., create accounts and join discussion groups. We are interested in user Bob and his activities in the system (a snapshot is depicted in Figure 2, where timestamps are expressed in minutes).

Let us suppose we want to check the compliance of Bob's personal data processing w.r.t. Bob's privacy preferences, as registered in his consent (see Figure 3(a)). The reasoning module receives the query, it looks for all processes using Bob's personal data and checks if the purpose of the process has been consented by Bob.

In our example, at time $\texttt{t} = 21$, Bob joins a group, which generates a marketing cookie using $\texttt{DP} = id\_bob$. At time $\texttt{t} = 26$, this cookie is used by the process $\texttt{P} = sendCookie$, associated to a purpose $\texttt{PU} = sendThirdParties$ (provided by the system). The solver tries to instantiate the predicate $\texttt{consent}(\texttt{DP}, \texttt{PU}, \texttt{T})$ with the previous values, however Bob has consented to use his identifier only for *analysis* purposes, i.e., $\texttt{consent}(\texttt{id\_bob}, \texttt{analysis}, \texttt{17})$. As a result, the system is non-compliant. The solver returns the non-compliant process details, displayed in the interface (see Figure 3(b)).

V. Conclusion

We have presented an extension of the provenance model to automate GDPR compliance verification. We have also developed a proof-of-concept prototype to demonstrate the feasibility our approach. Future work includes automating provenance graph generation on various scenarios (e.g., social networks, public services, webstores) for more extensive testing. We also plan to extend the approach to other regulations, such as GDPR-UK or the United States Health Insurance Portability and Accountability Act (HIPPA).

References

[1] D. Basin, S. Debois, and T. Hildebrandt. On purpose and by necessity: Compliance under the gdpr. In *Financial Cryptography and Data Security*, pp. 20–37. Springer Berlin Heidelberg, 2018.
[2] L. Moreau, et al. The Open Provenance Model core specification (v1.1). *Future Generation Computer Systems*, vol. 27, no. 6, pp. 743–756, June 2011.
[3] A. Tauqeer, A. Kurteva, T. Raj Chhetri, A. Ahmeti, and A. Fensel. Automated gdpr contract compliance verification using knowledge graphs. *Information*, vol. 13, no. 10, 2022.
[4] European Union. General data protection regulation, 2016. Accessed: 2024-08-23.

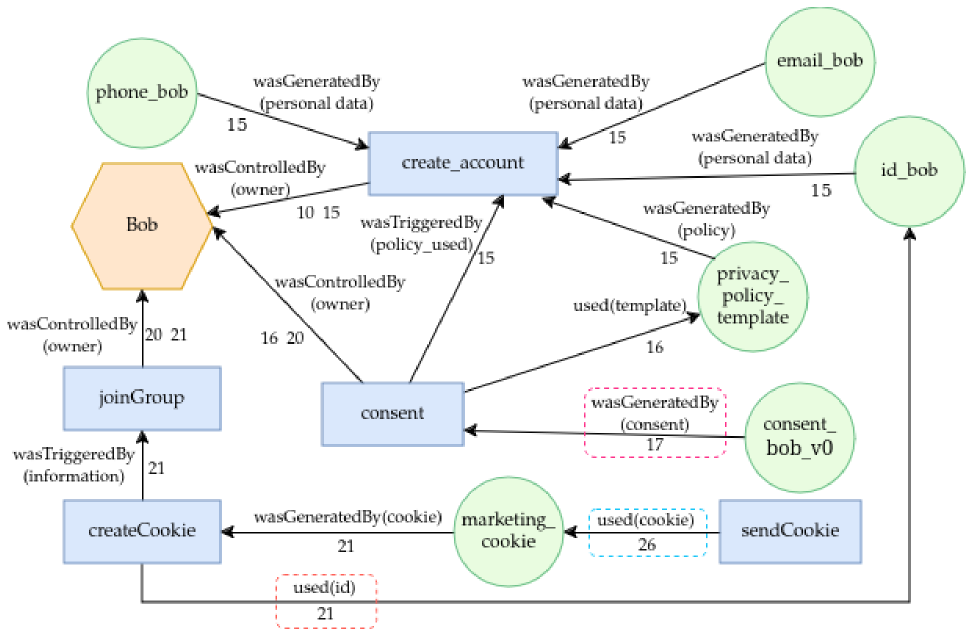
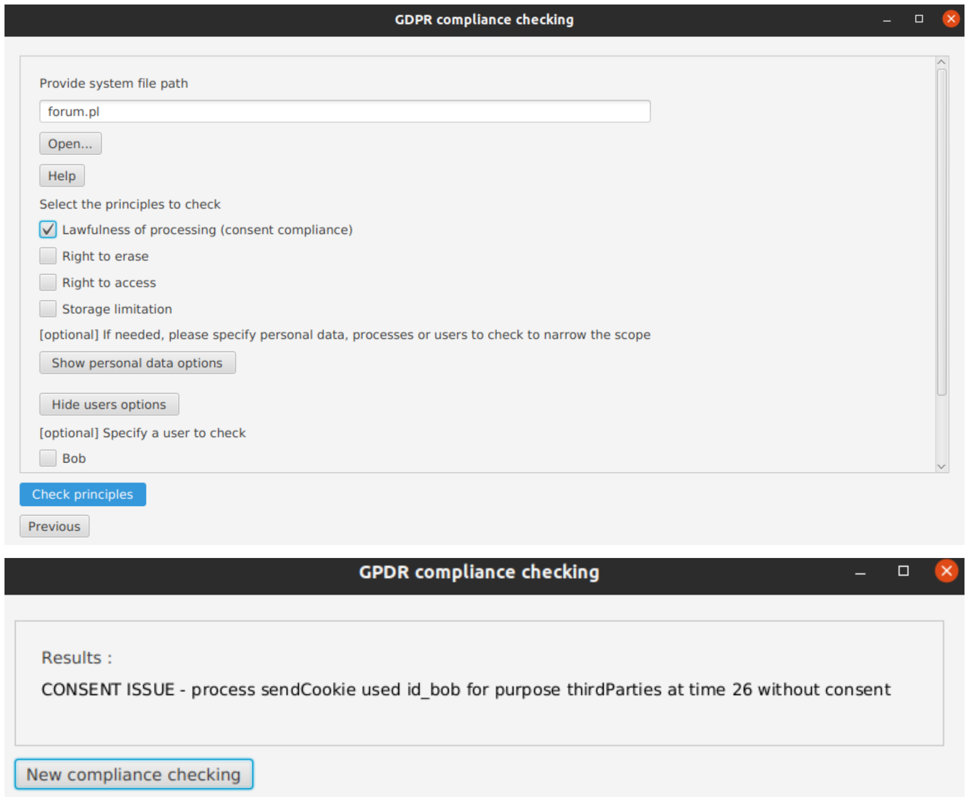

phone_bob
email_bob
id_bob
Bob
create_account
privacy_policy_template
joinGroup
consent
consent_bob_v0
createCookie
marketing_cookie
sendCookie
wasGeneratedBy (personal data) 15
wasGeneratedBy (personal data) 15
wasGeneratedBy (personal data) 15
wasControlledBy (owner) 10 15
wasTriggeredBy (policy_used) 15
wasGeneratedBy (policy) 15
wasControlledBy (owner) 16 20
wasControlledBy (owner) 20 21
used(template) 16
wasGeneratedBy (consent) 17
wasTriggeredBy (information) 21
wasGeneratedBy(cookie) 21
used(cookie) 26
used(id) 21

Fig. 2. Online forum application: provenance graph sample.



forum.pl

Select the principles to check

- [✓] Lawfulness of processing (consent compliance)
- [ ] Right to erase
- [ ] Right to access
- [ ] Storage limitation

[optional] If needed, please specify personal data, processes or users to check to narrow the scope

[optional] Specify a user to check

- [ ] Bob

Results :

CONSENT ISSUE - process sendCookie used id_bob for purpose thirdParties at time 26 without consent

Fig. 3. Interface: (a) option and (b) results screens.