# Policy-Driven XACML-Based Architecture for Dynamic Enforcement of Multiparty Computation

Arghavan Hosseinzadeh[a], Jessica Chwalek[b] and Robin Brandstädter[c]
*Department of Security Engineering ,Fraunhofer IESE, Kaiserslautern, Germany*
{*firstname.lastname*}*@iese.fraunhofer.de*

Keywords: Data Sovereignty, Data Usage Control, Policy Language, Privacy Enhancing Technologies, Multiparty Computation, MYDATA Control Technologies.

Abstract: The need to protect sensitive business and personal information while adhering to data protection regulations, along with the exponential growth of digital data, presents a significant challenge. Data sovereignty addresses this challenge by focusing on safeguarding data across different domains, such as business and healthcare. This objective is accomplished through the specification of Usage Control policies, implementation of data anonymization techniques, and enhancement of policy enforcement in distributed systems. In this work we present a data sovereignty solution that enhances the capabilities of the XACML framework within a data sharing ecosystem. When this solution is realized, the data providers can benefit from dynamic enforcement of Multiparty Computation (MPC) by specifying MPC-enabling policies. Following this approach, the data providers who seek to collaboratively compute a function over their inputs while keeping those inputs private can enforce MPC by specifying a corresponding policy at runtime. Resulting in heightened security and privacy preservation, our solution motivates data providers to engage in data sharing.

## 1 INTRODUCTION

As companies develop innovative business strategies for data monetization, it becomes crucial for them to not only safeguard their confidential business information but also address privacy concerns when dealing with personal data (Parvinen, 2020). Further, there is a growing trend in leveraging AI to study aggregated health data, leading to advancements in patient treatment and care.

While this practice of data sharing has become an integral aspect of business and research endeavors, public opinion tends to be sympathetic toward data sharing as long as privacy protection measures are in place (Kalkman et al., 2022). This presents a unique challenge of balancing data-driven insights against privacy preservation.

According to GDPR, data is of personal nature if it contains information that can directly or indirectly identify a data subject (European Parliament and Council of European Union, 2016). Privacy Enhancing Technologies (PETs) have been developed to help individual users control the amount of personal information they disclose in an online transaction (Seničar et al., 2003) and therefore, they are suitable tools to enforce GDPR requirements. Notably, it is because these technologies have undergone significant improvements in recent years.

In this context, PETs denote a set of technical solutions that assure the privacy of the individuals as well as supporting companies in complying to the data protection regulations. By the nature of PETs, data may, for example, be encrypted and anonymized and thus no longer falls under the GDPR (Veeningen et al., 2018; Thapa and Camtepe, 2021). There are still requirements to be met in order to fully comply with GDPR. These encompass pseudonymizing data, minimizing data collection, and providing data transfer in a secure and compliance-friendly way (Thapa and Camtepe, 2021). Multiparty Computation offers a means of achieving these goals. It guarantees privacy by distributing computation among parties and employing cryptographic protocols to ensure both the accuracy and confidentiality of the computation.

Yet, data providers shall be empowered to adjust the conditions under which their data is used. A comprehensive design for a data provision platform should offer the means to define and ascertain these

[a] https://orcid.org/0009-0001-8699-9972
[b] https://orcid.org/0009-0005-3406-8366
[c] https://orcid.org/0000-0001-8439-3697



Hosseinzadeh, A., Chwalek, J. and Brandstädter, R.
Policy-Driven XACML-Based Architecture for Dynamic Enforcement of Multiparty Computation.
DOI: 10.5220/0012254000003648

In *Proceedings of the 10th International Conference on Information Systems Security and Privacy (ICISSP 2024)*, pages 81-89
ISBN: 978-989-758-683-5; ISSN: 2184-4356

conditions. For instance, data providers might express the need to deny access to their data at any given moment, or to only allow data usage within *Europe* and only for an specific time interval. Moreover, data providers shall be able to demand for a set of obligations. They may require their data to be anonymized before sharing and be deleted after utilization.

In this work, we present an extended Usage Control architecture that can evaluate such policies that explicitly demand to enforce Multiparty Computation (MPC) as a prominent example of PET (Garrido et al., 2022). The data providers who are willing to contribute their data to computations while preserving their privacy can effortlessly specify MPC-enabling policies. Our proposed architecture ensures the effective enforcement of these policies, thereby addressing their demands.

## 2 BACKGROUND

### 2.1 Data Sovereignty

*Data Sovereignty* is generally defined as "a natural person's or corporate entity's capability of being entirely self-determined with regard to its data" (Otto et al., 2017; Zrenner et al., 2019). That is, both individuals with privacy concerns and business data providers that want to keep control of their valuable data benefit from a realization of Data Sovereignty.

In order to implement Data Sovereignty, a Usage Control architecture is necessary. According to Pretschner et al. a Usage Control architecture can either entirely inhibit the utilization of the data or apply modifications, and execute the required actions (Pretschner et al., 2008). Building upon this foundation, Jung et al. describe a Usage Control architecture based on the XACML model (Jung et al., 2022; OASIS Standard, 2013). This architecture, as illustrated in Figure 1, includes several components that work together to enforce Usage Control policies and is the basis of our work.

A *Policy Enforcement Point (PEP)* monitors and intercepts data flows and asks a *Policy Decision Point (PDP)* how to handle the data. Are there any policies restricting the use of the data, subject to specific conditions being met? Are there any policies demanding further modifications (e.g., filtering) of the data?

A PDP evaluates the policies that are stored in a repository (known as *Policy Retrieval Point (PRP)*) based on the information taken from *Policy Information Points (PIPs)*. Consequently, the PDP provides the PEP with the data treatment decision; in addition, it may invoke *Policy Execution Points (PXPs)*

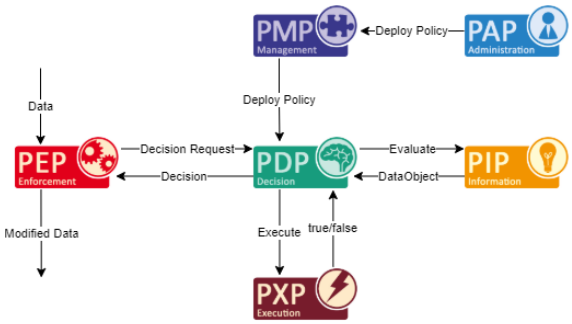


Figure 1: Usage Control Communication Flow.

for executing further obligations (e.g., sending a notification to the data provider when the data usage is allowed). Obligations refer to the extra value that Usage Control adds to the Access Control. After all, a Policy Management Point (PMP) manages these components. Lauf et al. implement this architecture and offer a solution in a scenario that involves medical data (Lauf et al., 2022). They provide patients with three different levels of agreement (i.e., broad consent, vicarious contract, and constrain provision for a project). By reconfiguring their individual policies, patients can control their personal medical data.

Ultimately, Usage Control policies play a pivotal role in the realization of Data Sovereignty. Policies are often represented in the form of *event-condition-action* rules, in which a specific event, in a particular situation, triggers a specific action. According to Singh et al., actions tend toward three categories: reconfiguration, message production, and policy management. Therefore, state changes can alter the set of active policies (Singh et al., 2014). For instance, one can define a context-aware policy as follows: "During a medical emergency situation, the sensitive health data shall be provided without restrictions, but once the emergency situation is over, the data can only be used for research purposes."

Depending on the scenarios and the preferences of data providers, policies may become more complex. These policies may not only permit or deny data usage in different situations, but they may also require data filtering and anonymization.

### 2.2 Multiparty Computation

Secure Multiparty Computation (MPC) is a cryptographic protocol that enables $m$ parties, each party owning a secret $x_i$, to compute a function $f(x_1,...,x_m)$ collaboratively while preserving the privacy of their individual inputs. This has been initially introduced by Yao and has been extended by Goldreich et al., who proposed the multiparty component (Yao, 1982; Goldreich et al., 1987).

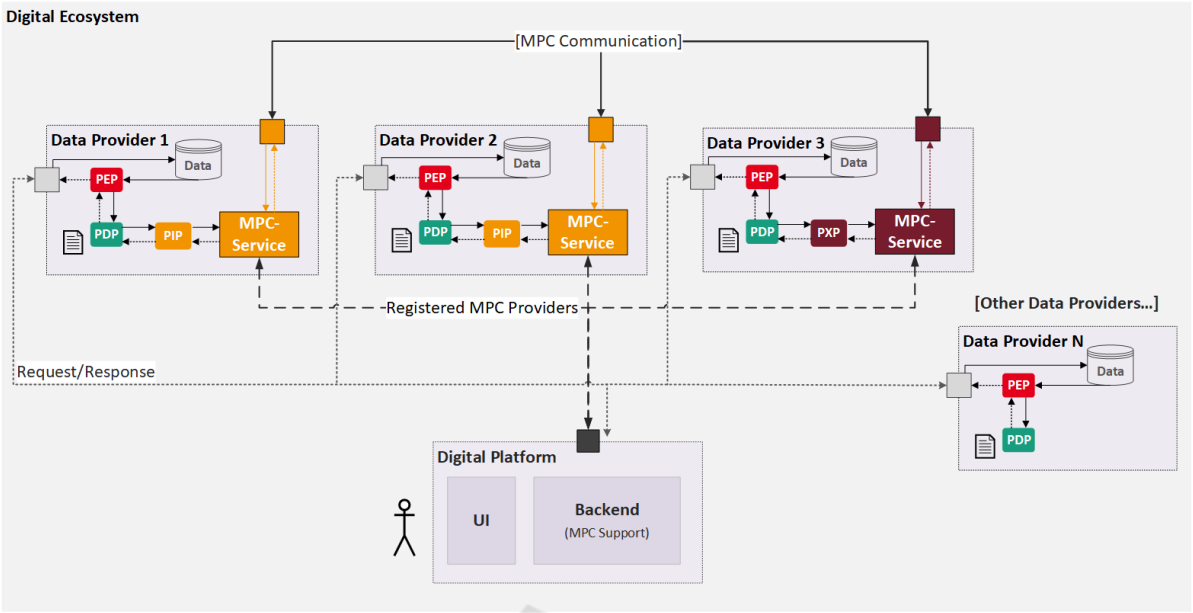


Figure 2: Extended XACML-based architecture to enforce MPC policy.

It utilizes advanced cryptographic techniques to ensure that no party gains knowledge about the private inputs of others. By distributing the computation across multiple parties, MPC allows for joint analysis of sensitive data without compromising privacy. Thus, it comes as no surprise that Deng et al. promote MPC as a countermeasure against linkability and losing confidentiality (Deng et al., 2011).

Today, a multitude of MPC implementations are available, including the EMP-toolkit (Wang et al., 2016) and MP-SPDZ (Keller, 2020) which are publicly available.

MPC has gained significant attention in the field of privacy-preserving computation and secure data analysis. It has applications in various domains, including financial transactions, healthcare data analysis, collaborative machine learning, and more.

For example, Kumar et al. introduce a solution in which the patients' data can be shared with hospitals in encrypted format, so that a proper diagnosis is found (Kumar et al., 2020). The goal is to preserve privacy of the patients while providing them the best possible treatments.

Agahari et al. present an application of MPC in business-to-business data sharing (e.g., in the automotive sector) (Agahari et al., 2022). They implement MPC as part of a data marketplace and show that using MPC can provide better control over data, lower the need for trust in other actors involved, and reduce competitiveness risks. They emphasize how MPC affects the companies' decision to share or not sharing data. However, in their work, the specification of MPC enforcement is static.

Similarly, Koch et al. propose a privacy-preserving data marketplace (Koch et al., 2022a). There, MPC is leveraged to enable data consumers evaluate expressive functions on a set of data. In addition, data providers can selectively authorize specific computations by setting corresponding policies. Analogous to the aforementioned solution, the policies shall be pre-defined and linked to the data upon its registration with the broker.

## 3 POLICY-DRIVEN ARCHITECTURE FOR ENFORCING MPC

In this section, we present an architecture that allows reconfiguration of policies at runtime in order to enforce MPC in a digital ecosystem. Koch et al. define a digital ecosystem as a socio-technical system connecting multiple, typically independent providers and consumers of assets for their mutual benefit (Koch et al., 2022b).

In an ecosystem, digital platforms provide digital services. An example of a digital ecosystem service is the provision of statistical data. This service involves collecting data from various providers, performing functions such as summation or averaging, and delivering the results to data consumers. In this work, we enhance the architecture of a platform that provides such a service to allow the data providers

to keep control over their data. They can store their data and the corresponding policies on their own sites while utilizing MPC for securely performing the requested calculation.

In this work, we present two possibilities to enforce MPC:

- Finite delay and Modification. A PIP wraps the MPC Service and performs the MPC computation synchronously. The PDP waits until the computation is over. The Modification mechanism is used to replace the original confidential value with the PIP result.
- Execution of Action. A PXP wraps the MPC Service and performs the MPC computation asynchronously. The result will be provided to the platform as soon as it is ready.

Figure 2 shows an overview of our proposed XACML-based architecture, to enforce MPC. In the scenario shown, we have a user who has the role of a data consumer in a digital ecosystem. Moreover, we have *N* data providers , each of whom can provide a specific piece of information. When a request is made by the user, the platform knows which providers to contact. These providers, however, do not know each other. Their MPC services will only obtain the information about the other involved parties of the platform if they configure a policy to allow data sharing under the condition of performing MPC.

Every data provider who wants to enforce Usage Control policies must have a policy repository and also requires a PDP component for the policy evaluation. In addition, each data provider must implement at least one PEP that can intercept the data flow and enforce the Usage Control decisions. After these components are implemented, a data provider can reconfigure corresponding policies to either allow or inhibit the provision of the data.

In order to enforce MPC-enabling policies, data providers must wrap their MPC services by either a PIP or a PXP. The computation of MPC is independent of which component triggers the MPC service.

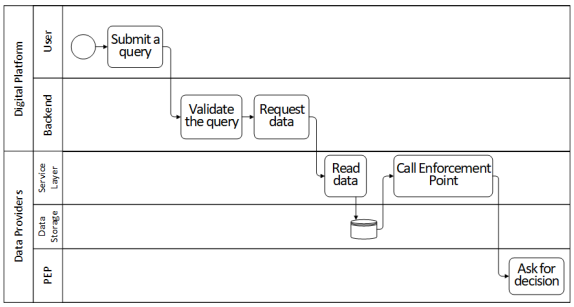


Figure 3: BPMN Diagram - Submit a Query.

To better clarify the process, we explain the stepwise communication process between a user, the backend implementation of a platform, and the Usage Control components of data providers. As illustrated in Figure 3, a user of a digital platform must first submit a query on the platform. For example, there can be a request for "total number of Methicillin-resistant Staphylococcus aureus (MRSA) cases during last year" on a digital ecosystem that connects health care organizations and researchers to exchange health data. Most MRSA infections occur in people who have been in hospitals, doctor's offices, dialysis centers, etc. Hospitals and other data providers may decide to provide this information only via MPC to avoid exposing themselves as origins of MRSA infections.

When a query is submitted, the platform validates it and requests to obtain data from the known data providers. On data provider side, the service layer reads the raw data from their repository and triggers the enforcement point. The PEP then intercepts the data flow and calls the PDP to determine whether the use of the raw data is permitted, prohibited, or subject to further modifications.

The PDP receives the request from the PEP and evaluates the existing policies. More information about the policy specification will be provided in the next section. At this point, the policies are already configured and stored by the data providers. As illustrated in Figure 4, four different decisions can be made depending on the last deployed policies:

1. Permit. The dynamic reconfigurable implementation of Usage Control gives us the ability to *revoke* a policy and simply permit the use of the data without further restrictions. However, we can explicitly allow the provision of original data. So, the PEP provides the raw data without any modification.
2. Inhibit. When the policy inhibits the use of the data, the PDP instructs the PEP to block the data flow. Accordingly, the PEP prepares a message to inform the platform.
3. Enforce MPC using a PIP. The PDP triggers a corresponding PIP that performs MPC synchronously. The PDP provides the raw data which is needed for the computation to the PIP and waits until the computation is completed.
4. Enforce MPC using a PXP. The PDP triggers a corresponding PXP to perform the MPC and provides the raw data to it for the computation. Furthermore, the PDP requests the PEP to inhibit the provision of raw data, and informs the PEP with a message that PXP will perform the MPC compu-

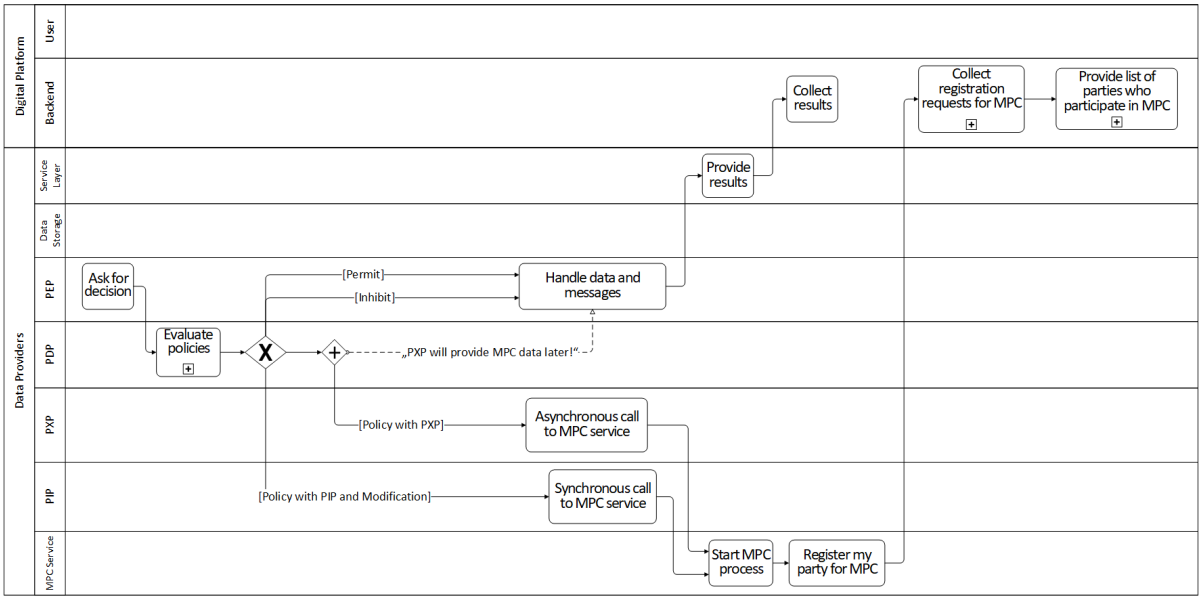


Figure 4: BPMN Diagram - Policy Evaluation.

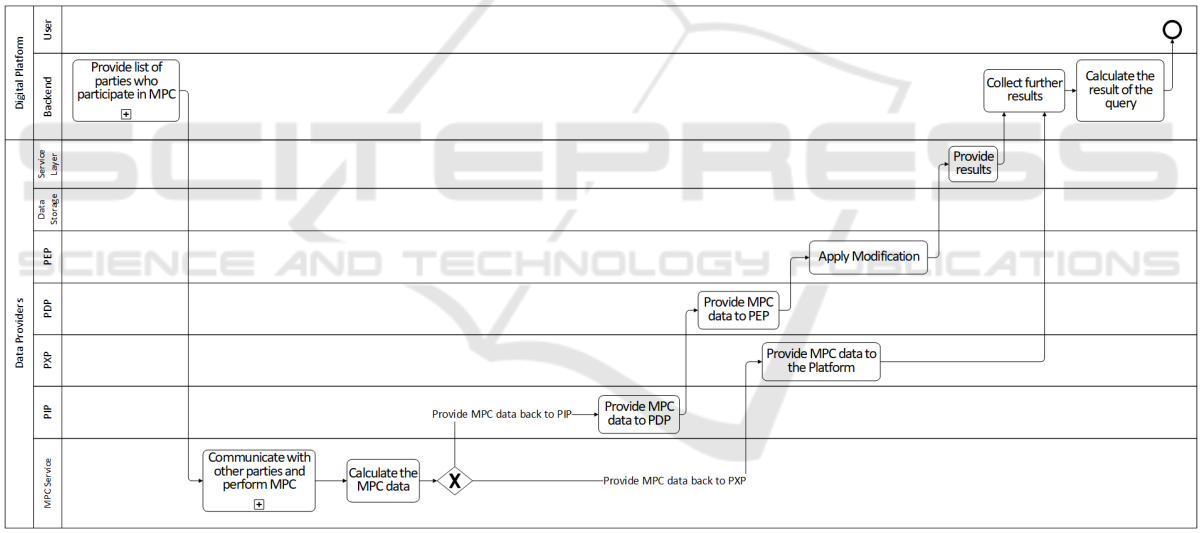


Figure 5: BPMN Diagram - MPC Calculation.

tation asynchronously. According to this decision, the PEP further informs the platform that MPC data will be provided later or can be retrieved from a defined endpoint. Which option is chosen is, of course, an implementation decision.

The platform knows the total number of data providers that are requested to provide their data. Therefore, the platform waits to receive responses from each of them. Some may provide their raw data to the platform and some may inhibit the use of their data. The remaining ones, who decide to perform MPC, must register themselves to the platform. According to the number of registered parties, the platform dynamically initiates the MPC environment. This step is important because only the platform knows which other parties are participating in the MPC and therefore, can provide all of them with the information about other participants.

Now, an MPC communication can start. The MPC services of the involved parties jointly compute the requested function over their inputs, regardless of whether an MPC service is encapsulated by a PIP or a PXP.

As shown in Figure 5, the MPC service sends the calculated MPC data back to the component that initiated the calculation. In case of a PIP, since it is a syn-

```
<policy id='urn:policy:platform:policy1' description='retrict the data usage to
    time and location'>
  <mechanism event='urn:action:platform:provide-MRSA-cases-for-summation'>
    <if>
      <and>
        <date is='before' value='01.04.2024'/>
        <pip:boolean method='urn:info:platform:spatial' default='false'>
          <parameter:string name='spatial' value='Germany'/>
        </pip:boolean>
      </and>
      <then>
        <allow/>
      </then>
    </if>
    <else>
      <inhibit/>
    </else>
  </mechanism>
</policy>
```

Listing 1: MYDATA Policy.

chronous call and the PDP is waiting, the result will be forwarded to the PDP and the PEP, respectively. The PEP applies the modification by substituting the raw data with MPC data and subsequently delivers the MPC data to the platform. In case of a PXP, data will be provided to the platform, as promised.

Depending on the scenario, one can choose to implement the MPC using a PIP or a PXP, since each option has its own advantages and disadvantages. The synchronous implementation may lead to *time-out*. However, the asynchronous implementation requires more complex message processing on the part of the MPC initiator.

# 4 IMPLEMENTATION

MYDATA Control Technologies is a technical implementation of data sovereignty (Fraunhofer IESE, 2018). It is based on IND$^2$UCE framework (Jung et al., 2014) and when realized, it lets users enforce their Usage Control policies while they can reconfigure their intended operations at runtime. In this work, we integrated MYDATA Control Technologies to enforce MPC policies and the additional MPC service is implemented using Cicada 0.8.1 (Berry et al., 2021).

There are two considerations regarding the performance of our solution. Firstly, the dynamic characteristic of this solution imposes a waiting period before an MPC communication begins. It is because the MPC participants are not known in advance and the platform must wait until all providers evaluate their policies and signal their intent to contribute in the MPC computation. Secondly, data providers rely on network communication to engage in MPC. Consequently, the performance is directly influenced by the efficiency of this network communication.

# 5 POLICY SPECIFICATION

A **Policy Administration Point (PAP)** is responsible for policy specification. The PAP of MYDATA Control Technologies offers a user-friendly editor for specifying policies in MYDATA language, which is XML-based and follows the *event-condition-action* schema. Listing 1 shows an example policy specified in MYDATA language. This policy restricts the time and the location of data usage to *before 01.04.2024* and *Germany*, respectively. As mentioned before, a comprehensive Usage Control policy not only expresses the conditions of data provision, but also defines the required obligations and modifications.

For instance, a data provider can specify a policy that not only checks the time and location restrictions, but also enforces MPC. The policy shown in Listing 2 declares a variable called *MpcPipValue* that stores the returned value from a PIP. The PIP is identified with *summationPip*. It provides data from an MPC service and receives three parameters. These parameters that are listed below, are mandatory for performing MPC (e.g., performing a summation function over inputs about MRSA cases).

- Id. This parameter refers to the unique identifier of the request. In case of several requests, a faultless parallel computation is only possible when the id is given.
- Requester. This parameter refers to the platform where the request originated. The MPC service

```
<policy id='urn:policy:platform:policy2' description='Use PIP to enforce mpc for
    data sharing.'>
  <variableDeclaration:number name='MpcPipValue'>
    <pip:number method='urn:info:platform:summationPip' default='NULL'>
      <parameter:string name='id'>
        <event:string eventParameter='requestId'/>
      </parameter:string>
      <parameter:object name='requester'>
        <event:object eventParameter='dataRequester'/>
      </parameter:object>
      <parameter:number name='realValue'>
        <event:number eventParameter='numberOfMRSACases'/>
      </parameter:number>
    </pip:number>
  </variableDeclaration:number>
  <mechanism event='urn:action:platform:provide-MRSA-cases-for-summation'>
    <if>
      <and>
        <not>
          <equals>
            <variable:number reference='MpcPipValue'/>
            <constant:number value='NULL'/>
          </equals>
        </not>
        <!--other conditions-->
      </and>
      <then>
        <modify eventParameter='numberOfMRSACases' method='replace'>
          <parameter:number name='replaceWith' >
            <variable:number reference='MpcPipValue'/>
          </parameter:number>
        </modify>
      </then>
    </if>
    <else>
      <inhibit/>
    </else>
  </mechanism>
</policy>
```

Listing 2: MPC Policy via Policy Information Point.

must register there to obtain information about the other participants and to start the MPC computation.

- RealValue. This parameter refers to the raw data (e.g., the number of MRSA patients occurred in a data provider's work place).

Moreover, the policy shown in Listing 2 specifies that once the MPC is computed successfully and the PIP returns a valid number, then the modification shall apply. As shown in the *then-block* of the policy, a modification method replaces the raw data with the MPC data, in transit.

Similarly, we can define a MYDATA policy that demands to enforce MPC using a PXP. As shown in Listing 3, the policy declares a variable called *MpcPxpValue*. This variable refers to a PXP that is identified with *summationPxp* and wraps the MPC service. The PXP receives the same parameters (id, requester and realValue). This is due to the fact that the MPC service that performs in the background is agnostic to the chosen form of synchronous or asynchronous communication. As mentioned before, the only difference is that the PDP does not wait for the result of the MPC calculation.

The PXP returns *true* when it is successfully triggered. If the conditions specified in the *if-block* are fulfilled, the PDP provides the PEP with an *inhibit* decision with a message attached to it; "PXP will provide MPC data later!". This decision and the attached message indicate that the provision of the raw data is prohibited, though PXP has received the order and MPC will be performed asynchronously. If the specified conditions are not fulfilled (e.g., if the PXP throws an error), the decision is simply to inhibit the data provision.

Depending on the implementation, the message

```
<policy id='urn:policy:platform:policy3' description='Use PXP to enforce mpc for
    data sharing.'>
  <variableDeclaration:boolean name='MpcPxpValue'>
    <execute action='urn:action:platform:summationPxp'>
      <parameter:string name='id'>
        <event:string eventParameter='requestId'/>
      </parameter:string>
      <parameter:object name='requester'>
        <event:object eventParameter='dataRequester'/>
      </parameter:object>
      <parameter:number name='realValue'>
        <event:number eventParameter='numberOfMRSACases'/>
      </parameter:number>
    </execute>
  </variableDeclaration:boolean>
  <mechanism event='urn:action:platform:provide-MRSA-cases-for-summation'>
    <if>
      <and>
        <variable:boolean reference='MpcPxpValue'/>
        <!--other conditions-->
      </and>
      <then>
        <inhibit reason='PXP will provide MPC data later!'/>
      </then>
    </if>
    <else>
      <inhibit/>
    </else>
  </mechanism>
</policy>
```

Listing 3: MPC Policy via Policy Execution Point.

includes information about how MPC data will be delivered; whether the data will be pushed to the platform, or whether the platform can pull it from a defined endpoint.

## 6 CONCLUSION AND FUTURE WORK

In summary, we introduced an extended XACML-based architecture that serves both data providers and data consumers that are involved in a data sharing ecosystem. The raw data remains on the data provider's side and will not be disclosed, even to the platform, without data provider's permission. In case of performing MPC, only the MPC values will be disclosed to the platform.

The Usage Control policies are specified and stored on the data provider's side, as well. Data providers can enforce their restrictions and obligations by reconfiguring their policies at runtime and without the necessity of changing the implementation of their Usage Control components (i.e., PEP, PIP and PXP). Considering the ability to enforce MPC and also to inhibit the provision of data at any time, data providers are encouraged to provide their data, which benefits data consumers as well.

Yet, in this work, we focused on implementing MPC. To expand it, future work can focus on enforcing other PETs such as Differential Privacy or Federated Learning while benefiting from dynamic reconfiguration of Usage Control policies by data providers.

## ACKNOWLEDGEMENTS

This research has been funded by the Federal Ministry for Economic Affairs and Climate Action (BMWK) in the project "Semantische Plattform zur intelligenten Entscheidungs- und Einsatzunterstützung in Leitstellen und Lagezentren" (grant agreement number FKZ 01MK21005B).

## REFERENCES

Agahari, W., Ofe, H., and de Reuver, M. (2022). It is not (only) about privacy: How multi-party computation redefines control, trust, and risk in data sharing. *Electronic markets*, 32(3):1577–1602.

Berry, J. W., Ganti, A., Goss, K., Mayer, C. D., Onunkwo, U., Phillips, C. A., Saia, J., and Shead, T. M. (2021). Adapting secure multiparty computation to support machine learning in radio frequency sensor networks. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States).

Deng, M., Wuyts, K., Scandariato, R., Preneel, B., and Joosen, W. (2011). A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. *Requirements Engineering*, 16(1):3–32.

European Parliament and Council of European Union (2016). 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, L 119, 4.5.2016, p. 1–88.

Fraunhofer IESE (2018). Mydata control technologies. https://www.mydata-control.de/.

Garrido, G. M., Sedlmeir, J., Uludağ, Ö., Alaoui, I. S., Luckow, A., and Matthes, F. (2022). Revealing the landscape of privacy-enhancing technologies in the context of data markets for the iot: A systematic literature review. *Journal of Network and Computer Applications*, page 103465.

Goldreich, O., Micali, S., and Wigderson, A. (1987). How to play any mental game. In *Proceedings of the nineteenth annual ACM conference on Theory of computing - STOC '87*, New York, New York, USA. ACM Press.

Jung, C., Dörr, J., Otto, B., Ten Hompel, M., and Wrobel, S. (2022). Data usage control. *Designing Data Spaces: The Ecosystem Approach to Competitive Advantage*, pages 129–146.

Jung, C., Eitel, A., and Schwarz, R. (2014). Enhancing cloud security with context-aware usage control policies. *GI-Jahrestagung*, 211:50.

Kalkman, S., van Delden, J., Banerjee, A., Tyl, B., Mostert, M., and van Thiel, G. (2022). Patients' and public views and attitudes towards the sharing of health data for research: a narrative review of the empirical evidence. *Journal of Medical Ethics*, 48(1):3–13.

Keller, M. (2020). Mp-spdz: A versatile framework for multi-party computation. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 1575–1590.

Koch, K., Krenn, S., Marc, T., More, S., and Ramacher, S. (2022a). Kraken: a privacy-preserving data market for authentic data. In *Proceedings of the 1st International Workshop on Data Economy*, pages 15–20.

Koch, M., Krohmer, D., Naab, M., Rost, D., and Trapp, M. (2022b). A matter of definition: Criteria for digital ecosystems. *Digital Business*, 2(2):100027.

Kumar, A. V., Sujith, M. S., Sai, K. T., Rajesh, G., and Yashwanth, D. J. S. (2020). Secure multiparty computation enabled e-healthcare system with homomorphic encryption. In *IOP Conference Series: Materials Science and Engineering*, volume 981, page 022079.

Lauf, F., zum Felde, H. M., Klötgen, M., Brandstädter, R., and Schönborn, R. (2022). Sovereignly donating medical data as a patient: A technical approach. In *HEALTHINF*, pages 623–630.

OASIS Standard (2013). extensible access control markup language (xacml) version 3.0. *A:(22 January 2013). URl: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html*.

Otto, B., Lohmann, S., Auer, S., et al. (2017). Reference architecture model for the industrial data space.

Parvinen, P. (2020). Advancing data monetization and the creation of data-based business models. *Communications of the association for information systems*, 47(1):2.

Pretschner, A., Hilty, M., Schütz, F., Schaefer, C., and Walter, T. (2008). Usage control enforcement: Present and future. *IEEE Security & Privacy*, 6(4):44–53.

Seničar, V., Jerman-Blažič, B., and Klobučar, T. (2003). Privacy-enhancing technologies—approaches and development. *Computer Standards & Interfaces*, 25(2):147–158.

Singh, J., Bacon, J., and Eyers, D. (2014). Policy enforcement within emerging distributed, event-based systems. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, pages 246–255.

Thapa, C. and Camtepe, S. (2021). Precision health data: Requirements, challenges and existing techniques for data security and privacy. *Computers in biology and medicine*, 129:104130.

Veeningen, M., Chatterjea, S., Horváth, A. Z., Spindler, G., Boersma, E., van der SPEK, P., van der Galiën, O., Gutteling, J., Kraaij, W., and Veugen, T. (2018). Enabling analytics on sensitive medical data with secure multi-party computation. In *MIE*, pages 76–80.

Wang, X., Malozemoff, A. J., and Katz, J. (2016). Emp-toolkit: Efficient multiparty computation toolkit.

Yao, A. C. (1982). Protocols for secure computations. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*. IEEE.

Zrenner, J., Möller, F. O., Jung, C., Eitel, A., and Otto, B. (2019). Usage control architecture options for data sovereignty in business ecosystems. *Journal of Enterprise Information Management*, 32(3):477–495.