A Federation of Federations:
Secure Cloud Federations meet European Identity Federations

Thomas Zefferer¹, Dominik Ziegler², Andreas Reiter³
A-Sit Plus GmbH¹, Know-Center GmbH² Graz University of Technology³
Austria

Abstract

In the field of information-technology (IT) federation of systems allows combining multiple components in order to provide better service quality or reduce administrative overhead. Two important domains in the field of federated solutions include cloud computing and electronic identity (eID). However, most research in this field has so far focused on one specific domain without considering cross-domain federations. The problem of creating a federation of federations, composed of multiple federated solutions from different domains into one holistic solution has not been adequately addressed up to now. As a result, potentials and limitations of combined federated solutions remain unclear. We bridge this gap by proposing a solution that integrates an existing pan-European federation of national eID systems into a framework that enables secure cloud federations. We further demonstrate the feasibility and great potential of combining multiple federations from different IT domains in a successful implementation and deployment.

1. Introduction

During the past years, the federation of information technology (IT) solutions has become an important topic in both research and industry. Indeed, federating IT services and solutions offers several advantages for both, service providers and users. On the one hand, users can stick with existing solutions they are already used to. On the other hand, federation frameworks can achieve interoperability between different services, enabling service providers to increase their feature set.

A typical example of federated IT systems is electronic-banking (e-banking) solutions. Each bank can operate an individual e-banking solution to serve the specific needs of its customers. Still, customers can initiate financial transactions across different e-banking solutions, as these solutions are federated by means of an appropriate interoperability framework. This way, interoperability between e-banking solutions is achieved, while all customers are still allowed to use the specific e-banking solution provided by their preferred bank.

Two additional domains, for which federation solutions have been investigated recently, are cloud computing and electronic identity (eID). In the field of eID, federation frameworks and solutions assure that users can identify and authenticate with one eID at multiple services from different domains. Popular eID-federation solutions are often based on OpenID Connect [1] or OAuth [2], which e.g. enable users to login to various services with their existing Google or Facebook accounts. During the past years, a federation layer has also been introduced in the European Union (EU) [3], in order to achieve interoperability between national eID systems. Foundations for this eID federation are laid down in the REGULATION (EU) No 910/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC (eIDAS Regulation) [3]. National eID systems assign each citizen an eID that can be used for authentication at public-sector online applications provided in the citizen’s home country. Most national eID solutions also support their integration into electronic services offered by the corporate sector. Examples for national eID systems are the Austrian Citizen Card [4], the Belgian eID card [5], or the Swedish eID [6]. National eID systems usually rely on strong user authentication based on hardware tokens such as smart cards. Unfortunately, existing solutions mainly meet country-specific requirements. Hence, current national eID solutions differ significantly in design and implementation, and thus lack interoperability. The interoperability layer defined by the eIDAS Regulation and developed during the past years addresses this issue and enables a cross-border applicability of national eIDs. All these activities make evident that achieving eID interoperability through federation solutions is a topic of increasing relevance.

In the field of cloud computing, the aim to achieve interoperability between multiple clouds through federation solutions is not yet as widespread as in the field of eID. There are several reasons for that. On the one hand, cloud-service providers do not necessarily benefit from interoperability features but rather aim to lock-in customers and prevent them from using other solutions [7]. On the other hand, especially in private-cloud deployments interoperability with external clouds might raise challenges, when a possible exchange of data between clouds threatens to violate
security requirements. Nevertheless, there exist several use cases, in which users and cloud operators could benefit from federated cloud resources. A prominent example is the public sector, where e.g. federating private clouds of different authorities can enable more powerful and more efficient e-government services. Possibilities to achieve a secure federation of clouds for such use cases have been successfully investigated and developed by the EU-funded research project SUNFISH1. The above-mentioned examples support the statement that achieving interoperability through federation solutions has become a relevant topic in different fields of application. Successfully developed and deployed solutions show that the federation concept already works satisfactorily in several domains. However, it usually remains unclear, if and to what extent federation solutions from different fields can be combined, in order to gain the best possible benefit of two worlds. In this article, we bridge this gap by combining the cloud-federation solution developed by the research project SUNFISH with the European federation solution for national eID systems defined by the eIDAS Regulation [3]. In other words, we enhance SUNFISH by means of strong and reliable user identification and authentication. This way, we make SUNFISH-based cloud federations ready for application in security-critical fields of application.

The remainder of this article resembles the methodology followed and is structured as follows. Section 2 introduces in more detail the two federation solutions that are combined by our work and briefly sketches related work. Subsequently, Section 3 introduces in detail the proposed solution to the problem addressed. This includes the identification of relevant use cases and the derivation of important requirements, on which the proposed solution is based on. Evaluation aspects are discussed in Section 4. Finally, conclusions are drawn in Section 5.

2. Background and Related Work

This article proposes a solution for combining federation solutions from two different domains. The two respective federation solutions are introduced in this chapter in more detail. Section 2.1 focuses on the research project SUNFISH, which has developed a secure and privacy-preserving way to federate arbitrary cloud-computing resources. Subsequently, Section 2.2 introduces the EU eIDAS Regulation, which defines a secure eID federation across Europe and already provides a production-ready solid foundation for state-of-the-art identity management with millions of users.

2.1. SUNFISH: Secure Cloud Federations

SUNFISH was a H2020 project funded by the European Union with the goal to enable secure data sharing and federation of (private) clouds. Key concepts applied by SUNFISH have been discussed by Suzic et al. [8]. SUNFISH ensures that data owners always remain in control of their data. For this purpose, SUNFISH provides data owners with the capability to define in detail who is allowed to access data for which purposes and to which extent. This is achieved by applying an extended data-security governance model. This model relies on classical eXtensible Access Control Markup Language (XACML) [9] based approaches, which are already widely used today in corporations to enforce data-access control. XACML is a de-facto standard in the field of data security policy languages [10, 11]. In earlier days, policies were directly included into applications and were mixed up with application logic. With the introduction of XACML, a decoupled model for policy-based access control is now available. SUNFISH takes advantages of XACML by basing its own mechanisms for data-access control on this technology.

The core of the XACML enforcement model comprises several well-defined entities such as the Policy Enforcement Point (PEP), the Policy Decision Point (PDP), the Policy Administration Point (PAP) and the Policy Information Point (PIP). This separation of components yields a clear assignment of responsibilities. The PEP is the contact point for applications and issues access-decision requests to the PDP. The responses respectively the decisions are then enforced by the PEP. The PEP may be included in applications or may be deployed as a separate component. Both approaches have their advantages and shortcomings. In both cases, responsibilities are shifted to the application. The PEP needs to be included in the application in a proper way in both cases, either as a component or by correctly interfacing with an external PEP. For an application-included PEP, trusted third parties need to assure that the integration is working correctly, and decisions are enforced as expected by the PDP. For application-external PEPs, trusted third parties need to assure that the PEP is contacted for all relevant access decisions.

The policy store itself is logically separated into the Policy Retrieval Point (PRP) and the Policy Administration Point (PAP). The flow for a usual policy-decision request starts at the PEP, where the request is generated, and is passed on to the PDP. The PDP gathers potentially missing attributes from connected PIPs and retrieves a list of matching policies from the PRP. Finally, the evaluation result is returned to the PEP, where the decision is enforced.

1 http://www.sunfishproject.eu/
The description given above elaborates on the generic XACML approach, ignoring any use case specific aspects. SUNFISH has customized this generic approach to meet the specific requirements of cloud federations. Concretely, the model targeted at cloud federations as developed by Suzic and Reiter [12] has been extended with additional components. The extended model provides a concrete implementation of the XACML approach for federated cloud environments. This way, it closes identified gaps of the XACML specification. Figure 1 illustrates this novel model. Tenants, as illustrated in Figure 1, are comparable to security compartments, which group work flows with similar requirements on data security. An infrastructure tenant is used to deploy common services like the policy-decision service and the policy store required for operation of the federation. The Ethereum Blockchain [13] is used to get additional integrity guarantees for stored policies.

All compartments dedicate computational resources, e.g. virtual machines, to the federation. These resources are used to deploy services and applications. The classical XACML model is extended by introducing the entity Policy Enforcement Gateway (PEG). The PEG extends the responsibilities of the PEP by means of gateway functionality. PEGs are deployed at the edge of each security compartment. They protect the entire incoming and outgoing communication of a security compartment.

![Figure 1. SUNFISH governance model](Image)

The PEG analyzes all data traffic and issues policy-decision requests to the PDP. Based on the policies defined by the respective data owner, a decision is derived. This way, it is for instance decided, whether an application from a certain compartment is allowed to access a service deployed in another compartment. Furthermore, it is decided whether a user is allowed to access a particular service in another compartment.

The SUNFISH cloud-federation solution can be operated such that it remains completely transparent to applications deployed and operated in security compartments. This way, SUNFISH supports unmodified legacy applications, which are not aware of the data-security governance infrastructure. At the same time, SUNFISH also supports applications that are already aware of the deployed data-security governance infrastructure. Such applications can even make use of additionally provided out-of-band information. As an example, applications or services can check the authenticity and integrity of received requests by validating an automatically attached digital signature. In either case, communication to other security compartments or services outside of the federation is required to pass through the PEG and is hence subject to enforcement of defined security and data-access policies. This way, many security-related responsibilities are taken from the applications and application developers.

Summarizing, SUNFISH and its data-security governance model enable the federation of clouds while still leaving control of processed and exchanged data in the hands of the respective data owners. Clearly, a major gap is the missing homogenous integration of already existing large-scale identity federation systems. SUNFISH’s data-security governance model is just as strong as the user-authentication mechanisms applied by the federated cloud environments to identify users. In this regard, SUNFISH can benefit from secure and reliable user authentication provided by the federation solution for national eIDs as defined by the eIDAS Regulation.

2.3. eIDAS: Secure Identity Federations

The Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC (eIDAS Regulation) [3] is the legal basis for provision and application of cross-border eID and trust services in Europe. Although covering different kinds of trust services, the regulation has a strong focus on eID and on achieving interoperability between existing national eID systems of EU Member States (MS). Essentially, the eIDAS Regulation defines how a European citizen can use an eID issued by MS X to authenticate at a service provided by MS Y.

Technical foundations of the eIDAS Regulation have been developed in several European large-scale pilots (LSPs) such as STORK² and STORK 2.0³. A thorough overview of these activities is provided by Leitold et al. [14]. The eID-interoperability solution developed by these LSPs has been evaluated by means of several pilot applications. Relevant results and findings from the LSPs and its pilot applications have been incorporated into design and implementation of

---
² https://www.eid-stork.eu/
³ https://www.eid-stork2.eu/
the eID-interoperability framework defined by the eIDAS Regulation.

The general architecture of the eIDAS interoperability framework is illustrated in Figure 2 by means of three exemplary EU Member States. Each MS operates an own identity provider (IDP) denoted as eIDAS Node. Each national eIDAS Node is able to identify and authenticate citizens from the Node’s own MS and accepts user-authentication requests from Service Providers located in the same MS. In order to allow citizens access to services in other MS as well, all eIDAS Nodes form an eID federation.

The functionality provided by this eID federation is explained best by means of a concrete example. Assume that (according to Figure 2) a citizen from MS X accesses a service provided by a Service Provider from MS Y, i.e. SP-Y. To identify and authenticate the citizen, SP-Y sends an identification/authentication request to the eIDAS Node of MS Y, i.e. IDP-Y. As the citizen is from MS X, IDP-Y is unable to identify and authenticate the citizen itself. Therefore, it sends an identification/authentication request to the eIDAS Node of MS X, i.e. IDP-X. IDP-X identifies and authenticates the citizen by means of his or her eID, issues an identity assertion, and returns this assertion to IDP-Y. Due to the established eID federation, IDP-X and IDP-Y can establish a trust relationship and mutually accept issued identity assertions. This way, IDP-Y can finally supply SP-Y with a valid identity assertion. SP-Y can finally use this assertion to identify and authenticate the citizen.

![Figure 2. eIDAS-based eID federation](image)

The eID-interoperability framework defined by eIDAS assures that all European citizens can continue to use their existing national eIDs. At the same time, national IDPs only need to support their own national eID. Support for foreign eIDs is achieved by federating IDPs (eIDAS Nodes) of other Member States. The technical interoperability framework defined by the eIDAS Regulation is currently being set up in EU Member States. Although full coverage has not been reached yet, the potential of the interoperability solution becomes already apparent from various successful deployments.

### 3. Proposed Solution

Combining the concepts of cloud federations and eID federations to a holistic solution has been defined as main goal tackled in this article. By integration the eIDAS-based federation of national eID solutions, SUNFISH’s cloud-federation model achieves a more secure and reliable authentication of users, which, in turn, enables a more reliable application of SUNFISH’s data-security governance model.

This section introduces a solution that achieves this goal by combining the concepts of SUNFISH (cloud federations) and eIDAS (eID federations). For this purpose, relevant use cases and requirements are identified first in Section 3.1. The architecture of the proposed solution that meets the relevant requirements identified in Section 3.1 is then presented in Section 3.2. Finally, typical process flows for the relevant use cases identified in Section 3.1 are discussed in Section 3.3. This way, this section provides a comprehensive overview of the proposed solution.

#### 3.1. Relevant Use Cases and Requirements

The overall goal of the solution proposed in this article is to enhance the existing SUNFISH architecture and functionality with secure and reliable means to authenticate users. Relevant use cases that need to be covered and requirements that must be met by the proposed solution are detailed in this section. Identification of relevant use cases is based on a few general assumptions. While these assumptions are necessary to define and limit the scope of the proposed solution, they do not restrict the solution’s applicability. All assumptions are rather intuitive and implicitly met when relying on eIDAS and SUNFISH.

- **Assumption 1**: Each SUNFISH tenant features some kind of identity provider (IDP). This assumption is met, as soon as users are authenticated by members of SUNFISH-based cloud federations.
- **Assumption 2**: The IDP of a SUNFISH tenant is able to identify and authenticate users originating from this tenant. This assumption should always be met, as identifying and authenticating users is the innate purpose of an IDP.
- **Assumption 3**: Applications deployed in a SUNFISH tenant can request the IDP of the same tenant to identify and authenticate users. This assumption should always be met, as otherwise operating an IDP would be useless.

While the assumptions made provide a solid basis for the proposed solution, they do not limit the
solution’s general applicability. From the assumptions made and taking into account the general SUNFISH architecture as depicted in Figure 1, the following two relevant use cases can be identified:

- **Use Case 1:** A user originating from SUNFISH Tenant A authenticates at an application deployed in SUNFISH Tenant A. This means that user and application originate from the same SUNFISH tenant. Note that an equivalent use case is yielded when a user originating from SUNFISH Tenant B authenticates at an application deployed in SUNFISH Tenant B. Due to their equivalence, these two use cases are not considered separately.

- **Use Case 2:** A user originating from SUNFISH Tenant A authenticates at an application deployed in SUNFISH Tenant B. In contrast to Use Case 1, user and application originate from different SUNFISH tenants in this case. Again, an equivalent use case is yielded when a user originating from SUNFISH Tenant B authenticates at an application deployed in SUNFISH Tenant A, but not considered separately.

Both relevant use cases are illustrated in Figure 3. The first use case resembles a classical setup of an identity management system consisting of a user, a service provider (application), and an identity provider (IDP). The IDP authenticates the user on behalf of the service provider. The main challenge in this scenario is the SUNFISH-specific PEG component acting as intermediary (i.e. gateway) between the user and the application to be accessed. This component must be integrated appropriately into the classical setup of an identity management system.

![Figure 3. Relevant use cases](image)

The second use case is more complicated. In this use case, user and application origin from different SUNFISH tenants. Hence, there is no single IDP that is able to receive identification/authentication requests from the application and identify/authenticate the user at the same time. Instead, two IDPs are involved in this use case. One IDP is able to communicate with the requesting application, the other one is able to authenticate the user. Hence, the two IDPs need to collaborate. In addition, the same challenge as in Use Case 1 applies: the SUNFISH-specific PEG residing between user and application must be integrated into the user-authentication process.

From the relevant use cases identified, relevant requirements can be derived. These requirements must be considered when integrating eID-federation solutions into SUNFISH-based cloud federations. In particular, the two use cases yield the following functional requirements:

- **Requirement 1:** Identity-management functionality must be integrated into SUNFISH-based cloud federations such that SUNFISH-specific architecture components like the PEG are adequately addressed.

- **Requirement 2:** IDPs deployed in different SUNFISH tenants must form some kind of federation, in order to assure that users can identify and authenticate at applications deployed in other SUNFISH tenants.

Especially Requirement 2 suggests reliance on an established eID-federation solution. In Europe, such a solution is defined by the eIDAS Regulation and currently set up in EU Member States. It is hence reasonable to employ the full potential of this approved solution and integrate it into the SUNFISH cloud federation. The architecture that results from combining the SUNFISH cloud federation with the eIDAS-based identity-federation framework is introduced and discussed in the following section.

### 3.2. Architecture

Figure 4 shows the architecture of the proposed solution. This architecture results from combining the SUNFISH cloud federation with the eIDAS identity federation. For the sake of simplicity and clarity, the architecture shown in Figure 4 sketches only two Member States participating in the eID federation. In reality, the federation spans all EU Member States that have successfully set up the interoperability framework defined by the eIDAS Regulation. Figure 4 shows that the proposed solution addresses already on architectural level the two functional requirements defined in Section 3.1.

Requirement 1 defines that SUNFISH-specific components relevant for the functionality of the cloud federation must be adequately taken into account. This especially applies to the PEG, which controls incoming and outgoing data transfers of SUNFISH.
tenants. The proposed solution meets this requirement already on architectural level. According to the architecture shown in Figure 4, the conceptual role of the service provider is transferred from the cloud application to the PEG. Accordingly, the PEG interacts with the corresponding IDP and receives identity assertions issued by this IDP. This implies that the user actually authenticates at the PEG and not at the cloud application itself. By shifting the role of the service provider to the PEG, this component is empowered to implement access-control mechanisms based on the users confirmed identity.

As a downside of this shift, the cloud application no longer receives the identity assertion from the IDP. Consequently, the cloud application cannot use this information to provide identity-based services or to apply additional identity-based access-control schemes itself. To remove this drawback, the proposed solution foresees that the PEG supplies the cloud application with required identity and role attributes. This way, the cloud application can still benefit from the user’s identity information without assuming the role of a service provider. Of course, this approach requires an established trust relationship between the PEG and the application. However, this should not be an issue in practice, considering the central role of the PEG for the respective tenant.

As a supplement to Requirement 1, Requirement 2 defines the need for a federation of involved IDPs. This requirement is met by integrating the eIDAS-based eID-federation framework. Reliance on this framework and on eIDAS-compliant IDPs guarantees that users can authenticate at arbitrary cloud applications, irrespective in which tenant the applications are deployed. The flexibility that comes with reliance on the eIDAS-interoperability framework is illustrated in Figure 4. Connections between PEGs and IDPs indicate that in principle, each PEG can access and use each IDP, yielding an N-to-N mapping. While this is true in theory and fully supported by the architecture, each PEG will most likely be able to communicate with one IDP only in practice. In most cases, this will be the eIDAS Node of the MS, in which the PEG is deployed. However, this does not limit functionality by any means, as all eIDAS-compliant IDPs are federated. Benefits of this federation become apparent from the process description provided in the following section.

3.3. Process Flows

In addition to the architecture introduced in Section 3.2, common process flows of the proposed solution are sketched in the following. Focus is put on the two relevant use cases defined in Section 3.1. The sketched process flows complete the picture of the proposed solution and support a deeper understanding of the proposed solution’s intended functionality.

Use Case 1: In Use Case 1, user and cloud application originate from the same SUNFISH tenant. Hence, both, user and cloud application support the same IDP. In this case, federation of multiple IDPs is not necessary. Still, the process of this use case is detailed in the following to illustrate the use of eIDAS-compliant IDPs within SUNFISH by means of a rather simple example.

All relevant processing steps are illustrated in Figure 5 by means of a simplified sequence diagram. Actors in the sequence diagram correspond to components of the architecture shown in Figure 4. The entire process flow is triggered by the user originating from SUNFISH Tenant A, who attempts to access a cloud application residing in the same tenant.

Following the common SUNFISH workflow, this access request is intercepted by the PEG of this tenant (PEG-A) (1). In order to know how to proceed with the user request, PEG-A requests an access decision from the PDP (2). As the user is not authenticated yet, the PDP denies access and requests user authentication (3). Consequently, PEG-A requests an identity assertion for the user from IDP-X (4). IDP-X authenticates the user by requesting provision of valid user-authentication data (5). The actual form of these data depends on the user-authentication method applied. In the simplest case, the user provides a secret password. In more secure scenarios, provision of user-authentication involves multi-factor authentication schemes, e.g. including hardware tokens such as smart cards. In any case, the user needs to supply IDP-X with valid authentication data that prove the user’s identity (6). After successful validation of the received authentication data, IDP-X creates and signs an identity assertion that confirms the identity of the user.
user (7). Subsequently, IDP-X issues the identity assertion to PEG-A (8). PEG-A sends another access-

decision request to the PDP, now containing confirmed identity information from the user (9). The
PDP is now able to authenticate the user and authorizes access to the requested cloud application (10). When access has been granted by the PDP, PEG-A assembles required identity and role attributes (11). These attributes are added to the user’s application-access request that is then forwarded to the cloud application.

Following these steps, the proposed solution enables eIDAS-based access control to cloud applications. Involved components are largely transparent to the user and assure that the solution can be applied for arbitrary cloud application and eIDAS-compliant IDPs.

**Use Case 2:** Use Case 1 already demonstrates that eIDAS-compliant IDPs can be employed by the SUNFISH cloud federation to implement identity-based access-control mechanisms. However, Use Case 1 does not employ the eID-federation feature that is provided by all eIDAS-compliant IDPs. This aspect is covered by Use Case 2, which assumes that a user stemming from SUNFISH Tenant A wants to access a cloud application located in SUNFISH Tenant B. It is further assumed that the user can only be authenticated by IDP-X, while the targeted cloud application is able to communicate with IDP-Y only. This setup requires IDP-X and IDP-Y to be federated, in order to be able to collaborate, to achieve a successful user authentication, and to enable identity-based access-control mechanisms.

The overall process flow for this scenario is illustrated in Figure 6 again by means of a simplified sequence diagram. As in Use Case 1, the user triggers the entire process flow by requesting access to a cloud application. Concretely, User-A (originating from SUNFISH Tenant A) attempts to access Cloud Application B deployed in SUNFISH Tenant B. As illustrated in Figure 6 and similar to Use Case 1, the access request is intercepted by PEG-B (1). To determine the further processing of the intercepted request, PEG-B sends an access-decision request to the PDP (2). The PDP denies access, as the user is not yet authenticated (3). To authenticate the user, PEG-B requests an identity assertion for the user from the identity provider used in Tenant B, i.e. IDP-Y (4). IDP-Y requests the user to declare his or her origin, which defines the (federated) IDP that is able to authenticate the user (5). The user selects his or her origin and transmits this information to IDP-Y (6). From the obtained selection, IDP-Y determines the IDP that is able to authenticate the user and sends an eIDAS-compliant identification/authentication request to this IDP, i.e. IDP-X (7). IDP-X authenticates the user by requesting respective user-authentication data such as passwords, etc. (8). The user authenticates himself or herself by providing the requested authentication data (9). If the provided data can be verified successfully, IDP-X creates an eIDAS-compliant identity assertion (10). This assertion is transmitted to IDP-Y via the established eIDAS-based eID-interoperability framework (11). If necessary, IDP-Y transforms the received assertion, e.g. to conform to some national protocol used by PEG-B (12). The resulting identity assertion is then returned to PEG-B (13). PEG-B uses this assertion to send another access-decision request to the PDP, this time including obtained user-authentication information (14). As the user is now authenticated, the PDP authorizes the requested access to the application (15). PEG-B assembles required identity and role attributes (16), which are added to the user’s application-access request that is finally forwarded to the cloud application (17).

Following these steps, the proposed solution enables identity-based access control to cloud applications within an eID federation. Again, involved components are largely transparent to the user. This way, the solution can be applied for arbitrary cloud application and eIDAS-compliant IDPs that form an eID federation.

**4. Evaluation**

To evaluate the feasibility of the proposed solution, we have implemented all relevant building blocks and deployed them in an evaluation environment. This way, we have shown that the proposed solution is feasible and that defined processes can be realized in practice.

Figure 7 illustrates the resulting deployment. The deployment spans across Europe by integrating the Austrian and Swedish eIDAS infrastructures with their already available eIDAS Nodes into the SUNFISH infrastructure. These countries have been chosen as their national eIDAS infrastructures are sufficiently set up.

Figure 7. Implementation architecture
In addition to the eIDAS Nodes, each MS is assumed to host a computational SUNFISH tenant in this evaluation setup. An integral part of the SUNFISH model is the Infrastructure Tenant, thus also part of the evaluation setup. The Infrastructure Tenant hosts components related to policy evaluation and supports the PEGs of other tenants in enforcing defined policies.

The deployment illustrated in Figure 7 slightly differs from the solution’s general architecture depicted in Section 3.2. For the sake of clarity, the proposed solution’s architecture has modeled the PEG on a rather high level of abstraction. Figure 7 shows the PEG in more detail and makes apparent that the PEG actually comprises the gateway components itself as well as two proxy components, i.e. the Reverse Proxy and the Transparent SUNFISH Proxy.

The Reverse Proxy serves as a basic router to handle incoming HTTP(S) requests. This approach allows mapping multiple services to the same instance of the Transparent SUNFISH Proxy. Consequently, the Reverse Proxy can provide transport layer security on a domain basis or load balancing for increased scalability and flexibility. Additionally, separating this component entails that services can be migrated fully transparent to the SUNFISH infrastructure, without the need to change existing application configurations.

The Transparent SUNFISH Proxy serves as an intermediate layer between legacy applications and the SUNFISH cloud-federation infrastructure. This architecture enables an efficient integration of legacy applications into the SUNFISH infrastructure, without interfering with existing workflows. Indeed, by aiming at a transparent deployment of service and by bridging the gap between legacy applications and managed cloud federations, a better acceptance of the adoption of the overall system can be achieved.

The solution proposed in this article supports the two basic use cases derived in Section 3.1. Figure 7 shows the more complex Use Case 2, where a user originating from the Swedish tenant aims to access a cloud application operated in the Austrian tenant. This scenario raises a challenge: There is no common IDP that is able to serve the Demo Application and at the same time is able to authenticate User Sweden. The proposed solution still enables a successful user authentication by supporting the following process flow, which is illustrated in Figure 8.

![Figure 8. Implementation process-flow](image)

User Sweden uses a client (e.g. a web browser) that is not aware of the underlying SUNFISH infrastructure. The user enters the publicly available service URL and has the impression that the request is issued to the target application. Actually, the SUNFISH infrastructure intercepts the service call using the Reverse Proxy in SUNFISH Demo Tenant Austria. In fact, the entered URL points to this
Reverse Proxy, there is no public URL to directly access the deployed application. The Reverse Proxy acts as a technical solution to intercept incoming requests (1). The Reverse Proxy forwards the intercepted request to the Transparent SUNFISH Proxy located in the same tenant. The Transparent SUNFISH Proxy acts as adapter for legacy (SUNFISH-unaware) applications and integrates them with the SUNFISH policy-enforcement infrastructure (2). The Transparent SUNFISH Proxy transforms the original request into a format suitable for the SUNFISH cloud-federation infrastructure (3). This application-access request is then forwarded to the PEG endpoint deployed in SUNFISH Demo Tenant Austria (4). The PEG issues an authorization request to the PDP (5). As User Sweden is not authenticated at this point, the PDP denies access to the requested Demo Application (6). Hence, the PEG endpoint requests an identity assertion from the Austrian IDP MOA-ID4, which represents an Austrian eIDAS Node. Note that the actual target application was not involved so far. Furthermore, the user authentication process is initiated without target-application involvement (7). This request sent to MOA-ID relies on an Austrian national protocol called PVP2. MOA-ID initiates the user-authentication process by displaying a web-based selection screen of all available and supported eIDAS-compliant IDPs (8). User Sweden then selects his or her country of origin (i.e. Sweden) (9). Based on this selection, MOA-ID requests the identity assertion from the Swedish eIDAS Node using the SAML2-based interoperability protocol (10).

The eIDAS Node Sweden authenticates User Sweden by means of the user’s national eID (11)(12). After successful verification of User Sweden’s identity, eIDAS Node Sweden creates an eIDAS-compliant identity assertion (13) and forwards it to MOA-ID (14). MOA-ID transforms the received assertion into the national PVP2 format (15). MOA-ID returns the resulting PVP2-based identity assertion to the PEG (16). As a result, the PEG can extract the identity and additional role attributes from the identity assertion (17). Still, the PEG needs to verify if the provided attributes are sufficient to grant access to the Demo Application. Thus, the PEG again issues an authorization request to the PDP, but this time includes available user information (18). If the data provided meets all defined access policies, the PDP grants access to the requested Demo Application (19). In the last step, the PEG forwards the initial user request to the Demo Application located in SUNFISH Demo Tenant Austria and includes the supplied user information.

The successful completion of the process steps depicted above shows that the solution proposed in this article works in practice and enables secure and reliable user identification and authentication in pan-European cloud-federation deployments. While the scenario depicted above has focused on the more complex Use Case 2 (as defined in Section 3.1), the simpler Use Case 1 is similar, but omits interaction with the second eIDAS Node.

The crux in this demonstrator is to showcase the simplicity of including the complex eIDAS workflow into cloud federations and their deployed applications. The invocation of the authentication and authorization procedures is completely controlled by the defined policies and the SUNFISH infrastructure, without changing the deployed application.

6. Conclusions

In this article, we have proposed a solution that combines emerging concepts of secure cloud federations with a pan-European eID federations. Concretely, we have presented a solution that extends SUNFISH-based cloud federations by means of secure and reliable user authentication provided by the eIDAS-based eID-interoperability framework. We achieved an integration of the eID-interoperability framework in full conformance to the SUNFISH processing model. Applications can use the full potentials of the eID federation without adapting their workflow. All aspects are controlled using separate security policies.

For future work, we plan to further increase usability by improving the seamless integration of the eIDAS-based user-authentication process in all relevant SUNFISH use cases. Furthermore, we plan to continue evaluation of the proposed solution within the scope of several productive SUNFISH pilot applications that are to be launched soon.

The solution introduced in this article is a solid foundation for these future activities. It shows by means of a working implementation that cloud and eID federations can complement each other by combining the best of two worlds.

7. References


4 https://joinup.ec.europa.eu/software/moa-idpsso ds de/description


