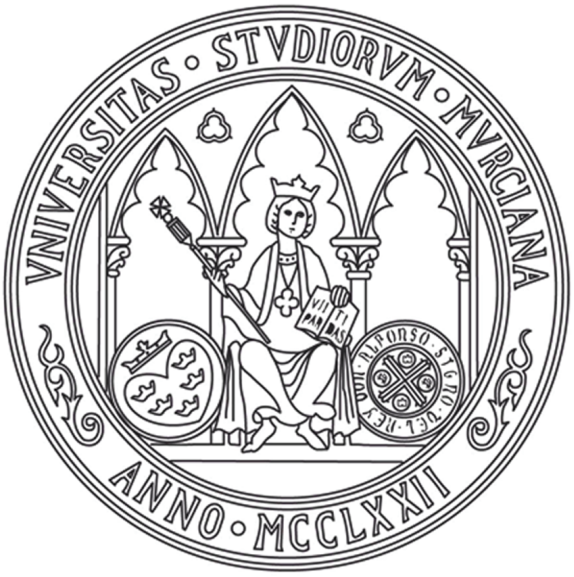


UNIVERSIDAD DE MURCIA

FACULTAD DE INFORMÁTICA

Digital Identity Management Through the Interoperability of Heterogeneous Authentication and Authorization Infrastructures

Gestión de la Identidad Digital a Través de la Interoperabilidad de Infraestructuras Heterogéneas de Autentificación y Autorización

Dña. Elena María Torroglosa García

2017

Universidad de Murcia

Facultad de Informática

DIGITAL IDENTITY MANAGEMENT THROUGH THE INTEROPERABILITY OF HETEROGENEOUS AUTHENTICATION AND AUTHORIZATION INFRASTRUCTURES

GESTIÓN DE LA IDENTIDAD DIGITAL A TRAVÉS DE LA INTEROPERABILIDAD DE INFRAESTRUCTURAS HETEROGÉNEAS DE AUTENTICACIÓN Y AUTORIZACIÓN

Tesis Doctoral

Presentada por:
Elena María Torroglosa García

Supervisada por:
Dr. Antonio Fernando Skarmeta Gomez

Murcia, Mayo de 2017

Resumen

La necesidad de interconectar personas y servicios crece cada día. La compañías están al tanto de la necesidad de unir y ofrecer servicios comunes como forma de ganar nuevos servicios y simplificar su gestión. Del mismo modo, los gobiernos y las instituciones ven la necesidad de migrar sus servicios al mundo digital para cubrir la creciente demanda de administración electrónica, ya que simplifica los procedimientos y ahorra costos.

Además, los usuarios demandan mecanismos que garanticen la privacidad y la seguridad en el uso de los sistemas informáticos, al tiempo que quieren que todos los servicios estén conectados y disponibles. Las instituciones públicas y privadas, así como las empresas trabajan duro para aumentar el uso y la calidad de sus redes y servicios, siempre estudiando nuevas formas de mejorar los recursos existentes y crear otros nuevos. Debido a las diversas opciones de federaciones de identidad con diferentes mecanismos de autenticación, se esta realizando un mayor esfuerzo en homogeneizar e integrar las infraestructuras de autenticación y autorización existentes (AAI).

Cualquier usuario estándar de Internet tiene que gestionar un gran conjunto de nombres de usuario y contraseñas a lo largo de su día en Internet. Los usuarios consumen gran variedad de servicios electrónicos los cuales les obligan a usar diferentes credenciales de autenticación en cada uno con el fin de garantizar la privacidad de identidad digital.

La identidad digital está formada en general por información personal y laboral, contactos, gustos y preferencias. Toda esa información puede ser solicitada por los proveedores de servicios como requisito para proporcionar o personalizar los servicios ofrecidos. La realidad es que cualquier usuario de Internet tiene que compartir parte de su información personal para poder usar los servicios de Internet. Por tanto, los usuarios necesitan usar herramientas específicas para administrar y proteger sus credenciales e información compartida.

Los Sistemas de Gestión de Identidad (IdM, abreviatura del término inglés *Identity Management*) ofrecen a los usuarios herramientas y mecanismos para ayudarles en la tarea de controlar las credenciales y la información personal. Estos mecanismos incluyen desde la gestión de credenciales y la seguridad de la privacidad hasta el *Single Sign-On*, entre

otros. Desde el punto de vista de los Proveedores de Servicios, los Sistemas de Gestión de Identidad permiten simplificar la gestión de los usuarios, gracias a la delegación del proceso de autenticación y el almacenamiento de credenciales.

Los actuales Sistemas de IdM, como parte de Infraestructuras de Autenticación y Autorización (AAI), ofrecen en general funciones básicas relacionadas con la privacidad y el control de los usuarios, mientras que los usuarios y proveedores de servicios exigen cada vez funciones más avanzadas para mejorar sus servicios y la seguridad.

Internet cada día está más arraigada en la vida de las personas, por lo que los usuarios necesitan más control y servicios más variados para seguir migrando. Las funciones avanzadas de gestión de identidades como el anonimato, potentes mecanismos de control de acceso para los usuarios, el SSO entre capas (entre la de red y la de aplicación), la integración transparente entre servicios y federaciones heterogéneos, etc. son exigidos para incrementar el uso potencial de sistemas e infraestructuras relacionados con la identidad digital, la gestión de identidades y las federaciones de identidad.

La integración de protocolos de autenticación y autorización heterogéneos, así como la interoperabilidad de información de identidad de diferentes fuentes no son tareas triviales debido a la gran variedad de tecnologías disponibles. Su integración gracias a los mecanismos de gestión de identidad permiten avanzar hacia nuevos escenarios más ricos y seguros, con nuevos servicios y procedimientos online que simplifican y agilizan la vida de las personas. La gestión de la identidad debe aprovecharse como una tecnología clave para el Internet del futuro, abordando problemas como la interoperabilidad de servicios y tecnologías desde la perspectiva de la gestión de la identidad, manteniendo la compatibilidad a la vez que se crean nuevas infraestructuras de control de acceso que son requeridos por las soluciones IdM.

En un nivel superior, las federaciones de identidad se basan en el establecimiento de acuerdos de confianza entre organizaciones que permiten a cualquier usuario de la federación acceder a los recursos y servicios de cualquier organización federada gracias a una identidad digital única común. Esta identidad federada, válida para todos los servicios dentro de la federación, simplifica el control de credenciales así como la gestión de usuarios por los proveedores de servicios.

En general, una vez que una federación ha sido establecida y desplegada se hace evidente el problema de estar aislada de otras federaciones y sistemas. Cada federación tiene sus propios usuarios y servicios centrados en su entorno específico, por lo que los usuarios de una federación no pueden interactuar con usuarios de otras federaciones y los servicios no pueden ser ofrecidos fuera de los límites de la federación debido a diferentes tecnologías e infraestructuras internas. La existencia de múltiples federaciones crea un problema

de interoperabilidad y es un inconveniente para las soluciones globales. La creación de mecanismos de integración entre federaciones de identidades permite, al mismo tiempo, unir los conjuntos de usuarios y servicios potenciales entre federaciones.

Para resolver el problema de la interoperabilidad, la mayoría de las soluciones de interfederación existentes implican la migración o adaptación de servicios a nuevos mecanismos de autenticación e interoperabilidad. Esto puede ser una buena opción en nuevos despliegues, pero en el caso de federaciones y servicios en producción, puede no ser práctico si implica la modificación de entidades y flujos de usuarios, la migración a nuevos protocolos o la adopción completa de una nueva capa de federación. Todos estos cambios implican generalmente el aumento de la complejidad de funcionamiento, del número de reglas y dificulta los acuerdos políticos.

Esta tesis estudia los mecanismos existentes de gestión de identidad así como las federaciones de identidad más populares para establecer características y necesidades comunes, para sobre ello definir nuevos y mejores mecanismos que permitan una gestión de la identidad y un control de la confianza más rico y potente así como funciones de control de confianza sobre ellos. El objetivo principal es aprovechar las posibilidades de interoperabilidad a diferentes niveles, desde los mecanismos internos utilizados en las federaciones de identidad hasta el nivel de interfederación, para proporcionar privacidad y seguridad a usuarios y servicios.

El trabajo comienza con nuestra colaboración en la definición de una nueva federación de identidad completa denominada SWIFT, centrada en desarrollar mecanismos de gestión de identidad que permitan funciones avanzadas de la privacidad para los usuarios, ofreciendo un control preciso de la información privada del usuario, anonimato, la definición avanzada de políticas para el control de acceso Y SSO entre capas (*cross-layer*); todo desde una perspectiva centrada en el usuario. Como resultado, se ha diseñado una arquitectura IdM ampliada que contiene mecanismos de seguridad y privacidad que responden a los requisitos de las soluciones IdM futuras. En particular, se ofrece una solución de privacidad entre capas e infraestructura de control de acceso que mejora las existentes. Para validar la solución, toda la información y los mensajes intercambiados han sido traducidos a una notación formal y en base a ella, como prueba funcional de funcionamiento se ha implementado y probado un prototipo completo con todas las entidades participantes.

El siguiente paso ha sido trabajar en mejorar el control de la confianza en las federaciones existentes. El objetivo aquí ha sido definir los mecanismos de interoperabilidad para proporcionar seguridad y control de confianza entre sistemas de seguridad heterogéneos conectados a través de un Bus de Servicio Empresarial común (ESB). Para ello, se ha propuesto la integración de la familia de estándares WS-* con el

protocolo OAuth2.0 con el fin de alcanzar la interoperabilidad entre los servicios SOAP tradicionales basados en SAML 2.0 y en certificados X.509 y los nuevos servicios REST basados en OAuth. Esto se ha aplicado a la definición e implementación del Servicio de Seguridad correspondiente al Proyecto GEMBus. Esto se traduce en un Servicio de Tokens de Seguridad (STS) compuesto por el Servicio de Traducción de Token (TTS) y el Servicio de Validación. El STS permite la traducción entre diferentes protocolos de seguridad y el uso de diferentes interfaces de comunicación (SOAP y REST) con el objetivo de interconectar servicios y usuarios de orígenes diferentes. Las funciones del STS son generar nuevas sentencias de seguridad y validarlas, bien por si mismo o delegando en el emisor original de la sentencia, ofreciendo un punto central de confianza para los servicios. Para la validación en este caso, junto a la implementación de un piloto funcional completo, se han realizado mediciones de tiempo que han permitido evaluar las variaciones del tiempo debido a las funciones de integración.

Finalmente, la investigación ha ido un paso más allá hasta el nivel de interfederación. Como veremos a lo largo de este trabajo, las federaciones de identidad están aisladas unas de otras. En general, están diseñados para integrar servicios individuales pero no para interconectar federaciones completas. Para estudiar y analizar el problema cuidadosamente, nos concentramos en dos federaciones importantes y populares, que tienen la particularidad de tener diferentes áreas de uso, lo que hace su interconexión especialmente interesante debido al gran beneficio que la integración implica para ambas federaciones. EduGAIN desde el punto de vista de las instituciones educativas y de investigación, y STORK desde el punto de vista gubernamental son las federaciones de identidad elegidas para trabajar en su conexión. La migración de una federación a otra o el establecimiento de una nueva capa de interoperabilidad entre ambas no son soluciones prácticas debido a su tamaño y al nivel de despliegue que ya tienen. Aunque ambas federaciones comparten puntos en común, usan diferentes restricciones de seguridad, arquitecturas, protocolos de seguridad, identificadores y conjuntos de atributos. El objetivo es establecer los mecanismos necesarios para obtener la interacción entre usuarios y servicios de ambas federaciones de manera bidireccional, manteniendo los requisitos de seguridad y privacidad necesarios, de la manera más transparente y sencilla posible ya que ambas federaciones están en producción. Para lograr esto, una entidad intermedia, denominada *eduPEPS*, ha sido diseñada para actuar como punto de confianza y traducción entre ambas federaciones. Esta entidad se encarga de simular el correcto funcionamiento de cada federación, así como la traducción de mensajes, identificadores y atributos, a la vez que proporciona credibilidad y validez a la información proporcionada a los servicios por los proveedores de la otra federación. Además del despliegue de un piloto internacional entre

Grecia y España para probar la funcionalidad y viabilidad de la solución propuesta, hemos evaluado la calidad de la experiencia ofrecida a los usuarios finales como posible medida que anticipa el éxito de la adopción por parte de los usuarios.

En todos los casos, el proceso de validación se ha centrado en la medición de la usabilidad y la calidad de la experiencia (QoE) desde el punto de vista del usuario, teniendo en cuenta la importancia de las interfaces de usuario como medida que anticipa el éxito de las soluciones diseñadas y su capacidad futura de adopción como un paso más allá de comprobar sólo la viabilidad técnica. En las pruebas, hemos visto que las soluciones propuestas añaden complejidad interna a los flujos de autenticación y autorización que pueden afectar la experiencia del usuario, añadiendo, por ejemplo, nuevos pasos a los flujos de interacción. Aun así, consideramos que las ventajas que ofrecen las soluciones propuestas en términos de simplificación de la gestión de las credenciales, nuevas posibilidades de servicio, sustitución de los procedimientos cara a cara, entre otros, compensan los aumentos en los tiempos de procesamiento o la necesidad de pasos adicionales, los cuales consideramos pueden optimizarse en el futuro.

Dado que las instituciones están demostrando un gran interés en la armonización de federaciones de identidad, pretendemos contribuir en esa dirección continuando nuestra investigación sobre la definición de concordancia entre atributos, prestando especial atención en garantizar la seguridad y la privacidad. Además, las implicaciones legales y filosóficas sobre dónde y cómo se puede desplegar el eduPEPS se deben analizar en detalle. Por último, STORK ha tenido su continuación en la regulación eIDAS, y consideramos que la solución propuesta con el eduPEPS puede ser extrapolada a la integración entre eIDAS y eduGAIN. Aunque eIDAS se basa en STORK, eIDAS tiene sus propias características que requieren una solución personalizada. En este caso, proponemos la incorporación de una nueva entidad que puede permitir que los mecanismos de traducción requeridos en este nuevo escenario resuelvan toda la interoperabilidad técnica ya mostrada con el eduPEPS. Por lo tanto, es necesario analizar en detalle cómo afecta a la integración propuesta y avanzar en esa dirección para alinear la integración con la nueva normativa de la federación. Además, el consorcio GÉANT, con el que colaboramos a través de RedIRIS, está ahora involucrado en GN4-2, un proyecto de 32 meses que comienza el 1 de mayo de 2016 y termina el 31 de diciembre de 2018. Dentro del proyecto se están realizando tareas específicas centradas en soluciones de armonización e interoperabilidad entre federaciones. En particular, el proyecto también analiza las posibilidades de interoperabilidad entre Estados Unidos y Europa a través de la interconnexión de la federación estadounidense InCommons (basada en eduGAIN) y la federación eIDAS, respectivamente. Hay contactos en curso entre varios socios para consolidar este objetivo en un proyecto internacional

llamada ALPHA, por lo tanto, existe un interés real en la evolución de la propuesta de integración descrita en esta tesis como base para trabajo futuro.

La interoperabilidad en los niveles de gestión de identidad y federaciones de identidad permite un mejor control de la privacidad y la seguridad de los datos personales de los usuarios, mejorando y simplificando la gestión y la interacción entre los usuarios y proveedores de servicios. Las administraciones públicas son actores relativamente nuevos en el mundo de la TI, con un gran potencial para armonizar las federaciones existentes, con la garantía de poder ofrecer información de alta calidad para los servicios y al mismo tiempo mantener el máximo respeto por la privacidad de los usuarios, lo que supone un gran impulso para seguir avanzando en la mejora del uso y la gestión de la identidad digital.

Agradecimientos

Quiero dedicar esta página a agradecer a todas las personas que a lo largo de mi etapa universitaria me han acompañado, guiado, enseñado y compartido su amistad conmigo, porque sin ellos, este camino estupendo y también complicado a veces, no hubiera sido así.

Por llevar un poco de orden, quiero dar en primer lugar las gracias a mis compañeros y amigos de carrera, por su amistad que ya perdura media vida; en especial a Jose Miguel, Mario y Abraham.

A mis compañeros de Dibulibu y de la facultad. A Manuel Bernal, a Pedro Martinez y a Alejandro Molina. A Alejandro Pérez porque hemos compartido amistad y proyectos desde el principio. A Jordi por ser buen campañero y mejor amigo, siempre ahí para ayudar y apoyarme, más aún cuando comenzó el otro gran proyecto de mi vida. Especial recuerdo para Manuel Sánchez Cuenca, porque aunque ya no está entre nosotros, siempre le estaré agradecida por apadrinarme y guiarme cuando inicié mi etapa investigadora.

También las gracias a todos los profesores y responsables que me han enseñado y me han conducido a dónde estoy. En especial a Óscar Cánovas, Gregorio Martínez, Diego R. López y sobre todo a Gabriel López, porque será siempre mi ejemplo de meticulosidad y buen hacer.

Por supuesto las gracias a Antonio Skarmeta, mi director tesis y de investigación a lo largo de estos años, por darme la oportunidad de trabajar en tantos y tan buenos proyectos, por traerme hasta aquí y hacer que esto, que era un sueño para mí, sea realidad.

Por último, quiero dar infinitas gracias a mi familia: a mi padre Vicente, a mi santa y maravillosa madre Juana Mari, a mis hermanos Vicente e Isi, y a mi suegra Isabel por cuidar tan bien de mi familia. A mi Eva preciosa que me alegra cada día y me impulsa a ser mejor persona. Quiero detenerme aquí especialmente para agradecérselo a mi marido Felipe que me ha apoyado, cuidado y querido tanto que no tengo palabras para expresarlo.

Por ello, a ti Felipe y a nuestra Eva, os dedico este trabajo.

Abstract

The need to interconnect people and services grows everyday. Companies are aware of the need to unite and offer common services as a way to win new users and simplify their management. Similarly, governments and institutions see the need to migrate their services to the digital world to cover the ever increasing demand for e-management, which streamlines procedures and saves costs.

In addition, users demand mechanisms that guarantee privacy and security in the use of IT systems while at the same time wanting every service to be connected and available. Public and private institutions and companies work hard to increase the use and quality of their networks and services and are always studying new ways to improve existing resources and to create new ones.

Due to the existence of various federation options each with different authentication mechanisms, there has been a huge effort to homogenize and integrate existing Authentication and Authorization Infrastructures (AAI).

Any standard Internet user has to manage a large set of usernames and password throughout her day on the Internet. Users consume a wide variety of electronic services which oblige them to use different authentication credentials in each one in order to guarantee their identity privacy.

Digital identity comprises, in general, personal and working information, contacts, tastes and preferences. All these data can be requested by service providers as requisites to for providing or customizing the service offered. The reality is that any Internet user has to share part of her private information in order to use Internet services, so users need specific tools to manage and protect their credentials and shared information.

Identity Management Systems offer users tools and mechanisms to help them in the task of controlling credentials and personal information. These mechanisms range from the credential management and privacy assurance to Single Sign-On among others. From the point of view of Service Providers, Identity Management Systems allow the simplification of user management, since they assume the delegation of the authentication process and credential storage.

Current Identity Management Systems, such as Authentication and Authorization Infrastructures (AAI), offer in general basic functions regarding user privacy and control. Users and services demand advanced features to improve services and security. Internet is increasingly more deep-rooted in people's lives, so they need more control and more varied services to migrate. Advanced identity management functions like anonymity, powerful access control by users, cross-layer SSO (between network and application layers), transparent integration between heterogeneous services and federations, etc. are demanded to increase the potential use of systems and infrastructures related to digital identity, identity management and identity federations.

The integration of heterogeneous authentication and authorization protocols as well as identity information from different sources are not trivial given the wide variety of technologies available. Their integration thanks to identity management mechanisms allow us to move toward new richer and safer scenarios, with new services and online procedures that simplify and streamline people's lives. IdM must be leveraged as a key technology of the Future Internet, tackling problems like the integration of heterogeneous services and technologies from an IdM perspective as well as backward compatibility and a new access control infrastructure that are required by IdM solutions.

On a higher level, Identity federations are based on the establishment of trust agreements between organizations that allow any user in the federation to access resources and services of any federated organization thanks to a unique digital identity, which is common to the whole federation. This federated identity, valid for all federated services, simplifies the credential control by the user and the user management by service providers.

In general, once a new federation has been established and has deployed the problem of its being isolated from other federations and systems arises. Each federation has its own users, and services focus on its specific environment, but the users from one federation cannot interact with users from other federations, and the services cannot be offered over borders due to different internal technologies and infrastructures. The existence of multiple federations creates the problem of a lack of interoperability and a drawback for global solutions. The creation of integration mechanisms between identity federations allows, at the same time, joint user bases and potential services among them.

To solve the interoperability problem, most existing interfederation solutions imply the migration or adaptation of services to new authentication and interoperation mechanisms. This may be a good option in the case of new deployments, but in the case of running federations and services, it might not be practical if it entails the modification of entities and user flows, the migration to new protocols or the adoption of a complete new federation layer. All these changes usually imply increased operation complexity and more rules, and

makes political agreements more difficult.

This thesis studies the existing identity management mechanisms and the most popular identity federations to establish common features and gaps in order to define new and better mechanisms that enable rich and powerful identity management and trust control functions over them. The main focus is on exploiting interoperability possibilities at different levels, from the internal mechanisms used inside identity federations to interfederation level, and so provide privacy and security for users and services.

The work begins with collaboration in the definition of a new complete identity federation named SWIFT, with the focus on identity management enablers that allow advanced functions for users' privacy, offering fine control disclosure of private user information, anonymity, advanced policy definition for access control and cross-layer SSO; all from a user centric perspective. As a result, an extended IdM architecture has been designed that contains security and privacy mechanisms addressing the requirements of future IdM solutions. In particular, we provide a cross-layer privacy solution that enhances existing work and a new access control infrastructure. To validate the solution, all the information and messages interchanged have been translated to a formal notation and based on this, a complete prototype of all entities has been implemented and tested as a functional test of operation.

The next step was to work on improving the trust control in existing federations. The objective here was to define interoperability mechanisms to provide security and trust control through heterogeneous security systems connected between a common enterprise service bus (ESB). We propose the integration of WS-* family standards with OAuth2.0 protocol to reach interoperability between traditional SOAP services based on SAML2.0 and X.509 certificates and new REST services based on OAuth. This has been applied to the definition and implementation of the Security Service inside the GEMBus Project. It is translated to a Security Token Service (STS) comprising the Token Translation Service and the Validation Service. The STS therefore allows translation between different security protocols and the use of different communication interfaces (SOAP and REST) with the aim of interconnecting services and users from heterogeneous origins. The Security Service generates new security statements and validates them, offering a central trust point for services. The validations can be done by the Security Services themselves or delegated to the original issuer of the security statements. In this case, together with the implementation of a complete and full functional pilot, we have made time measurements to evaluate the increase of time due to the integration functions.

Finally, the research has gone a step further toward the interfederation level. As we see throughout this work, identity federations are isolated from others. In general, they are

designed to integrate individual services but not to interconnect complete federations. To study and analyze the problem carefully, we focus on two important and popular federations, which have the particularity of having different areas of use, which makes their interconnection especially interesting due to the great benefits that the integration implies to both. EduGAIN, from the point of view of educational and research institutions, and STORK, from the governmental point of view, are the identity federations chosen to work with on their interconnection. Migrating from one federation to another or establishing a complete new interoperability layer between both is not practical, due to the size and level of deployment. Although both federation shared points in common, they use different security restrictions, architectures, security protocols, identifiers and attribute sets. The objective here is to establish the mechanisms needed for interaction between users and services of both federations in a bidirectional way, keeping the necessary security and privacy requirements as transparent and simple as possible, since both federations are in production. To achieve this, an intermediate entity, the eduPEPS, was designed to act as a trust and translation point between both federations. This entity is in charge of simulating the proper functioning of each federation as well as the translation of messages, identifiers and attributes, while providing credibility and validity to the information provided to the services by providers from the other federation. In addition to the deployment of an international pilot between Greece and Spain to test the functionality and feasibility of the solution proposed, we evaluate the quality of experience offered to the end users as a possible measure that anticipates the success of adoption by users.

In all the cases the validation process focuses on measuring the usability and the quality of experience (QoE) from the user point of view, taking into account the importance for the user of interfaces as a measure that anticipates the success of the solutions designed and their future adoption capacity as a step beyond merely checking the technical viability. In the tests, we see that the proposed solutions add internal complexity to the authentication and authorization flows that can affect the user experience, adding, for example, new steps to the interaction flows. Even so, we consider that the advantages offered by the proposed solutions in terms of simplification of credentials management, new service possibilities, replacing face-to-face procedures among others, compensate for increases in processing times or the need for additional steps, which we also consider may be optimized in the future.

Since institutions have demonstrated great interest in federation harmonization, the authors aim to help in that sense by continuing our research in this topic with work on attribute mapping definition, and with special attention to security and privacy assurance. In addition, the legal and philosophical implications about where and how the eduPEPS

is deployed are to be analyzed in more detail. Finally, STORK has its continuation in the eIDAS regulation, and the authors consider that the eduPEPS solution can be extrapolated to the integration between eIDAS and eduGAIN. Although eIDAS is based on STORK, eIDAS has its own characteristic, which requires a personalized solution. In this case, we propose the incorporation of a new entity that enables the translation mechanisms required in this new scenario to solve all the technical interoperability already shown with the eduPEPS. Therefore, it is necessary to analyze in detail how it affects the integration proposed and advances in that direction to align the integration with the new federation normative. In additiom, GÉANT consortium, with which we collaborate through RedIRIS, is now involved in GN4-2, a 32-month project beginning 1^{st} May 2016 and ending 31^{th} December 2018. Inside the project there are specific tasks working on interoperability and federation harmonization solutions. In particular, the project is also analyzing the interoperability possibilities between United States and Europe through the interoperation of American InCommons federation (based on eduGAIN) and eIDAS federation respectively. There are ongoing contacts between several partners to consolidate this goal in an international project called ALPHA, so there is real interest on the part of different actors in the work presented in this thesis as starting point for future work.

The interoperation at identity management and identity federations levels allows better privacy and security control of user personal data, improving and simplifying the management and the interaction with users and service providers. Public administrations are relatively new actors in the IT world, with great potential to harmonize existing federations, with the guarantee of being able to offer high quality information for services and, at the same time, maintain the utmost respect for the privacy of users.

Acknowledgements

I want to devote this page to thanking all the people who throughout my university period have accompanied, guided, taught and shared their friendship with me, because without them, this long, sometimes complicated journey, would not have been as great as it is.

In no specific order, I wish to thank all my friends and university friends, for what is now half a lifetime of friendship; especially Jose Miguel, Mario and Abraham.

To my Dibulibu and faculty fellows. To Manuel Bernal, Pedro Martínez and Alejandro Molina. To Alejandro Pérez because we have shared friendship and projects from the beginning. To Jordi Ortiz for being a good workmate and best friend, always there to help and support me, even more when the other great project of my life began. A special memory for Manuel Sánchez Cuenca, because although he is no longer among us, I will always be grateful to him for supporting and guiding me when I began my research stage.

I also thank all the teachers who taught me and led me to where I am. Especially Oscar Canovas, Gregorio Martinez, Diego R. Lopez and even more especially to Gabriel López, because he will always be my example of meticulousness and good work.

Of course thanks to Antonio Skarmeta, my thesis and research director over the years, for giving me the opportunity to work on so many and such good projects, to bring me here and make this, which was a dream for me, a reality.

Finally, I want to give infinite thanks to my family: my father Vicente, my wonderful mother Juana Mari, my brother Vicente and my sister Isi, and to my mother-in-law Isabel for taking such good care of my family. To my precious Eva who makes me happy every day and motivates me to be a better person. I want to pause here to say a huge thank you to my husband, Felipe, who has supported me, cared for and loved me so much that I have no words to express it.

For this reason, Felipe and our Eva, I dedicate this work to you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This first chapter gives a brief introduction to digital identity management and trust control in the context of authentication and authorization infrastructures as a starting point to establish the existing interoperability problem when interoperating between heterogeneous protocols and infrastructures. It also outlines the gaps, drawbacks and shortcomings that have motivated the work of this thesis. It then describes the main objectives of the thesis and its main contributions. Finally, the chapter details the structure of this document, and lists the publications that have resulted from the research carried out.

1.1 Contextualization

The need to interconnect people and services grows everyday. Companies are aware of the need to unite and offer common services as a way to win new users and simplify their management. Similarly, governments and institutions see the need to migrate their services to the digital world to cover the ever increasing demand for e-management, since it streamlines procedures and saves costs.

In addition, users' demand mechanisms that guarantee privacy and security in the use of IT systems, while at the same time wanting services to be connected and available. Public and private institutions as well as companies work hard to increase the use and quality of their networks and services and are always studying new ways to improve existing resources and to create new ones.

Due to the several federations options, with their different authentication mechanisms' appearance, bigger efforts to homogenize and integrate existing Authentication and Authorization Infrastructures (AAI) [5] are arising .

Any standard Internet user has to manage a large set of usernames and passwords

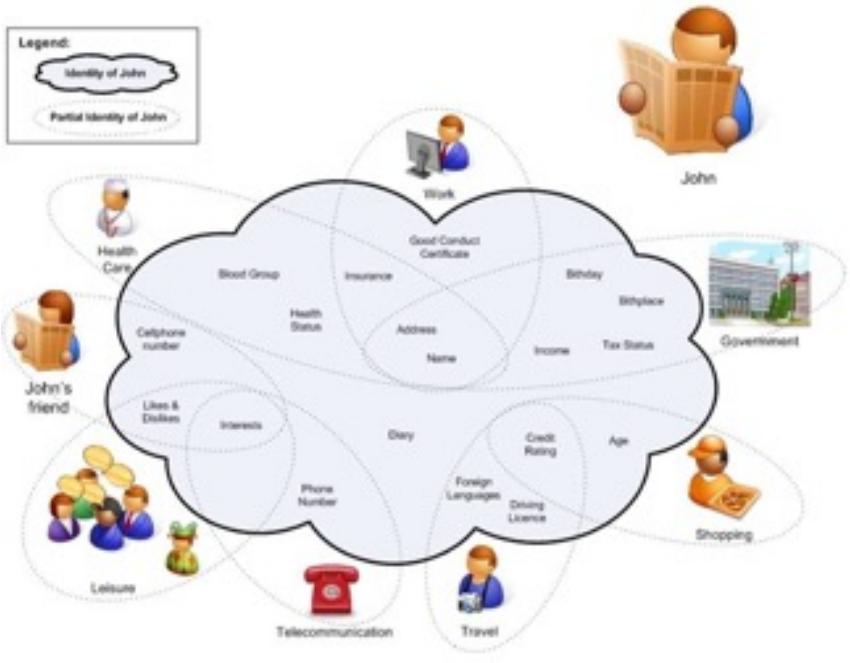


Figure 1.1: Digital identity.

throughout her day on the Internet. Users consume a wide variety of electronic services which oblige them to use different authentication credentials in each one, in order to guarantee their identity privacy. Service and identity providers make use of different authentication and authorization technologies.

Digital identity (see Figure 1.1) is formed in general by personal and working information, contacts, tastes and preferences. All these data may be requested by service providers as requisites to be able to provide or customize the service offered. The reality is that any Internet user has to share part of her private information in order to use Internet services, so users need specific tools to manage and protect their credentials and shared information.

Identity Management Systems, like Authentication and Authorization Infrastructures (AAI), offer users tools and mechanisms to help them in the task of control credentials and personal information. These mechanisms range from the credential management and privacy assurance to Single Sign-On, among others. Identity Management Systems can simplify user management for Service Providers, as they can assume the delegation of the authentication process and the credential storage.

The current Identity Management Systems offer, in general, basic functions regarding user privacy and control. Users and services demand advanced features to improve services

and security. Internet is increasingly more deep-rooted in people's lives, so they need more control and more varied services to migrate. Advanced identity management functions like anonymity, powerful access control by users, cross-layer SSO, transparent integration between heterogeneous services and federations, etc, they are required to increase the potential use of systems and infrastructures related to digital identity, identity management and identity federations.

At a higher level, identity federations are based on the establishment of trust agreements between organizations that allow any user in the federation to access resources and services of any federated organization thanks to a unique digital identity, which is common to the whole federation. This federated identity, valid for all federated services, simplifies the credential control by the user and the user management by service providers.

This thesis studies the existing identity management mechanisms and the most popular identity federations to establish common features and gaps in order to define new and better mechanisms that enable rich and powerful identity management and trust control functions over them. The main focus is on exploiting interoperability possibilities at different levels, from the internal mechanisms used inside identity federations to interfederation level to provide privacy and security for users and services.

1.2 Motivation and problem statement: digital identity management through the interoperability of heterogeneous AAI

The integration of heterogeneous authentication and authorization protocols as well as identity information from different sources are not trivial due to the wide variety of technologies available. Their integration via identity management mechanisms allow us to move towards new richer and safer scenarios, with new services and online procedures that simplify and streamline people's lives. IdM must be leveraged as a key technology of the Future Internet, tackling problems like the integration of heterogeneous services and technologies from an IdM perspective as well as backward compatibility and a new access control infrastructure that are required by IdM solutions.

In general, once a new federation is established and deployed it raises the problem of being isolated from other federations and systems. Each federation has its own users and services and focuses on its specific environment, but the users from one federation cannot interact with users from other federations, and the services cannot be offered

outside borders due to different internal technologies and infrastructures. The existence of multiple federations creates the problem of lack of interoperability and a drawback for global solutions. The creation of integration mechanisms between identity federations allows at the same time to join user bases and potential services between them.

To solve the interoperability problem, most existing interfederation solutions make use of the migration or adaptation of services to new authentication and interoperation mechanisms. This can be a good option in the case of new deployments, but in the case of running federations and services, it might not be practical if it entails the modification of entities and user flows, the migration to new protocols or the adoption of a complete new layer. All these changes usually imply increased operation complexity, a higher number of rules and difficult political agreements.

In particular, taking into account the functional and security features already supported by AAI-based technologies, a solution for *Digital Identity Management through the interoperability of heterogeneous AAI systems* should consider fulfilling the following requirements:

(R1) *Advanced identity management capabilities.* Identity Management Systems offer, in general, basic functions regarding user privacy and control. Advanced functions like anonymity, powerful access control by Service Providers, fine-grained policy disclose definition and cross-layer SSO must be considered to enhance identity management in interoperability solutions.

(R2) *Fine-grained authorization and trust control.* The authorization process intends, on the one hand, to ensure that the end user has the necessary privileges to access an application service under specific conditions, and none other. On the other hand, it allows users to define privacy policies to control personal data disclose. Service Providers need powerful tools to define access control mechanisms to protect their resources. In contrast, Identity Providers have to offer users fine-grained mechanisms to define under what conditions and with whom personal information is shared.

(R3) *Guarantee of more restrictive security requirements.* Taking into account that we are always referring to interoperability and integration, the solutions must ensure that the strictest security requirements are always met so that neither party is unprotected or breaks its security commitments with either end users or Service Providers.

(R4) *Data transport security.* The exchange of sensitive information such as credentials or identity information must be protected to assure its authenticity, integrity, and

confidentiality. That is, they must only be accessible to the intended recipient, and no one else.

(R5) *Reuse of existing infrastructures and standards.* Of course, when we talk about interoperability, it is assumed that we must work with the protocols and infrastructures involved. Even so, it may be necessary to design new components, so the solution will seek the reuse of components and be based on free standards, so that possible improvements revert in the community.

(R6) *Transparent integration.* This requisite refers to how the integration affects users, service providers and identity providers. The proposed interoperability mechanisms should seek to affect as little as possible the entities involved, such that the fewest possible changes are required and these are as subtle as possible so that the integration may even go unnoticed. We believe that transparent integration offers better levels of adoption and greater guarantees of success in its transition to production environments.

(R7) *Usability.* Linked to the previous requisite, it is very important not to lose sight of the usability of the interfaces and the ease of adoption of the proposed solutions. The use of integrating solutions should not imply making the end user experience worse. In contrast, the interoperability has to be seen as an opportunity to offer new features and functions to users and services through their usual input interfaces to the systems, so integration does not necessarily imply an increase in complexity.

(R8) *Protection trust relations control.* The integration solution implies the creation of new trust relationships between the parties involved. It is important to define mechanisms that allow us to respect original trust agreements and to maintain the control of new agreements between the parties involved.

1.3 Objective of this thesis

Although current state of the art approaches provide valid solutions for the *digital identity management through the interoperability of AAI* problem, they also present some gaps (especially related to R1, R2, R5 and R6) that leave an open door for further research and improvements.

To achieve the interoperability of authentication and authorization technologies, and fulfill the stated requirements, the general objective of this thesis can be expressed as:

> *To analyse, design, and validate solutions that enable digital identity management through the interoperability of authentication and authorization infrastructures at different levels, from the improvement of identity management and trust control mechanisms to the interfederation level.*

At internal AAI level, the objective is to offer advanced mechanisms that allow better digital identity management, anonymity, improved access control and SSO cross-layer.

In contrast, the ideal objective in the interoperation of two existing identity federations is to achieve bidirectional matching between identities with the specific properties of transparent, fluid and "on the fly" integration between users and services of both federations, not only allowing basic authentication but also complex scenarios like linking existing accounts and requesting additional attributes to provide identity management. In this case, we can speak of an ideal goal, because we are aware of the difficulty of setting such an ambitious goal, as it will not be possible in all situations.

1.3.1 Specific objectives

The general objective of this thesis can be split into these more specific objectives:

(O1) Analyze most important authentication and authorization technologies used as building blocks for identity management systems.

(O2) Analyze existing identity federations in the area of education, research and government institutions.

(O3) Improve identity management mechanisms with the aim of offering advanced features in the digital identity interoperability area like anonymity, partial identities, fine-grained privacy disclose control, advanced access control and SSO cross-layer. These mechanisms should be validated

(O4) Design a solution enabling trust control in heterogeneous scenarios through the interoperability of different authorization protocols.

(O5) Design a solution enabling interoperation between different identity federations allowing transparent bidirectional authentication and authorization flows, as well as real interoperation between identities from heterogeneous federations.

(O6) Validate the designed solutions by means of analytical models and prototype implementations, taking into account such important factors as functionality, security, feasibility, usability or performance.

1.4 Contributions

In order to accomplish the objectives described in Section 1.3, this thesis provides three separate blocks of contributions

- **To improve Identity Management capabilities**. This block of contributions, described in Sections 3.2, proposes interoperability mechanisms in the area of Identity Management System to integrate different digital identities and services, offering partial identities, anonymity, better privacy and access control. These mechanism have been included in a complete new architecture defined in the SWIFT project.

  In addition to contributing to the SWIFT architecture definition and implementation, the work done in relation to the Deductive Policies was sent to a W3C congress and discussed as a possible contribution to the working draft version of XACML 3.0 standard.

  Next, we summarize the main contribution in this area:

  - ⋆ Analysis of main authentication and authorization protocols in the area of identity management (addressed in P10).
  - ⋆ Advanced mechanisms to Identity Management architecture: partial identities, anonymity, better privacy and access control (P4, P5, P7, P8 and P12).
  - ⋆ Proposal of Deductive Policies as extension to XACML3.0 (P9).
  - ⋆ Definition in formal language of SWIFT architecture.
  - ⋆ Implementation of functional prototype to architecture validation.

- **To improve Trust Management**. This block of contributions, described in Sections 3.3, defines an integrating solution to interoperate different security technologies in the context of the GEMBus project with the aim of offering trust mechanisms to work between heterogeneous authorization protocols. The main contribution of this block is the Security Service, which allows the interoperation between traditional SOAP services based on WS-Trust family standards (SAML2.0, X.509, ...) and modern REST services based on OAuth 2.0 protocol.

  Next we summarize the main contribution in the trust management area:

  - ⋆ Implementation of WS-Trust library.

  - ⋆ Design and implementation of the GEMBus Security Service based on the Security Token Service and formed by the Token Translation Service and the Validation Service (addressed in P6 and P11).
  - ⋆ Proposal of interoperability solution to integrate WS-Trust and OAuth security standards (P2).
  - ⋆ Deployment of a GEMBus pilot to validate the architecture (P2).

- **To improve interfederation interoperability**. This block of contributions, described in Chapter 4 provides a detailed solution to integrate two important federations like eduGAIN and STORK2.0. As our starting point, we provide a survey that analyses the main federations in the area of education, research and government institution. Second, we analyze the main requisites and open challenges to be covered by an interfederation solution. Based on this, the authors propose an integration solution based on the introduction of an intermediate entity, called eduPEPS, that addresses all requisites and solves the translation needs.

  This is the list that summarize the contribution to this block:

  - ⋆ Survey of main identity federation in the area of education, research and government institution including: Moonshot, EUDAT, eduGAIN, STORK2.0 and eIDAS (addressed in P1).
  - ⋆ Study of main requisites and open challenges on the identity federation interoperability problem (P1).
  - ⋆ Analysis of similarities and differences between eduGAIN and STORK2.0 federations, focus on: protocols, architecture, entities, identifies and attributes (P3).
  - ⋆ Proposal of a solution to integration eduGAIN and STORK federations: the eduPEPS (P1 and P3).
  - ⋆ Design and implementation of eduPEPS software.
  - ⋆ Design and deployment of eduPEPS pilot, together with eduGAIN ans STORK 2.0 testbeds as proof concept and as medium to test and validate the proposed interoperability solution (P3).

1.5 Thesis structure

The remainder of this Thesis is organized as follows:

Chapter 2 is devoted to the background and state of the art, and provides a brief description of the most relevant technologies related to the identity management and identity federation and the authentication and authorization mechanisms associated.

Chapter 3 analyses the interoperability problem at internal federation level, proposing several mechanisms that enable advanced identity management features and better trust control, applied to two different projects: SWIFT and GEMBus. Each proposal includes implementation and validation tests to prove the viability of the solutions proposed.

Chapter 4 analyzes the problem of interoperability between identity federations, with the focus on the interconnection of eduGAIN and STORK2.0 federations. Together with the detailed analysis of the similarities and differences between both federations and the proposal of the mechanisms necessary to reach different scenarios and use cases, a validation of the solution is offered on a fully functional prototype of the integrated architectures.

Finally, Chapter 5 summarizes the thesis proposals and results, and gives some indications for future work.

1.6 Related publications

The research work carried out in this thesis has led to the publication of various papers at conferences, in research journals, and as book chapters. The most relevant contributions are presented below, in chronological order.

Indexed Journals (JCR)

(P1) Elena M. Torroglosa-Garcia, Antonio F. Skarmeta-Gomez.**Towards Interoperabilty in Identity Federation Systems**. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 8(2). 2013. Listed in Q2 (96/203) of the SJR 2015, area of Computer Science, Computer Networks and Communications. Pending publication in June 2017 [6].

Digital services aimed at humans need to ensure user identity. Governments and institutions confront the identity problem when migrating their face to face services to the digital world, where no facial identification is plausible. On the other hand, users concerns regarding their privacy and security are a barrier to be overcome during the migration. Identity federations are envisioned to unify and simplify user and service management through trust relationships. Recent trends indicate that federations are limited by target audiences and scope and are isolated from each other. It is necessary to go one step further and work in interoperability

mechanisms to develop the existing federations and improve user experience and service quality. This work reviews some of the most important identity federations, with the focus on well defined sectors such as research and education communities and governments, specifically Moonshot, eduGAIN, EUDAT, STORK and EIDAS. Based on their analyses, we consider interfederation scenarios between eduGAIN, STORK and eIDAS and propose interoperability mechanisms to reach interfederation solutions to extend the user's scope of each or with the others, and thus provide wider federation possibilities.

(P2) Elena M. Torroglosa-Garcia, Antonio D. Perez-Morales, Pedro, Martinez-Julia, and Diego R. Lopez. **Integration of the OAuth and Web Service family security standards**. *Computer Networks*, 57(10) 2233-2249, 2013 July 5th. ISSN 1389-1286. Listed in Q1 (42/190) of the SJR 2013, area of Computer Science, Computer Networks and Communications [7].

There are more and more scenarios requiring the transparent integration of heterogeneous security services in order to facilitate application development, simplify deployment and provide a seamless user experience. One of the most common cases occurs when resources make use of OAuth to provide a simple and flexible way to authorize clients in order to access protected resources. But different OAuth implementations normally use distinct types of authorization grant and access tokens. This heterogeneity can be tackled by leveraging on WS-Trust, which is especially intended to offer integration mechanisms among services that implement WS-* specifications. By integrating these mechanisms it is possible to reduce the complexity supported by the OAuth Authorization Server (AS), so easing the interoperability through the delegation of the issuance and validation processes. This work also proposes a solution to cover the needs of WS-Trust clients who intend to use OAuth resources.

(P3) Elena M. Torroglosa-Garcia, Jordi Ortiz-Murillo, Antonio F. Skarmeta-Gomez. **Matching federation identities, the eduGAIN and STORK approach**. Second review ongoing in *Journal of Future Generation Computer Systems*. Listed in Q1 (23/203) of the SJR 2015, area of Computer Science, Computer Networks and Communications.

Several identity federations with different authentication mechanisms exist in the area of governments and educational institutions. STORK from the European administration side and eduGAIN from research and educational institutions

side are the main exponents in their areas. Both federations are investing on harmonizing and integrating their federations with other Authentications and Authorization Infrastructures (AAI) to improve and gain users and services. This paper analyses each federation and different integration scenarios proposing an interfederation solution that interconnects both federations through ad-hoc interoperability mechanisms. These mechanisms are focused on translating and matching user identities. In addition, an international testbed has been deployed with the aim of probing the viability of the solution and its quality has been measured from the user experience point of view in order to estimate the impact on user adoption. The proposed integration allows users belonging to one federation to access services in other federations, which implies an explosion of user and services in the receiving federation. Allowing users from other federations permits at the same time the conversion of some face to face procedures to online transactions thanks to the strong authentication mechanisms. The mechanisms proposed do not guarantee the integration transparently for all the cases due to security restrictions of each federation and it will imply as future work the study of political, legal and administrative implications of a real integration between both federations.

Non-indexed Journals

(P4) Alejandro Pérez, Elena Torroglosa, Gabriel López, Antonio Gómez-Skarmeta, Joao Girao, Mario Lischka. **SWIFT - Advanced services for identity management**. *Upgrade*, XI(1), 13–20. 2010 February [8].

Traditional solutions for identity management, based on the end user authentication, usually by means of credentials such as username and password, have significantly improved in recent years with the incorporation of SSO (Single Sign-On) mechanisms and the concept of identity federations. However, both providers and end users are demanding additional services that are not yet available in current solutions. These additional advanced services such as anonymity, authorization based on end user attributes, and cross-layer SSO, would improve the usability and security of these systems. The SWIFT (Secure Widespread Identities for Federated Telecommunications) project aims to offer an identity management frame- work in which all these advanced topics can work together.

(P5) Alejandro Pérez, Elena Torroglosa, Gabriel López, Antonio Gómez-Skarmeta, Joao Girao, Mario Lischka. **SWIFT - Servicios avanzados para la gestión de identidad**. *Novatica*, 202. 2009 November. ISSN 0211-2124 [9].

This paper includes the same content as the previous one published in Upgrade with the title "SWIFT - Advanced services for identity management" since both, Upgrade and Novatica journals had publication agreements and they requested a translated version of the paper.

Conferences

(P6) Mary Grammatikou, Costas Marinos, Pedro Martinez-Julia, Jordi Jofre, Steluta Gheorghiu, Diego R. Lopez, Yuri Demchenko, Krzysztof Dombek, Roland Hedberg, Antonio F. Skarmeta, Elena Torroglosa. **Composable Network Services Framework: GÉANT Multi-domain Bus (GEMBus)**. *18th International Conference on Parallel and Distributed Processing. Techniques and Applications* (PDPTA'12). USA. 2012 July 16-19th [10].

Service-Oriented Architecture (SOA) has become a common technology for provisioning infrastructure services on demand. Service-Oriented Architecture (SOA) allows managing, maintaining and accessing heterogeneous and geographically sparse resources in a unified way. This paper introduces GEMBus (GEANT Multi-domain Bus), a service-oriented middleware platform that allows flexible services composition, and their on-demand provisioning and deployment to create new specialized task-oriented services and applications. GEMBus is built upon state-of-the-art Enterprise Service Bus (ESB) technologies and extend them with new functionalities that allow dynamic component services deployment, composition and management.

The GEMBus architecture incorporates different ESB instances at different management domains, orchestrated by the GEMBus core, constituted by the elements which provide the functionality required to maintain the federation infrastructure: service registry, message infrastructure, security service, accounting service, and composition services. The paper also discusses the general case for integration of Service-Oriented Architecture (SOA) principles and technologies with the provision and deployment mechanisms to support on- demand infrastructure services provisioning. This architecture has been validated with a series of use-cases in the GÉANT environment and is about to be applied to similar infrastructures in a wide range of application fields within the academic and research community.

(P7) Elena Torroglosa, Alejandro Pérez, Gabriel López, Antonio Gómez-Skarmeta, Óscar Cánovas. **SWIFT: Advanced identity management**. In proceedings

CHINACOM 2010. 2010 August [11].

Identity Management (IdM) emerges from the necessity to establish control of digital identities, protecting personal information and establishing confidence in the services providers. The SWIFT framework goes a step further of the traditional IdM solutions, and provides an environment for an advanced management of end users' identities. This framework will mainly provide end user identity aggregation from different individual identities, anonymous services access and cross-layer authentication and authorization.

(P8) Marc Barisch, Elena Torroglosa, Mario Lischka, Rodolphe Marques, Ronald Marx, Alfredo Matos, Alejandro Perez, Dirk Scheuermann. **Security and Privacy Enablers for Future Identity Management Systems**. In proceedings of *Mobile Summit 2010*. June 2010 [12].

In recent years, Identity Management (IdM) has gained a lot of attention in industry, standardization and academia. In particular, two research projects, such as Daidalos or Prime, have invested considerable effort to bring IdM forward, to take advantage of features like improved usability and security. Nevertheless, there are important issues that have not been addressed so far. The SWIFT project leverages IdM as a key technology of the Future Internet, tackling problems like the integration of the network and application layer from an IdM perspective as well as the use of electronic identity cards. Moreover, aspects like the integration of several user devices, backward compatibility and new access control infrastructure are required by future IdM solutions. We consider all these aspects by extending existing IdM solutions with six new security and privacy enablers that are part of the overall SWIFT framework. These enablers have been partially implemented towards a new IdM architecture. First evaluation results of the implementation are promising to pave the way towards future IdM solutions.

(P9) Mario Lischka, Yukiko Endo, Elena Torroglosa, Alejandro Pérez, Antonio G. Skarmeta. **Towards Standardization of Distributed Access Control**. *W3C Workshop on Access Control Application Scenarios*. 2009 November 17-18th. Luxemburgo [13].

This position paper presents a new approach on deduction of policy decisions in a distributed environment. This approach is based on an extension of the well known XACML, enhancing it with the key feature of distributed policy decisions as well as attributes from different domains. The key idea is to combine the policy decision of

another Authoritative Domain into those of the local domain. This introduces a new level of abstraction to the XACML based policies. The work on distributed policies is being carried out within the SWIFT project.

Book chapters

(P10) Elena M. Torroglosa-García, Gabriel López Millán. **Web Service Security: Authentication and Authorization Technologies**. *Network Security Technologies: Design and Applications*. IGI Global, 2014. 108-128. 2014 May 7th. DOI: 10.4018/978-1-4666-4789-3.ch008 [14]

The high adoption in our daily lives of services offered by the Web 2.0 has opened up a wide field for the proliferation of new Web-based services and applications. Social networks, as the main exponent of this new generation of services, require security systems to ensure end user authentication and access control to shared information. Another feature that is becoming increasingly important in these scenarios is the delegation of controlled access between the different API (Application Programming Interfaces) to integrate services and information. The safe use of these Web services requires end user security credentials and different authentication and authorization technologies. This chapter provides an introduction to the most relevant protocols and standards in the area of Web service security, which are able to provide authentication and authorization mechanisms.

(P11) Y. Demchenko, C. Ngo, C. de Laat, P. Martínez Julia, Elena Torroglosa, A.D Pérez Morales. M.Grammatikou, J.Jofre, S.Gheorghiu, J.A. Garcia-Espin. **GEMBus based Services Composition Platform for Cloud PaaS**. Book: *"Service Oriented and Cloud Computing"*, pp.32-47. Springer Berlin Heidelberg. 2012 September 19-21th. DOI: 10.1007/978-3-642-33427-6_3. ISBN 978-3-642-33426-9 [15].

Cloud Platform as a Service (PaaS) provides an environment for creating and deploying applications using one popular development platform. This paper presents a practical solution for building a service composition platform based on the GEMBus (GEANT Multi-domain Bus) that extends the industry accepted Enterprise Service Bus (ESB) platform with automated services composition functionality and core services to support federated network access to distributed applications and resources, primarily targeted for GEANT research and academic community. The ESB is widely used as a platform for SOA and Web Services based integrated enterprise

solutions. However in existing practices ESB design is still based on manual development, configuration and integration. GEMBus with its extended functionality and orientation on distributed resources integration can be considered as a logical choice for creating cloud PaaS services composition and provisioning platform. The paper describes a Composable Services Architecture that creates a basis for automated services composition and lifecycle management and explains how this can be implemented with GEMBus. The paper describes the combined GEMBus/ESB testbed and provides an example of the simple services composition.

(P12) Antonio F. Gomez-Skarmeta, Alejandro Perez Mendez, Elena Torroglosa Garcia, Gabriel Lopez Millan. **User-centric privacy management in future network infrastructure.** Book: George O.M. Yee. *Privacy Protection Measures and Technologies in Business Organizations: Aspects and Standards.* 2011 December. ISBN13: 9781613505014 [16].

Identity management is becoming increasingly important every day. Users need a way to centralize the management of their identity information, such as simplifying the access to services with mechanisms like Single Sign-On. Organizations need a means of obtaining reliable information about users of their services. While this is the main concern for service providers, users are more worried about how their information is treated, what information is provided to what entity, and how privacy is assured in general. IdM provides the means for adequate privacy protection.

While several IdM (Identity Management) solutions that work at Web layer exist, they usually lack integration with network layer services. Future network infrastructures should integrate IdM functionality in such a way that the user is provided with a unified and simplified vision of his identity, which results in an improved privacy and security protection. The IdM framework defined in the SWIFT (Secure Widespread Identities for Federated Telecommunications) project provides the means for cross-layer identity management as well as a set of advanced identity management concepts allowing improved privacy protection, simplification of user interactions, and extensible architecture.

Finally, it is analyzed how the inclusion of IdM in business organizations can provide economical benefits. These benefits range from a reduction in resource requirements to the increment of potential clients thanks to the incorporation of the organization in an identity federation. Special attention is placed on the case where the telecommunications operator is established as the main point of identity

providing, as a straightforward result of its already established trust relationships with a wide range of parties (clients and service providers).

Chapter 2

Background and State of the Art

The increasing importance of new services on the Internet, mainly due to the increasing popularity of social networks [17], results in users having to manage many accounts associated to these services, which contain information about their own identity. This situation leads to bad management of these accounts by end users and, therefore, to security risks [18].

In contrast, service providers have traditionally based end user access to services on end user information, linked to end user identity, which is used for authentication purposes. Nowadays, more and more providers are requesting solutions to incorporate additional end user information in order to decide whether or not an end user is allowed to access the service, that is, not just making use of login and password, but using additional attributes (such as organization, role, age, etc.) thus incorporating a complete authorization process.

Identity management is an aspect of Internet service provision that causes concern to network and security administrators of organizations willing to provide services over the Internet [19]. From the point of view of both organizations and end users, management of user accounts is of paramount importance both to protect service providers against malicious users, and to protect users against providers that want to obtain information [20] about user profiles, behavior patterns, site visited sites, etc.

It is also important to note that service providers are forming identity federations, in order to facilitate user access to different services by means of Single Sign-On (SSO) mechanisms [21]. These identity federations try to partially mitigate the problem of the existence of multiple user accounts, but they rarely offer solutions for the management of authorization, privacy nor anonymous access to services [19].

This thesis focuses on providing the necessary mechanisms to allow interoperability between different mechanisms of authentication and authorization in order to improve

identity federations internally and go a step further toward the interconnection between identity federations. Therefore it is necessary to perform a preliminary analysis of the different technologies available in the area of authentication and authorization as well as existing identity federations in order to get a consistent and solid foundation on which to base the rest of the work.

The following subsections introduce some of the most relevant definitions, protocols and technologies that are used in identity management systems and identity federations mainly focus on Web services, providing frameworks to manage end user's authentication and authorization requirements.

2.1 Identity Management

End users make use of a wide variety of Internet services. For each, a registration process is required in order to define an end user's service profile. This implies the management of new usernames and passwords, and a large amount of, usually private, information.

The Cambridge Dictionary Cambridge University Press [22] defines **identity** as "who a person is, or the qualities of a person or group which make them different from others". The reality is that anyone who wants to make use of an Internet service usually needs to share some private information with the service provider, be it a real need (in the case of address and billing information) or a requirement of the business model (for example, in the case of being asked for gender and age). These users need tools to make the management of their multiple identities in the network easier. An **identity management system** ought to provide end users with these mechanisms, from the management of simple service accounts, to offering value-added functions such as ensuring privacy, advanced access control or Single Sign On (SSO).

When organizations wish to share their resources among their registered end users, the concept of **identity federations** arises. Identity federations define how, by making use of trust relationships, end users of any of the involved organizations are able to request access to the services offered by the rest. Some identity management systems like Higgins [23] and Shibboleth [24] provide end users the ability to homogenize the use of **authentication credentials** (typically username and password) to deal with identity federations.

When an end user wants to access a Web service, the service provider needs to confirm that she is a valid end user (and usually identified as such) on its system. To carry out the **authentication process**, the service provider usually makes use of an identity management system, which is responsible for retrieving the end user's required

information and verifying the authenticity of her identity. Examples of those authentication mechanisms are HTTP Basic and Digest [25], Forms Based [26], digital certificates (TLS) [27], etc. If the authentication is successful, it generates an authentication proof, also called **authentication token**, to the service provider. With this token, the service provider will be sure that the end user has been authenticated in the system and can access the protected resource. Another relevant concept is the **Single Sign-On (SSO)** [28], which allows end users to access different service providers within an identity federation, with authentication only required the first time they access during a session lifetime. This mechanism provides significant advantages, such as saving re-authentication time and improving the user experience.

The **access control system** of a service provider may require, beside end user authentication, an **authorization process.** Authorization makes use of additional information in order to decide whether the end user meets the requirements imposed by the service or not. By definition, **authorization** is the process that determines what resources of a service provider the end user has permission to access. Furthermore, **access control** [29] is the process of gathering information and taking decisions about service delivery. For this, the service provider must contact the identity management system and request end user attributes. In this case, the amount of information gathered depends on the data the end user has added to her identity profile, and the available attributes disclosure policies.

This section introduces the existence of different types of entities, as shown in Figure 2.1. A **Service Provider** is an organization willing to offer Internet services, including Web, email, multimedia data, e-commerce or network access; providers dealing with end user's identity are known **Identity Providers**, which can be classified depending on the specific role they perform. For example, an **Authentication Provider** is responsible for demonstrating that an end user really is who she claims to be, offering an authentication token as proof of the successful authentication. Besides, the **Attribute Provider** is responsible for dealing with additional end user information (attributes).

A usual configuration for these entities is one Identity Provider that provides authentication and attribute information to several Service Provider about a large of Users in each organization. In the case of identity federated systems, there are various possibilities **topologies** depending on how the architectural entities (Service Provides and Identity Providers) are configured. Some of the most usual topologies are: centralized, distributed and hierarchical. In **centralized topologies**, there is a central entity that concentrates interactions with other system participants. In **distributed topologies**, the load of work and responsibility is divided per interconnected groups or areas of work; sometimes this

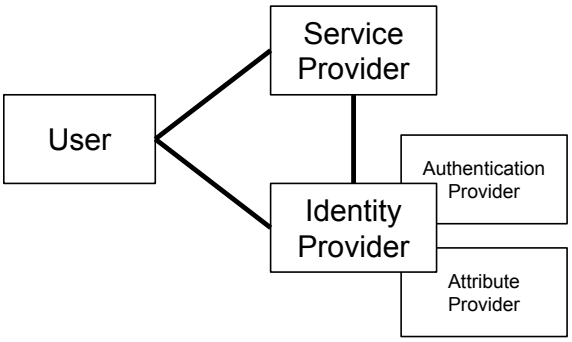


Figure 2.1: entities interaction diagram.

topology is similar to a mesh, due to the high level of interconnections between entities. Finally, in **hierarchical topologies** the entities distribution are asymmetric with more elements in the ends, and fewer, or just one, when you move to the root. In this topology, the intermediate entities are able to attend the request without overloading the root node. Figure 2.2 shows different examples of these configurations:

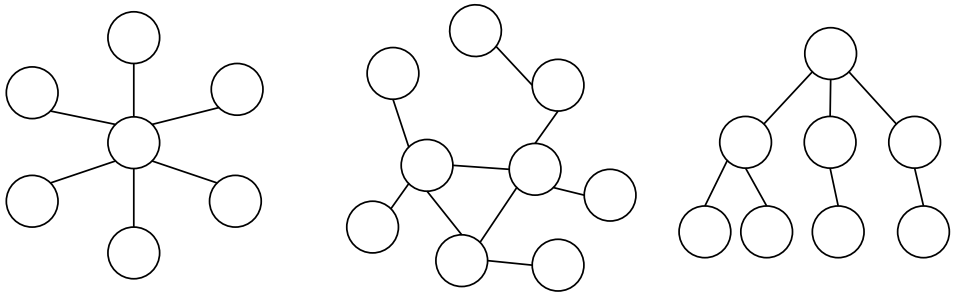


Figure 2.2: examples of possible topologies in identity federation systems. From left to right: centralized, distributed and hierarchical.

2.2 Base technologies

This section provides a survey of the most important protocols and standards in the area of authentication and authorization security mechanisms for Web services. In order to facilitate the establishment of similarities and differences between these mechanisms, we propose a running example: Alice, a traditional end user of Web services is registered with an identity provider (www.idp.com). She wants to buy the latest CD of her favorite music group at a digital music store *(www.music-store.com)*, and to receive it at home.

Note that in order to ensure the confidentiality of communications between those entities, Web technologies delegate to the security transport layer that protects the communication channel with protocols such as TLS 1.0 [30] or SSL3.0 [31].

2.2.1 Security Assertion Markup Language (SAML)

SAML is an open standard [32] developed and approved by the OASIS SSTC [33]. It uses the XML language [34] to exchange authentication and authorization data between security domains, usually between an identity provider and a service provider. SAML dates from January 2001, when the OASIS SSTC met for first time with the aim of defining an XML framework for exchanging authentication and authorization information. The current version, version 2.0, was released in March 2005. As mentioned earlier, the specification recommends, and in some cases requires, a variety of security mechanisms, such as the use of SSL 3.0 or TLS 1.0 for security at the transport level.

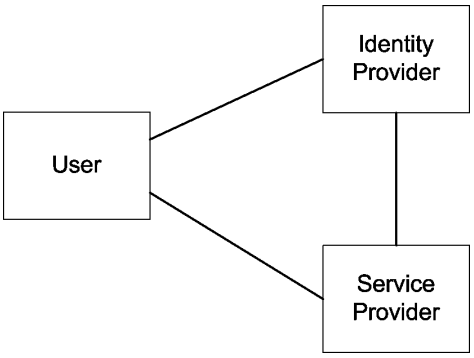


Figure 2.3: Relations between SAML entities

SAML defines three main entities, shown in Figure 2.3. The User is the one who wants to access a resource or Web service, and therefore needs to be authenticated and, optionally, authorized. The Service Provider (SP) is responsible for offering the service to the User, and relies on the Identity Provider (IdP) to perform her identification and authentication. Besides the authentication process, the Identity Provider could provide service providers with additional information (role, age, etc.) about the User in the form of attribute statements. Following the running example, Alice plays the User's role, the Identity Provider represents the www.idp.com entity and the Service Provider represents the music store service (www.music-store.com).

SAML defines the exchange of authentication, attributes and authorization decisions statements between the entities involved. Authentication statements, issued by Identity Providers, inform the Service Provider that the end user has, by means of a specific

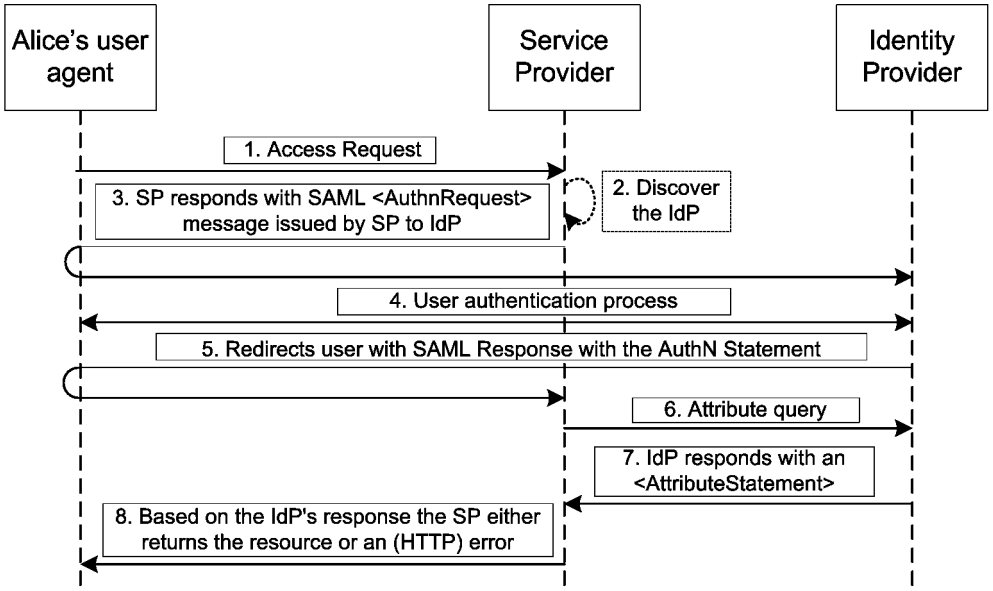


Figure 2.4: SAML Web Browser SSO Profile

authentication method, been successfully authenticated. Attribute statements provide name-value pairs with information associated with the subject (User) which can be used by the Service Provider to take the final access control decision. Authorization decision statements (deprecated in v2.0) assert that the subject has been authorized to perform a certain action on a specific resource based on some kind of evidence.

To request these statements, the standard defines several protocols [35] based on a request-response scheme, such as the *Assertion Query and Request Protocol*, the *Artifact Resolution Protocol* or the *Single Logout Protocol*, of which the most relevant is the *Authentication Request Protocol*.

In addition, SAML defines several bindings [36] that describe how to transport a SAML message in a standard format message using several communication protocols. Specifically, the *HTTP Redirect POST binding* defines the message exchange in base64-encoded format over HTTP redirections.

Finally, SAML defines several profiles [37], by defining specific sets of rules and syntax restrictions to solve specific business problems, such as the definition of constraints on the contents of SAML statements, protocols, and bindings; for example, the *Web Browser SSO Profile*, the *Assertion Query/Request Profile* and the *Artifact Resolution Profile*.

The authentication flow described in Figure 2.4 represents the *SAML Web Browser SSO Profile*. It starts when the User (Alice) wants to access (1) through her User Agent (for example, a Web browser) to the Service Provider (SP) (www.music-store.com). Alice is

```
<samlp:AuthnRequest
    xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    ID="identifier_1"
    Version="2.0"
    IssueInstant="2012-04-05T110:05:31Z"
    AssertionConsumerServiceIndex="1">
    <saml:Issuer>https://www.music-store.com/SAML2</saml:Issuer>
    <samlp:NameIDPolicy
      AllowCreate="true"
      Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"/>
</samlp:AuthnRequest>
```

Table 2.1: SAML Authentication Request

not authenticated, so SP initiates a discovery process (2) in order to ascertain know which her Identity Provider (IdP) is. This process is usually based on a MetaData service [38] known by the entities involved. Once SP knows about the Identity Provider, it issues (3) an explicit SAML authentication request, *AuthnRequest* (Table 2.1) to the IdP, through a Web redirection, which includes some relevant information such as the request's issuer (*saml:Issuer*). The IdP (4) authenticates Alice (the method is outside the scope of the standard), and responds (5) to the SP using a XHTML form inside a Web redirection that contains the SAML authentication statement (*AuthnStatement*).

Table 2.2 shows an example of an Authentication Response that includes the *Subject* inside the authentication statement. It is worth noting that the *NameID* field, which represents the end user identifier, is for privacy reasons a transient pseudonym and not the real identity. It is digitally signed by IdP and validated by SP, so a trust relationship must exist between both entities. It also contains the SP destination URI (www.music-store.com), the assertion's issuer (in this case: Alice's IdP) and the successful status of the authentication process, among other things.

The SP requests (6) additional User attributes (i.e. Alice's preferred language, age, entitlement, or postal address) from the IdP in order to take an access authorization decision. The IdP checks who is making the request and who is the subject (Alice's pseudonym). Then, it prepares the requested attributes and responds to the SP with a SAML *AttributeStatement* sentence (7). Using this information, the SP responds to Alice, allowing or denying her access to the desired service.

In SAML, the concept of SSO is managed with the help of HTTP [39]. From the point of view of the IdP, if further authentication requests from the same user are received during the validity period of the previously generated authentication statement, she is not asked for a new authentication process. HTTP-cookies [40] are used to manage an HTTP session between the IdP and the User.

```
<samlp:Response
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  ID="identifier_2" InResponseTo="identifier_1" Version="2.0"
  IssueInstant="2012-04-05T10:05:381Z"
  Destination="https://www.music-store.com/SAML2/SSO/POST">
  <saml:Issuer>https://wwww.idp.com/SAML2</saml:Issuer>
  <samlp:Status>
    <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
  </samlp:Status>
  <saml:Assertion
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    ID="identifier_3" Version="2.0" IssueInstant="2012-04-05T10:05:381Z">
    <saml:Issuer>https://www.idp.com/SAML2</saml:Issuer>
    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">...</ds:Signature>
    <saml:Subject>
      <saml:NameID format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient">
        3f7b3dcf-1674-4ecd-92c8-1544f346baf8
      </saml:NameID>
      <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
        <saml:SubjectConfirmationData
          InResponseTo="identifier_1"
          Recipient="https://www.music-store.com/SAML2/SSO/POST"
          NotOnOrAfter="2012-04-05T110:15:381Z"/>
      </saml:SubjectConfirmation>
    </saml:Subject>
    <saml:Conditions
      NotBefore="2012-04-05T10:05:381Z" NotOnOrAfter="2012-04-05T10:15:381Z">
      <saml:AudienceRestriction>
        <saml:Audience>https://www.music-store.com/SAML2</saml:Audience>
      </saml:AudienceRestriction>
    </saml:Conditions>
    <saml:AuthnStatement AuthnInstant="2012-04-05T10:03:381Z"
     SessionIndex="identifier_3">
     <saml:AuthnContext>
       <saml:AuthnContextClassRef>
         urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
       </saml:AuthnContextClassRef>
      </saml:AuthnContext>
    </saml:AuthnStatement>
  </saml:Assertion>
</samlp:Response>
```

Table 2.2: SAML Authentication Response

SAML standard is widely used in federated environments of universities and research networks like Cisco Networking Academy [41] or in large Web applications providers like Google [42]. In addition, there are multiple implementations of this mature standard, such as the Shibboleth package [24].

2.2.2 WS-Security and WS-Trust

WS-* is a family of Web services specifications developed by the OASIS WSS TC [43]. The aim of this specification is to provide a flexible set of mechanisms that can be used to construct a range of security protocols. The WS-Security specification [44] defines a flexible and feature-rich extension to SOAP [45], specifying how integrity and confidentiality in Web messages can be enforced and enable the communication between different kind of security tokens. The standard mainly focuses on the use of XML Signature [46] and XML Encryption [47] to provide end-to-end security, but by itself, it does not provide a complete security solution for Web services. So it is complemented with specifications like WS-Trust [48], WS-SecureConversation [49] and WS-SecurityPolicy [50]. This section focuses on WS-Security and WS-Trust.

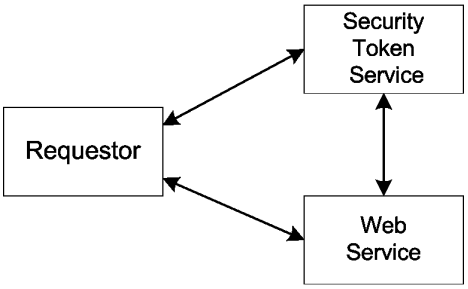


Figure 2.5: WS-Trust Security Token Service scheme

WS-Trust specification defines a Web service security model (Figure 2.5) that allows security tokens to be exchanged in order to enable the issuance and dissemination of credentials within different trusted organizations. The main entity of this model is the Security Token Service (STS), which is responsible for issuing, renewing and validating security tokens based on evidence in which the STS trusts. The use of different kinds of security tokens such as SAML sentences [51], Kerberos tickets [52] or X.509 certificates [53] are defined in the WS-Security specification.

Figure 2.5 shows the main entities in this specification. In order to establish a trusted communication between a Web Service and STS, the Web Service requires some proof, such as a signature or the proof of possession of a security token (or a set of them). The token

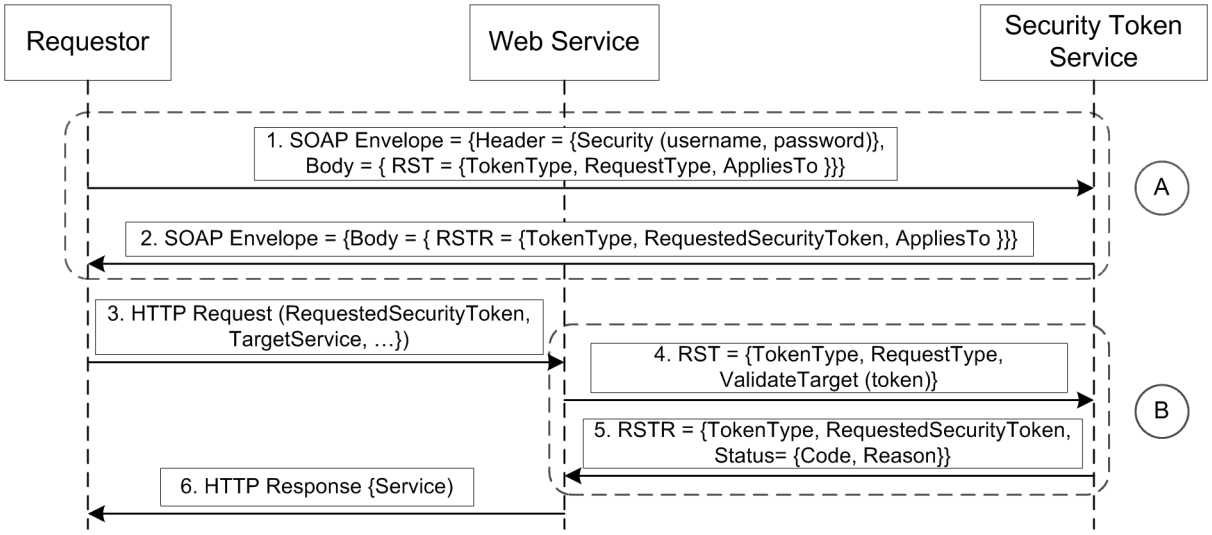


Figure 2.6: WS-Trust STS detailed interaction flow

generation process is usually carried out by the STS. The Requestor entity represents a service or client that wants to access a Web Service. This Web Service may require the incoming request message to prove the possession of a set of claims or statements. If the Requestor does not have the necessary token to prove the claims required by the Web Service, it contacts the appropriate authority (STS) and requests the specific token. The message formats to request and reply for the issuance, the validation and the renewing of security tokens are defined by the WS-Trust specification.

In this case, the comparison with the running example is not so immediate. Alice (Requestor) wants to access www.music-store.com (Web Service). This service requires a specific access or authentication token that Alice does not have. Alice accesses the www.idp.com (STS), in order to obtain the appropriate token. Alice can then provide www.music-store.com with the proof needed to gain the service access.

The STS issuance token process, Figure 2.6-A, uses the *RequestSequrityToken* (*RST*) request (1) that includes details (Table 2.3) about the type of the required token (*mySpecialToken*), the type of request *(Issue*) and other information that might be needed. Besides, the STS may require the authentication of the end users, so the request can include end user credentials (i.e. username and password) or another authentication information at *SOAP Security Header*, depending on the specific STS policies (outside the scope of the specification). The *RequestSecurityTokenResponse (RSTR)* message (2) provides the security token and other related information such as the type (*TokenType*) and the request context, as shown in Table 2.4.

```
<S11:Envelope xmlns:S11="..." xmlns:wsu="..." xmlns:wsse="..."
        xmlns:xenc="..." xmlns:wst="...">
    <S11:Header>
        ...
        <wsse:Security>
            <xenc:ReferenceList>...</xenc:ReferenceList>
            <xenc:EncryptedData Id="encUsername">...</xenc:EncryptedData>
            <ds:Signature xmlns:ds="...">
                ...
            <ds:KeyInfo>
                <wsse:SecurityTokenReference>
                    <wsse:Reference URI="#myToken"/>
                </wsse:SecurityTokenReference>
             </ds:KeyInfo>
            </ds:Signature>
        </wsse:Security>
        ...
    </S11:Header>
    <S11:Body wsu:Id="req">
        <wst:RequestSecurityToken>
            <wst:TokenType>
                http://www.idp.com/mySpecialToken
            </wst:TokenType>
            <wst:RequestType>
                http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue
            </wst:RequestType>
        </wst:RequestSecurityToken>
    </S11:Body>
</S11:Envelope>
```

Table 2.3: Request Security Token for a new token

```
<wst:RequestSecurityTokenResponse xmlns:wst="...">
  <wst:TokenType>http://example.org/mySpecialToken</wst:TokenType>
  <wst:RequestedSecurityToken>...token value...</wst:RequestedSecurityToken>
  <wst:Lifetime>...</wst:Lifetime>
  ...
</wst:RequestSecurityTokenResponse>
```

Table 2.4: Request Security Token Response

```
1  <wst:RequestSecurityToken xmlns:wst="...">
2      <wst:TokenType>...</wst:TokenType>
3      <wst:RequestType>
4          http://docs.oasis-open.org/ws-sx/ws-trust/200512/Validate
5      </wst:RequestType>
6      <wst:ValidateTarget>... </wst:ValidateTarget>
7      ...
8  </wst:RequestSecurityToken>
```

Table 2.5: Request Security Token for validation

```
1  <wst:RequestSecurityTokenResponse xmlns:wst="..." >
2      <wst:TokenType>...</wst:TokenType>
3      <wst:RequestedSecurityToken>...</wst:RequestedSecurityToken>
4      ...
5      <wst:Status>
6          <wst:Code>...</wst:Code>
7          <wst:Reason>...</wst:Reason>
8      </wst:Status>
9  </wst:RequestSecurityTokenResponse>
```

Table 2.6: Request Security Token Response for validation

The second part of Figure 2.6-B shows how the token is used to gain access to the Web Service (WS). The Requestor sends a *HTTP Request* (3) to the service, including the new token. Now, the WS can send it to the STS in order to check whether it is valid or not (4). The validation binding has a similar pattern to the previous messages. In this case (Table 2.5), the RST request includes the specific *RequestType* and the *ValidateTarget* element that the reference to the token typically contains. After being evaluated by the STS, it replies (5) to the WS with the RSTR indicating the token status (Table 2.6), including a specific code (*valid* or *invalid*) and, optionally, the reason as a human-readable text.

The proposed WS-Trust architecture does not consider the request for additional data (attributes). However, it is open to the use of access control policies in order to manage the different aspects to determine the requirements to be satisfied by the Requestor. Moreover, to make use of SSO mechanisms in federations, it is necessary to incorporate the WS-Federation specification [54], which extends the functionality of the STS and outlines how to carry out the interactions when more than one STS is involved.

WS-Trust and WS-Security are implemented within Web services libraries, provided by vendors or by open source collaborative efforts. Web services frameworks that implement the WS-* family protocols include: GlassFish Metro Project [55], Apache's Rampart (part of axis2) [56], Microsoft's WCF [57] and WIF [58].

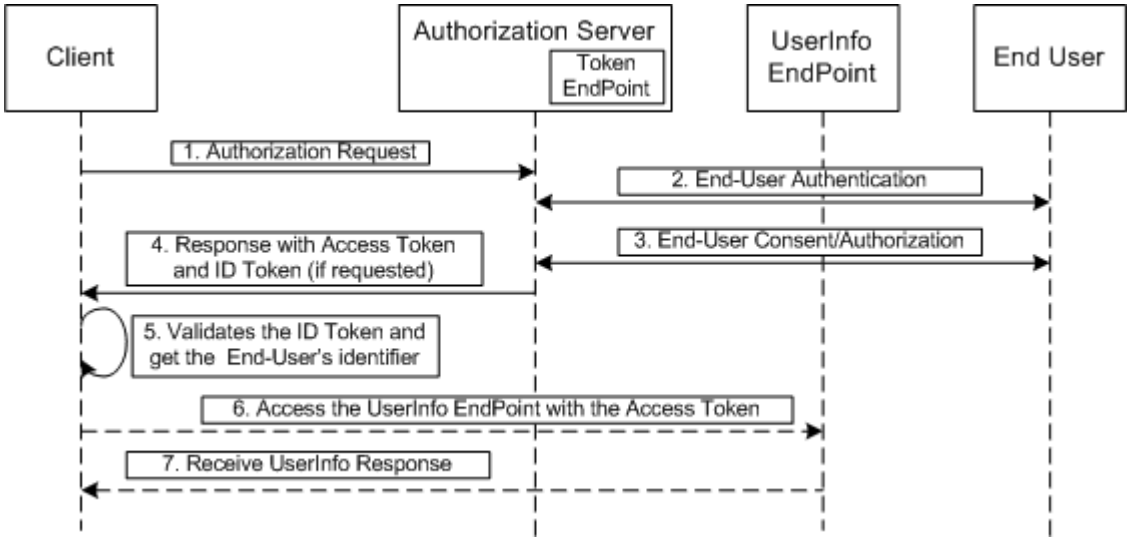


Figure 2.7: OpenID interaction flow schema.

2.2.3 OpenID

OpenID [59] is an open and free standard for authentication that provides end users with mechanisms to authenticate at a Web service in a decentralized environment making use of an URL [60] or XRI [61], like end user identifiers, which can be verified by any target service. It is also a personal data management system, such that when an end user authenticates in a new service for registration, she is requested to indicate the data she wishes to share with the new site.

The main advantage of a decentralized authentication system like OpenID is that the end user uses only one username/password credential to request access to hundreds of Web services without having to remember access passwords and names for each different site. In services supporting this standard, end users do not have to create a specific account to gain access to them. Instead, they only need an *OpenID identifier*, created at an OpenID Provider (OP), which allows the identity of the end user to be verified for any service compatible with the protocol.

OpenID specification defines three main entities . The User Agent is the entity acting in place of the end user (i.e. Web browser) when she wants to access a resource or Web service and needs to be authenticated. The Relaying Party (RP) offers a Web service to the end user and relies on the OpenID Provider to perform the task of the authentication process. Following the running example, the User Agent represents Alice's Web browser, the OpenID Provider is Alice's Identity Provider (www.idp.com) and the Relaying Party represents www.music-store.com.

```
1  https://www.idp.com/accounts/o8/id
2  ?openid.ns=http://specs.openid.net/auth/2.0
3  &openid.claimed_id=http://specs.openid.net/auth/2.0/identifier_select
4  &openid.identity=http://specs.openid.net/auth/2.0/identifier_select
5  &openid.return_to=http://www.music-store.com/checkauth
6  &openid.realm=http://www.music-store.com/
7  &openid.assoc_handle=ABSmpf6DNMw
8  &openid.mode=checkid_setup
```

Table 2.7: OpenID authentication request example

Figure 2.7 describes the message flow for the OpenID authentication. First, Alice starts the authentication process by sending her identifier (e.g. http://www.idp.com/alice/) to the RP (www.music-store.com) through the Web browser (1). The RP normalizes Alice's identifier and discovers the OpenID Provider (www.idp.com) associated to the identifier. Using this information, the RP can, optionally, establish an association with the OP (2). The association process calculates a shared secret (usually based on a Diffie-Hellman exchange [62] between the RP and the OP, which is used by the OP to sign subsequent messages and by the RP to verify those messages. If they avoid the shared secret generation step, the RP will need to do direct requests after each authentication request/response in order to verify the signatures. After that, the RP redirects Alice with an OpenID authentication request to the OP (3). An example of this request is shown in Table 2.7. It includes several fields, some related to the end user (such as *opened.claimed_id* and *opened.identity*), some related to the RP, like the *opened.realm* or *opened_return_to*. The authentication process (4) carried out by the OP is outside the scope of the OpenID specification.

Once Alice has been authenticated, she is redirected again (5) from the OP to the RP with the result of the authentication process, using the URI indicated at *opened.return_to*. This response (Table 2.8) includes the end user authentication information such as the end user identifier (*openid.identity*), the OP Endpoint URL (*openid.op_endpoint*), the list of signed fields (*openid.signed*) and a digital signature (*opened.sig*). Then, the RP verifies (6) the received statement using either the shared key established during the previous association (2) or by sending a direct request to the OP. If the verification is correct, RP allows (7) Alice access to the store.

OpenID does not specify which authentication mechanism must be used, so, the security depends on the trust in the OP. If it does not offer enough trust and the authentication mechanism is unreliable, the authentication will not be appropriate for services requiring strong security. Nevertheless, one popular advantage of OpenID is to allow any end user to set up their own authentication service, which would offer more confidence because they

```
1  http://www.music-store.com/checkauth
2  ?openid.ns=http://specs.openid.net/auth/2.0
3  &openid.mode=id_res
4  &openid.op_endpoint=https://www.idp.com/accounts/o8/ud
5  &openid.response_nonce=2012-05-18T04:14:41Zt6shNlcz-MBdaw
6  &openid.return_to=http://www.music-store.com:8080/checkauth
7  &openid.assoc_handle=ABSmpf6DNMw
8  &openid.signed=op_endpoint,claimed_id,identity,return_to,response_nonce,assoc_handle
9  &openid.sig=sgfiWSVLBQcmkjvsKvbIShczH2NOisjzBLZOsfizkI=
10 &openid.identity=https://www.idp.com/accounts/o8/id/id=ACyQatix...AY983DpW4UQV_U
11 &openid.claimed_id=https://www.idp.com/accounts/o8/id/id=ACyQat...AY983DpW4UQV_U
```

Table 2.8: OpenID redirect example after a successful user log-in and confirmation

have the service and personal data control.

The OpenID standard has been widely criticized for security and privacy reasons. On the one hand, the unification of all the end user identities in one OpenID entity entails high risks, because all the end user related information is protected by a single authentication process. On the other hand, the OP is the central point of authentication through which OpenID services must pass, so the OP knows the end user's information about the sites and services visited. Besides, if the OpenID Provider or end user's credentials are compromised, the entire digital life of the end user will be disclosed.

OpenID allows federated SSO login based on the use of the OpenID Identifier. The protocol enables Relaying Parties to validate the identity of an OpenID user at the OpenID Provider, including the optional ability to request end user's attributes by means of an extension to the standard named OpenID Attribute Exchange 1.0 [63].

This authentication protocol has been offered by some of the most important Internet companies such as Google [64], Yahoo [65], MySpace [66] or Paypal [67]. Many of these companies have currently migrated to OpenID Connect (see Section 2.2.5) as Google [68] and Paypal [69].

2.2.4 OAuth (Open Authorization)

OAuth is an open standard [70, 71] for authentication and authorization that allows end users to share limited access to their protected resources with third-party applications, orchestrating the approval interaction between a resource owner (end user) and a resource provider (service provider). Its functionality is based on the exchange of authorization codes and security tokens to manage the access control. The standard [72] allows Clients (consumers such as Web services and applications) to access protected resources stored at Web services (Resource Servers) through an API, without the end users (Resource Owners) having to reveal their access credentials stored in the Resource Servers to the Clients.

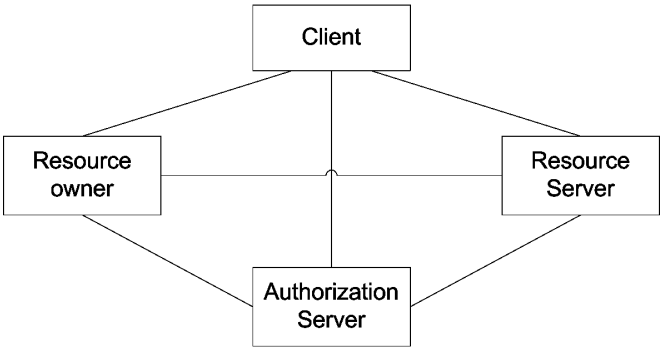


Figure 2.8: OAuth entities.

The OAuth2.0 version [73] was approved as RFC standard in 2012 and it has been widely used since its first drafts by the main social networks, like Twitter [74] and Facebook [75], with the support and adoption by companies like Google or Microsoft.

In November 2006, during the development of the OpenID implementation for Twitter, there emerged the need for a solution that allows end users to authorize certain applications access to their resources by delegating the access to them. The OAuth discussion group was created in April 2007 and soon had the support of Google. In October 2007, the final draft of the OAuth Core 1.0 was presented, and one year later, a minor revision called "OAuth 1.0 Revision A" [76] was published. Finally, in April 2010, RFC 5849 [72] was published with the title "The OAuth 1.0 Protocol". Only three months later, Twitter began to require all third party applications to use OAuth to access to Twitter's APIs. In late April 2010, OAuth IETF Working Group published the first draft of OAuth 2.0 protocol [73], whose standardization was completed in October 2012. This new version focuses on simplifying the development of clients, at the same time as providing specific authorization flows for Web services, desktop applications, mobile phones and home devices.

The OAuth architecture defines four types of roles that interact through the exchange of messages, as shown in Figure 2.8. First, the Resource Owner (RO) is the entity that grants access to the protected resource stored at the Resource Server (RS). The RS is responsible for answering protected resource requests. The Client represents the application that requests access to the protected resource on behalf of the Resource Owner through an authorization grant. Finally, the Authorization Server (AS) is responsible for issuing access tokens to the Client, so authenticating the Resource Owner and requesting her authorization.

OAuth 2.0 completely redefines the architecture of the previous versions to the point of not maintaining compatibility with them. Among the new features, the protocol

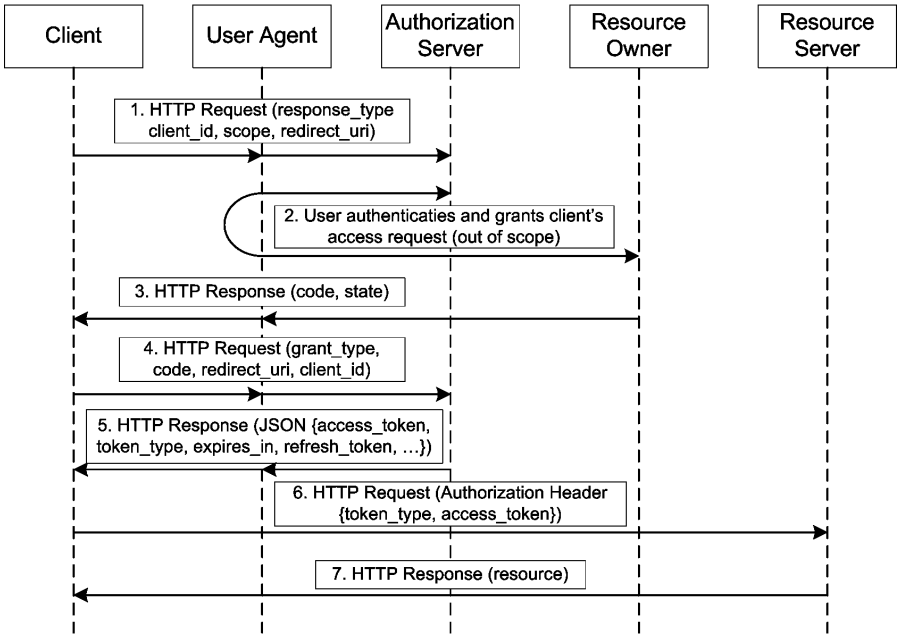


Figure 2.9: Auth Authorization Code flow.

defines three profiles by client type (Web application, user-agent and native application), four types of authorization grants used for request access tokens and provides a new mechanism called bearer token that relaxes the cryptography requirements and allows greater interoperability. In general, version 2.0 simplifies cryptography mechanisms and requirements for cryptographic signing of statements and tokens.

The OAuth architecture implies a change of perspective from the previous models. In this case, the Client represents www.music-store.com, that needs to access to Alice's postal address to be able to send the CD to her home after the purchase has been completed. Alice is represented by the role of the Resource Owner, and her identity provider (www.idp.com) would play the role of the Authorization Server. Finally, to cover all the roles, we assume that Alice's home postal address is stored in her identity provider's profile; so in this case, the www.idp.com would play the role of Resource Server too.

In more detail, Figure 2.9 shows the messages exchanged in the *authorization code grant* scenario. This scenario introduces the User Agent role, which consists of a public client application (e.g. Web browser) that is responsible for downloading and executing on the Alice's device (the Resource Owner) the Client code from a Web server.

The Authorization Code Grant flow (Figure 2.9) shows the scenario in which the Client (www.music-store.com) requests access to the protected resource (Alice's personal information) by making use of an authorization code grant to carry out the sale. First

(1), the Client has to request authorization (Table 2.9) from Alice (Resource Owner) by asking indirectly through the Authorization Server (www.idp.com), which receives the Client's identifier (*client_id*) and the redirection URI (*redirect_uri*) used to send back the response to Client. All messages are exchanged using HTTP protocol and protected by TLS. An example of a *HTTP Request* for the authorization request as is shown below:

```
1  GET /authorize?response_type=code&client_id= A2nJpRkay7&state=cba
2        &redirect_uri=http%3A%2F%2Fwww.music-store.com%0A HTTP/1.1
3  Host: authzserver.idp.com
```

Table 2.9: OAuth authorization request example

```
1  Location: https://www.music-store.com/ep?code=pKc3jkFoVnfCYkaybORrmdql&state=cba
```

Table 2.10: OAuth authorization response.

The Authorization Server (Alice's IdP) has to authenticate (2) Alice and asks whether she grants or denies the access request to her personal information. If Alice agrees, Client receives (3) an authorization code grant (*code)* as a proof of her authorization and any local state (*state*) provided by the Client's request, as shown at Table 2.10.

Using this credential, Client (4) requests an *access token* from the Authorization Server (Table 2.11). The Authorization Server authenticates the Client (*Authorization Header*), validates the *authorization code*, and makes sure that the redirection URI matches with the one received in the previous message. If the validation is correct, the Client receives (5) the access token (*access_token*) and, in some cases, a refresh token (*refresh_token*) as is depicted in the example (Table 2.12).

Now, the Client (www.music-store.com) requests access (6) to Alice's personal information (the protected resource), using the access token as proof of the access grant. As result of a correct process, the Client retrieves (7) Alice's attributes (e.g. her postal address) and completes the sale by sending the music CD to Alice's home. The access tokens can have different formats and structures based on the resource server security

```
1  POST /token HTTP/1.1
2  Host: authzserver.test.com
3  Authorization: Basic JQSzdZs0z2WJpmMpntEFn3S0zmCa
4  Content-Type: application/x-www-form-urlencoded;charset=UTF-8
5    grant_type=authorization_code&code=pKc3jkFoVnfCYkaybORrmdql
6    &redirect_uri=http%3A%2F%2Fwww.music-store.com%0A
```

Table 2.11: OAuth Access Token request example.

```
1  HTTP/1.1 200 OK
2  Content-Type: application/json;charset=UTF-8
3  Cache-Control: no-store
4  Pragma: no-cache
5  {
6    "access_token":"oic2ZFsaXG3nMzYWpEjrFA",
7    "token_type":"test_type",
8    "expires_in":3600,
9    "refresh_token":"IkueF2Tt0Qj9J1vGzhxPew"
10 }
```

Table 2.12: OAuth Access Token response example.

requirements, and can contain diverse end user attributes, but these details are beyond the scope of the OAuth specification and are not considered in the scenario.

In other scenarios, it is not always necessary to perform all the previously described steps. For example, in the Implicit Grant flow, the Client directly obtains the access token as the result of the authorization request, instead of the authorization code, so reducing the number of interactions required. In this case, the Client is not authenticated and the security checks fall on the redirection address used by the Client, which may be previously agreed on between the Client and the Authorization Server. Besides, OAuth defines an extension grant type based on bearer token. The possession of a token of this type allows Clients to request an access token (and access to the associated resource) without needing to demonstrate possession of a cryptographic key. All these features and the specific use of the bearer tokens are described in the RFC 6750 [77].

Although OAuth was designed as an authorization protocol, its use as an authentication mechanism is widely extended. OAuth allows Clients to perform a Resource Owner pseudo-authentication based on the authentication made by the AS before the authorization request. If the Client obtains the authorization grant, it means that the end user has been successfully authenticated. This mechanism offers an alternative to the Single Sign On by delegating the authentication process to the AS.

The OAuth protocol has been well received among the top Internet companies and, in particular, by the most popular social networks. Its architecture design and its approach toward Web services make it a lightweight authorization protocol and it facilitates the development of client applications. Besides, it seems to be the perfect complement for other protocols widely used in these scenarios, such as OpenID. For these reasons, companies like Google [78], Microsoft [79], Facebook or Twitter already use it in their services.

2.2.5 OpenID Connect

This new protocol [80] makes use of the OpenID concepts to add an identity layer on top of the OAuth 2.0 protocol. The framework, that follows the OAuth concept, allows clients of all types (including browser-based, mobile, and JavaScript clients) to verify the identity of an end user based on the authentication made by the (OAuth) Authorization Server (AS), before requesting the end user's authorization, and it provides mechanisms to obtain basic information about her profile.

OpenID Connect offers a set of specifications that provides a lightweight framework for identity interactions through the REST [81] APIs. The specification set is extensible, allowing the optional encryption of identity data, the discovery of OpenID providers, and advanced session management.

The objective of OpenID Connect is to integrate all the OpenID's features with OAuth protocol. In this sense, OpenID Connect incorporates into its specification the *authorization codes* and the *access tokens* defined in OAuth, adding a new third type of token, named *ID token*, which contains authentication information and can be used later to request additional end user information. The specification incorporates the new endpoint called *UserInfo EndPoint*, which corresponds with a protected resource that returns claims (attributes) about end user information. The Client can request access by submitting the respective access token. In the running example, this new role is performed by Alice's identity provider (www.idp.com), since it is responsible for protecting and providing Alice's private information. The answers provided by the *UserInfo EndPoint* can include three different types of claims: normal, aggregated (from other sources but provided by the OpenID Provider) and distributed (references are provided so that the Client can retrieve them).

Figure 2.10 depicts the OpenID Connect implicit authorization flow that consists of a scenario in which the Client (www.music-store.com) requests access to Alice's *UserInfo EndPoint* (www.idp.com) that contains her personal information (e.g. her postal address). First, the Client prepares an Authorization Request (1) and sends it to the AS. The request (Table 2.13) contains parameters such as *reponse_type* (fixed to "*token id_token*" for requesting both *access token* and *ID token*) and *scope* (set to *openid*, that informs the AS that it is a OpenID Connect request, and *profile* that requests the default profile claims of the *UserInfor EndPoint*). Other relevant parameters included in the request are the Client's ID (*client_id*) and the redirection URI (*redirect_uri*) used by the AS.

Once the request has been received, the Authorization Server has to authenticate Alice (2) and to obtain her consent/authorization (3) for the Client access. Both authentication

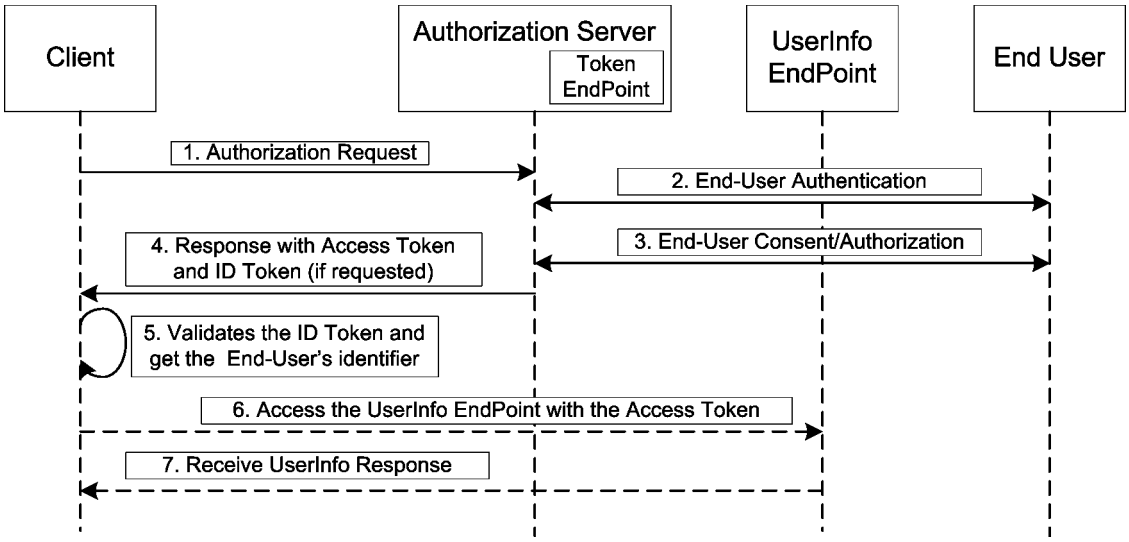


Figure 2.10: OpenID Connect Implicit code flow.

```
1  HTTP/1.1 302 Found
2  Location: https://server.example.com/authorize?response_type=token%20id_token
3           &client_id=s6BhdRkqt3
4           &redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
5           &scope=openid%20profile
6           &state=af0ifjsldkj
```

Table 2.13: OpenID Connect authorization request example to the Authorization Server

```
1  HTTP/1.1 200 OK
2  Content-Type: application/json;charset=UTF-8
3  Cache-Control: no-store
4  Pragma: no-cache
5  {
6    "access_token":"SlAV32hkKG",
7    "token_type":"bearer",
8    "expires_in":3600,
9    "refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA",
10   "id_token":"eyJ0NiJ9.eyJ1c...I6IjlifX0.DeWt4Qu...ZXso"
11 }
```

Table 2.14: OpenID Connect response example.

```
GET /userinfo?schema=openid HTTP/1.1
Host: www.idp.com
Authorization: Bearer "eyJ0NiJ9.eyJ1c ... I6IjlifX0.DeWt4Qu ... ZXso"
```

Table 2.15: OpenID Connect UserInfo request.

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
{
 "user_id": "248289761001",
 "name": "Alice Smith",
 "given_name": "Alice",
 "family_name": "Smith",
 "address": "211B Baker Street, London",
 "email": "alicesmith@idp.com",
 "picture": "http://www.idp.com/alicesmith/me.jpg"
}
```

Table 2.16: OpenID Connect UserInfo EndPoint's response.

and authorization mechanisms are outside the scope of the OpenID Connect standard. If Alice is successfully authenticated and consents to the Client grant, the AS sends (4) the End User back to the Client with the *access token* and the *ID token* as it was requested in the *response_type* field of authorization request from Table 2.13; a response example is shown in Table 2.14.

The Client verifies (5) the ID token itself by checking the validity of several fields, such as the Client's ID (*client_id*) and the audience restriction (*aud*), as detailed at OpenID Connect Message specification [82]. After successful validation, the Client (www.music-store.com) may send (6) a *UserInfo requests* (Table 2.15) with the ID token to the *UserInfo Endpoint* to obtain (7) further information (Table 2.16) about Alice, such as her postal address, which is needed to complete the sale. For federated login scenarios, in which the Authorization Services are federated, it is sufficient for Clients to use the ID token obtained from authorization requests, to obtain the SSO features.

As can be seen, OpenID Connect offers a complete and fully functional solution that promises a very interesting future. Currently its use is widespread among Internet giants like Google, Twitter, Facebook and Microsoft, who have been collaborating in its specification since its early drafts.

2.2.6 Future research direction in the area of base technologies

Once identity management systems have been consolidated to provide authentication and authorization frameworks for Web services, new challenges emerge in order to ensure a

better user experience, strong security and privacy.

The management of several digital identities and the use of identity federations can confuse end users about what kind of personal information is being revealed and to whom. The definition of tools to manage attribute reveal policies and the use of standard languages to define those policies (like XACML [83]) are topics that should be managed by Web services and identity providers.

Strong end user authentication also has to be taken into account. Although it could worsen the user experience, strong cryptography, like digital public key certificates, would improve end user security considerably.

In contrast, some users are requesting more privacy. Concepts like anonymity and pseudonymity are always present in the proposed solutions, but today there is no clear solution for that.

2.2.7 Summary

This section introduces the reader to some of the most relevant standards and protocols currently used by Internet organizations to ensure end users' identity for Web services. We have covered generic frameworks, like WS-Security and WS-Trust, providing extensible mechanisms and high degree of flexibility for deployment and specific and rigid proposals, like OpenID. In between, we can find solutions such as SAML and OAuth, providing different profiles and bindings and able to offer enough flexibility to be very attractive for Web services developers. In fact, the latter are the most common solutions adopted today by Internet Web services. We have to take into account that these are not independent proposals. For example, OpenID Connect has been born from the combination of OpenID and OAuth, seeking to take the advantages of each. WS-* was designed with SAML in mind as the security sentences definition language, and OAuth defines new profiles in combination with SAML.

Furthermore, for each of these solutions, we have presented the main use case, offering a brief description of the entities involved and the information flow. Additionally, we have provided a running example in order to offer a clear view of involved roles and entities.

Finally, because of the analysis performed in this section, we have noticed an increasing trend to provide, as far as possible, interoperability between the different existing technologies. Relevant services, like social networks, demand the ability to exchange authentication and authorization information to provide end users with a homogeneous view of their identity information across services.

2.3 Identity Federation review

Nowadays there are several projects working on different AAI solutions and identity federation technologies. In the area of science there are several options. EGI [84] is a highly distributed, multi-disciplinary resource infrastructure, integrating more than 300 resource centers (service providers) and almost 20,000 users. EUDAT [85] offers common data services, supporting multiple research communities as well as individuals resilient network of 33 European organizations based on is B2ACCESS [3] and Unity [86], ELIXIR [87] builds its own infrastructure for biological information, while Umbrella [88] is the pan-European federated identity system for the 30,000 users of the European large photon/neutron facilities. All are good AAI examples of this but, as we will see below, their numbers of users and services have no comparison with STORK [89] and eduGAIN [90].

Due to the high number of federations, this section focuses on identity federations based on SAML 2.0, since this is the authentication and authorization protocol par excellence. As shown in Section 2.2.1, SAML defines both the representation of the information and the protocols that are used to exchange this information. SAML assertions are the security statement format to transport information regarding a specific principal. To exchange them, SAML defines the use of several query/response protocols, like Authentication Request Protocol to request authentication information regarding a principal, and the Assertion Query and Request Protocol to obtain an assertion or Name Identifier Management Protocol to change the identifier given to a principal.

The following identity federation review analyzes, by each federation, different aspects such as the general characteristics, its architecture and entities, the basic flow of interaction, relevant protocols and software as well as the target audience and the use scope.

2.3.1 Moonshot

Moonshot promotes the development of a single unifying technology to extend the benefits of federated identity to a broad range of non-Web services [91], all in a manner that gives these users Single Sign On (SSO). It was developed by Jisc, the UK's National Research and Education Network (NREN, also known as JANET, in collaboration with partners from around the world.

Moonshot extends the federated identities advantages to a wide range of non-web services, including high performance computing, cloud and grid infrastructures and other non-web common services, such as instant messaging, file storage or mail [1].

The Moonshot technology is the implementation of the IETF's Application Bridging

for Federated Access Beyond web (ABFAB) [92] standards and makes use of EAP/RADIUS [93, 94] for authentication, as used in eduroam [95] and SAML [32] authorisation, as used in eduGAIN [90]. The Moonshot specification allows a Moonshot IdP (Moonshot Identity Provider) to retrieve user attributes from a SAML IdP or an LDAP directory. The information retrieved is sent by the Moonshot IdP to the SP in a standard SAML assertion.

Entities and architecture

Moonshot defines four main components (see Figure 2.11) to represent all the elements involved in its architecture. The interaction topology for these elements depends on the AAA architecture behind, but in general it should be hierarchical like classic RADIUS networks. Below is the description of the entities involved:

- Client: the end user's software installed in her computer and used for interacting with the service. The software allows the session to be started with the service and completes the authentication process at the IdP.
- Relying Party (RP): this consists of the service itself and the specific Moonshot module called Relying Party Proxy (RPProxy. It is a RADIUS server which allows the interconnection with the IdPs through the Trust Infrastructure (TI). During a new session, the Service part contacts the local RP Proxy, which uses its TI to find and forward the authentication request to the home IdP.
- Identity Provider (IdP): is the entity in charge of providing identity information about organisation members. The end user and IdP interact through RADIUS EAP secure tunnelling [1], which allows the obfuscation of user credentials for any intermediate party. After successful authentication, the IdP generates a SAML response with the success status and user's attributes, and sends it to the RP through the TI.
- Trust Infrastructure (IT): this is managed by the NREN and makes the trust relationship between the RP and IdP possible. There are several possible configurations for the TI based on the communication path between both endpoints (RP and IdP). On one hand, there is the classical hierarchical RADIUS network (i.e. eduroam structure), and on the other, it is possible to use Moonshot Trust Router Network, which allows direct communication between the Relaying Party and the Identity Provider, without passing through other proxyRADIUS servers.

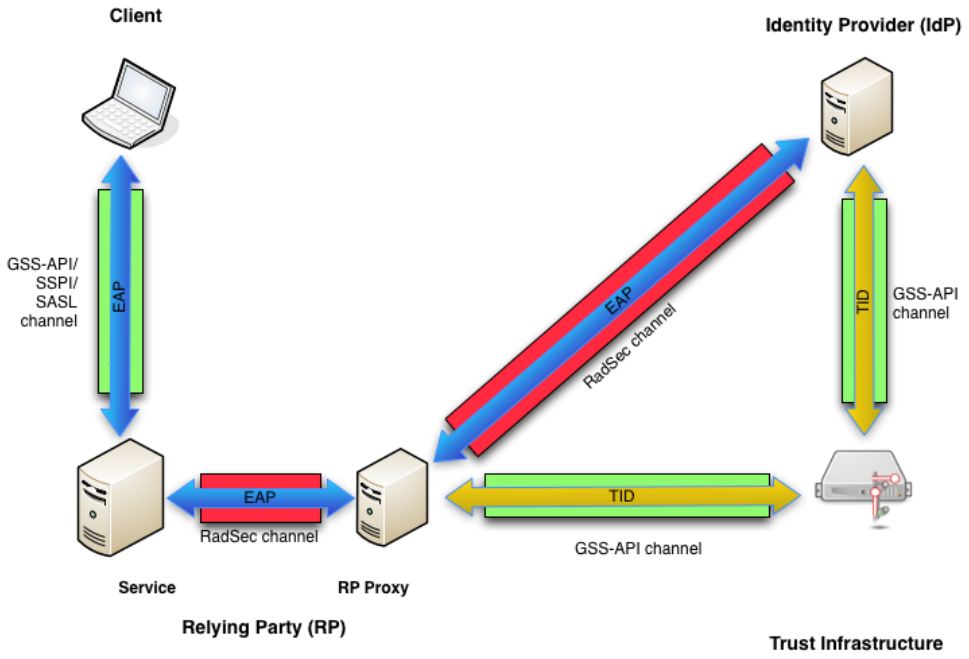


Figure 2.11: Moonshot architecture [1].

Interaction flow

The steps of the protocol flow to access an OpenSSH service are as follows [96]:

1. The application client attempts to connect to the OpenSSH application server using the application's standard protocol.

2. The client and the application server agree on the use of GSS-API authentication, so the client's device calls the GSS-EAP.

3. The GSS-EAP module requests user credentials for the service.

4. GSS-EAP mechanisms are agreed on to be used as authentication mechanisms from the GSS-APi.

5. The client creates an EAP request using its GSS-EAP module, with the NAI anonymized version. A GGS channel is established and used to send the EAP message to the server.

6. At the server, a RadSec/RADIUS connection is opened by the GSS-EAP module to its configured RP Proxy in order to send a RADIUS Access-Request message with

the client's EAP message. At the application server, GSS-EAP could also create a SAML authentication request to be added to the RADIUS Access-Request.

7. The RP Proxy receives this request. Depending on the realm, it finds the IP of the associated IdP, with which it can contact the IdP and interchange the keying material necessary to open a secure connection. The RP Proxy opens a RadSec/RADIUS connection, through which it forwards the RADIUS Access-Request with the encapsulated EAP message.
   After this step, a secure tunnel exists between the RPProxy and IdP and there is also a path between the client and its IdP.

8. Having the EAP channel between the client and the IdP, both choose the EAP method to use, in order to create a secure tunnel between them.

9. IdP and Client have to agree on the EAP method to verify the credentials. In addition, a channel bindings message is sent by the Client with the GSS name of the RP.

10. The IdP authenticates the client credentials and checks the GSS name. Then, EAP messages are exchanged so that the client and the IdP both end up with a copy of the EAP Master Session Key (EAP MSK) and EAP Extended MSK.

11. The IdP sends a RADIUS Access-Accept message with the encapsulated EAP success message to the RP Proxy with the EAP MSK, and possibly a SAML assertion.

12. The RP Proxy evaluates the user iaccording to its policies and the information retrieved from the IdP. In case of successful decision, the RP Proxy forwards the Access-Accept with an EAP positive response, EAP MSK, and possibly SAML assertion to the application server over the RadSec/RADIUS connection.

13. At the application server, the GSS-EAP module checks the EAP keys and, based on this information, it evaluates the user's access to the service. If it is positive, the server provides access to the client's application.

Protocols and technologies

Moonshot uses different technologies and protocols to implement the ABFAB architecture. The authentication process is done using EAP/RADIUS and the authorisation information is interchanged using SAML. The end user's Client needs to be compatible with them in

order to interact with the architecture. Therefore, Moonshot Project offers a GSS-EAP mechanism that allows the use of GSS-API authentication mechanisms in the EAP and RADIUS network. The support of GSS-EAP mechanism enables software to use Moonshot technology as an authentication and authorization option. As alternatives, there are SSPI and Security Support Providers (SSPs), which require the EAP-SSP module to allow the use of SSPI-enabled by applications. Finally, another popular and compatible security protocol is Kerberos, which acts as a mechanism for GSS-API and is supported by quite a few applications.

Audience and scope

Moonshot is general-purpose technology to build on and unify existing infrastructures (i.e. eduroam) and identity federations components (IdPs) with the aim of offering new services and possibilities based on them. Besides, it can be used in several scenarios regardless of the audience.

The unifying feature allows fewer usernames and passwords for the user. It offers SSO to all of the services, with more secure authentications provided by the user's home organization. In addition, it preserves user privacy by not releasing personally identifiable information - unless the user gives consent.

From the point of view of service administrators, Moonshot reduces the cost and effort of credential management, since the authentication and maintenance work is delegated to IdPs. At the same time, it allows flexible authorization, since the service can manage the access control on the basis of user attributes.

Summary

Moonshot is a Jisc project that promotes the development of a single unifying technology to extend the benefits of federated identity to a broad range services beyond the web. Its ABFAB technology makes use of common security mechanisms and protocols such as SAML, GSS-API, EAP, Radius or AAA. Moonshot is general-purpose technology that allows fewer usernames and passwords for common users. From the administrators' point of view, it reduces the cost and effort of credential management, since it allows the authentication and maintenance work to be delegated to IdPs.

2.3.2 eduGAIN

The eduGAIN project (EDUcation Global Authentication INfrastructure) [90] has its origins as a research activity in project GEANT2 (2004-2009), co-funded by the European Union and today it interconnects identity federations around the world in the field of global research and the education community, simplifying access to content, services and resources. eduGAIN infrastructure allows the trustworthy exchange of identity information related to authentication and authorization infrastructure (AAI). An example of its architecture is shown in Figure 2.12.

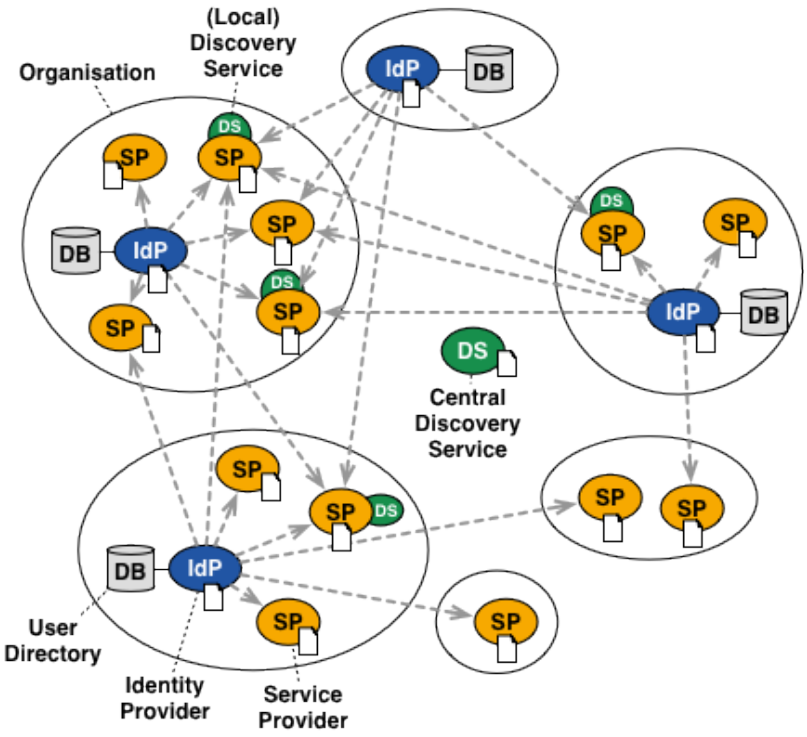


Figure 2.12: eduGAIN federation schema [2].

Thanks to eduGAIN, the IdPs can provide a wider range of services to their end users within an environment composed of multiple collaborative federations; The SPs can provide their services in different federations, amplifying their market possibilities; end users also see an enhanced service offering with security and trust. The Interoperable SAML 2.0 Profile ($SAML2_{Int}$) [97] is the only SAML 2.0 profile allowed in eduGAIN.

Entities and architecture

eduGAIN architecture offers different possibilities [2] in relation to the topology configuration. In general, it follows a distributed topology, but there are several options from fully distributed to centralized: full mesh, hub & spoke with distributed login and hub & spoke with centralized login.

Below is the list of the main eduGAIN entities [98] involved in a standard interaction flow:

- Service Provider (SP): this component controls the user access to services and resources. It evaluates the assertion generated by IdPs and uses the information retrieved to authorize access to protected services.
- Identity Provider (IdP): this entity is in charge of authenticating users and providing authentication and attributes statements about users to the requesting providers
- Discovery Service (DS): offers the user the possibility to indicate her Home Organization (HO) from a list so as to redirect her and thus delegate her authentication to its HO's IdP.
- Metadata Distribution Service (MDS): metadata files collect technical data and descriptive information about the IdPs and SPs. The MDS aggregates and validates upstream metadata delivered by the participating federations. Furthermore, it signs and republishes the gathered metadata for consumption by the member federations, playing the role of central trust point. Individual SPs and IdPs should not consume the eduGAIN metadata directly.

Interaction flow

The typical eduGAIN login flow is described below:

1. The user tries to access a web SP.
2. The SP requires user authentication so it redirects her to the DS in order to select her home IdP. Often the DS can be deployed with the SP or offered by the SP itself.
3. With this information, the SP redirects the user to her IdP with an authentication request.

4. At the IdP, the user is authenticated by introducing her credentials. Based on these, the IdP generates a SAML assertion with the result of the process and user information and redirects her again to the SP with a SAML response.

5. Based on the information received, the SP verifies the user identity information and authorizes access to the service.

Protocols and technologies

eduGAIN federations are based on $\text{SAML2}_{\text{INT}}$. This specification defines an interoperable subset of SAML 2.0 protocol [35] that guarantees the interoperability. The SAML 2.0 WebSSO INTeroperability Deployment Profile defines a minimum set required to be followed by entities participating in eduGAIN with regard to which bindings should be used, which parts of the SAML messages should be signed or encrypted and how, rules, etc.

As regards specific software, most of the eduGAIN deployment is based on Shibboleth [24]. The Shibboleth Internet2 middleware Project propose an architecture for identity management and federated identity-based AAI (Authentication and Authorization Infrastructure). It also offers an open-source implementation of this architecture based on SAML2.0. Hence, its software fits perfectly with the eduGAIN requirements so its SP and IdP software are widely used in eduGAIN environments.

Audience and use scope

eduGAIN project was designed to interconnect identity federations around the world in the field of global research and the education community. Its main promoters are the NRENs, which are part of GÉANT, but currently it is extended to Universities and research centers around the World and joins up 40 federations.

eduGAIN allows students, researchers and educators to access online services while minimizing the number of accounts users and service providers have to manage, which reduces the costs, the complexity and the security risks. In addition, it offers SPs access to a larger pool of international users, and makes access to resources of peer institutions or commercial or cloud services available to users using their one trusted identity.

Summary

The eduGAIN initiative is co-funded by the European Union and today interconnects identity federations around the world. It makes the trustworthy exchange of identity

information related to AAI possible It is based on SAML 2.0 Interoperable Profile, a subset of the SAML2.0 standard focused on easing the interoperability. eduGAIN was designed from its beginnings to interconnect global research and educational centers, and its main users are students, researchers and educators.

2.3.3 EUDAT

The EUDAT project [99] [85] is a pan-European data initiative that had a first stage between 2011-2014 and was extended in 2015 to run till 2018. The project started with a single consortium of 25 partners, including cooperating centers, thematic data centers and some of Europe's largest scientific data centers. It aims to support multiple research communities offering common data services, building a sustainable cross-disciplinary and cross-national data infrastructure that provide a set of shared services for accessing and preserving research data [100].

The AAI component of EUDAT is B2ACCESS [3]. It is an easy-to-use and secure AAI that can be integrated with different services, allowing users to log in with varied authentication methods (OAuth2, SAML, X.509 SLCs). When the user authenticates for the first time, she creates a new EUDAT ID in the B2ACCESS service using her home IdP (e.g. Facebook, Google, eduGAIN, X.509, etc.). The user's home IdP (external) is the recommended way to interact with EUDAT. In addition, B2ACCESS offers a range of tools and services to manage, store and share data (B2SHARE, B2SAFE, B2DROP, B2FIND, etc.).

Entities and architecture

EUDAT architecture bases its operation on B2ACCESS services that centralize all identity services. In contrast, it offers the possibility to replicate and distribute some services to adapt to the needs of the organizations that implement it. The entities involved and their functions are depicted in Figure 2.13 and described below:

- Service Providers: these entities, also called Downstream Services, offer services protected by the EUDAT architecture. When a user accesses the SP, it requests attribute assertions generated by the B2ACCESS service.
- Primary Identity Providers: these are external IdPs that provide identities and attributes using different technologies such as OpenID, SAML or X.509.

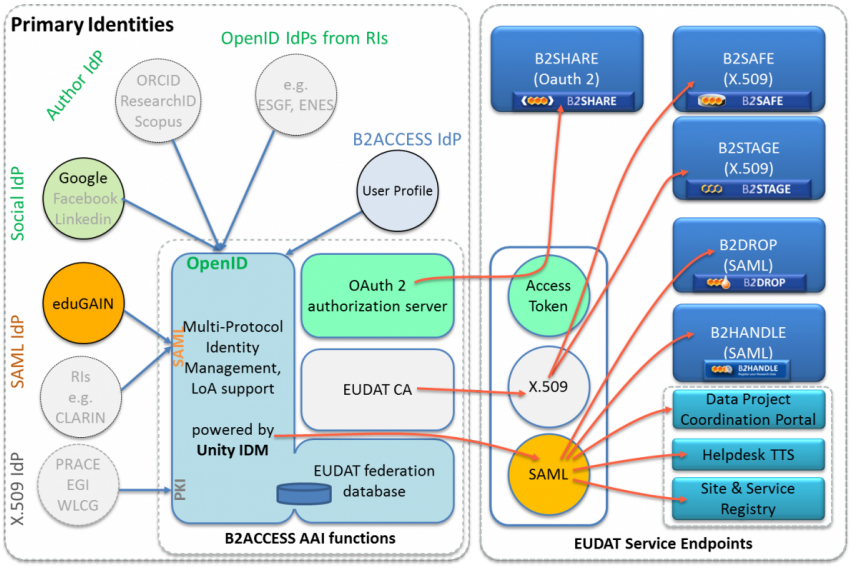


Figure 2.13: EUDAT internal architecture based en B2ACCESS [3]

- B2ACCESS AAI Functions: the Unity IdM component of the B2ACCESS service maps recovers information from the Primary Identity Providers onto an EUDAT identity. The role of B2ACCESS is to provide user authentication and authorization assertions to the SPs,.B2ACCESS can play the IdP role and authenticate its own users, using a specific EUDAT username and password.

- EUDAT Service Endpoints: These tools are responsible for managing, storing and sharing user data from the EUDAT identity.

Interaction flow

Due to the flexibility offered by the authentication methods, the flow will vary to suit the chosen mechanism. The login flow example described below is taken from [5] and corresponds to the SAML IdP scenario:

1. The end user wants access an EUDAT service that offers a protected resource (e.g. B2SHARE)

2. The user is not authenticated so she is redirected to B2ACCESS.

3. The B2ACCESS make it possible to select from a list the user's home organization IdP.

4. Based on the IdP selection, the user is redirected to her IdP login page, where she is authenticated with her credentials.

5. In the case of successful authentication, the user is redirected to the B2ACCESS with the authentication message from the IdP.

6. The end user is redirected to the SP by the B2ACCESS service with the EUDAT identity and attributes in the format used to integrate the service (i.e. OAuth2).

7. Then, the SP decides if the user is authorized or not on the basis of the information retrieved from B2ACCESS.

Protocols and technologies

EUDAT is able to work with several authentication protocols, such as SAML2.0, OAuth2, X.509 certificates. It makes interaction with other federation systems like eduGAIN, Google or Facebook possible. The B2ACCESS identity management software makes use of Unity Project [101], which offers "a complete solution for identity, federation and inter-federation management" [102]. It enables the authentication process using various protocols, with different configurations. Internally, Unity is composed of an orchestration platform with highly specialized extensions that provide support for the actual Unity features. The sources and binaries are protected by copyright, but it allows their redistribution and use under the same conditions and rights.

Audience and scope

EUDAT consortium comprises 36 European Partners, with more than half being Computing and Data Centers [103], managed by community data managers. Its target audience is citizen scientists and,staff and users from science organizations, research institutions and universities. Some deployment examples listed at [104] are: EPOS (Earth observation), ENES (climate), VPH (bio-medical sciences), ICOS, LTER and. CLARIN (linguistics),

Summary

The EUDAT project is a pan-European initiative formed by 36 partners. Its architecture is based on B2ACCESS AAI services and a complete set of backend services and tools to manage, store and share data (B2SHARE, B2SAFE, B2DROP, B2FIND, etc.). As it is based on Unity IdM software, EUDAT is able to work with several authentication

protocols such as SAML2.0, OAuth2 and X.509 certificates, which allows interaction with several identity federations. The project work is oriented toward Computing and Data Centers.

2.3.4 STORK

Secure idenTity acrOss boRders linKed 2.0 (STORK 2.0) [105] [89] is carried out by 19 EU/EEA Member States and 59 partners of different types, such as governmental institutions, banks or universities. The initiative works to offer citizens a single European electronic identification and authentication zone in collaboration with public and private service providers that allow new electronic relations between citizens and foreign e-services. This pilot experience has also contributed to the elaboration of the eIDAS normative, which will be reviewed in the next section.

STORK has implemented security guidelines and requirements [106] that all countries involved have to adhere to, and one of its most relevant characteristics is the high level of assurance (LoA) offers and requires in the authentication process, which is delegated to each Member State.

Its architecture is defined as a user centric system, so users must maintain the control of information shared with an entity, the specific origin and destination of their information. The user must be informed of the context, the sector (government, public, health, ...), the privacy or data usage policy, the liability disclaimer or any other aspect of each country's legislation where her informations will be used, and must give her consent The user's personal information revealed to an entity should be the minimum needed for the purposes of the service provided.

Entities and architecture

STORK architecture (Figure 2.14) has a hybrid topology based on the interaction between the PEPS and the rest of entities. For each country, STORK has one PEPS that acts as centralized point for all the users, Services Providers and Identity Providers of this country. In contrast, the topology between the PEPS is full distributed, with one-to-one trust relationships. This architecture allows that each country can maintain the control of what users and services providers are connected within its borders, while managing relations with the rest of the countries.

As detailed below, this is the list of main STORK entities involved in a standard interaction flow:

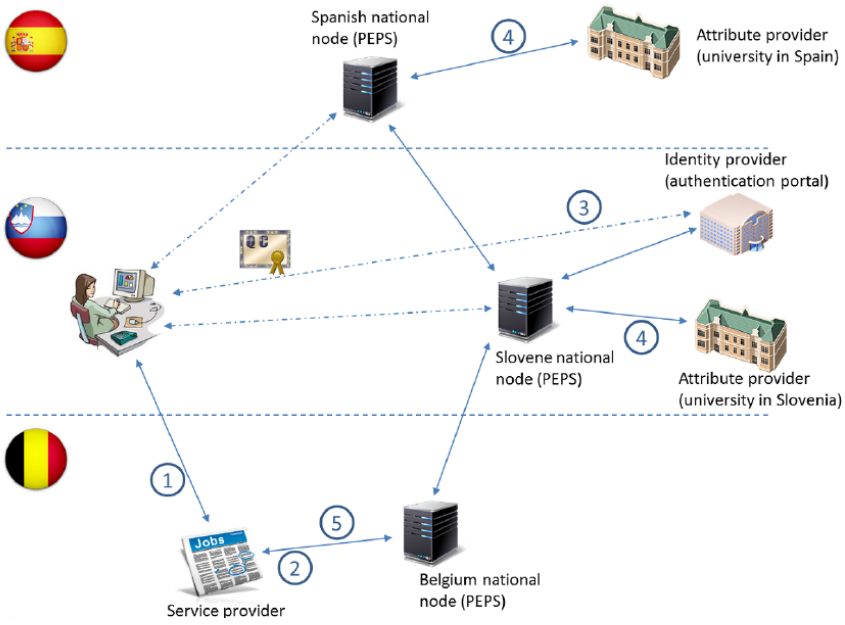


Figure 2.14: STORK 2.0 infrastructure example [4].

- PEPS (Pan European Proxy Server): Each PEPS includes the functionalities which are specific to its Member State, which are typically the interfaces with the local ID providers, domain-specific attribute providers and mandate providers. It plays two principal roles: the S-PEPS (Service PEPS) that attends SP and forwards it to her colleague PEPS or V-IDP and the C-PEPS (Citizen PEPS), which attends to citizen requests and resolves the requests received from her colleague PEPS or V-IDP.
- IdP (Identity Provider): this is responsible for managing and issuing identity information to service providers and doing the authentication.
- AP (Attribute Provider): this entity stores user attributes and returns them to the PEPSes
- SP (Service Provider) offers a protected resource or application to the user and usually initiates an authentication.

Interaction flow

The typical STORK 2.0 flow is described below:

1. Spanish STORK user tries to access a STORK SP that depends on the Italian infrastructure

2. The user indicates her country of origin in order to be authenticated by her home IdP. With this information, the SP redirects the user to the Italian S-PEPS.

3. The Italian S-PEPS uses the country information to redirect the user to her origin C-PEPS, in this case, the Spanish PEPS.

4. The Spanish C-PEPS requires explicit user consent prior to requesting her authentication and the requested attributes.

5. If the user agrees, she is redirected to the IdP, where she is authenticated using her national eID. The IdP requests the user's consent to share the requested attributes with the C-PEPS.

6. The Spanish C-PEPS redirects the user back to the Italian S-PEPS and the latter back to the SP. The PEPS's response includes the authentication statement and the user attributes.

7. Based on the information retrieved, the SP evaluates them and decides whether to grant the user access to the service.

Protocols and technologies

The protocol implemented in STORK 2.0 is called $\text{SAML}_{\text{STORK}}$, which has been designed as a subset of SAML2.0 standard with some extensions [107]. The SAML's extension capability has the advantage of allowing the customization of communication protocol and the transport of specific attributes. It makes it possible, among other functions, to request and transport specific information and attributes required by STORK such as QAA and AQAA. In contrast, the use of these extensions makes direct interaction with other pure SAML systems impossible.

Audience and scope

STORK 2.0 is divided into four areas of interest: eLearning, eHealth, eBanking and Public Services for Business, which are considered the most interesting to promote the use of the European eID. The pilot involves partners from all these areas, with the participation of end users, service providers and European Member States. In fact, STORK has as its target the whole European Union, its citizens and all the SPs that in one way or another want to take advantage of the STORK infrastructure and the security features it offers.

Summary

STORK 2.0 is a pilot based on the STORK project and carried out by 19 European Member States and 59 partners of different types, such as governmental institutions, banks or universities. The initiative was planed with the aim of being helpful in the preparation of the eIDAS regulation. STORK offers and requires high levels of security and privacy to users and services, due to work done at governmental levels. It is based on $SAML_{STORK}$, which uses SAML extension capabilities to introduce new attributes and custom information. These modifications and the security restrictions make STORK incompatible with other standard SAML federations.

2.3.5 eIDAS

eIDAS Regulation [108] on electronic identification and trust services for electronic transactions in the European internal market is based on the work done during the STORK and STORK2.0 projects and it has been designed as an evolution of both. It was published in regulation N°910/2014 and adopted by the co-legislator in July 2014 The European regulation obliges all Member States to become eIDAS compliant in 2018, thus enabling recognition of the notified e-ID means.

It offers a regulatory environment that guarantees people and services the use of their national electronic identifications (eIDs) to use public e-services in all European countries where the eIDAS is already available, so encouraging the creation of a European internal market for electronic Trust Services (eTS) and ensuring across border works, with he same legal reliability as traditional face-to-face or paper based procedures. Indeed, one of the main objectives is to cause the disappearance of the existing barriers to the electronic identification between citizens and services from different Member States using the eIDAS cross-border authentication for at least public services.

Entities and architecture

Like STORK, eIDAS has a hybrid topology based on the previous STORK architecture. For each country, eIDAS has one eIDAS Node that acts as a centralized point for all the users, Services Providers and Identity Providers of that country. At European level, all eIDAS nodes are organized in a distributed topology that allows each country to maintain the control of what users and services providers are connected within its borders, while managing fine grain control in the relations with the rest of the countries.

The entities involved in eIDAS architecture [109] are described below:

- Member State: State covered by the eIDAS regulation. It acts as Sending MS when it is in charge of authenticating the eID scheme and sending the ID data to the Receiving MS. In contrast, it acts as Receiving MS when one of its Relying Parties requests the authentication of a person through it.
- Relying Party (RP): this entity represents any natural or legal person, in general it is a service provider, that requires an electronic identification of a citizen or a trust service in order to provide a service.
- eIDAS-Node: this is an operational entity involved in cross-border authentication of persons. A Node can have different roles, which are distinguished in this specification (eIDAS-Connector/eIDAS-Service, see below).
  - ⋆ eIDAS-Connector: an eIDAS-Node requesting a cross-border authentication.
  - ⋆ eIDAS-Service: an eIDAS-Node providing cross-border authentication.
    - ∗ eIDAS-Proxy-Service: an eIDAS-Service operated by the Sending MS and providing personal identification data.
    - ∗ eIDAS-Middleware-Service: an eIDAS-Service running Middleware provided by the Sending MS, operated by the Receiving MS and providing personal identification data.
- eID Scheme: There are two options to provide the eID scheme according to the eIDAS-Service interface. The Proxy based scheme is a (notified) eID scheme which provides cross-border authentication via an eIDAS-Proxy-Service and the Middleware based scheme is a (notified) eID scheme which provides cross-border authentication via eIDAS-Middleware-Services

Interaction flow

This section describes the process flow to authenticate a person, enrolled in the eID-scheme of the Sending MS, to a relying party established in the Receiving MS [110].

1. A user tries to access a resource protected by the RP, which sends an authentication request to the eIDAS-Connector responsible for RP. The eIDAS-Connector can be directly attached to the RP (Decentralized MS) or operated by a separate entity (Centralized MS).

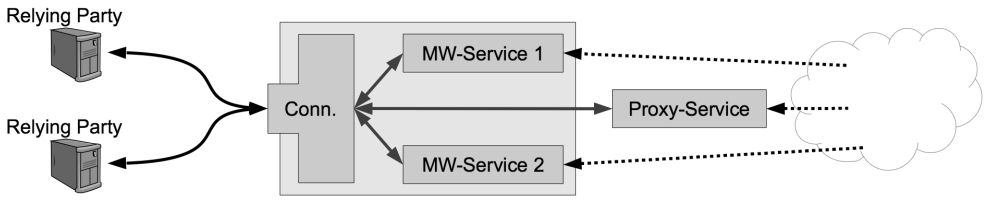


Figure 2.15: eIDAS components.

2. The eIDAS-Connector requests the MS in charge of authenticating the user on the basis of the eID scheme.

3. The eIDAS-Connector sends a SAML Request to the eIDAS-Service corresponding to the chosen MS.

4. The eIDAS-Service must verify the authenticity of the Request and check the terms of access and the Level of Assurance (LoA) required in the request.

5. The eIDAS-Service performs the authentication of the person according to the selected eID scheme and sends back a SAML Authentication Response.

6. The eIDAS-Connector verifies the Response message authenticity and decrypts the Assertion. The Connector must verify the LoA and send the received authenticated person identification data to the requesting party

Protocols and technologies

SAML 2.0 is the base protocol to communicate and encode eIDAS information and messages. As in STORK, eIDAS uses SAML extension capabilities to customize the information transmitted and also to encode specific information.

The eIDAS Project also provides software to simplify the adoption by Providers and Member States. DG DIGIT provides an implementation of the eIDAS-Connector and the eIDAS-Proxy-Service as a single package licensed under the EUPL, under the CEF (Connecting Europe Facility) program. CEF Management Board manages the implementation requirements, including provisioning and service support. This implementation is provided bundled with the Middleware provided by the Member States.

Audience and scope

eIDAS, as an evolution of STORK, has is target audience in all the Member States of the European Union, including their citizens and public and private services related with

government administrations.

The stakeholders of the eIDAS-Network are, in the first place, the citizens, who expect confidentiality with their person identification data at the same time that they require the eIDAS-Network to respect their privacy. In the second place, there are the operators of components of the eIDAS-Network with requirements derived from the requirements of the relying party and the citizens. Finally, all the relying parties that require authenticity/integrity of the personal identification data received to fulfill the data protection obligations relating to confidentiality and privacy.

Summary

The eIDAS Regulation was published in 2014 as a regulatory environment that guarantees people and services the use of their national eIDs to use public services in all European countries with the same legal reliability as traditional paper based procedures. eIDAS promotes and facilitates the use of cross-border electronic identification and trust services, and guarantees transparency and accountability. The regulation also contributes to the enhancement of trust and security of digital transactions and thus to the building of the Digital Single Market. The objective is to extend and popularize the use of eID among citizens of the European Union in their relations with institutions as well as in the private sector.

2.3.6 Other existing AAI solutions.

Shibboleth

Shibboleth [111] [112] is a web authentication and authorization infrastructure (AAI) based on the use of SAML and web redirections to determine if a user can access a resource through its web browser. This decision is based on the user's information kept at his home institution.

This mechanism enables the definition of identity federations such that the user always authenticates through his home institution, and then the information needed is sent where necessary to authorize him. Shibboleth defines a Service Provider (SP) and an Identity Provider (IdP). The former is at the institution providing resources and the latter is at the institution managing the user's identity. Thus, when the user accesses some protected resource, the SP asks the IdP where the user is from, in order to obtain information about him. If the user was previously authenticated, the IdP returns the needed information, otherwise the user is redirected to the IdP to be authenticated. In this way, web SSO is

provided. Moreover, this process is performed transparently to the user by means of web redirections.

Shibboleth defined a new service called WAYF (Where Are You From) to locate the user's IdP with or without user interaction. It is independent of both the SP and the IdP. This element is essentially a proxy for the authentication request passed from the SP to the SSO Service.

CardSpace

Windows CardSpace [113] is client software that enables users to provide their digital identity to online services in a simple, secure and trusted way. It is what is known as an identity selector: when a user needs to authenticate to a web site or a web service, CardSpace pops up a special security-hardened UI with a set of "information cards" for the user to choose from.

Each card has some identity data associated with it – though this is not actually stored in the card, which is either given to the user by an identity provider such as their bank, employer or government or is created by users themselves. Having the user as an identity provider sounds a bit strange on first acquaintance (who would trust the user?) but this is a very common scenario: this is what we do every time we register at a web site.

The CardSpace UI enables users to create Personal cards (aka self-issued cards) and associate a limited set of identity data. It also enables the user to import Managed cards from third party identity providers. When the user chooses a card, a signed and encrypted security token containing the required information (e.g. name and address, employer's name and address, or credit limit) is generated by the identity provider that created the card. The user, in control at all times, then decides whether to release this information to the requesting online service. If the user approves. then the token is sent on to this relying party, where the token is processed and the identity information is extracted.

Higgins

Higgins [114] is an open source framework for the web that allows users and systems to integrate identity information and profiles, besides information on social relations across multiple systems, applications and devices. The framework supports many identity management systems such as CardSpace (Higgins is a direct free software implementation) and OpenID; in addition, it is in the process of supporting Idemix. Its infrastructure is based on the most popular identity protocols such as WS-Trust, OpenID, SAML, XDI or

LDAP. The framework offers similar features in the treatment of privacy to CardSpace, with the added benefit of working with open source implementations and the use of standards.

The main advantage of Higgins for developers is that it facilitates the creation of applications and services that work with other identity managers. For the user, Higgins provides a graphical interface, called the Identity Selector, which provides control of digital identity management to the user.

The Higgins project also provides an API and a data model for integration and federation of digital identity and security information from multiple sources. It also offers adapters to enable integration in the framework of various data sources such as directories, communication systems, collaboration systems and databases with different protocols and schemas.

PrimeLife

The research project PrimeLife [115] is led by IBM and funded by the European Commission's 7th Framework Programme. The goal is to bring privacy and identity management to the collaborative environments of Internet, allowing people to control personal data trails left behind their navigation. PrimeLife is based on the success of the FP6's PRIME project (Leenes, 2008) which developed a working prototype of an identity management system that improves privacy, while it was also the starting point of the Idemix project.

Idemix

Idemix (Identity mixer) [116] is a suite of cryptographic protocols developed by IBM Research that allows strong authentication while maintaining privacy in identity management. Its operation is based on the use of an anonymous credential system that allows aggregation of attributes from different digital identities and partial disclosure of these.

The project also focuses on the development of new mechanisms to support online privacy and identity management in order to interact in new ways with the new services that have emerged on the Internet, communities and the Web 2.0. To do this, the project works on the design and the implementation of an appropriate policy language that allows web sites and end users to express their privacy policies and preferences. In turn, it undergoes a process of analysis and design in order to implement the prototype of a tool that allows users to evaluate the reliability of collaborative content, such as wikis and blogs.

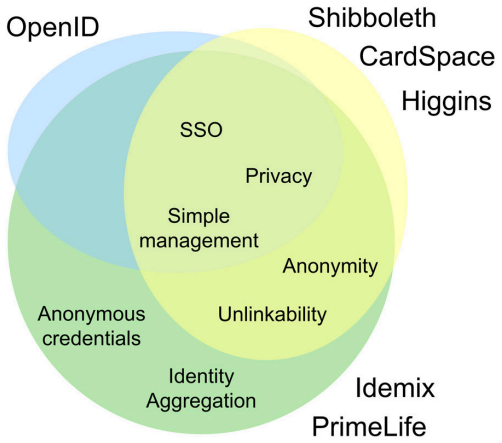


Figure 2.16: Summary of IdM solutions.

Summary

Identity management systems are focused on enabling the user to manage their digital identity on the network to improve their privacy offering different characteristics (Figure 2.16). OpenID proposes a solution based on a unique identifier to access the services provided by the management, but it also represents a single point to undermine security. Other systems such as CardSpace, Higgins or Shibboleth offer more advanced features such as anonymity and unlinkability. PrimeLife (based on the project Idemix) also provides identity aggregation and complex privacy mechanisms that allow the use of anonymous credentials to enhance privacy.

2.3.7 Identity Federation Summary

In order to appreciate the intense work done in the area of Authentication and Authorization Infrastructures to guarantee users' privacy and security, access control to the services and improve infrastructures and services deployed, this section has reviewed some of the most relevant identity federations based on SAML.

In the first place, we have analzsed Moonshot as a general purpose option that implements the ABFAB architecture based on EAP/RADIUS and SAML. We continue with eduGAIN and EUDAT, which are very important in educational and research sectors. Both federations have worked hard on the mechanisms of federating new organizations, eduGAIN through the use of metadata and EUDAT though UNITY technology and the integration of external IdPs for the first registration of a new user.

Finally, the review places special emphasis on those options related to the

Administration sector such as STORK and eIDAS that have European governmental support, offering as main feature the highest level of LoA thanks to eID infrastructure deployed at all European Member States.

Table 2.17 summarizes of all the identity federations studied in this section, with the aim of offering a general overview of their main characteristics.

	Moonshot	eduGAIN	EUDAT	STORK	eIDAS
Entities	Client, RP, IdP, Trust Infrastructure	SP, IdP, DS, MDS	SP, Primary IdPs, B2ACCESS AAI, EUDAT Service Endpoints	SP, PEPS (S-PEPS & C-PEPS), IdP, AP	RP, eIDAS Node, IdP
Topoloy	Hierachical	Distributed	Centralized	Hybrid	Hybrid
Protocols	EAP/ RADIUS, SAML2	SAML2	SAML2, OAuth2, X.509, ...	$\text{SAML}_{\text{STORK}}$	$\text{SAML}_{\text{STORK}}$
Software	GSS-API, SSPI Kerberos	Shibboleth	B2ACCESS (Unity)	Demo software for SP, PEPS, IdP& AP	Demo software for RP, Nodes, eID Scheme
Audience	general-purpose	Researchers, students, university staff	Researchers	European citizens	European citizens
Scope	general-purpose	Research centers, universities	Research and data centers	eLearning, eHealth, eBanking, Public Services for Busines	Public administrations, public and private services
Countries involved	Global	Global	Europe	Europe	Europe

Table 2.17: Identity federations summary.

2.4 Conclusions

People and services increasingly demand a greater interconnection to unite and offer common services as a way to win new users, improve existing services and simplify management. There exists a wide variety of identity federation systems according to the requirements of users, services and scopes, and the most active work in this IT area is being done by the research and administration sectors. The latest trends and advances focus not only on guaranteeing user privacy and improving the security, but rather on enhancing the improvement of existing federations, in some cases by adding new advanced features and in others through interconnection capabilities, which allows a larger user base, unified services and management, and creates a set of new services in line with the new possibilities.

This chapter has introduced some of the most relevant protocols and technologies currently used by Internet organizations to ensure end users' identity for Web services. Some of them, such as SAML2 and more recently OAuth2, are so widespread that they have become de facto standards for transporting and encoding authentication and authorization information. These standards, seen in Section 2.2, provide the fundamental tools for building identity federations, but while they are powerful enough to do so, how they are adopted and used determines the characteristics that identity federations end up offering.

As seen at Section 2.3, different federations working on the same standard (as SAML2.0) offer very different security and IdM characteristics. Advanced features such as anonymity, partial identities, cross-layer SSO, advanced access control or interoperability between heterogeneous services are often not covered. In the following chapters, we will present different scenarios that have not been resolved by the current federation systems and we will offer the necessary architectures, tools and mechanisms to demonstrate their feasibility.

Chapter 3

Integration Architectures for enabling digital identity interoperability

3.1 Introduction to digital identity interoperability

Generally users have one account or profile for each one of the services they use. This presents several drawbacks that limit the usability of this approach. Users need to remember lots of different user names and passwords, especially if they want to preserve their privacy and to prevent their activities being traced. Users are also requested to fill out registration forms for each service they register for, so having to insert again and again their identity information. This may lead to inconsistencies and to out-of-date information.

An individual digital identity collects information that characterizes the individual from the rest. Furthermore, once identity information is shared among services, it is of paramount importance for the user to control the access to this information, determining by whom, when and how information about himself can be retrieved, so the user has a high control over how his privacy is being preserved. Partial identities can be defined by a subset of information on specific context within a domain as a role for age, nationality or their preferences. Hence, one partial identity may be used for work, another for social relationships, or hobbies. On the other hand, users may want to make use of advances features such as accessing a service anonymously, providing just a little information about himself (e.g. he is authenticated and is over 18 years old).

Identity Management (IdM) emerges from the need to establish control of these digital identities, protect personal information, establish confidence in the services (Web and network access) and electronic transactions. IdMs allows the same identity information to be shared among the different services. Like this, users only need to remember a

small number of credentials, while services will be provided with reliable and up-to-date information. It offers user a way to centralize the management of their identity information, such as simplifying the access to services with mechanisms like Single Sign-On. Identity management should be applied in two different planes: horizontal (across domains and services) and vertical (across network layers).

From the point of view of services, there are also advanced features feasible like flexible service composition, accounting service, dynamic service discovery and security token service that allow rich federations with better use experience. There is a large heterogeneity between services due to the boom of new web services and different authentication and authorization protocols. There is a need to move towards integration scenarios that allow the interoperability between them with the aim of also allowing resource sharing and improve the user experience, always by means of mechanisms which ensure the privacy of the users.

The integration of heterogeneous authentication and authorization protocols as well as identity information from different sources are not trivial due to the wide variety of technologies available as seen in Chapter 2. Industry and academia have invested considerable effort to bring IdM forward, to take advantage of features like improved usability and security. Nevertheless, there are important issues that have not been addressed so far. IdM should be leveraged as a key technology of the Future Internet, tackling problems like the integration of heterogeneous services and technologies from an IdM perspective as well as backward compatibility and a new access control infrastructure are required by future IdM solutions.

We consider all these aspects by extending existing IdM solutions with new security and privacy enablers that are part of the overall SWIFT framework that will be present in Section 3.2. In second place, in Section 3.3 we present GEMBus Project (GEANT Multi-domain Bus), a service-oriented middleware platform that allows flexible services composition, and their on-demand provisioning and deployment to create new specialized task-oriented services and applications. It is built upon state-of-the-art Enterprise Service Bus (ESB) technologies and extends them with new functionalities that allow dynamic component services deployment, composition, and management, with an enhanced security service that enables the integration of services and users from different security technologies.

3.2 Interoperability mechanisms for Identity Management: SWIFT

The SWIFT project (Secure Widespread Identities for Federated Telecommunications) [8, 9, 11, 117] defines an identity management framework that aims to provide a solution to all these advanced aspects like digital identity, privacy and security requested by end users and service providers. SWIFT aims for users to be able to group all their accounts in the so called *virtual identities*. Moreover, it allows users to access the services anonymously, so avoiding traceability by third-parties.

Starting from this identity federation scenario, it is desired, moreover, to introduce SSO management not only at service layer such as web services, but also in those scenarios where network access requires a previous authentication step. That is, to perform a single authentication process throughout the network access and to allow SSO access to the upper layer services. Besides, it adds authorization management mechanisms to the traditional authentication process based on user attributes.

In order to describe these characteristics, this section is structured as follows. Section 3.2.1 offers an overview of how SWIFT incorporates this set of advanced identity management aspects. In section 3.2.2, the components of the architecture are briefly described, while section 3.2.3 depicts a use case illustrating the basic behavior of the different entities. In section 3.2.4 the architecture and the interchanged messages are validated using formal language. Next, section 3.2.5 describes the software developed to test the architecture. Finally, section 3.2.6 provides a summary of this project with an overview of related work.

3.2.1 Advanced identity management aspects in SWIFT

The following section provides a high-level overview of the most important advanced identity management aspects covered by SWIFT.

Identity Aggregation

Nowadays, users on Internet have a large number of accounts in different service providers (e.g. electronic bank, email, social networks, blogs, etc.). Each one of these accounts has associated authentication credentials and some amount of personal information related to the user (e.g. name, address, hobbies, work place...). This exposes the user to a series of drawbacks, like having to remember the credentials associated to each one of them,

the difficulty of controlling privacy (e.g. revoking access to my phone number in all the accounts) and the tedious need to fill in registration forms usually with the very same data.

SWIFT allows users to create virtual identities, that link or aggregate authentication credentials and attributes coming from different providers [118], as shown in Figure 3.1. These providers can play different roles: Authentication Providers (AuthNP) using Authentication Statements asserting to other entities that a particular user has been authenticated properly, or Attribute Providers (AttrP), storing and managing the various pieces of information on a particular identity. A single provider could play both roles if required.

Virtual identities are created and managed at special identity providers called Identity Aggregators, which act as an intermediary entity between Service Providers (SPs) and the end user information stored at the different identity providers. For each new virtual identity there exists a VID (virtual identifier), that identifies univocally such an identity within the context of an Identity Aggregator (e.g. alice@aggregator.org). They allow the drawbacks mentioned before to be solved, since it is not necessary to remember a large number of credentials (only the aggregated ones) and SPs obtain data dynamically from the virtual identity (so avoiding the problems of redundancy and privacy). A more detailed description of how these virtual identities are created and used can be seen in Section 3.2.4 and here [119].

Figure 3.1 shows an example where the user has defined a virtual identity using her credentials from one of her accounts at Identity Provider A (that will act as Authentication Provider). Besides, the end user has also aggregated some attributes from her accounts at Identity Providers A, B and C (that will play the role of Attribute Providers).

Identity-based access control

In an identity management system, the point of view of service providers regarding information releasing is the opposite of the one from end users. While the latter prefer to reveal the minimum amount of information as possible when they access to a service, service providers need to obtain some information about the users in order to, on the one hand, provide the service and, on the other, obtain a better knowledge about the profile of their users. Therefore, a good identity management system must find a compromise to cover the requirements from the end users and the service providers.

Thus, a service provider needs to know some identity information about the users to be able to decide whether the service will be provided or not. This process of gathering information and deciding about the access to a service is called *Access Control* [29] [118].

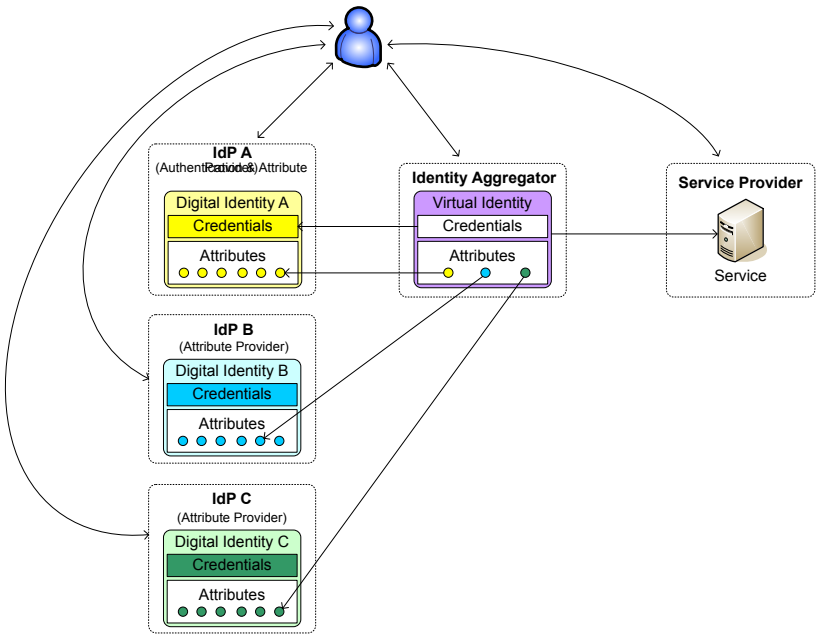


Figure 3.1: Identity aggregation.

In SWIFT, access control is based on the virtual identity of the user. When a user desires to access a service she presents an Initiation Statement or an SSO Token (if SSO is desired) [10]. However, the Service Provider requires information asserting that she is a valid and identified user in the system. Thus, it contacts with the Identity Aggregator and forwards the received information from the end user. Once the Identity Aggregator has determined the identity of the user, it provides the Service Provider with the *Authentication Statement*, which checks that the user has been authenticated by the Identity Aggregator.

In addition to the verification of the user's identity, the service provider may require an additional authorization process to verify that the end user satisfies certain requirements before providing her with the service (for example, to be over 18 years old, to have a certain nationality or to have a positive balance on a banking account). Thus, the service provider must contact the Identity Aggregator and request the end user attributes. The amount of user's information that will be available for the service provider will depend on the attributes that were aggregated by the end user to that virtual identity (Sec. 3.2.1) and the Attribute Release Policies (Sec. 3.2.1) that were defined by her. As the environment is highly distributed, policies have been introduced to specify the user's intentions [120].

To summarize, access control is performed at the service providers based on the authentication statements issued by the Identity Aggregator, as well as on the end user

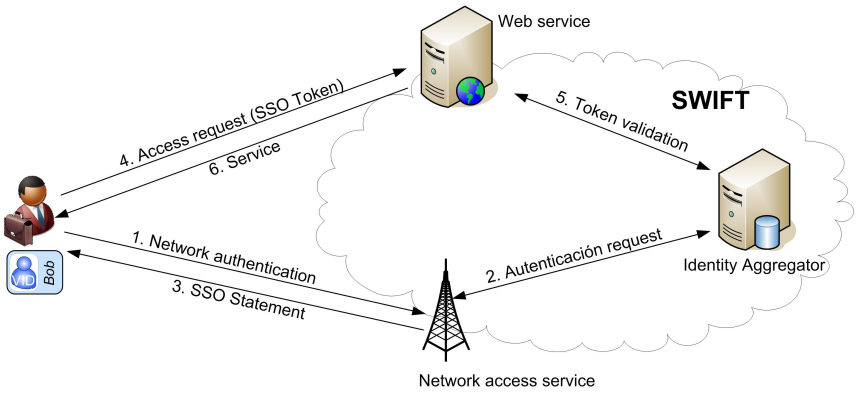


Figure 3.2: Cross layer SSO.

attributes, obtained from the Attribute Providers through the Identity Aggregator.

Cross-layer SSO

One of the main objectives of SWIFT is to provide a unified Single Sign On (SSO) mechanism [21] [121] that should work across different network layers, in such a way that it allows the reuse of the authentication performed at one layer to authenticate at a different one. This means that if an end user is authenticated during the access to the network using one of her virtual identities (for example by means of EAP-MD5, EAP-TLS, PEAP, ...) the authentication material obtained during the process will allow her to access later on services at the application layer (for example, web services) without repeating the authentication process again.

Figure 3.2 shows an example where a SWIFT end user is first authenticated to access the network, and then she makes use of the defined SSO mechanisms to access a web service. A complete authentication process is performed for the network access, by means of some network level authentication protocol (for example, EAP [93]). As a result of this authentication, the user obtains a *SSO Statement* generated by the Identity Aggregator. With this statement the user is able to generate the specific *SSO Token* used to access the web service. The service provider will request the validation of the token to the Identity Aggregator before providing any service to the user.

This provides an added value to end users, but with the drawback of not being transparent for them, as happens with the traditional methods. The management of SSO tokens and statements is described in [119].

Privacy management

User privacy management is one of the pillars of the SWIFT architecture. The objective is to provide the end user with the required mechanisms and procedures to control the amount of personal information that is revealed to the different entities in the system. In SWIFT, end user privacy is taken into account from the moment virtual identities are created. Throughout the creation process (called *enrolment*), an end user does not reveal the actual identifiers she has in the different involved providers. A pseudonym (identifier that is created by an entity to refer another entity) is established instead between those providers and Identity Aggregator. Hence, it is impossible for the Identity Aggregator to determine whether the same user has created more than one virtual identity.

As described in section 3.2.1, each virtual identity has a unique identifier called VID. This VID, therefore, must be presented by the end user on each access to a protected service, with the purpose of being identified by her Identity Aggregator. In order to protect user privacy, and since this information must go through the service provider, the VID is protected in such a way that only the Identity Aggregator is able to know all the details. Thus, service providers are not able to recognize the identity of the end user (anonymity) nor to relate different accesses of the same user to the same service (*unlinkability*) [118].

Besides protecting their identity, SWIFT allows end users to control what kind of information can be revealed and to whom. As indicated before, when an end user accesses the different available services, her intention is to reveal the minimum amount of information possible (section 3.2.1). For this reason, SWIFT makes use of advanced attribute release policies that are established by the end user at the Identity Aggregator, which becomes the trust point of the framework. These policies are expressed using the XACML language [83] or a newly introduced version of distributed XACML [120].

Virtual terminal

Currently, an end user has more than one device or terminal. Sometimes the use of certain digital identities is limited to be bound to some specific devices due to, for example, processing capacity restrictions, security constraints, etc., so conditioning the flexibility and usability of the system. With the aim of mitigating this problem, the concept of virtual terminal has been defined within SWIFT [122], which seeks to make identities usable among all the user's devices.

The use of virtual terminals allows, for example, the reduction in the number of authentication processes that must be performed among the devices, enabling the use of authentication capabilities of a device in another one (for example the use of a SIM

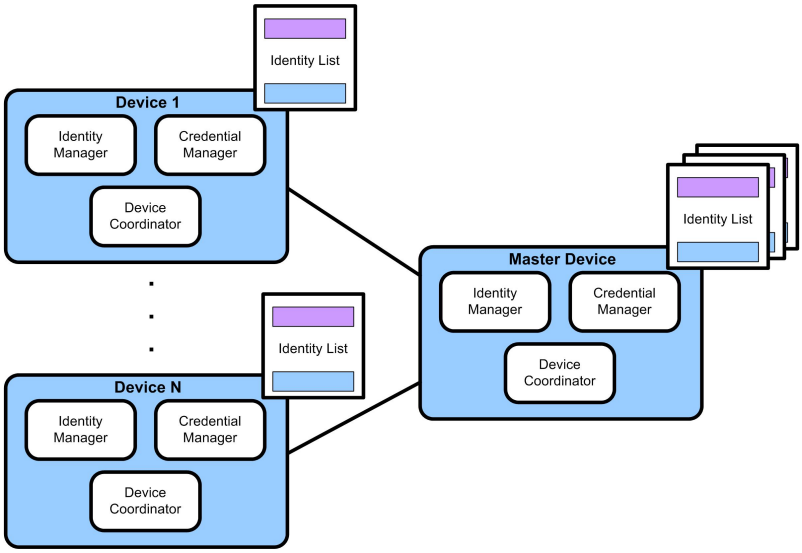


Figure 3.3: Virtual terminal architecture.

card [123] to authenticate a web video streaming session) and moving application sessions among devices in a single way for end users and service providers.

In the virtual device architecture [122], one of the devices assumes the role of *master device*. Its responsibilities are to coordinate the inclusion and removal of devices, as well as the announcement of capabilities and the selection of the most appropriate device to perform an operation. Figure 3.3 shows the basic architecture for the virtual terminal, where the different functional blocks that compose a device are depicted. Each of these functional blocks is in charge of a specific task: credential manager, identity manager and device coordinator. Therefore, this architecture allows the transparently use and managing of a group of devices as if they were a single device, so sharing identity information and session control.

Mobility

Identity management mechanisms provided by SWIFT, especially the SSO one, allow simplification of the management of terminal mobility [124]. Thus, when a terminal moves from one network to another, it can perform a faster authentication which involves a lower number of messages and computational overhead, by making use of the material obtained during the first authentication.

Besides optimizing the network authentication process, the possibility of joining identity management with mobility management improves the access and availability of dynamic user information (for example the current location of the user or her preferences for certain

services based on the current connexion capabilities). An example of mobility would be presented when Alice connects to the network with her PDA, using her home cable connection and then she decides to move to her car, where she wants to keep connected to the network (in this case using a UMTS connection) and to maintain the active session alive.

SWIFT defines a framework that allows the management of end user mobility based on the use of authentication tokens and the mediation of the Identity Aggregator.

Advanced policy management: deductive policies

Along with user's information, policies play a fundamental role in IdM. They model the expected behaviors of the entities. SWIFT puts the focus on those policies directly involved with user information, that is, attribute release policies and access control policies.

Traditional policy management assumes that the entity that is going to take a decision (usually denoted as PDP) has all the required policies defined in its own domain. In federated environments, this may not scale well, since every single organization within the identity federation must have a local copy of the policies. For example, if an organization O_1 would like their users to have limited access to services provided by another organization O_2, both should agree on the policies that should be defined in O_2 that enforce O_1 preferences. Additionally, O_2 should have access to all the information from O_1 that was required to take the decision. This last requirement may be inconvenient to organizations that do not wish to disclose internal information about their processes.

Instead, SWIFT defines a distributed and deductive policy framework, which allows distribution of authorization decisions across different domains. The different entities within the federation manage the policies regarding their own domain, but they offer an endpoint for deductive policy decisions. A policy deduction consists of a reference in a local policy that indicates to the PDP that, in order to be evaluated, it must first request a remote decision from a different administrative domain. This solution solves the issues presented in the example above, since O_2 access control policies for O_1's users could include a deduction that refers to an O_1 decision.

To achieve these objectives, the XACML language requires some modifications and extensions as is described in detail in [13] and [125], so as to include deductions in the policy structure. Figure 3.4 depicts an example scenario where deductive policies are applied. The Firewall Service (O_2) will only allow users to pass through it when they are authorized to access the target Web Service (O_1). To do this, the policies defined in the Firewall service include a reference to the policies in the Web Service. Thus, when a user

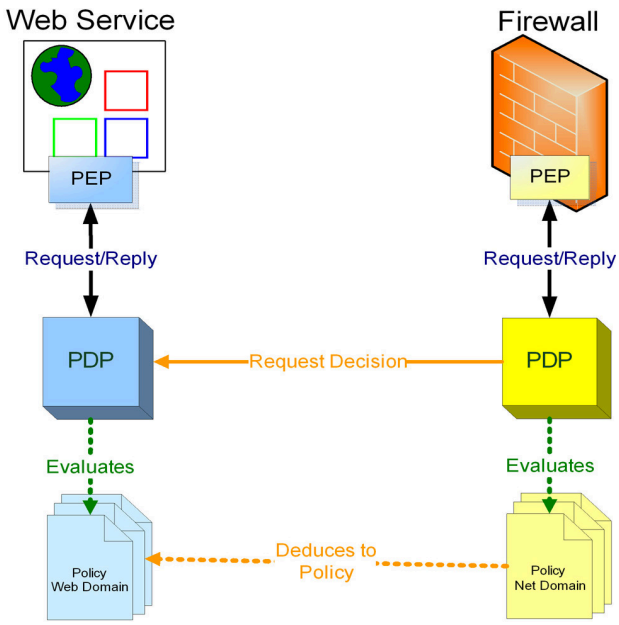


Figure 3.4: Deductive policy schema.

wants to pass through the firewall, the Web Service's PDP is queried by the Firewall's one to obtain a decision.

3.2.2 Components of the SWIFT architecture

While section 3.2.1 has described a series of important aspects within the SWIFT functionality, this section focuses on the description of the main components of the framework that perform the functionality described in this document.

- **Identity Aggregator (IdAgg or IA):** This is the central entity of the architecture. It is the center of trust for the components and acts as a mediator between the user, the services and the identity providers. Its main functions are the management of the virtual identity life-cycle, the control over the authentication and SSO mechanisms and the mediation in the attribute retrieving process.
- **Authentication Provider (AuthNP):** It is responsible for the end user authentication and for the provision of that information to the IdAgg. It can allow different authentication methods (SIM card, login/password, digital certificates, etc.).
- **Attribute Provider (AttP):** It manages end user identity information in form of attributes. These attributes are stored and distributed to the IdAgg when they are requested.

- **End User (EU):** This is the end user of the system that desires to access to the services offered by the different providers. She has the control over her different identity accounts, including her virtual identities.
- **Service Provider (SP):** Provides services to End Users. In the scope of SWIFT, a service is anything that provides an additional value to the user, including the network access service itself. In this last case, we would talk of Network Provider (NP) as a particular case of SP.

3.2.3 Use Cases

In this section we present a high level vision of four use cases that have been defined for SWIFT and that demonstrate SWIFT identity management capabilities, specifically the use of virtual identities, attribute aggregation and cross-layer SSO authentication.

Initial authentication and authorization for the network access

In this use case, Alice (EU) desires to access the network using one of her virtual identities. As shown in Figure 3.5, Alice initiates the process presenting an Initiation Statement to the network service provider (NP), including information about the VID (step 1). As described before in this document, this information is protected to prevent anyone except the IdAgg recognizing the EU. The statement is forwarded from the NP to the IdAgg, which identifies that it refers to Alice's virtual identity (alice@idagg.com) and determines the AuthNP that will be in charge of her authentication (steps 2 and 3). Once Alice has been authenticated (by means of some network authentication mechanisms) (4), the AuthNP sends an artifact to the end user (step 5), enabling her to retrieve a SSO Statement at a later point of time (step 14 and 15). Additionally the AuthNP provides an *Authentication Statement* to the IdAgg, where it is certified that Alice has been successfully authenticated (6). Following the same rationale, the IdAgg generates an *Authentication Statemen*t for the NP (7) where, besides asserting Alice's authenticity, a pseudonym is provided. Thus, the NP only needs to trust in the statement issued by the IdAgg, without knowing anything about the AuthNP that actually performed the authentication.

Once the authentication has been performed, the NP can request end user attributes to complete the process of authorization, prior to providing the service to Alice. First, the NP uses the pseudonym obtained in the previous step to request the attribute *age* from the IdAgg (8). The IdAgg determines which virtual identity is behind the indicated pseudonym, whether the requested attribute is available or not (i.e. it was aggregated

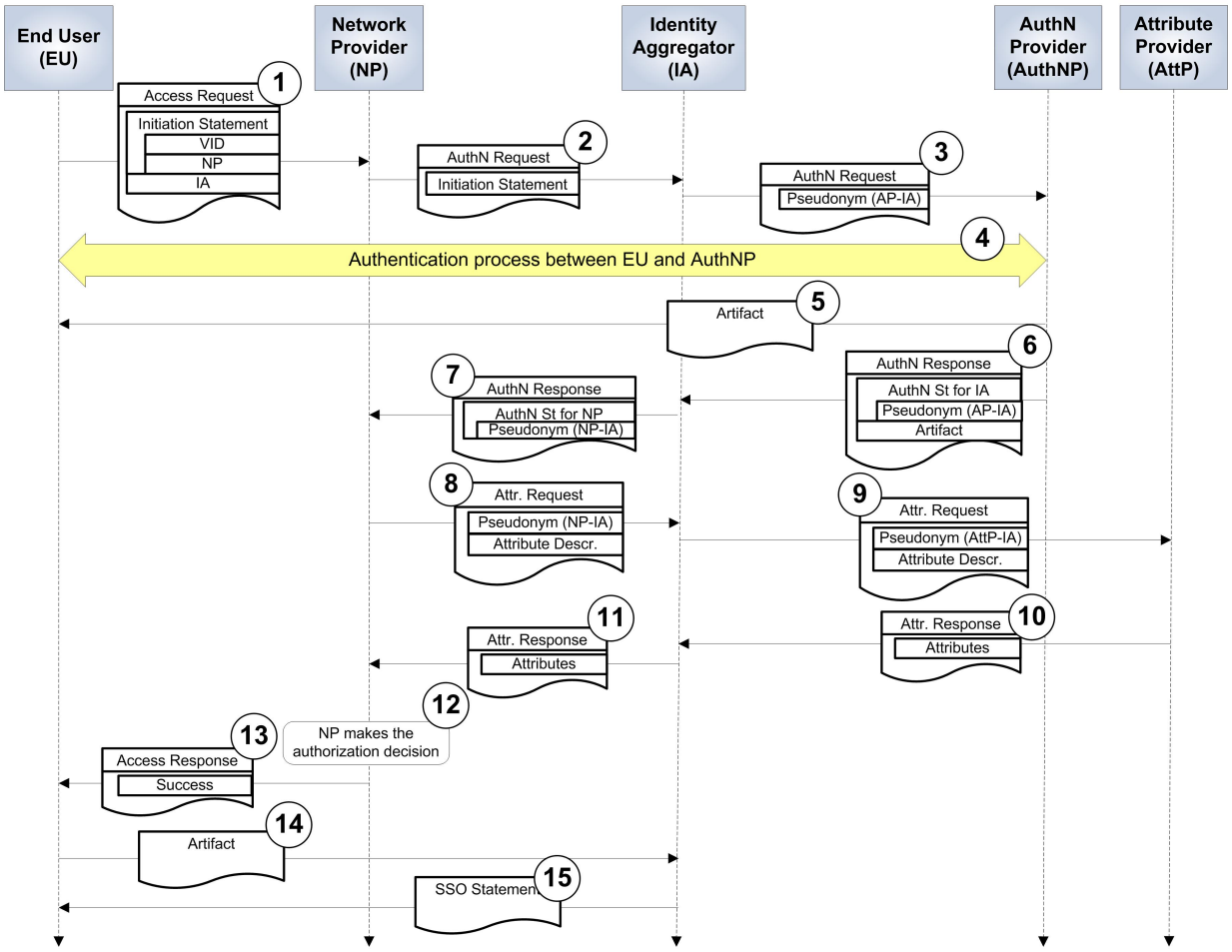


Figure 3.5: Message flow for the network authentication and authorization.

by the end user and the privacy policies allow to provide it to the NP). If so, the IdAgg contacts with the AttP and requests the attribute (9). The AttP provides the attribute to the IdAgg (10), which does the same for the NP (11). Likewise, the NP only trusts the IdAgg for attributes provision. Once the NP has obtained the attribute *age* it can verify, by means of the corresponding access control policies (12), that she can be provided with the service (13) since she is over 18 years old. During this process the NP never knows Alice's actual identity, thus preserving her anonymity. Finally, Alice can obtain the *SSO statement*, which is necessary to make use of SSO mechanisms (14 and 15) using as reference the artifact obtained during the authentication with AuthNP (5).

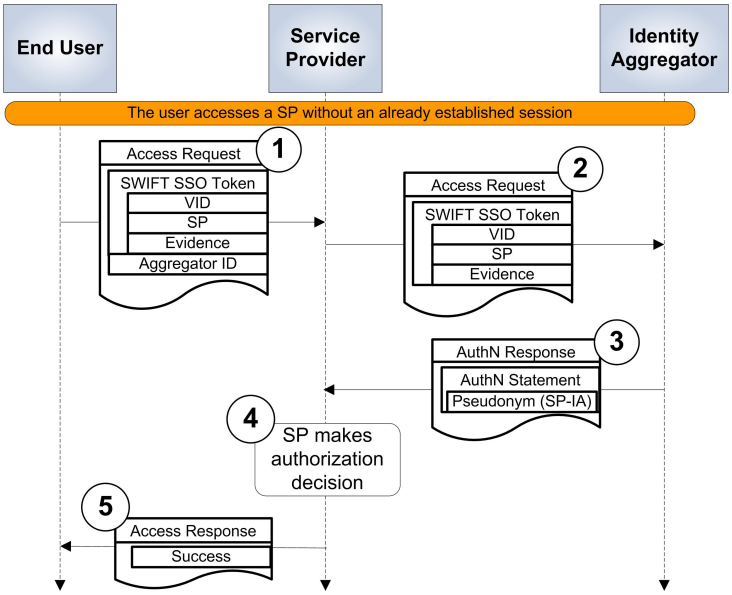


Figure 3.6: Message flow for the web SSO authentication.

SSO authentication for web access

In this use case, Alice (EU) wishes to access a web service using one of her virtual identities and the SWIFT SSO mechanisms. As a previous step, Alice generates a *SSO Token* based on the *SSO Statement* received during the network access authentication and the SP that she wants to access. This token is protected in such a way that only the IdAgg is able to read it. As shown in Figure 3.6, Alice starts the exchange of messages by sending this SSO Token to the SP (1). After that, the SP forwards the token to the IdAgg in order to validate its authenticity (2). Then the IdAgg determines the virtual identity of Alice (in this case, alice@idagg.com) and verifies that the *SSO Statement* used for its generation correspond to the one generated for Alice's last authentication. Once the IdAgg has validated the *SSO Token*, it generates and provides an *Authentication Statement* (3) to the SP. At this point the SP can perform the authorization process as described in the previous section (4) and provide the service to Alice (5).

As for the previous use case, the web access is performed in an anonymous way to the SP, which trusts the IdAgg for the identity management processing (authentication and attribute provision). Besides, the attribute retrieval is common for all the use cases, and is the pillar of the authorization process.

Initial authentication and authorization for the web access

This use case describes the situation when an unauthenticated EU (Alice), owning a virtual ID, wants to access a web service provided by a SP. Web redirections are used to carry out the process of authentication. Figure 3.7. depicts this use case.

Alice wants to access a web service provided by SP. As an initial step, she generates an Initiation Statement including the VID she desires to use for the access and the SP being accessed. Then, she contacts the SP and sends it a new Access Request message including the newly generated statement along with the identifier of the IdAgg that manages the virtual ID (1). With this information, the SP is able to redirect Alice to the appropriate IdAgg, including an Authentication Request message, which transports the received Initiation Statement from the user (2).

When the IdAgg receives this message, it looks for the virtual ID referenced by the received VID and determines the AuthNP and pseudonym that have to be used to authenticate Alice. Then the IdAgg redirects Alice to the AuthNP, including a new Authentication Request message (3) that indicates the pseudonym of the user to be authenticated. The AuthNP prompts Alice for her user name and credentials (4). If the authentication is successful the AuthNP generates an Authentication Statement, where it is asserted that Alice has been correctly authenticated. Then she is redirected back to the IdAgg including this statement along with the redirection (5).

When the IdAgg receives the statement, it generates a new one for the SP, as well as a SSO Statement for the EU. The new Authentication Statement contains a pseudonym that must be used by the SP to refer to Alice. Then the IdAgg redirects Alice to the SP including these two statements (6), though Alice extracts the SSO Statement before reaching the SP (7).

At this point, the SP is aware that Alice has been successfully authenticated by a trusted AuthNP, but without knowing which provider it was or the user name of Alice. Now, the SP can perform an authorization process (8) (as will be described in detail in the following use case and provide the service to Alice (9). This use case complies with the identity aggregation and privacy management objectives seen in Section 3.2.1, since a virtual identity is used to access the service, while Alice's actual identity is not revealed to the SP.

Attribute retrieval and authorization

In order to be granted access a service, Alice may require not only to be properly authenticated, but also to fulfill a series of requirements. This is called the Authorization

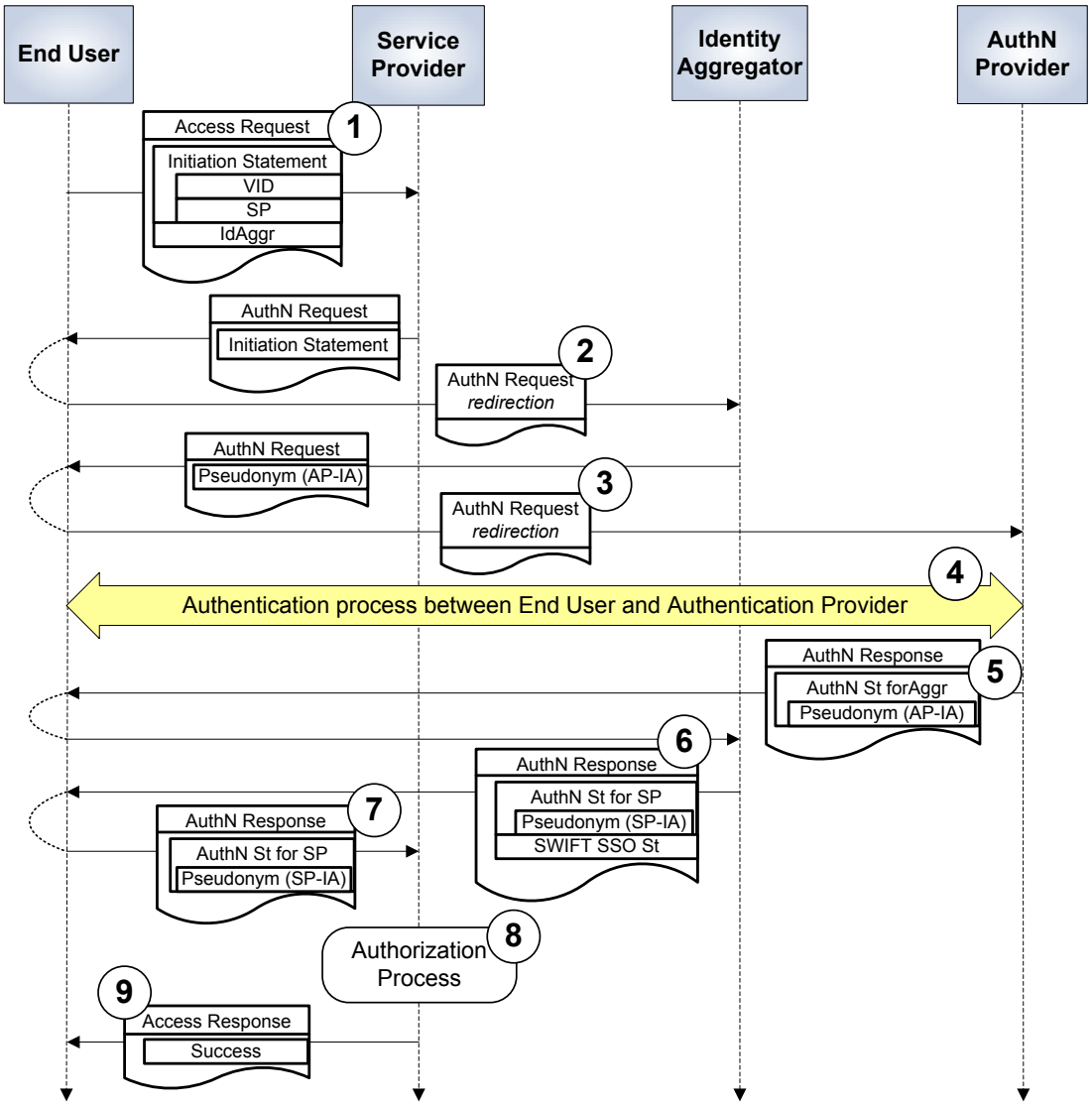


Figure 3.7: Message flow for the initial web authentication.

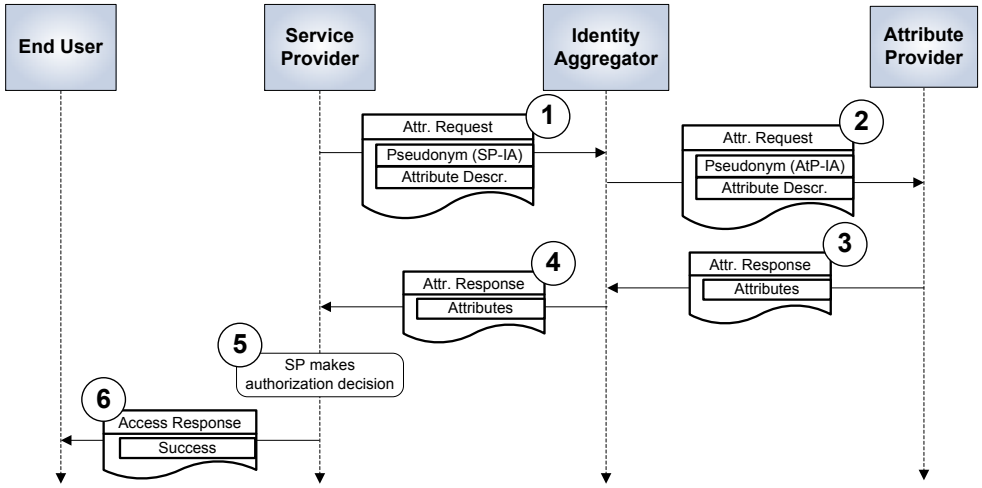


Figure 3.8: Attribute retrieval and authorization use case.

process. For example, a SP might require Alice to be over a determined age, to be from a specific organization or to have a valid credit card number. That is, the SP can impose certain restrictions on Alice's identity information prior to granting her access to the service. The SP can retrieve this information, in the form of attributes, from the IdAgg. These attributes are requested using the pseudonym received during the authentication process. Nevertheless, Alice could define access policies to decide which SP is allowed to access what information. These policies are called Attribute Release Policies (ARP).

This use case, depicted in Figure 3.8, describes the interactions that should be performed among the different framework elements to allow the SP to obtain Alice's attributes based on her virtual identity information.

Once Alice has been authenticated, for example making use of one of the authentication processes described above, the SP can retrieve some of her attributes to decide whether she is authorized or not to access the service. In order to do that, the SP constructs an Attribute Request message containing the pseudonym that identifies Alice between the SP and the IdAgg, and the list of required attributes (1). When the IdAgg receives such a request, it looks for the virtual identity referenced by the indicated pseudonym and checks whether the requested attributes were aggregated by Alice into that identity. If so, the IdAgg verifies, based on the ARPs previously defined by Alice, whether the SP can access the requested attributes or not.

Thus, if the SP is allowed to access them, the IdAgg contacts the different AttPs that actually store the attributes in order to retrieve them (2). To do that, the IdAgg uses the pseudonym established between them during the enrollment process to reference

Alice. Each AttP provides the requested attributes to the IdAgg included in an Attribute Statement (3). Once the IdAgg has collected all the attributes requested by the SP, it generates a new Attribute Statement containing all the information and delivers it to the SP (4). When the SP receives all the attributes, it can perform the authorization process and determine whether the EU can access the service or not (5). In order to perform this authorization process, the SP is subdivided following the architecture defined for XACML, including its different entities (PEP, PDP, etc.) and the communication among them. These interactions are not described here since it is outside the scope of this document. Finally, the SP provides the service to the EU (6).

This use case accomplishes the identity aggregation , privacy management and authorization based on attributes objectives seen in Section 3.2.1, since a virtual identity is used to obtain the required attributes to perform the authorization process, while Alice's actual identity is not revealed to the SP.

3.2.4 Formal validation for SWIFT interfaces

This section establishes the design to be followed by a user client that supports the requirements of the SWIFT architecture, for which it will be necessary to move from the virtual identity concept to a concrete design that is implementable in a real system. We will use a formal language to define interfaces, messages and information interchanged, with the aim of validating the architecture, which supposes an important reference previous to a real implementation and guarantees a more robust, secure and reliablefinal software.

Formal notation for APIs and messages

This first section offers a brief description of the nomenclature used to describe the roles involved and different parameters used. The following section describes the information flow, from and to the user client, and use cases for the creation of a new VID and for the access to a web service (with and without the evidence). A more detailed explanation of this notation can be seen in the final description of the architecture [126] [127] [128].

In order to perform a functional validation of interfaces, messages and the information exchange, we have used a formal notation similar to the notation used in validation tools like AVISPA [129], which is intended for security protocol validation, so it fits perfectly in our validation objectives. A complete analysis of SWIFT architecture and its security goals can be seen in this paper [130]

Roles

- EU → User
- Up → User Profile
- IA → Identity Aggregator
- SP → Service Provider
- AuthnP → Authentication Provider
- AttrP → Attribute Provider

Parameter abbreviations

- VID → Virtual IDentifier
- N → Nonce (Random load)
- Pse → Pseudonym
- attr_ref → attribute reference
- attr_policies → Attribute access control policy.
- artifact → reference to SSO_St_{IA}
- Evid → Evidence that will be calculated based on the SSO Statement provided by the IA.

Cryptography nomenclature

- Message X is signed by rol Y.
  $\{X\}_{inv(pk(Y))}$
- Message X is encrypted by rol Y.
  $\{X\}_{pk(Y)}$

Statements

- Initiation Statement:

  $\texttt{Init\_St}_{\texttt{vid}} \rightarrow \{\texttt{(Up,IA),SP,Nu}\}_{\texttt{pk(IA)}}$

- SSO Statement:

  $\texttt{SSO\_St}_{\texttt{IA}} \rightarrow \{\texttt{(Up,IA), hash(AuthnP,IA,Pse,N}_{\texttt{a}}\texttt{)}\}_{\texttt{inv(pk(IA))}}$

- Single Sign-On Token:

  $\texttt{SSO\_token}_{\texttt{vid,sp}} \rightarrow \{\texttt{(Up, IA), SP, N}_{\texttt{u}}\texttt{, Evid}\}_{\texttt{pk (IA)}}$

- Authentication Request:

  $\texttt{Authn\_Request}_{\texttt{IA}} \rightarrow \{\texttt{IA, AuthnP, N}_{\texttt{a}}\}_{\texttt{inv(pk(IA))}}$

- Authentication Statement issued by the *AuthnP*:

  $\texttt{Authn\_St}_{\texttt{AuthnP}} \rightarrow \{\texttt{AuthnP, IA, Pse, N}_{\texttt{a}}\}_{\texttt{inv(pk(AuthnP))}}$

- Authentication Statement issued by the *AttP*:

  $\texttt{Authn\_St}_{\texttt{AttrP}} \rightarrow \{\texttt{AttrP, IA, Pse, N}_{\texttt{a}}\}_{\texttt{inv(pk(AttrP))}}$

- Authentication Statement generated by the *IdA*:

  $\texttt{Authn\_St}_{\texttt{IA}} \rightarrow \{\texttt{IA, SP, Pse}_{\texttt{ia-sp}}\texttt{, N}_{\texttt{sp}}\}_{\texttt{inv(pk(IA))}}$

Use case APIs

This section describes in detail the APIs defined for each architecture component, depending on the corresponding use case.

Virtual Identity Enrolment: The process of creating and registering a virtual identity is performed by the user in the Identity Aggregator (IA). As can be seen in Figure 3.9, first, the user contacts the IA by providing an identifier (or VID) for the virtual identity, which should not be in prior use and also the Authentication Provider (AuthnP) that she wants to use to authenticate her identity. So, she is redirected to the AuthnP to prove her identity and confirm the use of this associated to the virtual identity. A similar process will then be performed to bind new attributes. In this case, the user must indicate which attributes she wants to bind to the new identity and in which Attribute Provider (AttrP) they are located. To confirm her identity in the AttrP, the user must be redirected to it to authenticate with her corresponding credentials in that service. From that point on, and thanks to the trust relationship between the AttrP and the IA, the attributes can be

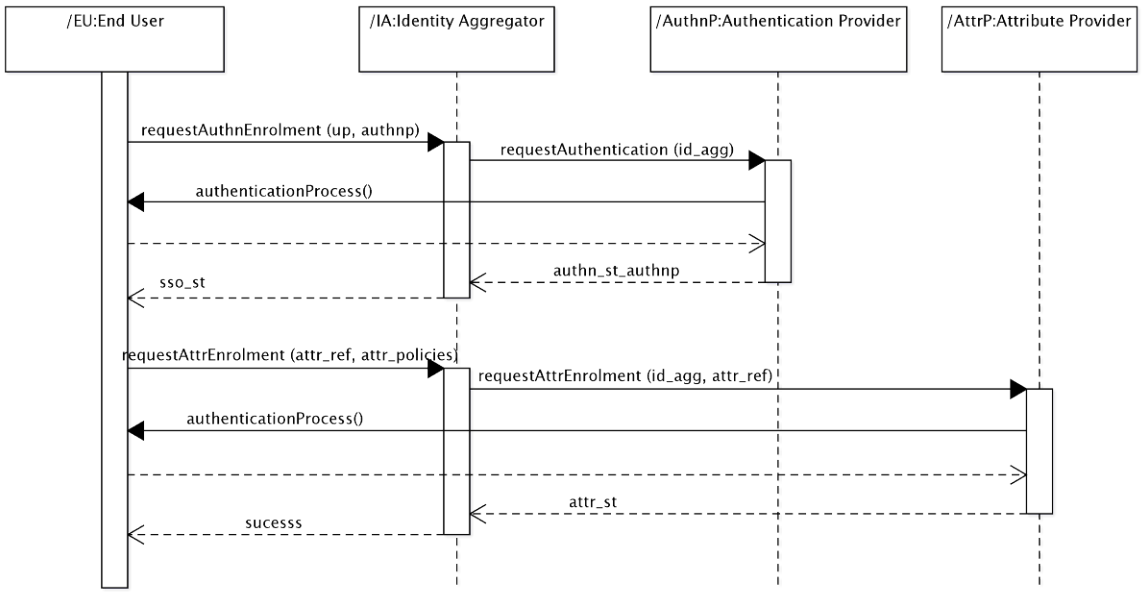


Figure 3.9: Enrolment sequence diagram.

retrieved by the IA without requiring explicit user authentication. In addition, the user can define specific security policies to control the access to attributes by the services, ensuring greater control of privacy.

1. EU → IA

   End User requests the registration of a new virtual identity named *up* from the IA, using it to authenticate the Authentication Provider named AuthnP. As a successful response to that registration request, she obtains a SSO_St statement signed by the IA.

   $$IA.requestAuthnEnrolment\ (up,\ AuthnP) = SSO\_St_{IA}$$

   $$IA.requestAuthnEnrolment\ (up,\ AuthnP) = \{(Up,IA),\ hash(AuthnP,\ IA,\ Pse,\ N_a)\}_{inv(\ pk(IA))}$$

2. IA → AuthnP

   The IA requests the AuthnP to authenticate the user. As a satisfactory answer, the IA gets an authentication statement from the AuthnP.

   $$AuthnP.requestAuthentication\ (Authn\_Request_{IA}) = Authn\_St_{AuthnP}$$

   $$AuthnP.requestAuthentication\ (\{IA,AuthnP,N_a\}_{inv(pk(IA))}) = \{AuthnP,\ IA,\ Pse,\ N_a\}_{inv(pk(AuthnP))}$$

3. EU ↔ AuthnP

An exchange takes place between the EU and the AuthnP to perform the user authentication process. How to perform this process is outside the scope of the framework.

4. EU → IA

   The user requests her IA to link a new attribute to her virtual identity. The request contains a reference or identifier to the attribute, and may include an access control policy to delimit later uses.

   `IA.requestAttrEnrolment (attr_ref, attr_policies) = success`

5. IA → AttrP

   The IA sends a request to the AttrP indicated by the user to determine whether or not to add the user attribute to her VID.

   $\texttt{AttrP.requestAttrEnrolment (Attribute\_Query}_{IA}\texttt{) = Attr\_St}_{AttrP}$

   $\texttt{AttrP.requestAttrEnrolment (\{IA, AttrP, attr\_ref\}}_{inv(pk(IA))}\texttt{) = \{AttrP,IA, Pse, N}_a\texttt{\}}_{inv(pk(AttrP))}$

6. EU ↔ AttrP

   There is an exchange between the EU and the AttrPto authenticate the user. The way to do this process is outside the scope of the framework.

Authentication initiated by the Service Provider (Web) In this use case, explained in Section 3.2.3, the user wants to access a resource or service using SWIFT framework. The EU sends an access request with the Initiation Statement to the SP. In order to validate the statement, the SP makes an authentication request to the IA that includes this. Because this type of statement does not include any evidence, the IA queries the AuthnP to perform the user authentication process. Once completed, AuthnP returns an authentication statement to the IA, generating a valid one for the SP. Finally, the latter gives the user access to the resource or service.

1. EU → SP

   The user requests web access to the SP using the *Initiation Statement*. After successful authentication, the user will have access to the resource or service.

   $\texttt{SP.requestInitialAccess (IA, Init\_st}_{vid}\texttt{) = Service || } \textit{failure}$

   $\texttt{SP.requestInitialAccess (IA, \{(Up, IA), SP, N}_u\texttt{\}}_{pk(IA)}\texttt{) = Service || } \textit{failure}$

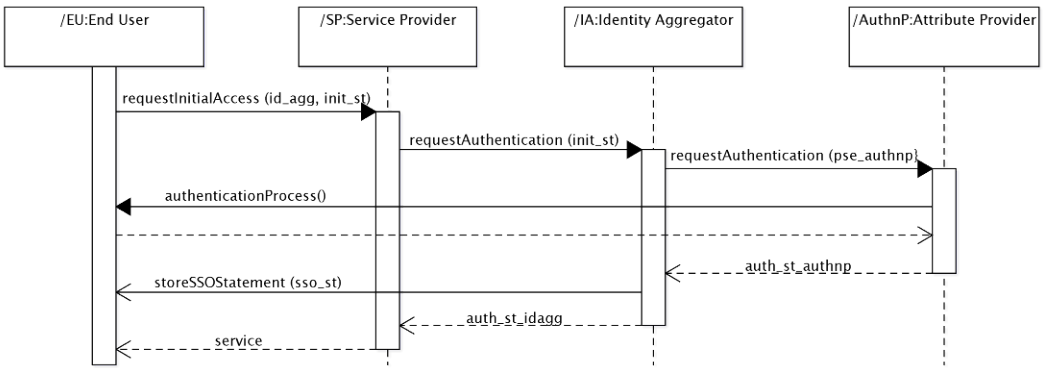


Figure 3.10: Sequence diagram for authentication initiated by the SP.

2. SP → IdAgg

   The SP requests the AI to authenticate the End User based on the *Initiation Statement*. If the authentication is successful, the SP receives an *Authentication Statement* from the IA.

   $IA.requestAuthentication\ ((Init\_st_{vid})_{inv\ pk(SP))} = Authn\_St_{IA}\ ||\ error\_response$

   $IA.requestAuthentication\ (\{\{(Up, IA), SP, N_u\}_{pk(IA)}\}_{inv(pk(SP))}) = \{IA, SP, Pse_{iasp}, N_{sp}\}_{inv(pk(IA))}\ ||\ error\_response$

3. IA → AuthnP

   The IA requests the AuthnP to authenticate the EU. As a satisfactory answer, the IA get an *Authentication Statement*.

   $AuthnP.requestAuthentication\ (Authn\_Request_{IA}) = Auth\_St_{AuthnP}\ ||\ error\_response$

   $AuthnP.requestAuthentication\ (\{\{IA, AuthnP, Pse_{ap}, N_a\}_{inv(pk(IA))}\}_{pk(AuthNP))} = \{AuthnP, IA, Pse_{IA\text{-}AuthnP}, N_a\}_{inv(pk(AuthNP))}\ ||\ error\_response$

4. IA → EU

   The IA asks the user to store the *SSO Statement* as proof of successful authentication.

   $EU.storeSSOStatement\ ((SSO\_st_{vid})_{inv(pk(IA))}) = success\_response\ ||\ error\_response$

   $EU.storeSSOStatement\ (\{(Up, IA), hash(AuthnP, IA, Pse_{ap}, N_a)\}_{inv(pk(IA))}) = success\_response\ ||\ error\_response$

5. EU ↔ AuthnP

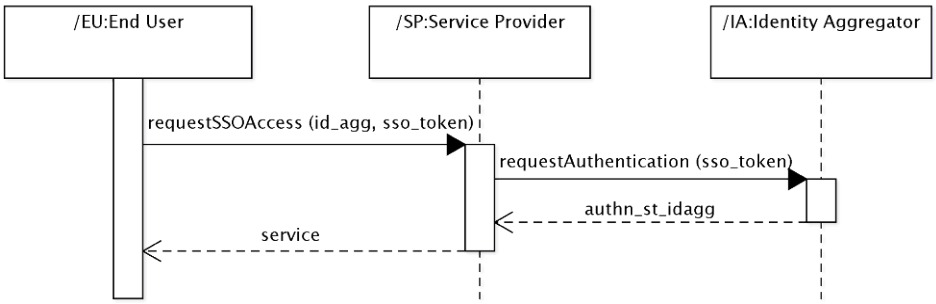


Figure 3.11: Sequence diagram for SSO in a web service

An exchange takes place between the End User and the AuthnP to perform the user authentication process. How to perform this process is beyond the scope of the framework.

SSO based access In this use case, previously described in Section 3.2.3 the user wants to access a resource or service using the Single Sing On (SSO) function. The EU sends an access request attaching the *SSO Token* to the SP. In order to validate the statement, the SP makes an authentication request to the IA including the statement. Because this type of statement includes an evidence, the IA checks the validity of the evidence and the *SSO Token* to authenticate the user. Once validated, the IA generates an *Authentication Statement* for the requesting SP that gives the user access to the resource or service.

1. EU → SP

   The user requests access to a resource to an SP based on the *SSO_ Token Statement*. After the process, the user can access the service.

   $SP.requestSSOAccess\ (IA,(SSO\_token_{sp}) = service\ ||\ failure$

   $SP.requestSSOAccess\ (IA,\{(Up, IA), SP, N_u, E_{vid}\}_{pk(IA)}) = service\ ||\ failure$

2. SP → IA

   The SP queries the IA about the validity of the *SSO_ Token Statement*. As a result, the IA returns an authentication statement to the SP.

   $IA.requestAuthentication\ (SSO\_token_{vid}) = Authn\_St_{IA}\ ||\ error\_response$

   $IA.requestAuthentication\ (\{\{(Up, IA), SP, N_u, Evid\}_{pk(IA)}\}_{inv(pk(SP))}) = \{Auth\_St_{IA}\}_{inv(pk(IA))}$ or $error\_response$

These sections have described in detail the main APIs for each component of the architecture, depending on the corresponding use case. This description will later serve as a basis for the correct implementation of the user client and the rest of the components of the architecture. A more complete description of the interfaces can be found in the final description of the SWIFT architecture [127].

3.2.5 Software development: the user agent software

The following subsections will show the set of characteristics that the user client must offer to cover the needs raised in the previous points when describing the life-cycle of the virtual identities and the general APIs of the architecture.

User interface features

The user client is the connection point of the user with the architecture. It must facilitate the virtual identity management to the user, and due to the internal complexity required for the management of sentences, identifiers, etc., make all this as transparent and simple as possible for the user.

In the graphical user interface, the client must display information clearly and intuitively about each of the user's virtual identities, such as the Virtual Identity Identifier (VID), usage status, or supplementary information that may be of interest.

In addition, the client must offer, through buttons or other means, how to perform actions on their virtual identities to use or create new ones, giving access to the internal functionality of the client, as will be seen later.

Internal client functionality

In addition to providing information to the user about her virtual identities and interacting with her, the client must offer an internal logic that supports the functions offered by the graphical interface for the complete operation of the architecture.

The client must be able to generate the *Initiation Stamentent* and *SSO Token statements* (including a valid evidence) to be sent to the SP in an access request to the resource or service. During the authentication process with these statements, the IA sends the user a statement called *SSO Statement*, which the client must receive and save for later generation, in future access, of the *SSO Tokens statements*.

In addition to offering the option through the graphical interface to the user, the client must be able to initiate the Enrollment process of new VIDs in the Identity Aggregator.

Because this process is performed outside the client, it has to provide the internal logic necessary to take the user to the IA.

Another important functionality is to provide persistence to users' VIDs, so that the user does not have to remember the identifiers of different VIDs, and to maintain the information necessary to generate *SSO Token Statements*.

Security Features

As seen previously, a user client that complies with framework requirements, it must be able to handle advanced cryptography and security functions, so management of digital certificates will be required for signature verification and the encryption of sentences.

Because the client manages user's sensitive private information that has to be protected, the user must have some mechanism to ensure the reliability of the tool that will be used to access the system. The proposed solution is that the client software be signed by the Identity Aggregator in which she trusts.

Another aspect to take into account when sending the sentences generated by the client is that they pass through unreliable entities such as service providers, so they must be protected to ensure that they are not used undesirably. For this, the SWIFT framework establishes that the sentences generated by the client must be encrypted for the Identity Aggregator to which they are addressed.

Finally, in the case of the client's implementing persistence, the information must be stored in such a way that only the client can access it. The proposed solution assumes the existence of a trusted infrastructure that allows validating the protected sentences that are exchanged between system entities. In most cases, identity federations define this trust by deploying public key infrastructures or PKIs. The interaction of the framework components with the services offered by this trust infrastructure is outside the scope of this work.

Technologies for the user interface

The web browser is the software that gives the user access to Internet resources. Indeed, it is the base for developing a user client that supports the architecture, so the chosen solution from a multitude of options that exist in the market must be safe, extensible and widely deployed. In principle, any browser should be supported by the infrastructure.

When providing the functionality of the client, the advantages of using a Java Applet versus the use of browser extensions or specific plugins are clear: multiplatform support, browser portability, a wide base of libraries to use in development, and the security of

the Java virtual machine. All these mean the solution chosen for user agent development should be done on Java in the form of an applet for the browser.

User Client final design

As a summary of what has been presented throughout these sections, the main component of User Agent Client is a Java Applet, which makes use of different libraries to support the architecture, running inside a Java virtual machine that needs a web browser. The applet will have access to the user's equipment in order to store and retrieve the VIDs (along with the corresponding information) between the different accesses to the services.

Design decisions to adapt the rest of components

Due to the specific design of virtual identities and the flow of interaction between the client and the architecture, it is necessary to apply certain changes in some components to adapt them to the proposed solution. The changes for each of them are described in a very simplified way below.

Service Provider For the adaptation of this component it will be necessary to modify the web pages that give access to the protected resources, in order to include the applet. In addition, the provider must receive and forward the client's SAML statements with the aim of including them in the requests to the Identity Aggregator. The remaining functionality will remain unchanged.

Identity Aggregator This component is specific to the proposed architecture and therefore, in its normal operation it expects to receive certain information from the SP that allows it to locate the user's virtual identity. Starting from this point, the IA must be adapted to retrieve the *Initiation Statement* or *SSO Token statement* from the SP's requests, and extract the information about the user needed to authenticate her. Depending on whether the evidence is included, and if the session is active, the IA must validate the user or put her in contact with her authentication provider to be authenticated. As a last step, the IA must be adapted to generate and send the *SSO Statement* to the user client, so that it stores it for later use.

Authentication Provider Although the Authentication Provider does not interact with the User client, its operation is affected by the specific design of the virtual identities. The only notable adaptation to the standard operation is to receive, within the SAML request

from the Identity Aggregator, the pseudonym that they share (IdA-AuthnP) to verify if the authenticated user is the one that actually corresponds with the VID presented to the Identity Aggregator.

Software design conclusions

Throughout this section, the design of the virtual identities, the user's client and their interactions with the rest of the architecture components has been established. For virtual identities their life cycle through the different use cases has been established, and a specific design has been set for the related statements over the SAML 2.0 language.

Once the design of the virtual identities is clear, the next step has been to establish the interactions between the architecture components taking into account: the client, the Identity Aggregator, the Service Provider and the Authentication Provider. Figure 3.12 shows a diagram of the final design of the various components.

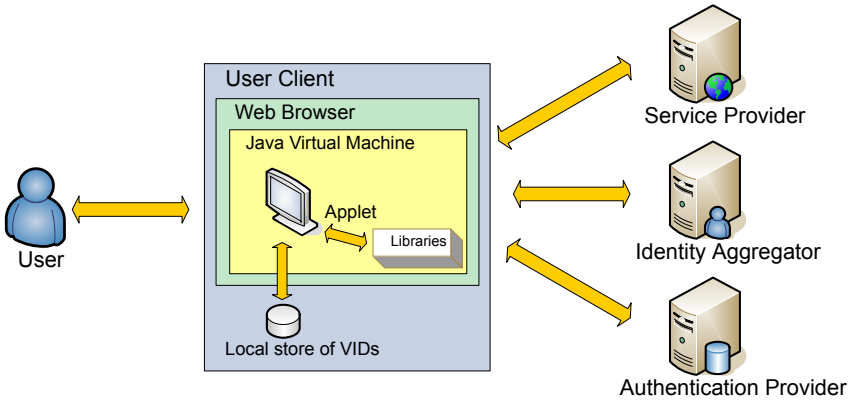


Figure 3.12: Detailed architecture schema

In the case of the client, the development of the functional prototype involves taking into account all the components of the architecture for its correct design. Section 3.2.5 states that the final client design based its operation on a JavaApplet that interacts through a web browser with the other components of the architecture. Internally, the Java Applet makes use of several libraries for the handling of the different technologies. On the other hand, the client must interact with the user's system to maintain information persistence between different uses.

The proposed design finally based on a Java applet offers clear advantages over other options. The decisions taken allow the resulting implementation to be used in most operating systems and browsers without having to make changes, so fulfilling one of the main objectives in identity management systems. In addition, thanks to the use of Java

and its great implementation in the field of communications security, the client can make use of already developed libraries that provide an additional guarantee of reliability.

3.2.6 Summary

Previous sections present a high-level vision of a framework offering solutions that enabling advanced features for identity management aspects to service providers and users. The possibility of aggregating end user accounts and thus simplify their management, the use of distributed policies that allow advanced policy definition, or the possibility of performing anonymous access to services, guaranteeing on the other hand that they are valid users that are actually authenticated and authorized, offer great possibilities for both, users and services. The possibility of, for instance, offering customized services depending on the end user attributes to anonymous users is a very rich, powerful and rare feature in current identity federations. The framework also allows for those organizations that require an authenticated network access (universities, research centres, companies, etc.), the association between the network authentication process and the access to higher-level services increases the usability of these.

All these concepts have being developed within the SWIFT project and offer a advance framework to keep working on other identity management aspects, like improved advanced access control mechanisms, identity delegation, service-bound accounting, etc.

3.3 Interoperability mechanisms for Trust Management: GEMBus

The number and variety of services that can be found on the Internet today are growing, and most of them need to be secured in order to not reveal information to those people not authorized to access it. Much of this information is related to private user data, so security takes on an important role when it deals with services accessing private information. There are many scenarios requiring the transparent integration of heterogeneous security services. This integration eases the application development, as well as simplifying deployment and providing a seamless user experience.

GEMBus is a framework that arises inside the National Research & Educational Network (NREN) community of Europe, within the GÉANT project. The research project started as part of the GN3 program (2009-2013) and continued in GN3+ (2013-2015), with the aim of addressing a key paradigm in the NREN innovative service environment:

seamless collaboration, moving beyond the current model of a community of collaborating network operators into an open cloud of resource services that users can freely compose to access network resources and other applications operating on top of the network. The scenario may include both services made available by network operators and services offered by the users themselves to other users, therefore making GEMBus both a facilitator and a consumer of cross-stratum optimization mechanisms.

The self-management of networked systems and services is a key challenge for the Future of Internet (FI) [131–133]. The Future Internet Assembly (FIA) [134] released the *Management and Service-aware Networking Architectures (MANA) for Future Internet* position paper [135] to support and guide the definition of management architectures for FI and the role of management operations within network services.

Developed as part of the GÉANT project [136] and in collaboration with user groups to identify and prototype use cases, the GEMBus framework aims to provide a new environment to enable users to create, compose, and consume service facilities on demand. It is intended to expand the current Service Oriented Architecture (SOA) model to produce the basic framework for a federated service bus as the foundation of future advanced network services for the European research and academic community.

As stated above, the main objective of the GEMBus framework is to provide mechanisms to define, discover, access, and combine network services, with the focus on covering security requirements required to offer access control and enforce security to users and services. Thus, the framework will be exposed to the application elements at the infrastructure-level and service-level. It is the basis for a federated, multi-domain service bus that will be able to integrate (compose) any network service, especially within the GÉANT infrastructure but open to any service-oriented infrastructure. Moreover, it allows flexible negotiation of service provision capabilities dynamically on a mutual basis, thus avoiding centralized service management as much as possible.

At the beginning of the research project, we studied the major use-cases for network service integration, as is described in [137, 138]. From the results of this study, we determined the essential mechanisms that should be provided by the framework, with the cross-layer service orientation and composition being the most important pitfalls to overcome. Finally, we put together all requirements and defined the boundaries of the GEMBus framework by providing a set of infrastructural services to be included in the platform. Thus, to achieve its objective, the common infrastructural services included in the framework, as illustrated in Figure 3.13, are:

- A distributed and federated registry for service descriptions to allow other services

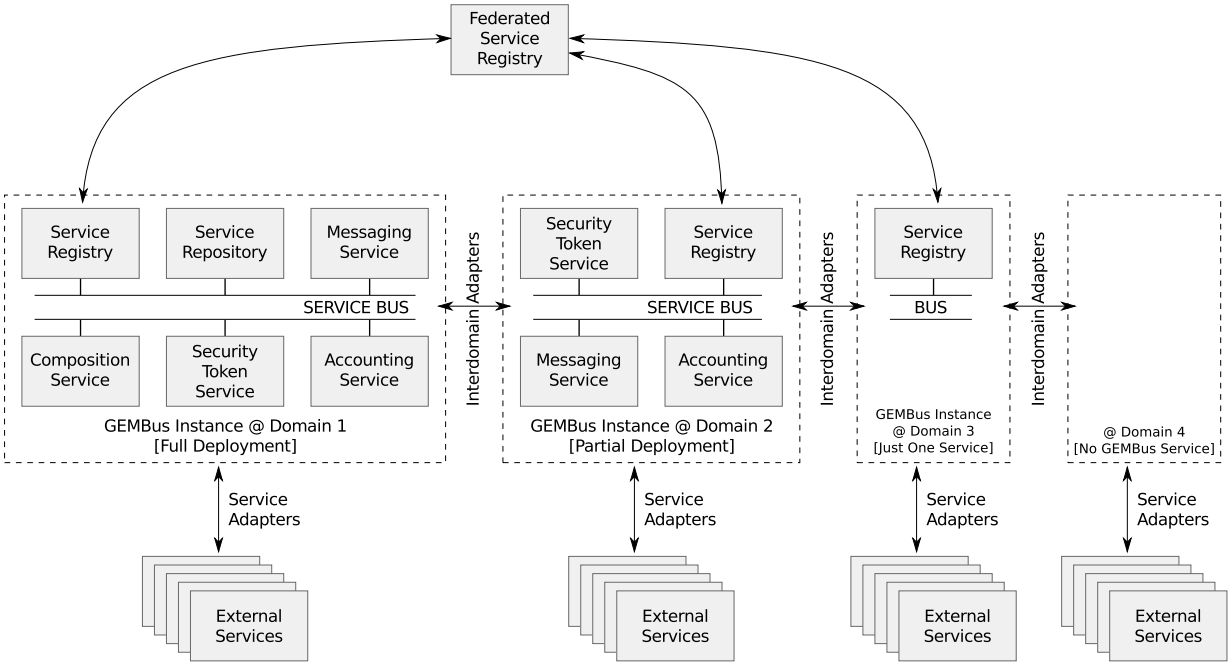


Figure 3.13: Overview of the GEMBus architecture.

to know where to find and access services.

- A messaging service that provides a flexible services interconnection functionality with high level of dynamicity.
- An accounting service for traceability and diagnostics through logging the desired message exchanges.
- A service repository to store service components to be easily found and deployed.
- A security service to provide/enforce access control and protect/enforce the rights of users and services over other services or the whole framework.
- A service composition engine to take different component services, either deployed locally or remotely, and thus build a new composite service.

The service bus concept used in the architecture is similar to the well-known Enterprise Service Bus (ESB) concept [139]. It is used as the basis to connect the services and to address the framework objectives. A service bus offers an effective method of integrating heterogeneous services, allowing their loose coupling, so gaining in flexibility and dynamism, as well as the simplification and standardization of the interfaces used by service providers and consumers.

The GEMBus framework provides the ability to compose services from the underlying virtualization systems to cover the requirements of the orchestration systems as defined in [135]. In fact, the GEMBus framework is fed by any interface that can be represented as a service in order to combine them to produce and deliver bigger services. Moreover, it can be placed in the middle of the current Cloud Computing infrastructures [140] to serve as Platform-as-a-Service (PaaS) technology and thus build PaaS services. In summary, it is a global middleware that *talks* in terms of services and produces combined functionality (composite services).

3.3.1 Trust enablers in Identity Federations

Traditional services are based on top of the SOAP protocol [45] because they are easy to use, their interfaces are defined by a contract (using Web Services Description Language (WSDL) [141]) and there are many development tools that help to build SOAP services. Moreover, many specifications were created in order to secure this type of services, such as WS-Security, WS-Trust, etc. (all of them members of the WS-* family of web service specifications). WS-Security incorporates security features in the header of a SOAP message and can be used in conjunction with other Web service extensions and higher-level application-specific protocols to accommodate a wide variety of security models and security technologies. WS-Trust, in turn, provides extensions to WS-Security and deals with the issuance, validation, renewing and cancellation of security tokens that are used to provide security using the mechanisms created by WS-Security.

In recent times, the services based on the REST protocol are gaining strength, because they are lightweight and easy to develop (specific tools are not needed) and widely supported by the growth of social networks. To secure REST services, one of the latest security protocols that has been created is OAuth. It is an authorization protocol and is oriented toward the interconnection and integration of service APIs used by Facebook and Twitter, the two major social networks. These social networks make use of the OAuth protocol to provide a way to manage the trust between services and users, and authorize clients for access to protected information.

Both OAuth and WS-Trust (along with WS-Security) are two of the main protocols used today to provide security functions to web services. However, different services using different security mechanisms need to be integrated in a transparent way. This integration is not trivial because different services may use different and incompatible security mechanisms. One of the most common use cases for such security services is found in the services making use of the OAuth protocol to provide a simple and flexible way

to authorize clients to access the protected information (resources). Due to the fact that different OAuth implementations normally use distinct types of authorization codes and access tokens, the integration between these implementations is not an easy task.

However, this heterogeneity can be tackled by leveraging on WS-Trust, which offers integration mechanisms between services which implement WS-* specifications. Thus, this work proposes a combined solution for this type of scenario. In this proposal, the integration among services that use different protocols (SOAP and REST) and different security mechanisms is achieved through the integration of the mechanisms proposed by OAuth and WS-Trust, which enables advanced trust management through heterogeneous security technologies. Also, by integrating these mechanisms it is possible to reduce the complexity supported by the OAuth Authorization Server (AS), so easing the interoperability by delegating the issuance and validation processes to the WS-Trust mechanisms which are more mature and are able to deal with a wide range of security tokens. In addition, this work also proposes a solution for the integration of WS-Trust clients which intend to use OAuth protected resources. It closes the cycle and opens a wide range of authorization possibilities to current and future web services.

3.3.2 Extended State of the Art

This section extends the background information provided in Chapter 2 about WS-* family (Sect 2.2.2) and OAuth (Sect. 2.2.4) protocols with the aim of going in depth into specific details that will be needed to define, design and implement the proposed solution.

WS-Trust

As seen in Section 2.2.2, WS-Trust [48] is a WS-* specification and OASIS standard that provides extensions to WS-Security, specifically dealing with issuing, renewing and validating security tokens, as well as how to establish, assess (the presence of) and broker trust relationships between participants in a secure message exchange. WS-Trust defines the concept of the Security Token Service (STS) as a web service that issues security tokens as defined in the WS-Security specification. Besides, the standard specifies the formats of the messages used to request security tokens and the responses to those messages and the mechanisms for a secure key exchange.

A STS is a Web service that issues, renews and validates security tokens based on evidence that it trusts. These security tokens are defined in the WS-Security specification. To communicate trust, a service requires a proof such as a signature, or the proof of knowledge of a security token (or a set of them). The token generation process can be

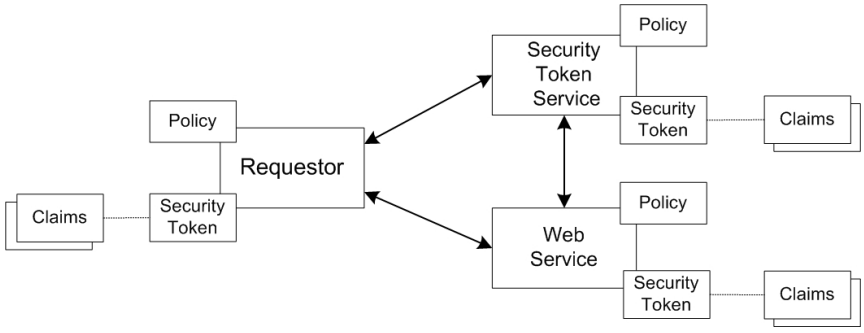


Figure 3.14: WS-Trust Security Token Service scheme

carried out by the service itself or can rely on a separate STS. In this case, the separate service issues the security token with its own trust statement.

Figure 3.14 shows the three main entities in the specification. The requester -Requestor- represents a service or client that wants access to a Web Service. This Web Service may require that the incoming message proves a set of claims. If the requester does not have the necessary token to prove the claims required by the service, it can contact appropriate authorities and request the information needed with the proper claims. This authority is STS, which may require its own set of claims for authenticating and authorizing the request from the requestor.

The WS-Trust specification defines the formats of the messages to request and reply for the issuance, the validation and the renewal of security tokens. The issuance process makes use of the `RequestSequrityToken` request that includes details about the type of the token required, the type of request and other information that might be needed, such as user credentials or other attributes. The reply provides the new security token and other related information, such as the type and the request context.

In the case of the renewal binding, the request is similar to the issuance one. The main difference is the required inclusion of `RenewTarget` element that identifies the token being renewed and may contain a reference to the token or, directly, the token to be renewed. In response to this request, the consumer receives the fresh token from the STS.

The validation binding shows a similar pattern to the previous messages. In this case, it is necessary to include `ValidateTarget` element that typically contains the reference to the token or could also carry the token to be validated directly. After STS has evaluated the token, it replies to the consumer with the token status, including a specific code (valid or invalid) and, optionally, the reason as a human-readable text.

OAuth 2.0 Protocol

OAuth 2.0 is an authorization protocol that enables a third-party application to obtain limited access to a service, either on behalf of a resource owner or by allowing the third-party application to obtain access on its own behalf.

In the traditional client-server authentication model, the client requests an access restricted resource (protected resource) from the server by authenticating with the server using the resource owner's credentials. In order to provide third-party applications access to restricted resources, the resource owner shares its credentials with the third-party. This creates several problems and limitations. Third-party applications are required to store the resource owner's credentials for future use, so the compromise of any third-party application results in a compromise of the end-user's password and all the data protected by that password. Resource owners cannot revoke access to an individual third-party without revoking access to all third-parties, and must do so by changing their password. Besides, third-party applications gain overly broad access to the resource owner's protected resources, leaving resource owners without any ability to restrict duration or access to a limited subset of resources.

OAuth 2.0 protocol addresses these issues (and more) by introducing an authorization layer and separating the role of the client from that of the resource owner. In OAuth, the client requests access to resources controlled by the resource owner and hosted by the resource server, and is issued a different set of credentials than those of the resource owner. Instead of using the resource owner's credentials to access protected resources, the client obtains an access token (a string denoting a specific scope, lifetime, and other access attributes). Access tokens are issued to third-party clients by an authorization server with the approval of the resource owner. The client uses the access token to access the protected resources hosted by the resource server.

Roles

The OAuth protocol defines four roles. The resource owner, which is an entity capable of granting access to a protected resource. The client, which is an application making protected resource requests on behalf of the resource owner and with its authorization (it could be an application executed on a server, desktop or other devices). The authorization server (AS), which is the server issuing access tokens to the client after successfully authenticating the resource owner and obtaining authorization. And finally, the resource server (RS) is the server hosting the protected resources, capable of accepting and responding to protected resource requests using access tokens. These roles are used in

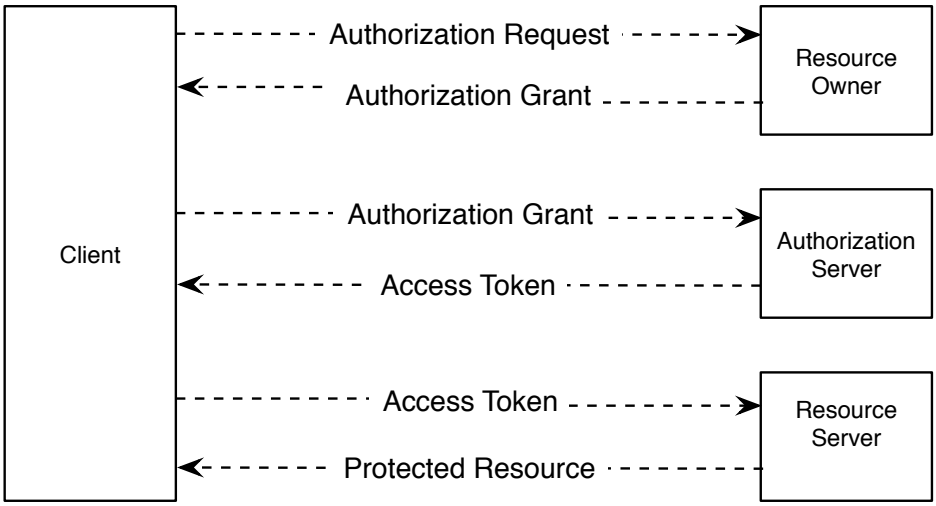


Figure 3.15: Abstract Protocol Flow

the OAuth architecture inside the flows depicted in Figure 3.15.

Architecture

The OAuth 2.0 architecture is very simple and its definition is based on the abstract protocol flow, as is depicted in Figure 3.15, which describes the interaction between the roles. As can be seen, the steps are as follows: first, the client requests and obtains authorization from the resource owner to access a protected resource. The client receives an authorization grant which is a credential representing the resource owner's authorization. Then, the client requests an access token by authenticating with the authorization server and presenting the authorization grant. When the authorization server receives the request, it authenticates the client and validates the authorization grant, and if it is valid, issues an access token to the client. The client then requests the protected resource from the resource server and authenticates by presenting the access token. The resource server receives the access token, validates it, and if it is valid, serves the request.

Access token request

To obtain an access token, the client makes a request to the token endpoint by adding the following parameters using the `application/x-www-form-urlencoded` format in the HTTP request entity-body:

- `grant_type` (required). The value MUST be set to "authorization_code".

- `code` (required). The authorization code received from the authorization server.
- `redirect_uri` (required, if the "redirect_uri" parameter was included in the authorization request)

For example, the client can make an HTTP request using TLS as shown in Table 3.1.

```
1 POST /token HTTP/1.1
2 Host: server.example.com
3 Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
4 Content-Type: application/x-www-form-urlencoded;charset=UTF-8
5
6 grant_type=authorization_code&code=SplxlOBeZQQYbYS6WxSbIA
7 &redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
```

Table 3.1: OAuth access token request example

Extension capabilities

For the definition of new extension grant types, OAuth takes into account the support of additional clients or the provision of a bridge between OAuth and other trusted frameworks. It also allows the definition of additional authentication mechanisms to be used by clients when interacting with the authorization server.

The OAuth 2.0 Assertion Profile [142] is an abstract extension to the specification that provides a general framework for the use of assertions as client credentials and/or authorization grants with OAuth 2.0. This profile provides a general framework for the use of assertions as client credentials and/or authorization grants with OAuth. It includes a generic mechanism for transporting assertions during interactions with a token endpoint, as wells as rules for the content and processing of those assertions. The main goal is to facilitate the use of OAuth in client-server integration scenarios where the end-user may not be present. OAuth 2.0 also provides additional profile documents for standard representations of these assertions in formats such as SAML and JWT.

An assertion can be used to request an access token when a client wishes to utilize an existing trust relationship. This can be done through the semantics of the assertion, and expressed to the authorization server through an extension authorization grant type. The processes of granting authorization and assertion acquisition are outside the scope of OAuth.

Below is the list of the most important fields that should carry an OAuth Authorization Grant, along with a brief description of each:

- `client_id` (optional). The client identifier.
- `grant_type` (required). The format of the assertion defined by the AS and expressed as absolute URI.
- `assertion` (required). The assertion used as an authorization grant.
- `scope` (optional). The request may contain a scope parameter, and it is expressed as a list of space-delimited strings.

Table 3.2 defines the use of assertions as authorization grants. In this case, the client must include the assertion using the following HTTP request parameters:

```
1 POST /token.oauth2 HTTP/1.1
2 Host: authz.example.net
3 Content-Type: application/x-www-form-urlencoded
4
5 grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Asaml2-
6 bearer&assertion=PEFzc2VydGlvbiBJc3N1ZUluc3RhbnQ9IjIwMTEtMDU
7 [...]aG5TdGF0ZW1lbnQ-PC9Bc3NlcnRpb24-
```

Table 3.2: SAML assertion as OAuth authorization grant

It should be noted that Bearer Token describes how to use bearer tokens in HTTP requests to access OAuth protected resources. Any party in possession of a bearer token can use it to get access to the associated resources.

3.3.3 OAuth 2.0 and WS-* integration

Today, there are more and more scenarios which are making use of OAuth 2.0 to provide a simple and flexible way to authorize clients in order to access protected resources. This is due to the many advantages OAuth provides, among which we highlight the following:

- the use of REST to make requests (lightweight, easy to implement, readable results...)
- the provision of a simple API which simplifies the creation of different clients. This is easier than dealing with large XMLs to request and obtain something

Different OAuth 2.0 implementations normally use different types of authorization grant and access tokens. The main complexity to reach the interoperability between OAuth implementations is to offer compatibility for these types of internal tokens.

The authorization grant and access tokens can be anything, for example, encrypted string, a SAML assertion [35], a X509 certificate, etc. So, the OAuth Authorization Service (AS) must be able to deal with the authorization grant types that it expects, in order to generate the access token if the authorization grant is valid.

By means of WS-* (Section 2.2.2), it is possible to reduce the complexity supported by the OAuth AS, so easing the interoperability through the delegation of the issuance and validation processes on the mechanisms introduced by WS-Trust (Section 3.3.2). The STS can deal with SAML2.0, X.509 and JWT tokens (using its extension capabilities) for validation and it can use different types of policy engines to perform the access control. Furthermore, it is able to generate different types of tokens such as SAML2.0 and JWT tokens and it can be extended to support more token types, such as OAuth access tokens.

On the other hand, WS-Trust arose as a standardization mechanism to offer security and interoperability between traditional Web Services (SOAP) that use heavyweight protocols such as SAML or X.509 certificates. WS-Trust is, in turn, a heavyweight protocol and does not seem to be suitable for the new scenarios which have appeared lately, such as social networks or cloud services.

However, the integration of scenarios which use OAuth 2.0 with the WST STS can be very helpful to extend the capabilities for authentication and authorization, and simplify the complexity of the OAuth Authorization Server. In the same manner, there are situations where a given service using WS-* for security needs to make use of OAuth protected resource, so the integration between OAuth 2.0 and WS-Trust must be done in both directions.

Scenario OAuth and the WST STS

Let us suppose a scenario in which a user tries to access a service which uses OAuth as authorization mechanism. The OAuth AS implementation only deals with a specific type of authorization grant, but the user may have a previous authorization grant, which can be of a different type to the one supported by this AS.

The question is clear, how can an access token be obtained to access the service using heterogeneous authorization grants?

It is in this type of situation that the WST STS can be very important. The main idea is to try to delegate the complexity supported by the OAuth AS in the Security Token Service described by WST so that the STS is responsible for authenticating, authorizing, generating and validating the tokens used.

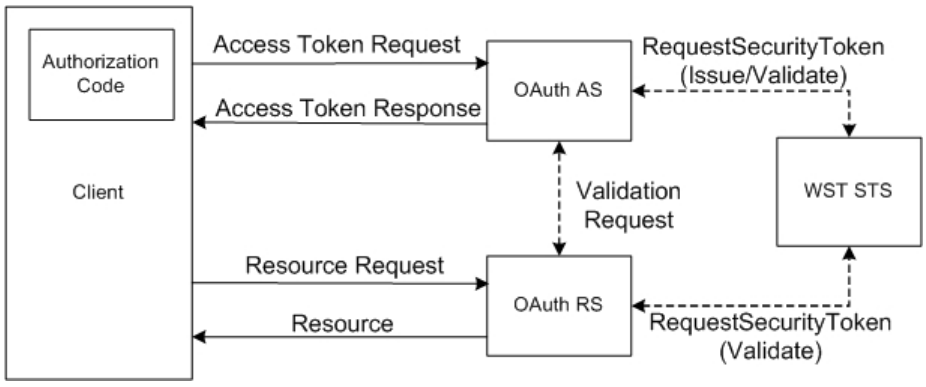


Figure 3.16: OAuth-STS Integration

Proposed solution Figure 3.16 illustrates the OAuth elements and the WST STS, and how a client, acting on behalf of a user using an authorization grant, can access a resource protected by a different OAuth implementation, that is to say, an implementation that is not able to deal with the authorization code supplied.

The operational flow of this proposed solution begins by obtaining an OAuth authorization grant based on the bearer token specification [77] (omitted in the figure). With it, the client, acting on behalf of a user, requests an access token to get access to the protected resource. The client sends the authorization grant to the Authorization Server along with other parameters (scope of the protected resource, client identifier...). At this point, the AS tries to validate the authorization grant.

Since the authorization grant type can be heterogeneous, the AS delegates to the STS the responsibility of validating the authorization grant and if it is valid, the STS will issue the requested access token. Once the access token has been obtained by the client, it will be used to access the protected resource.

The client sends the access token obtained to the Resource Server (RS). The access token must be validated by the RS so that it can return the protected resource. The RS can validate the access token by itself if it is able to deal with the type of the access token. If not, the RS can query the AS, which could either validate the access token by itself or, in the same manner as above, could delegate the process to the STS. Finally, if the access token is valid, the RS returns the protected resource.

As can be seen, it is sufficient for OAuth AS to be able to make WS-Trust requests (WS-Trust messages) to the STS and understand the WS-Trust responses. There is also the possibility that the RS can use the STS to validate the access tokens, but this would increase the complexity of the solution since the RS should be able to deal with WS-Trust

messages too. These decisions imply a trust relationship between OAuth AS/RS and STS.

To enable interoperability between OAuth AS and STS, the OAuth AS must be able to translate the OAuth REST requests to WS-Trust SOAP requests, that is to say, it must be able to create a WS-Trust message for issuance from the REST request received.

As shown in Section 3.3.2 on OAuth, the *Access Token Request* contains the following fields: `gran_type`, `code` and `redirect_uri`. The only field which is needed to obtain the access token from the STS is the `code` (authorization grant), so the OAuth AS must add the authorization code in a WS-Trust message and send it to the STS. The following example shows the mapping between OAuth and WS-Trust requests to obtain an access token using a bearer token received by the AS:

```
POST /token.oauth2 HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded;charset=UTF-8

grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Asaml2-bearer
&assertion=PEFzc2VydGlvbiBJc3N1ZUluc3RhbnQ9IjIwMTEtMDU
[...omitted for brevity...]aG5TdGF0ZW1lbnQ-PC9Bc3NlcnRpb24-
&scope=http%3A%2F%2Fwww.test-service.com

<RequestSecurityToken>
  <TokenType>urn:ietf:wg:oauth:2.0:oob</TokenType>
  <RequestType>
    http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue
  </RequestType>
  <AppliesTo>
    <EndpointReference>
      <Address>http://www.test-service.com</Address>
    </EndpointReference>
  </AppliesTo>
  ... assertion ...
</RequestSecurityToken>
```

Table 3.3: Mapping between OAuth access token request and WS-Trust issuance request

The `RequestType` element specifies that the request is an issuance request and its value is fixed, in this case http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue

According to the WS-Security specification, if the authorization grant (assertion) is a SAML assertion, then it is added to the WS-Trust message as another XML element, but if it is a encoded string (such as base64 encoded string) then it is added inside a `BinarySecurityToken` element of WS-Security, indicating the value and encryption type of the

code so that the STS can handle it.

If the scope parameter exists in the client request, then it is mapped as `Address` element of `EndpointReference` inside the `AppliesTo` element and, in this case, the `TokenType` element is not mandatory. This element is used to decide the type of access token to return. But if no scope is specified in the client request, then the OAuth AS must set the type of the access token it wants in the `TokenType` element of WS-Trust request. In this example `urn:ietf:wg:oauth:2.0:oob` indicates that the AS is requesting an OAuth token.

When the STS receives the request and validates the authorization grant, it responds to the OAuth AS with a WS-Trust `RequestSecurityTokenResponse` message containing either the access token if everything went well or an error indicating why the authorization grant is not valid.

Finally, the OAuth AS parses the response and extracts the access token issued by the STS and sends it back to the client in an OAuth access token response.

The following example shows the mapping between WS-Trust response and OAuth access token response to return an access token to the client to be used in an OAuth RS. That is, to translate the WS-Trust response into a JSON structure

```
1  <RequestSecurityTokenResponse>
2    <TokenType>urn:ietf:wg:oauth:2.0:oob</TokenType>
3    <RequestType>
4      http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue
5    </RequestType>
6    <RequestedSecurityToken>
7      ... access-token ...
8    </RequestedSecurityToken>
9    <Lifetime>
10     <Created>2017-03-27T10:00:00.503Z</Created>
11     <Expires>2017-03-27T10:30:00.503Z</Expires>
12   </Lifetime>
13 </RequestSecurityTokenResponse>
```

Table 3.4: WS-Trust RequestSecurityToken Response example

As can be seen, the access token returned by the STS (since it is defined by WS-Trust) is contained in the `RequestedSecurityToken` element with its token type (normally it will be the same as the one requested except when an `AppliesTo` element is used to specify the endpoint of the resource which the token will be applied to) is contained in the `TokenType` element. The `Lifetime` element states the creation and expiration dates of the token, so the AS must parse these dates in order to obtain the expiration time in seconds used by OAuth protocol.

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache

{
  "access_token":"... access-token",
  "token_type":"urn:ietf:wg:oauth:2.0:oob",
  "expires_in":1800,
  "other_parameters":"other-parameters-values"
}
```

Table 3.5: WS-Trust mapping to OAuth access token response example

Other parameters supplied in the OAuth requests (e.g. scope) can be for the response by the AS. In this case, the OAuth access token response is a mix between WS-Trust response elements and OAuth access token request parameters.

The main advantages of this solution are the removal of part of the AS complexity and to allow different OAuth implementations to be compatible between them, so an access token obtained in an OAuth AS can be used to access protected resources from different OAuth Resource Servers.

Scenario WS-Trust STS and OAuth

In contrast, there is a scenario in which a WS-Trust client tries to access an OAuth protected resource. The client requests a valid token from the STS for the resource using WS-Trust mechanisms. It should be noted that the token issuance can be done either by the STS itself or this can request the token from a trusted OAuth AS.

The client with the OAuth token would access the resource based on the needed protocol required by the resource provider, because the resource access is out of the OAuth's scope.

Proposed solution Figure 3.17 shows the proposed solution for this scenario. The client, which uses WS-Trust mechanisms, has to access an OAuth protected resource. In this case, the WS-Trust client throws a `RequestSecurityToken` request to the STS indicating the needed token type and the target service.

The STS evaluates the request and returns the required token to the client using a `RequestSecurityTokenResponse` message of WS-Trust standard.

The client uses this token to obtain the protected resource from an OAuth RS. As seen in the previous scenario, this access token must be validated by the RS so that it can return

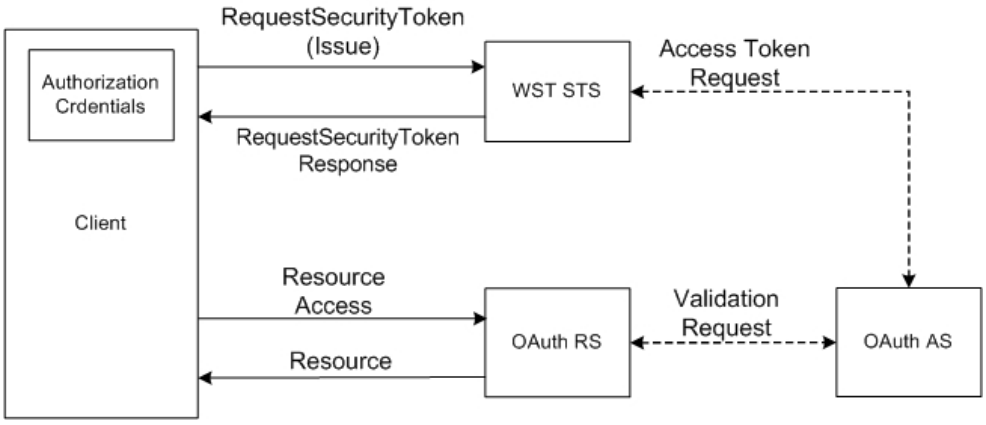


Figure 3.17: STS-OAuth Integration

the protected resource. The RS can validate the access token if it is able to deal with the type of the access token. If not, the RS can query the AS, which could either validate the access token or act as a proxy of the STS. Finally, if the access token is valid, the RS returns the protected resource to the client.

Internally, when the access token validation is delegated to the STS, the validation request made by the AS is as shown in Figure 3.6.

The `RequestType` element specifies that the request is a validation request and its value, in this case, is fixed at `http://docs.oasis-open.org/ws-sx/ws-trust/200512/Validate` and the access token is the token previously obtained. The `TokenType` is not needed, so it can be omitted.

The STS responds with a WS-Trust validation response, indicating the status of the token validation. The `Code` element, contained in the `Status`, indicates the result of the validation, where the possible values are `http://docs.oasis-open.org/ws-sx/ws-trust/200512/status/valid` and `http://docs.oasis-open.org/ws-sx/ws-trust/200512/status/invalid`.

Result summary

As stated in the introduction, the proposed mapping mechanisms between WS-Trust and OAuth protocol messages (SOAP and REST messages) allows full integration between these protocols. The resulting architecture offers an easy way to extend the security features of existing solutions based on OAuth or WS-Trust.

The integrated WS-* mechanisms allow the complexity of OAuth AS to be delegated to the STS. Furthermore, this takes advantage of the interoperability benefits offered by WS-Trust and the extension capabilities offered by the STS in order to improve the

```
<RequestSecurityToken>
  <TokenType>urn:ietf:wg:oauth:2.0:oob</TokenType>
  <RequestType>
    http://docs.oasis-open.org/ws-sx/ws-trust/200512/Validate
  </RequestType>
  <ValidateTarget>access_token</ValidateTarget>
</RequestSecurityToken>

<RequestSecurityTokenResponse>
  <TokenType>
    urn:oasis:names:tc:SAML:2.0:assertion
  </ns4:TokenType>
  <Status>
    <Code>
      http://docs.oasis-open.org/ws-sx/ws-trust/200512/status/valid
    </ns4:Code>
    <Reason>Valid token</ns4:Reason>
  </Status>
</RequestSecurityTokenResponse>
```

Table 3.6: WS-Trust Token Validation example

OAuth AS functionality and the range of token types accepted.

The integration in both directions has been achieved, so similarly, the STS can benefit of the interoperability with the newly emerged services in social network area and new web 2.0 applications.

3.3.4 Integrated design and implementation

After describing integration of OAuth and WS-* technologies, we decided to translate this to a practical (prototype) implementation and thus allow our approach to form part of a bigger solution. Therefore, we implemented the proposed approach for the GEMBus framework [137]. It is a complete framework to provide service composition in wide and federated scenarios, so it is a perfect candidate to benefit from the capabilities provided by our proposal. Also, as GEMBus is conceived for the research and education communities forming the GÉANT network, a European-wide research network, we find in it a large environment of already deployed and federated services that may benefit from our proposal. Furthermore, it can give us a wealth of useful scenarios to demonstrate the capabilities of our proposal.

GEMBus offers a Security Token Service (STS) which is responsible for managing

security credentials. Its design is based on the guidelines given by WS-Trust standard, which focuses on SOAP services. In the case of wanting to interact with other types of services such as REST services, it should make use of a solution, like the one given in the Section 3.3.3.

Below we describe the GEMBus architecture, the Security Token Service, the OAuth Authorization Service, and last, the resulting integrated solution with the proposed modifications to these elements in order to achieve interoperability.

The Security Token Service

The GEMBus Security Service must provide mechanisms to ensure security, privacy and simplicity for the communication that takes place within the GEMBus architecture. The *Security Token Service* (STS) is based on principles established by the WS-Security and WS-Trust specifications. As shown in section 2.2.2, the WS-Trust is a WS-* specification and OASIS standard that provides extensions to WS-Security, and it specifically deals with the issue, renewal and validation of security tokens, as well as how to establish, assess (the presence of) and broker trust relationships between participants in a secure message exchange.

Internal architecture

The GEMBus architecture defines its own Security Token Service, based on WS-Trust specifications, that offers all these security features. The GEMBus concept divides STS functionality into two different elements. First, the *Ticket Translation Service* (TTS) is responsible for generating valid tokens in the architecture according to credentials received from the consumer. This element includes the support of current authentication methods such as SAML or X.509 certificates, as well as extension mechanisms to incorporate others in the future. Second, token validation process is performed by the *Authorisation Service* (AS), which can also be associated to more complex authorization mechanisms that imply attribute request and check security policies.

Figure 3.18 illustrates a scenario in which the STS, extended with support for internal session tokens, is integrated in the GEMBus architecture. In this example, the consumer obtains an *identity token* (a SAML assertion, for example) from an identity infrastructure. Then it sends an authentication request to the STS using the identity token. The STS validates the consumer identity token and issues a *Security Token* (ST) to the consumer. With the new token, the consumer sends a request message to the provider that is intercepted by an element that extracts the ST and sends a token validation request to the

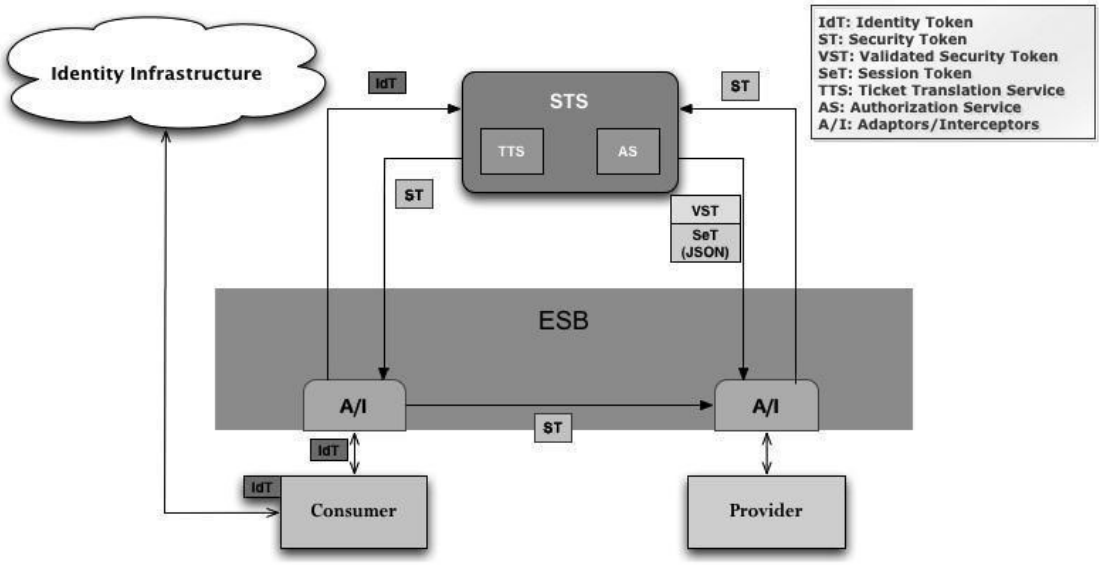


Figure 3.18: GEMBus interaction scheme

STS. The AS module validates the consumer token and issues a response with a validated security token, together with an optional *Session Token* (SeT). Finally, the interceptor passes the message to the provider, which processes the consumer request and sends a response message to the consumer. In addition to the flow described here, the service providers (SPs) deployed in GEMBus can validate the tokens by contacting the STS.

The STS standard concept is extended in the GEMBus architecture in order to provide additional functionalities. With the aim of improving the validation process, the STS is able to request attributes from external entities such as *Attribute Authorities* and *Identity Providers*. This new information could be used alongside that of the client in order to take an authorization decision from Policy Points using eXtensible Access Control Markup Language (XACML) [83].

STS Ticket Translation Service (TTS)

The *Ticket Translation Service* (TTS) is responsible for issuing, renewing and converting security tokens, responding to consumer requests for services that require it. These operations can only be done by the TTS, unlike security token validation, which can be done either by the service itself or at the framework integration elements, such as interceptors, message routers or binding components.

The main TTS operations are the issuance of new security tokens based on user's identity credentials, the token renewal and the conversion of one security token type to

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:sts="http://sts.wstrust.security.gembus.geant.net/">
  <soapenv:Header/>
  <soapenv:Body>
    <ns4:RequestSecurityToken
      xmlns="http://www.w3.org/2005/08/addressing"
      xmlns:ns2="http://schemas.xmlsoap.org/ws/2004/09/policy"
      xmlns:ns3="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
      xmlns:ns4="http://docs.oasis-open.org/ws-sx/ws-trust/200512/"
      xmlns:ns5="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <ns4:RequestType>http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue</ns4:RequestType>
      <ns4:TokenType>urn:geant:gembus:security:token:1.1:gemtoken</ns4:TokenType>
      <ns2:AppliesTo>
        <EndpointReference>
          <Address>www.sp2-gn3.org</Address>
        </EndpointReference>
      </ns2:AppliesTo>
      <ns4:Lifetime>
        <ns3:Created>2017-02-20T12:00:00.574Z</ns3:Created>
        <ns3:Expires>2017-02-20T14:30:00.574Z</ns3:Expires>
      </ns4:Lifetime>
      <ns5:BinarySecurityToken ValueType="X509v3" EncodingType="Base64Binary">
        MIIC...b21lLVN0YXRlMRAwDgYDVQFJssZ0=
      </ns5:BinarySecurityToken>
    </ns4:RequestSecurityToken>
  </soapenv:Body>
</soapenv:Envelope>
```

Table 3.7: Request Security Token (issuance) example

another compatible type. Table 3.7 shows an *Request Security Token* for the issuance of a new token. The most relevant fields are: `RequestType`, `TokenType`, the `AppliesTo` and the `BinarySecurityToken` that contains the identity credentials used by the STS to authenticate the user.

The TTS operation is as follows:

1. The consumer obtains an identity token (SAML Assertion, proof of access to a private key, etc.) from an identity infrastructure.

2. The consumer sends a request for issuance, renewal or conversion to the TTS using either the Identity Token (issuance) or a Security Token (renewal or conversion).

3. The STS validates the consumer's token (using security policies) and sends a security token to the consumer.

STS Authorization Service (AS)

The AS is responsible for supporting the token validation functions, responding to requests for validating tokens of consumers and services that require it.

The token validation process can be performed by the STS itself or act as a proxy, redirecting the validation process to the external service that generated it. For external validation, the Authorization Service consults an external service or IdP, and forwards the response to the STS consumer. When the Authorization Service itself performs validation, the process must verify the information contained in the token, checking the issuer, issue and expiration date, signatures, etc. In addition to the token, the Authorization Service can perform a more complex authorization process, retrieving attributes related to the token subject and consulting a Policy Decision Point (PDP) for authorization decisions.

Tokens

The GEMBus TTS supports transformations between different token formats, according to service descriptions as stored in the *GEMBus Registry*. Appropriate profile definitions describe these formats. Nevertheless, the canonical *GEMBus Security Token* (applicable by default in all GEMBus-supported exchanges) is the relayed-trust SAML assertion originally defined within the GN2 project [143] to provide identity information in scenarios where a service is acting on behalf of a user identified through an identity federation.

The WS-Security specification allows a variety of signature formats, encryptions algorithms and multiple trust domains. It is open to various security token models, such as X.509 certificates, userid/password pairs, SAML assertions and custom-defined tokens, as seen in Section 2.2.2.

JSON Web Token (JWT) [144] defines a compact token format intended for space constrained environments such as HTTP Authorization headers and URI query parameters that can encode claims transferred between two parties. The claims in a JWT are encoded as a JSON object [145] which is then optionally digitally signed and transmitted *base64url* encoded. JSON is a lightweight, text-based, language-independent data interchange format that defines a small set of formatting rules for the portable representation of structured data.

WS-Security does not define a specific profile for this kind of tokens. In order to make the use of JWT possible, the GEMBus STS uses the extension capabilities of the WS-Security based on the *Binary security tokens* element. It is used to embed X.509

certificates and Kerberos tickets, as well as other non-XML formats that require a special encoding format for inclusion. The `<wsse:BinarySecurityToken>` element defines two attributes that are used to interpret it. The `ValueType` attribute indicates what the security token is and the `EncodingType` that tells how the security token is encoded. An example of the internal GEMBus Token is shown in Figure 3.8.

```
1 <wsse:BinarySecurityToken
2   EncodingType="Base64Binary"
3   ValueType="urn:geant:gembus:security:token:1.1:gemtoken"
4   ...
5 </wsse:BinarySecurityToken>
```

Table 3.8: Internal GEMBus Token example

OAuth Authorization Service

The OAuth Authorization Service is the entity in charge of authenticating the credentials of users and clients, validating the authorization grants and generating the access tokens. The AS selected for this demonstration was developed by RedIRIS Institution as a testbed to allow the development of prototypes and interoperability models for future use.

Its core makes use of an OAuth library implemented following the 14th draft version of the standard and offers access to its functionality through a REST interface.

Integrated architecture

The proposed architecture that allows the integration between WS-* and OAuth systems is composed of four main entities: the client of the system that needs a specific token, the WS-Trust STS and the OAuth AS that are responsible for providing the requested token, and finally the service which the user wants to access and that will consume the token. Figure 3.19 shows the relationships between these entities.

The architecture proposed by GEMBus is based on message exchanges performed by different services that can be connected in many ways. Since the ESB is the main integration mechanism provided by GEMBus, and it can also act as a container, it is possible to develop and deploy a service directly on the bus. GEMBus offers, furthermore, integration capabilities, such as interceptors, message routers and binding components. Whether deployed inside the bus or running as an external service, the STS can be used directly through a SOAP interface.

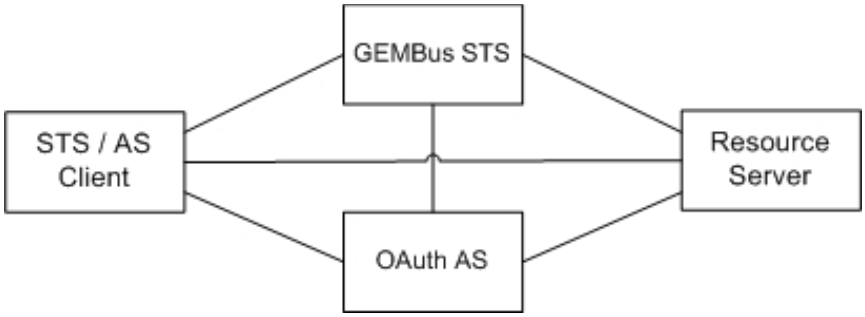


Figure 3.19: Integration relationship diagram

The GEMBus framework has been developed over FUSE ESB, adding to it the complete GEMBus ecosystem of services to enrich the original FUSE platform functionality. FUSE ESB is an open source integration platform with a pluggable architecture oriented to SOA solutions. The GEMBus modifications are focused in the STS, which has been extended with a module to allow the translation to REST OAuth message in order to use an OAuth AS for issuing tokens.

The OAuth Authorization Service development is based on the Draft 14 of OAuth specification. This element of the architecture is accessible through a REST interface which plays the role of the facade of the security system, but it has been adapted to support SOAP communications and to understand WS-Trust messages to interact with the STS.

As an additional feature, the extensions made to both STS and OAuth AS would allow us to see the OAuth AS like the REST interface of the STS and vice versa, the STS like the SOAP interface of the OAuth AS, that is to say, each security service may use the interface of the other service to manage requests.

Table 3.9 shows a list of main software components that take part in the testbed. The table includes software versions and the role that they play. The table 3.10 shows the most relevant standards used and implemented in the testbed demo. The software used in the demo has been developed from the beginning with the aim of making it flexible and configurable through configuration files. This allows us to use the same software in both scenarios only by adapting the settings.

3.3.5 Deployment and Evaluation

As part of this work, we set up a functional prototype where the concepts described in this thesis are put into practice. Using this, the objective of this section is to evaluate the proposed architecture in two different scenarios, one for the OAuth accessing the STS

Element	Software	Version
ESB	FUSE ESB	4.3.1
SOA Framework	GÉANT GEMBus	1.0.0
WS-* Implementation	GEMBus STS	1.0.0
OAuth library	RedIRIS OAuth library	1.0.0
OAuth AS	RedIRIS OAuth AS	1.0.0

Table 3.9: Used software.

Standard	Version
WS-Security	1.1
WS-Trust	1.4
SAML	2.0
OAuth	Draft 14

Table 3.10: The used standards and their versions.

and other for the STS accessing OAuth. Thus, we aim to demonstrate the feasibility and validity of integration models.

To perform the evaluation we deployed the functional prototype in a testbed composed of a single machine with the GEMBus software (including the STS), the OAuth Authorisation Server, and the OAuth Resource Server. We group these elements into one single machine to focus the evaluation on the software part and not on the network. In practice, we do not consider the cost of network jumps because it is almost the same in both scenarios. Moreover, the equipment used to obtain the measurements is an Intel(R) Pentium(R) Core 2 Duo CPU 2400MHz, with 8GB of RAM, running Mac OS X Lion 10.7.4. It has enough resources to host all components without disturbing each other.

Using this testbed and components we performed 500 executions of the whole process (the token request for both scenarios), to obtain the time measurements of the different steps of each process. Therefore, each scenario has been decomposed into several steps in order to analyze the time employed by each flow element, as can be seen in detail in the following sections. First, we start with the execution of a standalone scenario where the GEMBus STS and OAuth solutions run separately without any integration. Then, we proceed with the execution of the proposed integrated architecture.

Scenario 0: Standalone Execution

In this section we analyse standalone scenarios, where the STS and OAuth are running without being integrated, to obtain the base case to compare the scenarios using the integrated solution proposed in this work. From them, we measured the time spent in

the reception and initial processing of the client requests, the time spent by the OAuth Authorization Server (AS) and GEMBus Security Token Service (STS) in the processing, and the time spent sending the results back to the client. With the results from this evaluation we can perform an objective analysis of the proposed approach.

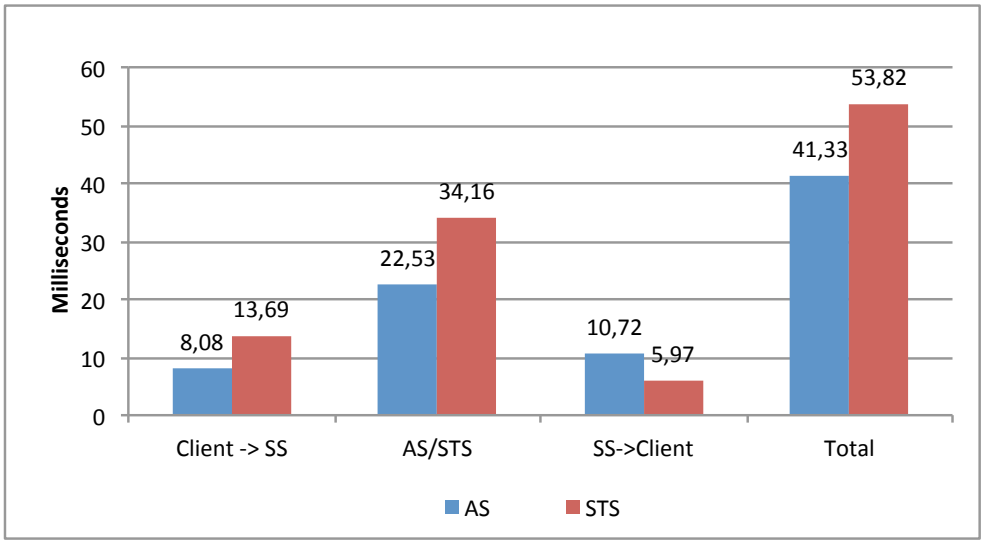


Figure 3.20: Means of execution times for standalone scenarios

Figure 3.20 shows the resulting times measured for the standalone scenarios where each security service (the OAuth AS or the GEMBus STS) produces its default token. As we can see in the figure, the times are relatively similar. The biggest difference is seen at the second step, where the specific security service has to check the validity of the credentials and then generate the new token. The main reason for this difference is that while the STS checks the signature of the credentials, the AS only checks some dates and fields of the request, because there is a previous trust relationship between the Client and the AS.

Scenario 1: OAuth AS accessing WST STS

In this section we analyze the scenario proposed in Section 3.3.3 in which an OAuth Client wants access to an OAuth protected resource that does not use the standard authorization token. For this reason, the OAuth Authorization Server (AS) delegates the generation of the token to the Security Token Service (STS). The interactions between the elements of this integration proposal are shown in Figure 3.16. The detailed steps of the scenario are:

1. The OAuth Client sends an OAuth Access Token request using the REST interface of

the OAuth AS including the required credentials, in this case a SAML 2.0 assertion. This step includes the message reception by the OAuth AS.

2. The OAuth AS validates the credential, and creates and sends a WS-Trust Request Security Token request to the GEMBus STS including the required credentials.

3. The GEMBus STS receives the request through its SOAP interface, validates the client credentials (including the signature) and generates a GEMBus Token in JWT format.

4. The GEMBus STS returns to OAuth AS a WS-Trust Request Security Token response that the AS has to process (convert) to obtain an OAuth Access Token response.

5. Finally, the AS returns the response to the Client, and the latter receives it for subsequent operations.

To obtain the time measurements in the integrated scenarios, we modified the components to log the time spent on the execution of each step. After the execution of the tests, we analyzed the log file to get their statistical data. Specifically, we calculated the mean of the time required to perform each step, the standard deviation, the confidence Interval at 95%, and the percent of the total time. These values are shown in Figure 3.21 and Table 3.11.

Step	Mean	Std.Dev.	C.I. 95%	Time %
Client → AS	6.46	4.79	0.42	10.32
AS: validation	10.23	5.76	0.50	16.35
STS: generation	33.63	11.48	1.01	53.73
AS: issuance	4.44	1.80	0.16	7.09
AS → Client	7.83	2.84	0.25	12.51
Total time	62.58	18.64	1.63	100.00

Table 3.11: Scenario 1 - Statistical data obtained from the measured times (in milliseconds)

To summarize, the time measures obtained show that the lengthiest step of the flow is the validation of the credentials and the token generation carried out by the STS in the third step ("STS: generation"). Thus, the STS needs 53% of the total time to complete these tasks. In the second place, the AS consumes 23% of the total time in processing the client request and asking the STS in the second step ("AS: validation"), and converting the STS response into the OAuth response. The remaining time is spent on the interactions with the Client.

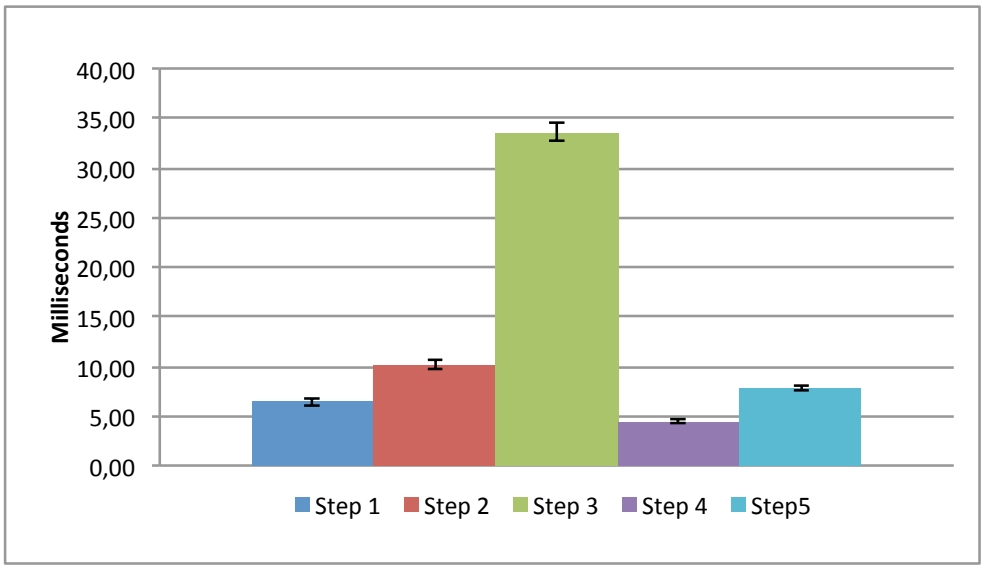


Figure 3.21: Scenario 1 - Means of execution times for the different steps (operations). Step 1 is the "Client → AS", step 2 is the "AS: validation", step 3 is the "STS: generation", and step 4 is the "AS → Client".

Scenario 2: WST STS accessing to OAuth AS

We have just studied the scenario where an OAuth client accesses an OAuth protected resource; in this section we proceed to study the scenario where a WS-Trust client accesses an OAuth protected resource. This scenario is discussed in Section 3.3.3 and Figure 3.17 shows the interactions between the elements of the integrated architecture. This comprises the following steps:

1. The Client sends a WS-Trust Request Security Token request over SOAP to the GEMBus STS, including the required credentials, in this case a SAML 2.0 assertion. This step also includes the message reception by the STS and its unmarshalled process to discover the type of required token.

2. The GEMBus STS validates the credentials (including the signature) and creates an OAuth Access Token Request for the OAuth AS.

3. The OAuth AS receives the request through its REST interface, validates client credentials and generates an OAuth Access Token in JWT format.

4. The AS returns an OAuth Access Token response to the GEMBus STS. It is processed by the STS and converted into a WS-Trust Request Security Token response.

5. Finally, the STS returns the response to the Client, and the latter receives and keeps it for subsequent operations.

As we did in the previous section, to obtain the time measurements in the integrated scenarios we modified the components to log the time spent on the execution of each step. Then, we analyzed the log file of the application to extract statistical data from the measurements. Again we specifically calculated the mean of the time required to perform each step, its standard deviation, the confidence Interval at 95%, and the percent of the total time. These values are shown in Figure 3.22 and Table 3.12.

Step	Mean	Std.Dev.	C.I. 95%	Time %
Client → STS	11.52	11.17	0.98	17.88
STS: validation	24.25	11.46	1.00	37.64
AS: generation	20.83	9.72	0.85	32.32
STS: issuance	2.34	2.01	0.18	3.63
STS → Client	5.50	4.81	0.42	8.53
Total time	64.44	26.89	2.36	100.00

Table 3.12: Scenario 2 - Statistical data obtained from the measured times (in milliseconds)

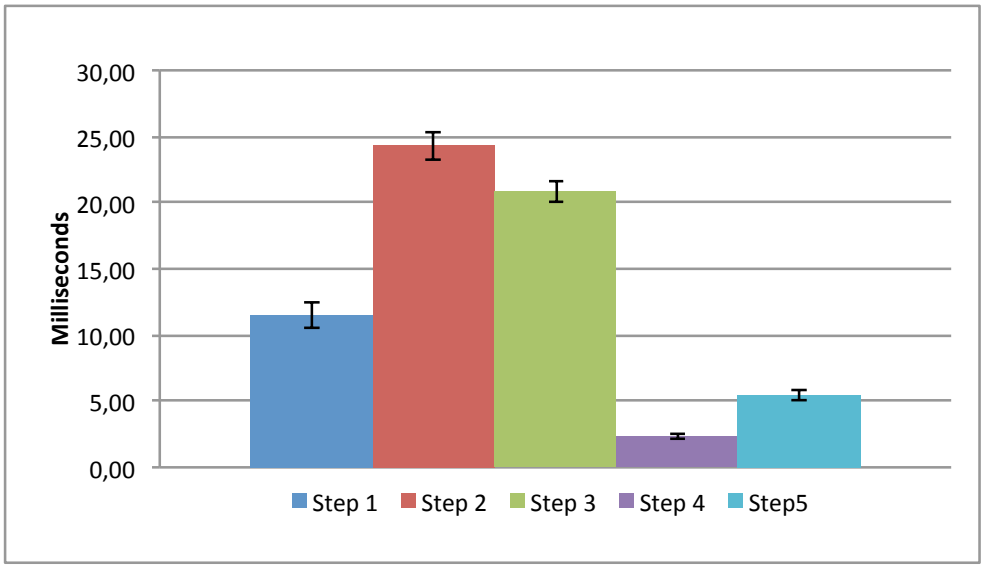


Figure 3.22: Scenario 2 - Means of execution times for the different steps (operations). Step 1 is the "Client → STS", step 2 is the "STS: validation", step 3 is the "AS: generation", and step 4 is the "STS → Client".

For this scenario, the time measures obtained show that the lengthiest step of the flow is the validation of the credentials and the token generation carried out by the STS in

the second step ("STS: validation"). Thus, the STS needs 37.64% of the total time to complete these tasks. In the second place, the AS consumes 32.32% of the total time when generating the OAuth Access Token ("AS: generation"). The fourth step ("STS: issuance") takes only 3.63% of the total time. Finally, the remaining time is spent on the interactions with the Client.

Discussion

In this section we discuss the results obtained in the evaluations performed in the two different scenarios and compare them with the results obtained in the standalone scenarios.

We discussed above that the steps that take a long time are, for all scenarios, those that validate and generate tokens. Comparing these times with the standalone results, we can effectively see that they are very similar. The generation step of the first scenario takes 33.63 milliseconds (ms) and the STS in the standalone scenario takes 34.16 ms. For the OAuth part, the second scenario takes 20.83 ms and the standalone scenario takes 22.53 ms. For the total time, we can see that, on the one hand, that the first scenario takes 62.58 ms, while the standalone OAuth solution takes 41.33 ms. This small overhead is clearly introduced by the extra STS generation step, so the cost of the added functionality is feasible. On the other hand, the second scenario takes 64.44 ms while the standalone STS solution takes 53.82 ms. Here, the difference is smaller and is only half the time spent by the OAuth AS generation step. This demonstrates a small overhead for the gains obtained from the integrated solution.

When comparing the two integrated scenarios with each other, from the total time measurements in both scenarios we can see that they are very similar. However, the time distribution is more homogeneous among the processing steps of the second scenario. The STS in the second scenario consumes more time than the OAuth AS, mainly due to the credential signature verification. Even so, the AS gains more prominence in the second scenario due to the validation and generation of new tokens. The first step consumes more time in the second scenario than the first because WS-Trust messages are more complex than OAuth messages.

In summary, as we mentioned, the total times of the integrated scenarios are slightly higher than the standalone scenarios. However, relating the results to overhead percentage, the integrated solution takes 56% more time than the standalone case for the OAuth operations and 16% more time than the standalone case for the STS operations. Although this overhead may seem large, in terms of absolute cost, taking into account that these operations are only run during authentication, the overhead is affordable and thus we

demonstrate the feasibility of the proposed solution. Anyway, the time overhead could be reduced with the optimization of the code but, even so, the benefits of the proposed integration of these two protocols for improving federation and interoperability are, in our opinion, much greater.

3.3.6 Summary

These sections present the current scenarios for authorization found in the context of web services and new platforms such as social networks. There are a variety of solutions to integrate technologies, but new security protocols arise faster than the integration mechanisms. New scenarios need lightweight protocols that ease the development of new services, as well as the integration among them, to allow resource sharing. As discussed throughout these sections, OAuth is becoming the *de facto* standard for this authorization task.

That said, this work also presents a route through different integration mechanisms for security protocols, focusing analysis on the WS-* specification family. Moreover, the GEMBus project has chosen this family of specifications to give security mechanisms to its framework, since it is specially oriented to the interoperability of web services. At this point, it is notable that both WS-Security along with WS-Trust and OAuth offer powerful extension mechanisms to be adapted to new security protocols.

After introducing the authorization and general security technologies used within web services mechanisms, we present two different scenarios to set out the integration from different points of view. These scenarios outline the interoperability problem between services and resource providers that implement different security protocols. As a possible solution, we propose the necessary integration and translation mechanisms to allow the interconnection of heterogeneous services from both the viewpoint of a WS-Trust service which wants to consume OAuth resources as well as when an OAuth architecture wants to integrate with a WS-Trust architecture to take advantage of the Security Token Service (STS).

On the one hand, the use of these integration mechanisms allows OAuth architectures to reduce the complexity of their authorization server by delegating part of their functions to the STS. On the other, the extension of WS-Security to give support to OAuth tokens and the mapping made between OAuth and WS-Trust messages allow OAuth resources to be consumed.

We evaluate the proposed architecture in two different scenarios to proof its feasibility and viability, thus obtaining a performance profile of the execution process. We

demonstrate that, in terms of absolute cost, the overhead introduced by the integrated solution is affordable and thus we demonstrate the feasibility of the proposal. However, future optimizations should be taken into account in order to reduce the overhead when instantiating the solution in production environments.

3.4 Summary

IdM is becoming more and more important every day. Users need a way to centralize the management of their identity information, such as simplifying the access to services with mechanisms like Single Sign-On. While organizations need a means of obtaining reliable information about users of their services, users are more worried about their privacy how their information is treated, what information is provided to what entity, and how privacy is assured in general.

SWIFT architecture offers a wide variety of advanced services focus on improve the identity management capabilities, enabling high levels of privacy, security and control from a user-centric perspective. Services are also benefit of advanced features such as cross-layer SSO and better access control and identity management. The viability of the proposed architecture has been demonstrated thanks to definition of all messages and information interchanged in SWIFT APIs in formal language which is very close to a real implementation language, which later has served for a security validation using the tool AVISPA. Finally, main SWIFT components, including user client, has been implemented and deployed on a demo testbed as definitive proof of feasibility for the architecture.

On the other hand, GEMBus put the focus on integration and trust control between heterogeneous services. It attempts to bring the advantages of SOA into the highly heterogeneous, open multi-domain environment that currently characterizes scientific e-infrastructures. The GEMBus commitment to openness and the support of high heterogeneity mean that it is also useful for environments in which a full SOA deployment is challenging, by proposing and enabling some core services that can create a basis for further migration to more consistent service oriented environment. On the core of these integration capabilities is GEMBUS Security Service, which is also enriched with the integration proposal between WS-Trust and OAuth 2.0 authorization technologies that allows to offer the interaction of services based on several security technologies as different as SAML, OAuth2.0, JSON tokens and x.509 certificates, even through different service interfaces as REST and SOAP. GEMBus architecture was implemented and deployed as functional prototype in order to test the viability of the solution offered. On this prototype,

we made several measures of time with the objective of evaluating the overload that implies the integration against the functional advantages that it offers. As a conclusion, we determined that despite the overload, the potential offered by the possibility of integrating such different services is compensated thanks to the flexibility it provides.

Future network infrastructures must integrate Identity Management functionality, in such a way that the user is provided with a unified and simplified vision of his identity, which results in improved privacy and security protection. As seen in this chapter, Identity federations are envisioned to unify and simplify user and service management through trust and IdM enablers. Their functions are not limited solely to identity management, but they can offer other advanced features related to the interoperability between services and heterogeneous technologies that enable us to create improved, richer and more powerful services that improve the user experience guaranteeing the privacy and the control of the information to the users. As seen through these two projects, some of these possible advanced features are the management of authorization and trust, privacy control, advanced identity management and anonymous access to services even through heterogeneous technologies. We have offered important mechanisms that enables identity federation to go an step further improving their capabilities with enriched and better services and characteristics.

Nevertheless, the interoperability problem is not limited to identity management and trust control level. It is necessary to go a step further to analyze the interoperability issues at interfederation level. Identity federations are limited by their specific sets of rules, protocols and attributes, and this raises the need for mechanisms to allow the integration of identity federations that create new opportunities and services. The next chapter analyzes the current status of interoperability between federations, establishing the open challenges and offering a solution for a specific case of study.

Chapter 4

Mechanisms for Federation Interoperability: the eduPEPS

Identity federations allow arrangements between several companies that let subscribers access services and the network of all companies in the group using the same digital identity. These agreements enable services to be unified and enable user based management. In addition, Identity federations allow better privacy and security control of user personal data, so improving and simplifying the management and the interaction with service providers. Therefore, federation systems are especially interested in harmonizing and unifying users' interaction with public administration, with the certainty that interoperability and authentication are critical for the successful advancement of digital technologies. In contrast, the increase in the number of federations, each focusing on different areas, revives the initial problem of the users need for a user account in each federation, and also implies the isolation of services and users due to the impossibility of interacting between them. This implies a problem of lack of interoperability and a drawback for global solutions. Thus, the creation of integration mechanisms between federations allows at the same time joint user bases and strengthens the services used among them.

Most existing interfederation solutions imply the migration or adaptation of in-production services to new authentication and interoperation mechanisms. This can be a good option in the case of new deployments, but in the case of running federations and services, it might not be practical if it entails the modification of entities and user flows, the migration to new protocols or the adoption of a complete new layer. All these changes usually imply the increase of operation complexity, number of rules and difficult political agreements.

We reviewed in Section 2.3 some of the most relevant identity federations in the

education, government and research sectors as well as the general public in order to establish their main characteristics, with the focus on federations oriented toward the administrative sector. Most of them are based on SAML protocol to encode and transmit the identity information, but this does not guarantee interoperability between them, due to the distinctive features that prevent interaction between federations. In general, deployed federations are isolated from others and users need specific credentials to access each one, and services are limited to their federation's audience. As we have seen, there are projects focused on promoting the integration of new services, such as eduGAIN (Sec. 2.3.2) and EUDAT (Sec. 2.3.3), but the problem persists in the case of deployed federations that have been successfully adopted and can not be directly integrated.

In contrast, some identity federations like STORK (Sec. 2.3.4) and eIDAS (Sec. 2.3.5) were born as federations with very specific and strict requirements in security and trust areas and are focused on keeping control in the hands of the end user at all times. The integration of this kind of federation requires additional and specific work focusing on solving technical interoperability gaps and the political and legal issues required to reach real interoperation.

The interconnections between STORK and eduGAIN are particularly beneficial since, while their initial scopes are different (public institutions vs educational sector), the integration of their users and services has enormous potential interest for both since, on the one hand, there is a great potential to increase the number of users, and on the other, it is possible to offer a large number of new services that were not included initially. Besides, the high level of security required in STORK means eduGAIN can take advantage of it to improve the LoA offered to its services.

Because of its nature, STORK is limited to a European Union scope mostly because obtaining an eID is restrained to EU member state citizenship. However, the services already deployed are of big impact and are business oriented. In contrast, eduGAIN is not limited by any political state border and, in consequence, it is already extended to other continents, although the services provided are of relative importance and focused on the research and educational business, and therefore limited in terms of daily life impact. All these factors promote interest from GÉANT and the European Commission to establish the necessary interoperability mechanisms.

In the previous chapter, we proposed several alternatives to improve the interoperability from an internal point of view for the identity federations, so enabling interconnection at user and service level. SWIFT offers a good solution to be deployed from the beginning and GEMBus works by providing interoperability at service level. As we will see in the following sections, other solutions usually imply the migration to new common protocols (such

as OpenID [146], OAuth [147] or OpenID Connect [80]), modifying all entities involved (Service Providers, Identity Providers, Attribute Providers, etc). Even when using the same protocol, it is necessary to adapt most of the federation components. The changes can even affect end users, since the process may not be transparent. In addition, the use of general superfederation solutions usually entails new legal and political issues for the existing federations.

Based on our collaboration experience in eduGAIN and STORK projects, we will work to define a specific and ad-hoc solution for the interoperability problem of integrating both federations. The proposal focuses on the establishment of interoperability mechanisms that allow transparent interaction for services and users from both federations thanks to identity management mechanisms and trust control. Therefore, users maintain their identities between federations and at the same time, they can access services of the other federation.

The following sections offer interoperability solutions between eduGAIN and STORK and eduGAIN and eIDAS. Our proposal focuses on offering a tailor-made solution that enables both federations to be integrated without having to migrate entities to new protocols, modify deployed services while supporting end user interaction flows as much as possible.

4.1 Specific background

there are several projects working on different AAI solutions and identity federation technologies. In the area of science there are several options. EGI [84] is a highly distributed, multi-disciplinary resource infrastructure, integrating more than 300 resource centres (service providers) and almost 20,000 users. EUDAT (Sec. 2.3.3) offers common data services, supporting multiple research communities as well as individuals in a resilient network of 33 European organizations based on B2ACCESS [148] and Unity [86], ELIXIR [87] builds its own infrastructure for biological information, while Umbrella [88] is the pan-European federated identity system for the 30,000 users of the European large photon/neutron facilities. All are good AAI examples, but as we will see below, their numbers in terms of users and services are not comparable with STORK (Sec. 2.3.4) and eduGAIN (Sec. 2.3.2).

The interconnection of identity federation is a relevant work area with several project and researchers working in parallel. Some very important work is being done in papers like [149] and [150] and the LIGO Project [150] to analyze the requirements of federation

and interfederation initiatives and propose alternative solutions to solve the problem like the GÉANT Trust Broker [151]. Kantara initiative [152], which is based on Liberty Alliance Identity Assurance Expert Group [153], works on create a common framework to harmonize baseline policies, business rules, and commercial terms against which identity trust services can be assessed and evaluated. Other projects, like InCommons [154] in the United States and AARC [155] in Europe, devote part of their efforts to expanding their base federations to interconnect with others.

The use of existing interfederation solutions to interconnect deployed federations implies the migration or adaptation of operating production services to new authentication and interoperation mechanisms. In some cases, for example new deployments, this is a good option, but when the work has to be done with running federations and services, it might not be a practical option because, in general, it entails the modification of the entities involved and user interaction flows. In the case of eduGAIN and STORK, both federations are based on SAML protocol, so even though there are other protocols like OAuth or OpenID Connect to transport authentication data, these imply the translation to or the migration from both sides. Other solutions based on SAML, like Shibboleth [24], imply the adoption of a complete new layer with a consequent increase in operation complexity, rules and political agreements management. These reasons lead to the design of an ad-hoc solution to resolve the specific problems of integrating these particular federations minimizing the number of added elements and modifications.

Both STORK and eduGAIN identity federations define a common set of policies, practices and protocols to manage identity as well as user and services trustworthiness across organizations. To design interfederation operation mechanisms, it is necessary to study the similarities and differences between federations in order to establish parallelism and **safe** the problematic points. The analysis will be focused on how each federation manages user identity, especially protocols and identifiers, insofar as they are the core of identity management.

As a consequence of STORK 2.0 project outputs, the European Union has invested in a new iteration of a European electronic identity federation represented by the eIDAS project, so interest in interconnecting with other federations, and with eduGAIN in particular, is still present. Despite the differences between STORK and eIDAS, the results of this study will be applied as the technical base to federate eIDAS and eduGAIN.

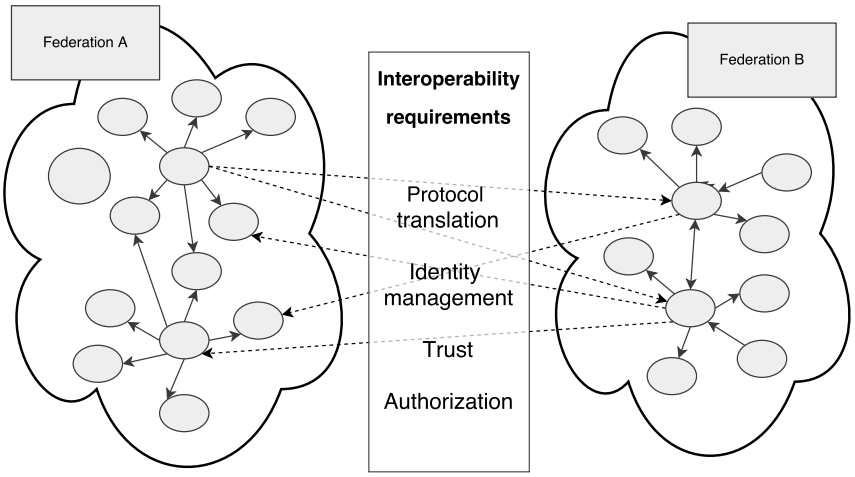


Figure 4.1: Interoperability requirements for existing federations.

4.2 Requirements and open challenges in federation interoperability

An interoperability solution has to offer mechanisms to solve different aspect related to protocol interoperability, identity management, trust and access control (authorization mechanisms) as is depicted in Figure 4.1 and seen previously.

The use of existing interfederation solutions to interconnect deployed federations can imply the migration or adaptation of operating production services to new authentication and interoperation mechanisms. The establishment of interoperability mechanisms can allow transparent interaction for services and users from different federations thanks to the integrations of their identity management mechanisms and trust control. An optimal integration also allows users to maintain their identities between federations and, at the same time, access to services of the other federation. There are many desirable features in the integration of existing federation related to different aspects:

- **Transparent integration:** This feature is one of the most important and can determine if an interoperability solution is adopted or not. It makes reference to how the integration affects users, service providers and identity providers depending on how many entities have to be involved in the integration process and how much; for example, if users have to learn new procedures to solve the same use cases or new emerging use cases, of if providers have to modify their configuration files or even their source code, changing their privacy and security policies (service and identity

providers), to name but some of the possibilities. Insofar as integration is more transparent and affects fewer entities, the integration process will be simpler, faster, cheaper and more comfortable and, therefore, with more chances of being successfully adopted .

- **Identity matching:** the use of services and resources from one federation for users of others is the final objective of the integration process, but when we talk about the integration of existing federations, it is necessary to consider the possibility that a user already has accounts created in both federations. The establishment of mechanisms that allow linking both digital identities, if the user so wishes, is a very interesting and desired feature that offers an improved user experience. This feature can be difficult to achieve, and even more so in a transparent manner.

- **Trust relationship:** In the integration process it will be necessary to establish new trust relationships between the entities involved. According to how the trust relationships are initially established in the federations (by the topology and the internal architecture) and the integration solution proposed, more or fewer entities will be involved in the process and required to create new trust agreements. In some cases, it is possible to maintain the trust control at federation level, which greatly simplifies the creation and management of new trust relationships.

- **Privacy and security:** user privacy is a key factor for the integration process. Depending on how the interoperability solution is designed, each federation can maintain the control of its users data or maybe, federations can share and mix their users database. In addition, the integration of users and services requires studying and comparing privacy policies, with the aim of maintaining the security and privacy levels offered to the users by the original federations. In the case of finding differences, the integration solution should work on guaranteeing the highest level of security and the most restrictive privacy policies.

- **Live migration:** another desirable feature in relation with designing an interoperability solution for existing and running federations is the possibility of offering the integration with long cuts in services. This feature is closely related to the transparency and security of the proposed solution.

The ideal objective in the interfederation problem of two existing identity federations is to achieve bidirectional matching between identities with the specific properties of transparent, fluid and "on the fly" integration between users and services of both

federations, not only allowing basic authentication, but also complex scenarios like linking existing accounts and requesting additional attributes to provide identity management. To reach total integration for all the use cases is a very ambitious objective and requires the study of different integration possibilities depending on the preconditions and the scenarios proposed. In order to define and interoperability solution between two federations, it is necessary to define the equivalence between entities and all the required mechanisms to translate and to make the matching between identifiers, messages and protocols.

4.3 eduGAIN and STORK: the interoperability problem

As commented on earlier, eduGAIN is the federation par excellence in the educational and research sector around the world, especially in universities and research centers. In contrast, STORK is focused on governmental services and institutions inside the European Union. Although STORK is based on SAML2.0, it implements some extensions that together with some particularities, such as the identifier and attributes used and its rigid infrastructure based on PEPS, make direct interconnection with eduGAIN federation impossible.

Both federations have entities in common with almost equal functions, such as users, SPs, IdPs and APs. The issues arise with intermediary entities like STORK's PEPS and eduGAIN's MDS that have no direct match. A detailed description of all entities involved was made in Chapter 2. As seen, STORK PEPS plays different roles depending on the entity with which it has to interact. When it plays the role of S-PEPS, it shows the list of available C-PEPS to the user so that she can choose her national PEPS. At the same time, eduGAIN offers a similar functionality through the WAYF Service (Where Are You From). In the second role, the C-PEPS redirects the citizen to the official IdP and allows her to choose which attributes have to be recovered in each IdP and AP. In eduGAIN, this functionality could be done by the IdP, the DS or even statically. Table 4.1 offers a side-by-side comparison between the entities of both federations.

SAML 2.0 is wide and flexible protocol that allows adaptation to the environment's needs. eduGAIN makes use of $SAML_{Int}$ [97] specification, a SAML subset that maintains the compatibility with the standard. STORK 2.0 makes use of SAML adaptation capabilities to add language extensions that allow including addition information needed to the STORK operation. These extensions make direct communication with the standard version incompatible. This SAML dialect will be denominated $SAML_{STORK}$ from now on and can be reviewed in the STORK D4.4 deliverable [107]. The appendixes provide

eduGAIN	STORK	Description
user	user	The user in both cases starts the use case by requesting a service that requires an authenticated user. In an educational scenario, the client may be a student who needs to carry out some procedure. The eduGAIN user is a student or employee of some institution. In the case of STORK, the client has to be registered at least by the government.
SP	SP	Service Providers are very similar. They request the user authentication and, in some cases, additional attributes before offering the service.
MDS/DS	S-PEPS Source Pan-European Proxy Service	The S-PEPS shows the list of available C-PEPS with the aim that the user chooses his national PEPS. In the case of eduGAIN, the DS offers a similar functionality asking the WAYF (Where Are You From) and redirecting the user to the IdP.
IdP DS or statically	C-PEPS Source Pan-European Proxy Service	The C-PEPS redirects the user to the official government IdP and allows the user to choose from which AP the attribute recovery should be done. In eduGAIN this functionality is not clearly defined, and could be done by the IdP, the DS or statically; it is an operational decision.
IdP	IdP	Identity Providers are responsible for authenticating the user and providing a signed assertion to the SP. The functionality is the same in both cases.
AttrP	AP	Attribute Providers offer additional user information (attributes) to the SP. The functionality is the same in both cases. Each platform supports a different set of attributes.

Table 4.1: eduGAIN and STORK side by side entity comparison.

examples of requests and responses in both SAML dialects that allow us to observe the differences between $SAML_{Int}$ of eduGAIN (Appendix B) and $SAML_{STORK}$ of STORK (Appendix C).

Apart from the differences with the SAML language, STORK does not support Single Sign-On (SSO). SSO is a mechanism that allows users to access multiple systems by performing a single authentication process [21]. $SAML_{STORK}$ also requires a full authentication to recover any attribute, due to the impossibility of establishing sessions. Because of its capability to recover attributes from different attribute providers (AP) in a single request, STORK allows multiple assertions per response, whilst eduGAIN forces just one assertion per response. Speaking of attributes, STORK defines a quality measurement level, named QAA and AQAA (Quality Authentication Assurance y Attribute Quality Authentication Assurance) for each attribute which can not be mapped to eduGAIN since they are part of $SAML_{STORK}$'s extensions.

As for the mechanism for identifying users, STORK 2.0 generally uses the national identification number of each country as a reference for users within the federation, while eduGAIN usually uses an identifier similar to a mailing address that identifies the user within the institution to which he belongs.

The STORK infrastructure is quite static and requires a manual registry to establish peer-to-peer trust relationships between PEPSes in different countries. PEPSes are government-managed entities and are in charge of retransmitting requests between countries, signing statements and enforcing STORK policies. On the other hand, eduGAIN offers dynamic metadata distribution through the MDS entities. Similarly, it contrasts STORK's centralized architecture (PEPS-centric) with the distributed eduGAIN architecture, where the only centralized points are located in Meta-Data Servers (MDSs).

Our proposed solution is to incorporate a new entity, denominated eduPEPS, situated between the STORK and eduGAIN federations. This intermediate entity would be in charge of the SAML translation (between $\text{SAML}_{\text{stork}}$ and SAML_{int}) as well as of the translation of identifier and attributes. The eduPEPS has to act as a trusted entity for both federations, playing the role of DS and MDS of any other eduGAIN federation for the eduGAIN side, and acting as a PEPS (S-PEPS or C-CPEPS depending on the scenario) of another country for the STORK side.

4.4 Matching federation identities between eduGAIN and STORK 2.0

As we have seen in the previous section, both federations have differences at several levels,from the kind and roles of the entities involved to the internal details, such as identifiers. To achieve the interoperability between both federations, we will have to define the equivalence between entities and all the required mechanisms to translate and perform the matching between identifiers, messages and protocols.

Although both federations are based on SAML, translation between their message protocols is not direct. As we have seen previously, eduGAIN uses $SAML_{INT}$, which defines a minimum set of bindings and rules that needs to be followed from the SAML standard to reach a minimum level of interoperability. On the other hand, STORK 2.0 is designed to be a subset of the standard OASIS SAML 2.0 specification but with some extensions [107]] which are used to allow additional STORK attributes to be requested at authentication time and also to allow the transport of additional information required for completing the authentication process. We will consider the existing possibilities of translating and working with the identifiers that represent those identities.

4.4.1 Main identifiers description

Each of the federations studied uses its own system of identify users. STORK only works with eIdentifier (eId) as a key to refer an user, while eduGAIN has more than one identifier. First, we will provide a review of the specific characteristics of each federation's identifiers, in order to analyze later the similarities and differences between both federations' identifiers and so define the most appropriate mechanisms for translation between them.

STORK identifier description

The main identifier in the STORK federations is the Electronic Identifier Number or eIdentifier (eId). This is what uniquely identifies a person within a service. Its usual format "NC/NC/xxxxxxxxx..." has three parts, which correspond to the Nationality Code of origin country followed by a slash ("/"), the Nationality Code of visited country followed by a slash ("/") and finally, a combination of readable characters, which uniquely identify a person in the country of origin of the identifier. This definition of the STORK eIdentifier is extracted from "D.4.9 Final Version of Functional Design" at section 2.1.1.3.1 [156]. Table 4.2 offers a detailed description of the STORK identifier.

Name	Description
Field Name	eIdentifier
Definition	Cross-Border Electronic Identity Number. This is a number which uniquely identifies a person within a service.
Domain	String of digits, upper and lower case letters (26 letters A-Z and a-z), and "+", "/", the characters of base64. Max length: 94 (3+3+88)
Original / Derived	Original
Format	Format:NC/NC/xxxxxxxxx...
Description	String of characters comprising three parts: • The first part is the Nationality Code of the identifier This is one of the ISO 3166-1 alpha-2 codes, followed by a slash ("/") • The second part is the Nationality Code of the destination country This is one of the ISO 3166-1 alpha-2 codes, followed by a slash ("/") ISO 3166-1 alpha-2 codes are two-letter country codes defined in ISO 3166 standard published by the International Organization for Standardization (ISO) to represent countries. • The third part is a combination of readable characters, which uniquely identify a person in the country of origin of the identifier. Examples: - ES/AT/02635542Y (Spanish eIDNumber for an Austrian SP), - GB/NL/274136A (UK eIDNumber for a Dutch SP).
Graphical representation	

Table 4.2: Detailed description of the STORK identifier

eduGAIN identifiers description

In eduGAIN there are two possible attributes used to identify the user, which can match with the eID. The formats [98] of both identifiers are clearly influenced by the SAML specification [35].

The `eduPersonTargetedId` is unique, non-reassigned, persistent, opaque and targeted identifier of the user and it is the usual key for eduGAIN. In our pilot, the format is 500255@gn-vho.grnet.gr, which is the typical serialized representation. It is used to designate a principal. In abstract terms, it is a tuple consisting of an opaque identifier for the principal, a name for the source of the identifier, and a name for the intended audience of the identifier. The name of the intended audience also takes the form of an absolute URI, and may refer to a single service provider or a collection of service providers (for which SAML V2.0 uses the term "Affiliation", not to be confused with the ordinary eduPerson use of the term).

On the other hand, the `eduPersonPrincipalName` (ePPN) is unique and persistent scoped identifier of the user but not opaque. It should be represented in the form "user@scope" where 'user' is a name-based identifier for the person and where the "scope" portion must be the administrative domain of the identity system where the identifier was created and assigned. Each value of 'scope' defines a namespace within which the assigned identifiers must be unique. Given this rule, if two ePPN values are the same at a given point in time, they refer to the same person. There must be one and only one "@" sign in valid values of `eduPersonPrincipalName`. Syntactically, ePPN looks like an eMail address but is not intended to be a person's published eMail address nor to be used as an eMail address. In general, name-based identifiers tend to be subject to some degree of expected change and/or reassignment (e.g. jdoe@example.org).

4.4.2 Identifier comparison and possible identity translation mechanisms

Since there are two options in eduGAIN for use as identifiers, this section presents two alternatives to define the translation between the eduGAIN and STORK identifiers.

eduGAIN eduPersonTargetedId vs STORK eIdentifier

The first matching option considers the integration between the eduGAIN `eduPersonTargetedId` and the STORK eIdentifier. On the basis of the description made in eduGAIN `eduPersonTargetedId` vs STORK eIdentifier Section (Sect. 2), the main difference

between both identifiers is that the eduGAIN identifier must be opaque. Although this is not a requisite for the STORK eIdentifier (eID), in some countries the eID is opaque too. While in some countries the third part of the eID is the national fiscal identifier or similar, in others it is a hash function result (opaque) of this number (e.g: Italy uses hashes of the 'Carta d'Identità').

If the translation is made from STORK eIdentifier to eduGAINTargetedId, the eIdentifier can be used in the transformation function as a uniform string without different parts. However, when the translation is in the opposite direction, from eduGAINTargetedId to eIdentifier, it is necessary to consider the three fields of STORK identifier. There are two country fields (first and second parts of eID). The first part is the Nationality Code of the identifier (citizen's home country), while the second part is the Nationality Code of the destination country. The letter assigned to second part depends of the STORK SP country (standard operation mode) but the first field has to be estimated from eduGAIN IdP. One option is to use a fixed free pair of letters to describe all eduGAIN identities. Another option is try to calculate them from the IdP domain or extract them from IdP metadata.

Finally, the third part of the STORK identifier makes reference to the specific user. It is a combination of readable characters with a amaximum of 88 characters which uniquely identify a person in the origin country of the identifier. The starting point to translate this part is the `eduPersonTargetedId`, which usually is transported inside a SAML structure or also as serialized string. Its eduGAIN value can have special characters such as '@', '-', '_' and '.', which do not fall within the readable category. For these reasons a transformation function is needed to convert from the `eduPersonTargetedId` identifier to the eIdentifier format. This function can work in different ways, for example, by only transforming the unreadable characters for other readable ones, or in a more complex way. This is design decision but it can have implications later, as we will see at the second use case. The transformation function should be applied in an intermediate point between the STORK federation and the eduGAIN IdP since it has the confidence of both parties.

eduGAIN eduPersonPrincipalName vs STORK eIdentifier

The second matching option considers the integration between the eduGAIN `eduPersonPrincipalName` and the STORK eIdentifier. This combination offers more integration possibilities since the usual value assigned at eduGAIN for this attribute is similar to an eMail format and sometimes can even play that role (the user's eMail). The `eMail` attribute is also usually available at STORK, so matching mechanisms could be established using

the STORK email attribute as additional information for the matching mechanisms. Since it is a standard attribute, it may not be unique, so this particularity must be taken into consideration if this option is selected.

In the case of considering the STORK eMail as candidate to make a matching to eduGAIN identifier, it is necessary to bear in mind that it is a standard attribute and not an identifier so it does not have the same properties. The attribute might not be unique, be unavailable or not exist, or change in function of the contexts: official/governmental vs educational/university.

On the other hand, the translation from eduGAIN authentication (based on `eMail` or the opaque identifier) to STORK eIdentifier is more complicated. There is no eduPerson attribute for the National Identity Number/Fiscal Number and the `eduPersonPrincipal` has a completely different meaning than STORK eIdentifier, so there is no direct translation in this direction. The best option in this case is to use the `eduPersonTargetedID` to calculate the third field of the STORK eID. Thus, the first two fields of the `eIdentifer` have been translated as we have defined in previous section.

4.5 Matching between attributes

EduGAIN and STORK have significant differences in structure, security and scope that profoundly influence to the point of determining the set and format of attributes that are exchanged. In this section we intend to study the particularities of each federation to establish the possible equivalences between the different attributes used in order to achieve the highest degree of integration possible.

4.5.1 STORK attributes

STORK makes use of the SAML extension mechanisms to transport a specific set of attributes in relation with the SP or the user. In addition, STORK allows us to establish how mandatory each attribute request is and defines the Attribute Quality Authentication Assurance (AQAA) levels that denote the confidence in each attribute. This meta-information relating to the attributes is characteristic of STORK and can be transmitted thanks to the extensions included in $SAML_{STORK}$.

STORK classifies the attributes into different subsets depending on the context in which they originate. Table 4.3 depicts the list of attributes related to personal user information. They can be queried about the subject of an authentication, of the user in session (representing person) and of the represented person in case of Authentication on

behalf of or Powers (for digital signature). They may be requested several times in one query, e.g. with isAgeOver this makes sense. Unknown attributes (attributes not listed in the following table) in a request are ignored, unless they are mandatory.

Another relevant set of attributes is related to Academia scope listed in Table 4.4. These are all attributes that can be queried about the subject, which can be provided by institutions of the Academia (higher education) sector. A request may be responded to several times in one response, e.g. with hasDegree this makes sense: a person may have several university degrees. As in the previous case, unknown attributes (attributes not listed in the following table) in a request are ignored, unless they are mandatory.

In particualr, `diplomaSupplement` and `currentDimplomaSupplement` are not further defined within the common interoperability layer, as they are considered irrelevant outside the academic world. So formally they are defined on the string domain, even though the underlying contents have an internal xml structure.

The involved stakeholders will not be limited to the academic world, but will also include legal entities from the private sector that will be enabled to access sets of academic attributes valuable for their purposes, ranging from a full qualification report (for job selection, for instance) to data about specific fields of expertise (for conducting opinion surveys among experts, for example) or simple attributes (date of birth, degree...) for targeting services to specific groups of individuals.

4.5.2 eduGAIN attributes

During eduGAIN's successful authentication process, the Identity Provider (IdP) supplies the Service Provider (SP) with a set of default attributes related to the authenticated user (even if no attribute request has been performed). These attributes define the authenticated user identity and should be used during the subsequent attribute retrieval process. In particular, a transient session identifier is provided in the authentication response and will be used as identifier in the attribute retrieval process, thus a certain degree of anonymity level is provided. The user related information is included in the response attributes, such as `givenName`, `email` or `eduPersonPrincipalName`.

It is recommended by Internet2 MACE-Dir Working Group [157] that eduGAIN Participant Federations ensure that Identity Providers supply the attributes listed in Table 4.5. A recommended attribute means that it is available, in general, for most end users. However, it can be left empty for those end users who do not qualify for any of the values in the vocabulary.

Friendly Name	Name	Name Format
eIdentifier	http://www.stork.gov.eu/1.0/eIdentifier	CC/CC/string
Given Name	http://www.stork.gov.eu/1.0/givenName	UTF-8
Surname	http://www.stork.gov.eu/1.0/surname	UTF-8
Inherited Family Name	http://www.stork.gov.eu/1.0/inheritedFamilyName	UTF-8
Adopted Family Name	http://www.stork.gov.eu/1.0/adoptedFamilyName	UTF-8
Gender	http://www.stork.gov.eu/1.0/gender	"M" or "F"
Date of Birth	http://www.stork.gov.eu/1.0/dateOfBirth	Date
Country of Birth	http://www.stork.gov.eu/1.0/countryCodeOfBirth	ISO-3166-3
Place of Birth	http://www.stork.gov.eu/1.0/placeOfBirth	UTF-8
Nationality	http://www.stork.gov.eu/1.0/nationalityCode	Country code
Marital Status	http://www.stork.gov.eu/1.0/maritalStatus	S = Single M = Married P = Separated D = Divorced W = Widowed
Text Residence Address	http://www.stork.gov.eu/1.0/textResidenceAddress	UTF-8 (with new lines)
Canonical Residence Address	http://www.stork.gov.eu/1.0/canonicalResidenceAddress	XML
eMail Address	http://www.stork.gov.eu/1.0/eMail	e-mail
Title	http://www.stork.gov.eu/1.0/title	UTF-8
Residence Permit	http://www.stork.gov.eu/1.0/residencePermit	UTF-8
Pseudonym	http://www.stork.gov.eu/1.0/pseudonym	UTF-8
Age	http://www.stork.gov.eu/1.0/age	Numeric
IsAgeOver	http://www.stork.gov.eu/1.0/isAgeOver	Requested age boundary if true, empty if false
Signed Document	http://www.stork.gov.eu/1.0/signedDoc	see below
Citizen QAA Level	http://www.stork.gov.eu/1.0/citizenQAALevel	Numeric {1:4}
Fiscal Number	http://www.stork.gov.eu/1.0/fiscalNumber	UTF-8

Table 4.3: Stork 1 attribute definitions.

Friendly Name	Name (in XML Format)
diplomaSupplement	http://www.stork.gov.eu/1.0/diplomaSupplement
currentStudiesSupplement	http://www.stork.gov.eu/1.0/currentStudiesSupplement
hasDegree	http://www.stork.gov.eu/1.0/hasDegree
isStudent	http://www.stork.gov.eu/1.0/isStudent
isAcademicStaff	http://www.stork.gov.eu/1.0/isAcademicStaff
isTeacherOf	http://www.stork.gov.eu/1.0/isTeacherOf
isCourseCoordinator	http://www.stork.gov.eu/1.0/isCourseCoordinator
isAdminStaff	http://www.stork.gov.eu/1.0/isAdminStaff
habilitation	http://www.stork.gov.eu/1.0/habilitation
acTitle	http://www.stork.gov.eu/1.0/actitle

Table 4.4: Stork Academia Data definitions

4.5.3 Attribute comparison and matching

Based on the above descriptions, Table 4.6 identifies where eduGAIN and STORK diverge using a side by side comparison as the same time that it establishes a matching proposal.

The differences between eduGAIN and STORK are difficult to save. In the first place, the attribute sets are different and only have equivalences in a very small subset of attributes. In addition, eduGAIN does not establish a clear and specific list of attributes, but leaves freedom for each federation to use what it deems appropriate, only recommending a (very) minimal set.

On the other hand, the AQAA and mandatory information does not exist in eduGAIN federation, so its values have to be ignored when the translation is from STORK to eduGAIN and calculated dynamically or fixed in advance when the translations is from eduGAIN to STORK, which implies a big lack of security especially from the point of view of STORK federation.

Appendix D depicts two tables with real examples of the translation from an eduGAIN identity (identifier and attributes) to STORK 2.0 format and vice versa.

4.6 Summary of eduGAIN and STORK differences

This section offers a summarized view of the most relevant differences analyzed throughout the previous sections. All these points must be solved to reach a functional and transparent

Friendly name	Defined in	Notes
displayName	[eduPerson]	Since other attribute types such as 'cn' are multivalued, an additional attribute type is needed. Display name is defined for this purpose. Syntax may be culturally dependent (for example, Firstname Lastname or Lastname Firstname).
common name (cn)	[eduPerson]	Syntax may be culturally dependent (for example, Firstname Lastname or Lastname Firstname).
mail	[eduPerson]	If populated, this must be the end user's valid personal mail address (not a shared mailbox).
eduPersonAffiliation and eduPersonScopedAffiliation	[eduPerson]	
eduPersonPrincipalName	[eduPerson]	
SAML2 Persistent NameID (eduPersonTargetedID)	[SAMLCore, eduPerson]	
schacHomeOrganization	[SCHAC]	
schacHomeOrganizationType	[SCHAC]	

Table 4.5: Minimun set of eduGAIN attributes

eduGAIN	Stork
displayName	pseudonim
commonName	givenName + surname
mail	eMail
eduPersonAffiliation & eduPersonScopedAffiliation	isStudentOf/isTeacherOf
ePPN (eduPerson Principal Name) / eduPersonTargetedID	eIdentifier
eduPersonTargetedID	N/A
shacHomeOrganization	nameOfInstitution (part of isStudentOf and isTeacherOf)

Table 4.6: Matching between basic attributes

integration between both federations.

- Identifiers: email (`eduPersonTargetedId` & `eduPersonPrincipalName`) vs `eID`. STORK does not use the Subject's NameID field to identify the user; it uses an attribute named eIdentifier that contains a fiscal number or any other government identifier with a country code. It is necessary to establish some mechanisms to make a matching between both identifiers.
- $SAML_{Int}$ vs $SAML_{STORK}$
  - ⋆ STORK does not offer any SSO mechanism, so each SP request requires a new authentication.
  - ⋆ protocol flow: $SAML_{STORK}$ does not allow session establishment, which means that an Attribute Query cannot be made after an Authentication Request without involving a new Authentication Request process. It is necessary to request the attributes needed at the same time as the AuthRequest.
  - ⋆ Different attribute sets. STORK makes use of the SAML extension mechanisms to transport a specific set of attributes with respect to the SP or the user. In addition, STORK allows us to establish how mandatory each attribute

request is, so it will be necessary to study how to solve this in the case of $SAML_{Int}$. Furthermore, STORK defines Attribute Quality Authentication Assurance (AQAA) levels that denote the confidence in each attribute. On the other hand, eduGAIN defines assurance levels focused on the authentication process, which are a set of URIs that assert compliance with specific standards for identity assurance.

  - ⋆ The $SAML_{STORK}$ responses may contain more than one assertion when there is a necessity to ask several APs, whereas the eduGAIN response only retrieves attributes from one AP. It should be necessary to establish a unification mechanisms for all the $SAML_{STORK}$ assertions in one, in order to resemble to the standard performance of $SAML_{Int}$.

- Optional automatic registration (eduGAIN MDS) vs forced manual registration (at STORK's PEPS). STORK does not publish information regarding metadata. The STORK configuration is done manually, establishing trust relationships one by one. On the other hand, eduGAIN establishes mechanisms to optionally automatically update the Metadata Service with the metadata of IdPs and SPs.

- Centralized vs distributed. STORK PEPSs suppose a centralized point where all the IdP, SPs and APs have to contact and trust. This scheme does not exist in eduGAIN, which is organized in a distributed manner. The only centralized point is the MDS, which is used to simplify the interconnection between federations.

4.7 Scenario integration proposal

Since we are working with two different identity federations like eduGAIN and STORK, there are multiple configuration possibilities according to the federation origin of the user's identity and the federation of the Service Provider. Depending on how the endpoints of each federation are combined with the others, we establish two different scenarios to solve. Scenario one (Fig. 4.2) consists of a STORK user trying to access an eduGAIN SP that requests her STORK identity. On the other hand, scenario two (Fig. 4.3) consists of an eduGAIN user trying to access a STORK SP that requests her eduGAIN identity.

For each one of these scenarios, we define two use cases in function of the user federation and the complexity of the authentication process. First use case consists of one federation's SP requesting the user's authentication and attributes to the IdP of the other federation

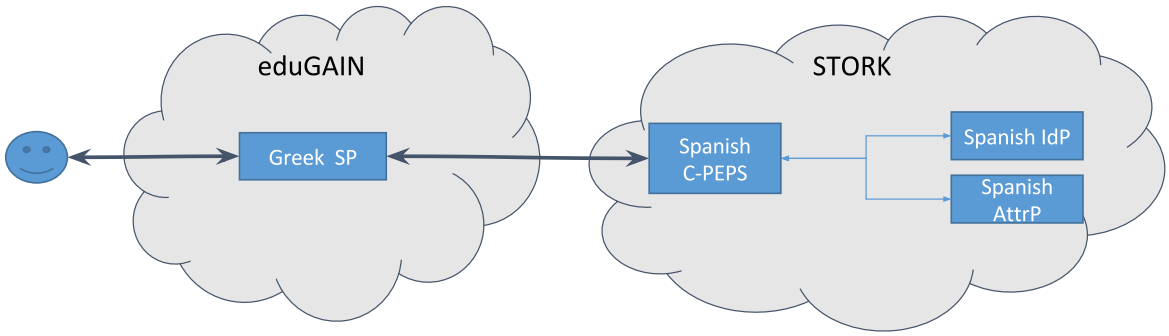


Figure 4.2: Scenario 1, STORK user visiting an eduGAIN service

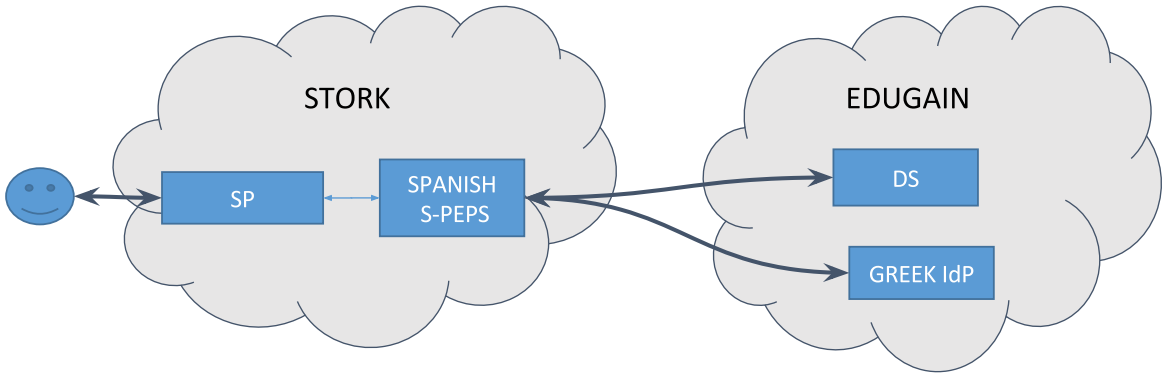


Figure 4.3: Scenario 2, eduGAIN user visiting a STORK service

without considering if the user already has an account on the visited SP or some possible subsequent connections to the user's home federation IdP to ask for additional attributes.

As an extension of the first use case, we consider whether it is possible to link the existing user profile in the visited service with the recovered user account's identifier from the home IdP. In this assumption, the user has two accounts, one in each federation, and an exiting user profile at the SP. The objective is to establish some mechanism to match between both identities at the SP, that is, based on the IdP's identity being able to identify the SP's federation user profile.

The second use case studies whether it is possible to ask for more attributes using the identifier translated in the first authentication process (use case one). In this second use case, the work is focused on how to use the identifier recovered in the first steps (use case one) to work with the IdP to recover more attributes.

4.7.1 Scenario 1: eduGAIN SP requesting STORK identity

Scenario 1 considers the situation in which a STORK user tries to access and use an eduGAIN SP. The objective is to get a bidirectional matching between the STORK identifier and the eduGAIN one with the aim of transparent, fluid and "on the fly" integration between users and services of both federations. As we will see, it is feasible to obtain a eduGAIN identifier from the STORK one, the most difficult task here is how to link this translated identifier with an existing eduGAIN identifier for the same user. A detailed analysis of the possible use cases is shown below. To solve these use cases, we are going use the intermediate entity proposed in Section 4.3, which will be denominated Intermediate Point.

First use case: initial STORK authentication and attribute recovery from eduGAIN SP.

The first use case consists of a STORK user accessing an eduGAIN SP. The user accesses SP with a standard `HTTP GET request` that initiates a redirection flow through the new Intermediate Point to the STORK IdP where the user is authenticated and his identity information is recovered. With these attributes, the user goes back to the SP, again through the Intermediate Point, to the SP where he or she is allowed to access the service as shown at Fig. 4.4.

STORK federation uses and returns eIdentifier as main identifier for its authentication, so it should be translated to the eduGAIN identifier, in this case, to edugainTargetedID. The STORK identifier is opaque and persistent so a transformation function must be

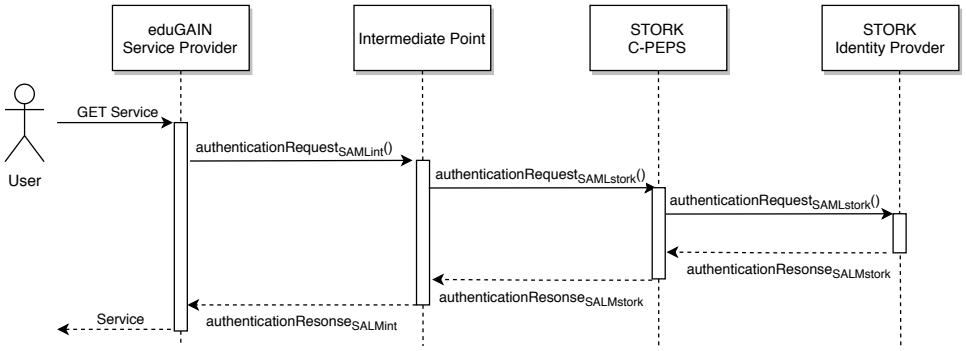


Figure 4.4: Initial STORK authentication and attribute recovery from eduGAIN SP

defined to make the translation between both identifiers. One possible implementation of this function could be to apply a hash function with the eIdentifier as input that produces an opaque and persistent identifier that can be used as edugainTargetedID.

The only requisite to use this solution is to add the Intermediate Point entity that plays the role of a proxy. This entity should have a trust relationship with both sides/federations playing STORK SP role for the STORK entities and eduGAIN IdP role for the eduGAIN SP. When the message arrives at the Intermediate Point the content is translated from one dialect to the other (from $SAML_{INT}$ to $SAML_{STORK}$ and vice versa) and by applying the function to translate from one identifier type to the other. The required mechanism to translate the rest of fields and attributes between both SAML message types is outside the scope of this work.

The Intermediate Point allows transparent integration between both federations without any specific change to the entities involved beyond establishing a trust relationship with the new entity.

First UC extension: linking the Stork identity with a previously existing eduGAIN user profile at eduGAIN SP.

In this case (shown in Fig. 4.5), the idea is to link the obtained STORK identity with a possible existing eduGAIN profile at the eduGAIN SP. Since the translation function must provide the same translated identifier between different access for the same user, it is possible to establish a persistent profile in base to the STORK authentication. The usual problem is that in most cases the obtained identifier from the STORK eIdentifier is different from existing one at the eduGAIN SP so a direct equivalence can not be established.

Based on a non transparent model, in which the SP knows that the translation process between federations is being made, a subsequent task can be carried out using other

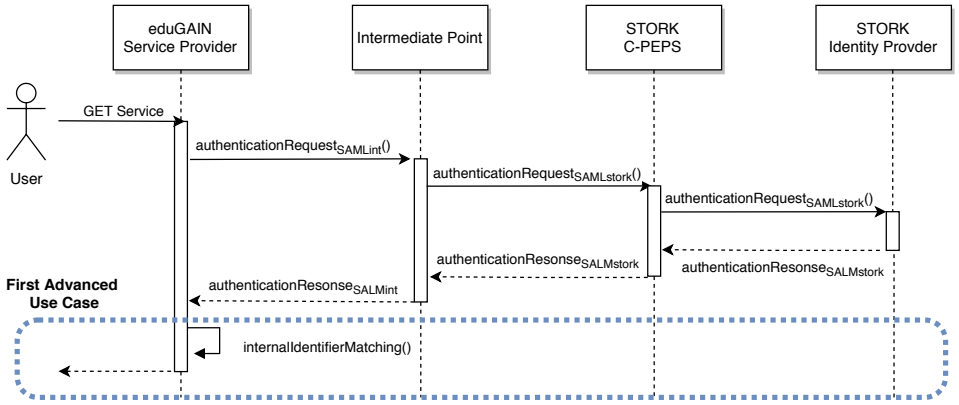


Figure 4.5: Linking the Stork identity with an existing eduGAIN user profile at eduGAIN SP

attributes and response messages information obtained in the authentication process. Under this assumption, the most interesting result is gained when working on the user `eMail` and the `eduPersonPrincipalName` (ePPN) attributes, since they are good candidates to find an equivalence between federations. If the retrieved STORK user identity contains the `eMail` attribute, it can be used to be translated to a possible ePPN, due to their similar formats. If the SP supports it, the ePPN can be used to establish the equivalence between the STORK identity and the user's account at the SP instead of using the `eduGAINTargetedID`. This mechanism is only a compatibility enhancement and does not guarantee anything even though the `eMail` attribute is available. Some of the possible reasons are that the user can have several eMails, and depending on the contexts, none may match to the specific case, or the ePPN may even not be a real eMail but just uses email format.

Second use case: requesting additional attributes.

New STORK attribute recovery always demands user authentication, as if it was a new use or the first time (see Fig 4.6), even though the SP can provide the user's identifier. From the eduGAIN side, this operation mode is unnatural since the SP can usually make an attribute query directly through the IdP in base of the user identifier. When this mechanism is available, the usual SAML binding [35] used is through an internal SOAP call between the SP and the IdP. In this case, the user browser (the user agent) does not participate and the redirection web flow is broken, so it can not be integrated with STORK.

To avoid this operation mode, the Intermediate Point (which is playing the role of an IdP) must not offer that option in its metadata [158] in order to force SP to try to ask for the attributes another way. The next SP's step depends of SP's specific implementation.

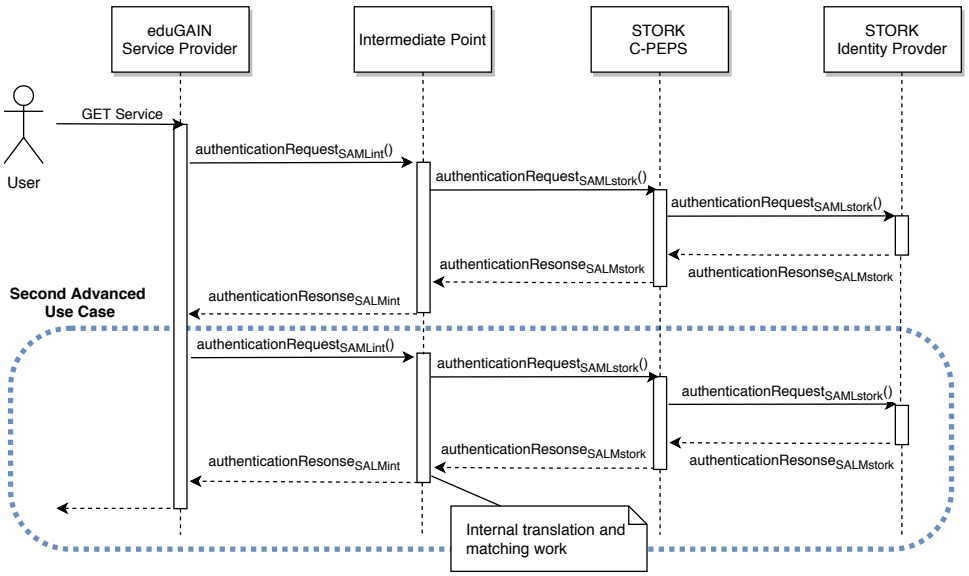


Figure 4.6: Use case 2, eduGAIN SP requests additional attributes from STORK IdP

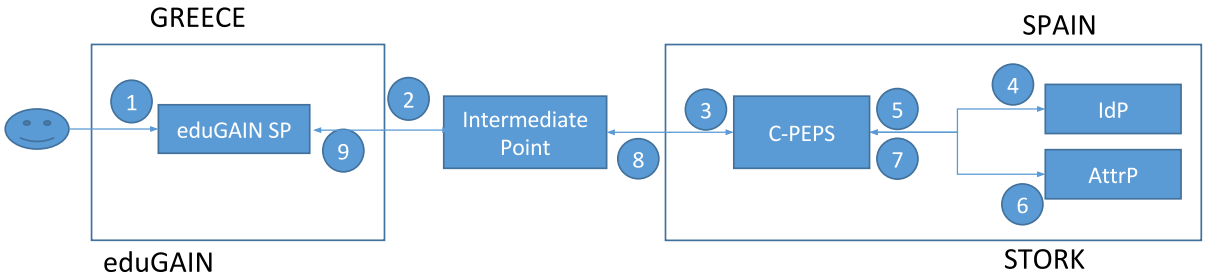


Figure 4.7: Scenario 1 flow schema, STORK user accesing an eduGAIN SP

If the SOAP binding can not be used, the process can fail and the SP displays an error to the user. Another option is that the SP makes a new authentication request including an explicit list of the required attributes. In this case, the flow will be similar to a new authentication request as described in the first use case, as is shown in Fig. 4.4.

Summary results

To sum up the general lines from both uses cases, it is necessary to use the Intermediate Point, as is depicted in Fig. 4.7, that has the confidence of both federations and is in charge of doing the translation work between SAML dialects (protocols) and identifiers. The Intermediate Point plays the role of STORK S-PEPS for STORK federation, and the role of an eduGAIN IdP for the eduGAIN SP.

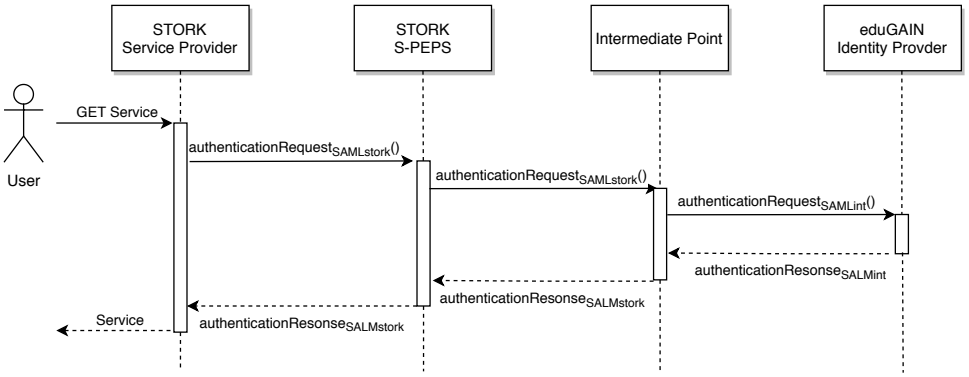


Figure 4.8: Initial eduGAIN authentication and attribute recovery from STORK SP

As can be seen in the Scenario 1 analysis, both use cases are possible. In contrast, the matching use case extension, although not impossible, can not be guaranteed, due to the complexity of making a match between existing accounts from both federations, with their identifiers and attributes.

4.7.2 Scenario 2: STORK SP requesting eduGAIN identity

Scenario 2 considers the situation in which an eduGAIN user tries to access and use a STORK service. The objective, as in Scenario 1, is to get a bidirectional matching between the eduGAIN identifier and the STORK one with the aim of transparent, fluid and "on the fly" integration between users and services of both federations.

First use case: initial authentication and attribute recovery

In this use case (Fig. 4.8), the eduGAIN user requests the access to a STORK service using his eduGAIN identity regardless of anything else (without considering a possible prior relationship with the STORK service, as will be studied at the use case 2).

As introduced above, eduGAIN works with two different identifiers: the `eduPersonTargetedId` and the `eduPersonPrincipalName`. Since eduGAIN uses and returns edugainTargetedId as its main identifier, it should be translated to the STORK eIdentifier.

The STORK identifier has three parts. The first two parts make reference to the destination and origin of the user, although they have no equivalent in eduGAIN, their value can be set easily via the Intermediate Point. The visited country is calculated from the SP request and the origin country from the user IdP. The third part can be calculated by applying a function similar to the one described in Sect. 4.4.1 using the eduGAIN identifier as input.

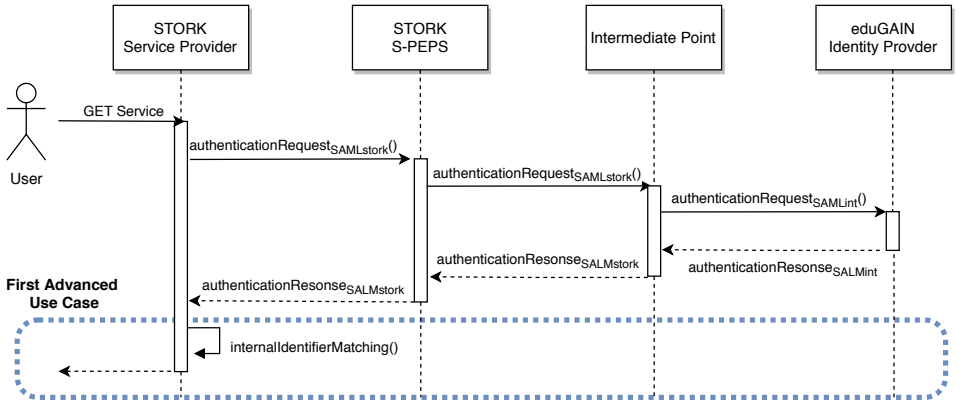


Figure 4.9: Linking the eduGAIN identity with a previously existing STORK user profile at STORK SP

The Intermediate Point is in contact with all the eduGAIN user personal information (identities and attributes) and adapts them to SAML STORK dialect. STORK offers and requires high levels of security and privacy [106] to and from users and services due to its work at governmental levels. eduGAIN also has security and privacy requirements [159] that obviously have to be covered too, so this is a critical point for all the proposed solutions. Due to the important labor done by the Intermediate Point and the information sensibility implications, we propose that it is designed as an extension module of the eduGAIN's IdPs or the STORK's PEPS, so there are no new parties, and the control remains with the original parties involved.

First UC extension: linking the eduGAIN identity with a previously existing STORK user profile at STORK SP

This use case (Fig. 4.9) considers how to link the recovered eduGAIN identity with an existing user profile at the STORK SP. As we have already seen in the previous use case, eduGAIN does not have any attribute similar to the fiscal number to make a direct translation from eduGAIN to STORK. So in this case, a transparent translation is virtually impossible.

If the STORK SP stored its own user information and it knew about the integration work, the SP could use all the collected attributes to search and match with all the stored user attributes at SP profile. This scenario is very complex and improbable as it implies modifying the services involved instead of using the Intermediate Point to make the whole process transparent.

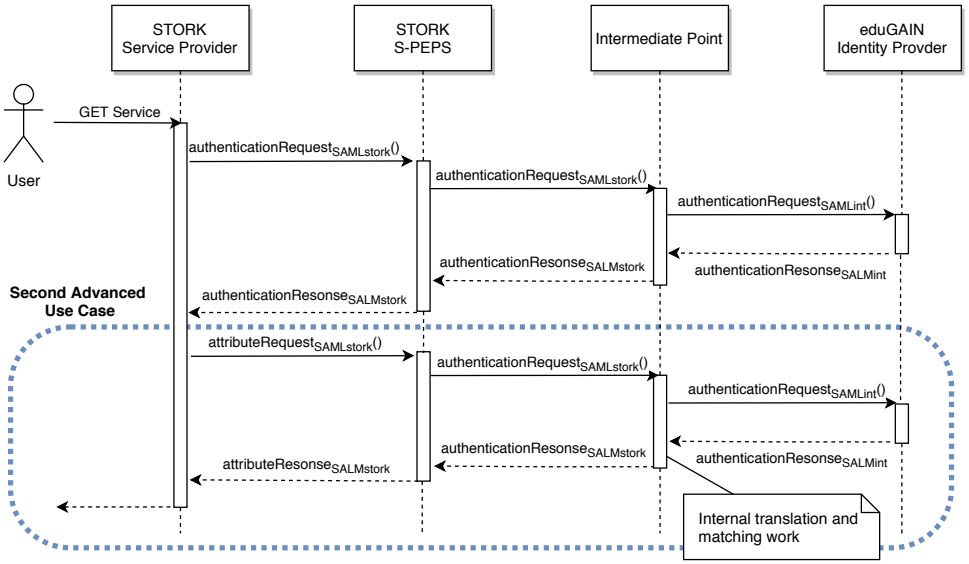


Figure 4.10: STORK SP requests additional attributes from eduGAIN IdP

Second use case: request of additional attributes

In this use case (Fig. 4.10), the objective is to recover additional attributes from eduGAIN IdP using the already translated STORK identity. According to the function selected in the base use case, the eduGAIN identifier should be able to be recovered.

On the STORK side, the S-PEPS is waiting for a STORK authentication so the Intermediate Point must be able to return a complete and new STORK authentication response. Since STORK does not maintain sessions and expects full authentication, there is no way to distinguish the second request from any other new authentication request. It does not include information to maintain and restore a session at the Intermediate Point that allows using the initial authentication information to directly request more attributes to eduGAIN IdP, as happens in pure eduGAIN scenario. Therefore, the Intermediate Point has to run a new authentication request including the required new attributes, and the eduGAIN user is going to be required to authenticate at his eduGAIN IdP again. The S-PEPS will obtain an authentication response which is returned to the STORK SP. In this use case, where the STORK SP asks for additional attributes is the SP, it is the Intermediate Point which has to check if both authentications return the same eIdentifier.

Summary results

As in the first scenario, Scenario 2 (Fig. 4.11) also requires defining and including the Intermediate Point to do the translation work between eduGAIN and STORK federations.

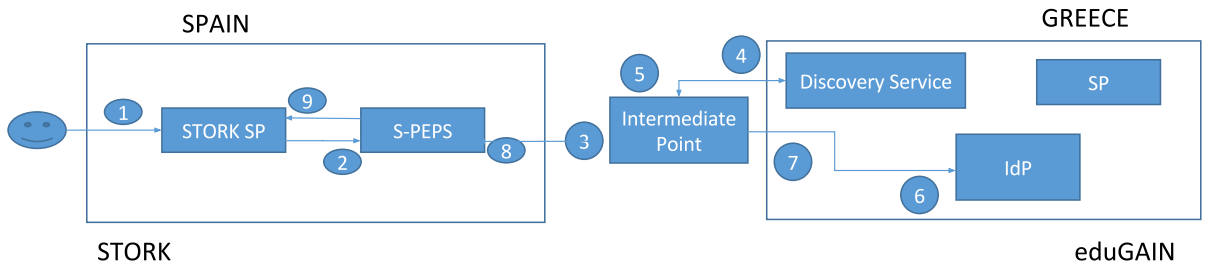


Figure 4.11: eduGAIN user accessing an STORK SP flow schema

This intermediate entity allows both federations to be interconnected transparently, including only this new entity as a trust point for both sides.

4.8 eduPEPS implications

From the study of the different scenarios and use cases raised in Sections 4.3 and 4.7 there follows the need for an intermediate element that performs the functions of translation and interconnection between eduGAIN and STORK.

Our proposed solution is to incorporate a new intermediate entity, denominated from now on eduPEPS, situated between the STORK and eduGAIN federations. This entity, which was presented in [160], would be in charge of the SAML translation (between $SAML_{STORK}$ and $SAML_{Int}$) as well as the translation of identifiers and attributes sets from both federations. The eduPEPS have also to act as a trusted entity for both federations, providing credibility and validity to the information provided to services by the providers from the other federation.

Although the entities at the endpoints are common in both federations (SPs and IdPs), the entities in charge of the infrastructure are different. The eduPEPS assumes the differences, allowing us to maintain the operation and security requirements of the endpoints of each federation. EduGAIN relies, on the one hand, on the MDS to manage and provide reliable metadata and, on the other, on the DS to locate the origin of the users. In STORK, the discovery function is generally performed by the SPs (as seen in Section 2.3.4), but the SP's PEPS is responsible for redirecting the user to her national PEPS and acts as centralized point of trust for each Member State, establishing one-to-one relations with SPs and IdPs on the one hand, and with the rest of national PEPSes on the other. In this way, from the point of view of eduGAIN, the eduPEPS will act as a DS and any MDS, while from STORK's point of view, it will act as a PEPS from another Member

State (both S-PEPS and C-PEPS pending on the scenario).

The eduPEPS element allows us to incorporate the concept of *Virtual Country* applied to the eduPEPS from the STORK point of view. This perspective allows us to treat eduGAIN federation as a set, that is, as a virtual country with the same security requirements and obligations as the other Member States. In STORK, each country needs legal agreements to be trusted by each national PEPS, so through this equivalence each STORK Member State can decide if it allows the eduGAIN integration or not. This concept opens up a wide range of possibilities at the same time as it simplifies in a conceptual and administrative way the management of trust and security, mainly for the STORK federation.

In close relation with the concept of Virtual Country, the eduPEPS can be located in different points between eduGAIN and STORK federations. These locations determine the functions the eduPEPS has to implement, and the role and visibility that eduPEPS offers to other entities of both federations. As an example, the eduPEPS can be located as PEPS level or just as IdP module doing only proxy functions. The following subsections provide a wide analysis of all possibilities and the implications for each case.

Among the different possibilities to locate the intermediate entity, the following are the best alternatives that have been considered.

4.8.1 Intermediate entity as PEPS adapter

STORK PEPSes are controlled by the governments of each Member State and act as gateways to their identity providers (authentication and attribute provision). Locating the adaptation functions as extension modules of the PEPS allows control to remain in the hands of each Member State. This facilitates the adaptation and maintenance, and helps to ensure the compliance with the strict STORK safety requirements. In this situation, the eduPEPS not only has to translate between both SAML dialects (between $SAML_{STORK}$ and $SAML_{Int}$), but it also has to cover all the differences of functioning between federations, as discussed above.

With this approach, any STORK Member State that wishes to allow eduGAIN users to access their services or wishes to provide authentication for eduGAIN services to their users will need to implement an adaptation module in their national PEPS and manage their eduGAIN trust relationships with independence of other Member States. Figures 4.12 and 4.13 depicts uses cases from both perspectives: STORK and eduGAIN users using eduGAIN and STORK services respectively.

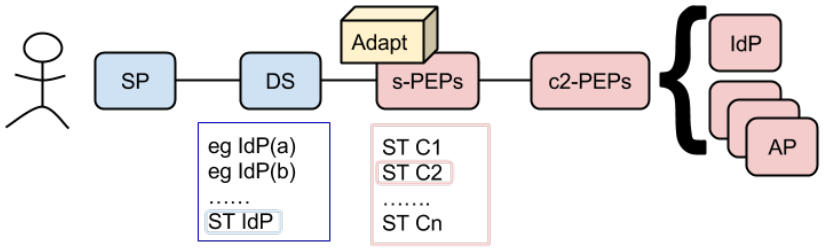


Figure 4.12: Use of an adapter in the PEPS when a STORK user accesses an eduGAIN SP.

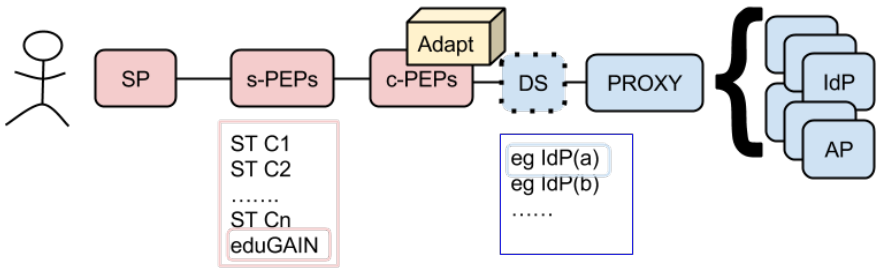


Figure 4.13: Use of an adapter in the PEPS when an eduGAIN user accesses a STORK SP.

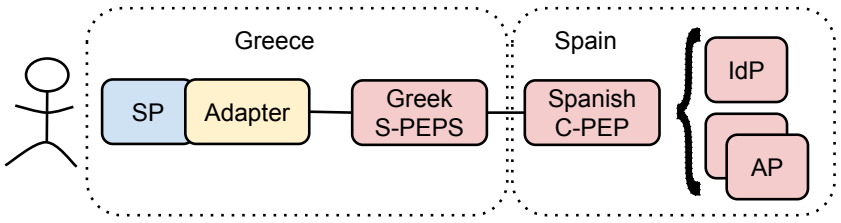


Figure 4.14: eduGAIN SP connected to STORK federation through a proxy.

4.8.2 Intermediate entity as proxy module for SPs and APs

With this approach, the intermediate entity is designed as a proxy module in the endpoints of both federations. The adapter can be deployed to work with SPs and Authentication and Attributes Providers of both federations. With this mechanism, the origin or destination of the service is hidden by the adapter and is transparent to all the other components. From the point of view of STORK, this transparency could be a disadvantage since it implies a loss of control and even a security gap, since the endpoint behind the adapter would be outside the STORK federation.

In the case of the adaptation to STORK federations of eduGAIN SPs and IdPs, the adapters have to have been registered and accepted by some national PEPSes, so the process is similar to be followed by a native STORK service. On the other hand, the case of STORK SPs and IdPs interaction with eduGAIN federation, the adapter has to be registered at DSs and MDSs in order to be available to the eduGAIN federation.

This solution really only makes sense if it is offered to the endpoints as an already developed module that brings together all the necessary functions with the only requisite of being configured to the specific case, since a development from scratch would be measureless in most cases to an individual provider.

As example, the Figure 4.14 depicts a Spanish STORK user visiting a Greek University belonging to eduGAIN. For this scenario, the adapter is located in eduGAIN SP and has to be registered and accepted by the Greek PEPS.

4.8.3 Independent intermediate entity: eduPEPS

Another possible approach is to introduce a complete new entity that performs all translation and adaptation functions as we already stated at the beginning of this section. The new entity, the eduPEPS, would be located between STORK and eduGAIN federations and would act as a trust entity for both federations.

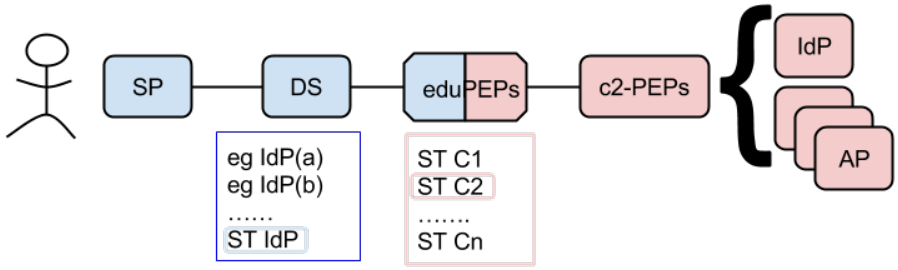


Figure 4.15: Use of eduPEPS for eduGAIN SP consuming STORK identity flow diagram

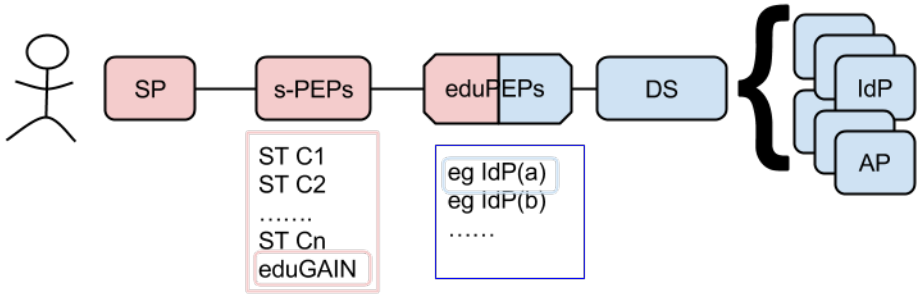


Figure 4.16: Use of eduPEPS for STORK SP consuming eduGAIN identity flow diagram

When a STORK user accesses an eduGAIN service (see Figure 4.15), the eduPEPS plays the role of an S-PEPS from the point of view of STORK and the IdP role from the point of view of eduGAIN. In addition, to translating the queries of the eduGAIN SP and the responses from the STORK IdP to be understandable by each of the pairs, the eduPEPS shows the user country selection options and manages the session level differences between both federations. The eduPEPS could also fulfill the role of the MDS and handle the dissemination of STORK metadata across the entire eduGAIN federation.

In the case of an eduGAIN user accessing a STORK service (see Figure 4.16), eduPEPS has to act as an SP from the point of view of eduGAIN and as a C-PEPS from STORK's point of view, with the particularity that instead of redirecting the user directly to his IdP, the eduPEPS has to provide the list of available eduGAIN federations supplying the role of DS. This function can be done in several ways, by redirecting the user to an eduGAIN own DS, by importing the metadata from an MDS or by directly offering a list of supported IdPs. This last approach entails certain disadvantages such as the large number of possibilities usually available to the user in eduGAIN that can be a bit unusable. In the role of C-PEPS, the PEPS interacts with the STORK SP through the SP's S-PEPS. The eduPEPS has to translate STORK request to eduGAIN IdP, and translate and sign the

eduGAIN IdP's responses, before sending them back to STORK S-PEPS.

STORK PEPS performs the role of central node in each country, establishing trust relationships with all service and identity providers within that country and acting as a bridge with the rest of PEPS from other countries. From this point of view, each PEPS represents a country, and based on this concept, we plumb a parallelism to use the eduPEPS as an entry point to the federation of eduGAIN. Thus, eduGAIN and its services can be considered as a kind of virtual country, using eduPEPS as a means to manage access to it, and this allows us to delimit the adoption of the integration between eduGAIN and STORK to the level of trust that each pure STORK PEPS give or do not give to the eduPEPS.

Although the number of eduPEPSes is neither limited nor determined, if only a few of them (let's say one) are to be deployed, only a few (one) trust relations would need to be created between each STORK participating country and the eduPEPS service. The eduPEPS could optionally feed and be fed with the MDS. In this case, the eduGAIN entities could retrieve the information of the different STORK entities and would be able, for instance, to provide multiple STORK entries in their dialogs, which would avoid one more step of selecting the country on the eduPEPS (as it would have been already selected on the eduGAIN federation DS), although the eduPEPS must continue working as relay and translator for any eduGAIN request. Furthermore, each STORK AP (Attribute provider) could be directly listed as an option for the user. This approach introduces complications regarding the different eduGAIN and STORK approaches.

When the user belongs to a STORK federation and accesses an eduGAIN service, the eduPEPS takes the role of a S-PEPS. Apart from translating the eduGAIN requests and STORK responses being understandable by each peer, the eduPEPS shows a country selection options to the user and manages the session level differences between the two worlds. The eduPEPS could in this case also update the eduGAIN level MDS which would lead to STORK metadata dissemination through the whole eduGAIN federation. The metadata dissemination could serve for various optimization mechanisms in the near future.

As for how the eduPEPS would interact with eduGAIN in the case of eduGAIN users visiting STORK services, the eduPEPS would act as C-PEPS with the exception that instead of redirecting the user directly to its IdP, the eduPEPS will provide a list of available eduGAIN federations and redirect the user to the selected federation DS. An enhancement to this approach would be that the eduPEPS imported the eduGAIN metadata from the MDS and directly offered all the IdP possibilities, so avoiding the use of DS. This approach, despite its benefits, has some drawbacks, like the large number of possibilities shown to

the user that could lead to a non human friendly system.

Within this approach, multiple eduPEPS could be instantiated at different levels (depending on operational or even political decisions) in order, for example, to classify eduGAIN services and IdPs according to the level of assurance (LoA) offered, since this is an important requisite in STORK federations. Nevertheless the idea of reducing to a minimum the eduPEPS instances is driven by the desire to reduce the number of agreements needed by each STORK federation, which are essentially hard and fatiguing to accomplish.

4.9 eduPEPS validation

The integration possibilities between eduGAIN and STORK have aroused great interest along the STORK 2.0 project, as well as from GÉANT community, receiving specific support for deploying and doing integration tests from GN4-1 and currently from GN4-2. In this context, the University of Murcia, as a GN4 member, has collaborated with GRNET (Greek Research & Technology Network) to create a crossborder, more realistic and worldwide testbed. The University of Murcia has a wide knowledge of STORK 2.0 architecture since it has participated as a partner in the STORK 2.0 eLearning Pilot, working on university eLearning platform connection with the Spanish PEPS as well as deploying the TADS (Trusted Attribute Display Service) and contributing with STORK as attribute provider. GRNET is also an active member of eduGAIN federation, thus it contributes with the eduGAIN elements to the testbed, as shown in Fig. 4.17.

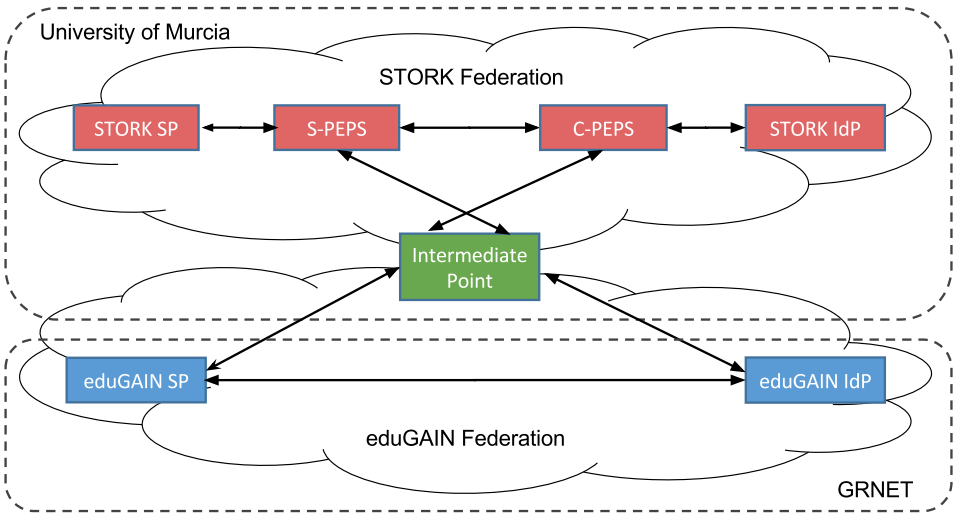


Figure 4.17: Testbed schema

The testbed objectives are, in the first place, to demonstrate the viability of the solution proposed and, based on this, to compare how user interaction complexity is affected when STORK and eduGAIN federations are joined together in integrated scenarios. The quality of experience (QoE) for users is an important factor to estimate the adoption level of new services and systems. In this case, we will consider the number of user interactions needed to complete the authentication process as a representative factor for the QoE. We define one user interaction as the required intervention of the user to select or accept flow steps; In general, user interactions match with each one of the different web forms that are shown to the user along the access to the service and the authentication process. More user interactions imply more complexity and time duration, together with the reduction of the usability and of the users interest.

4.9.1 Testbed description

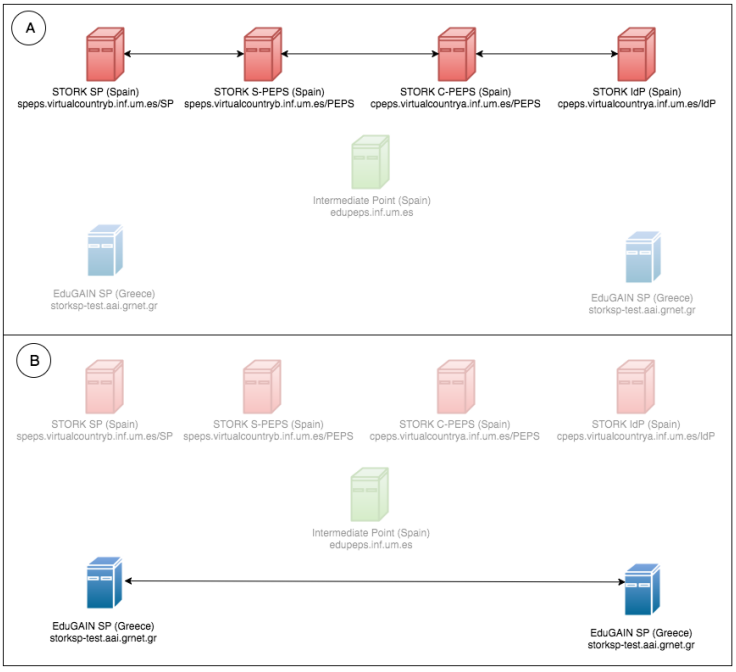
The designed testbed has two parts, one for deploying the STORK elements, and the other for deploying the eduGAIN elements. The STORK part (in red in Fig. 4.17) has been deployed at the University of Murcia and contains all the required elements to run a minimal STORK scenario: SP, S-PEPS, C-PEPS and IdP. On the other hand, the eduGAIN part has been deployed at GRNET (in blue) and includes one SP and one IdP, while the Discovery Service element is embedded in the SP. Both federations are interconnected through the Intermediate Point that acts as a proxy and is in charge of translating between SAML dialects and identifiers (in green). This element is deployed together with all STORK elements at the University of Murcia.

The demonstration considered four different scenarios, two pure and two integrated. Fig. 4.18 depicts all the configurations studied: Pure STORK (A), Pure eduGAIN (B), STORK SP & eduGAIN IdP (C) and eduGAIN SP & STORK IdP (D).

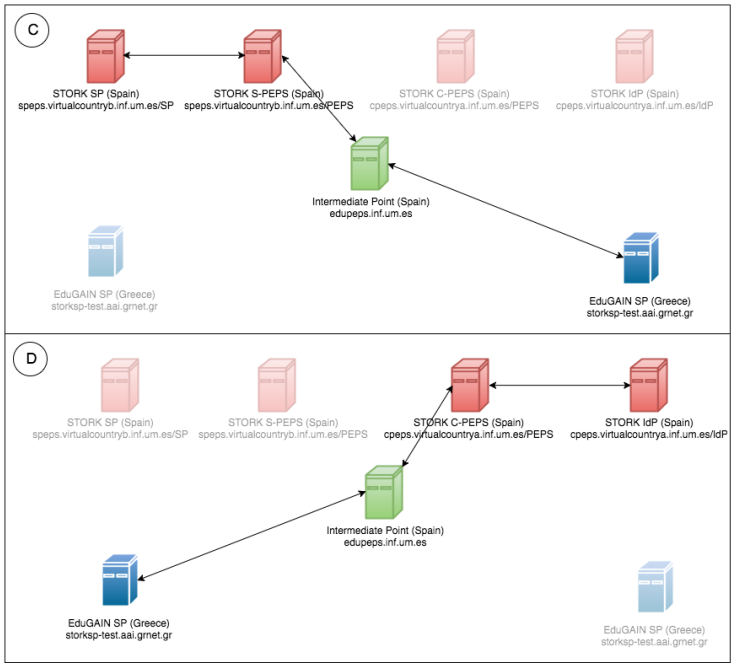
For each of these scenarios, we have analysed two use cases studied along Section 4.7 which are: “Basic authentication” (use case 1) and “Authentication and additional attribute recovery” (use case 2). The matching extension (for both scenarios and use cases) does not produce additional interactions since it only implies SP’s internal work, so it is not consideedr for the testbed.

4.9.2 Results

The testbed has to demonstrate the viability of the solution proposed as well as the quality of the experience from the user point of view. The prototype deployed offers a complete



(a) Standalone scenarios: Pure STORK & Pure eduGAIN



(b) Integrated scenarios: STORK SP – eduGAIN IdP & eduGAIN SP – STORK IdP

Figure 4.18: Testbed schema

testbed that integrates all the elements involved in both eduGAIN and STORK. The deployment has even been designed by integrating different institutions and locations with the aim of being more realistic. The results obtained demonstrate the real possibility of performing a complete authentication process between both federations while keeping the principles and requirements of each federation separate and, therefore, the viability of the solution is proved.

From the quality of experience (QoE) point of view, the testbed analyzes the number of user interactions required in order to measure the QoE since it is an important factor to estimate the adoption level of new services and systems. With this aim, the testbed considers the studied use cases, and for each one, four different configurations: standalone STORK, standalone eduGAIN and both integration scenarios. Table 4.7 provides the required user interaction data while Fig. 4.19 depicts the graphical representation of the results obtained. Table 4.8 shows how much the integrated scenarios are affected compared to specific standalone scenarios.

Numbers of user interactions	Pure STORK	Pure eduGAIN	eduGAIN SP & STORK IdP	STORK SP & eduGAIN IdP
Authentication (UC1)	5	4	6	5
Authentication + additional attribute recovery (UC2)	9 (5+4)	5 (4+1)	11 (6+5)	9 (5+4)

Table 4.7: User interaction results

User interactions	Pure STORK compared with eduGAIN	Pure eduGAIN compared with STORK	eduGAIN SP & STORK IdP		STORK & eduGAIN IdP	
			Compared with STORK	Compared with EduGAIN	Compared with STORK	Compared with EduGAIN
Authentication (UC1)	5 +25%	4 -20%	6 +20,00%	6 +50,00%	5 0,00%	5 +25,00%
Authentication + additional attribute recovery (UC2)	9 +80%	5 -44,5%	11 +22,22%	11 +120,00%	9 0,00%	9 +80,00%

Table 4.8: Comparative of user interaction number for each use case at each scenario using as base both pure scenarios.

Usage flow through STORK federation requires from the beginning a greater number of user interactions, since the user must consent (authorize) the attribute retrieval and

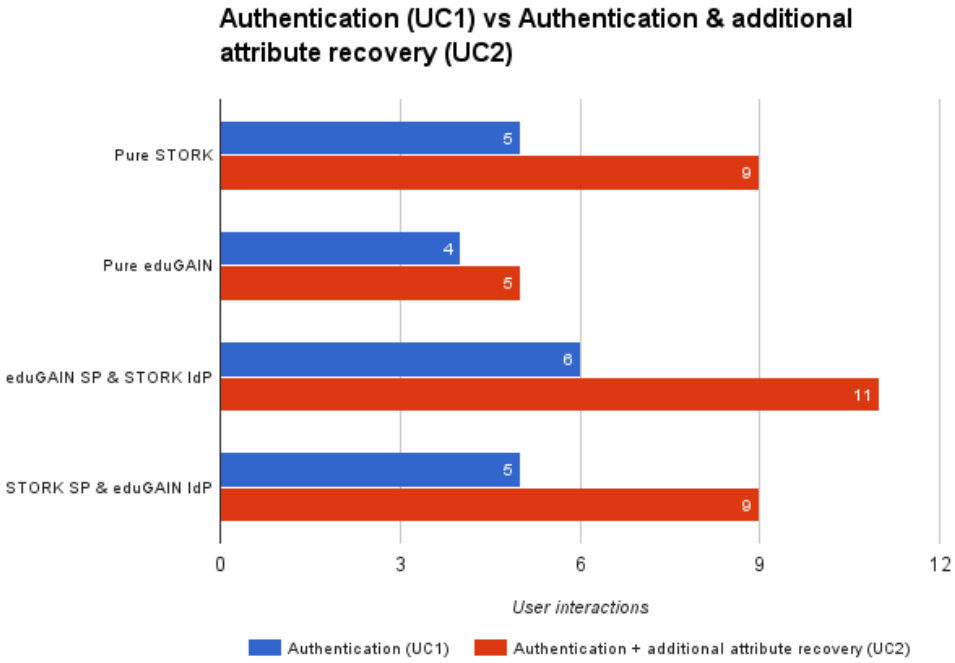


Figure 4.19: Graphical representation of user interactions at different use cases and scenarios

distribution, which is not a requisite in eduGAIN federation. In turn, each STORK access to retrieve attributes requires user authentication to be performed again and the whole consent process to be repeated. These differences mean that STORK has an overhead of 25% for basic authentication and 80% when an additional attribute recovery with respect to eduGAIN is necessary (Table 4.8).

The integrated scenarios have to meet the requirements of both federations so it can be seen that in both use cases the interaction number is equal (STORK SP & eduGAIN IdP scenario) or only one interaction more (eduGAIN SP & STORK IdP scenario) than in pure STORK.

If the starting point for the comparison is pure eduGAIN, the additional required user interactions in the integration flows are many more. Under this point of view, the overhead reaches 50% in basic authentication (UC1) and rises to 120% in the UC2, meaning more than twice the interaction numbers. These increased overheads compared to pure eduGAIN scenario are due to the process of recovering additional attributes being simpler in eduGAIN where the user does not have to choose again the identity provider or make any additional authorization, while in STORK the user has greater control about which attributes are accessed and for whom.

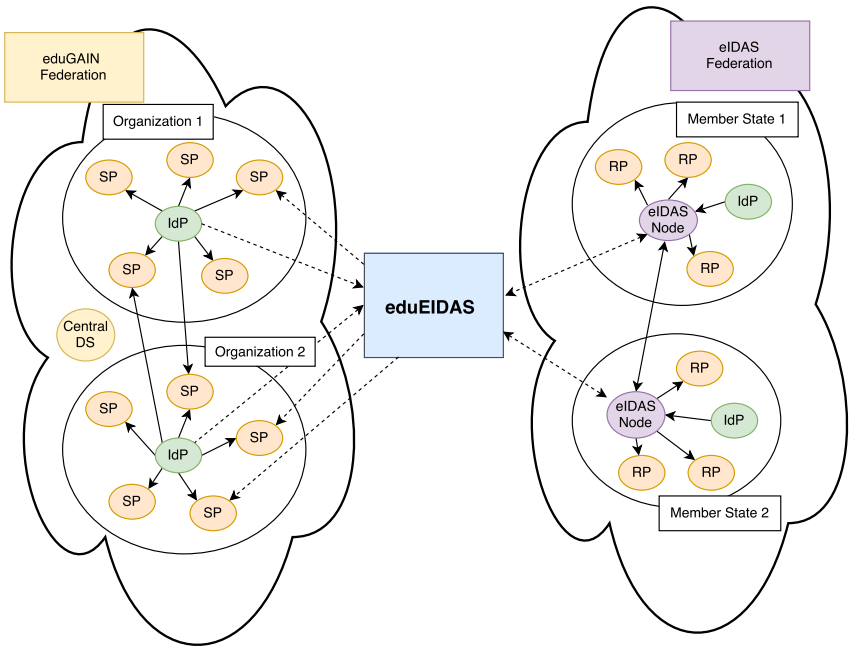


Figure 4.20: eduGAIN and eIDAS interfederation through eduEIDAS.

4.10 Evolution to EIDAS

eIDAS, as an evolution of the STORK project, has similar peculiarities and problems to interact with eduGAIN. The differences in SAML dialects, identifiers and attributes, together with high security and legal requirement and restrictions make this impossible.

As with STORK, the introduction of a new intermediate entity (Figure 4.20) enables all technical problems of compatibility between federations to be solved. The new entity, named in this case eduEIDAS, also has to translate between SAML_{INT} and $\text{SAML}_{\text{STORK}}$ dialects, translate and make the matching between attributes and identifiers, and act as trust point for both federations, so all the mechanisms and tools designed and developed for the previous integration scenario can be used as a base here.

In eIDAS, the central element of the architecture is the eIDAS Node, which has similar characteristics to the STORK PEPS. In general, the eIDAS Nodes is the central point of trust and interconnection of each Member State and is used as gateway in the cross-border operations. As we saw in the eIDAS section, it has two different roles depending on whether it acts as Connector or Service Node.

In the case of an eduGAIN user accessing an eIDAS RP, the eduEIDAS plays the role of the Service Node from the point of view of eIDAS and acts as a SP that requests user

authentication from the eduGAIN IdP. In the case of an eIDAS user accessing an eduGAIN SP, the eduEIDAS can play the role of DS and MDS and replace the IdP. From the point of view of eIDAS, it acts as Connector Node of the Receiving Member State.

eIDAS architecture also allows the use of the Virtual Country concept introduced in the previous scenario, which simplifies the trust management and allows selective adoption by each Member, since in eIDAS each country needs legal agreements to be trusted by each national eIDAS Node.

4.11 Summary and conclusions

Nowadays there is intense work in the area of AAI to improve the infrastructure uses and the services deployed over them. STORK 2.0 project from the governmental point of view and GÉANT with eduGAIN throughout its different phases GN2-GN4 from the educational and research point of view, are good examples of this effort. In spite of pursuing similar goals, each federation chooses its own tools to build their architectures. Even if both eduGAIN and STORK take SAML as common starting protocol, they are not directly compatible between them because, among other things, they are based on different profiles and identifiers.

A wide analysis of different integration possibilities has been made. We have taken into account the migration to other protocols such as OAuth or OpenID Connect as well as the use of existing federation mechanisms like Shibboleth, but all of them imply modifying all the entities involved and increasing the complexity of the operation and legal agreements between entities. Designing a customized solution allows us to minimize modifications to existing deployments and adapt the solution to the maximum to maintain control of information, privacy and trust.

Based on the previous collaboration and deep knowledge of the operations and specific needs of both federations, our research group has focused on fulfilling the harmonization interest of both federations, working on interconnection arrangements between them and proposing the identity management mechanisms needed to carry this out. The demonstration phase has focused on proving the interoperability between identifiers and users accounts, and measuring the complexity and quality of the solution proposed from the user point of view.

As an initial step, it was necessary to establish the equivalence for the roles played between federation entities, the trust relationships and the security requirements in each federation. The proposed integrated scenarios achieve the main security requirements

in both federations, among which stands out the obligation of the user to consent to each request and release her information as STORK established. This requirement has conditioned, as has been shown in the Validation section, the differences between scenarios.

Next, we analyszd eduGAIN and STORK identifiers to establish differences and similarities, studying interoperability mechanisms between them from both perspectives: a STORK user that tries to access an eduGAIN service, as well as an eduGAIN user that tries to access an STORK service. The biggest complexity at this point is the use of different identifiers and attribute sets. To solve this, we established translation mechanisms that allow it, even if there is no direct equivalence, and that recover the original identifier to request additional attributes. Our proposal also considers the possibility of making a matching between existing user profiles at SP using the user identity information recovered from the other federation.

The Validation Section has used the differences in the number of user interactions as a metric. This number has a direct relation with the complexity, usability and, in general, with the QoE and, therefore, user interest. The tests show that the increment for STORK users is minimum (from 0 to 22%), in contrast to that for eduGAIN users, where the increment rises to 120%, so there is a considerable overload. These bad comparative results of integrated scenarios when it is offered to eduGAIN users limits the interest in STORK integration. However, the integration provides the opportunity of providing eduGAIN users with access to a wide range of government and official services from STORK. For eduGAIN institutions (SPs), the integration with STORK federation could be very interesting in specific scenarios where STORK authentication can replace face procedure or where a high degree of STORK information assurance level is required. For STORK users the integration is very interesting in general, since the increase in the service offer is huge and the new flow is very similar to STORK standard flow, in exchange for a small increment of use cost. For STORK Service Providers the main incentive is to gain access to a huge eduGAIN user base with the limitation of different levels of assurance in the authentication process.

Since institutions have demonstrated great interest in federation harmonization, the authors aim to help in this sense by continuing our research in this topic by working on attribute mapping definition, with special attention to the security and privacy assurance. In addition, the legal and philosophical implications on where and how the Intermediate Point is deployed are to be analyzed shortly. Finally, STORK has its continuation in the eIDAS regulation, therefore it is necessary to analyze how it affects the integration proposed and advances in that direction to align the integration with new federation normative.

Chapter 5

Conclusions and Future Work

People and services are increasingly demanding greater interconnection to unite and offer common services as a way of winning new users, improving existing services and simplifying the management. There is a wide variety of identity management and identity federation systems according to the requirements of users, services and scopes, and the most active work in this IT area is being done by the research and administration sectors. The latest trends and advances focus on guaranteeing user privacy and improving security, and even on enhancing the improvement of existing federations through their interconnection, which allows a larger user base, unified services and management, and creates a set of new services in line with the new possibilities.

In order to appreciate the intense work done in the area of AAI to improve the infrastructure used and the services deployed, this thesis has reviewed some of the most relevant authentication and authorization protocols and, in particular, the most used identity federations based on SAML.

From the point of view of identity federations, we analyze Moonshot, first as a general purpose option, continuing with eduGAIN and EUDAT, which are very important in education and research sectors. Finally, the review places special emphasis on options related to the Administration sector, such as STORK and eIDAS. The review offers a general overview of the main characteristic of each federation.

Based on the identity federation analysis, we set out the problem of interconnecting federations in specific scenarios where the federations are already deployed and running. This thesis attributes the main problems to the differences in communication protocols, attributes sets and identifiers, as well as to the exiting differences in security restrictions and requirements. The interoperability between federations offers a wide range of benefits such as the huge increase in the number of users, the improvement of existing services and

the creation of new ones, as well as multiple security improvements. The definition and implementation of interoperability mechanism is critical for the successful advancement of digital technologies.

The first contribution consists of collaboration in the definition of a new complete identity federation named SWIFT, with the focus on identity management enablers, which has allowed advanced functions for users' privacy, offering fine control of disclosure of private user information, anonymity, advanced policy definition for access control and cross-layer SSO; all from a user centric perspective. As a result, an extended IdM architecture has been designed. It contains security and privacy mechanisms addressing requirements of future IdM solutions. In particular, we have provided a cross-layer privacy solution that enhances existing work and a new access control infrastructure. To validate the solution, all the information and messages interchanged have been translated to a formal notation and based on this, a complete prototype of all entities has been implemented and tested as functional proof of operation.

The next step was to work on improving the trust control in existing federations. The objective here was the definition of interoperability mechanisms to provide security and trust control through heterogeneous security systems connected between a common enterprise service bus (ESB). We propose the integration of WS-* family standards with OAuth 2.0 protocol to reach the interoperability between traditional SOAP services based on SAML2.0 and X.509 certificates and new REST services based on OAuth. This has been applied to the definition and implementation of the Security Service inside GEMBus Project. It is translated to a Security Token Service (STS) made up of the Token Translation Service and the Validation Service. Therefore, the STS allows the translation between different security protocols and the use of different communication interfaces (SOAP and REST) with the aim of interconnecting services and users from heterogeneous origins. The Security Service generates new security statements and validates them, offering a central trust point for services. The validations can be done by the Security Services themselves or delegated to the original issuer of the security statements. In this case, together with the implementation of a complete and full functional pilot, we have made time measurements to evaluate the increase of time due to the integration functions.

Finally, the research has gone a step further toward the interfederation level. As we have seen throughout this work, identity federations are isolated from others. In general, they are designed to integrate individual services but not to interconnect complete federations. To carefully study and analyze the problem, we focused on two important and popular federations, that have the particularity of having different areas of use, which made their interconnection especially interesting due to the great benefit that the integration implies

for both. EduGAIN, from the point of view of educational and research institutions, and STORK, from the governmental point of view, were the identity federations chosen to work with on their interconnection. Migrating from one federation to other or establishing a complete new interoperability layer between both is not practical, due to the size and level of the deployment. Although both federations share points in common, they use different security restrictions, architectures, security protocols, identifiers and attribute sets. The objective here is to establish the necessary mechanisms to provide the interaction between users and services of both federations in a bidirectional way while maintaining the necessary security and privacy requirements as transparently and simply as possible, since both federations are in production. To achieve this, an intermediate entity, the eduPEPS, was designed to act as a trust and translation point between both federations. This entity is in charge of simulating the proper functioning of each federation as well as the translation of messages, identifiers and attributes, while providing credibility and validity to the information provided to services by providers from the other federation. In addition to the deployment of an international pilot between Greece and Spain to test the viability and feasibility of the solution proposed, we have evaluated the quality of experience offered to the end users as an illustrative measure that anticipates the success of adoption by users.

In all the cases the validation process has been focused on measuring the usability and the quality of experience (QoE) from the user point of view, taking into account the importance of user interfaces, as a measure that anticipates the success of the designed solutions and their future adoption capacity in order to go a step beyond merely only the technical viability. In the tests, we see that the proposed solutions add internal complexity to the authentication and authorization flows that can affect the user experience, for example, by adding new steps to the interaction flows. Even so, we consider that the advantages offered by the proposed solutions in terms of simplification of credentials management, new service possibilities, replacing face-to-face procedures, among others, compensate for the increases in processing times or the need for additional steps, which we also consider can be optimized in the future.

To summarize the work done throughout this thesis to improve digital identity management through the interoperability of heterogeneous AAIs at different levels, we offer a list of the main contributions:

- To improve Identity Management capabilities: we have analyzed the main authentication and authorization protocols in the area of IdM, in order to propose advanced mechanisms for IdM architectures, being the main features: partial identities, anonymity functions, better privacy and access control policies (including

Deductive Policies). This latter contribution was sent as proposal to XACML 3.0 standardization group. Finally, we have defined SWIFT architecture using formal language, and used it as base for the complete implementation of a functional prototype of the architecture to its validation.

- To improve Trust Management: we have implemented the WS-Trust library to use it in the implementation of the GEMBus Security Service based on Security Token Service. To improve the trust management, we have proposed an interoperability solution to integrate WS-Trust and OAuth security standards that have enabled the interaction between heterogeneous authorization technologies. Finally, a complete GEMBus pilot has been deployed to validate the architecture and the interoperability proposal.

- To improve interfederation interoperability: we have reviewed the main identity federations in the area of in the area of education, research and government institution, identifying the main requisites and open challenges in the identity federation interoperability problem. We have analyzed and compared eduGAIN and STORK 2.0 federations, in order to define the required interoperability mechanisms to reach their integration, as result we have proposed the introduction of the eduPEPS. Finally, we have validated the integration proposed thought the eduPEPS implementation and the deployment of complete testbed.

Since institutions have demonstrated great interest in federation harmonization, as future work the authors aim to help in that direction by continuing the research into this topic by working on attribute mapping definition, with special attention to security and privacy assurance. Furthermore, the legal and philosophical implications on where and how the eduPEPS is deployed are to be analyzed in detail.

GÉANT consortium, with which we collaborate through RedIRIS, is now involved in GN4-2, a 32-month project beginning 1^{st} May 2016 and ending 31^{th} December 2018. Inside the project there are specific tasks working on interoperability and federation harmonization solutions. In particular, the project analyses the interoperability possibilities between United States and Europe through the interoperation of InCommons federation (based on eduGAIN) and eIDAS federation respectively. There are ongoing contacts among several partners to consolidate this goal in an international project called

ALPHA. For our part, as we introduced at the end of Chapter 4, we consider that eduPEPS solution can be extrapolated to the integration between eIDAS and eduGAIN.

Although eIDAS is based on STORK, eIDAS has its own characteristic that requires a personalized solution. Therefore, we propose the incorporation of a new entity that allows the translation mechanisms required in this new scenario to solve all the technical interoperability already shown with the eduPEPS. It will be necessary to analyze in detail how it affects the integration possibilities and to advance in that direction to align the integration with the new federation normative.

Identity federation interoperation allows better privacy and security control of user personal data, so improving and simplifying the management and the interaction with service providers. Public administrations are relatively new actors in the IT world, with great potential to harmonize existing federations, with the guarantee of being able to offer high quality information for services and at the same time maintain the utmost respect for the privacy of users.

Bibliography

[1] Moonshot. Moonshot WIKI, 2015.

[2] Thomas Baerecke. eduGAIN Federation Architecture. Technical report, G{É}ANT, 2014.

[3] EUDAT. B2ACCESS Project webpage, 2016.

[4] Tomaž Klobučar, Dusan Gabrijelcic, and Vladimir Pagon. Cross-Border e-Learning and Academic Services based on eIDs: Case of Slovenia. elfast, 2014. eChallenges.

[5] L Hämmerle, R Sabatino, T Lenggenhager, M Mantovani, P Pilt, L Toom, L Jensen, Elena M Torroglosa, Stefan Paetow, Peter Solagna, Willem Elbers, Andrea Ceccanti, Bas Wegh, Marcus Hardt, Paul Millar, and Johannes Reetz. GN4 - 1 White Paper : Comparison of Authentication and Authorisation Infrastructures for Research. Technical report, 2016.

[6] Elena M. Torroglosa-García and Antonio F. Skarmeta-Gómez. Towards Interoperabilty in Identity Federation Systems. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 8(2), 2017.

[7] Elena Torroglosa-Garcia, Antonio D Perez-Morales, Pedro Martinez-Julia, and Diego R Lopez. Integration of the OAuth and Web Service family security standards. *Computer Networks*, 57(10):2233–2249, 2013.

[8] Alejandro Pérez, Elena María Torroglosa, Gabriel López, Antonio F Gómez, Joao Girao, Mario Lischka, and Mario. Pérez, Alejandro; Torroglosa-Garcia, Elena; López, Gabriel; Gómez-Skarmeta, Antonio; Girao, Joao; Lischka. SWIFT – Advanced Services for Identity Management. *Upgrade*, XI(1):13–20, 2010.

[9] Alejandro Pérez, Elena María Torroglosa, Gabriel López, Antonio F Gómez, Joao Girao, and Mario Lischka. SWIFT - Servicios avanzados para la gesti{ó}n de identidad. *Novática*, pages 1–19, 2009.

[10] Mary Grammatikou, Costas Marinos, Pedro Martinez-Julia, Jordi Jofre, Steluta Gheorghiu, Diego R. Lopez, Yuri Demchenko, Krzysztof Dombek, Roland Hedberg, Antonio F. Skarmeta, and Elena Toroglosa. Composable Network Services Framework : GÉANT Multi-domain Bus (GEMBus). *18th International Conference on Parallel and Distributed Processing. Techniques and Applications*, 2012.

[11] Elena Torroglosa, Alejandro Pérez, Gabriel López, Antonio F Gómez-Skarmeta, and Oscar Cánovas. SWIFT: Advanced identity management. In *Proceedings of the 5th International ICST Conference on Communications and Networking in China*, pages 1–5. IEEE, 2010.

[12] Marc Barisch, Elena Torroglosa Garcia, Mario Lischka, Rodolphe Marques, Ronald Marx, Alfredo Matos, Alejandro Perez Mendez, Dirk Scheuermann, Elena Torroglosa Garcia, Mario Lischka, Rodolphe Marques, Ronald Marx, Alfredo Matos, A Perez Mendez, Dirk Scheuermann, Alejandro Perez Mendez, Dirk Scheuermann, A Perez Mendez, and Dirk Scheuermann. Security and Privacy Enablers for Future Identity Management Systems. In *Proceedings of the Future Network {&} MobileSummit (FutureNetw'10)*, pages 1–10. IEEE, 2010.

[13] Mario Lischka, Yukiko Endo, Elena Torroglosa, Alejandro Pérez, and Antonio G Skarmeta. Towards Standardization of Distributed Access Control. *Computers & Education*, 51(1):0–4, 2009.

[14] Elena M Torroglosa García and Gabriel López Millán. Web Service Security: Authentication and Authorization Technologies. In *Network Security Technologies: Design and Applications*, chapter 8, pages 108–128. 2014.

[15] Yuri Demchenko, Canh Ngo, Pedro Martínez-Julia, Elena Torroglosa, Mary Grammatikou, Jordi Jofre, Steluta Gheorghiu, Joan A Garcia-Espin, A D Antonio D Perez-Morales, and Cees De Laat. GEMBus based services composition platform for cloud PaaS. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7592 LNCS, pages 32–47, 2012.

[16] Antonio F Gomez-Skarmeta, Alejandro Perez Mendez, Elena M Torroglosa García, and Gabriel López Millán. User-Centric Privacy Management in Future Network Infrastructure. In *Privacy Protection Measures and Technologies in Business Organizations: Aspects and Standards*, chapter 2, pages 32–64. 2012.

[17] Nb Boyd, DM & Ellison. Social Network Sites: definitions, history, and scholarship. *Journal of Computer-Mediated Communication*, pages 1–11, 2007.

[18] E F Gehringer. Choosing passwords: security and human factors. In *IEEE 2002 International Symposium on Technology and Society (ISTAS'02). Social Implications of Information and Communication Technology. Proceedings (Cat. No.02CH37293)*, pages 369–373, 2002.

[19] E Maler and D Reed. The Venn of identity. *IEEE Security and Privacy*, 6(2):16–23, 2008.

[20] Jaideep Srivastava, Robert Cooley, Mukund Deshpande, and P N Tan. Web Usage Mining: Discovery and Applications of Usage Patterns fromWeb Data. *SIGKDD Explorations*, 1(2):12–23, 2000.

[21] T A Parker. Single sign-on systems-the technologies and the products. *Security and Detection 1995 European Convention on*, pages 151–155, 1995.

[22] Cambridge University Press. Cambridge Dictionary Online. http://dictionary.cambridge.org/, 2012.

[23] The Eclipse Foundation. Higgins Project - Personal Data Service, 2012.

[24] Shibboleth. Shibboleth Project, 2012.

[25] John Franks, Phillip Hallam-Baker, Jeffrey Hostetler, Scott Lawrence, Paul Leach, Ari Luotonen, and Lawrence Stewart. *HTTP authentication: Basic and digest access authentication*, 1999.

[26] Oracle Group. Understanding Login Authentication, 2010.

[27] Tim Dierks. RFC 5246: The transport layer security (TLS) protocol version 1.2, 2008.

[28] Tomi Määttänen. Single Sign-On Systems. Technical report, Helsinki University of Technology, 2002.

[29] E Damiani, S D C di Vimercati, and P Samarati. New paradigms for access control in open environments, 2005.

[30] T Dierks and C Allen. TLS Protocol (Version 1.0) – RFC 2246, 1999.

[31] A Freier, P Karlton, and P Kocher. Secure Socket Layer (SSL) Protocol Version 3.0 – RFC 6101, 2011.

[32] N Ragouzis, Hal Lockhart, Brian Campbell, N Ragouzis, Hal Lockhart, Brian Campbell, N Ragouzis, Hal Lockhart, and Brian Campbell. Security Assertion Markup Language (SAML) V2.0 Technical Overview, 2008.

[33] OASIS Security Services (SAML) Technical Committee, 2012.

[34] T Bray, J Paoli, C M Sperberg-McQueen, E Maler, and F Yergeau. *Extensible Markup Language (XML) 1.0 (Fifth Edition).* W3C Recommendation, 2008.

[35] Scott Cantor, John Kemp, Rob Philpott, and Eve Maler. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0 (SAML Core), 2005.

[36] Scott Cantor, Frederick Hirsch, John Kemp, Rob Philpott, and Eve Maler. Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0. Technical report, 2005.

[37] Scott Cantor, Frederick Hirsch, John Kemp, Rob Philpott, Eve Maler, and John Hughes. Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. Technical report, OASIS Standard, 2005.

[38] S Cantor, Jahan Moreh, Rob Philpott, and Eve Maler. *Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0.* OASIS Standard, 2005.

[39] R Fielding. RFC 5849 - Hypertext Transfer Protocol – HTTP/1.1, 1999.

[40] A Barth. RFC 6265 - HTTP State Management Mechanism, 2011.

[41] Cisco. Cisco Networking Academy, 2012.

[42] Google Developers. SAML Single Sign-On (SSO) Service for Google Apps, 2012.

[43] OASIS WSS TC. OASIS Web Services Security (WSS) TC, 2006.

[44] K Lawrence, K Kaler, C Kaler, A Nadalin, K Kaler, C Kaler, and A Nadalin. Web Services Security: SOAP Message Security 1.1 (WS-Security 2004), 2006.

[45] M Gudgin, M Hadley, N Mendelsohn, J Moreau, H Frystyk, A Karmarkar, and Y Lafon. SOAP Version 1.2 Part 1: Messaging Framework (Second Edition), 2007.

[46] D Eastlake, J Reagle, D Solo, Frederick Hirsch, and Thomas Roessler. XML Signature Syntax and Processing (Second Edition), 2008.

[47] D Eastlake and J Reagle. XML Encryption Syntax and Processing, 2002.

[48] Anthony Nadalin. WS-Trust 1.4, 2009.

[49] K Lawrence and C Kaler. WS-SecureConversation 1.3, 2007.

[50] K Lawrence and C Kaler. WS-SecurityPolicy 1.2, 2007.

[51] K Lawrence and C Kaler. Web Services Security: SAML Token Profile 1.1, 2006.

[52] K Lawrence and C Kaler. Web Services Security: Kerberos Token Profile 1.1, 2006.

[53] K Lawrence and C Kaler. Web Services Security: X.509 Certificate Token Profile 1.1, 2006.

[54] H Lockhart, S Andersen, J Bohren, and Others. Web Services Federation Language (WSFederation), 2006.

[55] Java.net. GlassFish Metro Project, 2016.

[56] Apache Foundation. Apache Rampart, 2016.

[57] MSDN Microsoft. Windows Communication Foundation Security, 2017.

[58] Microsoft MSDN. Windows Identity Foundation, 2016.

[59] OpenID Authentication 2.0, 2007.

[60] T Berners-Lee, R Fielding, and L Masinter. RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax. Technical report, Network Working Group. Internet Engineering Task Force (IETF), 2005.

[61] Peter Davis, Nat Sakimura, Mike Lindelsee, and Gabe Wachob. *Extensible Resource Identifier (XRI) Syntax V2.0*. OASIS Committee Specification, 2005.

[62] Whitfield Diffie and Martin Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, 1976.

[63] D Hardt, J Bufu, and J Hoyt. OpenID Attribute Exchange 1.0, 2007.

[64] Google. Federated Login for Google Account Users, 2012.

[65] Yahoo. Yahoo! meets OpenID, 2012.

[66] MySpace Developer Team, 2012.

[67] X.commerce. Standard OpenID Integration for PayPal Access Getting Started Guide, 2012.

[68] Google Developers. Migrating from OpenID 2.0 to OpenID Connect. Technical report, 2016.

[69] Paypal. Integrate Log In with PayPal, 2016.

[70] Eran Hammer. Introducing OAuth 2.0. Technical report, may 2010.

[71] Mitchell Anicas. An Introduction to OAuth 2. Technical report, Digital Ocean, 2014.

[72] E Hammer-Lahav. *RFC 5849 - The OAuth 1.0 Protocol*. Internet Engineering Task Force (IETF), 2010.

[73] Dick Hardt. The OAuth 2.0 Authorization Framework [RFC 6749]. *Internet Engineering Task Force (IETF)*, pages 1–76, 2012.

[74] Twitter Developers. OAuth: Send secure authorized requests to the Twitter API, 2017.

[75] Facebook Developers. Facebook login: manually build a login flow. Technical report, 2017.

[76] Mark Atwood. OAuth Core 1.0 Revision A, 2009.

[77] M Jones and D Hardt. RFC 6750 - The OAuth 2.0 Authorization Protocol: Bearer Tokens Usage. Technical report, Internet Engineering Task Force (IETF), 2012.

[78] Google Developers. Using OAuth 2.0 to Access Google APIs. Technical report, Google, 2016.

[79] Microsoft MSDN. OAuth 2.0, 2012.

[80] Nat Sakimura, John Bradley, Mithcael Jones, Breno de Medeiros, and Chuck Mortimore. Openid connect core 1.0. *The OpenID {{}{...}{}}*, 2014.

[81] L Richardson and S Ruby. *RESTful web services: Web services for the real world*. O'Reilly Media, may 2007.

[82] N Sakimura, J Bradley, M Jones, B de Medeiros, C Mortimore, and E Jay. OpenID Connect Messages 1.0 - draft 20. 2013.

[83] Tim Moses. *eXtensible Access Control Markup Language (XACML) Version 2.0*. OASIS Standard, 2005.

[84] Enol Fernández-del Castillo, Diego Scardaci, and Álvaro López García. The EGI Federated Cloud e-Infrastructure. *Procedia Computer Science*, 68:196–205, 2015.

[85] Ahmed Shiraz Memon, Jens Jensen, Aleš Černivec, Krzysztof Benedyczak, and Morris Riedel. Federated authentication and credential translation in the EUDAT collaborative data infrastructure. In *Proceedings - 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing, UCC 2014*, pages 726–731. Institute of Electrical and Electronics Engineers Inc., 2014.

[86] UNITY IdM Project, 2016.

[87] Mikael Linden and Michal Prochazka. ELIXIR authentication and authorization infraestructure AAI, 2016.

[88] umbrela Project webpage, 2016.

[89] A A I Workshop Brussels. Introduction to STORK2.0 project. Technical Report April, 2014.

[90] eduGAIN Portal, 2016.

[91] Alejandro Perez-Mendez, Fernando Pereniguez-Garcia, Rafael Marin-Lopez, Gabriel Lopez-Millan, Josh Howlett, Fernando Pereñígez-García, Rafael Marín-López, Gabriel López-Millán, and Josh Howlett. Identity Federations Beyond the Web : A survey. *IEEE Communications Surveys Tutorials*, 16(4):2125–2141, 2014.

[92] J (Jisc) Howlett, S (Painless Security) Hartman, H (ARM Ltd.) Tschofenig, and J (August Cellars) Schaad. RFC 7831: Application Bridging for Federated Access Beyond Web (ABFAB) Architecture. Technical report, Internet Engineering Task Force (IETF), may 2016.

[93] B Aboba, L Blunk, J Vollbrecht, J Carlson, and H Levkowetz (Ed.). RFC 3748: Extensible Authentication Protocol (EAP). Technical report, IETF Network Working Group, 2004.

[94] A. DeKok and A. Lior. RFC 6929: Remote Authentication Dial-In User Service (RADIUS) Protocol Extensions. 2013.

[95] eduroam Portal, 2017.

[96] Rhys Smith. The Architecture and Protocol Flows of Moonshot. Technical report, Moonshot, 2014.

[97] SAML2int. SAML2int - The Interoperable SAML 2.0 Profile, 2015.

[98] eduGAIN WIKI. eduGAIN WIKI Therminology, 2014.

[99] Damien Lecarpentier, Peter Wittenburg, Willem Elbers, Alberto Michelini, Riam Kanso, Peter Coveney, and Rob Baxter. EUDAT: A New Cross-Disciplinary Data Infrastructure for Science. *International Journal of Digital Curation*, 8(1):279–287, 2013.

[100] D Lecarpentier, A Michelini, and P Wittenburg. The building of the EUDAT Cross-Disciplinary Data Infrastructure. In *EGU General Assembly Conference Abstracts*, volume 15 of *EGU General Assembly Conference Abstracts*, pages EGU2013—-7202, 2013.

[101] Unity. UNITY - Identity relationship managment., 2017.

[102] Unity Team. Unity Manual Version 1.9.4. Technical report, 2016.

[103] EUDAT. EUDAT Partners, 2017.

[104] EUDAT Project. EUDAT Use Cases, 2016.

[105] STORK-eID Consortium. *STORK 2.0*. PhD thesis, 2016.

[106] STORK-eID Consortium. D5.8.3d Security Principles and Best Practices Deliverable. Technical report, STORK-eID Consortium, 2011.

[107] STORK-eID Consortium. D4.4 First version of Technical Specifications for the cross border Interface. 2011.

[108] European Commission. Trust Services and eID (eIDAS regulation), 2015.

[109] eIDAS Technical Subgroup. eIDAS - Interoperability Architecture. Technical report, 2015.

[110] eIDAS Technical Subgroup. eIDAS Technical Specifications v1.0. Technical report, 2015.

[111] Tom Scavo and Scott Cantor. Shibboleth architecture: Technical overview. Technical report, Internet2, 2005.

[112] Scott Cantor, Steven Carmody, Marlena Erdos, Keith Hazelton, Walter Hoehn, RL "Bob" Morgan, Tom Scavo, and David Wasley. Shibboleth Architecture: Protocols and Profiles. Technical report, Internet2, 2005.

[113] Arun Nanda. A Technical Reference for the Information Card Profile V1.0. Technical report, Microsoft Corporation, 2006.

[114] A. Shelat. Project Higgins: User-centric identity meta system and privacy. Technical report, IBM Research, 2006.

[115] Harald Zwingelberg, Katalin Storf, Dieter Sommer, and Sabrina De Capitani di Vimercati. PrimeLife dissemination report V2. Technical report, 2010.

[116] J Camenisch and E Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 21–30. ACM, 2002.

[117] SWIFT Project.

[118] Ronald Marx. SWIFT Deliverable D302. 2009.

[119] Gabriel López, Oscar Cánovas, Antonio F. Gómez Skarmeta, and Joao Girao. A SWIFT Take on Identity Management. *IEEE Computer*, 42:58–65, may 2009.

[120] Mario Lischka, Yukiko Endo, Elena Torroglosa, Alejandro Pérez, and Antonio G Skarmeta. Towards Standardization of Distributed Access Control. *Computers {&} Education*, 51(1):0–4, 2009.

[121] M Sánchez, G López, O Cánovas, and A F Gómez-Skarmeta. Bootstrapping a global SSO from network access control mechanisms. In *4th European PKI Workshop (EuroPKI'07)*. Computer Science (LNCS) Volume 4582/2007, jun 2007.

[122] Marc Barisch. D303. Specification of Identity-centric Security Modules and Cross-layer Interfaces. Technical report, SWIFT, 2009.

[123] Acclinks Communications. Subscriber Identity Module (SIM) Card, 2007.

[124] SWIFT. D402: SWIFT Mobility Architecture. Technical report, SWIFT, 2009.

[125] Mario Lischka, Yukiko Endo, Manuel Sánchez Cuenca, and Manuel Sánchez Cuenca. Deductive Policies with XACML. In *ACM Workshop on Secure Web Services*, pages 37–44, 2009.

[126] Hariharan Rajasekaran, Joao Girao, Hugo Santos, Mario Lischka, Nils Gruschka, Alejandro Pérez, Elena Torroglosa, Gabriel López, Antonio F. Gómez-Skarmeta, Ricardo Azevedo, Hervais C. Simo Fhom, Dirk Scheuermann, Ronald Marx, Marc Barisch, Alfredo Matos, Rodolphe Marques, and Rui Ferreira. D207: Final SWIFT architecture. Technical report, SWIFT consortium, 2010.

[127] Alejandro Pérez Méndez, Elena M. Torroglosa García, Gabriel López Millán, Antonio F. Gómez-Skarmeta, Marc Barisch, Dirk Scheuermann, Hervais Simo Fhom, Alfredo Matos, Rodolphe Marques, André Zúquete, and Mario Lischka. D207a: Final SWIFT architecture. pages 1–55, 2010.

[128] Ricardo Azevedo, Pedro Santos, Alfredo Matos, Rodolphe Marques, Rui Ferreira, Pedro Gonçalves, Rui Ribeiro, Ronald Marx, David Lutz, Marc Barisch, Alejandro Pérez Méndez, Elena Torroglosa, Gabriel López, Antonio F. Gomez-Skarmeta, Mario Lischka, Wolfgang Steigerwald, and Alexandra Mikityuk. D207b: Final SWIFT architecture. pages 1–63, 2010.

[129] AVISPA Project. Automated Validation of Internet Security Protocols and Applications (AVISPA), 2003.

[130] Alejandro Pérez, Gabriel López, Óscar Cánovas, and Antonio F. Gómez-Skarmeta. Formal Description of the SWIFT Identity Management Framework. *Future Gener. Comput. Syst.*, 27(8):1113—-1123, 2011.

[131] Ehab Al-Shaer, Albert Greenberg, Charles Kalmanek, David A Maltz, T S Eugene Ng, and Geoffrey G Xie. New Frontiers in Internet Network Management. *ACM SIGCOMM Computer Communication Review*, 39(5):37–39, 2009.

[132] J Schonwalder, M Fouquet, G Rodosek, and I Hochstatter. Future Internet = Content + Services + Management. *IEEE Communications Magazine*, 47(7):27–33, 2009.

[133] A Pras, J Schonwalder, M Burgess, O Festor, G M Perez, R Stadler, and B Stiller. Key Research Challenges in Network Management. *IEEE Communications Magazine*, 45(10):104–110, 2007.

[134] European Future Internet Portal. Future Internet Assembly, 2010.

[135] Alex Galis and Others. Position Paper on Management and Service-aware Networking Architectures (MANA) for Future Internet, 2010.

[136] DANTE Ltd. and Others. GÉANT Project, 2010.

[137] Pedro Martinez-Julia, Diego R Lopez, and Antonio F Gomez-Skarmeta. The GEMBus Framework and its Autonomic Computing Services. In *Proceedings of the International Symposium on Applications and the Internet Workshops*, pages 285–288, Washington, DC, USA, 2010. IEEE Computer Society.

[138] Pedro Martinez-Julia, Antonio Marin Cerezuela, and Antonio F Gomez-Skarmeta. A Service Oriented Architecture for Basic Autonomic Network Management. In *Proceedings of the IEEE Symposium on Computers and Communications*, pages 805–807, Washington, DC, USA, 2010. IEEE Computer Society.

[139] David Chappell. *Enterprise Service Bus*. O'Reilly Media, Inc., 2004.

[140] George Lawton. Developing Software Online With Platform-as-a-Service Technology. *Computer*, 41(6):13–15, 2008.

[141] Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. Web Services Description Language (WSDL) 1.1, 2001.

[142] C Mortimore, M Jones, B Campbell, and Y Goland. OAuth 2.0 Assertion Profile (draft-ietf-oauth-assertions-01). 2011.

[143] DANTE Ltd. and Others. GÉANT 2 Project, 2004.

[144] M Jones, D Balfanz, J Bradley, Y Goland, J Panzer, N Sakimura, and P Tarjan. JSON Web Token (JWT). Technical report, Network Working Group, Internet Engineering Task Force (IETF), 2011.

[145] D Crockford. The application/json Media Type for JavaScript Object Notation (JSON) - RFC4627, 2006.

[146] David Recordon and Drummond Reed. OpenID 2.0. In *Proceedings of the second ACM workshop on Digital identity management - DIM '06*, page 11. ACM Press, 2006.

[147] Dick Hardt. The OAuth 2.0 Authorization Framework. 2012.

[148] EUDAT Project. B2ACCESS, 2016.

[149] Thomas J Smedinghoff. Solving the legal challenges of trustworthy online identity. *Computer Law {&} Security Review*, 28(5):532–541, 2012.

[150] Jim Basney and Scott Koranda. A Study of Three Approaches to International Identity Federation for the LIGO Project. Technical report, 2013.

[151] Daniela Pöhn, Stefan Metzger, and Wolfgang Hommel. Géant-TrustBroker: Dynamic, Scalable Management of SAML-Based Inter-federation Authentication and Authorization Infrastructures. In Nora Cuppens-Boulahia, Frédéric Cuppens, Sushil Jajodia, Anas Abou El Kalam, and Thierry Sans, editors, *ICT Systems Security and Privacy Protection: 29th IFIP TC 11 International Conference, SEC 2014, Marrakech, Morocco, June 2-4, 2014. Proceedings*, pages 307–320. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[152] Kantara. Kantara Project, 2017.

[153] Liberty Alliance Project. Liberty Identity Assurance Framework. 2008.

[154] InCommons webpage, 2016.

[155] AARC. AARC Project, 2017.

[156] STORK-eID Consortium. D4.9 Final version of Functional Design. pages 1–229, 2011.

[157] Scott Cantor and Keith Hazelton. MACE-Dir SAML Attribute Profiles, 2008.

[158] Shibboleth WIKI. Metadata For IdP, 2016.

[159] GÉANT. eduGAIN technical website, 2016.

[160] Jordi Ortiz, Pedro Martinez-julia, Christos Kanellopoulos, and Antonio F Skarmeta. Scholar European Electronic Identity Federation. *TNC 2015*, pages 1–3, 2015.

Appendix A

SWIFT demo example: initial access to a music store

This appendix shows an example with various screenshots that illustrate a scenario in which a user named John Swift uses his virtual identity *john.swift@swift-idagg.no-ip.org* to access a online music store. It will be assumed that the user *John* has previously created the mentioned virtual identity in the Identity Aggregator (*swift-idagg.no-ip.org*) and he is already registered within the SWIFT client applet.

Figure A.1 shows the web access page to the music store that offers a dual authentication system. On the one hand, the user can choose the traditional login system based on username and password, and on the other hand, there is the access based on the federated system proposed by SWIFT architecture. To use the virtual identity with the service, the user must select in the applet what VID wants to use and then click on the button "Use It". Because the evidence is expired (as seen in the capture), the applet generates a new `Initiation Statement` that is sent to the Music Service Provider.

When the Service Provider receives the access request with the `Initiation Statement`, it redirects the user to the corresponding Identity Aggregator to authenticate the user. Figure A.2 shows a debug screen that would not be observed in a production system. The figure corresponds to a web page generated by the Identity Aggregator when it receives the Service Provider's request which contains information regarding the virtual identity such as the VID, the SP identifier, the associated Authentication Provider and the pseudonym used between them.

Because the received statement is of type Initiation Statement, it is understood that the user has not recently performed a valid authentication with the IdA, so it will be redirected to AuthnP.

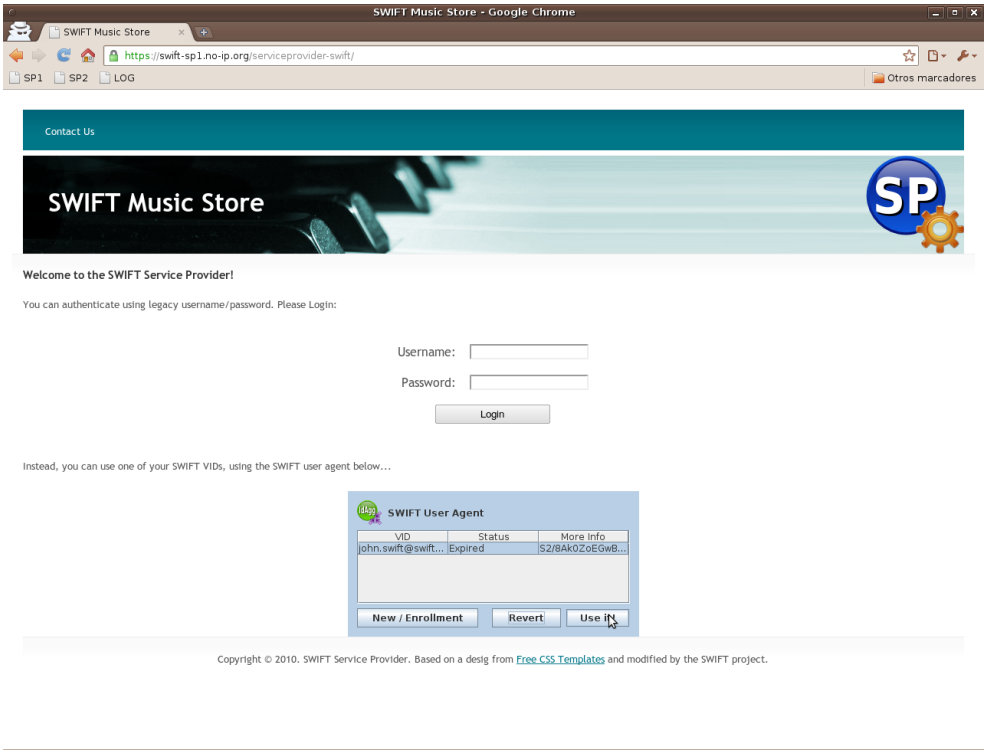


Figure A.1: Step 1: Applet display within the SP web page

To continue with the normal flow of execution, in the debug page the user must click on the hyperlink "Link" to be redirected to the AuthnP. Once in AuthnP (Figure A.3), the user must enter the credentials of the account associated in the enrollment process to authenticate with it. If the credentials are correct and are associated with the pseudonym that was established for the user between the IdA and AuthnP, the latter will generate an `Authentication Statement` confirming the authenticity of the user and with it, will redirect the user to the IdA.

Figure A.4 shows debug information generated by the IdA that would not be displayed in a production system. In the image you can see the information regarding the successful authentication made by AuthnP. At the bottom of the screen, the applet shows the results of the request made by the IdA to update the information related to the VID with the new evidence generated in the process. When the user clicks on "Click here to continue", the IdA redirects the user to the Service Provider with a successful authentication response.

Figure A.5 shows the screen snapshot in which the user has finally gained access to the protected area of the music service. At this point, the Service Provider knows that the user is an identified user in the federation, although the SP does not know what his real identity is.

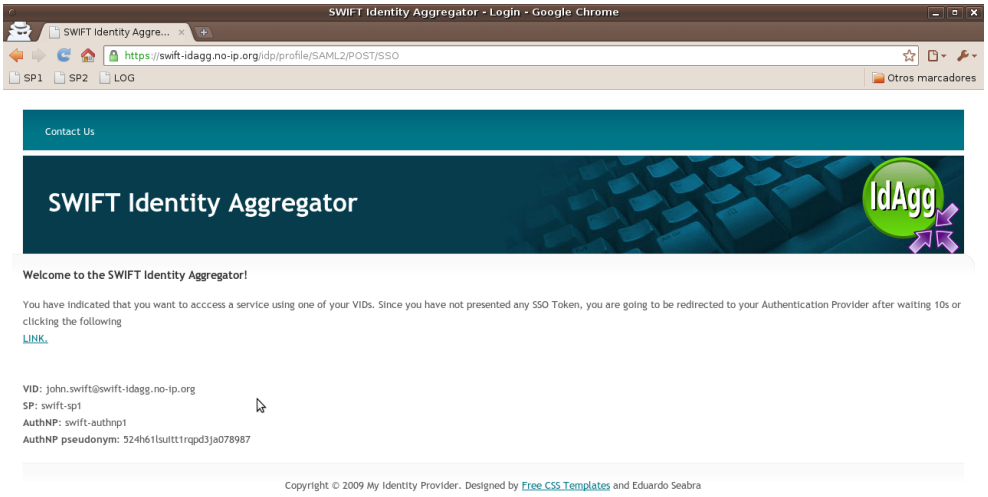


Figure A.2: Step 2: the *IdA* receives an authentication request

This scenario includes an example of the use of the user client developed in this work. It can be verified that in spite of the internal complexities of the architecture to protect the privacy and the anonymity, the use by the user is very simple. In a production system (without debugging pauses in IdA) the user would go from selecting the virtual identity to be used in the applet to authenticating to the AuthnP, and from there, directly to the protected resource.

In spite of the internal complexities of the architecture to protect the privacy and the anonymity of the user, the scenario clearly shows how the use and management of the virtual identities by the user is very simple. In a production system (without debugging pauses in the IdA), the user would only have to select the virtual identity in the applet that he wants to use, enter the corresponding user name and password in the AuthnP, and from there directly access to the protected resource. In the SSO access use case, the process would be even more immediate, since the user only has to select the virtual identity in the applet to then directly access the service.

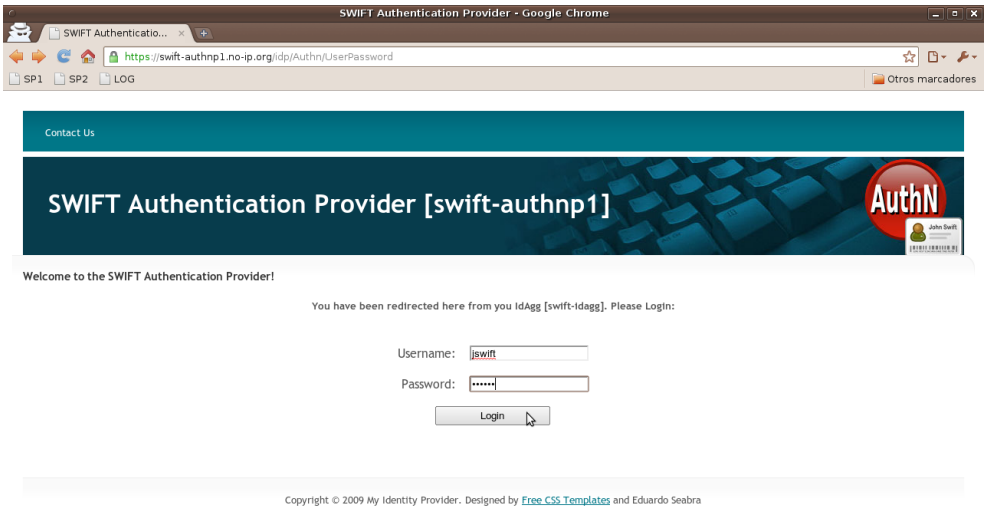


Figure A.3: Paso 3: autenticación del usuario frente al Proveedor de Autenticación

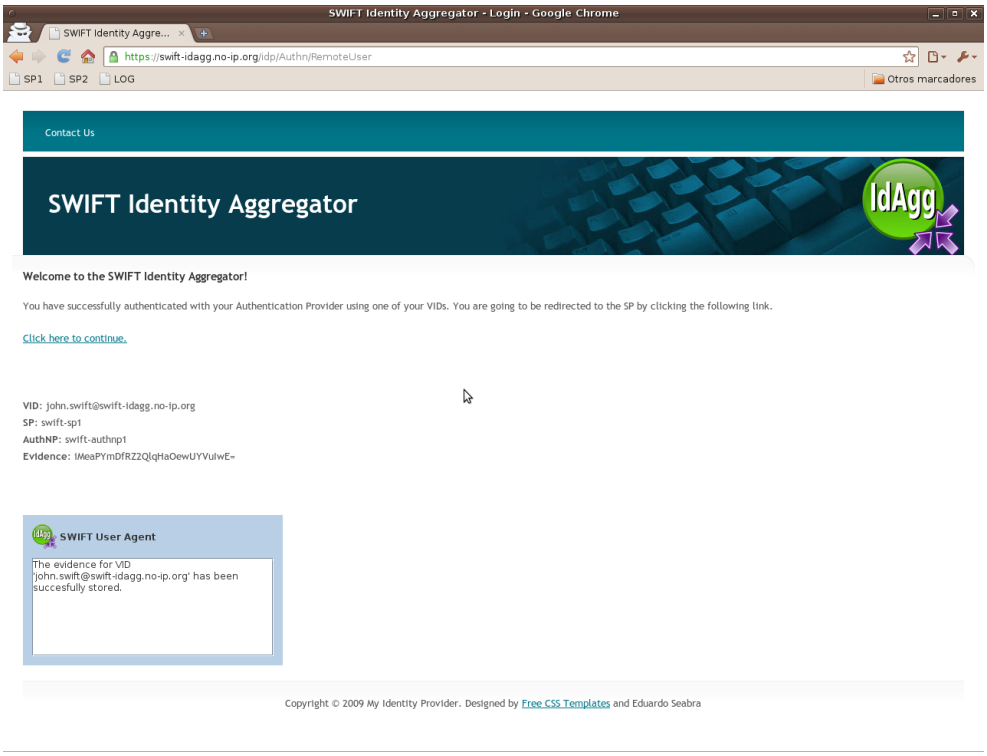


Figure A.4: Paso 4: envío de la sentencia SSO Statement desde el IdA al applet

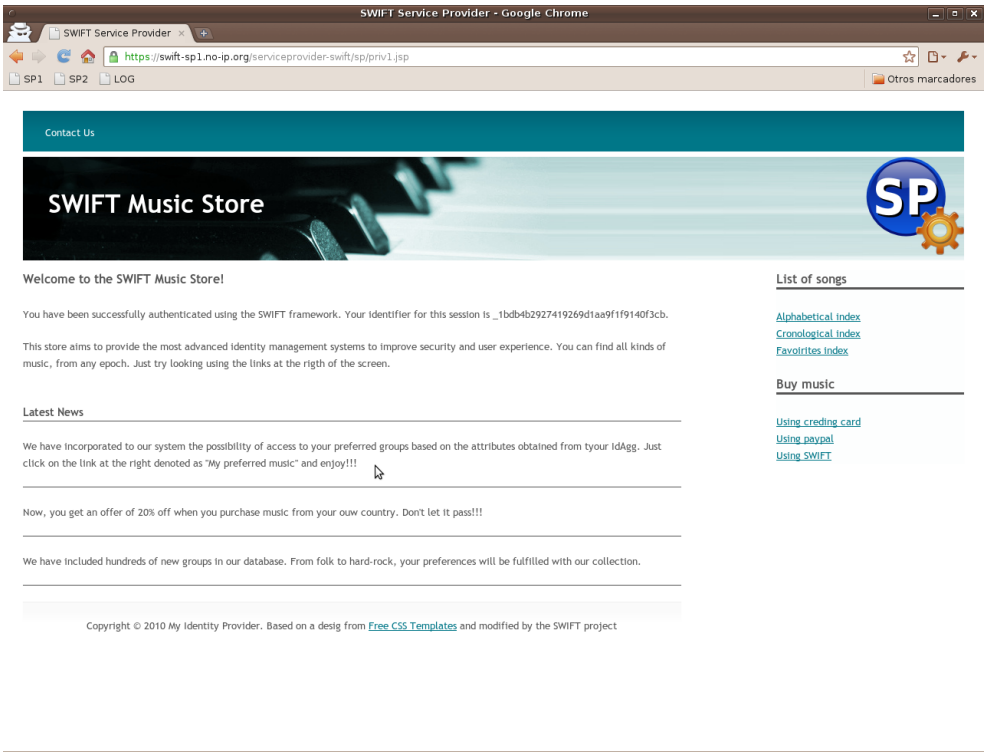


Figure A.5: Paso 5: el usuario consigue acceso al servicio gracias a su identidad virtual

Appendix B

eduGAIN SAML Request and Response Examples

Below an Authentication Request and Authentication Response example is detailed. During eduGAIN's successful authentication process, the Identity Provider (IdP) supplies the Service Provider (SP) with a set of default attributes related to the authenticated user (even if no attribute request has been performed). These attributes define the authenticated user identity and should be used during the subsequent attribute retrieval process. In particular, a transient session identifier is provided in the authentication response and will be used as identifier in the attribute retrieval process, thus a certain degree of anonymity level is provided. The user related information is included in the response attributes, such as `givenName`, `email` or `eduPersonPrincipalName`.

Listing B.1: eduGAIN SAML 2.0 AuthNRequest

```
<AuthnRequest
  Destination="http://idp.ssocircle.com:80/sso/SSORedirect/metaAlias/ssocircle"
  IssueInstant="2008-05-28T20:12:35.971Z"
  ID="xc6YVsq0cxOWkwdDqQwqCIMmHSu" Version="2.0"
  xmlns="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
    <saml:Issuer>edugain.showcase.surfnet.nl</saml:Issuer>
    <NameIDPolicy AllowCreate="true"/>
</AuthnRequest>
```

Listing B.2: eduGAIN SAML 2.0 Response

```
<Response
  InResponseTo="_78b579d14708bf36642f1375f6ecc9642199b6bfdc
  IssueInstant="2008-06-03T09:46:55.934Z" ID="Vzf4L8F_PaZ-JOy4cpgijVTu0I_"
  Version="2.0"xmlns="urn:oasis:names:tc:SAML:2.0:protocol"
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <saml:Issuer>edugain.showcase.surfnet.nl</saml:Issuer>
  <Status>
    <StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
  </Status>
  <saml:Assertion Version="2.0" IssueInstant="2008-06-03T09:46:55.947Z"
    ID="G4t1o38BoKjogdlV9AYnC7wDcOL">
    <saml:Issuer>edugain.showcase.surfnet.nl</saml:Issuer>
    <ds:Signature>
      <ds:SignedInfo>
        <ds:CanonicalizationMethod
        Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        <ds:SignatureMethod
        Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
        <ds:Reference URI="#G4t1o38BoKjogdlV9AYnC7wDcOL">
          <ds:Transforms>
            <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
          <ds:DigestValue>drF4K3fUbvfLYdKSk/Axynxu/8Q=</ds:DigestValue>
        </ds:Reference>
      </ds:SignedInfo>
      <ds:SignatureValue>
        SOa3qqE8fvojAcH0pD3KEDkqkgpdAELRrLYiCw8K0xn2i+Q2o/n2bZXLY/s1tZqhEoykE6A37Fd1
        O10z75MTbW0DX3oFCEwBT0l7bFV3zoqmbuwZ8+AWm3/86u4kSGuY1HJAtV7wp610j0X5I3rAejsc
        UyLO8gjre6W7dFkWqC4=
      </ds:SignatureValue>
      <ds:KeyInfo>
        <ds:X509Data>
          <ds:X509Certificate>
          MIICUzCCAbygAwIBAgIGARQGprr2MA0GCSqGSIb3DQEBBQUAMG0xCzAJBgNVBAYTAk5MMRAwDgYD
          VQQHEwdVdHJlY2h0MRMwEQYDVQQKEwpTVVJGbmV0IEJWMRwwGgYDVQQLExNNaWRkbGV3YXJlIFNl
          cnZpY2VzMRkwFwYDVQQDExBGZWRlcmF0aWUgQmVoZWVyMB4XDTA3MDcyNzA3NDkyM1oXDTEyMDcy
          NTA3NDkyM1owbTELMAkGA1UEBhMCTkwxEDAOBgNVBAcTB1V0cmVjaHQxEzARBgNVBAoTClNVUkZu
          ZXQgQlYxHDAaBgNVBAsTE01pZGRsZXdhcmUgU2VydmljZXMxGTAXBgNVBAMTEEZlZGVyYXRpZSBC
```

```
        ZWhlZXIwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAJV0dbL8fZe9quABgF8rnJPwjIp+FLIa
        DFypyXUu6ob7tQJtqmD8W7LawxjlGTnswoqOolMVEyQLbw+RLDwtfJ/45/65aA4J+/I8vD68H4go
        ice63Q81SIJLIPBmksdSICokDnv/FTyPuZqlPOzuapnaYs9UO9slpmgkNtCiiLh7AgMBAAEwDQYJ
        KoZIhvcNAQEFBQADgYEANVzWNWLxCIx3sWz6+BWUZzAMlKJovfmNK5kCROx+Fc7r/Tns4P0Xlwp2
        mLtgOqhhI4L3gpGFrUfN/tUfn0rjdjlDAxB206fDUslNOIrhyPjqaDgKbCxgrB7c9ybuxAq+5m97
        NQBAUrKDuTTbN7llgCpw+Ui8FusIorNTlGVcpC0=</ds:X509Certificate>
      </ds:X509Data>
      <ds:KeyValue>
        <ds:RSAKeyValue>
          <ds:Modulus>
            lXR1svx9l72q4AGAXyuck/CMin4UshoMXKnJdS7qhvu1Am2qYPxbstrDGOUZOezCio6iUxUTJAtv
            D5EsPC18n/jn/rloDgn78jy8PrwfiCiJx7rdDzVIgksg8GaSx1IgKiQOe/8VPI+5mqU87O5qmdpi
            z1Q72yWmaCQ20KKIuHs=
          </ds:Modulus>
          <ds:Exponent>AQAB</ds:Exponent>
        </ds:RSAKeyValue>
      </ds:KeyValue>
    </ds:KeyInfo>
  </ds:Signature>
  <saml:Subject>
    <saml:NameID SPNameQualifier="urn:mace:feide.no:services:no.feide.foodle"
      NameQualifier="edugain.showcase.surfnet.nl"
      Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient">
      SvmQaABKKVdE71wxeuuwXQacDmj
    </saml:NameID>
    <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <saml:SubjectConfirmationData
      InResponseTo="_78b579d14708bf36642f1375f6ecc9642199b6bfdc"
      NotOnOrAfter="2008-06-03T09:51:55.948Z"
      Recipient="https://foodle.feide.no/simplesaml/saml2/sp/AssertionConsumerService.php"/>
    </saml:SubjectConfirmation>
  </saml:Subject>
  <saml:Conditions NotOnOrAfter="2008-06-03T09:51:55.947Z"
    NotBefore="2008-06-03T09:41:55.947Z">
    <saml:AudienceRestriction>
      <saml:Audience>urn:mace:feide.no:services:no.feide.foodle</saml:Audience>
    </saml:AudienceRestriction>
  </saml:Conditions>
  <saml:AuthnStatement AuthnInstant="2008-06-03T09:46:55.946Z"
    SessionIndex="G4t1o38BoKjogdlV9AYnC7wDcOL">
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified</saml:AuthnContextClassRef>
    </saml:AuthnContext>
```

```
86      </saml:AuthnStatement>
87      <saml:AttributeStatement xmlns:xs="http://www.w3.org/2001/XMLSchema">
88        <saml:Attribute
89          NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
90          Name="urn:mace:dir:attribute-def:eduPersonAffiliation">
91          <saml:AttributeValue xsi:type="xs:string">
92          Medewerker
93          </saml:AttributeValue>
94        </saml:Attribute>
95        <saml:Attribute
96          NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
97          Name="urn:mace:dir:attribute-def:displayName">
98          <saml:AttributeValue xsi:type="xs:string">
99          Hans Zandbelt
100         </saml:AttributeValue>
101       </saml:Attribute>
102       <saml:Attribute
103         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
104         Name="urn:mace:dir:attribute-def:givenName">
105         <saml:AttributeValue
106         xsi:type="xs:string">Hans</saml:AttributeValue>
107       </saml:Attribute>
108       <saml:Attribute
109         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
110         Name="urn:mace:dir:attribute-def:mail">
111         <saml:AttributeValue xsi:type="xs:string"
112         >Hans.Zandbelt@surfnet.nl</saml:AttributeValue>
113       </saml:Attribute>
114       <saml:Attribute
115         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
116         Name="urn:mace:dir:attribute-def:uid">
117         <saml:AttributeValue
118         xsi:type="xs:string">hansz</saml:AttributeValue>
119       </saml:Attribute>
120       <saml:Attribute
121         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
122         Name="urn:mace:dir:attribute-def:cn">
123         <saml:AttributeValue xsi:type="xs:string">
124         Hans Zandbelt</saml:AttributeValue>
125       </saml:Attribute>
126       <saml:Attribute
127         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
128         Name="urn:mace:dir:attribute-def:sn">
```

```
129         <saml:AttributeValue
130         xsi:type="xs:string">Zandbelt</saml:AttributeValue>
131       </saml:Attribute>
132       <saml:Attribute
133         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic"
134         Name="urn:mace:dir:attribute-def:eduPersonPrincipalName">
135         <saml:AttributeValue xsi:type="xs:string">
136         hansz@surfnet.nl
137         </saml:AttributeValue>
138       </saml:Attribute>
139     </saml:AttributeStatement>
140   </saml:Assertion>
141 </Response>
```

Appendix C

Stork SAML Request and Response examples

Below an Authentication Request and Authentication Response example is detailed. As stated above, during the STORK's authentication, attributes can be retrieved at once. The service provider (SP) asks the Service Provider Policy Enforcement Point (S-PEPs) for the `eIdentifier` and whether user's age is above 16 or 18 years old as well as the birth date.

In response the Identity Provider will return through the Citizen Policy Enforcement Point (C-PEPs) the availability of each attribute with the corresponding value if available.

Listing C.1: STORK Authentication Request Example

```
<saml2p:AuthnRequest xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
xmlns:stork="urn:eu:stork:names:tc:STORK:1.0:assertion"
xmlns:storkp="urn:eu:stork:names:tc:STORK:1.0:protocol"
AssertionConsumerServiceURL="https://despanishpeps.redsara.es/SP/ReturnPage"
Consent="urn:oasis:names:tc:SAML:2.0:consent:unspecified"
Destination="https://despanishpeps.redsara.es/PEPS/ServiceProvider" ForceAuthn="true"
ID="_9d1e2953ffed26c5a9b77f0dfce34ab4" IsPassive="false"
IssueInstant="2016-07-23T10:25:34.522Z"
ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
ProviderName="DEMO-SP"
Version="2.0">
<saml2:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
  http://S-PEPS.gov.xx
```

```
16    </saml2:Issuer>
17    <ds:Signature>
18      <ds:SignedInfo>
19        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
20        <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" />
21        <ds:Reference URI="#_9d1e2953ffed26c5a9b77f0dfce34ab4">
22          <ds:Transforms>
23            <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
24            <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
25          </ds:Transforms>
26          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
27          <ds:DigestValue>D4iI81UNzHEkVclmuyk9SvEINZE=</ds:DigestValue>
28        </ds:Reference>
29      </ds:SignedInfo>
30      <ds:SignatureValue>
31        Q1zk2VLT58YEAB0PzpJpATcdgZQ3bqqHRrw9mBTm/e4LGdBpGTQ2TTVWltJ8/9TKBoXqoI8LbQHc
32        IdnKtzLPRbcMtwcyImYt0mB/f+vZF+1VSSqyzVR4XpOCGsLmXQE8/7AkW/zG2EaAzhE97toTXony
33        1nG9kFMbvwSDX+it4DrdrE5dxNxEZ/2axKJGiwAVriW5W8qg8KqSDGJFZmdNj4vGYXx+DAMVRahU
34        Vy3RAWFgDrBRuScMOeEPRSzOjjnJIXczPD6UGkSIg3lsxH5ASgWZ2REMl04fnx97f/OoeaBppk+o
35        e2gGS8akKLdBq4scpAIJu/2lB4EMLacdlg78nw==
36      </ds:SignatureValue>
37      <ds:KeyInfo>
38        <ds:X509Data>
39          <ds:X509Certificate>
40            MIIDYzCCAksCBFMwAw4wDQYJKoZIhvcNAQEFBQAwdjEdMBsGCSqGSIb3DQEJARYOZW1haWxAaW5k
41            cmEuZXMxCzAJBgNVBAYTAkVTMRAwDgYDVQQIDAdFc3Bhw7FhMQ8wDQYDVQQKDAZNSU5BSFAxEzAR
42            BgNVBAsMCk1pbmlzdGVyaW8xEDAOBgNVBAMMB0RlbW8tU1AwHhcNMTQwMzI0MTAwMzU4WhcNMjQw
43            MzIxMTAwMzU4WjB2MR0wGwYJKoZIhvcNAQkBFg5lbWFpbEBpbmRyYS5lczELMAkGA1UEBhMCRVMx
44            EDAOBgNVBAgMB0VzcGHDsWExDzANBgNVBAoMBk1JTkFIUDETMBEGA1UECwwKTWluaXN0ZXJpbzEQ
45            MA4GA1UEAwwHRGVtby1TUDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALStvQlp3FEk
46            1NMhl+xEyrH2TivaF2QJGpiZSgscnLaMS5K7hcSmmfDPPLa46EPIxa8ob7lO5FlgRxS08xcXjDMk
47            AqyMToLbHIblOWvm6UXG5ESzV3yn+OWale8kppMlfa5IUcn1m9brxqrtt97MbOnAvxGRdMZy/hBv
48            p0eh7RostSooBRuw8EsH0kn6keQnmw2ooClkNXZYFtyzRfj8bDQ11INcKFTSIULFuR9Fbec1UH4N
49            P4xmqWcOPEpk/yKpQBuTnfYsr1nNwGwW5ql9Xj7xJqJuyOb6flsYZnNFkbOA4LlioEFanRxa+CIC
50            rlUaZjD2/j+RlnIopFfq2e3DaisCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEAkKYcVU7yKPsdT+Fi
51            GWuhNFdWgYnE1qqfdBkYxX5Gs6qxwujYwfiIe4FDEc9zzIsw1l42ZwcjxVXEIIM9Ym+FC6lwYt2G
52            Cei+SYorZ2C3MbK3Mv+kiMb3IJf6konQqHO/mBBXEY9KzJYwvwOhsPJGjW10AMLAoF5niUH2k00u
53            hwl+P7JvzV9SpMWv0abJvkjjwV82r9i4bMpgj2GK7Sb7ihY78nJT004OmeV8n5TXF4GNCSBwADbO
54            t5EZPEEOh0KPAXAZQXrygx8V+uOzzjPw6xk85hf/11/xTGtfj6jES74ySvQc57XpnZZCaJNGMNwd
55            0iqmK9qS0pbeUYgYuZmCAQ==
56          </ds:X509Certificate>
57        </ds:X509Data>
58      </ds:KeyInfo>
```

```
59    </ds:Signature>
60    <saml2p:Extensions>
61      <stork:QualityAuthenticationAssuranceLevel>3</stork:QualityAuthenticationAssuranceLevel>
62      <stork:spSector>DEMO-SP</stork:spSector>
63      <stork:spInstitution>DEMO-SP</stork:spInstitution>
64      <stork:spApplication>DEMO-SP</stork:spApplication>
65      <stork:spCountry>ES</stork:spCountry>
66      <storkp:eIDSectorShare>true</storkp:eIDSectorShare>
67      <storkp:eIDCrossSectorShare>true</storkp:eIDCrossSectorShare>
68      <storkp:eIDCrossBorderShare>true</storkp:eIDCrossBorderShare>
69      <storkp:RequestedAttributes>
70      <stork:RequestedAttribute
71        Name="http://www.stork.gov.eu/1.0/eIdentifier"
72        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri" isRequired="true" />
73      <stork:RequestedAttribute
74        Name="http://www.stork.gov.eu/1.0/isTeacherOf"
75        NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri" isRequired="true" />
76      </storkp:RequestedAttributes>
77      <storkp:AuthenticationAttributes>
78        <storkp:VIDPAuthenticationAttributes>
79          <storkp:CitizenCountryCode>ES</storkp:CitizenCountryCode>
80          <storkp:SPInformation>
81            <storkp:SPID>DEMO-SP</storkp:SPID>
82          </storkp:SPInformation>
83        </storkp:VIDPAuthenticationAttributes>
84      </storkp:AuthenticationAttributes>
85    </saml2p:Extensions>
86  </saml2p:AuthnRequest>
```

Listing C.2: STORK Response Example

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <saml2p:Response xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
3     xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
4     xmlns:stork="urn:eu:stork:names:tc:STORK:1.0:assertion"
5     xmlns:storkp="urn:eu:stork:names:tc:STORK:1.0:protocol"
6     xmlns:xs="http://www.w3.org/2001/XMLSchema"
7     Consent="urn:oasis:names:tc:SAML:2.0:consent:obtained"
8     Destination="https://despanishpeps.redsara.es/SP/ReturnPage"
9     ID="_c4b10b1f4f982b86a3871fa8438d22d4" InResponseTo="_9d1e2953ffed26c5a9b77f0dfce34ab4"
10    IssueInstant="2016-07-23T10:28:04.241Z" Version="2.0">
11    <saml2:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
12      http://C-PEPS.gov.xx
```

```
13    </saml2:Issuer>
14    <ds:Signature>
15      <ds:SignedInfo>
16        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
17        <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" />
18        <ds:Reference URI="#_c4b10b1f4f982b86a3871fa8438d22d4">
19          <ds:Transforms>
20            <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
21            <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
22              <ec:InclusiveNamespaces mlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#" PrefixList="xs" />
23            </ds:Transform>
24          </ds:Transforms>
25          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
26          <ds:DigestValue>nj7GzrbH7SXcBKfd8hVc4HphNro=</ds:DigestValue>
27        </ds:Reference>
28      </ds:SignedInfo>
29      <ds:SignatureValue>
30        IKIbLpkUnVsYPzrvgJGh7fZ5wohDP6AajmmQ9BDmgcvoFDlCZ3SB+tYnOLZwAaB2CmVOn5Sq6jG5
31        kATMpVVzb3Mlq8AOdSfNEC6LoijdhQMD4IlANoc+ple9UYMyKw7x1xw/90t5raRxe3u7nljDKbdo
32        Rw3E4kffuR03+rNlVJ4=
33      </ds:SignatureValue>
34      <ds:KeyInfo>
35        <ds:X509Data>
36          <ds:X509Certificate>
37            MIID3zCCA0igAwIBAgIEPMjmCjANBgkqhkiG9w0BAQUFADA2MQswCQYDVQQGEwJFUzENMAsGA1UE
38            ChMERk5NVDEYMBYGA1UECxMPRk5NVCBDbGFzZSAyIENBMB4XDTExMDIxNDEzMjMzMVoXDTE1MDIx
39            NDEzMjMzMVowdDELMAkGA1UEBhMCRVMxDTALBgNVBAoTBEZOTVQxGDAWBgNVBAsTD0ZOTVQgQ2xh
40            c2UgMiBDQTERMA8GA1UECxMIUFVCTElDT1MxEjAQBgNVBAsTCTUwMDA3MDAxNTEVMBMGA1UEAxQM
41            Ki5SRURTQVJBLkVTMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDMGDWFpIGu/RVURk1GRnwG
42            oNkhHp8kDAPPsCT5CL/abcBtAHjAlIt66A2naDIllExrvOyhiyQ0eypqkuzz1NsXofldf1KZ3N44
43            YtgYrUt7GSsL3U6wrTp8E2bX0Dh9hQTDbZh794PkooRSYgg810KyGy6jzDdzxjTqcjwhUpjcWwID
44            AQABo4IBujCCAbYwgakGA1UdEQSBoTCBnqSBjTCBijEYMBYGCCsGAQQBrGYBDxMJUzI4MzMwMDJF
45            MUowSAYJKwYBBAGsZgEOFDtNSU5JU1RFUklPIERFIFBPTM1USUNBIFRFUlJJVE9SSUFMIFkgQURN
46            SU5JU1RSQUNJ004gUNpCTElDQTEiMCAGCSsGAQQBrGYBCBMTV0lMRENBUkQuUkVEU0FSQS5FU4IM
47            Ki5SRURTQVJBLkVTMAkGA1UdEwQCMAAwKwYDVR0QBCQwIoAPMjAxMTAyMTQxMzI2MjRagQ8yMDE1
48            MDIxNDEzMjYyNFowCwYDVR0PBAQDAgWgMBMGA1UdJQQMMAoGCCsGAQUFBwMBMBEGCWCGSAGG+EIB
49            AQQEAwIGQDAdBgNVHQ4EFgQU+5z/JLpZeyEK91PUQRx4Q4sMjb8wHwYDVR0jBBgwFoAUQJp2RJd0
50            B8SsFMsejU86RXww12EwWwYDVR0fBFQwUjBQoE6gTKRKMEgxCzAJBgNVBAYTAkVTMQ0wCwYDVQQK
51            EwRGTk1UMRgwFgYDVQQLEw9GTk1UIENsYXNlIDIgQ0ExEDAOBgNVBAMTB0NSTDgxNzQwDQYJKoZI
52            hvcNAQEFBQADgYEAUoCRFnubo2SPx7egn4V1aSzATtEWGytrkkjA0LUw7pdR7UvbIKq//o1WQ7qd
53            W2PB83vd1vtWuZY0ZLHAi8PKODj5YWrIPlpZho65/jld8ZCg5EwzRnuIMgD32QwW0jwmc5FHFgBJ
54            bhbXdZVf84eHR3C4lh5CAbPsJ6Ly4CT8Db4=
55          </ds:X509Certificate>
```

```
      </ds:X509Data>
    </ds:KeyInfo>
  </ds:Signature>
  <saml2p:Status>
    <saml2p:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />
    <saml2p:StatusMessage>urn:oasis:names:tc:SAML:2.0:status:Success</saml2p:StatusMessage>
  </saml2p:Status>
  <saml2:Assertion ID="_e277e04ff5701f5d259fdc3246898e9b" IssueInstant="2016-07-23T10:28:04.265Z" Version="2.0">
    <saml2:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
      http://C-PEPS.gov.xx
    </saml2:Issuer>
    <ds:Signature>
      <ds:SignedInfo>
        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
        <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" />
        <ds:Reference URI="#_e277e04ff5701f5d259fdc3246898e9b">
          <ds:Transforms>
            <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
            <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
              <ec:InclusiveNamespaces xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#" PrefixList="xs" />
            </ds:Transform>
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
          <ds:DigestValue>dwz/en1Btf+OBE1EWU75Pu0QesE=</ds:DigestValue>
        </ds:Reference>
      </ds:SignedInfo>
      <ds:SignatureValue>
        kKXZE3TeeZxB+mcpBj6U8YRfnG/OCTe/I1b2b+eL4eRS2RQrIcYG4aZeOVp7ttGrBw4+/L8QAMY0
        kIZsAMPBtDlLnv8KOVyCGhuKIK2fgdgsRDNzTdnYy+k6f2+u6soJx1r5oInGy7EPGTdfp3+MdrOO
        ve5XwU6LhKg65r/nfCk=
      </ds:SignatureValue>
      <ds:KeyInfo>
        <ds:X509Data>
          <ds:X509Certificate>
            MIID3zCCA0igAwIBAgIEPMjmCjANBgkqhkiG9w0BAQUFADA2MQswCQYDVQQGEwJFUzENMAsGA1UE
            ChMERk5NVDEYMBYGA1UECxMPRk5NVCBDbGFzZSAyIENBMB4XDTExMDIxNDEzMjMzMVoXDTE1MDIx
            NDEzMjMzMVowdDELMAkGA1UEBhMCRVMxDTALBgNVBAoTBEZOTVQxGDAWBgNVBAsTD0ZOTVQgQ2xh
            c2UgMiBDQTERMA8GA1UECxMIUFVCTElDT1MxEjAQBgNVBAsTCTUwMDA3MDAxNTEVMBMGA1UEAxQM
            Ki5SRURTQVJBLkVTMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDMGDWFpIGu/RVURk1GRnwG
            oNkhHp8kDAPPsCT5CL/abcBtAHjAlIt66A2naDIllExrvOyhiyQ0eypqkuzz1NsXofldf1KZ3N44
            YtgYrUt7GSsL3U6wrTp8E2bX0Dh9hQTDbZh794PkooRSYgg810KyGy6jzDdzxjTqcjwhUpjcWwID
            AQABo4IBujCCAbYwgakGA1UdEQSBoTCBnqSBjTCBijEYMBYGCSsGAQQBrGYBDxMJUzI4MzMwMDJF
            MUowSAYJKwYBBAGsZgEOFDtNSU5JU1RFUklPIERFIFBPTM1USUNBIFRFUlJJVE9SSUFMIFkgQURN
```

```
99          SU5JU1RSQUNJ004gUNpCTElDQTEiMCAGCSsGAQQBrGYBCBMTV0lMRENBUkQuUkVEU0FSQS5FU4IM
100         Ki5SRURTQVJBLkVTMAkGA1UdEwQCMAAwKwYDVR0QBCQwIoAPMjAxMTAyMTQxMzI2MjRagQ8yMDE1
101         MDIxNDEzMjYyNFowCwYDVR0PBAQDAgWgMBMGA1UdJQQMMAoGCCsGAQUFBwMBMBEGCWCGSAGG+EIB
102         AQQEAwIGQDAdBgNVHQ4EFgQU+5z/JLpZeyEK91PUQRx4Q4sMjb8wHwYDVR0jBBgwFoAUQJp2RJd0
103         B8SsFMsejU86RXww12EwWwYDVR0fBFQwUjBQoE6gTKRKMEgxCzAJBgNVBAYTAkVTMQ0wCwYDVQQK
104         EwRGTk1UMRgwFgYDVQQLEw9GTk1UIENsYXNlIDIgQ0ExEDAOBgNVBAMTB0NSTDgxNzQwDQYJKoZI
105         hvcNAQEFBQADgYEAUoCRFnubo2SPx7egn4V1aSzATtEWGytrkkjA0LUw7pdR7UvbIKq//o1WQ7qd
106         W2PB83vd1vtWuZY0ZLHAi8PKODj5YWrIPlpZho65/jld8ZCg5EwzRnuIMgD32QwW0jwmc5FHFgBJ
107         bhbXdZVf84eHR3C4lh5CAbPsJ6Ly4CT8Db4=
108       </ds:X509Certificate>
109     </ds:X509Data>
110   </ds:KeyInfo>
111 </ds:Signature>
112 <saml2:Subject>
113   <saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified"
114     NameQualifier="http://C-PEPS.gov.xx">
115       urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified
116   </saml2:NameID>
117   <saml2:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
118     <saml2:SubjectConfirmationData Address="155.54.204.235"
119     InResponseTo="_9d1e2953ffed26c5a9b77f0dfce34ab4"
120     NotOnOrAfter="2016-07-23T10:33:04.242Z"
121     Recipient="https://despanishpeps.redsara.es/SP/ReturnPage" />
122   </saml2:SubjectConfirmation>
123 </saml2:Subject>
124 <saml2:Conditions NotBefore="2016-07-23T10:28:04.266Z" NotOnOrAfter="2016-07-23T10:33:04.242Z">
125   <saml2:AudienceRestriction>
126     <saml2:Audience>http://S-PEPS.gov.xx</saml2:Audience>
127   </saml2:AudienceRestriction>
128   <saml2:OneTimeUse />
129 </saml2:Conditions>
130 <saml2:AuthnStatement AuthnInstant="2016-07-23T10:28:04.268Z">
131   <saml2:SubjectLocality Address="155.54.204.235" />
132   <saml2:AuthnContext>
133     <saml2:AuthnContextDecl />
134   </saml2:AuthnContext>
135 </saml2:AuthnStatement>
136 <saml2:AttributeStatement>
137   <saml2:Attribute Name="http://www.stork.gov.eu/1.0/eIdentifier"
138     NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
139     stork:AttributeStatus="Available">
140     <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:anyType">
141       ES/ES/01234567A
```

```
142         </saml2:AttributeValue>
143       </saml2:Attribute>
144       <saml2:Attribute Name="http://www.stork.gov.eu/1.0/isTeacherOf"
145         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
146         stork:AttributeStatus="Available">
147         <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:anyType">
148           <stork:course xsi:type="xs:anyType">STORK</stork:course>
149           <stork:role xsi:type="xs:anyType">Docente</stork:role>
150           <stork:AQAA xsi:type="xs:anyType">2</stork:AQAA>
151           <stork:nameOfInstitution xsi:type="xs:anyType">University of Murcia</stork:nameOfInstitution>
152         </saml2:AttributeValue>
153       </saml2:Attribute>
154     </saml2:AttributeStatement>
155   </saml2:Assertion>
156 </saml2p:Response>
```

Appendix D

Attribute matching between eduGAIN and STORK2.0 federations

As we analyze in Sections 4.4 and 4.5, the translation process between eduGAIN and STORK federation is quite complex and each case has particularities that must be taken into account. Figures D.1 and D.2 show concrete examples of the results obtained in the process of translating attributes between both identity federations.

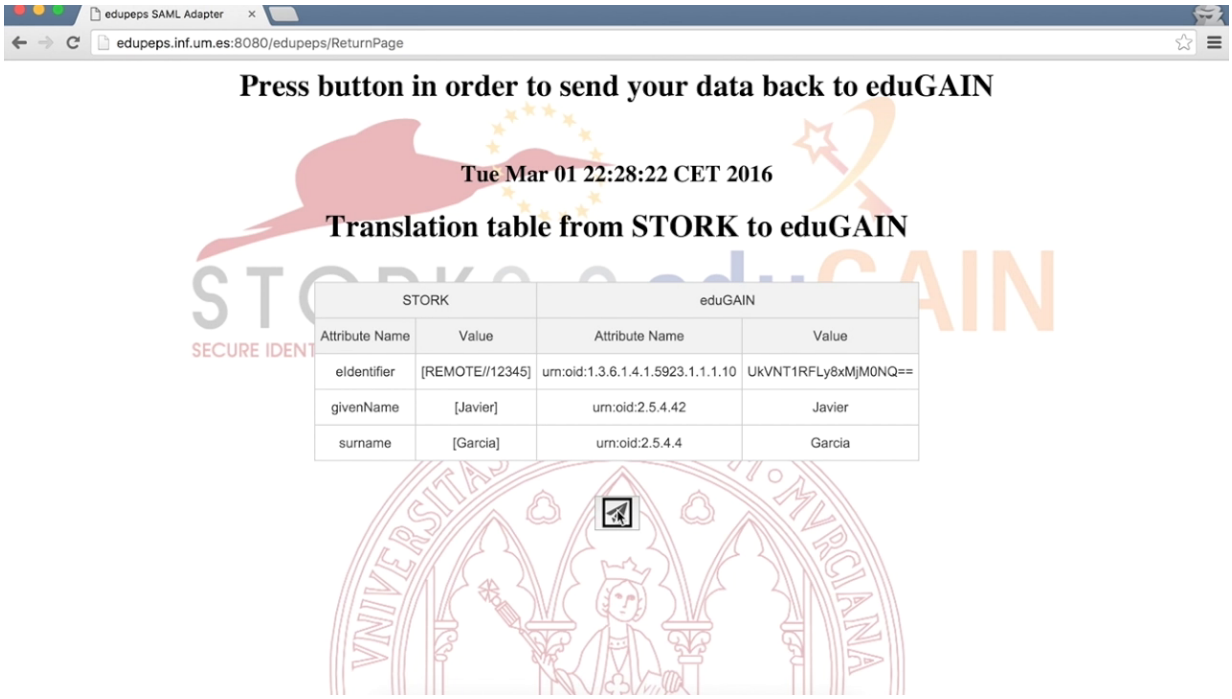


Figure D.1: Translation table example from STORK 2.0 identity to eduGAIN one.

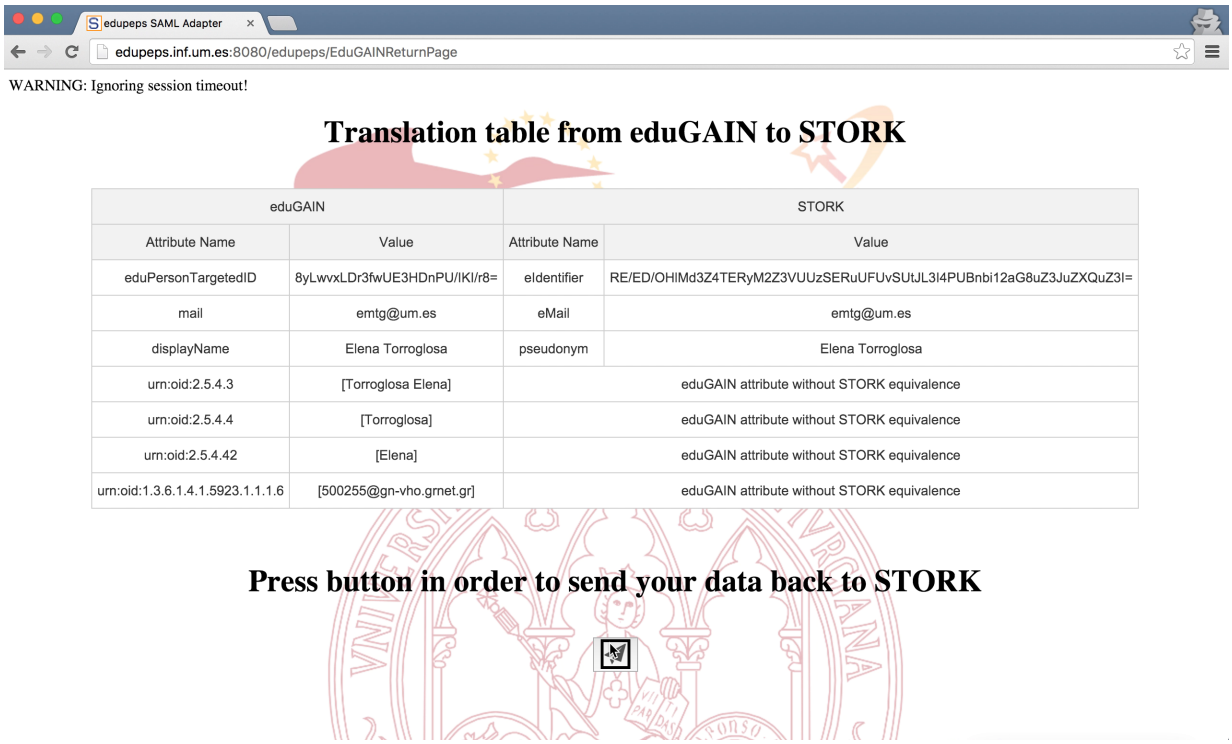


Figure D.2: Translation table from eduGAIN identity to STORK 2.0 one.