Servicio federado de eRúbrica para evaluación formativa

Federated e-Rubric service for formative assessment

José A. Accino, Elena Lozano

Resumen

En los últimos años hemos asistido a la consolidación de una nueva generación de tecnologías para la gestión de identidad y del acceso a los recursos. Una vez que los distintos mecanismos de gestión de autenticación parecen ya plenamente establecidos, la autorización constituye el siguiente problema a resolver. En esta comunicación se presenta una herramienta de gestión de grupos y permisos orientada al uso en un entorno federado. En torno a esa herramienta, se describe una arquitectura de agregación de servicios basada en tres elementos: proveedores de identidad, gestión de grupos y la aplicación o servicio que se presta. Se comentan algunos de los problemas encontrados, especialmente el del paso de atributos entre servicios y se describe la utilización de la biblioteca OAuth2lib, desarrollada por RedIRIS para implementar el Perfil de Aserción del protocolo OAuth 2.0, que se ha demostrado especialmente adecuada en este caso.

Palabras clave: Federación, autorización, gestión de grupos, evaluación, rúbrica.

Se presenta una herramienta de gestión de grupos y permisos orientada al uso en un entorno federado

Summary

In recent years we have witnessed the consolidation of a new generation of technologies for identity management and access to resources. Once the various mechanisms for authentication management seem to be fully established, authorisation becomes the next problem to be addressed. In this paper, a group and permission management tool is presented for use in a federated environment. Around this tool, a services aggregation architecture is described that is based on three elements: identity providers, group management and the application or service that is provided. We describe some of the problems encountered, especially in the transfer of attributes between services, and the use of the OAuth2lib library, developed by RedIRIS, for implementing the Assertion Profile of the OAuth 2.0 protocol, which has been shown to be especially suitable in this case.

Keywords: Federation, authorization, groups administration, evaluation, category.

1. Introducción

La autorización requiere una granularidad adecuada a cada aplicación o servicio

A lo largo de estos últimos años hemos asistido a la aparición y consolidación de una nueva generación de tecnologías en torno a la idea de federación para la gestión de identidad y del acceso a los recursos, a fin de resolver los problemas derivados de los procesos de identificación y autorización. Sin embargo, aunque los distintos esquemas de gestión de la autenticación parecen ya plenamente consolidados, la autorización sigue siendo un tema a resolver, principalmente debido a su mayor complejidad, ya que, mientras la autenticación es básicamente una cuestión de "todo o nada", la autorización requiere una granularidad adecuada a cada aplicación o servicio.

Resolver este problema resulta especialmente pertinente en un contexto como el actual en el que las instituciones pueden disponer y hacer uso de servicios dispersos por la red, muchos de los cuales, todavía, con mecanismos propios de identificación y la mayoría con esquemas internos de autorización. Es sabido que la *integración de identidades* es uno de los patrones detectados en la tendencia, creciente, hacia los entornos personales de aprendizaje (PLE), patrón estrechamente ligado al del *hub de conexión* (participación en diferentes redes y con distintos niveles de compromiso) [1]. Puede decirse que ambos patrones tienen su equivalente tecnológico en la gestión de identidad y en la gestión de autorización basada en roles y permisos.

2. Otras alternativas en gestión de grupos

Externalizar la autorización de un servicio implica la necesidad de mantener un sistema de control de roles y permisos en distintos grupos de usuarios. La gestión de grupos -u organizaciones virtuales- es un tema de actualidad y para ello existen ya varios desarrollos como Grouper, GMT o Sympa, así que ¿por qué no utilizar uno de estos?

En cuanto a Grouper [2], no es una solución que pueda llamarse "ligera", precisamente. Incluso con la interfaz desarrollada por CRU/ESCO no resulta demasiado amigable (aunque este aspecto ha mejorado con la interfaz de SURFnet). De todas formas, la gestión de los permisos atañe sólo al grupo y a sus documentos -no a las aplicaciones-, y es compleja de utilizar para el usuario final.

El GMT (Group Management Tool) de SWITCH [3] tiene dos limitaciones para el usuario: los grupos sólo los pueden crear los administradores globales y los roles sólo se pueden definir en la configuración inicial y por tanto sólo puede crearlos el administrador del sistema.

Por último, Sympa [4] es un excelente gestor de listas que puede exponer esas listas mediante una API pero, obviamente, eso no lo convierte en un gestor de grupos. De hecho, la "gestión" de grupos se maneja con un plugin Dokuwiki y se reduce a equiparar "administrador de la Wiki = dueño de la lista" y "usuarios de la wiki = suscriptores de la lista" [5].

En resumen, estas herramientas tienen su utilidad pero es necesario un gestor de grupos ligero, que pueda dar cabida a usuarios de distintas instituciones, que no tengan que ser expertos en tecnología, y que quieran disponer de una forma sencilla de manejar sus propios grupos -alumnos, participantes en un proyecto...-, con autonomía, asignando roles y permisos como consideren oportuno.

La gestión de los permisos atañe sólo al grupo y a sus documentos -no a las aplicaciones-, y es compleja de utilizar para el usuario final

3. Descripción del gestor de grupos

El gestor de grupos que presentamos (Figura 1) ofrece varias características diferenciales. Entre otras:

FIGURA 1: Portada del gestor de grupos

Sympa es un excelente gestor de listas que puede exponer esas listas mediante una API pero, obviamente, eso no lo convierte en un gestor de grupos

* Es una aplicación ligera, con una arquitectura modular ya probada en otras aplicaciones [6]. No se basa en ningún framework complejo y pesado, sino que utiliza una biblioteca de clases, Flourish [7], sencilla de utilizar pero potente, con prestaciones como ORM, Active Record y soporte de ACL, entre otras.

* Fácil de utilizar por los usuarios finales. Todas sus funciones se organizan en tres apartados: grupos y servicios, usuarios e invitaciones, roles y permisos.

* Los grupos se pueden activar y desactivar, editarlos para cambiar el nombre, las características, las fechas de inicio y final, asignarles los distintos servicios, y organizarlos jerárquicamente, es decir, como subgrupos de un grupo superior (Figura 2). La jerarquía se puede reorganizar en cualquier momento simplemente cambiando la asignación del correspondiente grupo padre.

Todas sus funciones se organizan en tres apartados: grupos y servicios, usuarios e invitaciones, roles y permisos

* Mecanismo integrado de invitación mediante direcciones de correo, que se introducen bien manualmente, bien leyéndolas de un archivo o ambas cosas al mismo tiempo. Las invitaciones se hacen siempre para uno de los roles establecidos para el grupo, ya sean los que se incluyen por defecto o los que el propietario del grupo haya definido.

Estas características están dirigidas a permitir a los usuarios finales una gestión autónoma de sus grupos, independientemente de la institución a la que pertenezcan, con el único requisito de que estén agrupadas en alguna federación, tal como el SIR de RedIRIS [8].

FIGURA 2: Grupos de un usuario

La jerarquía se puede reorganizar en cualquier momento simplemente cambiando la asignación del correspondiente grupo padre

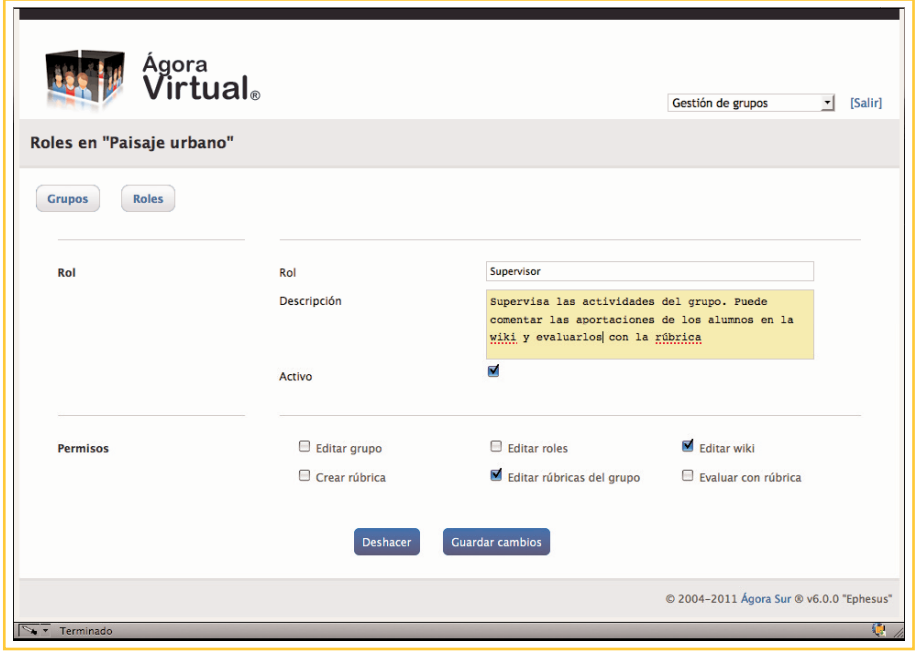
4. Roles y permisos

Para la gestión de la autorización se ha optado por listas de control de acceso (ACL) basadas en roles. La mayoría de los mecanismos de ACL existentes (phpGACL, Spiff Guard, CakePHP...) son globales por lo que las tablas que guardan los permisos tienen que saber también a qué tipo de objeto se están refiriendo. En nuestro caso, la ACL no necesita ser global, porque el grupo delimita de entrada a qué

objetos tiene acceso el usuario. Por tanto la ACL sólo tiene que controlar qué acciones se permiten dentro del grupo.

El propietario de un grupo puede gestionar fácilmente los roles del grupo, editar los existentes o añadir otros nuevos, activando para un determinado rol las funciones que considere oportunas, por ejemplo "editar wiki", "evaluar con rúbrica", dependiendo de los servicios que el grupo tenga asignados (Figura 3).

FIGURA 3: Edición de rol y permisos



El propietario de un grupo puede gestionar fácilmente los roles del grupo, editar los existentes o añadir otros nuevos

Naturalmente, el problema es cómo hacer que la aplicación sepa lo que significa un determinado rol y, sobre todo, lo que el propietario de un grupo quiere que ese rol signifique en ese grupo. Por tanto, habrá que ver cómo y dónde vincular cada rol con las funciones de la aplicación. Se podría hacer en la propia aplicación, pero entonces ésta debería tener conocimiento previo de todos los roles, con lo cual el propietario de un grupo no tendría libertad para definir los que quisiera. Es más fácil -y más lógico de administrar para el usuario final- que el gestor de grupos conozca las funciones que se hacen en las aplicaciones que no al revés, lo que significa que cuando se añade un nuevo servicio al entorno, el administrador del sistema deberá hacer saber al gestor de grupos los permisos pertinentes (por ejemplo, para una wiki: leer, editar, administrar), de manera que estén disponibles para que los propietarios de los grupos puedan utilizarlos en sus roles como vean oportuno.

El problema es cómo hacer que la aplicación sepa lo que significa un determinado rol y lo que el propietario de un grupo quiere que ese rol signifique en ese grupo

5. Una arquitectura de agregación de servicios

Como ejemplo de utilización se propone una arquitectura de servicios en la que los usuarios de varias instituciones pueden compartir el acceso a una herramienta de evaluación o Rúbrica.

FIGURA 4: Rúbrica

Criterios	Indicadores					Valor
Contextualización						
Justifica el tema / centro de interés	No hace ninguna justificación 0	Sí, pero brevemente 2.5 ✓	Justifica con el PCC 5	Justifica con el PCC y el PEC 7.5	Justifica con el PCC, el PEC y el currículo oficial de etapa 10	2.5/10
Selección de objetivos y contenidos						
Define los objetivos en términos de competencias	No lo hace en términos de competencias 0	No lo hace en términos de competencias, sólo algunos 2.5	Realiza todos los objetivos en términos de competencias 5 ✓	Realiza todos los objetivos en términos de competencias haciendo referencia a los objetivos de etapa y/o ciclo 7.5	Realiza todos los objetivos en términos de competencias haciendo referencia a los objetivos de etapa, ciclo y área 10	5/10
Realiza una presentación de los contenidos ajustada a criterios dados para el diseño de la unidad	Los confunde y hay faltas graves	identifica el carácter conceptual	identifica el carácter conceptual y procedimental	Identifica el carácter conceptual, procedimental y actitudinal	identifica el carácter conceptual, procedimental y actitudinal. Con clara relación con los objetivos y actividades	

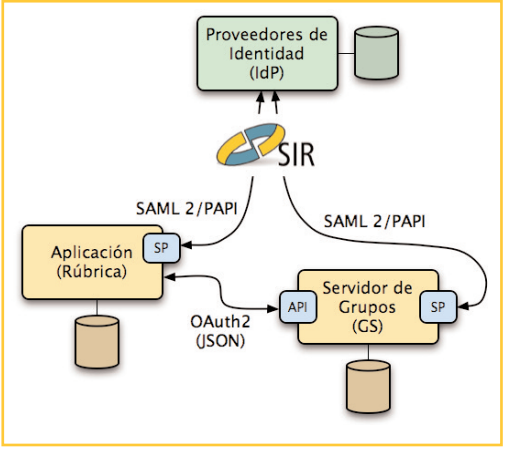
La Rúbrica consiste en una matriz -de una o más dimensiones- para evaluación de competencias o conocimientos

La Rúbrica (Figura 4) consiste en una matriz -de una o más dimensiones- para evaluación de competencias o conocimientos y constituye un marco de referencia para que profesores y alumnos tengan una idea clara sobre los criterios a aplicar en dicha evaluación [9]. La herramienta utilizada permite, -más allá de los simples editores HTML disponibles en la red- asignar criterios e indicadores y dar un peso específico a cada uno, entre otras prestaciones.

La actividad de la Rúbrica se organiza de forma natural en base a grupos

La actividad de la Rúbrica se organiza de forma natural en base a grupos: por ejemplo, el profesor puede evaluar una competencia en un grupo de alumnos, pero también los alumnos pueden evaluarse entre sí. Como es obvio, el servicio de Rúbrica podría gestionar esos grupos internamente, pero externalizando esa gestión podrán ser también utilizados por otras aplicaciones o servicios sin necesidad de recrearlos en cada uno de ellos.

FIGURA 5: Arquitectura del servicio

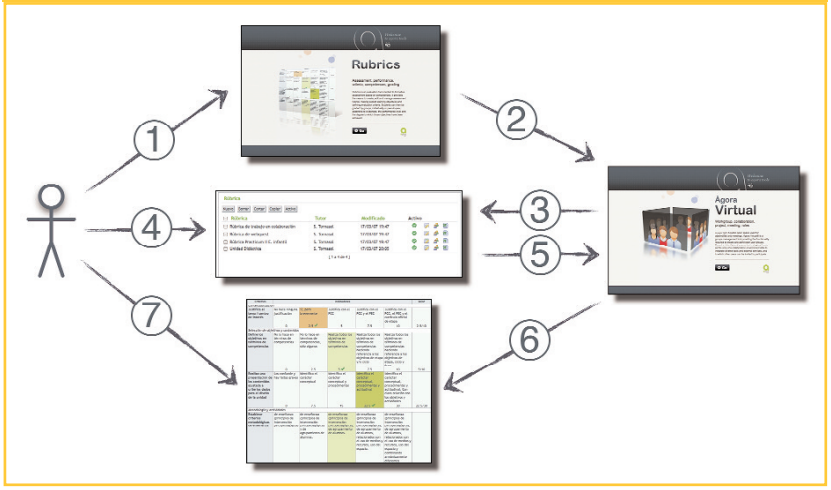


La arquitectura consta de tres bloques: proveedores de identidad dentro de la federación, el servicio de gestión de grupos y el servicio que se presta, la Rúbrica en este caso (Figura 5).

En esta arquitectura, una posible secuencia de autorización podría ser (Figura 6):

1 - Un usuario accede a la Rúbrica (y se autentica en el Proveedor de Identidad).
2 - La Rúbrica pide al Gestor de Grupos la lista de grupos a los que pertenece el usuario.
3 - Se le muestran las rúbricas de esos grupos.
4 - El usuario intenta una acción sobre una de ellas: ver, editar, evaluar...
5 - Se piden al Gestor de Grupos los permisos del usuario en el grupo al que pertenece esa rúbrica (que dependerán de su rol en ese grupo).
6-7- Se permite al usuario actuar, o no, según esos permisos (por ej. evaluar a un alumno).

FIGURA 6: Secuencia de autorización



6. La biblioteca oauth2lib

OAuth es un estándar abierto para incluir seguridad en la autenticación de aplicaciones web y de escritorio

Por tanto, para ser útil como fuente de autorización, el servicio de gestión de grupos debe tener un medio de proporcionar a las aplicaciones la información pertinente acerca de los grupos, sus usuarios y roles mediante algún protocolo que garantice seguridad e integridad. Para este fin se ha optado por utilizar el protocolo de autenticación y autorización denominado OAuth[10].

OAuth (Open Authorization = Autorización Abierta) es un estándar abierto para incluir seguridad en la autenticación de aplicaciones web y de escritorio. Consiste, a grandes rasgos, en una herramienta que permite a una aplicación cliente actuar en nombre de un usuario concreto para obtener unos recursos que son propiedad del mismo y que están alojados en un servidor de recursos determinado.

Los elementos básicos de la arquitectura de OAuth son los siguientes:

Para el caso de la aplicación de Rúbrica, el perfil necesario para su implementación es el denominado Perfil Autónomo

* **Aplicación cliente:** Aplicación que solicita y utiliza la información del usuario para un fin concreto. En este caso la aplicación cliente es la aplicación de Rúbrica.

* **Servidor de autorización:** Servidor que recoge la información de la aserción, de la aplicación cliente y del recurso al cual se desea acceder y devuelve un código de acceso a los recursos para ser utilizado en el servidor de recursos.

* **Servidor de recursos:** Servidor que permite acceder a los recursos con un código de acceso determinado. Ambos servidores, en el caso de uso de la aplicación de Rúbrica se establecen en el Servidor de Gestión de Grupos.

En la versión 2 del protocolo OAuth existen diferentes perfiles que se ajustan a distintos casos de uso. Para el caso de la aplicación de Rúbrica, el perfil necesario para su implementación es el denominado Perfil Autónomo. Este perfil se ajusta de forma más adecuada a este caso de uso ya que, en vez de utilizar al usuario directamente en todas las peticiones de acceso, se delega esta autorización en los atributos del usuario que están contenidos en una aserción. Por este motivo, además de los elementos comentados anteriormente, este perfil establece la figura del Proveedor de Identidad, el cual será el

encargado de generar las aserciones de los usuarios en base a sus atributos. En el caso que nos atañe, esta función la realiza un Servicio de Identidad como el SIR de RedIRIS.

Los pasos principales que se realizan para proveer acceso a unos recursos restringidos mediante OAuth son los siguientes (Figura 7):

1. El Cliente obtiene una aserción adecuada

La aplicación cliente, en este caso la Rúbrica, se comunica con un proveedor de identidad, el SIR, para que el usuario se autentique en este proveedor y se genere una aserción con los atributos del mismo, cuya duración vendrá determinada según el tiempo de vida que se haya configurado.

2. Obtención del Token de Acceso

La aplicación cliente, en este caso la Rúbrica, se comunica con un proveedor de identidad, el SIR

La aplicación cliente -la Rúbrica- envía la aserción obtenida en el paso anterior junto con el scope u objetivo al que desea acceder (una entrada a la API del Gestor de Grupos) y las credenciales de ella misma, como aplicación cliente. Éstas credenciales propias han debido ser obtenidas en pasos previos al flujo de autorización, mediante el registro y configuración de la misma en el servidor de autorización. Estos datos permitirán al servidor de autorización garantizar que la aplicación cliente está registrada y no es una aplicación fraudulenta que desea hacer un uso ilícito de los datos, que ésta puede acceder al scope especificado y que el usuario, representado por la aserción, tiene los privilegios suficientes para acceder a los recursos.

En el caso de que se cumplan todas las condiciones de seguridad, el servidor de autorización devolverá un token de acceso, el cual permitirá acceder en el servidor de recursos al scope especificado y tendrá un tiempo de vida concreto, que se habrá especificado previamente al configurar el servidor.

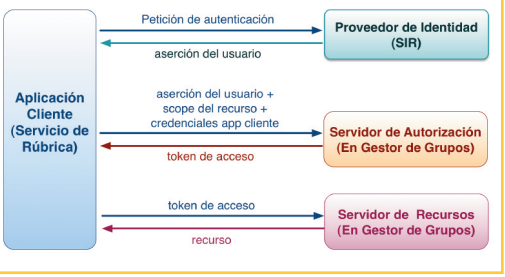
3. Obtención de los recursos

La aplicación de Rúbrica envía al servidor de recursos el token de acceso y, si este es válido y no ha expirado, el servidor responderá con el recurso solicitado por la aplicación cliente, en este caso una cadena JSON con la información requerida.

Para aplicar este protocolo a la aplicación de Rúbrica, se ha utilizado oauth2lib

Para aplicar este protocolo a la aplicación de Rúbrica, se ha utilizado oauth2lib[11] una biblioteca de código abierto desarrollada por RedIRIS que implementa el perfil autónomo o de aserción del protocolo OAuth en su versión 2. Además, se han añadido otras características de seguridad y de usabilidad para el funcionamiento más seguro y funcional de la biblioteca:

FIGURA 7: Uso del protocolo OAUTH2



* Integración con sistemas federados como el SIR mediante aserciones SAML2 y PAPI.

* Fácil adición de funciones de procesado de diferentes tipos de respuestas e integración de nuevos recursos gracias a la organización modular de la arquitectura de la biblioteca.

* Configuraciones sencillas y personalizadas mediante XML.

* Establecimiento de confianza entre Servidores de Autorización, aplicaciones Cliente y Servidores de Recursos que evita que algunos servidores o aplicaciones que se hagan pasar por otros válidos.

* Registro de aplicaciones clientes y scopes tanto en los Servidores de Autorización como en los de Recursos mediante archivos en XML que permiten garantizar que sólo accedan a los recursos los que tengan privilegios para ello.

* Posibilidad de establecer políticas de acceso diferentes según las aserciones y los atributos que en ellas se encuentren.

7. Conclusiones

Las tecnologías de identidad hacen posible un escenario más acorde con la evolución del entorno tecnológico-educativo en general, en el que los servicios pueden ofertarse a toda la comunidad a través algún esquema de federación, con la consiguiente economía de recursos y repercusión en las buenas prácticas de colaboración entre instituciones. Este modelo requiere, sin embargo, además de los esquemas de autenticación ya conocidos, un servicio en el que los usuarios puedan gestionar sus grupos y algún medio seguro de transferencia de información entre los distintos servicios. La arquitectura presentada para el caso específico de un servicio de Rúbrica es un ejemplo de uso de este modelo, en el que el usuario final tiene plena capacidad para gestionar sus grupos. A partir de la aserción emitida por el proveedor de identidad, la biblioteca Oauth2lib de RedIRIS permite a las aplicaciones acceder a la información necesaria relativa a las autorizaciones de que dispone el usuario según su rol en el grupo, a fin de ajustar su respuesta a dichas autorizaciones.

Las tecnologías de identidad hacen posible un escenario más acorde con la evolución del entorno tecnológico-educativo

Referencias

[1] Wilson, Scott : Patterns of personal learning environments. (Educational Cybernetics: Journal Articles) University of Bolton Institutional Repository. http://digitalcommons.bolton.ac.uk/cgi/viewcontent.cgi?article=1005&context=iec_journalspr

[2] http://www.internet2.edu/grouper/

[3] http://www.switch.ch/aai/support/tools/gmt.html

[4] http://www.sympa.org/contribs/sympaauth

[5] En nuestra opinión, debería ser al revés: en vez de utilizar las listas como fuente de grupos, se debería aprovechar la capacidad de Sympa para usar los grupos como fuentes para sus listas.

[6] Ágora Virtual: Una propuesta de entorno colaborativo y de enseñanza sobre interfaces OSID. Boletín de RedIRIS núm 76, abril 2006. http://www.rediris.es/difusion/publicaciones/boletin/76/enfoque1.pdf

[7] http://flourishlib.com/

La biblioteca Oauth2lib de RedIRIS permite a las aplicaciones acceder a la información necesaria

[8] http://www.rediris.es/sir

[9] Cebrián, M.; Accino. J.A.; Raposo, M.: Formative evaluation tools for the European Area of Higher Education: ePortfolio and eRubric. EUNIS Conference 2007. Grenoble (Francia). http://www.eunis.org/events/congresses/eunis2007/CD/pdf/papers/p85.pdf

[10] http://oauth.net

[11] http://www.rediris.es/oauth2/

José A. Accino
(accino@uma.es)
Servicio Central de Informática
Universidad de Málaga

Elena Lozano
(elena.lozano@prise.es)
PRiSE-Sevilla