PAPI-EE: Integrando PAPI con Shibboleth

PAPI-EE: adapting PAPI to Shibboleth

C. Rodríguez

Resumen
Tras la publicación de SAML por parte de OASIS como estándar para la intercomunicación entre los diferentes componentes que se encuentran en las infraestructuras de autenticación y autorización, ha tenido un fuerte crecimiento las migraciones de dichas infraestructuras hacia el protocolo de Shibboleth. Es por ello que PAPI debía tener la capacidad de comunicarse con este software, por lo que se convirtió en una de las líneas de trabajo principales de PAPI Enterprise Edition.

Palabras claves: privacidad, identificación, seguridad

Summary
Alter the publication of SAML, by OASIS, as the standard used for intercomucatting different components which are found in authentication and authorization infrastructures, migrations to Shibboleth-based protocols has gone up quite a bit. So PAPI must be able to communicate with this software, becoming one of the most important tasks in PAPI Enterprise Edition.

Keywords: privacy, identification, computer network security

1.- Introducción

Recientemente ha surgido la iniciativa PAPI Enterprise Edition (PAPI EE) [1] con el objetivo principal de desarrollar componentes y librerías en PAPI que sirvan de puente a los actuales ámbitos tecnológicos construidos a partir de Java.

Dentro de este entorno de actuaciones, una de las primeras líneas a realizar está siendo la interoperabilidad entre los distintos componentes de PAPI y Shibboleth [2]. Éste es una iniciativa de Internet2 para crear infraestructuras de autenticación y autorización, proporcionando mecanismos de Single Sign-On (SSO) [3] en un entorno tecnológico y de control de acceso a recursos protegidos.

Además, dicho organismo orientó el desarrollo del protocolo de comunicaciones de Shibboleth como la implementación de referencia del estándar Security Assertion Markup Language (SAML) [4] promovido por OASIS.

Actualmente, SAML es el lenguaje adoptado por la industria para comunicarse entre los distintos componentes que interactúan en una infraestructura de autenticación y autorización (IAA). Además, algunas IAA académicas, como las existentes en Estados Unidos y en Suiza también han adoptado SAML como la base de sus comunicaciones. Es por ello, que era necesario que PAPI pudiera entender dicho lenguaje.

2.- Shibboleth in a nutshell

Shibboleth se compone de los siguientes componentes principales:

- **Proveedor de Identidad** (IdP): gestiona las credenciales y los atributos de los usuarios. Cada vez que le llega una petición de una entidad de confianza, normalmente serán Proveedores de Servicios (SP), genera sentencias de

autenticación o atributos para incluirlos en la respuesta. Un IdP a su vez está compuesto, entro otros componentes, por:

  - **Servicio de Single Sign-On**: inicia el proceso de autenticación del usuario, redirigiéndolo a la aplicación que le devolverá sus credenciales en el caso de que no se haya identificado previamente.
  - **Autoridad de Atributos**: procesa todas las peticiones de atributos y emite asertos de atributos.

- **Proveedor de Servicio** (SP): gestiona los recursos protegidos, habilitando el acceso al usuario en función de los asertos obtenidos en su identificación en el IdP. El SP dispone de la ayuda de otros subcomponentes, como:
  - **Attribute Requester**: se encarga de obtener los atributos del usuario sin necesidad de pasar a través de su navegador, creando un canal seguro entre el IdP y el SP.

Como hemos comentado, Shibboleth utiliza SAML para el intercambio de mensajes entre sus componentes e implementa todos los perfiles de SAML. Un perfil SAML describe los intercambios HTTP que se requieren para transferir asertos entre un IdP y un SP. De esta forma, si un software es compatible con SAML lo es a su vez con Shibboleth y viceversa.

3.- Integración

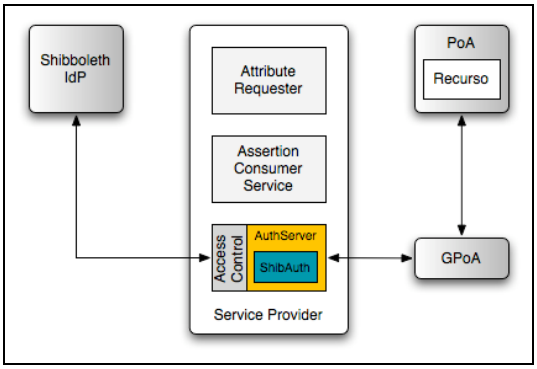
Para integrar PAPI con Shibboleth es necesario que sea posible la comunicación entre los distintos componentes de ambas tecnologías de manera que:

1. Un *PoA de PAPI* pueda responder a la petición de acceso de un usuario en función de las credencias y los atributos devueltos por un *IdP de Shibboleth*.
2. Un *SP de Shibboleth* pueda realizar las operaciones de control de acceso a través de las credenciales y atributos devueltos por un *AS de PAPI*.

El objetivo principal que se ha mantenido como base del proyecto de cara a esta tarea de hacerlos compatibles es reducir al máximo la generación de software para este proceso, puesto que aumentaría los costes de desarrollo y facilitaría la aparición de errores incontrolados.

3.1.- Interconexión entre un PoA y un IdP

En el siguiente diagrama podemos ver la solución propuesta para resolver la problemática del primer punto:



En este caso, necesitamos que un AuthServer de PAPI quede protegido por un SP de Shibboleth, de manera que cuando un usuario intente obtener sus credenciales, antes de acceder al AuthServer tendrá que autenticarse en el IdP de Shibboleth. Como el objetivo es que se valide el acceso (tanto con las credenciales como con los atributos) en

el PoA con los datos obtenidos en el IdP, el AuthServer estará configurado con el módulo ShibAuth, el cual autentica el usuario y emite los asertos de atributos en función de lo obtenido en la autenticación en el IdP.

La configuración del AuthServer describiría estas acciones de la siguiente manera:

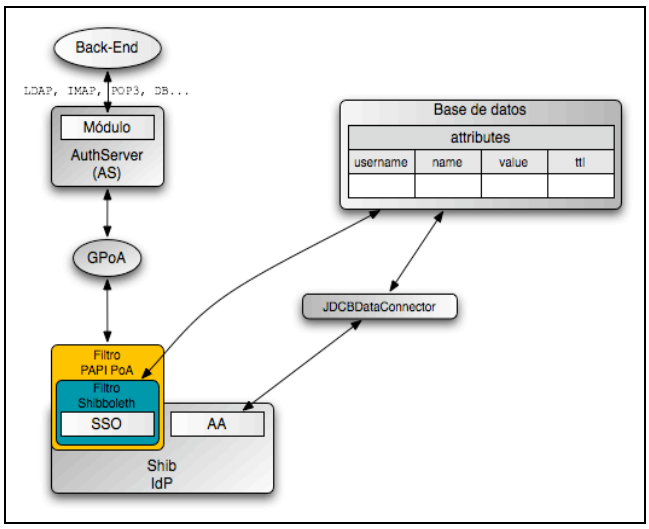
```
$$cfg{authenticationHook} = \&ShibAuth::VerifyUser;
$$cfg{credentialHook} = \&ShibAuth::UserCredentials;
$$cfg{attrRequestHook} = \&ShibAuth::UserAttributes;
$$cfg{SHIBUserHeader}='HTTP_SHIB_ORIGIN_SITE';
$$cfg{SHIBListAttr}={'HTTP_SHIB_USER_ROLE' => 'SHIBrole',
                     'HTTP_SHIB_USER_INST' => 'SHIBinst'};
$$cfg{defAssertion} = 'role=<papi var="SHIBrole"/>,
                       inst=<papi var="SHIBinst"/>';
```

Además de configurar el módulo de autenticación, utilizamos los siguientes parámetros de configuración:

- **SHIBUserHeader**: indicamos el nombre de la cabecera HTTP en el cual el SP va a escribir el identificador generado para el usuario. En el caso de que el usuario no haya obtenido unas credenciales válidas, esta cabecera no es enviada y el AuthServer entenderá de que no se ha autenticado correctamente.
- **SHIBListAttr**: realiza un mapeo entre cabeceras HTTP donde contienen valores de sus atributos y el nombre de la variable de PAPI a la cual se le copiará dicho valor.

3.2.- Interconexión entre un SP y un AuthServer

Para resolver la problemática del segundo punto utilizaremos un despliegue como describe el siguiente diagrama:



En este caso, necesitamos que un SP pueda resolver los derechos de acceso a través de la información emitida por un AuthServer. Al requerir un IdP el acoplamiento de un software que realice las funciones de autenticación, es posible presentar una solución muy transparente. Como vemos, en este caso protegemos en el Tomcat el módulo de SSO con un filtro de Shibboleth y éstos a su vez con un filtro de PoA.

De esta forma, cada vez que un SP pregunte al módulo de SSO de un IdP por las credenciales del usuario, primero el filtro de PoA capturará la petición y posteriormente el filtro de Shibboleth. Éste tiene como objetivo procesar las credenciales emitidas por

el AuthServer de PAPI y almacenar en una base de datos por cada uno de los atributos los siguientes datos:

- **username**: identificador del usuario, el cual es el utilizado también por el IdP y que es el propio filtro el que se lo transmite. Actualmente se genera un UUID como identificador.
- **name**: nombre del atributo.
- **value**: valor del atributo.
- **ttl**: tiempo de vida del atributo. Este valor se genera en función del tiempo de vida especificado en el archivo de configuración del filtro y se eliminarán aquellos atributos que hayan superado su valor.

De esta forma, se configura la Autoridad de Atributos (AA) para que utilice el conector de base de datos, llamado *'JDBCDataConnector'*, que es distribuido dentro del software de IdP. Éste buscará los valores de los atributos del usuario cada vez que reciba una petición en la base de datos.

4.- Conclusiones

En este artículo hemos presentado una solución transparente para interconectar todos los componentes principales de Shibboleth y PAPI, gracias a la cual es posible hacer compatible tanto un PoA y un IdP como un SP y un AuthServer.

Además, en el caso de que ya existiera un entorno PAPI desplegado en nuestra red, no habría que realizar ningún cambio significativo en dicho entorno a la hora de incluir un SP en él. De esta forma, se puede cubrir la reciente necesidad de interoperar con federaciones basadas en Shibboleth, ya que la mayoría de proveedores de contenido para redes académicas están comenzado a basar su infraestructura en la tecnología SAML.

Referencias

[1] *PAPI Enterprise Edition*. https://forja.rediris.es/projects/papi-ee/

[2] S. C. M. Erdos. *Shibboleth architecture draft v05*. http://middleware.internet2.edu/shibboleth/docs/draft-internet2-shibboleth-arch-v05.pdf, 2002.

[3] C. J. M. A. Pashalidis. *A taxonomy of single sign-on systems*. *Lecture Notes on Computer Science*, 2727:249–264, 2003.

[4] P. H.-B. et al. *Assertions and protocols for the oasis security assertion markup language (saml)*. *OASIS SSTC*, November 2002.