Trabajando con infraestructuras de autenticación y autorización

Working with Authorization and Authentication Federated Infrastructures

C. Rodríguez, D. R. López y J. M. Macías

Resumen

La implementación de una Infraestructura Federada de Autenticación y Autorización es una solución para la necesidad de que diferentes entidades accedan a servicios basados en red, la cual, generalmente, implica un alto coste para la organización. Para ayudarnos en el proceso de la integración, los autores presentan AA-RR, una herramienta que ayuda en la validación de la interoperabilidad entre diferentes componentes de Autenticación y Autorización.

Palabras clave: Infraestructura federada, autenticación y autorización, AA-RR

Summary

The implementation of an Authentication and Authorization Federated Infrastructure is a solution to the demand for inter-organizational access to networked services. Normally, this implies a high cost for the organization.

To help us in this integration, the authors present AA-RR, a tool conceived to help in the validation of the interoperability of a certain Authentication and Authorization component with others.

Keywords: Federate Infrastructure, authentication and authorization, AA-RR

1.- Introducción

Las Infraestructuras Federadas de Autenticación y Autorización (IFAA) permiten a los usuarios de las organizaciones federadas obtener acceso a los recursos protegidos, generalmente a través de protocolos de Single Sign-On (SSO). En este proceso los usuarios sólo necesitan identificarse una vez, realizándose normalmente en sus dominios locales, para poder obtener derechos de acceso a los recursos protegidos que poseen un control de acceso.

Las Infraestructuras Federadas de Autenticación y Autorización permiten a los usuarios obtener acceso a los recursos protegidos, generalmente a través de protocolos de Single Sign-On

Podemos encontrar numerosos proyectos de IFAA donde Liberty Alliance [1] y Shibboleth [2] utilizan *Security Assertion Markup Language* (SAML) [3, 4], un estándar XML para el intercambio de información de autenticación y autorización entre dominios de seguridad y publicado por OASIS. Pero de hecho, hay más proyectos que utilizan sus propios protocolos de comunicaciones, como PAPI [5] o A-Select [6].

Debido a que sigue creciendo la adopción de IFAA en los entornos distribuidos, se ha generado la necesidad de que puedan interoperar entre sí los distintos componentes tecnológicos de dichos entornos con los servicios que ofrece la infraestructura federada. El principal problema que nos encontramos es que aun queriendo los desarrolladores actualizar sus componentes, en la mayoría de los casos necesitan desplegar una IFAA para comprobar la interoperabilidad con ésta, con la problemática de tener que conocer todos los posibles mensajes en el protocolo de comunicaciones, lo cual complica nuestro conjunto de tests. Además, existen algunos protocolos de comunicaciones de autenticación adoptados por la industria, como por ejemplo Radius [7], con el que también podríamos querer ser compatibles.

Authentication and Authorization Requester-Responder (AA-RR) es una herramienta diseñada para ayudar en el proceso de la validación de la interoperabilidad de cierto componente de Autenticación y Autorización con otros

Authentication and Authorization Requester-Responder (AA-RR) es una herramienta diseñada con el objetivo de ayudar en el proceso de validación de la interoperabilidad de cierto componente de Autenticación y Autorización (AA) con otros. En AA-RR podemos emular el comportamiento de cualquier componente usando un fichero de configuración descrito en XML, donde definimos los

estados de la emulación y las acciones a realizar como cambiar de estado activo en función de los mensajes recibidos o enviados. Además, podemos elegir el protocolo de comunicaciones, como SAML versión 1.1, lo cual nos permite trabajar en aquellos productos que necesitan interoperar con Infraestructura de Autenticación y Autorización (IAA).

2.- Aprendiendo SAML

En la tecnología SAML 1.1, la información se transfiere entre elementos de afirmación (*asserting parties*), también conocido como autoridad SAML, y elementos de confianza (*relying parties*) utilizando mensajes XML. La comunicación necesaria para obtener dichos mensajes la realizaremos a través de peticiones y respuestas entre ambas partes, obteniendo como resultado una aserción. Éste es una unidad de información que contiene sentencias emitidas por una autoridad SAML para un sujeto determinado. Dentro de un aserto podemos añadir tres tipos de sentencias:

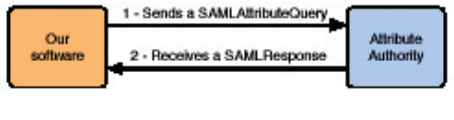
En la tecnología SAML 1.1, la información se transfiere entre elementos de afirmación y elementos de confianza utilizando mensajes XML

- *Autenticación*. Usamos esta sentencia cuando queremos afirmar que un usuario se autenticó a través de un mecanismo y momento determinado. Como podemos ver, sólo se obtiene información sobre la acción de autenticación, no la realización de dicha acción.
- *Decisión de autorización*. Usamos esta sentencia para informar si aceptamos o denegamos el acceso a un usuario a un recurso protegido. Además, podemos recibir información extra en una petición de decisión de autorización que el emisor haya considerado que pueda ser determinante a la hora de evaluar nuestra respuesta.
- *Atributo*. Usamos esta sentencia cuando queremos establecer una relación entre un determinado usuario y una serie de atributos.

  En SAML, enviamos información a través de un intercambio de mensajes SOAP sobre HTTP, el único *binding* definido en su versión 1.1, facilitando el despliegue del *software* distribuido en nuestro entorno. Pero, si queremos asegurarnos el éxito de la interoperabilidad de nuestro sistema con otros, debemos seguir un determinado flujo de mensajes de petición/respuesta llamado *perfil*.

Usamos la sentencia de autenticación cuando queremos afirmar que un usuario se autenticó a través de un mecanismo y momento determinado

FIGURA 1: CONSULTA DE ATRIBUTO ENTRE NUESTRO SW Y UNA AUTORIDAD DE ATRIBUTO



(a) Petición/Respuesta de atributos en SAML

(b) Ejemplo de petición de atributo en SAML

(c) Ejemplo de respuesta de atributo en SAML

2.1.- Peticiones

En una petición SAML podemos solicitar una aserción determinada, referencia por un identificador único, o pedir información sobre autenticación, decisión de autorización o atributos. En la figura 1, podemos ver un ejemplo de petición SAML así como el mensaje XML que se envía en dicha petición, representado por el elemento `<samlp:Request>`. Utilizando `<saml:AttributeQuery>` estamos indicando que es una petición de atributos, y a través de su atributo Resource referenciamos quién está interesado en la respuesta. Este elemento incluye los siguientes en la petición de atributo:

- `<saml:Subject>`. Define el usuario sobre el que solicitamos la petición, normalmente a través de `<saml:NameIdentifier>` el cual no sólo especifica la identificación del usuario, como *juan@rediris.es*, sino que además incluye el tipo de identificación.
- `<saml:AttributeDesignator>`. Usaremos este elemento cuando queramos especificar los atributos que deseamos consultar. En el caso de que queramos preguntar por todos los atributos posibles, no incluiríamos este elemento. Como podemos ver en nuestro ejemplo, estamos preguntando por el atributo *eduPersonScopedAffiliation*.

2.2.- Respuestas

Para responder a una petición SAML utilizaremos una respuesta SAML, informando de si ha ocurrido algún error e incluyendo una lista de aserciones. En la figura 1c podemos ver la respuesta de nuestro ejemplo, comenzando en el elemento `<samlp:Response>`. La petición que se ha recibido es correcta y por tanto informaremos de ello en el elemento `<samlp:Status>`. Además, añadiremos una aserción en la respuesta que incluya una sentencia de atributo, `<saml:AttributeStatement>`.

En una petición SAML podemos solicitar una aserción determinada o pedir información sobre autenticación, decisión de autorización o atributos

3.- Authentication And Authorization Requester-Responder (AA-RR)

Authentication and Authorization Requester-Responder es un sistema que utiliza un determinado tipo de *metadatos* para describir los requerimientos de una infraestructura. El objetivo principal es validar o testear la interoperabilidad de un determinado componente de AA con otros.

El grupo de investigación *IRTF Authentication, Authorization and Accounting Architecture* (AAAARCH) realizó un estudio sobre *Autenticación, Autorización y Accounting* (AAA) definiendo su arquitectura y *framework*. En nuestra herramienta podemos emular los siguientes componentes, fundamentales en dicha arquitectura:

- *Policy Information Point*. Proporciona a otros componentes de una IAA mecanismos de verificación sobre las credenciales y los atributos de un sujeto.
- *Policy Enforcement Point*. Representa el punto de acceso de un recurso protegido donde se realizará la comprobación de las credenciales del sujeto que intenta acceder.
- *Policy Decision Point*. Realiza una decisión de autorización basada en un conjunto de reglas o una política determinada.
- *Attribute Authority*. Gestiona los atributos de los usuarios pudiendo estos ser solicitados por otros componentes.

A la hora de interactuar con una IAA, disponemos de una serie de protocolos potenciales. Aunque está reconocido SAML como un estándar, hay otras IAAs en producción que tienen su propio protocolo de comunicaciones para interoperar y además, un conjunto elevado de aplicaciones que

Aunque está reconocido SAML como un estándar, hay otras IAAs en producción que tienen su propio protocolo de comunicaciones

están integradas en ellas. En nuestras emulaciones en el AA-RR, tenemos la posibilidad de elegir el protocolo de comunicaciones, siendo uno de los principales objetivos a la hora de diseñar la herramienta. Inicialmente, AA-RR ofrece compatibilidad con los protocolos SAML 1.1 y PAPI. Gracias a esta característica, podemos definir el comportamiento de nuestra emulación a través de un archivo de configuración neutro cuya estructura no depende del protocolo seleccionado.

Para facilitar el proceso de interacción, disponemos de un módulo que funciona como una base de datos preliminar, desde la cual podemos obtener información para nuestras peticiones y respuestas. Este módulo podemos configurarlo para que tome dicha información de un fichero XML, una base de datos o un repositorio de LDAP. De esta forma, podemos pedirle por los valores de algunos atributos cuando generemos una respuesta de atributo.

En nuestras emulaciones en el AA-RR, tenemos la posibilidad de elegir el protocolo de comunicaciones, siendo uno de los principales objetivos a la hora de diseñar la herramienta

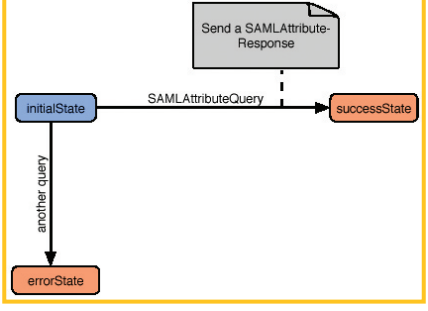
3.1.- Definiendo la interacción

Las interacciones de AA son un conjunto de peticiones de información y/o decisión de autorización así como sus respuestas. Cada petición contiene una lista de atributos, o un conjunto de pares atributo/valor para una decisión de autorización. Además, las IAAs podrían utilizar otro tipo de tecnologías para asegurar las interacciones, como certificados X.509 o claves compartidas.

Gestionaremos el comportamiento del AA-RR a través de un conjunto de reglas que especifican, de manera neutra y sin depender del protocolo seleccionado, qué peticiones o respuestas permitiremos recibir o enviar, qué atributos podremos solicitar o afirmar, qué condiciones de errores debemos detectar y qué resultado final esperaremos que se alcance.

Como vemos en la figura 2, representamos nuestra emulación en el AA-RR a través de un diagrama de estado, en el cual sólo puede existir un estado activo al tiempo. Definimos cada estado, identificado por su nombre, por un conjunto de reglas donde cada una tienen una serie de condiciones que deben cumplirse para realizar unas acciones determinadas. Estas reglas son procesadas secuencialmente, por lo que si una de ellas no cumple todas sus condiciones, saltaría la emulación a la siguiente. Pero en el caso de que se cumplan todas las condiciones, si una de las acciones a ejecutar es cambiar el estado activo a otro, no se procesarían más reglas de ese estado y saltaría al indicado.

FIGURA 2: DIAGRAGAMA DE ESTADOS DE LA EMULACIÓN



Las condiciones tienen como objetivo comprobar los eventos que se están produciendo en nuestra emulación

Las condiciones tienen como objetivo comprobar los eventos que se están produciendo en nuestra emulación, y podemos utilizar cualquiera de los siguientes:

- *Unidad de datos*. En esta condición podemos comprobar el tipo de datos que hemos recibido, identificado por un nombre específico que proporciona el adaptador de protocolo seleccionado, como por ejemplo, `SAMLAttributeQuery` en el adaptador de SAML.
- *Campo*. Podemos comprobar si se ha recibido un campo específico y su valor en la unidad de datos que recibimos, usando un parámetro que depende del adaptador de protocolo utilizado. En los adaptadores basados en mensajes XML, usaremos expresiones *XPath* para seleccionar campos y sus valores.

Las acciones definen el comportamiento de nuestra emulación, y podemos usar cualquiera de las siguientes:

- *Cambiar el estado activo*. Podemos cambiarlo a otro estado, identificando éste por su nombre.
- *Enviar datos*. Podemos enviar una unidad de datos del protocolo, el cual identificamos por un nombre determinado que nos proporciona el adaptador. Además, podemos asignar valores a sus campos.
- *Finalizar la emulación*. Podemos finalizar la emulación indicando si ha ido todo correcta, erróneamente o bien inconclusa.

FIGURA 3:
FICHERO XML QUE DEFINE EL DIAGRAGAMA DE ESTADOS

Una Autoridad de Atributo tiene como objetivo procesar todas las peticiones de atributo que se reciban

En el caso de que se haya recibido una petición SAML de atributo, con todos los parámetros indicados, se enviaría una respuesta SAML que contiene una aserción con una sentencia de atributo del sujeto

3.2.- Un ejemplo de uso: Emulando una autoridad de atributos

Una Autoridad de Atributo tiene como objetivo procesar todas las peticiones de atributo que se reciban y, basándose en una política de privacidad, enviar una respuesta que contenga las aserciones de atributo. Es un componente típico en una IAA y, por ejemplo, en Shibboleth está integrado dentro de la arquitectura del *Identity Provider*, en el cual se gestiona las credenciales y los atributos del sujeto. Vamos a emularlo en el AA-RR, simplificando su comportamiento ya que sólo queremos que responda a peticiones de atributo. Para este proceso necesitaremos definir un diagrama de estados y un conjunto de reglas que lo represente.

Antes de empezar nuestra emulación en el estado inicial, AA-RR crearía un fichero de log donde podremos leer toda la información sobre la ejecución. Como vemos en la figura 2, tenemos un estado inicial, llamado `initialState`, el cual espera recibir una petición SAML de atributo. Si no recibiéramos de este tipo o bien una petición que fuese errónea, la ejecución pasaría al estado `errorState` y finalizaría el archivo de log informando que la emulación no se ha realizado correctamente indicando la causa. En el caso de que se haya recibido una petición SAML de atributo, con todos los parámetros indicados, se enviaría una respuesta SAML que contiene una aserción con una sentencia de atributo del sujeto. Después de enviarlo, la emulación saltaría al estado successState y cerraría el fichero de log informando que todo ha ido como se esperaba. En el AA-RR, cuando se finaliza una emulación el proceso de ejecución vuelve a comenzar desde el estado inicial comenzando una nueva emulación de la *Autoridad de Atributo*.

Para definir esta emulación usaremos el conjunto de reglas que podemos encontrar en la figura 3. Como vemos, el estado inicial `initialState` se define por el elemento `<state`

name='initialState'>, el cual contiene tres reglas en donde comprobaremos, respectivamente, si es una petición SAML de atributo con una lista determinada de atributos, una petición SAML de atributo o cualquier otro tipo de petición. En el adaptador de protocolo de SAML, como dijimos antes, usaremos la tecnología *XPath* para referenciar a elementos en un mensaje SAML.

En la primera regla, <rule name='rule1-initialState'>, comprobaremos si hemos recibido una petición SAML de atributo que contenga un elemento AttributeDesignator, el cual se usa para especificar qué atributos se solicitan. A través del atributo variable del elemento <condition> tenemos una referencia del resultado que devuelve la expresión *XPath* usado en su atributo field. De esta forma, al hacer la respuesta SAML, como podemos ver en el elemento <action name='a1-rule1'>, podemos especificar el mismo sujeto y los mismos atributos que hemos recibido en la petición. De hecho, estamos indicando que queremos que la emulación busque los valores de esos atributos en la base de datos mydata. En caso de que todas las condiciones se satisfagan, la ejecución salta al estado *successState*.

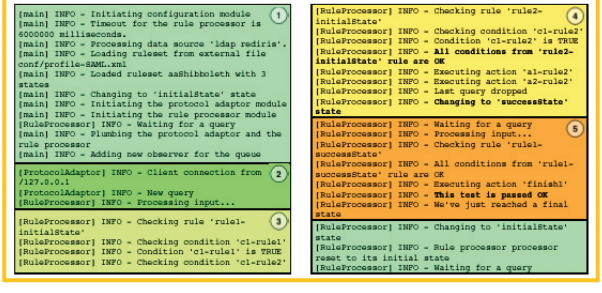
En el adaptador de protocolo de SAML, usaremos la tecnología XPath para referenciar a elementos en un mensaje SAML

En la segunda regla, <rule name='rule2-initialState'>, podemos comprobar que se ha recibido una petición SAML de atributo a través de la condición c1-rule2. En esta petición no hemos recibido ninguna lista determinada de atributos, así que una autoridad debe enviar todos aquellos que estén permitidos por una política de atributos. En nuestra emulación, hemos decidido enviar un único atributo, *urn:mace:dir:attribute-def:-eduPersonPrincipalName* con un espacio de nombres *urn:mace:shibboleth:-1.0:attributeNamespace:uri*. Como no necesitamos utilizar ningún elemento de la petición, usaremos el atributo value del elemento <action> para establecer el valor del atributo referenciado por la expresión *XPath*. En nuestro caso, enviaremos en este atributo el valor john.foo@bar.com. Cuando la emulación finaliza enviando la respuesta, elegiría como nuevo estado activo a successState.

En un primer momento AA-RR procesa todos los ficheros de configuración e inicializa el sistema para que se comporte como una Autoridad de Atributo de Shibboleth

En caso de que haya que procesar la siguiente regla, <rule name='rule3-initialState'>, significa que no hemos recibido una petición SAML de atributo, así que desplazaremos nuestra ejecución al estado errorState. Los dos estados finales, successState y errorState, tienen como objetivo finalizar la ejecución de la emulación y escribir en el fichero de log si ha ido todo correctamente o no.

FIGURA 4: INFORME DE LA EMULACIÓN



3.3.- Resultados de la emulación

En la figura 4 analizamos el fichero de log generado por nuestra emulación, el cual informa sobre los eventos y cambios de estado que han ocurrido. Como vemos, en un primer momento *(bloque 1)* AA-RR procesa todos los ficheros de configuración e inicializa el sistema para que se comporte como una Autoridad de Atributo de Shibboleth. En el *bloque 2* nos informa sobre la conexión desde una

dirección IP determinada el cual envía una petición a nuestro sistema. En el *bloque 3* nuestra herramienta informa de si la petición ha cumplido todas las condiciones de la primera regla del estado inicial. Como podemos ver, determina que ha recibido una petición SAML de atributo el cual no contiene un elemento `AttributeDesignator`, así que la ejecución pasa a la regla *rule 2*, como se describe en el *bloque 4*. Aquí vemos que la petición que recibimos cumple todas las condiciones de la *regla 2*, por lo que el sistema envía una respuesta y salta la ejecución al estado *successState*. En el *bloque 5* se informa sobre la ejecución de dicho estado, el cual finaliza la emulación e informa que todo ha ido correctamente.

3.- Conclusiones

En esta emulación, la herramienta ha sido capaz de comprobar si nuestro *software* interopera correctamente con una Autoridad de Atributo de Shibboleth. En el caso de que se hubiesen detectado errores, podríamos haber averiguado la causa. En el ejemplo ha ínteroperado correctamente con un coste mínimo para nosotros.

Como vemos, AA-RR representa una forma práctica y de confianza para comprobar si nuestro *software* interacciona de una manera correcta con diferentes componentes de una IAA. Los informes generados por las emulaciones nos permiten ver qué condiciones se satisfacen y cuáles no. Este aspecto es muy importante porque muchas IAAs comerciales funcionan como una *caja negra*, donde es muy difícil obtener más información sobre las razones de los errores que han ocurrido en la interacción. Además, esta herramienta nos permite invertir la mayor parte de nuestro tiempo en el desarrollo, puesto que no tenemos que desplegar una IAA en nuestra red (lo cual puede ser un proceso con un coste temporal elevado).

Los informes generados por las emulaciones nos permiten ver qué condiciones se satisfacen y cuáles no

De hecho, pensamos que AA-RR podría ser usado para facilitar el desarrollo de herramientas de testeo de IAA. A finales del 2005 se lanzó *HelloSAML*, un servicio de testeo que proporciona un interfaz web a AA-RR para comprobar infraestructuras basadas en SAML. A día de hoy, dicha herramienta está siendo utilizada por desarrolladores y compañías que quieran comenzar a interoperar con IAA basadas en SAML, puesto que facilita el proceso de aprendizaje.

AA-RR podría ser usado para facilitar el desarrollo de herramientas de testeo de IAA

Agradecimientos

Nuestro trabajo está respaldado por TF-EMC2 (TERENA Task Force on European Middleware Coordination and Collaboration) como una de sus líneas de actividades principales para los siguientes años. Además, queremos agradecer a la Fundación Española de Ciencia y Tecnología (FECYT) y el Centro Informático Científico de Andalucía (CICA).

Cándido Rodríguez
(contact@kan.es)
Universidad de Sevilla
Departamento de Tecnología Electrónica
Diego R. López
(diego.lopez@rediris.es)
José M. Macías
(macias@rediris.es)
RedIRIS

Bibliography

[1] L. A. Project. "Liberty Architecture Overview". July 2002 www.projectliberty.org

[2] M. Erds and S. Cantor. "Shibboleth Protocol Specification". Sept. 2005 shibboleth.internet2.edu/docs/internet2-mace-shibbolethарch-protocols-latest.pdf

[3] P. Hallam-Baker and E. M. eds. "Assertions and Protocol for the Oasis Security Assertion Markup Language (SAML)". *Oasis standard*. Nov. 2002 www.oasis-open.org/committees/security/docs/cs-sstc-core-01.pdf

[4] P. M. ed. "Bindings and Profiles for the Oasis Security Assertion Markup Language (SAML)". *Oasis standard*. Nov. 2002 www.oasisopen. org/committees/security/docs/cs-sstc-bindings-01.pdf

[5] R. C.-R. D. R. López. "Ubiquitous Internet Access Control: the PAPI System". *Database and Expert Systems Applications, 2002. Proceedings. 13th International Workshop on, pp. 441-445*. 2002.

[6] SURFnet. "The A-Select Authentication System" a-select.surfnet.nl

[7] W. S. S. W. C Rigney, A Rubens. "Remote Authentication Dial In User Service (RADIUS)". *Internet Request for Comments 2138*. Apr. 1997.