Artículo
Mathew Lambert · 7 mayo, 2020 Lectura de 4 min

Uso de OAuth2 con Servicios Web SOAP

¡Hola Comunidad!

Hace un par de días, un cliente me comunicó su intención de mejorar su aplicación legacy existente, que usa Servicios Web SOAP, por lo que comparte la misma autorización con su nueva API de aplicación basada en REST. Como su nueva aplicación usa OAuth2, el desafío estaba claro: cómo pasar un token de acceso con una solicitud SOAP al servidor.

Tras Googlear un poco, descubrí que una de las formas posibles de hacer esto era agregar un elemento de encabezado adicional al SOAP envelope y luego asegurarse de que la implementación del Webservice haga lo necesario para validar el token de acceso.

Por suerte, tenemos un mecanismo para poner encabezados a medida con una solicitud SOAP. Solo tuve que seguir la documentación (ver aquí los detalles) y obtuve las siguientes clases.

La clase de encabezado personalizada

```
Class API.SOAP.OAuth2Header Extends %SOAP.Header
{

Property accessToken As %String(MAXLEN = "");

}
```

Muy simple, realmente. Todo lo que necesito es pasar el token de acceso, pero también puede ampliarse para pasar otra información.

La clase de implementación del WebService

```
/// API.SOAP.MyService
Class API.SOAP.MyService Extends %SOAP.WebService [ ProcedureBlock ]
{

/// Name of the WebService.
Parameter SERVICENAME = "MyService";

/// TODO: change this to actual SOAP namespace.
/// SOAP Namespace for the WebService
Parameter NAMESPACE = "http://tempuri.org";

/// Namespaces of referenced classes will be used in the WSDL.
Parameter USECLASSNAMESPACES = 1;

/// TODO: add arguments and implementation.
/// GetVersion
```

```
Method GetAccountBalance(pAccNo As %String) As API.SOAP.DT.Account [ WebMethod ]

{
  #define APP       "ANG RESOURCES"
  try {
    #dim tAccessTokenHeader as API.SOAP.OAuth2Header=..HeadersIn.GetAt("OAuth2Header"
)

    $$$THROWONERROR(tSC,##class(%SYS.OAuth2.AccessToken).GetIntrospection($$$APP,tAcc
essTokenHeader.accessToken,.jsonObjectAT))

    /* service specific check */
    // check whether the request is asking for proper scope for this service
    if '(jsonObjectAT.scope["account") set reason="scope not supported" throw

    if '(##class(%SYS.OAuth2.Validation).ValidateJWT($$$APP,tAccessTokenHeader.access
Token,,,.jsonObjectJWT,.securityParameters,.tSC)) {
      set reason="unauthorized access attempt"
      throw
    }
    set tAccountObject=##class(API.SOAP.DT.Account).%New()
    set tAccountObject.accno=pAccNo
    set tAccountObject.owner=jsonObjectJWT."acc-owner"
    set tAccountObject.balance=$random(200000)
  } catch (e) {
    set fault=..MakeFault($$$FAULTServer,"SECURITY",reason)
    Do ..ReturnFault(fault)
  }
  Quit tAccountObject
}

XData AdditionalHeaders

{
<parameters xmlns="http://www.intersystems.com/configuration">
<request>
<header name="OAuth2Header" class="API.SOAP.OAuth2Header"/>
</request>
</parameters>
}
}
```

Vamos a comentar un poco la verificación del token de acceso. Como se puede ver, la primera tarea que hacemos es recuperar el token de acceso del encabezado personalizado y deserializarlo en una representación de un objeto.

Luego, podemos optar por verificar el alcance o hacer otras validaciones, como validar el token JWT (esto depende de cómo configuramos el servidor de autorización OAuth2 y si usamos OpenID, ¡lo que recomiendo muy especialmente!).

Veamos ahora el lado del cliente.

No explicaré en detalle cómo tu cliente recibe el token de acceso desde tu servidor de autorización OAuth2. Para eso podéis ver otros artículos. Simplemente vamos a analizar cómo proveer el token de acceso a un cliente del WebService. (Para generar la clase o clases de cliente WebService, debes ejecutar un Asistente SOAP estándar / Opción de cliente desde tu IDE Studio o Atelier).

Este es el fragmento de código de mi cliente

```
set tWSClient=##class(Web.WSC.MyServiceSoap).%New()
set tWSHeader=##class(Web.WSC.s0.OAuth2Header).%New()

set tWSHeader.accessToken=accessToken
do tWSClient.HeadersOut.SetAt(tWSHeader,"access-token")
#dim tAccountObject as Web.WSC.s0.Account=tWSClient.GetAccountBalance(tAccNo)
```

Consideraciones de configuración de seguridad en el servidor de recursos

La forma más fácil, pero también arriesgada, de configurar tu aplicación CSP en el servidor de recursos (WebServer) es hacer que permita usuarios no autenticados. Entonces, siempre será tu responsabilidad verificar el token de acceso en cada servicio y cada método, y validarlo. Si no existe un token de acceso o no es válido, DEBES devolver SOAP Fault.

La forma alternativa, pero mejor, es usar autenticación delegada y hacer que la rutina *ZAUTHENTICATE* recupere el token de acceso, asignando un nombre de usuario intencional (puede tomarse de JWT si el perfil de OpenID viene con un alcance dentro de la solicitud del token de acceso) con algún conjunto de reglas mínimas necesarias para ejecutar el método del webserver.

#Seguridad #SOAP #Caché #InterSystems IRIS

URL de fuente:https://es.community.intersystems.com/post/uso-de-oauth2-con-servicios-web-soap