記事
Toshihiko Minamoto · 2020年12月4日 5m read

SOAP（Web）サービスでの OAuth2 の使用

みなさん、こんにちは。

数日前、SOAP（Web）サービスを使用して、REST に基づく新しいアプリケーション API と同じ認証を使用できるように、既存のレガシーアプリケーションを拡張したい、とお客様から伺いました。新しいアプリケーションは OAuth2 を使用しているため、課題は明らかでした。SOAP リクエストを含むアクセストークンをどのようにしてサーバーに渡すか、ということです。

Google でしばらく調べてみたところ、SOAP エンベロープにヘッダー要素を追加してから、アクセストークンを検証するために必要なことを Web サービス実装が実行できるようにするのが 1 つの実現方法であることがわかりました。

幸い、私たちは、SOAP リクエストにカスタムヘッダーを提供するための仕組みを提供しています。そこで、ドキュメントを確認したところ（詳細はこちら）、次のクラスが出てきました。

カスタムヘッダークラス

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

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

}
```

確かに、非常に単純です。
必要なのはアクセストークンを渡すことだけですが、ほかの情報も渡すように拡張することもできます。

Webサービス実装クラス

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

/// Web????????
Parameter SERVICENAME = "MyService";

/// TODO: ?????? SOAP ????????????????
/// Web????? SOAP ???????
Parameter NAMESPACE = "http://tempuri.org";

/// ??????????????? WSDL ????????
Parameter USECLASSNAMESPACES = 1;
```

```
/// TODO: ??????????????
/// 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>
}
}
```

アクセストークンの確認について、少し説明しましょう。 ご覧のとおり、最初に行うタスクは、カスタムヘッダーからアクセストークンを取得して、オブジェクト表現に逆シリアル化することです。

その後は、スコープをチェックするかどうか、JWT
トークンの検証などさらなる検証を実行するのかを決めることができます（OAuth2
認証サーバーの設定方法や、私が非常にお勧めしているOpenID を使用するかどうかに依存しています）。

では、クライアント側を確認してみましょう。

クライアントが OAuth2 認証サーバーのアクセストークンをどのように受け取るのかについては他の記事で説明しているので、ここでは深く言及せずにおきますが、代わりに、アクセストークンを Web

サービスクライアントに提供する方法を確認することにしましょう。 （Web サービスクライアントクラスは、Atelier または Studio IDE から標準の SOAP ウィザード/クライアントオプションを実行して生成します。）

次は私のクライアントのコードスニペットです。

```
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)
```

リソースサーバーにおけるセキュリティ設定に関する考慮事項

リソースサーバー（Web サーバー）で CSP アプリケーションをセットアップするには、認証されていないユーザーを許可するのが最も簡単なやり方ですが、これにはリスクが伴います。サービスやメソッドごとにアクセストークンをチェックし検証するのはあなた次第です。 アクセストークンが存在しないか有効でない場合は、SOAP フォルトを返さなければなりません。

より優れた代替手段としては、委任認証を使用し、*ZAUTHENTICATE* ルーチンでアクセストークンを取得して、何らかの意図的なユーザー名（アクセストークンリクエストのスコープ付きで OpenID プロファイルが提供されている場合は JWT から取得可能）を、Web サーバーメソッドを実行するために最低限必要なロールで割り当てる方法があります。

#SOAP #セキュリティ #ベストプラクティス #Caché #InterSystems IRIS

ソースURL:
https://jp.community.intersystems.com/post/soap%EF%BC%88web%EF%BC%89%E3%82%B5%E3%83%BC%E3%83%93%E3%82%B9%E3%81%A7%E3%81%AE-oauth2-%E3%81%AE%E4%BD%BF%E7%94%A8