在Azure Pipeline中显示代码覆盖率

Azure API Management是微软云Azure上托管的API网关服务，具备完备的API管理功能。在安全方面，可以和 Azure Active Directory紧密集成，提供客户端OAuth 2.0的验证机制。官方文档<https://docs.microsoft.com/en-us/azure/api-management/api-management-howto-protect-backend-with-aad>有些简略，这里详细讲解验证机制和配置流程。

# 核心机制

在Azure AD中注册2个应用，一个对应API Management中创建的API，一个对应访问API的客户端应用。在Azure AD中为客户端应用授权可以访问后端API。在API Management中的API关联Azure AD中的后端应用。最后以客户端应用申请访问token，来访问后端API。

有关在API Management中创建API的操作本文不再赘述，如有需要，请参考官方文档 https://docs.microsoft.com/en-us/azure/api-management/add-api-manually。这里我们直接选用一个已经部署好的API Management中的API。

Graphical user interface, application

Description automatically generated

简化起见，我们把这个API设置成不要求订阅key，即在Settings中不勾选Subscription required。这样现在这个API就是没有任何限制，可以直接访问。就像：

curl https://contonso.azure-api.cn/contonso/name?name=test

{"message": "Hello, test! Welcome to Azure Function!", "timestamp": "2022-09-04, 14:35:34"}

下面我们会为这个API加上OAuth 2.0的token，把它保护起来。

# Azure AD部署

## 为后端API注册Azure AD应用

在Azure控制台进入Azure Active Directory, 选择 App Registrations。在右侧点击New registration。输入一个有意义的名字，比如这里我们命名为 name-backend。Redirect URI选择Web，其它保持默认，点击Register按钮。

Graphical user interface, text, application, email

Description automatically generated

创建成功后在Overview页，记录下这个应用的Application (client) ID。如这里是：

Display name: name-backend

Application (client) ID: 95ac855d-724d-4bf7-95b1-93244f4e2e5b

左侧导航链接在Manage部分点击Expose an API，在Application ID URI右侧点击Set链接。

Graphical user interface, text, application, email

Description automatically generated

使用默认值，点击 Save按钮保存。

Graphical user interface, application

Description automatically generated

把这个Application ID URI 也记录下来。

在Manage部分点击App roles 然后点击 Create app role。

Display name，输入一个有意义的名字，如Reader.

Allowed member types，选择Applications

Value输入有意义的值 ，如Read

Description 输入有意义的值 ，如Readers have rights to read.

Do you want to enable this app role? 勾选。

Graphical user interface, text, application, email

Description automatically generated

最后点击Apply保存。

设置返回token的版本

左侧导航菜单Manage下点击Manifest，右侧主窗格中找到 accessTokenAcceptedVersion,把它的值设为2。

Graphical user interface, application

Description automatically generated

这步很重要，因为API Management中的JWT仅支持 2.0 版的格式。如果不修改此值，调用Azure AD接口获取的JWT token中解析出来的token 发布者URL格式不对，后面验证就不能通过。

## 为客户端注册Azure AD应用

在Azure控制台进入Azure Active Directory, 选择 App Registrations。在右侧点击New registration。输入一个有意义的名字，比如这里我们命名为 name-client。Redirect URI选择Web，其它保持默认，点击Register按钮。

创建成功后在Overview页，记录下这个应用的Application (client) ID值。如这里的

Display name: name-client

Application (client) ID: 6bfe51f7-ea34-4096-aa7c-7e8df1043bbd

为这个客户端应用创建密钥。左侧导航菜单Manage下点击 Certificates & secrets。在New client secret下点击New client secret。在弹出层中Description输入name-client，再点击下面的Add按钮添加。

添加完成时，首次显示密钥值时一定要及时复制保存出来，刷新页面以后就不再显示了。如果没有及时保存下来，只能把它删除，再创建一个新的。

Graphical user interface, text, application, email

Description automatically generated

## 在Azure AD中为应用赋权

在Azure AD的App registrations中选择刚刚创建的客户端的应用，左侧导航Manage段下点击API permissions。右侧窗格点击App a permission。选中前面我们创建的后端应用name-backend。

Graphical user interface, text, application, email

Description automatically generated

再选择Application permissions，以及下面选择我们已经创建好的角色。点击Add permissions。

Graphical user interface, text, application, email

Description automatically generated

这时会回到Configured permissions列表，Status列都是空白了。

Graphical user interface, text, application, email

Description automatically generated

点击Grant admin consent for <your-tenant-name>，弹层点Yes。然后Status列显示为绿色对勾即可。

Graphical user interface, text, application, email

Description automatically generated

# 配置API Management中的JWT策略

先到 Azure AD控制台的Overview页，记录下当前的Tenant ID。

Graphical user interface, application

Description automatically generated

准备Open ID URL。使用Azure全球区域时，此URL规则是

https://login.microsoftonline.com/{Tenant-ID}/v2.0/.well-known/openid-configuration

使用Azure中国区域时，此URL规则是

https://login.partner.microsoftonline.cn/{Tenant-ID}/v2.0/.well-known/openid-configuration

在API Management控制台，打开我们已经准备好的API。在Design界面，Inbound policy点击Add policy。

Graphical user interface, application

Description automatically generated

再向下翻页，点选Validate JWT。

Header name输入Authorization。

Failed validation error message输入Unauthorized. Access token is missing or invalid.

点击Add claim，name输入aud。再点击Add value，Values输入前述记录下的Azure AD中注册的后端应用的client ID.

点击 Add Open ID URL，输入前述准备好的Open ID URL。点击最下面的Save按钮保存。

Graphical user interface, text, application, email

Description automatically generated

到此为API启用OAuth 2.0 验证配置完成。现在我们直接访问这个API，返回的是验证失败的错误了。

curl https://snowpeak.azure-api.cn/contonso/name

{ "statusCode": 401, "message": "Unauthorized. Access token is missing or invalid." }

# 验证测试

## 使用Azure AD中的客户端应用获取访问token

我们使用OAuth 2.0的标准流程，组织请求。使用前述Azure AD 的Tenant ID 组织目标地址。使用Azure全球区域时，目标地址是

https://login.microsoftonline.com/{Tenant-ID}/oauth2/v2.0/token

使用Azure 中国区域时，目标地址是

https://login.partner.microsoftonline.cn/{Tenant-ID}/oauth2/v2.0/token

参数说明如下：

|  |  |
| --- | --- |
| 参数名 | 参数值 |
| grant\_type | client\_credentials |
| client\_id | 前述客户端应用的Application (client) ID，如6bfe51f7-ea34-4096-aa7c-7e8df1043bbd |
| client\_secret | 前述客户端应用的密钥，如Lz36Zrilxa~01234567890#ABCDEFGH |
| Scope | 以后端应用ID按 {backend-application-Application-ID}/.default 格式组织  如95ac855d-724d-4bf7-95b1-93244f4e2e5b/.default |

组织好请求地址和参数后，我们以curl发送请求，并得到响应，注意再加一个Content-Type: application/x-www-form-urlencoded的请求头。

curl -X POST -H "Content-Type: application/x-www-form-urlencoded" -d 'client\_id=6bfe51f7-ea34-4096-aa7c-7e8df1043bbd&client\_secret=Lz36Zrilxa~pB3d6LAewX~33~6rWREFLC.&scope=95ac855d-724d-4bf7-95b1-93244f4e2e5b/.default&grant\_type=client\_credentials' 'https://login.partner.microsoftonline.cn/78c60b1a-3a93-4e0e-99ad-00dc772cfb6f/oauth2/v2.0/token'

{"token\_type":"Bearer","expires\_in":3599,"ext\_expires\_in":3599,"access\_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6Il9DSkFPdHlzWVZtNXhjMVlvSzBvUTdxeUJDUSJ9..2PsH5TO79q7Ix7FrkGP\_klLPjpkSC14mPhheBLWTig8d9UX565replym7baKbVX7VuXorwkVHdqLQr\_1AxzgylwMA7FWWTKeBEYWdfM7vcf58Ub46Q4Y2o0HD1jAqlRzyX2iLOvFsA5o0Hi4\_fYrJ6Nt8XW1HXdg2-PS8hdFTmudW\_NHVRD2B7FbmoAUwBV56MdMxmMuDY\_L6isBHCcaeqH9YmzTAosNg-ValNd67h8BLtStIlZOC3\_q6pzlhJB\_ox4wLsYel8tXPUzElli-S72Oo9WLVTB9UOhOTQfuknkpD2rPrGB\_IcQPkFUwo8HsaZ2azSzII5MJdfgUqzUMuQ"}

浏览器打开<https://jwt.ms/>，我们把返回的access\_token解析看一下。

Graphical user interface, text, application

Description automatically generated

可以看到解析出详细的token信息，其中 iss 就是token发布者的URL。这就是前面为后端应用设置accessTokenAcceptedVersion为2的效果。

最后我们添加请求头，再访问已经加上验证的API。

curl -H'Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIlvSzBvUTdxeUJDUSJ9..2PsH5TO79q7Ix7FrkGP\_klLPjpkSC14mPhheBLWTig8d9UX565replym7baKbVX7VuXorwkVHdqLQr\_1AxzgylwMA7FWWTKeBEYWdfM7vcf58Ub46Q4Y2o0HD1jAqlRzyX2iLOvFsA5o0Hi4\_fYrJ6Nt8XW1HXdg2-PS8hdFTmudW\_NHVRD2B7FbmoAUwBV56MdMxmMuDY\_L6isBHCcaeqH9YmzTAosNg-ValNd67h8BLtStIlZOC3\_q6pzlhJB\_ox4wLsYel8tXPUzElli-S72Oo9WLVTB9UOhOTQfuknkpD2rPrGB\_IcQPkFUwo8HsaZ2azSzII5MJdfgUqzUMuQ' 'https://snowpeak.azure-api.cn/contonso/name?name=test'

{"message": "Hello, test! Welcome to Azure Function!", "timestamp": "2022-09-05, 07:50:57"}

可以看到通过通过验证，并返回了正常的结果。