WorkshopPLUS – Modern Authentication and Authorization – Securing API with API Management

Student Lab Manual

V1.0, January 2017

Information in this document is subject to change without notice. The example companies, organizations, products, people, and events depicted herein are fictitious. No association with any real company, organization, product, person or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarked, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2017 Microsoft Corporation. All rights reserved.

Microsoft, MS-DOS, MS, Windows, Windows NT, MSDN, Active Directory, BizTalk, SQL Server, SharePoint, Outlook, PowerPoint, FrontPage, Visual Basic, Visual C++, Visual J++, Visual InterDev, Visual SourceSafe, Visual C#, Visual J#,  and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the U.S.A. and/or other countries.

Other product and company names herein may be the trademarks of their respective owners.

Contents

[Objectives 4](#_Toc53673804)

[Prerequisites 5](#_Toc53673805)

[Task 1 – Create an Azure API Management service/resource 5](#_Toc53673806)

[Task 3 – Add an existing REST API endpoint to your APIM instance 11](#_Toc53673807)

[Task 4 – Require valid JWT token to call the API 18](#_Toc53673808)

[Task 5 – Configure Azure AD to issue tokens to your API 21](#_Toc53673809)

[Task 6 – Configure APIM to use Azure AD for token issuance 22](#_Toc53673810)

[Task 7 – Update the API to use OAuth 2.0 Provider 27](#_Toc53673811)

[Task 8 – Test the API using the Developer Portal 28](#_Toc53673812)

[Task 9 – Remove/cleanup the APIM instance. 30](#_Toc53673813)

## Objectives

After completing the exercises in this lab, you will be able to:

* Create an Azure API Management service
* Configure an API to validate a JWT token.
* Configure an API to use Basic Authn to login to a back-end API

## Prerequisites

* Access to an Azure subscription

## Task 1 – Create an Azure API Management service/resource

Use instructions in <https://docs.microsoft.com/en-us/azure/api-management/get-started-create-service-instance> to create a new Azure APIM service instance.

Select Create a resource:

Diagram

Description automatically generated

Type API Management:

Graphical user interface, application

Description automatically generated

Create:

Graphical user interface, application, website

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Add a new product:**

On the APIM blade in the portal select Products:

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Select Products:

Graphical user interface, text, application

Description automatically generated

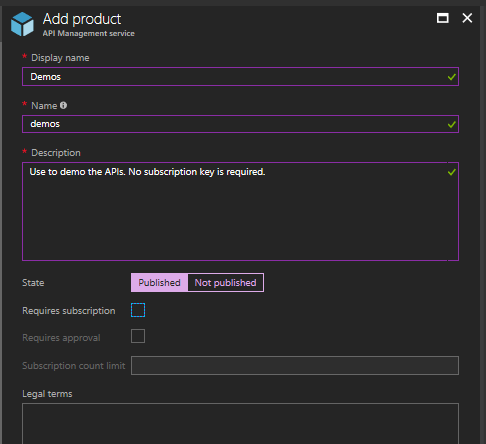


Complete the new product definition as follows. We will use it to test our API definitions without a subscription key – hence make sure *Requires subscription* is unchecked. Also, make sure that *Published* option is selected.

A screenshot of a cell phone

Description automatically generated





## Task 3 – Add an existing REST API endpoint to your APIM instance

1. Open the APIs blade from the **Azure portal APIM blade**:

Graphical user interface, application

Description automatically generated



1. Select *Blank API* to add a new API manually (no meta exists for the API).

A picture containing graphical user interface

Description automatically generated

1. Complete the *Create a blank API* form as follows and press the *Create* button “ON the top Full”.



Graphical user interface, text, application, email

Description automatically generated



Graphical user interface, text, application, email

Description automatically generated



1. Add one of the operations exposed by httpbin to API you have just created: select Add operator to the operation (browse to <http://httpbin.org> to see all operations exposed by this site):

Graphical user interface, text, application

Description automatically generated



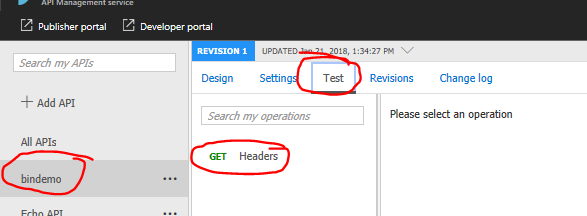
1. Complete the form as follows (the URL suffix must be as entered, you are free to enter a different Display name or Description) and click ***Save***:

Graphical user interface, text, application, email

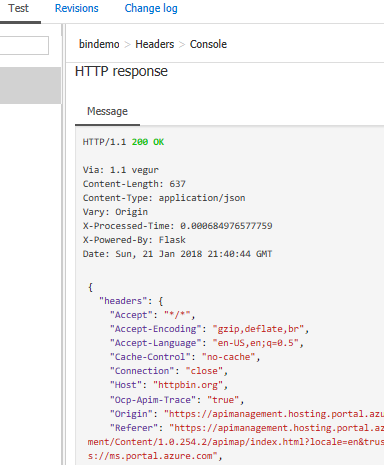
Description automatically generated



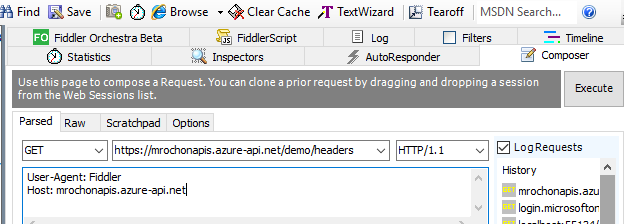
1. Test the API by the Test menu on for the API and the Get headers operation.



1. You should see the following return from the API when you press the Send button. The return contains all the http headers send in the request.



1. You can also use Fiddler to send the request to your APIM endpoint. Use the Fiddler composer with the following GET operation, after replacing the APIM service name (here *mrochonapis*) with the name of your service:



## Task 4 – Require valid JWT token to call the API

1. In the portal, select your API and All operations option. Then click on the ellipsis in the top right corner of the Frontend box and select Code editor.

Graphical user interface, application

Description automatically generated



1. In the xml window, add a new line before the <base /> element of the <inbound> policies. Then select + Validate JWT from the menu on the right:

Graphical user interface, text, application, email

Description automatically generated



Before



Graphical user interface, text, application

Description automatically generated

After



1. Replace the detailed definition of expected token issuer with the reference to its metadata as follows. Replace modernauth.onmicrosoft.com with the name of your own Azure AD tenant unless you plan to test the API using the predefined OAuth2 settings in the modernauthn tenant.

<validate-jwt header-name="Authorization"

failed-validation-httpcode="401"

failed-validation-error-message="Unauthorized">

<openid-config

url="https://login.microsoftonline.com/modernauthn.onmicrosoft.com/.well-known/openid-configuration" />

<audiences>

<audience>https://modernauthn.onmicrosoft.com/WebAPI</audience>

</audiences>

</validate-jwt>

<base />



Graphical user interface, text, application, email

Description automatically generated

## Task 5 – Configure Azure AD to issue tokens to your API

You can skip this step if you do not want to use your own Azure AD tenant. An application has already been registered in the modernaouthn.onmicrosoft.com tenant and its details will be provided to you in the next task, if you decide not to use your own Azure AD tenant.

Configure your Azure AD with two applications, both Web App/API type: one to represent the API fronted by Azure API Management (call it TestAPI) and another representing a client application calling the TestAPI (TestClient).

Modify the TestAPI manifest to include the following. This will allow clients to call it with Client Credential tokens:

"appRoles": [

{

"allowedMemberTypes": [

"Application"



],

"displayName": "General",

"id": "1b4f816e-5eaf-48b9-8613-7923830595ad",

"isEnabled": true,

"description": "Application access",

"value": "general"

}

],

App Id URI for TestAPI needs to be *https://modernauthn.onmicrosoft.com/WebAPI* or you will need to adjust the following to use whatever App ID you used.

TestClient must be configured to have permission to call TestAPI as an application and must have a symmetric key defined for it. Make sure to keep its App Id GUID and secret saved somewhere – you will need them in the following step.

## Task 6 – Configure APIM to use Azure AD for token issuance



In this task we will configure our API to use Azure AD to issue tokens to when testing the API from the developer console.

1. In <https://portal.azure.com> find your API Management blade.
2. Select the OAuth2 option on the left:

Graphical user interface, application

Description automatically generated



1. Click Add on the menu and configure AAD as provider with the following properties (leave other unchanged):

|  |  |
| --- | --- |
| **Item** | **Value** |
| Name | Any name, e.g. Modern Auth Directory or AAD Provider |
| Client registration page (not used but required) | *https://localhost* |
| Authorization code | Uncheck |
| Client credentials | Check |
| Authorization endpoint URL (not used) | *https://login.microsoftonline.com/modernauthn.onmicrosoft.com/*  *oauth2/authorize*  Replace modernauthn with your directory name. |
| Token endpoint URL | *https://login.microsoftonline.com/modernauthn.onmicrosoft.com/*  *oauth2/token*  Replace modernauthn with your directory name. |
| Additional body parameters… | Name: *Resource*  Value:[https://modernauthn.onmicrosoft.com/WebAPI](https://nam06.safelinks.protection.outlook.com/?url=https%3A%2F%2Fmodernauthn.onmicrosoft.com%2FWebAPI&data=02%7C01%7CNicholas.Switzer%40microsoft.com%7C83a4355c369d46c5dfa708d6d990633a%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C636935611491089339&sdata=n1RX3Hf3XJKqyU%2FzGuw2W%2B2LtcUIJdLp%2BuWxJNY%2BYBY%3D&reserved=0) |
| Client id | 1c635c96-739f-4a45-8020-0ee6ba133681 (or your own if you are using own Azure AD in Task 5) |
| Client secret | /VEB4SadZvIOJ/ETtFGQR1EN4bKOb2FMDmljAl7c0cs=  (if you are using your own application, then you have to use the client secret defined for your application) |

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

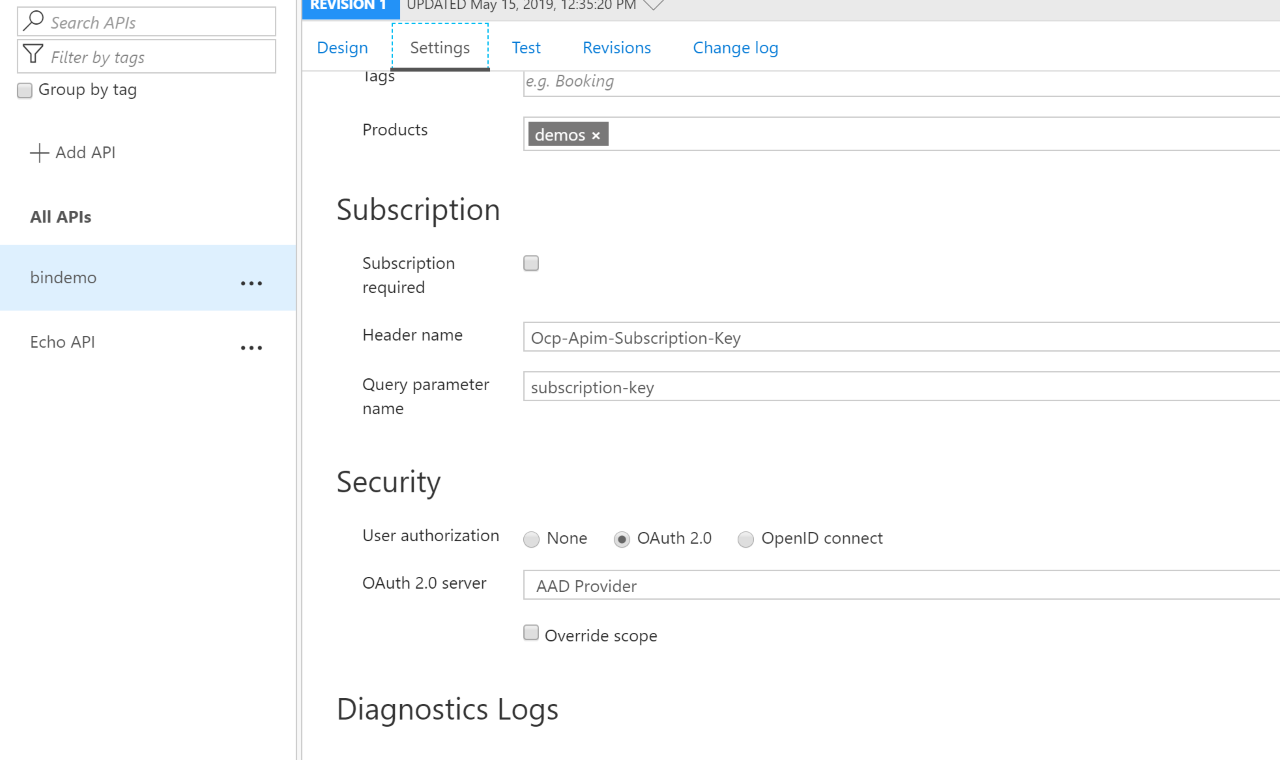
Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Task 7 – Update the API to use OAuth 2.0 Provider

1. Navigate back to your API
2. Click the *Settings* tab at the center of the screen
3. Scroll down to the *Subscription* section
   1. Ensure the *Subscription required* checkbox is **NOT** checked
4. Scroll down to the *Security* section
   1. Change *User authorization* to OAuth 2.0
   2. Select the *OAuth 2.0 Server* created from the previous task



1. Click Save

## Task 8 – Test the API using the Developer Portal

1. In portal.azure.com go to your API Management Intance blade. Select ***Developer portal*** menu option.

Graphical user interface, application

Description automatically generated

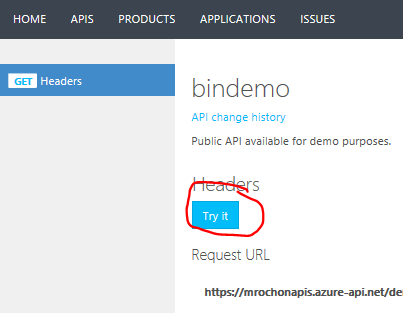
1. A new window will open **“Developer portal (legacy)”.** You will be logged in Administrator. Click the *APIs* menu option.

Graphical user interface, text, application

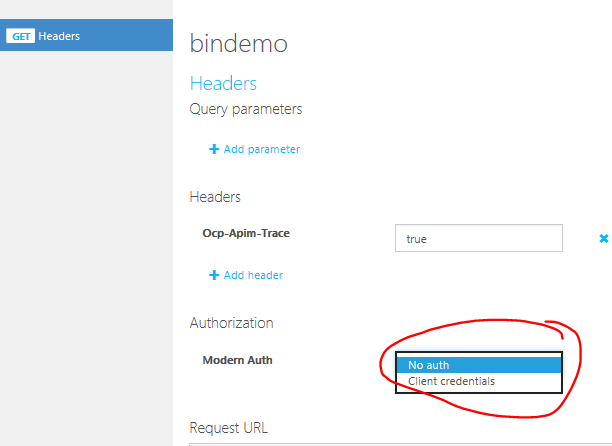
Description automatically generated



1. Select the *binddemo* API.
2. Click on the *Try it* button



1. Press the Send button at the bottom of the screen. You should get an error message (401) since your call did not include a token and our policy requires it.
2. Now select the Client Credentials option in the Authorization tab:





1. APIM will pop up a message telling you that a new token will be requested, press Authorize. APIM now has a new token with which to call the API. You can see the token in the Authorization property (and thus use it for debugging if needed).
2. Press send again. You should now get a valid (200 return code) message, which, in our case just replicates all the headers that were sent to the API. One of them is the JWT token itself. You have successfully validated that the JWT Policy we have configured in an earlier step is working!

## Task 9 – Remove/cleanup the APIM instance.

Follow instructions in <https://docs.microsoft.com/en-us/azure/api-management/get-started-create-service-instance#clean-up-resources> to remove the APIM service when no longer needed.



Graphical user interface, text, application, chat or text message

Description automatically generated

