Windows security, Active Directory and Azure AD

TD17 – Module 4 – Section 3

June 2020

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place 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, trademarks, 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.

The names of manufacturers, products, or URLs are provided for informational purposes only, and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third-party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

This training uses various tools and utilities downloaded from the Internet for the classroom environment.   
Downloading any tools, installing and using them should only be done at your own risk security checked the tools in a test environment.

© 2020 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at <https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx> are trademarks of the Microsoft group of companies. All other trademarks are the property of their respective owners.

Contents

[Hybrid Identity lab using Azure AD 1](#_Toc44074280)

[Abstract and learning objectives 1](#_Toc44074281)

[Overview 2](#_Toc44074282)

[Requirements 2](#_Toc44074283)

[Exercise 1: Azure AD consent and Permissions 3](#_Toc44074284)

[Task 1: Register the application in Azure AD 3](#_Toc44074285)

[Task 2: Configure the client application 5](#_Toc44074286)

[Task 3: Request OAuth tokens and access an API 5](#_Toc44074287)

[Task 4: Granting permissions to applications 8](#_Toc44074288)

# Hybrid Identity lab using Azure AD

## Abstract and learning objectives

This lab walks the reader through the step-by-step process of building a hybrid identity lab (Hybrid AD) using Azure Active Directory and Windows Server Active Directory. It builds upon knowledge gained from other classes for Active Directory and Azure Active Directory, and once completed, will allow the reader to be familiar with fundamental Azure Active Directory concepts and operations.

## Overview

In this Lab, attendees will create a new Azure AD tenant, enable synchronization between this tenant and an existing Active Directory domain and perform common Azure AD administration tasks.

## Requirements

1. Attendee’s machine:
   1. Ideal resolution 1920 x 1080
   2. An Internet browser
   3. An RDP client
   4. Internet access without restriction on outbound connections.   
      The following outbound TCP port must be accessible :

* **TCP/80 and TCP/443** to reach Azure Portal
* **TCP/3389** to establish RDP remote connection to virtual machines exposed directly to Internet

or

* **TCP/(49152 to 65535)** to establish RDP remote connection to virtual machines exposed by a Load Balancer

## Exercise 1: Azure AD consent and Permissions

Duration: 2 hours

Synopsis: In this exercise, you will register a new application in your subscription and use Azure AD as an identity provider for it.

For the purpose of the lab, you will use Google’s OAuth 2.0 playground. This application allows to test federated OAuth 2.0 authentication and displays the content of protocol exchanges between the different peers.

To simulate the application accessing an API to get data, you will finally query the Microsoft Graph API to retrieve authenticated user’s profile. More information about the Microsoft Graph API can be found at: <https://developer.microsoft.com/en-us/graph>

In order to see and analyze OAuth 2 token, you will also use the <https://jwt.io/> web application which allows parsing and interpretation of JWT token.

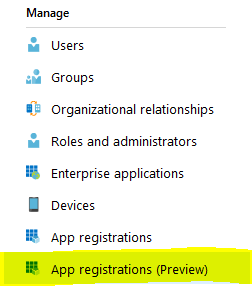
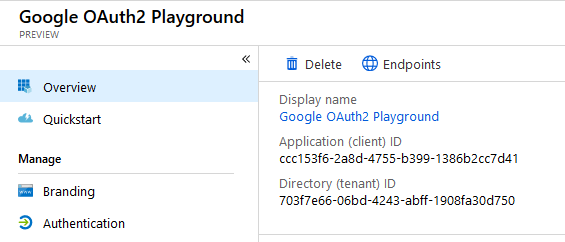
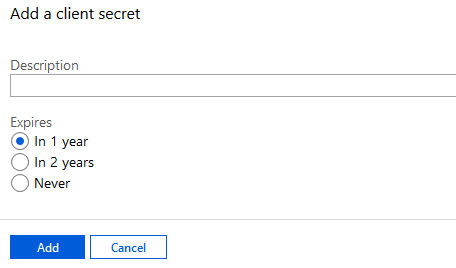
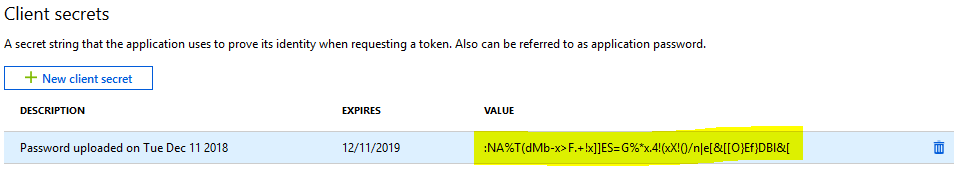
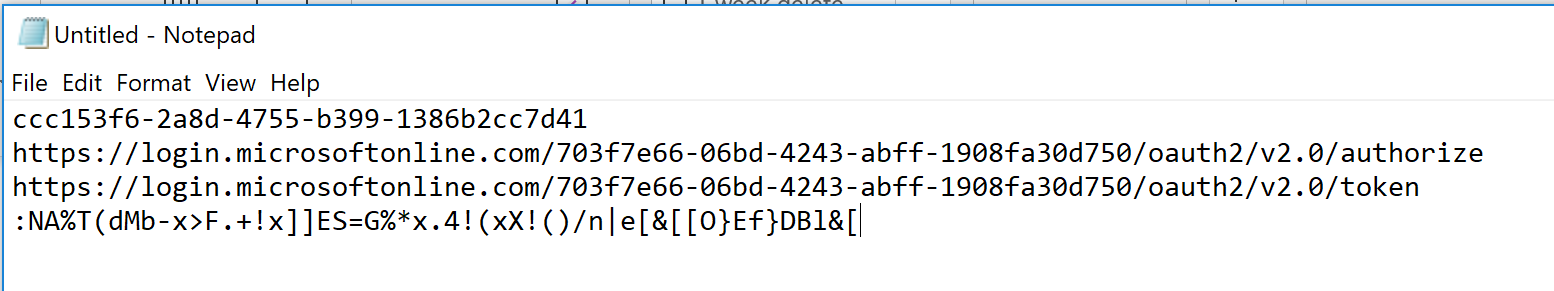
It is highly recommended you refresh you knowledge about OAuth2 by reading the following sections:

* OAuth2 roles : <https://tools.ietf.org/html/rfc6749#section-1.1>
* OAuth2 protocol flow : <https://tools.ietf.org/html/rfc6749#section-1.2>

#### Task 1: Register the application in Azure AD

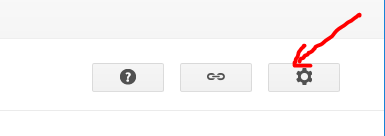
In this task, you will register the application in Azure AD so that it can authenticate users and access their data. This task is purely configuration. There is no question to answer.

**Important:** Unless you are currently signed in with your Office 365 trial subscription, ensure you sign out from any other Microsoft service you are currently signed in and clear your browser’s cookies. Otherwise, the portals will try to use your organizational credential which will fail.

1. Sign in to Azure AD by browsing <https://aad.portal.azure.com> and by using the credential you created in TD16, exercise 1.
2. For the purpose of the lab, you will be using the OAuth2 protocol and the Azure AD v2.0 endpoints to federate your application. At the time of writing, you need to use the preview version of the App Registration utility.  
   In the Azure AD menu bar, select **App Registrations (Preview):**
3. Select **New registration**
4. Fill in the form as instructed:
   1. Name: Google OAuth2 Playground
   2. Supported accounts types: Accounts in this organizational directory only
   3. Redirect URI: <https://developers.google.com/oauthplayground>
5. Click **Register** to validate application registration
6. If the registration was successful, you are now presented with the application’s Overview pane as in:  
   
7. Create a new text document and save the following data for later use:
   1. From the Overview tab, copy the **Application (client) ID** GUID to your text document
   2. Click the **Endpoints** button and copy the following data to your text document:
      1. OAuth 2.0 authorization endpoint (v2) URL
      2. OAuth 2.0 token endpoint (v2) URL  
         **Ensure you take the *(v2)* URLs**
8. Now, you need to create a secret key for the application to authenticate to Azure AD. In the application’s menu bar, select **Certificates & secrets**
9. In the **Client secrets** section, select **New client secret**
10. The **Description** field can be left empty. Select **1 year** as the expiration period and click **Add**.  
    
11. Now, **copy the secret value and save it to your text file**. This is important for the next steps and you won’t be able to retrieve this secret from the Azure portal after your leave the blade.  
    
12. Your text file should be similar to:  
    

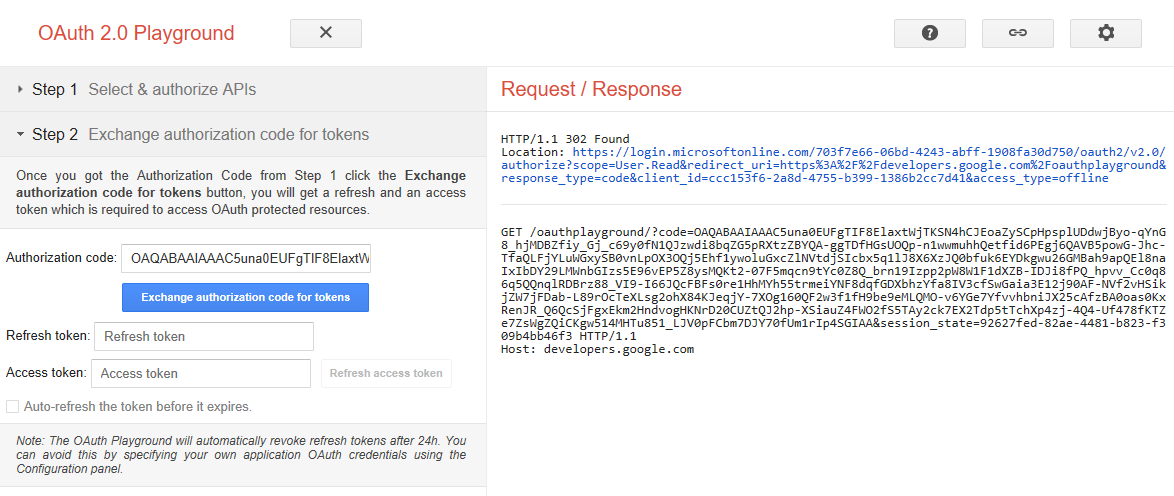
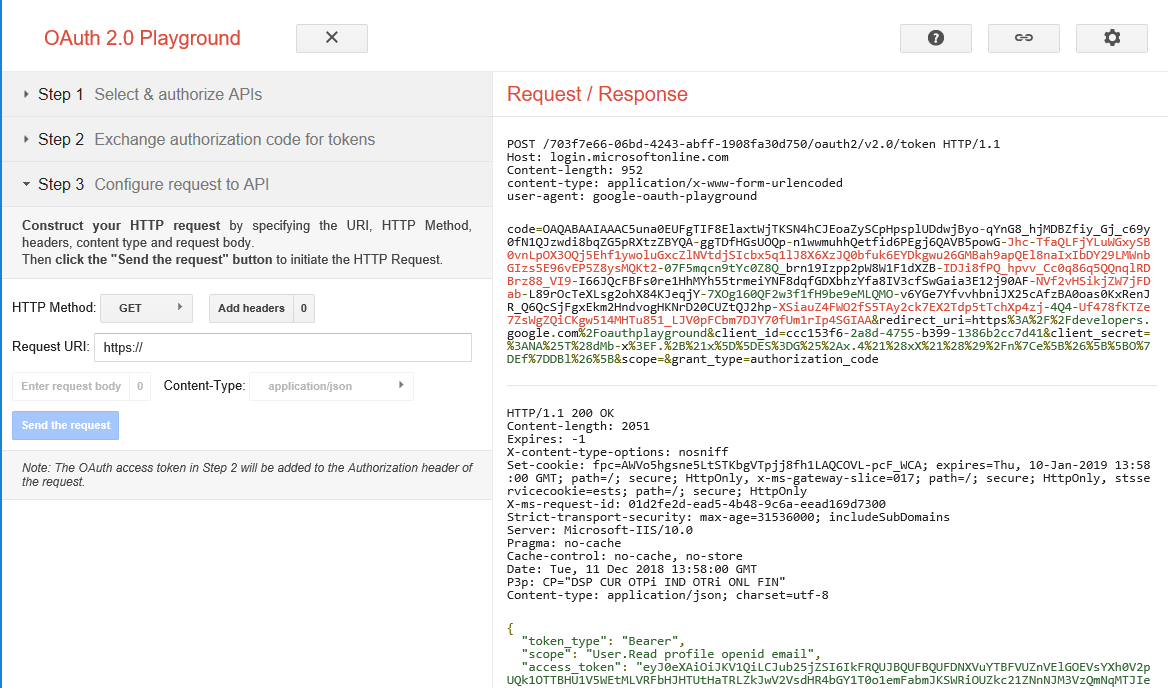
#### Task 2: Configure the client application

In this task, you will configure the Google’s OAuth2 Playground application to use Azure AD as the authentication source. This task is purely configuration. There is no question to answer.

1. Go to <https://developers.google.com/oauthplayground>
2. Click on the gear button in the top right corner  
   
3. Setup the **OAuth 2.0 configuration** panel as instructed:
   1. OAuth flow: **Server-side**
   2. OAuth endpoints: **Custom**
   3. Authorization endpoint: *take value from your text file. URL should end with /authorize*
   4. Token endpoint: *take value from your text file. URL should end with /token*
   5. Access token location: **Authorization header w/ Bearer prefix**
   6. OAuth Client ID: *take value from your text file*
   7. OAuth Client secret: *take value from your text file*
4. Click on **Close** to finish the configuration

#### Task 3: Request OAuth tokens and access an API

In this task, you will run the complete OAuth2 authorization flow from authentication to accessing an API. For demonstration purpose, you will use the Microsoft Graph API to retrieve details about the authenticated user.

1. In the OAuth2 playground application, locate the **Step 1** section of the left outer panel and type **User.Read** in the **Input your own scopes** field and click **Authorize APIs**This action will query the authorize endpoint URL and request an OAuth2 authorization code. You will be redirected to the Azure AD portal for authentication.
2. Go through all the Azure AD authentication screens and accept the consent prompt.
3. If authentication was successful, you should be redirected back to the Playground application with a view like:  
     
   Note: the right pane (Request / Response) displays the content of the respective HTTP request/response used in the process. The section above the line,
4. In order to access some APIs, the OAuth2 client needs to request an access token (Authorization Code cannot be used directly).  
   In the **Step 2** section of the left outer panel, click the **Exchange authorization code for tokens** button. Do not change the value of the other fields.  
   This action will query the token endpoint URL and request an OAuth2 access token using the Authentication Code obtained during the previous step.
5. If the access token was delivered you should now have a screen like:  
     
   Notes:  
   In the right part of the screen, the HTTP request is above the line and the response is below the line. The content of these messages is specific to the OAuth2 protocol. But, as OAuth2 and HTTP are tightly coupled, you can observe the HTTP response code to determine if the attempt was successful. 200 means the access token was successfully issued (you can see the token’s content in the JSON document provided in the answer) anything in the range 400-599 means an error. 4xx errors are due to a problem on the client (eg. Invalid request or access is denied) and 5xx errors are due to a problem on the server (eg. When the service is unavailable).
6. Use the <https://jwt.io/> to examine the content of the access\_token. JWT reference can be found at <https://tools.ietf.org/html/rfc7519>
7. Now is the time to query the API.  
   In the **Step 3** section of the left outer panel, sets the values as instructed then, click **Send the request**:
   1. HTTP Method: GET
   2. Request URI: <https://graph.microsoft.com/v1.0/me>

Note : the OAuth2 playground application will automatically add the Authenticate header containing the access token.

1. If the request was successful, you should get a JSON document containing most of the user’s personal attributes as they appear in Azure AD.

Additional Questions:

1. What is the difference between the Authentication Code and the Access Token?
2. Examining the content of the access token, what is the value of the Issuer and Subject claims?
3. What is the lifetime of the access token?
4. Copy-Paste here the result of the HTTP query to the Microsoft Graph API.

#### Task 4: Granting permissions to applications

In the previous task, you were able to query your own details because ability to read signed in user’s data had been granted by default when you registered the application.

In this task, you will try to list all users and read their profile in the Azure AD subscription from the OAuth2 playground application and set the permissions accordingly.

1. Using the instructions from <https://docs.microsoft.com/en-us/azure/active-directory/develop/quickstart-configure-app-access-web-apis#add-permissions-to-access-web-apis>, try to add the permission to list all users and read their basic profile.  
   Hint: for simplicity, you will use the **Read all users' basic profiles** permission which does not require an Admin consent
2. Using the same steps from previous task, try to list all users in the directory.  
   Note: the query to use for listing all users is:  
   - HTTP Method: GET  
   - Request URI : <https://graph.microsoft.com/v1.0/users>

Additional Questions:

1. What is the scope you must use to read all users’ basic profiles?
2. Is it possible to reuse the same Authentication Code from the previous task? Why?
3. Copy-paste the result of the HTTP query