**OAUTH FOR BUSINESS CENTRAL ON-PREMISES SERVICE TO SERVICE AUTHENTICATION**

## Introduction

There are already many great posts online and even Microsoft presentations that explain how to Authenticate to Business Central using OAuth 2.0. But most of them are focused on the cloud version.

As I’ve been asked to blog about **how OAuth 2.0 Business Central Authentication can be used for the On-Premise** version and I also worked on a related task recently, I thought it would be interesting to write about it.

The example is about Service to Service Authentication into Business Central and not OAuth authentication with user interaction. I think that many integrations will use this feature. Because when systems(or Business Central companies inside the same system) communicate with each other we usually don’t want user interaction.

At the moment I believe the most common way to access Business Central web services is with [Basic Authentication](https://docs.microsoft.com/en-us/dynamics-nav/web-services-authentication) (UserName and Password or Web Service Access Key).

## Why OAuth for Business Central?

Microsoft announced the **retirement of Basic Authentication in 2022 for the cloud** version of Business Central. At the moment it is not planned to retire Basic Authentication for On-Premise environments. However, for security reasons or to prepare for cloud and make sure that existing integrations can handle OAuth, there might be the need to use it for On-Prem environments as well.

## Steps to use OAuth in Business Central On-Premise

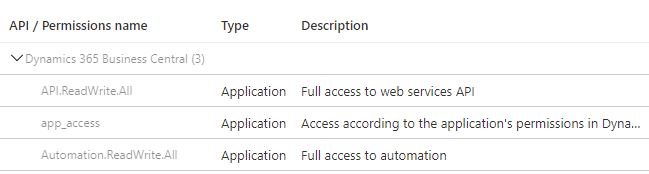
OAuth for Business Central On-Premises can be used only if Single Single-On (SSO) is enabled.

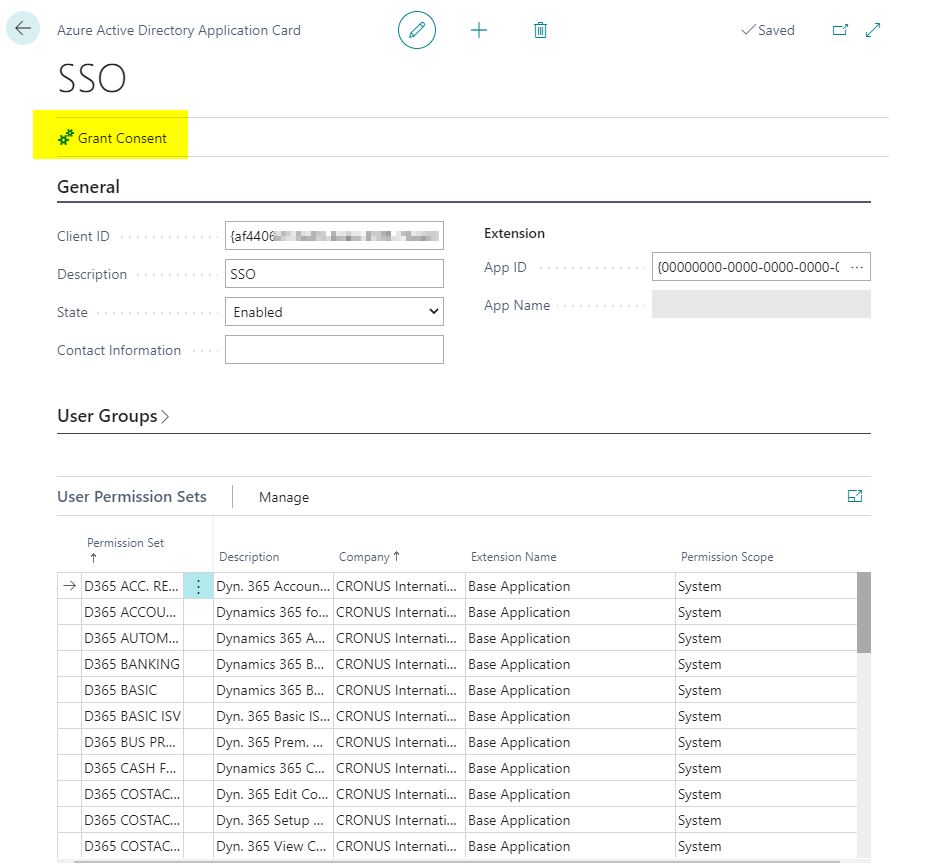
1) So, the first step is to **Configure SSO** for the Business Central Environment. All necessary steps can be found here:  
<https://docs.microsoft.com/en-us/dynamics365/business-central/dev-itpro/administration/authenticating-users-with-azure-active-directory>

**IMPORTANT**:

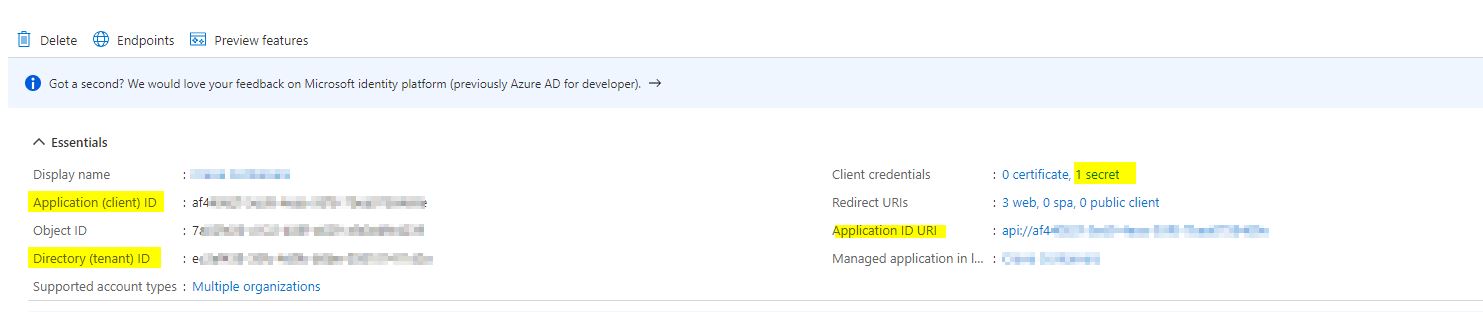
**Please note that if you set the BC Middle Tier Client Credentials type to “AccessControlService” (as mentioned in MS documentation), you will not be able to connect Visual Studio Code because at the moment VS Code does not support AAD authentication for OnPremise.  
The trick is to use Client Credentials type “NavUserPassword” instead. This way you are able to use AAD for Web Client and NavUserPassword for VS Code. We could not find this anywhere in MS documentation, but this combination works.**

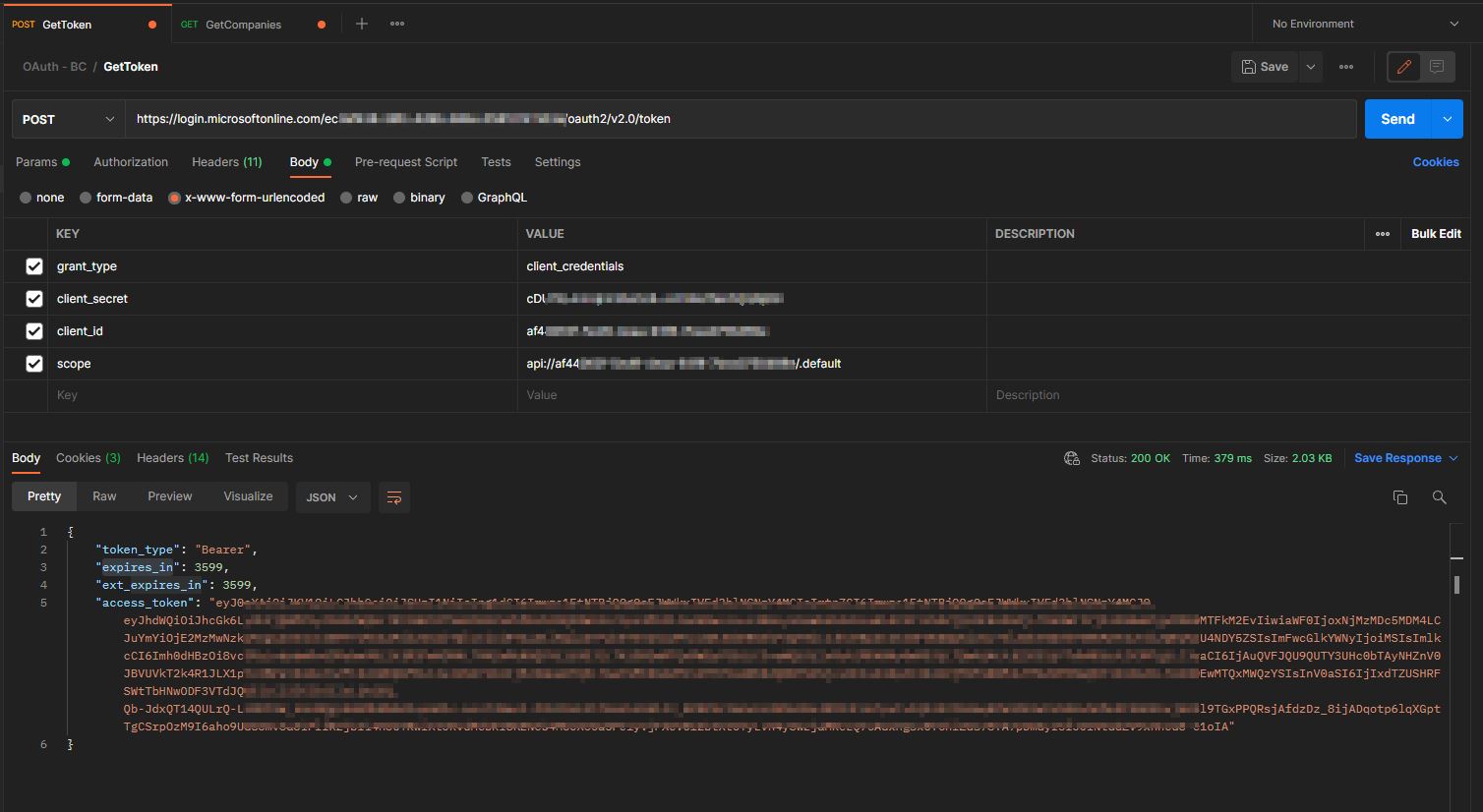
2) Create **Client Credentials secret**: select the Azure app created for Business Central in the previous step, press “Client Credentials” and create a new secret.

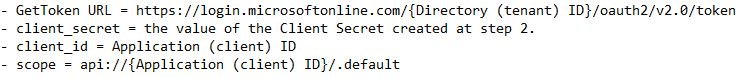
3) **Add Permissions in the Azure App**: go to “API Permissions” and add permissions for Business Central:  
[](https://andreilungu.com/wp-content/uploads/2021/10/01_Azure-DevOps-Permissions.jpg)

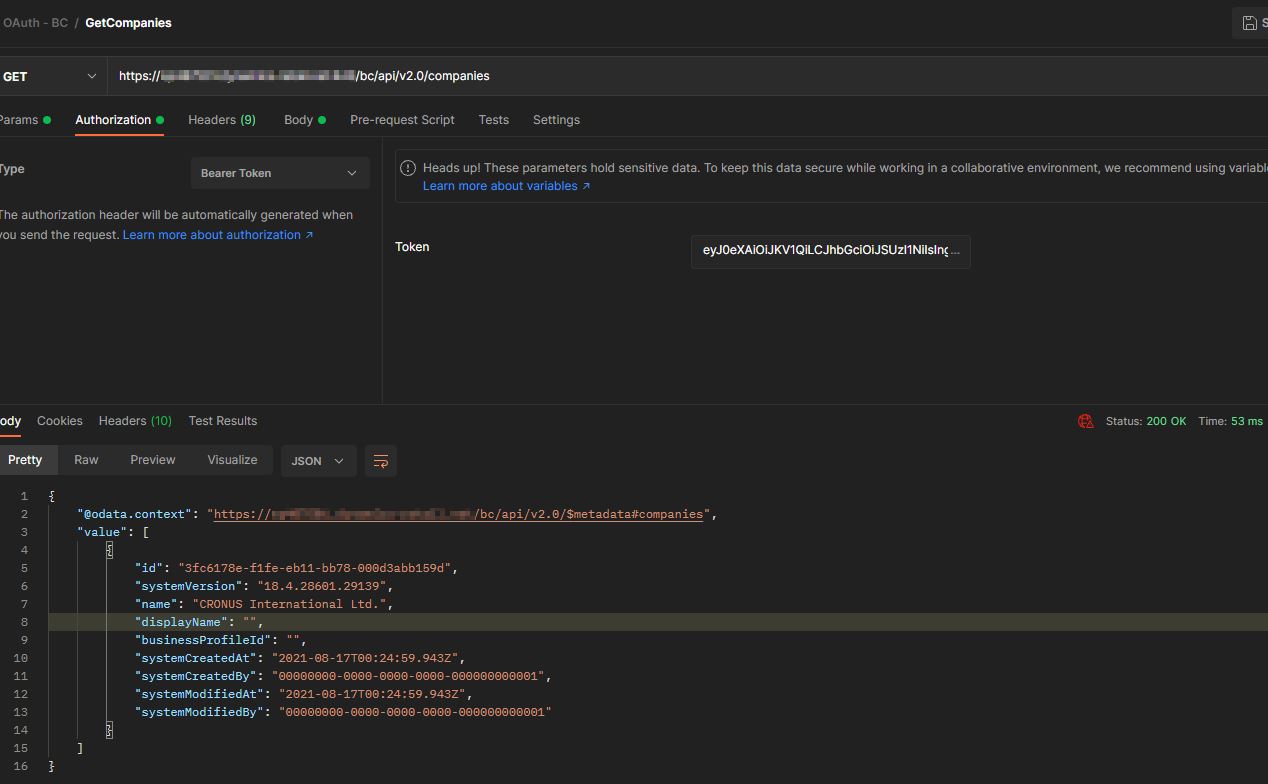
4) Register Azure App and **Add Permissions in Business Central**: Open Business Central Web Client, go to “Azure Active Directory Applications” and create a new application using the “Client Id” of the Azure App. Add BC permissions (please note that you cannot add SUPER permission). Then press action **“Grant Consent”**:  
[](https://andreilungu.com/wp-content/uploads/2021/10/02_BC-Permissions.jpg)

5) Now, if everything is setup correctly, you should be able to use OAuth2.0 client services flow:

So we have these parameters from the Azure app:  
[](https://andreilungu.com/wp-content/uploads/2021/10/03_Azure-AD-app-data.jpg)

And we can use them to:  
–> **Generate a Token** (using Postman in this example):  
[](https://andreilungu.com/wp-content/uploads/2021/10/04_Postman-Get-Token.jpg)

Mappings are:  
[](https://andreilungu.com/wp-content/uploads/2021/10/06_Mappings.jpg)

–> Copy the resulted **Token and use it to Authenticate** to Business Central API (companies in this example):  
[](https://andreilungu.com/wp-content/uploads/2021/10/05_Postman-Get-Companies.jpg)

Please note that token has expiration date 1 hour by default. I think this can be changed in Azure portal. But it means that any implementation of OAuth2.0 authentication should have a logic to check if existing Token is expired and if it is not expired yet, use it, if it is expired, generate a new valid token.

## Conclusion

To setup all of the above is not straightforward, but once all needed configurations are done, using OAuth for Service to Service Authentication in Business Central On Premises is not so scary anymore.