FY18 Azure Active Directory & Multitenant App

What is that “MULTITENANT”

Four simple characteristics

One reason:
LOWER COST
(for large numer of customers)

Multiple business clients – **tenants** (customers)

- Customer own users data repository (**Azure Active Directory**)
- Application trust that repository
- (Simple) self-service onboarding process

Single code base, single app instance for all customers

Single data repository

- Or – application will select data source based on tenant

(Maybe) custom logic for particular tenants

- "Workflow"
- Additional functionalities (modules etc.)
- **Configuration!**

Development costs
vs maintenance costs

Users in an application

Trivial – but necessary

Authentication – WHO, VERIFIED IDENTITY

- **Process of ascertaining that somebody really is who he claims to be.**
- How
  - Obsolete, use with caution: ~~User / Password in SQL, Active Directory + Kerberos / NTLM~~
  - Current, use external identity provider (why? security, GDPR/RODO, …)
    - Focus on PROTOCOL: SAML, SAML-P, **OAUTH2**, **OpenID Connect**
      (older, SOA times: WS-Federation / WS-Security)
    - Implementation: **Azure Active Directory**, **AAD B2C**, Identity Server (.NET Foundation), Auth0 (also broker), …

Authorization – WHAT CAN DO, PERMISSIONS

- **Rules that determine who is allowed to do what**
- Can (but not HAVE TO) use claims, roles, groups.
- Logic in App; framework (.NET Core: AddAuthorization, AddPolicy, ClaimsPrincipal, [Authorize] …)

Po polsku

Authentication: Uwierzytelnianie

Authorization : Autoryzacja

Helpful tool: Microsoft Graph API

REST API: relations between data in Office 365

Subset of API (Groups, Roles, Users) apply to Azure Active Directory

(Demos)

https://graph.microsoft.io/en-us/docs

https://graph.microsoft.io/en-us/graph-explorer

https://graph.microsoft.com/v1.0/me
https://graph.microsoft.com/v1.0/microsoft.com/me
https://graph.microsoft.com/v1.0/microsoft.com/me?$select=surname
https://graph.microsoft.com/v1.0/microsoft.com/users/piotrc@microsoft.com
https://graph.microsoft.com/v1.0/me/messages?$top=5&$skip=5&$orderby=createdDateTime
https://graph.microsoft.com/v1.0/me/events
https://graph.microsoft.com/v1.0/me/drive/root/children
https://graph.microsoft.com/v1.0/users/piotrc@microsoft.com/drive/root
https://graph.microsoft.com/v1.0/microsoft.com/groups?$filter=securityEnabled+eq+true
https://graph.microsoft.com/beta/me/workingWith
https://graph.microsoft.com/beta/me/trendingAround
https://graph.microsoft.com/beta/me/tasks
https://graph.microsoft.com/beta/me/notes

Authentication

Important:

Delegated – per user

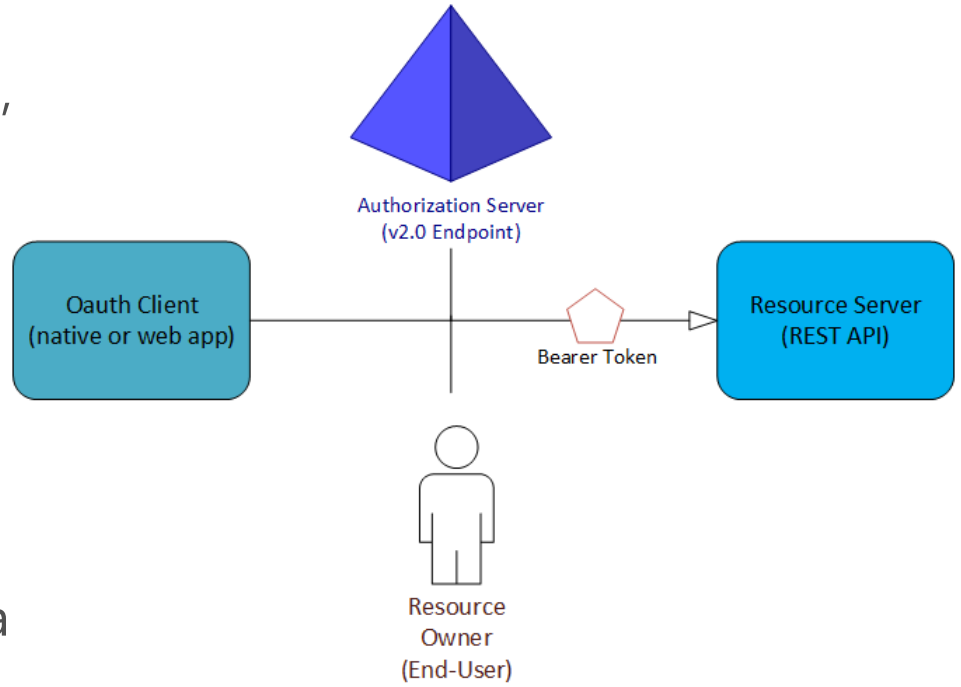
Per App - later

Basics – RFC6749,6750 + openid.net/connect

Authorization Server

Responsible for ensuring the user's identity, granting and revoking access to resources, and issuing tokens. Identity provider.

Handles anything to do with the user's information, their access, parties in an flow.



Resource Owner

Owns the data, allow 3rd parties to access data or resource.

OAuth Client

Identified by its Application Id. Party that the end-user interacts authorization server. The client must be granted permission to a owner.

Resource Server

Where the resource or data resides. Trusts the Authorization Server, authenticate and authorize OAuth Client, and uses (Bearer) access_tokens to ensure that access to a resource can be granted.

Type of tokens

Access Token

Credentials used to access protected resources

No defined format!!!

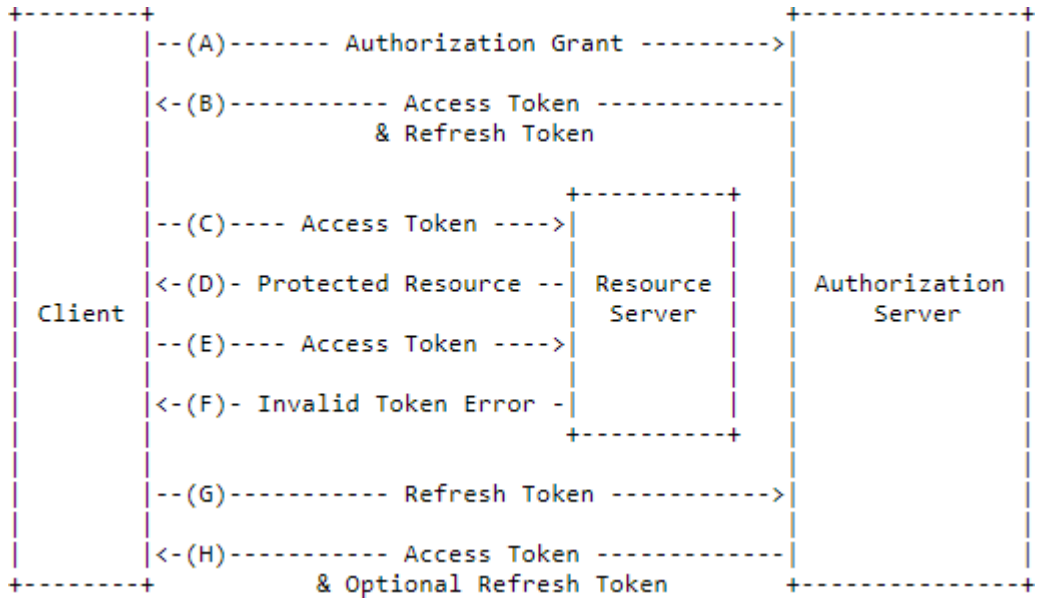
Refresh Token

Credentials used to obtain access tokens. Issued to the client by the authorization server and are used to obtain a new access token when the current access token becomes invalid or expires

RFC 6750 The OAuth 2.0 Authorization Framework

What is and how to use Bearer token in HTTP scenarios

JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication

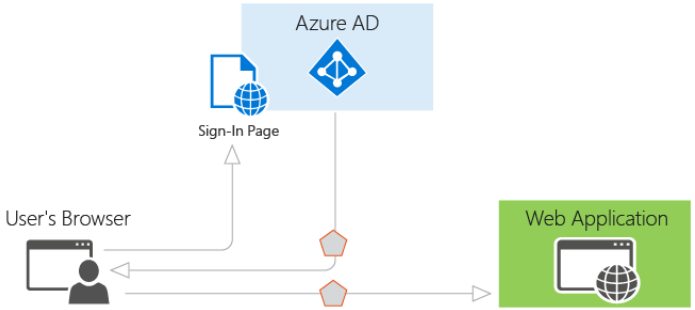


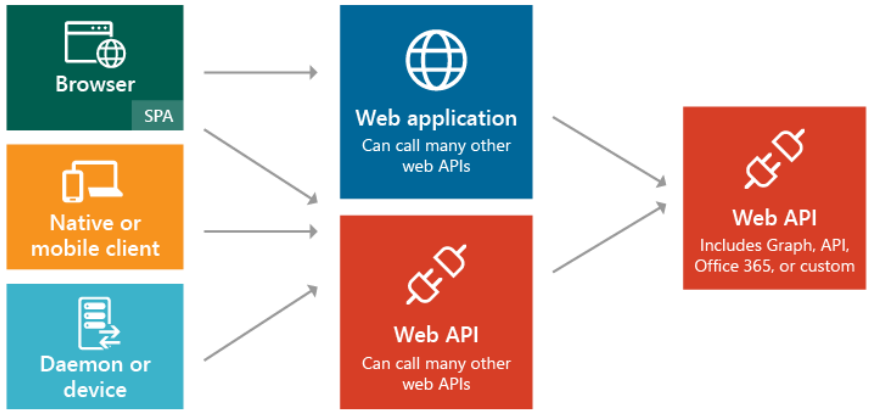
Token Lifetimes

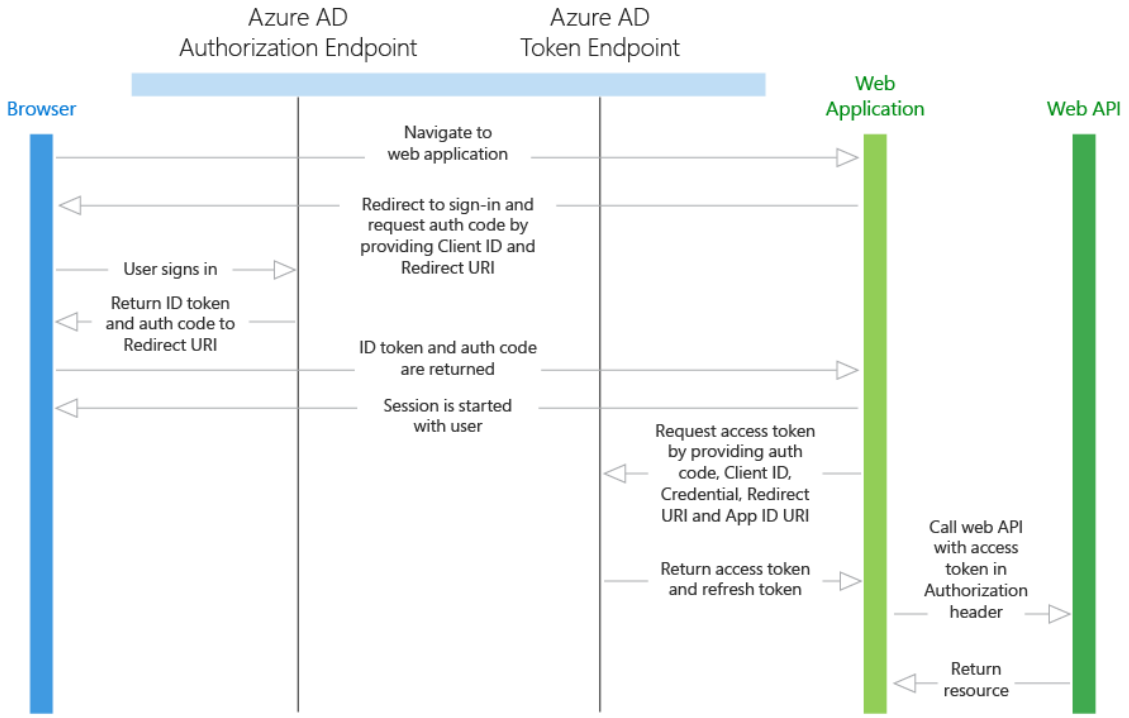
Property	Policy property string	Affects	Default	Minimum	Maximum
Access Token Lifetime	AccessTokenLifetime	Access tokens, ID tokens, SAML2 tokens	1 hour	10 minutes	1 day
Refresh Token Max Inactive Time	MaxInactiveTime	Refresh tokens	90 days	10 minutes	90 days
Single-Factor Refresh Token Max Age	MaxAgeSingleFactor	Refresh tokens (for any users)	Until-revoked	10 minutes	Until-revoked[1]
Multi-Factor Refresh Token Max Age	MaxAgeMultiFactor	Refresh tokens (for any users)	Until-revoked	10 minutes	Until-revoked[1]
Single-Factor Session Token Max Age	MaxAgeSessionSingleFactor[2]	Session tokens (persistent and nonpersistent)	Until-revoked	10 minutes	Until-revoked[1]
Multi-Factor Session Token Max Age	MaxAgeSessionMultiFactor[3]	Session tokens (persistent and nonpersistent)	Until-revoked	10 minutes	Until-revoked[1]

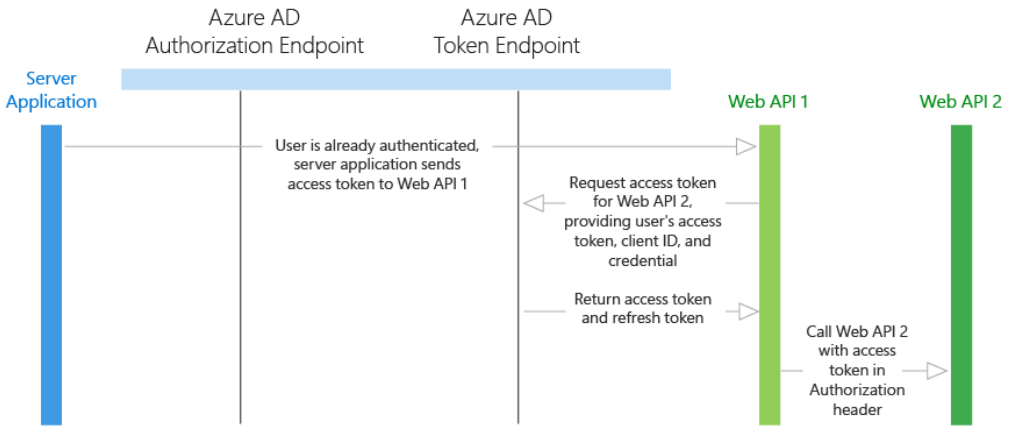
```
Connect-AzureAD

Get-AzureADPolicy

New-AzureADPolicy -Definition @('{"TokenLifetimePolicy":{"Version":1,"MaxAgeSingleFactor":"until-revoked"}}') -DisplayName "OrganizationDefaultPolicyScenario" -IsOrganizationDefault $true -Type "TokenLifetimePolicy“

Until revoked – generally up to 365 days
```

Authentication scenarios for Azure AD (v1)









OAuth2 vs OpenID Connect - remarks

Access token + id token (signed JWT with information about the authenticated user)

Standardizes: token format, instance scopes, endpoint discovery, and dynamic registration of clients.

Short: Identity FRAMEWORK

OpenID Connect - http://openid.net/specs

The OpenID Connect 1.0 specification consists of these documents:

Core – Defines the core OpenID Connect functionality: authentication built on top of OAuth 2.0 and the use of Claims to communicate information about the End-User

Discovery – (Optional) Defines how Clients dynamically discover information about OpenID Providers

Dynamic Registration – (Optional) Defines how clients dynamically register with OpenID Providers

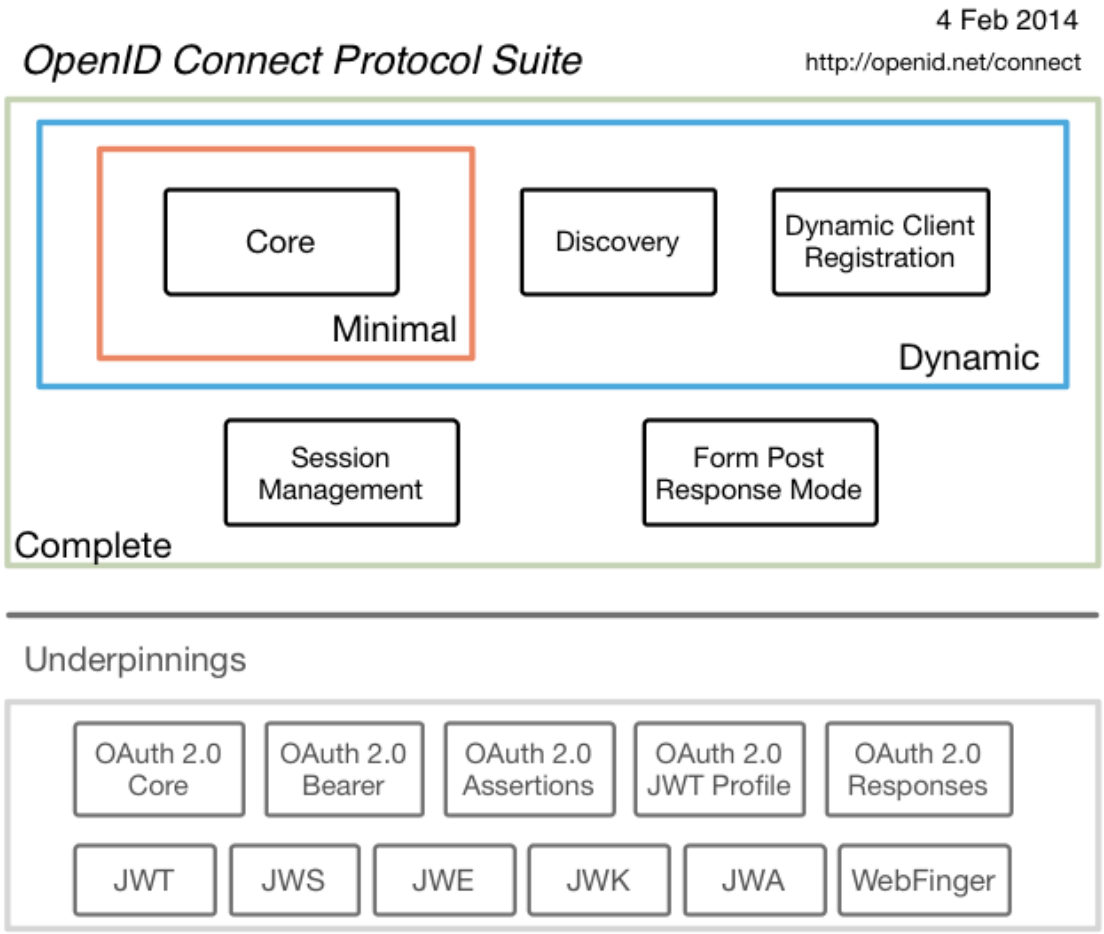
OAuth 2.0 Multiple Response Types – Defines several specific new OAuth 2.0 response types

OAuth 2.0 Form Post Response Mode – (Optional) Defines how to return OAuth 2.0 Authorization Response parameters (including OpenID Connect Authentication Response parameters) using HTML form values that are auto-submitted by the User Agent using HTTP POST

Session Management – (Optional) Defines how to manage OpenID Connect sessions, including postMessage-based logout functionality

Front-Channel Logout – (Optional) Defines a front-channel logout mechanism that does not use an OP iframe on RP pages

Back-Channel Logout – (Optional) Defines a logout mechanism that uses direct back-channel communication between the OP and RPs being logged out



OpenID - RequestToken

[OpenID Connect Implicit Flow]

[OpenID Connect Hybrid Flow]

Tokens:

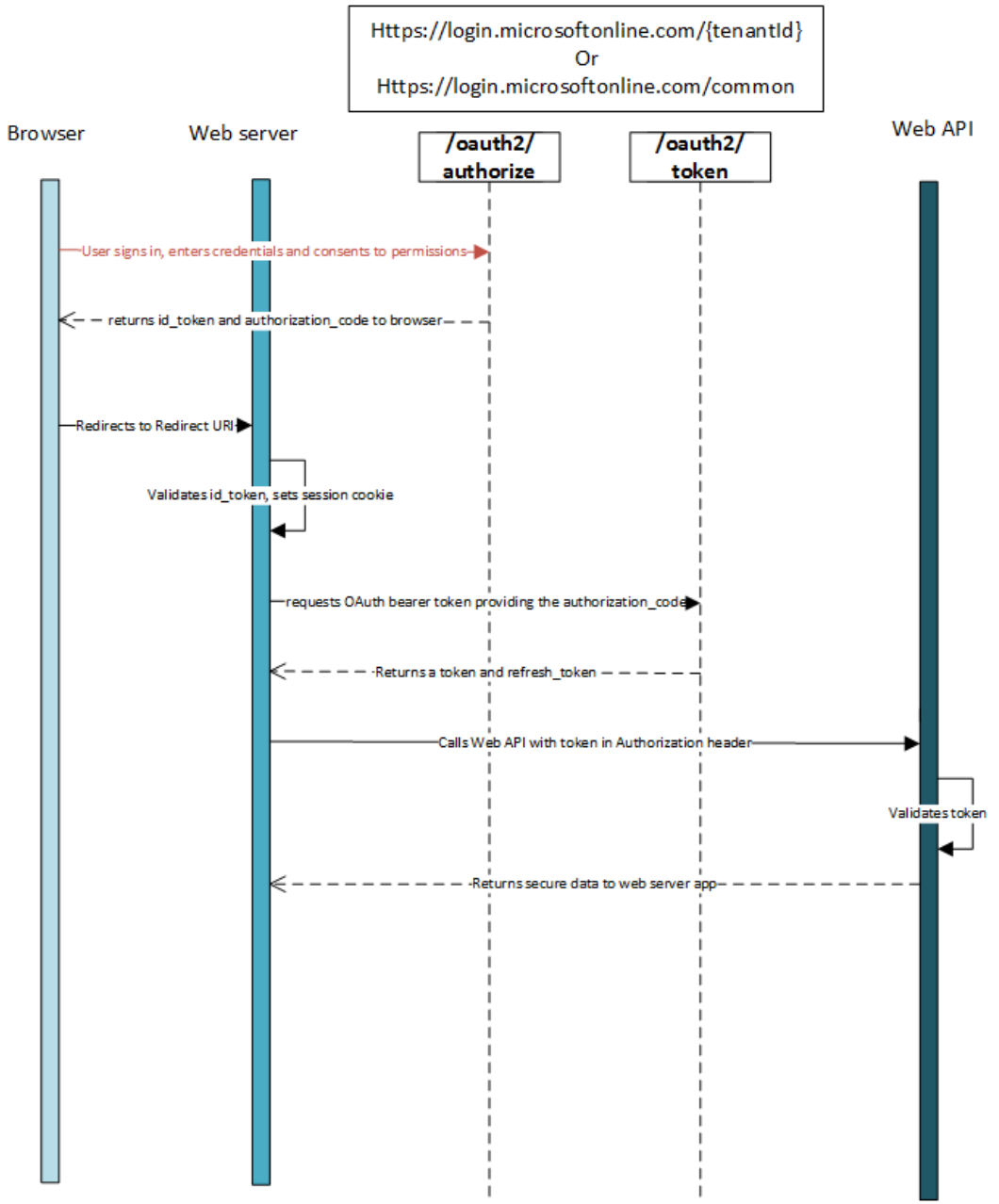
code - Authorization Code. To get (later) access token for resource.

To get access token: **ClientId**(from AAD registration), **ClientSecret** (from AAD registration), redirect URI (the same as for id_token!) and of course **code**

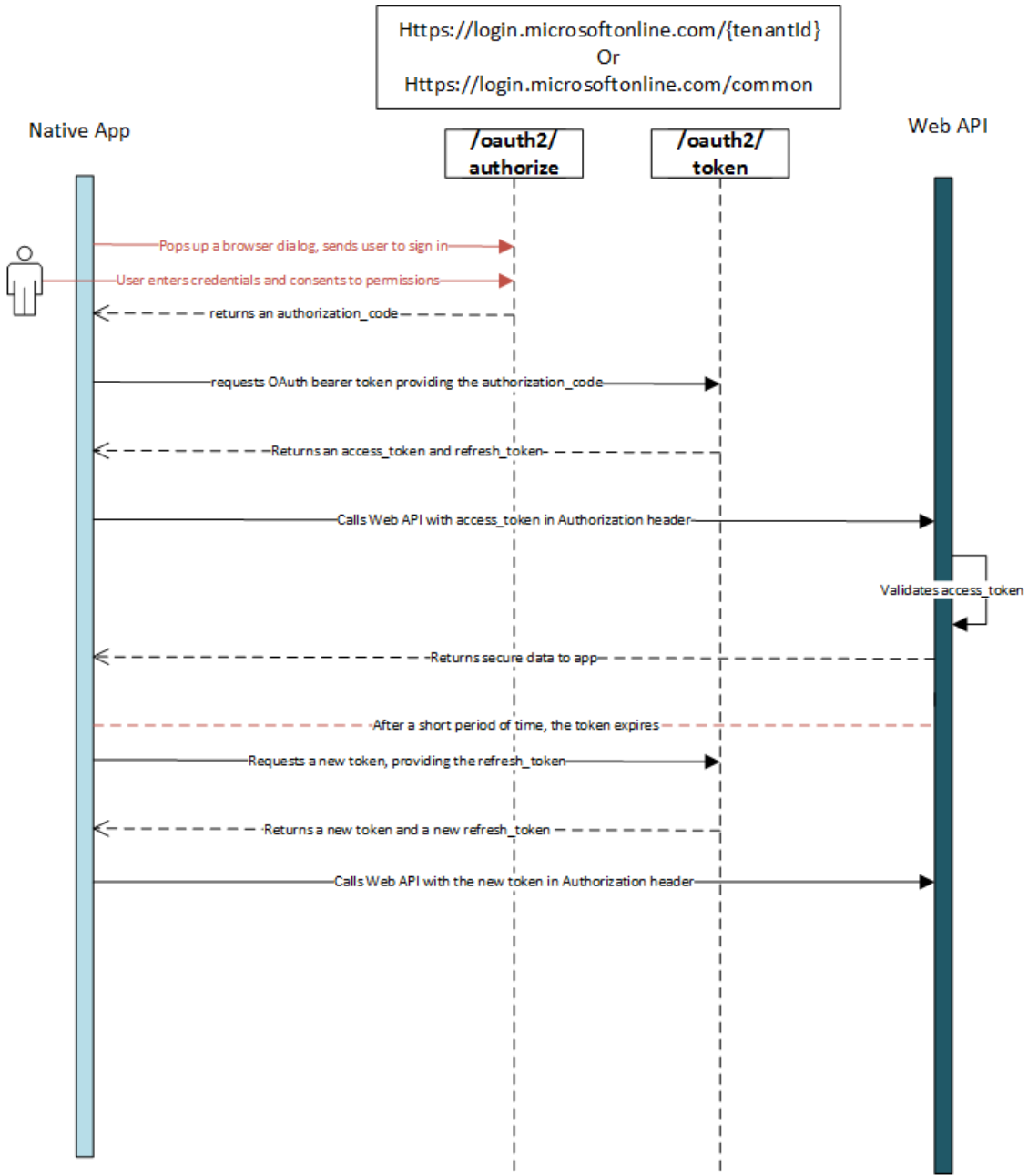
token - Access Token (implicit flow)

id_token - ID Token. Confirmation of identity, JWT

OpenID Connect (Web)

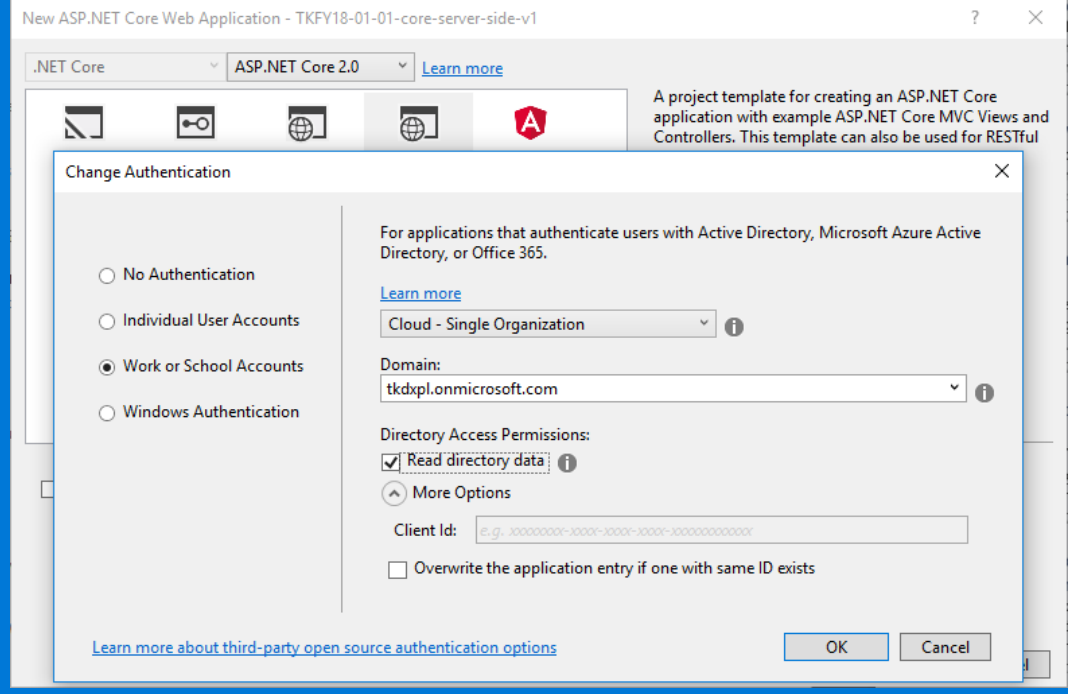


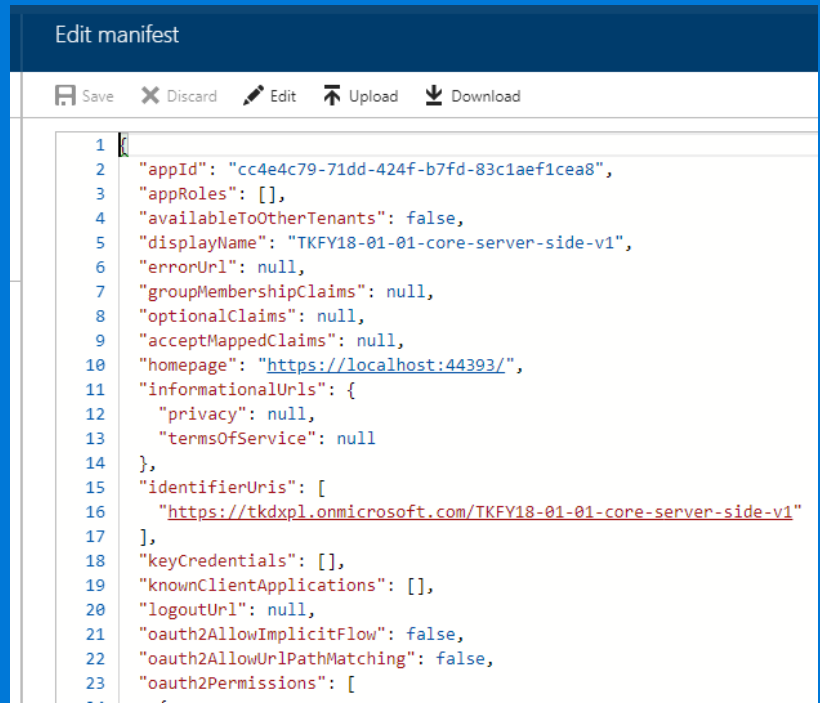
OAuth 2.0, not only Web



Demo 01-01

Server-side,
AAD v1, creator





groupMembershipClaims=null

TKFY18_01_01_core_server_side_v1 Home About Contact Hello tkadmin@tkdxpl.on

http://schemas.microsoft.com/claims/authnmethodsreferences: pwd

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname: Admin

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname: TK

name: tkadmin

http://schemas.microsoft.com/identity/claims/objectidentifier: 7fff78c6-c4aa-47ba-87b3-0b2ece50cac1

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier: Eu6TmafYxi5jrWoI5LhKWvMVl1iXmimky0p3Pwf-o0A

http://schemas.microsoft.com/identity/claims/tenantid: a757c7b8-69a2-4b92-b277-be767fc38487

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name: tkadmin@tkdxpl.onmicrosoft.com

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn: tkadmin@tkdxpl.onmicrosoft.com

uti: q_w6jtC9QkmASVk-zq8jAA

groupMembershipClaims=7

TKFY18_01_01_core_server_side_v1 Home About Contact Hello tkadmin@tkdxpl.or

http://schemas.microsoft.com/claims/authnmethodsreferences: pwd

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname: Admin

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname: TK

groups: 57fda17b-7e8d-4ba6-8e0d-8a8fe4539564

groups: 6da75b3f-a839-461e-9e2a-1305be67ff46

groups: 8542e184-3375-49de-8401-131a73ed9d9c

groups: 6ff8e6f5-38e6-4667-9805-f7e4dd58c45f

name: tkadmin

http://schemas.microsoft.com/identity/claims/objectidentifier: 7fff78c6-c4aa-47ba-87b3-0b2ece50cac1

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier: Eu6TmafYxi5jrWoI5LhKWvMVl1iXmimky0p3Pwf-o0A

http://schemas.microsoft.com/identity/claims/tenantid: a757c7b8-69a2-4b92-b277-be767fc38487

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name: tkadmin@tkdxpl.onmicrosoft.com

http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn: tkadmin@tkdxpl.onmicrosoft.com

uti: KDngVynU20OxQbgXQkUrAA

wids: 62e90394-69f5-4237-9190-012177145e10

Manifest – selected options

appID (client ID)

groupMembershipClaims

Null/0 (None) | 1 (security + roles) | 7 (security + roles + distribution)

knownClientApplications

Consent for additional WebAPI

oauth2AllowImplicitFlow

replyUrls

requiredResourceAccess

Implicit flow = JavaScript client

Flow:

The implicit grant is a simplified authorization code flow optimized for clients implemented in a browser using a scripting language such as JavaScript. In the implicit flow, instead of issuing the client an authorization code, the client is issued an access token directly (as the result of the resource owner authorization). The grant type is implicit, as no intermediate credentials (such as an authorization code) are issued (and later used to obtain an access token).

CAUTION

Short: we trust EVERY client

So: Implicit + CORS

Or: Trust redirect URI

10.16. Misuse of Access Token to Impersonate Resource Owner in Implicit Flow

For public clients using implicit flows, this specification does not provide any method for the client to determine what client an access token was issued to.

When issuing an access token during the implicit grant flow, the authorization server does not authenticate the client. In some cases, the client identity can be verified via the redirection URI used to deliver the access token to the client. The access token may be exposed to the resource owner or other applications with access to the resource owner's user-agent.

Demo 01-02

JavaScript, AAD v1

Demo 01-02 (Angular)

(vscode + edge) NodeJS + Angular + Graph API – V2
(far) Angular + WebAPI

https://blog.mehraban.com.au/2017/09/28/integrating-spa-azure-ad/

AAD – v1 vs v2 endpoint

V2: all apps are multitenant.

Consumer or business logins.

V2: Scopes not resources. Dynamic consent

V1: Change permission/manifest = remove registration on client tenant and/or force consent again
V2: App dynamically can change list of requested scopes

V2: WebAPI working, but we need to keep **the same Application ID** for client and WebAPI

V2: redirect URI: https only

V2: no SAML, WS-Federation

V2: Scopes not resources

Standard: openid, email, profile, offline_access

Graph API: User.ReadBasic.All (find more in docs)

Get Drive

Edit in GitHub

Retrieve the properties and relationships of a Drive resource.

A Drive is the top-level container for a file system, such as OneDrive or SharePoint document libraries.

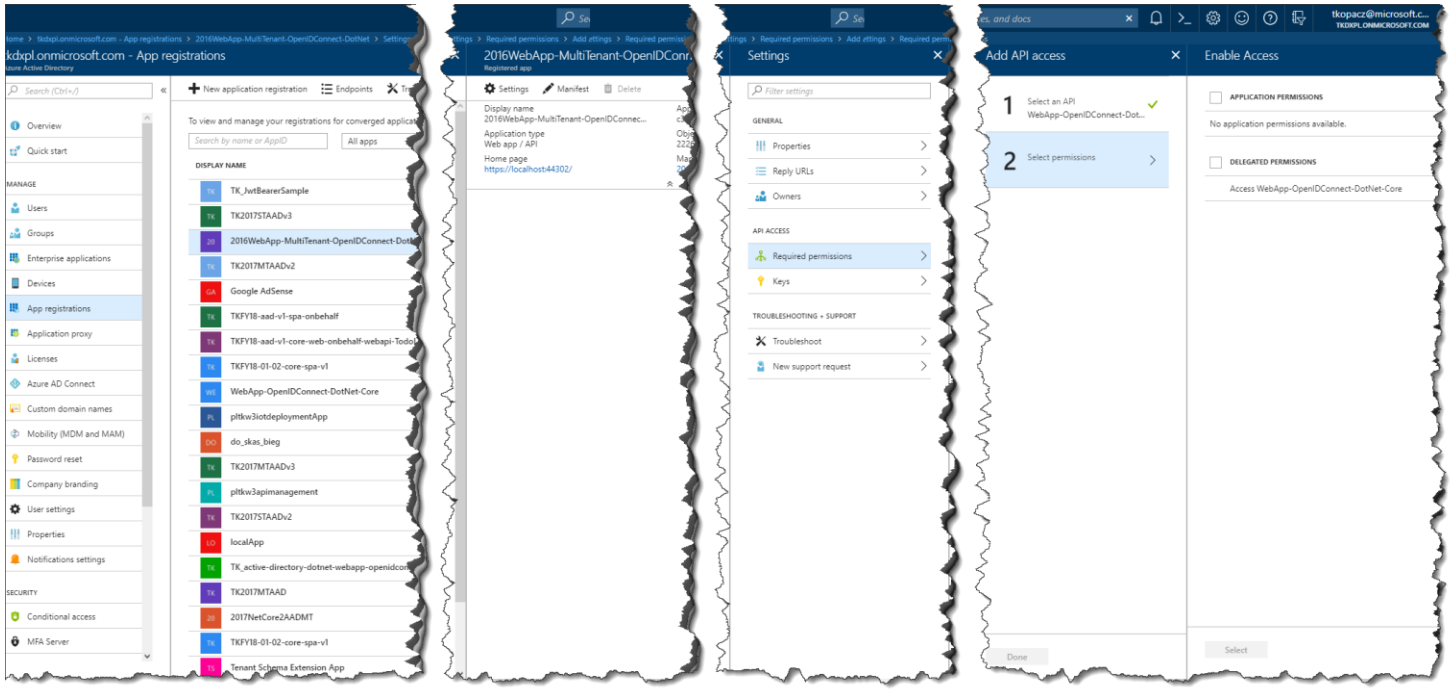
Permissions

One of the following permissions is required to call this API. To learn more, including how to choose permissions, see Permissions.

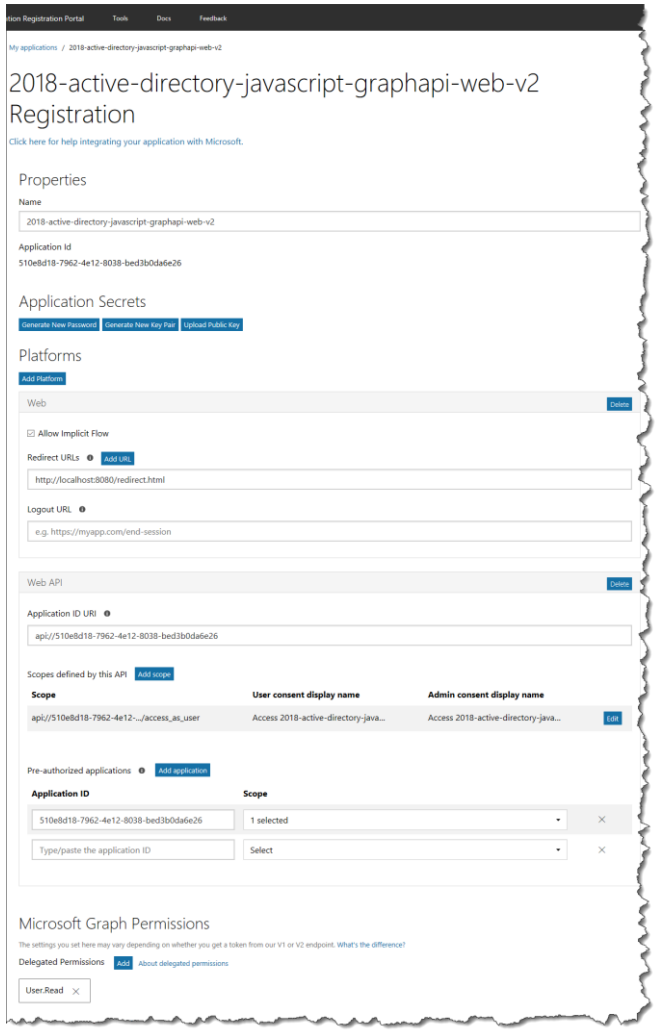
Permission type	Permissions (from least to most privileged)
Delegated (work or school account)	Files.Read, Files.ReadWrite, Files.Read.All, Files.ReadWrite.All, Sites.Read.All, Sites.ReadWrite.All
Delegated (personal Microsoft account)	Files.Read, Files.ReadWrite, Files.Read.All, Files.ReadWrite.All
Application	Files.Read.All, Files.ReadWrite.All, Sites.Read.All, Sites.ReadWrite.All

Registration

portal.azure.com



apps.dev.microsoft.com

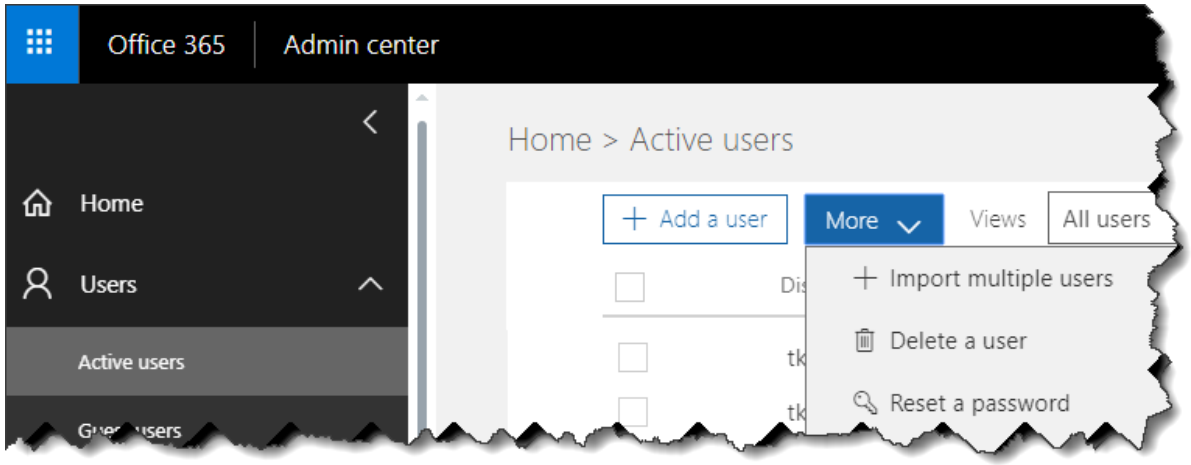


<Digression – Users & Federation>

Sources of users in Azure Active Directory

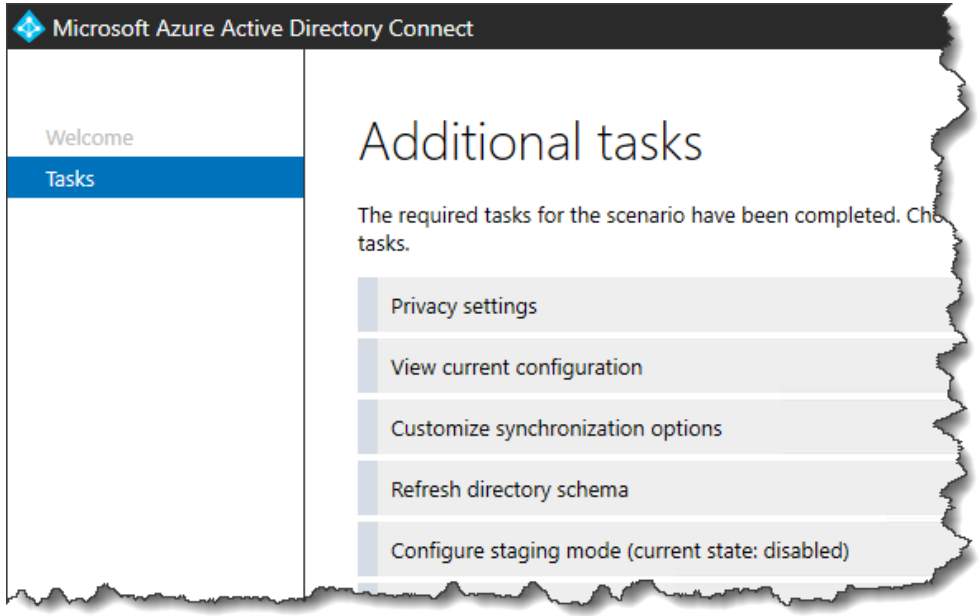
Manually (Powershell, CLI, CSV)

[az ad user create](), [New-AzureADUser]()
https://portal.office.com/adminportal
(even without M365)

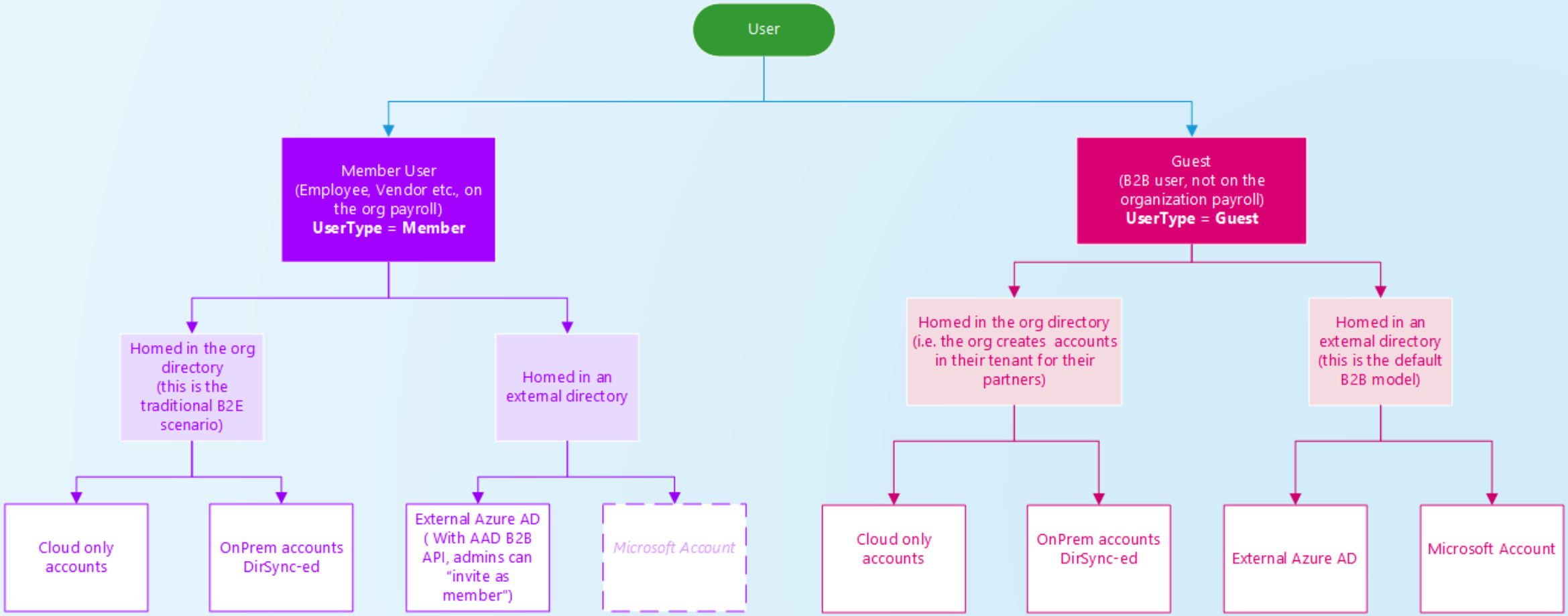


Active Directory (LDAP)

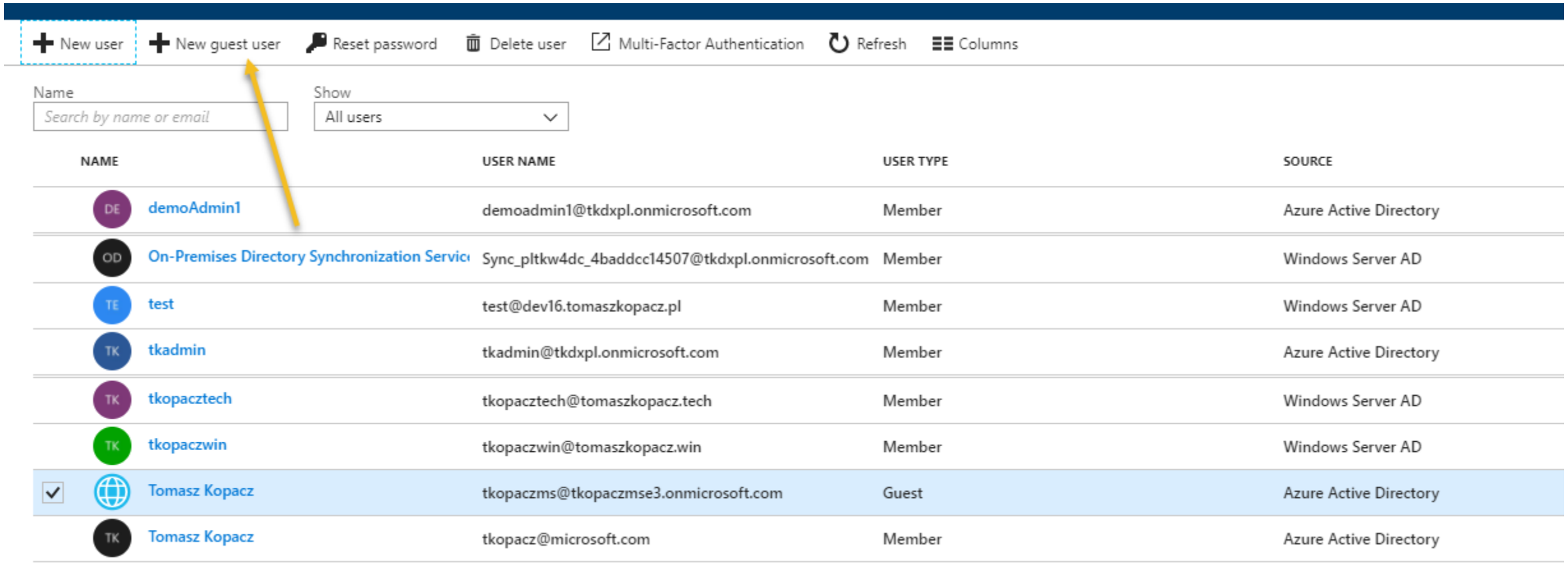
Using Microsoft Azure Active Directory Connect
- Federation (WAP Server + AD controller)
- Synchronization (could be run on AD controller)



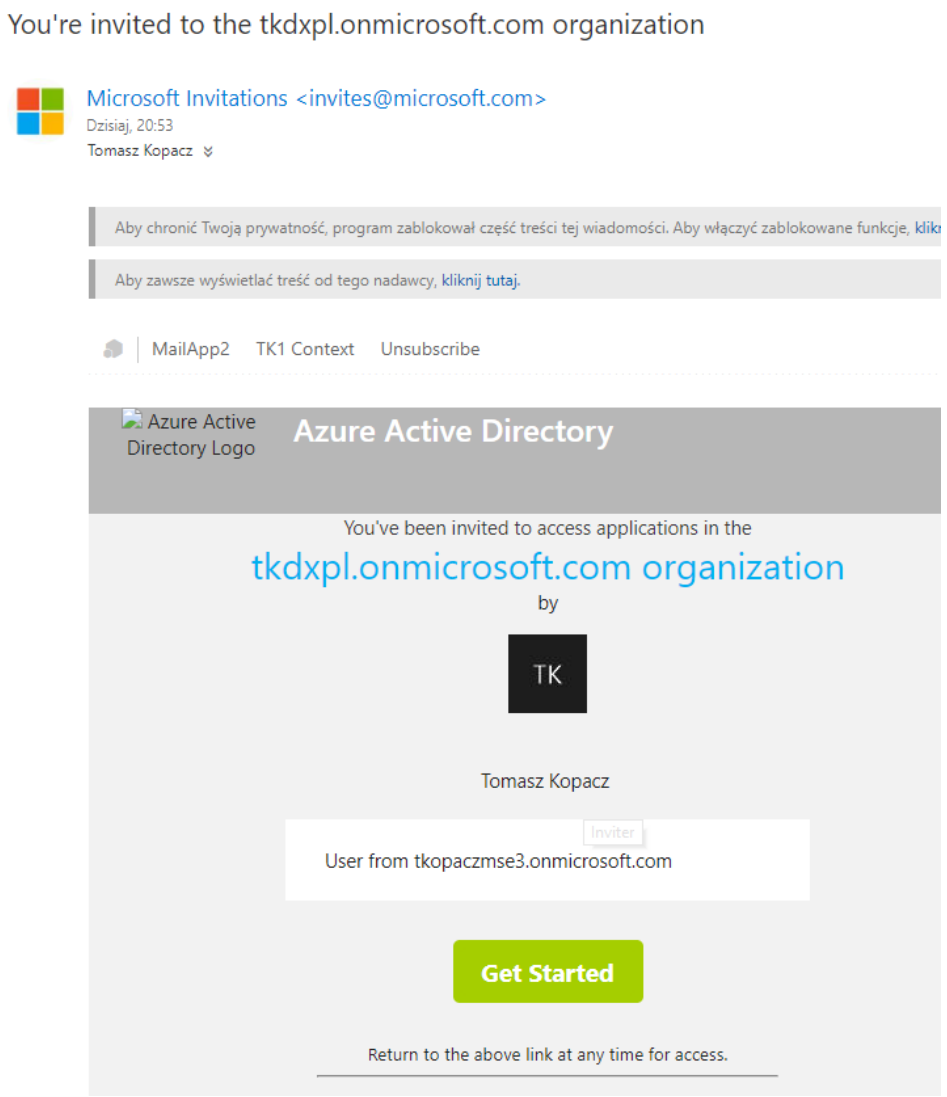
Azure B2B, aka „guest"



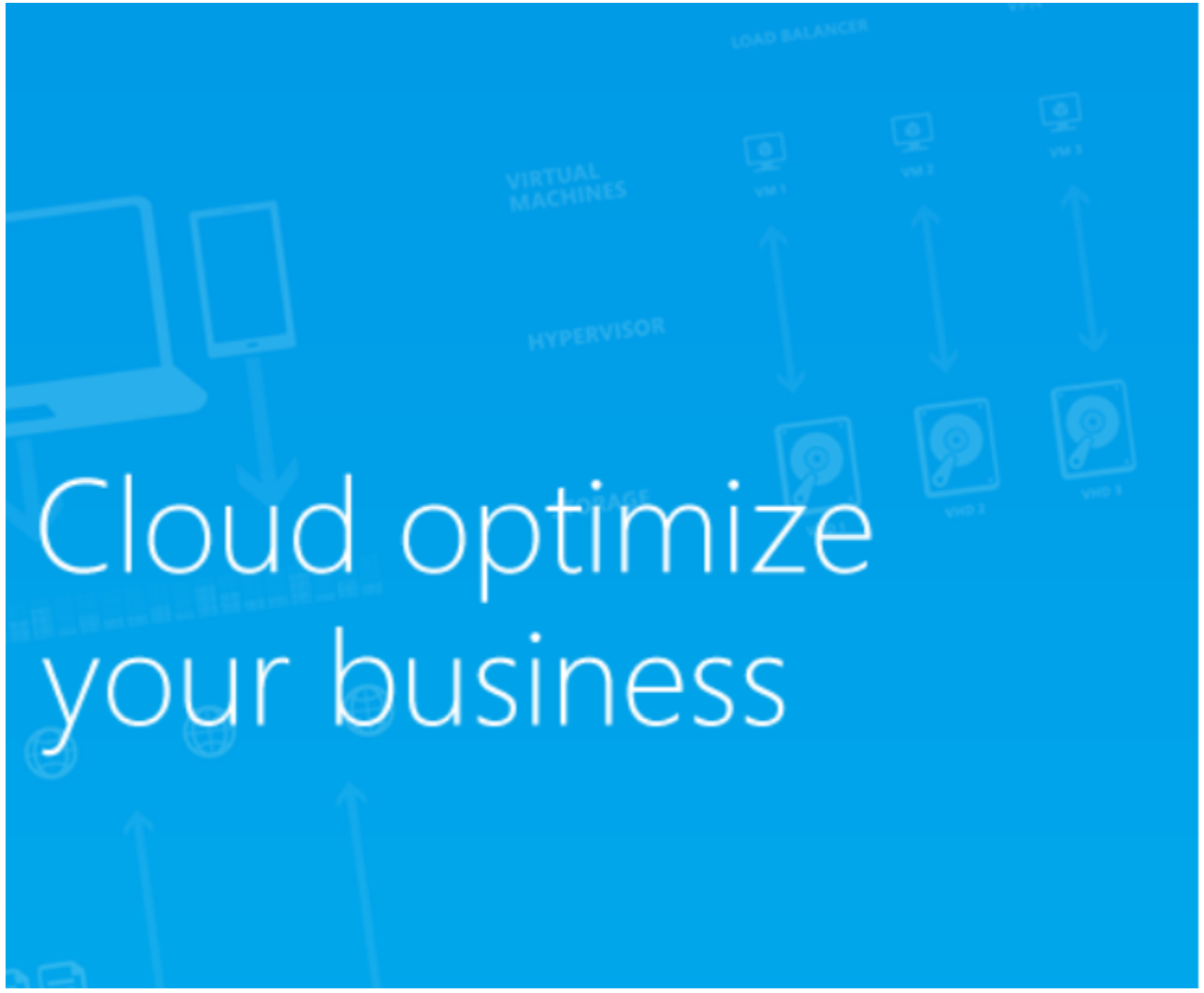
First, limited



Mail - invitation



Confirmation



Welcome to tkdxpl.onmicrosoft.com

You have been invited to access myapps.microsoft.com

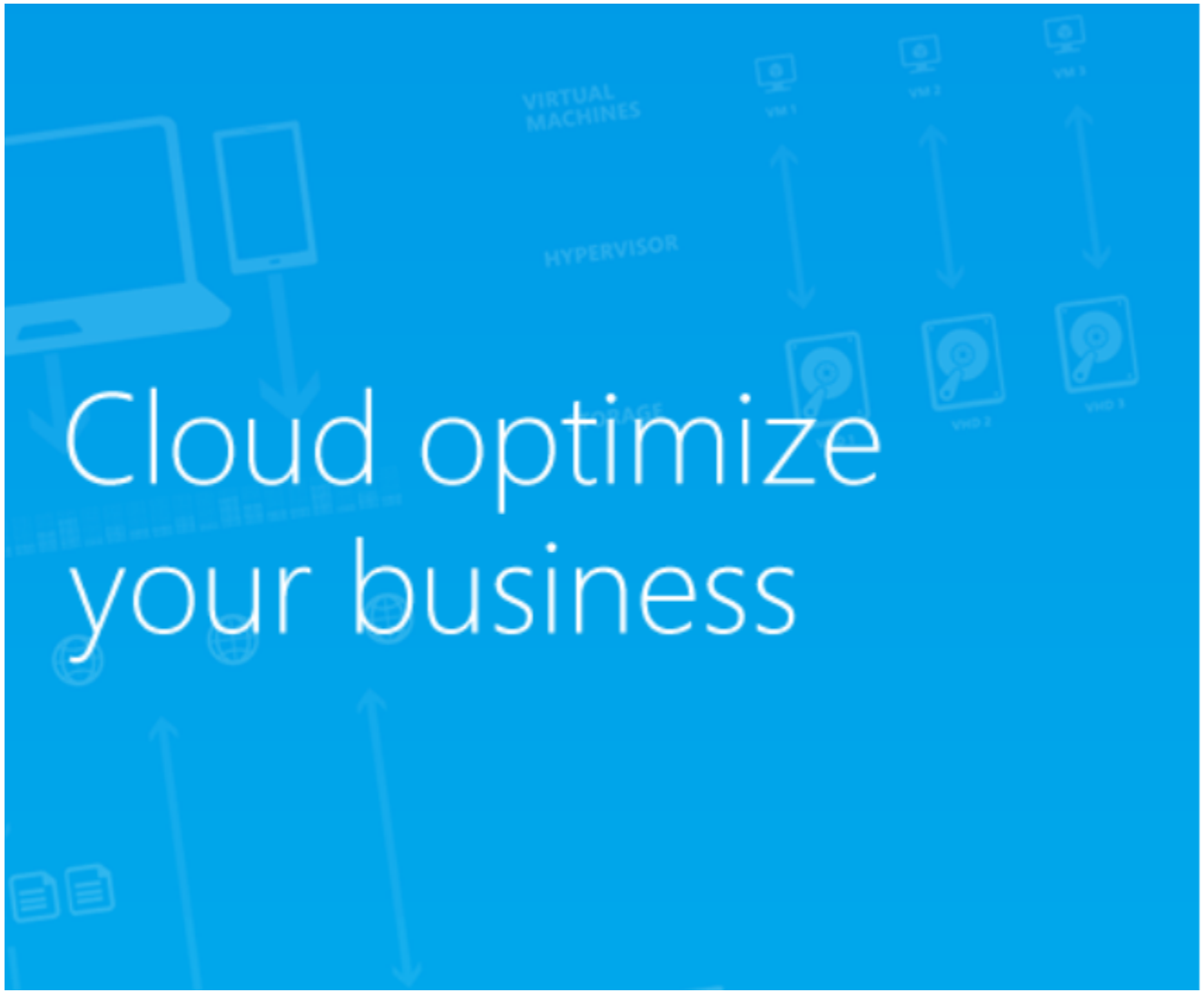
To access applications in the tkdxpl.onmicrosoft.com organization, you'll need to set up [redacted] as an account with Microsoft.

By clicking Next, tkdxpl.onmicrosoft.com will have access to your display name and email address.

Next

Note: After completing sign in you will be redirected to: https://myapps.microsoft.com/?tenantid=a757c7b8-69a2-4b92-b277-be767fc38487&login_hint=[redacted]

Setup password



Set up your account with Microsoft

You'll use it to access resources in the tkdxpl.onmicrosoft.com organization, and applications from Microsoft.

[redacted]

••••••••

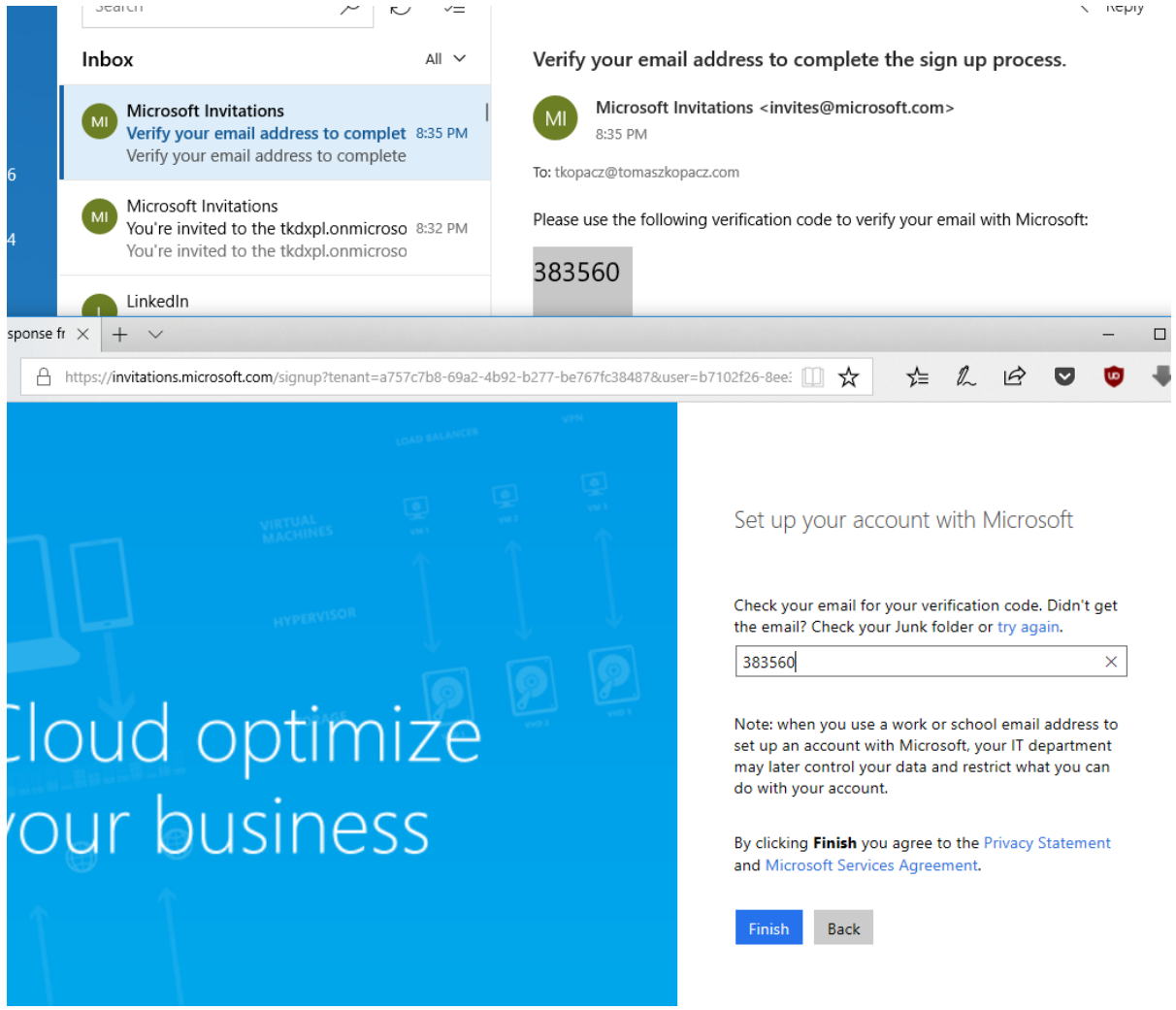
8-character minimum; case sensitive.

Tell us a bit more about yourself.

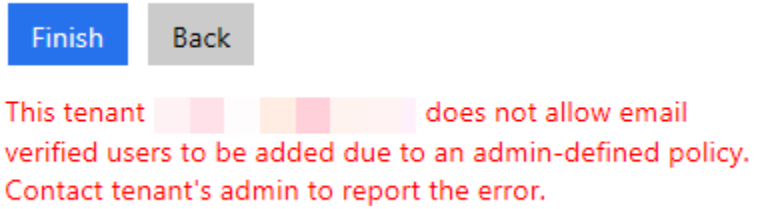
tkopacz (guest)

Next

Confirmation email



Change policy if needed



PowerShell

```
Install-Module MSOnline
Connect-MsolService
```

For: tkopaczmse3.onmicrosoft.com

```
Get-MsolCompanyInformation
Set-MsolCompanySettings -AllowEmailVerifiedUsers
$true
```

Demo

Browse tkdxpl configuration

</Digression – Users & Federation>

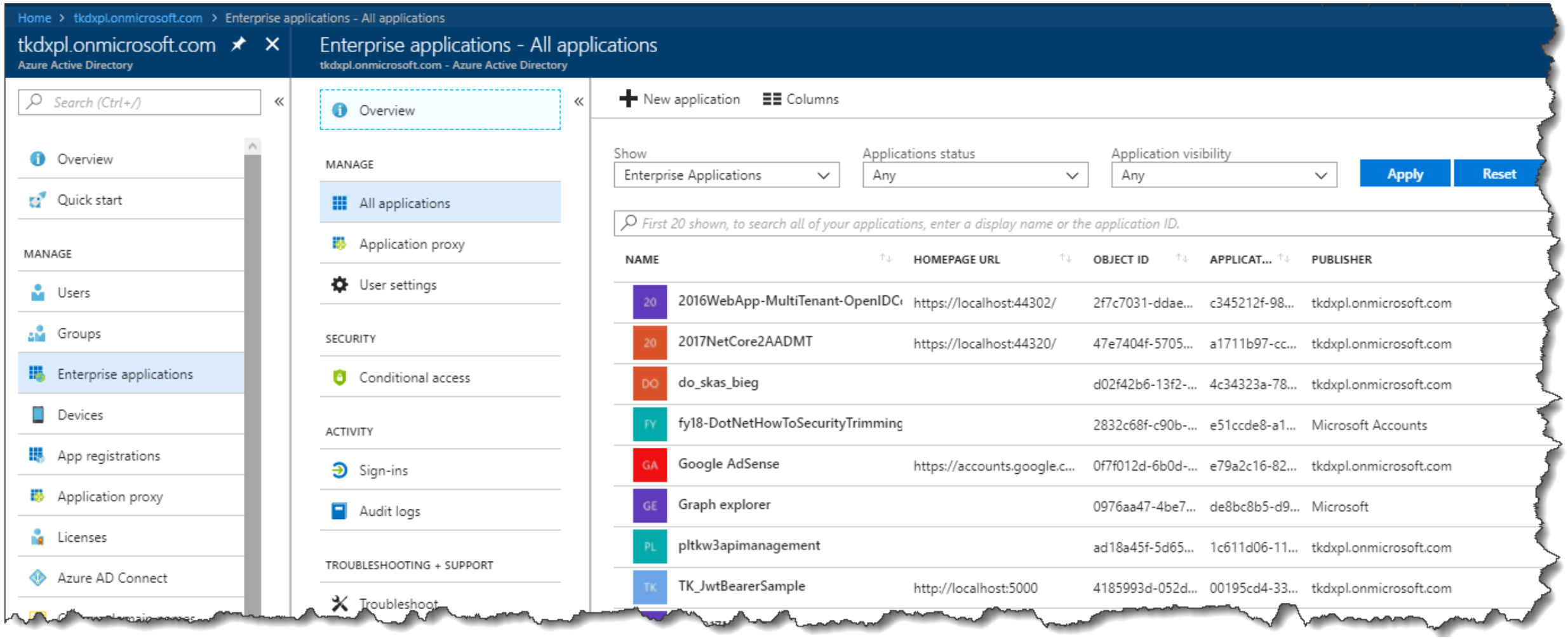
Fiddler + https://jwt.io

Demo 01-03

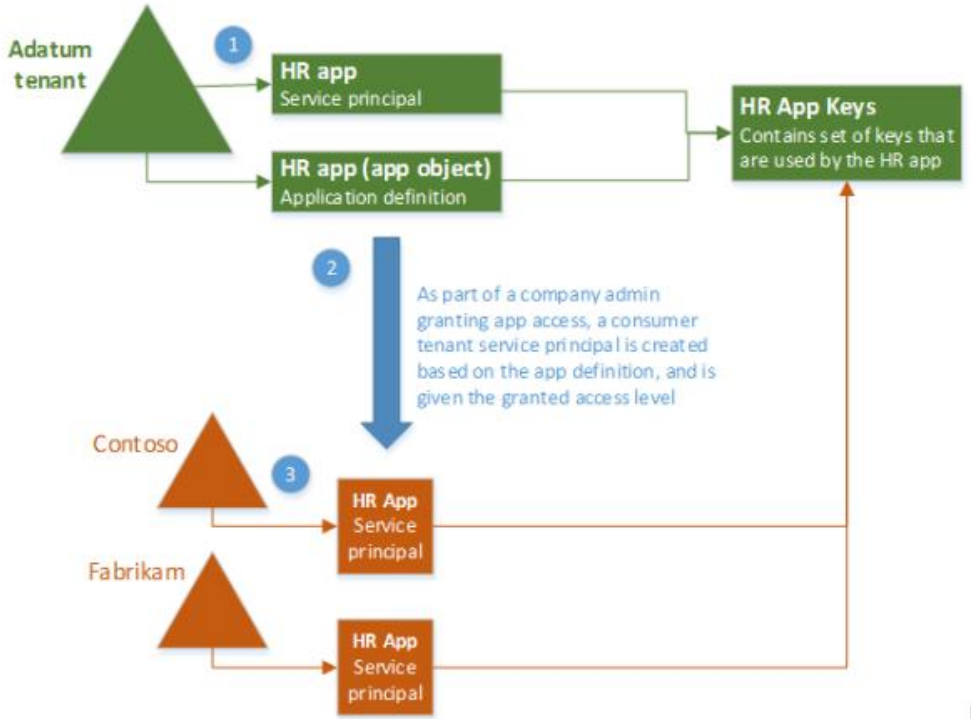
(far + Edge) JavaScript, AAD v2, No MS Libraries
(really, AAD use open standards)

<Digression – Consent and Apps – how to revoke>

AAD (admin) -> Enterprise Applications (v1 / v2)



How it works?

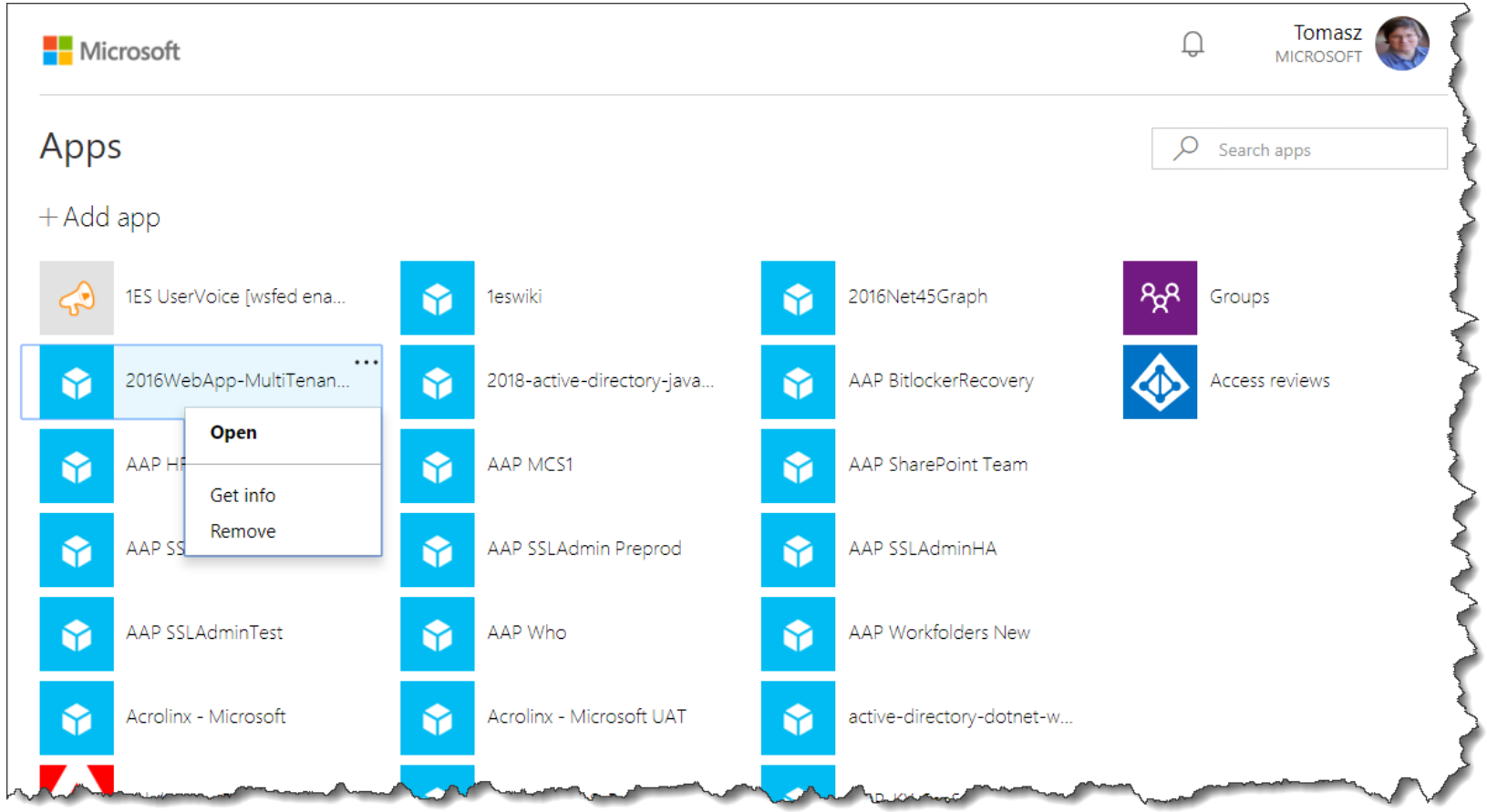



```
az login --tenant tkdxpl1.onmicrosoft.com
--allow-no-subscriptions
```

```
az ad sp list -o table
```

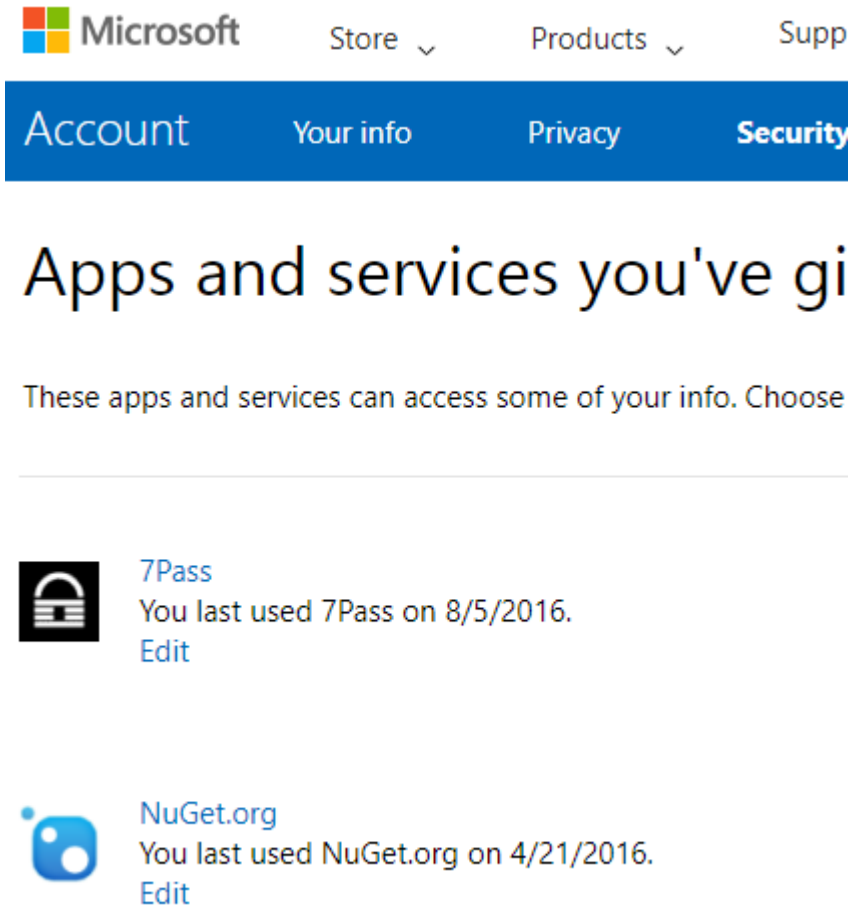
```
Z:\AzureFY18TK>az ad sp list -o table
AppId                                 DisplayName                          ObjectId                              ObjectType
------------------------------------  -----------------------------------  ------------------------------------  ----------
00000014-0000-0000-c000-000000000000  Microsoft.Azure.SyncFabric            03322556-19a6-473b-a2bc-8e46626e2ba6  ServicePrincipal
3a2eaceb-1ff3-49a4-9156-dc7fd7b15409  TK2017MTAADv3                        1dba6773-972c-4cd2-98e7-b426a1478806  ServicePrincipal
b4bddae8-ab25-483e-8670-df09b9f1d0ea  Signup                               209377d1-5699-42cf-9e15-d772552b41de  ServicePrincipal
982bda36-4632-4165-a46a-9863b1bbcf7d  O365 Demeter                         454a689c-85c1-42ef-a422-5221099b26b2  ServicePrincipal
00000013-0000-0000-c000-000000000000  Azure Classic Portal                 50f7df0b-0d14-4a3e-8b17-a98979a1d002  ServicePrincipal
37182072-3c9c-4f6a-a4b3-b3f91cacffce  AzureSupportCenter                   5573a483-0503-43b3-9902-918c32755b5d  ServicePrincipal
```

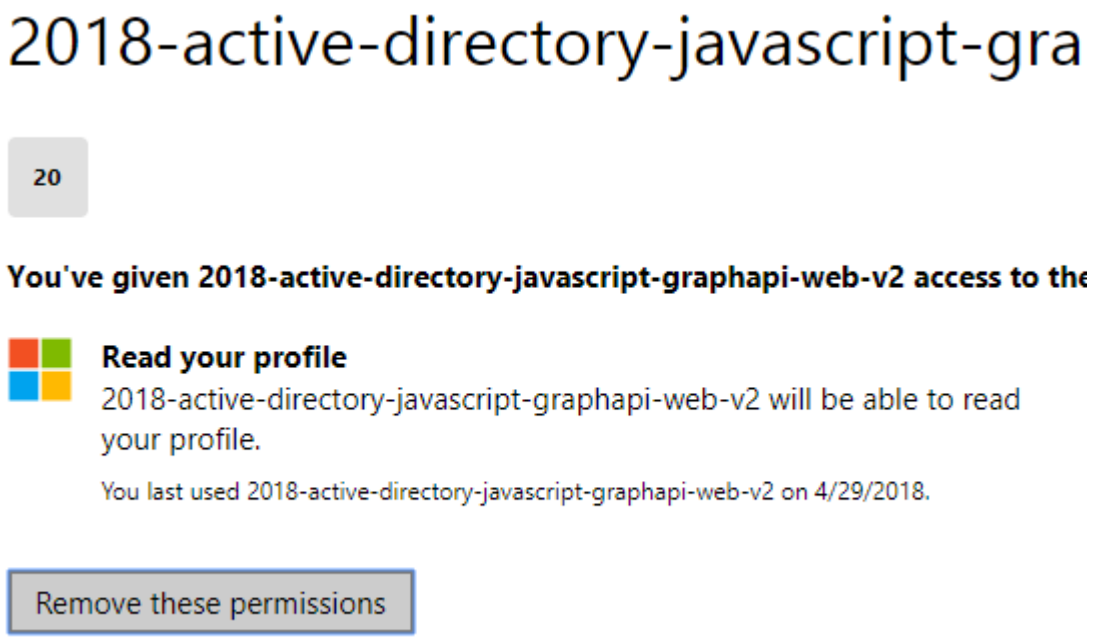
AAD, per account: https://myapps.microsoft.com/



Or: https://portal.office.com/myapps

Windows Account (personal): https://account.live.com/consent/Manage





</Digression – Consent and Apps – how to revoke>

Main libraries - summary

ADAL.JS

npm install --save adal-angular (single package for pure JS and angular)

ADAL .NET v3, ADAL for Python, OWIN, .NET Core, JWT

Links:

https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-authentication-libraries

https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-v2-libraries

Glossary

.NET Core – OpenID Events

```
// After "authorization code" is redeemed for tokens at the token endpoint.
public Func<TokenResponseReceivedContext, Task> OnTokenResponseReceived
```

```
// When a request is received on the RemoteSignOutPath.
public Func<RemoteSignOutContext, Task> OnRemoteSignOut
```

```
// Before redirecting to the identity provider to sign out.
public Func<RedirectContext, Task> OnRedirectToIdentityProviderForSignOut
```

```
// Invoked before redirecting to the identity provider to authenticate. Used to set persisted
ProtocolMessage.State. Can customize parameters sent to the identity provider.
public Func<RedirectContext, Task> OnRedirectToIdentityProvider
```

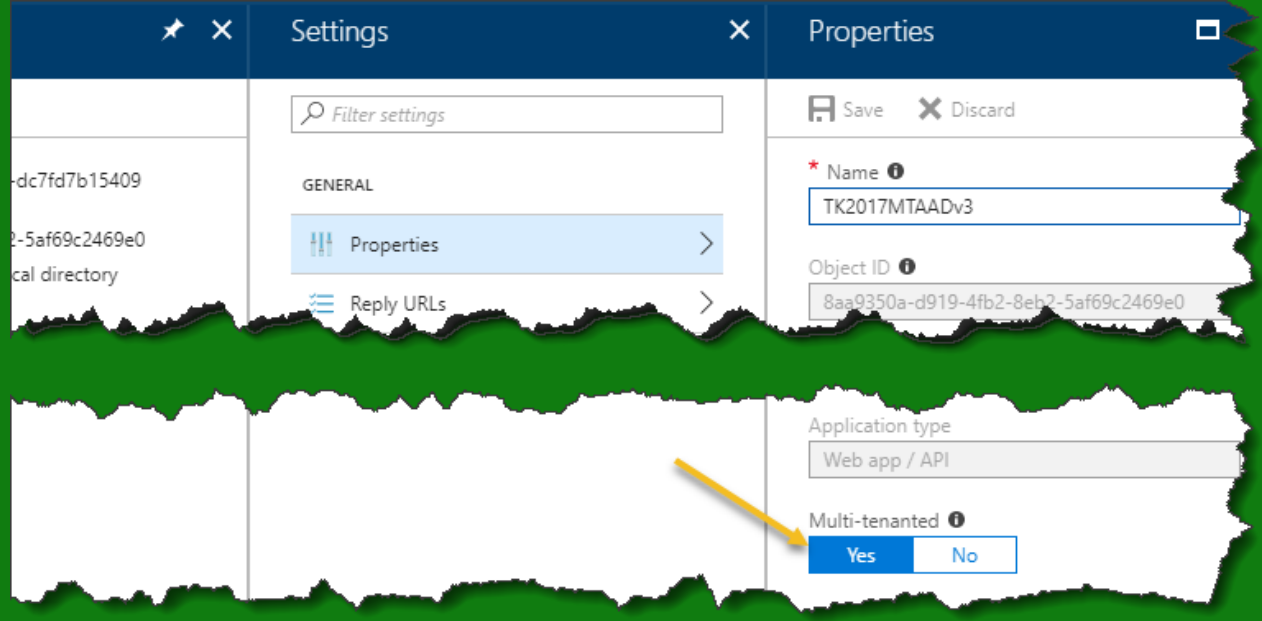
```
//  When a protocol message is first received.
public Func<MessageReceivedContext, Task> OnMessageReceived
```

```
// After security token validation if authorization code is present
public Func<AuthorizationCodeReceivedContext, Task> OnAuthorizationCodeReceived
```

```
// If exceptions are thrown. Will be re-thrown after this event unless suppressed.
public Func<AuthenticationFailedContext, Task> OnAuthenticationFailed
```

```
// Invoked when an IdToken has been validated and produced an AuthenticationTicket.
public Func<TokenValidatedContext, Task> OnTokenValidated
```

```
// Invoked when user information is retrieved from the UserInfoEndpoint.
public Func<UserInformationReceivedContext, Task> OnUserInformationReceived
```



.NET Core – mutlitenant?

Add VALIDATION to code (and registration in case of V1)

Onboarding

Tasks

Get Consent

Validate tenant / user – paid / free

(optional – consent for whole tenant – admin)

Flow – MTController

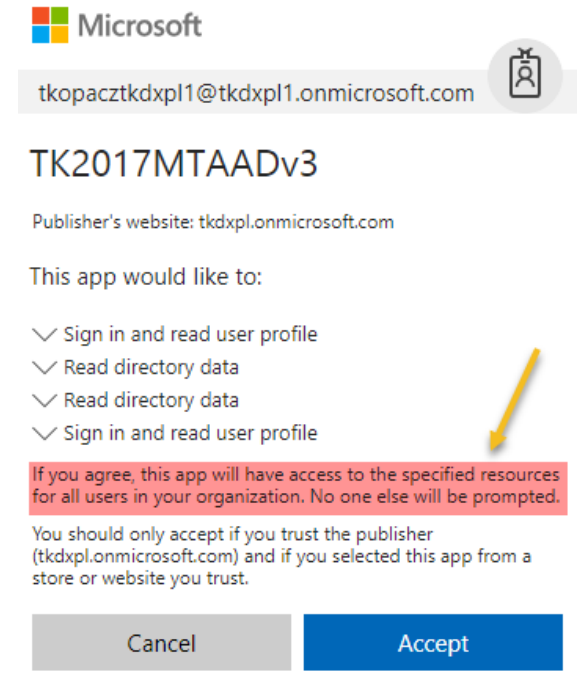
Form: **GroupGuid** (to validate which user in tenant is allowed to use app) / IsAdmin

SignUp (Post)

- Add to DB secret
- Redirect to

https://login.microsoftonline.com/common/oauth2/authorize?response_type=code&client_id=3a2eaceb-1ff3-49a4-9156-dc7fd7b15409&resource=https%3A%2F%2Fgraph.microsoft.com&redirect_uri=https%3A%2F%2Flocalhost%3A44349%2FMT%2FProcessCode&state=5521d92b-2187-4114-b9c7-e9a7581eccf2&prompt=admin_conse

- to get consent




```
string strRedirectUri = this.Request.Scheme +
    "://" + Request.Host + "/MT/ProcessCode";
string authorizationRequest =
    $"https://login.microsoftonline.com/common/oauth2/authorize?" +
    $"response_type=code" +
    $"&client_id={Uri.EscapeDataString(m_aadOptions.ClientId)}" +
    $"&resource={Uri.EscapeDataString("https://graph.microsoft.com")}" +
    $"&redirect_uri={Uri.EscapeDataString(strRedirectUri)}" +
    $"&state={Uri.EscapeDataString(stateMarker)}";
if (admin) authorizationRequest += $"&prompt=admin_consent";
return authorizationRequest;
```

Onboarding – Process Code

We have an auth code

Model:

- code
- error
- error_description
- resource
- State

So we can request Token And Call Graph API to get more information (here – only tenant!)

```
//Find Tenant based on secret
var t = m_db.Tenants.FirstOrDefault(p => p.Secret == model.state);
if (t!=null)
{
    var authContext = new AuthenticationContext($"{m_aadOptions.AzureAdInstance}"); //MT
    var creds = new ClientCredential(m_aadOptions.ClientId, m_aadOptions.ClientSecret);
    var redirectUri = new Uri($"{m_aadOptions.Domain}/MT/ProcessCode");
    //Get Tenant
    var authResult = await authContext.AcquireTokenByAuthorizationCodeAsync(
        model.code, redirectUri, creds,
        "https://graph.microsoft.com/");

    //Do we already registered that tenant?
    var tenantID = authResult.TenantId.ToLower();
    if (m_db.Tenants.FirstOrDefault(p => p.TenantGuid == tenantID) == null) {
        t.TenantGuid = tenantID;
    } else {
        //m_db.Tenants.Remove(t);
    }
    m_db.SaveChanges();
} //Else - wrong secret
```

Demo

MT – onboarding process

Login (Authentication & Authorization)

Authorization

Get all tenants from DB and for each tenant read group guid (to validate user)

AddAuthorization, AddPolicy with all guids – allowed users

(use [Authorize(Policy = "AdminPolicyByGuid")] for controller / WebAPI)

Authentication

o.ResponseType = "code id_token"; //We need identity and code to call API

OnTokenValidated

Check if tenant is allowed (DB)

context.SecurityToken.Claims.FirstOrDefault(p => p.Type == "tid").Value.ToLower();

And/Or check if user is allowed

context.SecurityToken.Claims.FirstOrDefault(p => p.Type == "upn").Value.ToLower();

OnAuthorizationCodeReceived

Get auth token based on code – to call Graph API and get more information (if needed)

This sample: names of groups (**need ADMIN consent!**)

Modify context.Principal.Identity (ClaimsIdentity) if needed!

OnTokenResponseReceived – get/cache access token

Demo

MT – login process & policies

<Digression>

If app is not Multitenat

Use – guest account / external user in AAD where app is registered

What about “services” (something “unattended”)

~~**NO USERNAME / PASSWORD IN APP!!!**~~

Ways to do that

(SPN – service account; usually for Azure Resources)

Use “App” permission

AuthenticationContext, **ClientCredential(clientId, appKey)**

Device Flow (device code) <- recommended!

Run in USER context | Multitenant

IETF Draft for OAuth Device Flow

https://login.microsoftonline.com/common/oauth2/deviceauth

Steps

Post to device code endpoint | 2. Wait for login (any device!) | 2. Pool to get response (bearer)

API (Microsoft.IdentityModel.Clients.ActiveDirectory):

AuthenticationContext for tenant / common

Check TokenCache

AcquireTokenSilentAsync | if fail -> AcquireDeviceCodeAsync

Manually: https://joonasw.net/view/device-code-flow

Demo

Unattended processes
App permission
DeviceCode
ADAL + Azure + SPN

OK, complex API
Web -> Web API -> Web API

First example – Passing only token_id (user)

Use V2(multitenant), OWIN, classic .NET

Same AppID, different Redirect URLs (limitation of v2)

Call:

Get: userObjectID, tenantID, authority

Get: ClientCredential (based on Secret)

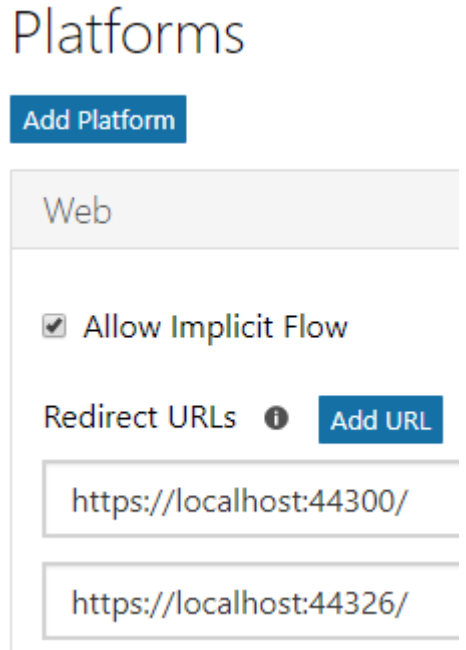
Get ConfidentialClientApplication (token cache) + AcquireToken(Silent)Async

Add bearer token to header

In Web API

Authentication: UseOAuthBearerAuthentication, JwtFormat, OpenIdConnectCachingSecurityTokenProvider

Use normal Claims



API – passing only USER ID (v2)

Second example – Passing on-behalf-of

(By the way – multitenant as well).

WebApp registration

add knownClientApplications and user_impersonation

NaiveSessionCache (in ASP.NET Core Session)

Call:

AuthenticationContext, ClientCredential(ClienID, ClientSecret),
AcquireTokenSilentAsync for URI / resourceID

Pass bearer token

WebAPI (to call GraphAPI)

Get token (from HttpContext)

Find userName – from Claims, UPN or Email

UserAssertion(token, "urn:ietf:params:oauth:grant-type:jwt-bearer", userName);

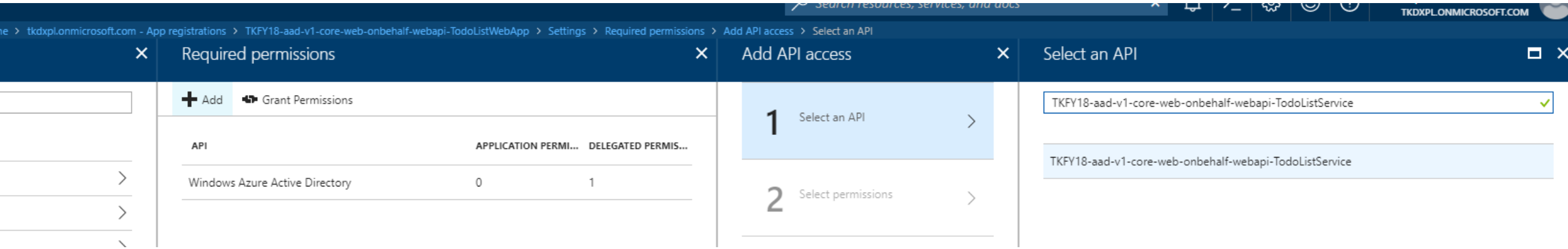
ClientCredential(ClientId, ClientSecret)

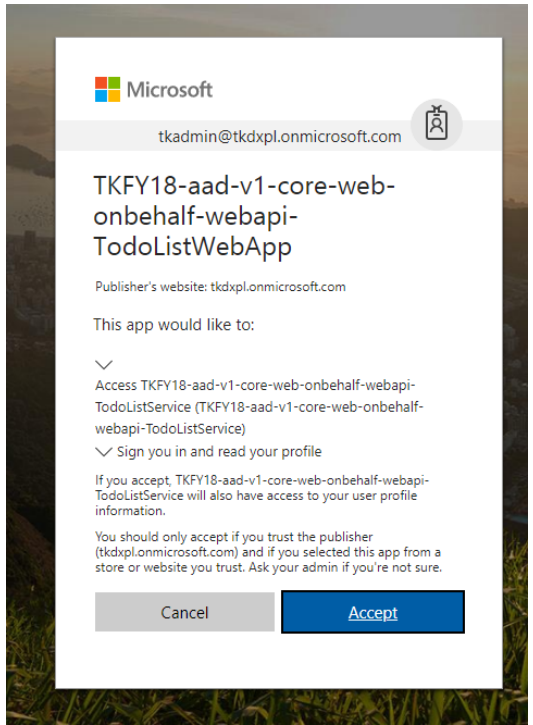
AuthenticationContext (for authority, $"{Instance}{TenantId}")

AcquireTokenAsync for URI / resourceID based on clientCredential, userAssertion);

Pass bearer token to Graph API & call API

API – AAD, registration





API – Flow on behalf (v1)

Short: Authorization (.NET Core)

Key concept – full (GOOD!!!) docs here

AuthorizeAttribute | AllowAnonymousAttribute

Policy

IAuthorizationRequirement (set of data parameters to authorize)
AuthorizationHandler (implement policy check – based on data)
Usage:

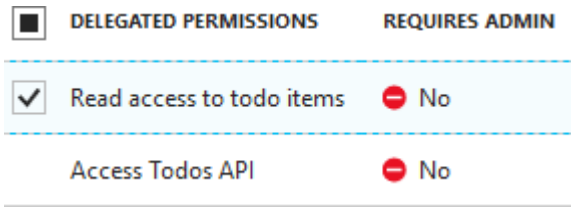
```
    services.AddAuthorization(options =>
    {
        options.AddPolicy("AtLeast21", policy =>
            policy.Requirements.Add(new MinimumAgeRequirement(21)));
    });

    services.AddSingleton<IAuthorizationHandler, MinimumAgeHandler>();
}
```

Custom Permission / Scopes

```
{
  "oauth2Permissions":[
    {
      "adminConsentDescription": "Allow access to read all users' todo items.",
      "adminConsentDisplayName": "Read access to todo items",
      "id": "43dc1069-125f-4aac-b554-7a837e049ed1",
      "isEnabled": true,
      "type": "User",
      "userConsentDescription": "Allow access to read your todo items.",
      "userConsentDisplayName": "Read access to your todo items",
      "value": "Todo.Read"
    }
  ]
}
```

JWT Token

```
{
  "appid":
  "family_name":
  "given_name":
  "name":
  "scp": "Todo.Read"
}
```



https://joonasw.net/view/defining-permissions-and-roles-in-aad

TailSpin Survey (see resources at the end)

Azure B2C (short!)

What is Azure B2C – for B2C Apps!

Scenarios

Enable a customer to sign up to use your registered application
Enable a signed-up customer to sign in and start using your application
Enable a signed-up customer to edit their profile
Enable multi-factor authentication in your application
Enable the customer to sign up and sign in with specific identity providers
Grant access from your application to APIs that you build
Customize the look and feel of the sign-up and sign-in experience
Manage single sign-on sessions for your application

Policies

Build-in, custom (Identity Experience Framework). What field do we need, additional query params, ….

Branding: Fully customized UI (HTML)

SSO to Azure AD

Fast Demo – MT, AAD B2C

Final remarks

Multitenant and code – important questions!

Dependency Injection?

- Singleton should depend on tenant / user / …
- AutoFac

Caching?

- Per tenant!

Static variables?

- Banned – unable to control!

Custom Business Logic?

- .NET: Roslyn, dynamically loaded assemblies, …

UI (branding) …

(Demo)

Roslyn and custom logic

Resources

V1: https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-developers-guide

V2: https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-appmodel-v2-overview

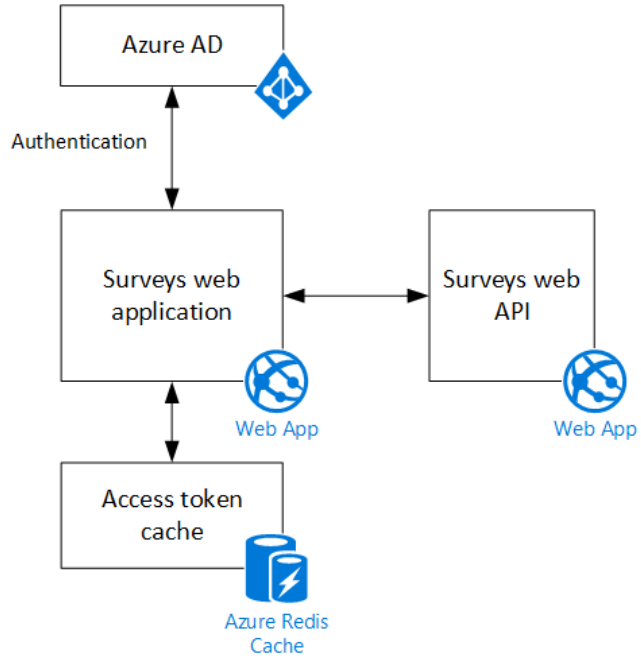
Architecture:

https://docs.microsoft.com/en-us/azure/architecture/

https://docs.microsoft.com/en-us/azure/architecture/multitenant-identity/
(old, but..)

Source code: https://github.com/mspnp/multitenant-saas-guidance
(TailSpin survey, V1, .NET)

- OpenID connect
- Redis as a token cache
- Web + WebAPI
- Run: run-the-app



END
Questions:
tkopacz@microsoft.com
(or during „lunch”/after etc.)