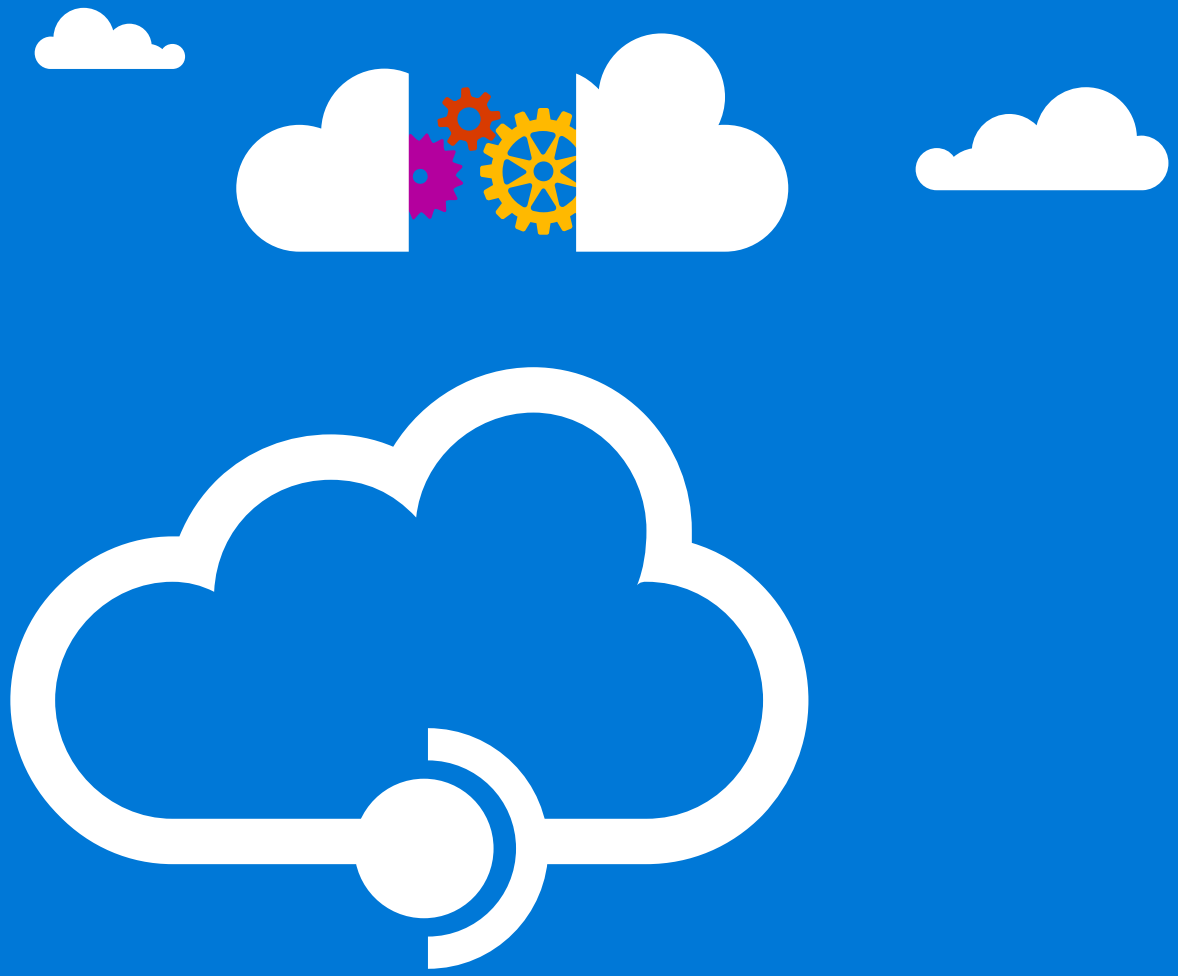
Azure Active Directory Application Security in action - Securing a Multi-Tier App using AAD



Ben Roscorla

Technical Evangelist

Microsoft UK

Microsoft

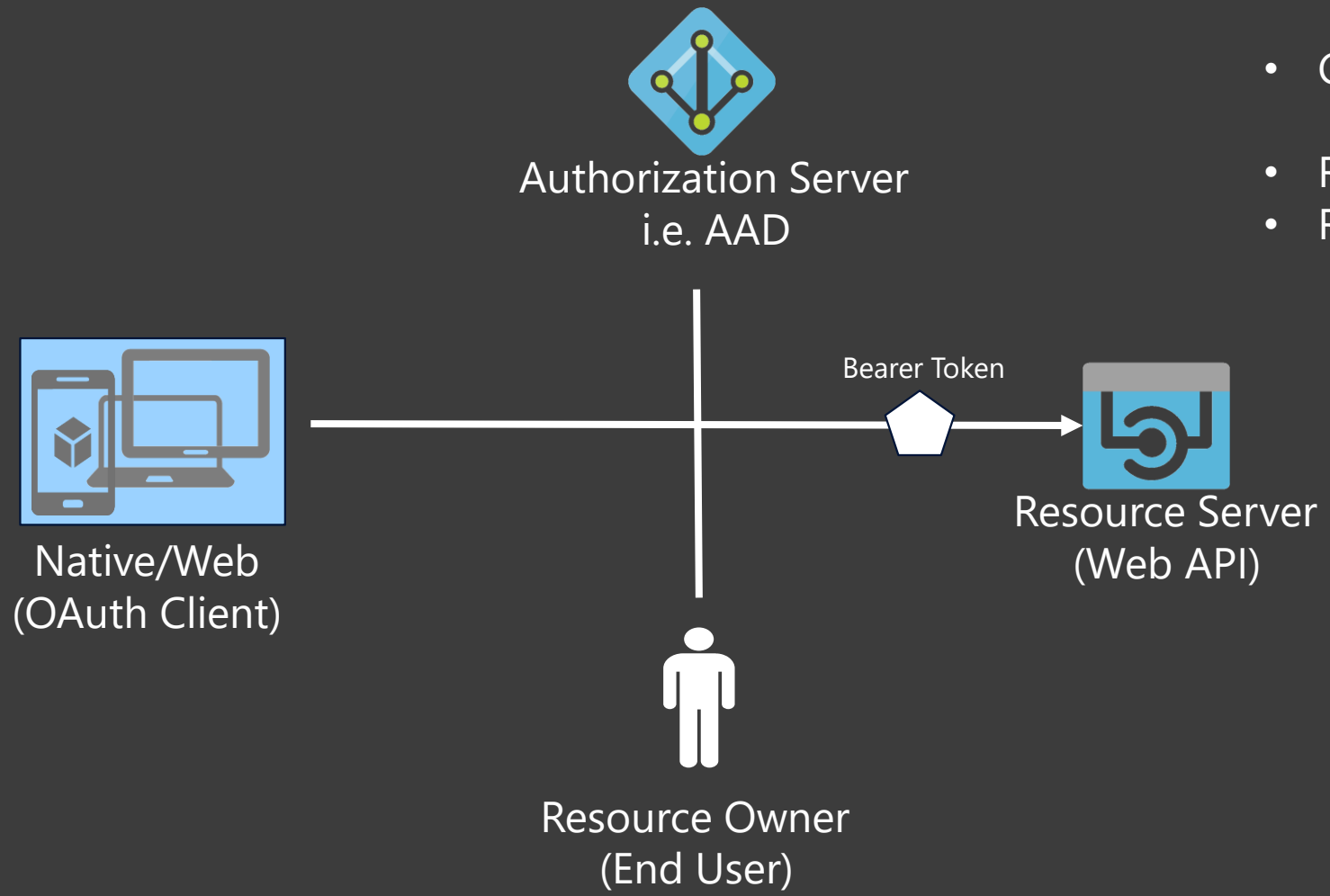
Azure AD

Azure AD B2C

Azure AD

Azure AD B2B

Terminology



- Client – Means Client Device – Not User
  - OAuth has Public and Confidential clients
- Resource Owner = User
- Resource Server is where the data resides

AAD OAuth Flows

Authorization Code	The quintessential OAuth grant
Implicit	The bad boy of OAuth - Recommend for SPA - Browser-based (JavaScript)
Client Credentials	Run as a Service - Client (not user!)
Resource Owner Password Credentials	A bit like a service account - username and password
Refresh Token	Get Token without re-authentication
JWT Bearer	‘on behalf of’ – multiple hops between services

4 Steps for Implementing Solution Azure AD

Design

Register in AAD

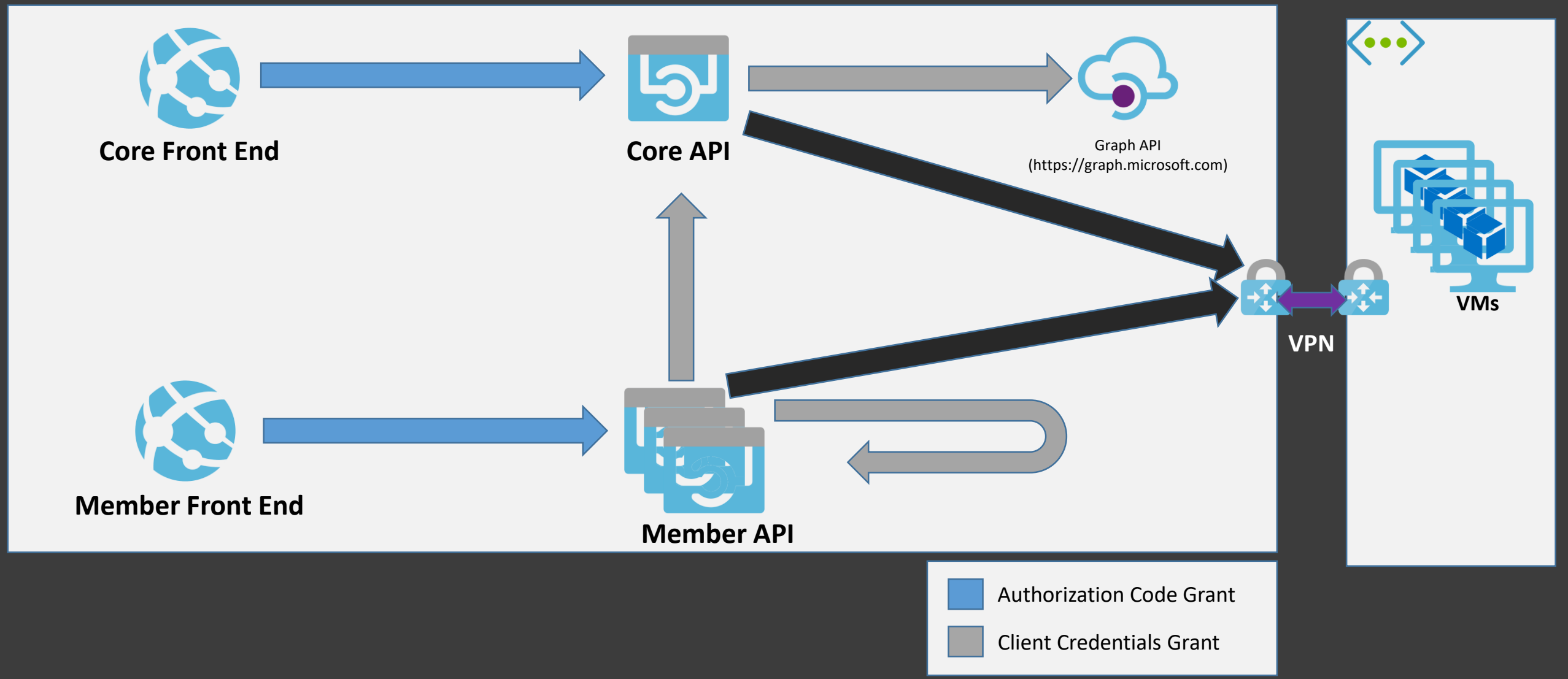
Implement Code

Consent

Initial AAD Setup

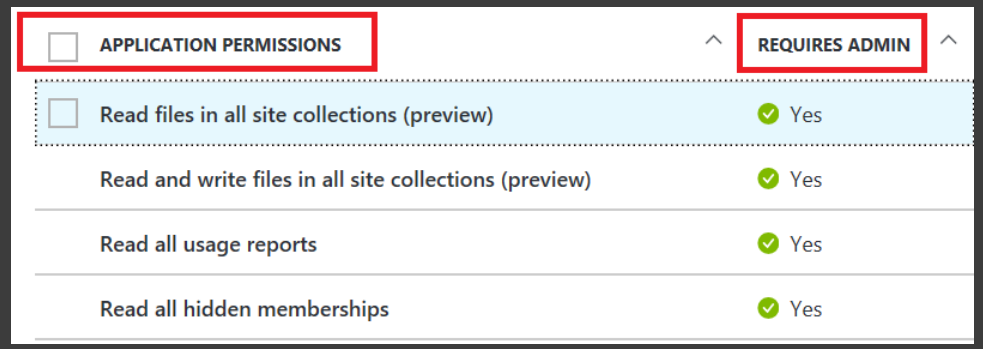
Azure AD

Design

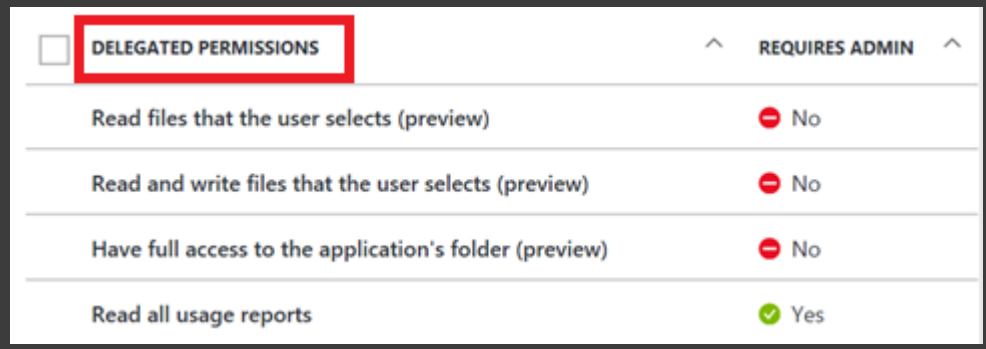


Register in AAD

- Specify the App Type – Web or Native
- Sign on URL – Reply URL and Home Page URL
- Retrieve the Client ID
- If an API – note APP ID
- Set up Keys/Secrets for Confidential clients
- Set up Permissions



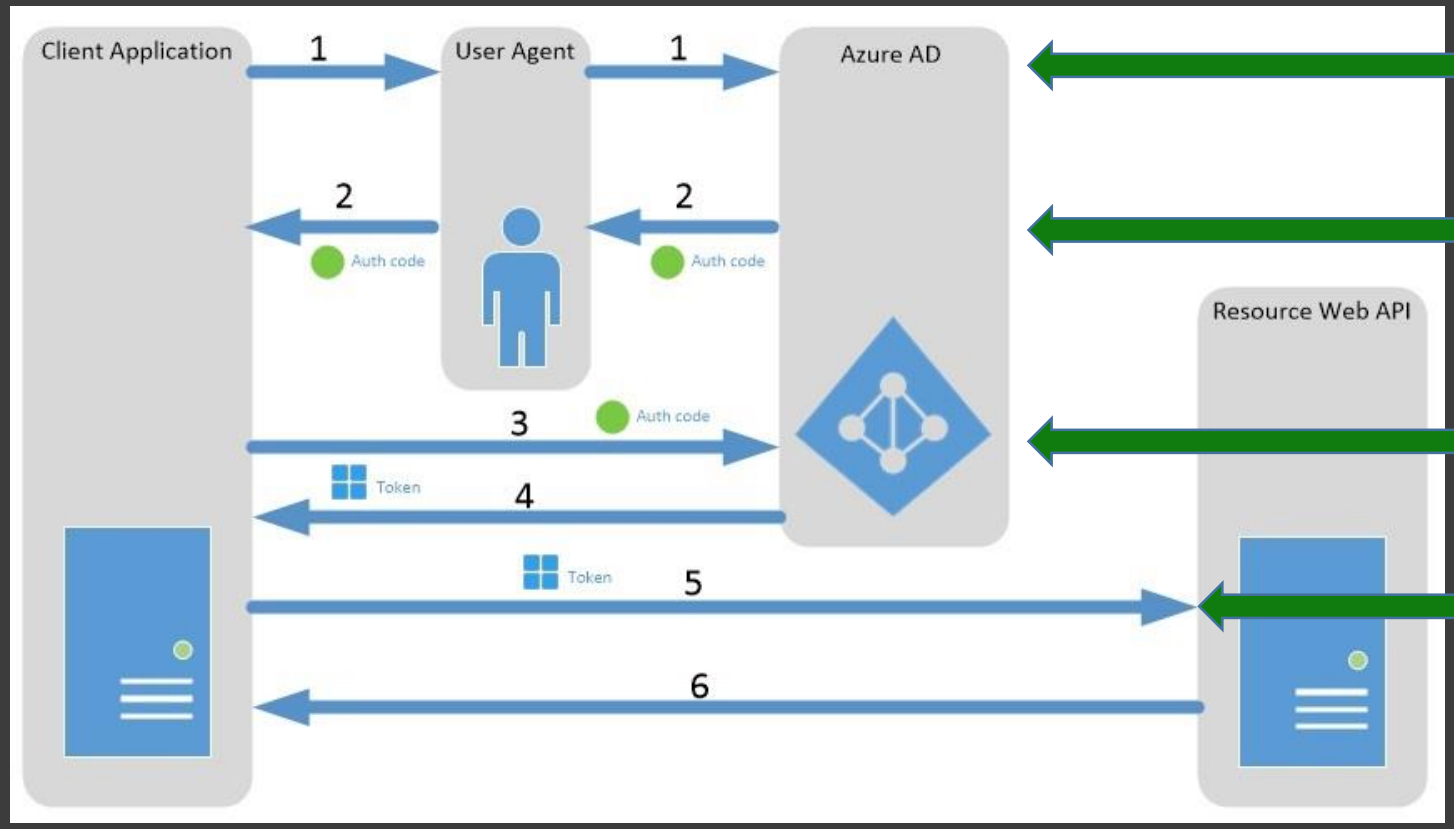
APPLICATION PERMISSIONS	REQUIRES ADMIN
Read files in all site collections (preview)	Yes
Read and write files in all site collections (preview)	Yes
Read all usage reports	Yes
Read all hidden memberships	Yes



DELEGATED PERMISSIONS	REQUIRES ADMIN
Read files that the user selects (preview)	No
Read and write files that the user selects (preview)	No
Have full access to the application's folder (preview)	No
Read all usage reports	Yes

You could hand roll….

Implement Code



`https://login.microsoftonline.com/{Tenant}/oauth2/authorize`

`Redirect URL`

`https://login.microsoftonline.com/{Tenant}/oauth2/token`

`Add Bearer to header for API call`

You could use our APIs….

Implement Code

Web Site

OWIN
UseOpenIdConnectAuthentication
Function/Class Decorators
[AllowAnonymous]
[Authorize]
ClaimsPrincipal

ADAL
AcquireTokenByAuthorizationCodeAsync
AcquireTokenSilentAsync

Web API

OWIN
UseWindowsAzureActiveDirectoryBearerAuthentication
Function/Class Decorators
[AllowAnonymous]
[Authorize]
ClaimsPrincipal

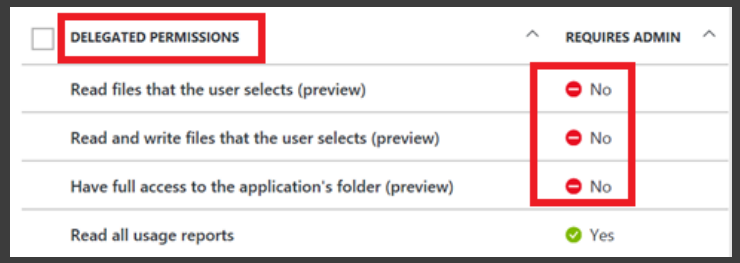
ADAL
AcquireTokenAsync
Client Credentials
User Assertion

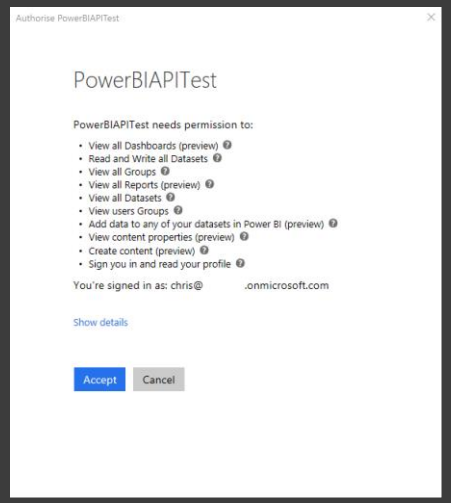
Native

ADAL
AcquireTokenAsync

Consent

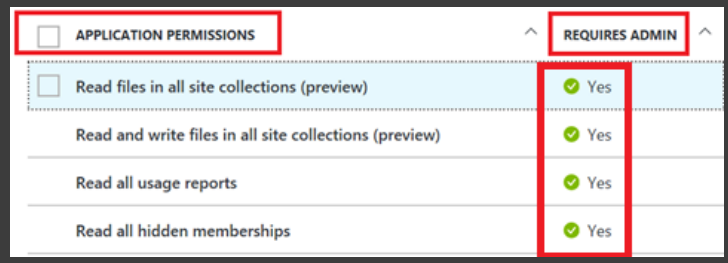
User

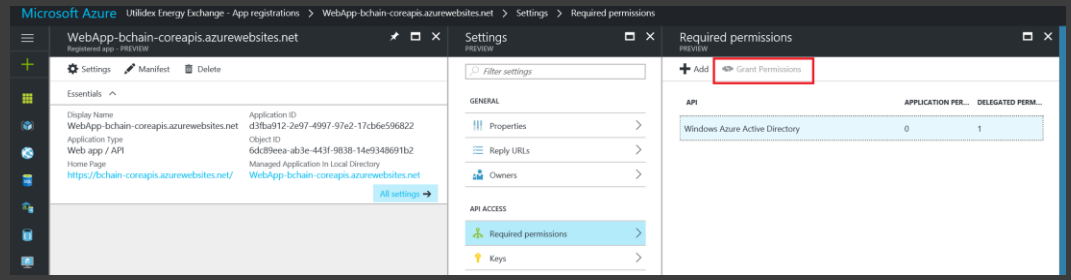




[Remove: Azure AD Access Panel]

Admin



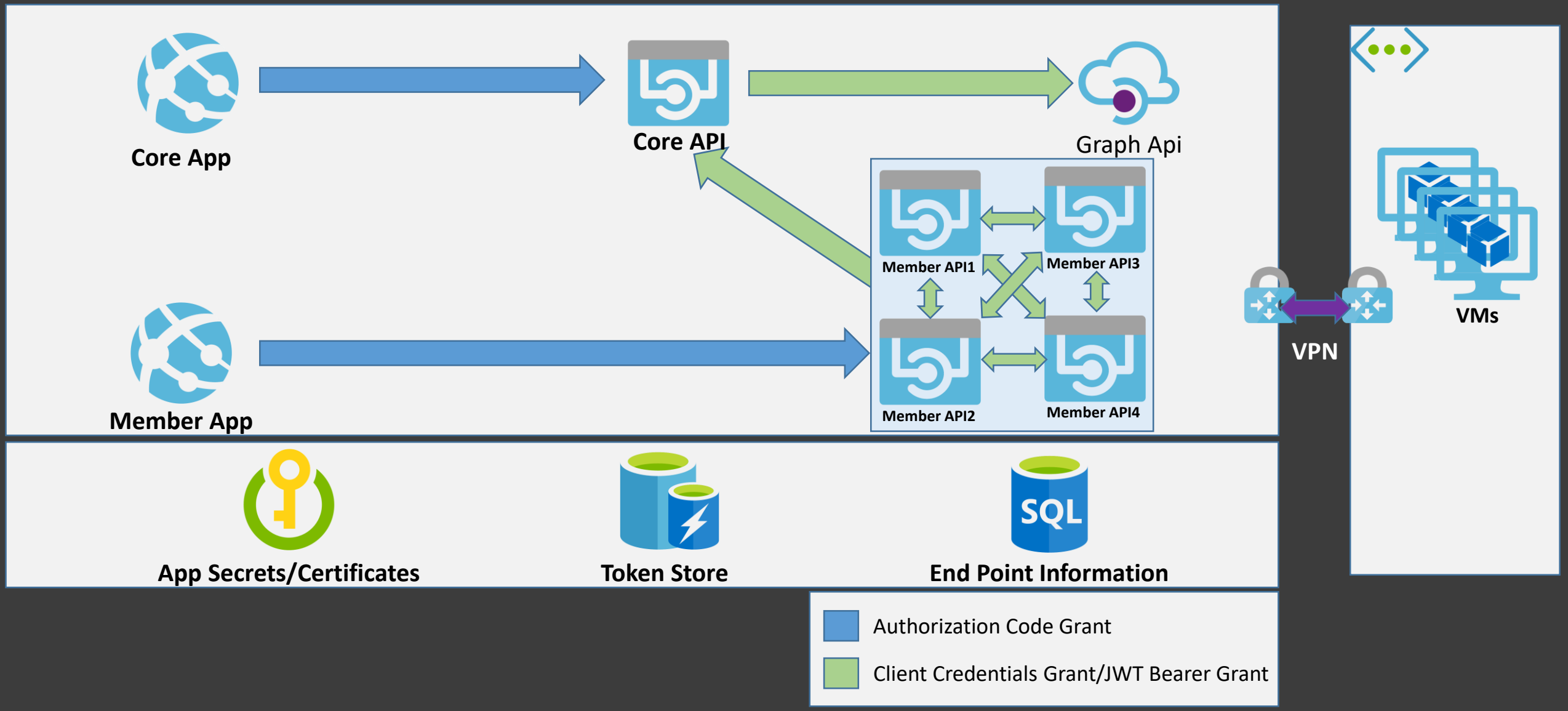


You can remove this consent by deleting the Application itself or the [Service Principal] which contains the consent information using a PowerShell command - [Remove-MsolServicePrincipal]

Final Setup

Azure AD

Design



Tips

- Easier to test locally
  - But remember to register your publish ReplyToUrls
- Always work with SSL turned on
- Double check the Reply URLs and other Registration properties
- Remember Azure AD supports multi-resource refresh tokens (MRRT)
- Consent and re-consent