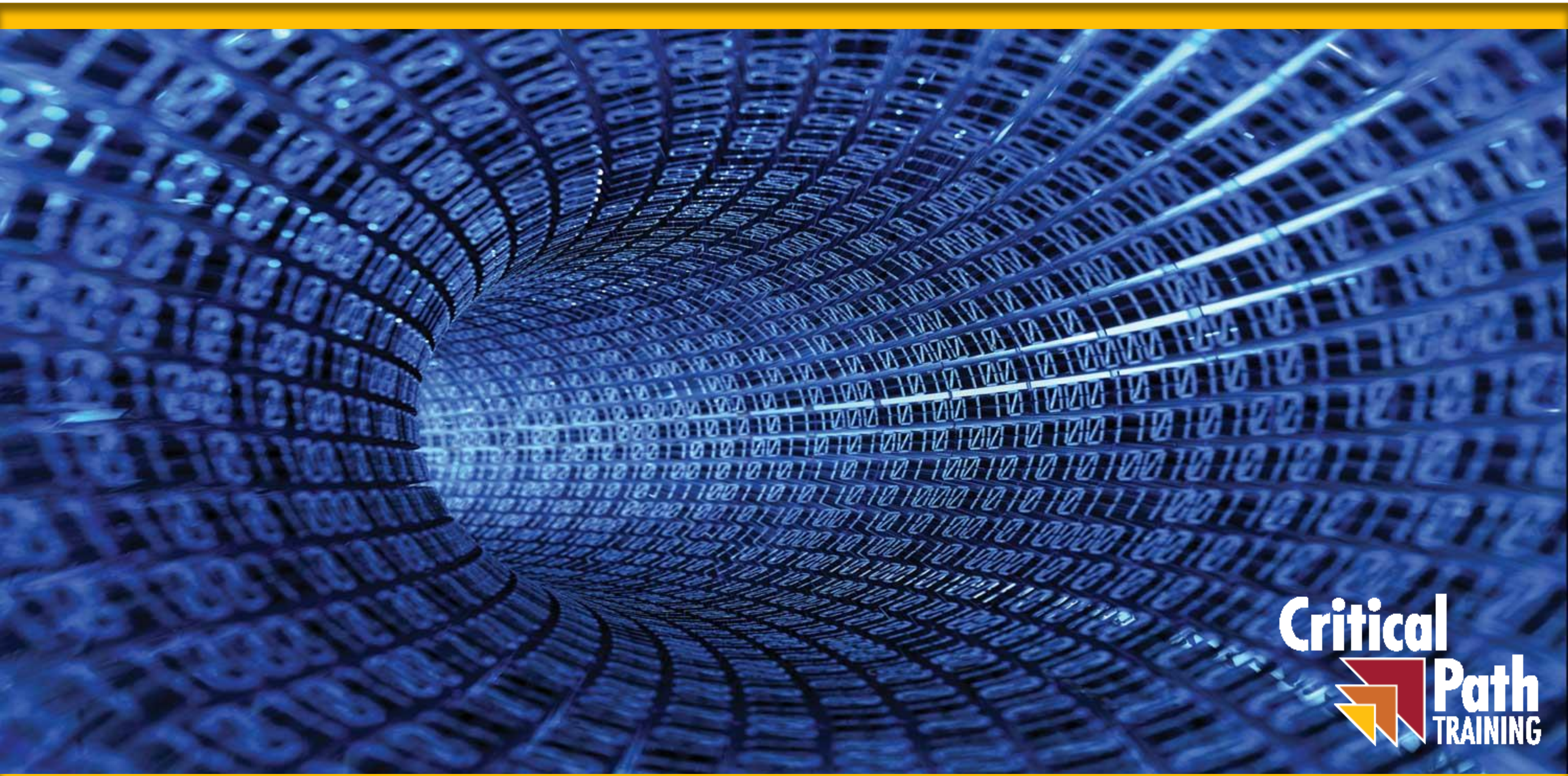
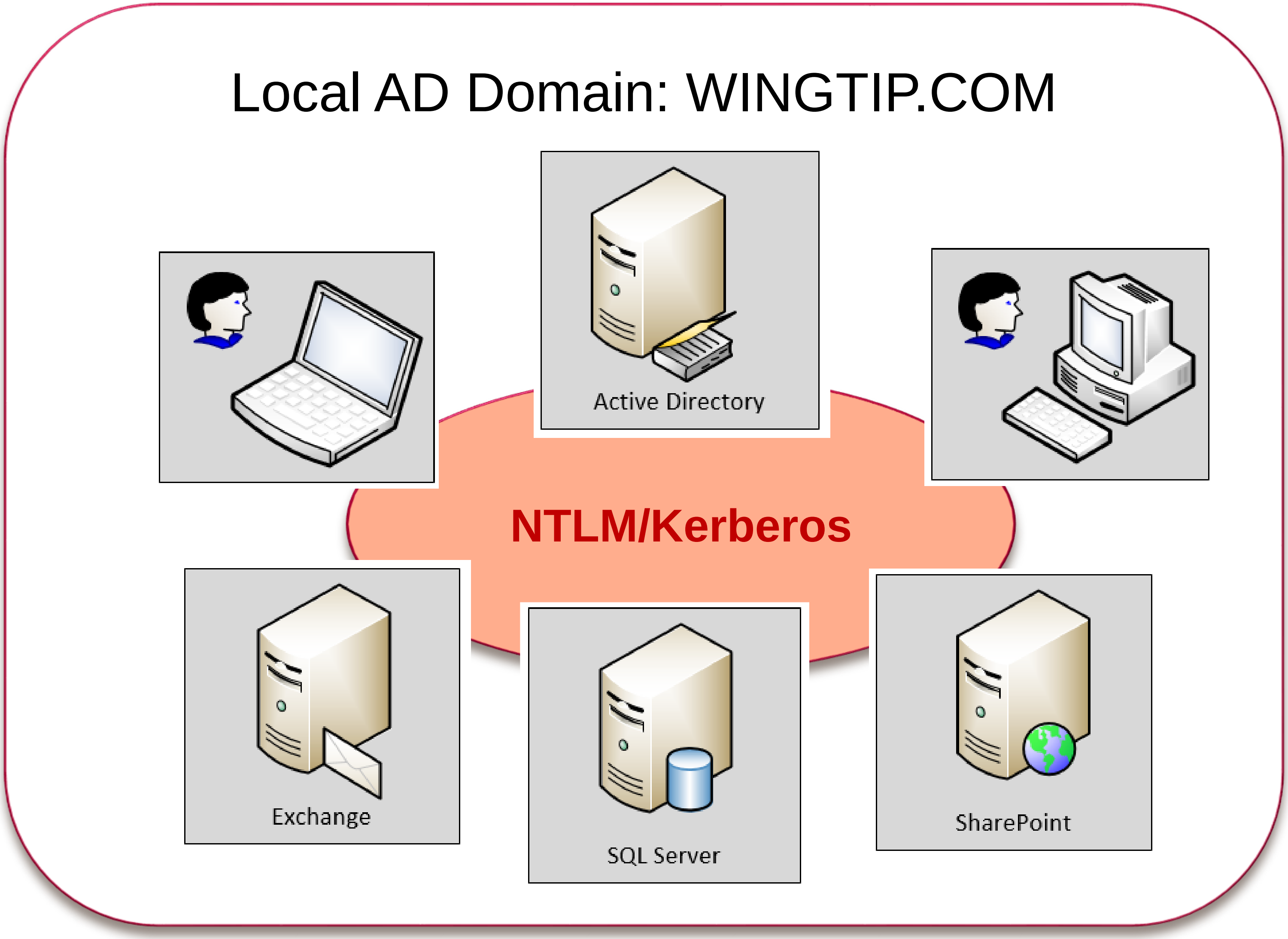
Developing Secure Applications using Azure AD



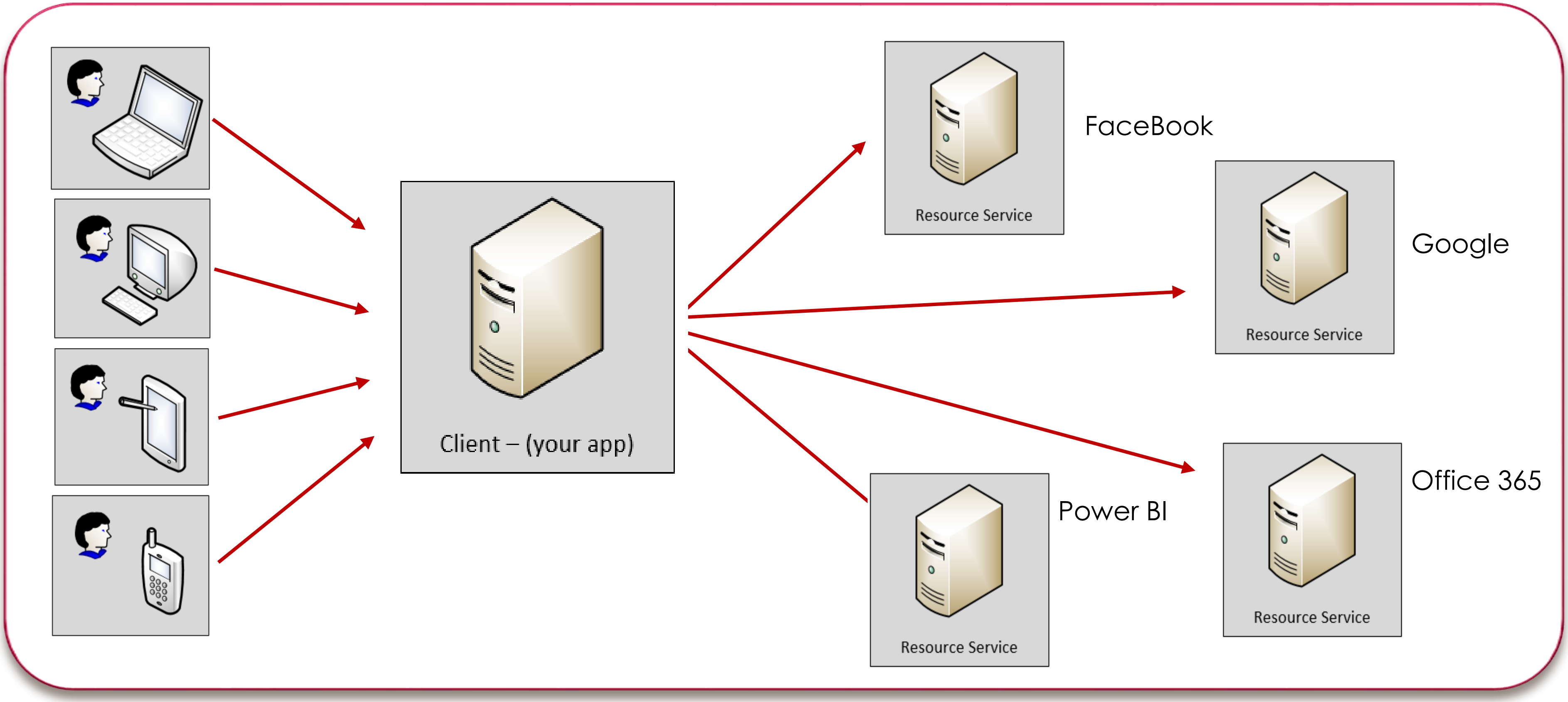
Agenda

- Understanding OAuth 2.0 and OpenID Connect
- The Role of Azure Active Directory
- Creating & Configuring Azure AD Applications
- Securing MVC Applications using ADAL and OWIN
- Securing SPAs using ADAL.js & Implicit Grant Flow

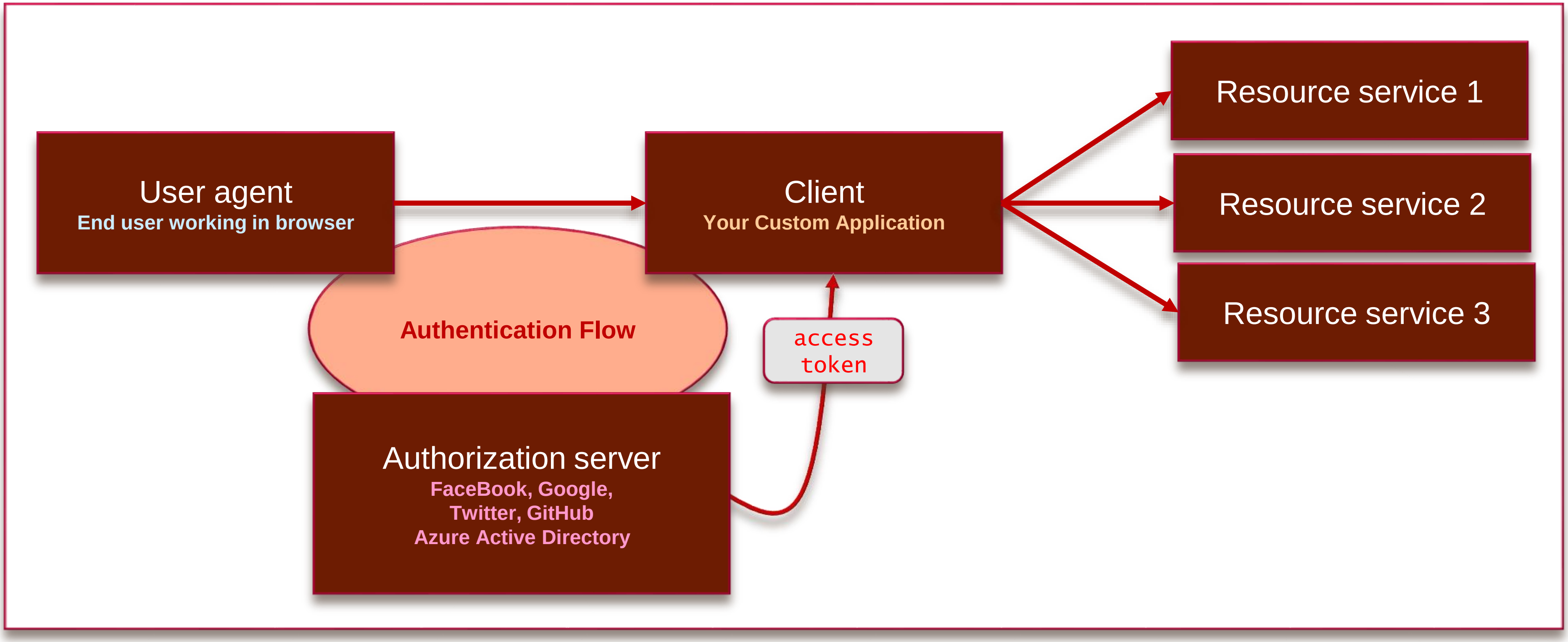
Old-school Enterprise Security



Internet Security



OAuth 2.0



User agent

End user working in browser

Client

Your Custom Application

Resource service 1

Resource service 2

Resource service 3

Authentication Flow

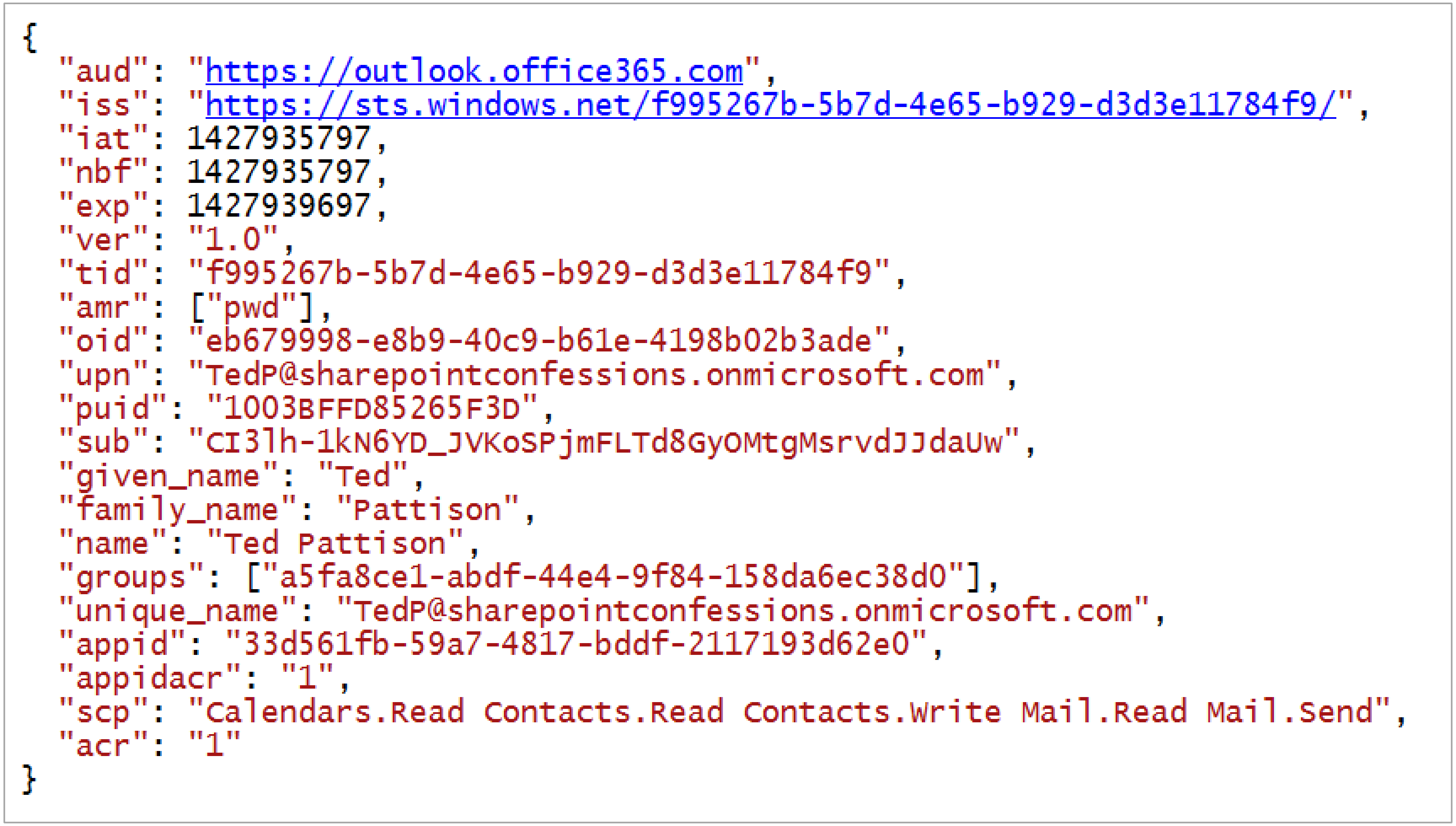
access token

Authorization server

FaceBook, Google, Twitter, GitHub

Azure Active Directory

View into an Access Token



```
{
  "aud": "https://outlook.office365.com",
  "iss": "https://sts.windows.net/f995267b-5b7d-4e65-b929-d3d3e11784f9/",
  "iat": 1427935797,
  "nbf": 1427935797,
  "exp": 1427939697,
  "ver": "1.0",
  "tid": "f995267b-5b7d-4e65-b929-d3d3e11784f9",
  "amr": ["pwd"],
  "oid": "eb679998-e8b9-40c9-b61e-4198b02b3ade",
  "upn": "TedP@sharepointconfessions.onmicrosoft.com",
  "puid": "1003BFFD85265F3D",
  "sub": "CI3lh-1kN6YD_JVKoSPjmFLTd8GyOMtgMsrvdJJdaUw",
  "given_name": "Ted",
  "family_name": "Pattison",
  "name": "Ted Pattison",
  "groups": ["a5fa8ce1-abdf-44e4-9f84-158da6ec38d0"],
  "unique_name": "TedP@sharepointconfessions.onmicrosoft.com",
  "appid": "33d561fb-59a7-4817-bddf-2117193d62e0",
  "appidacr": "1",
  "scp": "Calendars.Read Contacts.Read Contacts.Write Mail.Read Mail.Send",
  "acr": "1"
}
```

OAuth Client Registration

- Client must be registered with authorization server
  - Authorization server tracks each client with unique Client ID
  - Client should be registered with one or more Reply URLs
  - Reply URL should be fixed endpoint on Internet
  - Reply URL used to transmit security tokens to clients
  - Client registration tracks permissions and other attributes

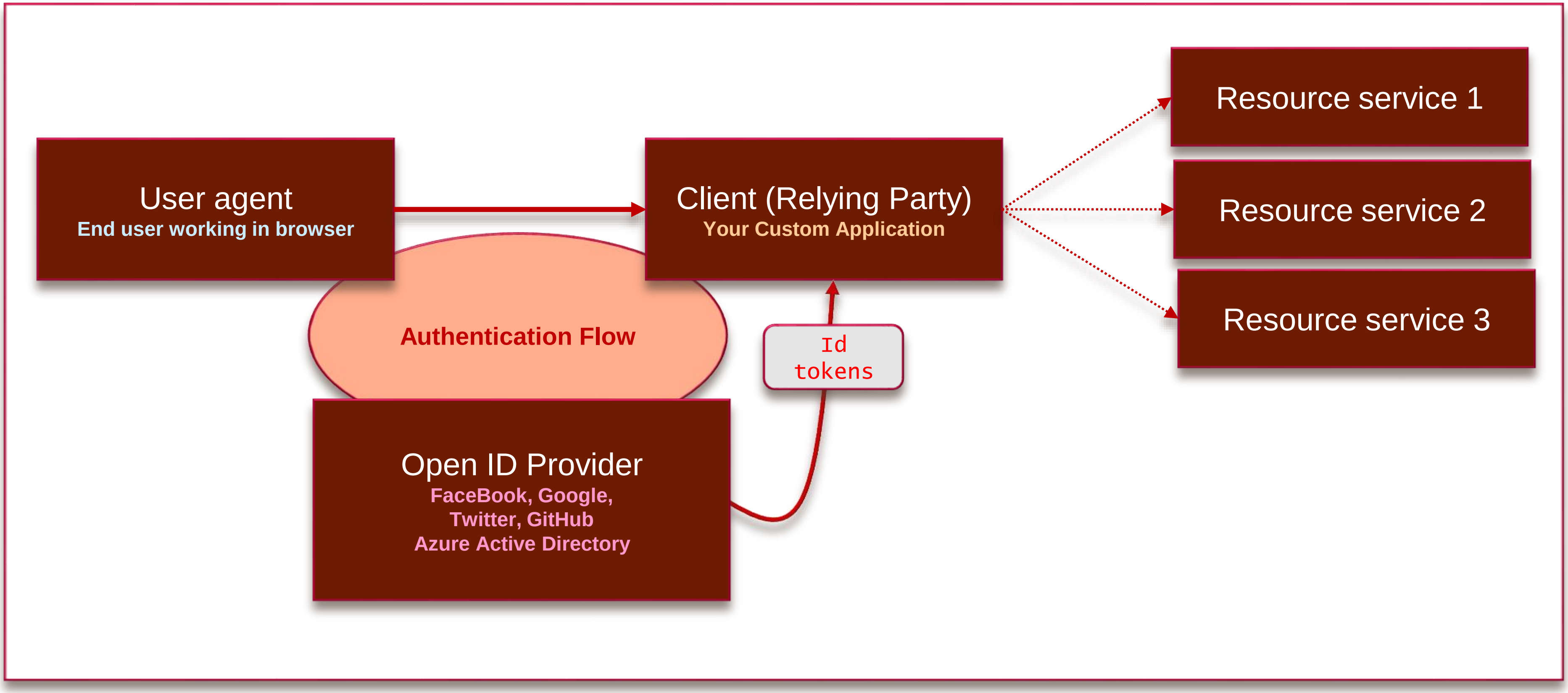
Authentication Flows

- User Credentials Flow *(public client)*
  - Used in Native clients to obtain access code
  - Requires passing user name and password
- Authorization Code Grant Flow *(confidential client)*
  - Client first obtains authorization code then access token
  - Server-side application code never sees user’s password
- Client Credentials Grant Flow *(confidential client)*
  - Authentication based on SSL certificate with public-private key pair
  - Used to obtain access token when using app-only permissions
- Implicit Grant Flow *(public client)*
  - Used in SPAs built with JavaScript and AngularJS
  - Application obtains access token w/o acquiring authorization code

OAuth 2.0 and Authentication

- OAuth 2.0 was designed for authorization
  - Creation of access token requires authentication
  - Authorization server passes access token to client
  - Client passes access token when calling resource services
  - Access token serves as app credentials for authorization
- Access token not intended for user authentication
  - Access token not designed to carry user identity data
  - OAuth 2.0 doesn't require validation of access token
  - Naïve OAuth 2.0 implementations subject to attack

Open ID Connect


User agent
End user working in browser

Client (Relying Party)
Your Custom Application

Resource service 1

Resource service 2

Resource service 3

Authentication Flow

Id tokens

Open ID Provider
FaceBook, Google,
Twitter, GitHub
Azure Active Directory

Agenda

- ✓ OAuth 2.0 and OpenID Connect
- ➢ Azure Active Directory
- Creating Azure AD applications
- Active Directory Authentication Library for .NET
- Programming Web Clients

Azure Active Directory (AAD)

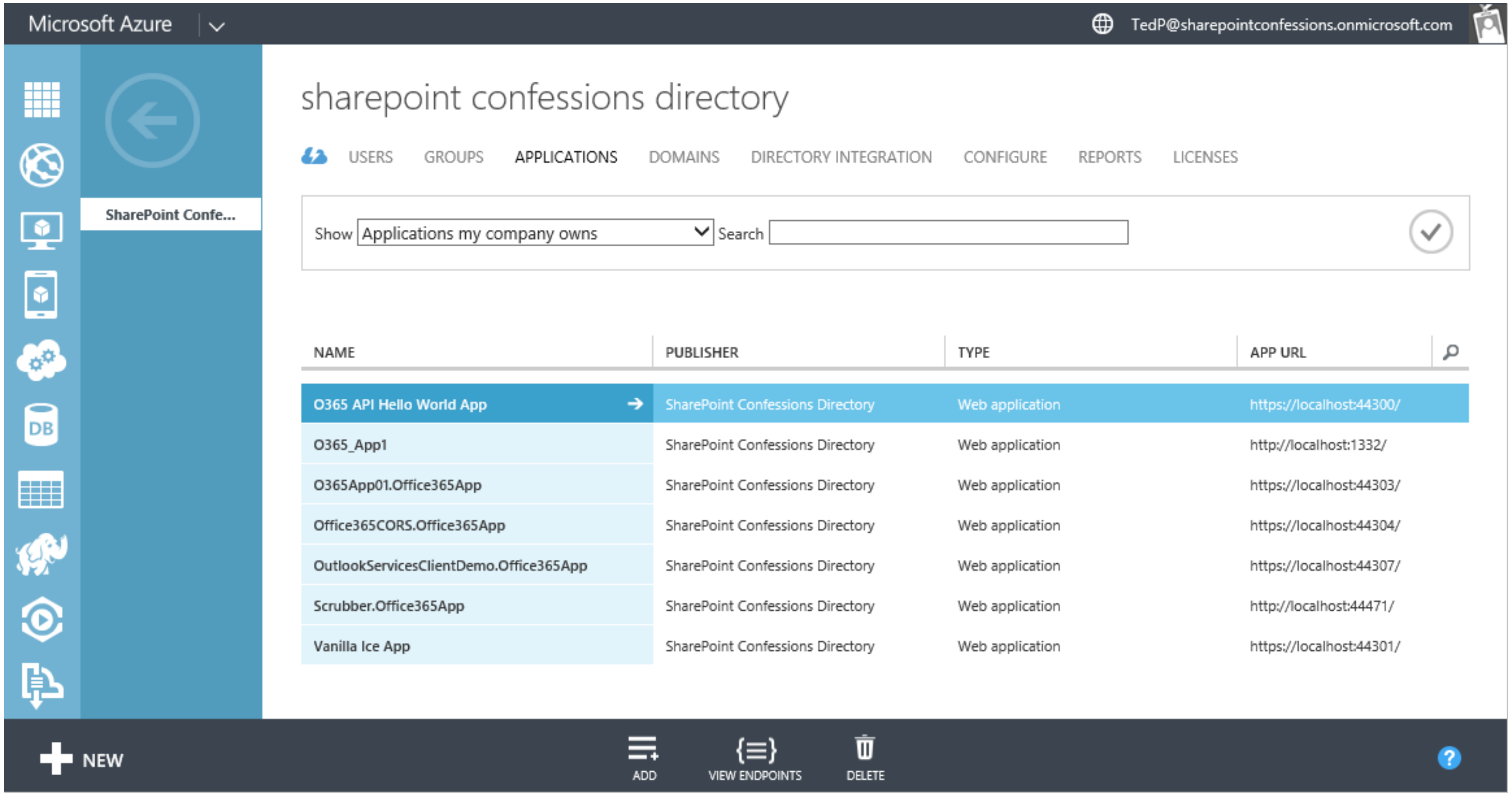
- AAD plays role of an OpenID Connect Provider
  - Creates access tokens based on OAuth 2.0
  - Creates id tokens based on OpenID Connect 1.0
- AAD provides authentication & authorization for…
  - Office 365, Exchange Online and SharePoint Online
  - Power BI REST API
  - Custom Web Applications and Web Services

Office 365 and Azure AD

- Office 365 environments are based on tenancies
  - Tenancy provides scope for creating and managing users
  - Tenancy provides a scope for site collections in SharePoint Online
- Office 365 is integrated with Azure Active Directory (AAD)
  - Each Office 365 tenancy is backed by an AAD directory
  - AAD directory can be managed using Office 365 administration
  - AAD directory can be managed using Windows Azure Portal
  - Azure support registering application within scope of AAD directory
- Application using Office 365 APIs must be registered with AAD
  - This means you must become familiar with Azure Active Directory

Azure Management Portal

- Provides management over one or more directories
  - View & configure AAD directory behind Office 365 developers site
  - Create, view and configure AAD applications during development



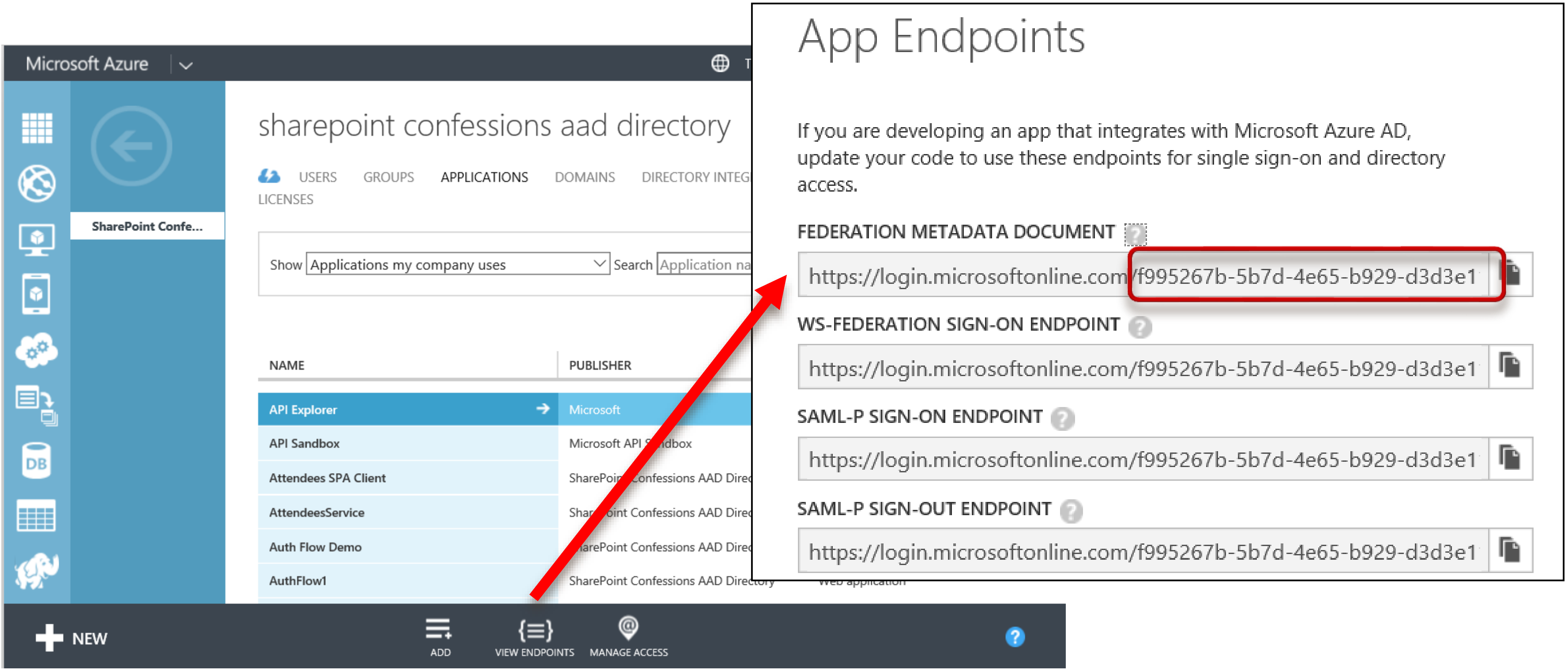
sharepoint confessions directory

USERS GROUPS APPLICATIONS DOMAINS DIRECTORY INTEGRATION CONFIGURE REPORTS LICENSES

Show: Applications my company owns Search:

NAME	PUBLISHER	TYPE	APP URL
O365 API Hello World App	SharePoint Confessions Directory	Web application	https://localhost:44300/
O365_App1	SharePoint Confessions Directory	Web application	http://localhost:1332/
O365App01.Office365App	SharePoint Confessions Directory	Web application	https://localhost:44303/
Office365CORS.Office365App	SharePoint Confessions Directory	Web application	https://localhost:44304/
OutlookServicesClientDemo.Office365App	SharePoint Confessions Directory	Web application	https://localhost:44307/
Scrubber.Office365App	SharePoint Confessions Directory	Web application	http://localhost:44471/
Vanilla Ice App	SharePoint Confessions Directory	Web application	https://localhost:44301/

Discovering Your Tenancy ID



sharepoint confessions aad directory

Show Applications my company uses Search Application na

NAME	PUBLISHER
API Explorer	Microsoft
API Sandbox	Microsoft API Sandbox
Attendees SPA Client	SharePoint Confessions AAD Direc
AttendeesService	SharePoint Confessions AAD Direc
Auth Flow Demo	SharePoint Confessions AAD Direc
AuthFlow1	SharePoint Confessions AAD Directory

App Endpoints

If you are developing an app that integrates with Microsoft Azure AD, update your code to use these endpoints for single sign-on and directory access.

FEDERATION METADATA DOCUMENT

https://login.microsoftonline.com/f995267b-5b7d-4e65-b929-d3d3e1

WS-FEDERATION SIGN-ON ENDPOINT

https://login.microsoftonline.com/f995267b-5b7d-4e65-b929-d3d3e1

SAML-P SIGN-ON ENDPOINT

https://login.microsoftonline.com/f995267b-5b7d-4e65-b929-d3d3e1

SAML-P SIGN-OUT ENDPOINT

https://login.microsoftonline.com/f995267b-5b7d-4e65-b929-d3d3e1

Agenda

✓ OAuth 2.0 and OpenID Connect

✓ Azure Active Directory

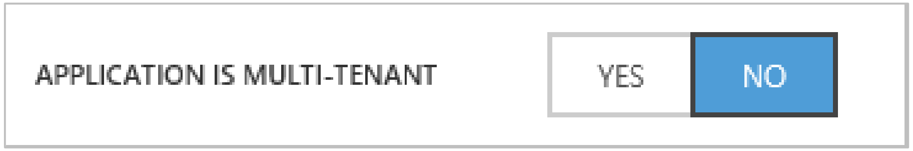
➢ Creating Azure AD applications

- Active Directory Authentication Library for .NET
- Programming Web Clients

Azure AD Applications

- Azure AD application configured with properties
  - Name
  - Sign-on URL
  - Logo
  - Single Tenant vs. Multi-tenant
  - Client ID
  - Keys (serves as password)
  - App ID URI
  - Reply URL
  - Application Permissions
  - Delegated Permissions

Single versus Multi-tenant

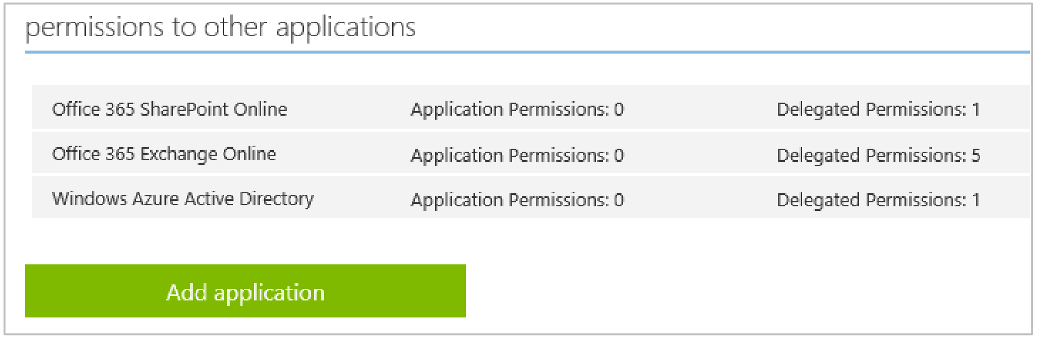


APPLICATION IS MULTI-TENANT YES [✓ NO]

- Single tenant application
  - intended for use within a single organization
  - line-of-business applications written by an Office 365 developer
  - only needs to be accessed by users in one Office 365 tenancy
  - typically registered by a developer in the organization
- Multi-tenant application
  - intended for use across many organizations
  - software-as-a-service (SaaS) applications written by ISVs
  - need to be provisioned in each directory where they will be used
  - requires user or administrator consent to register them

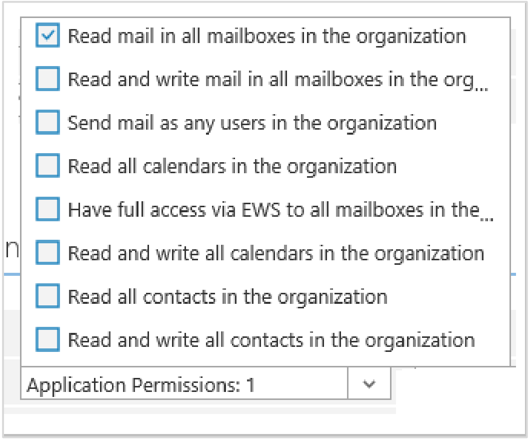
Application Permissions

- Applications can be granted permissions to other applications
  - Application permissions are app-only permissions
  - Delegated permissions are (app + user) permissions



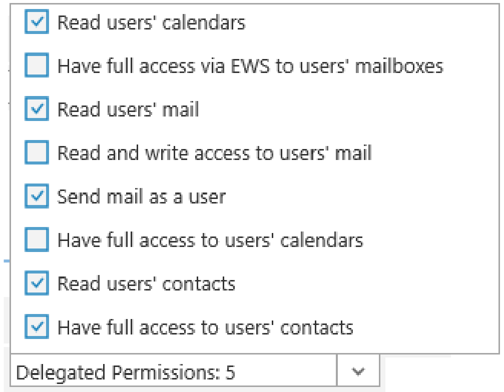
permissions to other applications

Office 365 SharePoint Online	Application Permissions: 0	Delegated Permissions: 1
Office 365 Exchange Online	Application Permissions: 0	Delegated Permissions: 5
Windows Azure Active Directory	Application Permissions: 0	Delegated Permissions: 1



- ☑ Read mail in all mailboxes in the organization
- □ Read and write mail in all mailboxes in the org...
- □ Send mail as any users in the organization
- □ Read all calendars in the organization
- □ Have full access via EWS to all mailboxes in the...
- □ Read and write all calendars in the organization
- □ Read all contacts in the organization
- □ Read and write all contacts in the organization

Application Permissions: 1



- ☑ Read users' calendars
- □ Have full access via EWS to users' mailboxes
- ☑ Read users' mail
- □ Read and write access to users' mail
- ☑ Send mail as a user
- □ Have full access to users' calendars
- ☑ Read users' contacts
- ☑ Have full access to users' contacts

Delegated Permissions: 5

DEMO

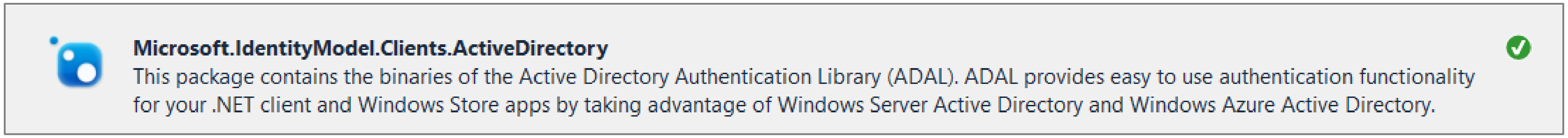
Creating an AAD Application

Agenda

- ✓ OAuth 2.0 and OpenID Connect
- ✓ Azure Active Directory
- ✓ Creating Azure AD applications
- ➢ Active Directory Authentication Library for .NET
- Programming Web Clients

ADAL for .NET

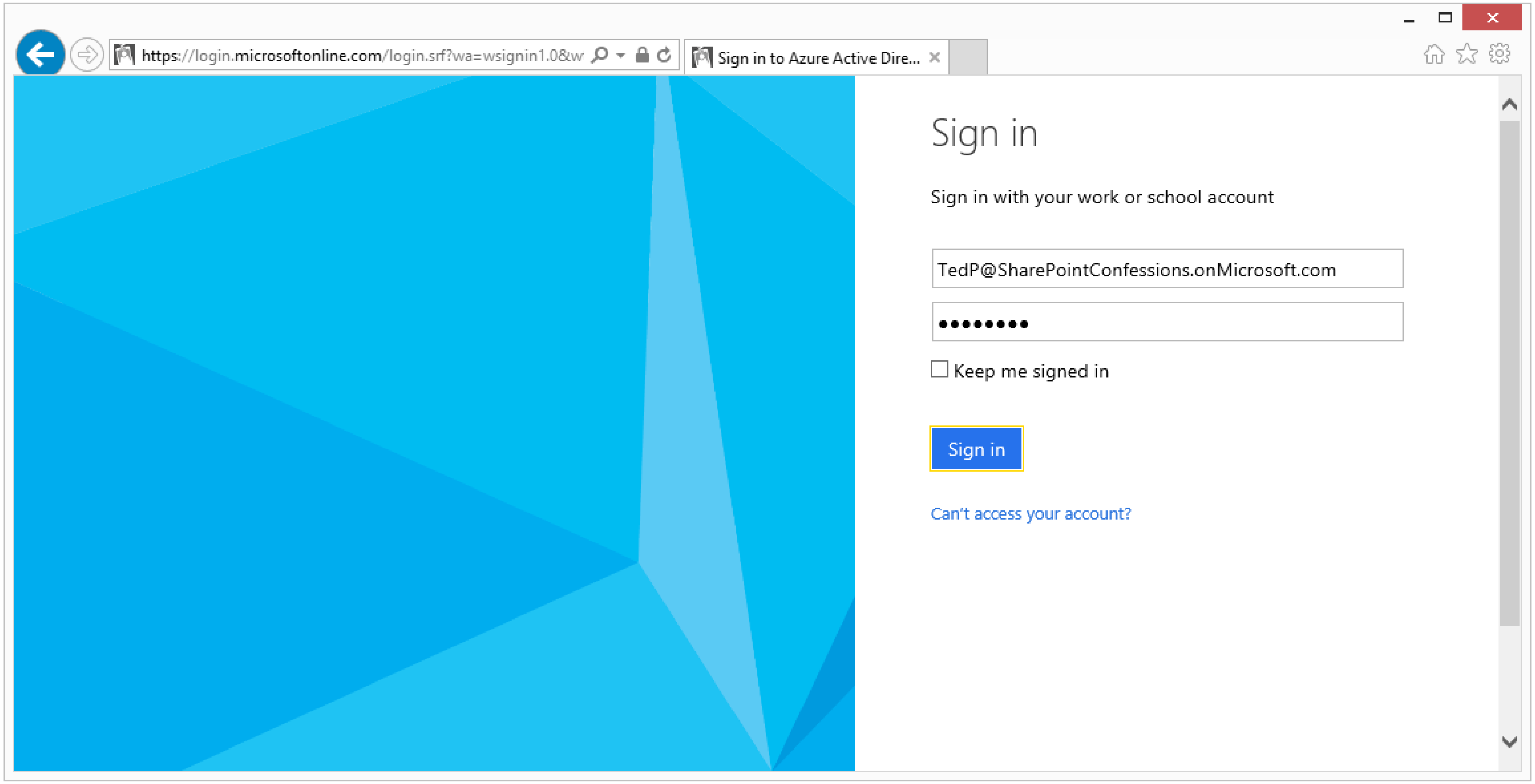
- Active Directory Authentication Library for .NET
  - Used in Native Clients and in Web Clients
  - Handles authentication flow behind the scenes
  - Provides token cache



Microsoft.IdentityModel.Clients.ActiveDirectory
This package contains the binaries of the Active Directory Authentication Library (ADAL). ADAL provides easy to use authentication functionality for your .NET client and Windows Store apps by taking advantage of Windows Server Active Directory and Windows Azure Active Directory.

- ADAL .NET installs as a NuGet Package
  - Version 2.x is latest stable version
  - Version 3.x is in prerelease

User Sign-in at https://login.microsoftonline.com



https://login.microsoftonline.com/login.srf?wa=wsignin1.0&w
Sign in to Azure Active Dire...

Sign in

Sign in with your work or school account

TedP@SharePointConfessions.onMicrosoft.com

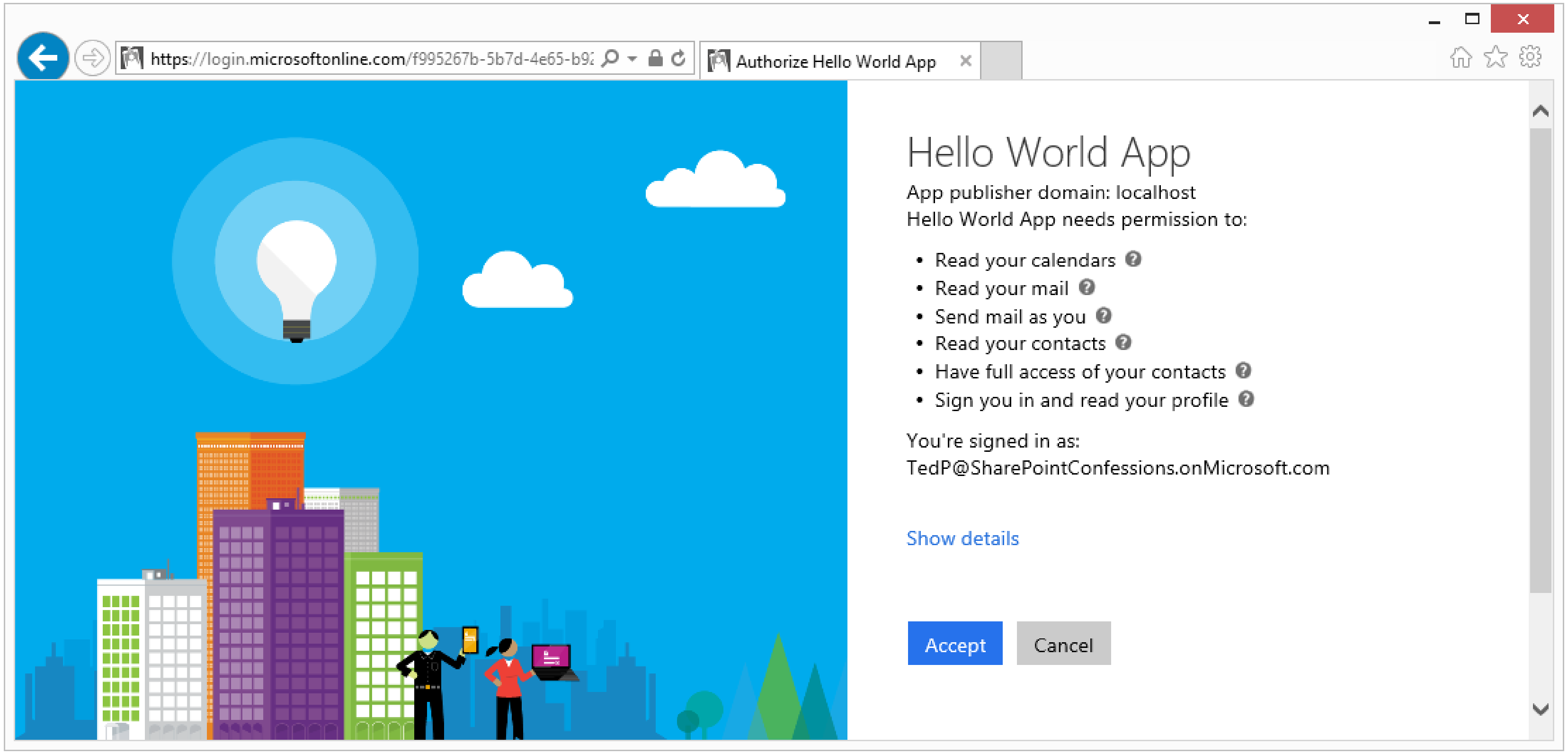
●●●●●●●●●

□ Keep me signed in

Sign in

Can't access your account?

Common Consent Experience (user)



https://login.microsoftonline.com/f995267b-5b7d-4e65-b9
Authorize Hello World App

Hello World App

App publisher domain: localhost
Hello World App needs permission to:

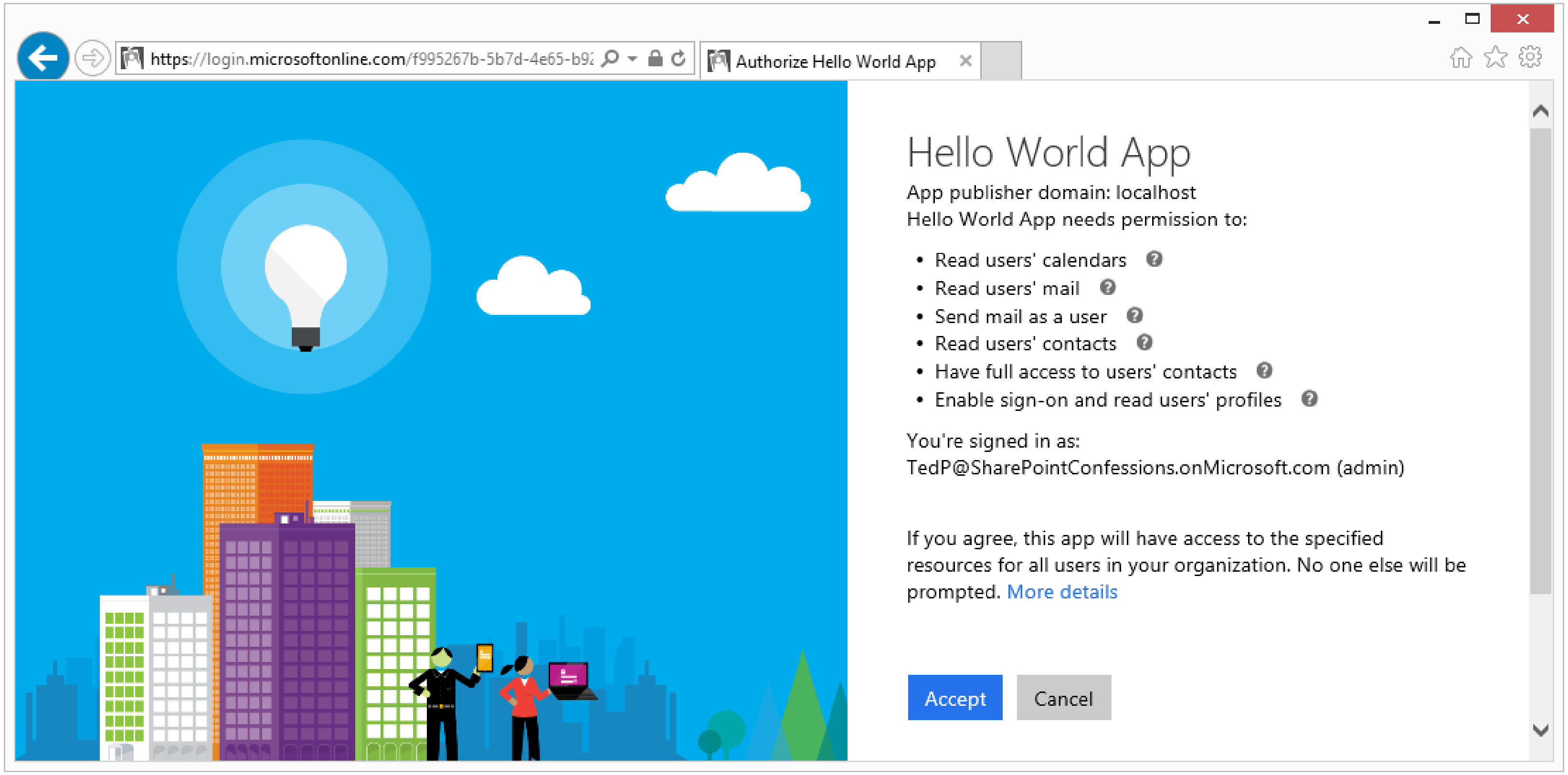
- Read your calendars
- Read your mail
- Send mail as you
- Read your contacts
- Have full access of your contacts
- Sign you in and read your profile

You're signed in as:
TedP@SharePointConfessions.onMicrosoft.com

Show details

Accept Cancel

Common Consent Experience (admin)



https://login.microsoftonline.com/f995267b-5b7d-4e65-b9:
Authorize Hello World App

Hello World App

App publisher domain: localhost
Hello World App needs permission to:

- Read users' calendars
- Read users' mail
- Send mail as a user
- Read users' contacts
- Have full access to users' contacts
- Enable sign-on and read users' profiles

You're signed in as:
TedP@SharePointConfessions.onMicrosoft.com (admin)

If you agree, this app will have access to the specified resources for all users in your organization. No one else will be prompted. More details

DEMO

Using ADAL in a Native Client

Agenda

- ✓ OAuth 2.0 and OpenID Connect
- ✓ Azure Active Directory
- ✓ Creating Azure AD applications
- ✓ Active Directory Authentication Library for .NET
- ➢ Programming Web Clients

Authentication Code Flow

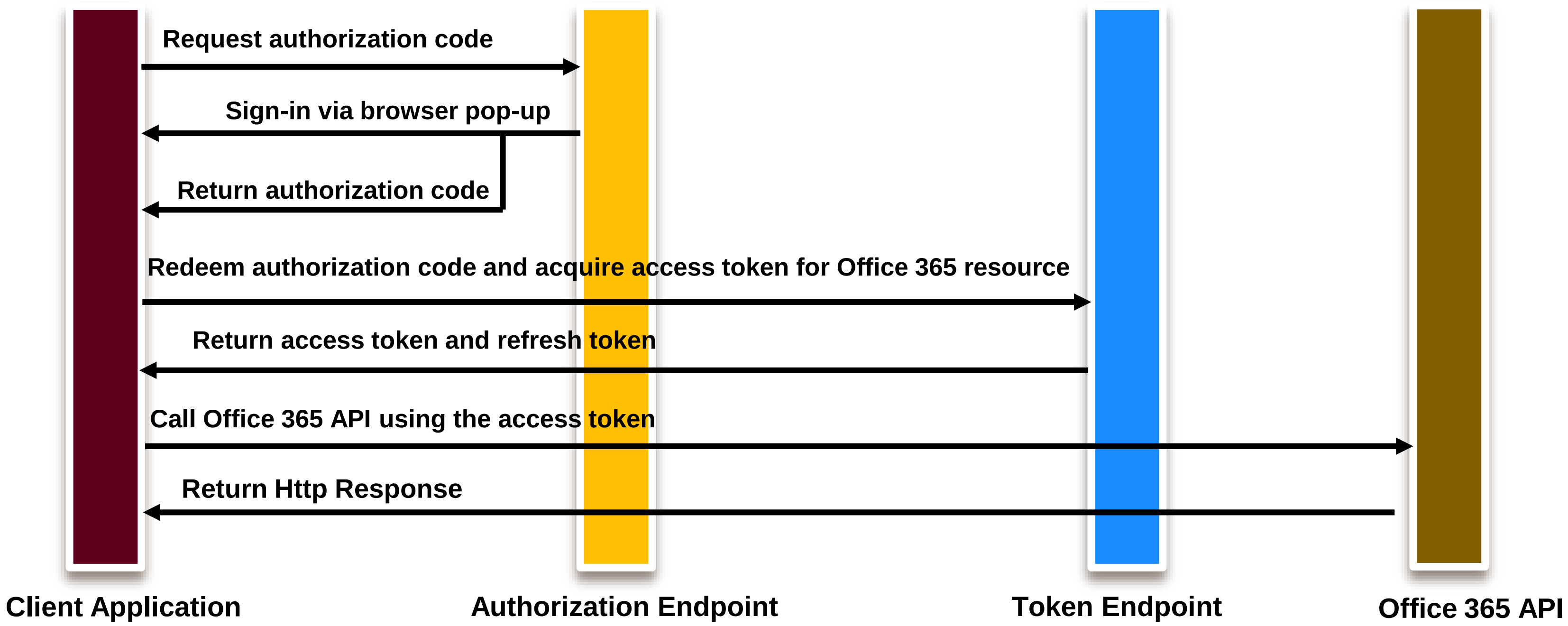
- Provides Highest Levels of Security
  - User credentials never seen by client
  - Access token passed to client with Reply URL
  - Access token not passed through user agent
- Refresh tokens used to get new access tokens
  - Access token lifetime is about 1 hour
  - Refresh token lifetime is 14 days
  - AAD supports multi-resource refresh tokens (MRRTs)

Authorization Code Grant Flow Example

- **Sign-on URL**
  - Development: **https://localhost:44300/**
  - Production: **https://www.MyDomain.com/**
- **Reply URL**
  - Development: **https://localhost:44300/AcceptDirect**
  - Production: **https://www.MyDomain.com/AcceptDirect**
- **Application ID URI**
  - String-based identifier for an application – *not a retrievable URL*
  - **https://sharepointconfessions.onmicrosoft.com/HelloWorldApp**
- **Client ID**
  - GUID-based identifier for a specific AAD application
  - **33d561fb-59a7-4817-bddf-2117193d62e0**
- **Key** (aka Client Secret)
  - Key that acts as a secret password between Azure AD and application
  - **ouWdhd2LxDl0Pcu2SKlujEiQ5GmSbKRbBM24nETb5dw=**

Authorization Code Grant Flow

- Sequence of Requests in Authorization Code Grant Flow
  - Application redirects to AAD authorization endpoint
  - User prompted to log on at Windows logon page
  - User prompted to consent to permissions (first access)
  - AAD redirects to application with authorization code
  - Application redirects to AAD access token endpoint


Request authorization code
Sign-in via browser pop-up
Return authorization code
Redeem authorization code and acquire access token for Office 365 resource
Return access token and refresh token
Call Office 365 API using the access token
Return Http Response
Client Application
Authorization Endpoint
Token Endpoint
Office 365 API

DEMO

Using ADAL in a Web Client

Summary of OAuth Client Types

	Web Client SPA	Hybrid Native Client	Web Application Client	Web Service Client
Client Type	Public	Public or Confidential	Confidential	Confidential
Verifiable Reply URL	Yes	No	Yes	Yes
Authenticates Client	No	It Depends	Yes	Yes
Token from Authorization Endpoint	Yes	Yes	No	No
Access Token from URI Fragment	Yes	No	No	No
Token from Token Endpoint	No	Yes	Yes	Yes
Can use refresh tokens	No	Yes	Yes	Yes
Permissions	Delegated	Delegated + App	Delegated + App	Delegated + App

Summary

- ✓ OAuth 2.0 and OpenID Connect
- ✓ Azure Active Directory
- ✓ Creating Azure AD applications
- ✓ Active Directory Authentication Library for .NET
- ✓ Programming Web Clients