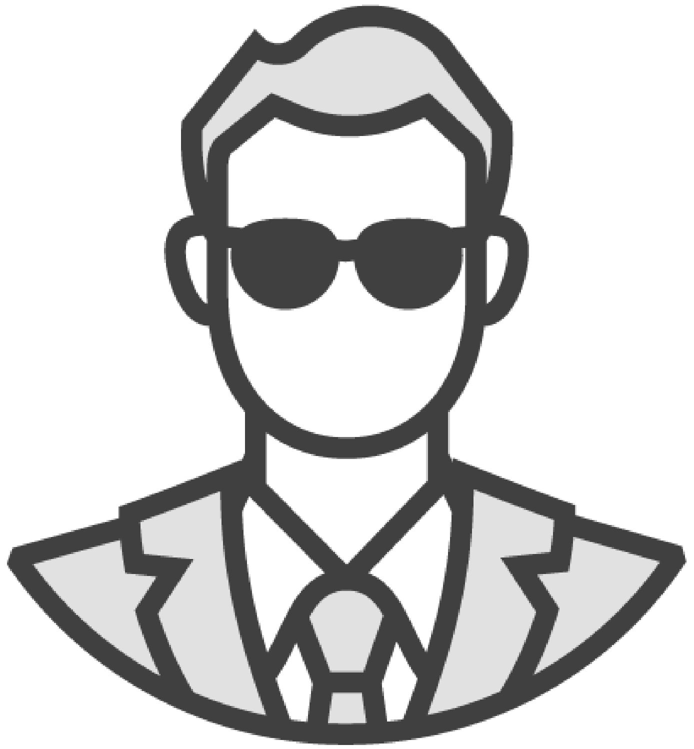
Securing ASP.NET Core 2 with OAuth2 and OpenID Connect

INTRODUCTION

Kevin Dockx
ARCHITECT

@KevinDockx https://www.kevindockx.com



Knowing how to secure applications is important…

… but knowing why we make certain decisions is, arguably, even more important

Coming Up

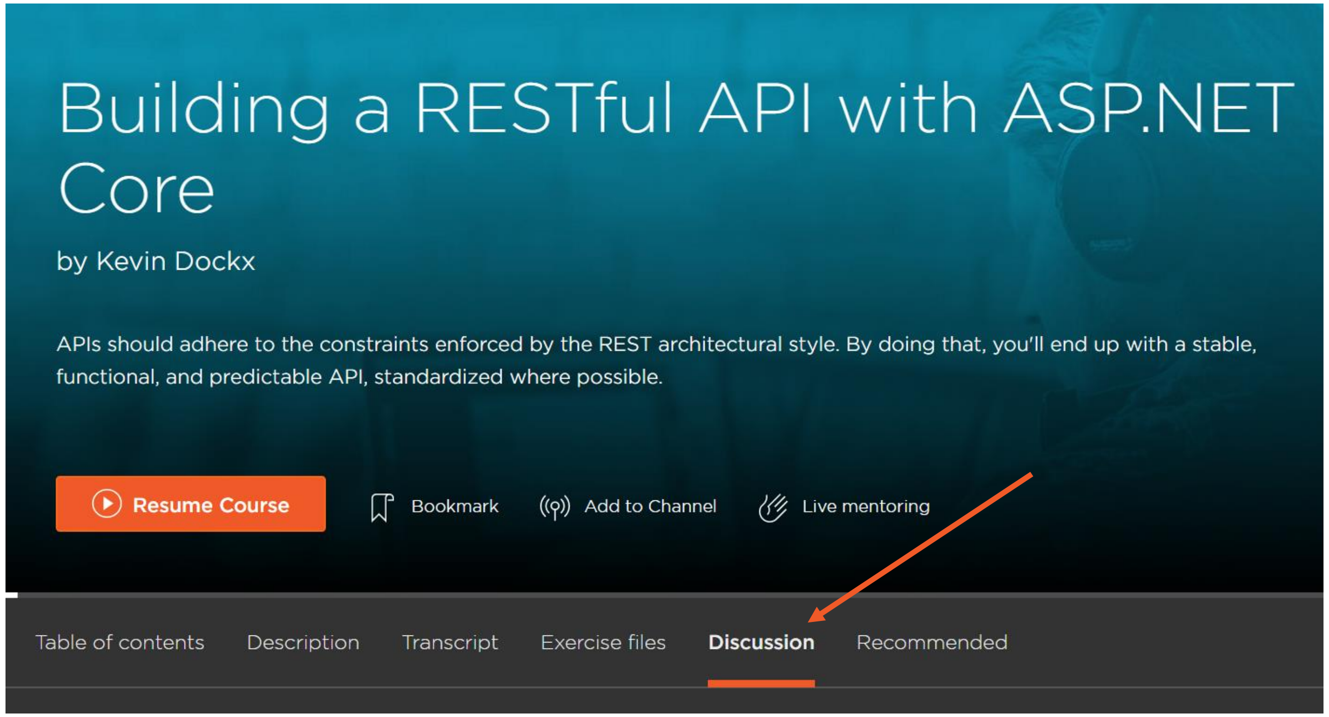
Course Prerequisites and Tooling

Application Architectures and Security

The Importance of a Central Identity Provider, OAuth2 and OpenID Connect

Discussion tab on the course page

Twitter: @KevinDockx


Building a RESTful API with ASP.NET Core

by Kevin Dockx

APIs should adhere to the constraints enforced by the REST architectural style. By doing that, you'll end up with a stable, functional, and predictable API, standardized where possible.

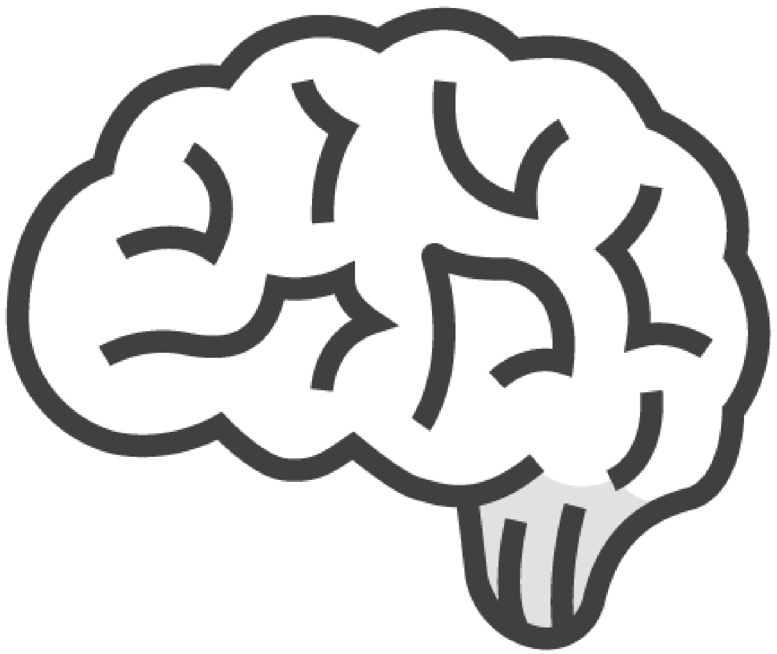
(Course shown is one of my other courses, not this one)

Course Prerequisites

Good knowledge of C#

Some knowledge of ASP.NET Core 2

Course Prerequisites



Building Your First ASP.NET Core 2.0 MVC Application with Visual Studio 2017 (Gill Cleeren)

Building Your First API with ASP.NET Core (yours truly)

Tooling

Visual Studio 2017

http://bit.ly/2dSGoN5

Visual Studio Code

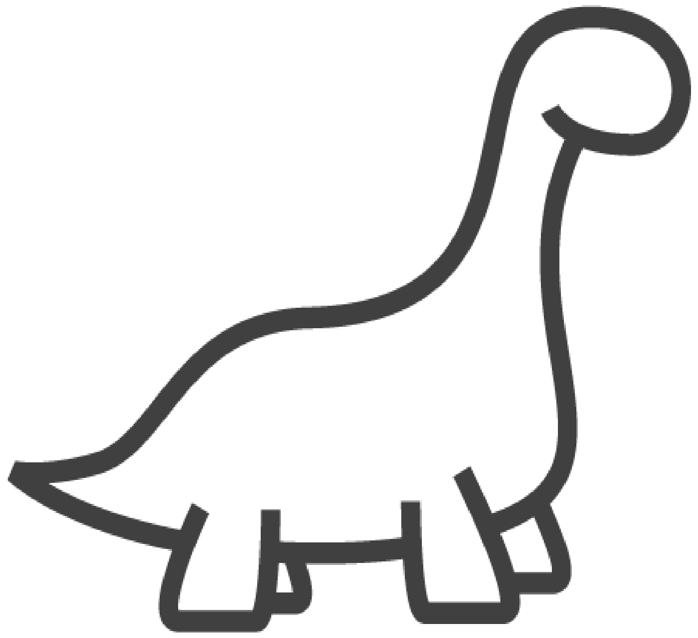
http://bit.ly/1J6QrU6

Visual Studio for Mac

http://bit.ly/2fXmQpH

.NET Core 2.0 SDK: http://bit.ly/2mJArWx

Application Architectures and Security



Thick client applications

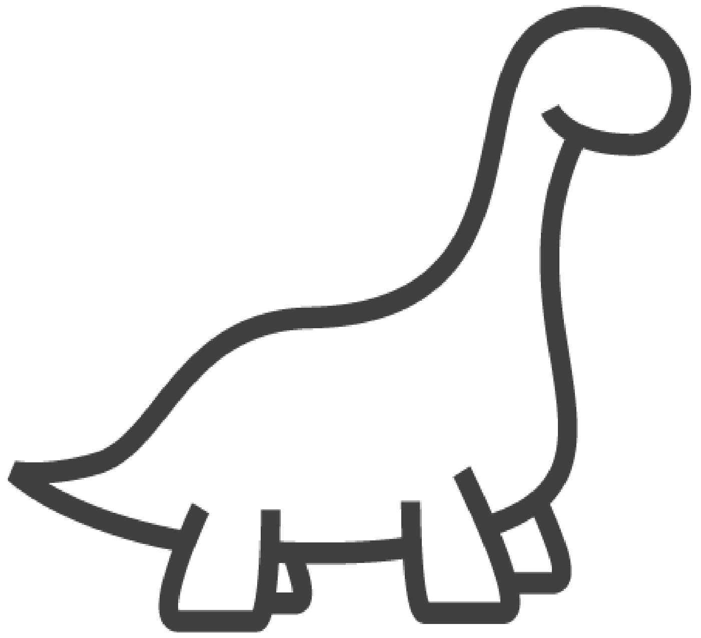
- Windows authentication

Server-side web applications

- Windows or Forms authentication

Not service-based

Application Architectures and Security



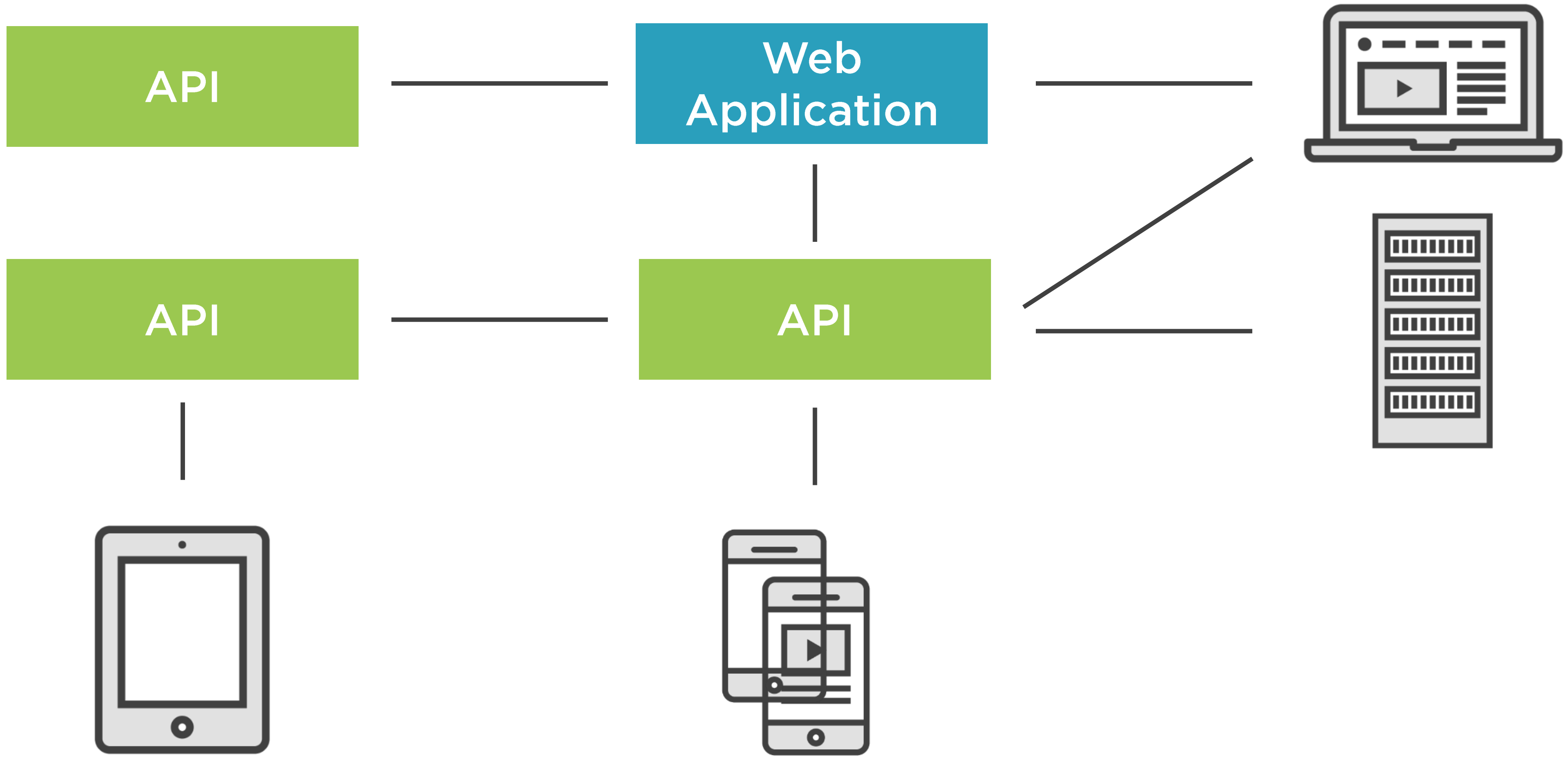
Service-based applications

- WS-Security (WCF)
- IP-level configuration (firewall)

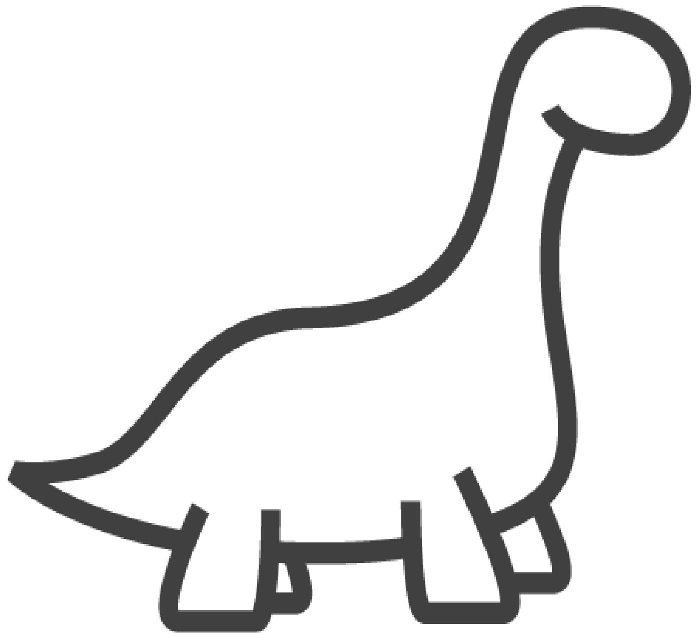
SAML 2.0

- Standard for exchanging authentication and authorization data between security domains

Application Architectures and Security



Application Architectures and Security

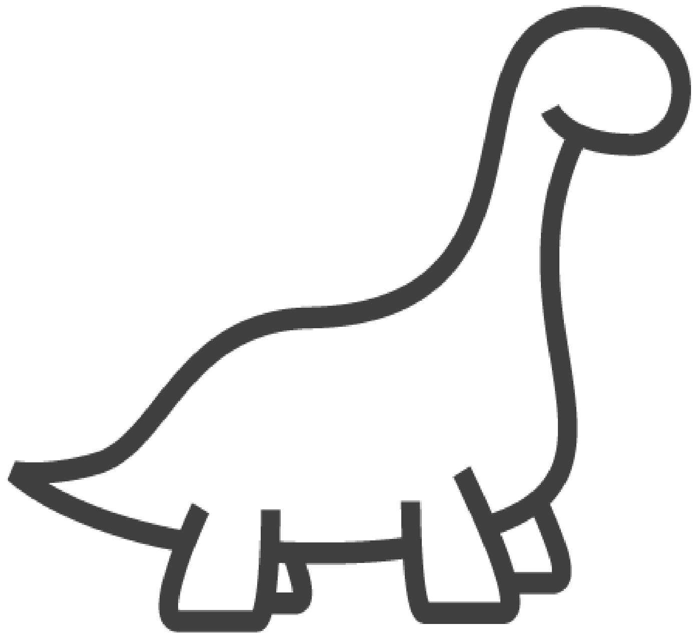


Client applications (often) require public APIs

Applications that live on the client can't be (decently) secured with means designed for use at the server

Sending username/password on each request proved to be a bad idea

Application Architectures and Security



Token-based security

- Client applications send tokens, representing consent, to API

Home-grown token services emerged...

- The application still has access to username/password

Reinventing the Wheel

Expiration

Token signing and validation

Token format

Authentication and authorization

Securely delivering tokens to different application types

...

Application Architectures and Security

A central identity provider

A protocol that's safe for authentication and authorization

Working Towards a Central Identity Provider

It’s the responsibility of an Identity Provider (IDP) to authenticate the user and, if needed, safely provide proof of identity to an application

Identity and Access Management-related Tasks

- User registration & management
- Locking out users
- Password policies, strength & resets

... are tedious tasks, prone to change

Handle them in a central location and reuse them across applications

Safely Storing Credentials

Early-days encryption mechanism can easily be brute forced

Key stretching algorithms discourage this…

… but the amount of stretching and the algorithms themselves are prone to change

Using Different Means of Authentication

Some systems might require certificates

Other systems might require a second or third factor of authentication

Common Tasks

User accounts are reused across applications

Identity and access management-related tasks are common concerns

Safely storing account-related information is prone to change

Means of authentication are added or changed

Working Towards a Central Identity Provider

A central Identity Provider (IDP) for Identity and Access Management (IAM) system solves these issues

Reinventing the Wheel (Reprise)

Expiration

Token signing and validation

Token format

Authentication and authorization

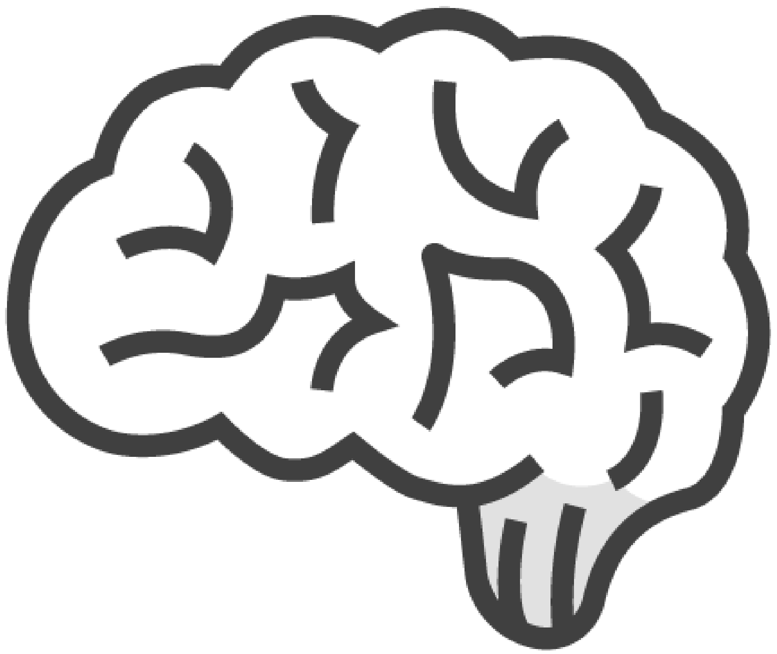
Securely delivering tokens to different application types

…

OAuth2

OAuth2 is an open protocol to allow secure authorization in a simple and standard method from web, mobile and desktop applications

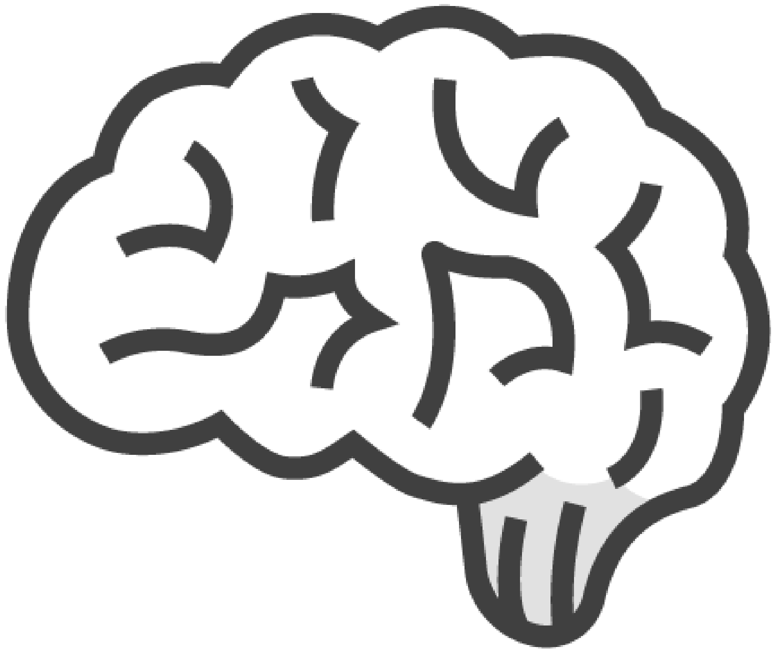
Introducing OAuth2



A client application can request an access token to gain access to an API

OAuth2 defines how a client application can securely achieve authorization

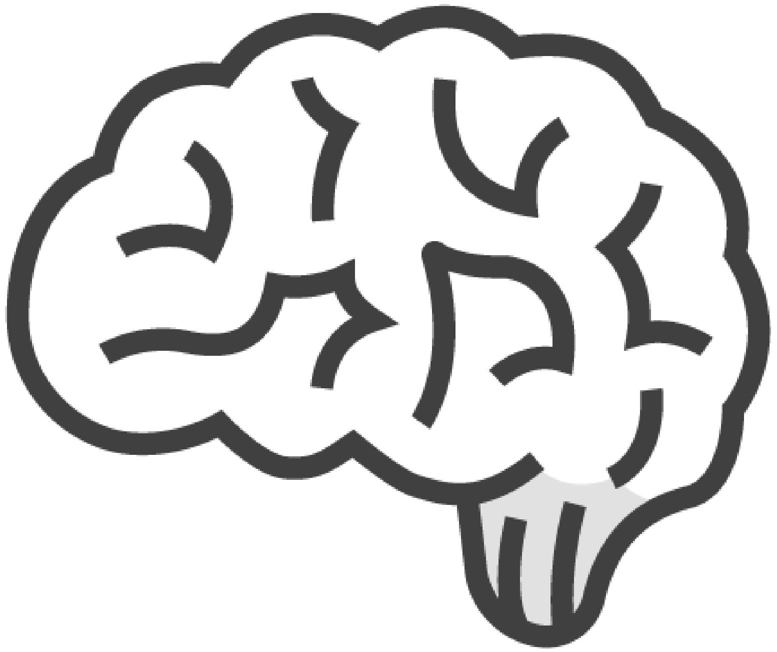
Introducing OAuth2



Homegrown endpoints are replaced by endpoints from the OAuth2 standard

The standard defines how to use these endpoints for different types of client applications

Introducing OAuth2



IdentityServer and Azure AD implement the OAuth2 standard

OpenID Connect

OpenID Connect is a simple identity layer on top of the OAuth2 protocol

Introducing OpenID Connect

A client application can request an identity token (next to an access token)

That identity token is used to sign in to the client application

The UserInfo endpoint allows a client application to get additional information on the user

Introducing OpenID Connect

OpenID Connect is the superior protocol: it extends and supersedes OAuth2

Even if the client application only requires authorization to access an API, we should use OIDC instead of plain OAuth2

Introducing OpenID Connect

OIDC isn't just for new or API-based applications

Demo

Introducing the Demo Application

Summary

Identity and Access Management (IAM) belongs at a central location: an Identity Provider (IDP)

That IDP should implement protocols that safely allow authentication and authorization: OpenID Connect (and OAuth2)