Authentication: Integrating with an Identity Provider

Kevin Dockx
ARCHITECT

@KevinDockx www.kevindockx.com

Coming Up

The Purpose of an Identity Provider

Introducing OpenID Connect

Signing in & Out

Protecting Angular Routes

An Identity Provider is...

a system entity that offers user authentication as a service

The Purpose of an Identity Provider

Login screen is at level of the IDP

- Avoids exposing user credentials to the relying party (client)
- Allows federation scenarios

Using an IDP centralizes common user-related responsibilities

Identity Provider Examples

Microsoft ADFS

- On-premise IDP

Azure AD

- Cloud-based IDP

IdentityServer4

- OpenID Connect and OAuth 2.0 framework for ASP.NET Core 2

Identity Provider Examples

Microsoft ADFS

- On-premise IDP

Azure AD

- Cloud-based IDP

IdentityServer4

- OpenID Connect and OAuth 2.0 framework for ASP.NET Core 2

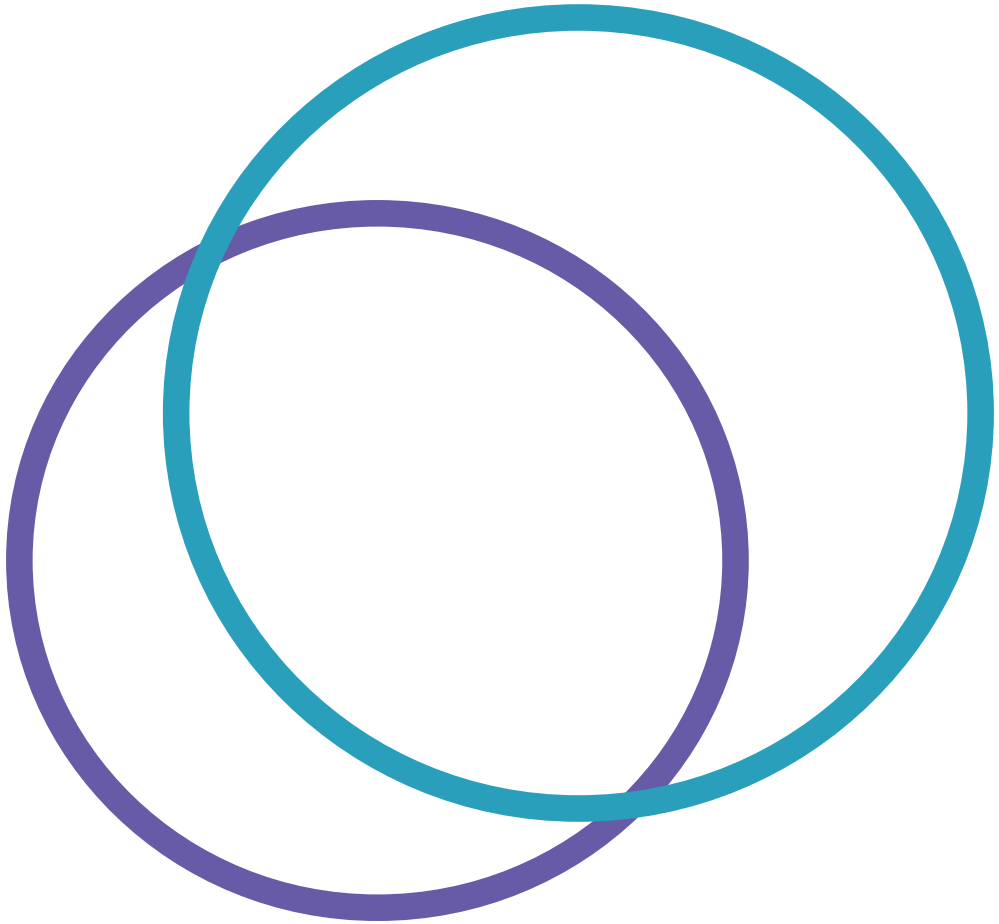
Integrating with an Identity Provider

We're going to integrate with an existing Identity Provider

Securing ASP.NET Core with OAuth2 and OpenID Connect

- http://bit.ly/2ksYDeo

Introducing OpenID Connect



OAuth 2.0

Authorization
Access token

Securing resources
(API)

OpenID Connect

Authentication
Identity token

Gaining identity
information and
signing in

Public and Confidential Clients

Confidential clients

Capable of maintaining the confidentiality of their credentials

(*clientid & clientsecret, or other means of authentication)*

Server-side web apps

Public clients

Incapable of maintaining the confidentiality of their credentials

(clientid & clientsecret, or other means of authentication)

Client-side applications (like Angular apps)

OpenID Connect Flows and Endpoints

The flow determines how the id_token (and potentially access_token) are returned to the client

A flow is a series of requests and responses between client and identity provider

Traffic should always be encrypted

OpenID Connect Flows and Endpoints

Endpoints at IDP Level

Authorization endpoint

Token endpoint

UserInfo endpoint

Discovery endpoint

Endpoints related to session management and logging out

Endpoints related to dynamic client registration

Endpoints at Client Level

Redirection endpoints

OpenID Connect Flows

Public Clients

Confidential Clients

Implicit

Authorization Code

Hybrid

OpenID Connect Flows

The implicit flow doesn't use the token endpoint, which requires client authentication

"Implicit" means the client is implicitly authenticated

The thing with security is that a lot of approaches will work, but most of them are not a good idea

A statement to remember

```
https://idphostaddress/connect/authorize?
client_id=tourmanagementclient
&redirect_uri=https://clientapphostaddress/callback
&scope=openid profile
&response_type=id_token
&nonce=324...zeIIEMc00
```

The Implicit Flow

Authentication request to the authorization endpoint

```
https://idphostaddress/connect/authorize?
client_id=tourmanagementclient
&redirect_uri=https://clientapphostaddress/callback
&scope=openid profile
&response_type=id_token
&nonce=324...zeIIEMc00
```

The Implicit Flow

The client identifier

```
https://idphostaddress/connect/authorize?
client_id=tourmanagementclient
&redirect_uri=https://clientapphostaddress/callback
&scope=openid profile
&response_type=id_token
&nonce=324...zeIIEMc00
```

The Implicit Flow

The redirection endpoint at level of the client application

```
https://idphostaddress/connect/authorize?
client_id=tourmanagementclient
&redirect_uri=https://clientapphostaddress/callback
&scope=openid profile
&response_type=id_token
&nonce=324...zeIIEMc00
```

The Implicit Flow

The requested scopes

- A scope, in this context, matches one or more claims about the user
- **openid** is a required scope that matches the user's identifier (sub)
- **profile** matches profile-related claims (given_name, ...)

```
https://idphostaddress/connect/authorize?
client_id=tourmanagementclient
&redirect_uri=https://clientapphostaddress/callback
&scope=openid profile
&response_type=id_token
&nonce=324...zeIIEMc00
```

The Implicit Flow

The response mode signifies the flow

- “id_token” and “id_token token” signify that the implicit flow will be used

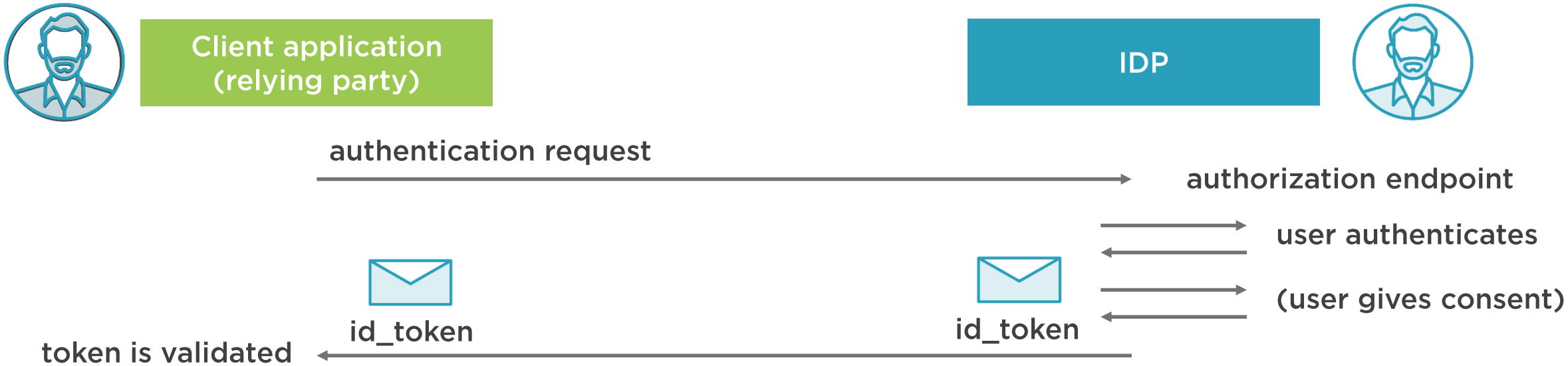
```
https://idphostaddress/connect/authorize?
client_id=tourmanagementclient
&redirect_uri=https://clientapphostaddress/callback
&scope=openid profile
&response_type=id_token
&nonce=324...zeIIEMcO0
```

The Implicit Flow

Nonce = number only used once

- Protects against replay attacks

The Implicit Flow

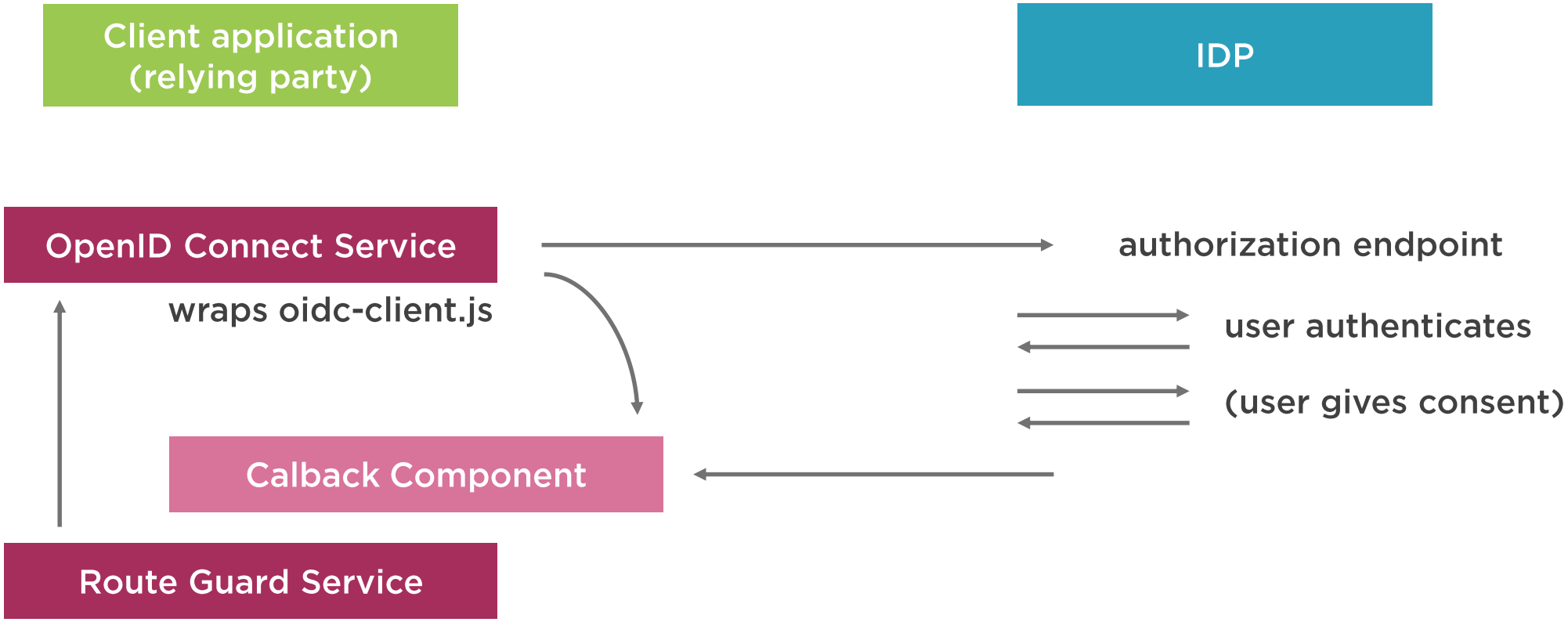


JavaScript Middleware

oidc-client.js

- OpenID Connect (OIDC) and OAuth2 protocol support for browser-based JavaScript applications
- http://bit.ly/2BRRmhW

Implementing OpenID Connect Support



Demo

Supporting TLS

Demo

Inspecting the IdentityServer Project

Demo

Creating an OpenID Connect Service

Demo

Adding a Callback Page

Demo

Signing In

Demo

Signing Out

Demo

Adding a Guard Service

Demo

Using Identity Claims in Our Application

Summary

OpenID Connect is an identity layer on top of the OAuth 2.0 protocol

- Authentication
- Identity tokens

OAuth 2.0 is an open protocol to allow secure authorization in a simple and standard method from web, mobile and desktop applications

- Authorization
- Access tokens

Summary

An Angular client is a public client

- Implicit flow

We cannot protect code that is already on the client