Understanding Authentication With OpenID Connect

Kevin Dockx
ARCHITECT

@KevinDockx https://www.kevindockx.com

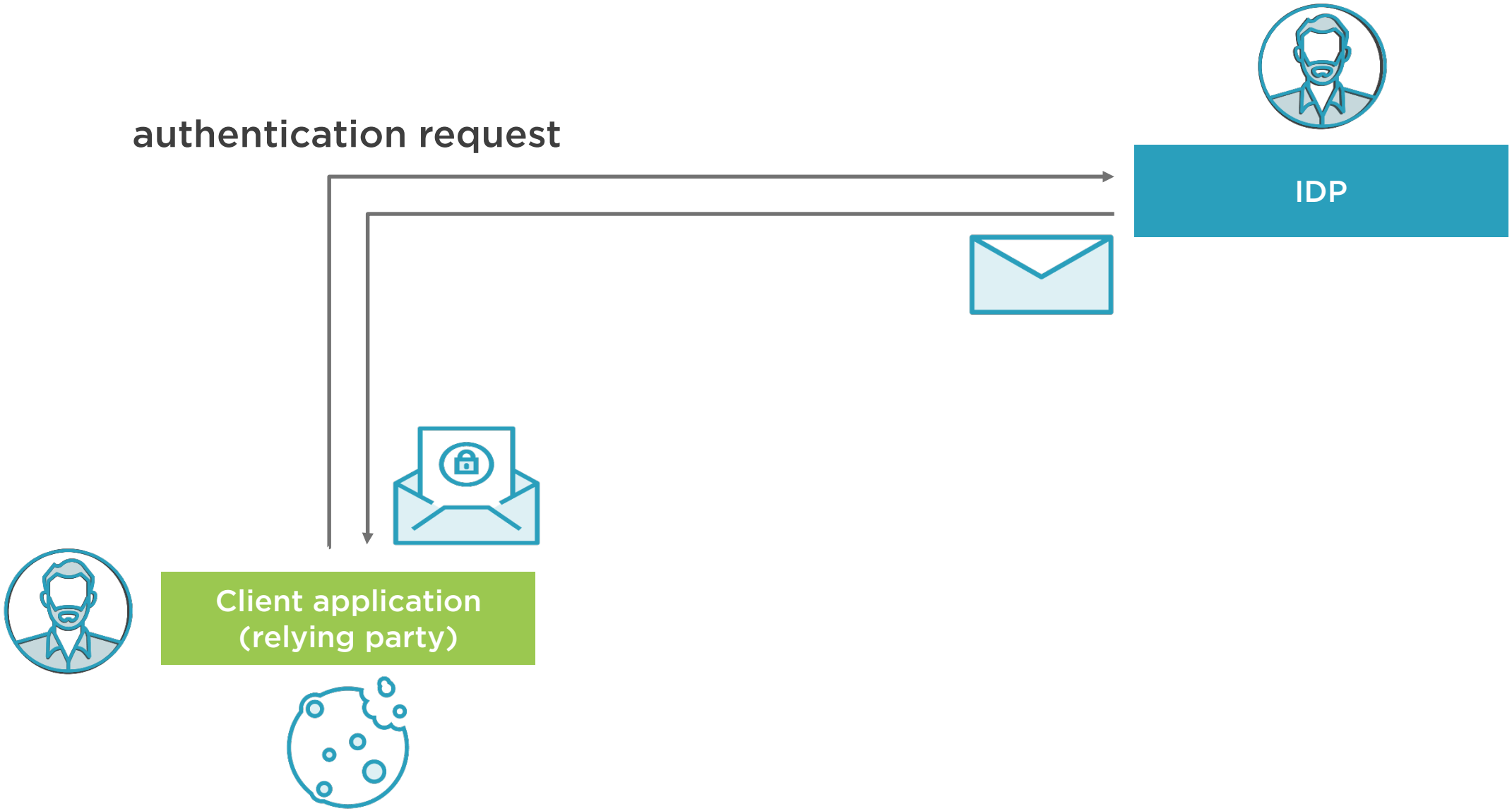
Coming Up

How OpenID Connect Works

Clients, Endpoints and Flows

Setting up an IDP: IdentityServer4

How OpenID Connect Works



Public and Confidential Clients

Confidential clients

Capable of maintaining the confidentiality of their credentials (clientid, clientsecret)

Live on the server

Eg: server-side web apps

Public clients

Incapable of maintaining the confidentiality of their credentials (clientid, clientsecret)

Live on the device

Eg: JavaScript apps, mobile apps

OpenID Connect Flows and Endpoints

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

Depending on application type, requirements, ... we must use a different flow

OpenID Connect Endpoints

Authorization endpoint (IDP level)

- Used by the client application to obtain authentication and/or authorization, via redirection

TLS (SSL) is a requirement (for all)!

OpenID Connect Endpoints

Redirection endpoint (client level)

- Used by the IDP to return code & token(s) to the client application

OpenID Connect Endpoints

Token endpoint (IDP level)

- Used by the client application to request tokens (without redirection) from the IDP

OpenID Connect Flows

Authorization Code

Tokens from token endpoint

Confidential clients

Long-lived access

Implicit

Tokens from authorization endpoint

Public clients

No long-lived access

Hybrid

Tokens from authorization endpoint & token endpoint

Confidential clients

Long-lived access

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

The most important statement of the entire course

OpenID Connect Flow for ASP.NET Core MVC

ASP.NET Core MVC

- Confidential client (server-side web app)
- We require long-lived access

Authorization Code used to be advised

Today, Hybrid is advised

- Verifiable id_token through authorization endpoint
- Tokens are linked

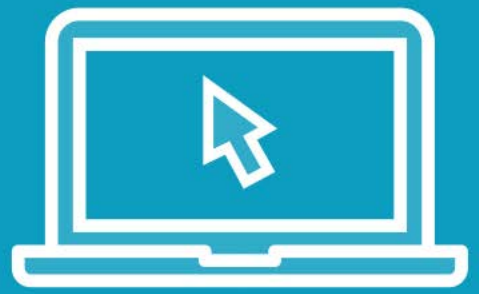
Introducing IdentityServer4

OpenID Connect and OAuth2 framework for ASP.NET Core by Dominick Baier and Brock Allen

Officially certified by the OpenID Foundation & part of the .NET Foundation

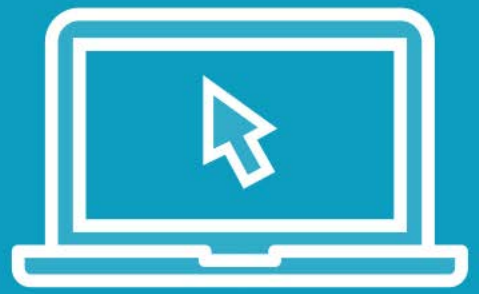
Open source: http://bit.ly/29HW80z

Demo



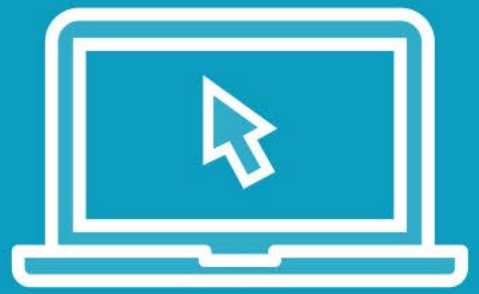
Setting up IdentityServer4

Demo



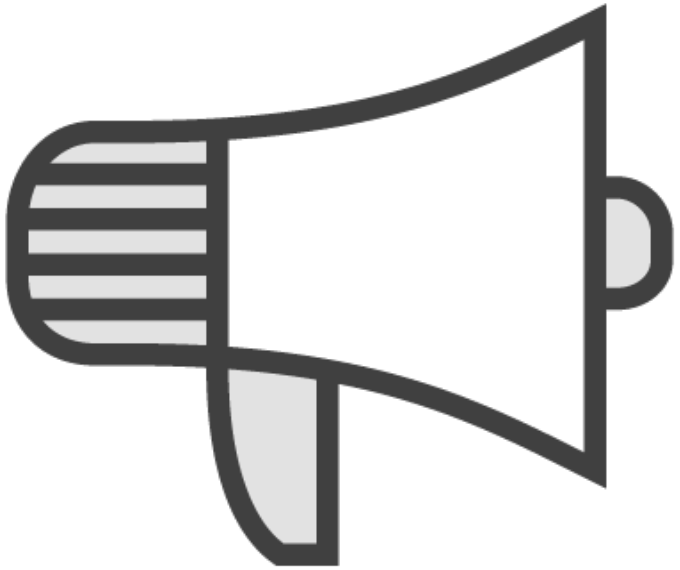
Adding a User Interface for IdentityServer4

Demo



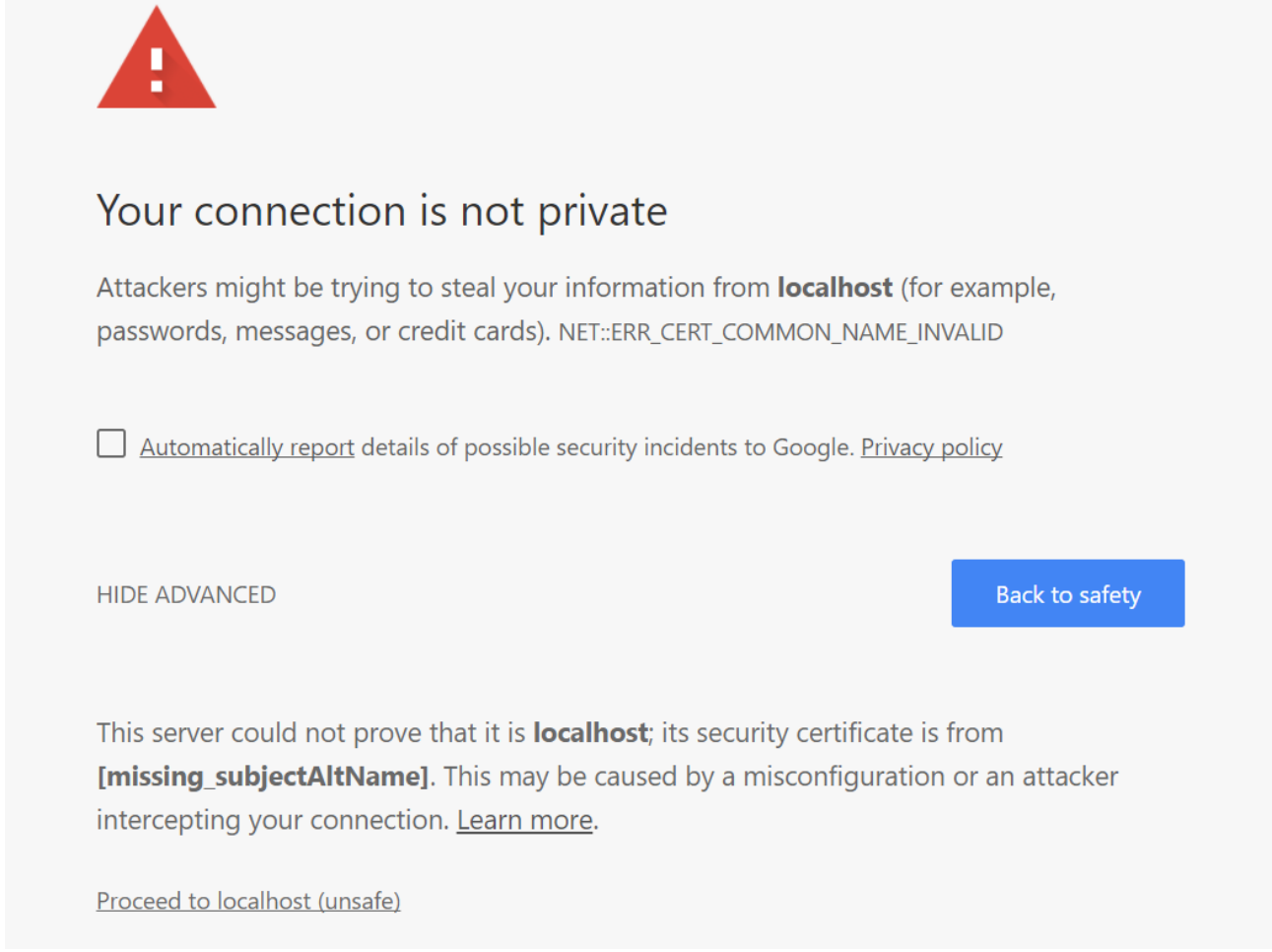
Ensuring Traffic Is Encrypted

A Note for Chrome Users



From version 58+, Google Chrome no longer considers the localhost certificate distributed with Visual Studio 2017 as safe

A Note for Chrome Users



Summary

A confidential client can safely store secrets

A public client can't safely store secrets

A flow can be seen as how an application can achieve authentication (and authorization)

Summary

Authorization endpoint (IDP)

- Used by the client application to obtain authentication and/or authorization

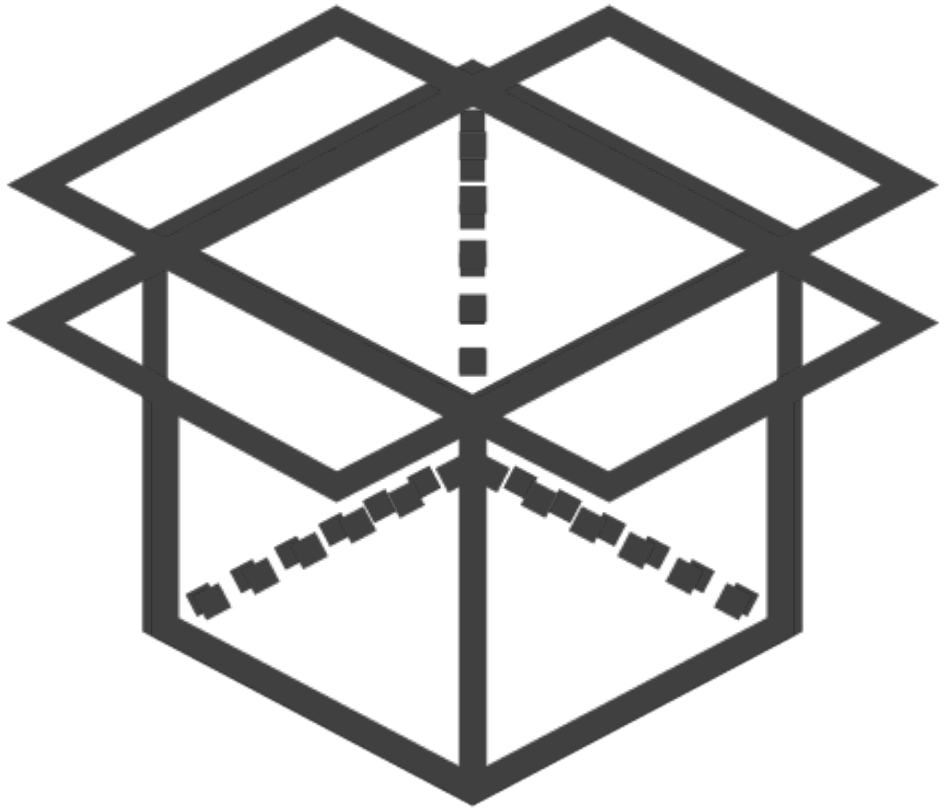
Token endpoint (IDP)

- Used by an application to programmatically request tokens

Redirection endpoint (client application)

- Where the tokens are delivered to from the authorization endpoint

TLS is a requirement!



It's our responsibility to keep the holes in this box as small as possible