

Getting Ready for Production



Kevin Dockx

ARCHITECT

@KevinDockx <https://www.kevindockx.com>



Coming Up



Using a signing certificate

**Persisting configuration and
operational data**



Using a Signing Certificate

builder.AddDeveloperSigningCredential()

- Load balancer can cause requests to end up at different servers
- Application pool recycling will reset credentials





Signing material

- Raw RSA (SHA256) keys
- Signing certificate
 - Best stored in a certificate store (or comparable)

Demo



Creating a signing certificate



Demo



Using a signing certificate



Configuration Data and Operational Data

Configuration data

Resources

Clients

Startup configuration data

Persistent store is advisable

Implement `IResourceStore`, `IClientStore`

Operational data

Authorization codes

Reference tokens, refresh tokens

Consent

Persistent store must be used

Implement `IPersistedGrantStore`



Demo



Persisting configuration data



Demo



Persisting operational data



Handling What's Next: Dealing with Users and Credentials

OpenID Connect doesn't deal with credentials, but applications do need to work with them

- Connecting to a user database
- Integrating with 3rd party providers
- Integrating with Active Directory
- User management
- 2FA, MFA
- ...



Handling
What's Next:
Dealing with
Users and
Credentials

**Dealing with Credentials when Securing an
ASP.NET Core 3 Application**



Summary



Use an SHA256 certificate, stored in a safe place (like a certificate store)



Summary



Configuration data should go in a persistent store

- Resources, clients, *startup configuration*

Operational data must go in a persistent store

- Authorization codes, reference tokens, refresh tokens, consent



you are ready



to be
Awesome!!!
ooo

@KevinDockx

