Securing Microservices in ASP.NET Core

SECURING YOUR FIRST MICROSERVICE

Kevin Dockx
ARCHITECT

@KevinDockx https://www.kevindockx.com

Coming Up

Course prerequisites and tooling

Inspecting the demo application

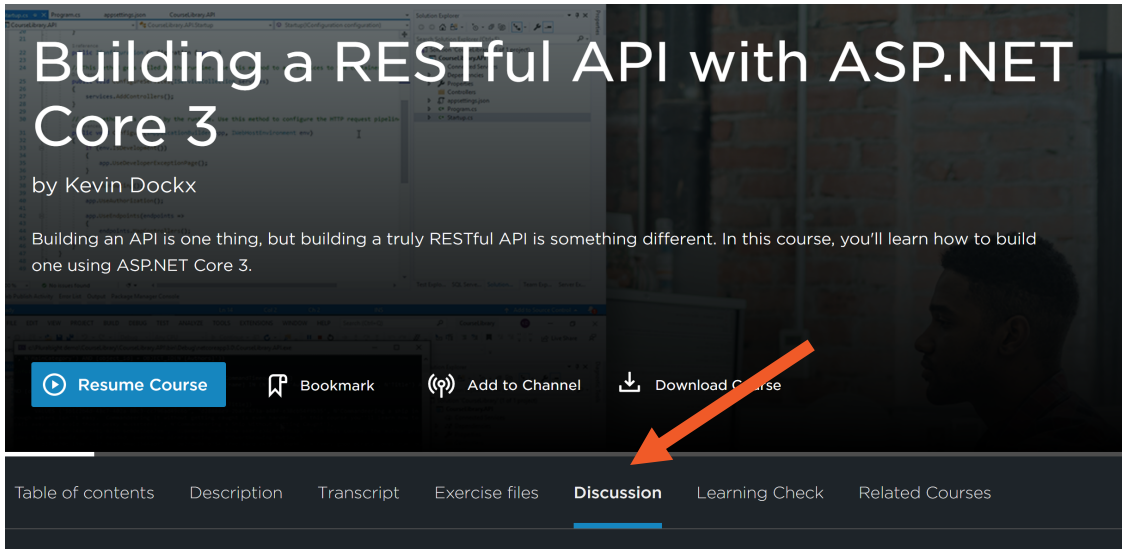
Token-based security for microservices

Accessing microservices

- On behalf of the client application
- On behalf of the user

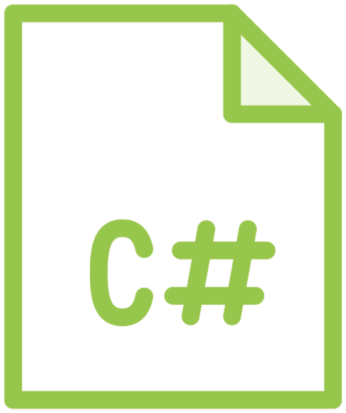
Discussion tab on the course page

Twitter: @KevinDockx



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

Course Prerequisites



Good knowledge of C#

Some knowledge of microservices in ASP.NET Core

Building Microservices with ASP.NET Core

This course is part of a learning path on Pluralsight

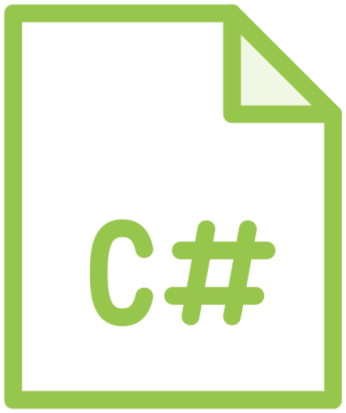
Microservices: the big picture	Getting started	Microservices communication
Data management	**Securing microservices**	Versioning
Deploying microservices	Cross-cutting concerns	Scalability and availability

Building Microservices with ASP.NET Core

This course is part of a learning path on Pluralsight

Microservices: the big picture	Getting started	Microservices communication
Data management	Securing microservices	Versioning
Deploying microservices	Cross-cutting concerns	Scalability and availability

Course Prerequisites



Good knowledge of C#

Some knowledge of microservices in ASP.NET Core

Some knowledge of OAuth2, OpenID Connect and IdentityServer4

Frameworks and Tooling

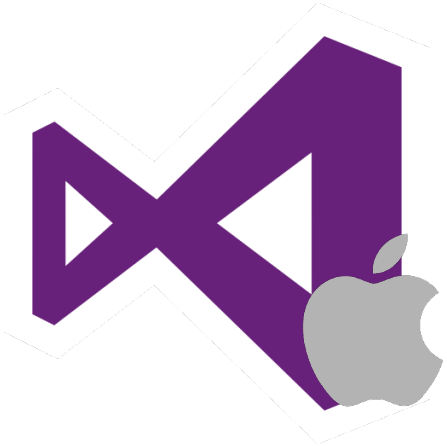
Visual Studio 2019
v16.4 or better

.NET Core 3.1

Frameworks and Tooling

Visual Studio 2019
v16.4 or better

Visual Studio
Code



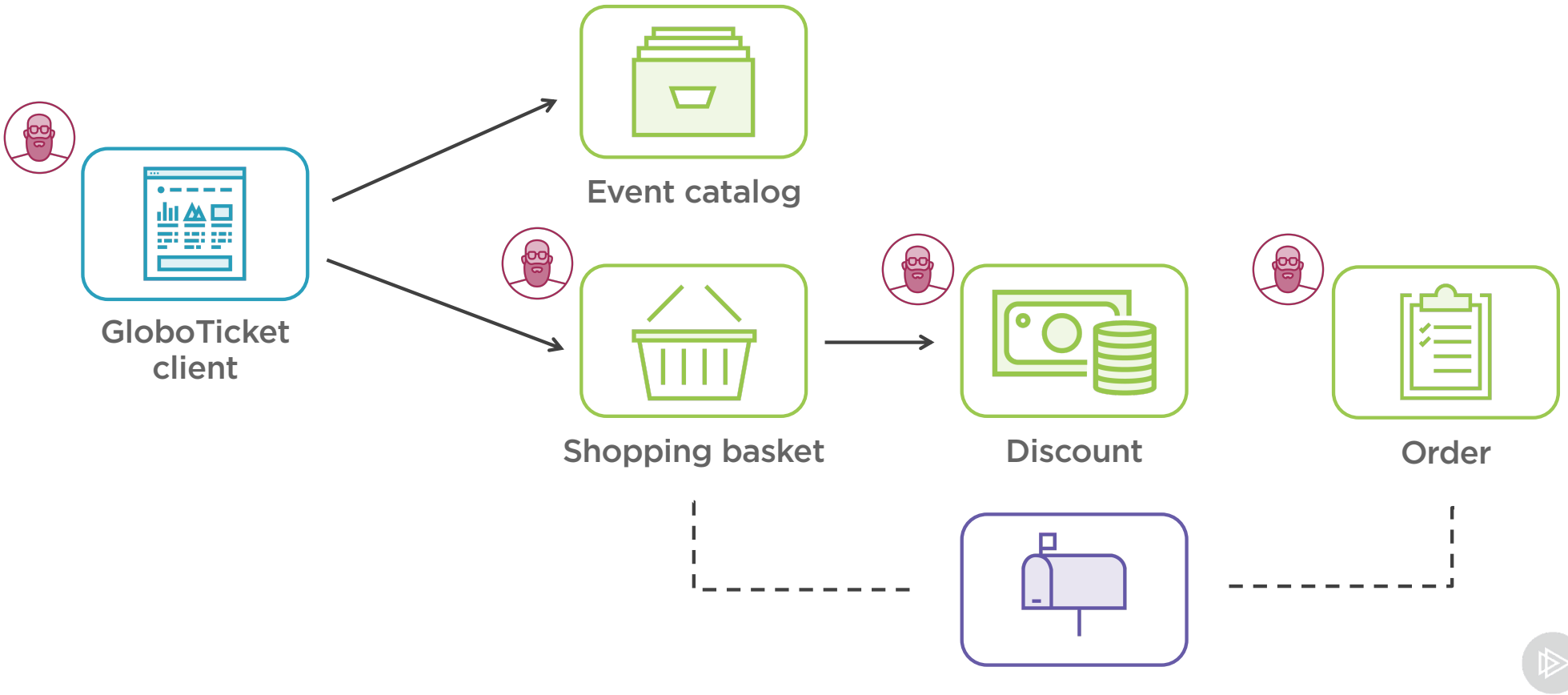
Visual Studio for
Mac

JetBrains Rider

Exercise files tab on the course page

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

Inspecting the GloboTicket Demo Application



Inspecting the GloboTicket Demo Application

The GloboTicket application as used in the path potentially contains additional microservices

- With our current set of microservices we can cover all security scenarios

Demo

Getting started with the GloboTicket demo application

Token-based Security for Microservices

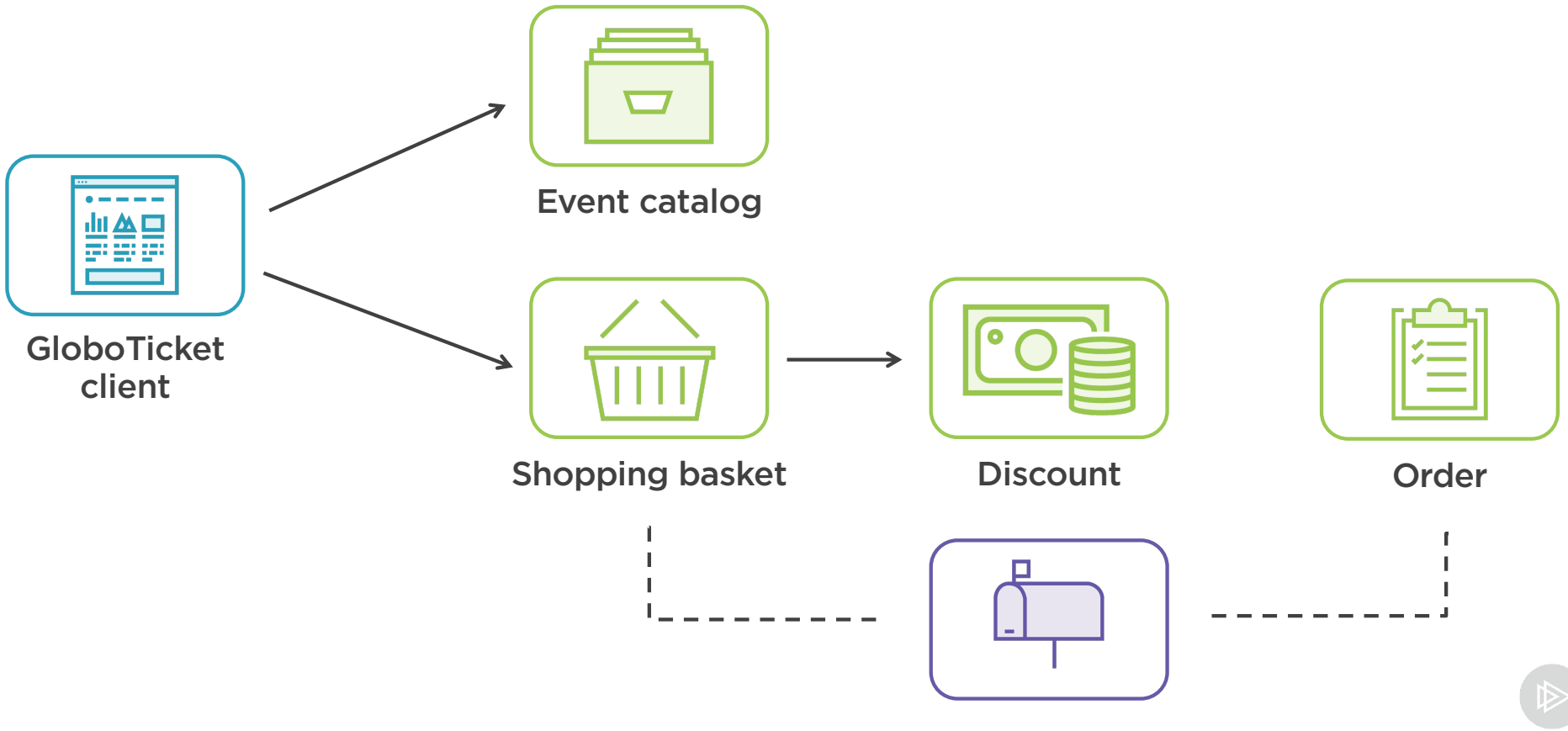
Multiple approaches to microservices architecture security exist

From quick & simple to complicated but best-of-class

Microservices architectures exist with or without an API gateway, with or without a BFF, with or without a bus, ...

These choices have an impact on what's possible, security-wise

Token-based Security for Microservices



Token-based Security for Microservices

Token-based Security for Microservices

We need a service that can generate tokens for us

- Tokens that provide access on behalf of a client application
- Tokens that provide access on behalf of a user

Identity service / identity provider

Token-based Security for Microservices

We don’t want to reinvent the wheel

- OAuth2 and OpenID Connect are proven and tested protocols

OAuth2

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

Introducing OAuth2

OAuth2 defines an access token

- A client application can request such an access token to gain access to an API / microservice
- It thus defines how a client application can securely achieve authorization

The access token does not have a notion of the user

OpenID Connect

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

Introducing OpenID Connect

OpenID Connect defines an identity token

- A client application can request an identity token (next to an access token) and use it to sign in to the client application
- The access token can be used to access an API / microservice

The access token has a notion of the user

Choosing an Identity Provider

Azure AD

Ping

Okta

Auth0

IdentityServer4

- http://docs.identityserver.io/

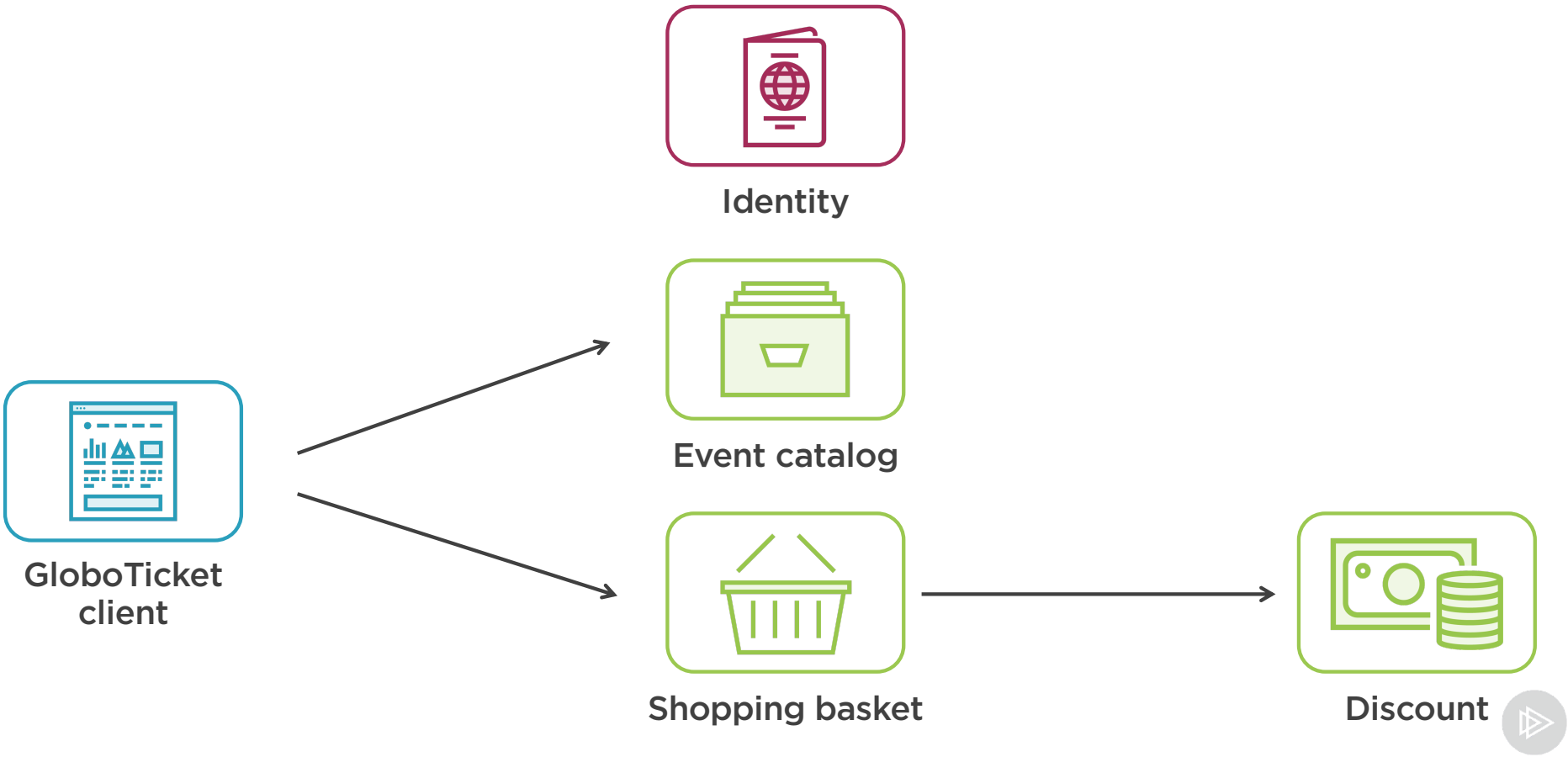
IdentityServer4 is an OpenID Connect and OAuth2 framework for ASP.NET Core

- De facto standard in the .NET Core world

Standardization is key

Everything we'll implement will be according to existing, proven and tested standards

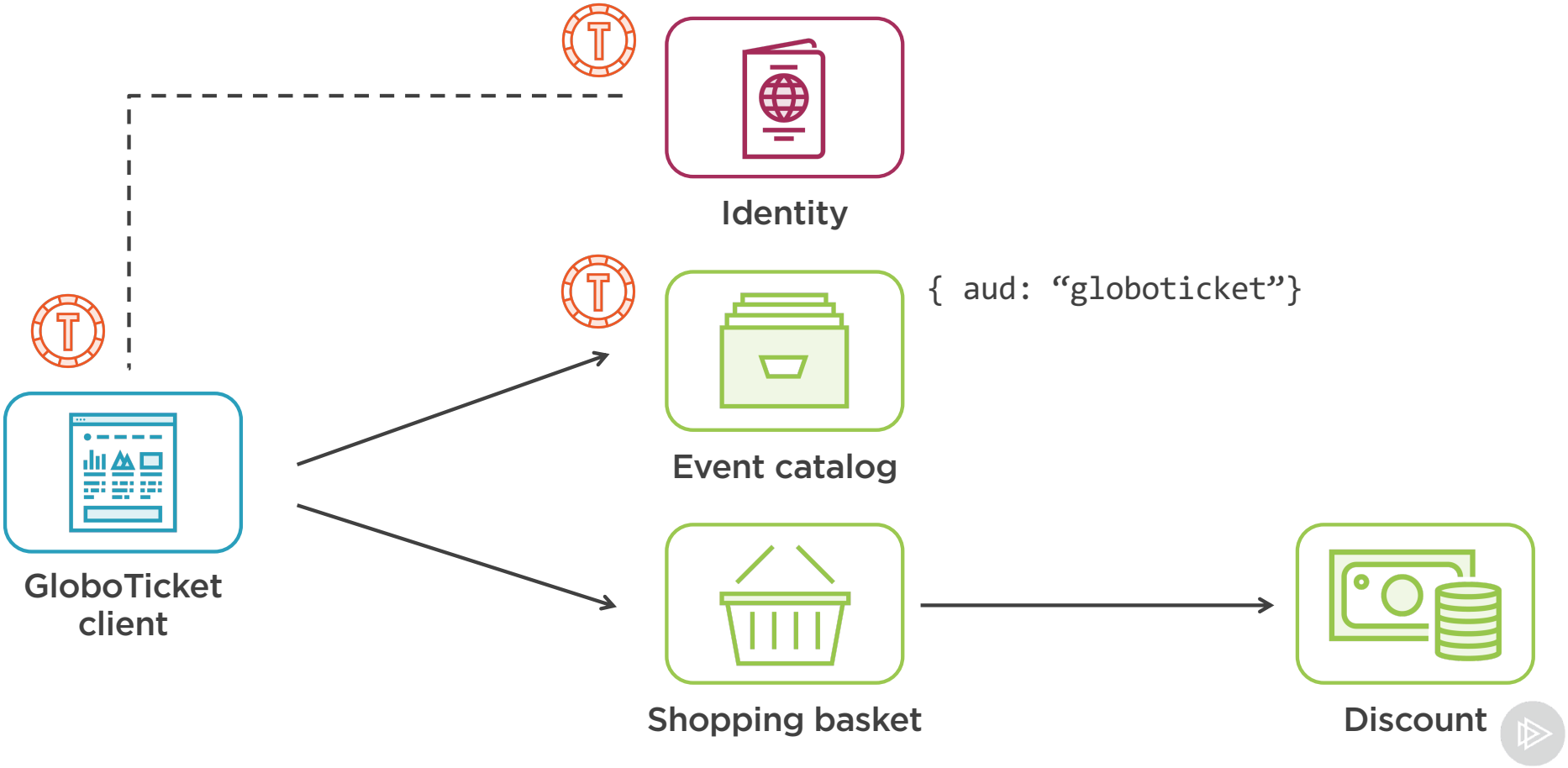
GloboTicket with Identity Service



Demo

Inspecting an identity service

Accessing a Microservice on Behalf of the Client



Client Credentials Flow

Demo

Blocking access to a microservice

Demo

Accessing a microservice on behalf of the client application

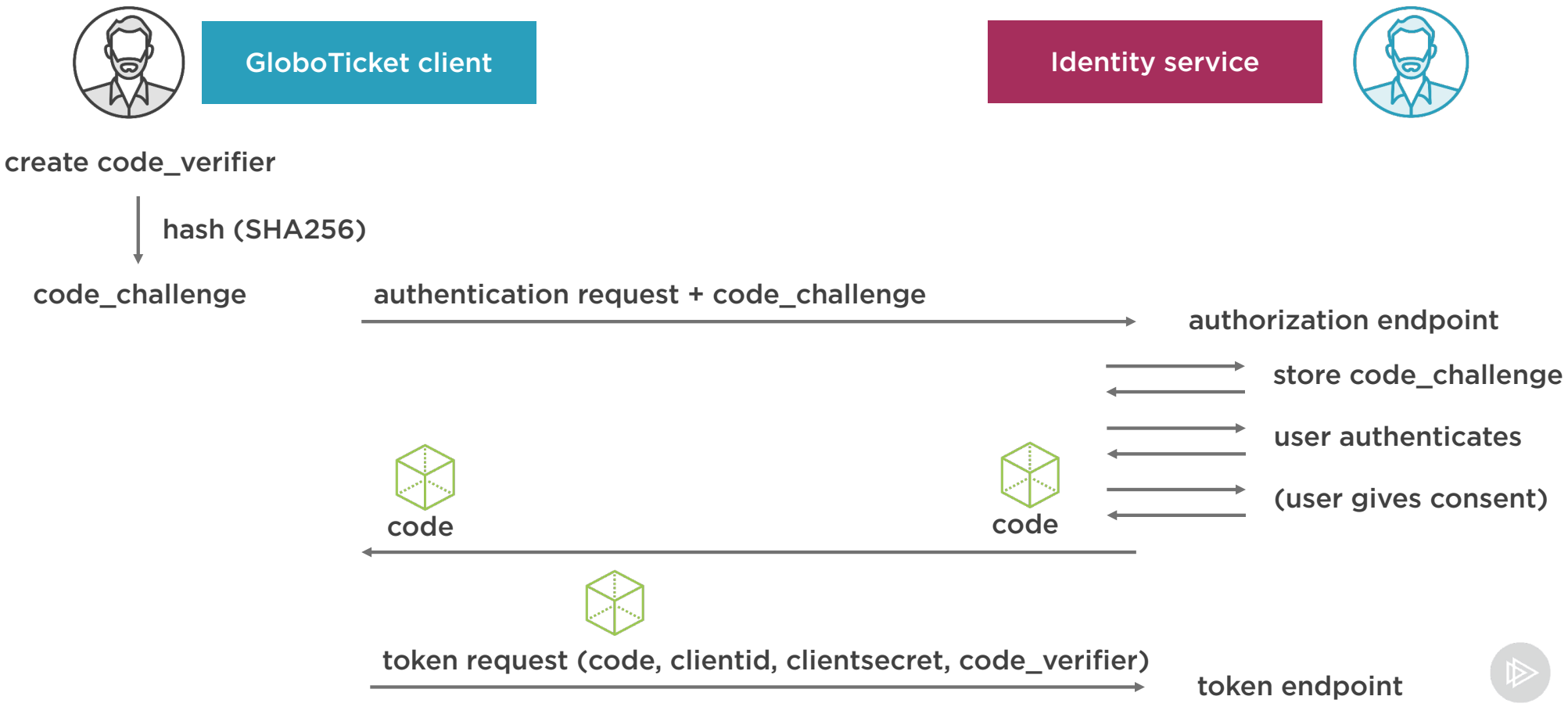
Using the Identity Microservice to Log In

More often than not, applications (clients and APIs) need to know who the user is

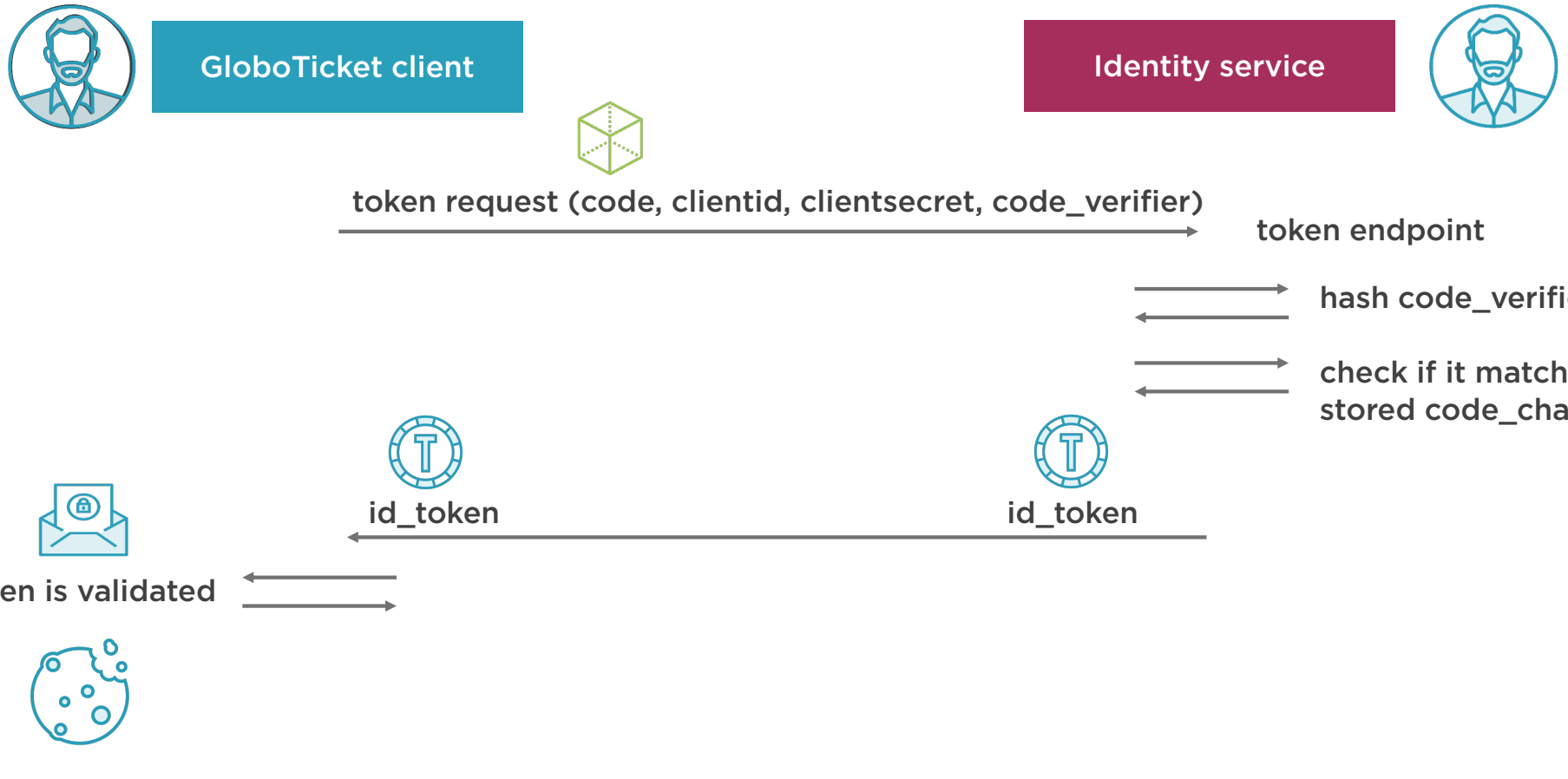
For client applications, that information is delivered in an identity token as proof of identity

- Used to log in
- "sub" claim defines the user

Authentication with an Identity Token



Authentication with an Identity Token



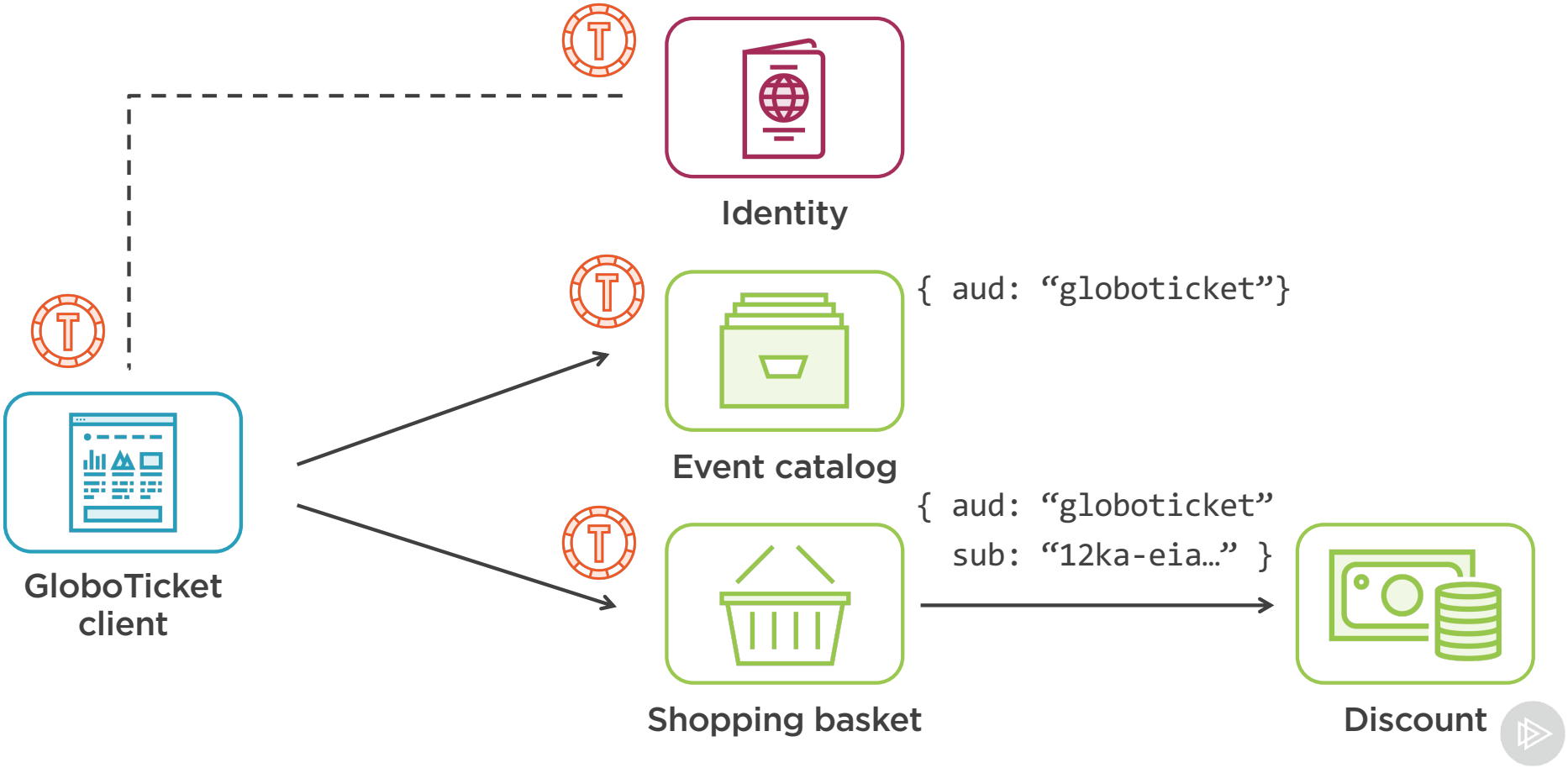
Demo

Using the identity microservice to log in

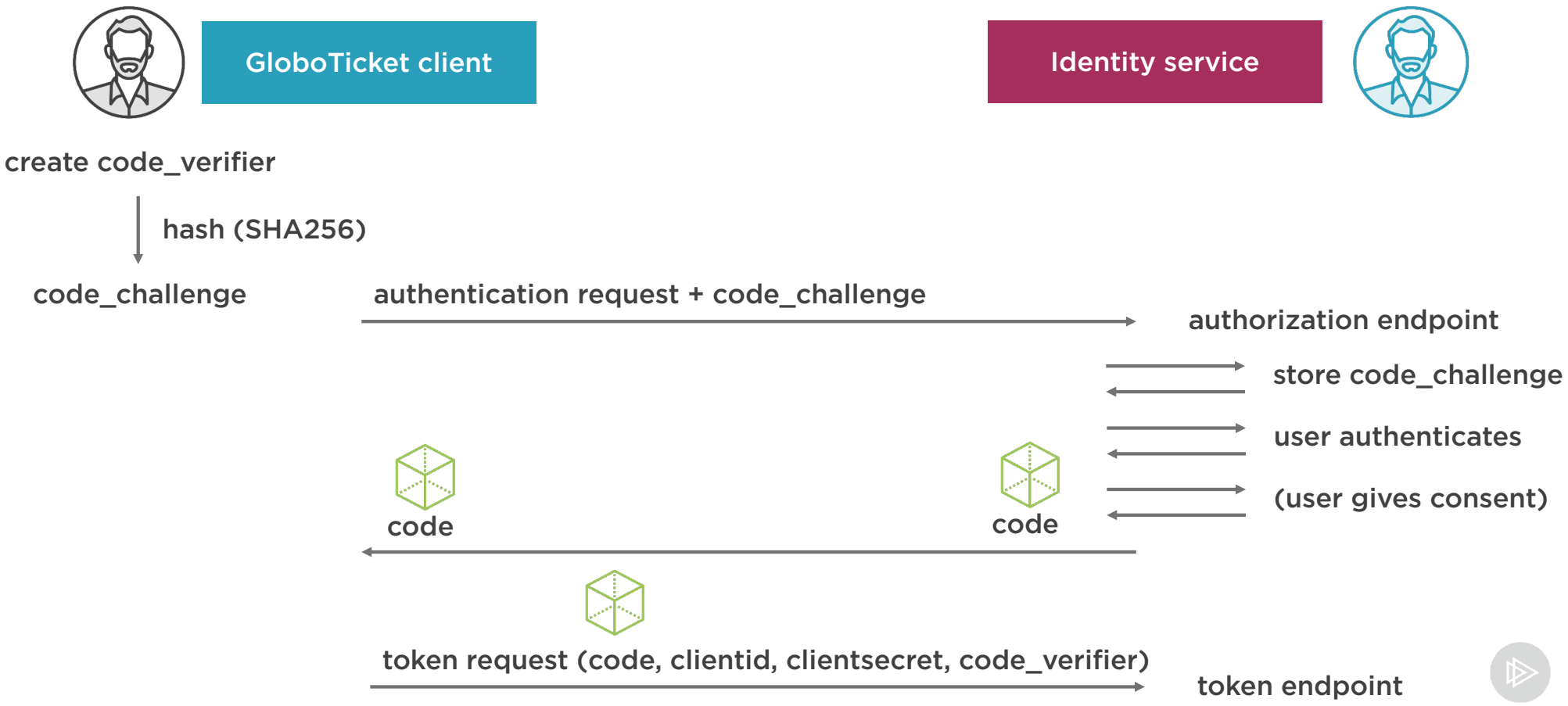
Demo

Logging out

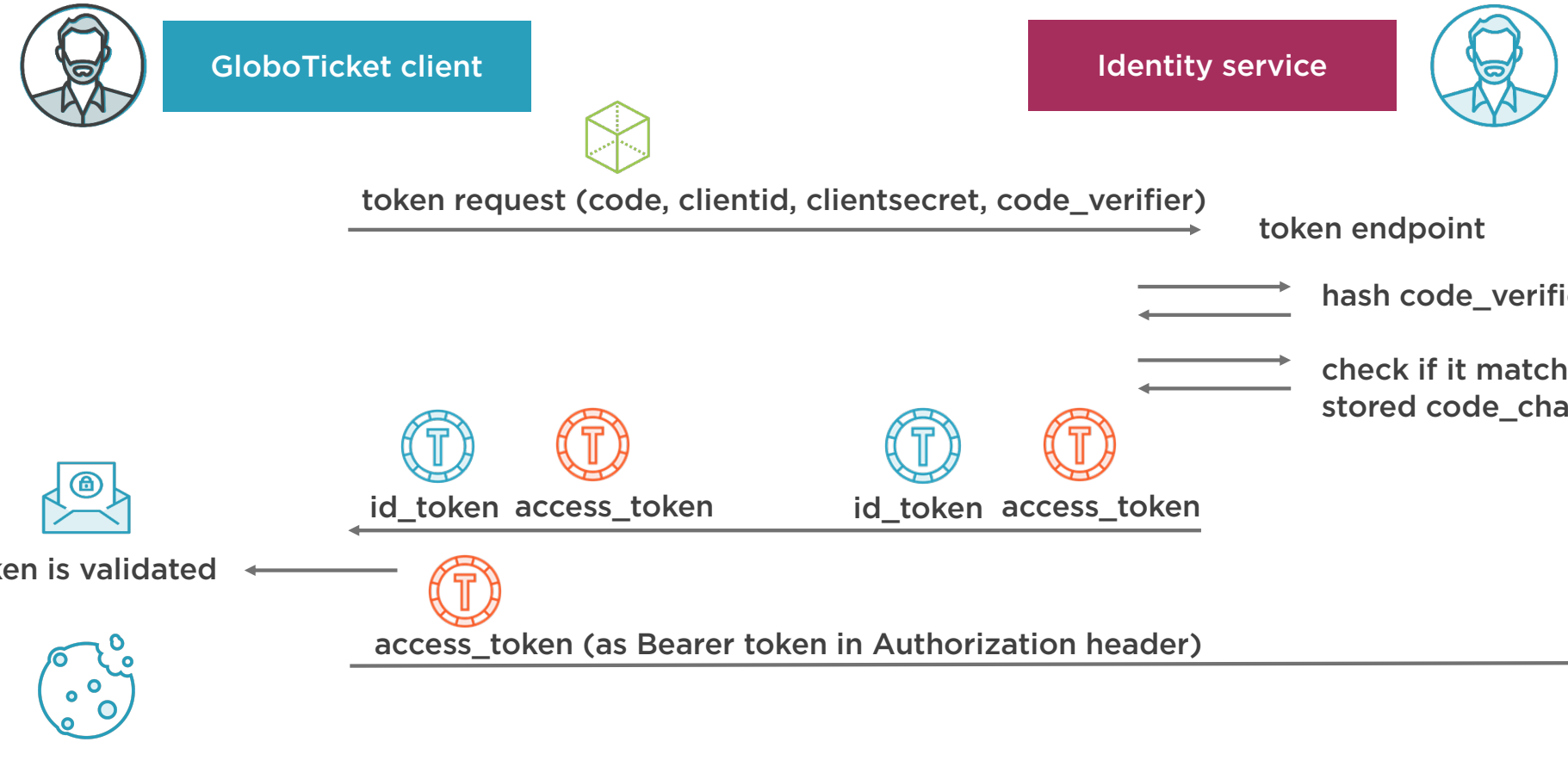
Accessing a Microservice on Behalf of the User



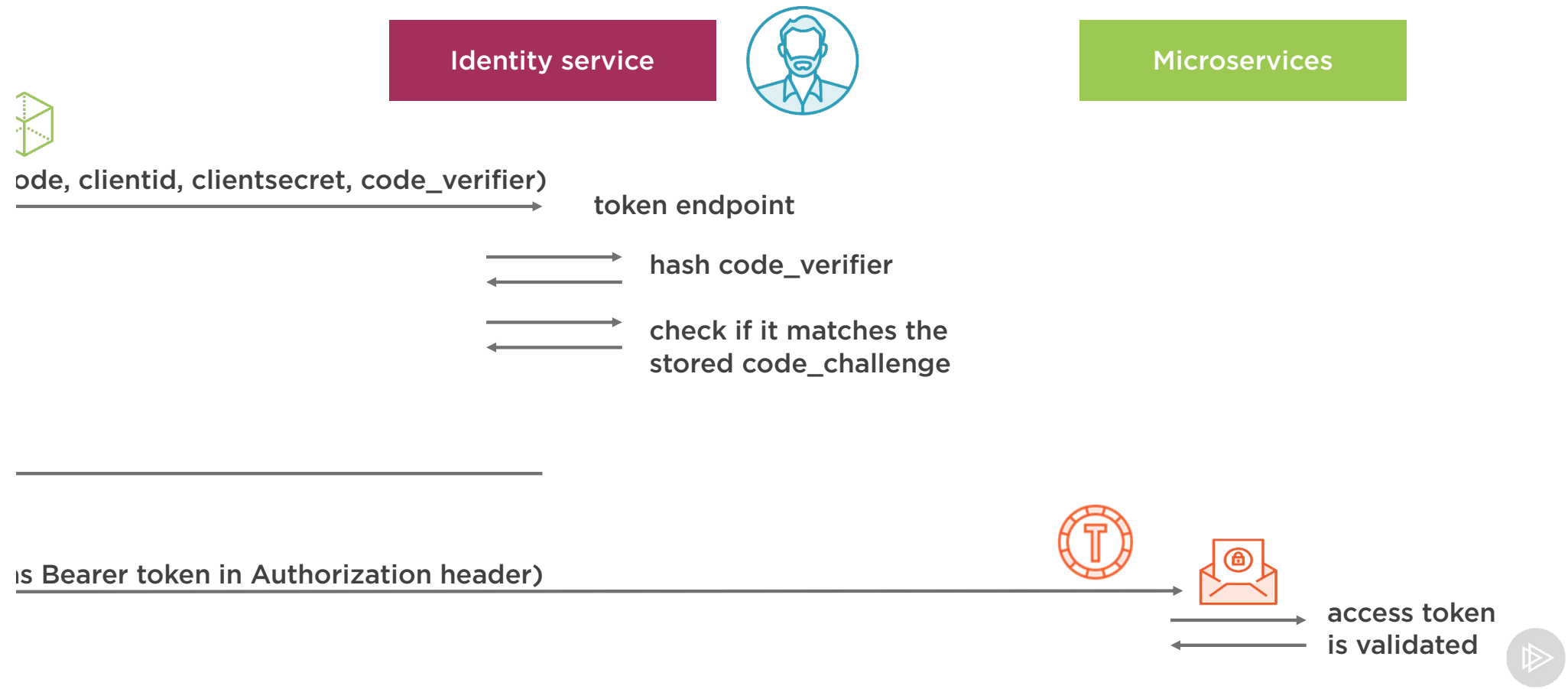
Authorization with an Access Token



Authorization with an Access Token



Authorization with an Access Token



Demo

Accessing a microservice on behalf of the user

Summary

Use token-based security to secure your microservices

- OAuth2, OpenID Connect

The microservice should check the incoming token for an audience value

- JwtBearerAuthentication middleware

Summary

Use the client credentials flow to

- Get an access token to access a microservice on behalf of the client

Use the code flow with PKCE protection to

- Sign in to the client application with an identity token
- Get an access token to access a microservice on behalf of the user