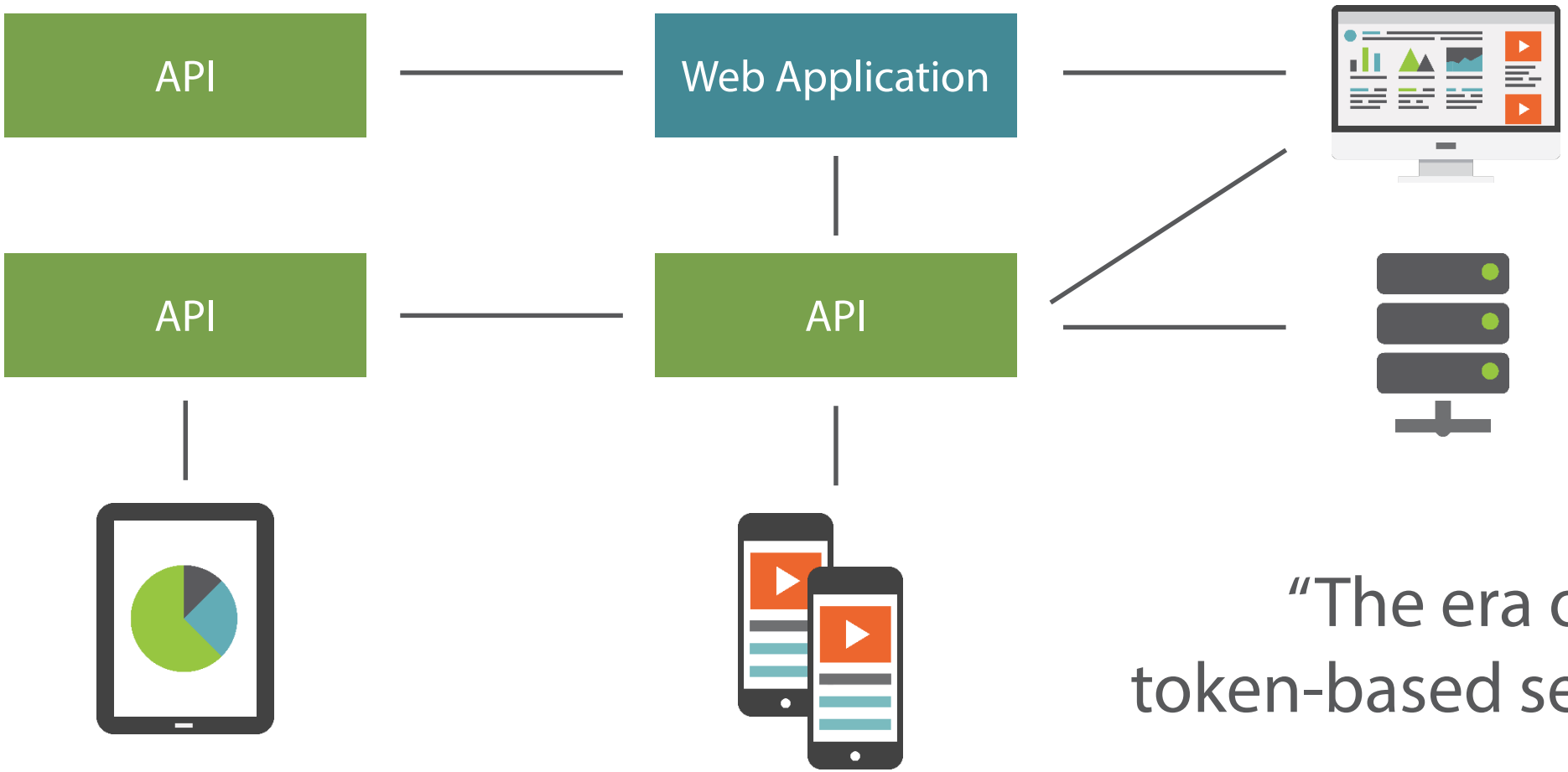
Understanding OAuth 2.0 Basics

Kevin Dockx

@KevinDockx | http://blog.kevindockx.com/

A Typical, Modern Application



"The era of token-based security"

cfr: https://identityserver.github.io/Documentation/

```
{
  "client_id": "tripgalleryimplicit",
  "scope": "gallerymanagement",
  "sub": "b05d3546-6ca8-4d32-b95c-77e94d705ddf",
  "amr": "password",
  "auth_time": 1437400047,
  "idp": "idsrv",
  "iss": "https://tripcompanysts/identity",
  "aud": "https://tripcompanysts/identity/resources",
  "exp": 1437403647,
  "nbf": 1437400047
}
```

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

OAuth 2.0 … allows secure authorization

OAuth 2.0 isn't about authentication. It's about authorization.
The standard doesn't say anything about the user.

OAuth IETF Standard: https://tools.ietf.org/html/rfc6749

… from web, mobile and desktop applications

Different types of applications require different means to achieve authorization

"Where can the token be delivered to?"

"Can the client application safely store secrets?"

OAuth 2.0 Flows

It's about making the right decision for the type of application you're building: "how can you safely achieve authorization?"

- Client Credentials
- Authorization Code
- Implicit
- Resource Owner Password Credentials

The Main Actors

Client

An application making protected resource requests on behalf of the resource owner and with its authorization

Authorization Server

The server issuing access tokens to the client after successfully authenticating the resource owner and obtaining authorization

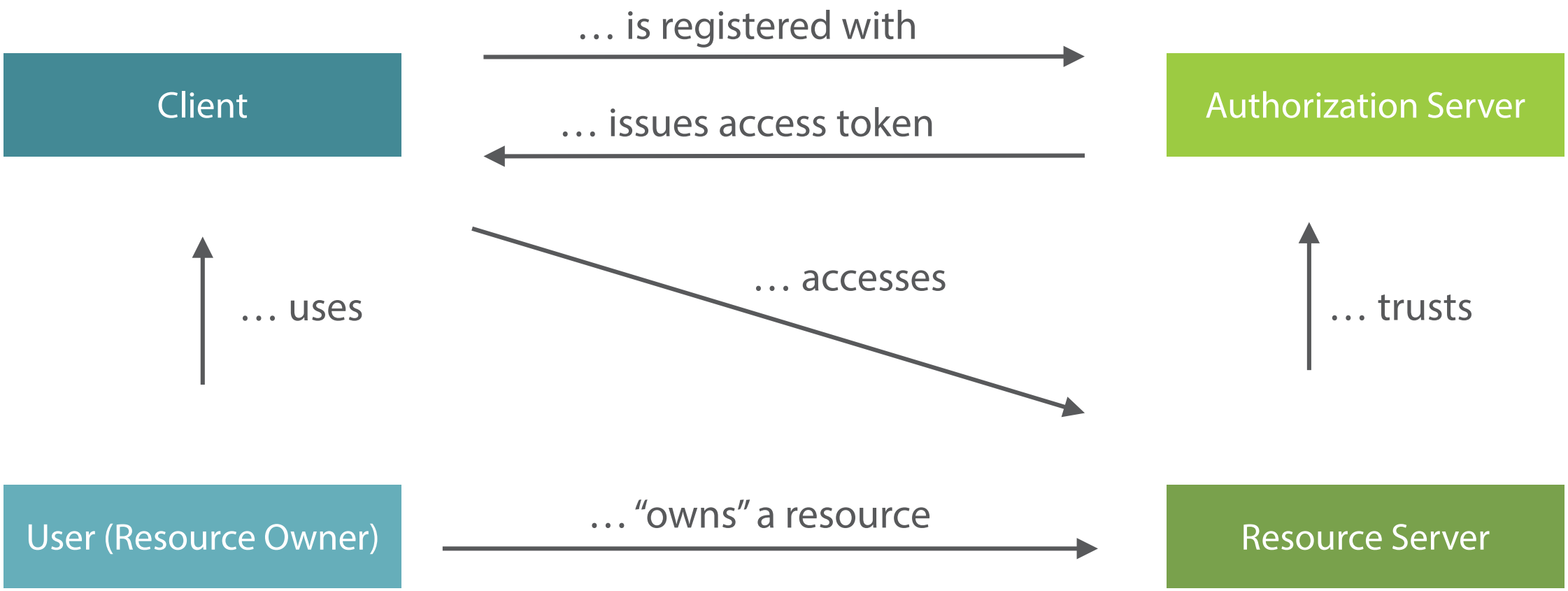
User (Resource Owner)

An entity capable of granting access to a protected resource

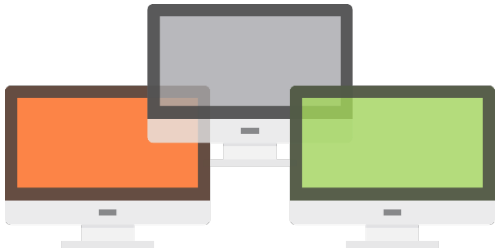
Resource Server

The server hosting the protected resources

The Main Actors



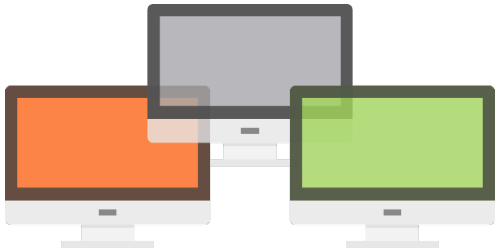
Confidential Clients



Clients capable of maintaining the confidentiality of their credentials

- Web application
  - Eg: our MVC application

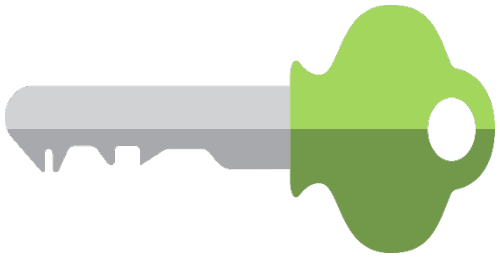
Public Clients



Clients incapable of maintaining the confidentiality of their credentials

- Native applications
  - An iOS, Android, Windows Phone app built in a native language or compile-to-native language
- User-Agent based applications
  - E.g.: JavaScript applications

Endpoints on the Authorization Server



Authorization endpoint

- used by the client to obtain authorization from the resource owner via user-agent redirection

Token endpoint

- used by the client to exchange an authorization grant for an access token, typically with client authentication

Endpoint on the Client

Redirection endpoint

- used by the authorization server to return responses containing authorization credentials to the client via the resource owner user-agent

IdentityServer v3

Created by **@leastprivilege** and **@brockallen**

Implements OAuth 2.0 and OpenID Connect

Highly optimized to solve the typical security problems of today's mobile, native and web applications

Part of the .NET Foundation

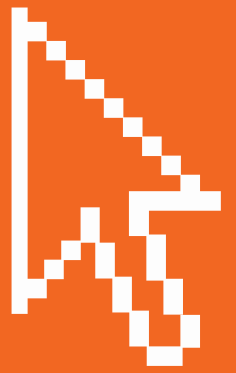
https://identityserver.github.io/

Setting up Identity Server

Learn how to set up a security token service

Requiring Authorization for API Access

Learn how to ensure our API isn't publicly accessible anymore



Summary

OAuth 2.0 allows secure **authorization** from web, mobile and desktop applications

Its main actors are the **resource owner**, **client**, **resource server** and **authorization server**

It defines **confidential** and **public** clients

It defines a **token**, **authorization** and **redirection** endpoint

IdentityServer implements OAuth 2.0 and OpenID Connect