Impersonating the User When Accessing the API

Kevin Dockx

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

Impersonating the User

[gallerymanagement]

"I can access the update method because I'm in the correct role"

"I can get my private trips, because the API knows who I am"

"I can delete the pictures I added, because the API knows who I am"

CLIENT

API

Ensuring the Access Token Contains the Scope We Need

Learn how to ensure the correct scope is included in the access token

Extending a Selection at API Level Based on the User

Learn how to ensure the correct resources are returned to the client thanks to impersonation

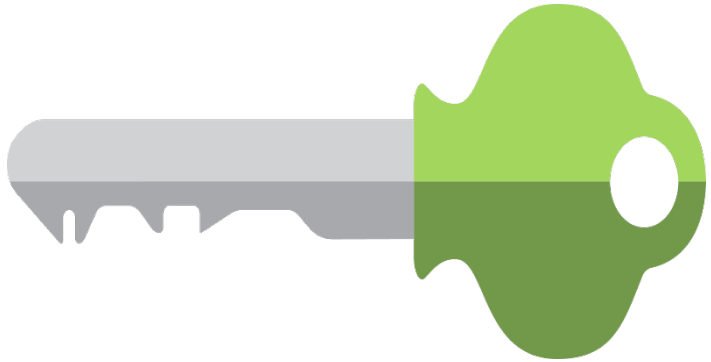
Blocking Functionality at API Level Based on the User

Learn how to ensure a user can only delete the pictures (s)he created, at API level, thanks to impersonation

Role-Based Authorization

We're not authorizing access to actions depending on the user
(due to functional requirements)

We can include additional claims in the access token and use those for authorization
A role claim is a good example, and allows role-based authorization



Role-Based Authorization

Learn how to authorize access to API resources depending on the user's role

Reusing Claims Across Scopes

Learn how to reuse the role claim across scopes by creating a custom identity scope

Summary

Use claims from the ClaimsIdentity at API level to know who the user is

Add an additional role scope to the access token to enable role-based authorization

Claims can be reused across scopes