Fluidez de la atención

José Berretta Moreno
@cocheok

AGENDA

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

- Autenticación con usuario y contraseña: reglas generales.
- Autenticación que no requiere contraseña.
- **Autenticación y autorización.**
  - **Open Authorization (OAuth)**
    - Tokens
    - OAuth 2.0
      - Consideraciones de seguridad
      - Covert Redirect

AUTENTICACIÓN CON USUARIO Y CONTRASEÑA: REGLAS GENERALES

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

- El ID de usuario no tiene que ser sensible a mayúsculas y minúsculas.
- Implementar controles adecuados de fortaleza de contraseña.
- Implementar un mecanismo seguro de recuperación de contraseña.
- Almacenar contraseñas de forma segura.
- Transmitir contraseñas solo sobre TLS u otro transporte fuerte.
- Solicitar volver a autentificarse para funciones sensibles.
- Utilizar la autenticación por múltiples factores.
- Manejar los mensajes de error en uno solo.
- Prevenir ataque por fuerza bruta.

AUTENTICACIÓN QUE NO REQUIERE CONTRASEÑA

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

AUTENTICACIÓN Y AUTORIZACIÓN

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

AUTENTICACIÓN: consiste en un sistema para certificar que el usuario es quien dice ser.

AUTORIZACIÓN: consiste en dar acceso a una serie de recursos a un usuario o sistema.

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Es un protocolo de **autentiticación** basado en HTTP que utiliza proveedores de identidad para validar que un usuario es quien dice ser. Es un protocolo muy simple que permite a un proveedor de servicios de identidad, un camino para el inicio de sesión único (SSO, por las siglas en ingñés de “singles sign-on”). Esto permite a los usuarios reutilizar una sola identidad dada a un proveedor de identidad OpenID de confianza y ser el mismo usuario en múltiples sitios web, exceptuando al proveedor de identidad OpenID.

Para entornos no empresariales, OpenID es considerado seguro y frecuentemente, la mejor opción, siempre y cuando el proveedor de identidad sea de confianza.

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

SAML a menudo se considera la competencia de Open ID. La versión más recomendada es la 2.0, ya que poseé características muy completas y proporciona gran seguridad.

Como con OpenID, SAML utiliza proveedores de identidad, pero a diferencia de este basado en XML proporciona mayor flexibilidad. SAML está basado en redirecciones del navegador, las cuales envian los datos en formato XML. A diferencia de SAML, OpenID no solo es iniciado por un proveedor de servicios, sino que también puede ser iniciado desde el proveedor de identidad. Esto permite al usuario navegar entre diferentes portales mientras que se mantiene autentificado sin tener que hacer nada, haciendo que el procesos sea transparente.

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

La Fast Identify Online (FIDO) Alliance ha creado dos protocolos para facilitar la autenticación online:

- Universal Authentication Framework (UAF): Está enfocado en la autenticación sin contraseña.

- Universal Secont Factory (U2F): Permite la adición de un segundo factor de autenticación basado en contraseñas existentes.

AMBOS PROTOCOLOS ESTÁN BASADOS EN UNA LLAVE PÚBLICA DE MODELO CRIPTOGRÁFICO DESAFIO - RESPUESTA.

UAF

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

- Toma ventaja de las tecnologias de seguridad existentes presentes en los dispositivos de autenticación, incluyendo sensores de huellas digitales, biométricos y otros.

- El protocolo está diseñado para conectar las capacidades de este dispositivo en un marco de autenticación común.

- Tabaja con aplicaciones nativas y web.

U2F

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

- Aumenta la autenticación basada en contraseñas mediante un token de hardware (típicamente un usb) que almacena llaves de autenticación criptográficas y las utiliza para firmar. El usuario puede utilizar el mismo token como un segundo factor para múltiples aplicaciones.

- U2F trabaja con aplicaciones web. Proveé protección contra phishing utilizando la URL del sitio web para buscar la llave de autenticación almacenada.

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Es un protocolo abierto, que permite autorización segura de un API de modo estándar y simple para aplicaciones de escritorio, web y mobile.

Permite a los usuarios compartir sus recursos privados (por ejemplo: fotos, videos, listas de contactos, etc) almacenados en un determinado sitio con otro sitio sin tener que entregar sus credenciales.

En lugar de las credenciales normales (nombre de usuario y password) envia un token.

¿Qué es un Token?

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

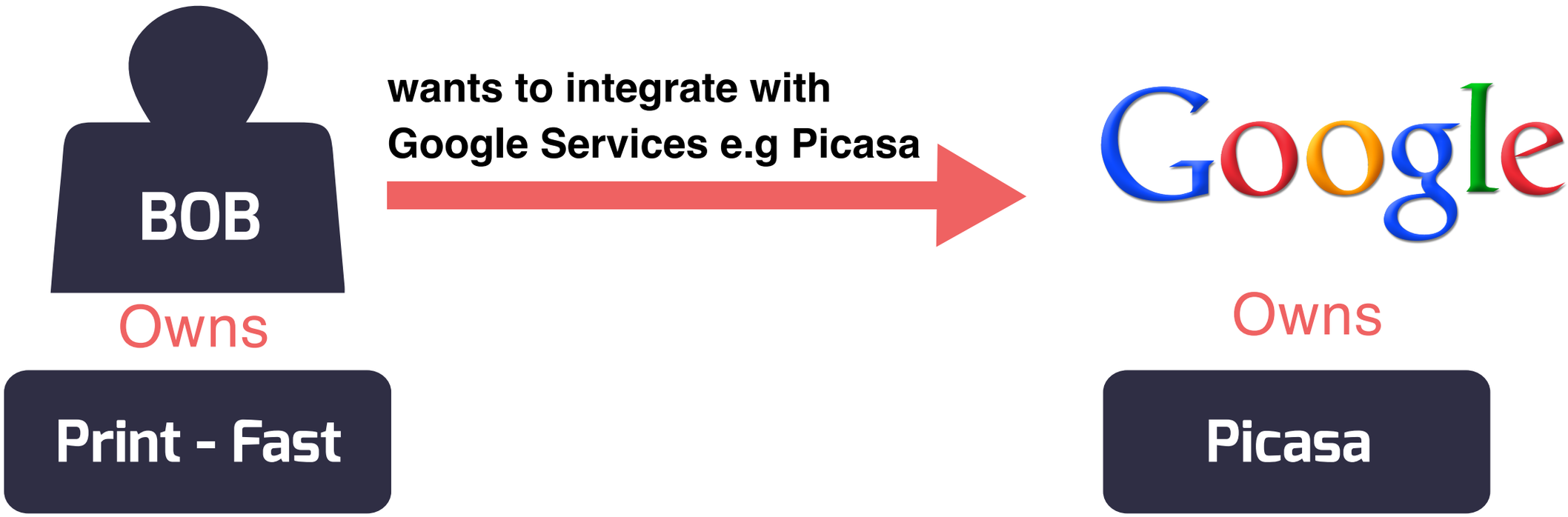
CADENA DE CARACTERES QUE TIENE LA INFORMACIÓN NECESARIA PARA LA IDENTIFICACIÓN DE UN USUARIO.

Cada token garantiza el acceso a un sitio específico para recursos específicos y para una duración determinada.

OAuth 2.0

ESCENARIO

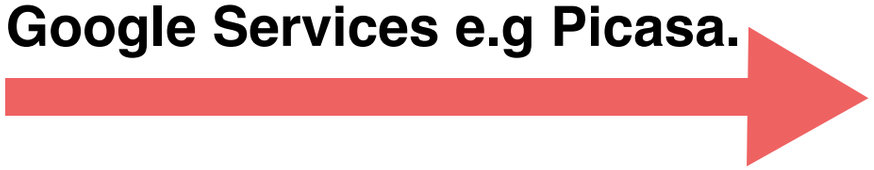
José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

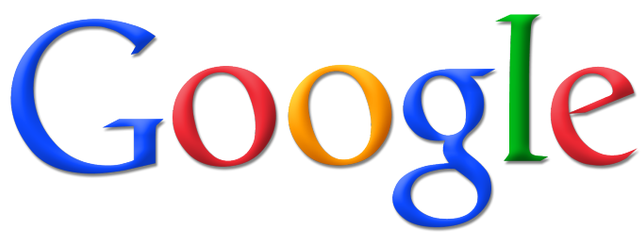


ROLES

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Wants to integrate with Google Services e.g Picasa.





Authorization Server

Print - Fast

Client

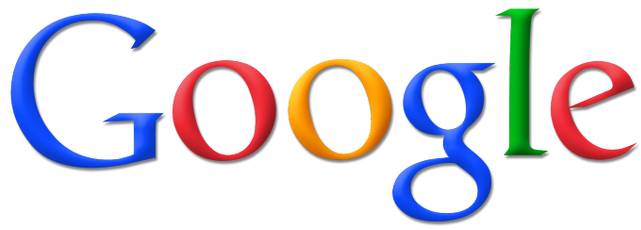
Picasa

Resource Server

ROLES

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Wants to integrate with Google Services e.g Picasa.



Authorization Server

Client

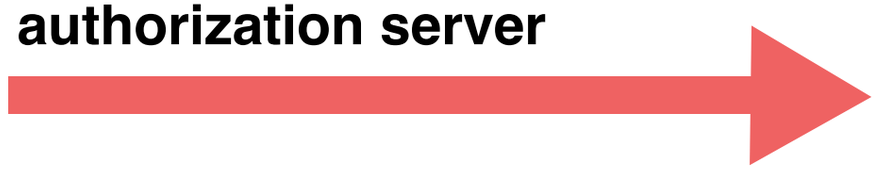
Resource Owner

Resource Server

CLIENT REGISTRATION

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Client register with authorization server



client_Id= print-fast
client_secret=xxx
redirect_uri=http://print-fast.com

Authorization Server

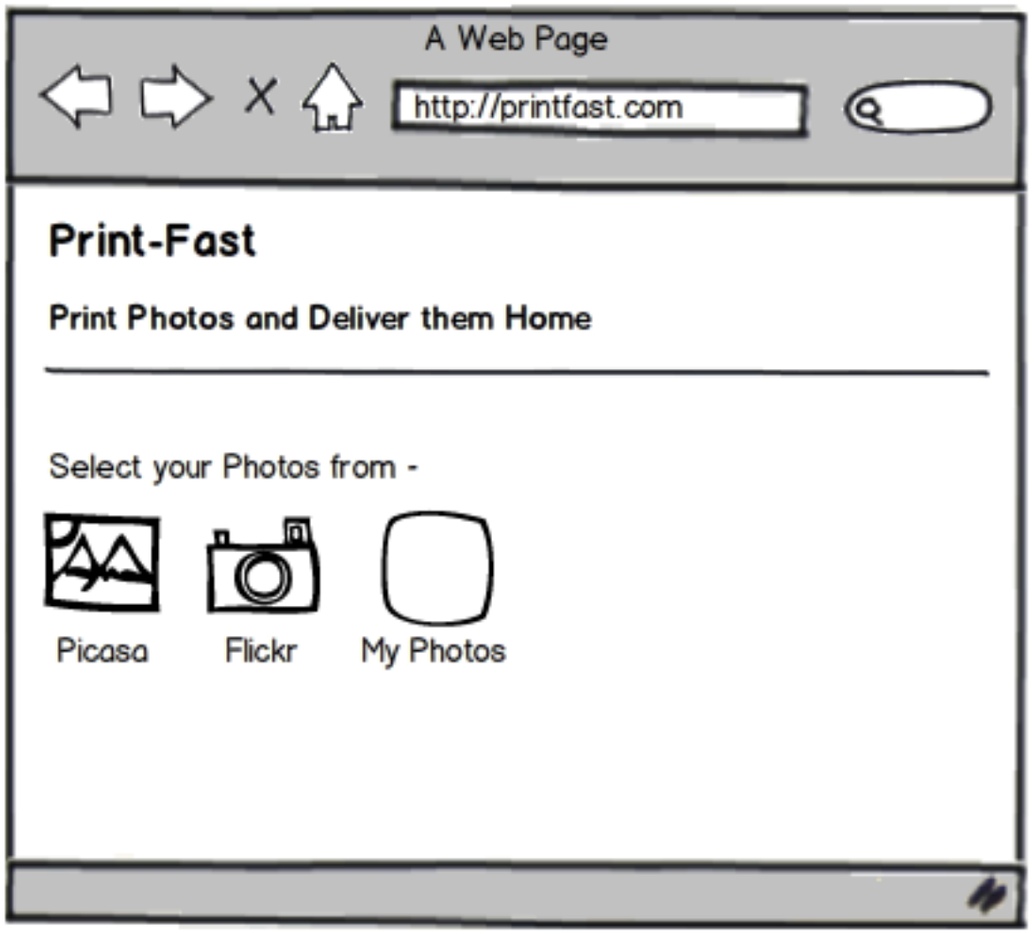
Client

Resource owner

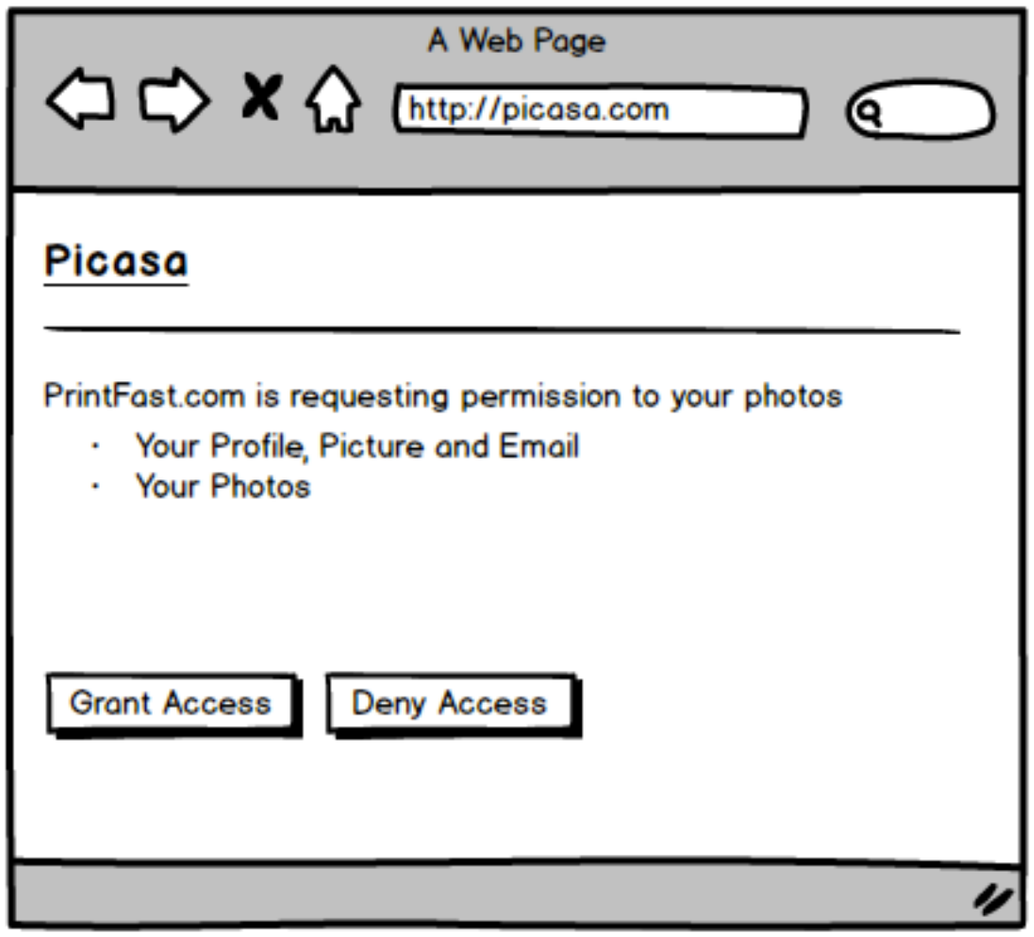
Resource Server

Flujo 1 - Obtener Authorization Grant

Authorization Request



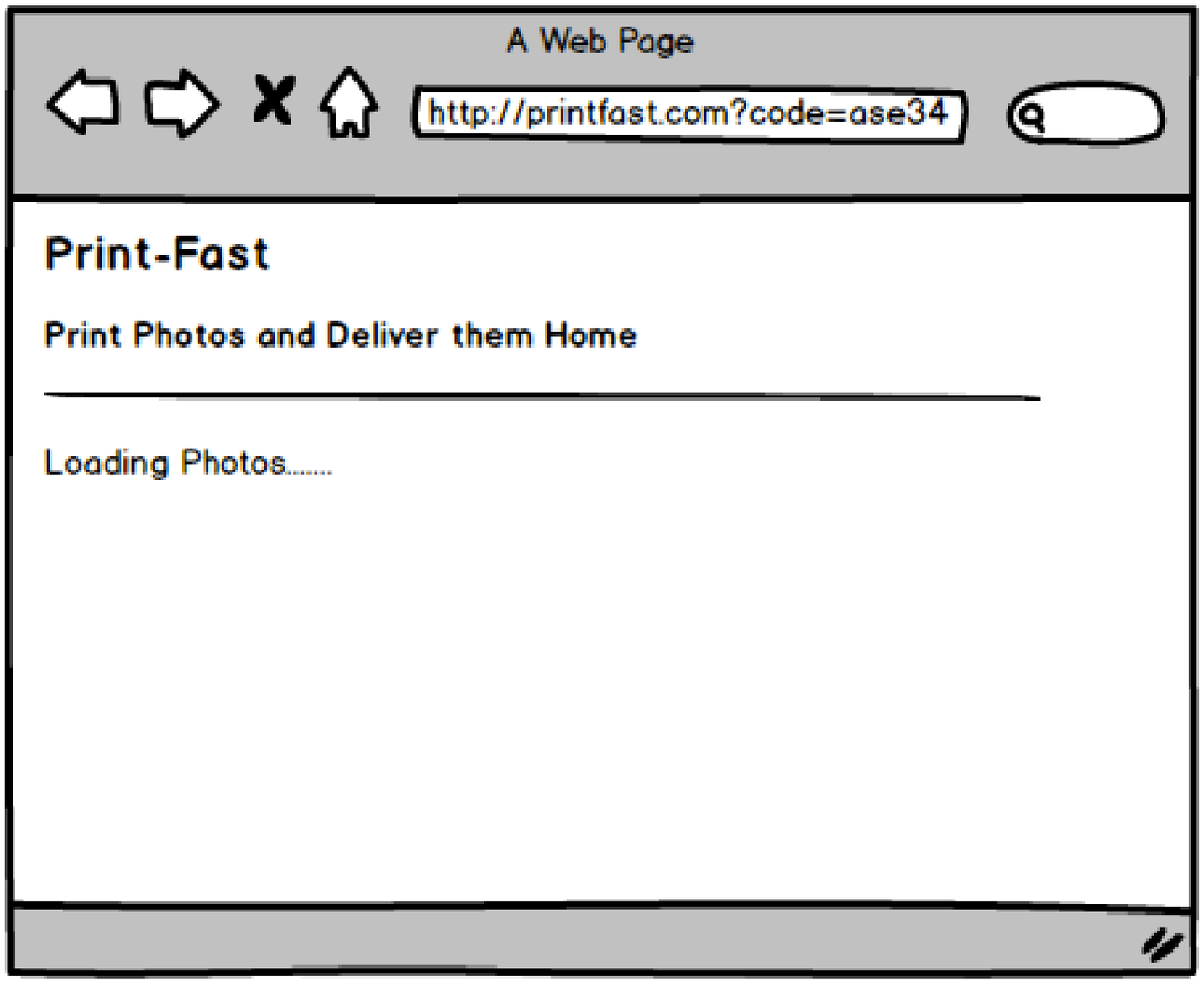
Authorization Grant



URL utilizada

http://picasa.com/?client_id=photo-fast&scope=profileemail,-photos&redirect_uri=http://print-fast.com&response_type=code&**state=aLo8fG3567a**

Authorization Grant



Code= ase34&state=aLo8fG3567a

Authorization Request

http://picasa.com/?client_id=photo-fast&scope=profile,email,photos

Authorization Grant

code= **ase34**

Representa una autorización exitosa de Alice

Client

Print - Fast

Resource Owner

Authorization Server

Resource Server

Flujo 2 - Obtener el Access Token

Client
Print - Fast

Authorization Grant

code= **ase34**
client_Id=print-fast
client_secret=xxx

Access Token

access_token=x3e4

Alice

Resource Owner

Google

Authorization Server

Resource Server

Representa la autorización que le da Alice al client, para acceder a sus recursos protegidos.

Flujo 3 - Acceder a los recursos protegidos

Alice

Resource Owner

Google

Authorization Server

Access Token

access_token=x3e4

Client

Print - Fast

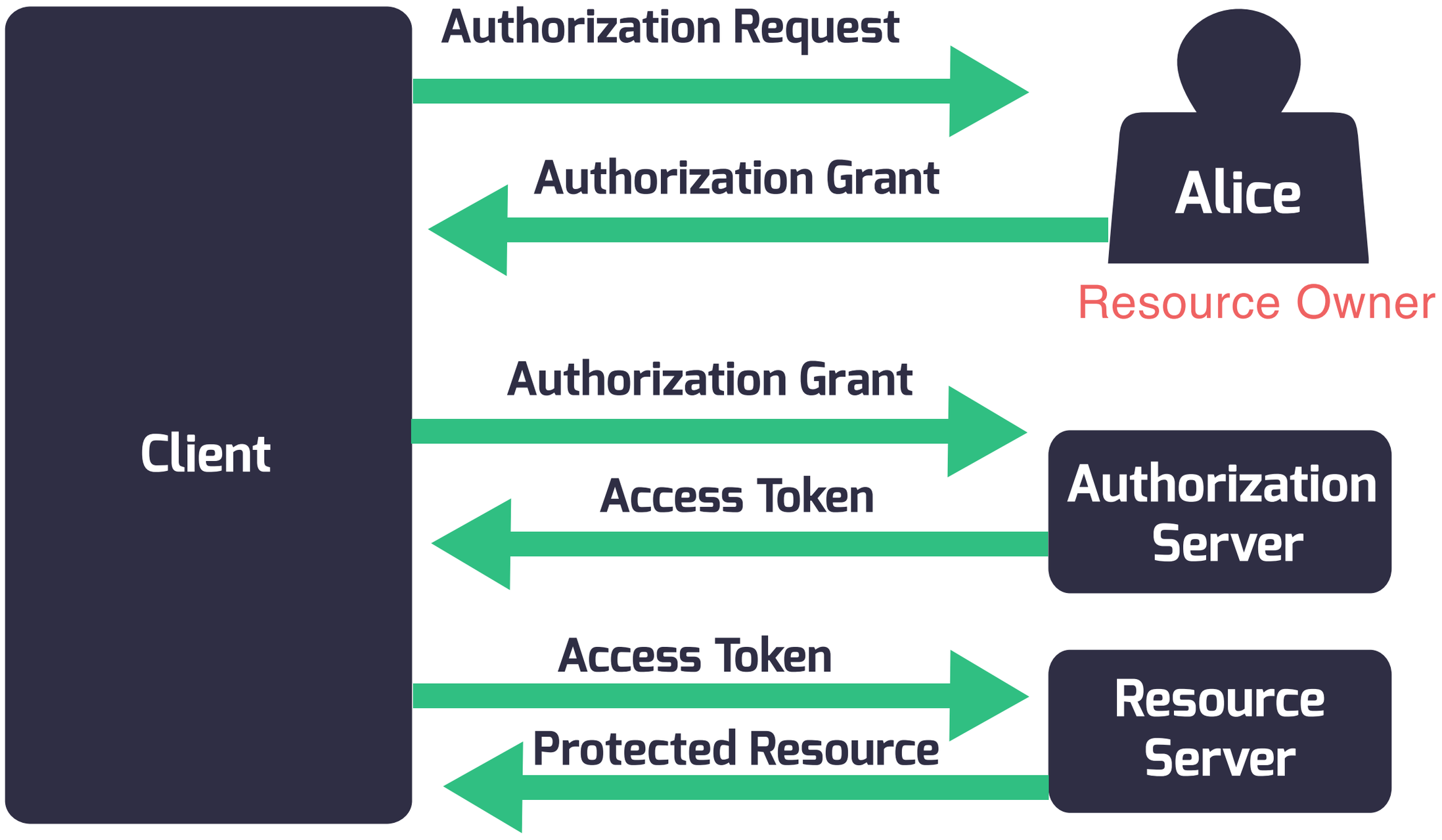
Picasa

Protected Resource

Resource Server

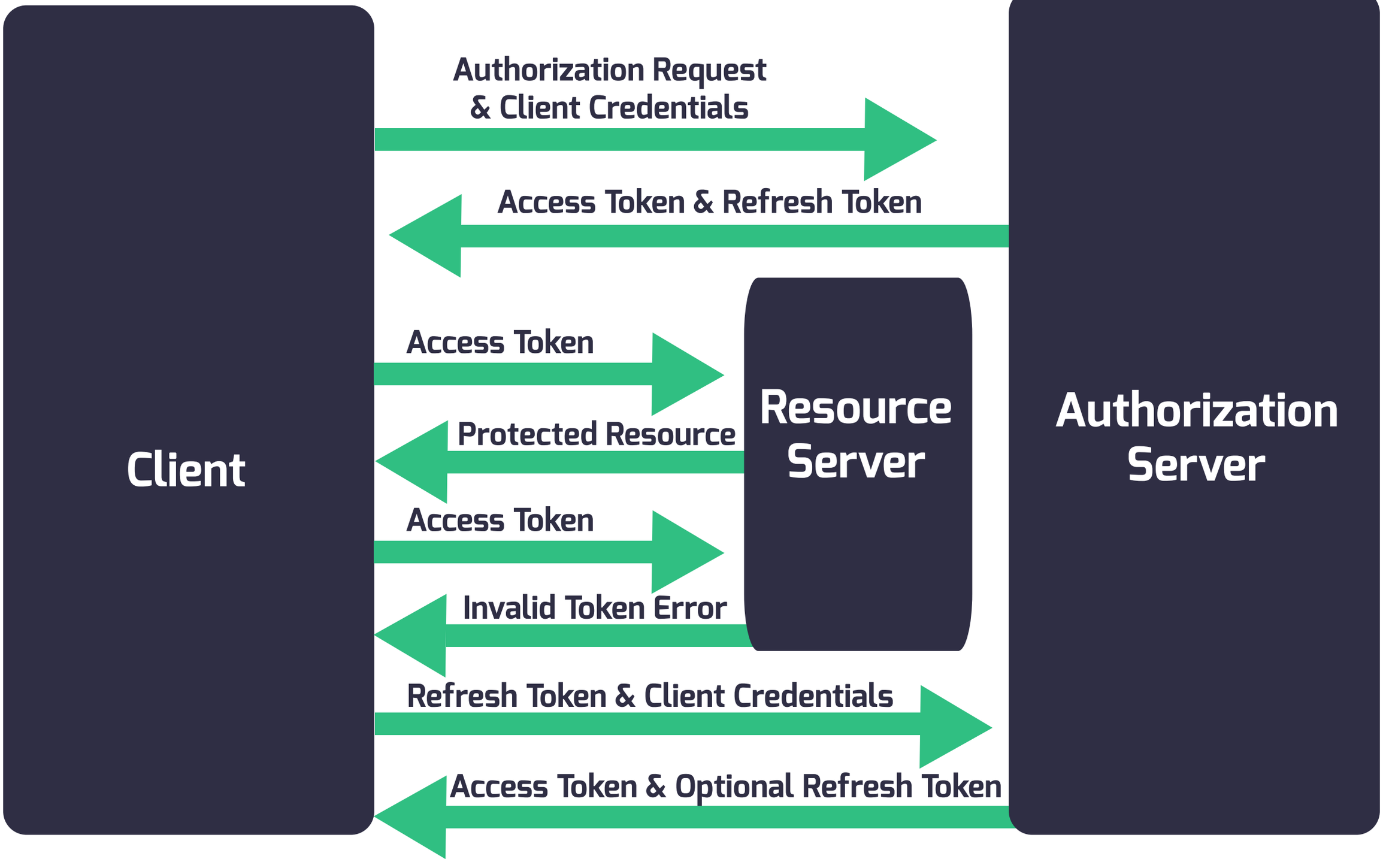
[“http://.../DSC34.jpg”, “http://.../DSC44.jp-g”,“http://.../DSC56.jpg”,“http://.../D-SC98.jpg”]

Flujo completo



Protocol Flow

Refresh Token


Client
Authorization Request
& Client Credentials
Access Token & Refresh Token
Access Token
Protected Resource
Resource
Server
Access Token
Invalid Token Error
Refresh Token & Client Credentials
Access Token & Optional Refresh Token
Authorization
Server

OAuth2

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Si alguién intercepta o captura el Access Token, ¿Podría alguién usarlo para solicitar información del usuario?

OAuth2

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Si alguién intercepta o captura el Access Token, ¿Podría alguién usarlo para solicitar información del usuario?

SI

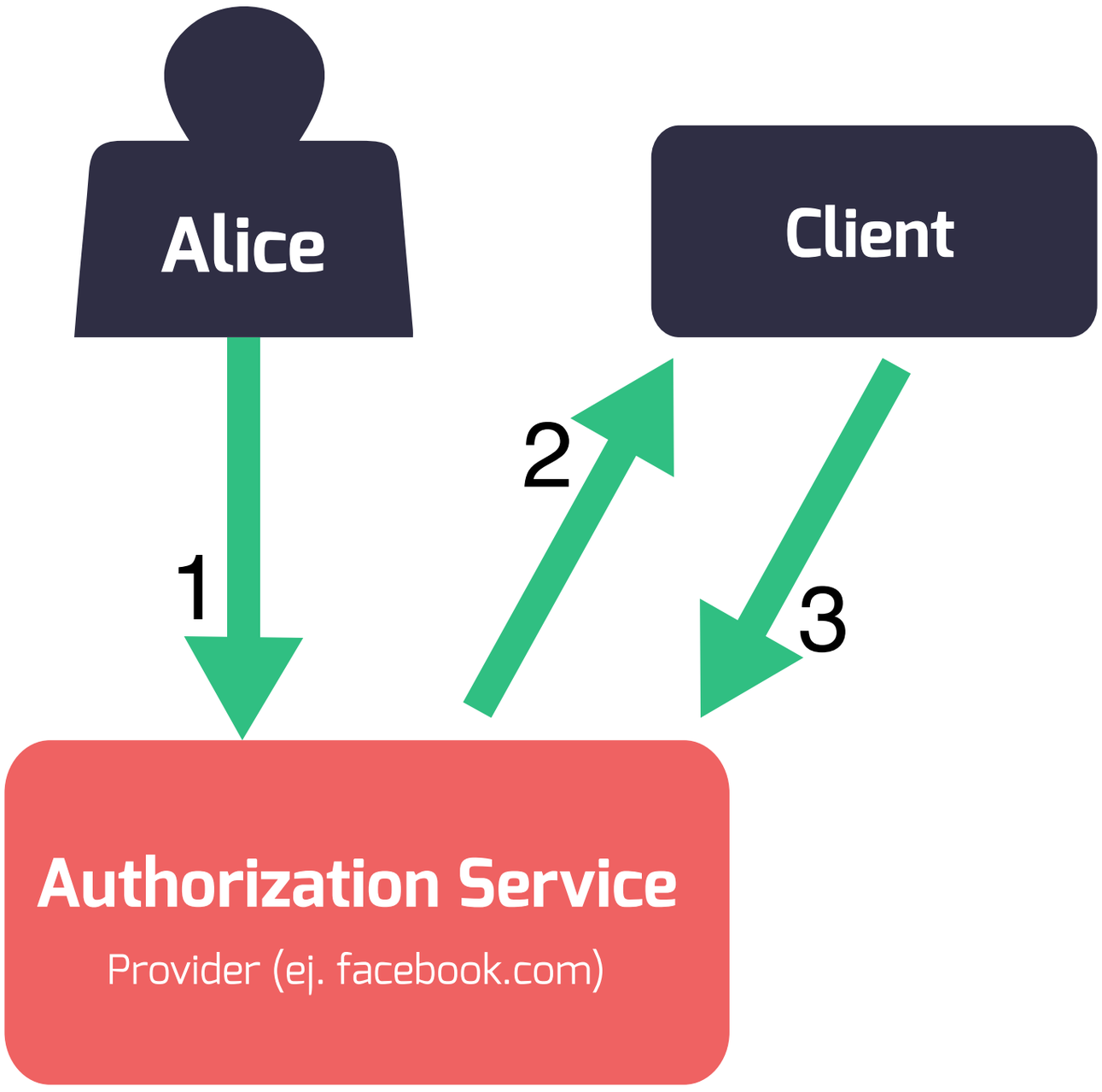
CONSIDERACIONES DE SEGURIDAD: cliente

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

- **Utilizar SSL para la comunicación**
- **No almacenar las credenciales en el código.**
- **Agregar el parámetro “state”:** evita los ataques de tipo CSRF (Cross-site request forgery o falsificaciones de solicitud entre sitios).
- **Limitar los tokens en alcance y duración:** depende del tipo de aplicación y el tipo de token, mientras menor duración tanto en tiempo como en uso tenga el token, menor es la probabilidad de que caiga en manos equivocadas.

COVERT REDIRECT: flujo normal

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com



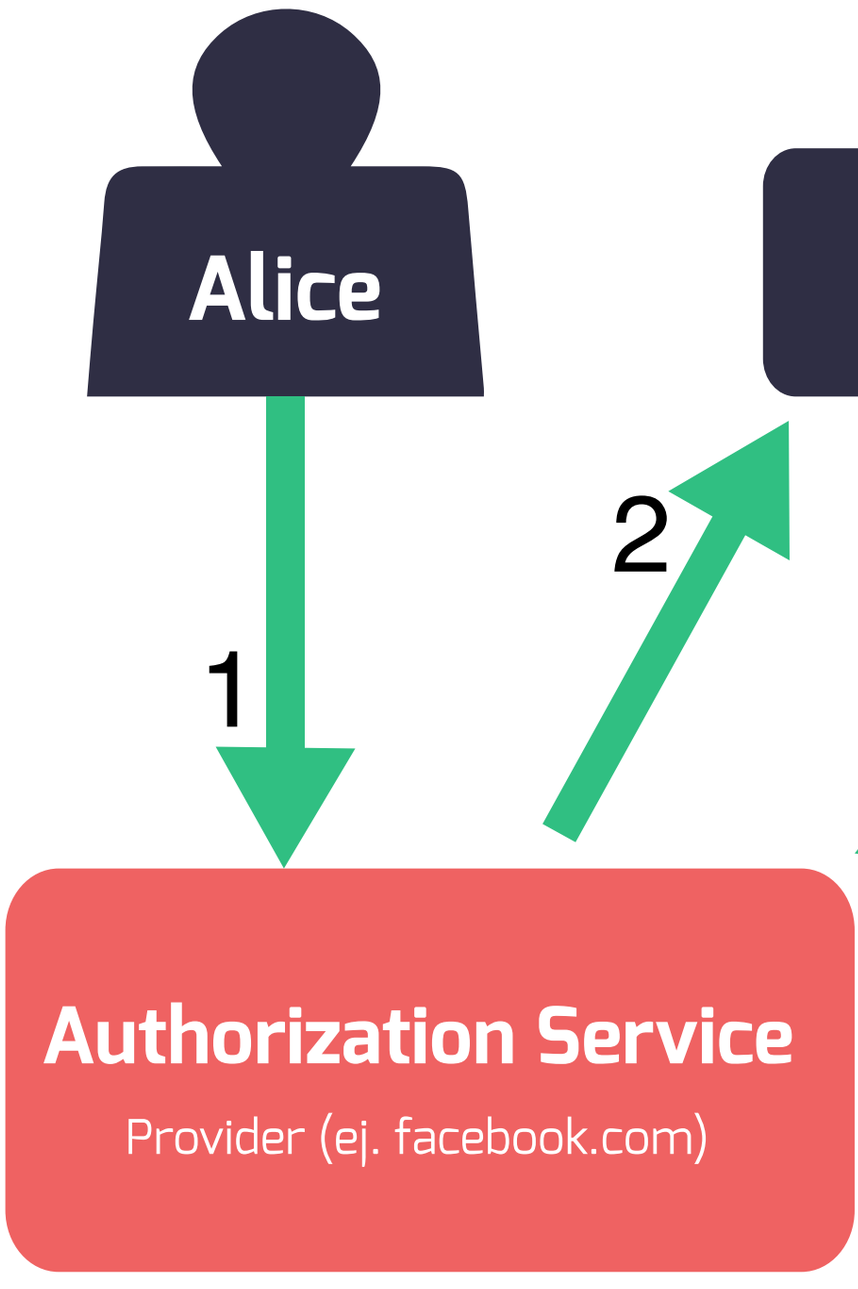
1- El cliente registrado solicita acceso un Authorization Server.

2- El usuario se loguea.

3- Luego de una verificación el cliente recive un Authorization Token y lo usa para acceder a su información.

COVERT REDIRECT:
flujo atacado

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com



3- Al resetear los valores de **redirect_uri**, el atacante ahora redirecciona al usuario al sitio malicioso.

4- El atacante puede ahora tener acceso a los datos del usuario utilizando el token.

COVERT REDIRECT: mitigación del riesgo

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

Agregar un whitelist de los sitios a redireccionar.

REFERENCIAS

José Manuel Berretta Moreno
@cocheok
jberrettamoreno.com
talkwithme@jberrettamoreno.com

OAuth: https://tools.ietf.org/html/rfc5849

OAuth2: https://tools.ietf.org/html/rfc6749

Consideraciones de seguridad OAuth2:
http://tools.ietf.org/html/rfc6819

Covert redirect: http://tetraph.com/covert_redirect/

OWASP Authentication: https://www.owasp.org/index.php/Authentication_Cheat_Sheet_Espa%C3%B1ol