Universidad de Oviedo

Departamento de Informática

Campus de Gijón

Autenticación de Usuarios (Red)

Presentación

Daniel F. García

Conexión remota y Sistemas distribuidos

Diferenciar diversos conceptos

- Autenticación local en un computador integrado en una red (típico en Windows)

  El usuario se autentica con una cuenta de Active Directory

- Autenticación remota en un computador

  El usuario desea tener acceso remoto a un computador

  Típico en UNIX → Secure Shell (SSH) + PuTTY

  Típico en Windows → Remote Desktop Protocol (RDP)

  Tecnología de escritorio remoto

- Autenticación de usuarios en sistemas distribuidos (red)

  El usuario se autentica una sola vez y
  obtiene acceso a múltiples recursos

Kerberos: Introducción

Kerberos es un protocolo de autenticación de usuarios para sistemas distribuidos

SD = Conjunto de workstations y servidores funcionando en la misma red

Escenario de funcionamiento https://www.kerberos.org/

Los usuarios de las workstations (clientes) desean acceder a servicios proporcionados por los servidores de la red

Kerberos implementa una **estrategia centralizada** para autenticar los clientes a los servidores (y opcionalmente los servidores a los clientes)

En vez de gestionar independientemente la autenticación de cada cliente con cada uno de los servidores

La gestión independiente es muy engorrosa si hay muchos clientes/servidores

Evolución https://web.mit.edu/kerberos/

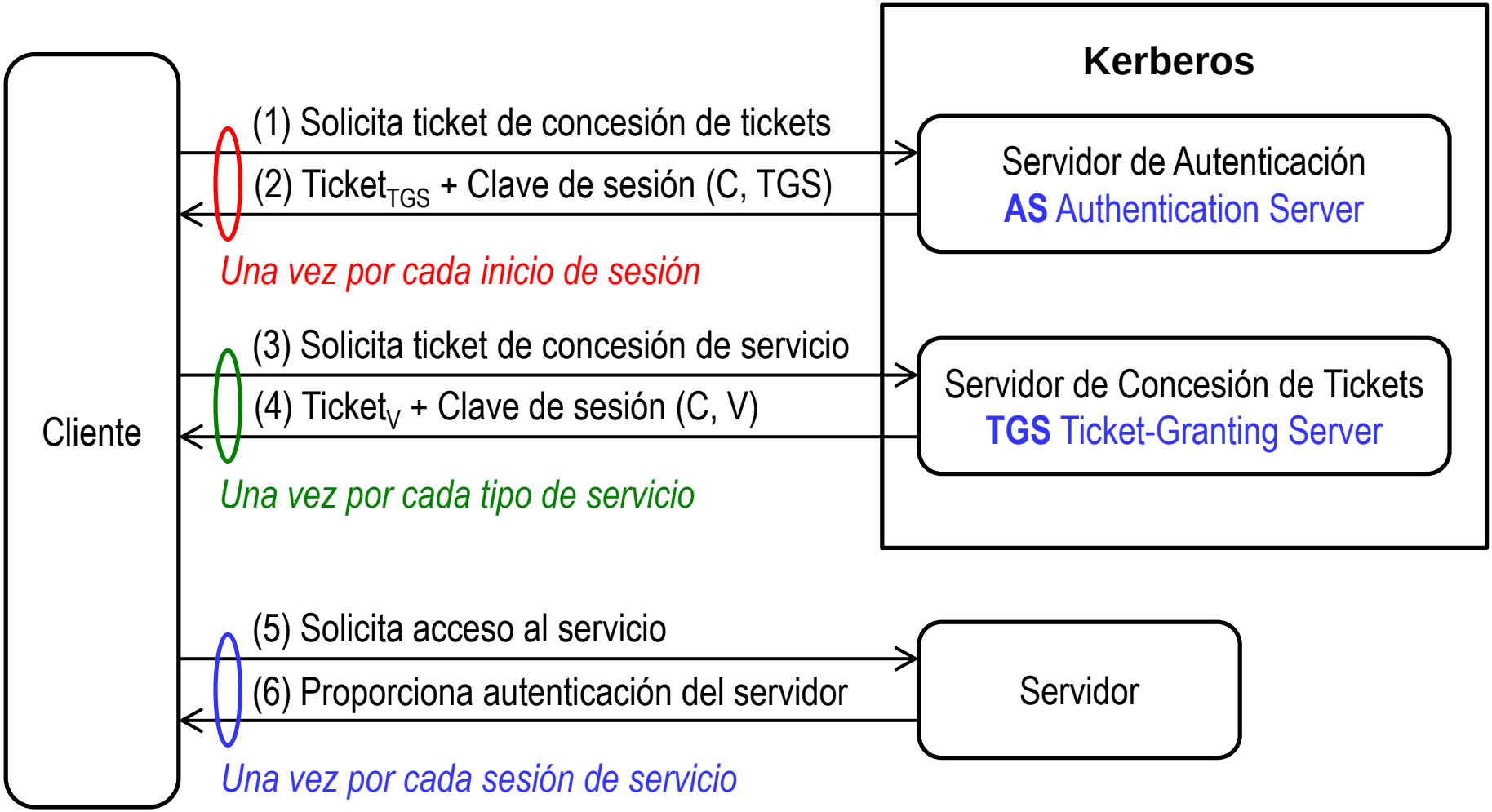
Versiones 1-3: Se desarrollaron en el MIT para el proyecto Athena (1983-1991)

Versión 4: Apareció en 1988 como el sistema de autenticación de Athena

Versión 5: Apareció en 1993 (RFC 1510) y se actualizó en 2005 (RFC 4120)

Ultima Versión 5 → krb5-1.21.2 de 15-agosto-2023

Kerberos: Visión general



Kerberos: Intercambio de mensajes en la **Versión 4**

Intercambio para obtener el ticket de concesión de tickets

(1) C → AS: ID_C ID_{TGS} TS_1

(2) AS → C: $E(K_C\ [K_{C,TGS}\ ID_{TGS}\ TS_2\ Lifetime_2\ Ticket_{TGS}])$

$Ticket_{TGS} = E(K_{TGS}\ [K_{C,TGS}\ ID_C\ AD_C\ ID_{TGS}\ TS_2\ Lifetime_2])$

Intercambio para obtener el ticket de concesión de servicio

(3) C → TGS: ID_V $Ticket_{TGS}$ $Authenticator_{C,TGS}$

$Authenticator_{C,TGS} = E(K_{C,TGS}\ [ID_C\ AD_C\ TS_3])$

(4) TGS → C: $E(K_{C,TGS}\ [K_{C,V}\ ID_V\ TS_4\ Ticket_V])$

$Ticket_V = E(K_V\ [K_{C,V}\ ID_C\ AD_C\ ID_V\ TS_4\ Lifetime_4])$

Intercambio para obtener el acceso al servicio

(5) C → V: $Ticket_V$ $Authenticator_{C.V}$

$Authenticator_{C,V} = E(K_{C,V}\ [ID_C\ AD_C\ TS_5])$

(6) V → C: $E(K_{C,V}\ [TS_5 + 1])$ para autenticación mutua

C = Client
V = Server
AS = Authentication Server
TGS = Ticket-Granting Server

Kerberos: Autenticación entre dominios

Un **dominio** Kerberos (*Kerberos realm*)
es un conjunto de nodos (clientes y servidores)
gestionados por el mismo servidor Kerberos

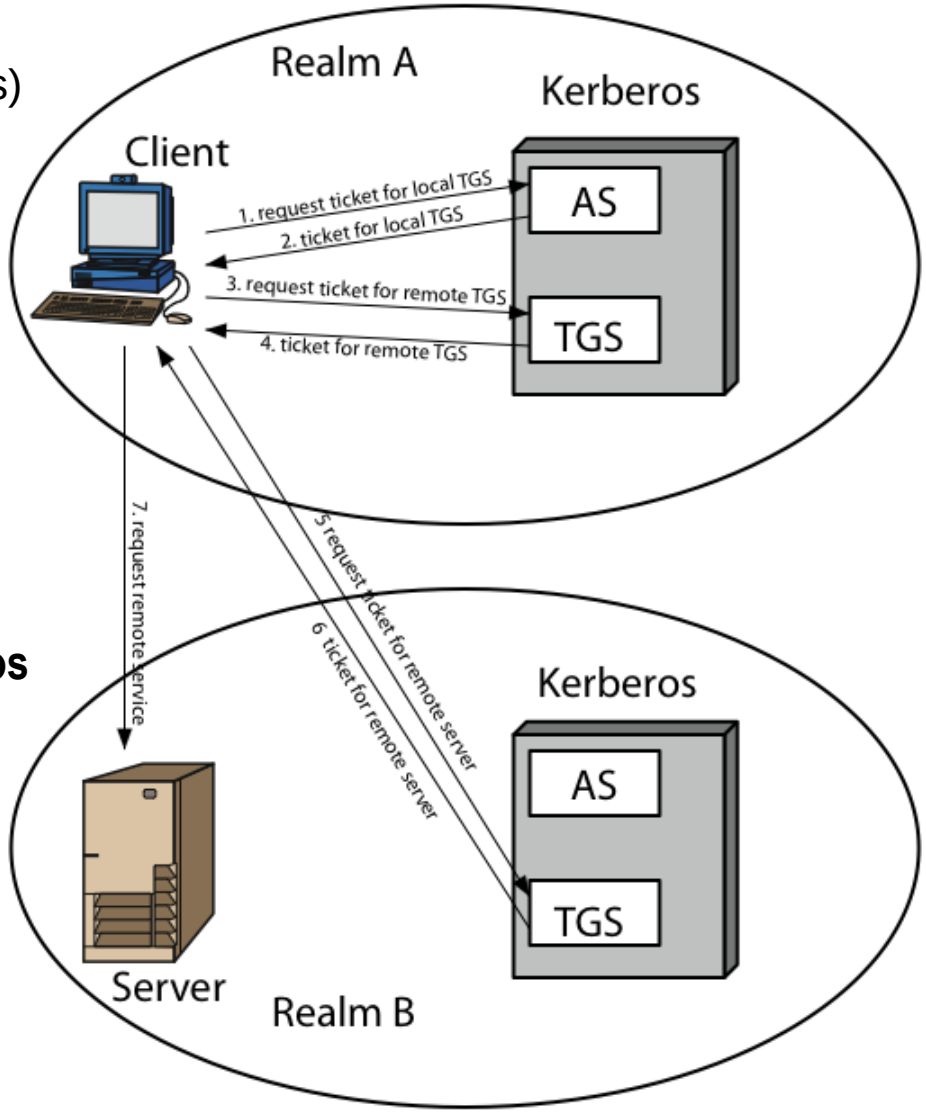
Un **principal** (*Kerberos principal*)
es cualquier usuario o servicio conocido
Se identifican por el nombre del principal

Se combinan nombres: Dominio + Principal

Kerberos soporta autenticación entre dominios

Cada servidor Kerberos …
- Comparte una clave secreta
- Está registrado

Con los otros servidores



Inicio de Sesión Único (SSO)

El Inicio de Sesión Único (*Single-Sign-on, SSO*) es una propiedad o característica de un sistema o procedimiento de control de acceso que permite a un usuario acceder a múltiples aplicaciones/sistemas/recursos autenticándose una sola vez

Se puede hablar también de Fin de Sesión Único (*Single-Sign-off*) como la propiedad de que al terminar una sesión se elimina el acceso a todos los sistemas

BENEFICIOS:

- Reduce el **número** de contraseñas que tiene que utilizar un usuario (*password fatigue*)
- Reduce el **tiempo** empleado reintroduciendo contraseñas para una misma identidad
- Reduce los **costes** de explotación → Menos asistencias para problemas con contraseñas

PROBLEMAS:

- El robo de una contraseña de un sistema SSO da acceso a muchos sistemas → Hay que proteger muy bien las contraseñas (en general, las credenciales del usuario)
- La falta de disponibilidad del sistema SSO impide el acceso a todos los sistemas cuyo acceso controla → El sistema SSO es un elemento crítico

Los sistemas SSO utilizan servidores de autenticación centralizados que usan todos los otros servidores/aplicaciones para propósitos de autenticación

Ej. Kerberos

Inicio de Sesión Web Único (Web-SSO)

El Inicio de Sesión Web Único (*Web Single-Sign-on, Web-SSO*) consiste en:

> El usuario se autentica en un servidor (sitio) web que le proporciona una cookie (≈token) de autenticación con la que puede acceder a múltiples servicios (sitios) web que aceptan la cookie sin re-autenticarse

Si el usuario intenta acceder a un sitio web sin autenticarse es redirigido automáticamente al servidor web de autenticación centralizada

> El concepto Web-SSO es simplemente el concepto SSO para aplicaciones y servicios a los que se accede mediante navegadores web

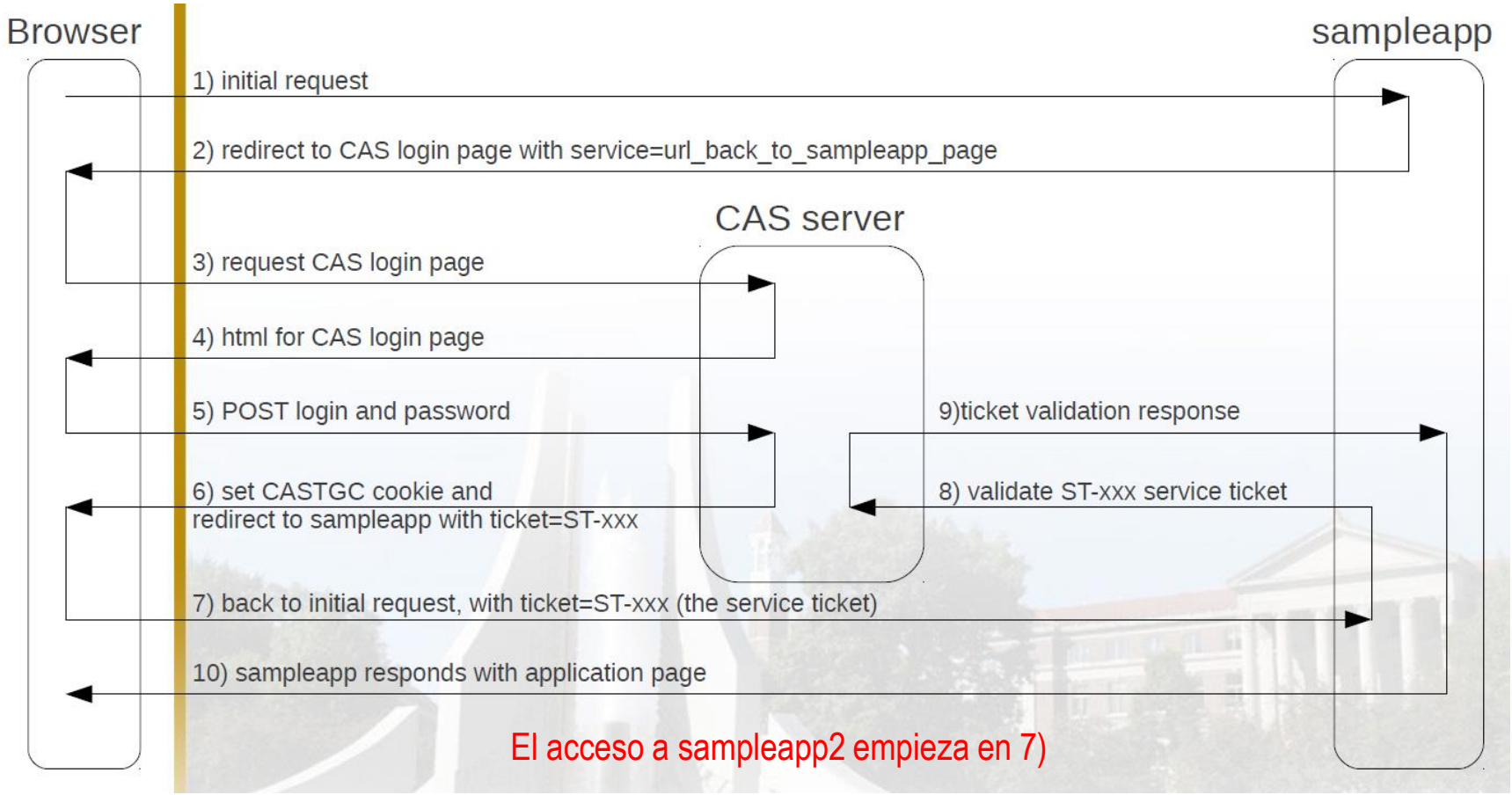
Al aparecer el concepto de Web-SSO …
Al control de acceso centralizado a servicios NO web se le denomina …
Enterprise SSO (E-SSO) ó Legacy SSO

Inicio de Sesión Web Único (Web-SSO): CAS

Ejemplo de Web-SSO → CAS (*Central Authentication Service*) open source

Desarrollado inicialmente en la Universidad de Yale https://www.apereo.org/projects/cas

En diciembre de 2004 su desarrollo pasó a JASIG (Java Architectures Special Interest Group)



CAS Server puede comprobar username / password en una BD de Kerberos o Active Directory

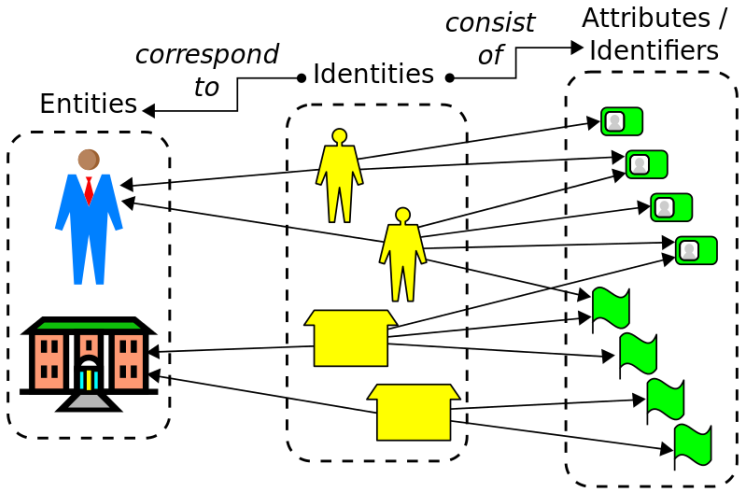
Gestión de identidades digitales

La gestión de identidad (Identity Management, IdM) es la tarea de controlar la información sobre los usuarios de los computadores de una o múltiples organizaciones

Esta información incluye:

- La que autentica la identidad del usuario
- La que indica que información y acciones está autorizado el usuario
- La que describe al usuario
- La que describe quien y como puede modificarla

Además de usuarios también se gestionan identidades de dispositivos, aplicaciones, servicios, etc.
En general se gestiona la identidad de “entidades”



Una entidad tiene múltiples identidades digitales

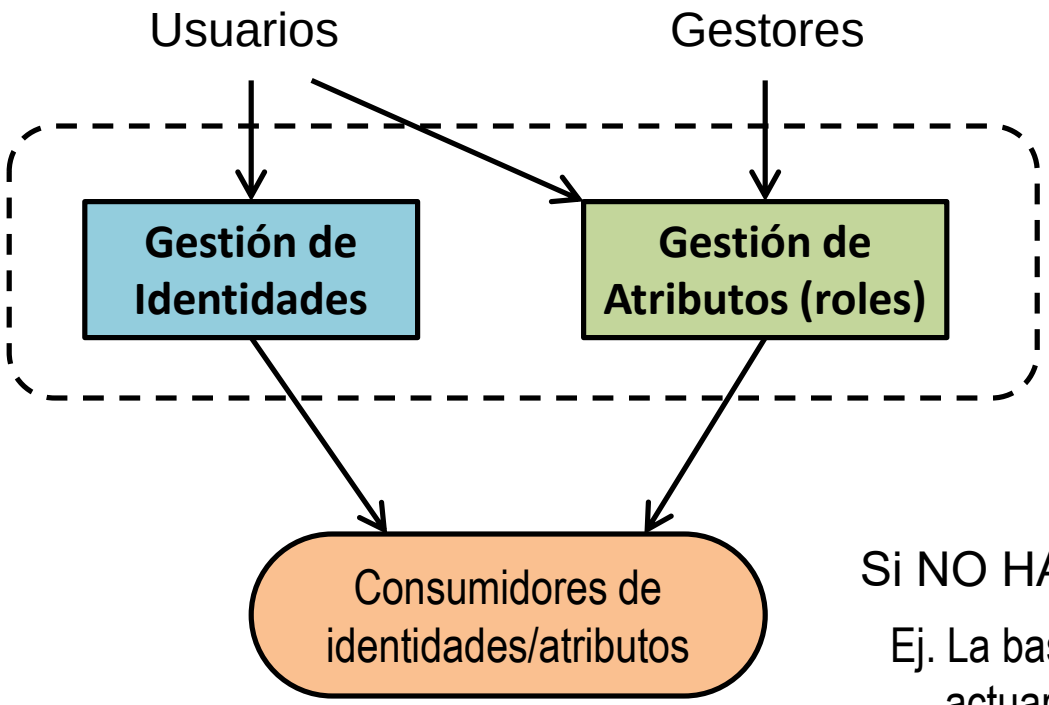
Un usuario { Tiene cuenta en varios computadores
Está registrado en muchos sitios web

Cada identidad consta de múltiples atributos
¿Consistencia identidad ←→ atributos?

Proveedores de Identidad Digital (Corporativos)

Para gestionar de forma centralizada las identidades digitales de los miembros de una corporación es necesario usar un sistema de gestión de identidades

Elementos de un sistema de gestión de identidades



Ej: Servicio de Directorio X.500 + Acceso con LDAP

Ej: Servicio de Directorio Activo (para Windows)

Si NO HAY Servicio de Directorio …

Ej. La base de usuarios de Kerberos puede actuar como Proveedor de Identidad

Proveedores de Identidad Digital (Globales)

Hay empresas de Internet que proveen servicios gratuitos a sus usuarios registrados

Servicios típicos: email, videoconferencia, almacenamiento, aplicaciones, etc.

Google Microsoft PayPal YAHOO!

Además están las empresas que soportan redes sociales

twitter facebook myspace.com LinkedIn

¡Todas estas empresas tienen millones de cuentas de sus usuarios!

- Tratan de operar como Proveedores de Identidad Globales
  - Ofertan servicios de autenticación de usuarios a otros sitios web
    - Así un pequeño sitio web no abre cuentas a sus usuarios sino que usa las de un proveedor
    - Básicamente proveen autenticación pues no tienen atributos (roles laborales) del usuario para gestionar de forma detallada la autorización de acceso a recursos
- Proveen servicios a usuarios autenticados por otros proveedores globales
  - A esta técnica de autenticación también se la denomina "*Social Login*" y "*Social Sign-in*"

Gestión de identidades federadas (1) Introducción

La gestión de identidades federadas es: (*Federated Identity Management*)

El conjunto de tecnologías, estándares, procedimientos, etc., que permiten la portabilidad de las identidades digitales (los atributos, los derechos de acceso a recursos, etc.) entre múltiples dominios de seguridad autónomos

Objetivo: Permitir a los usuarios de un dominio de seguridad acceder a los datos y servicios de otros dominios de forma transparente (= sin volver a autenticarse)

Problema:

Como los dominios de seguridad son autónomos o independientes …

NO es posible implementar un control centralizado

Las organizaciones deben formar una federación

basada en estándares y niveles de confianza mutua
para compartir (portar) las identidades de forma segura

Beneficios:

Cuando dos organizaciones federan sus gestiones de identidad …

Un empleado puede autenticarse en la intranet de su organización (ej. usando SSO)
Y luego, acceder a servicios de la otra organización sin re-autenticarse en ella

Gestión de identidades federadas (2) Antes

Para federar sus identidades, dos o más organizaciones tienen que acordar …

Representaciones de la identidad + atributos (roles) compatibles
y establecer mapeos entre las representaciones

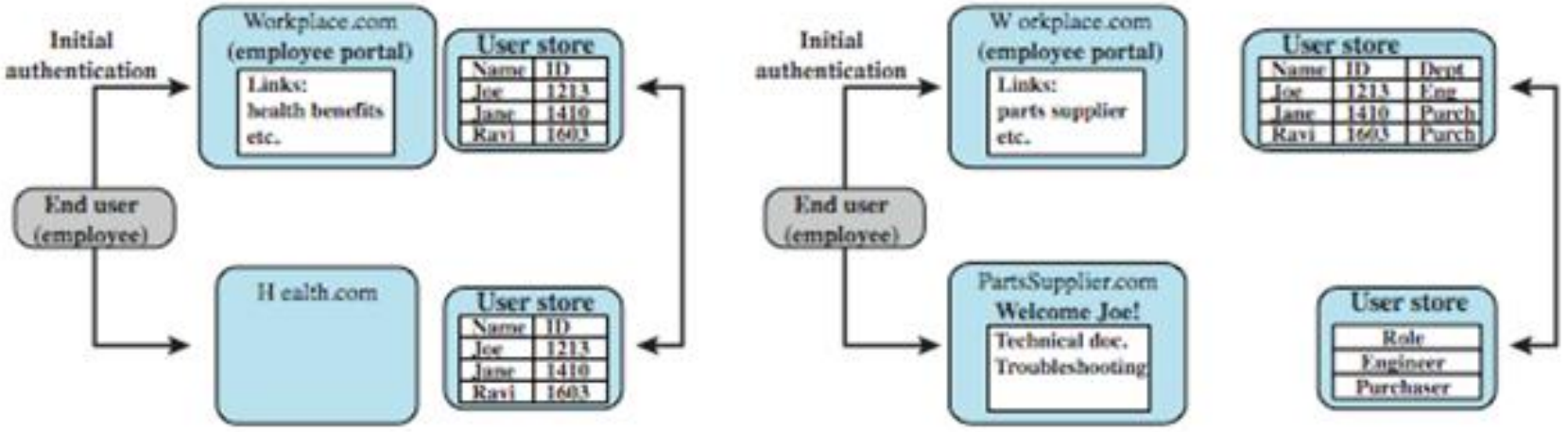
Formas antiguas de federar (basadas en centralizar elementos clave)

- Usar un meta-directorio que contiene una copia del directorio de cada organización
- Usar un directorio virtual que contiene apuntadores al directorio de cada organización

Estos mecanismos tienen problemas

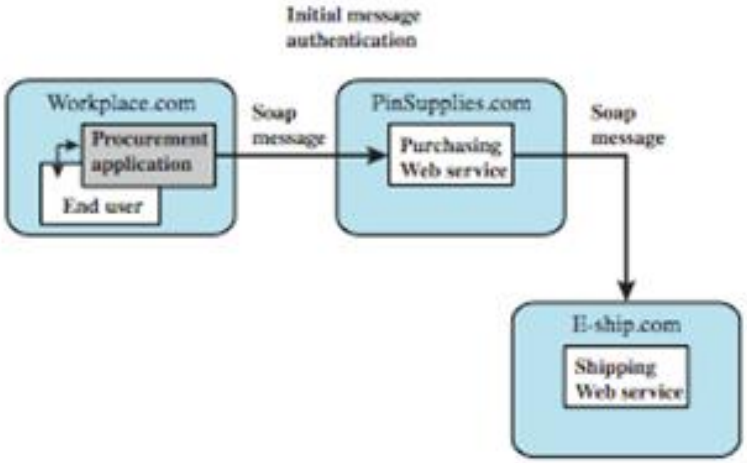
- No escalan bien con un número elevado de usuarios y/o organizaciones
- Exponen demasiada información de cada organización innecesariamente
- Los métodos de autorización de las organizaciones pueden ser incompatibles

Gestión de identidades federadas (3) Ejemplos



(a) Federation based on account linking

(b) Federation based on roles



(b) Chained Web Services

Gestión de identidades federadas (4) Ahora

Forma actual de gestionar la federación

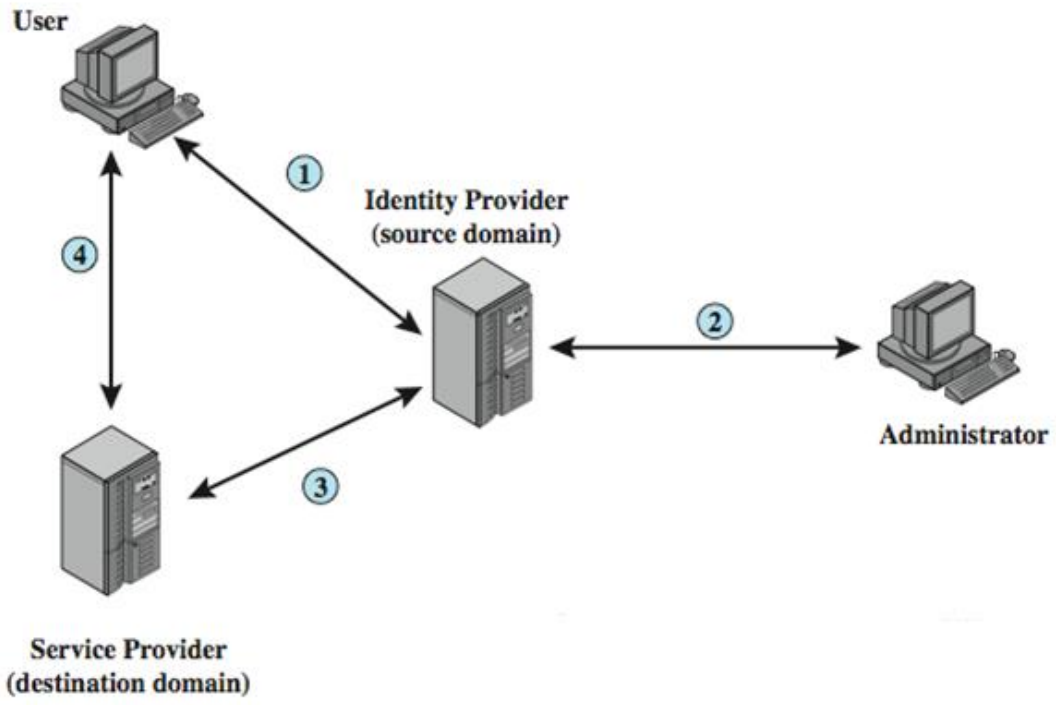
① El Usuario proporciona su identidad y atributos al Proveedor de Identidad (en su mismo dominio)

② Un Administrador (en el mismo dominio) asocia +atributos (roles) a la identidad del usuario

④ El Usuario solicita acceder a un recurso del Proveedor de Servicios (**en otro dominio**)

③ El Proveedor de Servicios solicita al Proveedor de Identidad que autentique al usuario

④ El Proveedor de Servicios proporciona al Usuario los servicios para los que está autorizado



Estándar SAML (1): Conceptos básicos

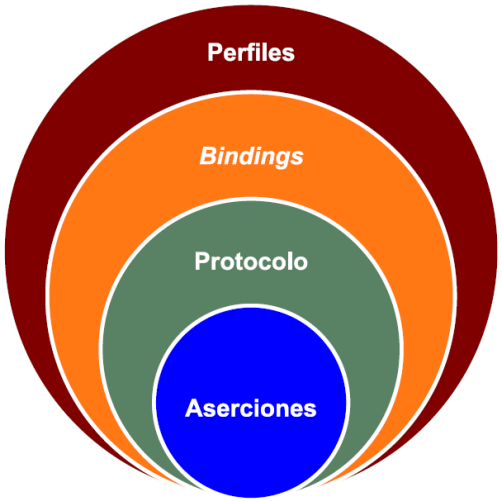
SAML (***S**ecurity **A**ssertion **M**arkup **L**anguage*) es un estándar abierto basado en XML para intercambiar datos de autenticación y autorización entre dos partes:
Un proveedor de identidad y Un proveedor de servicios

Fue diseñado por el Comité Técnico para Servicios de Seguridad de OASIS (*Organization for the Advancement of Structural Information Standards*)

2002 Nov SAML V1.0
2003 Sep SAML V1.1
2005 Mar SAML V2.0

→ Ambas en uso, pero son incompatibles

https://www.oasis-open.org/standards



Aserciones: Afirmación sobre un usuario que hace un Prov Identidad por petición de un Prov Servicio – Definidas en un esquema XML

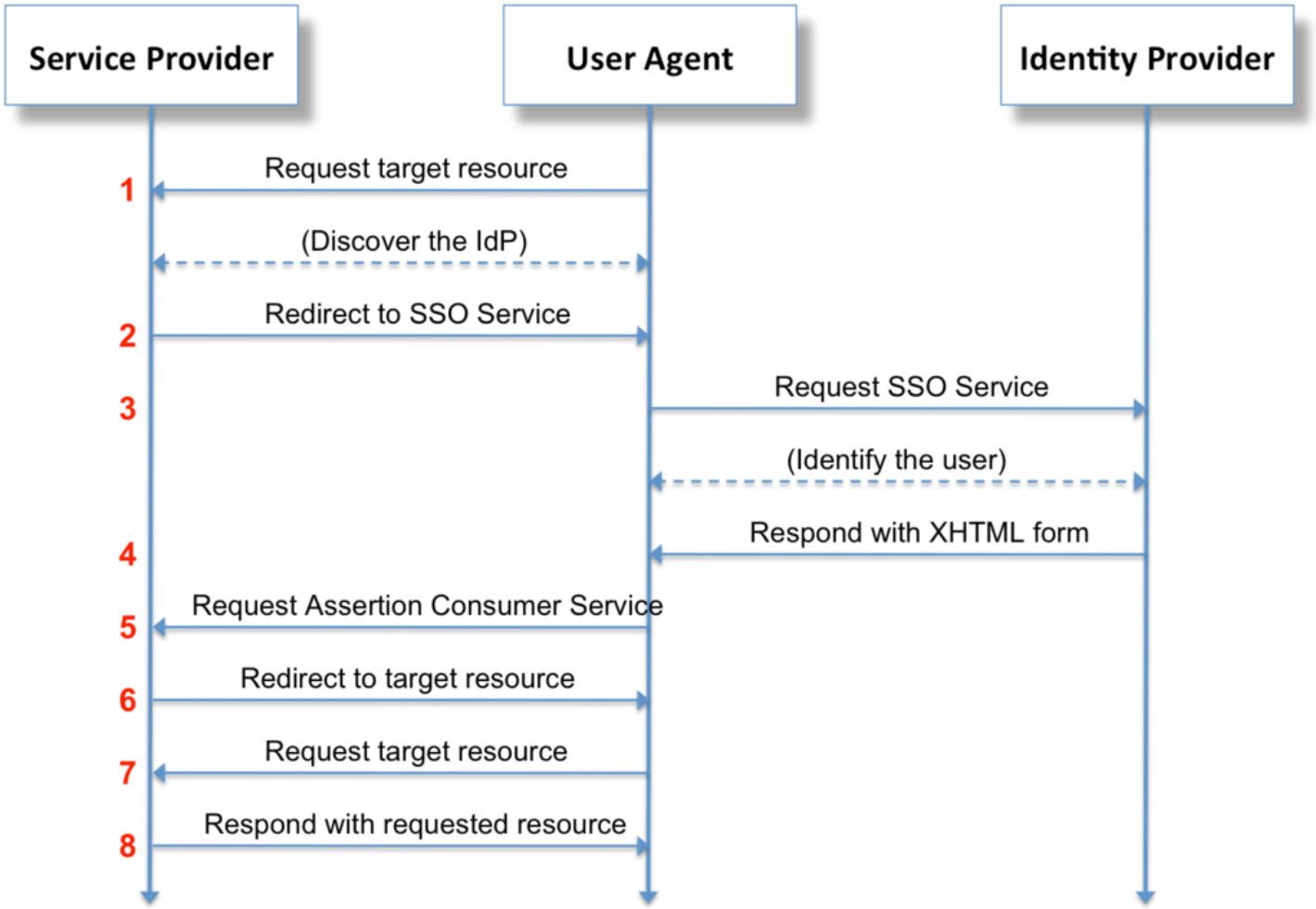
Protocolos: Definen los mensajes (solicitudes/respuestas) para pedir y obtener aserciones – Definidos en un esquema XML de protocolos

Bindings: Protocolos de comunicación (SOAP, HTTP) usados para transportar los mensajes del protocolo SAML

Perfiles: Casos de uso predefinidos combinando tipos de aserciones, protocolos y bindings (Ej. *Web browser SSO profile*)

Estándar SAML (2): Web browser SSO profile

Un usuario vía un ***User Agent*** solicita un recurso en un ***Service Provider*** (SAML enabled)
El ***Service Provider*** utiliza un ***Identity Provider*** (SAML enabled) para autenticar al usuario



Estándar OpenID (1)

OpenID es un estándar abierto para autenticar usuarios (usando navegadores web) en múltiples sitios web (*Relying Parties*, RP) utilizando un Proveedor de Identidad

https://openid.net/developers/specs/ Versión 2.0 (Final) → Diciembre 2007

VENTAJA para los WebMasters

No tienen que mantener cuentas de usuarios en su sitio web

VENTAJA para los usuarios

Consolidan sus identidades digitales

NO tienen que tener una cuenta en cada sitio web

SINO que tienen una sola cuenta en el proveedor de identidad

OpenID resuelve el problema Web-SSO en el contexto de Internet

OpenID es descentralizado

Ninguna autoridad central debe aprobar o registrar — Relaying Parties, OpenID Providers

Un usuario final puede — Elegir libremente el proveedor de OpenID; Conservar su identificador si cambia de proveedor

OpenID

OpenID Foundation Corporate Members

Google
Microsoft
PayPal
PingIdentity
Symantec
verizon
YAHOO!

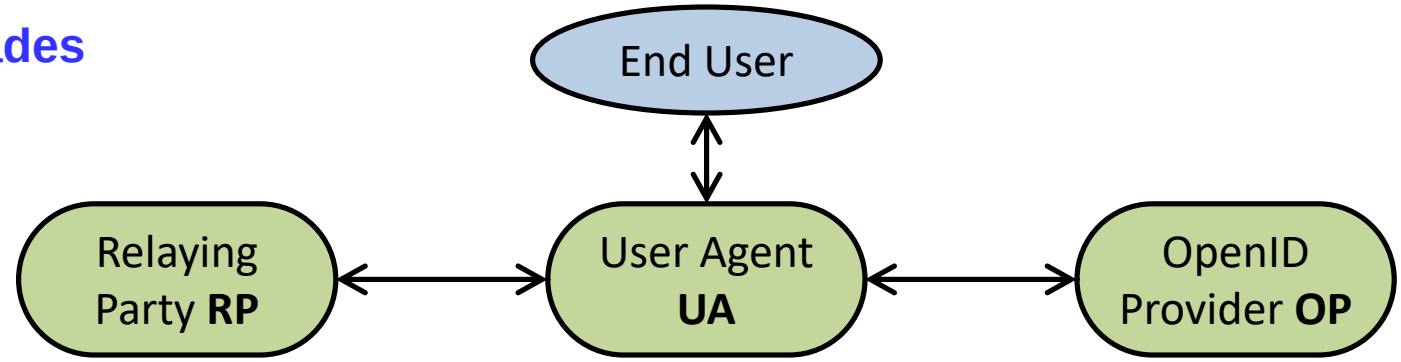
Estándar OpenID (2)

Mecanismos de autenticación

OpenID NO especifica ningún mecanismo concreto para la autenticación de los usuarios

Se pueden usar contraseñas, tokens, biometría, o combinarlos

Entidades



- **End User**: El usuario final es la entidad que desea afirmar (*assert*) su identidad
- **Relaying Party**: La parte que confía es el sitio web o aplicación que desea verificar el identificador del usuario final
- **OpenID Provider**: El proveedor OpenID es el que gestiona las identidades de las entidades
- **User Agent**: El agente del usuario es el programa (web browser) que usa la entidad para comunicarse con RP y OP

Estándar OpenID (3)

Registro de usuarios en el Proveedor

El usuario debe crear una cuenta en un proveedor de identidad que use OpenID

El proveedor asigna al usuario un Identificador OpenID

Hay 2 tipos de identificadores OpenID →
- URL: Uniform Resource Locator
- XRI: Extensible Resource Identifier

El usuario obtiene del Proveedor un URL habilitado para OpenID
que puede usar en los sitios web habilitados para OpenID

Típicamente si un usuario con el identificador "username"
Se registra en el proveedor OpenID "https://myprov.com/"
Se genera el identificador OpenID "https://username.myprov.com"

Posteriormente, el usuario puede usar un URL que controle personalmente
(un blog o una página personal) como un alias o una identidad delegada

Protocolo de autenticación

En desuso → Ahora se usa OpenID Connect – Basado en OAuth

Estándar OAuth (1)

OAuth es un estándar abierto de autorización (de acceso a recursos)

Proporciona un método para que un cliente (aplicación) acceda a un recurso en un servidor en nombre del propietario del recurso

Proporciona un método para que un usuario final autorice el acceso a sus recursos por terceros **SIN compartir sus credenciales** (generalmente: username + password)

https://oauth.net/2/

OAuth es un servicio distinto de OpenID pero puede complementarlo

Es un estándar soportado por el IETF:

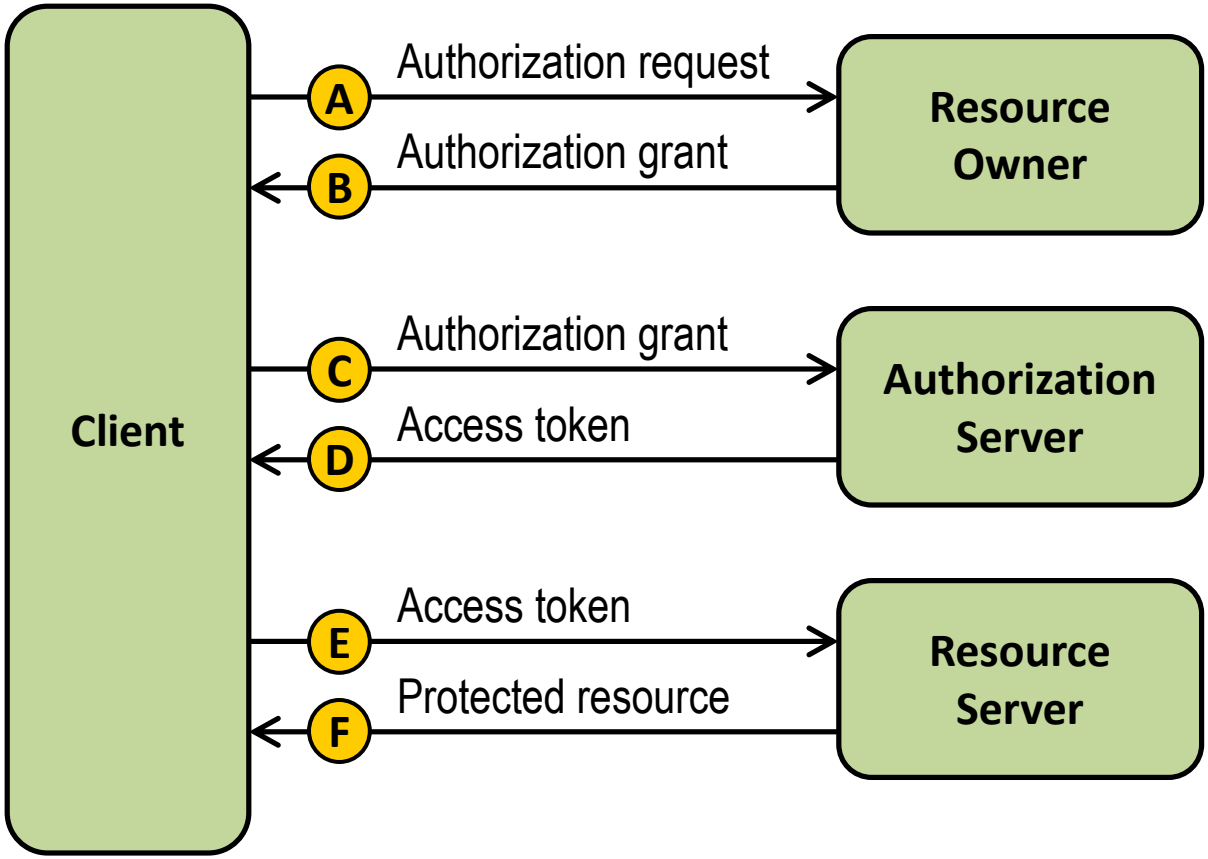
2010 Abr	RFC-5849	The OAuth 1.0 Protocol
2012 Oct	RFC-6749	The OAuth 2.0 Authorization Framework https://tools.ietf.org/pdf/rfc6749.pdf
	RFC-6750	The OAuth 2.0 Authorization Framework: Bearer Token Usage
2019 Ago	RFC-8628	OAuth 2.0 Device Authorization Grant

OAuth 1.0 es incompatible con OAuth 2.0

Estándar OAuth (2)

Roles en OAuth y sus interacciones

RFC 6749 Sec 1.2



Estándar OAuth (3)

La aplicación Cliente

RFC 6749 Sec 2

Registro

Antes de usar el protocolo, el cliente tiene que registrarse con el servidor de autorización (Generalmente requiere la interacción de una persona con un formulario de registro en html)

Al registrar un cliente, el desarrollador o responsable del cliente DEBE:

- Especificar el tipo del cliente
- Proporcionar los URIs de redirección del cliente

Tipos

En función de la capacidad del cliente para mantener la confidencialidad de las credenciales con las que se ha registrado en el servidor de autorización

- Cliente Confidencial → Es capaz de mantener la confidencialidad
- Cliente Público → Es incapaz de mantener la confidencialidad

Estándar OAuth (4)

La aplicación Cliente → Perfiles

RFC 6749 Sec 2

Aplicación web

Es un cliente confidencial ejecutándose en un servidor web

Ej. Aplicación servidora de un sitio web

El propietario accede al cliente mediante una interfaz html

No tiene acceso a las credenciales del cliente, ni a los mensajes del protocolo

Usa el flujo "*Authorization Code Grant*"

Aplicación basada en el agente del usuario

Es un cliente público cuyo código es descargado de un servidor web y ejecutado dentro del agente de usuario (navegador web) en un dispositivo usado por el propietario del recurso

Ej. Aplicación JavaScript ejecutada en el navegador del propietario

El propietario puede acceder a las credenciales del cliente y los mensajes del protocolo

Usa el flujo "*Implicit Grant*"

Aplicación nativa

Es un cliente público instalado y ejecutado en un dispositivo usado por el propietario del recurso

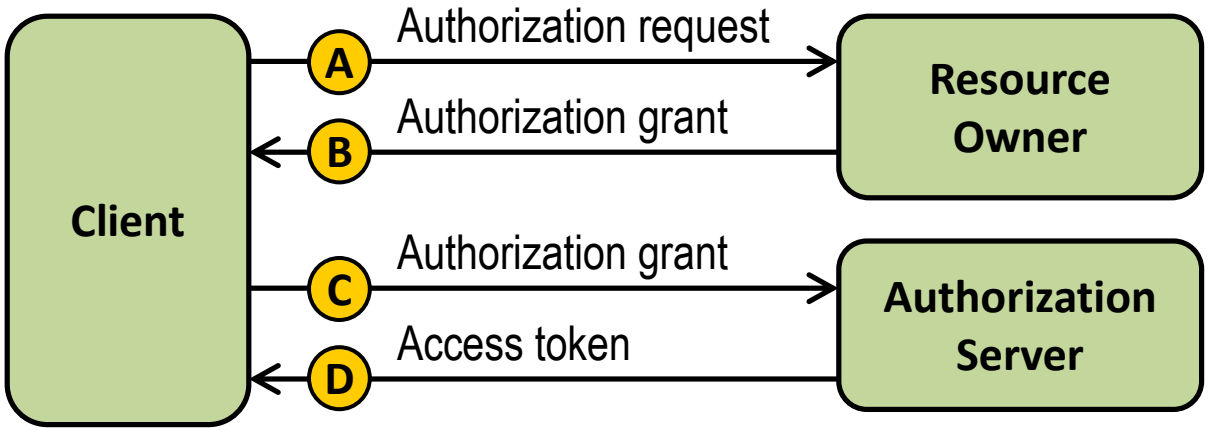
Ej. Una App ejecutada en el móvil del propietario

El propietario puede acceder a las credenciales del cliente y los mensajes del protocolo

Usa el flujo "*Resource Owner Password Credentials Grant*"

Estándar OAuth (5)

Interacciones genéricas del protocolo OAuth para obtener un token de acceso:

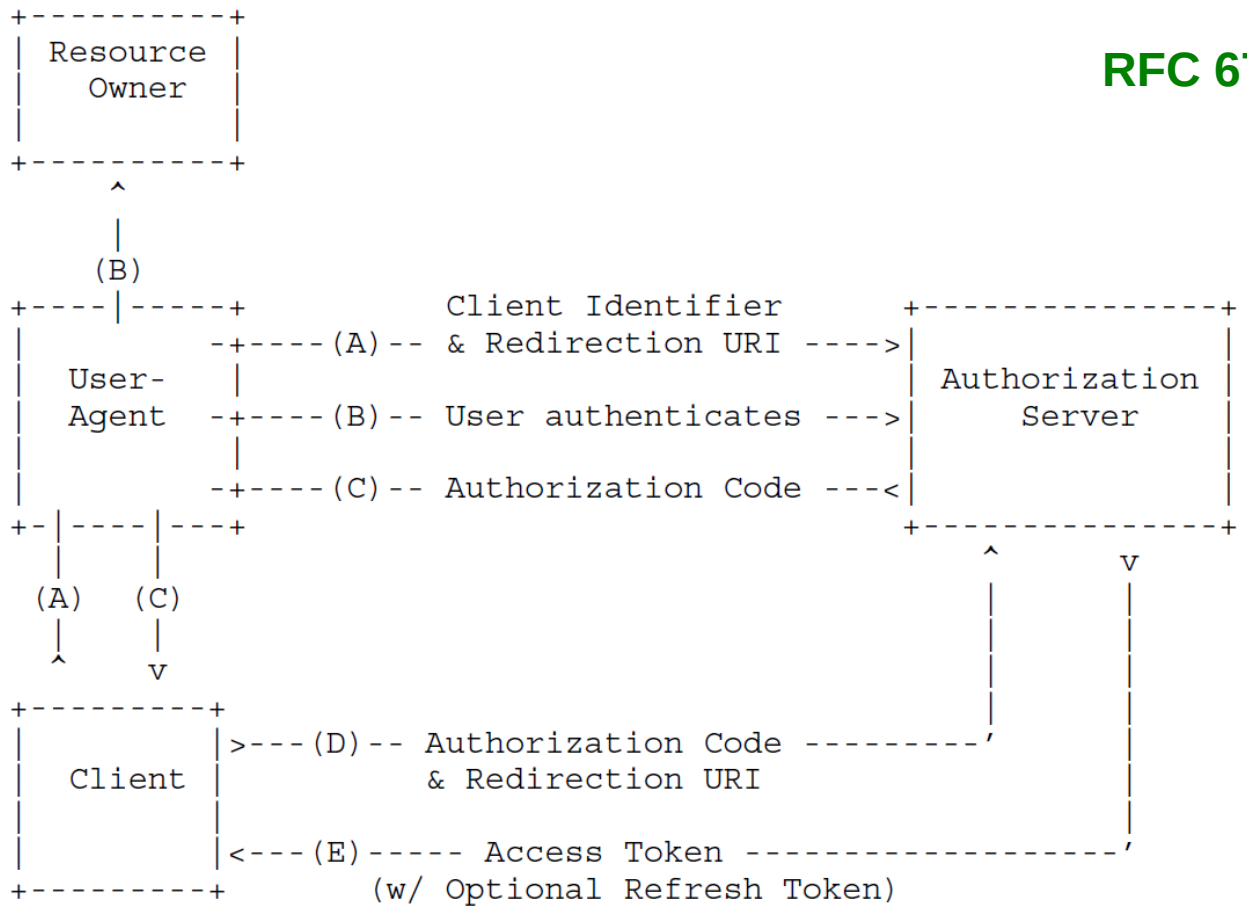


OAuth especifica 4 implementaciones de las interacciones genéricas → **Flujos**

1) Authorization Code Flow
2) Implicit Grant Flow
3) Resource Owner Password Credentials Flow
4) Client Credentials Flow

Estándar OAuth (6)

RFC 6749 Sec 4.1

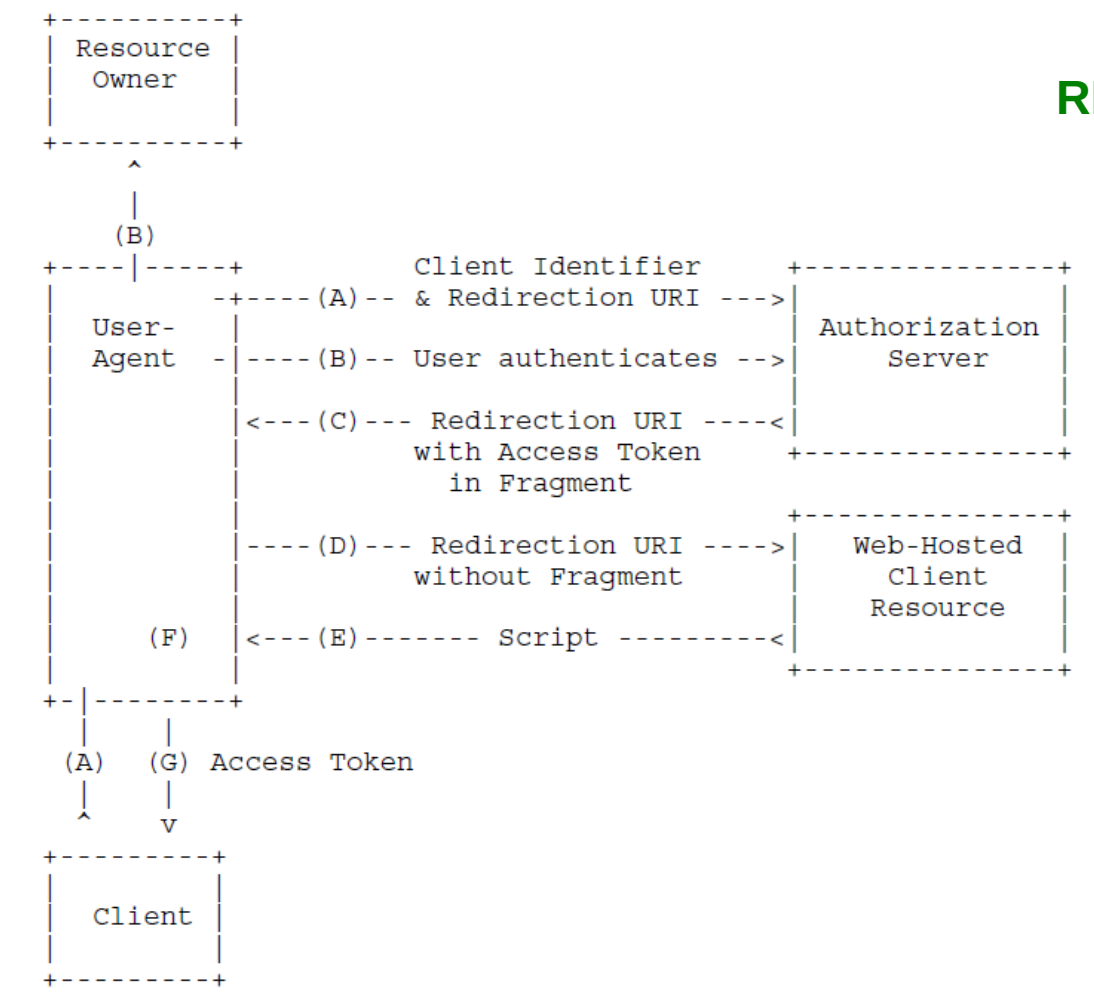


Note: The lines illustrating steps (A), (B), and (C) are broken into two parts as they pass through the user-agent.

Figure 3: Authorization Code Flow

Estándar OAuth (7)

RFC 6749 Sec 4.2



Note: The lines illustrating steps (A) and (B) are broken into two parts as they pass through the user-agent.

Figure 4: Implicit Grant Flow

Estándar OAuth (8)

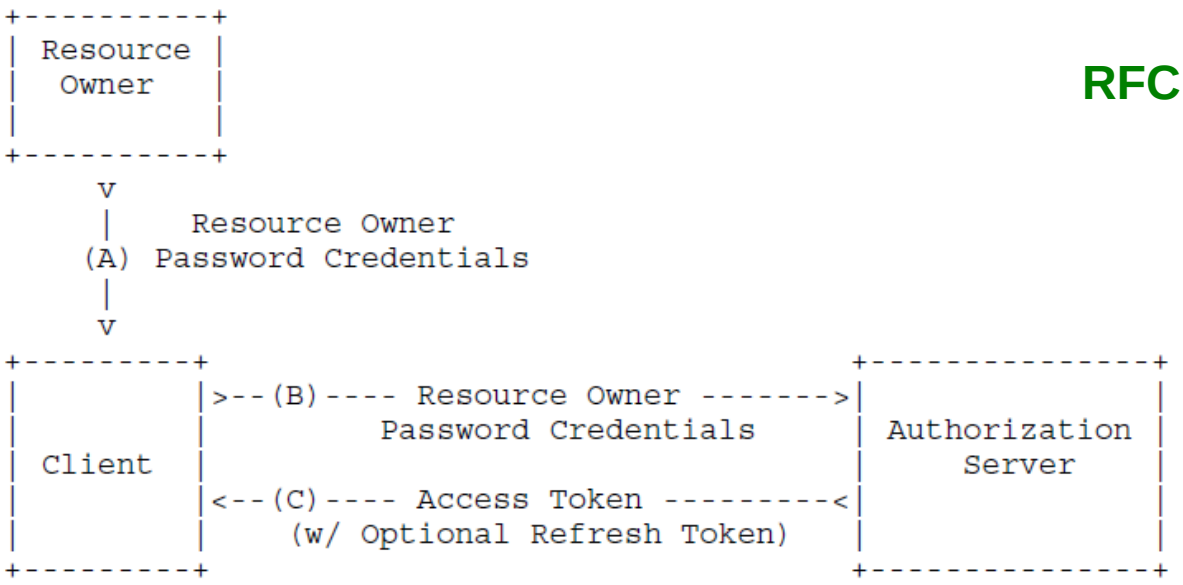


Figure 5: Resource Owner Password Credentials Flow

RFC 6749 Sec 4.3

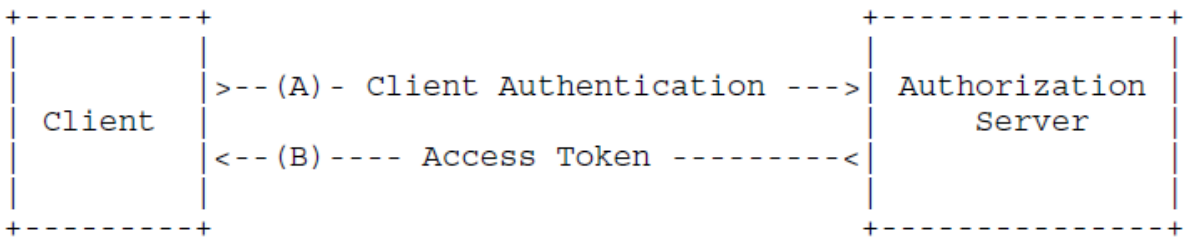


Figure 6: Client Credentials Flow

RFC 6749 Sec 4.4

OpenID Connect (1)

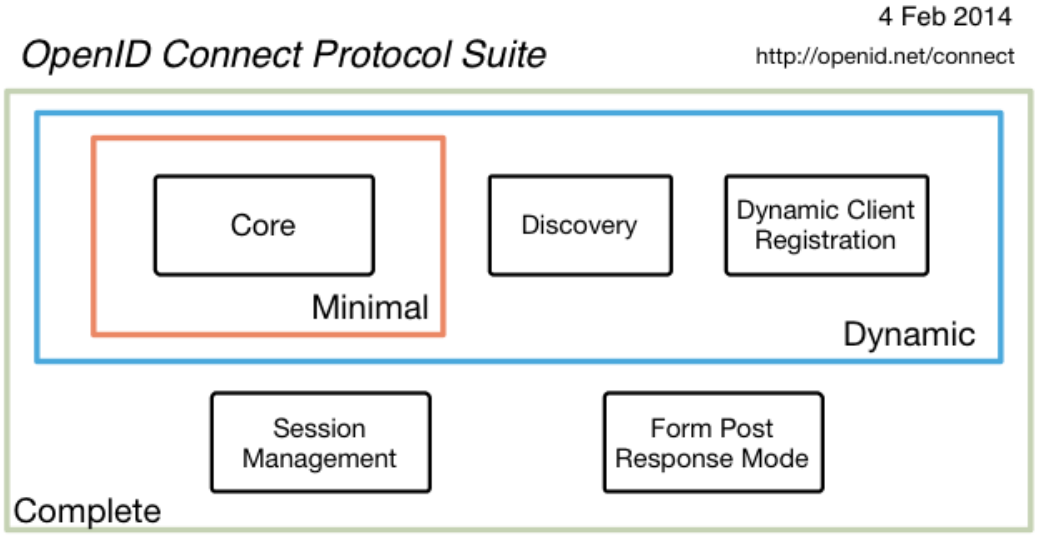
OpenID Connect 1.0 es una capa de gestión de identidad ubicada sobre el protocolo OAuth 2.0

Permite a los clientes: https://openid.net/developers/how-connect-works/

1.- Verificar la identidad del usuario final basándose en la autenticación realizada por un Servidor de Autorizaciones
2.- Obtener información básica sobre el perfil del usuario final de forma interoperable

Realiza las mismas tareas que OpenID pero es fácilmente utilizable por aplicaciones nativas

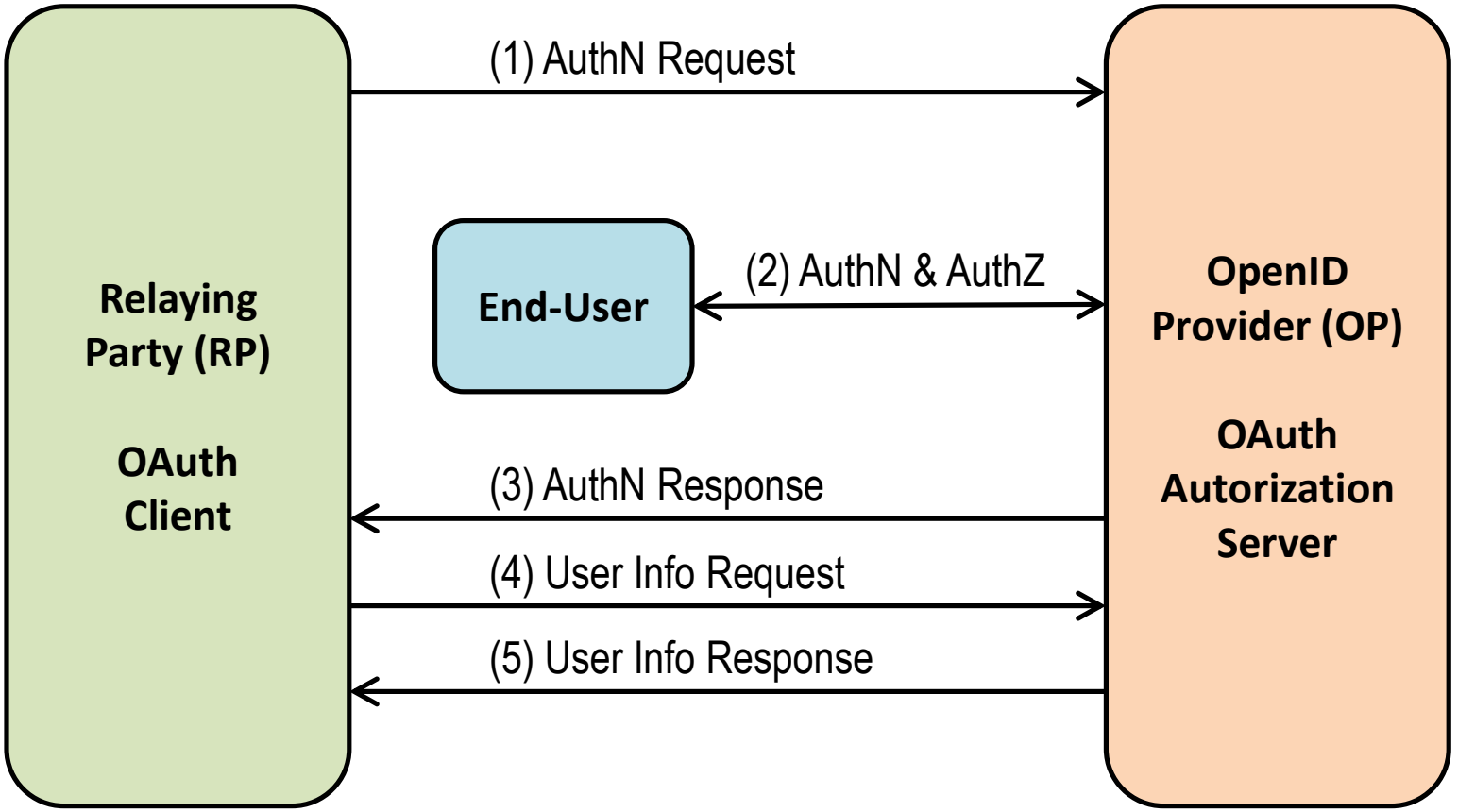
La especificación OpenID Connect consiste en varios documentos organizados así:



https://openid.net/developers/specs/

OpenID Connect (2)

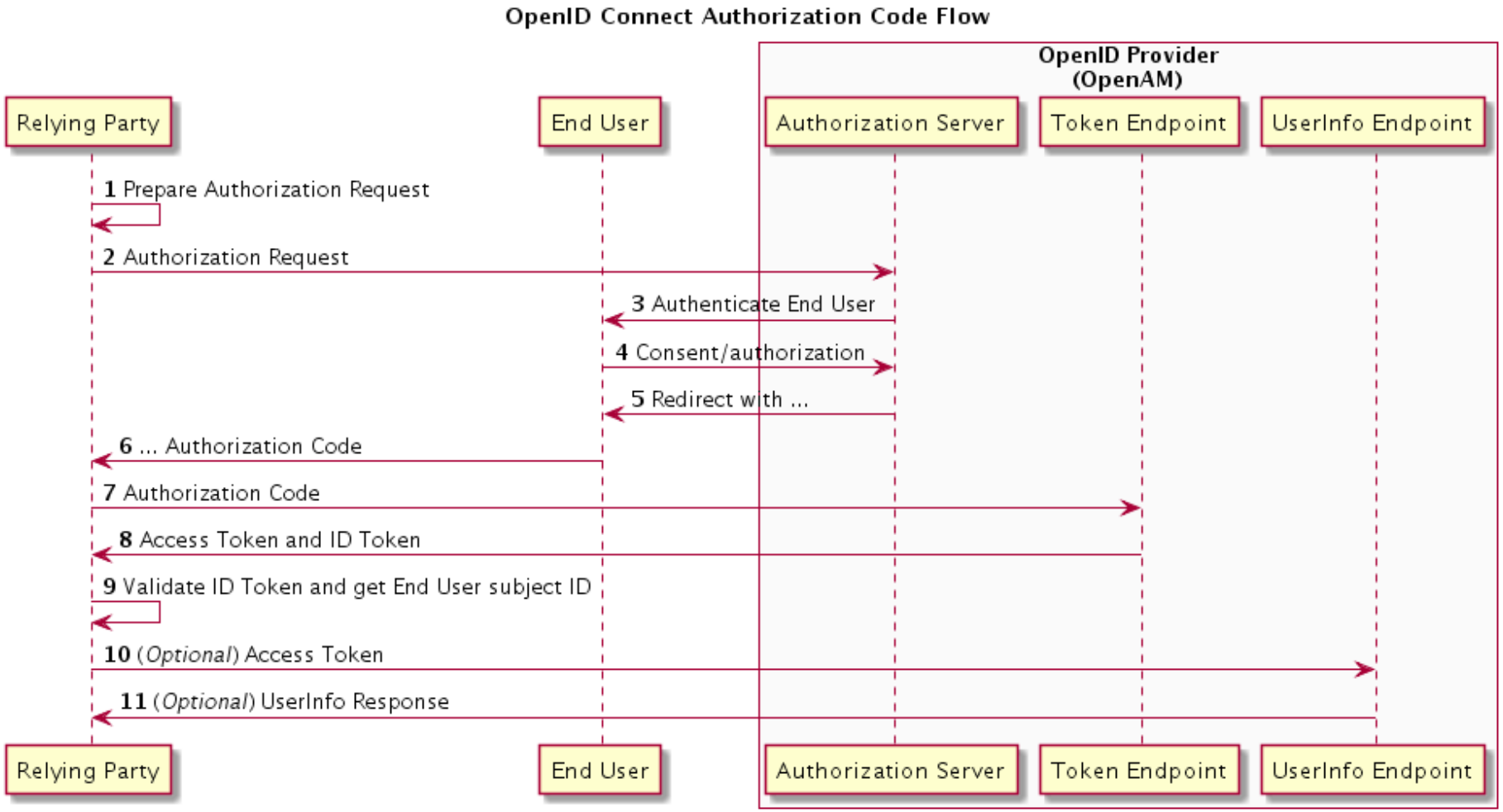
OpenID Connect Core 1.0 Sección 1.3 Overview



https://developers.google.com/identity/protocols/OpenIDConnect

OpenID Connect soporta 3 Flujos: (1) Authorization Code (2) Implicit (3) Hybrid

OpenID Connect (3)



OpenID Connect (4)