COMPUTACIÓN DISTRIBUIDA

# SISTEMAS DE AUTENTICACIÓN Y AUTORIZACIÓN EN INTERNET

v1.0

**Jordi Sánchez**

jun-2011

# CONTENIDOS DE ESTE DOCUMENTO

## RESUMEN

Con la proliferación de servicios en Internet, especialmente los relacionados con redes sociales, el problema de la **multitud de sistemas** de autenticación de usuarios y de acceso a los recursos y datos personales se hace cada vez más importante. En este trabajo se describen y discuten los principales **estándares y protocolos de autenticación y autorización en Internet** utilizados en la actualidad, no tanto desde un punto de vista técnico de funcionamiento de los protocolos (su especificación y documentación relacionada está disponible en la red), sino desde la perspectiva de los usuarios finales de esos servicios.

## 1. INTRODUCCIÓN: EL PROBLEMA

El uso de Internet y la oferta de servicios disponibles han crecido enormemente, y la mayoría de usuarios utilizan **múltiples aplicaciones de terceros**: por ejemplo, un usuario típicamente consultará su correo en un sistema de webmail, participará en varias redes sociales, etc.

> *Héctor podría ser un usuario típico de Internet. Como muchos otros, tiene su cuenta de correo electrónico en un webmail (GMail), y usa diariamente Facebook y Twitter para estar en contacto con sus amigos y conocidos. Además, cuando las necesita usa esporádicamente otras aplicaciones más especializadas, como Flickr para guardar algunas de sus fotos.*

La mayoría de esas aplicaciones requieren que el usuario **se autentique**; es decir, que demuestre de algún modo (habitualmente usando un identificador de usuario único y una contraseña asociada) que es quien dice ser. Y no solo eso; muchas de esas aplicaciones almacenan también **datos personales**, en muchas ocasiones comunes, como pueden ser la dirección de correo electrónico, una fotografía propia, la fecha de nacimiento, el domicilio, etc.

Al tratarse de aplicaciones independientes entre sí, en principio cada una de ellas utilizaría su **propio sistema de autenticación y de gestión de datos personales**, lo cual implica **inconvenientes** al usuario cada vez más graves a medida que el número de sistemas que utiliza crece: debe recordar diferentes identificadores de usuario y contraseñas para diferentes sistemas, y autenticarse en cada uno de ellos cada vez que desea usarlos. Además, debe mantener su información personal actualizada en todos ellos. El problema empeora si tenemos en cuenta que es habitual que los usuarios utilicen diferentes identidades (cuentas) en un mismo servicio.

> *Cada vez que tiene que entrar en GMail, Facebook o Twitter, Héctor debe recordar cuál es su nombre de usuario y contraseña en cada sistema. Cuando entra desde el ordenador de casa, el navegador ya los tiene almacenados y no tiene que introducirlos; pero cuando se conecta desde otras ubicaciones, suele costarle varios intentos e incluso alguna vez ha tenido que solicitar una nueva contraseña por e-mail. Además, cada vez que desea cambiar su foto de perfil, tiene actualizarla en diversas aplicaciones.*
>
> *Con las aplicaciones que utiliza con poca frecuencia es peor; es habitual que no recuerde la contraseña y/o que los datos que utilizan estén desfasados. Algunas de ellas tienen todavía una foto suya que desearía que ya no estuviera disponible en la red.*

Esta situación no sólo es **incómoda** para los usuarios, sino que lleva a que estos realicen prácticas que comprometen la **seguridad** de sus sistemas, como son:

- uso de contraseñas poco seguras;
- anotación de las contraseñas en papel, en documentos de texto, etc.;
- reutilización de contraseñas en diferentes sistemas [1]

Adicionalmente, los datos personales almacenados en los diferentes sistemas pueden quedar desactualizados o ser incoherentes entre sí.

El objetivo de las tecnologías y protocolos descritos en este trabajo es dar solución a esos inconvenientes, ofreciendo sistemas centralizados tanto de autenticación (conocidos como ***single-sign-on*** [2]) como de gestión de datos personales.

> *Para evitar tener que recordar tantas contraseñas diferentes, Héctor ha terminado por utilizar la misma en diferentes aplicaciones. Sabe que no es seguro, porque aumenta el riesgo de que alguien la averigüe y tenga acceso prácticamente a toda su información en la red; pero le parece que eso es mejor que tenerlas escritas en un papel o un Word.*
>
> *En cuanto a sus datos personales, habitualmente los tiene al día en Facebook, que es una aplicación que usa con frecuencia. En el resto de aplicaciones intenta mantener el mínimo de información posible, y no se preocupa demasiado si está desfasada.*
>
> *Sería más cómodo para él utilizar un sistema para entrar en todas esas aplicaciones con una única autenticación y que, además, sus datos personales estuvieran en un único lugar, y que las aplicaciones accedieran a ellos cuando lo necesitaran (y tuvieran permiso para ello).*

## 1.1. Autenticación vs. autorización

Antes de seguir, es necesario hacer una distinción clara entre dos tipos de servicios ofrecidos por las tecnologías que vamos a describir [3].

**Autenticación**: consiste en un sistema para certificar que el usuario es quien dice ser; lo más común es utilizar una combinación de identificador de usuario único y contraseña, aunque existen otros. Un sistema de *single-sign-on* consiste, por tanto, en un protocolo de autenticación que funcione en más de un sistema; es **federado** si el sistema responsable de la autenticación puede ser cualquiera que cumpla el estándar definido por el sistema; por otro lado, será un sistema **delegado** si el sistema que autentica es uno predeterminado [4].

**Autorización**: consiste en dar acceso a una serie de recursos a un usuario o sistema (para ello, el usuario o el sistema previamente tendrán que haberse autenticado). En este trabajo no nos centraremos en la autorización que se da a un usuario después de autenticarse en una determinada aplicación (que será gestionado por esa aplicación), sino en la **autorización que un usuario proporciona sobre sus recursos en un determinado sistema para que un tercer sistema tenga acceso a ellos**.

Así, podemos decir que las tecnologías de autenticación ofrecen soluciones al problema de tener que autenticarse en múltiples sistemas, mientras que las de autorización intentarán resolver los inconvenientes de tener datos y recursos personales repartidos en diferentes sistemas.

> *Para Héctor, el problema de tener que recordar diferentes contraseñas podría solucionarse con un sistema de autenticación único, mientras que el de tener sus datos personales en diferentes sitios podría resolverse autorizando a los sistemas a que accediesen a ellos en un repositorio único, donde estén actualizados.*

## 2. PROTOCOLOS Y ESTÁNDARES

Una primera solución evidente al problema de la autenticación sería el uso de algún tipo de **certificado personal** (basado, por ejemplo, en el estándar X.509 [5]) emitido y validado por alguna **autoridad central** de confianza; en España, sin ir más lejos, la versión digital del documento identificativo oficial (DNI electrónico) incluye certificados de usuario [6], y también permite obtener certificados para los navegadores de Internet [7].

Si bien este tipo de certificados son útiles para gestiones con determinadas entidades, **no son prácticos en general para el acceso a servicios de Internet**, ya que requerirían la presencia de autoridades centrales de confianza aceptadas globalmente que validen la identidad del usuario (cosa harto complicada en Internet), y además implicarían gestiones administrativas previas para obtener el certificado que complicarían el proceso.

Es por eso por lo que han surgido los protocolos de autenticación y autorización en Internet; a continuación se describen los más importantes y/o populares de ellos.

### 2.1. OpenID

El protocolo abierto **OpenID** [8], cuya primera versión fue definida en 2005 para su uso en el sitio web LiveJournal, es un **protocolo de autenticación federada**, y consiste básicamente en que el usuario selecciona un servidor externo (el "proveedor" de OpenID) que va a ser el que va a validar su identidad en un sistema determinado (el "consumidor" de OpenID).

El protocolo funciona así (Ilustración 1):

a. El usuario quiere acceder a su cuenta en un servidor.

b. Si ese servidor soporta el protocolo OpenID (es "consumidor" de OpenID), solicita al usuario su OpenID (la URL externa del "proveedor" de OpenID).

c. El usuario introduce o selecciona su OpenID (Ilustración 2).

d. El servidor redirige al usuario al proveedor de OpenID.

e. El usuario se autentica contra el proveedor de OpenID.

f. El proveedor de OpenID redirige al usuario de vuelta al servidor, validando su identidad.

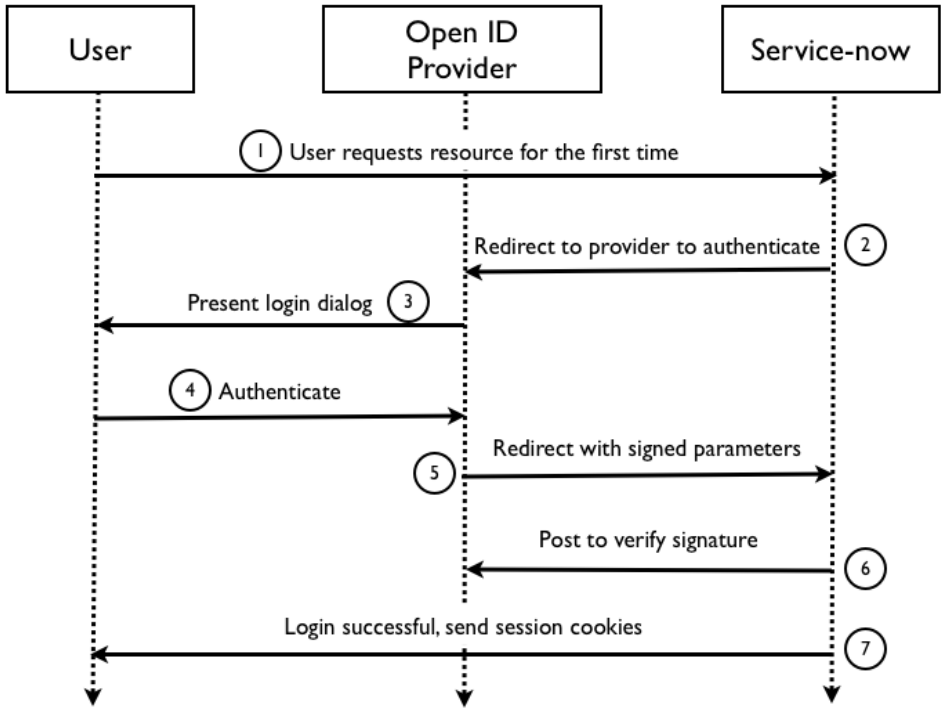


Ilustración 1. Secuencia de autenticación OpenID.

Utilizando siempre un mismo proveedor de OpenID, el usuario sólo necesita recordar su identificador OpenID y una única contraseña para autenticarse, con una misma identidad, en todos los servicios que acepten OpenID como sistema de autenticación.

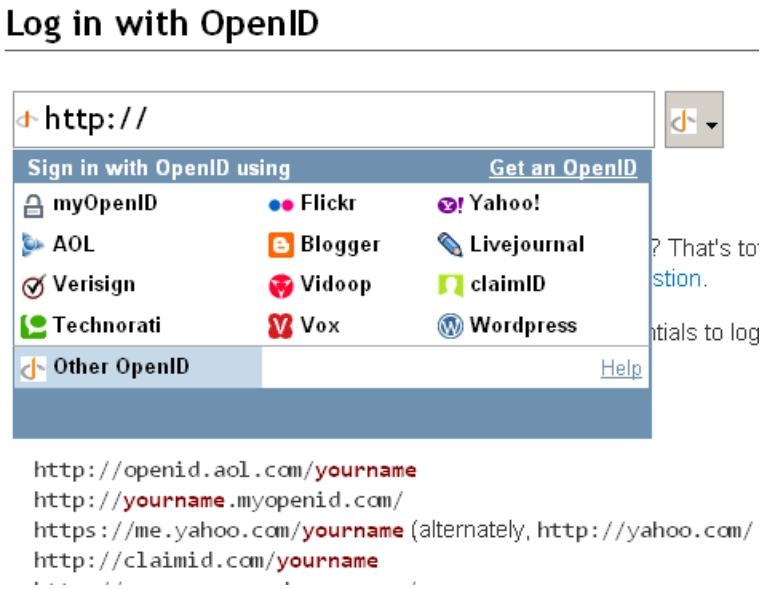


Ilustración 2. Ejemplo de una pantalla de selección del proveedor de OpenID.

Muchos proveedores de servicios actuales en Internet ofrecen la posibilidad de ser utilizados como proveedores de OpenID [9]: Google, Yahoo!, flickr, etc.; existen también servicios dedicados de OpenID, como myOpenID. El usuario incluso tiene la opción de implementar su propio proveedor de OpenID.

### Problemas de OpenID

Se han realizado múltiples críticas al protocolo OpenID [10], que se resumen básicamente en los siguientes aspectos:

- **Complejidad: no se implementa fácilmente.** OpenID ha recibido críticas por parte de los desarrolladores, que lo consideran excesivamente complejo de implementar [11].

- **Seguridad: es un protocolo muy vulnerable al *phishing***. El *phishing* [12] consiste en la suplantación maliciosa de una página de autenticación (*login*) con el objetivo de conseguir el identificador y la contraseña de un usuario. Habitualmente los sitios maliciosos tienen que engañar al usuario para que lleguen a la página suplantada (usualmente mediante enlaces en correos electrónicos); en un servidor que utilice OpenID ese acceso es más sencillo de conseguir: basta con ofrecer al usuario la autenticación mediante OpenID para acceder a una aplicación determinada (que puede ser falsa), y después redirigir al usuario a una página que suplanta la página del proveedor de OpenID para conseguir su contraseña.

- **Privacidad: los proveedores de OpenID tendrán mucha información del usuario**. Si los usuarios utilizan un proveedor de OpenID para autenticarse en múltiples servicios, ese proveedor obtendrá potencialmente muchísima información de la actividad de los usuarios en la red, lo que lleva a posibles problemas de privacidad.

- **Confianza: ¿quién es realmente el usuario?** El protocolo OpenID ofrece un mismo sistema de autenticación para diferentes servidores, pero no ofrece ninguna garantía de la identidad real del usuario. Nada impide, por ejemplo, que procesos de *spam* creen y validen identidades automáticamente mediante OpenID para tener acceso a determinadas aplicaciones.

- **Usabilidad: puede ser incómodo y/o complejo para el usuario**. El proceso de elegir el proveedor de OpenID y autenticarse en un servidor diferente al de la aplicación puede resultar confuso y/o complejo para los usuarios.

- **Adopción: los proveedores de servicios tienen pocos motivos para aceptarlo como autenticación**. El número de proveedores de servicios que aceptan OpenID como mecanismo de autenticación (es decir, que son consumidores) es relativamente bajo en comparación con el de proveedores de OpenID. El motivo es que a las organizaciones y empresas les resulta beneficioso tener las cuentas (los datos) de los usuarios en su servicio; por tanto, siempre estarán más interesados en que los usuarios utilicen las cuentas en su servicio para autenticarse en otros, que en utilizar la autenticación de proveedores externos para su propio servicio (con lo que dejarían de ser los gestores de los datos de los usuarios).

*Algunos de los servicios que Héctor utiliza esporádicamente aceptan que se autentique con un proveedor de OpenID. En esos casos, Héctor puede utilizar su usuario y contraseña de Google o Flickr, por ejemplo, y no necesita tener un usuario y contraseña específico para ese servicio.*

*Sin embargo, la mayoría de los sitios que usa con frecuencia (como GMail o Facebook) no aceptan esa autenticación, por lo que Héctor sigue teniendo los mismos inconvenientes con ellos.*

- **Disponibilidad: se incrementa la dependencia de servidores.** Por un lado, cualquier servicio requerirá el acceso a dos servidores: el de OpenID y el del propio servicio. Además, una caída en el proveedor de OpenID impedirá que el usuario pueda acceder a los servicios de los que depende.

- **Patentes: no es un protocolo "tan" abierto**. A pesar de que presume de ser un protocolo "abierto" porque en principio puede usarse cualquier servidor como proveedor de OpenID, en la práctica no se garantiza que los servidores acepten cualquier proveedor. Además, en un principio hubo conflictos en cuanto a la patente de la tecnología [13].

## 2.2. OAuth

El protocolo abierto **OAuth** [14], a diferencia de OpenID, es un **protocolo de autorización**; más exactamente, de delegación de acceso; es decir, permite definir cómo un tercero va a acceder a los recursos propios. Empezó a definirse en 2006 ante las carencias del protocolo OpenID, y en 2007 se publicó la primera versión oficial.

El propósito de este protocolo es, pues, que un usuario que tiene determinados recursos en un servidor (el "proveedor" de OAuth) pueda **dar acceso a un tercero** (el "consumidor", usualmente un sitio web) a parte o todos esos recursos, sin necesidad de que ese tercero conozca su usuario y contraseña, ya que con esos datos tendría el control total de la cuenta.

El protocolo es básicamente este (Ilustración 3):

a. El usuario dispone de una serie de recursos propios en un servidor (el "proveedor").

b. Un servidor externo (el "consumidor") desea acceder a un subconjunto de esos recursos.

c. El consumidor redirige al usuario hacia el proveedor.

d. El usuario se autentica en el proveedor (si no lo estaba previamente).

e. El proveedor pregunta al usuario si autoriza al consumidor a que utilice esos determinados recursos (Ilustración 4).

f. El usuario autoriza al consumidor a utilizar esos recursos.

g. El servidor externo (consumidor) consigue acceso a esos recursos.

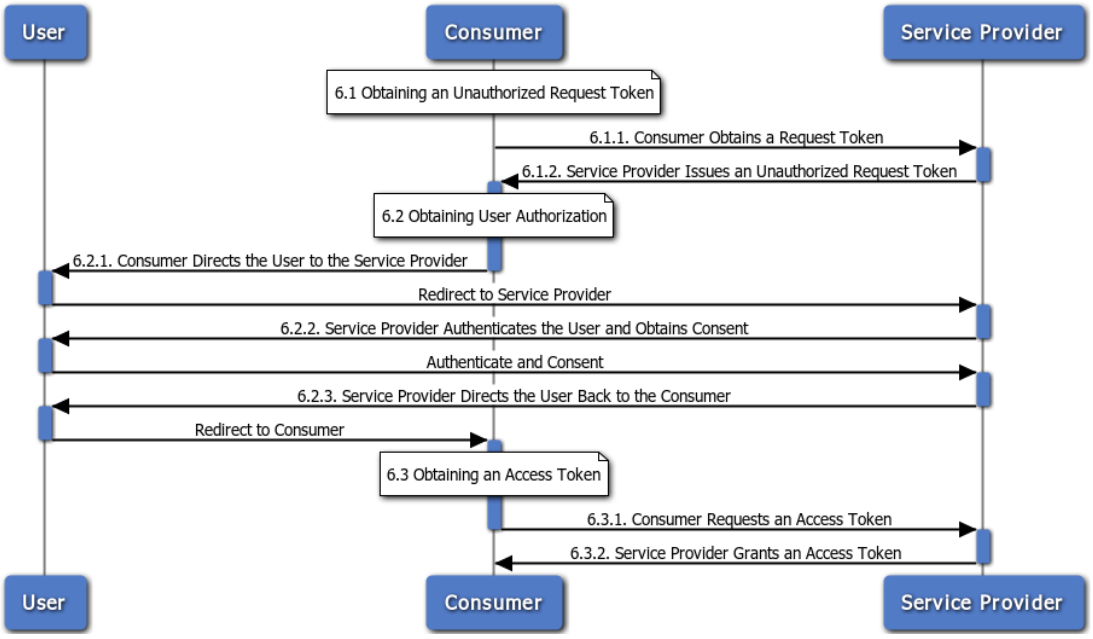


Ilustración 3. Secuencia de autorización OAuth.

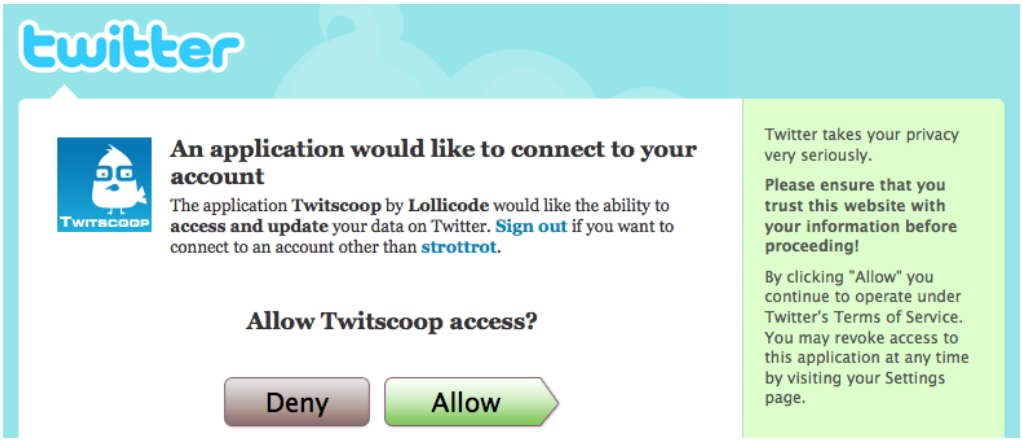


Ilustración 4. Pantalla de autorización de acceso a recursos de Twitter mediante OAuth.

### Problemas de OAuth

Estas son algunas de las críticas que se han realizado al protocolo:

- **Complejidad.** El protocolo ha sido definido como "una solución poco elegante a un problema complejo" [15]; eso hace que existan diversas implementaciones y librerías que realizan implementaciones poco robustas y poco documentadas.

- **Orientado a navegadores**. El protocolo está especialmente orientado a su uso en navegadores de Internet, de modo que su uso en otro tipo de aplicaciones (de escritorio, móviles, embebidas, etc.) resulta problemático, por ejemplo, por el uso de redirecciones entre URLs.

- **Seguridad**. En parte como consecuencia de los puntos anteriores, se han detectado algunos problemas de seguridad en el protocolo [16].

**OAuth 2.0**


Para solucionar parte de esos problemas, durante algún tiempo se trabajó en definir una versión de OAuth más sencilla, llamada OAuth WRAP [17], que fue abandonada en favor de la nueva versión 2.0 de OAuth.

**OAuth 2.0** [18] pretende ser una versión revisada y simplificada de OAuth, y a pesar de estar todavía en versión borrador, ya ha sido oficialmente **adoptada por grandes compañías** como Facebook, Twitter, Yahoo!, Google o Microsoft. Aventaja a la versión anterior en una mayor simplicidad de implementación, y en una arquitectura más robusta y que da soporte a mayor número de plataformas.

> *Desde hace un tiempo, Héctor se encuentra con aplicaciones que solicitan acceder a su información y recursos en diferentes servicios (por ejemplo, aplicaciones que utilizan sus datos de Facebook, o publican mensajes en Twitter automáticamente). Esto es posible gracias a OAuth, aunque eso es transparente para Héctor, que solo debe decidir si autoriza o no el acceso.*
>
> *Este tipo de funcionalidades son potentes y permiten la integración de datos y funcionalidades entre diferentes aplicaciones, aunque Héctor en ocasiones piensa que eso es a costa de sacrificar en cierta medida su privacidad, ya que cada vez más aplicaciones tienen acceso a sus recursos personales.*

**¿Es posible autenticación de usuarios con OAuth?**

Existe cierta confusión sobre si OAuth es o no un protocolo de autenticación de usuario. Estrictamente hablando **no lo es**, ya que no define ningún mecanismo explícito destinado a autenticar la identidad del usuario.

Sin embargo, uno de los pasos de este protocolo de autorización es la autenticación del usuario final en el servidor proveedor de OAuth para poder dar autorización a otros a acceder a tus recursos, antes tienes que autenticar quién eres tú realmente.

Por tanto, cuando se habla del "mecanismo de autenticación de OAuth", en realidad **se están refiriendo al mecanismo de autenticación propio del sitio web proveedor de OAuth**, que puede ser cualquiera: autenticación http básica, OpenID, etc.

## 2.3. OpenID OAuth Hybrid Protocol

Como hemos visto, OpenID y OAuth son protocolos con objetivos distintos aunque complementarios: autenticación de usuario (federada) y autorización, respectivamente. El **protocolo híbrido OpenID OAuth** [19] combina ambos sistemas, integrándolos en una **interfaz única** (Ilustración 5); de ese modo, el usuario se autentica en un servidor externo utilizando su proveedor de OpenID, y al mismo tiempo autoriza al servidor externo a que acceda a determinados recursos del proveedor. Sin este híbrido, la utilización de ambos protocolos implica que el usuario deba realizar dos acciones: la autenticación primero, y después la autorización.


Ilustración 5. Ejemplo de interfaz integrada de autenticación (con OpenID) y autorización (con OAuth).

Este protocolo híbrido ha sido adoptado por grandes proveedores como Google, Yahoo! o MySpace [20].

## 2.4. Facebook Connect


Debido al enorme incremento de usuarios y, en consecuencia, de datos personales que ha experimentado Facebook [21] en los últimos años, la compañía lanzó en 2008 su propio sistema conocido como **Facebook Connect** [22].

Con ese movimiento, Facebook pretendía posicionarse como repositorio central de identidad de los usuarios en Internet, ofreciéndose como proveedor de servicios tanto de **autorización como de autenticación**, compitiendo y yendo más allá de los que proporcionan los protocolos abiertos OpenID y OAuth:

- Autenticación (delegada) en servidores externos.
- Identidad real (Facebook intenta evitar la creación de usuarios "ficticios").
- Acceso de "amigos" a los datos.
- Privacidad dinámica (la configuración de privacidad se va actualizando con el tiempo).

En la actualidad Facebook parece haber **abandonado** Facebook Connect [23] para adoptar el protocolo 2.0 de OAuth como sistema de autorización, utilizándolo como base de su protocolo para la integración de servicios de redes sociales en páginas externas.

## 2.5. OpenID Connect


El protocolo **OpenID Connect** [24] es la última propuesta para reactivar **OpenID**. Su propósito es redefinir y simplificar el protocolo construyéndolo **sobre el protocolo OAuth**; de ese modo se aprovecha el trabajo desarrollado para OAuth, que parece estar extendiéndose rápidamente, dotándolo de una funcionalidad estándar de autenticación que, como hemos visto anteriormente, no posee. Esa propuesta permitiría, además, simplificar la implementación de OpenID que, como hemos visto anteriormente, ha recibido críticas por su excesiva complejidad.

La intención es, por tanto, seguir con la **idea original descentralizadora de OpenID**, permitiendo que el usuario pueda elegir entre diferentes proveedores de identidad, aprovechando el alto grado de adopción que está consiguiendo OAuth y que permite implementar OpenID sobre él [25].

## 2.6. Otros sistemas

Algunos de los grandes proveedores de servicios en Internet han definido, en algún momento, su **sistema propio** de autenticación y/o autorización. Sin embargo, la mayoría de ellos están adoptando **estándares** como OpenID y, especialmente, OAuth; es el caso, por ejemplo, de MySpace [26], Twitter [27], o Yahoo! [28].

Otro estándar abierto para el intercambio de recursos de identidad es **Security Assertion Markup Language (SAML)** [29]. Está basado en XML, y su principal propósito es servir de marco para protocolos de autenticación federada. Este protocolo sirve de base para algunos sistemas propietarios de *single-sign-on* [30], pero no es utilizado por los grandes proveedores de servicios en Internet.

## 2.7. Tabla comparación / resumen

En la siguiente tabla se muestra un resumen comparativo de diferentes sistemas que se han mencionado en este trabajo.

| Protocolo | Propósito | Última versión / estado | Ventajas | Inconvenientes |
|---|---|---|---|---|
| **OpenID** | Autenticación (federada) | 2.0 (de 2007) | *Single-sign-on* sin depender de ningún proveedor específico (federado). | Protocolo complejo y antiguo; potenciales problemas de seguridad, privacidad, usabilidad. Pocos incentivos para ser consumidor. |
| **OAuth** | Autorización | 2.0 (en borrador, pero muy utilizada) | Muy utilizado en servidores de Internet. | Dependencia de un servidor para la autenticación (delegada). |
| **OpenID OAuth Hybrid Protocol** | Autenticación (federada) + autorización; interfaz única | Borrador (pero adoptado por grandes proveedores) | Una única interfaz para autenticación (OpenID) + autorización (OAuth). | Depende de que los servidores implementen ambos protocolos. |
| **Facebook Connect** | Autenticación (delegada) + autorización + funciones red social | Abandonado en favor de OAuth 2.0 | Parte de una amplia base de usuarios y datos personales. | Muy ligado a un único proveedor. Problemas de privacidad. |
| **OpenID Connect** | Autenticación (federada) a partir de OAuth | Propuesta (sobre OAuth 2.0) | Las mismas ventajas que OpenID con una implementación más sencilla; aprovecha las implementaciones de OAuth 2.0. | Similares a OpenID; pocos incentivos para ser consumidor. |

# 3. DECISIONES Y ELECCIÓN DE UN SISTEMA

Ante esta variedad de sistemas, los usuarios se ven con frecuencia en la situación de tomar determinadas decisiones, en función de que sean desarrolladores o responsables de proveedores de servicios, o bien usuarios finales.

## 3.1. Para el proveedor de servicios

Los **pequeños proveedores** de servicios, sin una gran base de usuarios, pueden facilitar el acceso de sus usuarios delegando la autenticación mediante OpenID y/o en algún gran proveedor de servicios (Facebook, Twitter, etc.); adicionalmente, pueden optar por obtener recursos personales

de sus usuarios mediante OAuth. El uso de esos protocolos facilitará la gestión de los usuarios y sus datos, a costa de renunciar a ser los repositorios de esos datos; deberán utilizar los de terceros.

> *Un amigo de Héctor está desarrollando una pequeña aplicación en Internet para compartir información sobre mascotas. Podría utilizar su propio sistema de autenticación, pero si utiliza un proveedor externo (como Facebook o Google), se ahorrará tener que implementarlo, y además sus usuarios no tendrán que recordar una contraseña más. La desventaja es que dependerá de un servidor externo y de que los usuarios tengan cuenta en ese servidor.*
>
> *Adicionalmente, la aplicación necesita datos como la dirección de correo electrónico o la ciudad de residencia de los usuarios. El amigo de Héctor puede optar por que su aplicación solicite y almacene esos datos en su propio sistema, o bien solicitar autorización para leer esa información desde, por ejemplo, Facebook (donde es probable que esté actualizada).*

Por otro lado, los **grandes proveedores de Internet** (Facebook, Google, Twitter, etc.) están en la **lucha por ser los gestores de la identidad** y los datos de los usuarios [31], que son un valioso recurso (por ejemplo, para ofrecerles publicidad). En ese sentido, les conviene facilitar el uso de sus datos a terceros (siendo proveedores de autenticación y de recursos personales), pero no tanto aceptar los servicios de autenticación y/o recursos de otros.

## 3.2. Para el usuario final

Podemos partir de la hipótesis de que, por comodidad, la mayor parte de usuarios en Internet estarán interesados en los **sistemas que permitan centralizar sus identidades**. En tal caso, la decisión más importante que deben tomar es: ¿cuál será ese proveedor central? Es probable que esa decisión venga marcada por diversos criterios:

- existencia de cuentas de usuario anteriores, con datos personales ya introducidos;
- posibilidad de utilizar esas cuentas en otros servidores

Con factores como esos, los principales candidatos a ser los **repositorios de identidad** de muchos usuarios serán los "**grandes**": Facebook, Google, etc. Estos proveedores ya disponen de gran cantidad de cuentas y de datos de usuarios, y también se esfuerzan por ofrecerlos para su uso por parte de terceros (en muchas ocasiones mediante el protocolo OAuth).

Los usuarios quizás más **preocupados por su privacidad** pueden también optar por un **protocolo federado** de autenticación como OpenID y seleccionar un proveedor que no sea de uno de esas grandes compañías (incluso puede ser un servidor propio). Pero no es una decisión que parezca al alcance de los usuarios menos expertos, y además, como hemos visto, bastantes de los grandes proveedores de servicios no aceptan ese protocolo para acceder a sus sistemas.

Por último, los usuarios pueden también optar por **no centralizar** la gestión de su identidad, utilizando identidades diferentes en cada servidor. Sin embargo, esta alternativa resulta poco práctica ante el creciente número de servicios a los que se accede en Internet, y que resulta en los potenciales problemas que veíamos al principio.

*Cada vez más servicios en Internet le dan a elegir a Héctor entre diferentes opciones de autenticación: con Twitter; con Facebook; con OpenID;... Héctor, por comodidad, cada vez tiende más a utilizar Facebook, aceptado por la mayoría de ellos. También se da cuenta de que es más frecuente que las aplicaciones soliciten utilizar recursos personales que tiene en Facebook, aunque eso solo lo autoriza cuando ve muy claro su propósito.*

*En alguna ocasión Héctor se ha planteado que está dejando demasiado en manos de un servidor propiedad de una empresa privada, pero las alternativas son, o bien otra empresa (como Google o Twitter), o bien un sistema como OpenID que no es aceptado por muchos de los servidores que utiliza, y que además no acaba de entender bien.*

## 4. CUESTIONES DE INTERFAZ DE USUARIO

No cabe duda de que los diferentes sistemas de autenticación y autorización influyen de manera importante en la interfaz de los usuarios finales y en las acciones que necesitan realizar para acceder y utilizar diferentes recursos en sus proveedores de servicios. Los protocolos descritos en este trabajo tienen un **objetivo beneficioso** para los usuarios: evitar que tengan que recordar o gestionar diferentes datos de autenticación para acceder a diferentes servidores (a través de los estándares de autenticación), y/o evitar tener que almacenar y mantener sus recursos y datos personales en diferentes ubicaciones (mediante los protocolos de autorización).

Sin embargo, el uso de estos protocolos también tiene sus **puntos débiles**. Para empezar, los usuarios se ven en ocasiones enfrentados a **decisiones de carácter técnico** que pueden no comprender demasiado bien: por ejemplo, pueden verse obligados a **elegir** entre un sistema de autenticación u otro, o elegir un proveedor determinado de autenticación (Ilustración 6), o decidir si permiten que un determinado servidor acceda a sus recursos [32]. Todas esas opciones hacen que la interfaz sea habitualmente confusa, siendo muy probable que la mayoría de usuarios no disponga de los criterios suficientes para tomar la decisión más adecuada para ellos.

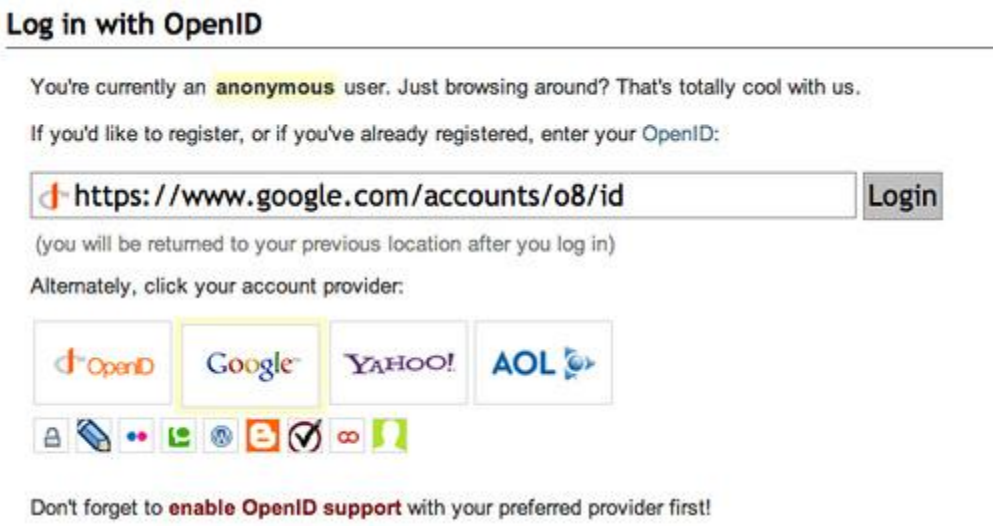


Ilustración 6. Múltiples opciones de autenticación en una única interfaz.

Por otro lado, una vez elegido un proveedor externo, la **interfaz de autenticación o autorización suele ser diferente** a la del servidor al que intentan acceder. Esto puede provocar incoherencias entre ambos sistemas, o incluso dudas sobre cuál es exactamente la función que se está

realizando (Ilustración 7). Además, como ya hemos visto, este uso de diferentes interfaces hace que los sistemas sean más propensos a ataques contra su seguridad, como ocurre con OpenID y el *phishing*.


Ilustración 7. Ejemplo de múltiples opciones confusas: elegir Facebook como sistema de autenticación (para un comentario) y, a su vez, elegir Yahoo! como sistema de autenticación en Facebook.

> *Cuando Héctor llega a una pantalla de autenticación que le solicita un usuario y contraseña, inmediatamente entiende su funcionamiento (otra cosa es que recuerde la contraseña). Sin embargo, las pantallas en que se le ofrecen diferentes sistemas de autenticación le siguen pareciendo confusas, y siempre le obligan a pensar durante unos instantes qué opción es la que le interesa, o la que utilizó la última vez.*
>
> *Es más; cuando utiliza su cuenta de Facebook para autenticarse en otros servicios menos conocidos, en ocasiones todavía tiene dudas sobre si no estará comprometiendo su contraseña o dando acceso a sus datos personales sin darse cuenta.*

Otra problemática a tener en cuenta es el caso de los usuarios que **ya tienen una cuenta en un determinado proveedor de servicios**, y empiezan a utilizar una autenticación de terceros; para evitar que el servidor los considere como usuarios distintos, es necesario que implemente alguna funcionalidad que permita al usuario "**enlazar**" ambas identidades, cosa que no resulta trivial.

> *En uno de los servicios que utiliza esporádicamente, Héctor pasó de autenticarse con usuario y contraseña a hacerlo con un proveedor externo (Facebook). Para ello, tuvo que utilizar una opción específica del servicio para indicarlo, ya que de otro modo lo hubiera considerado como usuarios diferentes.*

Así pues, las interfaces de usuario para este tipo de sistemas tienen que asumir un **compromiso** entre **proporcionar la suficiente información** al usuario de las opciones que tiene, y **ocultarles los detalles técnicos** que no tienen por qué conocer. Ya se han dado algunos pasos hacia la simplificación de las tareas a realizar por el usuario final, como OpenID OAuth Hybrid Protocol (descrito anteriormente), que integra en una única interfaz la autenticación mediante OpenID y la autorización mediante OAuth; o la **plataforma XAuth** [33], un **repositorio central** en el que los diferentes proveedores de servicios notificarían y consultarían cuáles son los servicios que está utilizando el usuario en un determinado momento, para adaptar la interfaz de usuario a la situación concreta (Ilustración 8).

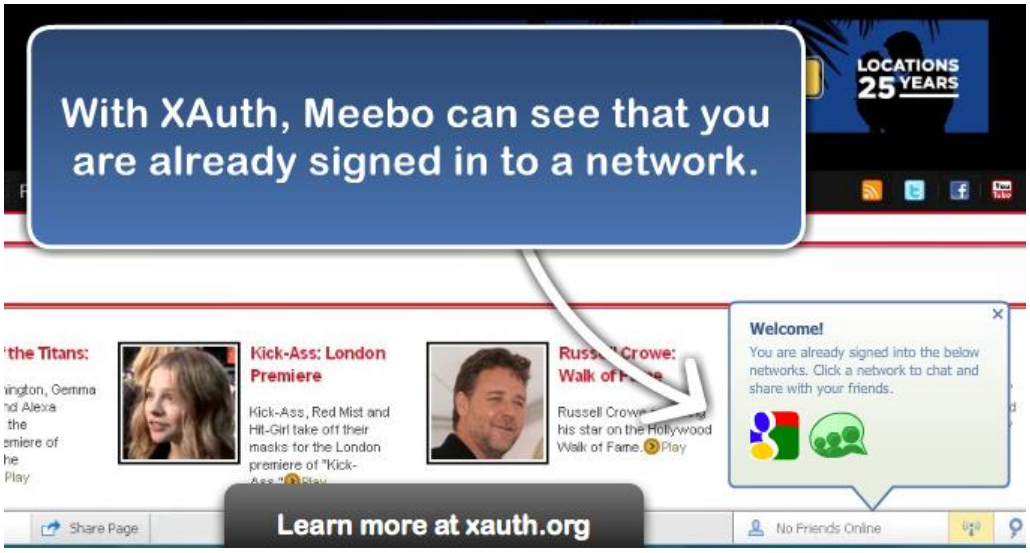


Ilustración 8.Ejemplo de utilización de XAuth en un proveedor (Meebo).

*Utilizando el servicio XAuth, los servidores en Internet podrían detectar si Héctor se ha autenticado previamente en Facebook durante esa sesión de navegación, y ofrecerle utilizar ese mismo sistema para autenticarse contra su sistema; de ese modo se evitaría una interfaz más compleja en la que se ofrecieran todas las opciones posibles cada vez.*

En definitiva, en la actualidad **no existe una solución única ideal** para las interfaces de usuario en este tipo de sistemas, más allá de seguir una serie de buenas prácticas determinadas a partir de algunos proyectos de investigación [34] [35].

## 5. ASPECTOS DE COMPUTACIÓN DISTRIBUIDA

Por definición, los sistemas descritos en este trabajo son de tipo distribuido, aunque la motivación de la arquitectura no proviene de una decisión por motivos de potencia computacional o de almacenamiento de datos, sino que **su propia naturaleza distribuida es la que los hace adecuados** para resolver el problema de la autenticación y autorización en sistemas heterogéneos.

Así pues, hay cuestiones generales de los sistemas distribuidos que afectan específicamente a los contemplados en este trabajo:

- Los sistemas proveedores deben ser **suficientemente potentes y robustos** como para proporcionar respuesta a las peticiones de diferentes consumidores.
- Los sistemas consumidores **trasladan parte de su problemática** de proceso, de almacenamiento de datos y de seguridad al proveedor.
- Por lo general, los sistemas contemplados son **poco robustos**, en el sentido de que la caída o malfuncionamiento de cualquiera de los servidores implicados lleva a que no se pueda completar la operación.

# 6. CONCLUSIONES Y SITUACIÓN ACTUAL

La principal motivación para el desarrollo de protocolos abiertos de autenticación y autorización en Internet es el **desarrollo espectacular de redes sociales** en Internet; ya no es una alternativa práctica que cada aplicación de ese tipo construya y mantenga su propia base de datos de usuarios, datos personales, relaciones, etc.

Protocolos y APIs de redes sociales como **Open Graph** [36] y **OpenSocial** [37] han surgido como propuestas para permitir que Internet se convierta en una verdadera red social global, donde las aplicaciones no son "jardines privados" sino que enlazan e intercambian datos y funcionalidades entre ellos. Este tipo de protocolos requieren, sin embargo, de **un paso básico: el usuario debe autenticar quién es, y qué permite hacer a los demás con sus datos**. Esa es la problemática en la que se ha centrado este trabajo.

En la actualidad parecen existir **dos tendencias** a la hora de dar soluciones a ese problema. Por un lado, los grandes servidores de redes sociales (como Facebook, Twitter o Google) pretenden convertirse en los elegidos por los usuarios como su **gestor principal de identidad**; eso les convierte en el repositorio donde los usuarios almacenan sus datos personales (además actualizados), lo que constituye un gran valor para las compañías. En ese sentido, es beneficioso para ellas convertirse en proveedores de autorización y autenticación, principalmente a través del protocolo OAuth 2.0, que a pesar de ser todavía un borrador, está siendo ampliamente aceptado por muchos proveedores; la lucha no es tanto por el uso de un protocolo u otro, sino **por convertirse en el proveedor elegido por los usuarios**. Eso significa que existe un fuerte incentivo para que los servidores se conviertan en proveedores y que terceros consuman sus datos y servicios, pero el incentivo para convertirse en consumidores de datos de terceros es menor.

Por otro lado, protocolos como OpenID y OpenID Connect representan un esfuerzo no sólo hacia la definición de protocolos abiertos, sino hacia la **descentralización de la gestión de identidad y de datos personales**. Si el interés de los proveedores de redes sociales es convertirse en el punto donde los usuarios tienen su identidad principal para atraer y gestionar sus datos, la propuesta de OpenID y OpenID Connect es que se pueda elegir entre diferentes servidores para ser utilizados como proveedores de identidad, y evitar esa centralización que puede provocar problemas de **privacidad**.

¿Hacia dónde evolucionará la situación en un futuro? Es difícil decirlo, aunque podría aventurarse que evolucionará hacia la típica situación de ***long tail*** tan habitual en Internet [38], en la que unos **pocos sitios webs** populares se conviertan en los gestores centrales de identidad de un **gran número de usuarios**, mientras que exista una **larga cola de proveedores minoritarios** pero que, en conjunto, administren la identidad de un número importante de usuarios, quizás los más preocupados por cuestiones de privacidad.

# REFERENCIAS

[1] InfoWorld. **Study finds hight rate of password reuse among users.**
http://www.infoworld.com/t/data-security/study-finds-high-rate-password-reuse-among-users-188

[2] Wikipedia. **Single sign-on.**
http://en.wikipedia.org/wiki/Single_sign-on

[3] Boston University. **Understanding Authentication, Authorization, and Encryption**.
http://www.bu.edu/tech/security/protect/bestpractice/auth/

[4] Nothing to See Here. **Delegated vs. Federated ID**.
http://www.personal.psu.edu/mhl100/blogs/2010/02/delegated-vs-federated-id.html

[5] International Telecommunication Union (ITU). **Recommendation X.509.**
http://www.itu.int/rec/T-REC-X.509

[6] DNI electrónico. **Certificados electrónicos**.
http://www.dnielectronico.es/obtencion_del_dnie/certificados.html

[7] Ceres. Fábrica Nacional de Moneda y Timbre. **Certificado de usuario con DNIe**.
http://www.cert.fnmt.es/index.php?cha=cit&sec=4&page=184&lang=es

[8] **OpenID Foundation website**.
http://openid.net/

[9] OpenID. **Get an OpenID**.
http://openid.net/get-an-openid/

[10] The Identity Corner. **The problem(s) with OpenID**.
http://www.untrusted.ca/cache/openid.html

[11] Dare Obasanjo aka Carnage4Life. **Learning from our Mistakes: The Failure of OpenID, AtomPub and XML on the Web**.
http://www.25hoursaday.com/weblog/2011/01/30/LearningFromOurMistakesTheFailureOfOpenIDAtomPubAndXMLOnTheWeb.aspx

[12] Wikipedia. **Phishing**.
http://en.wikipedia.org/wiki/Phishing

[13] TheServerSide.COM. **More patent joy: Sun says it won't enforce patents on OpenID**.
http://www.theserverside.com/news/thread.tss?thread_id=45552

[14] **OAuth Community Site**.
http://oauth.net/

[15] Ars Technica. **OAuth and OAuth WRAP: defeating the password anti-pattern**.
http://arstechnica.com/open-source/guides/2010/01/oauth-and-oauth-wrap-defeating-the-password-anti-pattern.ars

[16] Ars Technica. **Compromising Twitter's OAuth security system**.
http://arstechnica.com/security/guides/2010/09/twitter-a-case-study-on-how-to-do-oauth-wrong.ars

[17] OAuth Wiki. **OAuth WRAP**.
http://wiki.oauth.net/w/page/12238537/OAuth-WRAP

[18] OAuth.**OAuth 2.0**.
http://oauth.net/2/

[19] Step2. **Draft: OpenID OAuth Extension**.
http://step2.googlecode.com/svn/spec/openid_oauth_extension/latest/openid_oauth_extension.html

[20] OpenID. **OpenID: Now more powerful and easier to use!**
http://openid.net/2009/09/25/more-powerful-and-easier-to-use/

[21] **Facebook**.
http://www.facebook.com/

[22] Facebook Developers. **Announcing Facebook Connect**.
http://developers.facebook.com/blog/post/108/

[23] Mashable. **Facebook to Kill Facebook Connect**.
http://mashable.com/2010/04/21/facebook-kills-facebook-connect/

[24] **OpenID Connect**.
http://openidconnect.com/

[25] FactorCity. **Two tastes better together: Combining OpenID and OAuth with OpenID Connect**.
http://factoryjoe.com/blog/2010/05/16/combing-openid-and-oauth-with-openid-connect/

[26] MySpace Developer Platform. **MySpaceID**.
http://developer.myspace.com/myspaceid/

[27] Twitter Developers. **Authenticating Requests with OAuth**.
http://dev.twitter.com/pages/auth

[28] Yahoo! Developer Network. **OAuth / OpenID Guides**.
http://developer.yahoo.com/oauth/guide/

[29] **Online community for the Security Assertion Markup Language (SAML) OASIS Standard**.
http://saml.xml.org/

[30] **Ping Identity Corporation**.
https://www.pingidentity.com/

[31] Mashable. **Facebook vs. Google and the Battle for Identity on the Web**.
http://mashable.com/2010/11/11/facebook-google-identity-column/

[32] FactoryCity. **Does OpenID need to be hard?**
http://factoryjoe.com/blog/2009/04/06/does-openid-need-to-be-hard/

[33] XAuth. **XAuth Info**.
http://xauth.org/info/

[34] **Google's Internet Identity Research.**
https://sites.google.com/site/oauthgoog/Home

[35] Yahoo! Developer Network. **Yahoo! OpenID Usability Research**.
http://developer.yahoo.com/openid/bestpractices.html

[36] **The Open Graph Protocol**.
http://ogp.me/

[37] **OpenSocial**.
http://www.opensocial.org/

[38] Wikipedia. **Long Tail**.
http://en.wikipedia.org/wiki/Long_Tail