UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

Título: ESLIP - Easy Social Login Integration Plugin
Autores: Burghi Nicolás, Estigarribia Martín Miguel
Directores: Claudia Queiruga, Claudia Banchoff
Codirector: ---
Asesor profesional: ---
Carrera: Licenciatura en Sistemas

Resumen

Social Login es una metodología de identificación, la cual le permite al usuario registrarse en un sitio web utilizando una credencial creada en un proveedor de identidad, tal como las redes sociales Facebook o Twitter, sin la necesidad de crear un nuevo usuario específico para este sitio. Fue diseñado para simplificar el proceso de identificación para los usuarios finales, así como también para brindar información más fiable para los desarrolladores web.
Si bien existen varios protocolos para implementar la identificación o autorización de un usuario en un sitio web, los dos más utilizados actualmente son OAuth y OpenID, los cuales fueron usados para realizar el desarrollo de ESLIP.
Si bien existen gran cantidad de organizaciones que proveen servicios para facilitar la incorporación de Social Login, la mayoría de ellas son pagas, y además terminan siendo un intermediario entre el sitio web que contrata el servicio y el proveedor de identidad. Por otro lado, existen herramientas de código abierto pero son difíciles de integrar o poseen escasa documentación. En consecuencia se decidió crear ESLIP (Easy Social Login Integration Plugin), en español Plugin Simple para Integración de Social Login el cual es una herramienta que permite a los desarrolladores web incorporar de forma sencilla Social Login a sus aplicaciones o sitios web.

Palabras Claves

Social Login. Plugin. Proveedores de Identidad. Redes Sociales. OAuth. OpenID. Identificación. Autenticación

Trabajos Realizados

Se diseñó e implementó ESLIP, una herramienta que facilita a los desarrolladores a incluir Social Login en sus sitios web.
ESLIP cuenta con un asistente de configuración que guía al desarrollador para que en pocos pasos pueda configurar su widget de Social Login. A su vez cuenta con un módulo de administración desde el cual es posible agregar y quitar proveedores de identidad, habilitar y deshabilitar los mismos del widget, configurar el idioma, entre otras opciones.

Conclusiones

En esta tesina hemos puesto de manifiesto las ventajas y desventajas que tienen tanto un usuario al identificarse en un sitio web utilizando Social Login como el desarrollador del mismo.
La investigación realizada sobre los fundamentos de Social Login y los protocolos que se ven involucrados en esta metodología de identificación nos sirvieron como base para desarrollar ESLIP, una herramienta que facilita a los desarrolladores a incluir Social Login en sus sitios web.

Trabajos Futuros

En una primera etapa, se espera poder optimizar el diseño del widget de login, de acuerdo a los requerimientos de usabilidad que se planteen, brindando más opciones de personalización.
Otro trabajo que sería de mucha utilidad es la elaboración de diferentes versiones de ESLIP en forma de librerías o plugins para que pueda ser utilizado en diferentes frameworks y CMS's.
Además se prevé traducir ESLIP a más idiomas, ya que actualmente está disponible en español e inglés.

Fecha de la presentación: Junio de 2014

Agradecimientos

Nicolás Burghi

A mis padres Susana y Juan Carlos por el incansable esfuerzo que realizaron para darme la posibilidad de seguir esta carrera.

A mi novia Lucrecia quién siempre me brinda su cariño y apoyo incondicional para seguir adelante y conseguir mis objetivos

A mi amigo y compañero de Tesina Martín, con quién compartí de principio a fin esta hermosa etapa de la vida y confió en mí para realizar este trabajo.

Gracias.

Martín Estigarribia

A toda mi familia por acompañarme y ayudarme durante todos estos años y darme las fuerzas necesarias para afrontar los obstáculos de la vida.

A mi novia Florencia y toda su familia por el apoyo incondicional que siempre me brindan.

A mi amigo y compañero de Tesina Nicolás, con quién recorrí todo este camino y siempre me brindó su comprensión y paciencia en todo momento.

Gracias.

Nicolás Burghi y Martín Estigarribia

A nuestras directoras de Tesina Claudia Banchoff y Claudia Queiruga, por guiarnos y brindarnos todo el apoyo necesario para lograr este trabajo.

Gracias.

Índice

Capítulo 1

Introducción

1.1. Objetivos

Este trabajo de tesis tiene como objetivo realizar una investigación teórica sobre Social Login [1] y los protocolos más importantes que se ven involucrados en esta tecnología, así como también implementar un plugin al que bautizamos ESLIP (**E**asy **S**ocial **L**ogin **I**ntegration **P**lugin) que le permitirá a un desarrollador web integrar de manera sencilla e intuitiva Social Login en una aplicación o sitio web.

En lo que respecta a la investigación, se propuso estudiar el estado del arte de esta forma actual de identificación, cómo surgió, beneficios y alternativas disponibles en la actualidad para su integración. Se describe cómo el método de autenticación por medio de Social Login favorece a las empresas e instituciones que basan sus negocios en un sitio web y cómo afecta a la seguridad, analizando los posibles riesgos que pueden afrontar tanto los usuarios (ya que se ven involucrados datos personales) como también los sitios web que permiten este tipo de autenticación.

A su vez, se evalúa el funcionamiento de los dos protocolos más utilizados y populares: OAuth [2] y OpenID [3]. Éstos son las principales herramientas que los proveedores de identidad utilizan en la implementación de sus API's para proveer autenticación y autorización.

En base a lo analizado previamente se implementó el plugin ESLIP. El cual está destinado a los desarrolladores web, poniendo énfasis en la simpleza y facilidad de uso. Los usuarios desarrolladores deben configurar ciertos parámetros, como por ejemplo algunos datos de las aplicaciones, las cuales previamente deben ser creadas por ellos en los proveedores de identidad, que se utilizarán para poder identificarse en el sitio web. Esta configuración es muy simple y se lleva a cabo a través de un wizard [4] que guía al usuario en cada uno de los pasos. Una vez configurados los parámetros necesarios a través del wizard se provee un código para incluir en las páginas en las que se quieran visualizar la interfaz de Social Login.

1.2. Contexto y Motivación

Un proveedor de identidad es un sistema que permite que usuarios de Internet se identifiquen en él y además, ofrece un servicio de emisión de información de identidad para todos los sistemas (proveedores de servicios) que quieran obtener información o verificar la identidad de un usuario con el consentimiento del mismo. Para esto, primero el usuario debe iniciar sesión en el proveedor de

identidad para que luego éste se comunique con el proveedor de servicios que desea obtener información.

Gran parte de la población que tiene acceso a Internet está registrada al menos en un sitio que funciona como proveedor de identidad, siendo, las redes sociales, el tipo de sitio que predomina en este conjunto gracias a la popularidad creciente que han ganado en este último tiempo.

A raíz de esto, es que empieza a ganar terreno en la web algo conocido como Social Login, un modelo de autenticación y autorización que permite a los usuarios iniciar sesión en un sitio web particular utilizando sus cuentas de Facebook, Google, Twitter, Hotmail, entre otros proveedores de identidad.

Teniendo en cuenta, como se dijo anteriormente, que la mayoría de los usuarios es miembro de alguno de estos proveedores, los sitios web que quieren identificar a sus usuarios, están optando por incluir Social Login en vez de obligar al usuario a darse de alta completando un formulario de registro. De esta manera se le brinda al usuario la posibilidad de autenticarse utilizando la información que él habilita del proveedor de identidad que eligió para identificarse, lo cual favorece enormemente la accesibilidad al sitio en cuestión.

Por consecuencia, han ido surgiendo gran cantidad de empresas u organizaciones que proveen servicios para facilitar la incorporación de Social Login a un sitio web. De todas formas, estas empresas u organizaciones que proporcionan formas sencillas de integración, y que poseen buena documentación o soporte, son pagas [5], es decir, cobran un arancel por proveer el servicio de Social Login, y además terminan siendo un intermediario entre el sitio web que contrata el servicio y el proveedor de identidad, ya que los datos que se extraen de la red social o servicio en línea que el usuario elige para autenticarse, pasan antes por los servidores del prestador del servicio de Social Login y luego llegan al sitio web que el usuario está navegando.

Por otro lado, existen soluciones que son gratuitas, de código abierto [6] y en las cuales el manejo de la información queda bajo el control total del desarrollador web, pero actualmente hay muy pocas, y la mayoría de ellas son difíciles de integrar o poseen escasa documentación e incluso muchas están a medio desarrollar o manejan muy pocos proveedores de identidad y no son extensibles.

1.3. Organización del documento

Este informe se encuentra dividido en diversos capítulos que abarcan temas relacionados entre sí.

En el capítulo 2 se explica detalladamente el concepto de Social Login, una alternativa al tradicional inicio de sesión y proceso de registro en los sitios web. Entre otras cosas, se menciona por qué utilizarlo, así como también los factores a tener en cuenta, junto con sus ventajas y desventajas.

En el capítulo 3 se plantea un marco teórico de los distintos protocolos que se utilizan para implementar la identificación o autorización de un usuario en un sitio web. Conjuntamente se describen los conceptos en los que se hará mayor hincapié a lo largo de los diferentes capítulos, con el objetivo de facilitar la comprensión de la posterior lectura.

En el capítulo 4 se aborda el protocolo OpenID, el cual es un estándar de identificación digital descentralizado que permite, a una persona, usar una URL [7] como identidad. Se describe su funcionamiento, cuestiones referentes a la seguridad y se presentan sus beneficios e inconvenientes.

En el capítulo 5 se describe el protocolo OAuth, el cual permite que los usuarios aprueben que una aplicación actúe en su nombre sin compartir su nombre de usuario y contraseña. Se describe su funcionamiento, conjuntamente con sus ventajas y desventajas.

En el capítulo 6 se presenta ESLIP, un plugin que permite integrar fácilmente Social Login a un sitio web. Describiendo cómo surgió, argumentando los beneficios de su utilización. Así como también brindando detalles de su implementación y uso.

En el capítulo 7 se presentan las conclusiones finales obtenidas y los trabajos futuros que se esperan realizar con el fin de optimizar ESLIP.

Capítulo 2

Social Login

2.1. Introducción

Social Login es una metodología de identificación, la cual le permite al usuario registrarse en un sitio web utilizando una credencial creada en un proveedor de identidad, tal como las redes sociales Facebook o Twitter, sin la necesidad de crear un nuevo usuario específico para este sitio. Fue diseñado para simplificar el proceso de identificación para los usuarios finales, así como también para brindar información más fiable para los desarrolladores web.

Este mecanismo persigue el objetivo de que el ingreso en un sitio web debe ser una tarea fácil y gratificante, no debe ser un sufrimiento para los usuarios. Por tal motivo, ofrece una manera sencilla de identificarse.

Social Login no sólo acelera el proceso de registro en un sitio web, si no que provee información personal (en la mayoría de los casos real) del usuario con sólo un clic y hace que sean obsoletos los formularios de registro, brindando como ventaja para el usuario no tener que recordar una contraseña más.

2.2. ¿Por qué utilizarlo?

En pocas palabras la principal razón por la que se utiliza Social Login es porque facilita el registro a un sitio web.

La mayoría de los usuarios de Internet se encuentran registrados en las páginas que utilizan con cierta frecuencia, y es sabido que denotan cierto cansancio a la hora de tener que introducir, aunque sea cada cierto tiempo, sus datos de autenticación (usuario y contraseña) en cada una de ellas. En consecuencia, existen varias alternativas para tener que evitarlo. Una de ellas podría ser recurrir a complementos del navegador o aplicaciones específicas, pero, sin embargo, una de las alternativas más populares, hoy en día, son los logins sociales, más conocidos como “Social Login”.

Seguramente, la mayoría de los usuarios promedio de Internet, en más de una ocasión han entrado a una página que permite identificación directamente mediante una cuenta de Facebook, Twitter, Google u otro servicio web. Cada vez son más los sitios web que ofrecen esta posibilidad, por una obvia razón, porque facilita el registro, como bien se dijo anteriormente.

Contar con Social Login en un sitio no sólo beneficia al registro de usuarios, sino que también permite recolectar información sobre los mismos.

Con respecto a las páginas que eligen si implementar o no Social Login,

existen teorías encontradas. Hay muchas que los ofrecen como alternativas al registro tradicional pero manteniendo también éste. Otras se atreven incluso a apostar por los logins sociales como método único de registro y, a cambio, pierden potenciales usuarios que, o bien no están en estas redes, o no quieren utilizar sus cuentas en sitios de terceros. Finalmente, existen otras que directamente lo descartan.

Más allá de optar por implementar, o no, Social Login, en definitiva, lo que importa es que, al final, es el usuario el que decide si utilizar estos sistemas o no. Existen distintas cuestiones a valorar, por eso mismo, en las siguientes secciones se describen los factores a tener en cuenta, ventajas y desventajas, en cuanto a la utilización de Social Login.

2.3. Factores a tener en cuenta

Implementar Social Login en un sitio web es, potencialmente, una buena jugada. Sin embargo, existen varios factores que se deben tener en cuenta.

2.3.1. Asegurarse que Social Login es la solución adecuada para su organización

Así como por ejemplo, un blog [8] podría no ser lo más adecuado para ser el sitio web de una empresa, Social Login podría no ser la mejor opción para determinados tipos de sitios web. Una de las razones más importantes por la que podría no llegar a serlo, es que cuando un desarrollador (sea una empresa, un grupo de desarrolladores o un individuo) decide implementar Social Login, debe estar preparado para realizar los esfuerzos necesarios para que éste funcione adecuadamente y rinda sus frutos. Por ejemplo, en el caso de un sitio web de ventas online, es importante realizar buenas campañas de marketing en medios sociales, para atraer potenciales clientes. Por lo tanto, la falta de personal capacitado o de posibilidades de inversión pueden ser determinantes.

Resulta entonces, que es de vital importancia analizar si se cuentan con los recursos necesarios para llevar adelante la implementación y puesta en marcha de Social Login, tanto como evaluar si realmente Social Login es la solución que se está buscando.

2.3.2. Facilidad de integración

Es probable que el sitio web en el cual se desea integrar Social Login, esté construido sobre un framework de desarrollo web común y con buen soporte. O puede que se esté utilizando una mezcla de tecnologías. En cualquier caso, no hay garantía de que la integración de Social Login en dicho sitio web vaya a ser fácil.

Lo dicho anteriormente, obviamente, dependerá de la solución de Social Login que se elija y de la personalización que se desee alcanzar. Soluciones como HybridAuth [9], existen para una serie de frameworks de desarrollo web

populares. También existen soluciones empresariales, como Gigya [10] o Janrain [11], que son productos ampliamente utilizados y muy respetados, pero tienen su costo monetario.

A pesar de que este tipo de soluciones están disponibles, es de suma importancia tener en cuenta el tiempo necesario para integrarlas al sitio web. Asimismo, se debe analizar si se cuenta con el personal de desarrolladores adecuado e idóneo, para hacer el trabajo. O, si por el contrario, se decide externalizar la integración, entonces se tendrá que presupuestar cuidadosamente el tiempo y dinero para ese proyecto.

2.3.3. No olvidar los dispositivos móviles

Hoy en día, más y más vendedores en línea y servicios web tienden a ser "mobile" (aptos para teléfonos móviles, o incluso tablets). Los clientes lo exigen y a esta altura la mayoría ya asume que existe una versión "mobile" para cada aplicación web. A la hora de elegir la solución de Social Login, se debe tener en cuenta que si no tiene aún soporte básico para dispositivos móviles, entonces probablemente se estará alejando a un conjunto de clientes y/o visitantes y se estará repeliendo a los potenciales visitantes en el futuro.

Hoy en día, es de vital importancia asegurarse de que la solución de Social Login elegida sea compatible con múltiples dispositivos, y también con múltiples plataformas móviles.

2.3.4. Ofrecer múltiples soluciones

Con alrededor de 600 millones de usuarios, es tentador suponer que ofreciendo una única opción para el inicio de sesión mediante Facebook debería ser suficiente. Muchos sitios utilizan el plugin Facebook Login Button [12] como su tecnología de Social Login, para permitir autenticación mediante Facebook. Eso está bien, pero qué pasa si un bloque sustancial de clientes no tiene una cuenta de Facebook, y qué si no están realmente interesados en crear una.

Como se dijo anteriormente en este informe, elegir un único punto de entrada es casi tan malo como el uso de un registro web tradicional. Lo que es aún peor, es que se estaría bloqueando el acceso a un número considerable de clientes y potenciales clientes. Más aún, a lo mejor, se estaría haciendo más difícil el acceso para ellos.

Por ello, es que el verdadero beneficio de Social Login no se consigue con una única red o proveedor, sin importar que tan grande sea. Por esta razón, a continuación se presentan algunos factores a tener en cuenta al momento de contemplar cuántas opciones ofrecer para el inicio de sesión mediante Social Login, y por qué son importantes:

No ofrecer sólo una opción de Social Login

A pesar del dominio actual de Facebook en el ámbito social, nunca es una

buena idea ofrecer una única opción de Social Login. Para empezar, cualquier plataforma puede sufrir tiempos de inactividad imprevistos u otros problemas futuros que pueden evitar que los usuarios se autentiquen en el sitio. En cambio, tener varias opciones para el inicio de sesión es la mejor forma de mitigar estos problemas.

Más allá de los tecnicismos, una única opción de inicio de sesión mediante un único proveedor de identidad (por ejemplo Facebook) descuida grandes segmentos del mercado que prefieren utilizar otras redes por diversas razones. Gigya, un proveedor de soluciones sociales SaaS [13], hace un tiempo publicó un anuncio [14] donde informó que la mitad de sus usuarios ingresan a su sitio utilizando otras opciones de inicio de sesión que no sea Facebook. Del mismo modo, Janrain, también hace un tiempo informó mediante un artículo publicado en su blog [15] que Facebook ocupa el segundo lugar debajo de Google en su plataforma social.

Para evitar apartar estas grandes cantidades de usuarios potenciales y sus redes sociales, es recomendable asegurarse de tener por lo menos varias opciones viables para la audiencia a la que se apunta. Si es posible, se deben ofrecer por lo menos dos o tres posibilidades para que los clientes puedan iniciar sesión mediante Social Login.

Una comprensión clara sobre el público al que apunta resulta crucial para determinar qué opciones de redes sociales ofrecer. Por lo que se recomienda, como una buena práctica, acudir a los clientes o visitantes para ver qué servicios, o redes sociales están usando y cómo les resultaría más cómodo el ingreso al sitio. Esto puede llevarse a cabo ya sea a través de una encuesta en el sitio o poniéndose en contacto con ellos directamente. Es posible también considerar un ensayo corto para ver si los registros y conexiones aumentan o disminuyen.

Respetar identidades fragmentadas

La mayoría de los usuarios de Internet pasan casi una cuarta parte del tiempo que utilizan la computadora en las redes sociales y la explosión de las redes sociales para todas las tendencias permite a los usuarios gestionar sus relaciones en línea como nunca antes. La mayoría de las personas aprovechan tener varias redes sociales para su beneficio, fragmentando sus identidades y cultivando relaciones distintas en consecuencia, por lo que es difícil apegarse a cualquier elección.

La forma más eficiente para los consumidores que padecen las organizaciones para satisfacer estas necesidades diversas y cambiantes, es ofrecer varias opciones para el inicio de sesión. Si bien esto no sugiere que se deben tener decenas de opciones, sí significa que es recomendable contar con las redes sociales más populares, como se explicó anteriormente.

Aprovechar los datos de los usuarios

Social Login ofrece información valiosa desde la perspectiva del proveedor y

permite a los sitios aprovechar los grafos sociales [16] de sus usuarios. Si bien no hay duda de que Facebook ofrece un tesoro virtual para que los agentes de marketing puedan explorar en los datos de los usuarios en varios frentes, otras redes sociales ofrecen información única de los suyos que pueden ser más importantes para el proveedor.

Más allá de los datos, Social Login permite a los sitios web beneficiarse con las características sociales, como ser lo que comparten los usuarios en su red social, el último producto, artículo o comentario. El tráfico de las redes sociales está demostrando ser el mejor en todo, ya que es muy específico y prácticamente adaptado a usuarios individuales y sus redes. Tener varias opciones para el inicio de sesión asegura la ampliación del alcance del sitio y la conversión [17] simultáneamente.

2.3.5. No olvidarse de la privacidad

Independientemente de lo dicho anteriormente con respecto a los beneficios que se pueden obtener a partir de los datos de los usuarios, no debe olvidarse que la privacidad es importante. Los datos que se están aprovechando pertenecen al usuario, no al sitio que ofrece Social Login, el sitio sólo los usa. Siempre que sea posible, se debe permitir a los clientes optar por sí o por no. Como mínimo, se debe pedir su consentimiento antes de recolectar los datos.

2.3.6. Fomentar los registros de usuarios

Así es como se supone que debería ser: un cliente potencial visita un sitio web, se crea una cuenta, interactúa con el sitio, y se convierte en un cliente de por vida del sitio. Sería muy bueno que fuera así de fácil. El proceso de registro tiene un montón de baches en los que uno puede caer. Maximizar los registros online no es sencillo.

A continuación se presentan un conjunto de medidas que ayudan a aumentar los registros mediante Social Login:

- **Botones de inicio de sesión prominentes**. Utilizar los iconos de redes sociales para atraer la atención del visitante en el sitio web.

- **Rellenar previamente los formularios de registro**. Utilizando los datos de la cuenta de la red social, rellenar previamente los formularios de registro a los clientes para ahorrar tiempo y esfuerzo. Adicionalmente, si el sitio en cuestión necesita una gran cantidad de información para llenar el formulario, el mismo puede dividirse en dos sesiones de conexión diferentes.

- **Compartir a través de las redes sociales**. La mayoría de los proveedores de Social Login también ofrecen intercambio a través de múltiples redes sociales. Esta característica es ideal para referir amigos al sitio web.

- **Recompensar a los clientes de Social Login**. Enviar a los usuarios que han ingresado mediante Social Login un mensaje de agradecimiento con una promoción exclusiva y persuadir para que inviten a sus amigos a inscribirse.

- **Ofrecer acceso tradicional y Social Login, lado a lado**. Es una buena práctica para mantener la opción para que los usuarios se registren o accedan al sitio web con una cuenta tradicional. Esto facilitará la transición a los servicios sociales para los visitantes más tradicionales.

- **Ventana intermedia**. Utilizar una ventana intermedia después de un inicio de sesión tradicional delineando las características y beneficios de Social Login, tales como la comunidad y la capacidad de intercambio social.

2.4. Beneficios

Social Login es una gran solución para un problema que muchas personas enfrentan: la necesidad de tener un único par usuario y contraseña para cada sitio web que utilizan.

Como ya se ha expresado anteriormente, Social Login permite a las personas iniciar sesión en un sitio web con sus nombres de usuario y contraseñas de servicios como Twitter, Facebook, Google, o Yahoo!, entre otros. La herramienta que implementa Social Login integrada en un sitio web se comunica con estos servicios y los mismos confirman si una persona es quien dice ser.

Pero la tecnología Social Login no sólo realiza simplemente autenticación. También puede obtener información acerca de un usuario, puede ocuparse del relleno de los formularios de registro, y otros beneficios que se describen a continuación.

2.4.1. Establece Identidad

Si un usuario se registra en un sitio o simplemente lo está visitando, Social Login le permite, a los propietarios del sitio, verificar la identidad de ese usuario. En general, esto significa que los usuarios "son quienes dicen ser". Esto puede ser muy valioso para los sitios web que ofrecen comentarios, discusiones y otras interacciones entre los usuarios ya que estas interacciones tienden a ser más constructivas. Y puesto que los gigantes de la industria como Facebook y Google pasan gran cantidad de tiempo tratando de limitar las identidades fraudulentas, en consecuencia su sitio se ve beneficiado.

2.4.2. Obtención de datos de los usuarios

No importa qué red social o servicio utilicen las personas, por lo general tienen un perfil, el cual contiene cantidades variables de información sobre la persona.

Además del nombre de una persona y la dirección de correo electrónico, esta información puede incluir su fecha de nacimiento, la ubicación geográfica, el género, y tal vez incluso su grafo social. También puede incluir una lista de sus amigos y contactos.

Qué datos se recopilan depende de la herramienta de Social Login que se esté utilizando. Pero independientemente de la solución que se utiliza, los datos del perfil que se recogen pueden ser una poderosa herramienta para orientar el contenido del sitio a los usuarios. Esto puede conducir a un mayor compromiso y a una mayor lealtad al sitio.

2.4.3. Orientación de contenido

Generalmente las herramientas de Social Login obtienen información del usuario en a partir del perfil del proveedor de identidad que éste utilizó para realizar el ingreso al sitio web. De esta manera, se pueden guardar los datos y utilizar la información para enfocar campañas de marketing en base a los intereses y datos de los usuarios. Esta técnica es válida tanto para sitios web que ofrecen productos o para los que ofrecen un servicio o simplemente para los que quieren compartir su contenido.

2.4.4. Aumento de las tasas de conversión

Los sitios web que ofrecen a los visitantes la opción de identificarse o registrarse utilizando la identidad de alguna red social, verán aumentos en las tasas de conversión de su sitio, así como en el tráfico de referencias sociales, suponiendo que el sitio integra medios de interacción como por ejemplo los botones de “compartir”.

2.4.5. Vinculación de cuentas

Debido a que muchos usuarios prefieren la flexibilidad de usar múltiples identidades, otro de los beneficios de Social Login es que se puede crear con eficacia un sistema de respaldo para la autenticación. Dependiendo de la arquitectura del servidor, las aplicaciones pueden vincular fácilmente múltiples identidades de las redes sociales a una cuenta existente del sitio. Esto significa que los usuarios pueden iniciar sesión con el uso de sus credenciales del sitio o con cualquiera de las múltiples identidades de red.

2.4.6. Reducción del olvido de contraseñas

No hay duda que todos, en algún momento se han olvidado alguna contraseña. Y es sabido que la opción de recuperar contraseña es una gran molestia, ya que requiere tener que crear una nueva contraseña, la cual debe ser recordada.

En octubre de 2011, Janrain le encarga a la empresa de investigaciones Blue Research [18] que realice una investigación [19] para evaluar las actitudes de

los consumidores sobre los procesos tradicionales de registro en línea y las percepciones de la utilización de las identidades sociales existentes para accederá través de la web.
En base a un muestreo de 600 consumidores de Estados Unidos, el 86% de los consumidores está molesto por el registro en sitios web, y cuatro de cada cinco personas se sienten frustrados por la necesidad de crear nuevas cuentas al registrarse en un sitio web. Además, el 88% admite haber dado información incorrecta o incompleta al crear una nueva cuenta en un sitio web, y 9 de cada 10 personas admiten que han dejado de usar un sitio web al olvidarse su contraseña o información de autenticación, en lugar de tratar de recuperar su contraseña.

La posibilidad de acceder mediante un nombre de usuario y contraseña ya conocidos (credenciales de algún proveedor de identidad como Facebook o Twitter) podría aliviar el dolor. De hecho, según el estudio antes mencionado, casi ocho de cada diez personas quieren que se les ofrezca Social Login como una alternativa.

2.4.7. Promueve la utilización de múltiples identidades

Es una realidad que la mayoría de los usuarios de internet están familiarizados con varias cuentas de usuario, sin ir más lejos, resulta difícil encontrar a un usuario de internet que no tenga más de una cuenta. Por ejemplo, una persona puede tener varias cuentas de correo electrónico.

Mientras que a los grandes proveedores de identidad les gustaría que todos se autentiquen alrededor de la web con un solo servicio, es decir con su servicio, hay algunos argumentos muy buenos a favor de que las personas sigan utilizando múltiples identidades.

La confianza cambia

Incluso entre los grandes proveedores, la confianza que depositan los usuarios en un proveedor puede cambiar con el tiempo. Por ejemplo, la mayoría de los usuarios reaccionan ante los problemas de privacidad. Son, en gran parte, inconsciente de los debates de privacidad hasta que empiezan a leer acerca de temas nacionales en los medios de comunicación o escuchan noticias de la mano de gente conocida. La gran ventaja acerca de tener múltiples proveedores es que si la confianza en un determinado proveedor se rompe, los usuarios siempre tienen alternativas para autenticarse en un sitio.

No acceso al reino

Otra realidad ineludible es que la gente a veces comparte sus contraseñas con amigos o familiares para que puedan ver algún tipo de información, como una foto o un mensaje. Mientras que hay un cierto grado de confianza construida para hacer esto, la mayoría de la gente tiende a olvidar exactamente donde usan esta identidad. Autenticarse en diferentes sitios con múltiples proveedores previene que alguien, que podría tener la contraseña de un usuario, tenga

acceso a todos los sitios web que frecuenta el usuario.

2.5. Limitaciones y Desventajas

Luego de abordar los beneficios de Social Login y antes de empezar a esbozar una implementación de esta metodología, se deben tener en cuenta las limitaciones y desventajas que presenta.

2.5.1. Puede estar a merced de alguien más

Al utilizar Social Login, el usuario puede estar a merced de alguien más. Ese alguien más podría ser la red social, o servicio, que está permitiendo al usuario autenticarse mediante Social Login. Por ejemplo, podría ser Twitter, Google, Yahoo! o Facebook. Es importante destacar que hay un punto débil, la conexión entre el sitio proveedor de Social Login y el sitio o servicio que provee la identidad.

Una de las principales tecnologías clave que hace que Social Login funcione es una interfaz de programación de aplicaciones, es decir una API, la cual permite que dos sistemas muy diferentes y aparentemente incompatibles puedan hablar entre sí. Sin embargo, si el servidor de la API, al que se conecta el proveedor de Social Login, no está disponible, genera un gran inconveniente. Los usuarios y/o clientes no serán capaces de ingresar al sitio.

Mientras que el tiempo de actividad del servidor se ha vuelto menos y menos importante en los últimos años, las cosas suceden, y a menudo suceden cuando uno menos se lo espera o cuando en realidad, un servicio vital es absolutamente necesario.

2.5.2. No es para todo el mundo

Muchos sitios que implementan Social Login requieren que el usuario tenga una cuenta de Facebook o Twitter. Mientras que Facebook y Twitter tienen grandes bases de usuarios, que están creciendo aún más, no todo el mundo utiliza uno o ambos de ellos. Mediante el uso de sólo una o dos opciones de autenticación, podría denegarse el acceso a una gran cantidad de usuarios.

He aquí un ejemplo: A principios de 2011, el sitio de consejos eHow [20] implementó un sistema Social Login que utilizaba Facebook exclusivamente, no ofrecía otras opciones de inicio de sesión. Eso provocó una considerable reacción negativa de los usuarios, por lo que no tuvieron más remedio que ofrecer más opciones para el inicio de sesión.

Además de que no todas las personas poseen cuentas en las redes sociales más conocidas, también sucede que no todo el mundo utiliza Social Login, tal vez porque están conformes recordando otro nombre de usuario y contraseña, o son conscientes de que compartirán sus datos sociales y están preocupados por la privacidad.

Otra desventaja relacionada con lo dicho anteriormente, es que en ciertas industrias o sectores, Social Login simplemente no funciona. Por ejemplo, es muy difícil imaginar a alguien que acepte acceder a su cuenta bancaria en línea a través de sus credenciales de Facebook.

2.5.3. Puede producir costos adicionales

Es una realidad que hay costos involucrados en la implementación Social Login. Algunos de los proveedores de esta metodología de autenticación, no gratuitos, cobran extra cuando se llega a un cierto umbral de usuarios. Por ejemplo, si se contrata el servicio, podría tener que pagar $500 adicionales al mes si se pasa de 1.000 usuarios únicos. Para las organizaciones establecidas, ese gasto adicional probablemente valga la pena. Las organizaciones más pequeñas, que acaban de forjar sus nombres, podrían no ver un retorno de la inversión por un buen tiempo, o quizás nunca. Más usuarios no siempre se traducen en más ventas, por ello es que ese gasto tiene que equilibrarse con una visión realista de cómo crecerán los ingresos de la organización, y decidir a partir de ahí.

Social Login no es una solución mágica, por así decirlo, que de repente hace que un sitio web sea más social y, por ende, más atractivos para los clientes. De hecho, tal vez no sea la solución adecuada. Al comparar las ventajas de Social Login con sus limitaciones y desventajas, se puede obtener una mejor idea de si esta metodología es o no la solución más adecuada para un determinado sitio web.

2.6. Recomendaciones

Un sitio web que da un paso en falso puede ser muy vergonzoso. Cuando se considera incorporar Social Login a un sitio web es importante asegurarse que el sitio en cuestión no resulte ser de esos que tienen largos tiempos de carga o enlaces sin destino. Es importante pensar en una navegación más simplificada, gran diseño y una implementación de Social Login sin fisuras, para mejorar la experiencia del usuario.

Para evitar cualquier paso en falso en la implementación o integración de Social Login, se presenta a continuación una lista de recomendaciones [21]:

- **Integrar los campos de nombre de usuario/contraseña**, e iconos sociales en la misma zona. El propósito de Social Login, es simplificar el proceso de autenticación, por lo tanto, debe hacer que el registro sea fácil de encontrar y de seguir adelante. Borrar mensajes y colocar el inicio de sesión tradicional junto con Social Login, hacen de éste un proceso transparente.

- **Usar el nombre de usuario/contraseña del proveedor de identidad para autenticación**. Esta recomendación es muy útil ya que es sabido que existen sitios web que obligan a los usuarios a crear un nombre de

usuario/contraseña después de iniciar sesión con la cuenta de un proveedor de identidad. Esto es contraproducente e innecesario, ya que el proveedor de identidad ya ha realizado la autenticación.

- **Pre-llenar los campos con los datos devueltos por la API del proveedor de identidad**. Se trata de datos fácilmente disponibles y reducirá la cantidad de tiempo que el usuario tiene que pasar rellenando los formularios de registro. El tiempo ahorrado es directamente proporcional a la felicidad del usuario.

- **Reunir datos a través de múltiples visitas para reducir el abandono**. Hay un punto de inflexión entre pedir demasiada información y no reunir suficiente. Solicitar información adicional en una visita repetitiva es un buen compromiso para todos.

- **Personalizar el sitio web con la foto de perfil y nombre o nombre de usuario.** Esto es parte de la experiencia del consumidor. Si un usuario inicia sesión con una cuenta de sus redes sociales, espera ver su foto en el sitio web. Para ello se deben solicitar los permisos al usuario para adquirir la foto de perfil de su red social.

- **Fomentar múltiples conexiones de diferentes proveedores de identidad**. Para sacar el máximo provecho de la experiencia social, los usuarios pueden enlazar todos sus proveedores entre sí para el intercambio rápido y flexibilidad de autenticación.

- **Construir una estrategia de datos sociales y análisis de rutina**. Con tantos datos disponibles para ser utilizados fácilmente, se puede decidir cuáles son importantes para el sitio web y qué análisis utilizar como indicadores clave. Una rutina dos veces por semana para analizar estos indicadores clave y realizar ajustes al sitio web o ser proactivo será una fuerza impulsora para que los usuarios regresen o para planear estrategias de marketing en caso de tratarse de un sitio comercial.

2.7. Social Login y el Comercio Electrónico

Como hemos mencionado anteriormente, Social Login ha estallado en la escena como una alternativa al tradicional inicio de sesión y proceso de registro en los sitios web. Reduce la fricción asociada con el registro de usuarios, permite a los comerciantes capturar información más precisa del usuario y permite que tanto el vendedor como el usuario puedan aprovechar los beneficios del compromiso adquirido (es decir, información específica, ofertas y promociones).

El comercio social puede ser una nueva palabra de moda, pero se espera que crezca 6 veces en 2015 según la consultora mundial Booz & Co [22]. No es de extrañar, ya que un 71% de los entrevistados en una encuesta realizada por la consultora antes mencionada citó que las críticas de amigos y familiares ejercen la mayor influencia sobre las decisiones de compra de un cliente.

Proporcionar a los usuarios un medio eficaz para acceder a su red elegida, les permite discutir y compartir información acerca del producto o servicio ofrecido por el sitio. Utilizando una buena estrategia de marketing social, acorde a la actualidad, se puede promocionar la marca cada vez que el usuario interactúa en el sitio web por medio de Social Login y de esta manera logran un crecimiento significativo del negocio.

Pero una vez que los clientes, ya sean reales o potenciales, se registran en el sitio web, es cuando comienza el verdadero trabajo. Este trabajo está ligado a los clientes, manteniéndolos comprometidos, y a la transformación de clics en compras. Social Login es más que una herramienta para permitir que las personas se autentiquen cómodamente en un sitio web. También puede ser una forma eficaz de orientar el contenido a los clientes, aumentar la vinculación con el usuario, aumentar su fidelidad, e impulsar las ventas.

2.7.1. Conduciendo el compromiso entre los clientes y Social Login

Una de las metas de una marca o de una organización que utiliza los medios sociales es impulsar las conversiones, es decir, convertir las visitas y clics en ventas. Más importante aún, desea que los clientes sigan regresando. Lograr eso definitivamente no es fácil.

Para poder hacerlo de manera eficaz, es necesario involucrar a los clientes. El compromiso con el cliente está envuelto en una serie de factores, incluyendo el tener productos o servicios convincentes y siendo proactivo con los clientes.

Social Login puede ser eficaz en el impulso de la participación del cliente. A continuación se describen a algunas de las formas en que esta tecnología puede impulsar la participación del cliente.

Conveniencia

Con Social Login los clientes o visitantes pueden autenticarse mediante un proveedor de identidad. Hacer eso quita una capa de complejidad para conseguir que se enfoquen en los productos o servicios de la marca ofrecida. Con la complejidad hecha a un lado, existen más probabilidades de que se registren nuevos clientes y que clientes actuales no dejen de visitar el sitio.

Lealtad

Un componente clave para el compromiso es la lealtad. Muchas organizaciones con presencia en internet han encontrado que los usuarios que se registran en sus sitios son más fieles que el equivalente de los visitantes conducidos por internet. Mientras que la adición de Social Login a un sitio no garantiza la lealtad, puede mejorar las probabilidades de que los clientes se vuelvan, y permanezcan, leales. Y eso está atado con la conveniencia. Al bajar la complejidad para registrarse, clientes actuales y potenciales pueden ser más

propensos a registrarse. Lograr que se registren es un gran paso logrado.

Personalización

Las personas quieren que su experiencia en cada sitio web que utilizan sea única. Esto es especialmente cierto en el comercio electrónico. Dependiendo de la solución que se implemente, Social Login potencialmente puede traer más que la información del perfil de un usuario a un sitio web. Usando esa información, un sitio comercial puede dirigirse mejor a los actuales y potenciales clientes. Puede hacer recomendaciones personalizadas para atraerlos más. Y, según señalan los especialistas, hacer recomendaciones basadas en la información del perfil influye a los consumidores cuando están cerca de la conversión.

Además de eso, en base a los intereses y datos demográficos del cliente, se pueden hacer recomendaciones cuando este se autentica en el sitio. Mientras que a algunos clientes les resulta molesto, puede conducir a otros a ahondar un poco más en determinadas ofertas. Más aún, algunos de esos clics se podrían convertir en ventas.

A su vez, se puede añadir otra dimensión si algunos de los amigos o contactos del cliente, de una red social en particular, también se registran en el sitio, al observar lo que estos amigos o contactos han comprado o mirado en el sitio, se pueden hacer recomendaciones mejor orientadas. No es extraño que las personas compren productos o adquieran servicios basados en las recomendaciones de un conocido.

Sociabilidad

Por más que mucha gente odie admitirlo, uno siempre está tratando de mantenerse al día con sus amigos o conocidos. Con Social Login, parte de la información de perfil que se recolecta incluye información acerca de la gente en la red social de un cliente. Si los miembros de la red están utilizando un producto o servicio, Social Login ofrece la oportunidad de ver si alguien en la red de una persona utiliza ese producto o servicio, y cuáles son sus opiniones al respecto. Generalmente, la gente está más dispuesta a usar y comprometerse con una marca que, alguien de confianza y conocido, usa y se compromete.

Social Login no es una solución del tipo “mágica”, su aplicación no causará una revolución inmediata en la forma en que los clientes y potenciales clientes se involucran con una determinada marca. No va a solucionar todos los problemas de compromiso con el cliente, pero, sin duda puede jugar un papel importante en el impulso de la participación del cliente.

La aplicación de esta metodología le da la oportunidad al sitio web de hacer las cosas más conveniente y más personales para los clientes actuales y potenciales. Social Login puede generar lealtad y hacer que la experiencia de una persona sea más social con respecto a la marca del sitio web. Si un sitio web de comercio electrónico puede hacer todo eso, se puede decir que ha

dado un gran paso en la conducción y el mantenimiento de los clientes.

2.8. Principales proveedores de Social Login

Social Login, junto con sus plugins asociados y análisis, además de brindar una metodología de autenticación, proporciona acceso a un rico tesoro de datos que pueden ser utilizados para alimentar estrategias de marketing, publicidad creativa y publicación de anuncios, orientación del contenido y recomendaciones de productos.

Social Login contribuye a un conjunto, cada vez mayor, de soluciones conocidas como CRM [23] social. Mientras que en CRM tradicional, los profesionales de servicio al cliente y de ventas son responsables de la actualización de la base de datos y de llenarla con la información que conduce a más ventas o niveles más altos de servicio al cliente, CRM social se sirve de medios de comunicación social, así como la información recogida de los perfiles públicos de las redes sociales, para capturar datos sobre los clientes actuales y potenciales. Se agregan análisis y estadísticas a estos datos para predecir el comportamiento, y para que luego los sitios web que los utilizan pueden utilizarlos a su favor.

Contar publicaciones de Twitter o publicaciones en el muro de Facebook sin duda ayuda a las organizaciones a entender cómo es recibido su servicio en el mercado, pero tener acceso a los perfiles sociales completos de las personas es mucho, mucho más rico. Como tal, Janrain y Gigya son proveedores de Social Login que están a la vanguardia de la revolución de CRM social. Como tienen productos de análisis avanzados, están bien posicionados para integrar sus soluciones y añadir una capa de datos que pueden impulsar la comercialización en general, el contenido y la estrategia de producto.

Es evidente que los sitios web comerciales tienen una oportunidad importante para aumentar las tasas de conversión y el compromiso mediante la sustitución o complemento del registro tradicional con Social Login.

Según el último informe de Gigya [24], correspondiente al primer trimestre de 2014, la mayoría de los usuarios prefiere autenticarse a través de sus credenciales de Facebook. En el primer trimestre del año Facebook se ubicó como la red social de referencia para ingresar a otros sitios. Con un 53% de identificaciones, Facebook autentifica la identidad de los clientes ante las marcas, en segundo lugar se colocó Google+ con un 28% y con un 13% se ubicó Yahoo! en el tercer lugar. Twitter llegó al 4% y Linkedin obtuvo 1%. Lo dicho anteriormente se muestra a continuación en el gráfico de la Figura 2.1.

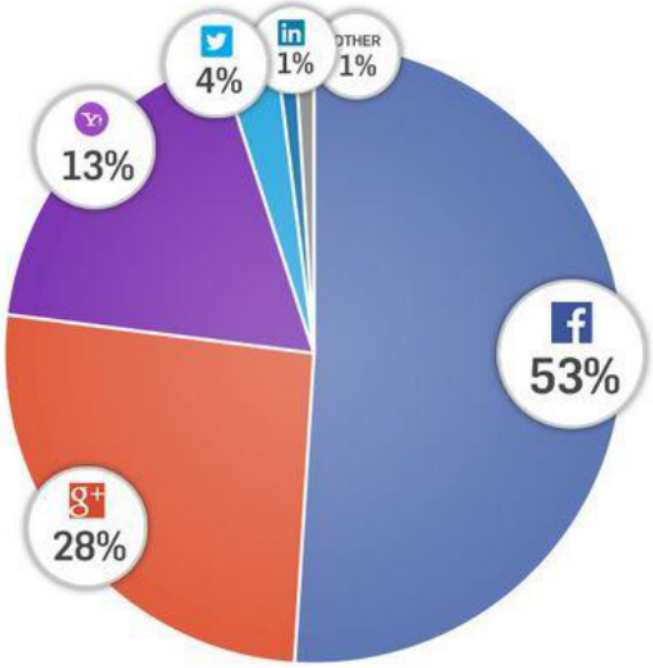

Figura 2.1: Gráfico que representa la tendencia de los usuarios a la hora de elegir un proveedor de identidad para autenticarse en un sitio web mediante Social Login.

Respecto a dispositivos móviles, tal como se representa en la Figura 2.2, Facebook cuenta con el 62% de los ingresos, mientras Google+ se queda con el 26%; en tercer lugar se ubica Twitter con el 6% y al final queda Yahoo! con el 4%.

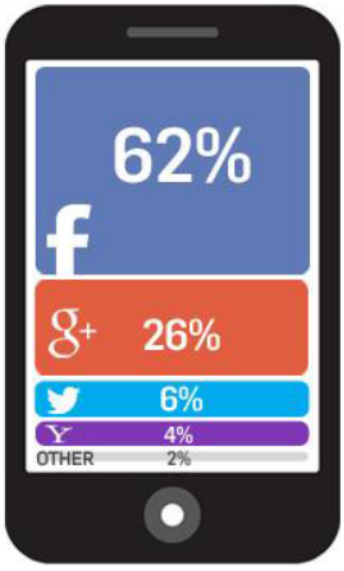

Figura 2.2: Gráfico que representa la tendencia de los usuarios a la hora de elegir un proveedor de identidad para autenticarse en un sitio web mediante Social Login en dispositivos móviles.

Las ventas en línea se vieron sumamente beneficiadas por Facebook, ya que fue el principal referente para los consumidores con el 77% de identificaciones. Google+ tuvo un 14% y Twitter contó con el 3%, números que se representan en la Figura 2.3.

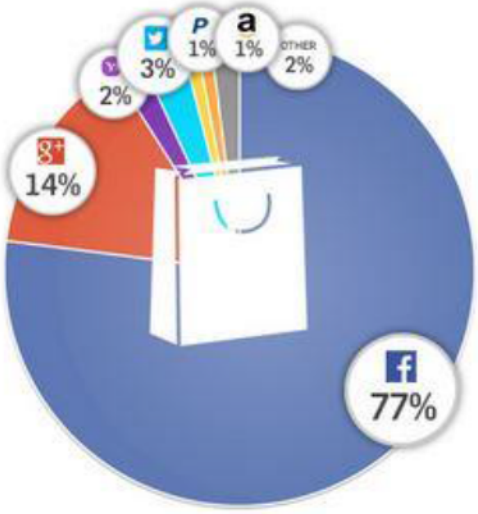

Figura 2.3: Gráfico que representa la tendencia de los usuarios a la hora de elegir un proveedor de identidad para autenticarse en un sitio web de comercio electrónico mediante Social Login.

2.8.1. Decodificación del valor de Social Login

Como ya se ha destacado anteriormente, más y más organizaciones están advirtiendo el valor que tiene proveer Social Login en un sitio web. Sin embargo, cuando se hace un relevamiento de los proveedores, no hay ninguna norma de precios o características, lo que hace difícil realizar un presupuesto. A continuación se listan las principales pautas sobre qué buscar y cómo calcular su costo. Estas pautas permiten establecer la relación costo/beneficio de incorporar Social Login en un sitio web.

- **Registros**: La mayoría de las tasas de los proveedores de Social Login se basan en registros de usuarios únicos por año. Esto significa pagar por cada usuario sólo una vez por año. Al final del año, se reinicia el reloj y se paga nuevamente por los usuarios registrados.

- **Anual vs Mensual:** Pagar anualmente es una buena manera de ahorrar presupuesto anual, pero en este caso el arancel se tiene que pagar por adelantado. El pago mensual es más fácil en el flujo de la caja, pero se podría estar perdiendo algún tipo de descuento.

- **Contrato vs Sin contrato**: Algunos proveedores requieren un compromiso anual. El desafío de comprometerse a un año es que hay que estimar, con un año de antelación, el número de usuarios que tendrá el sitio web y la cantidad de proveedores de identidad que se desea utilizar. Si se paga de más, se pierde el importe abonado o tal vez se pueda negociar un traspaso de usuarios no utilizados, y si se paga de menos, los usuarios adicionales se pueden añadir al próximo pago, por lo tanto, los precios serán más altos. Otros proveedores ofrecen planes de pagos mensuales basados en la cantidad de registros mensuales, en vez del consumo anual. En este caso, los pagos mensuales fluctúan pero no hay ningún compromiso o estimación.

- **Categoría vs Acumulativo**: Un modelo de pago por categoría significa que se estima el consumo anual, el cual se corresponde a una categoría y se paga esa cantidad durante todo el año. Cuanto mayor sea la categoría, menos se paga. En un modelo acumulativo, generalmente se paga la cantidad especificada en cada nivel en ese momento y los cambios en los costos se producen cuando se ingresa a una categoría de un nivel superior.

- **Varios dominios**: Múltiples dominios pueden o no estar incluidos en el precio base. Además, el número de usuarios se utiliza para determinar el nivel de precios, puede ser por dominio o puede ser para todos los dominios juntos.

- **Características**: Un modelo de precios basado en características, incluye o excluye características en los diferentes niveles. Por ejemplo, almacenamiento de datos, capacidad de exportación, el acceso a las estadísticas. Por lo general, es probable que todas las características estén disponibles pagando un precio complementario.

- **Estimación del presupuesto**: Se debe hacer un análisis profundo de los números para entender cuánto va a costar y evaluar lo que se puede obtener con cada proveedor.

  Se puede calcular el uso utilizando el número de miembros activos o usuarios que inician sesión anualmente, más cualquier aumento que se prevea teniendo en cuenta las próximas iniciativas y el futuro uso Social Login. A su vez, se pueden tomar como referencia el número de visitantes anuales de algún servicio de estadísticas de sitios web, en caso de que se utilice uno.

  Es importante confeccionar la lista de las características que se desean incluir del proveedor de Social Login. Se debe analizar lo que es importante para el fin del sitio web, para no contratar herramientas que luego no serán utilizadas o aprovechadas.

  Por último, se recomienda, como una buena práctica, obtener una revisión, de la documentación de cada proveedor, por parte de personal técnicamente capacitado. El precio no es el único costo, el tiempo y los recursos invertidos en la implementación y el mantenimiento, también deben tenerse en cuenta.

Capítulo 3

Protocolos

3.1 Introducción

Si bien existen varios protocolos para implementar la identificación o autorización de un usuario en un sitio web, los dos más utilizados actualmente son OAuth y OpenID, los cuales fueron usados para realizar el desarrollo de ESLIP, además de ser definidos y analizados en detalle en esta tesina. Asimismo se analizaron otros protocolos existentes, los cuales son descritos brevemente a continuación en esta sección para tener un conocimiento general de las alternativas que existen en este campo.

Al comenzar la lectura de material sobre los distintos protocolos lo primero que se notó fue que se confunden algunos conceptos como por ejemplo el de autenticación y autorización. Es por eso que a continuación se definen los más relevantes para la comprensión del contexto que engloba a estos protocolos.

3.2. Identidad

La identidad es algo que es usado para distinguir a algo o alguien, de otros. En el caso de la identidad de las personas, quizás esta puede ser el nombre de una persona, pero probablemente el nombre no sea suficiente para identificar a una persona, ya que otra persona puede tener el mismo nombre.

La identidad de una persona podría ser el número de documento, el color de ojos, el lenguaje que habla, el lugar donde vive, entre otras, aunque en la mayoría de los casos, la identidad es una combinación de los factores anteriormente nombrados.

3.3. Nombre de usuario y contraseña

Nombre de usuario y contraseña son comúnmente dos cadenas compuestas por números, letras y caracteres especiales, los cuales deben ser recordados.

A veces el nombre de usuario es conocido como User ID, Login ID, nombre (name en inglés) o simplemente ID. El mismo suele tener entre seis y ocho caracteres de longitud, pero algunas organizaciones pueden tener sus propios estándares para definir la longitud del nombre de usuario. A su vez, un nombre de usuario no suele ser siempre el nombre real de una persona, puede ser cualquier cadena de letras y/o números.

Una contraseña debe ser un conjunto complejo de caracteres en mayúscula y minúscula, números, y caracteres especiales. Diferentes políticas de las organizaciones pueden ser aplicadas para requerir contraseñas complejas, que

pueden variar según el tipo o rol del usuario.

Mientras que el nombre de usuario se mantiene, es recomendable cambiar la contraseña cada cierto tiempo. Algunos sistemas obligan a los usuarios a cambiar sus contraseñas luego de un determinado periodo de tiempo, o luego de una determinada cantidad de uso.

Comúnmente, se da que los usuarios tienen distintos nombres de usuario y contraseñas para todos los sistemas y sitios webs a los que acceden.

3.4. Autenticación

La autenticación es el proceso de verificar la identidad de un usuario, a sabiendas de que el usuario es quien dice ser.

En el mundo real, cuando alguien quiere corroborar una identidad, le pide al sujeto una identificación, por ejemplo su DNI, verificando que la imagen de la identificación corresponda a la semejanza del sujeto en cuestión.

En los sistemas de escritorio y web, la autenticación trata de saber que el usuario frente a la pantalla es el dueño de la cuenta. La autenticación se basa principalmente en preguntarle al usuario un nombre de usuario y una contraseña. El nombre de usuario representa la identidad alegada del usuario, y el sistema asume que si el usuario proporciona la contraseña correcta, este usuario es efectivamente quien dice ser.

3.5. Autenticación Federada

Aunque muchas aplicaciones tienen su propio sistema de cuentas (incluyendo nombres de usuario y contraseñas), algunas aplicaciones se basan en otros servicios para verificar la identidad de los usuarios. Esto se conoce como autenticación federada.

En un entorno empresarial, las aplicaciones pueden confiar en un servidor de Active Directory [25], un servidor LDAP [26] o un proveedor SAML [27] para autenticar usuarios.

En la web, las aplicaciones a menudo confían en proveedores OpenID (como Google o Yahoo!) para manejar la autenticación de los usuarios. Hay muchos beneficios con la federación, tanto para los desarrolladores de aplicaciones, como para los usuarios. OpenID es el protocolo abierto web más común para manejar autenticación federada.

3.6. Autorización

La autorización es el proceso de verificar que el usuario tiene el derecho de realizar alguna acción, como la lectura de un documento o acceder a una cuenta de correo electrónico. Esto normalmente requiere primero la

identificación válida para el usuario (autenticación) con el fin de comprobar si el usuario actual está autorizado.

Por ejemplo, cuando un oficial de policía detiene a una persona conduciendo un vehículo, primero identifica a dicha persona utilizando el carnet de conducir (para verificar su identidad), luego comprueba la licencia (fecha de vencimiento, restricciones, entre otros) y las cédulas verde y azul, para asegurarse de que la persona está autorizados para conducir el auto en el que se encuentra.

El mismo proceso ocurre en internet, una aplicación primero verifica la identidad del usuario mediante el inicio de sesión con nombre de usuario y contraseña, y luego se asegura de que este usuario sólo acceda a los datos y servicios para los que está habilitado, por lo general mediante la comprobación de una lista de control de acceso para cada operación.

3.7. Autorización Delegada

La autorización delegada permite otorgar acceso a una persona o aplicación para realizar acciones en nombre de otra.

Para entender mejor la idea se puede mencionar un ejemplo cotidiano como el de la cédula azul, en el cual el titular de un vehículo solicita al Registro de la Propiedad del Automotor la expedición de una o más Cédula/s Azul/es (Cédula/s de identificación para autorizado a conducir), a fin de instrumentar documentalmente la autorización a un tercero para usar el automotor.

OAuth funciona de manera similar, un usuario le permite a una aplicación realizar acciones en nombre del usuario y la aplicación sólo puede realizar las acciones autorizadas.

En respuesta a estos conceptos las tecnologías de autenticación como OpenID ofrecen soluciones al problema de autenticarse en múltiples sistemas, mientras que las de autorización como OAuth resuelven el problema de tener datos y recursos personales distribuidos en distintos sistemas. Como consecuencia de esto han surgido y evolucionado los protocolos de autenticación y autorización para los sitios web.

3.8. Protocolos y servicios

3.8.1. SAML 2.0

SAML 2.0, cuyas siglas significan Security Assertion Markup Language, o Lenguaje de Marcado de Aserciones de Seguridad, es la segunda versión del estándar propuesto por OASIS (Organization for the Advancement of Structured Information Standards) [28] para el intercambio de datos de autenticación y autorización entre dominios de seguridad. Su primera intención fue ofrecer una especificación dirigida al dominio de navegadores web para ofrecer un servicio de inicio de sesión único, aunque también fue diseñada de

forma modular y extensible para facilitar su uso en otros contextos.

Se trata de un protocolo basado en XML [29] que se usa en la comunicación entre un proveedor de identidad y un servicio web. La aserción, que se puede denominar como un token de seguridad XML, es lo que se emite desde el proveedor de identidad y que es consumido por el servicio web, el cual debe de confiar en el contenido de la aserción, más concretamente en su asunto (subject), por motivos relacionados con la seguridad.

3.8.2. Mozilla Persona

Mozilla Persona [30] es un sistema de autenticación descentralizada para la web, basado en el protocolo abierto BrowserID [31] creado por la empresa Mozilla [32].

Este sistema promueve una forma fácil de conectarse a los sitios web para que, en lugar de tener que inventar para cada sitio un nombre de usuario y una contraseña, Mozilla Persona permite usar una dirección de correo y una única contraseña para iniciar sesión e ingresar a cualquier sitio web que lo tenga implementado.

Persona se puso en marcha por Julio de 2011 y durante bastante tiempo se fueron viendo distintas versiones de prueba del sistema de autenticación. Pero a principios de Marzo de 2014 Mozilla anunció que el desarrollo de Persona fue transferido a la comunidad. Eso significa que Mozilla se encargará de garantizar la continuidad del servicio y de corregir fallos de seguridad y problemas críticos, pero no desarrollará nuevas características.

3.8.3. BrowserID

BrowserID surge a raíz de una propuesta de Mozilla Labs [33] para minimizar el uso de contraseñas desde cualquier navegador.

La idea que propone la Mozilla con BrowserID es utilizar el correo electrónico del usuario como clave de acceso para todos aquellos sitios y servicios que requieren autenticación, evitando la molestia de memorizar el conjunto usuario y contraseña.

BrowserID es un sistema descentralizado de identidad que permite a los usuarios demostrar la propiedad de las direcciones de correo electrónico de forma segura, sin necesidad de contraseñas para cada sitio.

La primera vez que se accede a BrowserID, se solicita una contraseña y una dirección de correo electrónico. Con esta información y a través de una tecnología denominada Verified Email Protocol [34], se comprueba la veracidad y pertenencia de la dirección suministrada.

Una vez realizado el paso anterior, se dispone de una identificación única para cualquier registro basada en criptografía asimétrica, con una clave pública

accesible para los proveedores de contenidos y una privada, que permanece en el navegador. La verificación de claves se realiza en los servidores de BrowserID.

BrowserID es similar a OpenID, aunque existen diferencias esenciales. En una nota publicada en el blog de Mozilla titulada "How BrowserID differs from OpenID" [35], se ponen de manifiesto las diferencias: correo electrónico como identificador, privacidad e identidad protegidas y que BrowserID está integrado en el navegador, entre otras.

3.8.4. WebID

Las especificaciones WebID [36] definen un conjunto de normas propuestas por la identidad, la identificación y la autenticación en redes basadas en HTTP [37].

WebID se propuso como un reemplazo para el inicio de sesión tradicional en los sitios web. Es un estándar abierto para la identidad y el acceso. Con WebID se busca no tener que recordar nombres de usuario o contraseñas para todos los sitios web. También es posible publicar la identidad donde cada uno quiera y elegir qué datos de la información personal se desean compartir con los sitios web.

WebID está en proceso de estandarización por parte de la W3C [38]. Para ello, ha organizado un grupo de trabajo al respecto para avanzar en la especificación del futuro estándar. Al encontrarse en este proceso, aún es recomendable su uso.

Capítulo 4

OpenID

4.1. Introducción

OpenID es un estándar de identificación digital descentralizado que permite a una persona usar una URL como identidad y usar esta misma identidad en múltiples sitios web que soporten OpenID. Los sitios web pueden usar esta URL de identidad para autenticación, autorización y otros propósitos. Es un concepto relativamente nuevo, el cual pone el control de la identidad en las manos de su dueño, es decir, en manos del usuario final. El dueño de la identidad puede decidir y tener control sobre qué información debe presentarse a una aplicación o sitio web, para realizar la autenticación en el mismo.

En los sitios web que soportan OpenID, los usuarios no tienen que crearse una nueva cuenta de usuario para obtener acceso. En su lugar, sólo necesitan disponer de un identificador creado en un servidor que verifique OpenID, llamado proveedor de identidad o IdP.

Anteriormente se definió que OpenID posee una estructura descentralizada, esto se debe a que permite que cualquiera pueda convertirse en un usuario de OpenID o un proveedor de OpenID de forma gratuita sin necesidad de registrarse o ser aprobado por una organización.

OpenID no especifica el mecanismo de autenticación, por lo tanto, la seguridad de una conexión depende de la confianza que tenga el cliente OpenID en el proveedor de identidad. Si no existe confianza en el proveedor, la autenticación no será adecuada para servicios bancarios o transacciones de comercio electrónico.

Entre otras cosas, OpenID permite a sus usuarios:

- Autenticarse a aplicaciones web o sitios web, sin introducir ningún nombre de usuario y contraseña.
- Decidir qué información será enviada al sitio web durante el proceso de autenticación.
- Elegir el conjunto de información (comúnmente conocido como perfil) que será enviado a cada sitio web, dependiendo de la necesidad y el nivel de riesgo en cada caso.
- Implementar una alternativa a los registros de usuarios convencionales.

En síntesis, OpenID es un sistema que permite a los usuarios utilizar una URL como identificador para autenticarse a sitios web que soporten OpenID, con el

beneficio de que, los usuarios del sistema OpenID, no tienen la necesidad de recordar nombres de usuarios y contraseñas para cada sitio web.

4.2. Definiciones

Antes de describir en detalle el protocolo es importante conocer las terminologías que OpenID utiliza. Cabe señalar que en algunos casos, ligeramente se usan diferentes términos para expresar lo mismo en la documentación de OpenID. Asimismo, las especificaciones de OpenID 1.1 y 2.0 tienen algunos pequeños cambios en la terminología.

4.2.1. Usuario Final

El usuario final es el usuario real o la persona real que utiliza OpenID para ingresar a diferentes sitios web usando sus credenciales, las cuales están almacenadas en el proveedor de identidad.

4.2.2. Consumidor o Usuario de Confianza

El consumidor es el sitio web dónde se quiere autenticar por medio de OpenID. Es llamado consumidor porque consume las credenciales de OpenID provistas por el proveedor de identidad.

Todos los sitios que permiten autenticación por OpenID son considerados consumidores. En la especificación de OpenID 2.0 el consumidor es a su vez llamado usuario de confianza o Relying Party en inglés.

4.2.3. Identificador

El identificador es la URL o la cadena que identifica la entidad del usuario final. Existen diferentes tipos de identificadores de OpenID, los cuales se detallan a continuación.

Identificador Local

Es un identificador que pertenece a un proveedor OpenID particular. Por lo tanto, no se encuentra bajo el control del usuario del final.

Por ejemplo, el siguiente identificador http://nicoburghi.myopenid.com/ pertenece a el proveedor MyOpenID [39], el usuario no es dueño de esa URL, si no que la misma está bajo el control de dicho proveedor.

Identificador Reclamado

Es un identificador que el usuario reclama que le pertenece.

Si un usuario posee su propia URL, por ejemplo http://nicoburghi.com.ar/, puede utilizarla como su identificador OpenID, siendo este un identificador

reclamado. Siempre y cuando tenga una cuenta en un proveedor de OpenID.

Por ejemplo, si el usuario posee un identificador OpenID perteneciente a un proveedor como ser http://nicoburghi.myopenid.com/ y posee su propia URL http://nicoburghi.com.ar/, puede utilizar esta URL como su identificador reclamado, delegando la funcionalidad del proveedor de OpenID a MyOpenID u otro proveedor que se desee. Es decir que el identificador reclamado funciona como un puente hacia el identificador local.

Para implementar lo descrito en el ejemplo anterior, el documento HTML [40] que se obtiene a partir de la URL del identificador reclamado, debe tener los siguientes elementos en la cabecera del documento:

```
<link href='http://www.myopenid.com/server'
rel='openid2.provider openid.server'/>

<link href='"http://nicoburghi.myopenid.com/'
rel='openid2.local_id openid.delegate'/>
```

Con este enfoque nunca se limita a un solo proveedor de OpenID. De manera tal, que si se pierde confianza en un proveedor de OpenID, es posible mudarse a otro proveedor manteniendo el identificador de OpenID original, o sea el identificador reclamado seguirá siendo el mismo.

Identificador XRI

Un identificador XRI (eXtensible Resource Identifier) [41] es un identificador global único, tal como lo son los nombres de dominio. También se lo define como un identificador abstracto, que sirve para obtener identificadores concretos. Estos últimos son identificadores que se utilizan, por ejemplo, para representar recursos reales en una red. URL's, emails, números telefónicos, son identificadores concretos.

En resumen, XRI es un identificador abstracto que puede ser mapeado a identificadores concretos.

La sintaxis de XRI define dos tipos de identificadores:

- i-names:
  - Son más amigables al humano, por ejemplo: =nicoburghi
  - Están destinados a ser identificadores reasignables como los nombres de dominio.

- i-numbers:
  - Son más amigables para las computadoras, por ejemplo: =!BFC9.75B7.9B2.11C4
  - Están destinados a ser persistentes.
  - Si un usuario posee un identificador OpenID de tipo i-number, nunca lo perderá.

4.2.4. Proveedor de Identidad

El proveedor de identidad es el host donde se almacenan las credenciales de un usuario. El identificador de OpenID apunta al proveedor de identidad. Durante el proceso de autenticación, el consumidor validará la identidad del usuario intercambiando mensajes con el proveedor de identidad. También se lo suele conocer como el Servidor OpenID, Proveedor OpenID o simplemente OP (derivado del inglés OpenID Provider).

4.2.5. Agente de Usuario

El agente de usuario es simplemente el navegador. Un usuario interactúa directamente con él.

4.3. Comunicación entre componentes de OpenID

Existen tres componentes principales en un sistema OpenID: el consumidor, el proveedor de identidad y el agente de usuario, los cuales interactúan entre sí durante el proceso de autenticación.

El consumidor es el sitio web dónde el usuario trata de autenticarse utilizando OpenID. Este interactúa con el proveedor de identidad y el agente de usuario. Durante el proceso de autenticación el consumidor enviará mensajes al proveedor de identidad tanto directamente como a través del navegador con la ayuda de la redirección de mensajes HTTP.

Por otro lado, el proveedor de identidad es el servidor de OpenID, quien almacena las credenciales del usuario final. El proveedor de identidad es el encargado de validar la propiedad de una URL de identidad al consumidor usando dos mecanismos básicos que serán explicados posteriormente.

Por último, un usuario final interactúa con el consumidor y el proveedor de identidad usando el navegador web.

Durante el proceso de autenticación, el navegador actúa como un intermediario entre el proveedor de identidad y el sitio web consumidor para algunos mensajes. Comúnmente, un consumidor interactúa tanto con el navegador como con el proveedor de identidad, pero en algunos casos, el consumidor puede utilizar las claves almacenadas para autenticar un usuario sin ninguna comunicación directa con el proveedor de identidad.

La Figura 4.1 muestra la ruta de comunicación entre las tres entidades. Aquí se asume que la URL de identificación reside en el servidor del proveedor de identidad.

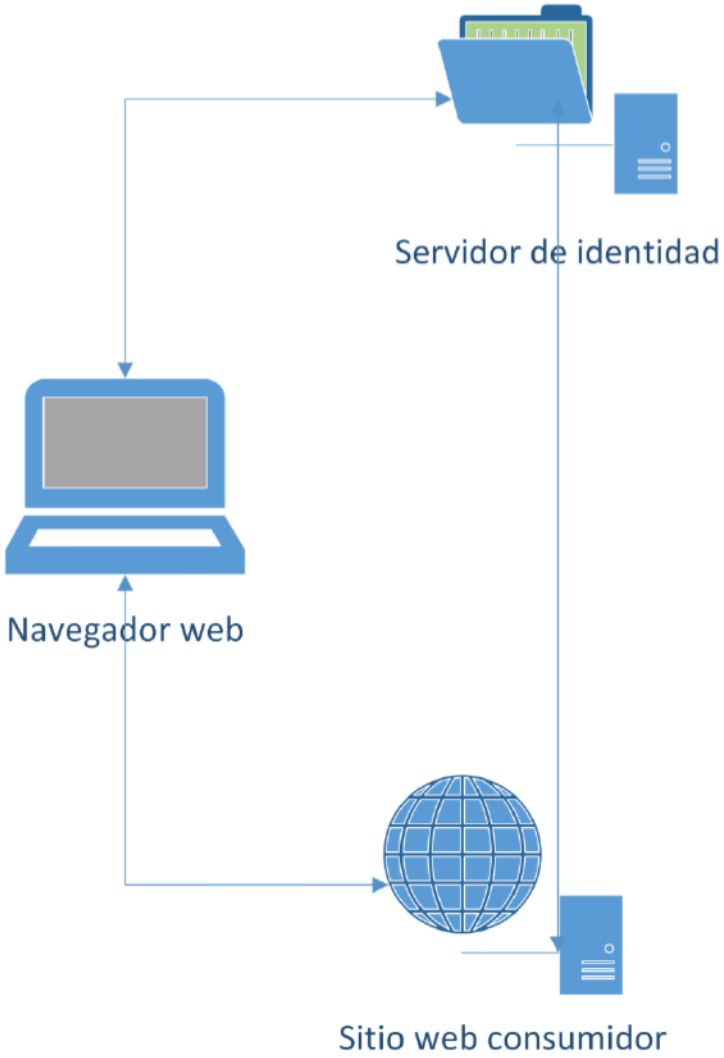

Figura 4.1: Comunicación entre los diferentes componentes de OpenID con la URL de identificación y el proveedor de identidad en la misma máquina.

Notar que un usuario puede tener la URL de identificación apuntando a la máquina de un proveedor de identidad o a un lugar diferente. De hecho, se puede alojar un identificador en cualquier máquina que se desee. Es importante destacar que desde que la URL es la identidad de un usuario, éste tiene el control de esa URL. El propietario de la URL de identificación puede usar diferentes servidores manteniendo su identidad intacta.

La Figura 4.2 muestra la ruta de comunicación cuando la URL de identificación es alojada en un lugar diferente al del proveedor de identidad.

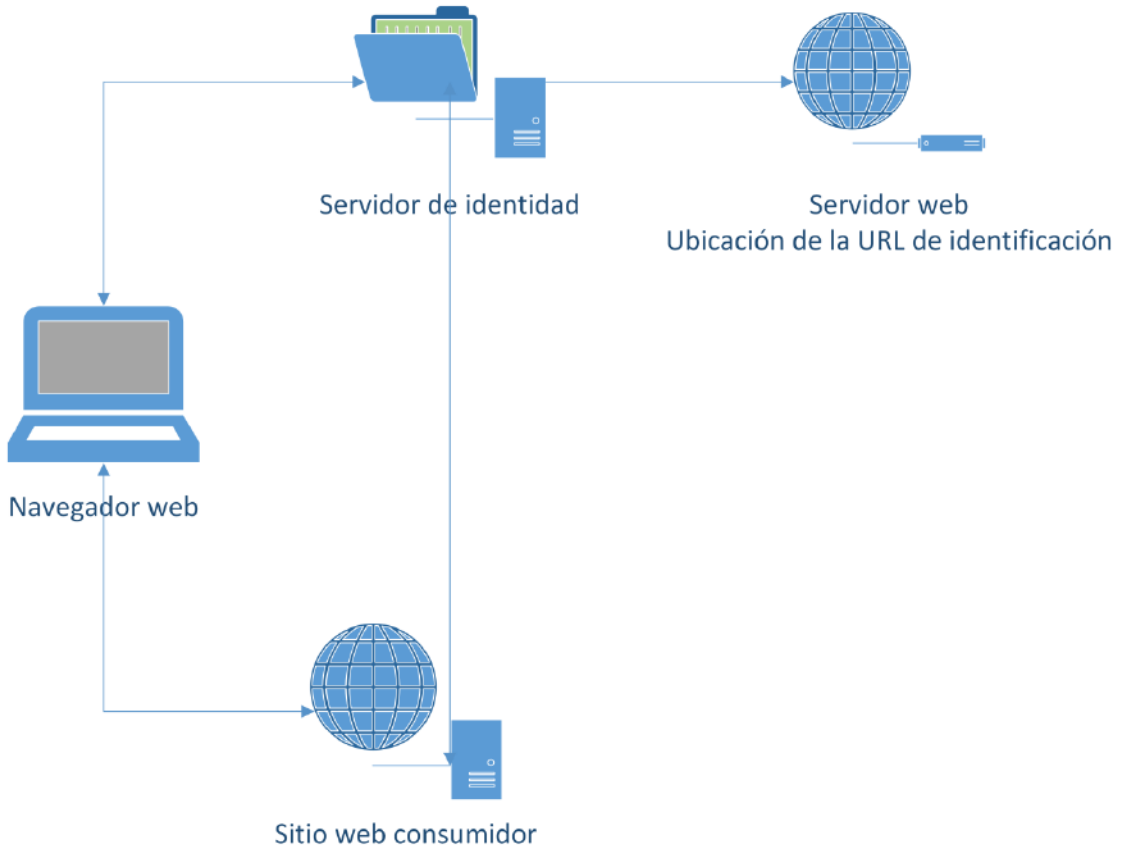

Figura 4.2: Comunicación a través de diferentes componentes de OpenID con la URL de identificación y el proveedor de identidad en diferentes máquinas.

4.3.1. Comunicación Directa e Indirecta

Existen dos métodos básicos de comunicación entre diferentes entidades en un sistema OpenID: comunicación directa y comunicación indirecta.

En el mecanismo de comunicación directa, dos entidades hablan directamente una con la otra, usando el protocolo HTTP. El método HTTP POST [42] es utilizado en este tipo de comunicación.

Por otro lado, con comunicación indirecta, dos entidades hablan una con la otra por medio de una tercera entidad. Esta última es generalmente el navegador web. La comunicación indirecta puede ocurrir a través de redirección HTTP o mediante redirección de un formulario HTML.

4.4. Iniciación y descubrimiento

En esta sección se describen las primeras acciones que se realizan para iniciar el proceso de autenticación de un usuario en un sitio web mediante OpenID. Estas acciones abarcan desde que se presenta el formulario de autenticación al usuario, pasando por la normalización del identificador ingresado, hasta que se descubre el proveedor de OpenID.

4.4.1. Iniciación

Para iniciar la autenticación con OpenID, el sitio web consumidor debe presentar al usuario final un formulario que contenga un campo para ingresar el identificador de dicho usuario.

El atributo "name" del campo del formulario debe tener el valor "openid_identifier", de manera tal que los navegadores puedan determinar automáticamente qué se trata de un formulario OpenID. En caso de tener otro valor, algunas extensiones de los navegadores u otros sistemas que soporten autenticación mediante OpenID puede que no lo detecten.

4.4.2. Normalización

La cadena que el usuario final ingresa en el campo del formulario de OpenID debe normalizarse en un identificador de la siguiente manera:

Si el usuario ingresa una cadena que comienza con el prefijo "xri://", este prefijo debe quitarse, de manera tal que el XRI se utilice en su forma canónica.

Si el primer caracter de la cadena resultante es un símbolo del contexto global de XRI ("=", "@", "+", "$", "!"), la cadena de entrada debe ser tratada como un XRI.

En caso contrario, la cadena de entrada debe ser tratada como una URL HTTP. Si la cadena no comienza con "http://" o "https://", debe agregarse como prefijo la cadena "http://". Si la URL contiene una sección fragmentada, debe ser removida junto con el caracter "#" que delimita el fragmento.

Los identificadores de tipo URL debe entonces ser normalizados mediante la aplicación de las reglas definidas en la sección 6 de la especificación "URI Generic Syntax" (RFC3986) [43].

4.4.3. Descubrimiento

El identificador OpenID, como se mencionó anteriormente en este informe, puede ser una URL o un XRI el cual facilita la autenticación descentralizada. El usuario final debe ingresar el identificador OpenID en el correspondiente formulario de autenticación del sitio web consumidor. Este último, a partir del identificador debe obtener o descubrir cuál es el proveedor OpenID correspondiente. A este proceso se lo denomina "descubrimiento" o "discovery" en inglés.

OpenID tiene tres caminos a través de los cuales puede realizarse el proceso de descubrimiento.

En primer lugar, si el identificador es un XRI, el mismo entregará un documento XRDS [44] que contiene la información necesaria.

Por otro lado, si se trata de una URL, se utilizará el protocolo Yadis [45] en el primer intento de descubrimiento. Si se tiene éxito, el resultado será de nuevo un documento XRDS. Pero, por el contrario, si el protocolo Yadis falla y no se recupera ningún documento XRDS válido o no se encuentra ningún elemento "service" en el documento XRDS, se recupera la URL y se procede a intentar un descubrimiento basado en HTML.

4.4.3.1 Información descubierta

Al completar con éxito el descubrimiento, el consumidor obtendrá uno o más conjuntos de la siguiente información:

- URL del proveedor de OpenID
- Versión del protocolo

Si el usuario final no ingresó un identificador de un proveedor OpenID, también se presentará la siguiente información:

- Identificador reclamado
- Identificador local del proveedor de identidad

4.4.3.2 Descubrimiento basado en XRDS

Si se utiliza un XRI o un descubrimiento mediante Yadis, el resultado será un documento XRDS (eXtensible Resource Descriptor Sequence). Este es un documento XML con entradas para los servicios que están relacionados con el identificador.

Ejemplo de documento XRDS:

```
<?xml version="1.0" encoding="UTF-8"?>
<xrds:XRDS xmlns:xrds="xri://$xrds" xmlns="xri://$xrd*($v*2.0)"
xmlns:openid="http://openid.net/xmlns/1.0">


 <XRD ref="xri://=example">

   <!-- service section -->

   <!-- XRI resolution service -->
   <Service>
   </Service>

   <!-- OpenID 2.0 login service -->
   <Service priority="10">
   </Service>

   <!-- OpenID 1.1 login service -->
   <Service priority="20">
   </Service>

 </XRD>
```

Un documento XRDS puede definir múltiples servicios:

```
<!-- XRI resolution service -->
<Service>
     <ProviderID>xri://=!F83.62B1.44F.2813</ProviderID>
     <Type>xri://$res*auth*($v*2.0)</Type>
     <MediaType>application/xrds+xml</MediaType>
     <URI priority="10">http://resolve.example.com</URI>
     <URI priority="15">http://resolve2.example.com</URI>
     <URI>https://resolve.example.com</URI>
</Service>

<!-- OpenID 2.0 login service -->
<Service priority="10">
<Type>http://specs.openid.net/auth/2.0/signon</Type>
<URI>http://www.myopenid.com/server</URI>
<LocalID>http://example.myopenid.com/</LocalID>
</Service>

<!-- OpenID 1.0 login service -->
<Service priority="20">
<Type>http://openid.net/server/1.0</Type>
<URI>http://www.livejournal.com/openid/server.bml</URI>
<openid:Delegate>
          http://www.livejournal.com/users/example/
   </openid:Delegate>
</Service>
```

4.4.3.3 Descubrimiento basado en HTML

En este tipo de descubrimiento, un documento HTML debe estar disponible en la dirección que denota la URL del identificador reclamado, de manera que el consumidor pueda recuperar el documento mediante una petición HTTP.

El descubrimiento basado en HTML es utilizado únicamente para el descubrimiento de identificadores reclamados, ya que los identificadores de OpenID deben ser XRI's o URL's que soporten descubrimiento basado en XRDS.

Dentro del elemento HEAD, del documento HTML que se obtiene, debe haber un elemento <link> con un atributo "rel" con valor "openid2.provider" y un atributo "href" con la URL del proveedor de OpenID.

Por ejemplo, si se ingresa en un navegador un identificador OpenID de tipo URL como este: http://nicoburghi.myopenid.com/, y se inspecciona su código fuente se deberían encontrar los siguientes elementos en la cabera del documento HTML obtenido:

```
<link rel="openid.server" href="http://www.myopenid.com/server" />
<link rel="openid2.provider" href="http://www.myopenid.com/server" />
```

Estas etiquetas indican quien es el proveedor de identidad del identificador. Ambas referencian al mismo proveedor ya que la primera, cuyo atributo "rel" tiene el valor "openid.server" se utiliza para la versión 1.1 de OpenID, mientras que la otra, cuyo atributo "rel" tiene el valor "openid2.provider" se utiliza para la

versión 2.0 del protocolo.

Como se puede observar, los elementos <link> no son idénticos en todas las versiones de OpenID. Mientras que transmiten los mismos datos, los nombres cambian, lo que permite determinar la versión del protocolo que se está utilizando.

4.5. Modos de operación de OpenID

OpenID tiene dos modos principales de funcionamiento: el modo “sin estado” (conocido como “dumb mode” en inglés) y el modo con estado o modo inteligente (“smart mode”). Estos modos se basan en cuan inteligente es el sitio web consumidor. En el modo sin estado, se puede decir que el consumidor no es tan inteligente y tiene que realizar pasos adicionales cada vez que un usuario se autentica. En cambio, en el modo con estado, el consumidor mantiene información de estado y almacena, en una memoria temporal, claves compartidas para su futuro uso.

4.5.1. Modo sin estado

En este modo, el consumidor no mantiene el estado de la conexión. Entonces, cualquier información que haya sido usada en una autenticación previa no podrá usarse nuevamente. O lo que es lo mismo, cada vez que un usuario final se autentique en un sitio web consumidor sin estado, el mismo proceso es repetido.

A continuación se presenta el paso a paso de la autenticación e inicio de sesión, donde se lleva a cabo la comunicación entre el consumidor, el navegador y el proveedor de identidad.

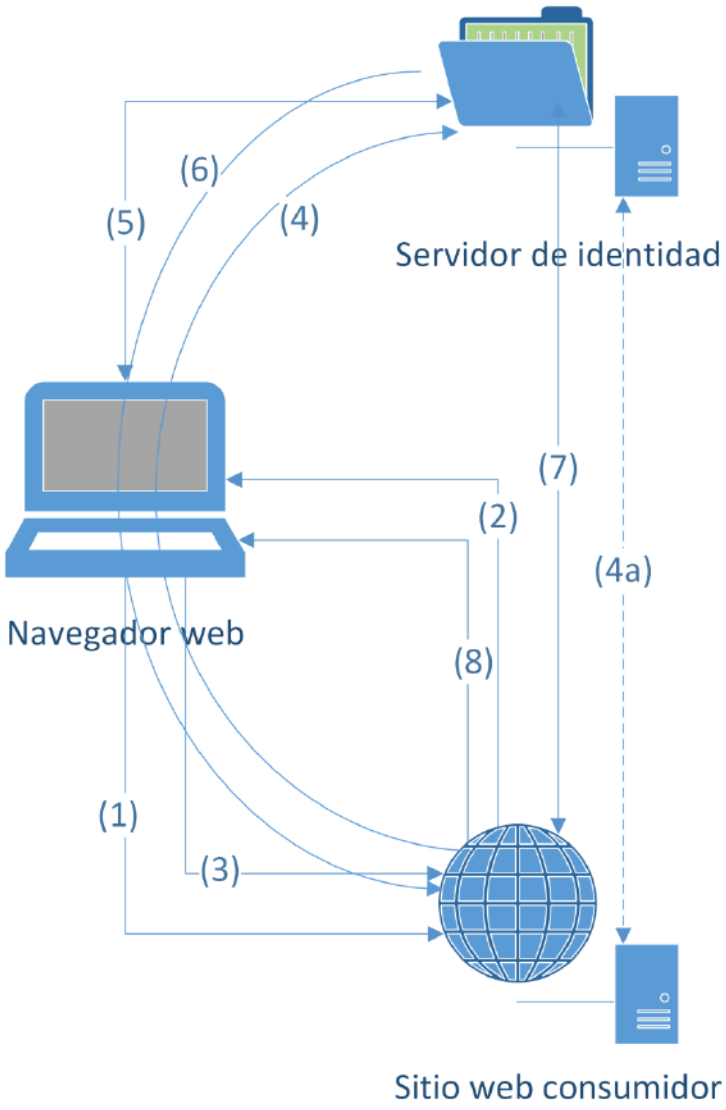

Figura 4.3: Comunicación Modo Sin Estado.

1. El usuario final visita el sitio web consumidor, en el cual desea iniciar sesión.

2. El sitio web presenta una página donde se solicita la URL de identidad. Comúnmente luego se escribir la URL de identidad, se debe hacer clic en un botón de ingreso. La especificación de OpenID requiere que todos los consumidores deben ser suficientemente inteligentes para interpretar la URL en distintos formatos.

3. El sitio web consumidor reformará o limpiará la URL de identificación y la traerá de su ubicación actual. Esta ubicación podría ser la misma que la del proveedor de identidad, o podría ser un servidor diferente.

4. Luego de obtener la página, el consumidor analizará la misma para determinar la ubicación del proveedor de identidad, es decir, la ubicación del servidor OpenID. Esta información será embebida dentro de una página web HTML. Este proceso de análisis es el conocido como “descubrimiento”. Luego del análisis, el consumidor redireccionará al navegador web al proveedor de

identidad para obtener la información de confirmación. Opcionalmente, el consumidor puede establecer, en este punto, una comunicación con el proveedor de identidad e intercambiar una clave secreta compartida para una futura comunicación. Esto se representa posteriormente con una línea punteada en el paso 4a de la Figura 4.4.

5. Si el usuario final no ha iniciado sesión aún en el proveedor de identidad, éste le requerirá al usuario que lo haga. Notar que esta parte ocurre fuera de las especificaciones de OpenID, quedando a consideración del proveedor de identidad decidir cómo autenticar al usuario final. En algunos casos, si el usuario final ya se encontraba autenticado en el sitio web del proveedor de identidad, este paso se omite.

6. El proveedor de identidad retornará la información de confirmación, con su firma, al consumidor, a través de una redirección del navegador. Esta confirmación representa tanto una autenticación exitosa, como una autenticación fallida. Se debe tener en cuenta que esta es la comunicación indirecta entre el proveedor de identidad y el consumidor.

7. Durante la confirmación exitosa, el consumidor establecerá conexión directa con el proveedor de identidad, preferentemente sobre una sesión segura SSL [46], solicitará la información de autenticación directamente al proveedor de identidad, y la comparará con la información de confirmación recibida a través del navegador web. Esto es para realizar un doble chequeo de la validez de la confirmación, para evitar un posible ataque malicioso sobre el navegador web.

8. Si hay una coincidencia en el paso anterior, el usuario final podrá acceder al sitio web. De lo contrario el inicio de sesión fallará.

4.5.2. Modo con estado

El modo de autenticación sin estado es similar al modo con estado, con la excepción del paso número 7 descrito anteriormente. En este caso, el consumidor ya tiene la clave secreta compartida (correspondiente al paso 4a de la Figura 4.4) y puede descifrar y verificar la confirmación del paso 6 y determinar si el proveedor de identidad realmente firmó, es decir si aceptó la autenticación.

Se debe tener en cuenta que el paso 4a se ejecutará sólo en ciertas ocasiones, cuando el consumidor necesite actualizar la clave secreta almacenada o tenga que obtenerla por primera vez. A continuación se describe el proceso paso a paso:

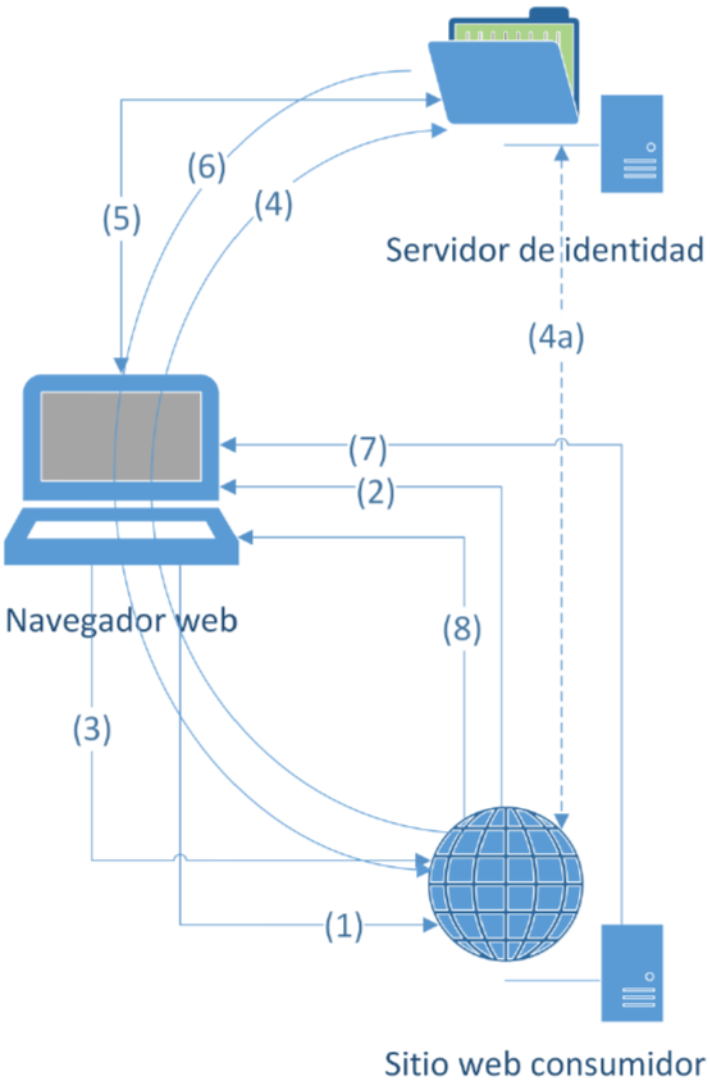

Figura 4.4: Flujo de comunicación del modo con estado durante la primera sesión.

1. El usuario final visita el sitio web consumidor.

2. El sitio web presenta una página donde se solicita la URL de identidad, el usuario final ingresa la URL y hace clic en el botón de ingreso.

3. El sitio web consumidor reformará o limpiará la URL de identificación y la traerá de su ubicación actual.

4. Luego de obtener la página, el consumidor analizará la misma para determinar la ubicación del proveedor de identidad, es decir, la ubicación del servidor OpenID. Luego del análisis, el consumidor redirigirá al navegador web al proveedor de identidad para obtener la información de confirmación de acceso. Opcionalmente, el consumidor puede enviar una solicitud de asociación con el proveedor de identidad e intercambiar una clave compartida como se muestra en el paso 4a en la Figura 4.4.

5. Si el usuario final no ha iniciado sesión aún en el proveedor de identidad, éste le requerirá al usuario que lo haga.

6. El proveedor de identidad retornará la información de confirmación, al consumidor, a través de una redirección del navegador. Esta confirmación representa tanto una autenticación exitosa, como una autenticación fallida.

7. Después de la confirmación exitosa, el consumidor verifica la confirmación usando la clave compartida almacenada. SI hay coincidencia en el paso anterior, el usuario final iniciará sesión en el sitio web. De lo contrario, el inicio de sesión fallará.

En muchos casos cuando ya se han establecido las credenciales con el proveedor de identidad, el mismo no solicitará nuevamente el inicio de sesión. Existen varias técnicas para conseguir este objetivo que pueden ser usadas por los proveedores de identidad. En el contexto mencionado anteriormente, y asumiendo que el consumidor tiene en almacenada la clave secreta compartida del proveedor de identidad, la comunicación se simplifica y el usuario final iniciará sesión en el sitio web consumidor sin interactuar con el proveedor de identidad.

Tener en cuenta que el usuario final obtendrá acceso al sitio web inmediatamente después de introducir la URL de identidad. Este caso se describe a continuación:

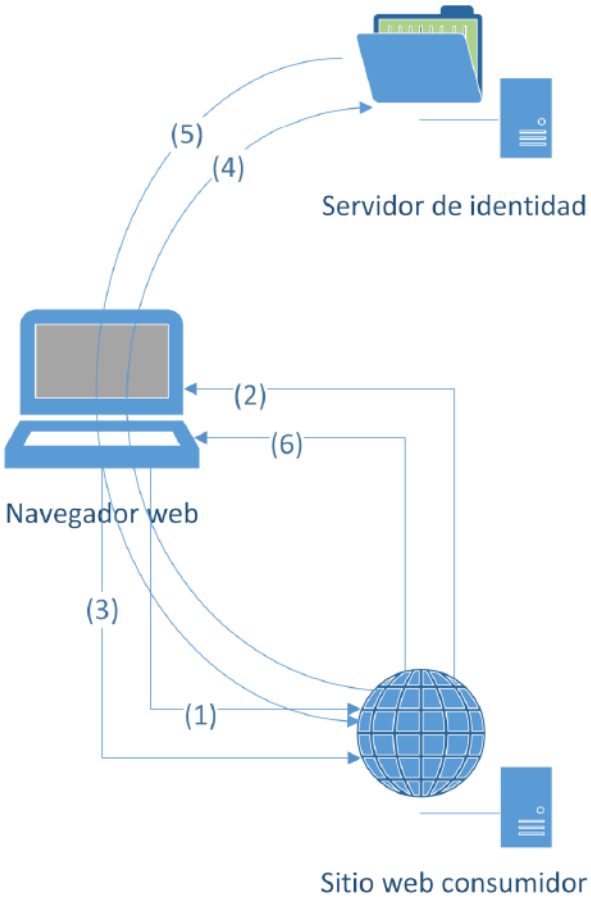

Figura 4.5: Flujo de comunicación del modo con estado y el posterior inicio de sesión en que el consumidor y el proveedor de identidad ya han establecido un secreto compartido.

1. El usuario final visita el sitio web consumidor.

2. El sitio web presenta una página donde el usuario introduce la URL de identidad.

3. El sitio web consumidor reformará o limpiará la URL de identificación y la traerá de su ubicación actual.

4. Luego de obtener la página, el consumidor analizará la misma para determinar la ubicación del proveedor de identidad, es decir, la ubicación del servidor OpenID. Luego del análisis, el consumidor redirigirá al navegador web al proveedor de identidad para obtener la información de confirmación.

5. El proveedor de identidad retornará la información de confirmación al consumidor a través de una redirección del navegador. Esta confirmación representa tanto una autenticación exitosa, como una autenticación fallida.

6. Después de la confirmación exitosa, el consumidor verifica la confirmación usando la clave compartida almacenada. SI hay coincidencia en el paso anterior, el usuario final iniciará sesión en el sitio web. De lo contrario, el inicio de sesión fallará.

Es importante tener en cuenta que, la comunicación con el proveedor de identidad es transparente al usuario y las operaciones ocurrirán detrás de escena mediante redirecciones del navegador.

4.6. Mensajes de OpenID

Las partes que componen el proceso de autenticación mediante OpenID intercambian diferentes mensajes, los cuales tienen un formato bien definido y a los cuales los sitios web consumidores y los proveedores de identidad deben adherirse para que ocurra una comunicación exitosa.

Cada mensaje tiene una combinación de solicitud y respuesta, donde la solicitud y la respuesta pueden tener diferentes conjuntos de parámetros.

Dependiendo del tipo de mensajes, el consumidor y el proveedor de identidad pueden usar los métodos de comunicación directa o indirecta.

Notar que los métodos HTTP POST son usados en la comunicación directa, mientras que los métodos HTTP GET [47] son usados en la comunicación indirecta.

Existen cuatro mensajes básicos que se utilizan en los sistemas de OpenID:

1. El mensaje de asociación.
2. El mensaje check_inmmeadiate.
3. El mensaje check_setup.
4. El mensaje check_authentication.

4.6.1. El mensaje de solicitud de asociación

El mensaje de asociación es enviado por el consumidor para el proveedor de identidad. El principal propósito de este mensaje es establecer un secreto compartido entre el consumidor y el proveedor de identidad. Se debe tener en cuenta que el sitio web consumidor puede enviar este mensaje al proveedor de identidad en cualquier momento que sea requerido.

Como se trata de una comunicación directa entre el consumidor y el proveedor de identidad, se utiliza el método HTTP POST para este mensaje. Mientras se inicia la petición de asociación, el sitio web consumidor enviará un número de parámetros junto con la solicitud. A continuación se presenta una lista de los parámetros que pueden ser enviados con la solicitud de asociación:

- openid.ns
- openid.mode
- openid.assoc_type
- openid.session_type
- openid.dh_modulus
- openid.dh_gen
- openid.dh_consumer_public

openid.ns

Este es un parámetro opcional, cuyo propósito es definir el número de versión de OpenID que se utilizará para un mensaje particular. El valor de este parámetro será “http://specs.openid.net/auth/2.0” para la versión 2.0. Si este parámetro no está presente, o su valor es “http://openid.net/signon/1.1” o “http://openid.net/signon/1.0”, el proveedor de identidad se ejecutará en modo de compatibilidad con versiones anteriores. Este parámetro no está presente en la especificación de OpenID versión 1.1, sino que fue incluido en la versión 2.0. En el futuro cuando se presenten versiones nuevas de OpenID, el valor de este parámetro cambiará acorde a las nuevas versiones.

openid.mode

Este parámetro muestra el tipo de mensaje y se utiliza para distinguir qué mensaje está siendo enviado o recibido. El mismo está presente en todos los mensajes de OpenID, por ejemplo, el valor de este parámetro será “associate” para el mensaje de solicitud de asociación.

openid.assoc_type

Este parámetro se utiliza para transmitir el algoritmo utilizado para firmar el mensaje. Si el valor de este parámetro es “HMAC-SHA1” entonces el mecanismo de la firma es HMAC-SHA1 [48]. En cambio si el valor es “HMAC-SHA256” el mecanismo de la firma es HMAC-SHA256 [49]. Actualmente estos son los dos valores soportados por OpenID.

openid.session_type

Este parámetro se utiliza para mostrar el tipo de encriptación usado en el mensaje para encriptar la clave MAC (Message Authentication Code) [50]. La MAC es un código corto que es calculado usando una clave secreta y un mensaje de entrada. El algoritmo MAC toma la clave privada y el mensaje como entrada y provee como salida el código MAC. HMAC (Hashed Message Authentication Code) es un tipo de MAC.

En los mensajes de OpenID, se utilizan diferentes tipos de encriptación para encriptar la clave MAC:

- "DH-SHA1" para Diffie-Hellman [51] SHA1
- "DH-SHA256" para Diffie-Hellman SHA256
- "no-encryption" para enviar MAC sin encriptar.

Se recomienda siempre usar encriptación. Sin embargo, si se utiliza una conexión segura SSL entre el proveedor de identidad y el sitio web consumidor, se puede elegir no encriptar la MAC ya que la capa de transporte estaría proporcionando el encriptado.

Se debe tener en cuenta que si el proveedor de identidad no soporta los parámetros openid.assoc_type y openid.session_type presentados en la solicitud, contestará con una respuesta de asociación fallida.

openid.dh_modulus, openid.dh_gen y openid.dh_consumer_public

Si se eligió DH-SHA1 o DH-SHA256 como valor del parámetro openid.session_type, los tres parámetros openid.dh_modulus, openid.dh_gen y openid.dh_consumer_public serán parte del mensaje de asociación. Las especificaciones de OpenID [52] describen cómo se deben generar estos parámetros.

Ejemplo de solicitud de asociación

A continuación se muestra un ejemplo de solicitud de asociación:

```
openid.mode=associate&openid.assoc_type=HMACSHA1&openid.session_
type=DHSHA1&openid.dh_consumer_public=KC6IpA00A6slCikafFSlrTGql9
H8+de6GFi5YLKz4pyDxUMS5Z8pMOm/Ptr1gFmCcgAXjFbuxS73ZutDTFJYpADoIn
tFVrah9eaezMcw6SDR24cnFjNc14xq0zGt3QcRLXaNTRVKfMW8evDAmLCrvEhU5c
7B3eqmk+bMMrbQpE=&openid.dh_modulus=ANz5OguIOXLsDhmYmsWizjEOHTdx
fo2Vcbt2I3MYZuYe91ouJ4mLBX+YkcLiemOcPym2CBRYHNOyyjmG0mg3BVd9RcLn
5S3IHHoXGHblzqdLFEi/368Ygo79RnxTkXjgmY0rxlJ5bU1zIKaSDuKdiI+XUkKJ
X8Fvf8W8vsixYOr&openid.dh_gen=Ag==
```

4.6.2. El mensaje de respuesta de la asociación

El mensaje de respuesta a la petición de asociación es enviado desde el proveedor de identidad al sitio web consumidor. El mismo es un mensaje HTTP 200, que indica si la asociación tuvo éxito o si falló. En caso de una asociación

exitosa, el mensaje contendrá un manejador para ese mensaje y tiempo de vida para ese manejador, expresado en segundos. Por otro lado, en caso de una asociación fallida, se retorna un error.

A continuación se presenta una lista con los parámetros que forman parte de una respuesta exitosa. Notar que los parámetros que fueron explicados anteriormente sólo se listan, pero no se explican.

- openid.ns
- openid.assoc_handle
- openid.session_type
- openid.assoc_type
- openid.expires_in
- openid.mac_key parameter
- openid.server_public
- openid.enc_mac_key

openid.assoc_handle

Este parámetro se utiliza como una clave para referirse a la asociación en cuestión en los mensajes subsecuentes. Es una cadena imprimible ASCII [53] con un tamaño máximo de 255 caracteres. Comúnmente como la asociación dura por un tiempo, este identificador se usa para determinar qué clave debe reutilizarse para la encriptación/desencriptación.

openid.session_type

Este parámetro es el mismo que el de la solicitud si la asociación es exitosa, pero si falla, el valor de este parámetro será “unsuccessful response” (respuesta sin éxito). Puede haber distintas razones para una asociación fallida, por ejemplo, si el consumidor solicita utilizar SHA256 y el proveedor de identidad solo soporta SHA1.

openid.assoc_type

Este parámetro se maneja igual que openid.session_type, si la asociación es exitosa, su valor será el mismo al que tenía en el mensaje de solicitud, y en caso de que la asociación falle, su valor será “unsuccessful response”.

openid.expires_in

Es el tiempo en segundos después del cual caduca la asociación. Al caducar la asociación el consumidor deberá solicitar una nueva asociación.

openid.mac_key parameter

Este parámetro sólo se utiliza si el valor de “openid.session_type” fue “no-encryption”. El valor de este parámetro es la clave MAC codificada en base-64 [54]. Esta clave también se denomina clave secreta compartida, o simplemente secreto compartido.

openid.server_public

Es la clave pública del proveedor de identidad. Este parámetro se usa si el algoritmo Diffie-Hellman fue utilizado en la solicitud.

openid.enc_mac_key

Es la clave MAC encriptada. Este parámetro se usa si el algoritmo Diffie-Hellman fue utilizado en la solicitud.

Ejemplo de respuesta

A continuación se muestra una respuesta típica para un mensaje de solicitud de asociación:

```
assoc_handle:{HMAC-SHA1}{4607344a}{oDFF0g==}
assoc_type:HMAC-SHA1
dh_server_public:AIPkx6xJ3b1Wnr1olWL7suoZnABDc+lJRR9DeNIBolGXQX3
W2e+4udY2p+dUcF5jKE6uoZuXLVPbimHbndBOYhUDUfkKaAjQtVvONerAjd5RHyt
2i2AoYrkjD26traC4jzg7NukZlmrRjfPRg4q3gwW+EZEXvz+ba9JnQfsXx+iH
enc_mac_key:UtQHBswQimAZAp4s/9sfSQSpuq0=
expires_in:1209600
session_type:DH-SHA1
```

En caso de una asociación fallida, los parámetros “error” y “error_code” son también retornados. Estos son los más importantes pero cabe destacar que también existen otros parámetros opcionales que se retornan en una asociación fallida.

4.6.3. Los mensajes de solicitud checkid_setup y checkid_immediate

Los mensajes checkid_immediate y checkid_setup son usados para obtener información de aserción del servidor de OpenID. Estos mensajes son iniciados por el sitio web consumidor y utilizan comunicación indirecta, por ende, el consumidor usará el método HTTP GET para enviar y recibir estos mensajes y los mismos pasarán a través del navegador web.

Estos mensajes son similares, con algunas diferencias de menor importancia y casos de uso.

A continuación se presenta una lista con los parámetros que utiliza el mensaje de solicitud checkid_setup. Los parámetros explicados previamente, no se vuelven a explicar.

- openid.ns
- openid.mode
- openid.claimed_id
- openid.identity
- openid.assoc_handle
- openid.return_to
- openid.realm

openid.mode

En este tipo de mensaje, este parametro contendrá el valor “checkid_setup”.

openid.claimed_id

Parámetro opcional con el identificador reclamado (también conocido como claimed id). Este identificador es una URL que el usuario final dice poseer pero aún no fue verificada.

openid.identity

Parámetro opcional que sirve como identificador local del proveedor de identidad. En caso de que este parámetro no esté presente, se utiliza el openid.claimed_id como valor para openid.identity.

openid.assoc_handle

Parámetro opcional, que si está presente, muestra la asociación que ya fue establecida por el servidor OpenID y el consumidor. En caso de no estar presente, la transacción se llevará a cabo en un modo sin estado.

openid.return_to

Este parámetro se utiliza para informar al proveedor de OpenID la ubicación de la URL donde debe redirigir al navegador después de procesar la solicitud. El proveedor OpenID usará esta URL para enviar la respuesta al sitio web consumidor.

openid.realm

Este es otro parámetro opcional, es una URL utilizada por el proveedor de OpenID para identificar al consumidor de una manera única. El mismo puede contener comodines como “*” para definir la URL. Por ejemplo: http://*.conformix.com.

Hay que tener en cuenta que las especificaciones de OpenID 1.1, los parámetros openid.claimed_id y openid.realm no están presentes. En su lugar se encuentra el parámetro openid.trust_root, el cual es opcional y muestra la URL actual del sitio web consumidor.

También se debe tener en cuenta que el mensaje de solicitud llega al proveedor de OpenID en dos pasos. En el primero, el consumidor realiza un redirección HTTP 302 hacia el navegador web. En el siguiente paso, el navegador envía una petición HTTP GET al proveedor de OpenID. Esto se muestra en la Figura 4.6.

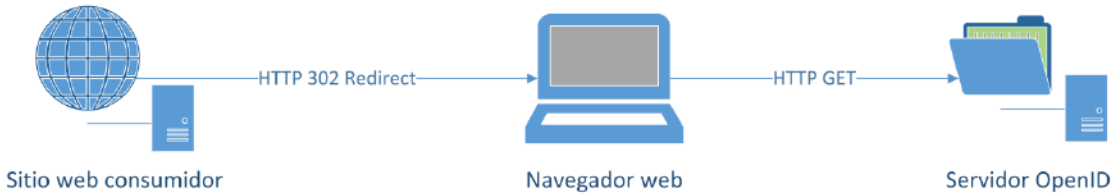

Figura 4.6: Flujo para el mensaje de solicitud checkid_setup.

Ejemplo de mensaje checkid_setup

A continuación se presenta un ejemplo de un mensaje checkid_setup capturado de una comunicación real entre un servidor OpenID y un sitio web consumidor.

```
GET
/index.php/serve?openid.assoc_handle={HMACSHA1}{46071e25}{Tt8MwQ
==}&openid.identity=http://idp.conformix.com/?user=openidbook&op
enid.mode=checkid_setup&openid.return_to=http://consumer.conform
ix.com:80/finish_auth.php?nonce=nC5sKquX&openid.sreg.optional=em
ail&openid.trust_root=http://consumer.conformix.com:80/ HTTP/1.1
```

4.6.4. Los mensajes de respuesta checkid_setup y checkid_immediate

Un vez que el proveedor de OpenID recibe el mensaje checkid_setup, realiza algún tipo de procesamiento y envía una respuesta al consumidor a través del navegador web. Opcionalmente, el proveedor de OpenID puede solicitar al usuario final que se autentique para asegurar que sólo el propietario de la URL de identificación autorice el acceso a ese identificador. La implementación de esta funcionalidad queda en manos del proveedor de OpenID y no está incluida en las especificaciones de OpenID.

En la respuesta, el proveedor de OpenID enviará múltiples parámetros al sitio web consumidor, algunos de ellos son opcionales y otros requeridos, los parámetros que ya fueron detallados anteriormente son omitidos en la descripción.

- openid.ns
- openid.mode
- openid.op_endpoint
- openid.claimed_id
- openid.identity
- openid.assoc_handle
- openid.return_to
- openid.response_nonce
- openid.invalidate_handle
- openid.signed
- openid.sig

openid.mode

Este parámetro tendrá el valor “id_res” en caso de éxito. Mientras que si la asociación falla, el valor será “setup_needed”, en caso de que la solicitud haya sido checkid_immediate, o será “cancel” si la solicitud fue checkid_setup.

openid.op_endpoint

Este parámetro contiene la URL del proveedor de OpenID.

openid.assoc_handle

Es el identificador que se utiliza para firmar este mensaje. El consumidor va a utilizar este identificador para propósitos de verificación. Este identificador puede ser el mismo que el que fue enviado por el consumidor con el mensaje de solicitud (en caso de que ya exista una asociación). Puede que sea diferente si el proveedor de OpenID no reconoce el identificador original, en tal caso, el servidor guarda un registro del nuevo identificador para que el consumidor puede usarlo para realizar la correspondiente verificación haciendo uso del mensaje check_authentication.

openid.return_to

Este parámetro es una copia de la URL que el consumidor envió en el mensaje de solicitud.

openid.response_nonce

Este parámetro se utiliza para evitar ataques y es único para cada mensaje. El tamaño máximo es de 255 caracteres y consiste de una secuencia de caracteres, que denotan la hora y fecha del servidor y caracteres ASCII adicionales para hacerlo único. El tiempo es tomado en formato UTC16 [55]. Se debe tener en cuenta que el consumidor puede rechazar la asociación si el valor de este parámetro es demasiado lejano a la hora actual.

openid.invalidate_handle

Este parámetro es utilizado para mostrar si el identificador openid.assoc_handle adjuntado con la solicitud fue válido o no. Si fue válido, este parámetro es opcional, en caso contrario, es requerido, ya que ayudará al consumidor a borrar los identificadores inválidos de sus registros.

openid.signed

Contiene una lista de parámetros firmados, separados por coma. Esta lista está formada por los nombres de los parámetros sin el prefijo "openid."

openid.sig

Contiene la firma codificada en base-64.

Como con el mensaje de solicitud, el mensaje de respuesta alcanza al sitio web consumidor en dos pasos. En el primer paso, el proveedor de OpenID usa el método HTTP 302 para realizar una redirección del proveedor de OpenID al navegador web. Luego en el siguiente paso, el navegador envía una solicitud HTTP GET al consumidor. Esto es representado en la Figura 4.7.

Figura 4.7: Flujo del mensaje de respuesta checkid_setup

Ejemplo del mensaje de respuesta checkid_setup

A continuación se muestra un ejemplo del mensaje de respuesta checkid_setup extraído de una comunicación real entre un servidor OpenID y un sitio web consumidor.

```
GET
/finish_auth.php?nonce=nC5sKquX&openid.assoc_handle={HMACSHA1}{4
6071e25}{Tt8MwQ==}&openid.identity=http://idp.conformix.com/?use
r=openidbook&openid.mode=id_res&openid.return_to=http://consumer
.conformix.com:80/finish_auth.php?nonce=nC5sKquX&openid.sig=nXWc
+07GLaSf+RghmGubGPPglZc=&openid.signed=mode,identity,return_to,s
reg.email&openid.sreg.email=rr@conformix.com HTTP/1.1
```

Es importante tener en cuenta que en las especificaciones de OpenID 1.1, si la confirmación falla, el parámetro "openid.user_setup_url" es enviado en el mensaje de respuesta, el cual contiene una URL que puede ser usada para redirigir al navegador web a los siguientes pasos.

4.6.5. El mensaje de solicitud check_authentication

Los mensajes de solicitud y respuesta check_authentication son una herramienta necesaria para verificar la respuesta exitosa recibida del navegador web. Esto es para asegurar que un atacante no está enviando mensajes de afirmación hechos a mano en nombre del proveedor OpenID. Se deben tener en cuenta lo siguientes puntos acerca de los mensajes check_authentication:

1. Este mensaje no es enviado si ya existe una asociación entre el sitio web consumidor y el proveedor de OpenID.

2. Si se utiliza una asociación existente, el consumidor enviará el parámetro "openid.assoc_handle" en la solicitud y el proveedor de OpenID devolverá el mismo valor en la respuesta. Si esto pasa, el sitio web consumidor sabrá que el proveedor de OpenID está de acuerdo en usar el identificador de asociación existente.

3. Si el proveedor de OpenID no está de acuerdo en usar el identificador de asociación provisto por el sitio web consumidor, la respuesta incluirá el parámetro "openid.invalidate_handle" y un diferente "openid.assoc_handle". A su vez, el consumidor usará el mensaje check_authentication para validar la afirmación.

4. Cuando se utiliza el modo sin estado, este mensaje de respuesta se usará siempre porque el consumidor no maneja estados y no guarda registros de ningún identificador de asociación previo.

Se debe tener en cuenta que este mensaje realizar una comunicación directa entre el consumidor y el proveedor de OpenID utilizando el método HTTP POST.

Ejemplo de un mensaje check_authentication

La solicitud contiene el parámetro "openid.mode" con el valor "check_authentication" y todos los parámetros restantes que formaban parte del mensaje de respuesta anterior. Un mensaje típico se ve como el siguiente:

```
openid.assoc_handle={HMACSHA1}{460730e1}{zr1gKg==}&openid.identi
ty=http://idp.conformix.com/?user=openidbook&openid.invalidate_h
andle={HMACSHA1}{46071e25}{Tt8MwQ==}&openid.mode=check_authentic
ation&openid.return_to=http://consumer.conformix.com:80/finish_a
uth.php?nonce=mAotRbGM&openid.sig=4hwwyWbPtSAmP2dYxEC+dq605Os=&o
penid.signed=mode,identity,return_to,sreg.email&openid.sreg.emai
l=rr@conformix.com
```

4.6.6. El mensaje de respuesta check_authentication

En respuesta al mensaje check_authentication, el proveedor de OpenID envía un mensaje con los siguientes parámetros:

- openid.ns
- is_valid
- invalidate_handle

is_valid

Este parámetro puede ser "true" o "false" (verdadero o falso), dependiendo si la solicitud fue válida o no.

invalidate_handle

Este parámetro es opcional y en caso de que el parámetro is_valid sea verdadero, el consumidor eliminará el identificador de asociación de su lista.

Ejemplo de un mensaje check_authentication

A continuación se muestra un ejemplo de este mensaje:

```
invalidate_handle:{HMAC-SHA1}{46071e25}{Tt8MwQ==}
is_valid:true
```

Notar que este es un mensaje muy corto y la respuesta incluye sólo el éxito o fallo de la comprobación de autenticación.

4.7. Funcionamiento de OpenID en algunos escenarios

Como se ha mencionado anteriormente, OpenID se utiliza para autenticar a un usuario en diferentes sitios web usando una credencial almacenada en un proveedor de identidad. Hay diferentes escenarios de cómo la autenticación mediante OpenID se lleva a cabo.

4.7.1. Primer acceso a un sitio web usando OpenID en modo sin estado

Cuando un usuario se autentica a un sitio web con OpenID por primera vez se realizan ciertas acciones. A continuación se lista una serie de pasos:

1. El usuario ingresa su identificador de OpenID en la página de inicio de sesión del sitio web consumidor y presiona el botón de ingreso.

2. El sitio web consumidor localiza el proveedor de OpenID y probablemente utilice el protocolo Yadis para descubrir el proveedor provisto en la URL de identificación. Luego redireccionará al navegador web a ese proveedor de OpenID para obtener las credenciales del usuario.

3. Como es la primera vez que el usuario se autentica en el sitio web, el proveedor de OpenID no sabe si dicho usuario confía o no en el sitio web, por tal motivo muestra una pantalla para que el usuario se autentique con sus credenciales de OpenID.

4. Luego el usuario marca este sitio web como un sitio web de confianza en el proveedor de OpenID.

5. El usuario es redirigido nuevamente al sitio web consumidor.

6. El sitio web comprueba la autenticación del usuario con el proveedor de OpenID.

7. La autenticación se completa.

Vale la pena destacar que en el modo con estado, tanto el sitio web consumidor como el proveedor de OpenID establecen una asociación y guardan registro de esta, mediante un identificador, para usarla en el futuro. Pero en modo sin estado, no se establece la asociación y el consumidor no guarda ningún registro.

4.7.2. Acceso a un sitio web de confianza usando OpenID en modo con estado

Un usuario puede visitar sitios web de manera regular. Si estos sitios soportan OpenID, el usuario puede marcar estos sitios web como "sitios de confianza" en el proveedor de OpenID donde se almacenan sus credenciales. Una vez que estos sitios fueron marcados como “sitios de confianza”, el proceso de autenticación se realiza automáticamente a través de los siguientes pasos:

1. El usuario ingresa su identificador de OpenID en la página de inicio de sesión del sitio web consumidor y presiona el botón de ingreso.

2. El sitio web consumidor establece la asociación con el proveedor de OpenID, si no existe una asociación válida.

3. El sitio web localiza al proveedor de OpenID y redirige al navegador a dicho proveedor para obtener las credenciales del usuario. Posiblemente se utilice Yadis para el descubrimiento de los servicios.

4. El proveedor de OpenID sabe que este sitio web es de confianza para el usuario, y sabe qué parámetros debe pasarle. Si el usuario ya se ha autenticado en el proveedor de OpenID, el mismo proveerá las credenciales necesarias para que se pueda llevar a cabo la autenticación en el sitio web.

5. El usuario se encontrará autenticado en el sitio web y todos los pasos se ejecutarán automáticamente sin intervención por parte del usuario.

4.8. Seguridad

En esta sección se analiza la seguridad de OpenID desde el punto de vista de los actores principales que tiene el proceso de autenticación: el usuario final, el sitio web consumidor y el proveedor de identidad. Se debe tener en cuenta que las cuestiones enumeradas aquí no son solamente relacionadas con el protocolo, son una combinación de deficiencias del protocolo, el navegador y las prácticas de implementación.

4.8.1. El Usuario Final

El usuario final puede causar problemas al sitio web consumidor, cuando este último le solicita al usuario por su identificador de OpenID.

Este identificador es comúnmente una URL, la cual es usada en la fase de descubrimiento. El identificador es adquirido a través de un campo de texto donde el usuario ingresa lo que desea. Por lo tanto, el usuario final puede forzar al servidor del sitio web consumidor a contactar la URL que él específica. Con lo cual el usuario podría realizar las acciones que se describen a continuación.

Obtener acceso a hosts internos

Primero que nada, el usuario podría forzar al sitio web consumidor a contactar hosts o servidores internos por dentro del firewall externo.

Por ejemplo, suponiendo que el usuario final ingresa una URL como la que se muestra a continuación como su identificador OpenID:

http://localhost/admin.php?action=sql&query=DROP TABLE user

Esto puede causar que el servidor del sitio web consumidor se contacte a sí mismo a través de una petición HTTP GET, lo que es una amenaza seria a la seguridad, ya que la mayoría de los servidores toman menos medidas de seguridad contra las conexiones locales entrantes.

Forzar contacto con hosts externos

El usuario final puede también forzar al sitio web consumidor a contactar otros hosts y servidores.

Por ejemplo, suponiendo que el usuario final ingresa la siguiente URL como su identificador:

http://www.example.com/securityflaw.php?attack=true

Si un usuario final encuentra una falla de seguridad en algún sitio web, puede ser aprovechada de forma remota desde el sitio web consumidor, lo que termina haciendo parecer que la culpa la tiene el sitio web consumidor.

Inundación de datos

Los sitios web deben protegerse de grandes cantidades inusuales de datos que pueden ingresar a través de la URL de OpenID del usuario final. Por ejemplo, si se utiliza la siguiente URL como identificador de OpenID, www.example.com/largefile.bin, esta puede causar problemas al sitio web consumidor en caso de que el archivo “largegile.bin” sea demasiado grande.

Ataque de denegación de servicio

Por último el sitio web consumidor debe estar atento a los ataques de denegación de servicio. Este tipo de ataques se produce cuando el servidor es inundado por más solicitudes de las que puede manejar y por lo tanto deja de estar disponible.

Solución: Filtrar el identificador de OpenID

Una posible solución para evitar los ataques por parte del usuario, es que el sitio web consumidor filtre el campo en donde se ingresa el identificador de OpenID.

Primero que nada, no se deben permitir las direcciones IP [56] privadas [57] ni los posibles nombres de dominio internos, ya que no deberían usarse como identificadores de OpenID. Se podría contemplar también la prohibición de todas las direcciones IP por completo, pero esto probablemente no sea aconsejable, ya que algunos usuarios podrían realmente usar su dirección IP como su identificador OpenID.

Este filtrado resuelve los problemas mencionados anteriormente, donde un usuario final podía forzar al sitio web consumidor a contactar a los hosts internos.

En segundo lugar, el consumidor debería considerar la prohibición de los hosts que ingresan repetidamente utilizando identificadores que no superan la fase de descubrimiento, lo cual indica que el identificador suministrado no conduce a un proveedor de OpenID en funcionamiento. Más aún, el consumidor debe prohibir los hosts que intentan iniciar demasiadas peticiones de autenticación en un intervalo corto de tiempo, esto para evitar un ataques de denegación de servicio.

La implementación de estas reglas de filtrado protege al sitio web consumidor de la mayoría de los ataques maliciosos que el usuario final puede perpetrar. Estas precauciones son fáciles de implementar y son indispensables para cualquier sitio web consumidor serio.

4.8.2. El sitio web consumidor

Uno de los ataques más serios que pueden presentarse en OpenID es el denominado “phishing” [58]. Este tipo de ataque consiste en un engaño al usuario por parte del sitio web consumidor, es decir, en el proceso de autenticación, en vez de redirigir al usuario final al sitio web del proveedor de OpenID, el consumidor redirige al usuario a algún otro servidor. Si este servidor web presenta una interfaz que simula ser el proveedor de identidad que el usuario final utiliza para autenticarse, el mismo podría tratar de iniciar sesión en un proveedor de identidad falso. Si el proveedor del usuario utiliza un método de autenticación débil, como un nombre de usuario y contraseña, el usuario final dará, sin saberlo, sus credenciales de acceso al proveedor falso.

Este tipo de ataque, como se dijo anteriormente, es conocido como phishing, ya que el consumidor está a la pesca de las credenciales de autenticación de los usuarios. El phishing es excepcionalmente difícil de descubrir para un usuario final, el único signo revelador real es la URL. Esta es la única parte, de la apariencia del proveedor, que es difícil de falsificar.

Solución: Estar atento al Phishing

El phishing es un problema de seguridad tanto para el usuario final como para el proveedor de identidad, por ello a continuación se listan posibles soluciones para ambos.

Existen dos soluciones principales para el usuario final. La primera es manual y la segunda es automática.

La solución manual se trata de verificar la URL del proveedor de identidad, es decir que el usuario final debe verificar la URL de la página donde está intentando realizar la autenticación. Por lo general la URL de la página falsa, que trata de engañar al usuario, tiene un dominio totalmente diferente al del proveedor de identidad del usuario, lo cual es una indicación de que alguien puede estar intentando un ataque de phishing.

El segundo método, es un método automático y consiste en extensiones para el navegador que ayudan a detectar el phishing. Estos sistemas automáticos usan el mismo método que el usuario, verificar la URL, especialmente el nombre de dominio. Un buen ejemplo es VeriSign's OpenID SeatBelt Plugin [59].

Aunque estas herramientas automáticas son una buena alternativa, se recomienda a todos los usuarios de OpenID que presten especial atención y verifiquen las URL's a las que se autentican, ya que esto reduce enormemente el riesgo de ser víctima de phishing ya sea para obtener sus credenciales de acceso del banco o simplemente para su cuenta de OpenID.

Por otro lado, la solución para el proveedor de identidad es hacer la página de autenticación fácil de reconocer para el usuario final, pero difícil de duplicar para evitar el phishing. Hay varias maneras de hacer esto, uno de los conceptos más exitosos e interesantes es lo que se conoce como sello de inicio de sesión. Este sello consiste en una pequeña imagen o texto personalizado, que se vincula con cada computadora, de manera tal, que cada vez que el usuario se autentique desde una computadora específica verá su sello único antes de ingresar sus credenciales. Si el usuario final no ve su sello cuando se le presenta la página de autenticación de su proveedor de identidad, debería tener cuidado porque podría estar frente a un posible ataque de phishing.

La desventaja que tiene esta técnica es que sólo funciona para las computadoras para las cuales el usuario creó un sello, por lo tanto los usuarios que cambian constantemente de computadora no se beneficiarán con la solución.

4.8.3. El Proveedor de Identidad

El proveedor de OpenID tiene poder sustancial ya que es quien tiene el privilegio de realizar la autenticación del usuario.

Esto significa que el proveedor puede hacerse pasar fácilmente a sí mismo como uno de sus usuarios, lo cual puede ser un grave problema de seguridad ya que un proveedor de OpenID podría estar relacionado con muchos sitios webs diferentes para el usuario final.

Elegir el proveedor correcto

La posibilidad de encontrarse con un proveedor de identidad malicioso puede causar grandes problemas y es un problema técnicamente imposible de resolver ya que la arquitectura se construye alrededor de un proveedor de autenticación unilateralmente. Si el proveedor decide "portarse mal" nadie puede detenerlo.

La clave para evitar este problema es la flexibilidad inherente de la arquitectura centrada en el usuario que presenta OpenID. La misma permite a los usuarios elegir un proveedor en quién confían o si lo desean, pueden iniciar su propio proveedor de autenticación. Aunque esto requiere que los usuarios finales tengan los conocimientos suficientes para darse cuenta de lo potencialmente destructivo que un proveedor de identidad puede llegar a ser.

Delegación de la autenticación

Es cierto que un sistema donde la seguridad queda dada únicamente a la elección del correcto proveedor de identidad no es algo muy serio. Por eso es que se pueden tomar otras medidas de seguridad como la delegación de la autenticación.

Una de las claves para protegerse contra los proveedores maliciosos es hacerse la pregunta de quién es dueño de mi identidad.

En el caso de OpenID, para responder esta pregunta, hay que señalar que el término proveedor en "proveedor de OpenID" se refiere a proporcionar un medio de autenticación y no a proporcionar una identidad. Lo cual lleva a pensar en la delegación de autenticación, que describe cómo un usuario final puede utilizar su propia URL como identidad, para luego ser redirigido (o delegado) a un proveedor.

Este es un sistema bastante único ya que el usuario final en este caso posee su propia identidad (la URL) y sólo delega la autenticación. Esto es extremadamente importante, ya que resuelve el problema de los proveedores que se vuelven maliciosos, ya que si el proveedor de OpenID se vuelve malicioso el usuario final puede simplemente delegar la autenticación a un nuevo proveedor que crea más seguro.

4.9. Ventajas y Desventajas

OpenID promete marcar el comienzo de una nueva etapa donde se simplifiquen las autenticaciones en los sitios web, brindando ventajas tanto para el usuario promedio como también para los desarrolladores de sitios web. Pero, sin duda, esta tecnología afecta también a los administradores de seguridad web, así como a las empresas que trabajan con sitios web. A continuación se presentan los beneficios e inconvenientes claves de OpenID.

4.9.1. Ventajas

Uno de los grandes beneficios de OpenID es que es un protocolo relativamente seguro, basado en la confianza. El hecho de que un sitio web utilice OpenID puede servir para que este sitio se vea más confiable ante los ojos de los usuarios, más aún, muchos usuarios probablemente ya poseen cuentas de OpenID, por ejemplo Google y Yahoo! son proveedores de OpenID.

Muchos usuarios de internet utilizan la misma contraseña en varios sitios web. Y puesto que las contraseñas tradicionales no son administradas centralmente, si un problema de seguridad se produce en cualquier sitio web que utilice el usuario, un atacante puede tener acceso a su contraseña con la cual podrá acceder a varios de los sitios que el usuario frecuenta. Con OpenID, las contraseñas no se comparten con los sitios web, y si ocurre un problema, sólo se tiene que cambiar la contraseña de la cuenta de OpenID.

Debido a que el foco principal de la mayoría de los proveedores de OpenID es en la gestión de la identidad, puede que la protección de dicha identidad sea más completa. Es probable que la mayoría de los administradores de sitios web convencionales no se dediquen en profundidad a la protección de la identidad del usuario como lo hacen los proveedores de OpenID.

Los sitios web que utilizan OpenID podrían no implementar su propio sistema de autenticación delegando esta tarea a OpenID, es decir a proveedores externos, por lo que se vuelve un recurso invaluable. Para las empresas, el hecho no tener que implementar su sistema de autenticación significa una reducción de costos y mantenimiento asociados con el almacenamiento y gestión de la información de perfil del usuario y manejo de contraseñas.

Otro factor a destacar es que OpenID favorece al aumento de tráfico en los sitios web. No es un secreto que la mayoría de los usuarios odian registrarse en los sitios web y consideran el proceso como molesto, intrusivo y como pérdida de tiempo. Al simplificar los inicios de sesión y eliminar la necesidad de que los usuarios completen un formulario de registro para cada sitio que visitan, OpenID facilita a que los usuarios de internet, que navegan por distintas páginas, utilicen sitios que recientemente hayan descubierto, pasando más tiempo interactuando con el sitio web y menos tiempo completando las páginas de registro. Asimismo, los administradores de los sitios se beneficiarán también a medida que más visitantes se autentiquen para utilizar sus sitios.

A su vez, OpenID reduce la frustración de los usuarios asociada con el mantenimiento de múltiples nombres de usuario y contraseñas. La mayoría de los usuarios tienen dificultades para recordar múltiples combinaciones de nombres de usuario y contraseñas necesarias para acceder a cada uno de sus sitios web favoritos, y el proceso de recuperación de contraseña puede ser tedioso. A su vez, el uso de la misma contraseña en cada uno de sus sitios web favoritos supone un riesgo de seguridad. Con OpenID, se puede utilizar una cuenta única para acceder a miles de sitios web sin necesidad de crear otro nombre de usuario y contraseña.

Adicionalmente, con OpenID el usuario puede obtener un mayor control sobre su identidad. OpenID es un estándar descentralizado, lo que significa que no está controlado por ningún sitio web o proveedor de servicios. El usuario final es quien controla la información personal que desea compartir con los sitios web que aceptan OpenID.

A diferencia de OAuth, OpenID solo permite autenticación. En general, carece de autorización para acceder a información específica. Por el contrario, OAuth permite el intercambio de información entre los sitios web, cosa que no ocurre utilizando el protocolo OpenID, en este caso, los sitios web no intercambian información de los usuarios entre ellos. Para muchas personas que se preocupan por la privacidad, esto es considerado como un rasgo positivo. Sin embargo, para algunos sitios web, la falta de intercambio de información a través del protocolo OpenID puede ser considerado como un inconveniente.

Si bien no es equivalente a los protocolos de autorización, OpenID potencialmente puede ofrecer acceso a la información pública de los usuarios finales, los cuales ofrecen una gran cantidad de material demográfico que puede ser muy útil para campañas de marketing, u otro uso estadístico.

4.9.2. Desventajas

Controlar y manejar usuarios en forma tradicional para el proceso de autenticación tiene sus beneficios, especialmente en sitios orientados a la comunidad o sitios basados en el diálogo, principalmente porque se tiene un control total sobre los datos que se le solicitan al usuario.

A pesar de contar con un identificador OpenID, aún no hay muchos lugares donde utilizarlo. El proceso de autenticación puede resultar confuso para el usuario, ya que requiere redirección hacia otro sitio, en lugar de realizar todo el proceso de inicio de sesión en un único sitio, como están acostumbrados los usuarios en general, puede que no resulte amigable al usuario.

Por otra parte, los problemas de seguridad no se han resuelto por completo. Al tercerizar la autenticación siempre se pueden presentar ataques de phishing contra el usuario.

Asimismo, se puede considerar una desventaja la pérdida inconsciente de anonimato. Cada sitio web al que el usuario se registra conoce los datos que el usuario completa. Con OpenID, a pesar de que se pueden mantener múltiples identidades, el usuario está atado a una gran cantidad de servicios, lo cual puede resultar en la pérdida de cierto grado de anonimato.

También OpenID puede provocar una pérdida de control, ya que coloca la mayoría de las responsabilidades de seguridad de sitios web en manos de un tercero. Eso no es un problema si todo funciona sin inconvenientes. Sin embargo, si el proveedor de OpenID experimenta algún tipo de problema de seguridad, como los que se han detallado previamente, los sitios web que

utilizan OpenID probablemente se verán afectados, de alguna manera, así como también peligra la privacidad de los datos del usuario.

Por último, OpenID dificulta la recaudación de información. Es sabido que muchas organizaciones dependen de los formularios de registro para que los visitantes proporcionen datos claves, pero con OpenID, la tarea de recopilar datos personales de visitantes se convierte en una tarea mucho más difícil. Por este motivo, en caso de usar OpenID, las organizaciones tendrán que buscar otros incentivos para lograr que los visitantes del sitio web proporcionen datos personales.

4.9.3. En Conclusión

OpenID puede ser una gran ventaja para muchos sitios web, ofreciendo autenticación simple y segura para los usuarios y al mismo reduciendo efectivamente los costos de implementación y mantenimiento.

Mientras que OpenID se está volviendo cada vez más conocido y es un claro líder en el campo de la autenticación, aún está lejos de ser un componente fundamental para un sitio web, o de ser una opción para la autenticación en la mayoría de los sitios web. Esto sucede ya que existen beneficios en la utilización de sistemas de autenticación tradicionales, por eso se aconseja que cada usuario tenga una forma sencilla de registrarse y acceder a un sitio web junto con la posibilidad de utilizar el protocolo OpenID.

Capítulo 5

OAuth

5.1. Introducción

La página web oficial de OAuth define OAuth como: "*un protocolo abierto que permite la autorización segura desde aplicaciones web, móviles y de escritorio de una manera simple y estándar.*"

5.1.1. Análisis de la definición

Es un protocolo ya que es un conjunto de reglas y se dice que es abierto ya que sus características son de libre acceso, tanto a empresas como a usuarios, los cuales pueden obtener la suficiente documentación para su implementación.

Esencialmente, OAuth permite que los usuarios aprueben que una aplicación actúe en su nombre sin compartir su nombre de usuario y contraseña. Por ejemplo, permite que los usuarios compartan con otro sitio sus recursos o datos privados (lista de contactos, documentos, fotos, vídeos, etc.) almacenados en un sitio, sin que los usuarios tengan que informar sus credenciales (generalmente, el nombre de usuario y la contraseña).

En cuanto a lo estándar, un protocolo abierto se convierte en estándar, cuando aparece un organismo normalizador que publica una serie de normas bajo las cuales debe regirse. En el caso de OAuth, el organismo en cuestión es la IETF [60] o Grupo de Trabajo de Ingeniería de Internet (en inglés, Internet Engineering Task Force).

5.1.2. La necesidad de OAuth

La necesidad de un protocolo de autorización como es OAuth suele ser la confianza. O mejor dicho, la falta de ella. Un usuario puede no confiar en una aplicación lo suficiente como para darle sus datos de autenticación pero por otro lado quiere permitir que dicha aplicación realice algo en su nombre. Y no es lo mismo realizar algo en el nombre de alguien, que suplantar a éste alguien. Esto es precisamente lo que permite OAuth: que alguien realice tareas en nombre de otro, pero sin poder suplantarlo: en todo momento se sabe quién es realmente el que está realizando las tareas. Al tener la autorización separada de la autenticación, se puede en cualquier momento desautorizar a quien se haya autorizado previamente. Es decir, impedir que siga realizando tareas en su nombre. Además de que se puede hacer esto sin necesidad de modificar el usuario o la contraseña.

Por otro lado, lo que le ofrece OAuth a un proveedor de servicios es la posibilidad de generar un ecosistema de terceros que usen sus servicios. Un ejemplo podría ser Facebook el cual no expone su API para su propia aplicación móvil, sino que la expone para permitir el desarrollo de un ecosistema de aplicaciones integradas en Facebook, pero realizadas por otros. Tanto Facebook como el creador de la aplicación ganan. El segundo puede aprovechar todo el potencial social que Facebook puede generar, y el primero obtiene más interacciones que es lo que más necesita una red social. Por supuesto que todo esto se podría conseguir sin OAuth, pero en tiempos donde se están mandando tantos (acertados) mensajes contra el phishing y que continuamente se le dice al usuario que vigile sus credenciales, no estaría bien que cualquier sitio web o aplicación pidiese el nombre de usuario y contraseña de Facebook. Además, incluso en el caso de que la aplicación fuese bienintencionada, el hecho de que todas las aplicaciones conectadas a Facebook tengan el nombre de usuario y contraseña de un usuario implica que si alguna vez, por cualquier razón, éste modifica sus credenciales, deba informarle a todas esas aplicaciones de nuevo las nuevas credenciales. Como se verá más adelante OAuth permite obviar todo esto, exponiendo claras ventajas tanto para usuarios como para desarrolladores de servicios.

5.1.3. Algo de Historia

OAuth surgió de las conversaciones entre desarrolladores de Twitter y Ma.gnolia [61] que querían autorizar a las aplicaciones de escritorio para acceder a sus servicios. Un grupo de trabajo se formó en 2007 para redactar una propuesta de estándar abierto. Los desarrolladores de Google y Yahoo! también contribuyeron a este trabajo.

El primer borrador de OAuth 1.0 fue lanzado a finales de 2007. Luego durante la 73ª reunión de la IETF, celebrada en Minnesota, Estados Unidos en 2008, se llevó a cabo un debate para determinar si el OAuth debe gestionarse como un estándar IETF. Una revisión menor (OAuth 1.0 Revisión A) se publicó en junio de 2009 para corregir un agujero de seguridad. Esta revisión no es un estándar de la IETF, pero es estable y bien entendido.

Finalmente en abril de 2010, el estándar de protocolo OAuth 1.0 fue lanzado como RFC 5849 [62].

Varias compañías y servicios de Internet adoptaron OAuth 1.0, pero muchos desarrolladores encontraron al protocolo complicado de entender e implementar por varias razones que describen más adelante en este informe.

En mayo de 2010 se comenzó a trabajar en la versión 2.0 del protocolo OAuth. Esta nueva versión es incompatible con OAuth 1.0 y se centra en la simplicidad para su implementación.

En Octubre de 2012 OAuth 2.0 es publicado como estándar oficial en IETF RFC 6749 [63].

En mayo de 2013, la especificación de OAuth 2.0 ganó el "European Identity & Cloud Award 2013" a la "Mejor Innovación / Nuevo estándar en seguridad de la información".

5.1.4. OAuth 1.0 vs OAuth 2.0

Si alguien se pregunta "*¿Qué hay de malo en OAuth 1.0?*" podría llegar a decirse "¡Nada!, Twitter sigue funcionando perfectamente bien con OAuth 1.0 y sólo comenzó a apoyar una pequeña parte de la especificación 2.0". OAuth 1.0 es una especificación bien pensada y permitió el intercambio seguro de información secreta sin la sobrecarga impuesta por SSL.

La razón por la que se necesitó una renovación y por la cual se comenzó a desarrollar una nueva versión del protocolo se basó principalmente en torno a la complejidad que enfrentan los desarrolladores en la implementación de la especificación.

Las siguientes son algunas áreas donde OAuth 1.0 no impresionó:

- **La firma de cada petición**: que el cliente genere firmas en cada solicitud de API y la validación en el servidor cada vez que se recibe una solicitud, ha demostrado ser una adversidad importante para los desarrolladores, ya que tuvieron que analizar, codificar y clasificar los parámetros antes de realizar un pedido. OAuth 2.0 elimina esta complejidad, simplemente enviando las señales a través de SSL, resolviendo el mismo problema a nivel de red. Así con OAuth 2.0 no se necesitan firmas.

- **Abordar aplicaciones nativas**: Con la evolución de las aplicaciones nativas para dispositivos móviles, el flujo de OAuth 1.0 basado en web parecía ineficaz, obligando al uso de agentes de usuario como un navegador Web. En OAuth 2.0 se agregan más flujos que se adaptan específicamente a las aplicaciones nativas.

- **Clara separación de roles**: OAuth 2.0 proporciona la tan necesaria separación de roles para el *servidor de autorización* autenticando y autorizando al cliente, y para el *servidor de recursos* manejando las llamadas API para acceder a recursos restringidos.

5.2. ¿Cómo funciona?

A continuación se explica cómo funciona el protocolo en su versión 2.0 y los distintos conceptos mencionados en su especificación. Se decidió desarrollar en detalle la versión 2.0, ya que es la última y la más utilizada por la comunidad de desarrolladores.

5.2.1. Entidades y Relaciones

Hay varios actores clave en los flujos del protocolo OAuth:

Servidor de recursos: Es un servidor que aloja recursos de usuarios, y estos están protegidos por OAuth. Suele ser una API que bloquea y protege los datos, como fotos, vídeos, calendarios o contactos.

Propietario de los recursos: Normalmente es el usuario de una aplicación, el propietario del recurso tiene la capacidad de permitir el acceso a sus propios datos alojados en el servidor de recursos.

Cliente: Una aplicación que hace peticiones a la API para realizar acciones en recursos protegidos en nombre del propietario del recurso con su autorización.

Servidor de autorización: El servidor de autorización tiene el consentimiento del propietario del recurso y emite tokens de acceso a los clientes para acceder a los recursos protegidos alojados en un servidor de recursos.

API's más pequeñas pueden usar la misma aplicación y URL para el servidor de autorización y servidor de recursos.

A continuación, la Figura 5.1 muestra el flujo general para la obtención de un recurso:

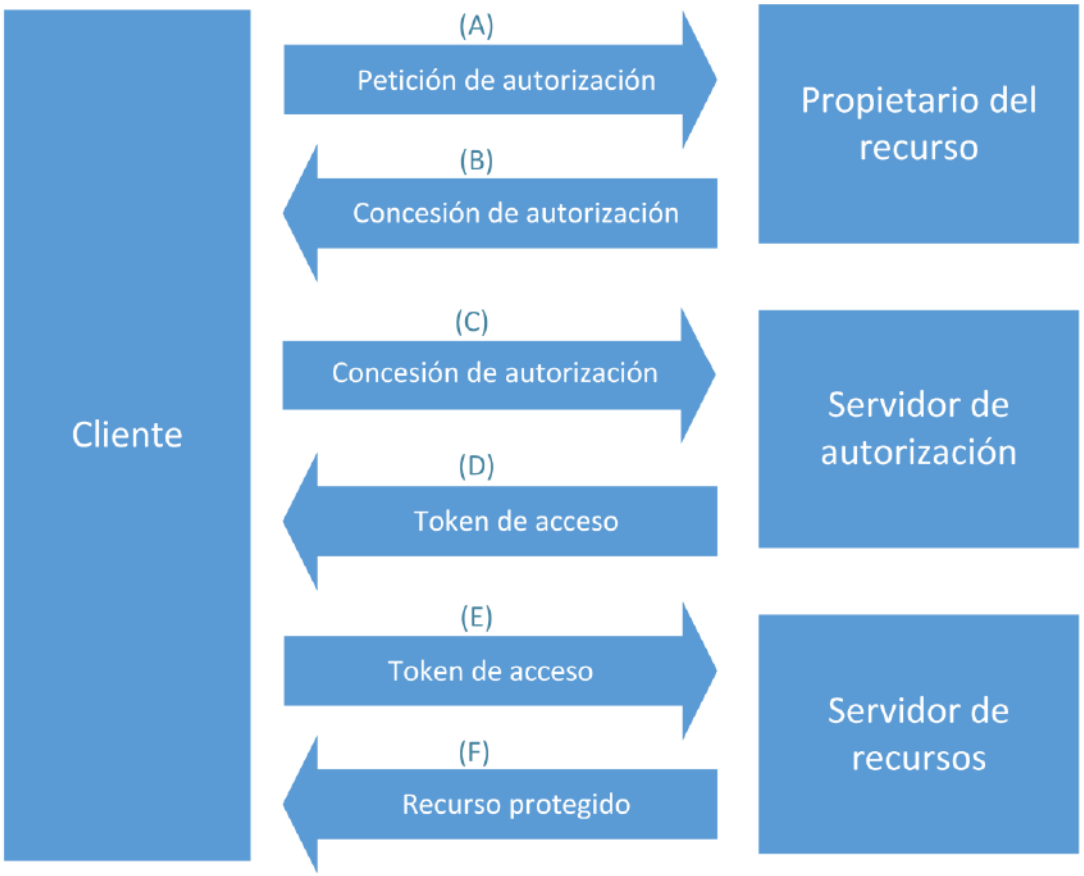

Figura 5.1: Flujo general para la obtención de un recurso.

(A) El cliente solicita la autorización del propietario del recurso. La petición de autorización se puede hacer directamente al propietario del recurso (como se muestra), o de manera indirecta, la cual es preferible, usando como intermediario el Servidor de Autorización.

(B) El cliente recibe una concesión de autorización, que es una credencial que representa la autorización del propietario del recurso, expresado mediante uno de los cuatro tipos de concesión definidos en este protocolo. El tipo de concesión depende del método utilizado por el cliente para solicitar autorización y los tipos soportados por el Servidor de Autorización.

(C) El cliente solicita un token de acceso mediante la autenticación con el Servidor de Autorización presentando la concesión de autorización.

(D) El Servidor de Autorización autoriza al cliente y valida la concesión de autorización. Si es válido, emite un token de acceso.

(E) El cliente solicita un recurso protegido al Servidor de Recursos y se autentica mediante la presentación del token de acceso.

(F) El Servidor de Recursos valida el token de acceso y, si es válido, atiende la solicitud.

5.2.2. Concesión de Autorización (Authorization grant)

Una concesión de autorización no es más que una credencial que representa la autorización dada por el propietario del recurso para que se pueda acceder a sus recursos, la cual va a ser usada por el cliente para obtener un token de acceso.

OAuth 2.0 define 4 tipos diferentes de concesiones de autorización. Para obtener cada uno de ellos existe un mecanismo o flujo definido, es decir, dependiendo del tipo de concesión de autorización, será el flujo que se lleve a cabo. A continuación se describe cada tipo.

- **Código de autorización (Authorization code)**: Es un tipo de concesión de autorización que se utiliza para obtener tanto tokens de acceso como tokens de refresco y está optimizado para clientes confidenciales. Como se trata de un flujo basado en la redirección, el cliente debe ser capaz de interactuar con el agente de usuario del propietario del recurso (típicamente un navegador web) y capaz de recibir solicitudes (a través de la redirección) desde el servidor de autorización.

  El código de autorización se obtiene usando un servidor de autorización como intermediario entre el cliente y el propietario del recurso. En lugar de hacer la petición de autorización directamente al propietario del recurso, el cliente redirige al propietario del recurso al servidor de autorización, que a su vez redirige al propietario del recurso de vuelta al cliente con el código de autorización.

Antes de dirigir al propietario del recurso de vuelta al cliente con el código de autorización, el servidor de autorización autentica al propietario del recurso y obtiene autorización. Las credenciales del propietario del recurso nunca se comparten con el cliente, dado que el propietario del recurso solo se autentica con el servidor de autorización.

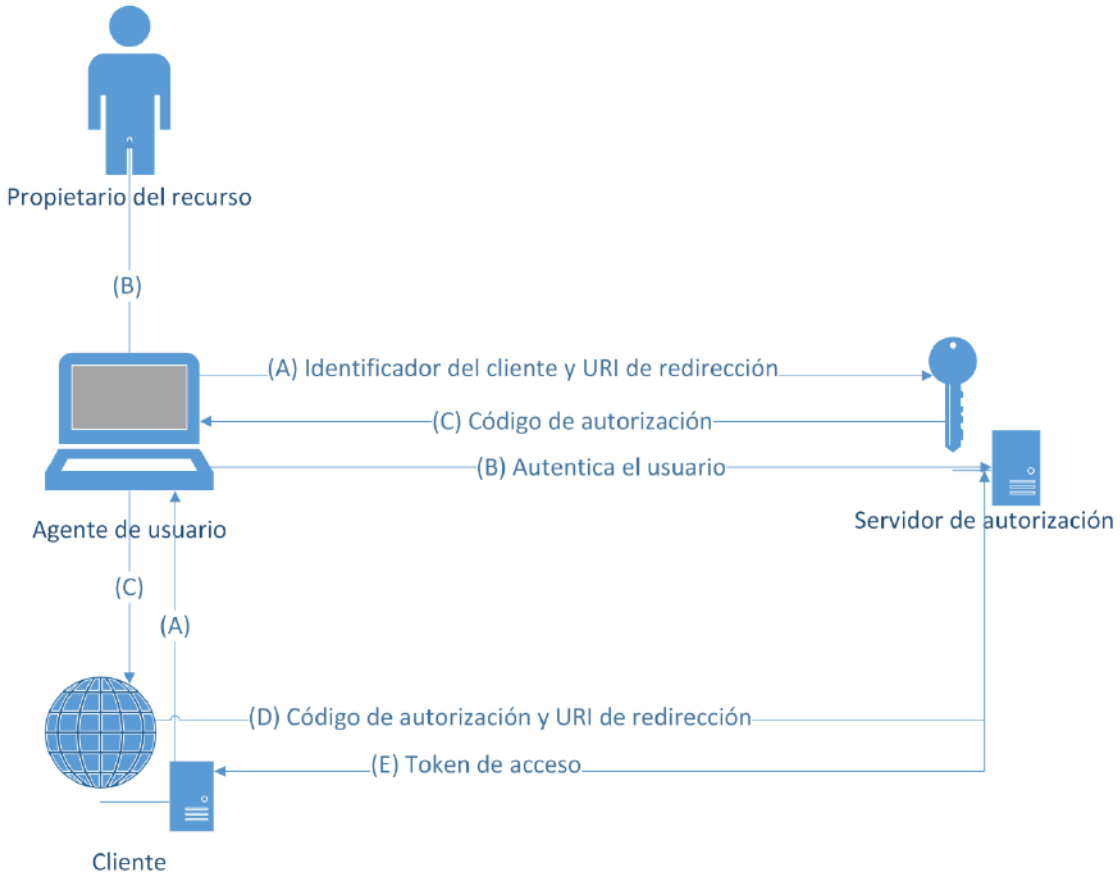

Figura 5.2: Código de autorización.

(A) El cliente inicia el flujo redireccionando el agente de usuario del propietario del recurso al punto de redirección (típicamente un formulario). El cliente introduce su identificador, su ámbito de acceso, estado y una URI de redirección a la que el servidor de autorización mandará al agente de usuario una vez el acceso finalice (satisfactoriamente o erróneamente).

(B) El servidor de autorización autentica al propietario del recurso (a través del agente de usuario) y establece si el propietario del recurso concede o deniega la petición de acceso del cliente.

(C) Asumiendo que el propietario del recurso concede el acceso, el servidor de autorización redirecciona el agente de usuario de vuelta al cliente usando la URI de redirección proporcionada anteriormente (en la petición o durante el registro del cliente). La URI de redirección incluye un código de autorización y cualquier estado local proporcionado por el cliente anteriormente.

(D) El cliente pide un token de acceso al servidor de autorización incluyendo el código de autorización en la petición, obtenido en el paso anterior. Cuando hace la petición, el cliente se autentica con el servidor de autorización. En cliente incluye la URI de redirección usada para obtener el código de autorización, con propósitos de verificación.

(E) El servidor de autorización autentica al cliente, valida el código de autorización y se asegura que la URI de redirección recibida coincide con la usada para redirigir al cliente en el paso.

(C). Si todo es correcto, el servidor de autorización responde con un token de acceso y, opcionalmente, con un token de refresco.

El código de autorización ofrece unos añadidos importantes de seguridad, tales como la capacidad de autenticar al cliente, o la transmisión del token de acceso directamente al cliente sin pasar por el agente de usuario del propietario del recurso y potencialmente exponerlo a otros, incluyendo al propietario del recurso.

- **Implícita (Implicit)**: La concesión implícita o implicit grant es una versión simplificada de la concesión con código de autorización pensada para clientes implementados en un navegador usando un lenguaje del lado del cliente. En este flujo, en lugar de emitir un código de autorización al cliente, se le emite directamente un token de acceso. La concesión es implícita dado que no se emiten credenciales intermedias como el código de autorización.

  Cuando se emite un token de acceso durante la concesión implícita, el servidor de autorización no autentica al cliente. En algunos casos, la identidad del cliente puede ser verificada mediante la URI de redirección usada para entregar el token de acceso al cliente. El token de acceso puede ser que se exponga al propietario del recurso o a otras aplicaciones con acceso al agente de usuario del propietario del recurso.

  La concesión implícita mejora la respuesta y la eficiencia de algunos clientes (por ejemplo los implementados en una aplicación de navegador web), dado que reduce el número de mensajes requeridos para obtener el token de acceso. Sin embargo, esta conveniencia debería ser valorada ante las implicaciones de seguridad que conlleva la concesión implícita, especialmente cuando la concesión con código de autorización está disponible.

- **Contraseña de las credenciales del propietario del recurso (Resource Owner Password Credentials)**: Este tipo de autorización por credenciales (por ejemplo nombre de usuario y contraseña) se puede usar directamente como una concesión de autorización para obtener el token de acceso. Las credenciales solamente deberían ser utilizadas cuando hay un alto grado de confianza entre el propietario del recurso y el cliente (por ejemplo, el cliente es parte del sistema operativo del dispositivo o una aplicación con altos privilegios), y cuando otros tipos de concesiones de autorización no

están disponibles (como la concesión con código de autorización).

Aunque este tipo de concesión requiere acceso directo del cliente a las credenciales del propietario del recurso, estas son usadas para una única petición e intercambiadas por un token de acceso. Esta concesión puede eliminar la necesidad del cliente de almacenar las credenciales del propietario del recurso para un futuro uso, intercambiando las credenciales por un token de acceso de larga duración o un token de refresco.

- **Credenciales de cliente (Client credentials)**: Las credenciales de cliente (u otras formas de autenticación de cliente) se pueden usar como una concesión de autorización cuando el contexto de autorización está limitado a los recursos protegidos que están bajo control del cliente, o a los recursos protegidos previamente acordados con el servidor de autorización. Las credenciales de cliente, típicamente se usan como concesión de autorización cuando el cliente actúa en su propio nombre (el cliente es también el propietario del recurso) o si solicita acceso a recursos protegidos basado en una autorización previamente arreglada con el servidor de autorización.

A parte de estas cuatro formas de conseguir una concesión de autorización, OAuth permite un mecanismo de extensión por el cual se pueden definir nuevos métodos para obtener una concesión de autorización.

5.2.3. Token de acceso (Access token)

Los tokens de acceso son credenciales que se utilizan para acceder a recursos protegidos. Un token de acceso es una cadena de caracteres que representa una autorización emitida al cliente. Los tokens representan alcances y duraciones de acceso específicos, garantizadas por el propietario del recurso, y reforzadas por el servidor de recursos y el servidor de autorización.

El token de acceso proporciona una capa de abstracción, en sustitución de las diferentes formas de autorización (por ejemplo, nombre de usuario y contraseña) con un único token entendido por el servidor de recursos. Esta abstracción permite la emisión de tokens de acceso más restrictivos que la concesión de autorización utilizada para su obtención, así como la eliminación de la necesidad de que el servidor de recursos comprenda una amplia variedad de métodos de autenticación.

5.2.4. Token de refresco (Refresh token)

Los tokens de refresco son credenciales utilizadas para obtener tokens de acceso. Los mismos son emitidos al cliente por el servidor de autorización, y sirven para que una vez caducado el token de acceso, poder obtener un token de acceso nuevo.

La emisión de un token de refresco es opcional a discreción del servidor de autorización. Si el servidor de autorización emite un token de refresco, éste es

incluido a la hora de emitir el token de acceso.

A diferencia de los tokens de acceso, los tokens de refresco están provistos para ser usados sólo con servidores de autorización y nunca se envían a los servidores de recursos.

5.2.5. Registro de clientes

Antes de iniciar el protocolo, el cliente tiene que estar registrado en el servidor de autorización. Los medios por los cuales el cliente se registra en el servidor de autorización están fuera del alcance de la especificación del protocolo pero generalmente implica la interacción del usuario final con un formulario de registro.

El registro de clientes en servidores de autorización no requiere una interacción directa entre el Cliente y el Servidor de Autorización. Si el registro es soportado por el Servidor de Autorización, este puede recurrir a otros medios para establecer la relación de confianza y la obtención de las propiedades requeridas del cliente (URI de redirección, tipo de cliente, concesión de autorización que se va a usar, entre otras).

Cuando se registra un cliente, el propietario o administrador de ese cliente debe:

- Especificar el tipo de cliente
- Ofrecer una URI para la redirección del cliente
- Incluir otro tipo de información que sea requerida por el Servidor de Autorización, por ejemplo: nombre de la aplicación, descripción, el consentimiento a términos legales, etc.

5.2.5.1. Tipos y Perfiles de Cliente

El rol de cliente de OAuth 2.0 se subdivide en un conjunto de tipos de clientes y perfiles.

Tipos

El protocolo define dos tipos de cliente, basados en su habilidad para autenticarse de manera segura con el Servidor de Autorización.

- **Confidencial**: son los clientes capaces de mantener la confidencialidad de sus credenciales, por ejemplo, un cliente implementado en un servidor seguro con acceso restringido a las credenciales del cliente. También puede ser capaz de garantizar la seguridad de las credenciales por otros medios.

- **Público**: son clientes incapaces de mantener la confidencialidad de sus credenciales, por ejemplo, un cliente que se ejecuta en un dispositivo del propietario del recurso, como puede ser una aplicación nativa instalada

en un dispositivo móvil o una aplicación web basada en navegador.

La designación del tipo de cliente se basa en la definición de autenticación segura que se haya implementado en el Servidor de Autorización y en los niveles de exposición de las credenciales del cliente.

Perfiles

La especificación de OAuth 2.0 también menciona un conjunto de perfiles de clientes. Estos perfiles son tipos concretos de aplicaciones, que puede ser confidenciales o públicos.

- **Aplicación Web**: una aplicación web es un cliente confidencial que es ejecutado en un servidor web. Los propietarios de recursos acceden al cliente a través de una interfaz de usuario HTML, representada en un agente de usuario, como por ejemplo un navegador web, en el dispositivo del propietario del recurso. Las credenciales del cliente, así como cualquier token de acceso emitido al cliente son guardados en el servidor web y no son accesibles por el propietario del recurso.

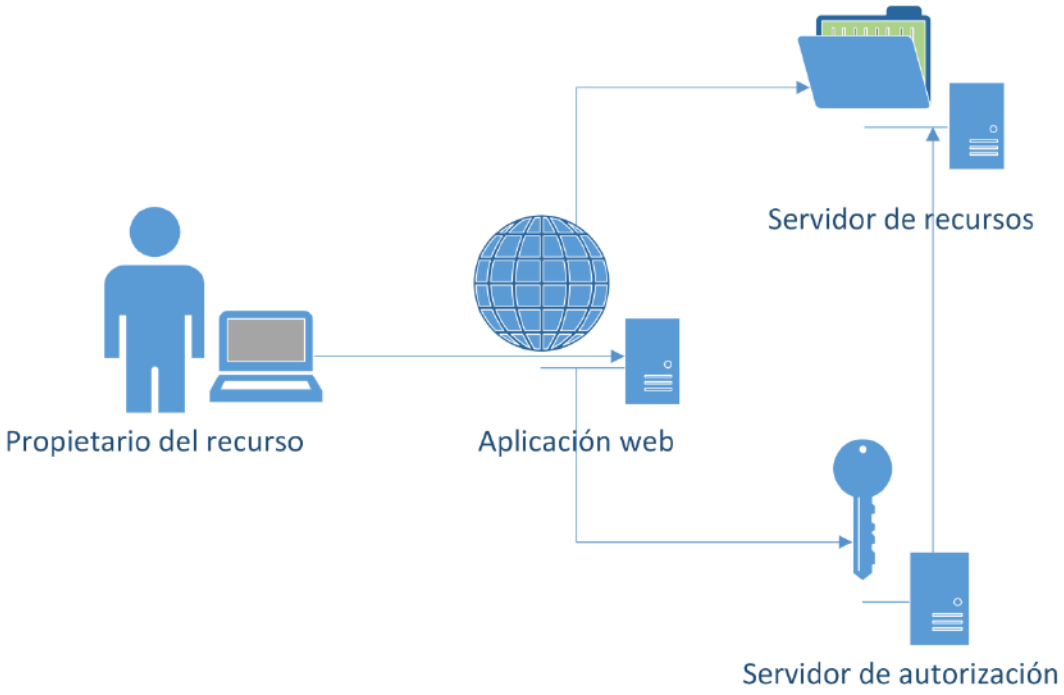

Figura 5.3: Aplicación Web.

- **Aplicación basada en el agente de usuario**: una aplicación de usuario basada en el agente de usuario es un cliente público en el que el código del cliente se descarga de un servidor web y se ejecuta dentro de un agente de usuario en el dispositivo del propietario del recurso. Los datos y credenciales del protocolo son fácilmente accesibles (y usualmente visibles) al propietario del recurso. Dado que estas aplicaciones residen en el agente de usuario del propietario del recurso, pueden hacer un uso transparente de las capacidades del agente de usuario al solicitar la autorización.

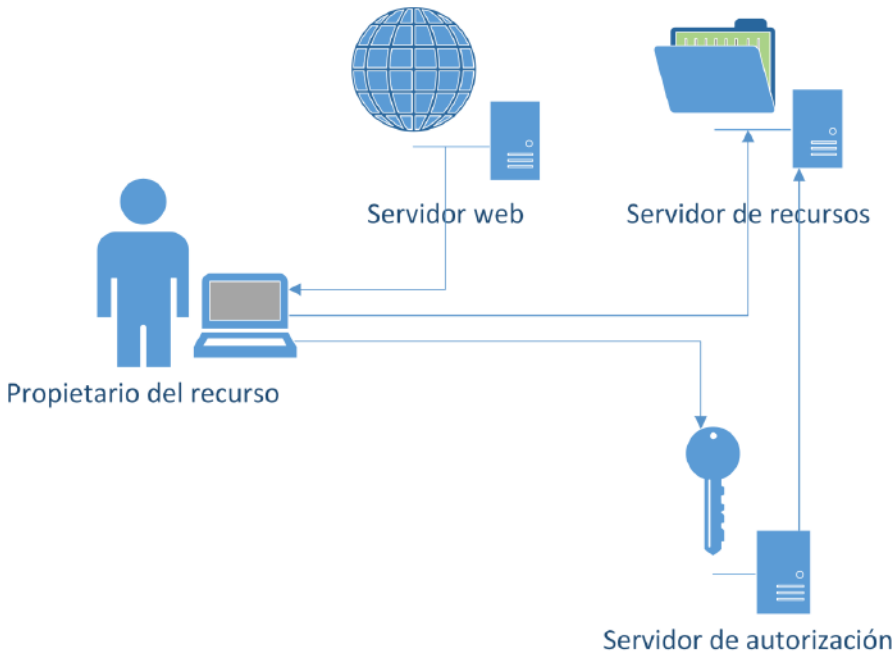

Figura 5.4: Aplicación basada en el agente de usuario.

- **Aplicaciones Nativas**: Una aplicación nativa es un cliente público instalado y ejecutado en el dispositivo del propietario del recurso. Los datos y credenciales del protocolo son accesibles por el propietario del recurso. Se asume que cualquier credencial de autenticación del cliente incluida en la aplicación puede ser extraída. Por otro lado, las credenciales emitidas dinámicamente, tales como el token de acceso, sí tienen un nivel aceptable de protección, garantizando como mínimo que esas credenciales están protegidas de servidores hostiles que puedan interactuar con la aplicación, o de aplicaciones dentro del mismo dispositivo.

Figura 5.5: Aplicaciones nativas.

5.2.5.2. Identificador de cliente

El Servidor de Autorización otorga al cliente, durante el proceso de registro, un identificador de cliente, que no es secreto, está expuesto al propietario del recurso, y no debe ser usado en solitario para la autenticación del cliente.

5.2.5.3. Autenticación del cliente

Si el cliente es de tipo confidencial, el Servidor de Autorización y el Cliente deben establecer un método de autenticación adecuado a los requerimientos de seguridad del Servidor de Autorización. Este puede aceptar cualquier tipo de autenticación de clientes siempre que concuerde con los requerimientos de seguridad.

Los clientes confidenciales suelen establecer una serie de credenciales de cliente que se usan para la autenticación con el Servidor de Autorización.

El Servidor de Autorización no debe de hacer suposiciones sobre el tipo de cliente o aceptar el tipo de información recibida sin haber establecido previamente una relación de confianza con el cliente o su desarrollador. El Servidor de Autorización puede establecer métodos de autenticación con clientes de tipo público. Sin embargo, no debería confiar en autenticaciones de clientes públicos con el propósito de identificar al cliente.

El cliente no puede usar más de un método de autenticación en cada petición.

5.2.5.4. Secreto de cliente

Los Clientes en posesión de un secreto de cliente, deben usar el esquema de autenticación “HTTP Basic” definido en RFC 2627 [64]. Según este esquema, se debe de incluir en la petición del token de acceso al Servidor de Autorización una cabecera HTTP con el siguiente prototipo:

Authorization : Basic <VALOR CIFRADO>

Siendo <VALOR CIFRADO> el valor del identificador del cliente cifrado mediante el método HMAC-SHA256, usando como clave el secreto de cliente.
De forma alternativa, el Servidor de Autorización puede permitir la inclusión de las credenciales del cliente en el cuerpo de la petición, pero está práctica es altamente desaconsejable debido a los problemas de seguridad y suplantación de la identidad de clientes que puede conllevar.

El Servidor de Autorización debe exigir el uso por parte del cliente de mecanismos de seguridad en la capa de transporte (SSL y TLS) al enviar las peticiones de tokens de acceso, ya que sino podrían extraerse de la transmisión las credenciales del cliente en texto en claro. También debe protegerse de ataques de fuerza bruta que traten de obtener el secreto del cliente.

5.2.5.5. Clientes no registrados

La especificación de OAuth 2.0 no excluye el uso de clientes no registrados. Sin embargo, el uso de este tipo de clientes queda fuera del alcance de esta especificación, ya que requiere de análisis de seguridad adicionales y de la revisión de su impacto interoperacional.

5.2.6. Puntos de entrada del protocolo (Protocol Endpoints)

El proceso de autorización utiliza dos puntos de entrada (endpoints) al flujo de ejecución de un proceso de petición de recursos mediante OAuth:

- Punto de entrada para la autorización (Authorization Endpoint)
- Punto de entrada para la obtención de un token (Token Endpoint)

5.2.6.1. Punto de entrada para la autorización (Authorization Endpoint)

Usado para obtener autorización del propietario del recurso a través de una redirección del agente de usuario.

El punto de entrada para la autorización se utiliza para interactuar con el propietario del recurso y obtener una concesión de autorización. El servidor de autorización debe primero verificar la identidad del propietario del recurso. La forma en la que el servidor de autorización autentica el propietario del recurso (por ejemplo, inicio de sesión con nombre de usuario y contraseña) está más allá del alcance de la especificación de OAuth. El medio a través del cual el cliente obtiene la ubicación del punto de entrada para la autorización también está más allá del alcance de la especificación, pero la ubicación está por lo general dentro de la documentación del servicio.

El punto de entrada para la autorización es utilizado tanto en el flujo para obtener una concesión de autorización de manera “Implícita” como a través de un “Código de autorización”. El cliente informa al servidor de autorización del tipo de concesión de autorización que desea obtener enviando el parámetro: response_type con el valor “code” para solicitar un código de autorización o “token” para solicitar un token de acceso (implícita).

5.2.6.2. Punto de entrada para la obtención de un token (Token Endpoint)

Usado para el intercambio de una concesión de autorización por un token de acceso durante la operación de autenticación de un cliente.

Este punto de entrada es usado por el cliente para obtener un token de acceso, presentando para ello una concesión de autorización o un token de refresco. El

punto de entrada para la obtención de un token es usado por todos los tipos de concesiones de autorización, excepto por el tipo "Implícita" (ya que un token de acceso se emite directamente). El medio por el cual el cliente obtiene la ubicación (URL) del punto de entrada para la obtención de un token está fuera del alcance de la especificación de OAuth.

No todas las concesiones de autorización utilizan ambos tipos de puntos de entrada. Los tipos de concesión extendidos (extension grant types) pueden definir puntos de entrada adicionales según necesiten.

5.2.7. Campo de aplicación del token de acceso (Scope)

El campo de aplicación (a partir de ahora se usará el término anglosajón *scope* para referirse a este ámbito de aplicación) del token de acceso es un parámetro que se ha de enviar al punto de entrada para la obtención de un token (también al punto de entrada para la autorización en caso de existir) para permitir que el cliente pueda especificar la amplitud de la petición de acceso, entendiendo por amplitud el número de recursos a los que es capaz de acceder. A su vez, el Servidor de Autorización usa el parámetro scope que incluye en la respuesta al cliente para informarle del alcance del token de acceso emitido.

El valor del parámetro scope se ha de expresar como una lista de cadenas de caracteres separada por espacios en blanco y con diferenciación entre mayúsculas y minúsculas. Estas cadenas han de ser definidas por el Servidor de Autorización. Si el valor contiene múltiples cadenas separadas por espacios en blanco, el orden de estas cadenas no es relevante, y cada cadena añade un rango de acceso adicional al scope requerido.

El Servidor de Autorización puede ignorar total o parcialmente el scope requerido por el cliente, basándose en las políticas implementadas en el Servidor de Autorización.

5.2.8. Obteniendo autorización: solicitudes y respuestas

Cuando una aplicación solicita la autorización del cliente y tokens de acceso, ésta envía peticiones HTTP al servidor de autorización al punto de entrada para la autorización y al punto de entrada para la obtención de un token. Las solicitudes y respuestas dependen del tipo de concesión de autorización. A continuación se detallan las diferentes solicitudes y respuestas para cada flujo correspondiente al tipo de concesión de autorización.

5.2.8.1. Código de autorización

El tipo de concesión de autorización a través de un código de autorización consta de 2 solicitudes y 2 respuestas en total. Una solicitud de autorización y su respuesta y una solicitud de token y su respuesta.

Solicitud de Autorización

La solicitud de autorización se envía al punto de entrada para la autorización para obtener un código de autorización. Los parámetros que se utilizan en la solicitud son:

response_type	*Obligatorio*. En este caso debe ser el texto “code”
client_id	*Obligatorio*. Es el identificador de cliente asignado por el servidor de autorización cuando el cliente se ha registrado.
redirect_uri	*Opcional*. El URI de redirección registrado por el cliente.
scope	*Opcional*. El scope de la solicitud.
state	*Opcional* (recomendado). Es un valor utilizado por el cliente para mantener el estado entre la solicitud y la respuesta. El servidor de autorización incluye este valor al redirigir al agente de usuario de vuelta al cliente. De esta manera el parámetro puede ser utilizado para prevenir ataques de falsificación de solicitudes.

Respuesta Satisfactoria de la solicitud de Autorización

La respuesta de autorización contiene el código de autorización necesario para obtener un token de acceso. Los parámetros incluidos en la respuesta son:

code	*Obligatorio*. El código de autorización.
state	*Obligatorio*, si el parámetro "state" estuvo presente en la solicitud de autorización del cliente. El mismo valor exacto debe ser recibido en la respuesta.

Respuesta de error de la solicitud de autorización

Si se produce un error durante la autorización, dos situaciones pueden ocurrir: la primera es que el cliente no está autenticado o reconocido. Un ejemplo podría ser que el parámetro redirect_uri enviado en la solicitud sea incorrecto. En ese caso, el servidor de autorización no debe redirigir al propietario del recurso a la URI de redireccionamiento. En su lugar, debe informar al propietario del recurso del error.

La segunda situación es que el cliente se autentica correctamente, pero por otro motivo falla la autorización. En este caso, la siguiente respuesta de error se envía al cliente incluida en la redirect_uri:

error	*Obligatorio*. Debe ser uno del conjunto de códigos de error predefinidos por el protocolo. Para más información hay que consultar la especificación.
error_description	*Opcional*. Un código legible por personas, codificado en UTF-8 [65], y que proporciona información adicional sobre el error, proporcionando asistencia al desarrollador del cliente sobre el tipo de error que se ha producido.
error_uri	*Opcional*. Una URI que identifica una página web con información acerca del tipo de error.
state	*Obligatorio*, si el parámetro "state" estuvo presente en la solicitud de autorización del cliente. El mismo valor exacto debe ser recibido en la respuesta.

Solicitud de token

Una vez que se obtiene el código de autorización, el cliente puede utilizar ese código para obtener un token de acceso. Los parámetros de la petición de token de acceso son:

grant_type	*Obligatorio*. En este caso debe ser el texto “authorization_code”.
code	*Obligatorio*. El código de autorización obtenido en la solicitud de autorización.
redirect_uri	*Obligatorio*, si el parámetro redirect_uri fue incluido en la solicitud de autorización. Debe ser idéntico.

Respuesta satisfactoria de la solicitud de token

El Servidor de Autorización emite un token de acceso construyendo la respuesta añadiendo los siguientes parámetros al cuerpo de la respuesta HTTP:

access _token	*Obligatorio*. El token de acceso emitido por el Servidor de Autorización.
token_type	*Obligatorio*. El tipo de token emitido. Los tipos de token de acceso serán estudiados más adelante.
expires_in	*Opcional*. El tiempo de vida en segundos del token de acceso. Por ejemplo, un valor de 3600 en este parámetro indica que el token de acceso expirará en una hora desde que la respuesta fue emitida.
scope	*Opcional*. El campo de aplicación o ámbito del token de acceso.

Estos parámetros han de ser incluidos en el cuerpo de la respuesta HTTP que emite el Servidor de Autorización. Para ello, los parámetros son serializados en una estructura de tipo JSON [66]. El orden de los parámetros no es relevante y puede variar según la respuesta.

El Servidor de Autorización debe incluir las siguientes cabeceras HTTP en cualquier respuesta que contenga un token de acceso, credenciales de cliente u otra información sensible:

```
Cache--Control : no--store
Pragma: no--cache
```

Así, un ejemplo de una respuesta satisfactoria emitida por un Servidor de Autorización tendría el siguiente formato:

```
HTTP/1.1 200 OK
Content--Type : Application/json: charset = UTF--8
Cache-Control: no-store
Pragma: no-cache
{
“access_token” : “3423kjhjkKJH786jHJIUiuohj”,
“token_type” : “ejemplo”,
“expires_in” : “3600”,
”example_parameter” : “valor_de_ejemplo”
}
```

El cliente debe de ignorar los parámetros que no reconoce de la respuesta emitida por el Servidor de Autorización. El tamaño de los tokens de acceso y de otros parámetros emitidos en la respuesta se ha dejado sin definir, ya que es el propio Servidor de Autorización el encargado de definir estos tamaños.

Respuesta de error de la solicitud de token

El *Servidor de Autorización* debe de responder con el código de estatus HTTP 400 (Bad request, petición errónea) incluyendo los siguientes parámetros en la respuesta:

error	*Obligatorio*. Un código de error de entre los siguientes: • invalid_request: La petición carece de algún parámetro requerido, incluyendo valores no soportados de algún parámetro, repite alguno, incluye múltiples credenciales, utiliza más de un mecanismo para autenticar al cliente, o está mal formada en algún sentido. • invalid_client: La autenticación del cliente ha fallado (por ejemplo, porque el cliente no es conocido, o porque el método de autenticación no es soportado por el servidor). El Servidor de Autorización puede devolver un código de estatus HTTP 401 (no autorizado) en vez de HTTP 400 • invalid_grant: La concesión de autorización proporcionada por el cliente es inválido, está expirado, ha sido revocado, no concuerda la redirección URI usada en la petición de autorización o ha sido emitida para un cliente distinto al que hace la petición. • unauthorized_client: El cliente autenticado no está autorizado para utilizar el tipo de concesión de autorización requerida. • unsupported_grant_type: El tipo de concesión de autorización no es soportado por el servidor de autorización. • invalid_scope: El alcance solicitado es inválido, desconocido, está mal formado o excede los límites impuestos por el propietario del recurso.
error_description	*Opcional*. Un código legible por personas, codificado en UTF-8, y que proporciona información adicional sobre el error, proporcionando asistencia al desarrollador del cliente sobre el tipo de error que se ha producido.
error_uri	*Opcional*. Una URI que identifica una página web con información acerca del tipo de error.

Así, un ejemplo de respuesta de error emitida por un *Servidor de Autorización* tendría en siguiente formato:

```
HTTP/1.1 400 Bad Request
Content--Type: application/json: charset=UTF--8
Cache--Control: no--store
Pragma: no--cache
{
“error” : “invalid_request”
}
```

5.2.8.2. Implícita

El tipo de concesión de autorización “Implícita” consta sólo de una solicitud y una respuesta.

Solicitud

Consta de los siguientes parámetros:

response_type	*Obligatorio*. En este caso debe ser el texto “token”.
client_id	*Obligatorio*. Es el identificador de cliente asignado por el servidor de autorización cuando el cliente se ha registrado.
redirect_uri	*Opcional*. El URI de redirección registrado por el cliente.
scope	*Opcional*. El scope de la solicitud.
state	*Opcional* (recomendado). Es un valor utilizado por el cliente para mantener el estado entre la solicitud y la respuesta. El servidor de autorización incluye este valor al redirigir al agente de usuario de vuelta al cliente. De esta manera el parámetro puede ser utilizado para prevenir ataques de falsificación de solicitudes.

Respuesta satisfactoria

La respuesta a la solicitud de concesión de autorización del tipo “implícita” contiene los parámetros descritos a continuación. Hay que tener en cuenta que la respuesta a este tipo de solicitud no es en formato JSON.

access_token	*Obligatorio*. El token de acceso emitido por el Servidor de Autorización.
token_type	*Obligatorio*. El tipo de token emitido. Los tipos de token de acceso serán estudiados más adelante.
expires_in	*Opcional*. El tiempo de vida en segundos del token de acceso. Por ejemplo, un valor de 3600 en este parámetro indica que el token de acceso expirará en una hora desde que la respuesta fue emitida.
scope	*Opcional*. El campo de aplicación o ámbito del token de acceso.
state	*Obligatorio*, si el parámetro "state" estuvo presente en la solicitud de autorización del cliente. El mismo valor exacto debe ser recibido en la respuesta.

Respuesta de error

Si se produce un error durante la autorización, dos situaciones pueden ocurrir: la primera es que el cliente no está autenticado o reconocido. Un ejemplo

podría ser que el parámetro redirect_uri enviado en la solicitud sea incorrecto. En ese caso, el servidor de autorización no debe redirigir al propietario del recurso a la URI de redireccionamiento. En su lugar, debe informar al propietario del recurso del error.

La segunda situación es que el cliente se autentica correctamente, pero por otro motivo falla la autorización. En este caso, se envía al cliente una respuesta de error incluida en la redirect_uri con los siguientes parámetros:

error	*Obligatorio*. Debe ser uno del conjunto de códigos de error predefinidos por el protocolo. Para más información se debe consultar la especificación del protocolo.
error_description	*Opcional*. Un código legible por personas, codificado en UTF-8, y que proporciona información adicional sobre el error, proporcionando asistencia al desarrollador del cliente sobre el tipo de error que se ha producido.
error_uri	*Opcional*. Una URI que identifica una página web con información acerca del tipo de error.
state	*Obligatorio*, si el parámetro "state" estuvo presente en la solicitud de autorización del cliente. El mismo valor exacto debe ser recibido en la respuesta.

5.2.8.3. Contraseña de las credenciales del propietario del recurso

El tipo de concesión de autorización del tipo “Contraseña de las credenciales del propietario del recurso” consta de una solicitud y de su correspondiente respuesta.

Solicitud

Consta de los siguientes parámetros:

grant_type	*Obligatorio*. En este caso debe ser el texto “password”.
username	*Obligatorio*. El nombre de usuario del propietario del recurso, codificado en UTF-8
password	*Obligatorio*. La contraseña del propietario del recurso, codificada en UTF-8
scope	*Opcional*. El scope de la solicitud.

Respuesta

La respuesta es una estructura JSON que contiene el token de acceso. A continuación se detallan todos los parámetros que incluye:

access_token	*Obligatorio*. El token de acceso emitido por el Servidor de Autorización.
token_type	*Obligatorio*. El tipo de token emitido. Los tipos de token de acceso serán estudiados más adelante.
expires_in	*Opcional*. El tiempo de vida en segundos del token de acceso. Por ejemplo, un valor de 3600 en este parámetro indica que el token de acceso expirará en una hora desde que la respuesta fue emitida.
refresh_token	*Opcional*. Contiene un token de refresco que se puede utilizar para obtener nuevos tokens de acceso utilizando la misma concesión de autorización, una vez que el token de acceso devuelto en esta respuesta ya no sea válido.

Un ejemplo de la estructura JSON devuelta se muestra a continuación:

```
{
"access_token"  : "...",
"token_type"    : "...",
"expires_in"    : "...",
"refresh_token" : "..."
}
```

5.2.8.4. Credenciales de cliente

Dado que la autenticación del cliente se utiliza como concesión de autorización, no se necesita ninguna solicitud de autorización adicional.

Solicitud

La solicitud de una concesión de autorización del tipo “credenciales de cliente” contiene los siguientes parámetros:

grant_type	*Obligatorio*. En este caso debe ser el texto “client_credentials”.
scope	*Opcional*. El scope de la solicitud.

Respuesta

La respuesta es una estructura JSON que contiene el token de acceso. A continuación se detallan todos los parámetros que incluye:

access_token	*Obligatorio*. El token de acceso emitido por el Servidor de Autorización.
token_type	*Obligatorio*. El tipo de token emitido. Los tipos de token de acceso serán estudiados más adelante.

expires_in	*Opcional*. El tiempo de vida en segundos del token de acceso. Por ejemplo, un valor de 3600 en este parámetro indica que el token de acceso expirará en una hora desde que la respuesta fue emitida.

Un ejemplo de la estructura JSON devuelta se muestra a continuación:

```
{
"access_token"  : "...",
"token_type"    : "...",
"expires_in"    : "..."
}
```

No se debe incluir un token de actualización en este tipo de solicitud de autorización.

5.2.9. Accediendo a recursos protegidos

El cliente tiene acceso a los recursos protegidos mediante la presentación del token de acceso al servidor de recursos.

El *Servidor de Recursos* debe validar el token recibido y asegurarse de que no está expirado y de que el *scope* cubre al recurso solicitado. El método por el cual el *Servidor de Recurso* valida un token de acceso está fuera del alcance de la especificación de OAuth 2.0, pero normalmente conlleva una interacción entre el *Servidor de Autorización* y el *Servidor de Recursos*.

La forma en la que el cliente utiliza el *token de acceso* para autenticarse con el *Servidor de Recursos* depende del tipo de token emitido por el *Servidor de Autorización*. Normalmente involucra el uso de la cabecera HTTP "Authorization", usando un esquema de autenticación definido para cada tipo de token.

5.2.10. Tipos de token de acceso

El tipo de *token de acceso* proporciona al cliente la información necesaria para utilizarlo de manera satisfactoria a la hora de realizar una petición al *Servidor de Recursos* de un recurso protegido. El cliente no debe de usar *tokens de acceso* si no comprende o confía en el tipo del token proporcionado.

5.2.10.1. Token de portador (Bearer Token)

Un token de portador (a partir de ahora se empleará el término *bearer token*) es un token de seguridad con la propiedad de que cualquier entidad (en nuestro caso esta entidad serían un cliente) en posesión de un token (un portador, *bearer*) puede usarlo de una manera única, es decir, ningún otro *bearer* podrá usar ese mismo token de la misma manera. El uso de *bearer tokens* debe de implicar la utilización de mecanismos de transporte seguro y de cifrado de datos.

Existen tres métodos por los cuales un cliente debe de ser capaz de enviar un *bearer token* en la petición al *Servidor de Recursos* de un *recurso compartido*, no pudiendo usar más de uno en cada petición. Estos tres métodos son:

1. **Campo en la cabecera de la solicitud de autorización (Authorization Request Header Field)**: Cuando se envía el token en el campo "Authorization" de la cabecera HTTP de la petición del recurso, el cliente está haciendo uso del esquema de autenticación Bearer para transmitir el token de acceso. Por ejemplo:

```
GET /recurso/1 HTTP/1.1
Host : ejemplo.com
Authorization : Bearer <TOKEN DE ACCESO>
```

   Donde <TOKEN DE ACCESO> representa el token de acceso obtenido del Servidor de Autorización. Debe de estar codificado en base-64.

   Los Servidores de Recursos están obligados a soportar este método de envío de tokens, siendo el método más recomendado a usar.

2. **Parámetro en el cuerpo (Form-Encoded Body Parameter)**: También es posible enviar el token de acceso en el cuerpo de la petición HTTP. Para ello, habría que añadir el valor del token al parámetro "access_token" del cuerpo de la petición. El cliente no debe de usar este método a menos que se cumplan todas las siguientes condiciones:

   - La petición contiene la cabecera HTTP "Content-Type" : "application/x-www-form-urlencoded".
   - El cuerpo sigue los requerimientos de codificación de la cabecera Content-Type definidos por la W3C para HTML 4.01
   - El cuerpo de la petición HTTP es de una sola parte.
   - El contenido a ser codificado en el cuerpo de la petición debe contener únicamente caracteres ASCII
   - El método *GET* no puede ser usado.

   Un ejemplo del uso de este método está en la siguiente petición HTTP:

```
POST /recurso HTTP/1.1
Host : servidorRecursos.com
Content-Type : application/x-www-form-urlencoded
access_token = sdj76sdj7iYk
```

   Este método no se debe usar, excepto en el contexto de aplicaciones donde el navegador de los participantes no tenga acceso al campo "Authorization" de la cabecera HTTP de la petición.

3. **Parámetro en la URI (URI Query Parameter)**: El último método es enviar el token de acceso directamente en la URI de la petición HTTP, de la siguiente forma:

```
GET /recurso?access_token=d658Uih9Uh HTTP/1.1
Host : servidorRecursos.com
```

Un ejemplo de una URI con un token de acceso incluido sería la siguiente:

https://servidorRecursos.com/recurso?access_token=d658Uih9Uh

Debido a las debilidades de seguridad asociadas con este método, tales como la modificación del token, repetición de tokens o la redirección de la URI destino del token, ya que existe una alta probabilidad de que la URL que contiene el token de acceso sea accesible por atacantes externos. Por ello, este método sólo debe de usarse en el caso de que sea imposible de enviar el token de acceso mediante alguno de otros dos métodos.

5.2.10.2. Token MAC

Utiliza una clave *MAC* junto con el token de acceso. Esta clave se usa para firmar ciertos parámetros de la petición HTTP.

Igual que OAuth 1.0, este tipo de token utiliza la firma en lugar de SSL. La especificación sigue cambiando por lo que se recomienda esperar para utilizar este tipo en una implementación. Una vez que esté más estable, el uso del token MAC va a permitir autorizar usuarios de forma segura y sin encriptar todo el tráfico (esta será una buena opción para las API's que necesitan la seguridad de OAuth, pero manejan solicitudes o respuestas muy grandes donde SSL es ineficiente).

Un ejemplo de una petición que incorporé un token de tipo MAC sería la siguiente:

```
GET /recurso/1 HTTP/1.1
Host : ejemplo.com
Authorization : MAC id = "dfsd796ojLh"
nonce="df8TY876shj"
mac="dda897UYhjksdsa87hjsSCDS="
```

5.2.10.3. Token SAML

La especificación principal de OAuth 2.0 no menciona el uso de una aserción SAML para realizar una autorización. Sin embargo, existe una extensión a la especificación del protocolo que define el uso de aserciones SAML 2.0 para solicitar tokens de acceso, así como la autenticación del cliente. El perfil de aserción OAuth 2.0 es una extensión abstracta a OAuth 2.0 que proporciona un marco para la utilización de aserciones como credenciales de cliente y/o concesiones de autorización. Esto permite ampliar el protocolo OAuth 2.0 con un método que puede escalar sus posibilidades y ofrecer alternativas a los tradicionales métodos de concesiones de autorización.

Perfil de Aserción OAuth 2.0 (OAuth 2.0 Assertion Profile) [67]

OAuth 2.0 provee una especificación general para usar una aserción como credenciales de cliente para obtener una concesión de autorización, o para que la propia aserción sirva como concesión de autorización. La intención de este perfil es mejorar la seguridad usando firmas digitales y métodos criptográficos para garantizar la autenticidad de los datos enviados desde un cliente, además de facilitar la integración de OAuth 2.0 en escenarios cliente-servidor donde el usuario final puede no estar presente. Una aserción no es más que una serie de atributos de usuario, que suelen llegar en forma de XML (SAML) o de cadena de texto (PAPI), y que son emitidas por un proveedor de identidad para que sean consumidas/utilizadas por un proveedor de servicios. Esta especificación es un poco general, sólo da ideas de los mecanismos genéricos para el transporte de las aserciones en la interacción del cliente con el Servidor de Autorización. Actualmente este perfil es demasiado general, por lo que se han creado otros perfiles más específicos. En la siguiente sección se pasará a explicar detalladamente el perfil de aserción SAML 2.0 para OAuth 2.0.

Perfil de Aserción SAML 2.0 para OAuth 2.0 (SAML 2.0 Bearer Assertion Profile for OAuth 2.0) [68]

El protocolo SAML 2.0 se detalló anteriormente en la sección 3.6 Protocolos y servicios, por lo tanto a continuación se definirá como una aserción SAML que puede ser usada para pedir un token de acceso (al Servidor de Autorización), siempre en el contexto de que el cliente quiere hacer uso de la existencia de una relación de confianza (como en entornos federados), expresada a través de la semántica de la aserción SAML, sin necesidad de la autorización expresa del propietario del recurso al servidor de autorización, haciendo que el flujo natural de OAuth se salte el paso de la obtención de la concesión de autorización, ya que, como se comentó anteriormente, la propia aserción SAML se usará como concesión de autorización.

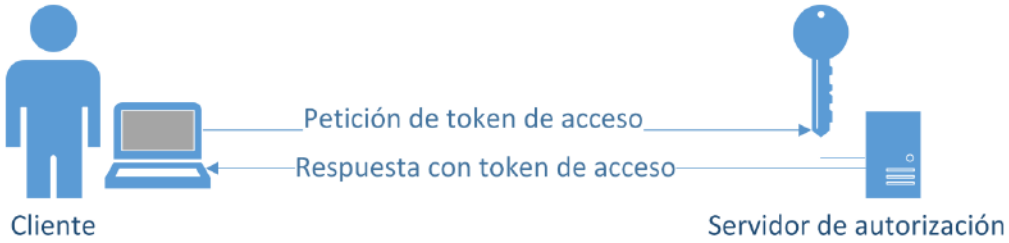

Figura 5.6: Perfil de Aserción SAML 2.0 para OAuth 2.0.

1. El cliente realiza una petición de un *token de acceso* al *Servidor de Autorización* presentando una aserción SAML, la cual actuará como garante de la autorización del propietario del recurso, y unas credenciales de cliente.

2. El *servidor de autorización* verifica la validez de la aserción, enviando de vuelta al cliente un *token de acceso* si no ha habido problemas, o una respuesta de error si la aserción no es válida.

El *servidor de autorización* debe también validar las credenciales del cliente, las

cuales no se deben confundir con las del *propietario del recurso* (las credenciales del propietario del recurso no son conocidas en ningún momento), son las credenciales de la aplicación cliente (normalmente un identificador de cliente y un secreto de cliente). Además, el *servidor de autorización* debería de emitir *tokens de acceso* con un tiempo de vida limitado, que en caso de perder la validez obligará al cliente a solicitar un nuevo *token de acceso* al *servidor de autorización*, presentando la misma aserción (en caso de ser todavía válida), o con una nueva.

5.3. Ventajas y Desventajas

5.3.1 Ventajas

OAuth 2.0 tiene dos grandes ventajas, soluciona el problema de la confianza entre un usuario y aplicaciones de terceros, y a su vez permite a un proveedor de servicios facilitar a aplicaciones de terceros a que amplíen sus servicios con aplicaciones que hacen uso de los datos de sus usuarios de manera segura y dejando al usuario la decisión de cuándo y a quién, revocar o facilitar acceso a sus datos, creando así un ecosistema de aplicaciones alrededor del proveedor de servicios.

La confianza, o la falta de ella, es el motivo principal para querer implementar un protocolo de autorización como OAuth 2.0. La falta de confianza de un usuario con una aplicación de terceros puede propiciar que el usuario quiera permitir a la aplicación realizar tareas y obtener datos en su nombre pero sin darle las credenciales de autenticación a dicha aplicación.

OAuth 2.0 como protocolo de autenticación y autorización es una muy buena opción en cuanto a servicios de provisión de identidad. Sin embargo, se puede pensar que existen alternativas igual de buenas para esta tarea. Un ejemplo de alternativa podría ser el protocolo OpenID.

Ventajas respecto a OpenID

OpenID es un protocolo abierto de autenticación. Cuando se usa para autenticar al usuario en un servicio, este tiene que dar sus credenciales al proveedor de identidad que implementa OpenID para que, de forma similar a OAuth 2.0, este ceda información de la identidad del usuario al consumidor y así proceder a la autenticación en el servicio.

Al ser un protocolo abierto descentralizado, nadie posee el control sobre éste. Cualquier servicio con control de autenticación de usuarios puede implementar OpenID para que sea usado como proveedor de identidad en otro servicio.

Estos puntos que se han mencionado son comunes con OAuth 2.0, sin embargo, con OpenID no se puede ofrecer un acceso a los recursos del proveedor, mientras que con OAuth 2.0, siempre que el usuario lo haya autorizado y el token de acceso tenga validez, una aplicación de terceros podrá acceder a esos recursos. Con OpenID únicamente se ofrece acceso a recursos

de identidad básicos, como pueden ser el nombre o el e-mail del usuario.

5.3.2 Desventajas

Hay algunos cabos sueltos en la especificación, ya que no define adecuadamente unos pocos componentes requeridos o deja que se decida en cada implementación, tales como:

- **Interoperabilidad**: Al agregar demasiados puntos de extensión en la especificación da como resultado implementaciones que no son compatibles entre sí. Esto significa es que no hay que esperar para escribir un código que utilice un punto de entrada genérico el cual detecte los puntos de entrada provistos por las diferentes implementaciones e interactuar con ellos, sino que hay que implementar códigos distintos para Facebook, Google, etc. Incluso la especificación admite este fracaso como una renuncia [69].

- **Poco tiempo de vida de los Tokens**: La especificación no establece cuál debería ser la vida útil y el alcance de las señales emitidas. La aplicación es libre de establecer que un token pueda vivir para siempre. Aunque la mayoría de las implementaciones proporcionan tokens de acceso de corta duración y un token de refresco, el cual es utilizado para obtener un nuevo token de acceso.

.

- **Seguridad**: La especificación sólo "recomienda " el uso de SSL/TLS mientras se envían los tokens en texto plano a través del cable. Sin embargo la mayoría de las implementaciones importantes tiene esto como requisito para comunicarse con los diferentes puntos de entrada, de lo contrario sería demasiado fácil para un atacante espiar la comunicación y descifrar los tokens.

5.3.3 En Conclusión

OAuth 2.0 es sin duda una mejora respecto a su predecesor. OAuth 2.0 proporciona varios tipos de concesiones de autorización nuevos, que pueden ser utilizados para soportar muchos casos de uso como aplicaciones nativas, pero el principal atractivo de esta especificación es su simplicidad respecto a la versión anterior.

A pesar de su significativa actual adopción, existen algunas críticas a OAuth 2.0. Como se detalló anteriormente, dificulta la interoperabilidad proporcionando tantas opciones y alternativas dentro del protocolo. Esta flexibilidad también tiene un impacto negativo en la seguridad: al dejar tantas opciones disponibles, aumenta la probabilidad de implementaciones inseguras.

Concluyendo, se recomienda leer un artículo [70] escrito por Dare Obasanjo, programador de Microsoft, en su blog personal acerca de la renuncia de Eran Hammer-Lahav, unos de los editores más importantes del protocolo OAuth 2.0, y del controversial artículo que éste escribió para dar a conocer su decisión.

Capítulo 6

ESLIP Plugin

6.1. Introducción

Easy **S**ocial **L**ogin **I**ntegration **P**lugin (ESLIP), en español Plugin Simple para Integración de Social Login, permite integrar fácilmente Social Login a un sitio web, tal como lo indica su nombre. El principal objetivo de esta herramienta es permitir a los desarrolladores web incorporar de forma sencilla un widget de inicio de sesión que ofrezca la posibilidad de que los usuarios se autentiquen en un sitio web por medio de las cuentas de redes sociales y servicios en línea más populares, como por ejemplo Facebook, Twitter, Google, entre otros.

ESLIP es un software de código abierto. Por lo tanto los usuarios pueden estudiar, modificar y mejorar su diseño e implementación mediante la disponibilidad de su código fuente, el cual se encuentra alojado en GitHub [71] (https://github.com/eslip/eslip), un entorno gratuito de desarrollo en colaboración para proyectos de software libre, con el fin de que, este plugin siga creciendo por la colaboración de la comunidad.

Si bien existen en la web, diferentes tipos de plugins que brindan Social Login, ESLIP se diferencia del resto por su premisa de sencillez de integración para desarrolladores web. ESLIP está preparado para ser integrado y configurado en simples pasos. Asimismo, cuenta con la gran ventaja de poder fácilmente incorporar proveedores de identidad a la configuración para adaptarse a los cambios repentinos que abundan hoy día en el mundo web.

ESLIP cuenta con un wizard de configuración que guía al desarrollador para que en pocos pasos pueda configurar su widget de Social Login. A su vez cuenta con un módulo de administración desde el cual es posible agregar y quitar proveedores de identidad, habilitar y deshabilitar los mismos del widget, configurar el idioma, entre otras opciones que serán explicadas en profundidad más adelante en este informe.

6.2. ¿Cómo surgió?

Implementar Social Login es una tarea difícil desde el principio, más aún si se planea que lo haga una sola persona. Las grandes redes sociales de todo el mundo ofrecen Social Login y no toma mucho tiempo darse cuenta de que es casi una misión imposible integrar más de dos redes sociales sin asignar una significante cantidad de recursos humanos para la implementación y el mantenimiento de la autenticación.

Cuando comienza la investigación, los desarrolladores empiezan a darse cuenta que muy pocas de las API's, de estos proveedores de identidad, siguen

las mismas normas y que lo que podría haber funcionado ayer cambia con una actualización hoy. Por ejemplo, mientras que Twitter sigue con OAuth 1.0, Google cierra Buzz que se ejecutaba con un protocolo híbrido (OAuth 1.0 combinado con OpenID) para luego lanzar Google+ sobre OAuth 2.0. Esta versión no es la misma que la de OAuth 2.0 que Facebook está utilizando. Y ahí empieza la pesadilla para los desarrolladores.

Una de las principales ventajas de la integración de Social Login es la riqueza de datos puestos a disposición por los proveedores de identidad. Sin embargo, no hay una estandarización en el campo de la estructura y la nomenclatura para facilitar la programación. Cada vez que se quiera añadir un nuevo tipo de datos que ofrece uno o más proveedores de identidad, será necesaria la programación.

Luego de la implementación llega la etapa del mantenimiento, el cual no es un tema menor. En los últimos años, Facebook ha lanzado importantes cambios en su API, Google también, Twitter está detrás de la tecnología por lo que un cambio es inminente, sin olvidarse de Microsoft y otros, que también han hecho cambios y han liberado mejoras importantes en sus API's. Estos grandes cambios requieren que los desarrolladores realicen investigaciones para actualizar su código. Los cambios no son siempre esperados y esto puede paralizar la autenticación de los usuarios hasta que se actualicen los proyectos correspondientes para arreglarlo.

Por otro lado, no hay que olvidar la importancia de los datos de los usuarios. Social Login devuelve a los sitios web grandes cantidades de datos y su uso puede ser un reto cuando se trata de ofrecer acceso mediante múltiples proveedores de identidad, ya que cada uno devuelve diferentes cantidades de datos, en diferentes formatos y con diferentes estructuras.

En términos de análisis, Facebook es el único proveedor de identidad que ofrece cifras interesantes en cuanto al uso de su API. Se puede ver el número de inicios de sesión, las acciones, el alcance, los grupos de edades y el desglose por sexo, de los usuarios y algunos otros detalles. Facebook podría ser el único proveedor que ofrece análisis, pero el inconveniente es que es una perspectiva muy macro, no brinda estadísticas sobre el comportamiento de los usuarios individuales, sólo ofrece tendencias.

Con el tiempo, el desafío será el seguimiento de los análisis de todos los proveedores de identidad mediante los que se ofrece autenticación ya que los datos no estarán disponibles en un lugar central, sino en cada proveedor de identidad.

Para no tener que lidiar con las cuestiones planteadas anteriormente existen soluciones pagas para implementar Social Login en un sitio web, como por ejemplo Gigya y Janrain (nombradas anteriormente en el Capítulo 2). Estos proveedores, ofrecen sencillez, personalización, y análisis de datos, entre otras cosas. Estas características pueden potencialmente ser muy beneficiosas para los sitios web de comercio electrónico, ya que brindan estadísticas y análisis de los datos de los usuarios.

Contratar un proveedor de Social Login pago puede ser una buena decisión, ya que, básicamente resolvería toda la complejidad que conlleva implementar un Social Login propio. Incluso se podría contar con soporte post implementación, dependiendo del tipo de servicio contratado. Pero, como todo, la utilización de servicios externos tiene sus contras. Principalmente, incluir a un tercero, el proveedor, tiene el inconveniente de que se estaría compartiendo cierta información con éste, más que nada, se estarían dejando los datos de los usuarios en manos del proveedor de Social Login. Es una realidad que no todos los desarrolladores se sienten cómodos teniendo un intermediario entre el usuario y su sitio web, más aun sabiendo que este intermediario procesará y analizará la información que pase por sus manos. El hecho de elegir utilizar un proveedor de Social Login depende exclusivamente de la confianza que el desarrollador tenga en él.

Por otro lado, existen soluciones que son gratuitas, de código abierto y en las cuales el manejo de la información queda bajo el control total del desarrollador web, pero tienen la desventaja de ser muy difíciles de integrar, y de poseer escasa documentación. Incluso muchas de ellas están a medio desarrollar o manejan muy pocos proveedores de identidad y no son extensibles.

Por ejemplo GITkit (Google Identity Toolkit) [72] es una buena opción gratuita, pero que aún se encuentra en desarrollo. También podemos contar con HybridAuth uno de los plugins de Social Login de código abierto más completos, que ofrece autenticación por medio de una gran cantidad de proveedores de identidad, pero su configuración no es tan simple como dice ser, es necesario codificar unas cuantas líneas para lograr un alternativa de autenticación íntegramente funcional, y ni hablar si se pretende realizar alguna extensión. Aun así cabe destacar que cuenta con una buena documentación y con continuidad en su desarrollo y evolución, lo cual es muy positivo.

A pesar de todas las opciones que existen para que un desarrollador pueda implementar Social Login en sus sitios web, ninguna de ellas logra satisfacer la necesidad más importante que tiene hoy en día el desarrollador promedio, que la mayoría de los casos es: facilidad y rapidez de integración y de adaptación. Con esta premisa se comienza a pensar en ESLIP, un plugin sencillo, que sea fácil de integrar, que sea él quien guía al usuario en la configuración, que genere código para que el desarrollador lo utilice. Más aún, que se pueda adaptar fácilmente a los nuevos proveedores de identidad que surgen día a día en la vorágine que es internet.

6.3. ¿Por qué utilizarlo?

A nuestro parecer, existe una vasta cantidad de buenas razones para que un desarrollador elija implementar su solución de Social Login mediante ESLIP, pero la principal razón se basa explícitamente en el motivo por el cual se decidió llevar adelante el desarrollo de este plugin, y es la simpleza, de la mano de la facilidad de integración. Partiendo de esta característica, se ramifican los demás beneficios que contrae utilizar ESLIP.

Asimismo, se debe destacar que el plugin apunta a desarrolladores web en general. Dicho de otro modo, para poder integrar ESLIP a un sitio web, no es necesario contar con desarrolladores expertos, basta con conocimientos mínimos de programación web y con seguir las instrucciones que brinda el plugin. De esta manera, siguiendo los pasos que propone ESLIP con atención, en un lapso breve se podrá contar con un widget de Social Login funcional para realizar autenticación de usuarios en un sitio web.

Para poder guiar al desarrollador, ESLIP cuenta con un wizard de configuración, es decir cuenta con una interfaz de usuario que presenta una secuencia de cuadros de diálogo que conducen al usuario a través de una serie de pasos bien definidos. De esta manera, la tarea de configuración, que en la mayoría de los plugins es compleja, resulta ser más fácil de realizar mediante este asistente.

Adicionalmente el plugin cuenta con un módulo de administración, el cual provee una interfaz de usuario para modificar la configuración forma muy sencilla, permitiendo agregar proveedores de identidad, actualizar los datos de los mismos, activar y desactivar los distintos proveedores, entre otras cosas. Cabe destacar que este módulo de administración brinda la posibilidad de agregar nuevos proveedores de identidad, extendiendo así la funcionalidad del widget de Social Login, para ofrecer a los usuarios nuevas opciones de autenticación. El único requisito necesario para poder agregar un proveedor de identidad nuevo es que debe soportar el protocolo OAuth o el protocolo OpenID en cualquiera de sus versiones.

Más aún, para facilitar las cosas, ESLIP cuenta con una lista de proveedores de identidad precargados, listos para ser configurados y activados de forma muy simple, (sólo deben configurarse dos o tres datos dependiendo del proveedor en cuestión). A su vez, esta lista se irá actualizando en razón de la popularidad que vayan adquiriendo los distintos proveedores de identidad que abundan en la web, de manera tal, que el plugin siga creciendo para dar soporte a la mayor cantidad de proveedores posible.

Uno de los factores más importantes a destacar es que con ESLIP, el desarrollador posee control total sobre los datos de los usuarios, es decir, el plugin se encarga de realizar la autenticación del usuario frente al proveedor de identidad seleccionado y luego retorna un objeto con los datos de ese usuario, que fueron solicitados mediante la configuración de dicho proveedor. De esta manera ESLIP no se interpone entre el sitio web y los datos del usuario, sólo verifica que el usuario sea quien dice ser y pone, en manos del sitio web, los datos del usuario que retornó el proveedor de identidad.

Por último, es importante mencionar que esta herramienta está disponible tanto en español como en inglés. Basta con configurar el idioma deseado en el módulo de administración. Pero si se necesita en otro idioma, resulta muy simple extenderlo para tal requerimiento. Por otra parte, se espera que, en un futuro, los desarrolladores que utilicen ESLIP colaboren con el proyecto y, entre otras cosas, compartan las distintas traducciones para así expandir el uso del plugin más allá de las fronteras.

6.4. Implementación

El plugin ESLIP consta de dos partes bien diferenciadas:

- El widget que se muestra en el sitio web con los proveedores de identidad disponibles para que el usuario se identifique. Este widget interactúa con lo que se denomina "core" o motor del plugin, quién es el encargado de llevar a cabo todo el proceso de comunicación con los proveedores de identidad y así obtener la identificación o autenticación del usuario.

- El wizard de configuración y el administrador a los cuales solo pueden acceder los desarrolladores para establecer configuraciones.

Técnicamente hablando, ambas partes del lado del servidor están desarrolladas utilizando el lenguaje PHP [73]. Se decidió utilizar este lenguaje dado que es uno de los lenguajes más populares y más utilizados en el desarrollo de aplicaciones web. Además es un lenguaje multiplataforma, de código abierto y posee una comunidad muy grande de desarrolladores.

Por último, para almacenar los datos configurados por los usuarios de ESLIP, se utiliza un archivo XML. En un principio se consideró almacenar las configuraciones propias de ESLIP y la de los proveedores de identidad en una base de datos, pero finalmente se decidió utilizar un archivo XML por una cuestión de portabilidad y flexibilidad para no obligar al desarrollador a contar con una base de datos en su sitio web.

Hasta aquí se ha presentado un pantallazo general de la arquitectura o distribución de ESLIP, pero a continuación se indaga aún más en el diseño e implementación del mismo.

6.4.1. Widget de Social Login

El widget [74] es el lugar donde el usuario del sitio web puede elegir entre los proveedores de identidad activos para identificarse.

Para realizar la visualización del widget en el sitio, se utilizan el lenguaje JavaScript [75], CSS [76] y un elemento HTML " " en donde se inserta dinámicamente el código del widget generado. Para ello se le pide al desarrollador que incluya en el sitio una hoja de estilos CSS y un archivo JavaScript.

La creación del widget se realiza en el archivo JavaScript. El primer paso es obtener, por medio de una petición Ajax [77] a la API de servicios de ESLIP, los datos necesarios como son por ejemplo los proveedores de identidad activos y las configuraciones de personalización del widget, entre otros. Luego se genera el bloque de código HTML correspondiente al widget, el cual, para finalizar, es incluido en el elemento HTML " " con id "eslip-plugin".

Cabe destacar, que el código HTML generado fue validado satisfactoriamente en el sitio web de la W3C, así como también la hoja de estilos CSS provista. Por lo tanto, la inclusión de ESLIP no altera el estado de validación del sitio o página en cuestión.

Cuando un usuario del sitio, donde se implementa el widget, elige un proveedor de identidad para identificarse, inmediatamente el widget se comunica con el core del plugin y es allí donde se realiza todo el procesamiento.

6.4.2. Core del Plugin

El core del plugin se encuentra del lado del servidor y es el conjunto de scripts y clases PHP que se encargan de llevar a cabo todo el proceso de comunicación con los proveedores de identidad y así obtener la identificación o autenticación del usuario.

A continuación se detallan los archivos que lo componen:

eslip_protocol.php

Define la clase “eslip_protocol”, la cual es la súper clase de los protocolos de autenticación OAuth y OpenID donde se define funcionalidad común a las clases en donde se implementan ambos protocolos.

eslip_oauth.php

Define la clase “eslip_oauth” que implementa el protocolo OAuth para autenticar a un usuario intercambiando mensajes con las API's de los Proveedores de Identidad. Esta clase soporta las versiones 1.0, 1.0a y 2.0 de OAuth.

eslip_openid.php

Define la clase “eslip_openid” que implementa una interfaz para autenticar un usuario a través del protocolo OpenID. La misma soporta tanto las versiones 1.0 como la 2.0 del protocolo.

eslip.php

Se definen las clases “Eslip” y “EslipXMLApi”, y se crean por cada clase una variable que la instancia para ser utilizada cada vez que se incluya este archivo.

La clase Eslip incluye funcionalidad genérica para todo el plugin. Se definen métodos generales del plugin para ser utilizados donde esté disponible un objeto de esta clase. Además se incluye el archivo de lenguaje correspondiente al idioma configurado en el administrador del plugin el cual define las constantes correspondientes para todos los textos que se utilizan en el plugin.

La clase EslipXMLApi implementa una API para manipular el archivo XML de configuración. Pone a disposición una interfaz para obtener y guardar fácilmente información en el archivo XML de configuración llamado config.xml.

eslip_services.php

Define la clase "EslipServices", la cual es la súper clase de las API's de servicios de ESLIP. Aquí se define la funcionalidad común a las API's de servicios.

eslip_frontend_services.php

Define la clase "FrontendServiceApi" que implementa una API de servicios para ser utilizados por el plugin para mostrar el widget de Social Login.

eslip_callback_process.php

Este script se ejecuta luego de realizar el proceso de autenticación. El mismo envía, mediante HTTP POST, los datos obtenidos del proveedor de identidad a la URL configurada por el desarrollador para procesar esos datos, conocida como "callbackUrl" que se explica en detalle más adelante en este informe.

eslip_helper.php

En este archivo se declaran funciones auxiliares que utiliza el plugin.

Figura 6.1: Diagrama de Clases de ESLIP

6.4.3. Wizard de configuración y Administrador

Existe un módulo de administración, donde se pueden configurar los distintos proveedores de identidad y los datos generales del plugin. A su vez, el plugin cuenta con un wizard, el cual debe ejecutarse inicialmente para configurar el plugin por primera vez, además de establecer aquí un usuario y una contraseña para luego acceder al administrador. De esta manera se provee un resguardo para no permitir que usuarios anónimos cambien la configuración de ESLIP. Esto debe estar acompañado por la seguridad de cada servidor donde se utiliza ESLIP, protegiendo el archivo "config.xml" (que se encuentra dentro del directorio ./eslip/), para que no sea reemplazado o modificado externamente por usuarios anónimos.

Para el desarrollo del wizard, en lo que respecta a la programación del lado del cliente, se decidió utilizar la librería JQuery [78] y el framework Bootstrap [79], por la versatilidad y facilidad de uso de ellos, basándonos principalmente en el plugin jQuery Templates [80] para el renderizado de las páginas HTML y el plugin Bootstrap Application Wizard [81] utilizado para implementar el wizard de configuración, elegido por su flexibilidad y su potencial.

En cuanto al desarrollo del módulo de administración, del lado del cliente, al igual que con el wizard, se utilizó la librería JQuery y el framework Bootstrap, complementadas con el plugin jQuery Templates.

Tanto el wizard como el módulo de administración interactúan con el servidor por medio de llamadas Ajax, utilizando servicios PHP expuestos por la clase "BackendServiceApi" definida en el archivo "backend_services.php" (que se encuentra dentro del directorio ./eslip/backend/). Estos servicios básicamente se comunican con la API XML para obtener y actualizar los datos de configuración del usuario. A su vez, junto con los datos de configuración de ESLIP, estos servicios devuelven las etiquetas de texto estático que utilizan estos módulos en el lenguaje correspondiente dependiendo el idioma configurado, de esta manera, la internacionalización de los textos del plugin es resuelta del lado del servidor.

6.5. Proceso de Login y Casos de Uso

6.5.1. Proceso de Login

El proceso de login con el plugin ESLIP se muestra a continuación en la Figura 6.1

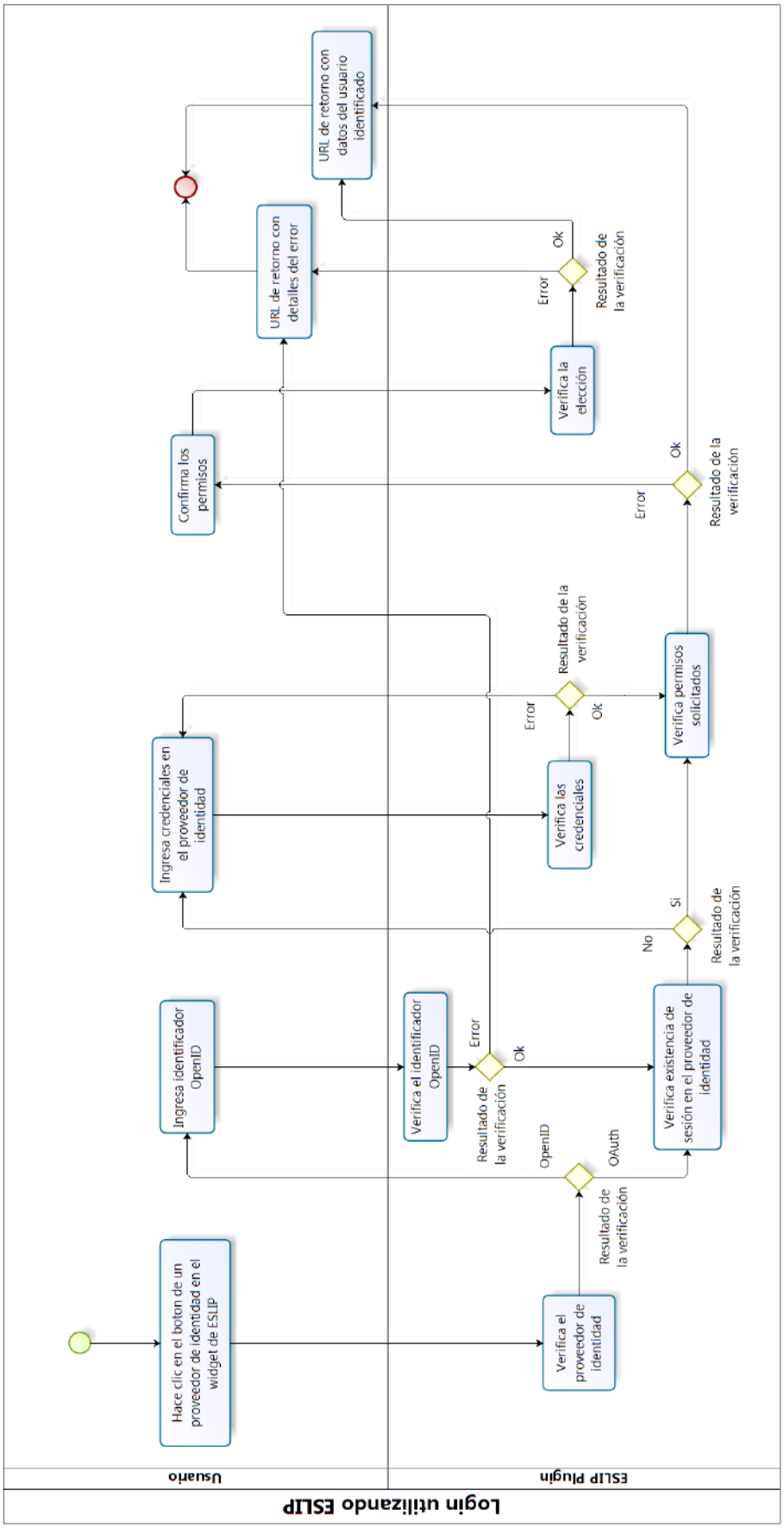

Figura 6.2: Proceso de Login con ESLIP.

Tal como representa la Figura 6.2 durante el proceso de login se llevan a cabo las siguientes tareas:

1. Inicio

   a. Comienzo del proceso de autenticación. Continúa el proceso 2.

2. Solicitud de login mediante un proveedor de identidad

   a. El usuario presiona el botón correspondiente al proveedor de identidad mediante el cual desea identificarse. Continúa el proceso 3.

3. Verificar el tipo de proveedor de identidad

   a. El plugin verifica el proveedor de identidad seleccionado dependiendo el protocolo que utiliza el proveedor elegido (OpenID u OAuth). Continúa la decisión 4.

4. Resultado de la verificación de proveedor de identidad

   a. En caso de tratarse de OpenID se continúa con el proceso 5.
   b. En caso de tratarse de OAuth se continúa con el proceso 8.

5. Ingreso de identificador de OpenID

   a. El usuario ingresa el identificador de OpenID. Se continúa con el proceso 6.

6. Verificar identificador de OpenID

   a. El plugin verifica que el identificador sea correcto. Continúa la decisión 7.

7. Resultado de la verificación de identificador OpenID

   a. En caso de ser errónea la verificación se continúa con el proceso 19.
   b. En caso de ser correcta la verificación se continúa con el proceso 8.

8. Verificar existencia de sesión en el proveedor de identidad

   a. El plugin corrobora la existencia de una sesión ya iniciada en el proveedor de identidad. Continúa la decisión 9.

9. Resultado de la verificación de existencia de sesión
   a. En caso de no existir la sesión se continúa con el proceso 10.
   b. En caso de sí existir la sesión se continúa con el proceso 13.

10. Ingreso de credenciales en el proveedor de identidad

    a. El usuario ingresa las credenciales de identificación correspondientes

al proveedor de identidad. Continúa el proceso 11.

11. Verificar credenciales de autenticación

    a. El plugin verifica los datos de autenticación contra el proveedor de identidad. Continúa la decisión 12.

12. Resultado de la verificación de credenciales

    a. Si la verificación es incorrecta, se continúa con el proceso 10.
    b. Si la verificación es correcta, se continúa con el proceso 13.

13. Verificar permisos solicitados por el proveedor de identidad

    a. El plugin verifica si ya fueron aceptados los permisos que solicita el proveedor de identidad. Continúa la decisión 14.

14. Resultado de la verificación de permisos

    a. En caso de no haber sido aceptados, se continúa con el proceso 15.
    b. En caso de haber sido aceptados, se continúa con el proceso 18.

15. Confirmación de permisos

    a. El usuario responde el pedido de aceptación o rechazo de permisos. Continúa el proceso 16.

16. Verificar aceptación o rechazo

    a. El plugin verifica el resultado de la solicitud de permisos. Continúa la decisión 17.

17. Resultado de la verificación de aceptación de permisos

    a. En caso de no aceptación, se continúa con el proceso 19.
    b. En caso de aceptación, se continúa con el proceso 18.

18. Callback URL con datos del usuario autenticado

    a. El plugin muestra al usuario la callback url con la información del usuario logueado. Continúa el proceso 20.

19. Callback URL con detalles del error

    a. El plugin muestra al usuario la callback url con la información del error. Continúa el proceso 20.

20. Fin

    a. Finaliza el procedimiento.

6.5.2. Casos de Uso

6.5.2.1. Caso de Uso de OAuth 2.0

El diagrama de la Figura 6.3 representa la identificación de un usuario en un sitio web que implementa el plugin ESLIP, utilizando uno de los proveedores de identidad mediante el protocolo OAuth 2.0.

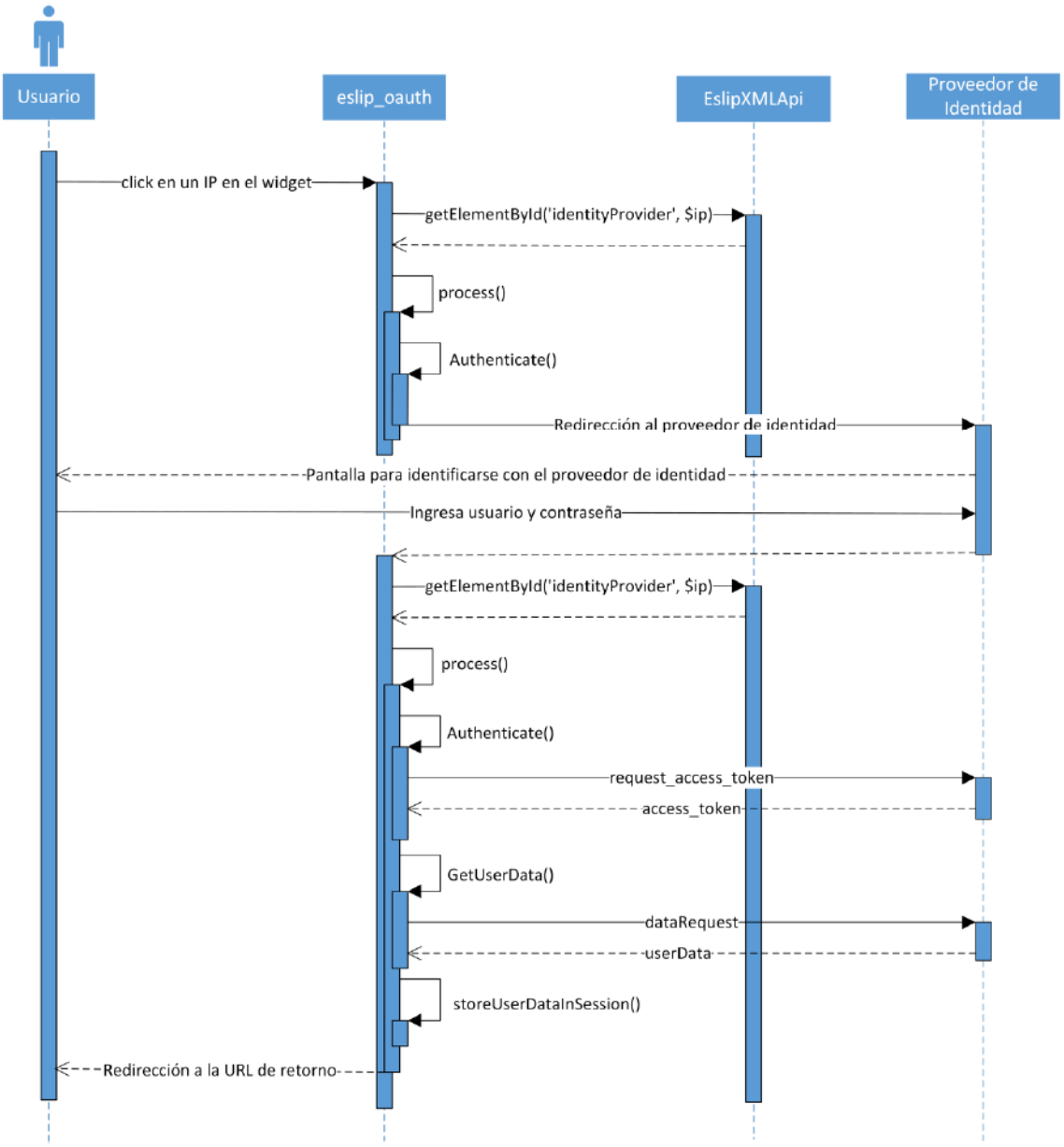

Figura 6.3: Caso de uso OAuth

1. El usuario hace clic en uno de los proveedores de identidad que se encuentran dentro del widget que proporciona el plugin que no sea OpenID.

2. Se abre una ventana emergente, dirigiendo al usuario al archivo eslip_oauth.php dentro del directorio del plugin. Aquí se crean las instancias de

las clases eslip_oauth, la cual implementa el protocolo OAuth para autenticar a un usuario intercambiando mensajes con las API's de los proveedores de identidad, y de la clase EslipXMLApi, la cual implementa una API para manipular el archivo XML de configuración. Al crear una instancia de eslip_oauth lo primero que se realiza es obtener, a través de una petición al objeto EslipXMLApi, los parámetros de configuración del proveedor de identidad elegido para realizar la identificación.

3. Una vez obtenida toda la información necesaria se llama al método process() del objeto eslip_oauth el cual realiza el llamado de todos los métodos necesarios por ESLIP para realizar la identificación y autorización del usuario.

4. El método process() lo primero que realiza es llamar al método authenticate() el cual realiza el procesamiento de la interacción entre el plugin y el servidor OAuth, de acuerdo a la especificación del protocolo OAuth. En este llamado, este método redirige al usuario a la URL de autenticacion y autorizacion. Allí el proveedor de identidad verifica si el usuario está identificado.

5. En caso de no estar identificado se le mostrará la pantalla de inicio de sesión en el proveedor de identidad. Una vez identificado, el proveedor de identidad verificará si la aplicación creada allí por el sitio web ya está autorizada por el usuario para obtener los datos requeridos de su perfil. Si no está autorizada el proveedor de identidad le mostrará al usuario una pantalla en la que deberá autorizar o no a la aplicación. Una vez iniciada la sesión, y autorizada la aplicación del sitio web el usuario es redirigido de vuelta al archivo eslip_oauth.php del plugin.

6. En este paso se realiza el mismo procedimiento que en la anterior llamada: primero se obtienen los parámetros de configuración del proveedor de identidad y luego se llama al método process() y éste al método authenticate().

7. En este nuevo llamado al método authenticate(), al ya estar autenticado el usuario, lo que se realiza es una llamada a la API del proveedor de identidad solicitando un token de acceso. Una vez obtenido el token de acceso finaliza la ejecución del método authenticate() continuando la ejecución del método process().

8. A continuación se llama al método getUserData() del objeto eslip_oauth para obtener los recursos que se quieren del usuario. Este método realiza una llamada a la API del proveedor de identidad solicitando esos recursos utilizando el access token obtenido previamente.

9. Una vez obtenidos los recursos, estos son almacenados en la sesión PHP y posteriormente el plugin cierra la ventana emergente abierta, redirigiendo al usuario a la página configurada por el desarrollador del sitio web donde se procesarán los recursos del usuario obtenidos por el plugin.

6.5.2.2. Caso de Uso de OpenID

El diagrama de la Figura 6.4 representa la identificación de un usuario en un sitio web que implementa el plugin ESLIP, utilizando el protocolo OpenID.

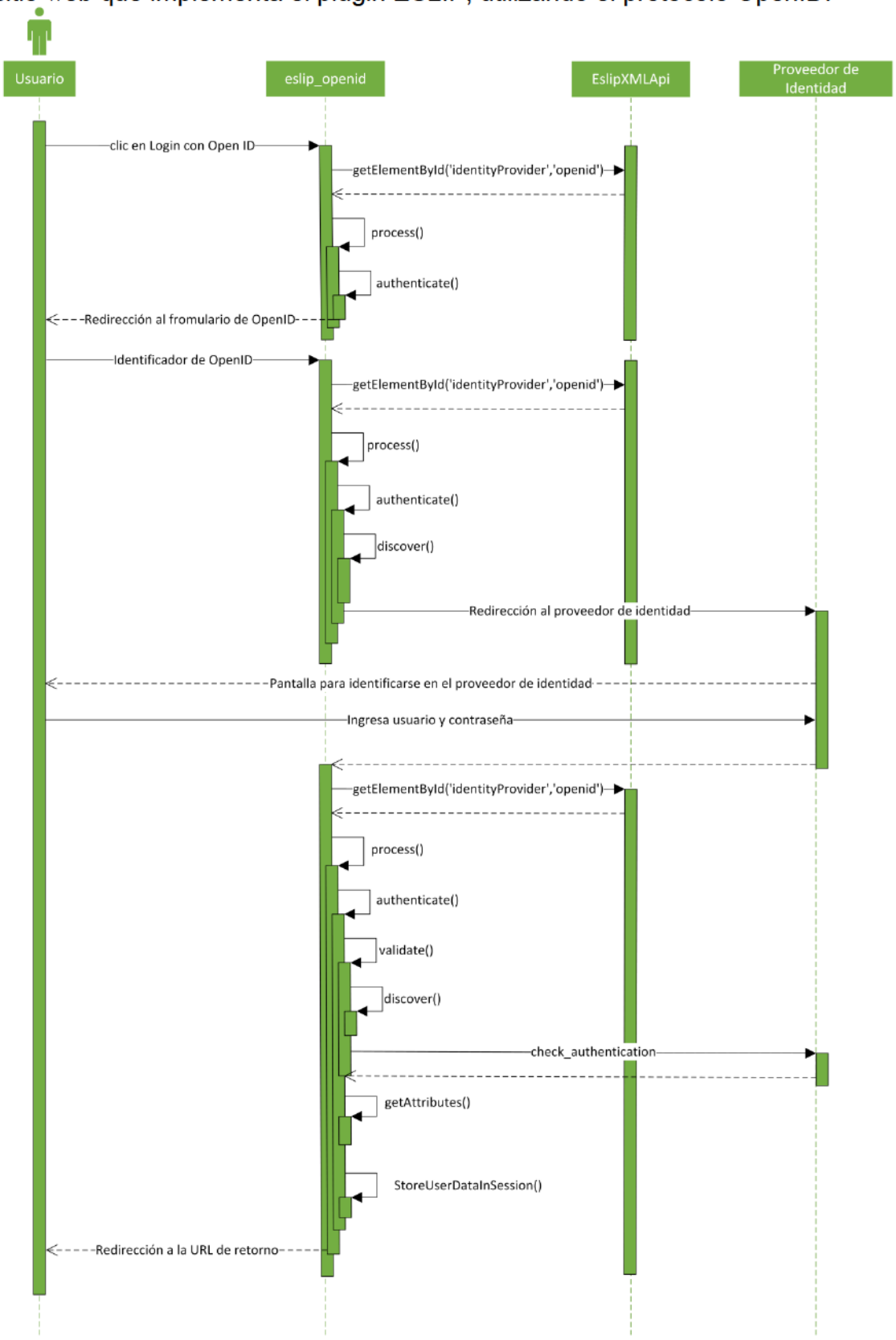

Figura 6.4: Caso de uso OpenID

1. El usuario hace clic en el botón de OpenID dentro del widget que proporciona el plugin.

2. Se abre una ventana emergente, dirigiendo al usuario al archivo eslip_openid.php dentro del directorio del plugin. Aquí se crean las instancias de las clases eslip_openid, la cual implementa el protocolo OpenID, y de la clase EslipXMLApi, la cual implementa una API para manipular el archivo XML de configuración. Al crear una instancia de eslip_openid lo primero que se realiza es obtener, a través de una petición al objeto EslipXMLApi, los parámetros de configuración de OpenID.

3. Una vez obtenida toda la información necesaria se llama al método process() del objeto eslip_openid el cual realiza el llamado de todos los métodos necesarios por ESLIP para realizar la identificación del usuario.

4. El método process() lo primero que realiza es llamar al método authenticate() el cual realiza el procesamiento de la interacción entre el plugin y el servidor OpenID, de acuerdo a la especificación del protocolo OpenID. Este método primero verifica si se recibe como parámetro un identificador OpenID o una respuesta de un proveedor de identidad. Al ser este llamado el primero, ningun parametro es recibido por lo tanto se redirige al usuario al formulario provisto por el plugin para que ingrese su identificador OpenID.

5. Una vez ingresado el identificador, se redirige al usuario de nuevo al archivo eslip_openid.php. Aquí se realiza el mismo procedimiento que en la anterior llamada: primero se obtienen los parámetros de configuración del proveedor de identidad y luego se llama al método process() y éste al método authenticate().

6. En esta llamada, el método authenticate() captura el identificador ingresado por el usuario. Por lo tanto, se realiza el proceso de descubrimiento del proveedor de OpenID mediante el llamado al método discover() . Al completar con éxito el descubrimiento se habrá obtenido la URL del proveedor de OpenID y la versión del protocolo.

7. Con la información obtenida en el descubrimiento, el plugin redirige al usuario a la URL obtenida. Allí el proveedor de identidad verifica si el usuario está identificado. En el caso en el que no esté identificado se le mostrará la pantalla de inicio de sesión en el proveedor de identidad. Una vez identificado, el proveedor de identidad le solicitará al usuario autorización para brindarle información al sitio web en el que se quiere identificar. Una vez iniciada la sesión, y autorizado el proveedor de identidad el usuario es redirigido de vuelta al archivo eslip_openid.php del plugin.

8. Una vez más se realiza el mismo procedimiento que en las llamadas anteriores: primero se obtienen los parámetros de configuración del proveedor de identidad y luego se llama al método process() y éste al método authenticate().

9. En este tercer llamado al método authenticate(), al ya estar identificado el usuario, lo que se realiza es una validación de la autenticación realizada en el proveedor de identidad. Para ello se vuelve a realizar el proceso de descubrimiento y se hace una llamada a la API del proveedor de identidad para que verifique la autenticación. Si la validación es correcta finaliza la ejecución del método authenticate() continuando la ejecución del método process().

10. A continuación se llama al método getAttributes() del objeto eslip_openid para obtener los atributos que se solicitaron del usuario al proveedor de identidad.

11. Una vez obtenidos los recursos, estos son almacenados en la sesión PHP y posteriormente el plugin cierra la ventana emergente abierta, redirigiendo al usuario a la página configurada por el desarrollador del sitio web donde se procesarán los recursos del usuario obtenidos por el plugin.

6.6. Manual de Usuario

En esta sección se describe en forma general como instalar y utilizar el plugin ESLIP. Esta misma información se encuentra disponible en su sitio web oficial http://eslip.com.ar

6.6.1. Descarga del Plugin

Como primer paso se debe descargar la última versión del plugin ESLIP. Éste estará en forma de archivo comprimido si la descarga se realiza desde el sitio web oficial de ESLIP o desde su página de GitHub https://github.com/eslip/eslip. Una vez realizada la descarga, se procede a descomprimir el archivo.

Figura 6.5: Captura de pantalla de la página oficial de ESLIP.

Otra forma de obtener el plugin es directamente mediante GIT [82] realizando la clonación del proyecto. Esto se puede llevar a cabo mediante el comando:

$ git clone git://github.com/eslip/eslip.git

Como resultado, de cualquiera de las alternativas de descarga antes mencionadas, se debería obtener la estructura de archivos representada a continuación por la Figura 6.6

Figura 6.6: Captura de pantalla del directorio de ESLIP, para mostrar la organización de sus archivos.

Dentro de la carpeta ***example/*** se encuentra un ejemplo de cómo funciona el plugin. En la misma se incluye un archivo ***index.php*** en donde se encuentra el código que renderiza el widget de Social Login. Allí también se encuentra un archivo llamado ***login.php*** que es el archivo a donde se retorna una vez realizado todo el procesamiento por parte del plugin y donde se muestra cómo recuperar los datos que fueron obtenidos del proveedor de identidad.

Por otro lado, en la carpeta ***docs/*** se encuentra la documentación del código fuente de los archivos más importantes del plugin. Dicha documentación fue generada con la herramienta phpDocumentor [83]. Esta documentación ofrece una visión en profundidad del proyecto orientada tanto a desarrolladores como a consumidores y contribuyentes.

Por último, en la carpeta ***eslip/*** se encuentra el plugin. Esta es la carpeta que se debe alojar en el sitio web donde se desea integrar el plugin.

6.6.2. Subir el Plugin al Servidor

Como se mencionó anteriormente, una vez descargado el plugin se debe proceder a colocar la carpeta ***eslip/*** (la cual contiene el código del plugin) en el servidor web correspondiente. Se puede alojar en cualquier ruta. Por ejemplo si se aloja en la raíz del sitio web www.example.com la ruta sería www.example.com/eslip.

6.6.3. Configuración del Plugin

Una vez subido el plugin al servidor, ya se encuentra apto para ser configurado. Para ello se debe escribir en la barra de direcciones del navegador la ruta donde fue ubicada la carpeta del plugin y de esta manera se accede al administrador.

Al ser la primera vez que se ingresa al administrador, se debe ejecutar el Wizard de configuración. Esta acción se lleva a cabo haciendo clic en el link del cartel informativo, representado en la Figura 6.7, o accediendo directamente al Wizard por medio la URL ***/eslip/backend/setup.php***.

Figura 6.7: Captura de pantalla del formulario de login para acceder al administrador de ESLIP (primer ingreso).

6.6.4. Wizard de Configuración

6.6.4.1. Selección de lenguaje

Como ya se mencionó, ESLIP se encuentra disponible en inglés y español, aunque es posible extender y agregar más lenguajes. Por ello, en el primer paso del wizard de configuración, se presenta ante el usuario una ventana emergente con la lista de lenguajes soportados por el plugin. En este paso se debe seleccionar el idioma de visualización que será utilizado tanto en el wizard como el administrador.

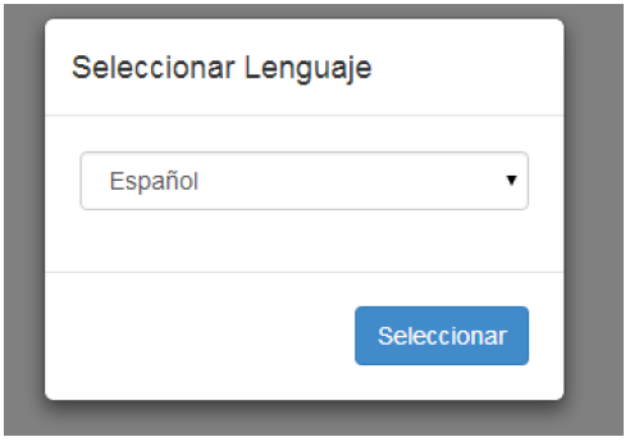

Figura 6.8: Captura de pantalla del selector de idioma del wizard de configuración.

6.6.4.2. Creación del usuario administrador

Se deben establecer las credenciales para acceder luego al administrador de forma segura.

Los datos requeridos son un nombre de usuario y una contraseña. Cabe destacar que el plugin sólo permite crear un único usuario, esto se debe a la búsqueda de sencillez que promueve el plugin.

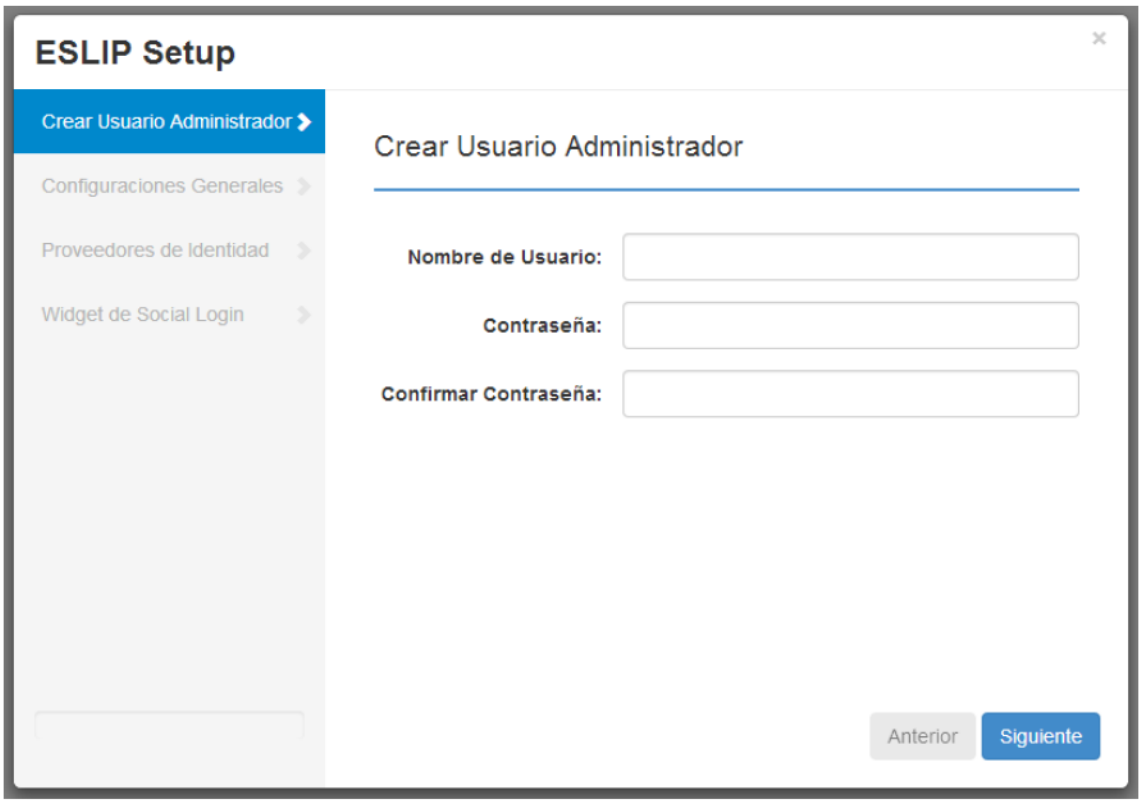

Figura 6.9: Captura de pantalla del paso de creación de usuario del wizard de configuración.

6.6.4.3. Configuraciones generales

En este paso se establecen las URL's que son utilizadas internamente por el plugin para poder realizar correctamente las distintas comunicaciones con los proveedores de identidad. Es de vital importancia que no haya errores en estos datos. Para facilitar la configuración, ESLIP sugiere las URL's en base a la ruta donde se encuentre alojado el plugin, pero por varios motivos puede ser que estas no sean las URL's deseadas.

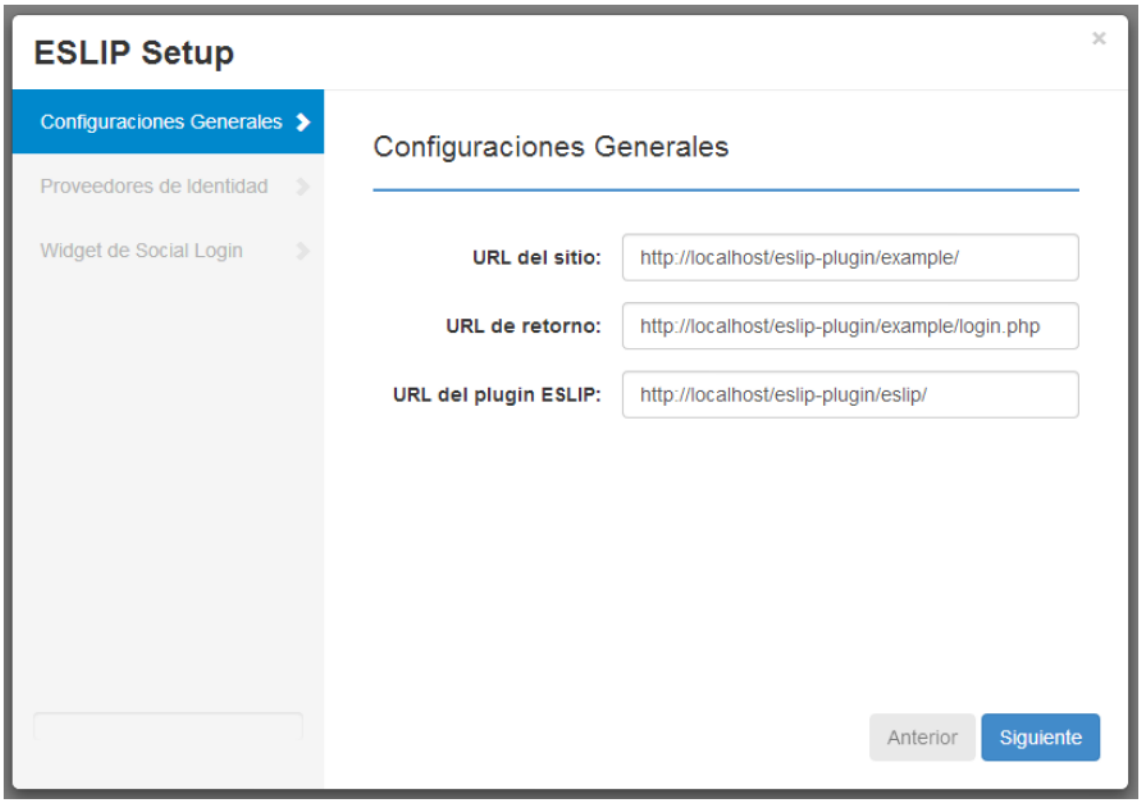

Figura 6.10: Captura de pantalla del paso de configuración general del wizard de configuración.

6.6.4.4. Configuración de los proveedores de identidad

En esta sección se muestran los proveedores de identidad que están dados de alta en el plugin. La primera vez que se ejecuta el wizard, se visualizará la lista de proveedores que ya vienen precargados con el plugin. Inicialmente se encuentra cargada la configuración de los siguientes proveedores de identidad:

- Dropbox
- Facebook
- Flickr
- Foursquare
- Github
- Google
- LinkedIn
- OpenID
- Twitter
- Windows
- Yahoo!

En este paso se deben activar los proveedores de identidad que se desee utilizar como medio de autenticación para el sitio web. De todas formas esto puede modificarse luego desde el módulo de administración del plugin, se podrán activar o desactivar los proveedores según sea necesario.

Con excepción de OpenID, cada proveedor de identidad requerirá para su activación que se cree una aplicación externa, es decir una aplicación propia del proveedor de identidad, para, de esta manera contar con un medio de vinculación entre el sitio web y la API propia del proveedor de identidad.

Estas aplicaciones de los proveedores de identidad exponen una gran cantidad de métodos y propiedades para que diferentes aplicaciones web puedan

establecer un canal de comunicación, el cual puede ser utilizado tanto para autenticación como para recolección de datos y comportamientos de los usuarios. En el caso particular de ESLIP, estas aplicaciones son utilizadas para autenticar, o sea para corroborar que un usuario es quien dice ser, y para recolectar datos del perfil de usuario.

Para la activación de cada proveedor de identidad, exceptuando OpenID, ESLIP solicita los siguientes datos:

- **ID de Cliente**: Identificador provisto por la API OAuth para la aplicación externa creada en el proveedor de identidad correspondiente.
- **Secreto de Cliente**: Clave secreta provista por la API OAuth para la aplicación externa creada en el proveedor de identidad correspondiente.
- **Scopes**: Recursos que se desean obtener y que su propietario, quien se autentica a través del proveedor de identidad, debe autorizar para que sean concedidos.

Para el caso particular del proveedor OpenID, se solicitan los siguientes datos:

- **Scopes Requeridos**: Atributos que se desean requerir al proveedor de identidad. Deben estar separados por coma.
- **Scopes Opcionales**: Atributos que se desean requerir al proveedor de identidad pero que serán opcionales. El proveedor de identidad puede o no retornarlos. Los diferentes scopes que se pueden solicitar pueden ser consultados en http://openid.net/specs/openid-attribute-properties-list-1_0-01.html#Prop_List. De cada scope se debe descartar la primer parte correspondiente a “http://openid.net/schema/” la cual se repite en todos los scopes, y solamente se debe utilizar lo que se encuentra a continuación.

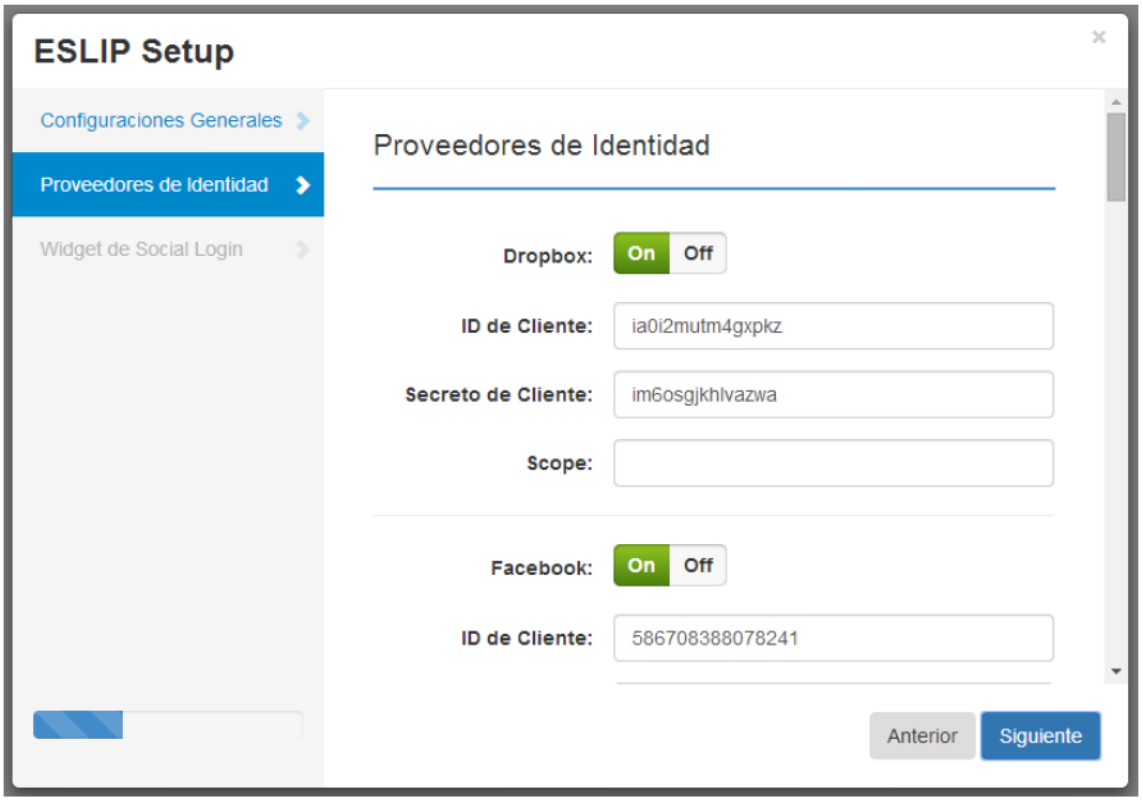

Figura 6.11: Captura de pantalla del paso de configuración de proveedores de identidad del wizard de configuración.

Cabe destacar que para poder completar estos datos, el desarrollador debe recolectar sus valores por fuera del contexto del plugin ESLIP, por medio de aplicaciones que se deben crear en cada proveedor de identidad. Se puede encontrar información detallada sobre dónde y cómo crear estas aplicaciones externas en la documentación presentada en la página web oficial del plugin ESLIP, más específicamente en la URL http://eslip.com.ar/#/documentation.

6.6.4.5. Widget de Social Login

En este paso se configuran los parámetros necesarios para personalizar el widget que será presentado en el sitio web con los botones de login correspondientes a los proveedores de identidad habilitados. Aquí se establece el tamaño del widget, si se debe o no mostrar el texto del botón y la disposición de los botones dentro de él, configurando cantidad de columnas y cantidad de filas.

A su vez, se facilita al desarrollador una vista previa del widget en donde se ven reflejados los valores a medida que éstos van siendo establecidos.

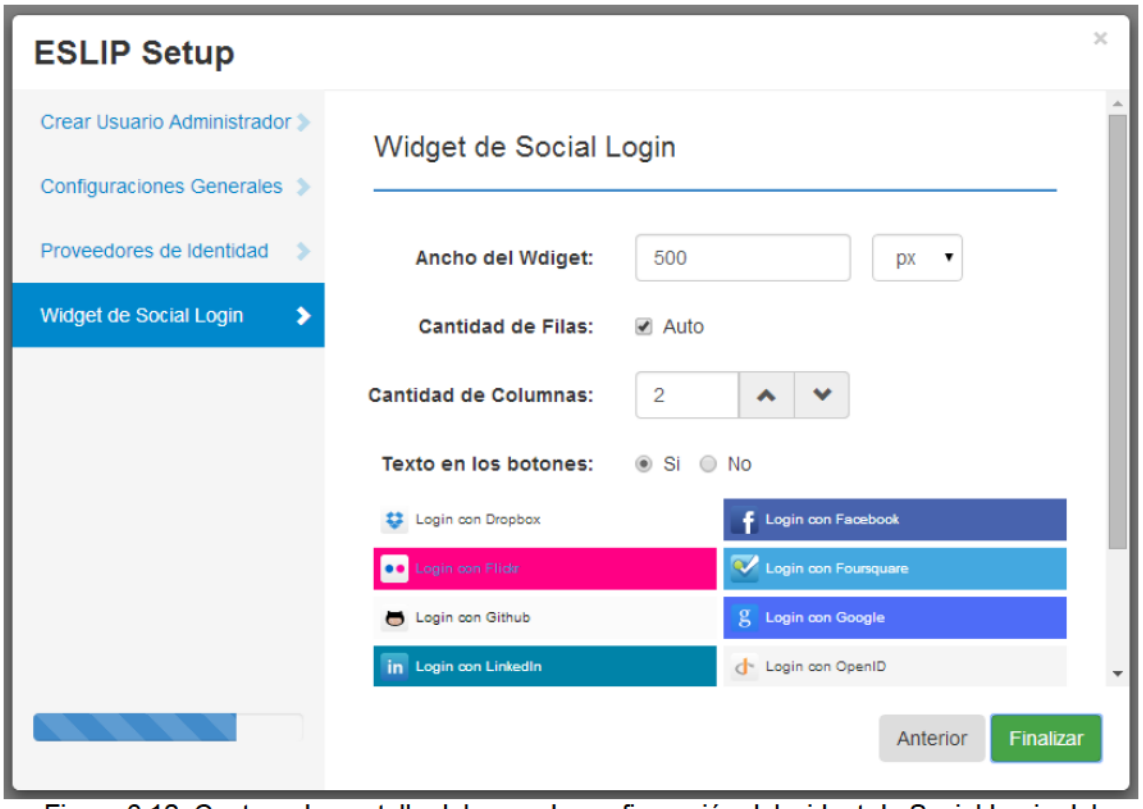

Figura 6.12: Captura de pantalla del paso de configuración del widget de Social Login del wizard de configuración.

6.6.4.6. Código a incluir en el sitio web

Como último paso, luego de realizar las configuraciones necesarias y activar los proveedores de identidad se muestra el bloque de código que debe ser copiado y pegado en las secciones del sitio web donde se desea visualizar el widget se Social Login.

Para una explicación más detallada se podría dividir el código en dos partes. Por un lado se encuentra las referencias a una hoja de estilos y un archivo JavaScript que hacen que se visualice el widget. Se recomienda que las mismas se incluyan en la sección <head> de las páginas HTML donde se desee presentar el widget. Por otro lado, un elemento HTML con el identificador “eslip-plugin” que debe ser incluido en la parte del documento HTML donde se quiere que aparezca el widget. El plugin necesita que el elemento “eslip-plugin” esté presente en la página para poder mostrar el widget.

A continuación, en la Figura 6.14 se puede ver el código que se debe incluir en el sitio web.

Figura 6.13: Captura de pantalla del código HTML del widget de Social Login.

Notar que todas estas configuraciones también están disponibles en el administrador del plugin. El wizard es una herramienta que provee ESLIP para configurar por primera vez el plugin de una manera sencilla, aunque opcionalmente puede ejecutar el wizard cuantas veces quiera, con la salvedad de que la primera vez que se ejecute deberá configurar las credenciales para acceder al módulo de administración.

6.6.5. Módulo de Administración del Plugin

Como se mencionó anteriormente, para poder acceder al administrador, primero se debe ejecutar el wizard de configuración para poder establecer las credenciales de ingreso. Una vez hecho esto, en la pantalla de login del administrador se debe ingresar el nombre de usuario y la contraseña elegida.

Figura 6.14: Captura de pantalla del formulario de login para acceder a los módulos de ESLIP.

Una vez dentro del administrador se pueden observar cinco secciones:

- Configuraciones generales
- Proveedores de identidad
- Configuraciones de usuario
- Configuración de idiomas
- Widget de Social Login

6.6.5.1. Configuraciones generales

Aquí se pueden modificar los valores que se configuraron en la sección con el mismo nombre dentro del wizard. Se establecen las URL's que se van a utilizar internamente en el plugin para poder realizar correctamente las distintas comunicaciones con los proveedores de identidad.

Es de vital importancia que no haya errores en estos datos. Desde esta sección también se puede modificar el lenguaje de la interfaz del plugin y de su administrador.

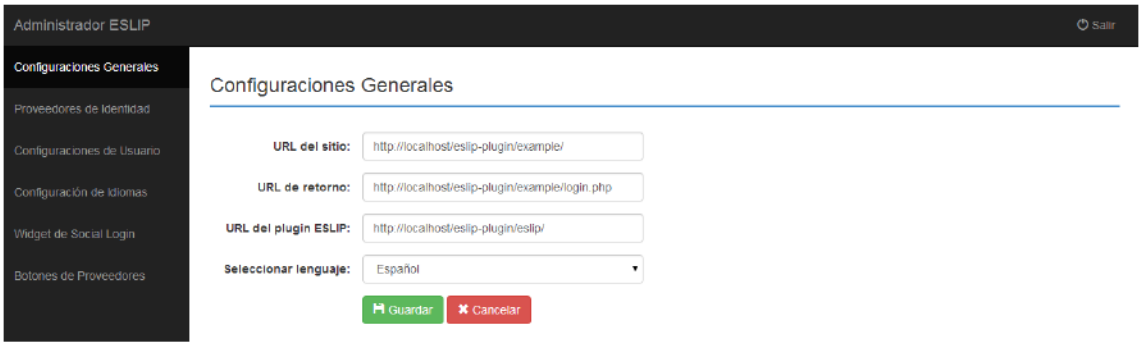

Figura 6.15: Captura de pantalla de la sección de configuraciones generales del módulo de administración.

A continuación se detallan los datos que se pueden configurar en esta sección:

- **URL del Sitio**: URL del sitio web en donde está incluyendo el plugin. Por ejemplo: http://www.example.com

- **URL de retorno**: URL a donde se debería retornar una vez realizada la identificación. Aquí el desarrollador procesará los datos devueltos por el plugin como resultado de su interacción con el proveedor de identidad. Los datos son enviados a esta URL por el método HTTP POST además de ser almacenados en la Sesión PHP. Por ejemplo: http://www.example.com/index.php

- **URL del Plugin ESLIP**: URL donde se encuentra alojado el plugin dentro de su sitio web. Por ejemplo: http://www.example.com/eslip/

- **Lenguaje**: Aquí debe seleccionar el idioma en que quiera que se muestren los textos de la interfaz del plugin y del administrador.

6.6.5.2. Proveedores de identidad

En esta sección se administran los proveedores de identidad de ESLIP. Aquí se puede editar o eliminar los existentes como también agregar nuevos proveedores.

Esta sección se divide en dos vistas. Por un lado, una estándar, en donde se pueden realizar las mismas acciones que en el wizard: activar y desactivar proveedores de identidad y establecer los principales parámetros de cada uno. Y por otro lado, una vista avanzada, en donde por cada proveedor de identidad se da la posibilidad de editar todos sus parámetros o también de eliminarlo del plugin.

Se debe recordar que con excepción de OpenID, cada proveedor de identidad requerirá para su activación que se cree una aplicación externa, es decir una aplicación propia del proveedor de identidad, para, de esta manera contar con un medio de vinculación entre el sitio web y la API propia del proveedor de identidad.

Vista Estándar

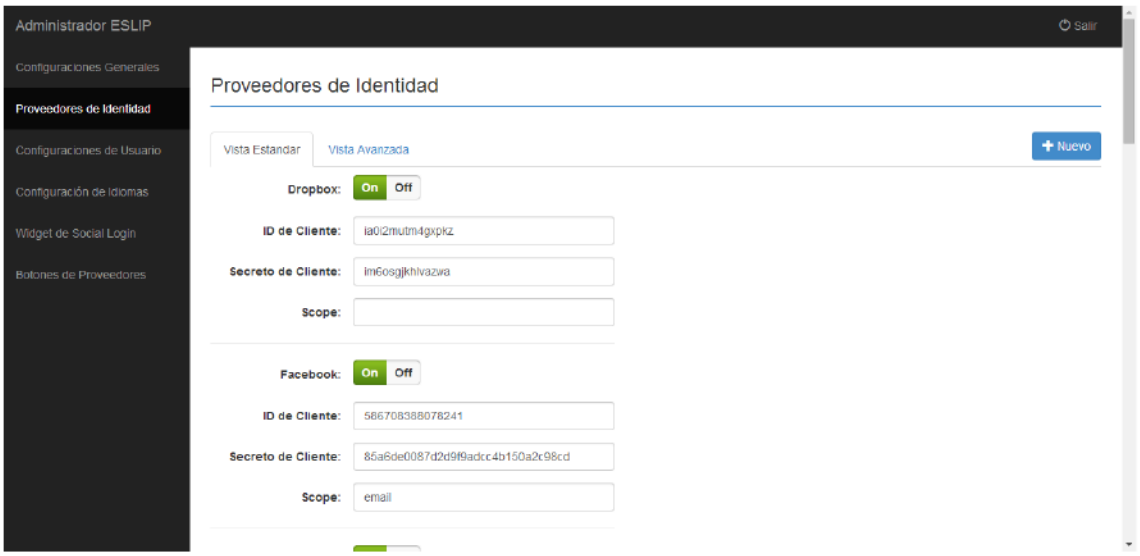

Figura 6.16: Captura de pantalla de la sección de configuración de proveedores de identidad (vista estándar) del módulo de administración.

Vista Avanzada

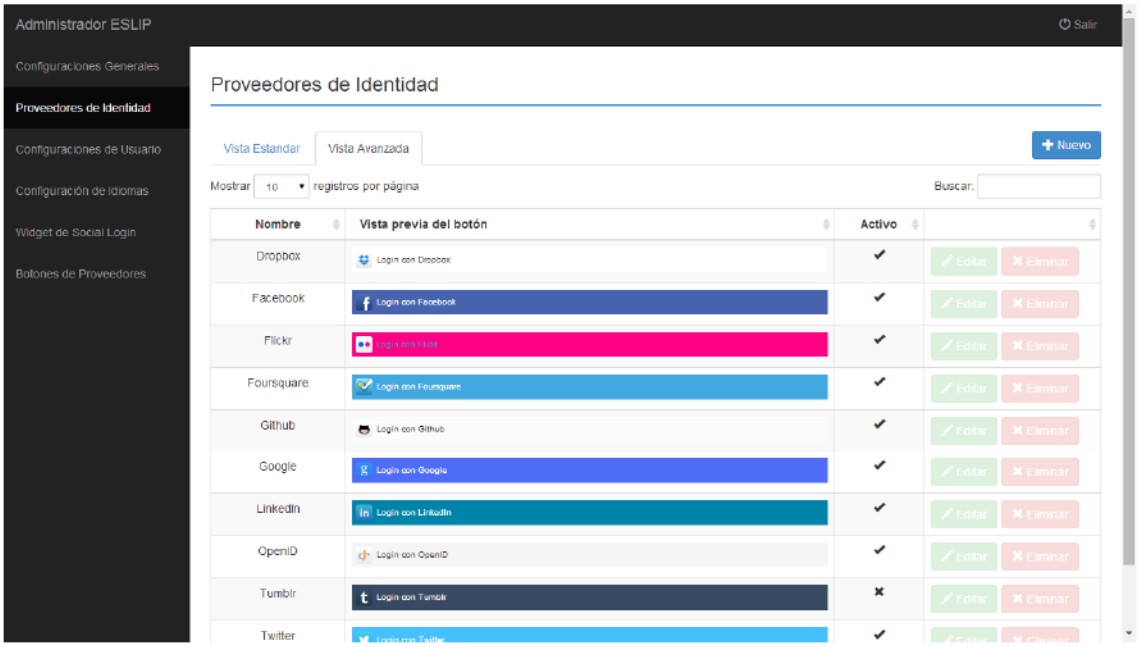

Figura 6.17: Captura de pantalla de la sección de configuración de proveedores de identidad (vista avanzada) del módulo de administración.

A continuación se detalla por cada vista, todos los parametro disponibles que se pueden configurar.

Vista Estándar

Como se dijo recientemente, esta vista es la misma a la presentada en el wizard de configuración inicial.

Para la activación de cada proveedor de identidad, exceptuando OpenID, ESLIP solicita los siguientes datos:

- **ID de Cliente**: Identificador provisto por la API OAuth para la aplicación externa creada en el proveedor de identidad correspondiente.
- **Secreto de Cliente**: Clave secreta provista por la API OAuth para la aplicación externa creada en el proveedor de identidad correspondiente.
- **Scopes**: Recursos que se desean obtener y que su propietario quien se autentica a través del proveedor de identidad debe autorizar para que sean concedidos.

Para el caso particular del proveedor OpenID, se solicitan los siguientes datos:

- **Scopes Requeridos**: Atributos que se desean requerir al proveedor de identidad. Deben estar separados por coma.
- **Scopes Opcionales**: Atributos que se desean requerir al proveedor de identidad pero que serán opcionales. El proveedor de identidad puede o no retornarlos. Los diferentes scopes que se pueden solicitar pueden ser consultados en http://openid.net/specs/openid-attribute-properties-list-1_0-01.html#Prop_List. De cada scope se debe descartar la primer parte correspondiente a "http://openid.net/schema/" la cual se repite en todos los scopes, y solamente se debe utilizar lo que se encuentra a continuación.

Vista Avanzada

Para configurar cada proveedor de identidad, se debe hacer clic en el botón "Editar" correspondiente al proveedor. A continuación se abrirá una ventana emergente con campos para la personalización de los siguientes parámetros:

Parámetros disponibles para todos los proveedores de identidad:

- **Texto del botón**: Texto que aparece en el botón del proveedor de identidad correspondiente dentro del widget.
- **Activo**: Con esta opción puede activar o desactivar este proveedor de identidad para que no sea visible en el widget.
- **Recurso utilizado como identificador único del usuario**: Nombre de la clave en el objeto que retorna el proveedor de identidad como respuesta a la petición de recursos, la cual se utiliza para acceder al identificador único del propietario de los recursos. Si la clave con el nombre ingresado existe en el objeto devuelto por el proveedor de identidad, su valor es enviado a la URL de retorno establecida por el desarrollador en un objeto junto al resto de los recursos obtenidos bajo la clave "id". Este objeto también es almacenado en la sesión PHP.

Parámetros disponibles sólo para OpenID:

- **URL del formulario**: dirección relativa dentro de la carpeta del plugin donde se encuentra el formulario donde el usuario ingresa su identificador OpenID.

- **Autenticación Inmediata**: Aquí se establece si la autenticación se va a realizar utilizando el modo inmediato.

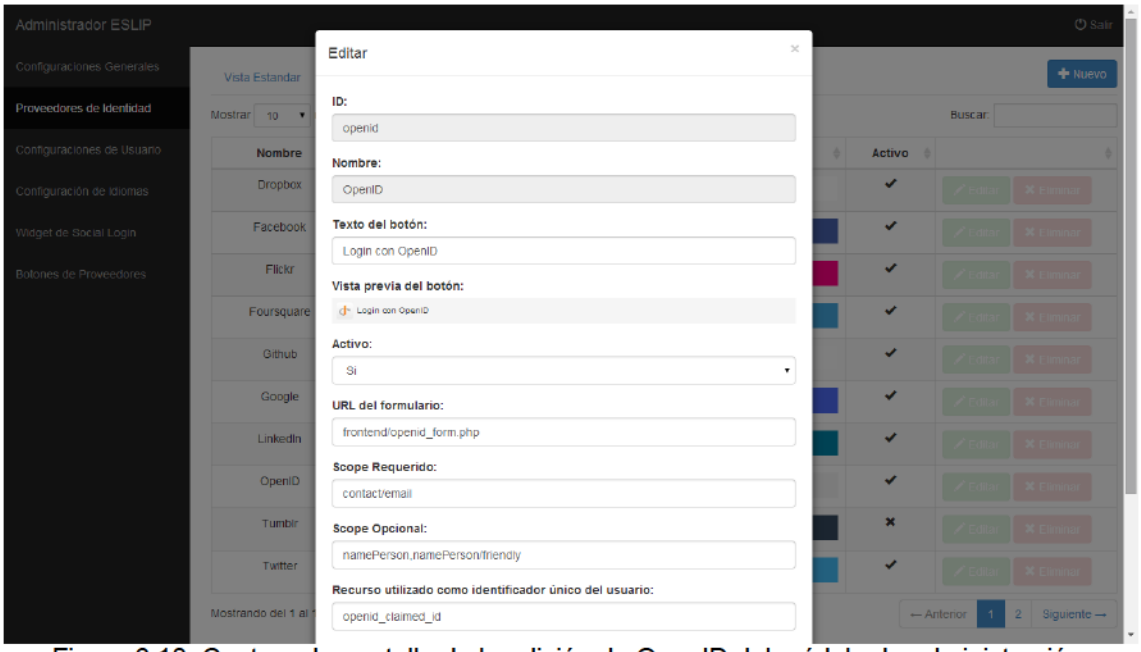

Figura 6.18: Captura de pantalla de la edición de OpenID del módulo de administración.

Parámetros disponibles sólo para proveedores de identidad que utilizan el protocolo OAuth:

- **ID**: Identificador único para cada proveedor de identidad. ESLIP ofrece una lista de identificadores precargados. El plugin provee un estilo de botón junto con el correspondiente icono para cada identificador.

- **Nombre**: Nombre para identificar el proveedor de identidad dentro del administrador del plugin.

- **Versión de OAuth**: Versión del protocolo que soporta el servidor OAuth del proveedor de identidad.

- **URL del punto de entrada para la solicitud del token inicial**: URL del servidor OAuth del proveedor de identidad para solicitar el token inicial cuando trabajamos con servidores OAuth 1.0 y 1.0a.

- **URL del punto de entrada para la autorización**: URL del formulario de login del proveedor de identidad. Se redirigirá al propietario del recurso a dicha URL para que inicie sesión y autorice o no a la aplicación creada en dicho proveedor de identidad a acceder a los recursos solicitados.

- **URL del punto de entrada para la obtención del token de acceso**: URL del servidor OAuth del proveedor de identidad que retornará el token de acceso.

- **Parámetros para la obtención de recursos del usuario**: Cadena con los parámetros que se le envían a la API del proveedor de identidad, entre los cuales debe estar el token de acceso.

- **Parámetros en la cabecera de la petición**: Establece si la API del proveedor de identidad requiere, a la hora de realizar la petición del token inicial, que se le envíen los parámetros en la cabecera de la petición HTTP. Si no se está seguro de ello dejar la opción en "Si".

- **Parámetros en la URL**: Establece si la API del proveedor de identidad requiere, a la hora de realizar la petición del token inicial, que se le envíen los parámetros en la URL. Si no se está seguro de ello dejar la opción en "Si".

- **Parámetro adicional en la respuesta a la petición del token de acceso**: Señala si la API OAuth del proveedor de identidad devuelve un parámetro extra en la respuesta a la petición de token de acceso necesario para la autenticación.

- **Nombre del parámetro adicional**: Nombre de la llave dentro de la respuesta a la petición de token de acceso para acceder al parámetro extra en caso de que el proveedor de identidad devuelva uno. Un ejemplo es Yahoo!, el cual devuelve el parámetro "xoauth_yahoo_guid" necesario para obtener los recursos solicitados.

- **URL del punto de entrada para la obtención de recursos del usuario**: URL que proporciona la API OAuth del proveedor de identidad a la cual se le realiza la petición para obtener los recursos requeridos.

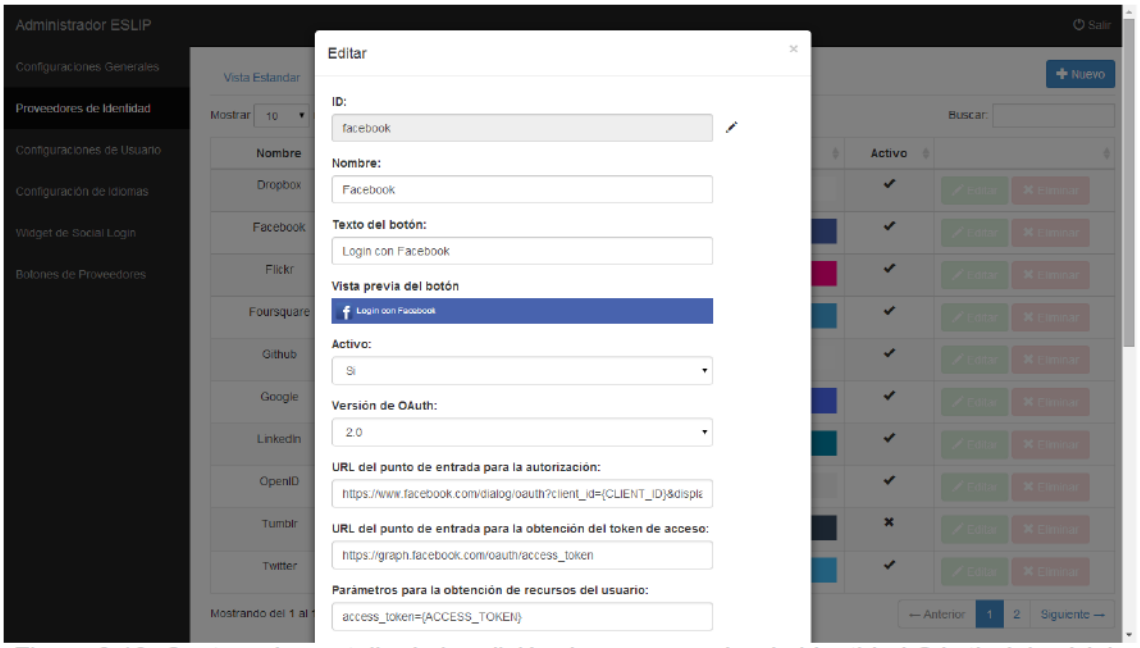

Figura 6.19: Captura de pantalla de la edición de un proveedor de identidad OAuth del módulo de administración.

6.6.5.3. Configuraciones de usuario

Desde aquí se pueden modificar las credenciales para iniciar sesión en el administrador del plugin. Es decir, se pueden modificar el nombre de usuario y la contraseña que fueron establecidas desde el wizard de configuración.

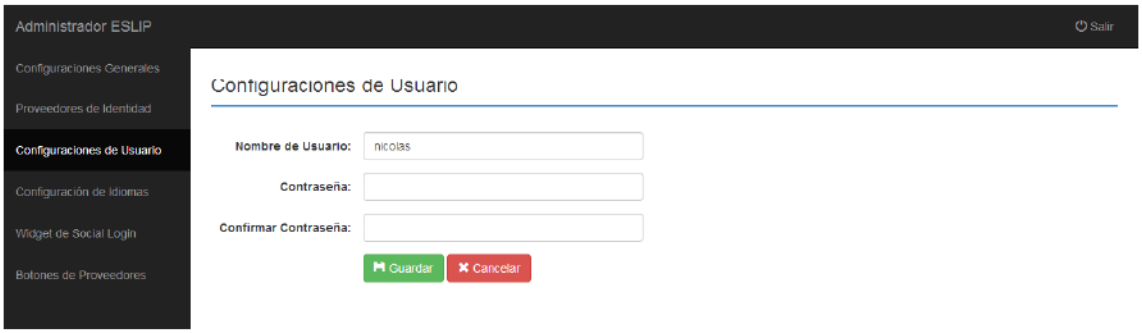

Figura 6.20: Captura de pantalla de la sección de configuración de usuario del módulo de administración.

6.6.5.4. Configuración de idiomas

Figura 6.21: Captura de pantalla de la sección de lenguajes del módulo de administración.

En esta sección se pueden agregar nuevos lenguajes para la visualización de textos del plugin. Para realizar esta acción, primero se debe descargar la plantilla o template. Cada línea de dicha plantilla se compone de la siguiente manera:

etiqueta = “texto”

```
[ WIZARD ]

;step titles
CreateAdminUser = "Crear Usuario Administrador"
GeneralConfigs = "Configuraciones Generales"
IdProviders = "Proveedores de Identidad"

;labels
AdminUser = "Nombre de Usuario"
AdminPass = "Contrase&ntilde;a"
AdminPassConfirm = "Confirmar Contrase&ntilde;a"

;texts
LoginFormDesc = "Debe copiar el siguiente c&oacute;digo en su p&aacute;gina de Login"

;buttons
next = "Siguiente"
previous = "Anterior"
cancel = "Cancelar"
finish = "Finalizar"

;dialogs
SelectLangTitle = "Seleccionar Lenguaje"
WizardEndTitle = "Configuración Finalizada"
WizardEndSubTitle = "Gracias por usar ESLIP Plugin"

[ ADMIN ]

;title
adminTitle = "Administrador ESLIP"
Login = "Iniciar Sesi&oacute;n"

;menu
```

Figura 6.22: Captura de pantalla de la plantilla para crear un idioma.

Se deben traducir los textos de cada etiqueta al nuevo idioma que se desea incorporar, sin alterar los nombres de las etiquetas y guardar el archivo con el correspondiente código de lenguaje ISO de dos letras en minúsculas [84].

Por ejemplo si se traducen los textos a ruso el archivo debe nombrarse “ru”. Una vez que se tiene el archivo traducido y guardado con el correspondiente nombre se lo debe subir al servidor desde esta misma sección y completar el

campo con el nombre del nuevo idioma. Siguiendo con el ejemplo, se debería completar el campo con “Ruso”.

6.6.5.5. Widget de Social Login

En esta sección se configuran los parámetros necesarios para personalizar el widget que será presentado en el sitio web con los botones de login correspondientes a los proveedores de identidad habilitados. Aquí se establece el tamaño del widget y la disposición de los botones dentro de él, entre otras opciones. A su vez, se facilita al desarrollador una vista previa del widget en donde se ven reflejados los valores a medida que éstos van siendo establecidos. A continuación se describen los parámetros disponibles para la personalización:

- **Ancho del widget**: Ancho que tendrá el widget. Se puede establecer en pixeles o en porcentaje dependiendo la necesidad de cada desarrollador.

- **Cantidad de columnas**: Cantidad de columnas de botones que se mostrarán en el widget. Es importante el valor que se establece aquí, ya que a partir de ésta cantidad se calcula el ancho de cada botón que se muestra en el widget.

- **Cantidad de filas**: Cantidad de filas de botones que se mostrarán en el widget. Si se opta por la opción “auto”, todos los botones se mostrarán en forma vertical obedeciendo la cantidad de columnas fijadas sin importar la cantidad de filas utilizadas. Mientras que, si se establece un valor mayor o igual a uno y la cantidad de botones a mostrar excede la capacidad del widget, acorde a la cantidad de columnas y filas configuradas, se generará una especie de galería con paneles horizontales deslizantes, en donde por cada panel se exhibirá el número de botones correspondientes.

- **Texto en los botones**: Define si se mostrará en los botones solo el icono del correspondiente proveedor de identidad o se mostrará junto a él un texto descriptivo.

Por otro lado, en esta sección también se encuentra el código HTML que se debe incluir en el sitio web para visualizar el widget de Social Login. El mismo se corresponde con el que se muestra en el último paso del wizard configuración.

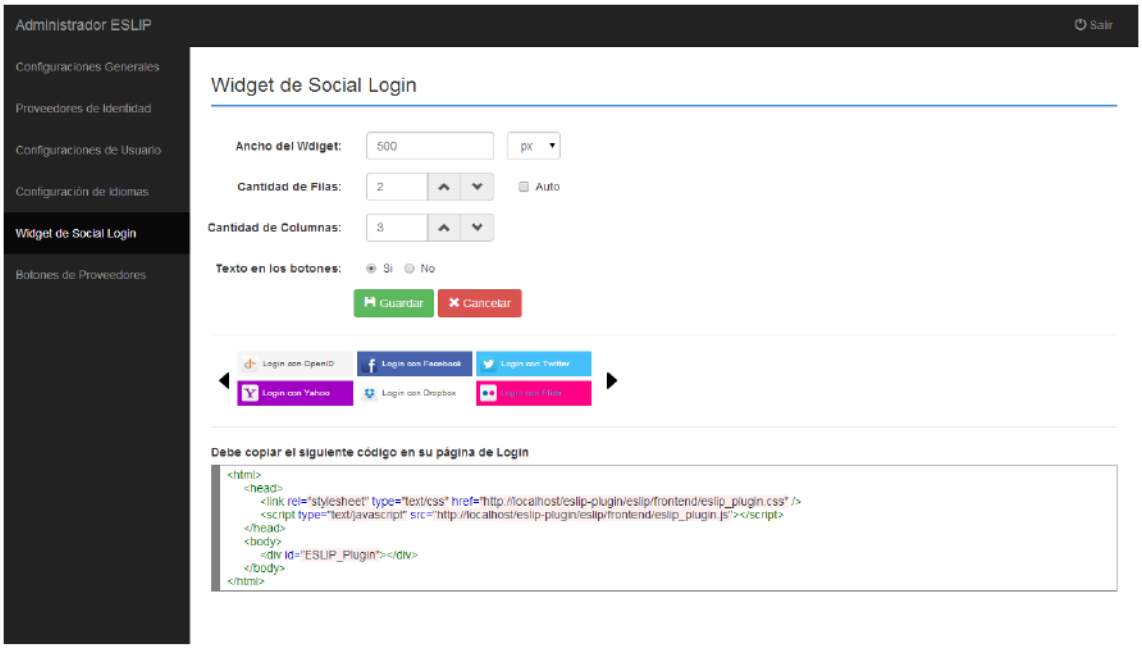

Figura 6.23: Captura de pantalla de la sección de configuración del widget de Social Login del módulo de administración.

6.6.5.6. Botones de proveedores

En esta sección se pueden administrar, es decir, editar y borrar, todos los identificadores existentes en el plugin, así como también agregar nuevos.

Cada identificador corresponde a un proveedor de identidad y está compuesto por un ID que identifica al proveedor de identidad (el cual debe ser único), su logo, y los colores de texto y de fondo que se utilizarán a la hora de mostrar su botón en el widget.

Tanto cuando se desea editar un identificador como cuando se desea crear uno nuevo, se abre un pequeño formulario con los campos detallados a continuación:

- **ID**: Debe ser único y se utilizará para identificar cada una de los proveedores de identidad.
- **Logo**: Logo del proveedor de identidad que está asociado a este identificador.
- **Color de texto**: Color que será utilizado en el texto dentro del botón del proveedor de identidad que se muestra en el widget cuando éste está activo.
- **Color de fondo**: Color que será utilizado como fondo del botón del proveedor de identidad que se muestra en el widget cuando éste está activo.

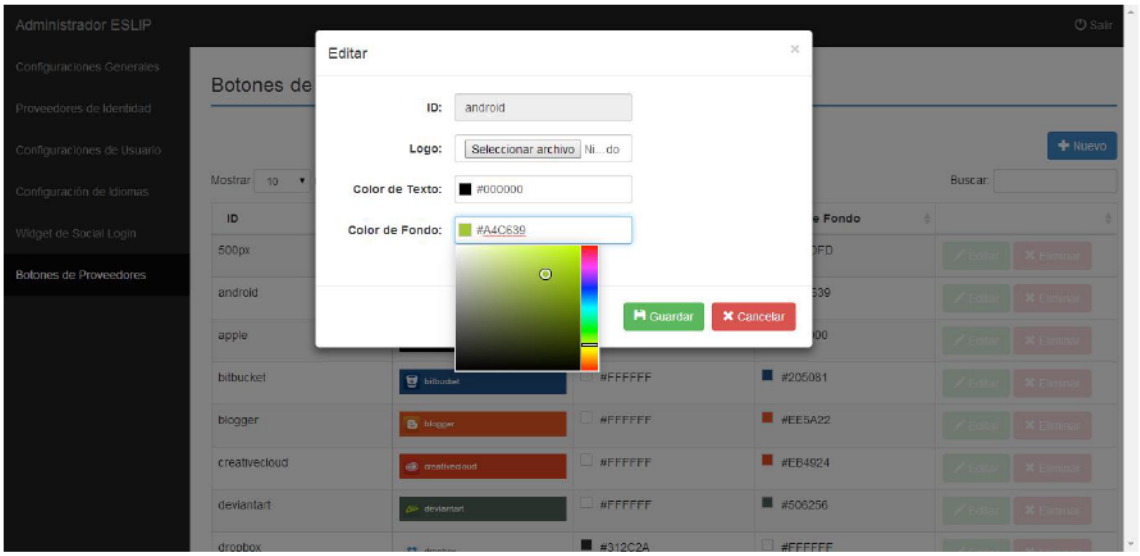

Figura 6.24: Captura de pantalla de la sección de configuración de los botones de proveedores del módulo de administración.

6.7. Licencia MIT

Como se dijo desde un principio, ESLIP es un proyecto de código abierto (Open Source). Se tomó la decisión de que sea de código abierto para que los desarrolladores o los dueños de los sitios en los que están interesados en integrar Social Login puedan estudiar, modificar y mejorar el código.

Los usuarios pueden adaptarlo a sus necesidades y corregir errores dando como resultado una mejora del plugin.

Luego de analizar diferentes licencias de código abierto, se optó por elegir la licencia MIT [85] para el plugin ESLIP. Esta licencia pone como condición que la nota de copyright y la parte de los derechos se incluya en todas las copias o partes sustanciales que se utilicen del software (en este caso el plugin).

Los derechos que otorga la licencia MIT son muchos: sin restricciones; incluyendo usar, copiar, modificar, integrar con otro software, publicar, sublicenciar y/o vender copias del software, y además permitir a las personas a las que se les entregue el software hacer lo mismo.

6.8. Contribuir con el proyecto

ESLIP es un proyecto de código abierto, como ya se ha mencionado anteriormente, por ese motivo es que se pone a disposición de la comunidad, tanto su código, como su documentación. Invitando a colaborar con el desarrollo, a quien desee hacerlo, buscando permanente el crecimiento y la evolución del plugin. Asimismo, contribuir con código no es la única manera de ayudar, también es posible traducir ESLIP a un idioma determinado o simplemente compartir los links de la página del plugin, para que más gente pueda conocerlo.

Para contribuir con el desarrollo del plugin se recomienda seguir el siguiente proceso:

1. Instalar Git (http://git-scm.com/downloads).

2. Registrarse en GitHub. (https://github.com/join)

3. Crear un fork del repositorio de ESLIP https://github.com/eslip/eslip. Clonar el fork en la máquina donde se va a desarrollar. Configurar los remotos. Crear un nuevo branch (del branch principal de desarrollo del proyecto) para contener su mejora, cambio o arreglo. (Detalles en https://help.github.com/articles/fork-a-repo).

4. Agregar, quitar o modificar lo que se crea necesario para la mejora del plugin.

5. Una vez realizadas las modificaciones, realizar un push de los cambios locales al fork en GitHub. Luego crear una solicitud de pull desde el branch creado al branch principal (master) (Detalles en https://help.github.com/articles/using-pull-requests). En la solicitud de pull, describir lo que hacen los cambios y mencionar el número de issue que se encuentra involucrado. Por ejemplo, "Cierra # 123".

6. Probablemente se origine una discusión sobre la solicitud de pull y, de ser necesario se realizará algún cambio, si todo resulta bien, se realizará el merge al branch principal del proyecto.

Recordar que quien realice la contribución estará de acuerdo en permitir a los propietarios del proyecto licenciar su trabajo bajo los términos de la licencia MIT.

Capítulo 7

Conclusiones y Trabajos Futuros

7.1. Conclusiones

Como conclusiones finales del trabajo realizado, podemos decir que hemos puesto de manifiesto los beneficios que tiene un usuario cuando inicia sesión en un sitio web utilizando las credenciales de su red social preferida o de su correo electrónico. Por otro lado descubrimos algunas desventajas que expone Social Login, pero que no son significativas en comparación a las facilidades que este le brinda al usuario.

El principal beneficio de Social Login es permitir que los usuarios accedan a un sitio web sin necesidad de un nuevo proceso de registro. Esta habilidad, sin duda, aumenta la posibilidad de que los usuarios se identifiquen en el sitio. Comprendimos que tampoco es bueno disponer sólo de Social Login, ya que ciertos usuarios sienten temor a utilizar sus cuentas de redes sociales para identificarse en otros sitios, ni tampoco sólo implementar un formulario de registro convencional, llegando a la conclusión que la mejor solución es incluir ambas opciones de identificación.

La investigación realizada sobre los fundamentos de Social Login y los protocolos que se ven involucrados en esta metodología de identificación nos sirvieron como base para desarrollar ESLIP, una herramienta que facilita a los desarrolladores a incluir Social Login en sus sitios web.

Por último creemos que ESLIP es un aporte a la comunidad de desarrolladores web, ya que no existe actualmente una herramienta de este tipo que sea de código abierto y con bajo costo de implementación. Cabe destacar que comprendemos que aún le falta maduración, pero estamos convencidos de que a medida que comience a ser utilizada y se obtengan sugerencias de los usuarios, ESLIP irá creciendo paulatinamente.

7.2. Trabajos Futuros

Un desarrollo como el de ESLIP, destinado a ofrecer facilidades para la implementación de Social Login, podría no tener fin, ya que cualquiera de los aspectos constituyentes de éste podría ser susceptible a un estudio de mejora.

Los siguientes temas son aquellos en los que creemos que pueden existir puntos de interés para la ampliación de este trabajo.

- En una primera etapa, se espera poder optimizar el diseño del widget de Social Login, de acuerdo a los requerimientos de usabilidad que se planteen, brindando más opciones de personalización y la posibilidad de

confeccionar temas de visualización. Así como también brindar una alternativa de widget apta para dispositivos móviles.

- Un trabajo que sería de mucha utilidad y que complementaria la presente tesina es la elaboración de diferentes versiones de ESLIP en forma de librerías o plugins para que pueda ser utilizado en diferentes frameworks y CMS's.

- Una mejora que podría llevarse a cabo a corto plazo sería traducir ESLIP a más idiomas, ya que actualmente solo está disponible en español e inglés

- Finalmente se tiene previsto agregar la posibilidad de exportar ESLIP a otros lenguajes de programación web como pueden ser Java, Python, Ruby o NodeJS.

Obviamente que el uso masivo de la herramienta generará requerimientos de cambios y mejoras las cuales serán consideradas a fin de perseguir la mejora continua de ESLIP.

Referencias

[1] Social Login [http://en.wikipedia.org/wiki/Social_login]. Accedido en Marzo 2014

[2] OAuth [http://oauth.net/]. Accedido en Marzo 2014

[3] OpenID [http://openid.net/]. Accedido en Marzo 2014

[4] Wizard [http://en.wikipedia.org/wiki/Wizard_(software)]. Accedido Enero 2014

[5] Ejemplos de empresas u organizaciones que proveen servicios pagos:
Janrain: [http://janrain.com/products/engage/social-login/]. Accedido Marzo 2014
Oneall: [http://www.oneall.com/services/single-sign-on/]. Accedido Marzo 2014
LoginRadius: [https://www.loginradius.com/]. Accedido Marzo 2014
Gigya: [http://www.gigya.com/social-login/]. Accedido Marzo 2014

[6] Código abierto [http://opensource.org/definition]. Accedido Marzo 2014

[7] URL [https://tools.ietf.org/html/rfc1738]. Accedido Marzo 2014

[8] Blog [http://lema.rae.es/drae/?val=blog]. Accedido Marzo 2014

[9] HybridAuth [http://hybridauth.sourceforge.net/]. Accedido Marzo 2014

[10] Gigya [http://www.gigya.com/social-login/]. Accedido Marzo 2014

[11] Janrain [http://janrain.com/]. Accedido Marzo 2014

[12] Facebook Login Button [https://developers.facebook.com/docs/plugins/login-button/]. Accedido Marzo 2014

[13] SaaS [http://en.wikipedia.org/wiki/Software_as_a_service]. Accedido Marzo 2014

[14] Anuncio publicado por Gigya [http://blog.gigya.com/which-identities-are-we-using-to-sign-in-around-the-web-infographic/]. Accedido Marzo 2014

[15] Artículo publicado por Janrain [http://janrain.com/blog/social-login-trends-across-the-web-for-q3-2013/]. Accedido Octubre 2013

[16] Grafo Social [http://es.wikipedia.org/wiki/Grafo_social]. Accedido Marzo 2014

[17] Conversión [http://en.wikipedia.org/wiki/Conversion_marketing]. Accedido Mayo 2014

[18] Blue Research [http://blue-research.com/]. Accedido Mayo 2014

[19] Investigación de Janrain sobre Social Login [http://www1.janrain.com/rs/janrain/images/Industry-Research-Consumer-Perceptions-of-Online-Registration-and-Social-Login-2012.pdf]. Accedido Mayo 2014

[20] eHow [http://www.ehow.com/]. Accedido Mayo 2014

[21] Lista de recomendaciones al implementar Social Login [http://socialmediatoday.com/beth-thouin/565839/do-s-don-ts-every-website-social-login-should-read-infographic]. Accedido Marzo 2014

[22] Booz & Co [http://www.booz.com]. Accedido Marzo 2014

[23] CRM [http://www.crmespanol.com/crmdefinicion.htm]. Accedido Marzo 2014

[24] Informe de Gigya correspondiente al primer trimestre [http://janrain.com/blog/social-login-trends-report-q1-2014/]. Accedido Mayo 2014

[25] Active Directory [http://support.microsoft.com/kb/196464/es]. Accedido Marzo 2014

[26] LDAP [http://ldapman.org/articles/sp_intro.html]. Accedido Marzo 2014

[27] SAML [http://saml.xml.org/saml-specifications]. Accedido Marzo 2014

[28] OASIS [https://www.oasis-open.org/]. Accedido Marzo 2014

[29] XML [http://www.w3.org/TR/REC-xml/]. Accedido Marzo 2014

[30] Mozilla Persona [https://developer.mozilla.org/es/docs/Persona]. Accedido Marzo 2014

[31] BrowserID [http://identity.mozilla.com/post/7616727542/introducing-browserid-a-better-way-to-sign-in]. Accedido Marzo 2014

[32] Mozilla [http://www.mozilla.org/]. Accedido Marzo 2014

[33] Mozilla Labs [https://mozillalabs.com]. Accedido Marzo 2014

[34] Verified Email Protocol [https://wiki.mozilla.org/Labs/Identity/VerifiedEmailProtocol]. Accedido Marzo 2014

[35] Publicación sobre diferencias entre BrowserID y OpenID [http://identity.mozilla.com/post/7669886219/how-browserid-differs-from-openid]. Accedido Marzo 2014

[36] WebID [http://www.w3.org/2005/Incubator/webid/spec/]. Accedido Marzo 2014

[37] HTTP [https://tools.ietf.org/html/rfc2616]. Accedido Marzo 2014

[38] W3C [http://www.w3.org/]. Accedido Mayo 2014

[39] MyOpenID [https://www.myopenid.com/]. Accedido Marzo 2014

[40] HTML [https://tools.ietf.org/html/rfc1866]. Accedido Marzo 2014

[41] XRI [https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xri]. Accedido Marzo 2014

[42] HTTP POST [http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html]. Accedido Marzo 2014

[43] URI Generic Syntax [https://tools.ietf.org/html/rfc3986]. Accedido Marzo 2014

[44] XRDS [http://docs.oasis-open.org/xri/2.0/specs/xri-resolution-V2.0.html] Sección 4. Accedido Marzo 2014

[45] Yadis [http://infogrid.org/trac/wiki/Yadis]. Accedido Marzo 2014

[46] SSL [http://tools.ietf.org/html/rfc6101]. Accedido Marzo 2014

[47] HTTP GET [http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html]. Accedido Marzo 2014

[48] HMAC-SHA1 [http://tools.ietf.org/html/rfc2104]. Accedido Marzo 2014

[49] HMAC-SHA256 [http://tools.ietf.org/html/rfc2104]. Accedido Marzo 2014

[50] MAC [http://es.wikipedia.org/wiki/Message_authentication_code]. Accedido Marzo 2014

[51] Diffie-Hellman [http://tools.ietf.org/html/rfc2631]. Accedido Marzo 2014

[52] Especificaciones de OpenID [https://openid.net/developers/specs/]. Accedido Marzo 2014

[53] ASCII [https://tools.ietf.org/html/rfc20]. Accedido Marzo 2014

[54] base-64 [https://tools.ietf.org/html/rfc3548]. Accedido Marzo 2014

[55] UTC16 [https://tools.ietf.org/html/rfc3339]. Accedido Marzo 2014

[56] Direcciones IP [https://tools.ietf.org/html/rfc791]. Accedido Marzo 2014

[57] Direcciones IP privadas [https://tools.ietf.org/html/rfc1918]. Accedido Marzo 2014

[58] Phishing [http://es.wikipedia.org/wiki/Phishing]. Accedido Marzo 2014

[59] VeriSign's OpenID SeatBelt Plugin [https://pip.verisignlabs.com/seatbelt.do]. Accedido Marzo 2014

[60] IETF [http://www.ietf.org/]. Accedido Marzo 2014

[61] Ma.gnolia [http://en.wikipedia.org/wiki/Ma.gnolia]. Accedido Marzo 2014

[62] RFC 5849 [http://tools.ietf.org/html/rfc5849]. Accedido Marzo 2014

[63] RFC 6749 [http://tools.ietf.org/html/rfc6749]. Accedido Marzo 2014

[64] RFC 2627 [http://tools.ietf.org/html/rfc2627]. Accedido Marzo 2014

[65] UTF-8 [http://tools.ietf.org/html/rfc3629]. Accedido Marzo 2014

[66] JSON [http://www.json.org/]. Accedido Marzo 2014

[67] OAuth 2.0 Assertion Profile (Perfil de Aserción OAuth 2.0) [http://tools.ietf.org/html/draft-ietf-oauth-assertions-12]. Accedido Marzo 2014

[68] SAML 2.0 Bearer Assertion Profile for OAuth 2.0 (Perfil de Aserción SAML 2.0 para OAuth 2.0) [http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer-17]. Accedido Marzo 2014

[69] Interoperabilidad en la implementación de OAuth [http://tools.ietf.org/html/rfc6749#section-1.8]. Accedido Marzo 2014

[70] OAuth 2.0: The good, the bad and the ugly [http://www.25hoursaday.com/weblog/2012/07/30/OAuth20TheGoodTheBadAndTheUgly.aspx]. Accedido Mayo 2014

[71] GitHub [https://github.com/]. Accedido Marzo 2014

[72] GITkit [https://developers.google.com/identity-toolkit/?hl=es]. Accedido Marzo 2014

[73] PHP [http://hybridauth.sourceforge.net/]. Accedido Marzo 2014

[74] Un widget web es una pequeña pieza de código proporcionada por terceros, que se puede añadir en cualquier tipo de página web que se ha creada usando HTML (o XHTML). Las tecnologías que se utilizan en los widgets web son JavaScript, Ajax y Adobe flash, pero en algunos casos también se utilizan otras tecnologías como HTML combinado con CSS y Java Applets. Un widget web se puede utilizar para:

- Mostrar información útil proveniente de sitio web.
- Proveer alguna funcionalidad extra (como: el voto, la votación).
- Haciendo suyos los productos y servicios de un sitio web.

[75] Javascript [http://www.w3ctutorial.com/js-basic/js-intro]. Accedido Marzo 2014

[76] CSS [http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo]. Accedido Marzo 2014

[77] Ajax [http://www.w3ctutorial.com/ajax-basic/ajax-intro]. Accedido Marzo 2014

[78] JQuery [http://jquery.com/]. Accedido Marzo 2014

[79] Bootstrap [http://getbootstrap.com/]. Accedido Marzo 2014

[80] jQuery Templates [https://github.com/BorisMoore/jquery-tmpl]. Accedido Marzo 2014

[81] Bootstrap Application Wizard [https://github.com/amoffat/bootstrap-application-wizard]. Accedido Marzo 2014

[82] Git [http://git-scm.com/]. Accedido Marzo 2014

[83] phpDocumentor [http://www.phpdoc.org/]. Accedido Marzo 2014

[84] Código de lenguaje ISO de dos letras en minúsculas [http://reference.sitepoint.com/html/lang-codes]. Accedido Marzo 2014

[85] Licencia MIT [http://www.opensource.org/licenses/mit-license.php]. Accedido Marzo 2014

Figuras

[Figura 6.21] Captura de pantalla de la sección de lenguajes del módulo de administración.

[Figura 6.22] Captura de pantalla de la plantilla para crear un idioma.

[Figura 6.23] Captura de pantalla de la sección de configuración del widget de Social Login del módulo de administración.

[Figura 6.24] Captura de pantalla de la sección de configuración de los botones de proveedores del módulo de administración.

Bibliografía

1. Rafeeq Rehman. The OpenID Book: Draft Version Revision 15. Conformix Books, 2007. ISBN: 0-9724031-2-4

2. D. Recordon, B. Fitzpatrick. OpenID Authentication 1.1. Mayo 2006

3. specs@openid.net. OpenID Authentication 2.0 - Final, Diciembre 2007

4. Eugene Tsyrklevich, Vlad Tsyrklevich. Single Sign-On for the Internet: A Security Story. BlackHat USA, 2007

5. Max Charas. Security in OpenID: An overview of OpenID from a security perspective. Master of Science Thesis Stockholm, 2009

6. Pavol Sovis, Florian Kohlar, Jorg Schwenk. Security Analysis of OpenID. Ruhr-University Bochum

7. Drummond Reed, Les Chasen, William Tan. OpenID Identity Discovery with XRI and XRDS

8. Ryan Boyd. Getting Started with OAuth 2.0. O'Reilly Media, Inc., 2012

9. Greg Brail, Sam Ramji. OAuth - The BIG Picture.

10. Estándar Framework OAuth 2.0. [http://tools.ietf.org/html/rfc6749]. Accedido Abril 2014

11. Estándar Protocolo OAuth 1.0. [http://tools.ietf.org/html/rfc5849]. Accedido Abril 2014

12. OAuth 2.0 - Centro para desarrolladores de Live Connect [http://msdn.microsoft.com/es-es/library/live/hh243647.aspx]. Accedido Abril 2014

13. Scott Nesbitt. Three Limitations and Disadvantages of Using Social Sign-On. Social Technology Review, Julio 2011

14. Scott Nesbitt. Using Social Sign-on to Target Content to Customers. Social Technology Review, Julio 2011

15. Scott Nesbitt. Driving Customer Engagement with Social Sign-On. Social Technology Review, Junio 2011

16. Bonnie Boglioli-Randall. Social Sign-On: Good Things Come in Multiples. Social Technology Review, Junio 2011

17. Brad Prescott. Social Sign-On: What is it and How Does It Benefit Your Web Site?. Social Technology Review, Enero 2011

18. Brad Prescott. Social Sign-On: The Case for Multiple Identities. Social Technology Review, Diciembre 2010

19. Jake Wengroff. Forget Your Password? Social Login Is There to Help. Social Media Today, Febrero 2012

20. Beth Thouin. Do's and Don'ts Every Website with Social Login Should Read [Infographic]. Social Media Today, Julio 2012

21. Beth Thouin. 4 Examples of Driving Marketing Strategies with Social Login Analytics. Social Media Today, Mayo 2012

22. Beth Thouin. 5 Ways to Increase Online Registrations Using Social Login. Social Media Today, Mayo 2012

23. Beth Thouin. 4 Hurdles to Overcome When Installing Social Login. Social Media Today, Mayo 2012

24. Beth Thouin. Decoding Social Login Pricing. Social Media Today, Mayo 2012

25. María González. Facebook, Google, Twitter, Amazon... ¿Por qué todos quieren su propio login social? Genbeta, octubre de 2013