André Azevedo Vargas
Gabriel Garcia Gava
Lucas Pereira da Silva

OpenID Connect

Junho de 2013

1. Introdução

O OpenID Connect define uma camada de identidade digital sobre o framework OAuth 2.0 (Open Authorization), e fornece a possibilidade dos clientes verificarem a identidade do usuário através de servidores autorizados e obterem informações da identidade do usuário.

Identidade digital é definido por um conjunto de informações que descrevem unicamente uma pessoa ou entidade, e também os interesses dessa entidade e como ela se relaciona com outras entidades. O OpenID Connect implementa a camada de identidade focando no usuário, onde os usuários “apresentam” sua identidade quando querem se identificar, e essa identidade tem a garantia de uma entidade confiável. Estes fundamentos pertencem ao Identity 2.0, diferentemente do Identity 1.0, onde o foco é a identificação focado no provedor do serviço, o que exige que o usuário crie uma identidade para cada entidade que ele queria obter os serviços.

1.1. Objetivos

Os objetivos gerais desse trabalho são:

- Pesquisar sobre o OpenID Connect, a nova especificação do OpenID Foundation para uma camada de identidade;
- Fazer um relatório sobre o OpenID Connect;
- Elaborar um protótipo para testar o funcionamento do OpenID Connect.

Quanto aos objetivos específicos, temos:

- Pesquisar sobre o histórico do OpenID;
- Pesquisar sobre a fundação OpenID;
- Estudar as especificações do OpenID Connect;
- Comparar as vantagens do OpenID Connect em relação à camada de identidade convencional;
- Construir um protótipo para simular casos de uso utilizando o OpenID Connect;
- Relatar os resultados obtidos em relação ao funcionamento do protótipo;

- Tirar conclusões sobre esta tecnologia de camada de identidade, e propor que tipos de trabalhos futuros são possíveis em relação ao OpenID Connect.

1.2. Metodologia

Para a criação do protótipo foram usados as seguintes ferramentas:

- Uma máquina virtual onde toda a plataforma de execução pudesse ser salva e executada;
- Uma servidor Web que irá pedir que o usuário se autentique para obter seus serviços;
- O site GitHub como provedor de identidade, que irá prover uma identidade certificada do usuário para o provedor de serviço.
- Uma aplicação Web com a qual o usuário irá interagir.

2. OpenID Connect

O OpenID Connect é um conjunto de especificações leves que provêm um ambiente para operações relativas à identidade via REST como APIs. Um simples implementação básica do OpenID Connect permite que clientes usando navegadores, dispositivos móveis, ou mesmo um cliente javascript, possam requisitar e receber informações sobre identidades e sesoẽs autenticadas. Porém especificações mais robustas suportam a cifragem dos dados, saber quem é o provedor de Identidade, e gerenciamento avançados da sessão.

O OpenID Connect consiste de um conjunto de especificações que definem uma camada de identidade. Trata-se da implementação de uma camada de identidade utilizando um conjunto de métodos chamado de identity 2.0, em cima de uma plataforma de autenticação chamada OAuth 2.0.

2.1. Histórico

O OpenID, originalmente chamado de Yadis (Yet another distributed system), foi criado por Brad Fitzpatrick em Maio de 2005, e ganhou esse nome após conseguir o domínio openid.net. Em março de 2006, JanRain, uma empresa de soluções Web envolvida no projeto, desenvolveu uma extensão para troca de perfis, posteriormente formalizando as extensões para o OpenID.

Em Junho de 2006, com a troca de URL com a identidade completa, para o uso de URL de provedores de Identidade, o OpenID começou a evoluir para uma verdadeira ferramenta de identidade digital, e assim passou a ser chamado de OpenID 2.0. Um ano depois, os líderes do projeto OpenID criam a fundação OpenID, um corporação de benefício público que lidasse com as questões de propriedade OpenID.

Em maio de 2011, a fundação OpenID cita pela primeira vez um nova especificação em que eles estão trabalhando, o OpenID Connect, que surge como objetivo principal uma especificação simplificada do OpenID anterior, que passou por várias mudanças e adição de extensões, provendo assim um serviço com as funcionalidades interessantes mais integradas, como o protocolo OAuth 2.0

2.2. OAuth 2.0

O OAuth é o padrão aberto para autorização. Ele fornece métodos para que o usuário se autentique perante a um site, e este possa ceder recursos que são relativos ao usuário que se autenticou. Além disso, ele permite que estas informações possam ser passadas para outro site, chamado de consumidor, sem que seja necessário que o consumidor necessite de informações secretas do usuário.

Já a versão 2.0 do OAuth é direcionada na simplicidade para desenvolvimento, ou seja, com uma API mais amigável. Ele também contém alguns métodos específicos para autorização em aplicações web, equipamentos móveis, entre outros.

2.3. Identity 2.0

Identidade é aquilo que define unicamente cada pessoa. Já na internet, identidade define unicamente cada usuário que a ela utiliza. O identity 2.0 é um conjunto de métodos para verificação de identidade digital na internet, ou seja, serve para que os provedores de serviço consigam saber quem está querendo acessar seus serviços.

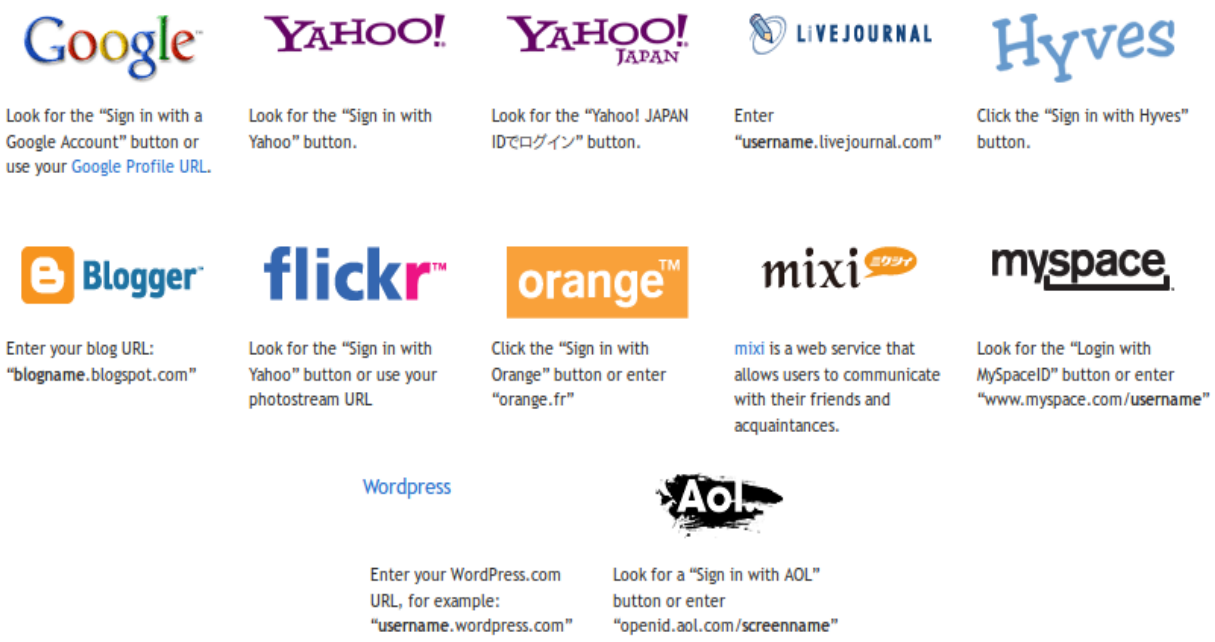
A grande vantagem do identity 2.0 em relação à sua antiga versão, é que ele é centrado no usuário, isto é, o usuário tem uma identificação que pode ser utilizada para acessar os serviços de diferentes provedores, diferente do método centrado nos serviços, onde os usuários tinham que ter uma identidade relacionada com o serviço para ter acesso a ele. Esta visão centrada no usuário traz uma maior transparência para ele em relação aos diferentes serviços, e elimina a necessidade de ter uma identidade para cada serviço, tendo assim, que lidar com uma quantidade muito grande de informações secretas (diferentes senhas, por exemplo).

2.4. Do OpenID para o OpenID Connect

O OpenID Connect é um sucessor do OpenID 2.0. Ele surgiu de forma a acabar com a fraquezas do OpenID 1.0. Entre as mudanças propostas, há algumas mais importantes. Primeiro, o OpenID Connect traz uma API mais amigável para os desenvolvedores, utilizando o formato JSON para a discrição das informações transmitidas. Outra mudança bastante significativa, é a integração com um framework de

autorização. Enquanto o OpenID 2.0 necessitava de uma extensão para poder usar o OAuth para a autenticação e autorização, esta nova versão já tem o OAuth integrado. O OpenID Connect é implementado em cima do framework OAuth 2.0.

O OpenID Connect também traz uma facilidade maior para se registrar nos sites, além de funcionar melhor do que anteriormente em dispositivos móveis. Ele também traz um nível de segurança variável, podendo alterá-lo dependendo da níveis requeridos de segurança. E por último, suas especificações são modulares, ou seja, os desenvolvedores precisam apenas implementar as funcionalidades que suas aplicações vão precisar.



Atualmente muitos sites já utilizam o OpenID Connect, incluindo grandes provedores de serviços, alguns dos quais podem ser vistos na figura acima, que juntos contabilizam mais de um bilhão de usuários. Ele fornecem ao usuário links diretos para provedores de identidade mais conhecidos, e assim não precisam que o usuário digite a URL de sua identidade. As maiores empresas que apoiam o OpenID são Google, Microsoft, Yahoo, MySpace, Verisign, GMX/Web.DE e France Telecom, provendo essa

alta contagem de usuários. Além disso, muitas empresas já oferecem suporte de autenticação OpenID, incluindo Facebook, Google, LinkedIn, Sears, Kmart.

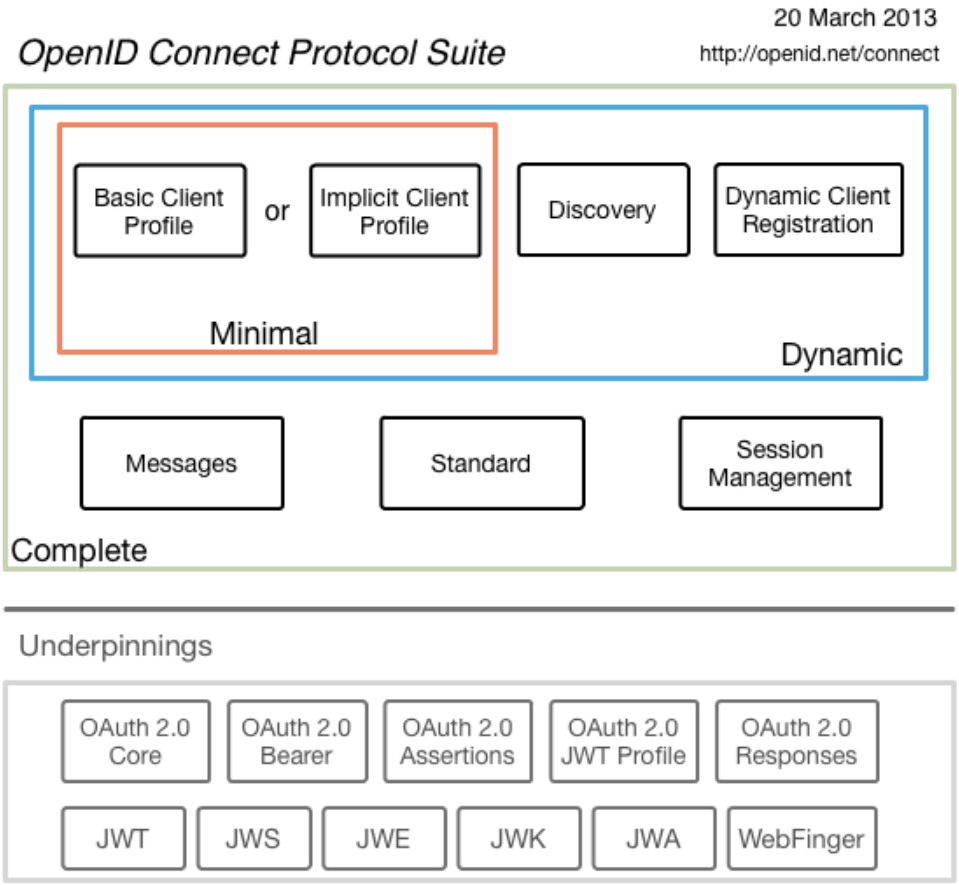
3. Funcionamento

Para que o cliente faça um pedido OpenID Connect, ele precisa ter as seguintes informações sobre o servidor:

- **Id do cliente**: um identificador único emitido para o cliente a identificar-se com o servidor de autorização.
- **Client secret**: um segredo compartilhado estabelecida entre o servidor de autorização e o cliente usado para assinar as solicitações.
- **Endpoint autorização do usuário final**: endpoint HTTP do servidor de autorização capaz de autenticar o usuário final e obter a autorização.
- **token endpoint :** endpoint HTTP do servidor de autorização capaz de emitir tokens de acesso.
- **informação do usuário endpoint:** é um recurso protegido que, quando apresentados com um token pelo cliente, retorna informações autorizadas sobre o usuário atual.
- **check id endpoint :** é um recurso protegido que quando apresentados com um token de ID do cliente, verifica a assinatura e retorna informações sobre a sessão do usuário (revogado 2012/03/03: pode voltar como um OAuth endpoint introspecção símbolo genérico).

3.1. Organização das Especificações

O openId Connect 1.0 é composto por oito documentos. São esses o Basic Client Profile, Implicit Client Profile, Discovery, Dynamic Registration, Messages, Standard, Session Management , OAuth 2.0 Multiple Response Types .



3.1.1. Basic Client Profile

OpenID Connect Basic Client Profile é um perfil que é projetado para ser fácil de ler e implementar pelos provedores de serviço baseado em Web usando OAuth com privilegio do tipo *authorization_code.* Esta especificação intencionalmente possui duplicatas das especificações Messages e Standard para fornecer um perfil de execução independente e auto-contido para partes confiáveis baseados em Web usando esses privilegios.

3.1.2. Implicit Client Profile

O OpenID Connect Implicit Client Profile é um perfil com base na Web confiavel usando o tipo de subsídio implícito OAuth. Provedores OpenID e aplicações não baseadas na Web devem consultar as especificações Messages e Standard. Porem este perfil omite implementação e considerações de segurança para provedores de OpenID e

aplicativos não baseados em Web.

3.1.3. Discovery

Para que um OpenID Client utilize os serviços OpenID Connect, para se comunicar com um End-User, o cliente precisa saber onde o provedor OpenID está. Utiliza-se webfinger para localizar o provedor para um end-user.

Uma vez que um provedor de OpenID é identificado, o endpoint e outras informações de configuração para que o OP é obtido a partir de um local conhecido como um documento JSON.

3.1.4. Dynamic Registration

Para que um cliente OpenID Connect usufrua dos serviços OpenID para um usuário final, o cliente precisa se registrar com o provedor OpenID, afim de adquirir um ID do cliente e segredo compartilhado. Este documento descreve como um novo cliente pode se cadastrar com o OP, e como as informações de registro para o cliente pode ser recuperada.

O registro Endpoint do cliente pode ser co-residente com o Endpoint token como uma otimização em algumas implementações.

3.1.5. Messages

Este documento apenas especifica as mensagens utlizadas pelo OpenID Connect, e o fluxo de troca delas. Porém é necessário que estas mensagens sejam usadas juntamente com algum protocolo de ligação, como o OpenID Standard. Mas também pode ser possível criar ligações para esta especificação em outros protocolos, como o IMAP ou XMPP.

3.1.6. Standards

O OAuth 2.0 especifica como que aplicações podem obter acessos limitados a recursos HTTP, ele define o uso de Access Tokens para o acesso de recursos, mas não dá suporte quanto à informações de Identidade.

Este documento faz a ligação do OpenID Messages com o OAuth 2.0, provendo

mensagem relativas a identidade e autenticação, construindo assim o OpenID Connect sobre o framework OAuth 2.0, e permitindo que aplicações que utilizem essa especificação também possam se comunicar com sistemas que usem o OAuth.

3.1.7. Session Manangement

Esta especificação está entre as opcionais. Ela trata da sessões entre o provedor de serviços, e o usuário final, tendo como assunto principal o logout. Ela diz sobre quanto que o provedor de serviços deve encerrar a sessão.

Para suportar esta funcionalidade, é preciso que o provedor de serviços obtenha a URL do provedor de identidade, e isto pode ser conseguido utilizando o OpenID Discovery já explicado anteriormente.

3.1.8. OAuth 2.0 Multiple Response Types

Este documento especifica como que as requisições de autorização do OAuth 2.0 devem ser tratadas, além de vários novos tipos específicos de resposta, respeitando as regras do OAuth.

4. Protótipo

Para a realização do protótipo decidimos criar um exemplo onde um usuário de nossa aplicação se autenticaria através de um provedor de identidade do OpenID Connect. Em um primeiro momento, nossa ideia foi utilizar o serviço de identidade oferecido pelo Google. Iniciamos então a implementação, porém nos deparamos com alguns problemas em relação a biblioteca disponibilizada pelo próprio Google para realizar os passos de autenticação e autorização. Além disso, a biblioteca do Google oferece uma visão bastante alto nível e, dessa forma, esconde muitos detalhes que são interessantes de serem mostrados. Devido a esses fatores, decidimos então procurar um outro provedor de identidade para realizar o nosso protópo e chegamos ao GitHub.

O GitHub fornece uma documentação bastante boa e atualizada sobre como utilizar os serviçoes do OpenID Connect fornecidos por eles. Entretanto, ao contrário do Google, eles não oferecem nenhuma biblioteca para facilitar a utilização desses serviços. O fato de não oferecem uma biblioteca para a utilização dos serviços dificutou um pouco a implementação, porém permitiu que entendêssemos melhor alguns detalhes referentes ao OpenID Connect.

Para a criação de nossa aplicação de exemplo utilizamos a linguagem Java no servidor e HTML em conjunto com JavaScript no lado cliente. A ideia do protótipo é que um usuário qualquer acesse a página cliente de nossa aplicação e se autentique conosco através de sua identidade fornecida pelo GitHub. Além disso, assim que o usuário se autentica solicitamos autorização dele para acessar os seus emails cadastrados no GitHub. Dessa forma, mostramos dois serviços importantes oferecidos pelo OpenID Connect, a autenticação e autorização.

4.1.Exemplo prático

Para o nosso exemplo considere um usuário que possui uma conta no GitHub e deseja utilizar a nossa aplicação. Apenas para fins de demonstração, o que a nossa aplicação irá fazer é: autenticar o usuário, requisitar autorização do usuário para obter informações dele no GitHub e, por fim, exibir os emails do usuário que estão cadastrados

no GitHub.

O fluxo se inicia quando o usuário acessa a nossa aplicação. Ao acessar, ele receberá uma página HTML que possui um botão para autenticação. O fluxo de interações entre a nossa aplicação, o usuário e o GitHub é o que segue:

1. Usuário clica no botão autenticar e uma requisicação é enviada ao servidor da nossa aplicação.
2. O servidor da nossa aplicação verifica que o usuário não está autenticado com a nossa aplicação e redireciona o usuário para a página de autenticação do GitHub. O servidor passa nos parâmetros desse redirecionamento o identificador da nossa aplicação cadastrada no GitHub (`client_id`), a URL de retorno para onde o usuário será redirecionado após se autenticar e dar autorização para acesso aos seus dados (`redirect_url`) e um item indicando quais são os dados do usuário que nossa aplicação deseja ter acesso (`scope`).
3. Após o usuário ser redirecionado para o GitHub pode acontecer dois cenários diferentes: no primeiro, o usuário já está autenticado no GitHub e no segundo não. No primeiro caso o GitHub redirecionará o usuário para o passo de autorização (item 5). Vamos considerar aqui o segundo caso, onde o usuário ainda não está autenticado junto ao GitHub. Nesse caso, o usuário é redirecionado para a página de autenticação (item seguinte).
4. Na página de autenticação o usuário irá fornecer as suas credenciais para que possa se autenticar. Caso a autenticação falhe, o usuário permanecerá na mesma página até que consiga realizar a autenticação.
5. Uma vez estando autenticado junto ao GitHub, podem ocorrer dois cenários diferentes: no primeiro, o usuário já deu autorização para que a nossa aplicação acesse aos seus dados no GitHub e no segundo essa autorização ainda não foi dada. Caso o usuário já tenha dado autorização então ele será retornado para a URL de retorno especificada no começo desse fluxo (item 7). Vamos considerar aqui o segundo caso, onde usuário

ainda não deu autorização, Nesse caso o GitHub redirecionará o usuário para a página de autorização (item seguinte).

6. Na página de autorização o usuário terá uma lista de quais dados a aplicação deseja acessar. O usuário pode então autorizar ou negar o acesso a esses dados. Caso autorize, o usuário será então redirecionado para uma URL da nossa aplicação que foi especificada no começo desse fluxo.
7. Após o usuário estar autenticado e ter cedido autorização para acesso aos seus dados no GitHub, ele é redirecionado para a URL especificada pela nossa aplicação no inicio desse fluxo. Além disso, é pasado como parâmetro nessa URL um atributo (`code`) fornecido pelo GitHub.
8. O servidor da nossa aplicação receberá então a requisição à URL mencionada no item anterior e irá obter dela o atributo `code`. Esse atributo `code` permitirá duas coisas: permitirá que a nossa aplicação verifique junto ao GitHub se o usuário realmente se autenticou e permitirá também que a nossa aplicação receba um `access_token` com o GitHub. Esse `access_token` possibilitará que a nossa aplicação tenha acesso aos dados do usuário no GitHub. Dessa forma, a nossa aplicação envia uma requisição para o GitHub e passa como parâmetro o `code`. Caso o GitHub responda com o `access_token`, então isso significa que tudo ocorreu dentro do previsto, ou seja, o usuário se autenticou e cedeu autorização aos seus dados. Caso contário, então ocorreu algum erro em algum dos passos, ou na autenticação ou na autorização.
9. Uma vez de posse do `access_token`, a nossa aplicação faz uma requisicação ao GitHub pedindo os dados do usuário e passa como parâmetro o próprio `access_token`.
10. Ao receber a resposta do GitHub com os dados do usuário, a nossa aplicação poderá então responder ao usuário. Como dito no início desse exemplo, para fins demonstrativos a unica coisa que a nossa aplicação fará

com os dados do usuário será retornar os emails dele cadastrados no GitHub.

5. Resultados

A criação do protótipo gerou uma aplicação cliente servidor simples. O servidor, escrito na linguagem Java, faz a intermediação entre o nosso cliente e o GitHub. O cliente é a interface direta com o usuário que deseja utilizar a nossa aplicação e é uma página Web escrita com HTML e JavaScript.

Para a disponibilização do protótipo foi criada e configurada uma máquina virtual. Essa máquina virtual foi criada a partir da ferramenta Virtual Box e possui o sistema operacional Ubuntu 13.04 de 32 bits. A máquina virtual foi então exportada para o formato OVA e ela pode ser rodada diretamente no Virtual Box bastando apenas que se importe o arquivo OVA. O protótipo foi então adicionado a máqina virtual que foi disponibilizada no Dropbox que é um serviço de armazenamento em nuvens.

A máquina virtual com o protóripo pode ser acessada através do endereço https://www.dropbox.com/sh/izr2m8w2ebadf49/mKeeRSSyiu.

Além da máquina virtual com o protóripo, esse trabalho gerou outros entregáveis como o relatório parcial, os slides comentados da apresentação e este próprio relatório final.

6. Conclusão

Como a própria fundação OpenID citou na criação do OpenID Connect, este é o “Segundo Ato do OpenID”. Podemos perceber como o conceito da Identity 2.0 está cada vez mais presente nos sites, e o OpenID Connect surge então para facilitar a utilização deste, permitindo assim que o OpenID possa ser mais difundido. No quesito de segurança o OpenID Connect também traz seus benefícios, visto que agora ele é implementado sobre uma plataforma de autorização poderosa como o OAuth 2.0.

Podemos ver também que esta tecnologia já é uma realidade, visto que as enormes empresas do campo da informática estão o utilizando, como a Google, Microsoft e Facebook. E isto tem seus motivos evidentes, pois ao mesmo tempo que os usuários não precisam mais criar cadastros em todos os sites que utilizam, e que provedores de serviço não precisam de tanta burocracia para conversar com um cliente, todo esse sistema garante níveis de segurança desejáveis, onde apenas é necessário que se confie no provedor da Identidade, ou seja, o OpenID Connect fornece simplicidade e facilidade, de forma segura e confiável.

Por útilmo, como ela se trata de uma tecnologia bastante nova, trabalhos futuros como estudos de utilização do OpenID Connect são bem vindos, por exemplo, a criação de aplicações que explorem especificamente o funcionamento do OpenID Connect, e não apenas como uma ferramenta.

Referências

- http://openid.net/connect/ (último acesso em: 26/06/2013).
- http://civics.com/wp-content/uploads/2012/04/OpenID-Connect-Lecture-for-MIT.pptx (último acesso em: 26/06/2013).
- http://stackoverflow.com/questions/10488973/openid-connect-providers (último acesso em: 06/06/2013).
- http://www.slideshare.net/oliverpfaff/openid-connect-an-emperor-or-just-new-cloths (último acesso em: 06/06/2013).
- http://openid.net/developers/libraries/ (último acesso em: 06/06/2013).
- http://nat.sakimura.org/2012/01/20/openid-connect-nutshell/ (último acesso em: 06/06/2013).
- http://developer.github.com/v3/oauth/ (último acesso em: 26/06/2013).