OpenID Connect for SURFconext

Assessment of the OpenID Connect protocol for Federations of Higher Education and Research

Project	: Samenwerkingsinfrastructuur
Projectjaar	: 2012
Projectmanager	: Bas Zoetekouw
Auteurs	: Martijn Oostdijk (Novay)
Opleverdatum	: December 2012
Versie	: 1.1

Summary

OpenID Connect is an upcoming standard for authentication, developed by the OpenID Foundation. It is based on OAuth 2.0, and the Service Provider side can be implemented relatively effortlessly, especially if the Service Provider already uses OAuth 2.0 for delegating access to other APIs.

In this report we evaluate the OpenID Connect protocol in the context of NREN identity federations, and more specifically, SURFconext. We describe the similarities and differences between OpenID Connect and SAML 2.0, which is typically used in identity federations nowadays, and we explore if and how OpenID Connect could be useful in the context of SURFconext.

Colofon

Programme line	: Collaboration Infrastructure 2012
Part	: Infrastructuur
Activity	: OpenID Connect
Deliverable	: SII12-19
External party	: Novay
Access rights	: public

This project was made possible by the support of SURF, the collaborative organisation for higher education institutes and research institutes aimed at breakthrough innovations in ICT. More information on SURF is available on the website www.surf.nl

Management Summary

OpenID Connect is an upcoming standard for authentication, developed by the OpenID Foundation. It is based on OAuth 2.0, and the Service Provider (Client) side can be implemented relatively effortlessly, especially if the Service Provider already uses OAuth 2.0 for delegating access to other APIs. OpenID Connect is designed for the consumer-to-social-network scenario, but can potentially be deployed in different environments such as the enterprise or federations for higher education and research. The OpenID Connect standard is not final at this point, yet it is stable enough that software developers are implementing it.

Some analysts have suggested that OpenID Connect (or a similar protocol) will replace SAML 2.0 in the long run. While this remains to be seen, it is certainly possible that certain categories of Service Providers will support OpenID Connect first to allow social login and only implement SAML if there is a clear business case for enterprise and/or higher education and research clients. Some big players (Google, Microsoft) support the OpenID Foundation in their development effort.

OpenID Connect achieves the same goals that SAML 2.0 is currently used for in SURFconext. It has certain features, such as asking for user consent, built in and as default. It can deal with higher levels of assurance and supports IdP discovery, dynamic client registration, and session management, though these are optional and/or under development. More static trust management is possible, of course, and standardization for typical NREN federation scenarios is being studied by NRENs and certain vendors.

Several use cases in which OpenID connect can play a role within federations for higher education and research are explored.

Whether OpenID Connect in its current form is here to stay is not yet known. OAuth 2.0 appears to be the protocol of choice for delegated access to APIs. A Service Provider that already implements OAuth 2.0 and that has a need for relaying authentication to an external IdP (within a federation) will surely look at OpenID Connect.

Table of Contents

1 Introduction

OpenID Connect is an authentication protocol specified by the OpenID Foundation[1]. As can be expected from the name it is based on an open, community driven design effort. It inherits its name from the OpenID protocol [OpenID 2.0] but is technically not directly related to earlier versions (see also Section 3.2). Instead it is based on the OAuth 2.0 [OAuth 2.0] standard. The purpose of the OAuth 2.0 standard is delegating access ("Open Authorization") by a user to a third party website to resources that typically take the form of a RESTful data-API. OAuth 2.0 is now an IETF standard, and is used by service providers in the social networking arena (Facebook, Twitter, LinkedIn, etc.) to allow client web sites and mobile apps access to resources of these service providers. Increasingly OAuth 2.0 is also used by cloud providers (Google Apps, Salesforce.com).

The typical use case of delegated access on behalf of users requires that users authenticate as part of most OAuth 2.0 flows, and as such the OAuth 2.0 protocol has been used explicitly for externalization of authentication in the past, for instance by Facebook[2]. Some of the other big players (e.g. Google, Microsoft) have come to the conclusion that OAuth 2.0 is a good foundation for authentication, and rather than having each potential identity provider implement their own externalization of authentication solutions on top of OAuth 2.0[3], OpenID Connect is an attempt, backed by many of these parties, to standardize this particular OAuth 2.0 use case.

OpenID Connect addresses the same problem that SAML 2.0 [SAML 2.0] addresses. SAML 2.0 has gained many supporters in the NREN world (and also in the enterprise world), yet apparently SAML 2.0 was not considered the optimal solution for the consumer-to-social-network use case of externalization of authentication. Some analysts have even speculated that SAML 2.0 will be replaced (as the de-facto standard for web based authentication) by competing protocols such as OpenID Connect[4].

This report places OpenID Connect in context and compares it to SAML 2.0 for use in a federation such as SURFnet's SURFconext.

1.1 Reading guide

This report focuses on OpenID Connect and assumes that the reader has some high-level technical knowledge of identity federations, in particular of the general features of SAML 2.0 and OAuth 2.0.

This report uses NREN federation terminology, such as Identity Provider (IdP), Service Provider (SP) etc., as well as terminology from the OpenID Connect and OAuth 2.0 specifications, such as Authorization Server, Resource Owner, Client. For the reader's convenience a mapping of terminology is given in Table 1.

---

[1] See http://openid.net/foundation/.

[2] Facebook Login uses OAuth 2.0 (and not the proprietary but popular Facebook Connect protocol), see https://developers.facebook.com/docs/concepts/login/login-architecture.

[3] See http://www.thread-safe.com/2012/01/problem-with-oauth-for-authentication.html.

[4] See http://blogs.kuppingercole.com/kearns/2012/07/31/the-death-and-life-of-a-protocol/.

Chapter 2 looks at the protocol, compares it to SAML 2.0 and discusses technical features such as client registration, IdP discovery, levels of assurance, security, and user centricity.

Chapter 3 looks at the application of OpenID Connect to SURFconext, NREN based federations in general, and other adoption issues.

Table 1 below maps the terminology used in OpenID Connect and OAuth 2.0 to terms that are more familiar to people with a SAML federation background:

Federation	OpenID Connect	OAuth
IdP	Authorization Server	Authorization Server
	UserInfo endpoint	Resource Server
SP	Client	Client
User	Resource owner	Resource owner
WAYF	Discovery	
Metadata	Client registration	Client registration
Single Sign Out	Session management	

Table 1: Terminology mapping

2 OpenID Connect

2.1 Specification

The diagram in Figure 1 (taken from the OpenID Foundation's website) gives an overview of the documents that make up the OpenID Connect specification and how they relate to one another. The OpenID Connect specification [OpenID Connect] is described in a central document[5] called *Standard*, which specifies the protocol itself and references the *Messages* document. The specification suite also contains the optional *Discovery*, *Dynamic Client Registration* and *Session Management* documents. Of course all these documents are based on a stack of other protocols, notably OAuth 2.0, and JSON Web Token [JWT], which in turn rely on standards such as JSON, HTTP, etc. The *Standard* is about 35 pages and is considered (at least by the authors of this report) to be concise, especially when compared to the SAML 2.0 [SAML 2.0] specification.

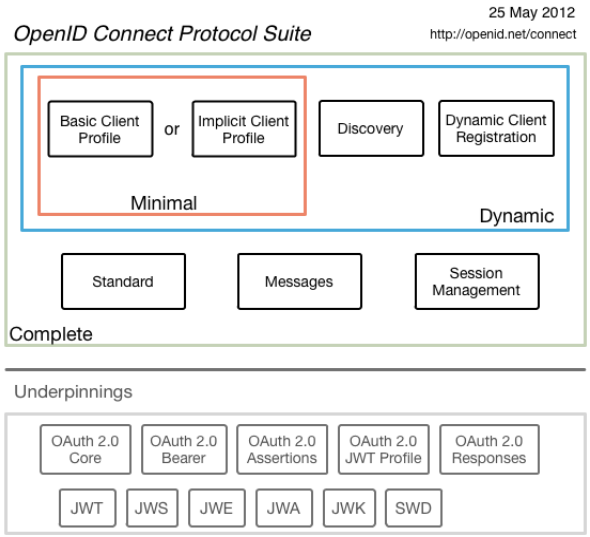

Figure 1: OpenID Connect specification overview

The *Basic Client Profile* and *Implicit Client Profile* are provided for the convenience of developers who only have a need for implementing the client side of the protocol for the more common use cases: accessing third-party content from a webserver-based client, a browser, or a (non-web) mobile app. SPs in a federation will typically base their implementation on the *Basic Client Profile* (or perhaps on the *Implicit Client Profile*); IdPs will base their implementation on *Standard*.

[5] At the time of writing this is: "OpenID Connect Standard 1.0 – draft 14".

2.1.1 Building on top of OAuth 2.0

Given that, in general, OAuth 2.0 implements delegated access to some data API (which is itself out of the scope of the OAuth 2.0 specification) the design of the OpenID Connect protocol can be best described as:

"Using OAuth 2.0 to give a client application access to authentication status and attribute information"

The authentication status is explicit. If the user fails to authenticate, or does not give consent then the IdP will return an error code. The authentication response can also, optionally, contain so-called claims (attribute name–value pairs) about the authentication event. After authentication the client can fetch additional claims about the user by accessing the UserInfo API.

2.1.2 High-level flow

OpenID Connect uses OAuth 2.0. The OAuth 2.0 protocol provides the client with a so-called *access token*. This access token is stored by the client and is subsequently used to access a data-API to fetch user information in the form of claims.

As part of the process to obtain the access token, the client passes a scope parameter containing a list of values that indicate specifically which (set of) attributes the client is interested in. The scope "openid" is mandatory and will yield a (for this client) persistent identifier. Other defined scopes are "profile", "email", "address", and "phone", all of which are optional.

In order for the client to obtain the access token, it needs to redirect the user to an authorization server (the IdP), which handles authentication, and which will ask for the user's consent to release the requested identity information (the requested scope, and specific claims) to the client.

The user is then redirected back to the client. Subsequent requests for user information take the form of accessing a data-API (the so-called UserInfo endpoint) protected by OAuth 2.0 to which the access token grants access. Note that no user interaction is required beyond the initial authentication and consent step as this takes place on a back-channel.

Other OAuth 2.0 services (APIs) offered by the IdP can be accessed by extending the scope of the original request to include additional scope values. This allows presenting the user consent page in a single overview page ("Do you want to share the following identity information with this third party *and* allow access to the following resources?").

2.2 Comparison with SAML 2.0

SURFnet's collaboration infrastructure SURFconext uses the SAML 2.0 Web SSO profile for authentication. In fact, most deployments of SAML follow this profile. The comparison is therefore limited to this particular SAML 2.0 profile. From a high level the two protocols achieve the same goal and do so in the same way, using slightly different technical means. The table below is partially based on slides from a presentation[6] by Roland Hedberg and is an attempt to enumerate the (apparent) differences:

Aspect	SAML 2.0	OpenID Connect
"Centricity"	Organization	User (but see below…)
Message format	XML	JSON, REST
Channel	HTTPS	HTTPS
Security	XMLDSig, XMLSEC	HTTPS, JWT
Client registration	Static	Dynamic
IdP discovery	Static	Dynamic
Attribute fetching	Front channel (but see below…)	Back channel

- Both SAML 2.0 and OpenID Connect are based on common web technologies; the message formats however are different. The difference can be summarized as: "JSON and REST" versus "XML and SOAP". For implementers, in the absence of a heavy-weight software library to support XML, JSON and REST are simpler to implement in common implementation languages as used on the web.
- OpenID Connect uses a back-channel between SP and IdP once an access token has been granted to the client, where SAML (at least in a Web SSO setting) will exclusively use the front-channel (through the user's browser). The primary use case is: mobile clients for which front-channel use is disruptive (typically the system browser is started to authenticate the user and to ask for consent, for all subsequent request the client uses the back-channel). Other SAML profiles (ECP) can use a back-channel as well.
- There is a difference in focus (at least according to Hedberg): OpenID Connect is user centric, while SAML 2.0 is organization centric. Certainly user consent is explicitly part of the OpenID Connect flow, yet is also possible using SAML 2.0[7]. Interestingly, though, the OpenID Connect "Basic Client Profile" uses a back channel between IdP and SP (see below), bypassing the user, which, while also possible in SAML (through Attribute Queries), is not commonly used in SAML setups. Some analysts point out that the pre-defined set of user attributes offered by OpenID Connect's UserInfo endpoint are more geared toward consumer-to-web service provider

[6] See http://youtu.be/doUZ715BHTY.
[7] See http://www.novay.nl/onze-mensen/maarten-wegdam/user-centric-saml/7466, and https://blogs.oracle.com/yvonne/entry/user_centricity_trust_technology_or.

scenarios than enterprise scenarios (where one would expect roles, entitlements, etc.), however, SAML also does not prescribe specific attribute names for enterprise situations.

– Security in SAML is typically implemented at the message level and uses the standard XML-based technologies for signing and encrypting (parts of) messages. Additionally (and typically) transport level security can be implemented using TLS. In OpenID Connect (and OAuth 2.0), on the other hand, transport level security is mandatory for all flows and message level security is optional. The IdP will typically demand that clients implement additional message level security, for instance in user-not-present situations (for browser-based clients the user is always present). Message level security is made possible through encrypting and signing JWT objects (see Section 2.3 for details).

– OpenID has some dynamic aspects; SAML deployments typically use a static trust configuration setup through meta-data which needs to be edited to add SPs. OpenID Connect allows "Dynamic Client Registration" where new clients only need to be provided with an access token (out of band) and can then visit a client registration endpoint to be registered with an IdP and receive further client credentials.

All in all OpenID Connect solves the same problem that SAML 2.0 does, and the protocols do so in a comparable way. Differences are mostly in flavour and default settings.

2.3 Security

The security of OpenID Connect, at the protocol level, inherits some features from OAuth 2.0. Unlike earlier versions of OAuth, which relied on cryptographic signatures for security that were perceived to be complex by many developers, OAuth 2.0 has been designed for a wide audience of potential developers. Therefore, the default security in OAuth 2.0 relies on TLS (a secure channel between protocol principals over which a so-called bearer tokens are exchanged)[8].

The security model used in OpenID Connect is described by some as layered[9]. For the most basic browser-based client flow (the "Implicit Client Profile") the only security mechanism supported is TLS at the HTTP level. The token is placed in the fragment part of the URL, which ensures that only the client can access it. For such clients this level of security is deemed appropriate, and is comparable to the security level achieved by web applications that handle their own authentication over HTTP/TLS.

In the basic web server based client flow (the "Basic Client Profile") an intermediate credential called an authorization code is used instead of a bearer token. The authorization code is later exchanged for an access token, but this is done on a back channel between the client and the authorization server. As the user (or malware at the user's side) cannot influence this back channel this can be seen as an additional layer of security.

---

[8] On OAuth 2.0 security, see http://hueniverse.com/2010/09/oauth-bearer-tokens-are-a-terrible-idea/. The author argues that additional layers are needed on top of bare OAuth 2.0 (which is what OpenID Connect does).

[9] See, for instance, the Google IO 2012 talk by Ryan Boyd: http://youtu.be/YLHyeSuBspI, which describes the Google interpretations of OAuth 2.0 and OpenID Connect (called "OAuth for Authentication" at Google) and considerations for adding layers of security for higher levels of access.

In still other scenarios (perhaps dictated in future profiles) actual message level security can be employed. Tokens implemented through JWT [JWT] may contain signatures and may have an encrypted payload. It is up to the stakeholders involved to decide whether such higher layers of security are necessary.

In the end, OpenID Connect can be made as secure as SAML 2.0, albeit with different mechanisms and a more granular, layered approach.

2.4 Levels of authentication assurance

OpenID Connect, like SAML, is agnostic to the authentication technology used: it demands that the user is authenticated by the authorization server, but it does not prescribe how to do this. Still, clients might need information about the level of assurance at which a user's identity has been established during authentication. OpenID Connect (optionally) supports signalling of level of assurance through a so-called *ACR* claim (ACR stands for *AuthenticationContextReference* as seen in SAML 2.0) in the ID token. The claim can appear both in the request (specifying a requirement by the relying party on the authentication process at the IdP) and in the response (specifying the actual level of assurance).

2.5 Dynamic aspects

Most of the optional parts of the OpenID Connect specification deal with dynamic aspects. Being able to dynamically add clients or to dynamically discover IdP-settings given only a user identifier (typically the user's e-mail address) are important features when trying to design a consumer authentication protocol that needs to scale to the whole of the Internet (an open-ended user community, and an open-ended developer community). Even though these aspects are, perhaps, less relevant in a federation (or enterprise) scenario, they are discussed here as they may become more important in future use cases such as collaborations across different federations.

2.5.1 Client registration

Social network sites (Facebook, Google, etc.) typically want to exercise some control over their developer community using the site's APIs (to prevent rogue developers from offering malicious services to their users, perhaps). Statically setting up a whitelist of developers does not scale, of course, and therefore a more dynamic solution is needed.

OpenID Connect allows "Dynamic Client Registration": An IdP implements a registration end-point that can be accessed by clients that want to register with the IdP. The endpoint can be protected with OAuth 2.0, so that new clients only need to be provided with an access token (out of band[10]) and can then visit a client registration endpoint to be registered with an IdP and receive further client credentials.

[10] Typically a developer signs in as a normal user, requests developer status from the Social Network site, is then given an "API key" which is either an authorization code or an access token.

Client registration in OpenID Connect is a further specialization of Client registration in OAuth 2.0. OpenID Connect specifies a number of additional parameters in the request that deal with typical OpenID Connect related issues (the various endpoints, ACR, etc.).

2.5.2 Discovery

The Where-Are-You-From (WAYF) problem that federations face (especially when the number of entities in a federation grows) also surfaces in a consumer-to-social-network-site scenario[11]. Users need to be able to indicate to the SP to which IdP they want to be redirected, and OpenID Connect requires that clients have some information about that IdP, specifically the location of the various endpoints. In the consumer-to-social-network scenario the client (SP) may never have dealt with a particular IdP before. The optional "Discovery" part of the OpenID Connect specification relies on SWD (Simple Web Discovery), which derives from a user provided identifier (the user's email address) a standard location at the IdP's site where this information can be found.

To help client (sites) in implementing the WAYF/Discovery flow, the OpenID Foundation has another working group that works on a product called Account Chooser[12]. Account Chooser is a browser-based application that allows users to register IdP accounts. Both OpenID Connect and SAML 2.0 are supported by Account Chooser. A (web server or browser-based) client embeds the Account Chooser dialog, showing users which account is currently being used by the site and can send a user to the Account Chooser application where they can select a previously registered account at an IdP, or add a new one.

2.5.3 Session management

The ability to dispatch events from an OpenID Connect IdP to all clients that the user is currently logged in to, is a very strong feature (indeed it would be the "holy grail" for federated identity management). Specifically it makes single-sign-out possible, where, on the event of a user logging out of the IdP, the IdP signals to all SPs that this event took place. The optional "Session Management" part of the OpenID Connect specification deals with the session life-cycle. This part of the specification[13] appears to be in draft at the time of writing of this report. The solution is based on HTML iframes (and thus would only work for web server based clients).

2.5.4 Authentication request can target specific attributes

OpenID Connect requests (as sent by clients) can contain claims, which can be viewed as filters specifying to the IdP which attributes to fetch. In addition to the scope mechanism this means that clients can be very specific as to which attributes are needed. The IdPs need only reveal information that is relevant to clients. SAML has a similar mechanism, but the SAML solution is more static[14].

---

[11] In this context the problem is often referred to as the NASCAR page problem, as SPs typically confront a user with a page filled with IdP logos similar to the NASCAR race event's sponsor logos.

[12] See https://www.accountchooser.com, and also the various demos at https://sites.google.com/site/oidfacwg/cdsdemo.

[13] See "Session Management" draft 08, see http://openid.net/specs/openid-connect-session-1_0.html.

[14] See Oliver Pfaff presentation on slideshare: http://www.slideshare.net/oliverpfaff/openid-connect-an-emperor-or-just-new-cloths.

3 OpenID Connect for SURFconext

Applying OpenID Connect in a federative context (a closed trust circle) is new. At this moment federations for higher education and research are SAML-based. Gartner's 2012 Hype Cycle on Cloud Computing places OpenID Connect "on the rise"[15], which means that it might become mainstream technology in a couple of years. There is certainly an interest from the NREN community in OpenID Connect. Feide is attempting to come up with a profile for setting up trust relations in a federation[16]. Gluu (an Open Source software vendor, which has deployed SAML based products in NREN situations) appears to define a similar profile for using OpenID Connect in multi-party federations[17]. There is a test service for OpenID Connect implementations (mentioned below) operated by Roland Hedberg and Andreas Solberg, both active members of the NREN community.

Adoption of OpenID Connect in NREN federations may change if and when OpenID Connect-only SPs start to emerge. It is, at this point, not clear what category of SP this will be.

- Traditional service providers for higher education: publishers like Elsevier, "SURFspot"
- SURFnet-hosted service providers
- Local services, running at the IdP, (perhaps Student Information Systems) from which authentication has been externalized
- Cloud service providers: Google Apps, Microsoft Live@Edu, etc.
- Collaboration-oriented service providers

Out of these, the cloud service providers are the most likely candidates.

[15] Can be purchased from http://www.gartner.com. See http://my.gartner.com/portal/server.pt?open=512&objID=202&&PageID=5553&mode=2&in_hi_userid=2&cached=true&resId=2096517&ref=AnalystProfile for a table of contents which already shows OpenID Connect's position.
[16] See https://rnd.feide.no/2012/08/24/openid-connect-federations/.
[17] See http://wiki.openid.net/w/page/59727624/Multi-Party%20Federations.

3.1 Adoption, maturity, software support

In March and June 2012, OpenID Connect interoperability events took place[18]. A list of implementations was extracted from the resulting web page. Another source of OpenID Connect implementations is the online test service run by UNINETT[19], which implementers[20] can use to test their implementations in a number of flows.

Vendor	Website / demo site / software repository	Type
AOL	http://www.aol.com	Demo?
eBay	https://openidconnect.ebay.com/	Source?
Edmund Jay	http://openid.bitbucket.org/	Source
Emmanuel Raviart	https://gitorious.org/wenou	Source
Gluu	http://www.gluu.org/cloud-identity/open-source/overview/	Source
Google	http://oauthssodemo.appspot.com	Demo
Heroku		
IBM	http://www.ibm.com	Demo?
Layer 7	http://www.layer7tech.com	Vendor
Mitre	http://mitre.org	Vendor?
Nov Matake	http://matake.jp, https://github.com/nov/openid_connect	Source
oic.info		
oic4us		
Orange	http://pub-openid-int.orange.fr/oauth	Demo?
Ping Identity	http://www.pingidentity.com	Vendor
Roland Hedberg	https://github.com/rohe/pyoidc	Source
Ryo Ito	http://d.hatena.ne.jp/ritou, https://openidconnect.info/	Demo?

[18] See http://osis.idcommons.net/wiki/OC4:OpenID_Connect_Interop_4.
[19] See http://openidtest.uninett.no/results.
[20] At the time of writing: November 29th 2012.

All in all there are at least 15 different OpenID Connect implementations. There is even an OpenID Connect IdP implementation for iPhone[21].

There are many more OAuth implementations that claim OAuth 2.0 compatibility out there. Assessing the quality of these implementations is out of scope of this report.

3.2 Why did previous versions of OpenID not succeed?

As outlined in the introduction, OpenID Connect does not technically derive from earlier versions of OpenID (version 1.x, 2.0, simply referred to as "OpenID" below), instead it was built on top of OAuth 2.0. A complete analysis of why OpenID failed to succeed as the ultimate consumer identity protocol for the Internet is outside of the scope of this report. Instead the following points about OpenID (and how OpenID Connect differs) are presented here:

- Many parties implemented the IdP side of OpenID, yet not that many implemented the SP side. OpenID Connect's relation with OAuth 2.0 and OAuth 2.0's popularity may make it easier for SPs to adopt OpenID Connect.
- To developers implementing the SP side the original OpenID protocol appeared easy to implement, but there were subtle pitfalls. Many developers implemented the protocol from scratch (not based on a third party APIs), and this resulted in non-interoperable and/or insecure (see also [Van Delft, Oostdijk, 2010]) deployments. The "too complex to implement securely" argument has been voiced by David Recordon, one of the original authors of the protocol[22].
- OpenID used a URL to identify a user, even though users generally associate an email address with a user account. In general there were some user-experience issues to be solved in OpenID. Third party identity selectors (similar to Account Chooser in Section 2.5.2) only partially solved this problem.
- Solutions based on OAuth and Facebook Connect started to appear, were easier to implement, and so-to-speak "ate OpenID's lunch".

3.3 Use cases

3.3.1 OpenID Connect to SAML gateway

A possible use case to explore could be for SURFconext to act as a gateway to connect SPs that cannot or will not implement SAML. This makes sense as soon as such SPs (of interest to institutions of higher education and research) start to appear in significant numbers.

In principle a gateway could also be built to allow users to log in to non-federative SPs using the account at their home institution's IdP (i.e. allow access to non-registered SPs). This is *not* advisable for the well known reasons: federation accounts will cease to exist as students graduate or employees leave the IdP's organization, malicious phishing sites may tempt unsuspecting users to enter their credentials, etc.

---

[21] See http://nat.sakimura.org/2012/04/15/openid-connect-idp-on-iphone/.

[22] For instance here: http://lwn.net/Articles/390626/, and here: http://blog.mastermaq.ca/tag/openauth/.

Supporting OpenID Connect on the IdP side is currently not interesting to explore as most IdPs are perfectly capable of implementing SAML. A possible exception could special IdPs, for example for guest accounts (in temporary Virtual Organization collaboration context).

3.3.2 Non-web Federated Authentication

OpenID Connect's *Implicit Client profile* shows developers how to use the protocol from a mobile device based client. There are best practices how to deal with directing the user to the IdP's sign on page and consent page (use the system browser so that existing session information (for SSO) can be re-used, do not ask for a password from the app as the user may user two-factor authentication). OpenID Connect appears to be the obvious candidate for Non-web Federated Authentication.

3.3.3 Provisioning revisited

[Oostdijk, 2010] studies a comprehensive set of provisioning scenarios and gives appropriate provisioning mechanisms (supported by protocols such as SAML Web SSO, SAML attribute query, SPML, direct LDAP access, etc.) that map to the different application archetypes corresponding to the scenarios. One of the conclusions was that "just-in-time-provisioning", based on Web SSO style attribute release, was sufficient for the more common scenarios. Some more exotic scenarios needed direct, back channel communication between IdP and SP, initiated by either IdP or SP. OpenID Connect (optionally) supports "Session management", and (always) an OAuth 2.0 style back channel communication based on access tokens. Therefore OpenID Connect may be an interesting candidate for scenarios where both SAML JIT provisioning and SPML failed.

3.3.4 Level of assurance service

Finally, it may be worth integrating support for OpenID connect in the Step-up Authentication-as-a-Service offering SURFnet is considering to implement. OpenID Connect allows an authentication context reference to be passed in requests and responses, just like SAML. The service might be of interest to a wider audience if it supports both SAML and OpenID Connect.

4 Conclusion

In this document, we have evaluated OpenID Connect in the context of NREN identity federations in general and SURFconext specifically.

We can conclude that OpenID Connect is a promising technology that offers similar functionality to the SAML 2.0 protocol, which is the main technology used in modern identity federations. OpenID Connect is built on top of the OAuth 2.0 authorisation protocol, which is quickly becoming the de factor standard to handle authorisation for cloud services. This makes OpenID Connect a lot easier to implement than SAML 2.0, both for Service Providers and for Identity Providers, in particularly if they are already using OAuth 2 to control access to backend resources.

As of yet, the uptake of OpenID Connect in the general marketplace is not very large. It is unclear if this will change, and in which timeframe service providers will start adopting OpenID Connect to allow "social logins" to their services, if at all. Furthermore, the standard set of OpenID Connect functionality offers the same functionality as typical identity federations (including SURFconext) can deliver. Therefore, there doesn't seem to be a urgent need to implement OpenID Connect support in current identity federations.

The extended parts of the OpenID Connect protocol, however, do offer some interesting new technology, for example *IdP-discovery*, *dynamic client registration*, and *session management*. Development of the protocol in these areas should be closely followed and might lead to interesting new functionality in the near future.

References

Burr, B., Polk, T., Dodson, D., *Electronic Authentication Guideline, NIST Special Publication 800-63 version 1.0.2, April 2006*

Delft, B. van, Oostdijk, M., *A Security Analysis of OpenID, Proc. IFIP Advances in Inf. And Comm. Technology, Vol. 343,* http://dx.doi.org/10.1007/978-3-642-17303-5_6*, 2010*

Cantor, Kemp, Philpott, Maler, *Security Assertion Markup Language (SAML) V2.0, OASIS Standard, 15 March 2005*

Jones, Mike, *JSON Web Token,* http://self-issued.info/docs/draft-ietf-oauth-json-web-token.html*, 2012*

Oostdijk et al., *Provioning Scenarios in Federations, SURFnet,* http://www.surfnet.nl/nl/Innovatieprogramma's/gigaport3/Documents/EDS-4%20Provisioning%20Scenarios%20in%20Federations%20Final.pdf , 2010

Pfaff, Oliver, *OpenID Connect an Emperor or just new Clother?*, http://www.slideshare.net/oliverpfaff/openid-connect-an-emperor-or-just-new-cloths, last consulted on November 2012

Sakimura, N., Bradley J., et al., *OpenID Connect Standard 1.0 – draft 14,* http://openid.net/specs/openid-connect-standard-1_0.html*, December 2012*