


Monday, June 25, 2007

## Refactoring an OpenID Library

Since I have a three track mind (but only one dedicated to PHP), I'm stuck with another OpenID post. Over the weekend, I managed to grab a few hours to dig around my OpenID library with the ultimate development tools: patience and experimentation. There are also a few integration tests as a safety net though 

. So I will be sending a copy around to a few people including Dmitry who has posted a Zend Framework proposal for OpenID on the wiki (which is currently down, as usual, so I haven't managed to add a comment on it yet - should be more stable in the near future I hear).

The refactored library changed a few aspects of the original code I wrote last Autumn. The original placed a lot of responsibilities in the Consumer class which meant it was very difficult to actually change things without knock on effects elsewhere. What I've refactored towards is a splitting of the OpenID process based on three categories: Request, Redirect, Response. The OpenID Specification is all about communication, and this type of splitting seems to have worked out very well. The other change was to centralise all values which are effectively constants to the top-level OpenID class. This class also manages the current version via a static public method - `OpenID::setVersion()`.

So what does the library not include (always a good question!).

1. XRI Support is not complete. I need to add an interface to the Yadis object to extract an XRI's Canonical ID which is the actual value used as an OpenID claimed identifier.
2. Stateless Mode is not supported. Also called Dumb Mode, it's necessary when the server cannot store state between requests. Since PHP is perfectly stateful I omitted it for now.
3. Good session integration for frameworks. The library currently directly accesses `$_SESSION` under an "OPENID\_SESSION" namespace. For the Zend Framework and others which have a Session class (e.g. `Zend_Session_Namespace`) this may not work well if sessions are being written to the database, or session expiry dates being set.

What does it support?


It supports almost the entire OpenID 2.0 Authentication Specification (draft 11 since it's not finalised). It also transparently covers a few common edge cases such as OP Servers which do not always return an "ns" namespace value in responses. It also downgrades and tracks the association type when an OP cannot use SHA256 (e.g. a number of PHP4 based providers). It also supports OpenID Extensions - you can add extension types quickly to the current list (i.e. the Simple Registration Extension (sreg) for now).

Improvements over original library?

It now has wider support for Diffie-Hellman and HMAC using one of mhash or hash extensions (both cryptographic algorithms are segregated to their own class also). Classes are now loosely coupled, so maintenance and individual class testing is simpler. Logic is more dispersed across smaller methods - again making it easier to maintain, test, and modify discrete chunks of logic. I've also centralised the constant values, and amended the API to be similar to that used by JanRain's PHP4 library. There are of course differences from JanRain's API but the flow is very similar.

Other stuff?

Except for some API improvements I should make, it includes a functional Yadis implementation. A task for later is allowing the OpenID library check whether an OP supports extensions like Sreg by checking the list of available Service types discovered by the Yadis protocol. This only needs a minor change - I first need to check whether a missing sreg service in an XRD document is a definitive sign it is not supported, or whether it's entirely optional to show it as a

service. (More spec reading this evening   
).

Aside from an example to be passed to the Zend Framework proposal, the library in it's refactored form will be proposed to PEAR. I think I'll propose Services\_Yadis first since it's a component of the larger OpenID scheme.

Posted by Pádraic Brady in Openid and Yadis, PHP General, PHP Security, Zend Framework at 19:03