Loudhailer security

ThenMedia's Loudhailer connects to your social networks using OAuth. This is a secure protocol for transmitting data between servers. The following information has been compiled from oauth.net and is adapted from Explaining OAuth, published in 2007 by Eran Hammer-Lahav. It explains what OAuth is and why it is essential for keeping your social network profiles secure whilst using our services.

A Little Bit of History

OAuth started around November 2006, while Blaine Cook was working on the Twitter OpenID implementation. He got in touch with Chris Messina looking for a way to use OpenID together with the Twitter API to delegate authentication. They met with David Recordon, Larry Halff, and others at a CitizenSpace OpenID meeting to discuss existing solutions. Larry was looking into integrating OpenID for Ma.gnolia Dashboard Widgets. After reviewing exiting OpenID functionality, as well as other industry practices, they came to the conclusion that there was no open standard for API access delegation. The conversation continued online and off for a few months.

In April 2007, a Google group was created with a small group of implementers to write a proposal for an open protocol. It turned out that this problem wasn't unique to OpenID and when DeWitt Clinton from Google caught wind of the project, he expressed his interest in supporting the effort, if only as a stakeholder. In July 2007 the team drafted an initial specification and the group was opened to anyone interested in contributing. After many online and face to face discussions, the OAuth 1.0 Draft is due out next week.

What is it For?

Everyday new website offer services which tie together functionality from other sites. A photo lab printing your Flickr photos, a social network using your Google address book to look for friends, and APIs to build your own desktop application version of a popular site. These are all great services – what is not so great about some of the implementations available today is their request for your username and password on the other site. Using Twitter as an example, the site simple yet powerful API created a rich community of applications built on top of its platform. But in order for those applications to update your status on Twitter, they must ask for your password. When do so, not only you expose your password to someone else (yes, that same password you also use for online banking), you also give them full access to do as they wish. They can do anything they wanted – even change your password and lock you out.

This is what OAuth does, it allows you the User to grant access to your private resources on one site (which is called the Service Provider), to another site (called Consumer, not to be confused with you, the User). While OpenID is all about using a single identity to sign into many sites, OAuth is about giving access to your stuff without sharing your identity at all (or its secret parts).

OAuth and OpenID

OAuth is not an OpenID extension and at the specification level, shares only few things with OpenID – some common authors and the fact both are open specification in the realm of authentication and access control. 'Why OAuth is not an OpenID extension?' is probably the most frequently asked question in the group, and also my first when I joined the OAuth effort. The answer is simple, OAuth attempts to provide a standard way for developers to offer their services via an API without forcing their users to expose their passwords (and other credentials). If OAuth depended on OpenID, only OpenID services would be able to use it, and while OpenID is great, there are many applications where it is not suitable or desired. Which doesn't mean to say you cannot use the two together. OAuth talks about getting users to grant access while OpenID talks about making sure the users are really who they say they are.

You can learn more about the OAuth security system here: **http://oauth.net/**