WHITEPAPER

Achieving Scalable Access Control and Privacy Protection With User-Managed Access

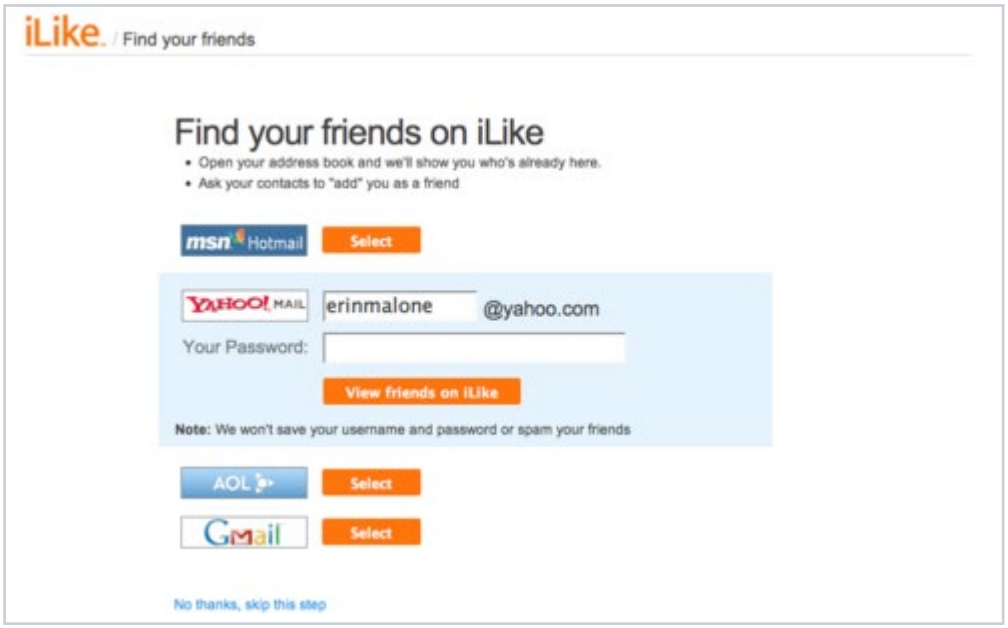
Benjamin Franklin once quipped that three people might keep a secret if two of them are dead. However, absolute secrecy of private or confidential information is impractical because, often enough, you must share it for good reason, such as disclosing your income to your tax accountant. In addition to software safeguards, ensuring that proprietary information is delivered only into the right hands at the right time requires preplanning and tool support.

As part of its OpenUMA project, ForgeRock is implementing the User-Managed Access specification. UMA is a profile of OAuth that offers standardized, modern Web APIs, resolving many issues in authorization, privacy, and consented sharing. Most important, UMA facilitates optimum user control, a key requirement in privacy protection.

Pitfalls and Challenges in Protected Access

In the early days of the Web, personal information, mostly collected through completed online forms, was error-prone because life is change. A common occurrence like moving outdated all the online records that contained your old address.

To head off that problem, some sites adopted a practice known to security experts as the password anti-pattern for information sharing. That is, once the user had signed in, the site prompted for that individual's login credentials to third-party sites. Below is a typical example.

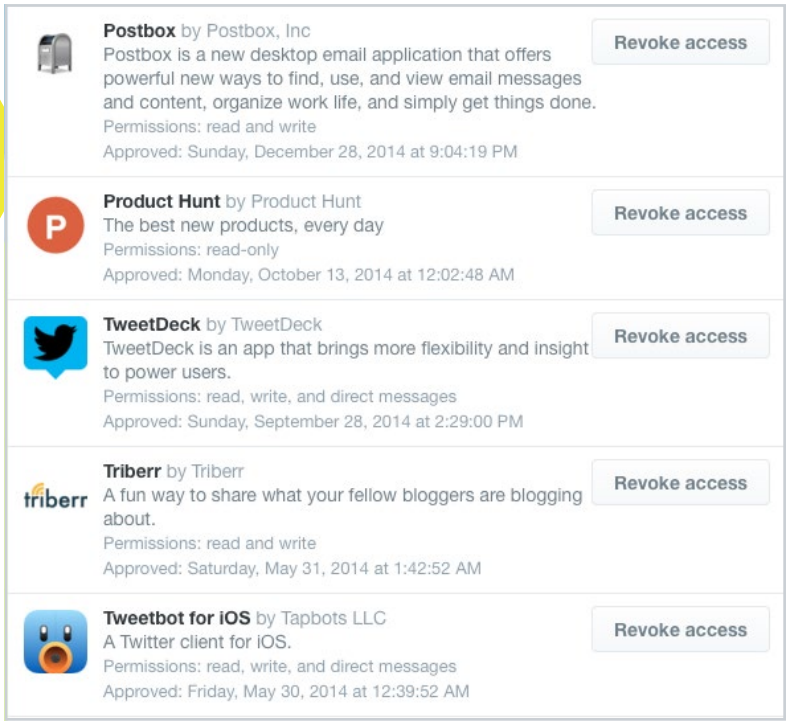


Source: Designing Social Interfaces

The password anti-pattern is insecure because you're telling a third-party site your password for a different site (also known as a "static shared secret") and, the more you share it, the less it's a secret. Also, consider apps like Mint, which enable periodic data imports that are based on storage of user names and passwords. Despite the convenience, ease, and time economy, propagating login credentials all over the Web opens a worrisome security loophole. In the case of banking apps, a data breach could cause devastating loss of valuable assets.

Two Important Protocols, One Caveat

In time entered the OAuth protocol, which is replacing the password anti-pattern. Another standard, OAuth-based OpenID Connect, added a lightweight yet robust federated single-sign-on feature for the modern era. With OAuth and OpenID Connect, social logins become feasible, meaning that you can access apps with an identity already established through popular services, such as Twitter, as illustrated below. Justin Pirie's post on social logins sheds more light on this subject.



In time, that list of apps grows and risks and inconveniences abound. A case in point: You might inadvertently authorize access to rogue apps. How to revoke authorizations can be tough to figure out because the interfaces on those services vary, let alone that you must revoke authorizations service by service. A one-stop-shop solution would be a godsend.

And, what about sharing between people? We don't often stop to think about it, but apps such as Flickr, Google Docs, and Tripit contain natural examples of person-to-person sharing. For instance, with the popular Share button in Google Docs, you can share with a collaborator a spreadsheet or document for editing or just viewing. The share paradigm works like a "push" version of a consent dialog box, which offers a powerful perspective on online privacy.

Those apps, however, rely on a proprietary authorization layer, which requires separate development efforts because OAuth and OpenID Connect do not deliver such a solution out of the box.

Consistent Security and Control From UMA

For bona fide security and privacy in information sharing, individual control is nonnegotiable. Hand in hand with control is choice, the result of which reflects human judgment. In this context, users would be well served with opportunities and interfaces not only to smoothly authorize or reject sharing, but also to make the right choices at convenient and appropriate times. And developers need standard architectures for building open API and application ecosystems for users.

In 2009, Eve Maler, now ForgeRock's VP of Innovation and Emerging Technology, founded the UMA Work Group at the Kantara Initiative and has been chairing it since. In March 2015, the UMA V1.0 specifications were finalized as Kantara Initiative Recommendations, the highest level of technical standardization Kantara Initiative can award.

Resource-Sharing Scenarios

Thanks to the standard UMA APIs, a helpful feature like the Share button can do wonders. In UMA nomenclature, the sharing party is called the resource owner; and the person with whom that owner shares data, the requesting party. UMA supports the three scenarios of authorized resource sharing that exist today:

- From the resource owner to himself or herself, as in the case of from Alice to herself. This scenario is similar to how OAuth and OpenID Connect work today.
- From the resource owner to someone else (a human requesting party), as in the case of Alice sharing her salary information with Bob.
- From the resource owner to an entity (an organizational requesting party in the form of an autonomously running Web service), as in the case of Alice sharing her whereabouts to take advantage of merchandise discounts offered by companies according to geographical location.

Furthermore, the resource owner can store the policies for sharing in one place but have resources live all over the Web. Centralizing user management of access in that manner is simple, intuitive, and easy to implement.

Configurations by Administrators

While deploying the authorization server of the upcoming release of OpenAM with the UMA-enabled feature, administrators can configure the settings in the administration console.

For example, see the details on how to set up a UMA provider.

In addition, administrators can configure the following:

- **Realm-based service**
  - Partitioning of user communities. In other words, the service that exposes an API (which UMA calls a resource server) and the client application that a requesting party is using have different user bases.
  - Partitioning of application resource sets. For example, the resource server and client application might use different resource realms even if they share the same user data store.
- **Data stores for resource sets and audit history**

Choices for Resource Owners

In a future release, resource owners can specify these conditions in an authorization:

- How long the authorized access is valid for.
- Whether the authorized party must perform a two-step authentication.
- Whether certain resource scopes apply to the requesting party, for example, whether Bob, the accountant, can view and also edit the books for a particular account.

Common scenarios might include the following:

- **For your tax accountant to view your bank account:** For example, verify that the requesting party's email address is TP1234@gmail.com and that the request originated from TaxHelp. Cancel the authorization on April 15, 2015.
- **For two health consultants to view the data generated by your scale and fitness wear:** For example, verify that the requesting party is either Health Aggregation or Live Oaks Care. Limit the access period to three months from the date of initial authorization.

In addition to preauthorizations, resource owners can share on request, such as when someone asks to view a protected Google document. All it takes is an API that supports that capability. Resource owners can cancel authorizations any time.

The Horizon Ahead: User-Controlled Privacy and Consent

UMA-enabled authorization and authentication are poised to play an important role in according peace of mind when you share proprietary information, whose security is doubly ensured with your prior consent. Applicable are numerous use cases in many sectors—health care, business or personal finance, education, government, media—across the Web, mobile devices, IoT, and APIs.

Do you have comments, ideas for enhancements, or suggestions of use cases? Get involved with ForgeRock's OpenUMA Community! Experiment with the open-source code and preliminary product features and let us know what you think. We appreciate your feedback as we work toward delivering user-controlled privacy and consent in OpenAM through UMA.

SAN FRANCISCO +1-415-599-1100
VANCOUVER +1-360-229-7105
OSLO +47-2108-1746
BRISTOL +44-1935-804797
GRENOBLE +33-625-14-96-92
LONDON +44-20-3598-4786
SINGAPORE +65-6709-5705