EmID: Web Authentication by Email Address

Ben Adida[*]

Abstract

We suggest that OpenID should use email addresses rather than URLs as identifiers, and show how OpenID can be adapted accordingly with relative ease. Email addresses provide better backwards compatibility with existing web authentication paradigms, map to existing trust decisions more closely, and provide a smoother, less intrusive adoption curve. Of particular interest to privacy advocates, a username-and-domain approach to web identifiers lends itself more naturally to cryptographic authentication protocols where the identity provider need not learn its users' every authentication event.

1 Background and Proposal

OpenID [7] is an open standard for web-based single-sign-on. A user's identity is a URL, and the OpenID login process proves that the user controls the content of the web page at that URL:

- Alice enters her OpenID URL to log in to a third-party web site.
- The third-party site fetches her OpenID URL, looks within its content for the appropriate metadata that indicates which OpenID identity provider to use, and redirects Alice's browser to the identity provider.
- Alice authenticates with the identity provider.
- The identity provider redirects Alice's browser to the third-party web site along with an authentication token that the third party can verify.

Thus, if Alice controls the content of the web page at a given URL, she can authenticate to an OpenID-compatible service as the owner of that URL. The URL becomes, in effect, her identity.

OpenID is generally marketed as the solution to low-security applications, such as blog commenting, online project management systems, etc. There are significant phishing concerns related to the plain implementation of OpenID [3, 6], though those can be addressed at the identity provider level with browser extensions or the use of client-side certificates.

1.1 The Problems with URLs as Identifiers

The OpenID protocol is particularly lean and flexible: the specifics of authentication between a user and her identity provider are left open, and the use of URLs and HTTP makes for easy cross-platform implementations. Unfortunately, this use of URLs can be problematic, too.

[*]Center for Research on Computation and Society, Harvard University / Children's Hospital Boston, Harvard Medical School. Email: `ben_adida@harvard.edu`.

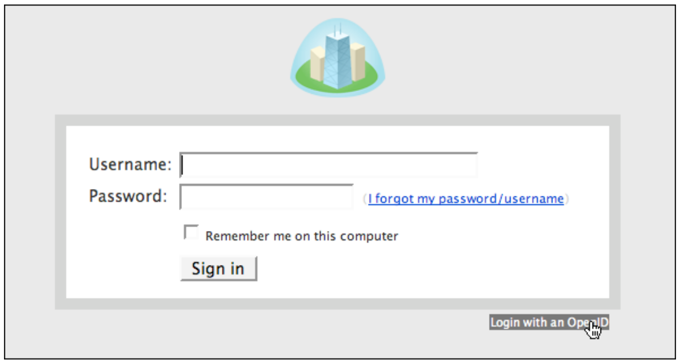

Figure 1: Logging in with OpenID: users are presented with a bifurcated login screen, because an OpenID identifier is not the same as a normal username. Users must be aware of OpenID. (Here at `37Signals.com`.)

A different login paradigm and process. Most users are not familiar with using a URL as their login "name". Though URLs-as-usernames is a paradigm that programmers and web experts may understand, it is particularly difficult to explain to a non-technical person who thinks of a URL as a way to fetch a document. And because URLs look nothing like existing identifiers, the login process for OpenID is forcefully bifurcated: at many sites, the login page forces users to choose between a "normal" username and an OpenID, as shown in Figure 1. It is worth noting that some tools, e.g. the Sxipper toolbar [1], abstract out the OpenID process for users, thus making it less confusing. However, if a user ever needs to use a browser without the specific extension, they are still required to know their single-sign-on URL identity.

An extraneous player. In the OpenID world, users are forced to think of a new player in their everyday online life: the identity provider. When signing up for an OpenID, the user must choose: which identity provider should I trust with the keys to all of my web accounts? If the answer is "use multiple providers," the problem becomes worse: users must negotiate the mappings between multiple web accounts and multiple OpenID providers. Though confusion may be alleviated as well-known companies begin to provide OpenID services, the decision to "go OpenID" remains particularly mysterious for the average Web user.

No messaging. An integral part of online authentication is messaging: web sites often need to contact their users for various purposes, and, in the case of OpenID, identity providers may well want to contact their users regularly, too. URLs provide no mechanism for messaging the user privately. To fill this gap, most OpenID providers simply request the user's email address, and sites that use OpenID for authentication often expect to receive the email address through the OpenID attribute exchange protocol. If they do not, third-party sites often extend the registration process by asking for and independently verifying the user's email address, typically via mailback. A process that was supposed to be made simpler by OpenID thus becomes more complicated than it was before OpenID.

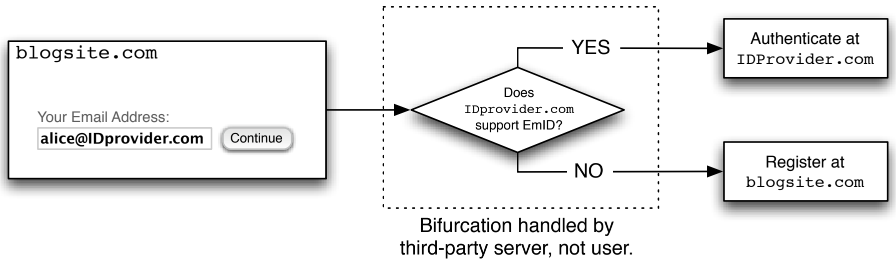
blogsite.com
Your Email Address:
alice@IDprovider.com
Continue
Does IDprovider.com support EmID?
YES
NO
Authenticate at IDProvider.com
Register at blogsite.com
Bifurcation handled by third-party server, not user.

Figure 2: Logging in with EmID: Alice is presented with a *unified login experience.* The rest of the work, in particular the process bifurcation, is done by the third-party site (`blogsite.com`) and her email/identity provider (`IDprovider.com`).

1.2 Using Email Addresses as Identifiers

Consider, instead, OpenID implemented differently: identifiers are email addresses of the form `user@domain`. We'll call this system EmID. Domains that support EmID may advertise this fact in a DNS SRV record [5] or, if a domain is unwilling to add a DNS entry, by convention at a given URL, e.g. `https://emid.domain`. Then, users have one simplified path for logging in: provide a valid email address. If the domain supports EmID, the third-party web site can simply redirect the user to the identity provider, as specified in the DNS SRV record, for authentication. If the domain does not support EmID, the third-party web site can perform the usual mailback verification it would likely perform anyways: send an email to the user with a secret token embedded in a confirmation links. In other words, with EmID, the onus is on the web site to do the authentication work, not on the user to understand the separate mechanisms for authentication (see Figure 2.)

2 Advantages of EmID

Existing uses of email addresses as usernames. Numerous web sites already make use of email addresses as identifiers. For example, a Google account [2], which comes with a calendar, a newsreader, an online document editor, and other applications, is associated with one (or more) email addresses, and the user logs in with an email address as username. Thus, the user experience for registering and logging in at web sites that support EmID would change very little from what they know today: users are always prompted for an email address as username.

Users already trust their mail servers. Existing web sites already assume that users trust their email servers: forgotten credentials are recovered by "mailback," where a password-recovery link is sent to the user's email address on record. It is sensible to expect that email servers would become EmID identity providers, and users would have much less room for confusion: their email address *is* their identity, and their choice of email provider includes identity services. Even the idea of having multiple identities/personas online maps nicely to email addresses, since users already handle the management of multiple email addresses for work, home, etc.

[1]`http://sxipper.com`
[2]`http://google.com/accounts/`

Gradual adoption without user intervention. Domains can implement EmID progressively. When Alice registers with a third-party web site, the site can check if Alice's provided email address domain is EmID-enabled. If it is, she will be redirected to the single sign-on process, and if it is not, she will simply create a normal account with her email address as identifier. When her domain *does* end up adopting EmID, the third-party web site can easily notice and shift Alice's authentication process to the EmID method, without Alice having to intervene.

Messaging. An EmID identity provides an established messaging protocol: SMTP. This simplifies the registration process for web sites that require an email address which they often cannot obtain from an OpenID transaction. One potential criticism of EmID is that web sites now automatically get the user's email address. We do not believe this would be a significant impediment in practice, as users can generate secondary, pseudonymous email addresses for less-important web sites, the way they already do to limit spam. Because many web sites already require an email address at registration time, users would see little difference in practice.

Future Messaging Protocols. As new, alternative messaging protocols are implemented and deployed, the email address format `user@domain` can still serve as a messaging handle. It is encouraging to see that a number of existing instant-messaging systems have opted for this backwards-compatible approach, e.g. MSN Messenger, Jabber and Google Talk. As users move to messaging systems other than email, EmID can evolve accordingly with only server-side modifications: the user's identifier remains an email-address-like token, though the messaging protocol might vary.

Enabling better privacy. With EmID's domain-derived identifiers, it may be easier to implement cryptographic authentication protocols that do not require user-specific information exchange between the third-party web site and the identity provider. For instance, one could repurpose Lightweight Email Signatures (LES) [2], originally proposed for email authentication. In schemes like LES, a third party would only need to fetch the domain-level public key for verification, thereby ensuring that the identity provider doesn't learn about every login event. This level of privacy protection will likely become quite important as users realize the paradigm change in moving to mediated single-sign-on solutions, where the identity provider is involved in every login event.

3 Security Considerations

A shift from OpenID to EmID should be examined for potential security ramifications. In this section, we point out a few issues worth exploring further.

Phishing. In a plain browser, EmID will be just as vulnerable to phishing as OpenID: a malicious third-party site might behave as if it is redirecting the user to her identity provider while instead sending her to a look-alike phishing site. Identity providers for EmID will need to deploy defenses against phishing similar to those recommended OpenID, e.g. browser extensions or BeamAuth [1]. One particular case of phishing should be further explored in developing EmID: as the user will not be presented with a bifurcated login experience, she might be more easily tricked into entering her EmID password *on* the actual third-party site, rather than at her identity provider.

Pharming. As EmID may depend on DNS records, one should consider DNS attacks, often called "pharming" given their similarity to phishing. That said, for an attack to be successful, the third-party *server*, not the end-user, would have to be targeted. It is likely that a DNSSEC [4] solution would be preferable, but the risk here is no worse than with OpenID, where the third-party site must resolve the user's OpenID URL. If EmID uses DNSSEC, it will likely be more secure against server-side pharming than OpenID, which cannot ensure that all users' OpenID URLs are SSL-enabled (since the user chooses her identity URL.)

User Expectations and Behavior. Although users already treat email addresses as identifiers, it is unclear how they will react to a new login experience which first prompts them for an email address, then determines whether to prompt them for a password or redirect them to their identity provider (depending on whether their domains supports EmID). Examining user behavior when using email addresses as identifiers will be an important aspect of a full EmID deployment. Again, to the average web user, the login experience should be quite a bit more familiar than that of OpenID. That said, only a real user test will help determine exact usage patterns.

4 OpenID 2.0 and Beyond

Interestingly, the latest OpenID specification, v2.0, indicates that OpenID is moving away from using strict URLs are usernames. Specifically, with Yahoo's implementation of OpenID v2.0, users enter only "`yahoo.com`" as their identifier, at which point OpenID 2.0 can look up the Yahoo identity provider without yet knowing which user is about to be authenticated. While this feature of "directed identity" is meant primarily as a means to limit information disclosure to the third-party site, this approach also hints that Yahoo believes users shouldn't have to manage a URL identifier. And, since the third-party web site will almost certainly ask the user for an email address anyways, this change is a minuscule step away from using the actual email address, `user@yahoo.com`, as the login identifier.

However, while OpenID 2.0 simplifies the user experience a bit, it remains fundamentally more complicated than EmID. The login process is still bifurcated, and mediated-login privacy remains a concern: a redirect to the identity provider is still required to perform authentication.

5 Conclusion

We propose that, while the OpenID paradigm is powerful and stands to simplify web logins, it would be much better if it used email addresses as identifiers rather than URLs. OpenID v2.0 shows that there is awareness, within the OpenID community, that URLs are not necessarily the best identifiers. It would be useful if the OpenID community continued further down this path. Email addresses may seem antiquated, but they are the ultimate established user identifier. A system like EmID would provide a more backwards compatible user experience and deployment plan, not to mention additional privacy options for those who want them. We hope to use this blueprint to prototype a system like EmID in the near future.

6 Acknowledgments

The author wishes to acknowledge the insightful anonymous W2S&P 2008 reviewers, as well as Rachna Dhamija and Adam Barth for encouraging this line of thinking.

References

[1] Ben Adida. BeamAuth: Two-Factor Web Authentication with a Bookmark. In *CCS 2007, Proceedings of the Fourteenth ACM Conference on Computer and Communications Security*, October 2007.

[2] Ben Adida, David Chau, Susan Hohenberger, and Ronald L. Rivest. Lightweight email signatures (extended abstract). In *Fifth Conference on Security and Cryptography for Networks (SCN'06)*, volume 4116 of *Lecture Notes in Computer Science*, pages 288–302. Springer Verlag, 2006.

[3] Kim Cameron. As simple as possible – but no simpler. `http://www.identityblog.com/?p=649`.

[4] D. Eastlake. RFC 2535: Domain Name System Security Extensions, March 1999.

[5] A. Gulbrandsen, P. Vixie, and L. Esibov. A DNS RR for specifying the location of services (DNS SRV). `http://en.wikipedia.org/wiki/SRV_record`.

[6] Ben Laurie. OpenID: Phishing Heaven. `http://www.links.org/?p=187`.

[7] D. Recordon and B. Fitzpatrick. OpenID Authentication 1.1, May 2006. `http://openid.net/specs/openid-authentication-1_1.html`.