Yahoo! OpenID and OAuth

Allen Tom
Yahoo! Membership Architect
OpenID Foundation Board Member
atom@yahoo-inc.com
• UC Berkeley BA CS – 1996
• Netscape 1997-1999
  – LDAP, Netscape Directory Server
• AOL 1999-2005
  – Online shopping, browser assistants, web toolbars, streaming media
• Yahoo 2005-Present
  – Membership Architect
    • Account lifecycle, Authentication, Security, Abuse
Yahoo! Membership

– Over 350 million active users
  (unique users who have signed into the Yahoo website in the last month)

– Account Lifecycle:
  • Login
  • Registration
  • Account Recovery
  • Customer Care tools
  • Suspension / Deletion

– Security, Abuse

– Single Sign On (OpenID, SAML)

– Authorization (OAuth)
Yahoo! Login: 350 million unique users monthly
Yahoo! Registration
Yahoo! Login: Devices and rich apps

Yahoo! Mail for iPhone

Yahoo! Instant Messenger
Profile Information
Edit profile details

Contact Information
Update your contact information
Choose how Yahoo! contacts you

Allen Tom
Male · January 01, 1990 · San Francisco, California

Yahoo! Email
allentomtest@yahoo.com

Home Email
atom@yahoo-inc.com

Home Address
1 Market St, San Francisco, CA 94133
• OpenID – Authentication
  – “logging in” to a website
  – Share basic profile information using Attribute Exchange
    • Name, profile picture, email address

• OAuth – Authorization
  • API access
  • Needed to call web services on behalf of the user
Yahoo Membership and the Open Web

• OpenID: Opens Yahoo’s Membership platform to all websites
  – Users who have a Yahoo Account can log in with it at any website that accepts OpenID

• OAuth: Authorization protocol (access control) for Yahoo Data and APIs
  – Contacts (Address Book)
  – Yahoo Mail
  – Photos
Yahoo OpenID + OAuth

• Yahoo users can sign into websites using their Yahoo ID via the OpenID Protocol
• Users can authorize data access via OAuth
  • Share your Yahoo Address Book
  • Let the 3rd party update your Status
  • Upload photos
OpenID: Authentication

- OpenID is for “logging in” to a website
- Use your Yahoo/Google/AOL account to log into any website that accepts OpenID
- OpenID lets users prove ownership of a userid
OpenID: History

• OpenID originally invented to allow bloggers to comment on each other’s blogs
  – Too much friction involved in registering a new username/password for every blog you comment on
  – OpenID 1.0 finalized in 2005 by grassroots community
  – OpenID 2.0 finalized in December 2007
• OpenID Foundation is the custodian of OpenID intellectual property
• OpenID is an Open Standard: free to implement
• http://openid.net
OpenID: Authentication

• OpenID
  – Supported by Yahoo, Google Accounts, Google Apps, AOL, MySpace. Microsoft/MSN LiveID (in alpha)
  – Yahoo OpenID support launched in January 2008
  – Allows users to login to websites that accept OpenID

• Decentralized
  – No central authority
  – Anyone can setup an OpenID Provider
• My YahooID is allentomdude@yahoo.com
• My OpenID identifier is https://me.yahoo.com/allentomdude
• OpenID lets me prove that I control https://me.yahoo.com/allentomdude
Yahoo OpenID Example

Log in with OpenID

Click your OpenID account provider:

Google
Yahoo!
myOpenID
AOL

Or, manually enter your OpenID URL:

http://yahoo.com/

Login
Click "Agree" to sign in to stackoverflow.com using your Yahoo! ID and allow sharing of Yahoo! info.

**You are sharing the following:**

**Allen Tom**
allentomdude@yahoo.com

**Select your OpenID identifier:**
[https://me.yahoo.com/allentomdude](https://me.yahoo.com/allentomdude)

**Agree**

By clicking Agree you are agreeing to the Yahoo! Additional Terms of Service.
Stackoverflow.com is asking for some information from your Google Account allentomdude@gmail.com

- Email address: allentomdude@gmail.com

[Allow] [No thanks]

[✓] Remember this approval
Log in with OpenID

Click your OpenID account provider:

- Google
- Yahoo!
- myOpenID
- AOL

Or, manually enter your OpenID URL:

allentom.com

Login
Alternative “popup” UI
• Single Sign On using Open Standards
  – Anyone is free to implement the protocol
• Login to websites using existing identities – intended to streamline or even eliminate website specific registration
  – No need for users to create a site-specific userid or password
• Support from Yahoo, Google, AOL, MySpace, PayPal
  – Facebook and Microsoft are active contributors to the core specifications
• OpenID Provider (OP)
  – The site where you have your account (Yahoo)
• Relying Party (RP)
  – the site that you’re logging into (Stackoverflow)
1) User clicks on the Yahoo! Button
2) Stackoverflow redirects browser to Yahoo’s OpenID Endpoint
3) User authenticates at Yahoo and approves the authentication request
4) Yahoo redirects browser back to stackoverflow with an assertion containing the user’s identifier
5) Stackoverflow makes a direct server call back to Yahoo to verify the assertion
6) Stackoverflow sets browser session cookies. The user is now signed into stackoverflow
OpenID Authentication

- User selects their OpenID Provider (Yahoo)
  - The user clicks on the Yahoo Button
  - Alternatively, the user can type “yahoo.com”
- Relying Party requests the OP to authenticate the user
  - The RP Performs Discovery on [http://yahoo.com](http://yahoo.com) to find the OpenID Endpoint
  - The RP redirects the user’s browser to the OpenID endpoint with an Authentication Request
- The OP responds with an assertion
  - The OP redirects the browser back to the RP with an assertion
  - The RP verifies the assertion by making a direct request back to the OP
Interesting OpenID Features

• OpenID Identifiers are URLs
  – https://me.yahoo.com/allentomdude
  – http://allentom.com
  – No standard way to transform an email address to an OpenID URL (there are several proposals)

• OpenID Discovery
  – Fully automatic discovery of OpenID endpoints and capabilities
  – No metadata files
  – RPs and OPs can interop without any manual setup or configuration

• OpenID Attribute Exchange
  – Share user attributes (name, email address, profile picture, etc)
User Benefits

• Faster & Easier registration and login
• Portable Profile, interests, contacts
• Personalization (even on the first visit)
• Social Filtering
• Better Security
Why is Yahoo supporting OpenID?

• Have a stronger relationship with our users
  – Users are Yahoo’s #1 asset

• Yahoo IDs are more valuable – used for logging into Yahoo and other websites

• More insights into user behavior on Yahoo and everywhere else
  – Needed for ad targeting and content personalization

• Open Standard:
  – No need to invent yet another auth protocol
  – Can leverage industry best practices
  – Open Source libraries, documentation
  – Developers can implement the same interface across all Ops
    Yahoo/Google/AOL are almost completely interoperable
Why should sites accept OpenID?

- New user on boarding experience is getting increasingly difficult
  - Username/password
  - Name/email address
  - Profile Picture
  - Location
  - Gender
  - Friends
  - CAPTCHA

- Security, Abuse, Account Recovery can be outsourced to the OpenID Provider

- New users already have a reputation
  - Abuse, expertise, etc

- Content and Ads can be personalized and relevant even on the first visit
Attribute Exchange

- RPs may optionally ask for user data via the Attribute Exchange Extension (supported by all major OpenID Providers)
  - Name
  - Email Address
  - Profile Picture
  - Age
  - Gender
  - Location
• Instead of exchanging metadata files, OpenID defines a discovery mechanism for RPs and OPs to publish metadata about themselves.
OpenID Discovery on Yahoo

$ curl -L --head \
   -H "Accept: application/xrds+xml" \
   http://yahoo.com

HTTP/1.1 200 OK
[....]
X-XRDS-Location:

Yahoo OpenID Provider metadata
• OpenID endpoint URL
• Version
• Supported Extensions
Interoperable Profile via OpenID

• Same interface can be used with Yahoo, Google, AOL, and Microsoft LiveID – Nearly 90% of the internet population
  – Name
  – Email address
  – Profile URL
  – Profile Pic
  – Gender
  – Zipcode

• The same code can be used for all OpenID Providers
OpenID Security

• Implementations can be buggy
• No standard test suite
• Issues with outsourcing authentication
• Orphaned accounts
• Phishing
• Privacy – correlation
• No Single Log Out
No Single Log Out

• Authentication sessions at the OP and RPs are not synchronized
• By design – logging out of Huffington Post should not log the user out of Yahoo Mail
• Difficult to synchronize logout events across multiple RPs
• Very controversial topic: Most OPs are strongly against it
• Sites that accept OpenID effectively outsource authentication to the user’s OpenID Provider
• The OP can be buggy or insecure
• Employees at the OP may be able to login as any of their users
Losing access to your OpenID results in losing access to all sites that you use

– Account compromised – the attacker can gain access to all your accounts
• Most websites will allow passwords to be reset via email

• This means that if your email address is hacked, the attacker can gain access to all the sites you use that allow email based account recovery

• Same thing as having your OpenID compromised
Many websites allow Account Recovery via email.
Outsourced Account Recovery to the user’s Email provider.
• Almost all consumer oriented websites allow passwords to be reset via email
  – Some sites will require the user to answer a secret question before resetting the password via email
  – Secret questions are not secret and don’t work
• Resetting your password is the same thing as logging in
• Therefore, from a security perspective, sites that allow email based account recovery already have outsourced their authentication to the user’s email provider
OpenID Challenges

• UI is confusing
  – “signing in with an account you already have” is a new concept
  – Users assume some sort of business relationship between the OP and RP
  – The OpenID Login Call to Action is not scalable past 2 or 3 choices
More challenges

• What if the user loses access to their OpenID?
  – Account was hacked/phished
  – The user graduated, changed jobs, etc
  – Deactivated for Terms of Service violations
  – User can’t remember/recover password
  – No support for client apps

• Only supports logging in to websites
  – Doesn’t work for logging into apps (desktop, mobile, etc)
    • We’re working on it!
Too Many Choices

OpenID

HuffPost Social News

Already have an account?
Enter your HuffPost username and password.

ENTER USERNAME

ENTER PASSWORD

Sign in

Forgot your Username or Password?

Sign up now!

Connect with Facebook

Sign in through Yahoo!

Sign in with Twitter

Join this site

Google Friend Connect
The OpenID NASCAR Problem
Potential Solutions

• 2 step login:
  – Enter email address
  – If the user has a password, display the password field
  – Otherwise, if the domain is OpenID enabled, send the user through the OpenID flow
    • user@yahoo.com goes to Yahoo
    • user@gmail.com goes to Google
UI Issues

• Plenty of usability testing in progress behind the scenes
• Many top designers are working on this
Questions?

• OpenID is an Open Protocol
  – Fully transparent development process
  – Follow us on http://openid.net
    • (check with your IP lawyers first)
    • All contributors must have their employers sign an IP contribution agreement

• Internet Identity Workshop
  – May 18-20
  – Mountain View, CA
  – Everyone working on OpenID will be there
OAuth

• Authorization (not Authentication)
• The “Access Control” layer of the Open Web
• Sometimes called the Valet Keys of the web
• OAuth is needed to authorize API access
  – OpenID does not pass any credentials to the RP
OAuth – Valet Keys for the Web

- OAuth allows users to grant credentials to their account, without giving up their password.
- Larger Service Providers usually issue scoped credentials.
- Credentials can usually be revoked.
• Many sites ask for your password so that they can access the data that you have on other sites
Sharing passwords is a bad idea….

- **Passwords are not scoped**
  - Instead of authorizing data sharing for a specific resource, you’re giving up access to your entire account

- **Most users don’t use different passwords for different sites**

- **No way to revoke access without changing your password**

- **Even if the site is honest, mistakes happen**
  - Extensive logging – passwords get saved to log files
  - Security compromises
Why OAuth?

• Prior to Oauth, most service providers defined proprietary delegated auth protocols

• OAuth combined the best practices of Yahoo BBAuth, Google AuthSub, Flickr Auth, Windows Live Delegated Auth
  – All of them did the same thing, slightly differently
The 3 Legs of OAuth

User: Jane

Service Provider: faji

Consumer: beppa
Hi Jane,

You have uploaded 2 photos:

[Images of uploaded photos]

Share with Friends
oauth_flow_begins
user_authentication
beppa.com is requesting access to your private photo albums:

If approved, beppa.com will have read only access for the next 1 hour

Approve  Deny
get_protected_resources
oauth_success
• Consumer – The application (client)
• Service Provider – The site that the consumer is trying to access
• Protected Resource – The service that the consumer is accessing (typically a Web Service API)
• 3 Legged Oauth
  – User
  – Consumer
  – Service Provider
• 2 Legged Oauth
  – Consumer
  – Service Provider
OAuth Terminology

- Consumer Key – Client identifier, equivalent to AppID or API Key in other protocols
- Consumer Secret – shared secret used to authenticate the Consumer
- Access Token – Credential used by the Consumer to access Protected Resources
- OAuth Dance – Browser gymnastics where the browser bounces between the Consumer to the Service Provider and back for the user to approve an Access Token
- Oauth WRAP – Next generation OAuth
• Application developers usually must go to the Service Provider to register for a Consumer Key
  – Process for obtaining a Consumer Key is Service Provider specific and is not defined in the OAuth Protocol
  – SPs usually require developers to register to agree to Terms of Service, as well as to satisfy the SP’s legal, business, and product requirements
Yahoo OAuth Consumer Key Registration

So, you want to use some Yahoo! APIs...

We need some information from you... To use Yahoo! Web Services, we need some information about you and the application you're building, understanding of how Yahoo! Web Services are being used and to protect the security and privacy of Yahoo! users. All questions are required.

About your application

Application Name: [ ]
Kind of Application: [Web-based]
Description: 250 characters or less
Application URL: [http://www.yoursite.com/thisApp/welcome.html]
Favicon URL: [http://www.yoursite.com/thisApp/favicon.png]

Optional; specify a URL to a 16x16 JPG, PNG or GIF image.
Yahoo Consumer Key Registration

- Application name, description, logos
- Developer name, contact information (email, phone number, website)
- Terms of Service agreement
Invite Your Friends

From: Allen Tom

To: (use commas to separate emails)

Message: (optional)

Invite Your Friends

Web Email (Hotmail, Gmail, Yahoo, etc.)

Invite contacts from your email account.

Your Email: allentomdude@yahoo.com

Find Friends

Facebook will not store your password. Learn More.
OAuth Protocol – Request Permission

1) Consumer gets a Request Token

Facebook

browser

Yahoo!
OAuth Protocol – Request Permission

1) Consumer gets a Request Token

2) Consumer redirects browser to Yahoo OAuth server with request token
Link your Yahoo! Account with Facebook

Yahoo! Facebook

Share your Yahoo! info with Facebook
Share your Facebook activities and info with Yahoo!

What am I sharing?

Please verify your password

Yahoo! ID: allentomdude
Sign in as a different user.
Password:

Sign In Cancel

By signing in, you agree to the Yahoo! Additional Terms of Service

Copyright © 2010 Yahoo! Inc. All rights reserved. Copyright/IP Policy | Terms of Service | Guide to Online Security To learn more about how we use your information, see our Privacy Policy
User Agrees

1) YOAuth redirects browser back to consumer to indicate successful authorization.
User Agrees

1) YOAuth redirects browser back to consumer to indicate successful authorization

2) Consumer exchanges Request Token For Access Token/ATSecret

- The Access Token is not returned on the redirect back to the consumer, instead the consumer must make a backend server-to-server call to retrieve the Access Token
- Enables Token to be exchanged over HTTPS
Yahoo OAuth implementation

1) Exchange Request Token for Access Token and Access Token Secret

2) Request user’s Address Book
   CK, AT, sig(CS, ATS)

Facebook

Yahoo! ADDRESS BOOK

OAuth Service
OAuth Deployment Challenges

- Scalability
  - Token Revocation: Database lookup required to determine if AT is still valid
  - Very difficult to scale for globally distributed web applications
  - Heterogeneous and distributed production environment
  - Security issues
    - Consumer Secret has to be widely distributed to all service providers
    - Address Book needs the consumer secret to verify the request
      - Very difficult to properly safeguard the consumer secret
    - Access Token (Permanent credential) directly handled by Service Providers
      - Issues with security compromises on Service Providers
- The Yahoo OAuth Service is much more secure than the rest of Yahoo
  - Closely monitored and audited
  - Allows the rest of Yahoo quickly deploy new services
• Yahoo’s proprietary auth protocol uses short-lived (1 hour) bearer tokens to access protected resources

• Service Providers can verify the bearer token locally without a database lookup
  • Bearer tokens are signed by the Auth service

• Persistent credentials (consumer secrets, Access Tokens) are not used to access any services
Deployment Issues with OAuth at Yahoo

- Oauth signatures are nearly impossible to generate without a library
- Libraries are buggy too
- #1 problem is that developers can’t generate signatures

2) Request user’s Address Book
CK, AT, sig( CS, ATS)
• OAuth signatures are used to protect against replay attacks
• Why not just use HTTPS?
• Data is flying around in the clear
• Does not match “Cookie” auth for Browsers
  • All protected resources can be accessed via the user’s browser
  • Browsers are far more susceptible to MITM than web applications
• OAuth is so difficult that most developers end up phishing the user for their password instead

2) Request user’s Address Book
   CK, AT, sig( CS, ATS)

Yahoo! Address Book

Consumer
OAuth WRAP

• WRAP == Web Resource Authorization Protocol
• Started at Internet Identity Workshop in May 2009
• Contributors:
  – Yahoo
  – Google
  – Microsoft
  – Facebook
• Goals
  – Make it easier for developers to use
  – Split OAuth’s Service Provider into 2 parts:
    • Token Issuer – Can be a completely separate entity
    • Service Provider
  – Define different “Profiles” for different use cases
    • Web Application Profile
    • Rich client app Profile (no browser)
WRAP Profiles

• WRAP defines several mechanisms for users to authorize an application
• Web App Profile
  – User wants to authorize a website
  – User experience identical to Oauth
• Username/Password Profile
  – User wants to authorize a rich client application
  – No browser is required
• WRAP Profile only define different mechanisms for clients to obtain an Access Token
• All Access Tokens are used the same way regardless which Profile was used to obtain them
• Service Providers do not need to care how the Access Token was issued
• Consumers are issued an Access Token with a 1 hour lifetime, and a Refresh Token

• Consumers access Protected Resources using the Access Token. No Signatures are required.
  • Passed in the HTTP Authorization: header

• If the Access Token is expired, the Consumer can request a new Access Token from the Token Issuer using the Refresh Token
OAuth WRAP Example

- `$ curl -header \\
  'Authorization: [WRAP] access_token="access token"' \\
  http://api.example.com/get_data`

HTTP/1.1 200 OK

[...]

Data
Expired Access Token Example

• $ curl -header 'Authorization: [WRAP] access_token="access token"' \ 
  http://api.example.com/get_data

  HTTP/1.1 401 Unauthorized

• $ curl -d wrap_refresh_token=refresh_token \ 
  https://auth.example.com/refresh_token

  HTTP/1.1 200 OK
  access_token=access_token2

• $ curl -header 'Authorization: [WRAP] access_token="access token2"' \ 
  http://api.example.com/get_data

  HTTP/1.1 OK
  data
• $ curl --header \n  "Authorization: [WRAP] AccessToken" \n  http://api.example.com/get_data

HTTP/1.1 200 OK
[...]
Data
WRAP Advantages

• Service Providers can verify credentials without a DB lookup
• Consumer Secrets and Persistent credentials do not need to be distributed to Service Providers
• Developers can just curl the requests
• Rich client apps do not need to use a browser
• Service Providers and Auth Servers can be separate entities – Assuming they can agree on the token format
  – Simple Web Token (SWT) Spec
Deploying WRAP at Yahoo

- WRAP Protected Resources only need to know how to validate a WRAP Access Token
  - Access Token contains
    - UserID
    - Scope that was authorized (eg: Address Book)
    - Consumer Key
    - Expiration time
    - Signed by the Yahoo Auth Server
  - Services don’t need to worry about determining if the user or the consumer key is valid
    - The Yahoo Auth Server verifies that both the user and the consumer are still valid before issuing the Access Token
    - Cons: it can take an hour to revoke access to an app or to deactivate a user
Why does Yahoo support OAuth?

• Open standards make it easier for developers to use our APIs
  – Reuse experience, documentation, and expertise across Service Providers

• Leverage industry best practices
OAuth / OAuth WRAP questions?

• Google Groups
  – OAuth
  – OAuth-wrap-wg

• Internet Identity Workshop
  – Same group of people who work on OpenID
  – OpenID, Oauth, and Oauth-WRAP all resulted from discussions at IIW
What’s next?

• OpenID+OAuth are the building blocks of the next generation of internet identity
  – Eliminating login/registration friction can eliminate the Anonymous Web
    • Users must be recognized to serve optimized content and ads
• Over 60 million unique users login to websites using Facebook Connect
  • Proprietary solutions are often better, but users and the market demand open and interoperable standards
• Nearly 100% of all internet users have an account that can be used to sign into other sites
  • Yahoo/Google/Google Apps/AOL/Hotmail/Facebook/Twitter
• Site specific registration will become a thing of the past
Allen Tom
atom@yahoo-inc.com

http://openid.net
http://groups.google.com/
  –OAuth
  –OAuth-WRAP-WG
http://www.internetidentityworkshop.com/