Outline for Today

- OpenID Connect and OAuth2 protocol overview
- MITREid Connect open source project
- Use cases for OpenID Connect
OpenID Connect

and OAuth2

A protocol overview
Delegated Authorization

- How can a resource owner authorize a piece of client software to access protected data on their behalf?
- How can we scale security decisions to work on the internet?
- How can we make a system that’s both secure enough for data protection and usable enough for average end users?
The Triangle of Nonlocal Auth

Front Channel: HTTP Redirect through browser

End User (and user agent: UA)

Back Channel: direct HTTP connection

Client (“Relying Party”: RP)

Service Provider (“Identity Provider”: IdP)
OAuth

- Started in late 2006 to connect sites using OpenID for logins
- Version 1.0(a) standardized in IETF: RFC5849
- Version 2.0 modularized concepts, added explicit extensibility, and removed major pain points of 1.0
  - Standardized in IETF: RFC6749, RFC6750
  - Continued extension development today
OAuth 2

Temporary credential, user authorization

Client credential, tokens, API
The Authorization Code Flow

The most common OAuth2 Pattern
The players

Resource Owner & User Agent

Authorization Server

Client

Protected Resource

©2013 The MITRE Corporation
The goal:

Connect the Client to the PR
End-user initiates Client action

UA

AS

C

PR
Client redirects User to AS

UA

AS

C

PR
User authenticates to AS
User authorizes Client

UA

AS

C

PR

©2013 The MITRE Corporation
AS issues Authorization Code

UA

AS

C

PR

©2013 The MITRE Corporation
AS redirects User to Client
Client sends code to AS
AS issues token(s)

UA

AS

C

PR

©2013 The MITRE Corporation
AS issues token(s)
Client accesses PR

UA

AS

C

PR

©2013 The MITRE Corporation
What is OAuth used for?

- Avoiding password proliferation
  - User’s credentials never go to the client
- API protection
  - Hundreds of thousands of sites, projects, and systems ... and growing
- Mobile access to server systems
- Authentication (sign-on) protocols
  - Facebook Connect, Log In With Twitter, etc.
What OAuth tells you

- Somebody authorized the Client
- Hints at what the client is authorized to do (scope) and for how long (expiration)
What OAuth doesn’t tell you

- Who the user is
- If the user is still there
- What to call the user
- How the user authenticated in the first place
How to use OAuth to log in:

- Create an identity API, protect it with OAuth
  - Authorization Server becomes Identity Provider
  - Client becomes Relying Party
- Standardized user info coming from this API
  - Name, email, picture, etc.
- Session management
  - Is the user still logged in?
  - Request log out
- Communicate authentication information
  - Step up to high levels of authentication
- Keep compatibility with basic OAuth2
OAuth2 + Identity API = OpenID Connect
OpenID Connect (OIDC) is built on experience with OpenID 2, OAuth, SAML, Facebook Connect, etc.

Developed by the OpenID Foundation
- http://openid.net/connect
Start with a solid base

- OAuth 2 authorization
  - Authorization Server becomes Identity Provider
  - Client becomes Relying Party

- JSON Web Tokens
  - Structured token format

- Can work in fully-distributed mode
  - Dynamic discovery and registration
  - Self-issued identities

- “Make the simple things simple, make the difficult things possible.”
The players

- Resource Owner & User Agent
- Authorization Server
- Client
- Protected Resource

©2013 The MITRE Corporation
The players (with OpenID Hats)

- End User & Web Browser
- Identity Provider
- Relying Party
- Identity API
How it works

- Use OAuth2 to get a regular access token, as well as an ID token
- Use access token to call User Info Endpoint
  - Standardized user profile (name, email, etc.)
  - Standardized scopes (profile, phone, etc.)
- Parse and use ID token
  - Standardized format (JWT)
  - Standardized claims set (user identifier, authentication aspects, etc.)
What else it can do

- Higher levels of assurance
  - Signed and encrypted requests
  - Signed and encrypted responses
- Fine-grained claims management
- Distributed and aggregated claims
- Self-issued identities
- IdP-initiated login
  - Kicks off the standard flow “remotely”
- Can get very complex if you want it to
  - “SAML with curly braces”
A protocol proving ground

- OAuth 2 in the wild
- Real-life interoperability testing
- Real deployments, large and small
- Generalization of protocols
  - OIDC Discovery → Webfinger
  - OIDC Registration → OAuth 2 Dynamic Client Registration
  - JWT Claims
    - Subject, audience, authorized presenter
MITREid Connect
https://github.com/mitreid-connect
Server and client built on Spring Security

Supports key features:
- Signed tokens
- Request objects
- Authorization code and implicit flows

General purpose OAuth2 server
- Flexible scope definitions
- Dynamic registration
- Token introspection
- Token chaining
Interoperability testing

- Interoperability testing with working group
  - Nomura Research Institute (PHP client)
  - OIDC-PHP (PHP Client)
  - IBM (Java client)
  - Nov Matake (Ruby client and server)
  - OIDC test suite (Python)
  - ... and others

- Pending Interoperability event hosted by MIT Kerberos Consortium
Enterprise features

- Enterprise-friendly platform (Java Spring)
- Administration consoles
- Programmable API
- Modern UI
- Event and action logging
- Pluggable primary user authentication
- General-purpose OAuth 2.0 service
Welcome!

OpenID Connect is a next-generation protocol built on top of the OAuth2 authorization framework. OpenID Connect lets you log into a remote site using your identity without exposing your credentials, like a username and password.

About

This OpenID Connect service is built from the MITREid Connect Open Source project started by The MITRE Corporation.

Contact

For more information or support, contact the administrators of this system.

Current Statistics

There have been 1 user of this system who have logged in to 1 total site, for a total of 1 site approval.
<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>edit</th>
<th>whitelist</th>
<th>delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>client</td>
<td>Test Client</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>e4287521-0666-457e-9e4f-059c873313fb</td>
<td>adkikcfghjkl,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45172aeb-cf00-4da5-b016-ad2283bbeb0</td>
<td>Simple Web App</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Edit Client

## Main

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client name</td>
<td>sdklkdfghjkl..</td>
</tr>
<tr>
<td>Client ID</td>
<td>e4257621-0056-467e-9a4f-059ac</td>
</tr>
<tr>
<td>Redirect URI(s)</td>
<td>http://</td>
</tr>
<tr>
<td>Description</td>
<td>Type a description</td>
</tr>
<tr>
<td>Logo</td>
<td>http://</td>
</tr>
</tbody>
</table>

## Access

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terms of Service</td>
<td>http://</td>
</tr>
<tr>
<td>Policy</td>
<td>http://</td>
</tr>
<tr>
<td>Home Page</td>
<td>http://</td>
</tr>
</tbody>
</table>

## Credentials

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contacts</td>
<td>new contact</td>
</tr>
</tbody>
</table>

## Tokens

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
</table>

## Other

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
</table>

[MITREId Connect](http://mitreidconnect.com) © 2013 The MITRE Corporation and MIT KIT.
Approve New Site

**Caution:**
This client was dynamically registered and has very few other users on this system.

Do you authorize "Simple Web App" to sign you into their site using your identity?

Redirect URL: http://nivers.richer.org:8080/simple-web-app
/openid_connect_login

**Access to:**
- [x] log in using your identity
- [ ] basic profile information
- [ ] email address
- [ ] physical address
- [ ] telephone number

**Remember this decision:**
- [ ] remember this decision until I revoke it
- [ ] remember this decision for one hour
- [ ] prompt me again next time

[Authorize]  [Deny]
Available on Maven Central

<dependency>
    <groupId>org.mitre</groupId>
    <artifactId>openid-connect-parent</artifactId>
    <version>1.0.0</version>
    <type>pom</type>
</dependency>

*Also:
openid-connect-common, openid-connect-client, and openid-connect-server

©2013 The MITRE Corporation
Additional pieces

- **Account Chooser**
  - Javascript application to let a relying party select between multiple identity providers

- **JSON Web Key Generator**
  - Generates public and private keys (not certificates) and prints them in JWK format

- **Simple Web App**
  - Sample application showing how to configure the OpenID Connect client filter in Spring Security

- **Example Overlay**
  - Sample application showing how to configure a server overlay
https://github.com/mitreid-connect

Please join us!
Case Study: RHEx
RESTful Health Exchange
What is RHEx?

- A RESTful system for exchange of health data
  - Best practices around HTTP protocols
  - Links to protected data

- Distributed user base
  - Medical practitioners in different organizations
  - Links to files on remote systems
Primary Care Physician Patient Records

Link to record with hData

Fetch Record with OAuth

Authenticate With OpenID

Consulting Physician’s System

Consulting Physician Identity Provider

©2013 The MITRE Corporation
Beyond RHEX: Blue Button

- Blue Button+ initiative on health record exchange
- Using OAuth 2 for protecting records in RESTful API
- Using dynamic discovery and registration
OpenID for MITRE users

- OpenID 2.0 prototype running since 2009
  - OpenID Connect available since 2012
- Backed by MITRE’s identity infrastructure
  - If you are a MITRE person, you have an OpenID
- Usable on any site that supports OpenID
- Nearly-singe-sign-on experience with minimal integration efforts
  - Many web apps supported OpenID 2.0 natively
MITREid by the numbers

- Over **6200** people have used it at least once
  - Ease of developer integration
  - Ease of user experience

- Users have accessed over **340** distinct sites
  - Handshake is largest, whitelisted by admins
  - Most sites are not whitelisted
  - Serves the “long tail” of user needs
Number of users per site

*logarithmic scale
Case Study: Clipper Web Services
Adapting a legacy application
What is Clipper?

- Trans-lingual web browsing and searching project from MITRE
- Originally built as a monolithic application, being redeveloped as a set of services
  - Modular architecture
  - HTTP interconnections
Advanced OAuth

- **Token introspection**
  - Resource servers call back to authorization server to validate tokens at runtime

- **Token chaining**
  - Resource servers need to call each other
  - Downscope tokens for minimal access
User login
Access a service

User Interface → Dictionary Service → OpenID Connect → AuthN Provider

Search Service

©2013 The MITRE Corporation
Chaining a service request

User Interface

Dictionary Service

OpenID Connect

Search Service

AuthN Provider
Turtles all the way down...
Case Study:
MITRE Partnership Network
What is the MPN?

- Services and applications to help MITRE interact with external partners
- Loosely-coupled architecture
  - Based on open standards
  - User-driven experience
  - Opt-in for applications
Making life easy

- Leverage what’s working on the internet
  - Build on existing code and libraries
- Give users a consistent experience
  - One account across applications
  - No password at MITRE if we can avoid it
- Let developers focus on their applications
- Make it easy to do the right thing
What is openPDS?

- Personal Data Store from Human Dynamics lab at MIT
  - Collecting and distributing personal information such as health data for individuals
- Using MITREid Connect as identity provider and authorization layer
Here to stay

- Massive adoption across the public internet and within enterprise
- OpenID Connect an OAuth2 aren’t going anywhere for a while
  - (Until we invent something better)
MITREid Connect project

- In active development by MITRE and others
- Being taken over by the MIT Kerberos Consortium
  - Community building
  - Continued development
  - Outreach and education
Thank you!

jricher@mitre.org
Backup Slides

(Here there be dragons)
A delicious metaphor

Chocolate VS Fudge

OAuth is Chocolate

- Delicious on its own
- Versatile ingredient
  - Useful in many circumstances
- Can be used to make fudge
Sign-on is Fudge

- A confection with several ingredients
- Can be made with chocolate
  - But needs more than just chocolate
  - Could be made without chocolate
Project components

Per-server overlays (not public)

Hosted on GitHub

Open Source, owned by SpringSource

MITREid Connect
Open Source Project

SECOAUTH

Spring Security

Spring

Java

Server A  Server B  …
Internal UA, Internal RP

Corporate Firewall

External OP

Database

Internal OP

User Data

Corporate SSO

DMZ

Intranet
Internal UA, External RP

DMZ

Intranet

Corporate Firewall

External OP

Database

Internal OP

User Data

Corporate SSO

©2013 The MITRE Corporation
External UA, External RP

DMZ

Intranet

Corporate Firewall

External OP

Two-Factor Signon

Database

Internal OP

User Data

©2013 The MITRE Corporation
External UA, Internal RP