Keycloak Integration: Identity Providers

Trey Hayden

After reviewing these slides, you will hopefully:

- Know how Keycloak defines an external Identity Provider (IdP)
- Know which kinds of Identity Providers can be use with Keycloak
- Learn how Keycloak integrates with the example provider (GitHub)
- Learn some of the nuances regarding the system
- (Optional) Follow along and confirm that the example provider works as intended

Outline

- Introduction
  - What is Keycloak?
  - What is an Identity Provider?
  - Why bother?
  - (Optional) Getting Started
- Configuring Identity Providers
- Example Using GitHub's API
- (Coming soon) Example Using a generic SAML provider
- Integration Nuances

Introduction: Keycloak

- What is Keycloak?
  - Keycloak is an open-source identity and access management system
  - It functions as a Single-Sign On (SSO) system for modern services
- As a developer following ADL, you've probably already encountered it in the TLA
- Weeds:
  - Java application that runs on the open-source Wildfly runtime
  - Uses OpenID Connect (a profile of OAuth 2)

KEYCLOAK

Introduction: Identity Providers

- Definition:
  - *An **identity provider** (abbreviated "IdP") is a system entity that creates, maintains, and manages **identity** information for principals while providing authentication services to relying applications within a federation or distributed network.*
  - TL;DR: A system capable of maintaining and authenticating users
- Examples:
  - Social Media (Microsoft, Google, Twitter, Facebook, GitHub, etc)
  - OpenID Connect (Keycloak)
  - SAML v2.0

Introduction: Why Bother?

- Account fatigue
  - Remembering multiple login credentials
  - Creating a new account for a service you use infrequently
- Increase modularity of your service ecosystem
  - Integration with **existing** Identity Providers
  - Instantly gives system access to an entire database users
- It's easy
  - You can quickly set up a Keycloak instance
  - Smart people have already done the hard parts for you

Introduction: Getting Started (Optional)

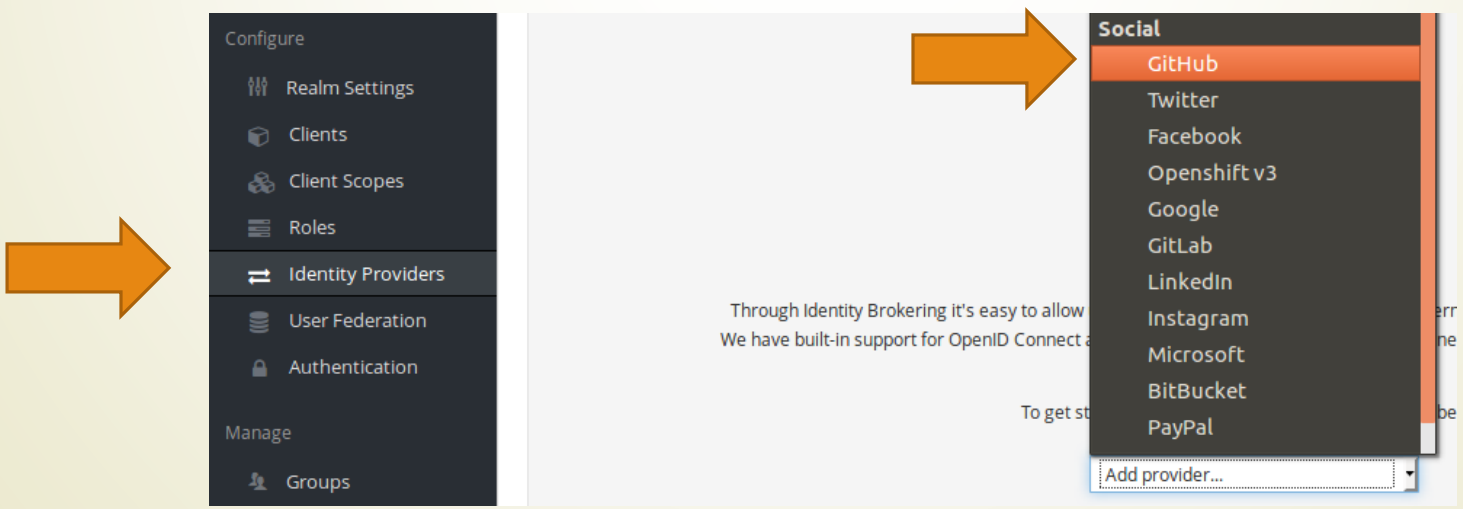
- Create an Ubuntu VM (16+)
- TL;DR:
  - `git clone https://github.com/vbhayden/keycloak-federation-examples`
  - Modify the docker-compose.yml file's KEYCLOAK_URL variable
    - Set it to whatever the VM's local IP is
  - `sudo ./install-reqs.sh`
  - `sudo ./rebuild.sh`
- This should have set up the following:
  - A Keycloak instance with Wildfly
  - A PostGreSQL database
  - An OpenLDAP instance with a web-accessible admin UI
  - An example service protected by Keycloak

Configuring Identity Providers

- To configure an Identity Provider with Keycloak, there are two parts. At a high level:
  - Register a “client” (or “app”) within the Identity Provider
    - Identity Providers usually require you to register some sort of client that will be responsible for the authentication handshakes between it and your OpenID Connect provider (Keycloak). This is for a few different reasons, including client permissions, client revocation, modularity within the provider, etc.
    - Some providers are better than others with their documentation, so finding out how to actually create one of these clients can be tricky.
    - You will usually need your Keycloak instance running in order to provide a redirect URL for the provider.
  - Use those “client” credentials to enable the Identity Provider within Keycloak
    - As this process is actually a transaction of user data from the Identity Provider’s API to Keycloak, we will require some sort of permission (usually through the form of a “client secret”) to get that user information.
    - Once you have those credentials, Keycloak will use them to interact with the Identity Provider and complete the login transactions.
    - There are presets for popular Identity Providers (Facebook, Twitter, etc), but you can configure anything that uses the OpenID Connect or SAML v2.0 specifications.

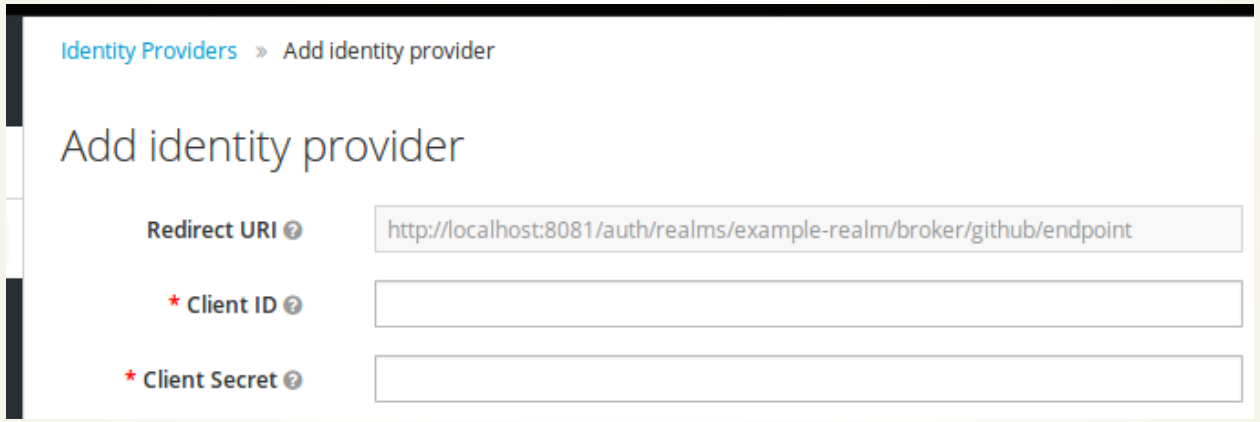
GitHub Example: Keycloak Setup

- Since the GitHub API will require our Keycloak address, let's set that up first.
- If you followed the instructions earlier, you should already have a Keycloak instance up!
  - If not, just enjoy the show
- Click on Identity Providers
- Select **GitHub** from the list



GitHub Example: Keycloak Setup

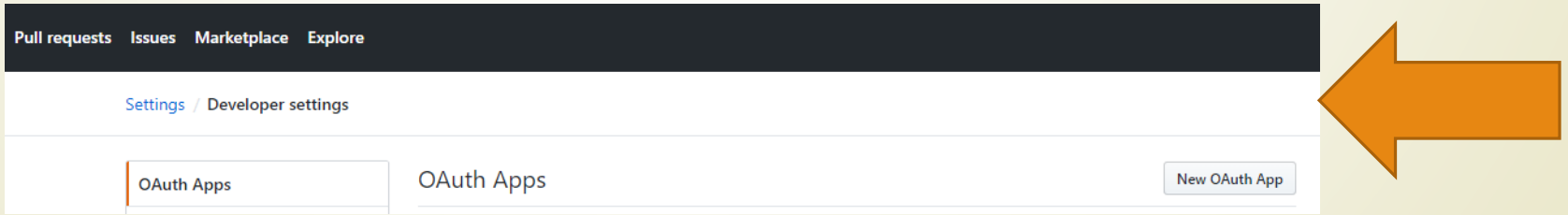
- You should now see a screen for configuring a GitHub Identity Provider



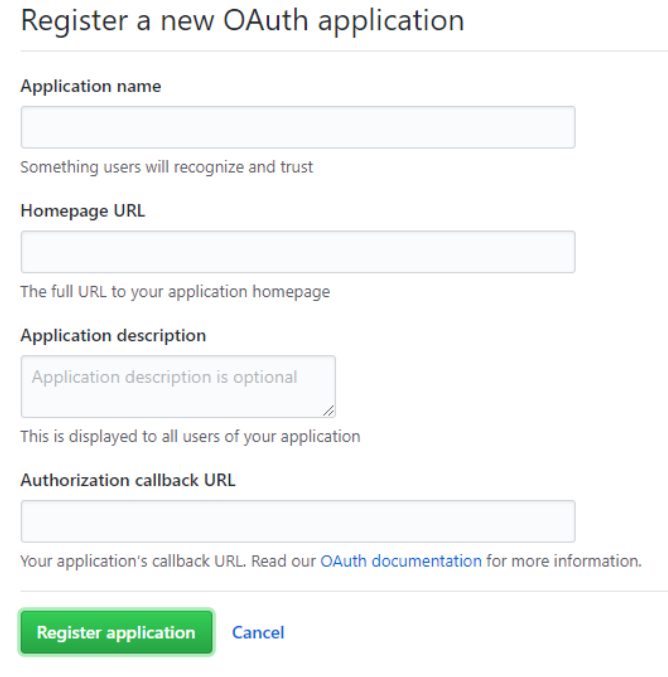
- We're only interested in these 3 topmost fields
  - Client ID and Client Secret we will get from GitHub
  - We use the Redirect URI to configure the GitHub client itself
- **Done for now, time to configure our GitHub client!**

GitHub Example: Client Setup

- Before we start, what is actually going on with this process?
  - First, we will need to register our Keycloak service as an OAuth app. In addition to letting GitHub know that we will be taking user information from GitHub's API using this app, we will also tell GitHub where to redirect the user once the login transaction is complete.
  - In our case, this will be our Keycloak server's location. It can either be the DNS name or an IP address. Since the redirect happens on the browser, local IPs are valid for testing purposes.
  - **You will need your Keycloak machine's address to complete this process**.
- Browse to: https://github.com/settings/developers
- Click "New OAuth App" to start creating the client



GitHub Example: Client Setup

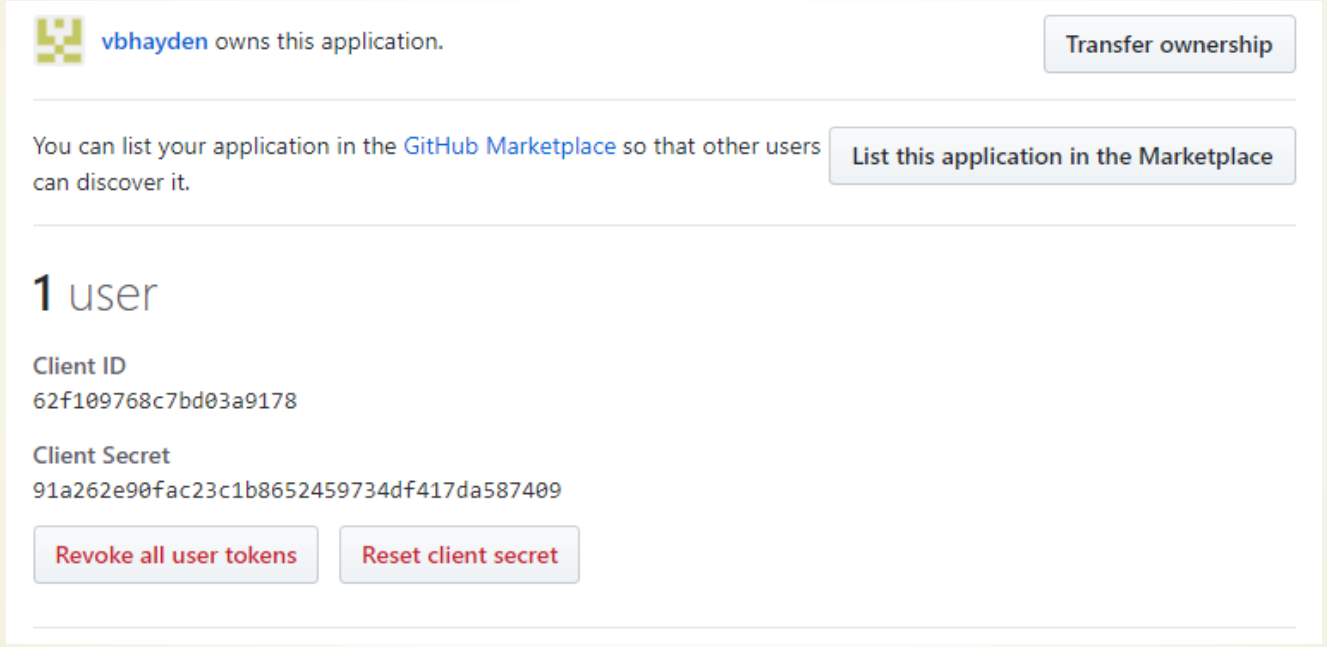


- Application name (Required; String)
- Homepage (Required; URL)
- Description (Optional; String)
- Callback URL (Required, Important)
  - This is how GitHub will know to send the user back to Keycloak once they've logged in on GitHub

- The callback URL itself can be found in Keycloak, but it likely uses **localhost**
- Assuming your Keycloak uses **192.168.30.153** on port **8081**, your URL would be:
  - http://**192.168.30.153:8081**/auth/realms/example-realm/broker/github/endpoint
- Click "Register application"

GitHub Example: Client Setup

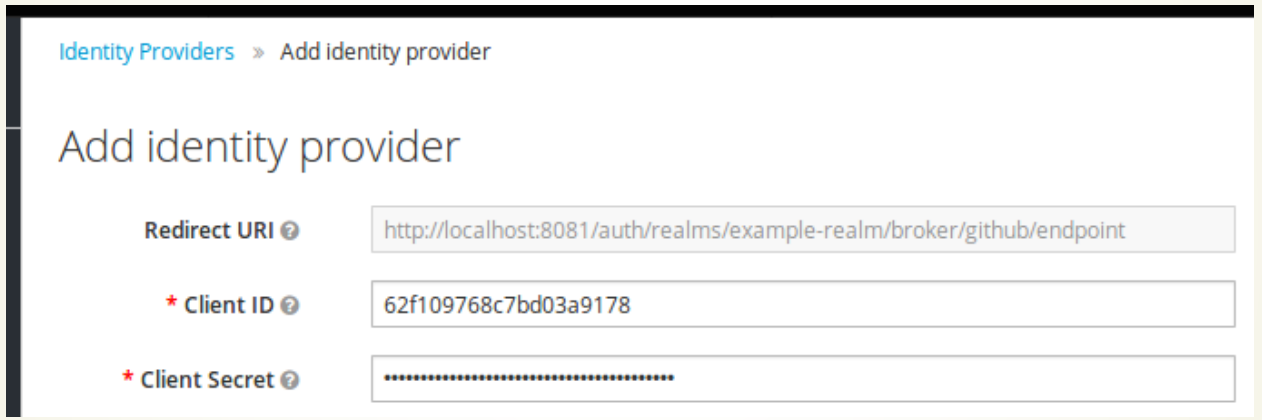
- Once that application is registered, GitHub will assign it an ID and a Secret



- And with that, the GitHub portion of our integration is done!
- Back to Keycloak …

GitHub Example: Finalizing Keycloak

- We now have the Client ID and Secret values to use for our Identity Provider:



- Save the settings and you're done.

GitHub Example: Checking Our Work

- If everything went according to plan, users can now
  - 1) Browse to your Keycloak-protected service
  - 2) Be redirected to your Keycloak login page
  - 3) Choose to sign-in with their GitHub account
  - 4) Be redirected back to the Keycloak service
  - 5) Be redirected to your service