EngineAuth Documentation

Release 0.1

Kyle Finley

August 06, 2015

Contents

EngineAuth is a standardized approach to third party authentication / authorization, designed to be as simple as possible, both for the developer and the end user.

CHAPTER 1

Disclaimer

Warning: EngineAuth is in the very early stages of development and the api is likely to change frequently and in non-backwards compatible ways. Please provide any issues, suggestions, or general feedback through the Issue Tracker, or in the comments section of this documentation.

CHAPTER 2

Demo

EngineAuth Example - Example site

CHAPTER 3

How it Works

Note: If you are unable to view the above image. Please log into your Google Docs account, or log out of Google altogether. There's currently a Google Docs bug that requires a user to be sign in to Google Docs to view public content.

CHAPTER 4

Supported Strategies & Providers

New strategies will be written as needed. If there's a particular strategy that your interested in please create a new issues using the *strategy request* label.

4.1 OAuth

- Twitter

4.2 OAuth2

- Facebook
- Google

4.3 OpenID Provider

- All - via App Engine OpenID

4.4 Email & Password

If the provider that you need isn't provided not to worry, adding additional providers is simple, and in many cases only requires a few lines of code.

CHAPTER 5

Requirements

- Google App Engine running Python 2.7

CHAPTER 6

Installation

Copy the `engineauth` directory and the contents of `lib` directory to your project's `root` directory.

6.1 Dependencies

- oauth2client - Required for OAuth2 Strategies
- httplib2 - Required for OAuth and OAuth2 Strategies
- uri-templates - Required for OAuth and OAuth2 Strategies
- python-gflags - Required for OAuth and OAuth2 Strategies
- python-oauth2 - Required for OAuth Strategies

6.2 Configuring EngineAuth

In your `appengine_config.py` add:

```
def webapp_add_wsgi_middleware(app):
    from engineauth import middleware
    return middleware.AuthMiddleware(app)

engineauth = {
    'secret_key': 'CHANGE_TO_A_SECRET_KEY',
    'user_model': 'engineauth.models.User',
}

engineauth['provider.auth'] = {
    'user_model': 'engineauth.models.User',
    'session_backend': 'datastore',
}

# Facebook Authentication
engineauth['provider.facebook'] = {
    'client_id': 'CHANGE_TO_FACEBOOK_APP_ID',
    'client_secret': 'CHANGE_TO_FACEBOOK_CLIENT_SECRET',
    'scope': 'email',
}

# Google Plus Authentication
```

```
engineauth['provider.google'] = {
    'client_id': 'CHANGE_TO_GOOGLE_CLIENT_ID',
    'client_secret': 'CHANGE_TO_GOOGLE_CLIENT_SECRET',
    'api_key': 'CHANGE_TO_GOOGLE_API_KEY',
    'scope': 'https://www.googleapis.com/auth/plus.me',
}

# Twitter Authentication
engineauth['provider.twitter'] = {
    'client_id': 'CHAGNE_TO_TWITTER_CONSUMER_KEY',
    'client_secret': 'CHAGNE_TO_TWITTER_CONSUMER_SECRET',
}
```

6.3 Acquiring Client Keys

6.3.1 Facebook

1. Go to: https://developers.facebook.com/apps
2. Select your application
3. Under `Select how your app integrates with Facebook` click `Website`. In the `Site URL:` field enter your domain E.g. http://example.com/ or http://localhost:8080/ be sure to include the closing `/`.
4. Copy `App ID/API Key` as `client_id`
5. Copy `App Secret` as `client_secret`

Note: Zuckerberg won't allow you to specify multiple callback domains for a single application. So for development you must create a separate application. Then, in your `appengine_config.py` you can specify which config will be loaded at runtime.

```
import os
ON_DEV = os.environ.get('SERVER_SOFTWARE', '').startswith('Dev')
if ON_DEV:
    # Facebook settings for Development
    FACEBOOK_APP_KEY = 'DEVELOPMENT_APP_KEY'
    FACEBOOK_APP_SECRET = 'DEVELOPMENT_APP_SECRET'
else:
    # Facebook settings for Production
    FACEBOOK_APP_KEY = 'PRODUCTION_APP_KEY'
    FACEBOOK_APP_SECRET = 'PRODUCTION_APP_SECRET'
engineauth['provider.facebook'] = {
    'client_id': FACEBOOK_APP_KEY,
    'client_secret': FACEBOOK_APP_SECRET,
    'scope': 'email',
    }
```

6.3.2 Google Plus

1. Go to: https://code.google.com/apis/console
2. Select your application or create a new one.
3. Choose `API Access`

4. Click `Create an OAuth 2.0 client ID..`
5. Enter Product name -> Next
6. Select `Web application`
7. Under `Your site or host` select `(more options)`
8. Under `Authorized Redirect URIs` add your domain name followed by `/auth/google/callback` E.g. `http://localhost:8080/auth/google/callback`, `http://YOUR_DOMAIN.COM/auth/google/callback`
9. Click `Create client ID`
10. Copy `Client ID` as `client_id`
11. Copy `Client secret` as `client_secret`

6.3.3 Twitter

1. Go to: https://dev.twitter.com/apps
2. Select your application or create a new one.
3. Make sure the you set the callback to `http://YOUR_DOMAIN.COM/auth/twitter/callback`. It's fine to set this to your production url, EngineAuth passes a redirect url while authenticating so there's no need to specify `localhost:8080` here.
4. Go to Details OAuth settings
5. Copy `Consumer key` as `client_id`
6. Copy `Consumer secret` as `client_secret`

6.3.4 LinkedIn

1. Go to: https://www.linkedin.com/secure/developer?newapp
2. Fill in required fileds. You may leave `OAuth Redirect URL:` blank.
3. Click `Add Application`
4. Copy `API Key` as `client_id`
5. Copy `Secret Key` as `client_secret`
6. Click `Done`

6.3.5 Github

1. Go to: https://github.com/account/applications/new
2. Fill in required fileds. For `Callback URL` enter ‘‘http://YOUR_DOMAIN.COM/auth/github/callback‘
3. Click `Create Application`
4. Copy `Client ID` as `client_id`
5. Copy `Secret` as `client_secret`
6. Click `Done`

6.3.6 App Engine OpenID

1. Go to: https://appengine.google.com
2. Select your application
3. Choose `Application Settings`
4. Choose `(Experimental Federated Login)` from the `Authentication Options` drop down
5. Click Save

CHAPTER 7

Objectives

7.1 User

When beginning any new web application, that involves users, you've probably asked yourself:

- How can I verify my user's identities?
- How do I protect their privacy?
- How can I make the signup process as simple as possible?
- How do I save my user from entering their information on yet another sight?
- How can I leverage the wealth of information that my users have entered into third party sights?

Which brings us to:

Note: Objective #1

Provide a clear path for Authentication / Authorization, that is secure, simple to use, and allows users to share their information, effortlessly.

7.2 Developer

And from a development standpoint you've probably ask:

- How can I save myself from writing yet another authentication strategy?
- As developers why are we all writing the same code, over and over again?
- How can I share what I've learn with others?

Which brings us to:

Note: Objective #2

The solution should be easy to implement, and easy to extend and share.

CHAPTER 8

Credits

EngineAuth brings together ideas and code from many projects:

- Google App Engine and the Google App Engine Team: Obviously.
- Rodrigo Moraes: many aspects of this project were derived form his work on webapp2. Including sessions, models, test setup, and even this documentation.
- Google Api Python Client: this library provides the foundation for EngineAuth's Authentication and Authorization.
- OmniAuth: the basic structure for `Provider Strategies` comes from OmniAuth
- TODO: add others.

CHAPTER 9

License

EngineAuth is licensed under the Apache License 2.0.