Trusted Publishers in PyPI

About Me

Jordan Duabe -- `@j4ckofalltrades`

- Software Engineer @ Anaqua
- Command-line ninja 🥷
- Has one too many MKBs ⌨️

Agenda

- Intro to PyPI
- Packaging (and Publishing) Flow
- Trusted Publishing

The Python Package Index (PyPI)

What is PyPI

- The official third-party software repository for Python hosted at pypi.org
- aka the Cheese Shop
- Launched in 2003
- Hosts > 450,000 packages (as of May 2023)

TestPyPI

A separate (test) instance of the Python Package Index (PyPI) accessible at test.pypi.org

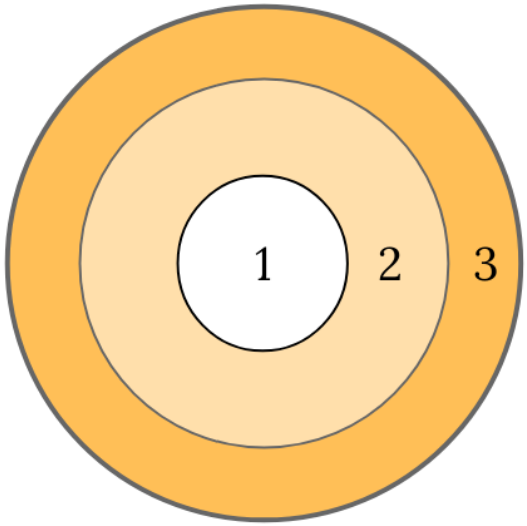
Packaging (and Publishing) Flow

Package types

Packaging for Python **tools** and **libraries**

1. **`.py`** - standalone modules
2. **sdist** - Pure-Python packages
3. **wheel** - Python packages

(With room to spare for static vs. dynamic linking)



Excerpted from *The Packaging Gradient (2017)*

Package structure

```
sampleproject/
├── LICENSE
├── pyproject.toml
├── README.md
├── src/
│   └── sampleproject/
│       ├── __init__.py
│       └── example.py
└── tests/
```

Configuration file

```
[build-system]
requires = ["setuptools"]
build-backend = "setuptools.build_meta"

[project]
name = "sampleproject"
version = "3.0.0"
description = "A sample Python project"
readme = "README.md"
requires-python = ">=3.8"
license = {file = "LICENSE.txt"}
keywords = ["sample", "setuptools", "development"]
authors = [
  {name = "A. Random Developer", email = "author@example.com" }
]
maintainers = [
  {name = "A. Great Maintainer", email = "maintainer@example.com" }
]
```

Generating distribution archives

```
python3 -m pip install --upgrade build
python3 -m build

# output
ls dist
dist/
├── sampleproject-0.0.1-py3-none-any.whl
└── sampleproject-0.0.1.tar.gz
```

Built Distribution (wheel)

```
sampleproject-3.0.0.dist-info
    entry_points.txt
    LICENSE.txt
    METADATA
    RECORD
    top_level.txt
    WHEEL
sample
    __init__.py
    package_data.dat
    simple.py
```

Source Distribution (sdist)

```
sampleproject-3.0.0
    LICENSE.txt
    PKG-INFO
    pyproject.toml
    README.md
    setup.cfg
    src
        sample
            __init__.py
            package_data.dat
            simple.py
        sampleproject.egg-info
            dependency_links.txt
            entry_points.txt
            PKG-INFO
            requires.txt
            SOURCES.txt
            top_level.txt
    tests
        test_simple.py
```

Uploading the distribution archives

This requires a (1) PyPI account and an (2) API token.

```
python3 -m pip install --upgrade twine
python3 -m twine upload dist/*

# prompt for PyPI credentials
Enter your username: __token__
Uploading sampleproject-0.0.1-py3-none-any.whl
100% ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.2/8.2 kB • 00:01 • ?
Uploading sampleproject-0.0.1.tar.gz
100% ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.8/6.8 kB • 00:00 • ?
```

Automation with GitHub Actions

```yaml
jobs:
    pypi-publish:
        name: upload release to PyPI
        runs-on: ubuntu-latest
    steps:
        # build distributions

    - name: Publish package distributions to PyPI
      uses: pypa/gh-action-pypi-publish@release/v1
      with:
        username: __token__
        password: ${{ secrets.PYPI_API_TOKEN }}
```

Trusted Publishing

Info

GitHub is currently the only OIDC identity provider that is supported.

Publishing with OpenID Connect (OIDC)

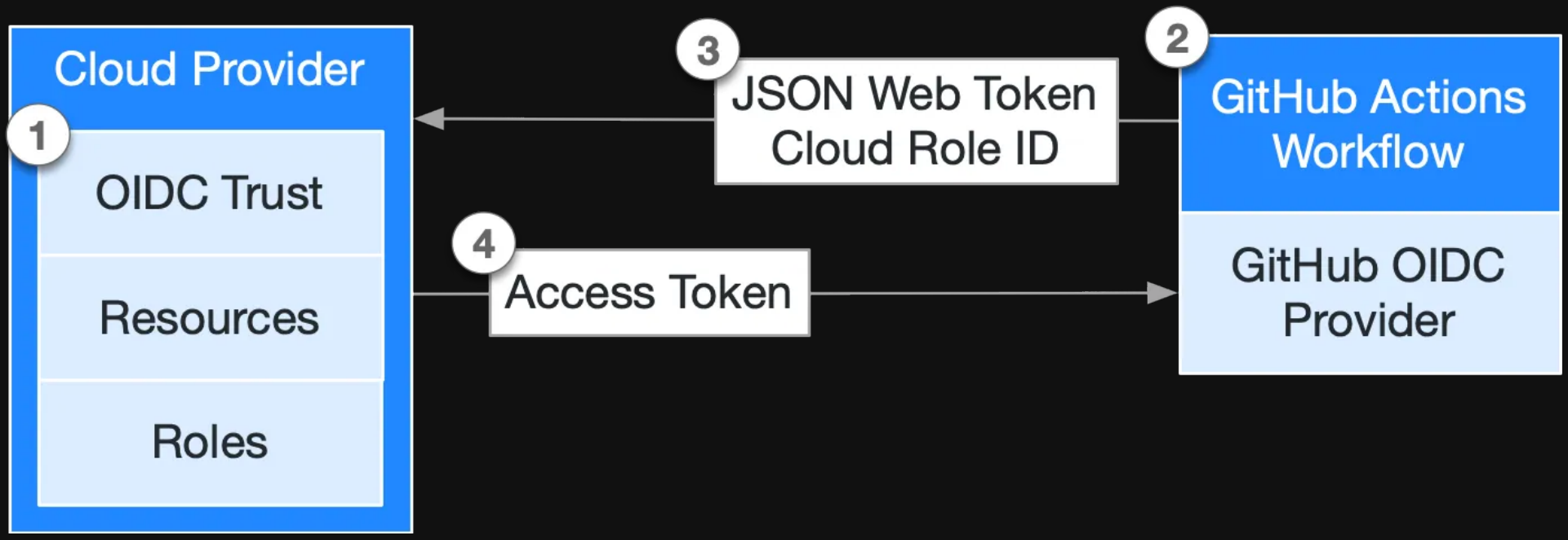


Diagram from *OpenID Overview* on GitHub

Configuring a Trusted Publisher in PyPI

Add a new publisher

GitHub

Read more about GitHub Actions's OpenID Connect support here.

Owner (required)

octo-org

The GitHub organization name or GitHub username that owns the repository

Repository name (required)

sampleproject

The name of the GitHub repository that contains the publishing workflow

Workflow name (required)

release.yml

The filename of the publishing workflow. This file should exist in the `.github/workflows/` directory in the repository configured above.

Environment name (optional)

release

The name of the GitHub Actions environment that the above workflow uses for publishing. This should be configured under the repository's settings. While not required, a dedicated publishing environment is **strongly** encouraged, **especially** if your repository has maintainers with commit access who shouldn't have PyPI publishing access.

Add

JWT (Encoded)


```
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.ey
JzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ikpva
G4gRG9lIiwiYWRtaW4iOnRydWUsImlhdCI6MTUx
NjIzOTAyMn0.NHVaYe26MbtOYhSKkoKYdFVomg4
i8ZJd8_-
RU8VNbftc4TSMb4bXP3l3YlNWACwyXPGffz5aXH
c6lty1Y2t4SWRqGteragsVdZufDn5BlnJl9pdR_
kdVFUsra2rWKEofkZeIC4yWytE58sMIihvo9H1S
cmmVwBcQP6XETqYd0aSHp1gOa9RdUPDvoXQ5oqy
gTqVtxaDr6wUFKrKItgBMzWIdNZ6y7O9E0DhEPT
bE9rfBo6KTFsHAZnMg4k68CDp2woYIaXbmYTWcv
bzIuHO7_37GT79XdIwkm95QJ7hYC9RiwrV7mesb
Y4PAahERJawntho0my942XheVLmGwLMBkQ
```

JWT (Decoded)


```
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "example-thumbprint",
  "kid": "example-key-id"
}
{
  "jti": "example-id",
  "sub": "repo:octo-org/octo-repo:environment:prod",
  "environment": "prod",
  "aud": "https://github.com/octo-org",
  "ref": "refs/heads/main",
  "sha": "example-sha",
  "repository": "octo-org/octo-repo",
  "repository_owner": "octo-org",
  "actor_id": "12",
  "repository_visibility": "private",
  "repository_id": "74",
  "repository_owner_id": "65",
  "run_id": "example-run-id",
  "run_number": "10",
  "run_attempt": "2",
  "runner_environment": "github-hosted"
  "actor": "octocat",
  "workflow": "example-workflow",
  "head_ref": "",
  "base_ref": "",
  "event_name": "workflow_dispatch",
  "ref_type": "branch",
  "job_workflow_ref": "octo-org/octo-automation/.github/workflows/oidc.yml@refs/heads/main",
  "iss": "https://token.actions.githubusercontent.com",
  "nbf": 1632492967,
  "exp": 1632493867,
  "iat": 1632493567
}
```

Verifying the JWT

The GitHub OIDC provider configuration contains the following information:

- A `claims_supported` JSON array that lists all supported JWT claims
- A `jwks_uri` that contains the JSON Web Key Set used to verify the JWT

Using Trusted Publishing with GitHub Actions

```
jobs:
   pypi-publish:
     name: upload release to PyPI
     runs-on: ubuntu-latest
+    permissions:
+      # IMPORTANT: this permission is mandatory for trusted publishing
+      id-token: write
     steps:
       # retrieve your distributions here

       - name: Publish package distributions to PyPI
         uses: pypa/gh-action-pypi-publish@release/v1
-        with:
-          username: __token__
-          password: ${{ secrets.PYPI_API_TOKEN }}
```

Putting it all together

Trusted Publisher workflow in action

github.com/j4ckofalltrades/powerline-k8s

Why use Trusted Publishers?

Why use Trusted Publishers?

1. **Usability.** With a trusted publisher, no manual API token management is necessary: configuring the publisher is a one-time action for each project.

Why use Trusted Publishers?

1. **Usability.** With a trusted publisher, no manual API token management is necessary: configuring the publisher is a one-time action for each project.
2. **Security**. Short-lived (effectively ephemeral) tokens reduces potential damage (if the token gets "leaked") as it expires automatically.

Resources

- PyPI Docs
- PyPA Packaging User Guide
- GitHub Overview of OpenID Connect
- JWT.io

Slides