Building Apptainer Containers on Demand Using Globus Compute

Ben Galewsky, Ritwik Deshpande, Sindhu Inuganti
About me...

• Sr Research Software Engineer at NCSA
• Background in IT Consulting in Industry
• Contributor to
  – Globus Compute
  – Garden
  – Institute for Research and Innovation in Software for High Energy Physics (IRIS-HEP)
  – Data Management Consulting
Overview

- Dependency management in Globus Compute
- Lessons learned from Container Service
- Introducing the custom-image-builder library
- Try it out!
Dependency Management in GC

- Imports must sit inside the function
- The worker in the endpoint must have all dependencies installed

```python
def inflation(csv_url: str):
    import pandas as pd
    df = pd.read_csv(csv_url)
    return df.mean()
```
Dependency Management in GC

- Imports must sit inside the function
- The worker in the endpoint must have all dependencies installed

```python
def inflation(csv_url: str):
    import pandas as pd
    df = pd.read_csv(csv_url)
    return df.mean()
```

ModuleNotFoundError: No module named 'pandas'
Current Solution

• In the endpoint configuration

```yaml
engine:
  provider:
    worker_init: pip install pandas
```
Drawbacks

- Hardcoded
- Each unique environment requires its own endpoint
- Opaque
Container Service

• Service deployed inside Globus Compute Cluster
• Accepts repo2docker style container spec
• Builds on demand
• Publish to DockerHub
• Conveniently integrated with Globus Compute container objects
Container Service: Lessons Learnt

• Very convenient for users
• Unbounded compute requirement
• Difficult to manage docker image repository
• Only works on Kubernetes based endpoints
Enter the Custom-Image-Builder

- Pip installable library
- Runs as Globus Compute Task on any endpoint
- Accepts repo2docker style container spec
- Writes out an apptainer env file
- Executes apptainer build
- Registers container
def build_and_register_container(
gcc_client: Client,
endpoint_id: str,
image_file_name: str,
base_image_type: str,
base_image: str,
payload_url: str = None,
pip_packages: list = None,
conda_packages: list = None,
apt_packages: list = None) -> str:
Example

```python
image_builder_endpoint = "5cdc5147-378c-4ed9-8ede-25fa3614e6aa"
gcc_client = Client()

container_id = build_and_register_container(gcc_client=gcc_client,
                                         endpoint_id=image_builder_endpoint,
                                         image_file_name="my-pandas-image",
                                         base_image_type="docker",
                                         base_image="python:3.8",
                                         pip_packages=["pandas"]
)

with Executor(endpoint_id=image_builder_endpoint,
               container_id=container_id) as ex:
    fut = ex.submit(my_function)
```
Try it out!

% pip install custom-image-builder

Current caveats

• Only works with globus-compute-endpoint 2.2.0
• Requires python 3.9
Gratitude

- Ben Blaiszik
- Kyle Chard
- Ryan Chard
- Will Engler
- Steve Goldstein
- Ari Scourtas
- KJ Schmidt
- Owen Price Skelly
- Steve Wangen

This project is supported by the National Science Foundation under Award 2004932.
Find Out More....

https://pypi.org/project/custom-image-builder/

https://github.com/ncsa/CustomImageBuilder

bengal1@Illinois.edu