High-Performance Data-Intensive Computing Systems Laboratory

Running JupyterHub in Kubernetes

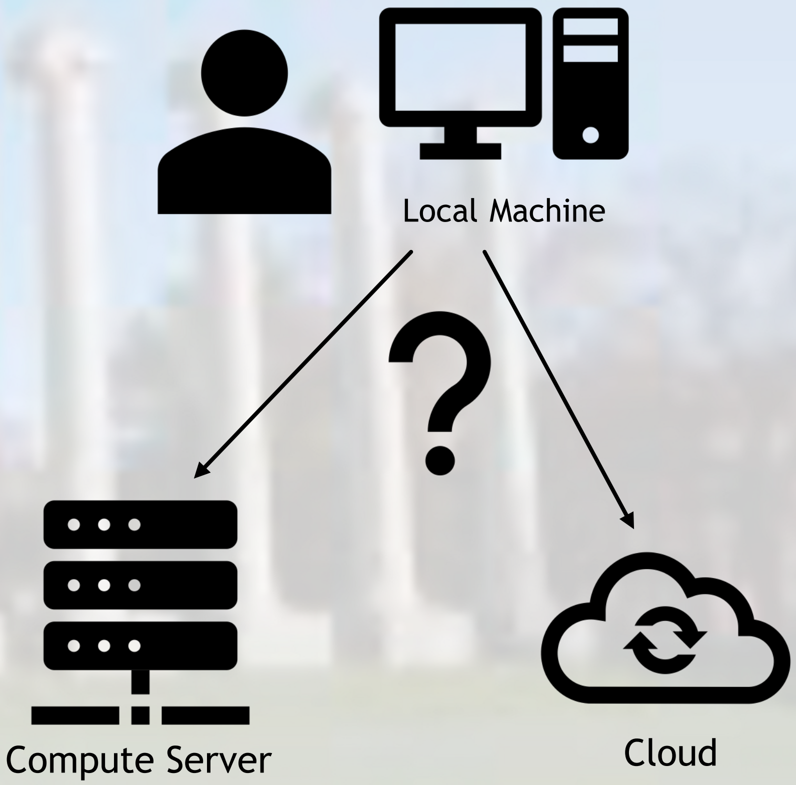
MORENet Technical Summit

20 Feb 2023

University of Missouri

The Problem: Scalability & Reproducibility

- How do we ensure reliable portability of software developed on local development machines to other computational environments?
- How do we move code from local development on one machine to hundreds/thousands of machines?



JupyterHub

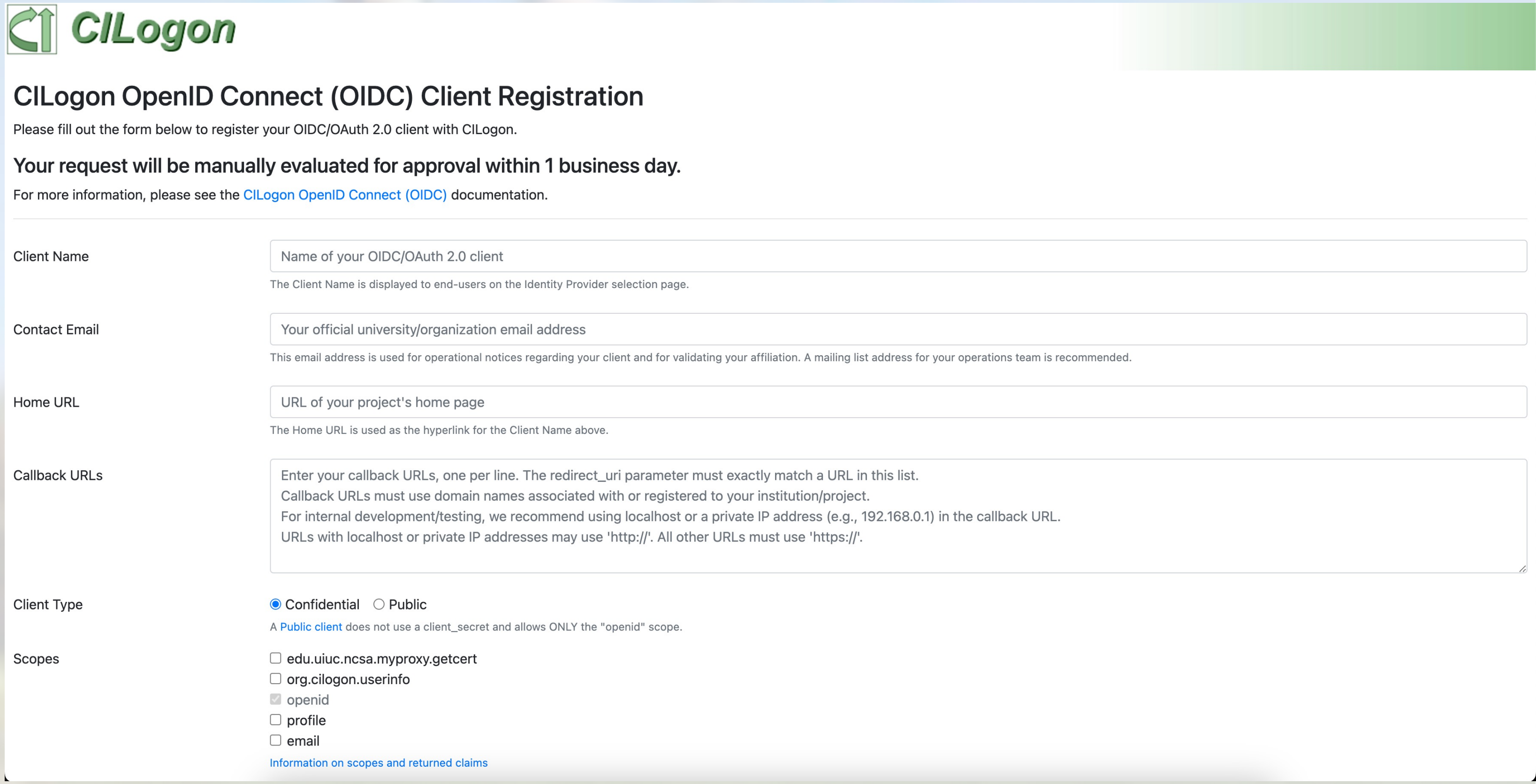
- JupyterHub is a browser-based interactive environment that can be utilized for many different applications
  - Teaching
  - Research
- Using JupyterHub for STEM instruction provides an opportunity for hands-on, interactive education
- JupyterHub can authenticate with CILogin, meaning no additional authentication management is necessary

Running JupyterHub on Nautilus

1. Register CiLogon Application
2. Create Kubernetes Namespace
3. Request removal of Pod Limitations
4. Install Helm
5. Build Configuration YAML
6. Deploy JupyterHub

Step 1: Register CiLogon Application

- https://cilogon.org/oauth2/register
  - Callback URL: https://YOURNAME.nrp-nautilus.io/hub/oauth_callback
  - Client Type: Confidential
  - Scopes:
    - org.cilogon.userinfo
    - openid
    - Profile
    - Email
  - Refresh Tokens: No
- **Important: Keep the Client ID and Client Secret!**



CILogon

CILogon OpenID Connect (OIDC) Client Registration

Please fill out the form below to register your OIDC/OAuth 2.0 client with CILogon.

Your request will be manually evaluated for approval within 1 business day.

For more information, please see the CILogon OpenID Connect (OIDC) documentation.

Client Name: Name of your OIDC/OAuth 2.0 client
The Client Name is displayed to end-users on the Identity Provider selection page.

Contact Email: Your official university/organization email address
This email address is used for operational notices regarding your client and for validating your affiliation. A mailing list address for your operations team is recommended.

Home URL: URL of your project's home page
The Home URL is used as the hyperlink for the Client Name above.

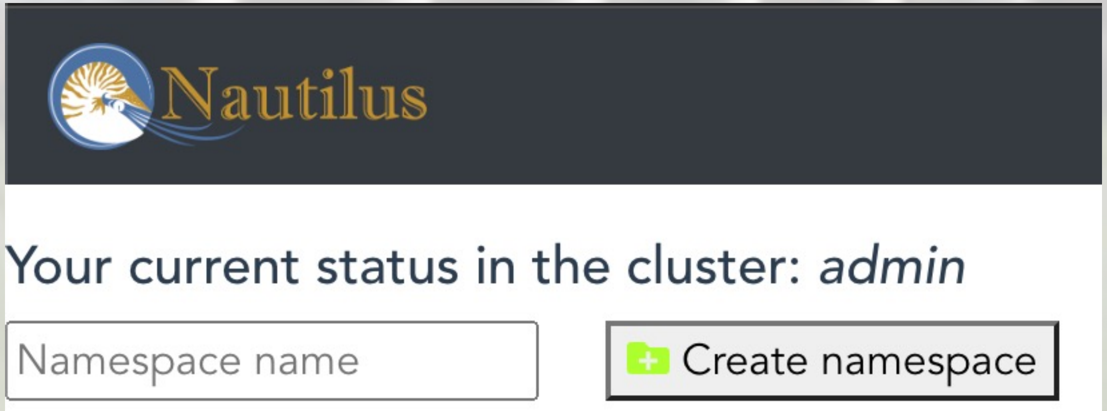
Callback URLs: Enter your callback URLs, one per line. The redirect_uri parameter must exactly match a URL in this list.
Callback URLs must use domain names associated with or registered to your institution/project.
For internal development/testing, we recommend using localhost or a private IP address (e.g., 192.168.0.1) in the callback URL.
URLs with localhost or private IP addresses may use 'http://'. All other URLs must use 'https://'.

Client Type: ● Confidential ○ Public
A Public client does not use a client_secret and allows ONLY the "openid" scope.

Scopes:
☐ edu.uiuc.ncsa.myproxy.getcert
☐ org.cilogon.userinfo
☑ openid
☐ profile
☐ email
Information on scopes and returned claims

Step 2: Create a Kubernetes Namespace

- Recall: **Namespaces** provide a way for K8s to partition cluster resources across multiple or many users in an exclusive way.
- All of the instances of Jupyter Hub running will be in a separate pod in a namespace
- To create the namespace on the Nautilus cluster, a cluster administrator will need to go to the following URL: https://portal.nrp-nautilus.io/profileN



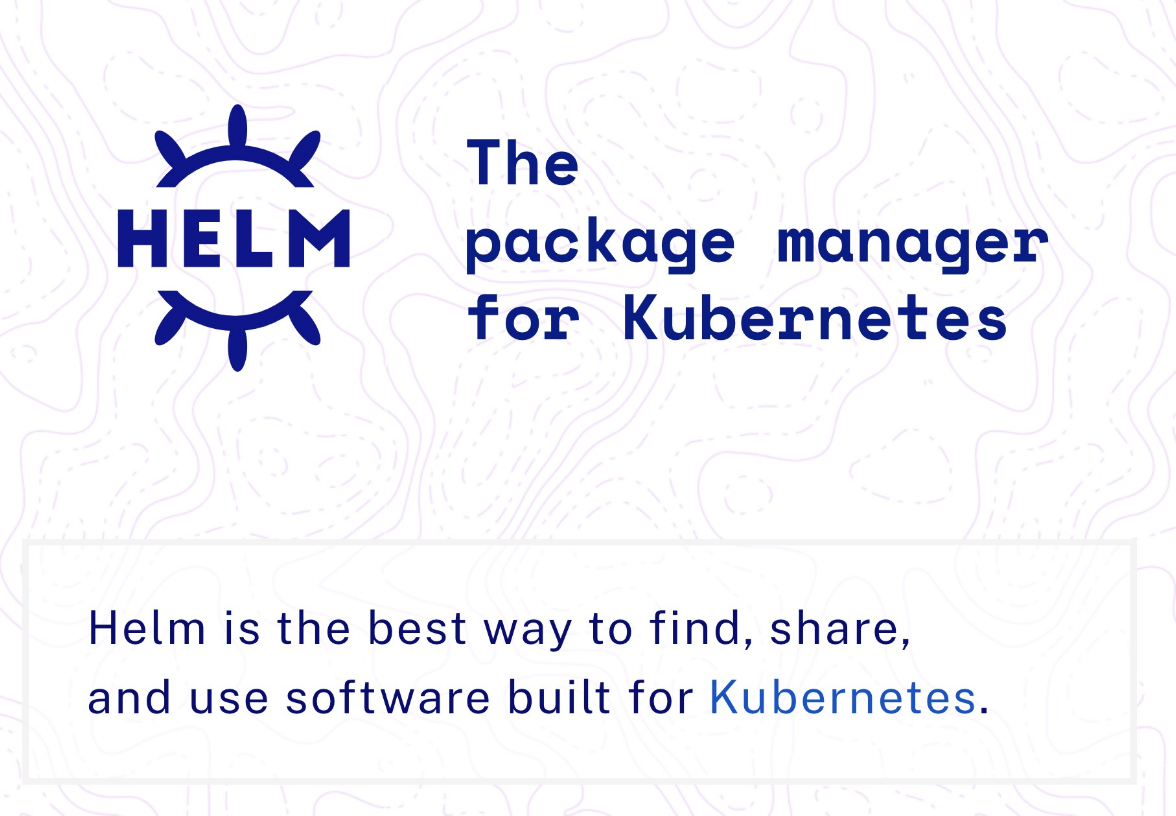
Your current status in the cluster: *admin*

Step 3: Request Removal of Pod Restrictions

- By default, pods on the Nautilus cluster can only run for 6 hours
- If you have users that may use their Jupyter Hub environment for more than 6 hours, their resources will be released at the 6 hour mark
- You can request that pod restrictions be removed on your namespace by going to the [Matrix] chat for PRP Nautilus here: https://element.nrp-nautilus.io/#/welcome
- Once you have access to the server, send a message in **Nautilus Support** channel containing the name of the namespace you are using for Jupyter Hub

Step 4: Install Helm

- Helm is a package manager for Kubernetes, and enables software deployment on K8s Clusters
- We will use helm to manage our JupyterHub installation
- To download and install: https://helm.sh/docs/intro/install



The package manager for Kubernetes

Helm is the best way to find, share, and use software built for Kubernetes.

Step 5: Build Configuration YAML File

- Helm is configured, similarly to K8s, with YAML files
- We need to build a YAML config file to tell Helm how we want our JupyterHub application configured
- We can download the template for JupyterHub on Nautilus here:
  - https://ucsd-prp.gitlab.io/userdocs/jupyter/values.yaml
- Fields to Update in the Template:
  - `secret_token` should be updated to the output of this command: `openssl rand -hex 32`
  - `client_id` and `client_secret` should be updated to match what you received when you registered you CiLogon client
  - `admin_users` should be set to who you'd like to have administrative access inside Jupyter Hub
  - `oauth_callback_url` should updated to: https://YOURNAME.nrp-nautilus.io/hub/oauth_callback
  - `ingress.hosts` should be updated to: https://YOURNAME.nrp-nautilus.io/

Step 6: Deploy JupyterHub with Helm

- Add the Jupyter Hub repository:

Bash

```bash
$ helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/ &&
  helm repo update
```

- Deploy the application:

Bash

```bash
$ helm upgrade --cleanup-on-fail --install jhub jupyterhub/jupyterhub
  --namespace <namespace> --version=1.2.0 --values config.yaml
```

  - where `config.yaml` is the configuration file you setup in Step 5 and `<namespace>` is the namespace you created in Step 2.