



Karvdash: A complete software stack for performing data science on Kubernetes

Antony Chazapis

chazapis_at_ics.forth.gr

FOSSCOMM 2021, 13-14/11/2021

The problem

Large scale computing is extremely difficult for people who actually need it

- Because of complexity
 - Time wasted in devops
 - Time wasted in rewriting the application for a specific environment
 - Steep learning curve
 - Vendor lock-in
- Because it's expensive
 - It is difficult to estimate requirements
 - It is difficult to estimate cloud-provider costs

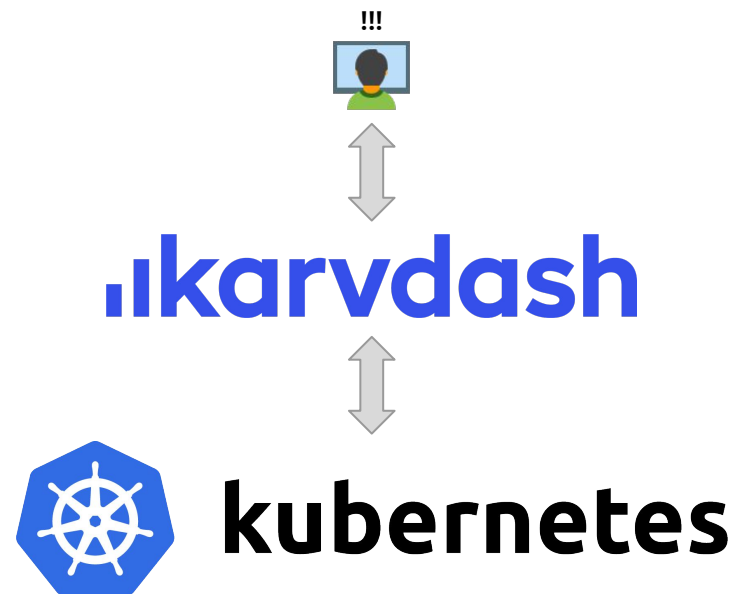


Scaling out to the cloud?

The solution

Provide a human interface to do actual work that is agnostic of infrastructure

- Hide the complexity
 - No devops
 - Minimal rewriting of the application, mostly packaging in containers
 - Well-known and standard tools, like notebooks and workflows to streamline the process
- Make it affordable
 - Run on any infrastructure (small or large scale: laptop or Cloud)
 - Autoscale to match requirements



Karvdash provides a user-friendly environment on top of Kubernetes

Notebook interface

- Combine paragraphs with different execution environments (shell, Python, etc.)
- Describe the configuration and write the code in the same context
- Portable, redistributable format
- Save the notebook with the data outputs for data provenance

The screenshot shows a Jupyter Notebook window titled "Untitled - Jupyter Notebook". The browser address bar shows the URL: <https://jupyterhub.platform.science-hangar.eu/user/admin/notebooks/Untitled.ipynb>. The notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and execution. The main content area displays the output of a cell, which includes GPU configuration details and the execution of TensorFlow code.

```

NVIDIA-SMI 440.82 Driver Version: 440.82 CUDA Version: 10.2
-----
GPU Name Persistence-M Bus-Id Disp.A Volatile Uncorr. ECC
Fan Temp Perf Pwr:Usage/Cap Memory-Usage GPU-Util Compute M.
-----
0 Quadro P1000 Off | 00000000:84:00:0 Off | 0% N/A
34% 31C P8 N/A / N/A | 0MiB / 4040MiB | 0% Default
-----
Processes:
GPU PID Type Process name GPU Memory
Usage
-----
No running processes found
-----
In [6]: import tensorflow as tf
print(tf.config.list_physical_devices('GPU'))
tf.debugging.set_log_device_placement(True)

# Create some tensors
a = tf.constant([[1.0, 2.0, 3.0], [4.0, 5.0, 6.0]])
b = tf.constant([[1.0, 2.0], [3.0, 4.0], [5.0, 6.0]])
c = tf.matmul(a, b)

print(c)

[PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
Executing op MatMul in device /job:localhost/replica:0/task:0/device:GPU:0
tf.Tensor(
[[22. 28.]
 [49. 64.]], shape=(2, 2), dtype=float32)

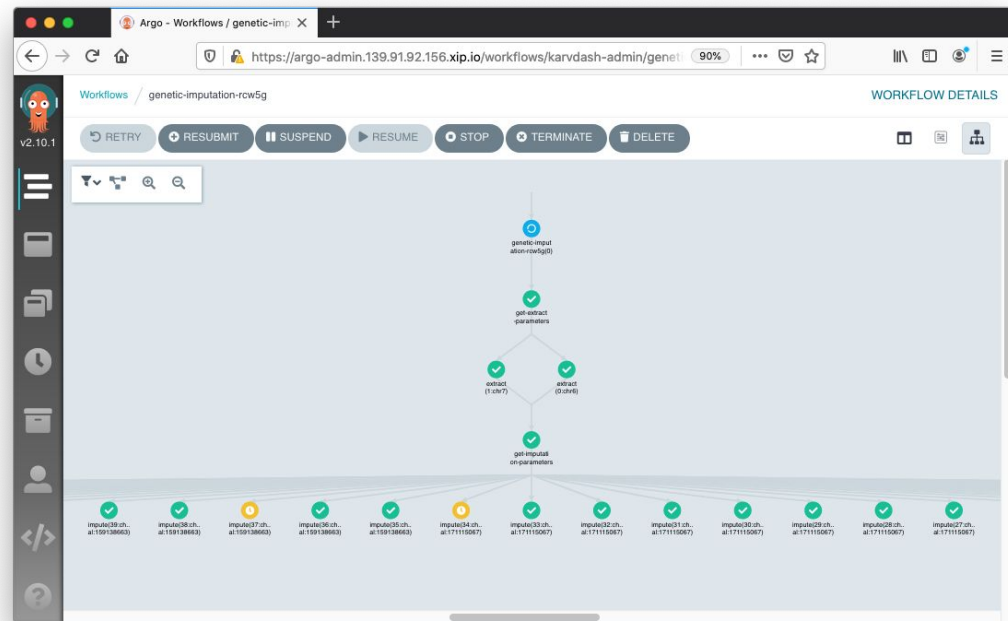
In [ ]:

```

Notebook example

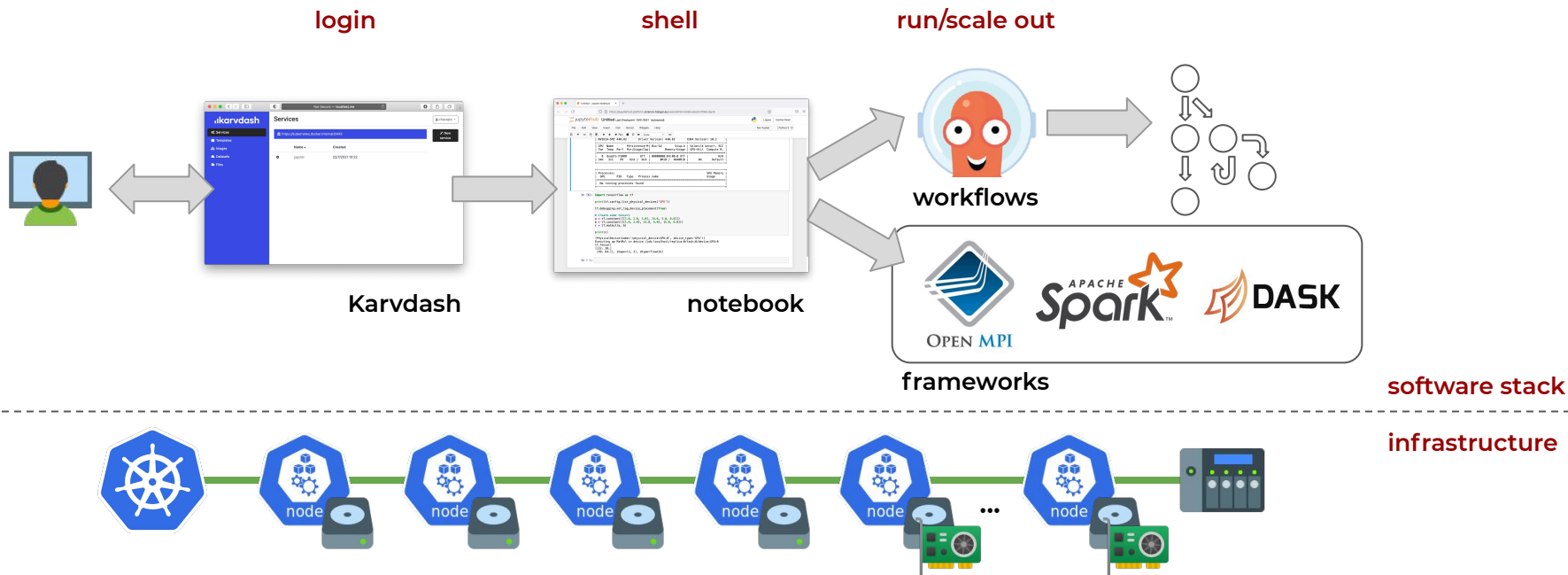
Workflows

- Describe execution as a DAG
- Extended features, such as loops, conditionals, recursion, etc.
- Each step is encapsulated in a container
- Scale out to all Kubernetes nodes
- Web-based interface
- Notebook integration



A running workflow with multiple parallel steps

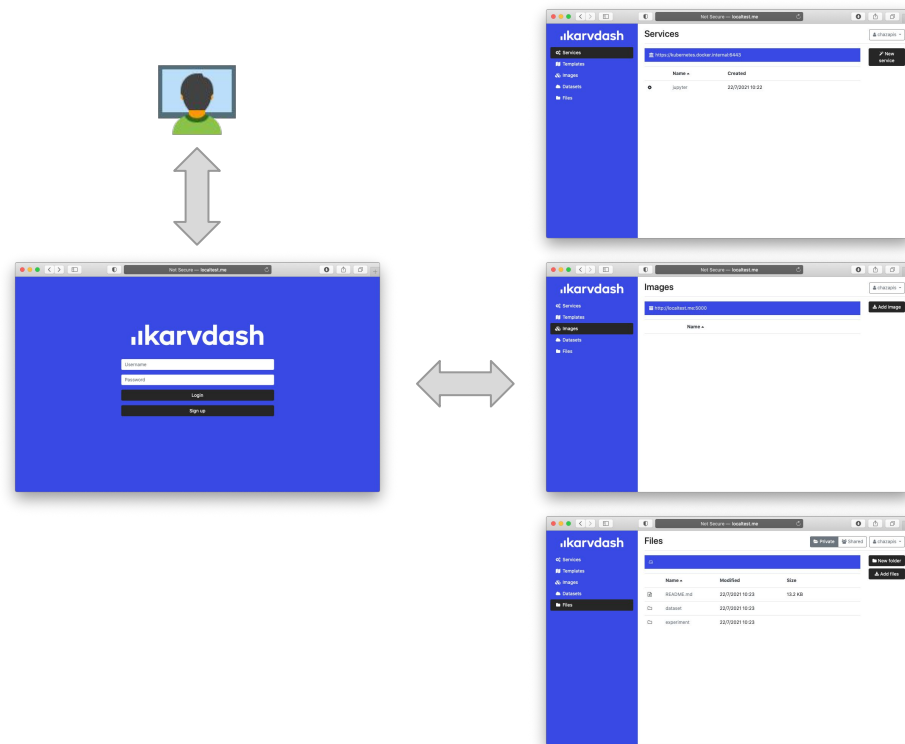
Overall layout and usage workflow



Clear separation of hardware infrastructure and software stack, allows to scale the hardware independently of the software

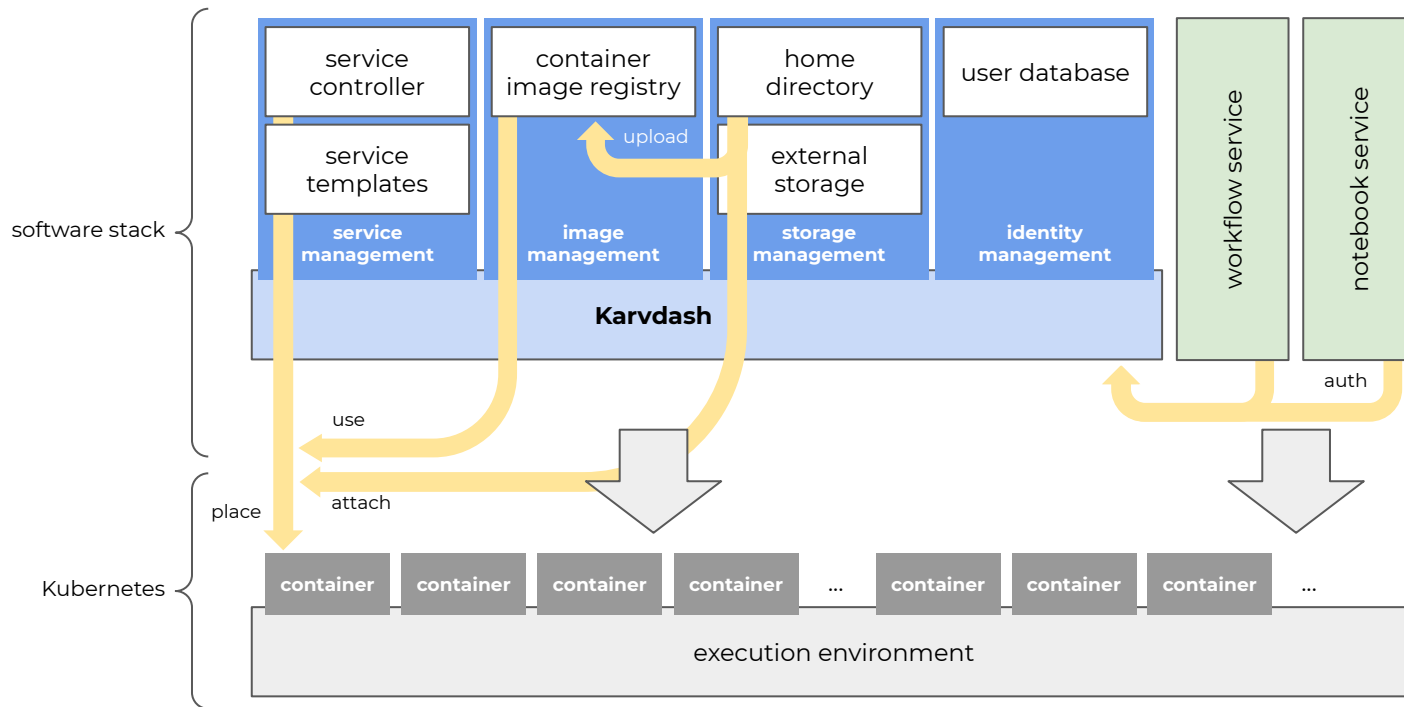
Karvdash features

- User management and a centralized login point
- Service management with customizable templates
- Private Docker registry interface
- File collections (automatically bound to all containers)
- External Cloud dataset connectivity (S3)
- Isolation of execution per user in separate namespaces
- External service provisioning over HTTPS
- User authentication for external services (OAuth 2.0/OIDC)
- API for integration with the notebook environment



Karvdash provides users service, container, and file management features

Internal components



Service templates

```

kind: Deployment
apiVersion: apps/v1
metadata:
  name: $NAME ←
spec:
  ...
  template:
    ...
    spec:
      containers:
        - name: $NAME ←
          image: postgres:12.3
          ports:
            - containerPort: 5432
          env:
            - name: POSTGRES_PASSWORD
              value: $PASSWORD ←
            - name: PGDATA
              value: ${PRIVATE_DIR}/.postgresql/data ←

```

```

kind: Template
name: PostgreSQL
description: Database server
singleton: yes
datasets: no
variables:
  - name: NAME
    default: postgresql
  - name: PRIVATE_DIR
    default: /private
  - name: PASSWORD
    default: postgres
    help: Password for postgres user

```

template
variables

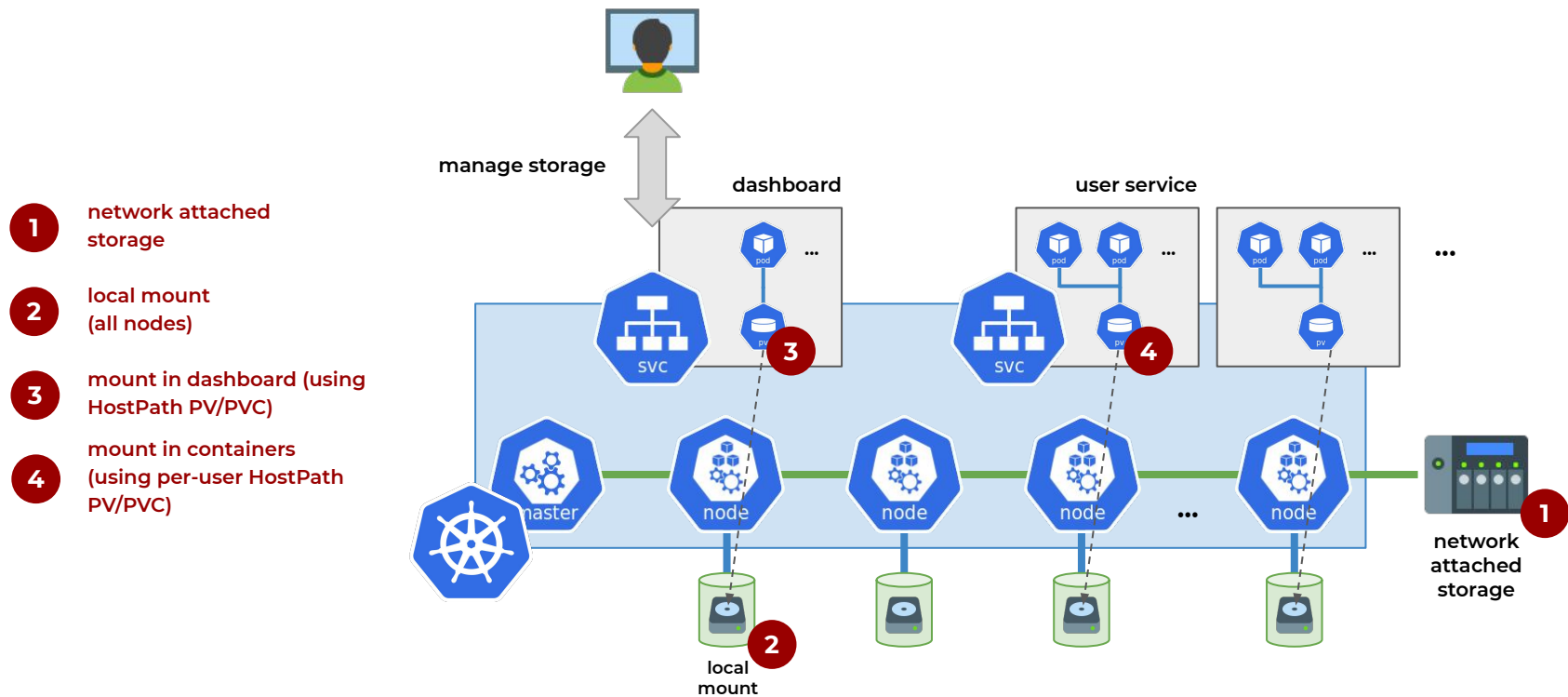


substitute in other
YAML parts



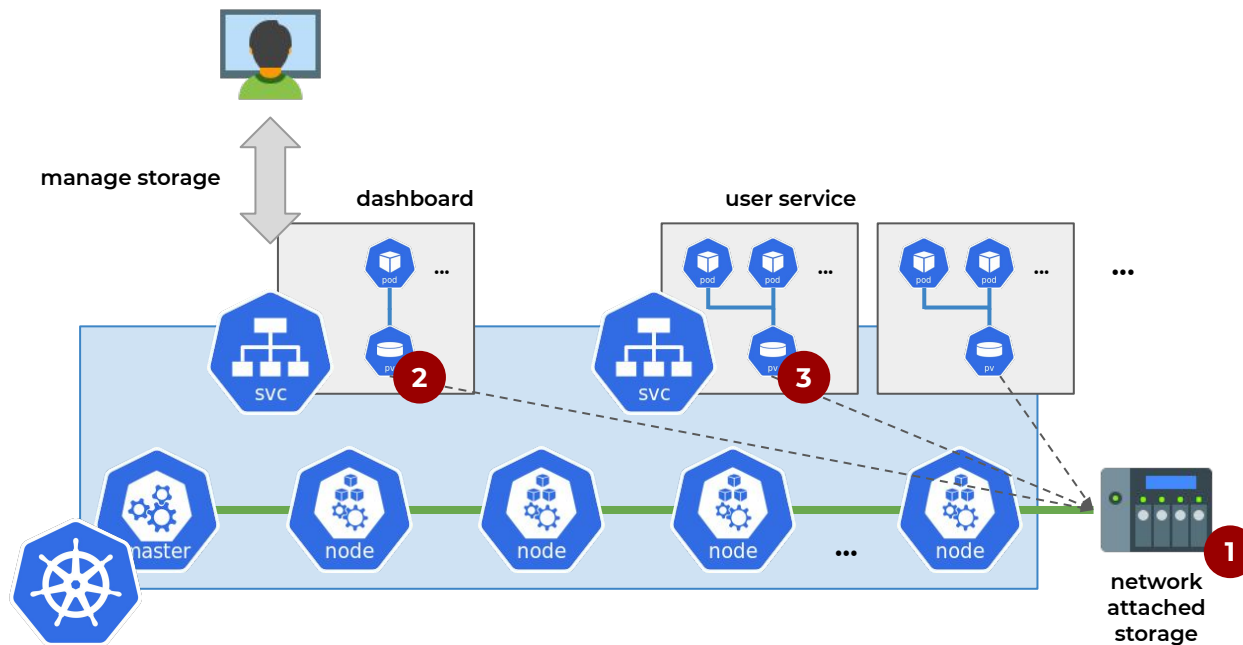
send to
Kubernetes

Storage handling with local storage



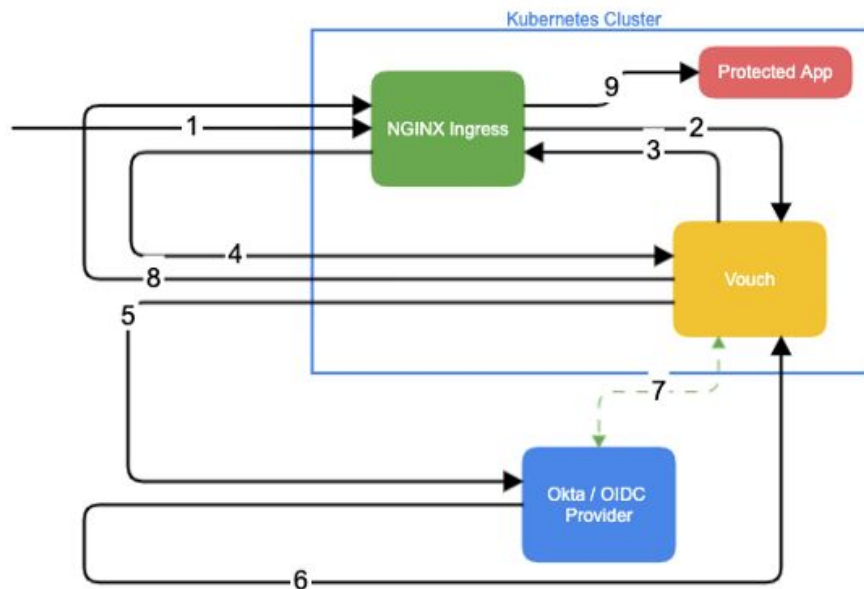
Storage handling with NFS

- 1 network attached storage
- 2 mount in dashboard (using NFS PV/PVC)
- 3 mount in containers (using per-user NFS PV/PVC)



Identity services

- Karvdash is an OAuth 2.0/OIDC-compatible identity provider
- We use Vouch identity proxy to authenticate users into their own services
- External services can use Karvdash for user authentication
- Karvdash is deployed side-by-side with Argo and JupyterHub



Source:

<https://medium.com/@msvechla/attribute-based-access-control-with-oidc-and-nginx-ingress-controller-in-kubernetes-169481ff2f22>

Development

Visit: <https://github.com/CARV-ICS-FORTH/karvdash>

Some details:

- Written in Django (Python)
- Documentation available (user and technical)
- Easy local development
- Helm chart and images for both amd64 and arm64 available
- Automated packaging and release of docs, container images, Python package, and Helm chart

Check our other projects: <https://github.com/CARV-ICS-FORTH>

Funded as part of the EVOLVE H2020 project:
<https://evolve-h2020.eu>

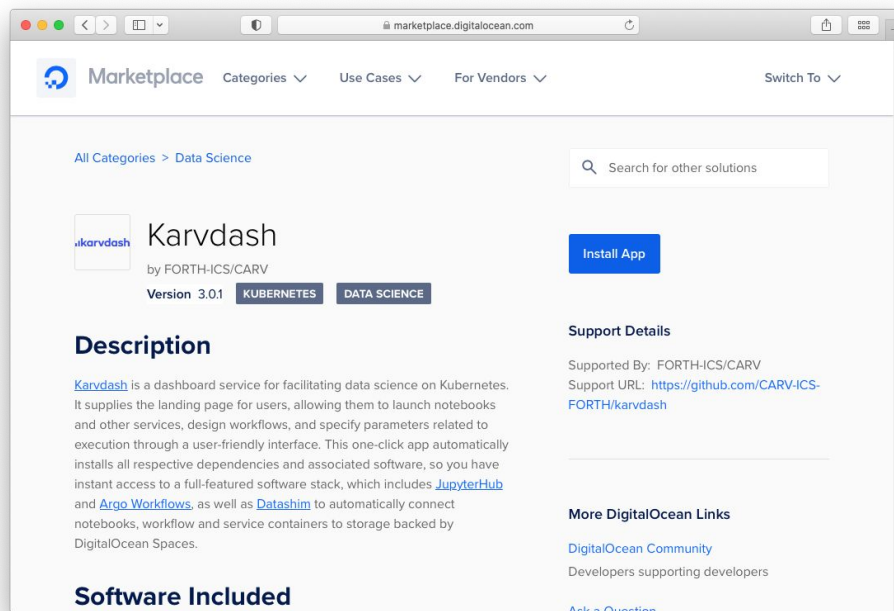


GitHub Pages

 **docker hub**



Easy evaluation



Try it out in the Cloud!

Full stack deployment available (with Argo and JupyterHub)

at: <https://marketplace.digitalocean.com/apps/karvdash>

Thank you!

The screenshot displays the GitHub repository page for `CARV-ICS-FORTH/karvdash`. The repository is public and has 11 stars and 0 forks. The main content area shows a list of files and folders with their commit history:

File/Folder	Description	Commit Date
<code>.github/workflows</code>	Do not build the chart for release candidate versions	4 months ago
<code>chart/karvdash</code>	Raise timeout values	2 months ago
<code>client</code>	Remove hard-coded themes, use volume to customize (#24)	4 months ago
<code>dashboard</code>	Set notebook directory permissions	3 months ago
<code>docs</code>	Update service layout image in docs	2 months ago
<code>karvdash</code>	Remove hard-coded themes, use volume to customize (#24)	4 months ago
<code>templates</code>	Fix MiniIO template	2 months ago
<code>.dockrignore</code>	Use Vouch Proxy for authentication of services	4 months ago
<code>.flake8</code>	Clean up	3 months ago
<code>.gitignore</code>	Use Vouch Proxy for authentication of services	4 months ago
<code>Dockerfile</code>	Configure Vouch secret in the proper namespace (#26)	2 months ago
<code>LICENSE</code>	Start working on dashboard	2 years ago
<code>Makefile</code>	Retry certificate until cert-manager webhook is ready	2 months ago

The right sidebar contains the following sections:

- About:** Karvdash provides a complete software stack for performing data science on Kubernetes. Includes links to Readme and Apache-2.0 License.
- Releases:** 35 tags.
- Packages:** No packages published.
- Languages:** A bar chart showing the distribution of code languages: Python (67.5%), HTML (22.3%), Makefile (4.2%), JavaScript (3.7%), Dockerfile (1.0%), CSS (0.9%), and Shell (0.4%).

<https://github.com/CARV-ICS-FORTH/karvdash>