

SCAILFIN

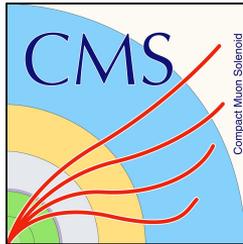
*Scalable CyberInfrastructure for Artificial Intelligence
and Likelihood Free Inference*

VC3, REANA, HPC-sites

Motivation

Major Multi-User Research Facilities involve the comparison of data collected from experiments with “synthetic” data, produced from computationally-intensive simulations.

Comparisons of experimental data and predictions from simulations are abstractions of the specific data analysis techniques developed by the respective communities over several decades. E.g.:



ICECUBE
SOUTH POLE NEUTRINO OBSERVATORY



Motivation

Many of these data analysis tasks are often conducted manually or through *ad hoc scripts* that might not be well maintained, difficulting reproducibility and reusability. Many of these tasks do have a well-defined workflow that make automation possible though.

REANA was created (in collaboration with DASPOS, DIANA and CERN) to address the reproducibility and reusability of the analysis pipeline.



Reproducible research data analysis platform

Motivation

In parallel:

Interest in leveraging Machine Learning (ML) and Artificial Intelligence (AI) techniques, to enhance the analysis of data from these facilities.

In particular, its application with emergent **Likelihood-Free Inference (LFI) techniques** when the predictions for the data are implicitly defined by the simulation, often leading to an intractable likelihood function. This can apply to analysis of data from LHC, LIGO, etc, but such Likelihood-Free algorithms have so far been **implemented mostly on individual machines and in ad hoc scripts.**

Motivation

SCAILFIN: Scalable CyberInfrastructure for Artificial Intelligence and Likelihood Free Inference

The SCAILFIN project aims to deploy artificial intelligence and likelihood-free inference techniques and software using scalable cyberinfrastructure (CI) that is developed to be integrated into existing CI elements, such as the REANA system.

The topic today:

SCAILFIN goals

- ...
- One of the goals of the project is **extending the REANA** platform to allow remote **submission** of workflows to **HPC facilities**.
- ...

Background

As mentioned before, SCAILFIN plans to use:

- **REANA** as the Cyber Infrastructure element to deploy AI and Likelihood-Free inference techniques.
- We are also leveraging on **VC3** (Virtual Clusters for Community Computation) in order to scale REANA to HPC resources.

Let's see a bit about these 2 components...

REANA: Reproducible Research Data Analysis Platform

Features

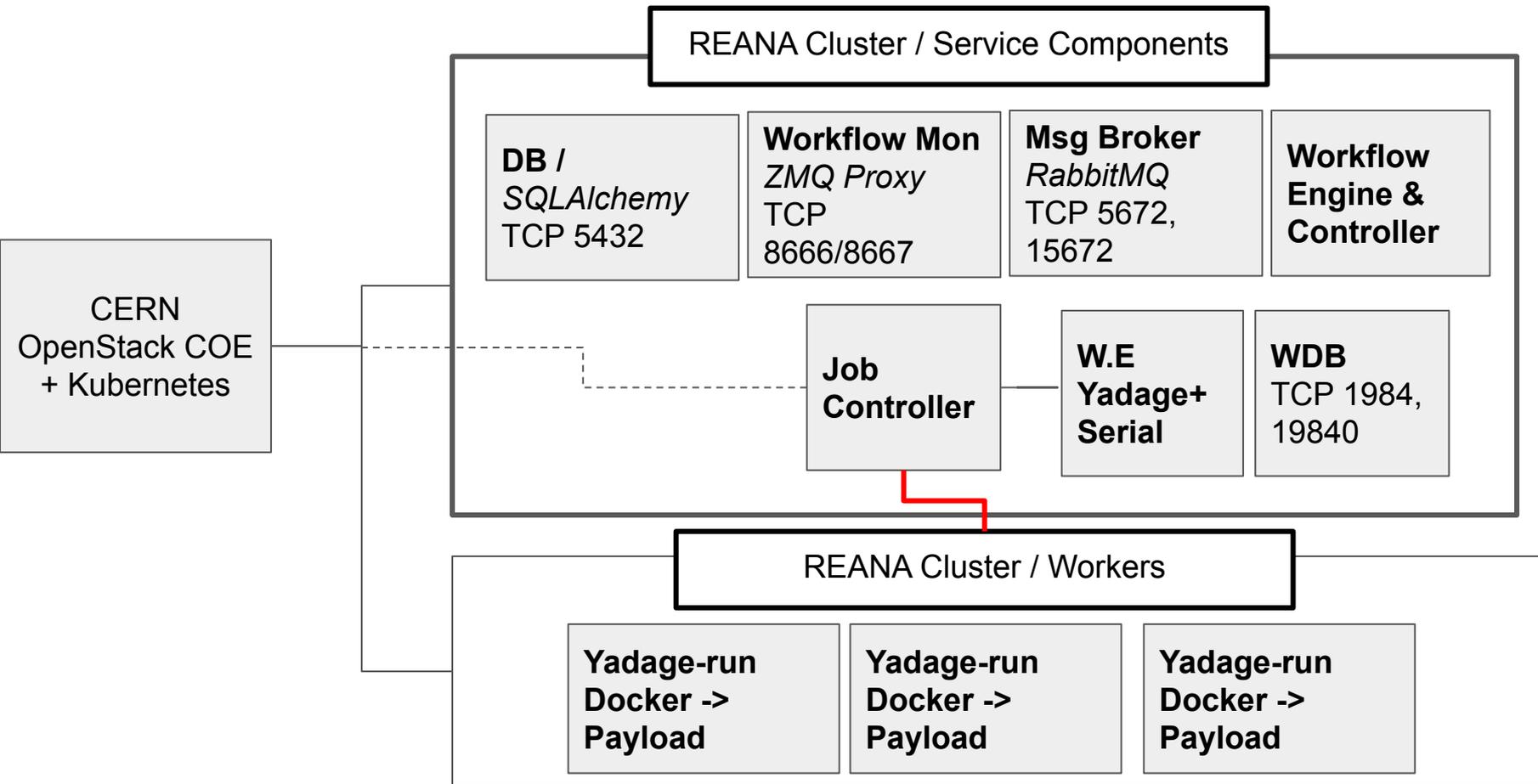
- Allows creation of tightly defined, container encapsulated workflows
- Whole purpose is to allow complete reproducibility
- Sharing workflows is as easy as sharing a specification
 - (and inputs!)
- Different workflow engines supported. E.g.:
 - CWL (Common Workflow Language) : <https://www.commonwl.org/>
 - Yadage (YAML based adage): <https://yadage.readthedocs.io>

REANA: Reproducible Research Data Analysis Platform

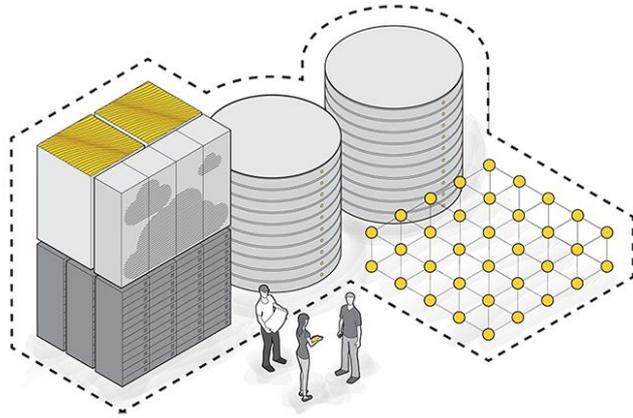
Components

- Two major components each consisting of many sub-components.
 - reana-client: User facing component.
 - Accepts workflows and is used as interface to entire REANA system (for user).
 - reana-cluster: Workhorse.
 - Consists of many small pieces which handle workflows, dish out jobs, coordinates results, can be thought of as the job scheduler. Jobs are scheduled via Kubernetes.

reana-cluster - Simplified Diagram



VC3: Virtual Clusters for Community Computation



VC3: A platform for provisioning cluster frameworks over heterogeneous resources for collaborative science teams

<https://www.virtualclusters.org>

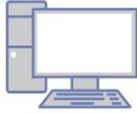
VC3: Virtual Clusters for Community Computation

VC3 Architecture

User adds an SSH public key to the system that will be used to grant access to the VC3 dynamic headnode

start here

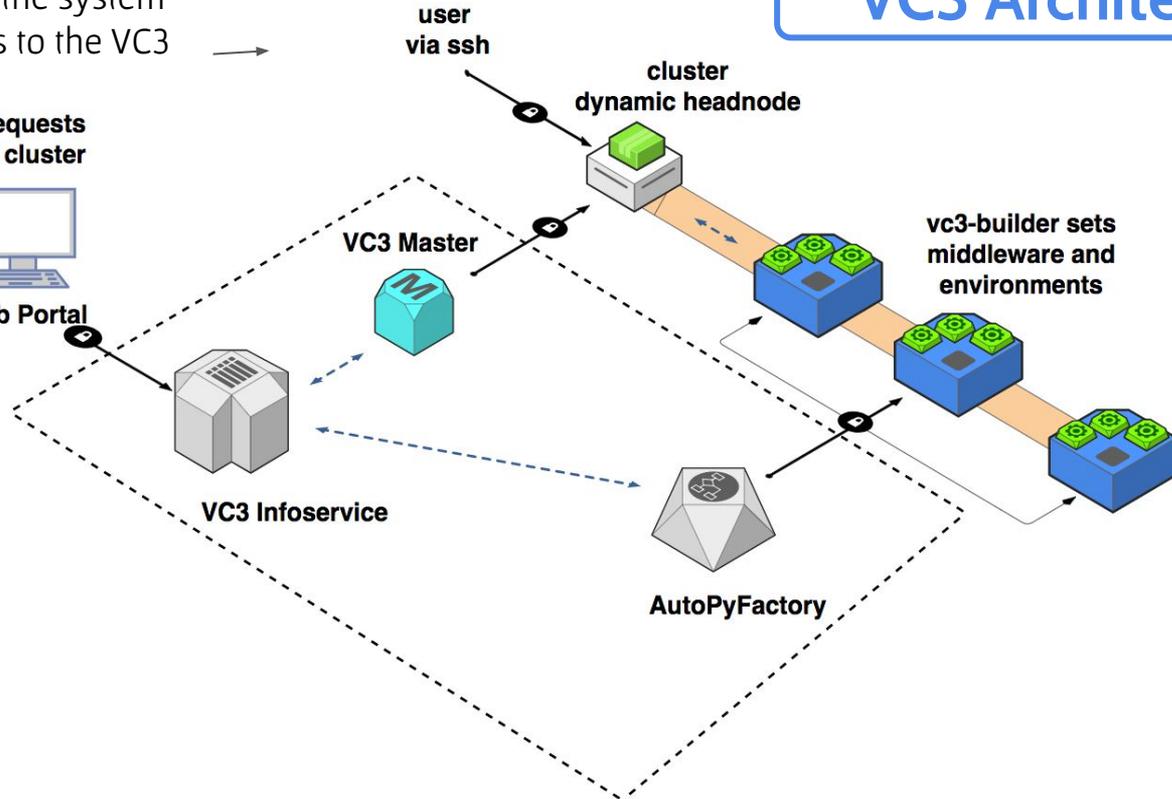
user requests virtual cluster



Web Portal

User defines an allocation. Two authentication mechanisms are supported:

- SSH keys: the user is handed a public ssh key to add to his/her submit node resource.
- GSI-SSH proxies: The user provides a proxy that VC3 can use.



VC3: Virtual Clusters for Community Computation

Features

- The user can select its own middleware for submission (E.g.: HTCondor, WorkQueue, Spark, REANA+HTCondor).
- It doesn't matter what the resource target batch system is (as long as it is supported by glite/blah, our translation layer for submission).

E.g.: Torque (Bluewaters), SLURM (NERSC, PSC-Bridges, Stampede2), HTCondor, LSF, SGE, PBS.

VC3: Virtual Clusters for Community Computation

Constraints

- At present, workers need outgoing network connection for a virtual cluster to work.
 - So, resources like ALCF/Theta are out of the scope with this approach.
 - But e.g.: XSEDE resources like NERSC, Bluewaters, Stampede or PSC-Bridges do meet the outgoing network requirement for example.

VC3: Virtual Clusters for Community Computation

Use-case example

Let's say we have an allocation to an HPC resource, or a pilot-based HTCondor pool like OSG Connect / CMS Connect / ATLAS Connect.

We want to use our allocation in this resource in order to create a SPARK cluster.

Creating a Spark Cluster – Step 1

← → ↻ <https://www.virtualclusters.org/resource>

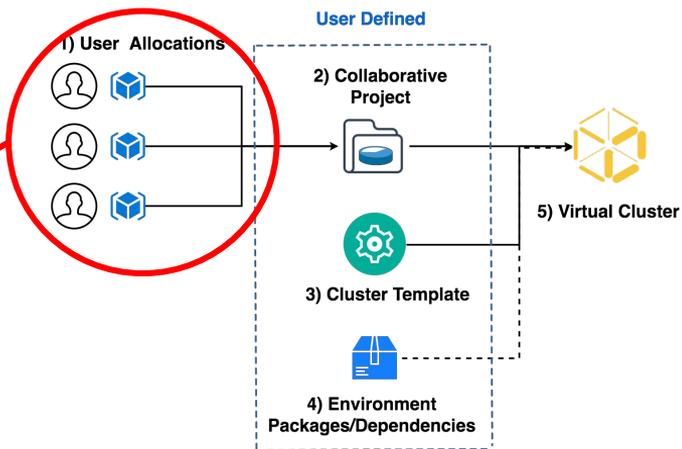
VC3  News Community Documentation

- Resources
- Allocations
- Projects
- Environments (beta)
- Virtual Clusters
- Monitoring
- Admin

	Research Computing Center (RCC)	Chicago Research Computing Center (RCC)
Stampede2	Texas Advanced Computing Center (TACC)	Stampede2 Super Computer
CMS Connect	CMS	CMS Connect
CoreOS	University of Chicago	CoreOS/Kubernetes Cluster with HTCondor Overlay
UCT3	University of Chicago - Enrico Fermi Institute	UChicago ATLAS Tier 3
ND CCL	University of Notre Dame Cooperative Computing Lab (CCL)	Notre Dame CCL Job Gateway

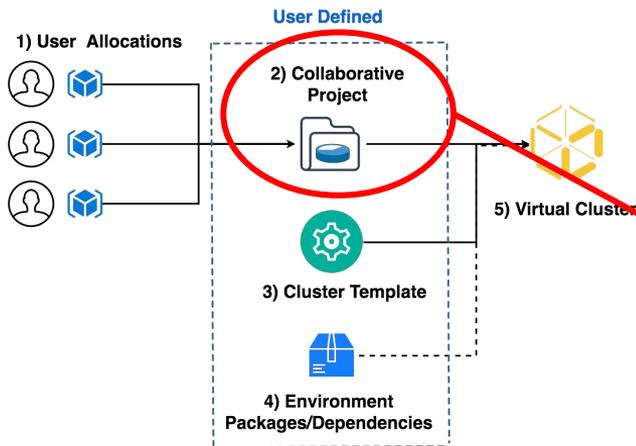
VC3 website:

<https://www.virtualclusters.org/>



Using CMS Connect to access the CMS Global Pool

Creating a Spark Cluster – Step 2



Create a New Project

A project connects your allocations to your team members

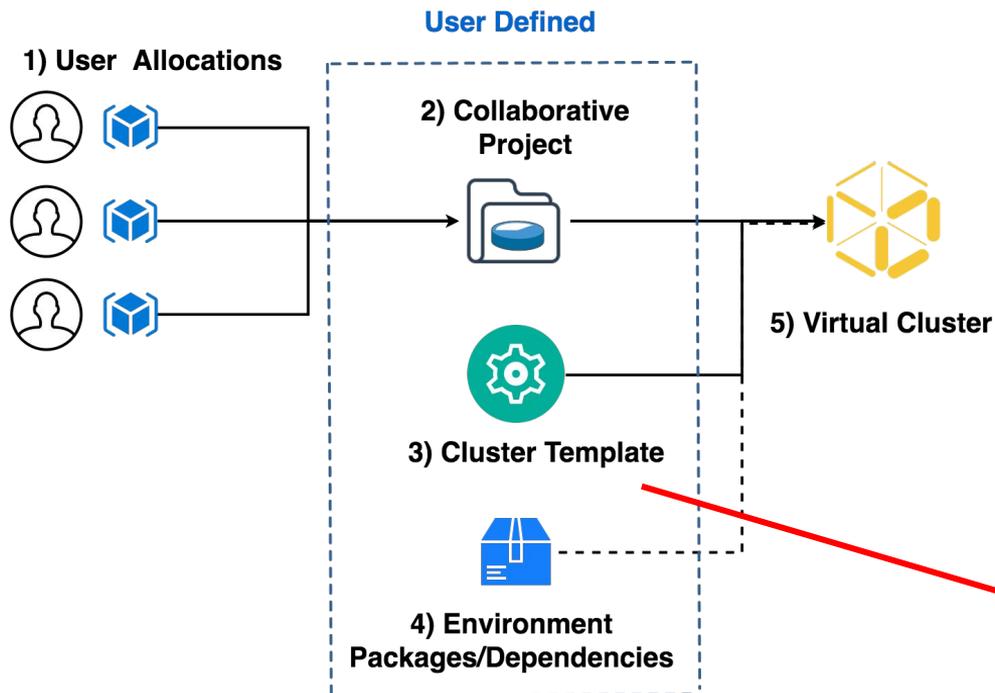
▪ - INDICATES REQUIRED FIELD

PROJECT NAME [▪]

PROJECT MEMBERS

SELECT ALLOCATION:

Creating a Spark Cluster – Step 3



Use CMS Connect to access the CMS Global Pool

Launch New Virtual Cluster

Project: vc3-team

* - INDICATES REQUIRED FIELD

VIRTUAL CLUSTER NAME (A-Z, 0-9, _ AND -)*

Your Virtual Cluster name

CLUSTER TEMPLATE FRAMEWORK*

- ✓ Please Select Framework
- HTCondor
- WorkQueue
- JupyterLab
- Spark**

Creating a Spark Cluster – Step 4



Spark Master at spark://128.135.158.246:7077

URL: spark://128.135.158.246:7077

REST URL: spark://128.135.158.246:6066 (*cluster mode*)

Alive Workers: 10

Cores in use: 20 Total, 0 Used

Memory in use: 74.2 GB Total, 0.0 B Used

Applications: 0 Running, 0 Completed

Drivers: 0 Running, 0 Completed

Status: ALIVE

Workers

Worker Id	Address
worker-20181024210235-169.228.132.112-39013	169.228.132.112
worker-20181024210254-169.228.132.112-33352	169.228.132.112
worker-20181024210842-169.228.131.229-37426	169.228.131.229
worker-20181024212048-169.228.132.142-39969	169.228.132.142
worker-20181024212233-169.228.130.186-30117	169.228.130.186
worker-20181024212246-169.228.130.186-5423	169.228.130.186
worker-20181024212423-169.228.131.46-42113	169.228.131.46
worker-20181024212423-169.228.131.46-43409	169.228.131.46

Virtual Cluster: khurtado-spark_ucsd

Terminate Cluster

STATE OF VIRTUAL CLUSTER

Running

All requested compute workers are running.

Owner

Kenyi Anampa

Project

sparkcms

Status

KHURTADO-SPARK_UCSD

Cluster Framework: spark

Requested 10

Running 10

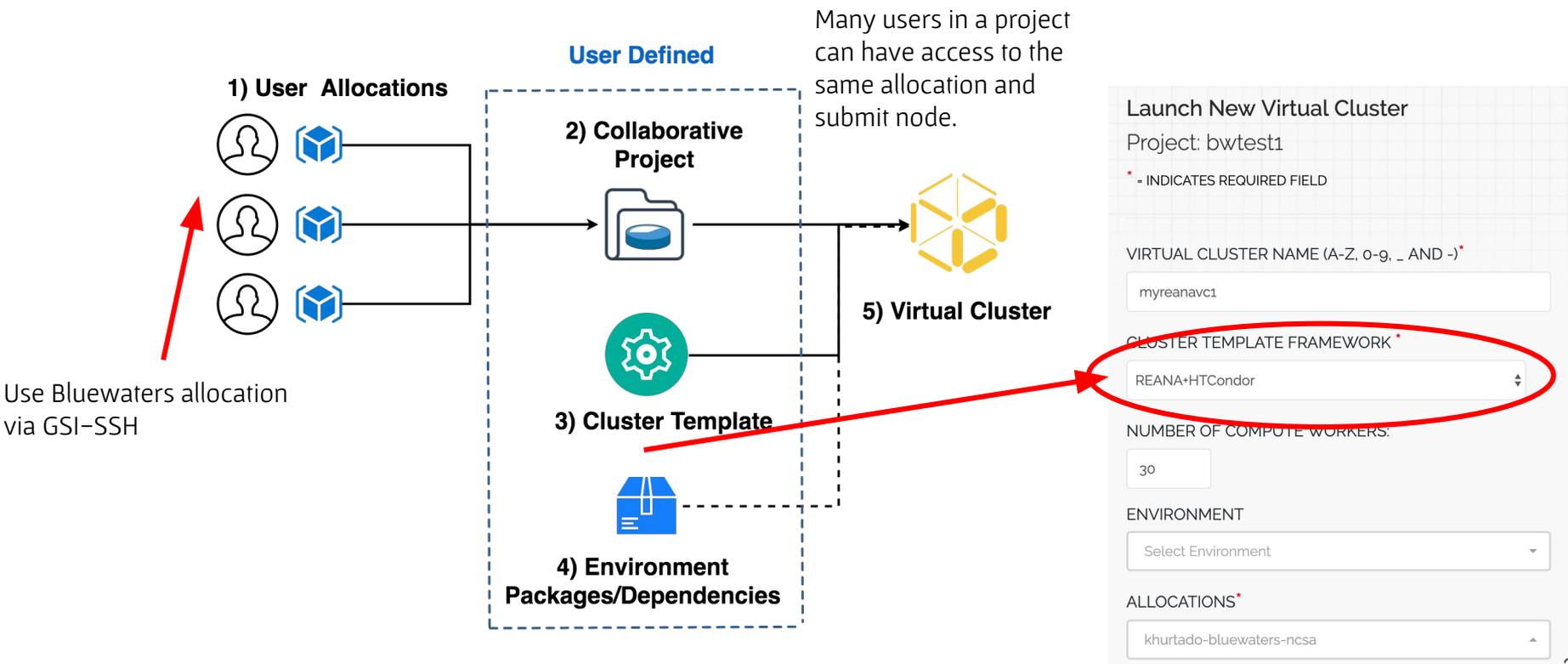
Queued 0

Error 0

Putting it all together

REANA + VC3 with HTCondor on Blue Waters
(in progress)

Creating a REANA Cluster Template for VC3



Creating A REANA cluster on Blue Waters

Portal / Allocations / Register New Allocation

Register New Allocation

An allocation is an allotted amount of CPU hours or service units (SU) of computing time on a specific resource

* = INDICATES REQUIRED FIELD

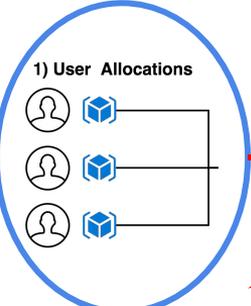
RESOURCE *

Blue Waters

ACCOUNT NAME ON RESOURCE *

khurtado

Cancel Register Allocation



```
hurtadoa@h2ologin1:~> myproxy-logout -s  
tfca.ncsa.illinois.edu -p 7512  
Enter MyProxy passphrase:  
A credential has been received for user hurtadoa in  
/tmp/x509up_u57980.  
hurtadoa@h2ologin1:~> cat /tmp/x509up_u57980
```

Register New Allocation

An allocation is an allotted amount of CPU hours or service units (SU) of computing time on a specific resource

* = INDICATES REQUIRED FIELD

PRIVATE TOKEN X509 PROXY *

```
-----BEGIN CERTIFICATE-----  
MIIEUzCCAzugAwIBAgIDAt84MAoGCSqGSIb3DQEBCwUAMIG  
GMQswCOYDVQQGEwJV  
UzE4MDYGA1UEChMvTmFoaWVWwGQzVudGVyIGZvcjBTd  
XBldmNybXB1dGluZyBB  
cHBsaWVWwGQzVudGVyIGZvcjBTd  
GhvcmloaWVzMRsw  
GQYDVQQDEwUud28gRmFidG9vLENBIDlwMTMwHhcNMTkw  
-----END CERTIFICATE-----
```

Cancel Register Allocation



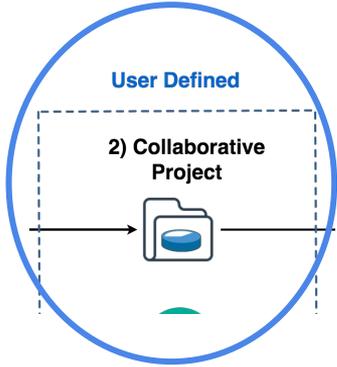
Allocation authentication mechanisms

The auth mechanisms available in VC3 are:

- SSH keys
- GSI tokens
 - Available on XSEDE HPC Systems
 - Requires renewal. The website shows when the proxy expires and the "Edit Allocation Name" button allows users to change
 - Some HPC centers have procedures to increase the default 12 hours expiration time to e.g.: 10 days (NERSC case, and previously done at BW)

The screenshot shows the 'Allocation: khurtado-bluewaters-ncsa' page. At the top, there are navigation links: 'Portal Home / Allocations / khurtado-bluewaters-ncsa'. Below the title, there are two buttons: 'Edit Allocation Name' (circled in red) and 'Delete Allocation' (circled in red). A green bar indicates the allocation is 'Ready'. The main text states: 'Allocation is ready to be used. This allocation may be added to any project in order to launch a Virtual Cluster.' The page is divided into three steps: 'Step 1: Log Into Resource' (with terminal command: `ssh hurtadoa@h2ologin.ncsa.illinois.edu`), 'Step 2: Access Resource' (with password field: `h2ologin.ncsa.illinois.edu`), and 'Step 3: Add Allocation SSH Public Key to Resource' (with a 'Copy to' link). On the right side, there are three panels: 'Owner' (Kenyi Hurtado), 'Resource' (Blue Waters, Account Name: hurtadoa), and 'Expiration' (5 hours, 55 minutes and 49 seconds, circled in red).

Creating A REANA cluster on Blue Waters



Portal Home / Projects / **bwtest1**

Project: **bwtest1**

Description: None Delete Project

Members **Allocations**

Project Members Filter

Name	Email	Organization	
Kenyi Hurtado (Owner)	khurtado@nd.edu	University of Notre Dame	
Cody Kankel	ckankel@nd.edu	University of Notre Dame	Rem

SELECT NEW USERS

Nothing selected

Portal Home / Projects / **bwtest1**

Project: **bwtest1**

Description: None Delete Project

Members **Allocations**

Allocations Filter

Name	Owner	Organization	
khurtado-bluewaters-ncsa	Kenyi Hurtado	University of Notre Dame	Rem

SELECT ALLOCATION:

Nothing selected

Add Allocation to Project

25

Creating A REANA cluster on Blue Waters

Launch New Virtual Cluster

Project: bwtest1

* = INDICATES REQUIRED FIELD

VIRTUAL CLUSTER NAME (A-Z, 0-9, _ AND -)*

reanabwv1

CLUSTER TEMPLATE FRAMEWORK*

REANA+HTCondor

NUMBER OF COMPUTE WORKERS:*

2

ENVIRONMENT

Select Environment

ALLOCATIONS*

khurtado-bluewaters-ncsa

EXPIRATION

If not specified, expiration defaults to 6 hours from launch, when your virtual cluster will automatically be terminated.

HOURS:

98

Cancel

Launch Virtual Cluster

Creating A REANA cluster on Blue Waters

My Virtual Clusters Filter						
Name	Project	Head Node	Workers	State	Cluster Template	
khurtado-ndvc1	scailfin-dev	128.135.158.232	<ul style="list-style-type: none">3 Requested3 Running0 Queued0 Error	<div>Running</div> <p>All requested compute workers are running.</p>	reana+htcondor	
khurtado-reanabwv1	bwtest1	128.135.158.188	<ul style="list-style-type: none">2 Requested2 Running2 Queued0 Error	<div>Running</div> <p>Requesting 0 less compute worker(s).</p>	reana+htcondor	

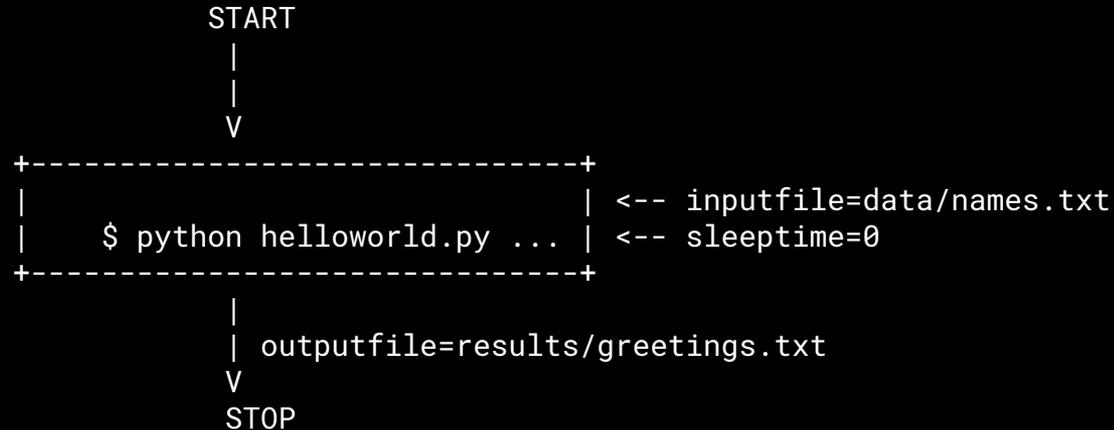
End result will be A REANA cluster deployed on the VC3 headnode

Components are deployed via Kubernetes (minikube)

```
(reana) [khurtado@khurtado-reanabwv1 ~]$ reana-cluster status
COMPONENT      STATUS
job-controller  Running
server          Running
db              Running
workflow-controller Running
message-broker  Running
REANA cluster is ready.
(reana) [khurtado@khurtado-reanabwv1 ~]$ kubectl get pods
NAME                                                    READY   STATUS    RESTARTS   AGE
batch-serial-7e79ee48-036f-4049-87ee-a3dc66d8a1da-tl7zd 0/1     Completed 0           5h54m
db-69744557df-wg4mt                                     1/1     Running   0           5h55m
job-controller-5c7f4c8b4f-sgnj6                         1/1     Running   0           5h55m
message-broker-b7d66cf55-m9p4n                          1/1     Running   0           5h55m
server-58dc985c77-n2qpn                                  2/2     Running   0           5h55m
workflow-controller-668f69d4bc-x62w7                    2/2     Running   0           5h55m
```

reana-client

Basic Example



1. Create workflow in reana-client
2. Upload code (helloworld.py) and data (names.txt)
3. Start workflow
 - a. Executes Python code and creates a result directory with the names found in names.txt
4. Download results

reana-client

Basic Example

```
$ reana-client create -n my_stuff
my_stuff.1
$ export REANA_WORKON=my_stuff
$ reana-client upload ./code ./data
File code/helloworld.py was successfully uploaded.
File data/names.txt was successfully uploaded.

$ reana-client start
my_stuff is running
$ reana-client status
NAME          RUN_NUMBER    CREATED                STATUS    PROGRESS
my_stuff      1             2019-06-04T21:13:21   running  0/1
$ reana-client status
NAME          RUN_NUMBER    CREATED                STATUS    PROGRESS
my_stuff      1             2019-06-04T21:13:21   finished 1/1

$ reana-client ls

NAME          SIZE    LAST-MODIFIED
code/helloworld.py  2991   2019-06-04T20:24:36
data/names.txt      20     2019-06-04T20:24:36
results/greetings.txt 67     2019-06-04T20:25:40
```

reana-client

Results from Blue Waters

```
(reana)[ckankel@ckankel-bw_scaifin_test1 reana-demo-helloworld]$ reana-client ls
```

NAME	SIZE	LAST-MODIFIED
code/helloworld.py	2991	2019-06-04T20:24:36
data/names.txt	20	2019-06-04T20:24:36
results/greetings.txt	67	2019-06-04T20:25:40

```
(reana)[ckankel@ckankel-bw_scaifin_test1 reana-demo-helloworld]$ reana-client download
```

```
File results/greetings.txt downloaded to /home/ckankel/reana-demo-helloworld.
```

```
(reana)[ckankel@ckankel-bw_scaifin_test1 reana-demo-helloworld]$ cat results/greetings.txt
```

```
Working from hostname: nid25354
```

```
Hello Jane Doe!
```

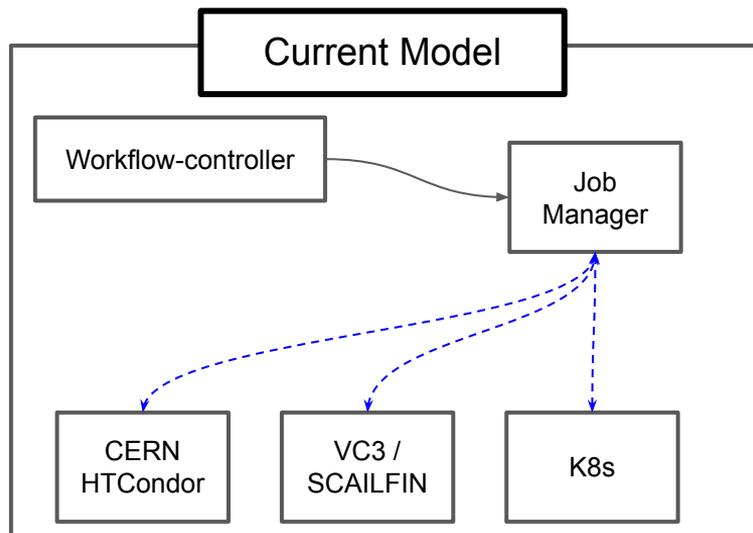
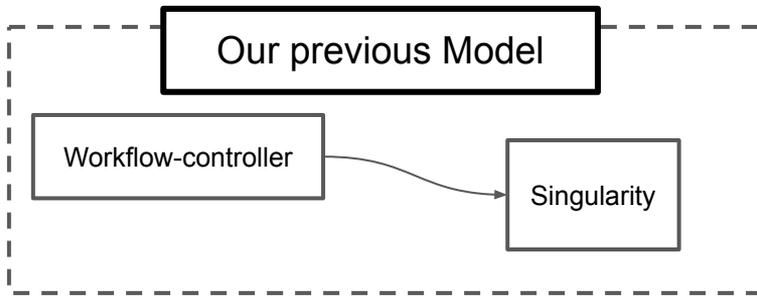
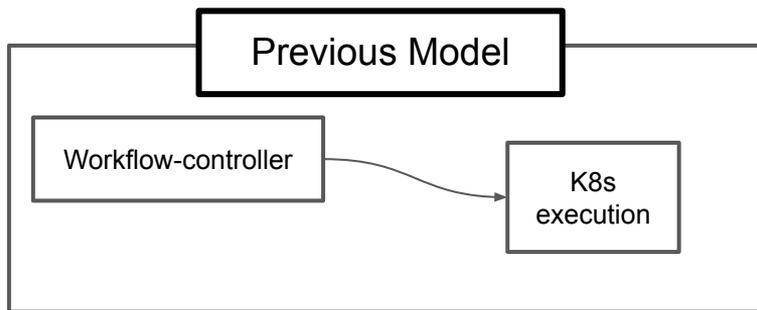
```
Hello Joe Bloggs!
```

SCAILFIN's REANA modifications

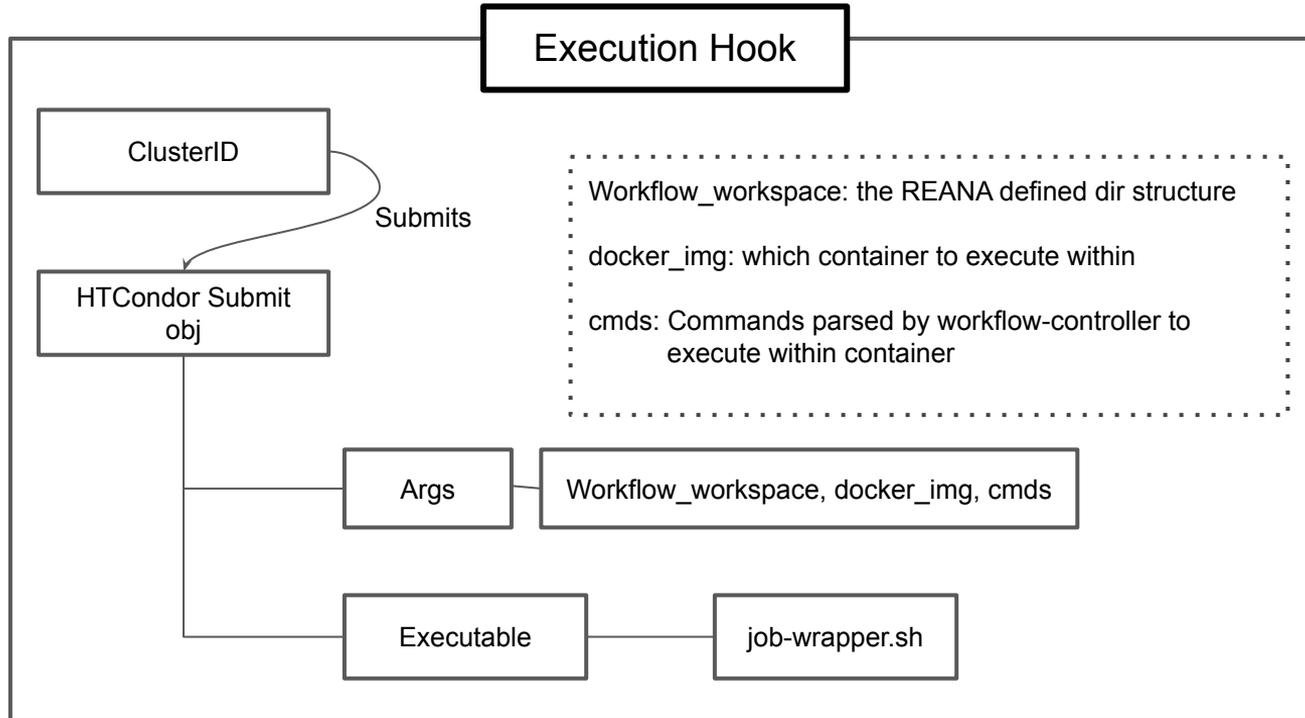
- REANA requires some form of docker supporting container technology
 - Singularity and Shifter support in the works. Possibly CharlieCloud
- REANA expects to submit to a kubernetes cluster
 - Added support for VC3 specialized HTCondor submissions through a modified reana-job-controller and a job_wrapper for every workflow step.
 - The modified reana-job-controller submits each workflow step to a local condor scheduler
- Job wrapper
 - Each workflow step is wrapped by a script which searches for container technology and launches each workflow step into the available container (shifter, singularity)

SCAILFIN's REANA modifications

- REANA developers allowed abstraction of reana-job-controller's backend
 - Allows for “plug and play” of backends / job execution component

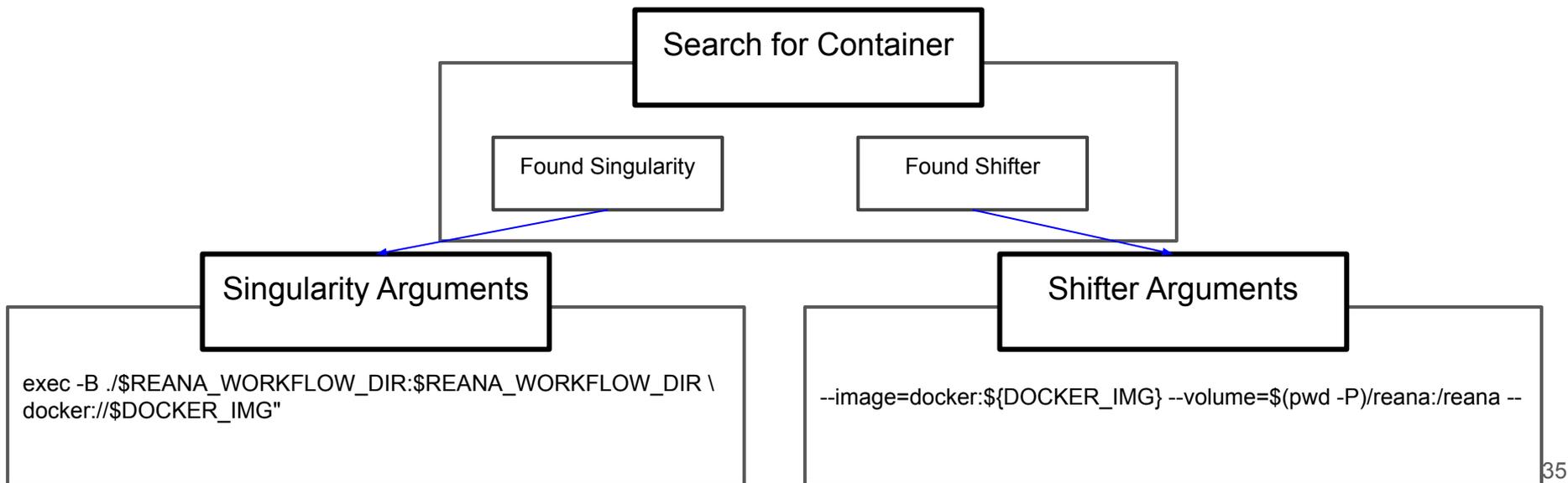


SCAILFIN's REANA modifications



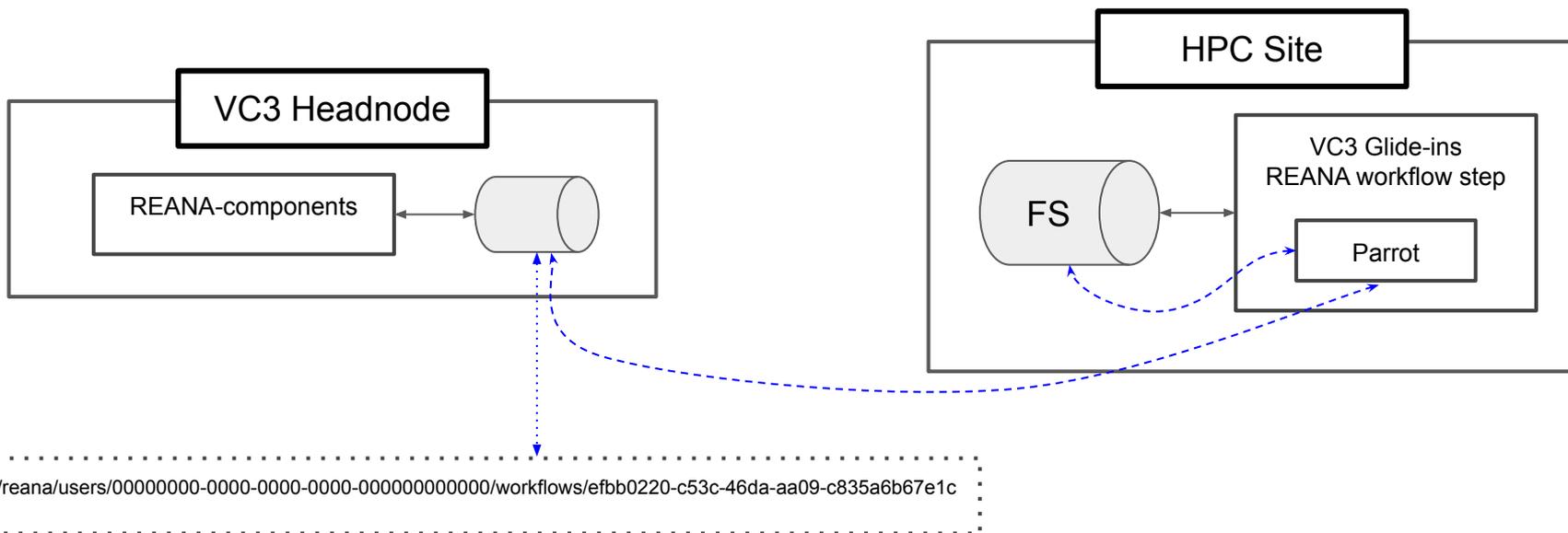
Job-wrapper Container management

- Performs check for available container technologies
 - Checks Binaries in \$PATH (VC3 may auto-load these)
 - Attempts to load modules for Singularity and Shifter
 - Executes workflow-step within discovered container
 - Will choose depending on the currently set \$default



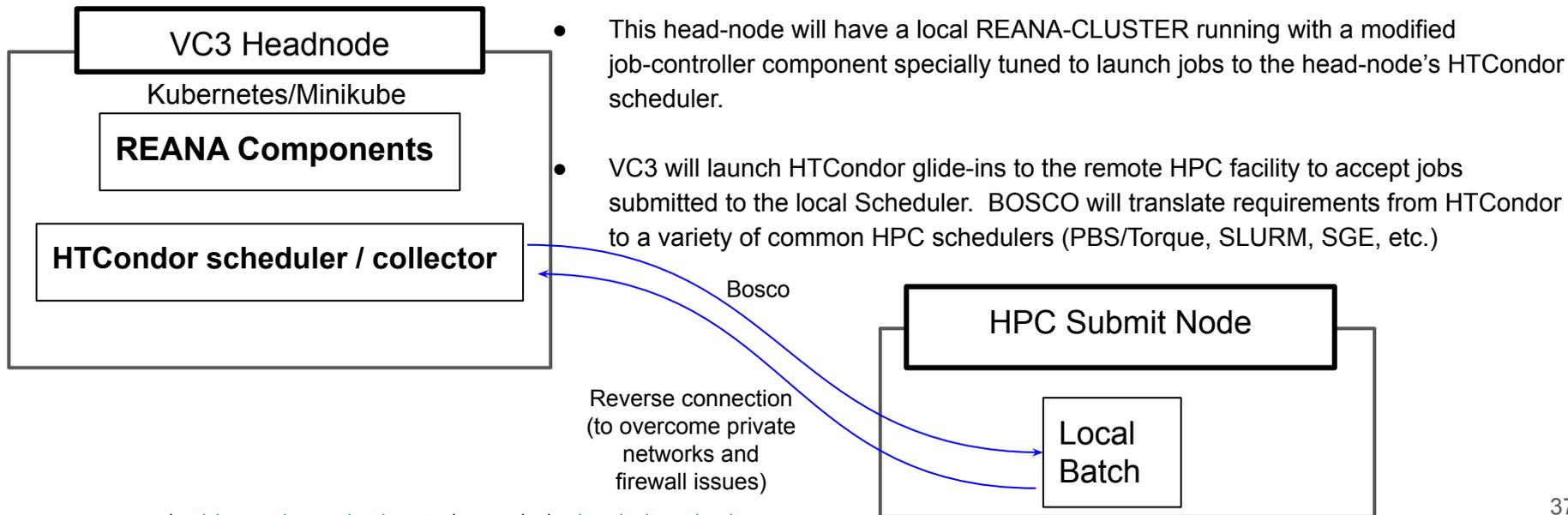
Job-wrapper File management

- REANA expects files to be where the workflow controller placed them
 - We must simulate this directory structure on the HPC infrastructure
- We can easily transfer entire directory structures with HTCondor's [chirp protocol](#) and [Parrot](#)



SCAILFIN and VC3

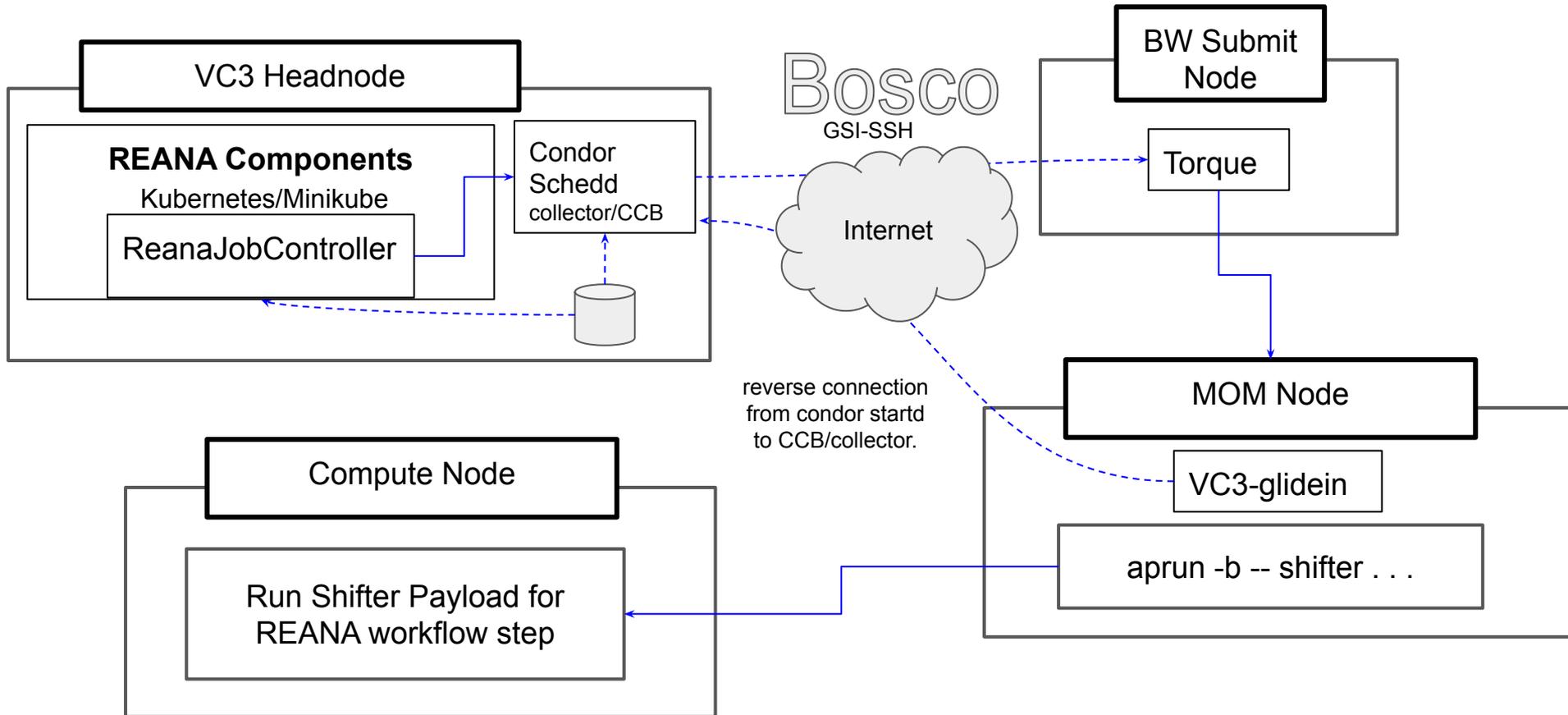
VC3 is utilized for remote connections to clusters.



SCALFIN's VC3 modifications

- Cluster template for REANA+HTCondor
 - Uses the standard HTCondor template as the base to create a condor pool that sends jobs to HPC resources, translating the job to the corresponding batch system submission syntax via bosco.
 - Deploys Kubernetes via minikube
 - Deploys the REANA cluster and client and set up the environment, so the user can interact with them as soon as the VC3 headnode is created.
- GSI-SSH support
 - The GSI-SSH authentication mechanism was added in the infrastructure, in order to support e.g. XSEDE HPC centers like Blue Waters, Stampede, NERSC.
 - Proxies can be renewed through the VC3 website for the virtual cluster allocations to remain active.
- Additional patches to support Cray Linux environments (NERSC, Blue Waters)

SCALFIN on Blue Waters



Links

- SCAILFIN Source code:

- SCAILFIN's modified RJC
https://github.com/scailfin/reana-job-controller/tree/job_manager
- REANA
 - <https://github.com/reanahub>
- VC3
 - <https://github.com/vc3-project>

- Websites

- <https://www.virtualclusters.org>
- <http://www.reanahub.io/>

Contacts

- Kenyi Hurtado
 - khurtado@nd.edu
- Cody Kankel
 - ckankel@nd.edu