



Uniform Resource Access Compute and Cloud Resources at JSC

2024-05-22 | Björn Hagemeier | Juelich Supercomputing Centre



Workflows



Jobs

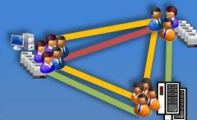


Data Management



Discovery

Services



Federations



Part I: UNICORE

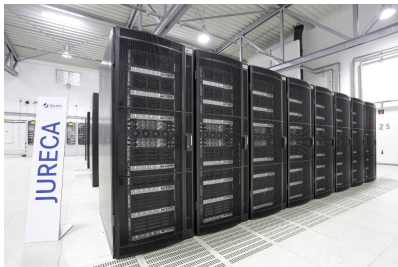
Motivation

Differences of systems

Uniform Interface to Computing Resources

Various RMS on systems

- JUQUEEN: IBM LoadLeveler
- JURECA: Slurm
- Different job description languages for specifying # of nodes, memory requirements, wall time, ...
- Different parameters on the command line
- Unify and simplify supercomputer access



Slurm



Load Leveler

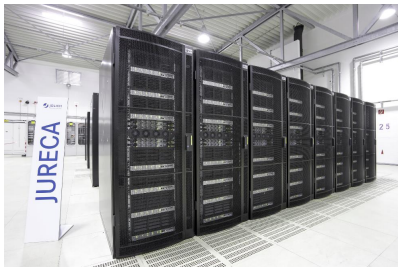
Motivation

Differences of systems

Uniform Interface to Computing Resources

Various RMS on systems

- JUQUEEN: IBM LoadLeveler
- JURECA: Slurm
- Different job description languages for specifying # of nodes, memory requirements, wall time, ...
- Different parameters on the command line
- Unify and simplify supercomputer access



Slurm



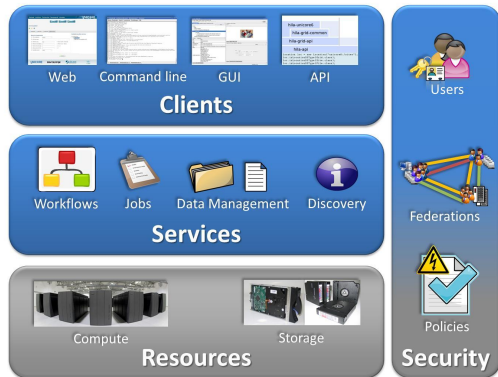
Slurm

Why UNICORE

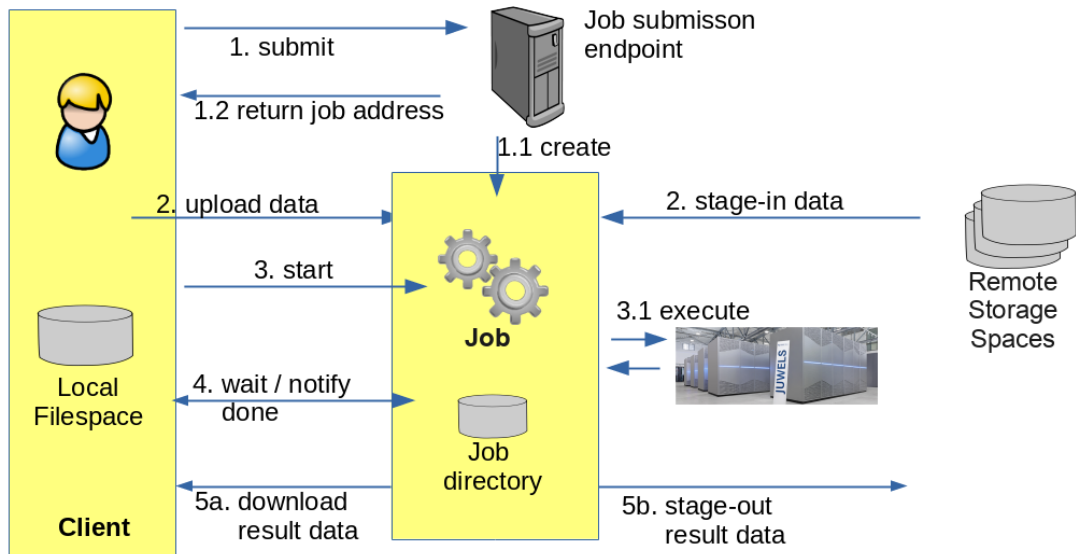
Advantages

- Hide system specific commands
- Create, submit and monitor jobs
 - Seamless, secure, and intuitive access to distributed compute and data resources
- Multiple clients
- Integrated **data management**
- **Federated identities**
- Open Source:
<https://github.com/UNICORE-EU>

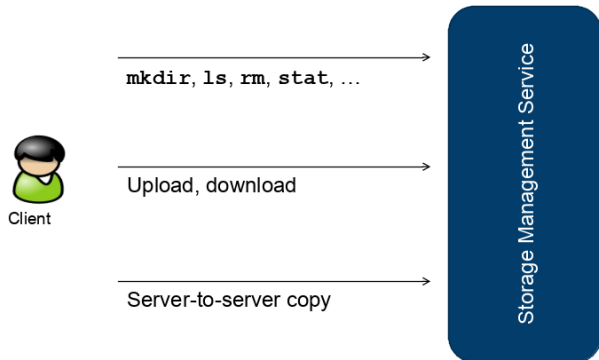
UNICORE



Job execution model



Data management and file transfer



- File Systems

- Apache HDFS



- S3



- iRODS



Efficient file transfer

UFTP

- Data **streaming** library and **file transfer** tool
- Fully integrated into UNICORE
- Standalone (non-UNICORE) client available
- Client to server and server to server data transfers
- Data staging among UFTP-enabled sites
- Efficient **synchronization** of individual local and remote files using the **rsync** algorithm
- Optional **compression** and **encryption** of data streams

Efficient file transfer

UFTP

- Data **streaming** library and **file transfer** tool
- Fully integrated into UNICORE
- Standalone (non-UNICORE) client available
- Client to server and server to server data transfers
- Data staging among UFTP-enabled sites
- Efficient **synchronization** of individual local and remote files using the **rsync** algorithm
- Optional **compression** and **encryption** of data streams
- Awarded “best systemic approach” in SC Asia Data Mover Challenge 2020



Source: SC Asia web site

PyUNICORE API

Features

- Job submission and monitoring
- File transfer handling
- Mounting filesystems remotely via UFTP
- Workflow management

```
$ pip install pyunicore[crypto,fs,fuse]
```

```
import pyunicore.client as uc_client
import pyunicore.credentials as uc_credentials
import json

base_url = "https://localhost:8080/DEMO-SITE/rest/core"

# authenticate with username/password
credential = uc_credentials.UsernamePassword("demouser", "test123")
transport = uc_client.Transport(credential)

client = uc_client.Client(transport, base_url)
print(json.dumps(client.properties, indent = 2))
```

Clients and APIs

- Commandline tools

- UNICORE Commandline Client (UCC): <https://sourceforge.net/projects/unicore/files/Clients/Commandline%20Client/>
- UFTP client for high-performance data access:
<https://sourceforge.net/projects/unicore/files/Clients/UFTP-Client/>

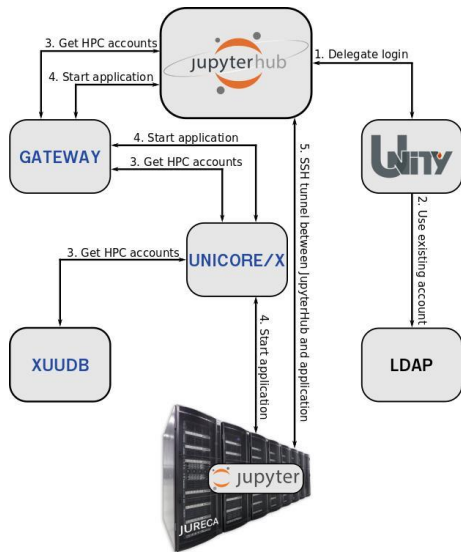
- RESTful APIs

- curl, Python Requests
- https://sourceforge.net/p/unicore/wiki/REST_API/
- PyUNCIORE client library: <https://github.com/HumanBrainProject/pyunicore>

Jupyter Hub @JSC

HPC in your web browser

- UNICORE is an integral part of the Jupyter offering at JSC
- Start Jupyter Labs on JUWELS, JURECA-DC, JUSUF, DEEP, HDFML, or a cloud based VM
- <https://jupyter-jsc.fz-juelich.de/>
- Foundation for NFDI base service Jupyter4NFDI



Additional information and support

UNICORE

- Project web site: <https://www.unicore.eu/> for downloads and documentation
- Product support: unicore-support@lists.sourceforge.net

UNICORE at FZJ

- User support email: ds-support@fz-juelich.de
- Registry: https://fzj-unic.fz-juelich.de:9112/FZJ/rest/registries/default_registry
- Documentation: <https://www.fz-juelich.de/en/ias/jsc/services/user-support/jsc-software-tools/unicore>



Part II: JSC Cloud

Overview

- OpenStack Infrastructure-as-a-Service (IaaS) environment
 - Compute, storage, network, orchestration, load balancing
 - Run VMs to provide services **linked to LARGEDATA**
 - Orchestration using OpenStack Heat
 - Load Balancer as a Service (LBaaS) using OpenStack Octavia
- Intended and existing workloads
 - Scientific services connected to HPC
 - Jupyter JSC
- Further information and reference: <https://apps.fz-juelich.de/jsc/hps/jsccloud/>

JSC Cloud	
vCPU	9312
Memory	30.5 TB
GPU	12x NVIDIA A100 80GB 16x NVIDIA V100 16GB 4x AMD MI210
NVMe	32x 1TB 6x 3.84 TB

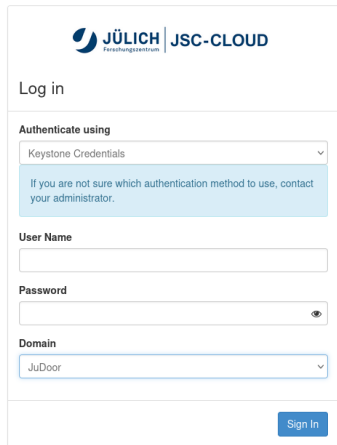


- OpenStack Zed release
 - released 2022-10-05
 - we try to balance stable operation and tracing current versions
- Services
 - Keystone – authentication and service registry
 - Horizon dashboard – convenient Web UI appropriate for many simple tasks
 - Nova compute – virtual machine (VM) service
 - Neutron networking – software defined networks
 - Cinder volume – virtual block devices
 - Glance images – template images for VMs
 - Heat orchestration – infrastructure management
 - Octavia load balancing – load balancing as a service
 - Neutron VPNaaS – cross-site (or project) VPNs
 - Sahara – data processing through virtual clusters

Authentication

There are two ways to authenticate

- JSC account
 - username and password
 - usable from both commandline interface and Web UI
- Helmholtz login
 - directly usable only from Web UI
 - commandline access through application credentials
- **However:** you need a project and allocated resources before using JSC Cloud



The screenshot shows the login page for Jülich JSC-Cloud. At the top, there is the Jülich Forschungszentrum logo and the text 'JSC-CLOUD'. Below this is the heading 'Log in'. A dropdown menu labeled 'Authenticate using' is set to 'Keystone Credentials'. A light blue informational box contains the text: 'If you are not sure which authentication method to use, contact your administrator.' Below this are three input fields: 'User Name', 'Password' (with an eye icon for visibility), and 'Domain' (set to 'JuDoor'). A blue 'Sign In' button is located at the bottom right of the form.

Nova

Virtual machine service

Nova manages the lifecycle of virtual machines (VMs) that have

- a number of CPUs
- an amount of main memory
- storage: **system**, ephemeral, swap
- data storage: volumes
- network ports
- a template image containing an operating system

Nova

Virtual machine service

Nova manages the lifecycle of virtual machines (VMs) that have

- a number of CPUs
- an amount of main memory
- storage: **system**, ephemeral, swap
- data storage: volumes
- network ports
- a template image containing an operating system

} Nova

Nova

Virtual machine service

Nova manages the lifecycle of virtual machines (VMs) that have

- a number of CPUs
- an amount of main memory
- storage: **system**, ephemeral, swap
- data storage: volumes
- network ports
- a template image containing an operating system



Nova

Virtual machine service

Nova manages the lifecycle of virtual machines (VMs) that have

- a number of CPUs
- an amount of main memory
- storage: **system**, ephemeral, swap
- data storage: volumes
- network ports
- a template image containing an operating system

} Nova

← Cinder

← Neutron

Nova

Virtual machine service

Nova manages the lifecycle of virtual machines (VMs) that have

- a number of CPUs
- an amount of main memory
- storage: **system**, ephemeral, swap
- data storage: volumes
- network ports
- a template image containing an operating system

} Nova

← Cinder

← Neutron

← Glance

Nova

Flavors

We are using as subset of the Sovereign Cloud Stack (SCS) flavors of the form

Example

SCS-16L:32:20n or SCS-nL:m:l-features

with

- n → number of CPUs (L for low performance or high oversubscription)
- m → amount of main memory (RAM) in GB
- l → size of root disk in GB (n for network shared storage)

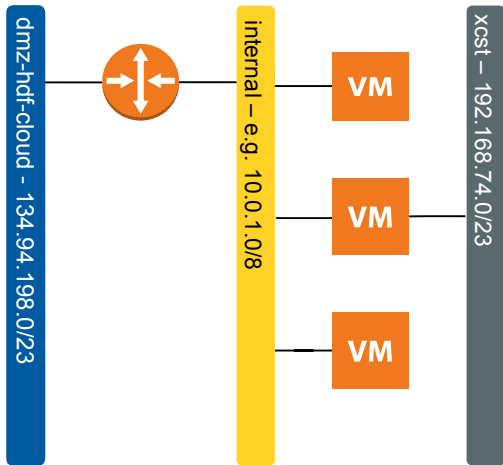
Use `openstack flavor list` or the Web UI to see available flavors. Some examples:

- SCS-2L:4:20n
- SCS-16L:64:20n-z2-nvme
- SCS-1L:1:20n

Networking

Specific networks at JSC

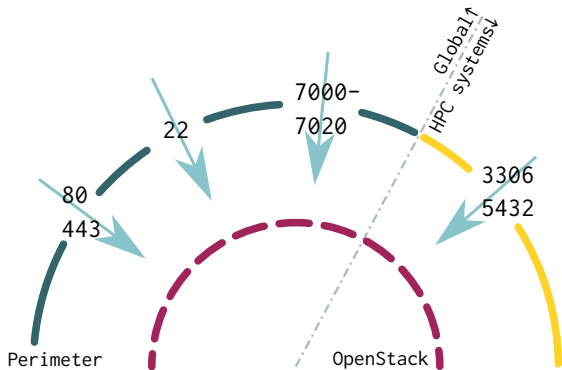
- **floating IPs** realized in router as DNAT/SNAT
- VMs without floating IPs not accessible from the outside and SNATed in outbound connections
- all new projects will be equipped with a **router** and **internal network**, such that you can immediately start working. JSC's **DNS servers** will be configured in the internal network
- **xcst** is a special purpose network for accessing DATA HPC file system



Network setup

Security groups and perimeter firewall

- OpenStack firewall freely configurable
- Restrictions apply for inbound connections in perimeter firewall
 - Globally available services and ports: HTTP (80), HTTPS (443), SSH (22), 7000–7020
 - Available from HPC systems: MySQL (3306), PostgreSQL (5432)
- Outbound connections: anything but MTA (25) aka. SMTP



Commandline interface

Prerequisites

- Python virtual environment
- Download credential files from the web interface (cf. authentication)

Run the following in your shell:

```
$ python3 -m venv openstack
$ source openstack/bin/activate
$ pip install python-openstackclient
```

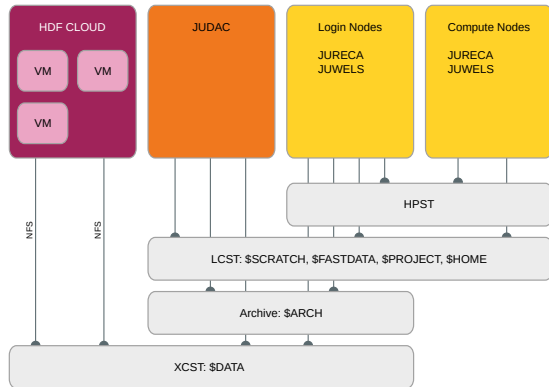
Authentication:

- Option 1: Download and `source openrc.sh`
- Option 2: Download `clouds.yaml`, put it in one of
 - current working directory as `clouds.yaml` or
 - `~/.config/openstack/clouds.yaml`

JSC Storage Landscape

Availability of file systems

- XCST
 - \$DATA on JUDAC and login nodes
 - dedicated NFS export to VMs
- Archive
 - \$ARCH on JUDAC and login nodes
- LCST
 - \$SCRATCH, \$FASTDATA, \$PROJECT, \$HOME on JUDAC, login and compute nodes
- HPST
 - Login and compute nodes



Data access

VMs and the DATA file system

- JSC Cloud / OpenStack cluster
 - Hosts virtual machines (VMs) for communities
 - Potentially administered by externals, bound by acceptable use policy
- Enable access to data beyond perimeter of SC facility
 - Web interfaces, databases, post processing, . . .
 - Users of service likely unknown to SC directory information service
- Access Method
 - POSIX file systems (\$DATA) accessible in VMs via NFS mount from CES servers
 - Server side UID squashing
 - ensures consistency
 - requires services to manage data accordingly
 - read-write or read-only



Summary

JSC Cloud

OpenStack

- Project web site: <https://www.openstack.org/>
- Documentation: <https://docs.openstack.org/>

JSC Cloud:

- User support: sc@fz-juelich.de
- Web dashboard: <https://cloud.jsc.fz-juelich.de/>
- Documentation: <https://go.fzj.de/jsc-cloud>