On-Grid GPU development

via interactive HTCondor jobs and Analysis Facility style workflows

Albert (Bruno) Borbely, David Britton, Emanuele Simili, Gordon Stewart, Samuel Skipsey

Introduction

Project Goals

- Streamlining the user development environment and reduce the overhead required to submit Grid jobs:
  - By bringing development onto the Grid, and allow users to install packages on the fly
  - Containerising the development environment
  - Allow users to test their software on the Grid hardware
  - Convert the development environment (DE) to the submission environment (SE) in a seamless way
  - And allow job submission to the wider Grid

Key Challenges

- User Authentication
- User Connection
- Data management
- Interactive development
- Development and Submission environments
- Integration of interactive and batch queues
- Submission framework to wider Grid queues

Aim: to find solutions with the pre-existing tools

On-Grid Interactive GPU development

Many problems to solve

Problems vaguely fall into 2 categories:

GPU problems

- Variety of cards (what to target)
- Variety of software tools
- Large dependancy issues
- Because of their nature user jobs tend to be small enough to not need to scale out to the Grid
- The idea is to facilitate on-Grid development and reduce the overhead in submitting Grid jobs via a submission engine

Interactive job develop problems (Analysis Facilities?)

- User authentication
- Flexible development environment
  - —> allow users to install packages
  - —> maintain site security
- Data storage / integration
- Scalability, i.e. easily scale out to the rest of the Grid

User Authentication

Who is Who?

- Ideally want to use something that is already in place?
  - Grid certificates were re-used for this purpose
  - Can be migrated to tokens with rest of site
  - Identity verification is already in-place
  - Already capable of handling multiple VOs
- CERN SSO and IRIS IAM were also considered:
  - CERN SSO not available for non-CERN VOs i.e. LIGO
  - Didn’t want to commit to a web-based login —> minimise required development

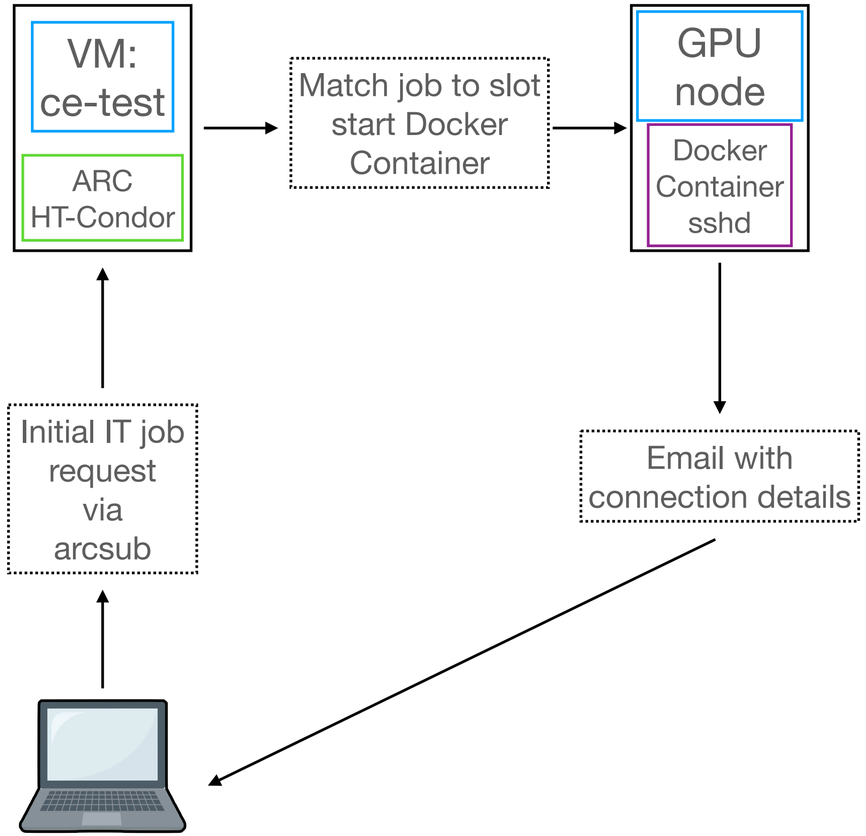
User Connection

via ARC, email and SSH

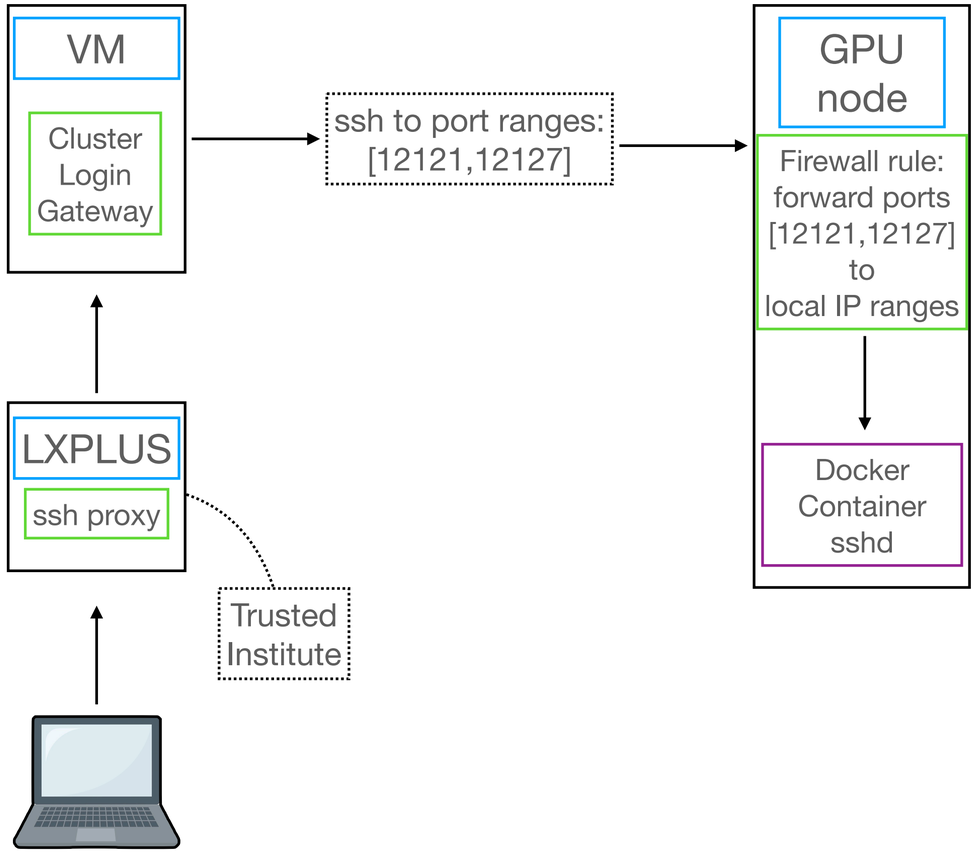
- Initial interactive session requested via arc job submission and providing a public ssh key:
  - Requesting a new session, list sessions or attach a previous session
  - The user then gets emailed a connection script which manages the connection via ssh
  - To minimise user error, a separate ssh config is created in .BanaF in the user’s home directory —> to mitigate any interaction with a user's pre-existing ssh config
  - For site security reasons only connections from trusted institute IP addresses are allowed with initially: Glasgow, CERN, Nikhev, DESY
  - LXPLUS (CERN) is the default ssh proxy, can be changed by the user
  - Then users connect to a VM which manages the users ssh keys and forwards the connection to the container

User Connection

1 - Interactive (IT) ARC request



2 - SSH connection



User Connection

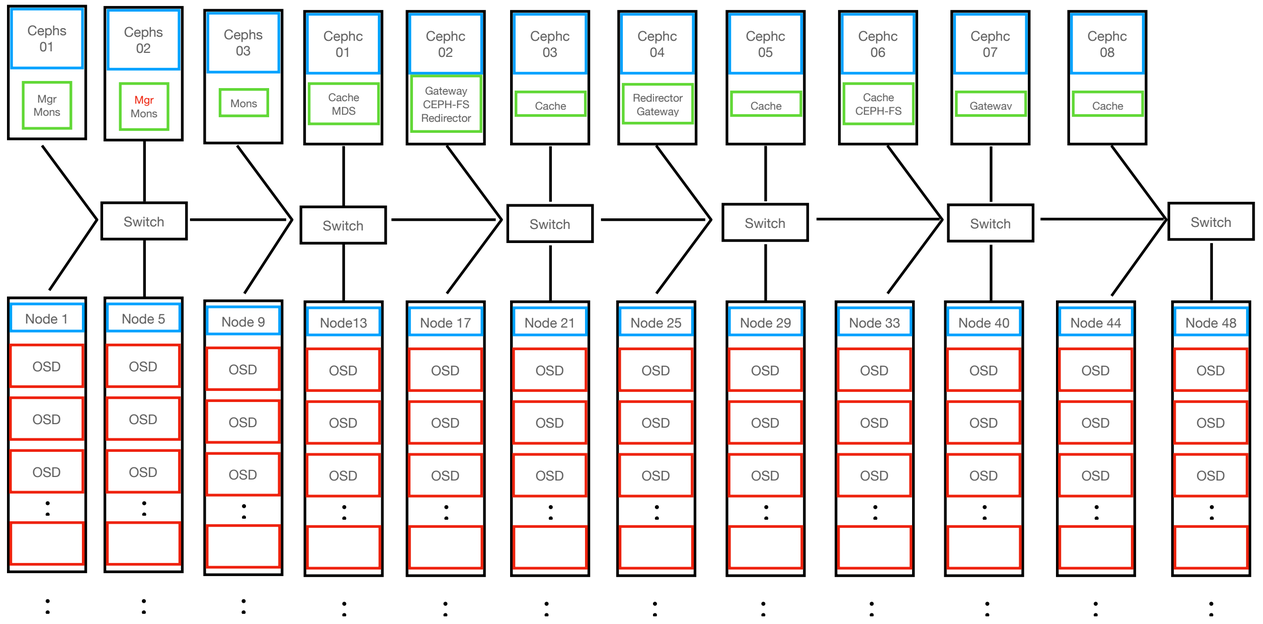
Removing User error

- Initially users were emailed ssh instructions that need to get implemented manually
- This proved to be clunky and would often created problems with users pre-existing ssh configs
- Since the connection is required to tunnel through a trusted institute, it also added issues of site system configuration, where I am not a sys-admin
  - Most testing was done with LXPLUS (CERN) as I have access
- Users are now only required to supply public ssh-key, users email is parsed from x509 proxy
  - This solves the problem of key distribution
- Now users are emailed a tar file with a setup script
  - The setup script moves all the files into ~/.BanaF directory where alternate ssh config doesn't interfere with the user's pre-existing setup
  - The script takes LXPLUS username as input, and is used as the default trusted institute
  - Tools like VS-Code can use the alternate config file to seamlessly connect to the session, edit files and browse ROOT files etc.

Data Management

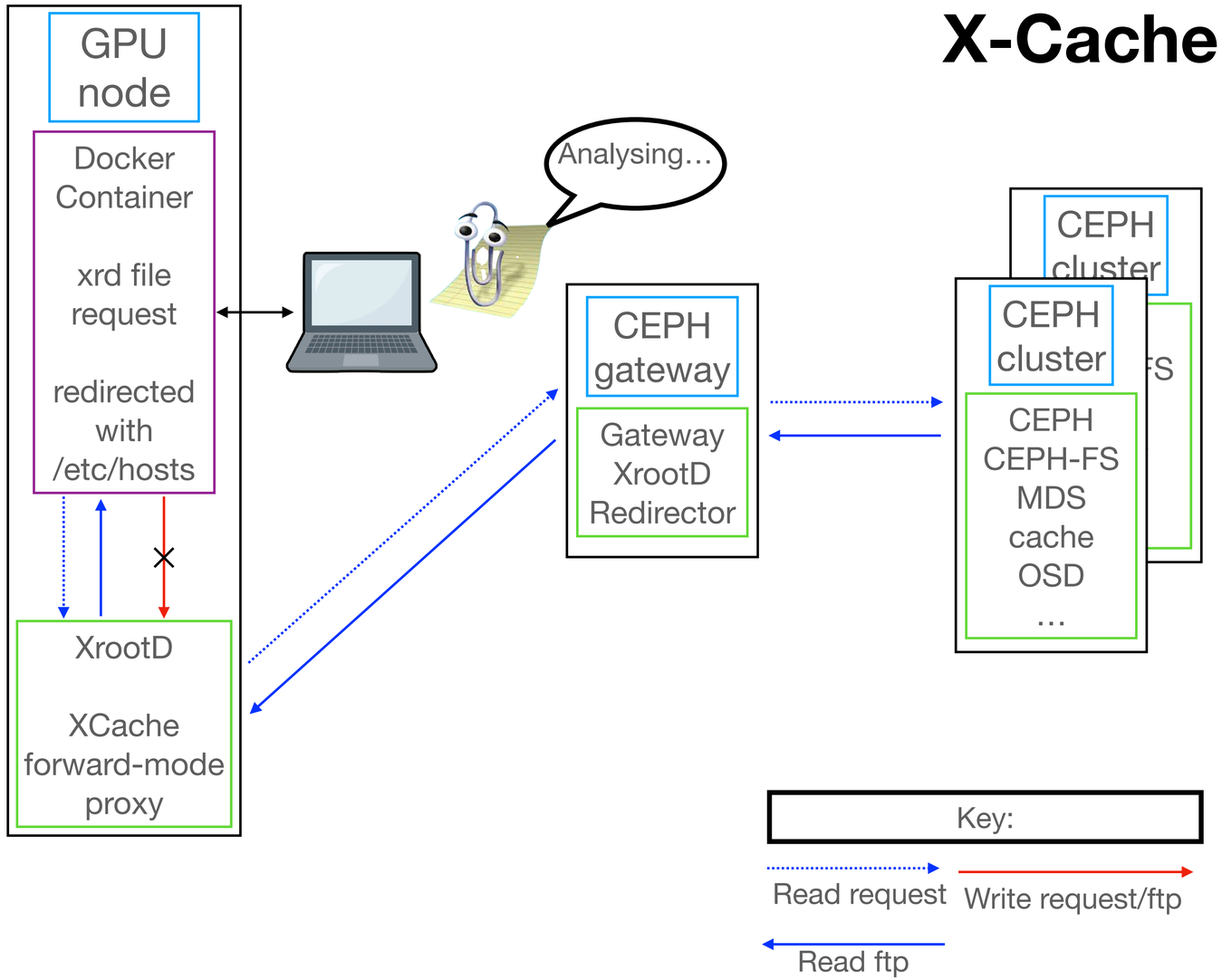
CEPH Tier-2

- Ceph-FS partitions can directly be mounted with XrootD into a directory in the container
- This works for any type of file i.e. hdf5 etc. not just root files.
- Allows users to natively access files in a POSIX file system directly on the Ceph cluster, allowing to easily read/write. Care should be taken when doing I/O heavy operations.
- Draw backs: heavily relies on XrootD and Ceph-FS. Mounted instances have crashed before causing bizarre behaviour.
- In the past the XrootD was running on the host system, this has now been moved in to the container.
- Legacy VOs (ATLAS) will remain as object stores.



Data Management

X-Cache



- Using hosts file in container to redirect traffic to proxy
- Proxy can only handle read-requests
- Write requests need to be directed to alternate gateway manually
- Only intended to cache object store
- CEPH-FS portions will be mounted via alternate gateway to avoid duplicate file write issues
- Rely on users to manage these two options
- Will also test libXrdClProxyPlugin.so

Heard rumours that a new feature will drop with arc 7

Container Environments

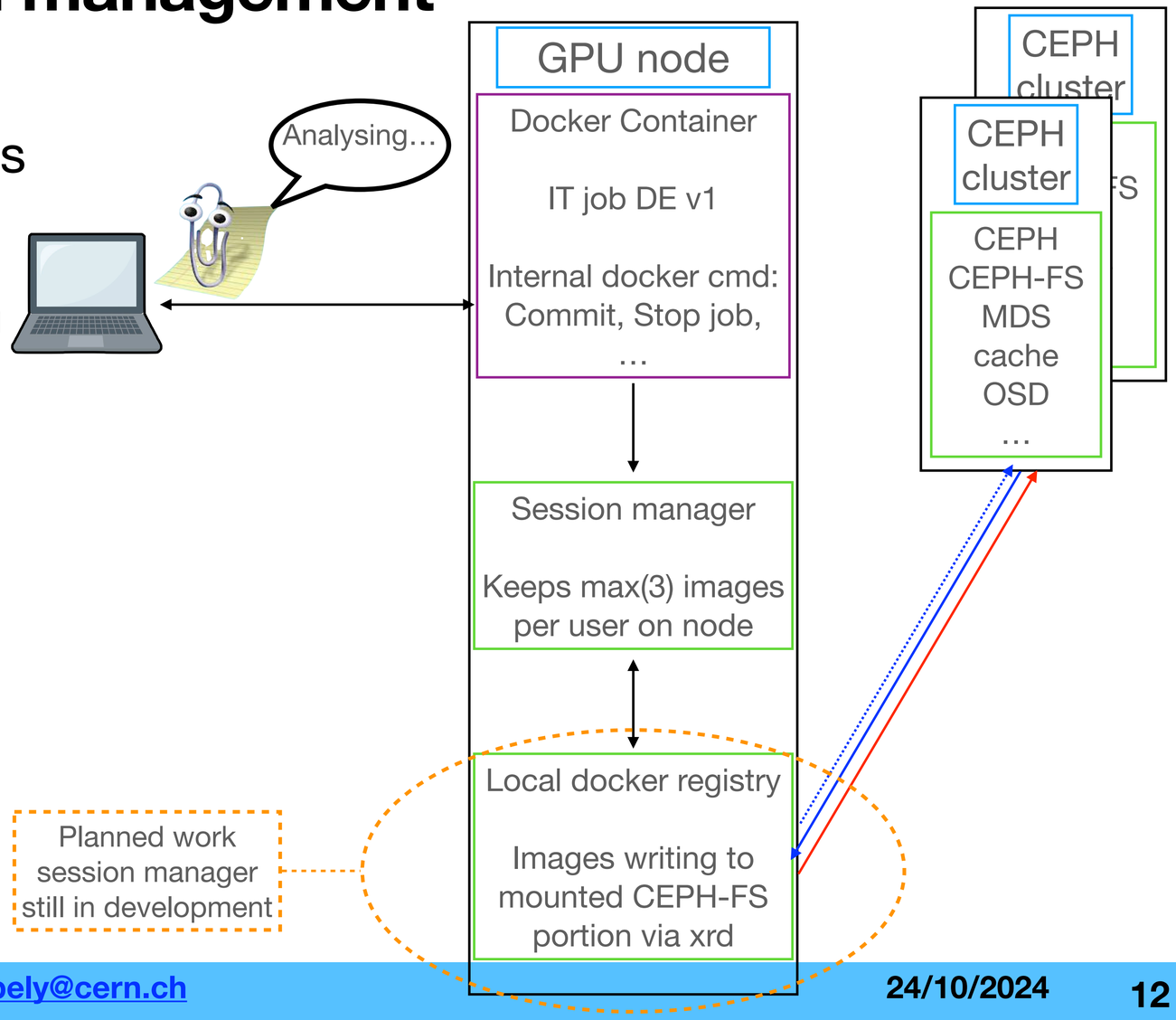
For Development and Submission

- Docker vs Apptainer (Singularity) were tested for this
- Docker wins, especially for interactive GPU development use
- Apptainer essentially doesn't allow for interactive containers and GPU interaction simultaneously
  - No matter what SUID variations you use
- Docker —> Apptainer conversion is straightforward
- Once development is over and a user wants to submit their job to the Grid it can be converted to Apptainer (the standard Grid tool)
- This allows for environment transportability

Container Environments

Job / session management

- Users need a way to manage their sessions
- Users can talk with the session manager via job submission and get responses with email i.e. list sessions
- Need to run some docker cmd from within the container e.g. docker commit, docker stop, docker run etc.
- Currently implemented with cron jobs
- Or from within an interactive session via cron jobs i.e. commit and stop



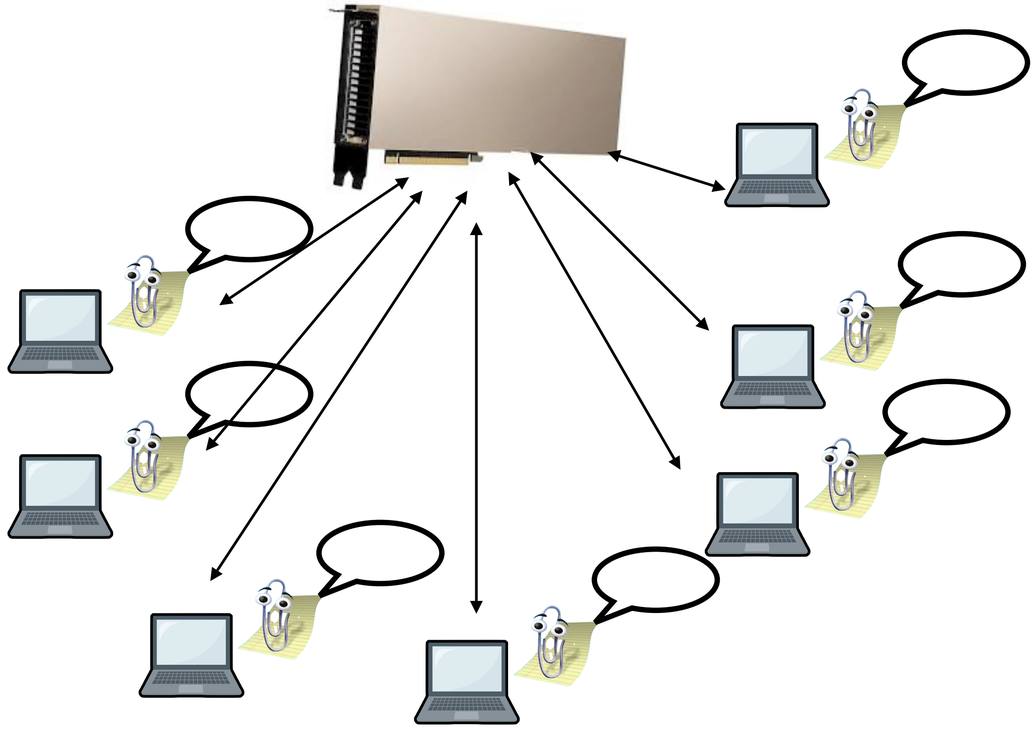
Interactive Development

Packages and Initial environment

- Initial environment is a docker file hosted on github
- This gets pulled and built when an interactive job is requested
- Users / groups can submit alternate docker files, which will require manual review before being merged
- Users ssh into the container as a test user, never have privileged access
- Packages can be installed with spack, pip and conda into home directory
- I was able to install and run Celeritas for benchmarking this way, was able to install all the packages etc. and compile the code
- The job should be made to be launched from an entrypoint.sh script

Merging Interactive and Batch queues
Via HT-Condor

- Currently node has 2 GPUs, plan to use one interactively and one in batch mode
- Slice one GPU with MIG into 7, with 6x10 and 1x20 GB of RAM
- Allows for 7 concurrent users
- Currently condor (10) assigns all GPUs to all slots
  - Plan to match correct gpu to correct slot with manual script
  - Hopefully this changes with newer versions of condor



Submission framework

to the wider Grid

- Docker wins for interactive development, but is not a standard Grid tool
- Apptainer is standard at Grid sites, supports conversion from docker images
- This feature is still in development, will plan to test with Manchesters GPU queue
- Will strip out the sshd and look at ways of slimming down the image size
- Plan to simply submit with arc, can hopefully be integrated into existing VOs submission engines
- Where to store the arc jobs .xml file?
  - In the interactive jobs container? With the session manager? Email it to the user?

Development problems

Developing on an active Tier-2

- Testing problems sometimes our test-ce gets found by a VO….
  - … and flooded with test jobs, blocking interactive job requests
- The CEPH cluster is actively used by site jobs
  - … can highlight I/O bottlenecks for interactive use
  - … hopefully these can be mitigated with Xcache, currently only providing read access
  - … currently fast write speed is only supported on-node, two mirrored SSDs, then manually replicated to CEPH cluster, managed by the user

Summary

Key Takeaways

- This was intended as a proof of concept
- Some solutions may be a bit hackey
- The ability for users to install packages on the fly is very useful
  - Solved with Spack + pip
  - No need for root permissions
- Currently testing with a handful of users
- Need a better way to interact with the session manager
  - Ideally with a web interface
  - Currently request have to be submitted via jobs…
  - These can get stuck in the queue and reduce usability
  - Will also need some queue management