Basic/default Connection

Connecting to:

Cloud10 / ARC3/4:

1) ssh -Y username@remote-access.leeds.ac.uk
   Password:
   Duo two-factor login for username@leeds.ac.uk
   Passcode or option (1-2): 1
   Success. Logging you in...
2) ssh -Y foe-linux-04
   username@foe-linux-04's password:
3) ssh -Y cloud10 / ssh -Y arc4.leeds.ac.uk

Running ParaView on cloud10:

1) paraview & (no MPI and no job submission, although can also use pvserver if desired)

Running ParaView on ARC4:

0) module add mesa paraview

Parallel:

1) qrsh -cwd -V -l h_rt=00:20:00 -l nodes=1 pvserver --server-port=11111 &
   [1] 191362
   Quite often ARC4 is busy, and will print this message:
   [username@login1.arc4 ~]$ Your "qrsh" request could not be scheduled, try again later.
   At this point, waiting for when it becomes less busy is one option, but can also try the serial approach below (or Advanced Connection).

Serial:

1) qrsh -cwd -V -l h_rt=00:20:00 -l h_vmem=10G pvserver --server-port=11111
   Waiting for client...
   Connection URL: cs://1234567.arc4.leeds.ac.uk:11111
   Accepting connection(s): 1234567.arc4.leeds.ac.uk:11111

Next, MUST set-up ssh-tunnel!

Open new terminal:

1) ssh -L 11111:1234567:11111 login1.arc4.leeds.ac.uk

NOTE: 1234567 is a connection ID given automatically, see Connection URL above (just copy that)

Open ParaView, connect as usual, details here:
https://arcdocs.leeds.ac.uk/software/applications/paraview.html

Connecting to and running ParaView server on:

ARCHER2:

1) ssh username@login.archer2.ac.uk
2) Apply for SHORTQOS (maximum 20 minutes 00:20:00) free
   srun --nodes=1 --exclusive --time=00:20:00 --partition=standard --qos=short --reservation=shortqos --pty /bin/bash
   or if needed for a longer interactive session:
   srun --nodes=1 --exclusive --time=00:50:00 --partition=standard --qos=standard --pty /bin/bash
   run ParaView interactively:
   username@nid004526:/tmp>
   ```
   cd /work/e710/e710/username/
   module load paraview/5.10.1
   srun --oversubscribe -n 32 pvserver --mpi --force-offscreen-rendering
   ```
   Waiting for client...
   Connection URL: cs://nid001023:11111
   Accepting connection(s): nid001023:11111

Next, MUST set-up ssh-tunnel!

Open new terminal:

1) `ssh -L 11111:nid001023:11111 username@login.archer2.ac.uk`

Open ParaView, connect as usual (details here: https://docs.archer2.ac.uk/data-tools/paraview/)

Advanced Connection

ARC3/4:

For test modules, run:

```
module add test
```

```
then module avail -t paraview
```

Interactive job request:

1) qrsh -l h_rt=1:00:00,h_vmem=16G -pe smp 8
   [1] 191362
   [username@login1.arc4 ~]$ Your "qrsh" request could not be scheduled, try again later.
   Can also apply for a GPU node:
   {GPU NODE:
   qrsh -l h_rt=0:20:00,coproc_k80=1
   }

Running ParaView on ARC4 (WITH MPI)

2) module swap openmpi intelmpi
3) module add mesa/18.3.6 paraview-osmesa/5.3.0

MPI:

4) mpirun pvserver --mesa --server-port=11118 --use-offscreen-rendering
   Waiting for client...
   Connection URL: cs://d11s3b2.arc4.leeds.ac.uk:11111
   Accepting connection(s): d11s3b2.arc4.leeds.ac.uk:11111

Next, MUST set-up ssh-tunnel!

Open new terminal:

5) ssh -L 11111:d10s0b1:11111 login1.arc4.leeds.ac.uk

ParaView with TTK and VTK-m on Archer2

TTK and VTK-m on ParaView is available on the ParaView Superbuilt. Member from the ARCHER2 Service Desk Team has made a Singularity container containing this ParaView Superbuild. This singularity container is now located at /work/e710/shared/paraview-5.10.1-ttk-1.1.0-mpich-4.0.2.sif. To test it, you will need a local installation of ParaView 5.10.1 to act as the client.

I've run tests on the ARCHER2 data analysis nodes. To try a similar test, you could start an interactive job by running:

srun --time=00:20:00 --partition=serial --qos=serial --hint=nomultithread --account=e710 --ntasks=4 --mem=16G --pty /bin/bash

This gives you 4 cores and 16 GB of memory for 20 minutes (use of these nodes is uncharged).

Once the job has started, you should take note of the node name -- when running on the data analysis nodes, this will be either dvn01 or dvn02. You should then run:

```
export OMP_NUM_THREADS=1
export SINGULARITYENV_LD_LIBRARY_PATH="/opt/cray/pe/mpich/8.1.4/ofi/gnu/9.1/lib-abi-mpich:/opt/cray/pe/pmi/6.0.10/lib:/opt/cray/libfabric/1.11.0.4.71/lib64:/usr/lib64/host:/usr/lib/x86_64-linux-gnu/libibverbs:/.singularity.d/libs:/opt/cray/pe/gcc-libs:\$LD_LIBRARY_PATH"
export SINGULARITY_BIND="/opt/cray,/usr/lib64/libibverbs.so.1,/usr/lib64/librdmacm.so.1,/usr/lib64/libnl-3.so.200,/usr/lib64/libnl-route-3.so.200,/usr/lib64/libpals.so.0,/var/spool/slurmd/mpi_cray_shasta,/usr/lib64/libibverbs/libmlx5-rdmav25.so,/etc/libibverbs.d,/opt/gcc,/work/e710/e710/username"
```

These environment variables ensure no OpenMP, and bind ARCHER2's MPICH libraries into the container then modified LD_LIBRARY_PATH to use them. It also binds your work directory into the container so you'll be able to see it from within ParaView.

You should then be able to run ParaView with the following command:

srun --oversubscribe --ntasks=4 singularity exec paraview-5.10.1-ttk-1.1.0-mpich-4.0.2.sif pvserver --force-offscreen-rendering --mpi

If you don't run this command from the same directory as the .sif file, you'll need to provide the full path. You'll see some text come up similar to

```
Waiting for client...
Connection URL: cs://dvn01:11111
Accepting connection(s): dvn01:11111
```

At this point the server is running and waiting for you to connect. Open an SSH tunnel to the node running pvserver by opening a new terminal on your own machine and running:

ssh <your usual ssh options to log in> -L 11111:dvn01:11111

replacing 'dvn01' with whatever else the name of the node running pvserver is. The '-L' option is to open a tunnel connecting port 11111 on your machine to port 11111 on dvn01 on ARCHER2. Once that's connected, you can just leave that window open until you're finished with ParaView.

Now, start ParaView 5.10.1 on your own computer. To connect to ARCHER2, click on the little Connect icon (two servers sat next to each other with a green circle below) or go to File -> Connect. You can add a server, giving it the default options: Server type 'Client/Server', Host 'localhost', and Port '11111'. The name doesn't matter so much, but you can call it 'ARCHER2' if you like.

Once added, click on 'Connect'. This finally should connect the client running on your machine to the server running from the container on ARCHER2. The terminal running the job will show:

Client connected.

At that point, you should be able to open files within your work directories on ARCHER2. You may also need to go to Tools -> Manage Plugins to enable TTK locally. It *should* be loaded automatically on the server.

William
The ARCHER2 Service Desk Team
helpdesk@archer2.ac.uk

File Transfer

feng: vglrun /usr/not-backed-up/Paraview/bin/paraview

Copying:

ARC4

PC → ARC4:

rsync -Pv -e"ssh -c aes128-gcm@openssh.com -i path/to/ssh/rsa/file" ./file.py username@arc4.leeds.ac.uk:/destination/

ARC4 → PC

rsync -Pv -e"ssh -c aes128-gcm@openssh.com -i path/to/ssh/rsa/file" username@arc4.leeds.ac.uk:/source .

ARCHER2 → ARC4

Repeatedly poll file transfer:

[username@login1.arc4 ~]$ for i in {1..20}; do ls -lt; date ; sleep 600; done

(source: for i in {1..10}; do echo -n "This is a test in loop $i "; date ; sleep 5; done

1. http://paulsoftech.blogspot.com/2016/02/to-run-command-repeatedly-in-linux.html
2. https://www.tecmint.com/run-repeat-linux-command-every-x-seconds/

scp -3r username@login.archer2.ac.uk:/src username@arc4.leeds.ac.uk:/dst/

source:

1. https://serverfault.com/questions/449705/why-is-it-not-possible-to-use-two-remotes-for-rsync

ARCHER2

ARCHER2 → PC:

scp username@login.archer2.ac.uk:/src .

PC → ARCHER2:

rsync -Pv -e"ssh -c aes128-gcm@openssh.com -i path/to/ssh/rsa/file" ./animation_loader.py username@login.archer2.ac.uk:/work/e710/e710/username/

feng-linux

ARC4 → feng (run from feng terminal)

rsync -Pv -e"ssh -c aes128-gcm@openssh.com -i path/to/ssh/rsa/file" username@arc4.leeds.ac.uk:/src /dest

feng → ARC4 (run from feng terminal)

rsync -Pv -e"ssh -c aes128-gcm@openssh.com -i path/to/ssh/rsa/file" /src/ username@arc4.leeds.ac.uk:/dest

archer2 → feng-linux :

rsync -Pv --verbose -e"ssh -c aes128-gcm@openssh.com -i path/to/ssh/rsa/file" username@login.archer2.ac.uk:/src /dest/

How to use GPU on ARC4 from IT support:

Paraview on ARC4, using GPUs.

Assuming there was a free GPU:

Login to ARC4:

```
module add test paraview-egl
qrsh -V -cwd -l coproc_v100=1,h_rt=1:0:0 -pty y pvserver -p 11111
```

It should output something like:
Waiting for client...
Connection URL: cs://db04gpu3.arc4.leeds.ac.uk:11111
Accepting connection(s): db04gpu3.arc4.leeds.ac.uk:11111

Once that's running, do a separate connection on your local device to make the tunnel to match the host you're allocated. In my case this is:

```
ssh -NL 11111:db04gpu3:11111 arc4.leeds.ac.uk
```

Then run on your local device, paraview to use that tunnel to talk to the remote paraview:

```
paraview --url cs://localhost:11111
```

You know that's connected as you should get "Client connected." appearing on your ARC session.

If you look in Help/About Paraview/Connection Information, it shows that you're connected to a system that's using a Tesla V100 for the rendering with EGL, which all looks good.