Cornell University
Center for Advanced Computing

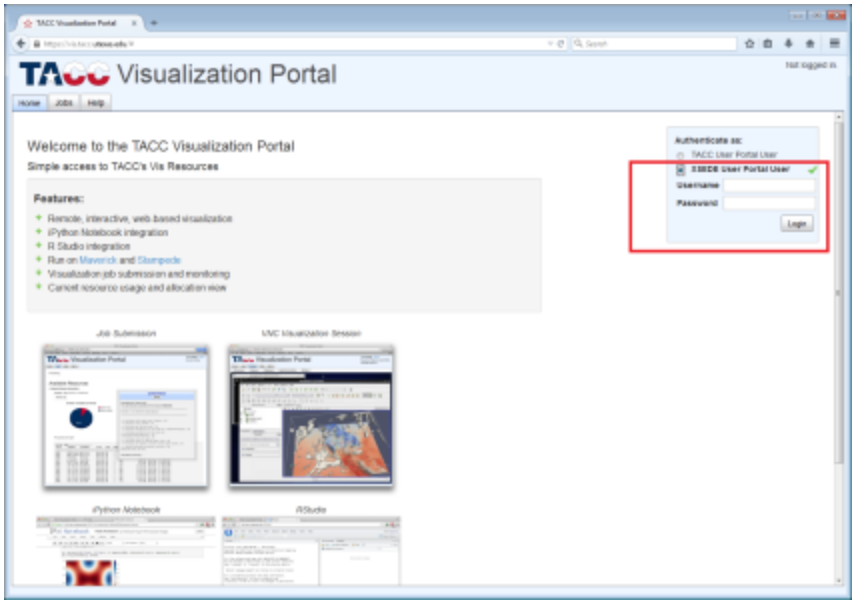
Visualization Labs

Aaron Birkland
Cornell CAC

High Performance Computing on Stampede
January 15, 2015

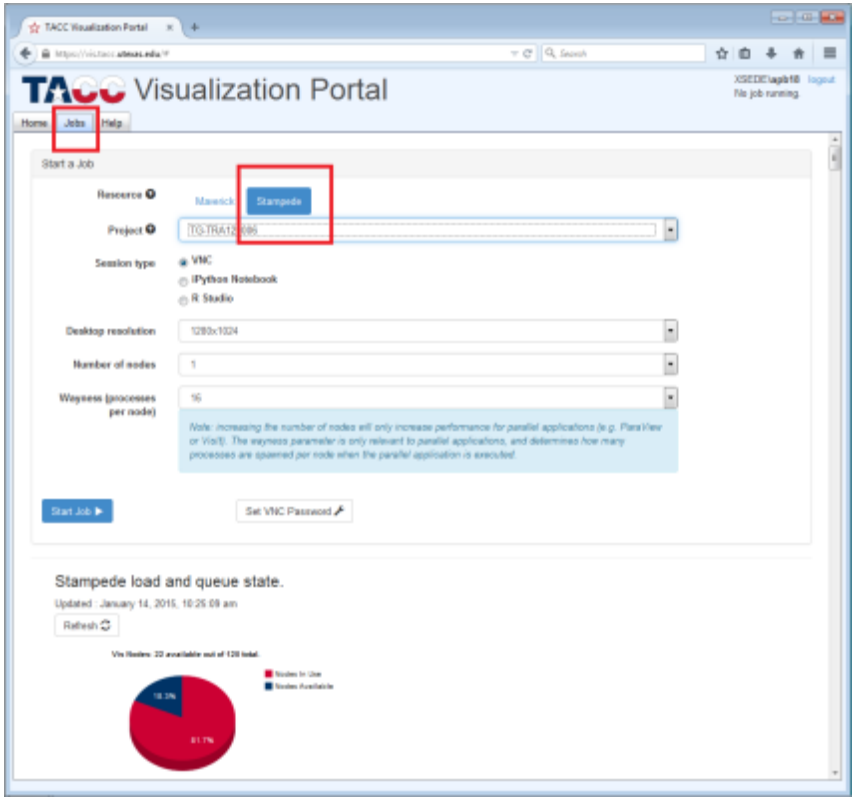
Visualization Portal: Getting Started: Login

- https://vis.tacc.utexas.edu
- Use XSEDE username/password, or local TACC credentials (if you have it)



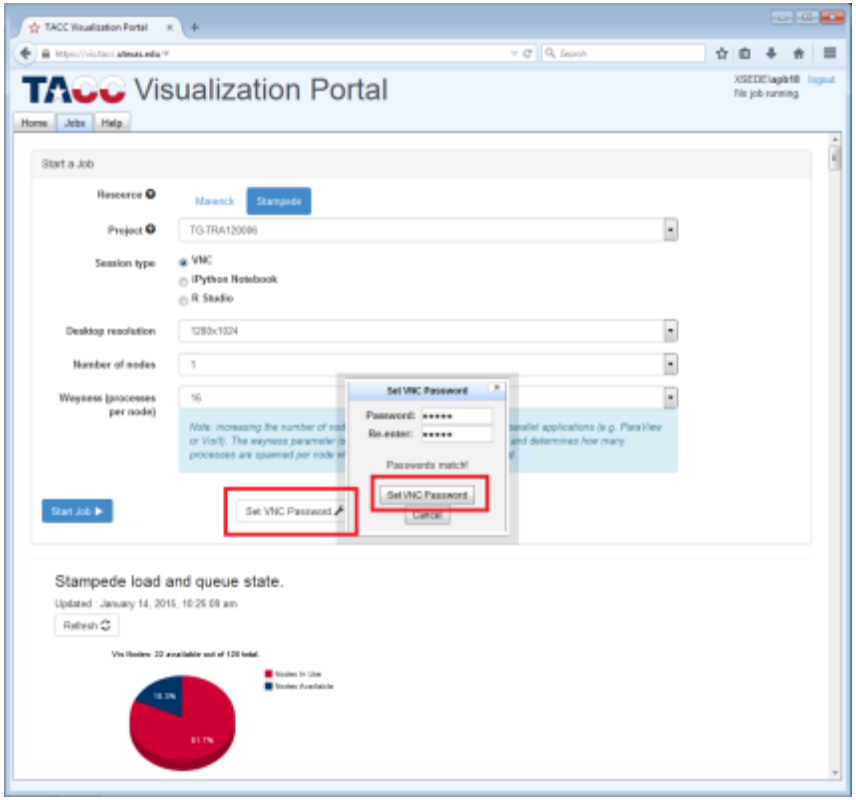
Visualization Portal: Getting Started: Jobs

- Click the ‘jobs’ tab, if you aren’t there already
- Click the ‘Stampede’ button in the ‘Resource’ section.
- Scroll down to the ‘Stampede load and queue state’ pie chart. Are there enough vis nodes to run your job?



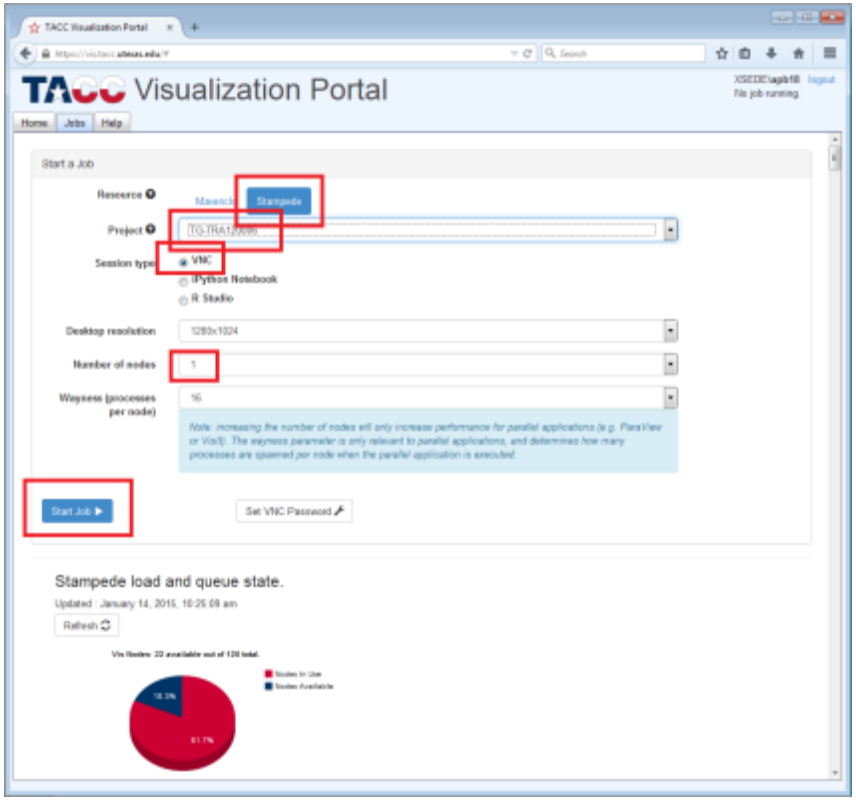
Visualization Portal: Getting Started: VNC passwd

- This needs to be done only once, ever
- DO NOT use your login password! This password protects the VNC graphical session only.
- Sometimes, you want to intentionally share VNC password with collaborators



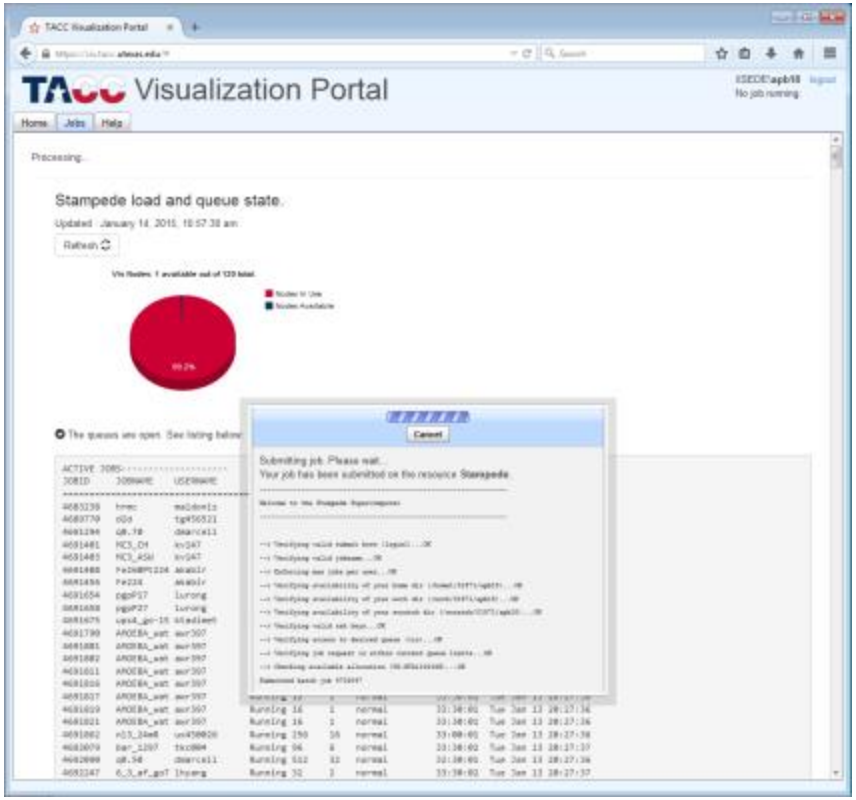
Visualization Portal: Getting Started: Jobs

- Pick job parameters. For this example, we're just opening a desktop on one node
  - Make sure 'Stampede' is selected, as well as the right account.
  - Session type is VNC, One node. Wayness irrelevant for this example
- Start Job!



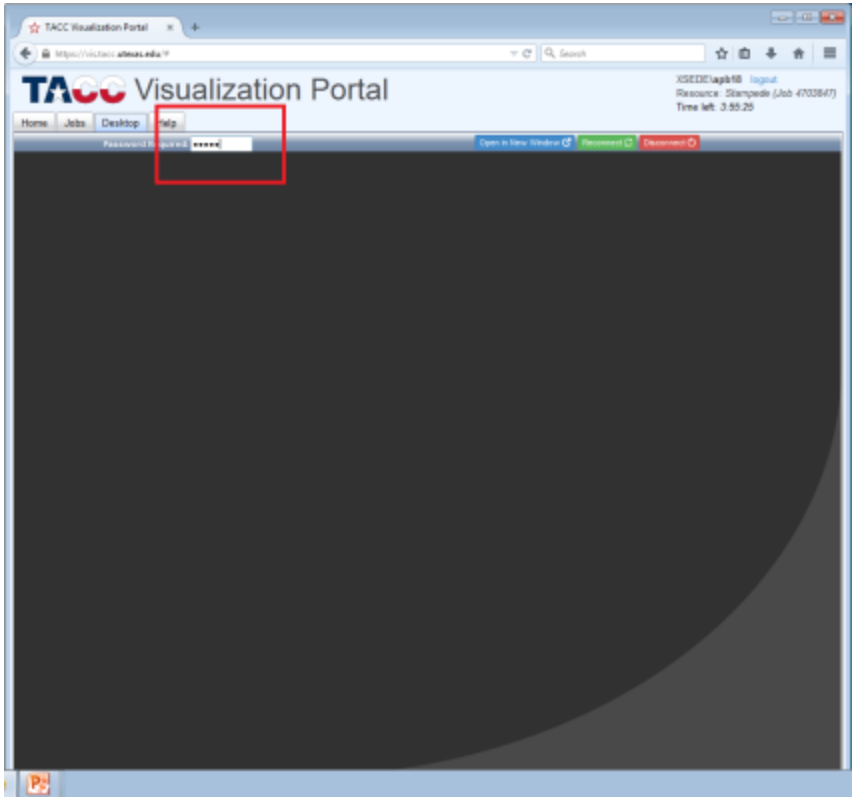
Visualization Portal: Getting Started: Jobs

- Various status screens will pop up until the job starts running
- You cancel the job submission via ‘cancel’ if you need to (i.e. you notice queue is full, don’t want to wait)



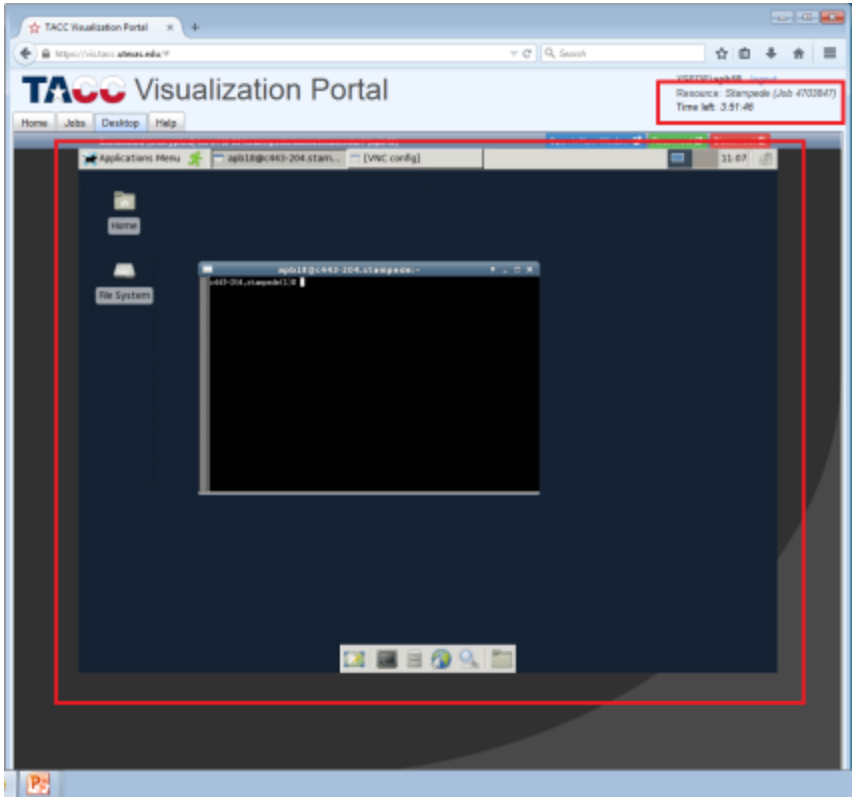
Visualization Portal: Getting Started: Jobs

- Once job is running, you will be directed to a new ‘desktop’ tab
- Type your VNC password in the box, then HIT ENTER KEY. Do not click on any of the tempting colored boxes.



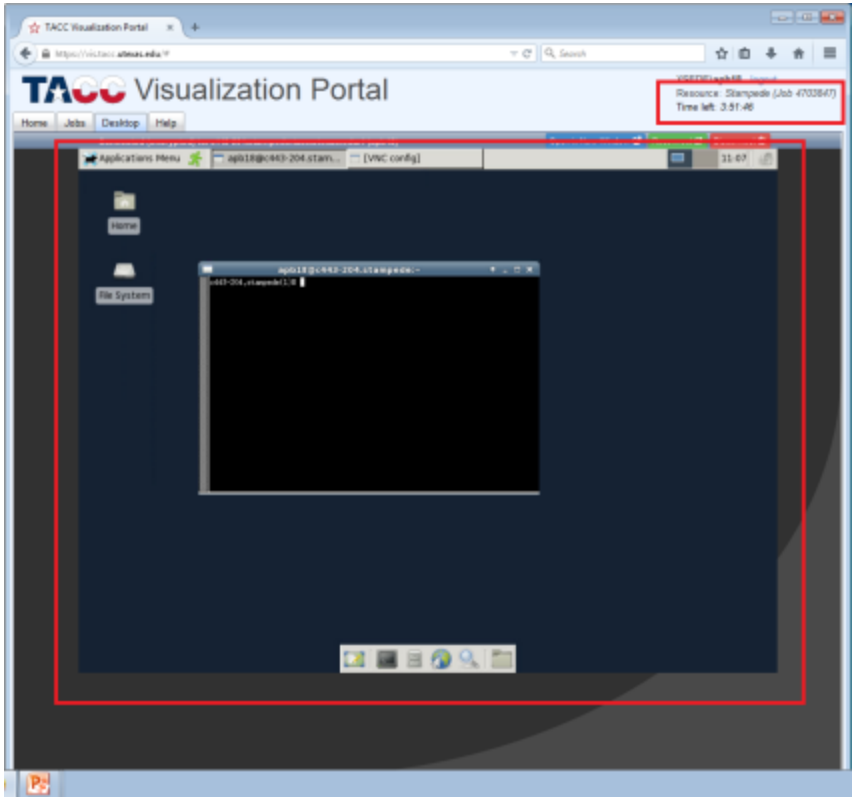
Visualization Portal: Getting Started: Jobs

- Your job is now running, and you have a desktop on one of the compute nodes!
- Move windows around and click on things. You can even use the 'Applications' menu in the upper left, or the 'dock' buttons at the bottom.
- Job status/timer in corner



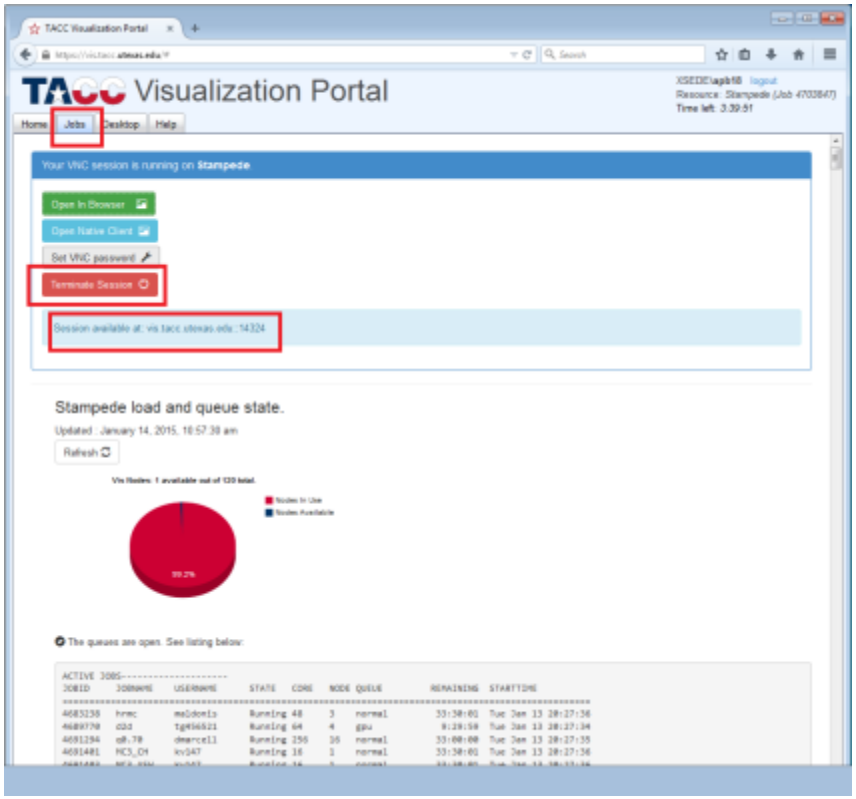
Visualization Portal: Getting Started: Jobs

- Do not type ‘exit’ in the default xterm, that will end your session!
- Use the ‘dock’ or application menu to launch a new terminal, or just type in ‘xterm’ to launch one manually.



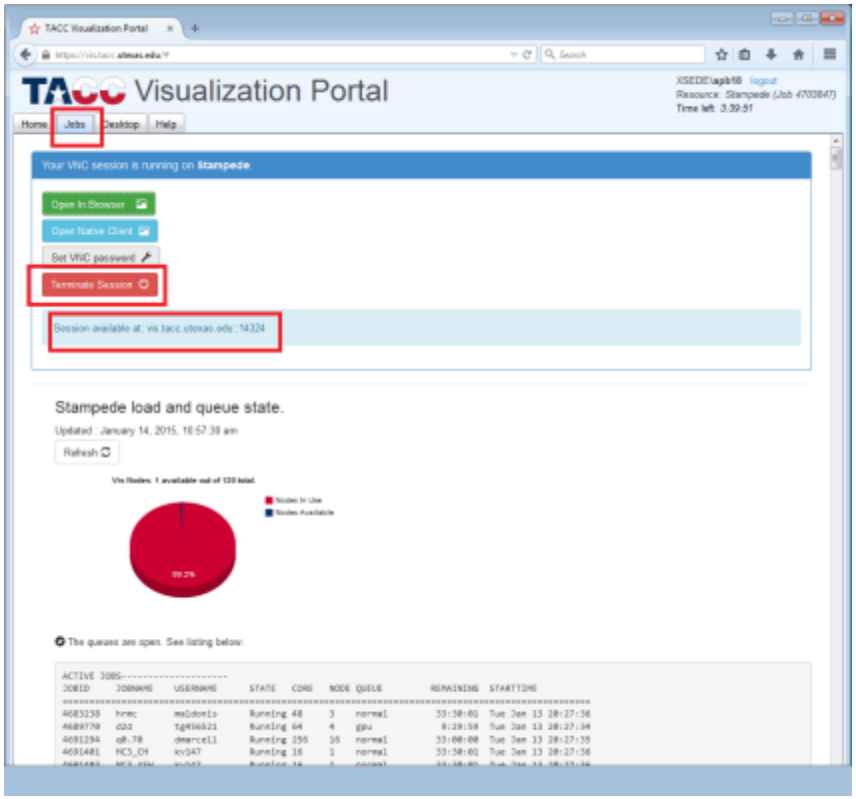
Visualization Portal: Getting Started: Jobs

- Click on ‘jobs’ tab to see job details, and perform various actions
- Look at vnc connection URL. This is used for connecting with a standalone VNC client
- When finish, terminate your session by clicking on the red button.



Visualization Portal: Sharing VNC

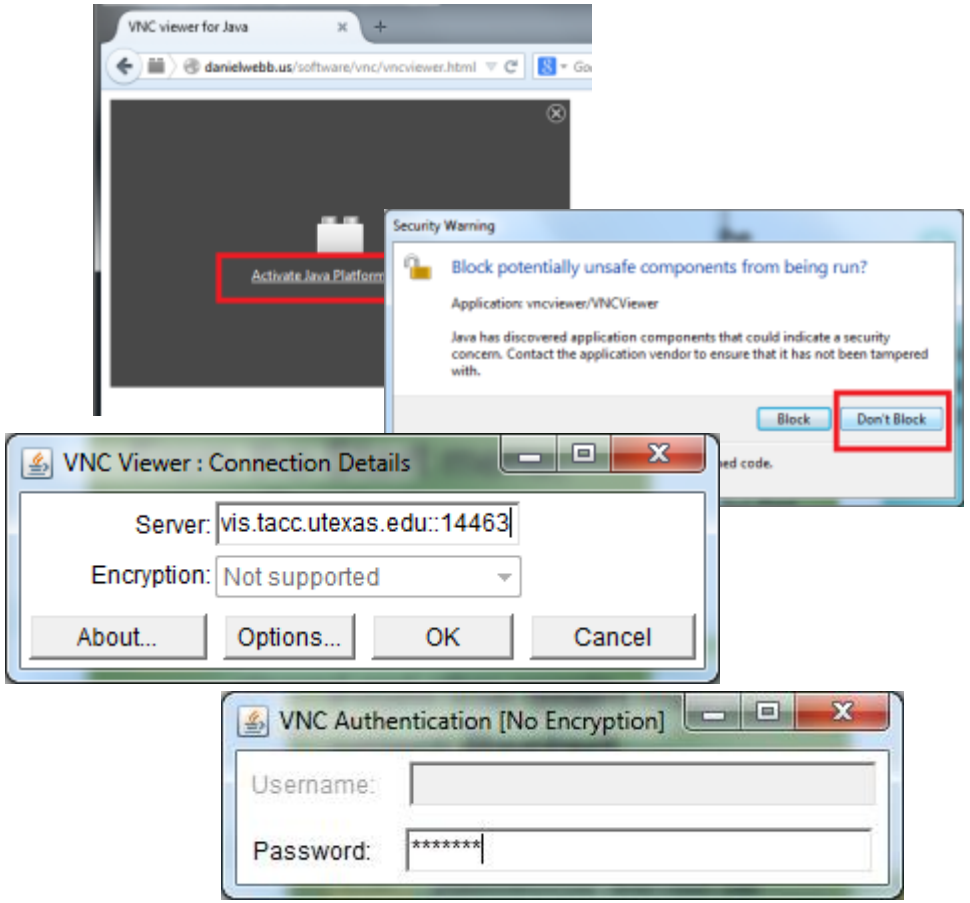
- VNC session can be shared by many people (fight for the mouse) by sharing the VNC connection URL and password
- Portal is only useful for connecting to your own sessions. Everybody else needs their own standalone client



Visualization Portal: Sharing VNC

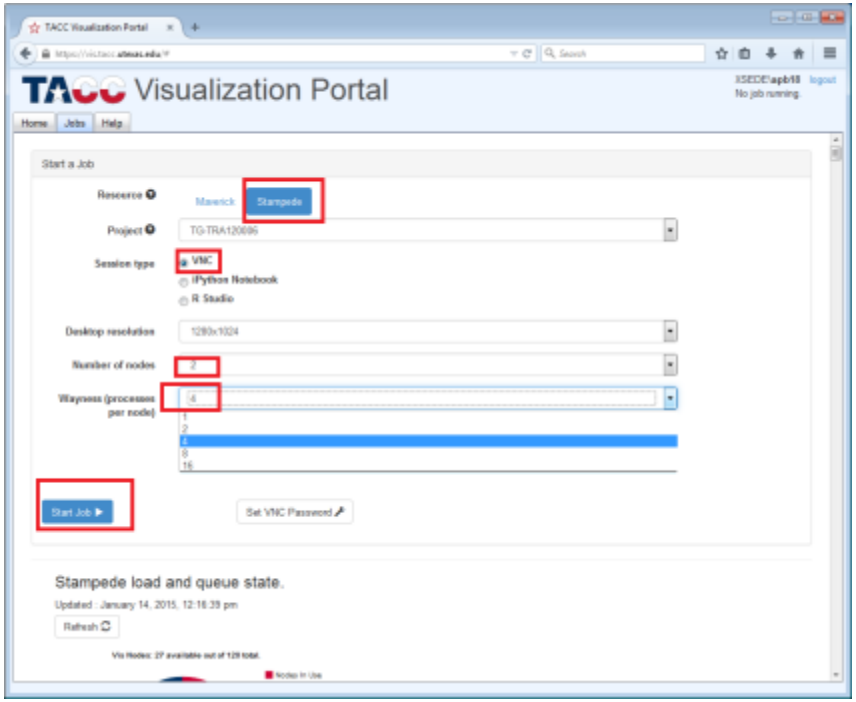
- Somebody has put up the java VNC client for people to use:
- http://danielwebb.us/software/vnc/vncviewer.html

- Click through all the scary messages about running java applets in order to run. Type in VNC URL exactly as it appears in the portal



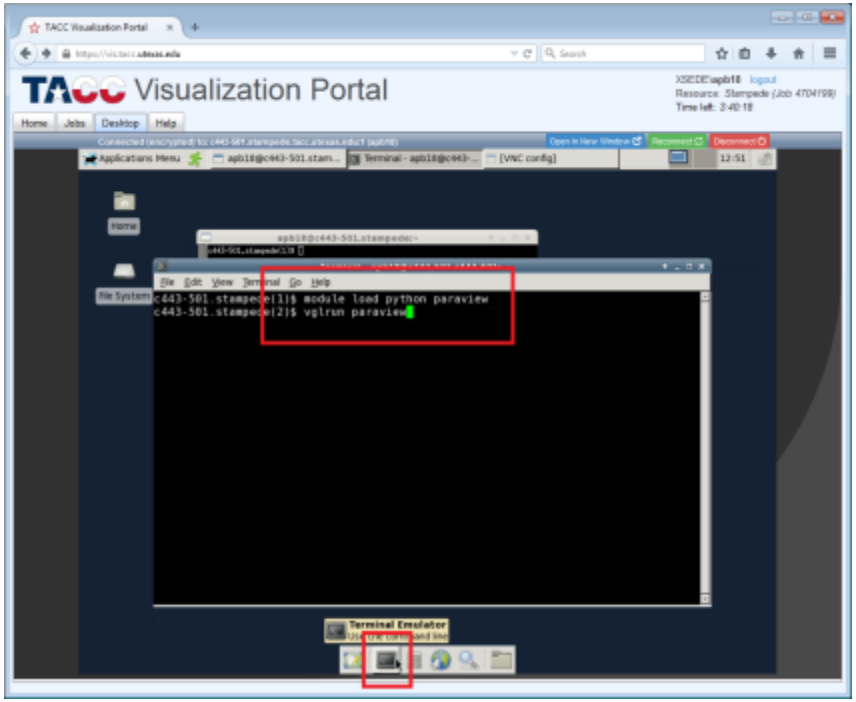
Visualization: ParaView in parallel

- Use the portal to launch a 2-node job, with 4 processes on each node.
  - Go back to the ‘Getting Started’ slides for step by step instructions if anything is unclear
- Once the job is running, connect to the VNC desktop.



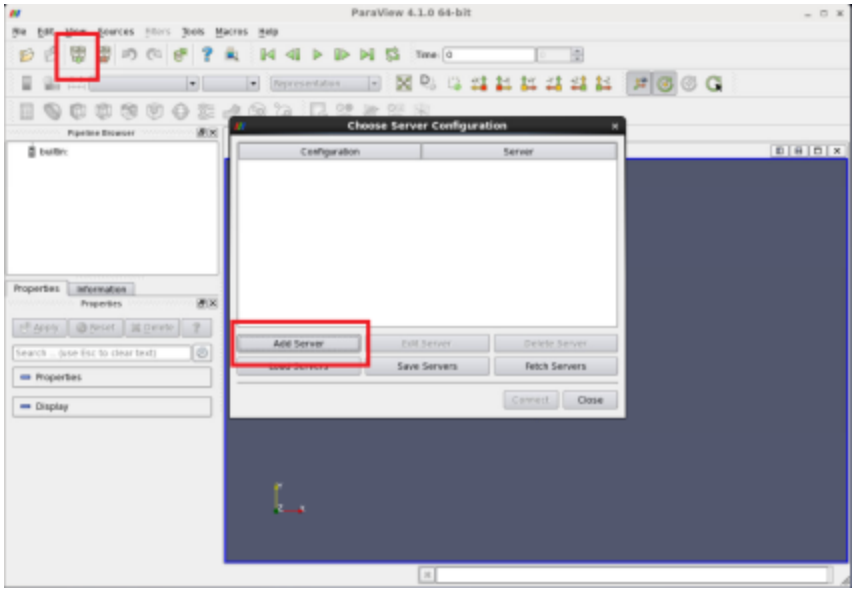
Visualization: ParaView in parallel

- Open up a new terminal, load the paraview module, and run
  - `module load python paraview`
  - `vglrun paraview`
- Notice that you still need to load any modules you need, and need to wrap 3D applications in `vglrun` in order for them to run on the VNC desktop



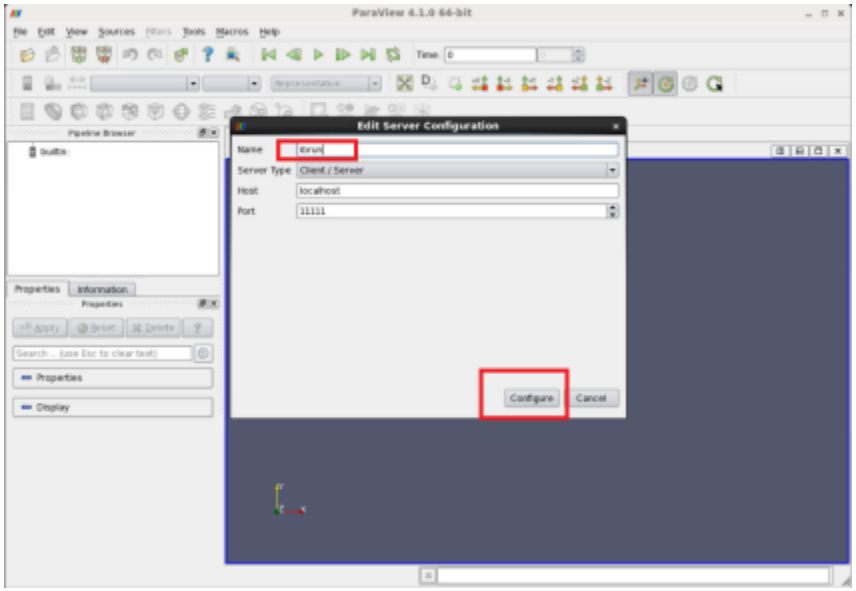
Visualization: ParaView in parallel

- Click the “connect” button (or choose file->connect)
- This will bring up a “choose a server” dialog.
- Select “add server”
- Paraview can launch backend parallel processes for us and connect to them, but we need to configure it to do so.



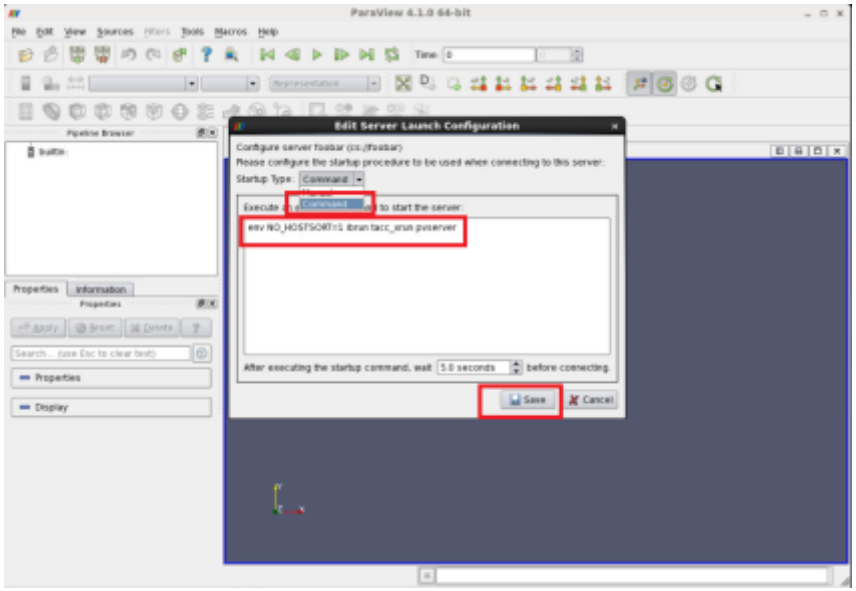
Visualization: ParaView in parallel

- Once you click “add”, an “edit server configuration” dialog will pop up.
- Give it this configuration a name (e.g. “ibrun”) in the “name” field”
- Leave the other fields alone
  - Server Type: Client/Server
  - Host: Localhost
  - Port: 11111
- Click “configure”



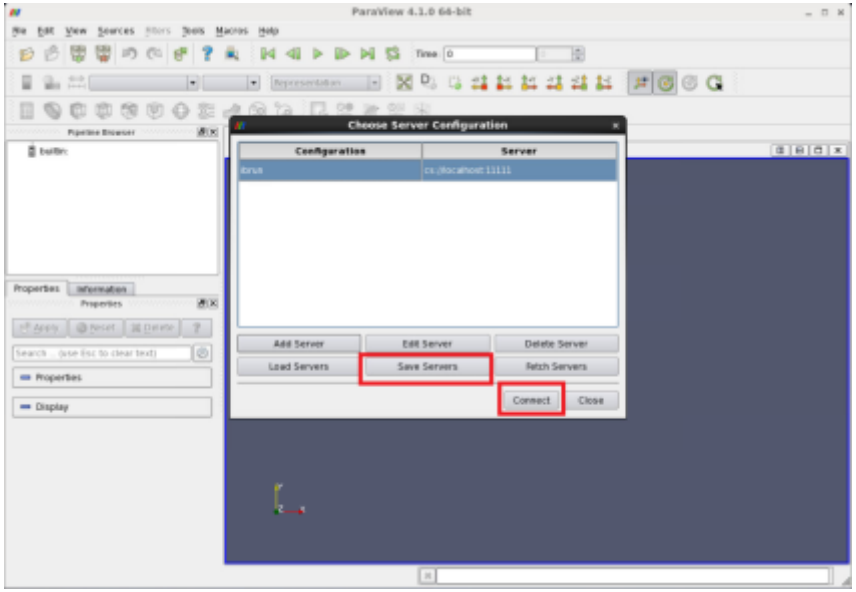
Visualization: ParaView in parallel

- Select Startup Type: "Command" from the dropdown menu
- In the large text box, type in "`env NO_HOSTSORT=1 ibrun tacc_xrun pvserver`"
  - This is the command that paraview will use to launch parallel backends
- Click "save"



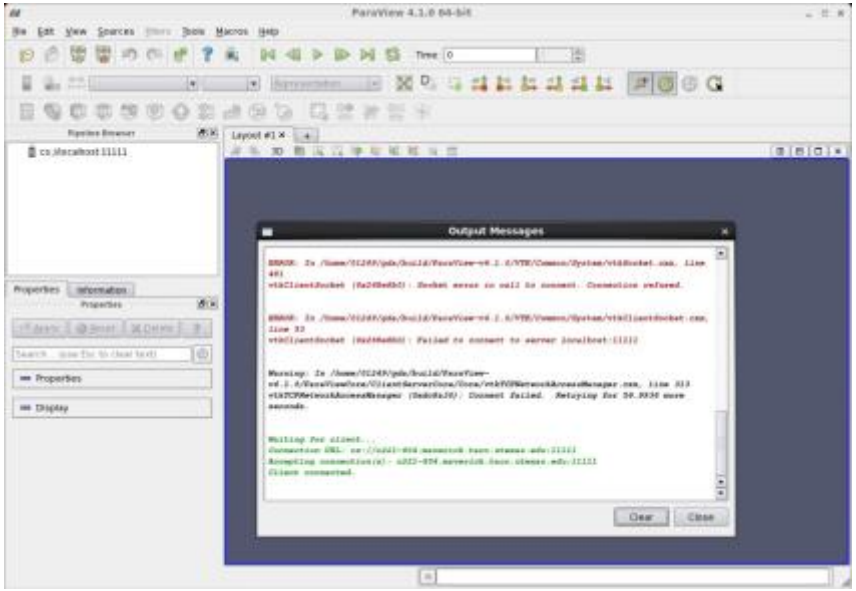
Visualization: ParaView in parallel

- Select “Save Servers” to have paraview remember these settings
  - Otherwise, you’ll need to type this information every time you restart paraview!
- It will prompt you for a filename. Just type in something like “servers” and click OK
- Click “connect” to have paraview launch the parallel backends.



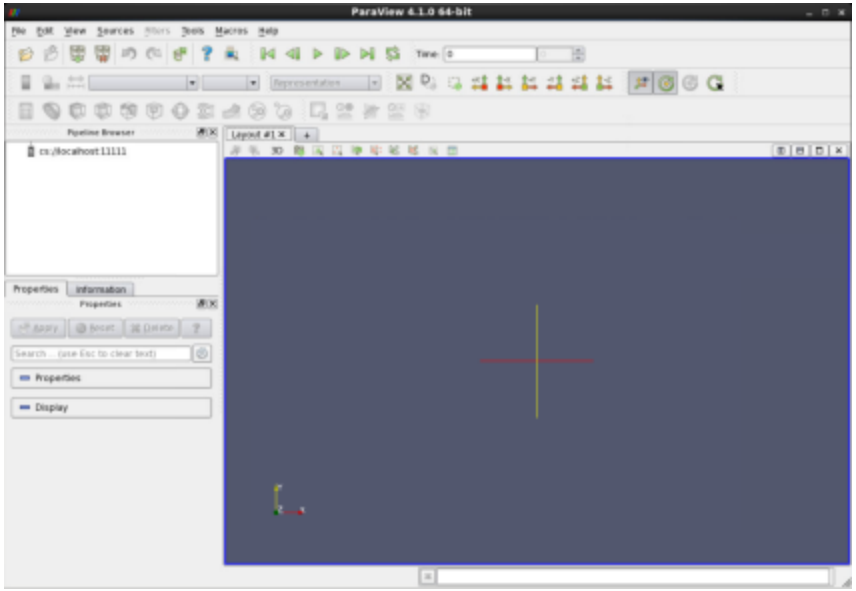
Visualization: ParaView in parallel

- You will see several windows pop up that report status.
- Wait for the green “Client Connected” message, ignore connection failure messages unless they are persistent. Close the window when it finishes connecting.
- The number and location of backends is automatically determined by your initial settings from submitting the VNC job.



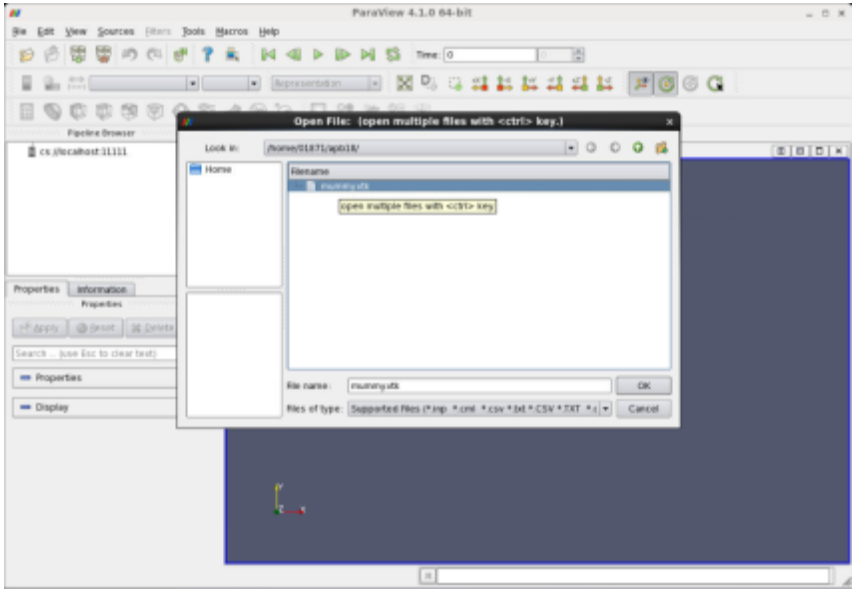
Visualization: ParaView in parallel

- ParaView is now ready to use. It looks no different from “normal”
- The only visible difference: the pipeline browser now starts with “cs://localhost:11111”
- You can proceed as usual, ParaView will automatically use all available resources in parallel.



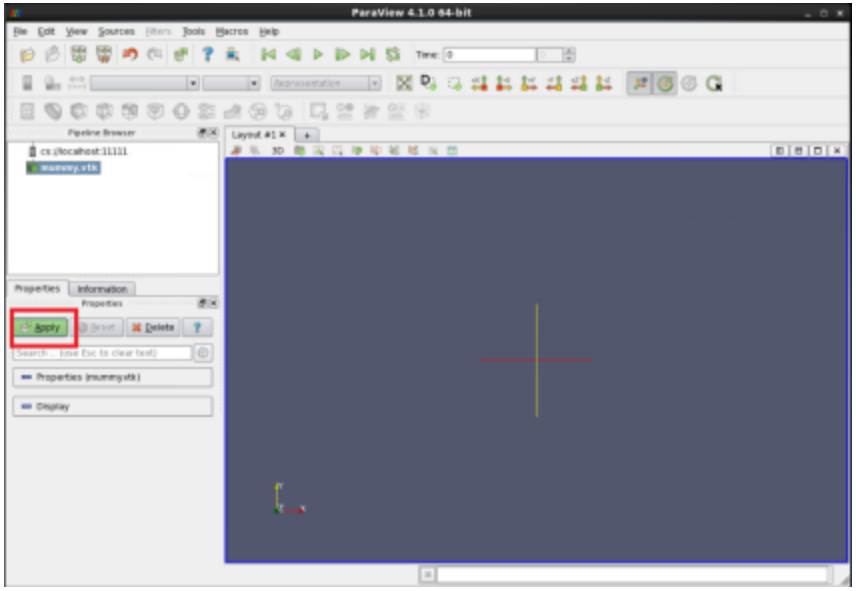
Visualization: ParaView in parallel

- Select File->Open to load the example dataset
- Navigate to `~tg459572/LABS/`
- Load the mummy dataset: `mummy.vtk`, click OK.



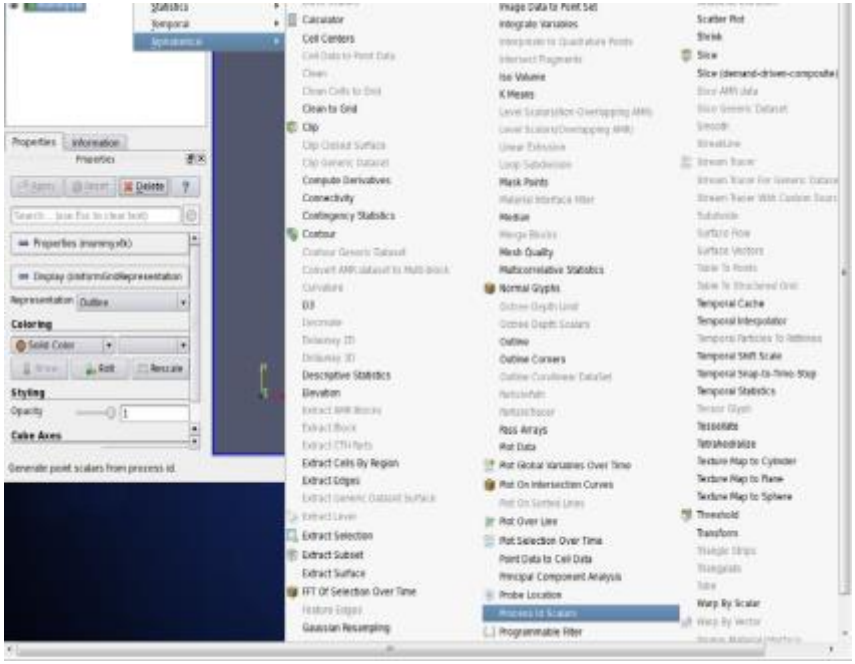
Visualization: ParaView in parallel

- Once loaded, click “apply” in the “properties” tab of the object inspector
- We will now add a filter which augments this data by adding another variable representing backend process ID for each point. This will indicate which backend is processing which portion of the data.



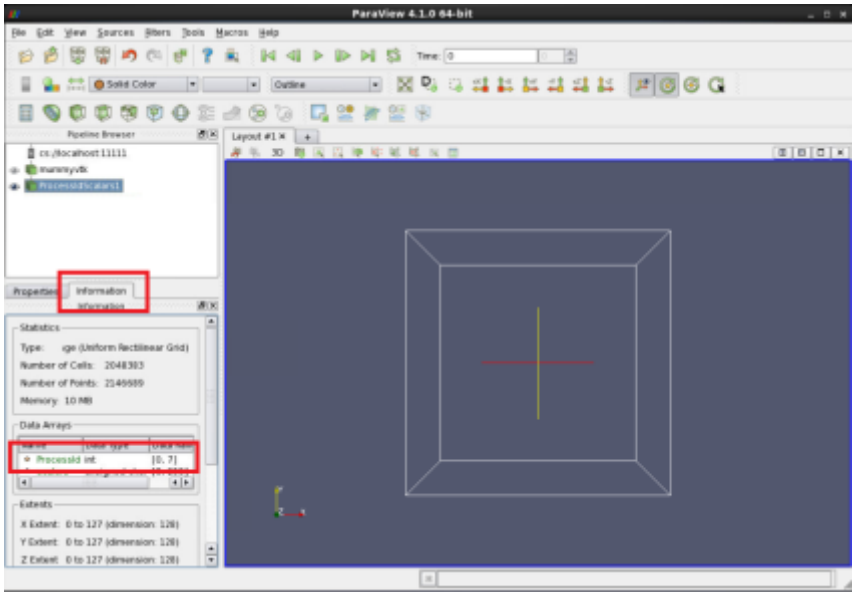
Visualization: ParaView in parallel

- In the “Filters” menu, go down to “Alphabetical”
- This will open a large menu showing all available filters. Navigate to the “Process Id Scalars” filter and choose it.
- Click “apply” in the “properties “tab of the object inspector for the ProcessID Scalars filter (as you did in the previous slide)



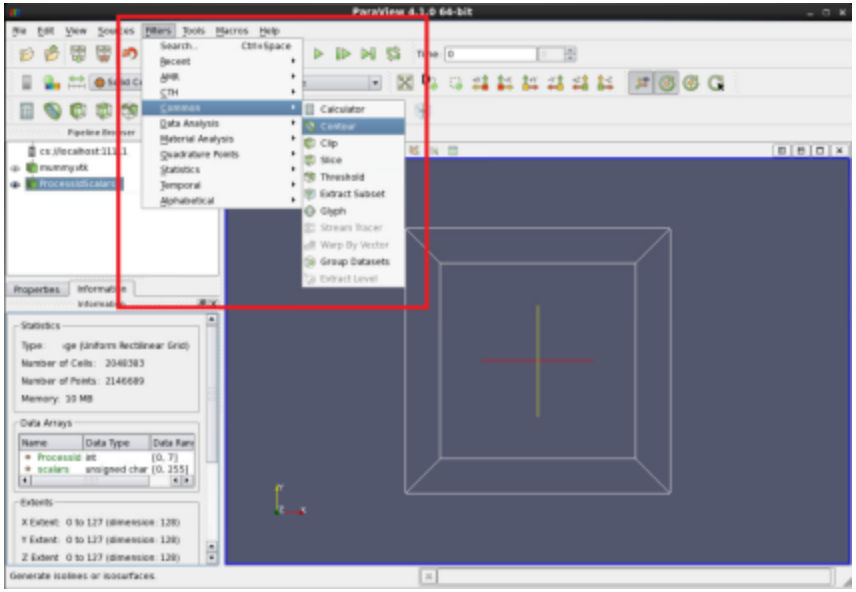
Visualization: ParaView in parallel

- Go to the “information” tab of the object inspector for the ProcessID Scalars filter. Scroll down until you can see the “Data Arrays” section.
- Observe that there is a new array named “ProcessId” containing integers ranging from 0-7.
  - These values map to our eight backend servers



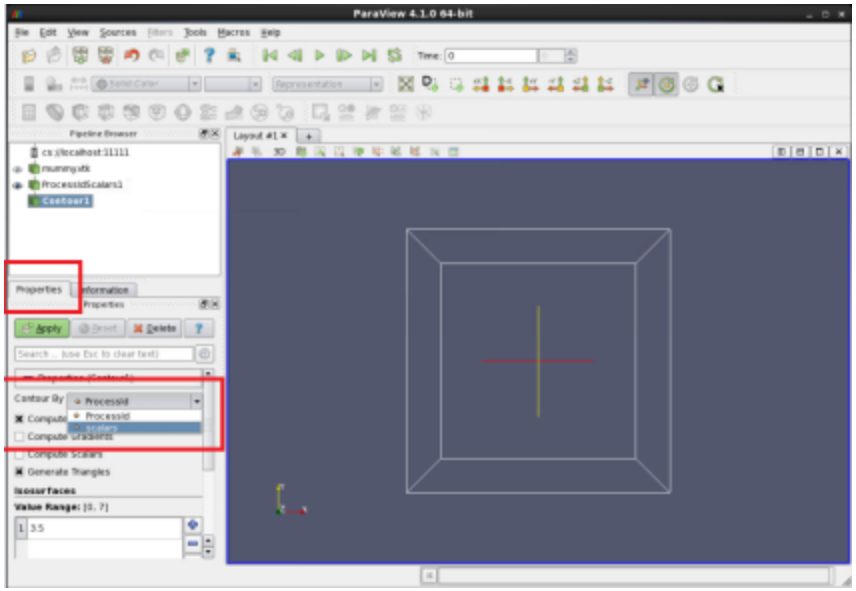
Visualization: ParaView in parallel

- Add and apply a contour filter via Filters -> Common -> Contour
- We will use this to produce an isosurface of the mummy skull wherever the value is 128.
- We will color this surface by ProcessId
- Be sure to click “Apply” after selecting the Contour filter, as you have done before



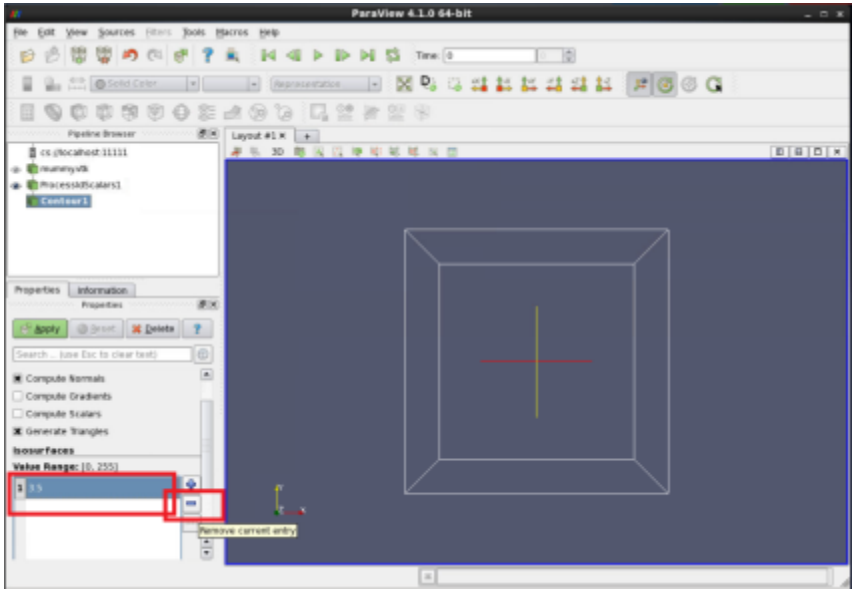
Visualization: ParaView in parallel

- In the “properties” tab of the object inspector for the contour filter, find the “contour section”
- The value will likely be “ProcessId” change it to “scalars”
  - We want to use the “scalars” data to produce the surface.



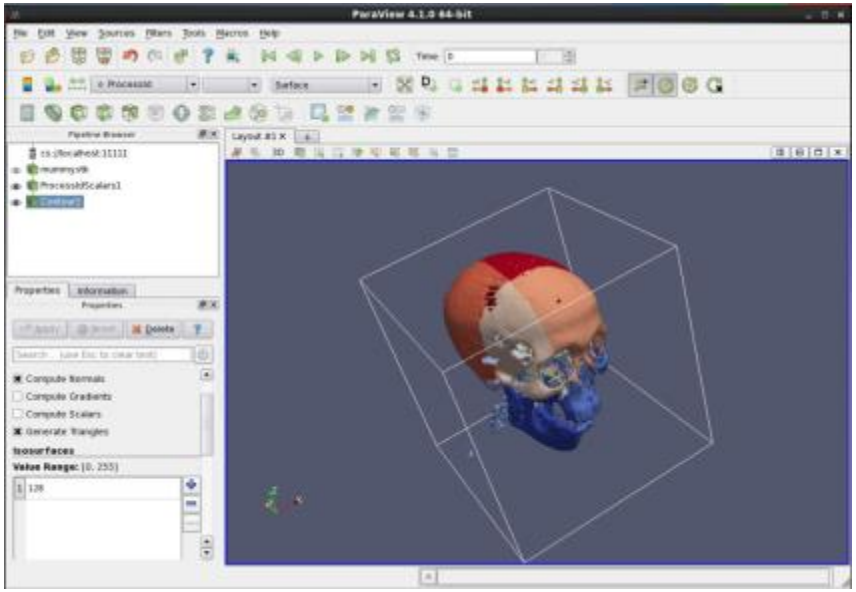
Visualization: ParaView in parallel

- Still in the “properties” tab, scroll down to the “isosurfaces” section.
- Delete the existing value (probably) 3.5 by selecting it and clicking on the minus (-) button
- Click on the plus (+) button to add a new value of 128.
- Click ‘apply’ when finished, as usual



Visualization: ParaView in parallel

- You should now see a rendering where the contour surface is determined by the original data, but colored by process ID.
- Since we have 2 nodes at 4 processes per node, we should see 8 distinct colors. This is showing data parallelism.
- Click on the image and drag the mouse to rotate, zoom, etc.



Visualization: ParaView in parallel

- ParaView will automatically determine if backends perform data processing only (sending triangles for client to render), or perform rendering as well (send the pixels to directly display).
- This can be tweaked via going into “settings” from the “Edit” menu.
  - Select “Server” under “Render View” in the tree on the left of the dialog box