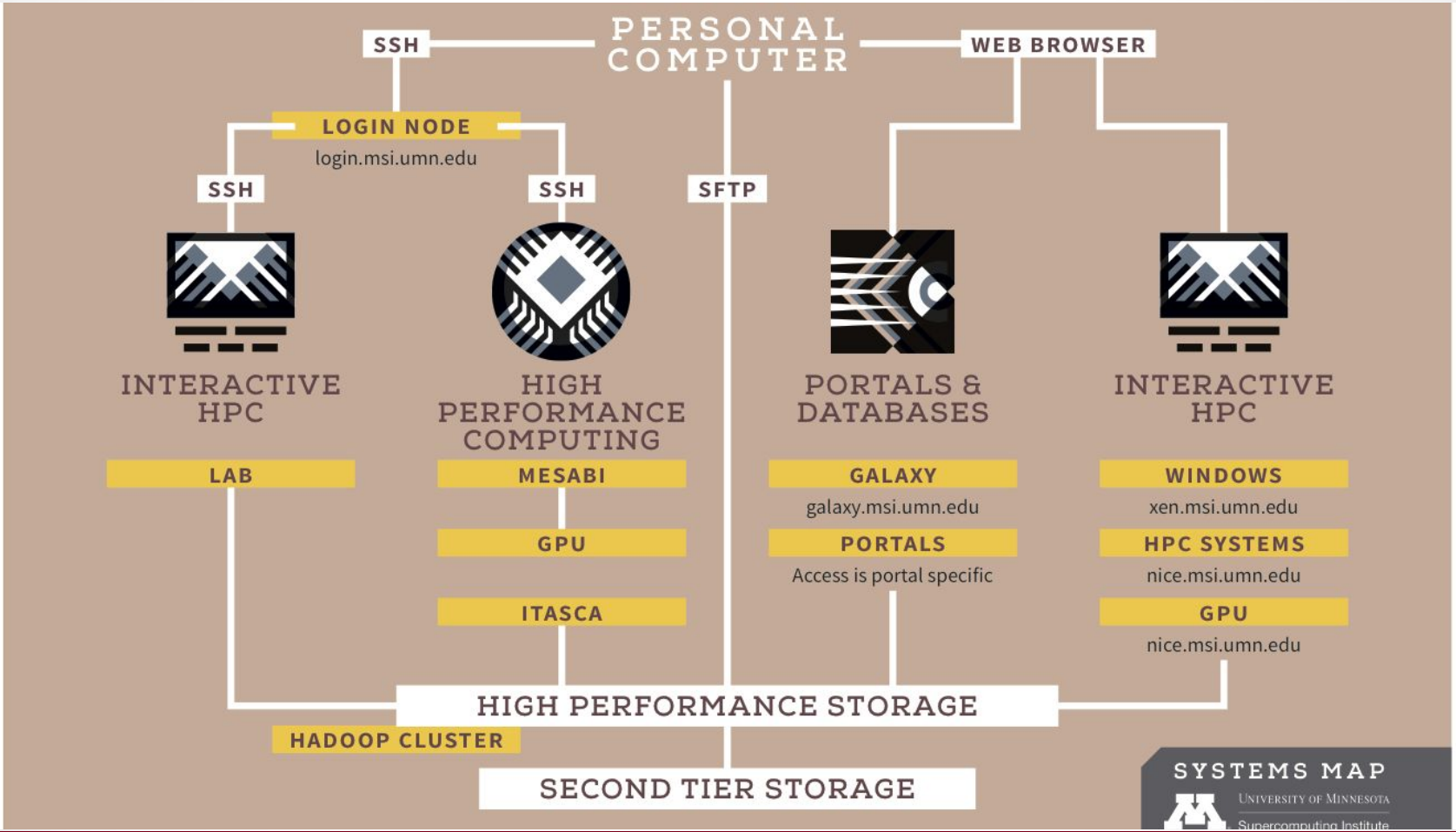
JupyterHub at MSI:
Building an Interactive HPC Gateway

Michael Milligan

July 22, 2016

MSI Systems Overview



Use Cases at MSI

- Supercomputer Gateway for Interactive HPC
- Enabling technology for Data Science Gateways

Use Cases at MSI

- Supercomputer Gateway for Interactive HPC
  - Interactive prototyping/visualization environment for popular language kernels (currently Python and R)
  - Provide access to parallel tools (ipyparallel, future: dask)
  - An alternative to ssh (esp notebook and terminal)
  - An alternative to remote desktop (terminal and visualization)
- Enabling technology for Data Science Gateways

Use Cases at MSI

- Supercomputer Gateway for Interactive HPC
- Enabling technology for Data Science Gateways
  - Used as a portal for interacting with data projects (built in collaboration with MSI researcher users)
  - Leverage widgets, Jupyter dashboard tools
  - Typically running in containers segregated from main systems

Technology Stack

Apache web server
[reverse proxy, SSL termination, Shibboleth auth]

⇩

configurable-http-proxy

⇩

JupyterHub server
[batchspawner, profiles, remoteuser]

⇩

Job scheduling engine (Torque)

BatchSpawner

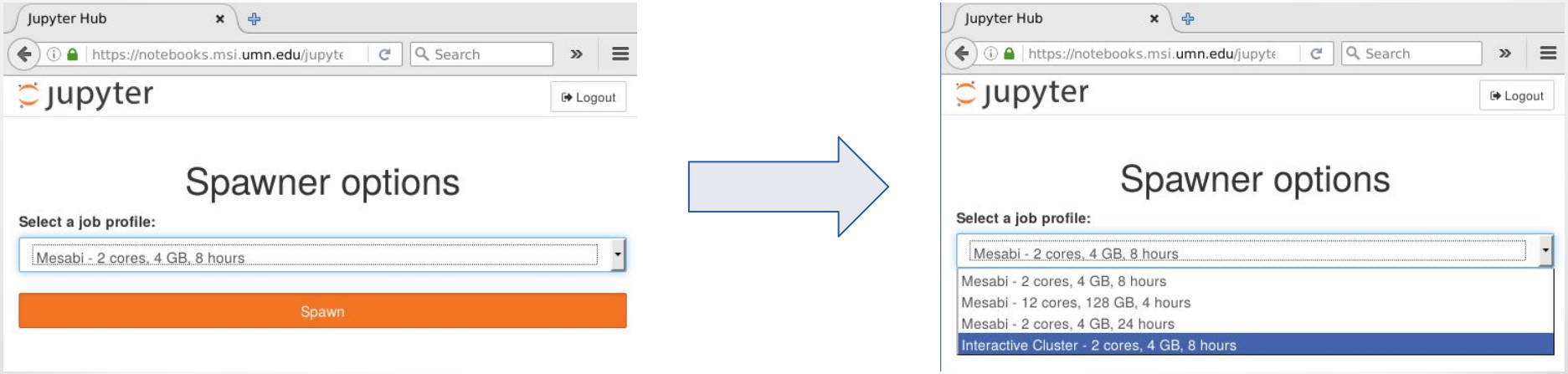
https://github.com/jupyterhub/batchspawner

- Generic interface to job scheduling engines
  - currently supports Torque, SLURM, SGE
  - support new engines in ~20 LoC
- Designed for every-site-is-unique HPC world
  - heavily customizable via templates controlled by site admin
  - no particular assumptions about architecture or security model
- In use now
  - in public-beta production at MSI
  - various prototype deployments at ~dozen institutions

Profiles Spawner

Included in BatchSpawner repo

- Allows user control of spawning process **without** introducing unsanitizable user data



Check out my 2016 SciPy talk

JupyterHub as an Interactive Supercomputing Gateway

for more details

Feature Requests

- User-facing feature requests
  - reinstate ipyparallel Jupyter tab (solved?)
  - Related: several Jupyter extensions we'd **really** like to see
    - JS-driven VNC client similar to JS terminal tool
    - Integration with Globus file transfer APIs
    - ability to forward web dashboards created by misc services running on (otherwise inaccessible) compute nodes
      e.g. Dask progress display
- Developer-facing needs
  - better understanding of how to plug external modules (i.e. batchspawner) into Jupyter testing framework