# Introduction to OSG

**Tim Cartwright**

*University of Wisconsin–Madison*

***OSG School Director***

*OSG Campus Coordinator*

# Overview

So far, we have seen how to use HTC on one cluster

***Don't let computing be a barrier to research!***

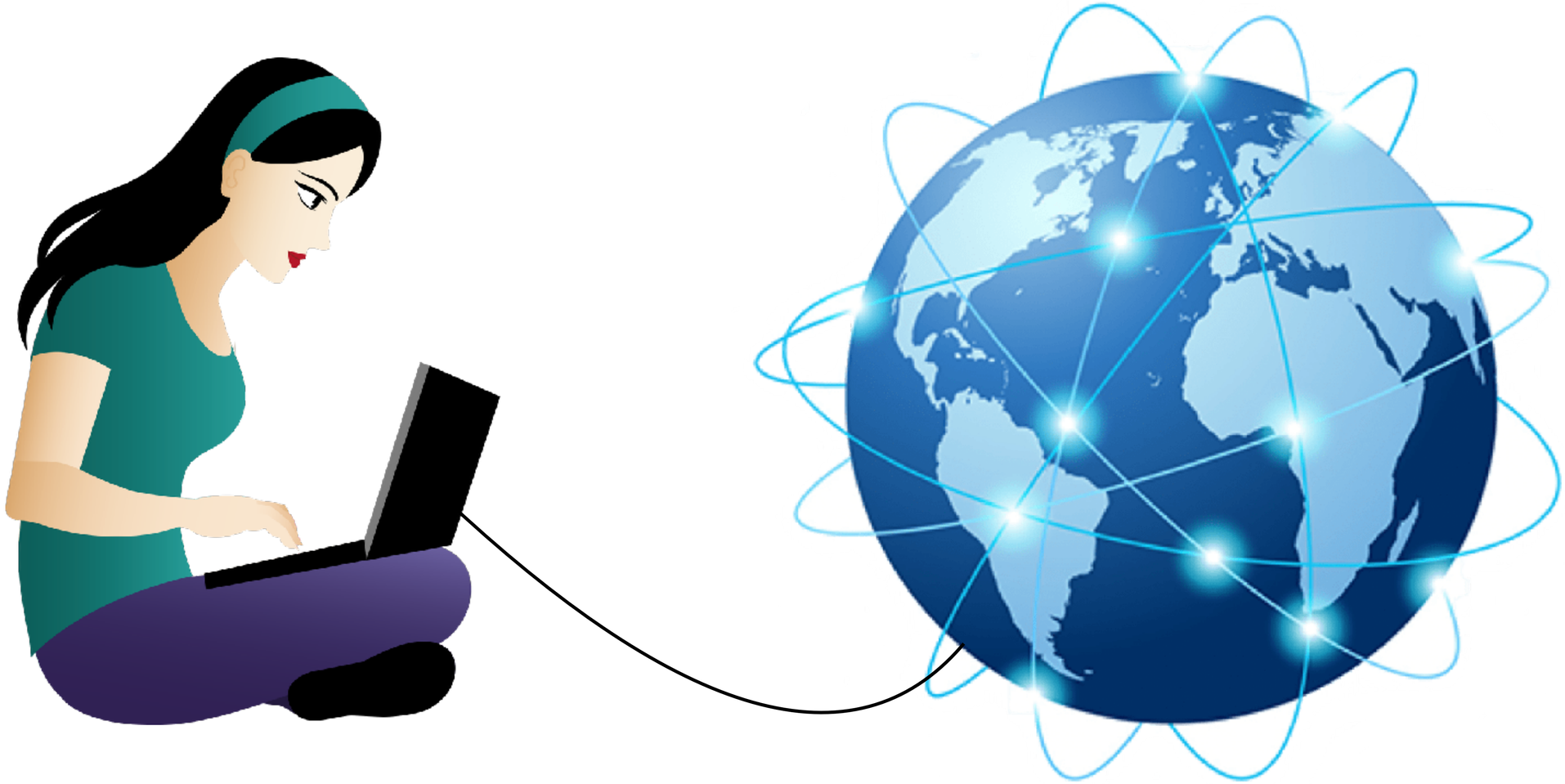
*(Sometimes, one cluster is not enough)*

Today, we see what it takes to get more capacity *

* ***Caveat:*** *I will focus on compute capacity; Wed. will focus on data.*

# What Researchers Want

## Submit locally, run globally



# Yesterday

**Access Point**

```
$ ls
science.sub
input.dat
$ condor_q
```

**PATh**

https://sweetclipart.com/
https://www.flaticon.com/free-icons/server

# The Goal

```
$ ls
science.sub
input.dat
$ condor_q
```

Access Point

Maine

PATh

Nebraska

NMSU

https://sweetclipart.com/
https://www.flaticon.com/free-icons/server

# What That Takes

- ✓ Lots of networked computing capacity
  - OSPool capacity are contributed by campuses!
  - (PATh Facility, used yesterday, is owned by PATh)
- ✓ Capacity owners who are motivated to share
- ✓ Trust among owners, researchers, OSG staff
- ✓ Automation to make it work at scale
- ✓ No allocations or charges would be nice!

# Demo, Part I

# Behind the Curtain

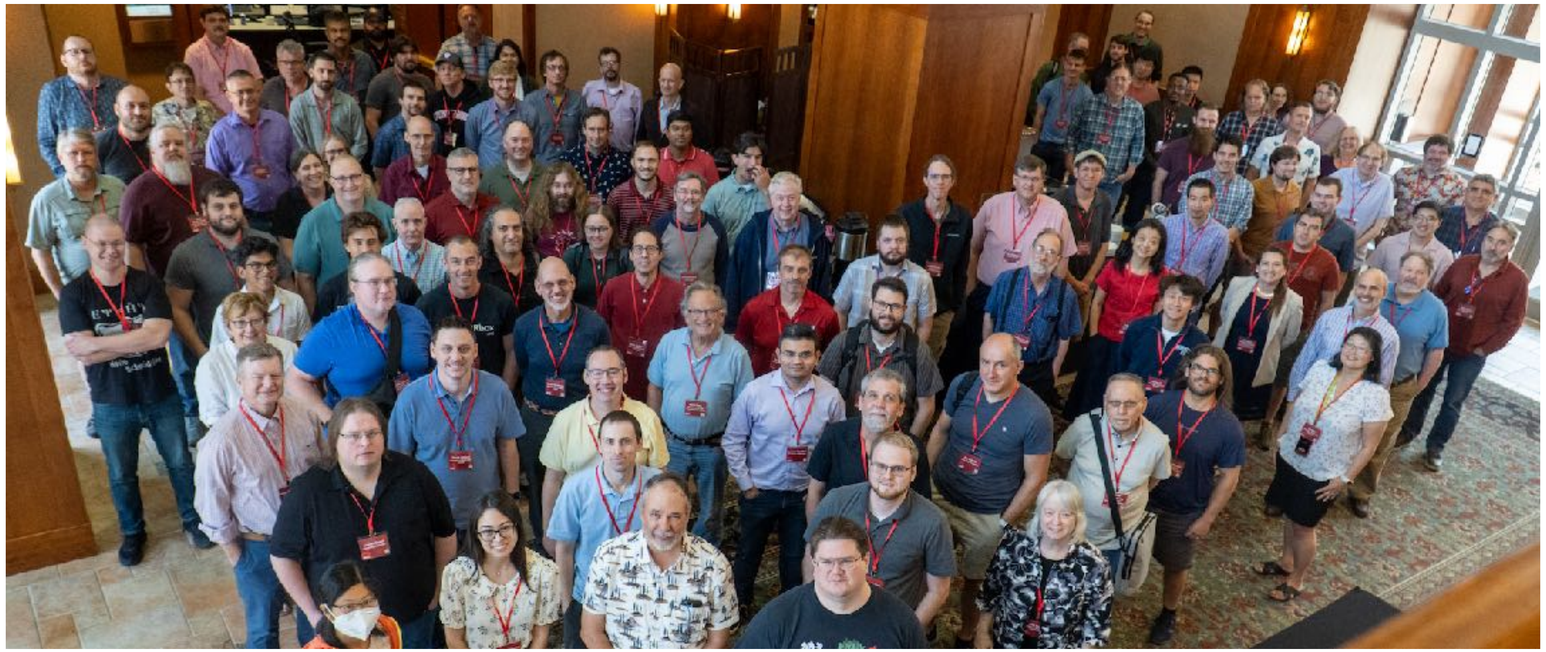
# Reasons for Continuing

- So why learn more about how the OSPool works?
  - May change how you plan to run computing there
  - May change the way you use the Access Point
  - May change how you handle issues that arise
- What is there to learn? (outline of rest of talk)
  - Concepts of OSG, Pool, and Access Point
  - How the OSPool gets resources
  - How the OSPool differs from a local cluster

# What is OSG?

# OSG Defined, Version 1

- OSG Consortium – in this view, OSG ***is*** people:
  - Users: individuals PI/students to collabs. of 1,000s
  - Resource owners/contributors
  - Team: provides infrastructure, support, features, …



# OSG Defined, Version 2

- Pools of capacity
  - **Capacity:** compute, storage, and other systems that can be used for research workflows
  - **Services:** software infrastructure that manages capacity and makes features available

https://www.pngall.com/wp-content/uploads/5/Server-Rack-PNG-Free-Image.png

# OSG Defined, Version 3

- OSG Access Point
  - Where you go to do computing
  - Has access to capacity (constantly changing)
  - Provides means for accessing data (see Wednesday)

```
[tim.cartwright@ap40 ~]$ condor_version
$CondorVersion: 10.7.0 2023-07-10 BuildID: 659788 PackageID: 10.7.0-0.659788 RC $
$CondorPlatform: x86_64_AlmaLinux8 $
[tim.cartwright@ap40 ~]$
```

# Getting Capacity for OSPool

# The Goal

**Access Point**

```
$ ls
science.sub
input.dat
$ condor_q
```

Maine

PATh

Nebraska

NMSU

https://sweetclipart.com/
https://www.flaticon.com/free-icons/server

# The Goal

Access Point

```
$ ls
science.sub
input.dat
$ condor_q
```

Maine

PATh

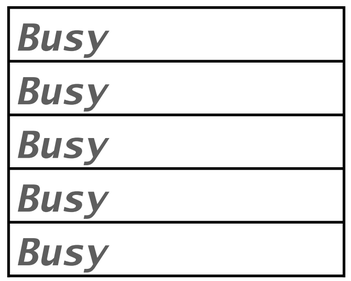
Nebraska

NMSU

# OSPool dHTC Diagram

## 1. Before OSPool

Nothing available at Wisc. 😭




## Access Point

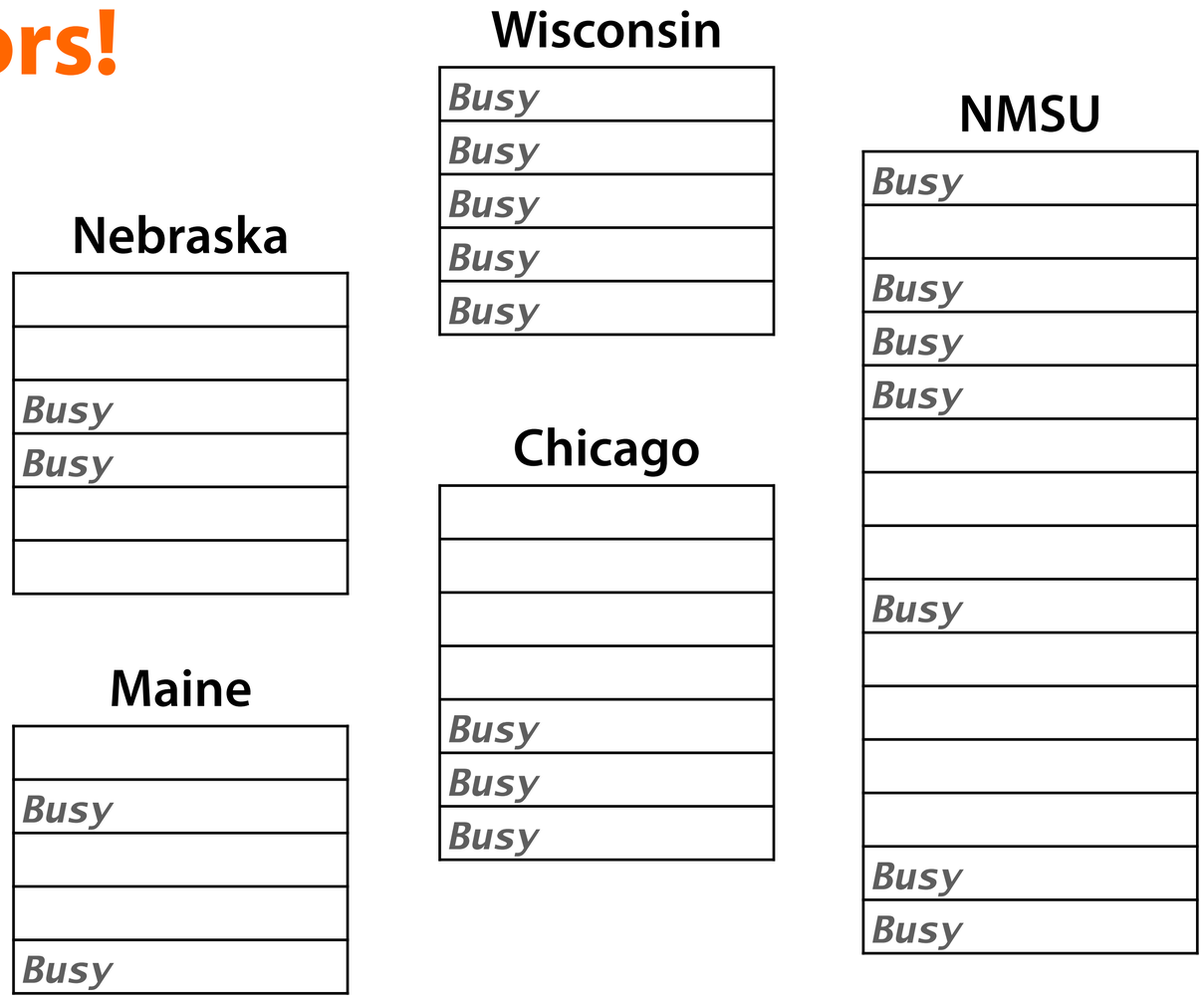
```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

# OSPool dHTC Diagram

## 2. Add capacity contributors!

### Access Point

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```



# OSPool dHTC Diagram

## 3. Request capacity (method #1)

Start Execution Points on clusters

### Access Point

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

**Nebraska**

| |
|---|
| OSPool EP NU1 |
| OSPool EP NU2 |
| *Busy* |
| *Busy* |
| |
| |

**Wisconsin**

| |
|---|
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |

**NMSU**

| |
|---|
| *Busy* |
| OSPool EP NM1 |
| *Busy* |
| *Busy* |
| *Busy* |
| OSPool EP NM4 |
| OSPool EP NM2 |
| OSPool EP NM3 |
| *Busy* |
| |
| |
| |
| |
| *Busy* |
| *Busy* |

**Chicago**

| |
|---|
| |
| |
| OSPool EP UC2 |
| OSPool EP UC1 |
| *Busy* |
| *Busy* |
| *Busy* |

**Maine**

| |
|---|
| OSPool EP ME2 |
| *Busy* |
| OSPool EP ME1 |
| OSPool EP ME3 |
| *Busy* |

# OSPool dHTC Diagram

## 4. EPs add capacity to Pool

(I am not explaining how yet)

### AP

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

### Pool

| | |
|---|---|
| OSPool EP NU1 | *idle* |
| OSPool EP NU2 | *idle* |
| OSPool EP ME1 | *idle* |
| OSPool EP ME2 | *idle* |
| OSPool EP ME3 | *idle* |
| OSPool EP UC1 | *idle* |
| OSPool EP UC2 | *idle* |
| OSPool EP NM1 | *idle* |
| OSPool EP NM2 | *idle* |
| OSPool EP NM3 | *idle* |
| OSPool EP NM4 | *idle* |
| | |

**Nebraska**

| |
|---|
| OSPool EP NU1 |
| OSPool EP NU2 |
| *Busy* |
| *Busy* |
| |
| |

**Maine**

| |
|---|
| OSPool EP ME2 |
| *Busy* |
| OSPool EP ME1 |
| OSPool EP ME3 |
| *Busy* |

**Wisconsin**

| |
|---|
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |

**Chicago**

| |
|---|
| |
| |
| OSPool EP UC2 |
| OSPool EP UC1 |
| *Busy* |
| *Busy* |
| *Busy* |

**NMSU**

| |
|---|
| *Busy* |
| OSPool EP NM1 |
| *Busy* |
| *Busy* |
| *Busy* |
| OSPool EP NM4 |
| OSPool EP NM2 |
| OSPool EP NM3 |
| *Busy* |
| |
| |
| |
| |
| *Busy* |
| *Busy* |

# OSPool dHTC Diagram

## 5. Run jobs

HTCondor with AP & Pool

### AP

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

### Pool

| | |
|---|---|
| OSPool EP NU1 | Job1.4 |
| OSPool EP NU2 | *idle* |
| OSPool EP ME1 | Job1.0 |
| OSPool EP ME2 | Job1.3 |
| OSPool EP ME3 | *idle* |
| OSPool EP UC1 | Job1.2 |
| OSPool EP UC2 | Job1.6 |
| OSPool EP NM1 | Job1.8 |
| OSPool EP NM2 | Job1.12 |
| OSPool EP NM3 | Job1.10 |
| OSPool EP NM4 | *idle* |
| | |

**Nebraska**

| |
|---|
| NU1 > Job1.4 |
| NU2 > *idle* |
| *Busy* |
| *Busy* |
| |
| |

**Maine**

| |
|---|
| ME2 > Job1.3 |
| *Busy* |
| ME1 > Job1.0 |
| ME3 > *idle* |
| *Busy* |

**Wisconsin**

| |
|---|
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |

**Chicago**

| |
|---|
| |
| |
| UC2 > Job1.6 |
| UC1 > Job1.2 |
| *Busy* |
| *Busy* |
| *Busy* |

**NMSU**

| |
|---|
| *Busy* |
| NM1 > Job1.8 |
| *Busy* |
| *Busy* |
| *Busy* |
| NM4 > *idle* |
| NM2 > Job1.12 |
| NM3 > Job1.10 |
| *Busy* |
| |
| |
| |
| |
| *Busy* |
| *Busy* |

# OSPool dHTC Diagram

## 6. Get resources (method #2)

Direct contributions

### AP

```
Job1.0
Job1.1
Job1.2
Job1.3
…
Job1.1999
```

### Pool

| | |
|---|---|
| OSPool EP NU1 | Job1.4 |
| OSPool EP NU2 | *idle* |
| OSPool EP ME1 | Job1.0 |
| OSPool EP ME2 | Job1.3 |
| OSPool EP ME3 | *idle* |
| OSPool EP UC1 | Job1.2 |
| OSPool EP UC2 | Job1.6 |
| OSPool EP NM1 | Job1.8 |
| OSPool EP NM2 | Job1.12 |
| OSPool EP NM3 | Job1.10 |
| OSPool EP NM4 | *idle* |
| NM Contrb EP1 | *idle* |

**Nebraska**

| |
|---|
| NU1 > Job1.4 |
| NU2 > *idle* |
| *Busy* |
| *Busy* |
| |
| |

**Maine**

| |
|---|
| ME2 > Job1.3 |
| *Busy* |
| ME1 > Job1.0 |
| ME3 > *idle* |
| *Busy* |

**Wisconsin**

| |
|---|
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |
| *Busy* |

**Chicago**

| |
|---|
| |
| |
| UC2 > Job1.6 |
| UC1 > Job1.2 |
| *Busy* |
| *Busy* |
| *Busy* |

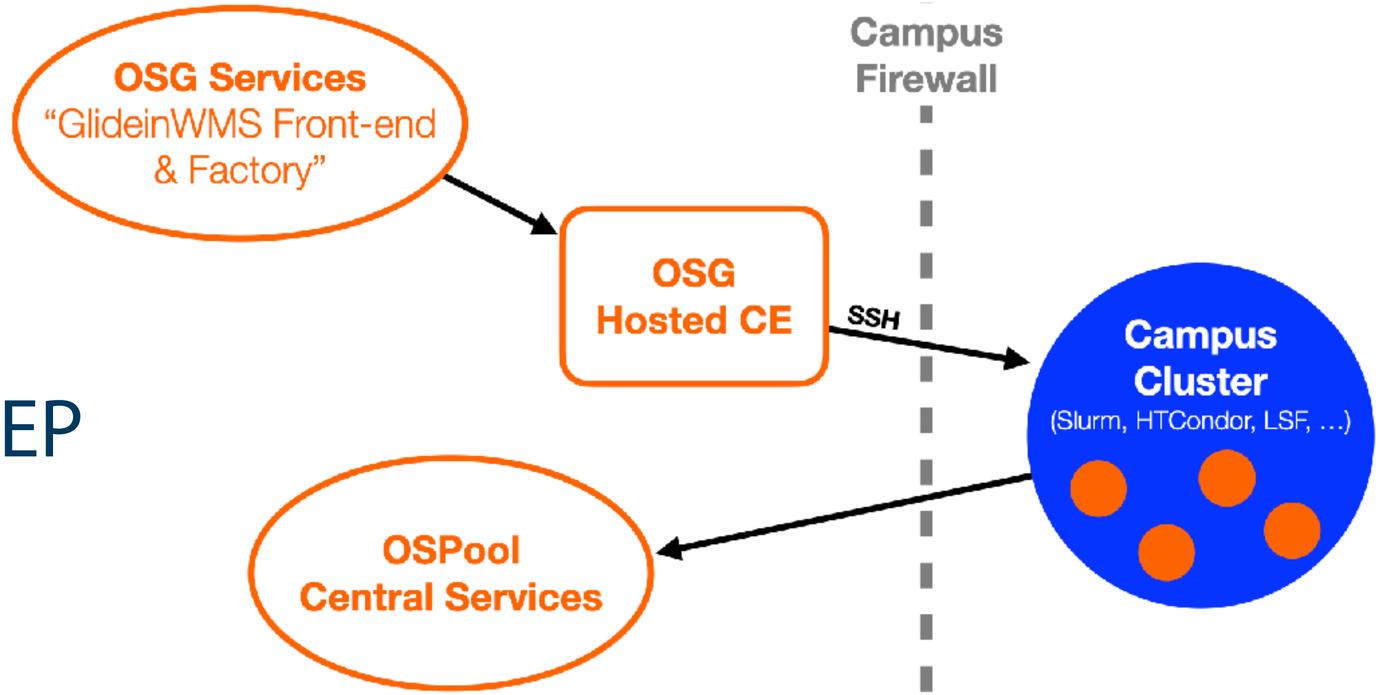
**NMSU**

| |
|---|
| *Busy* |
| NM1 > Job1.8 |
| *Busy* |
| *Busy* |
| *Busy* |
| NM4 > *idle* |
| NM2 > Job1.12 |
| NM3 > Job1.10 |
| *Busy* |
| |
| NM Contrb EP1 |
| |
| |
| *Busy* |
| *Busy* |

# OSPool dHTC – A Few Details

## For a batch scheduler:

- CE requests capacity (as jobs) based on demand
- Scheduler may run req.s
- Our SW creates *Execution Point* & adds to OSPool
- OSPool workloads run on EP



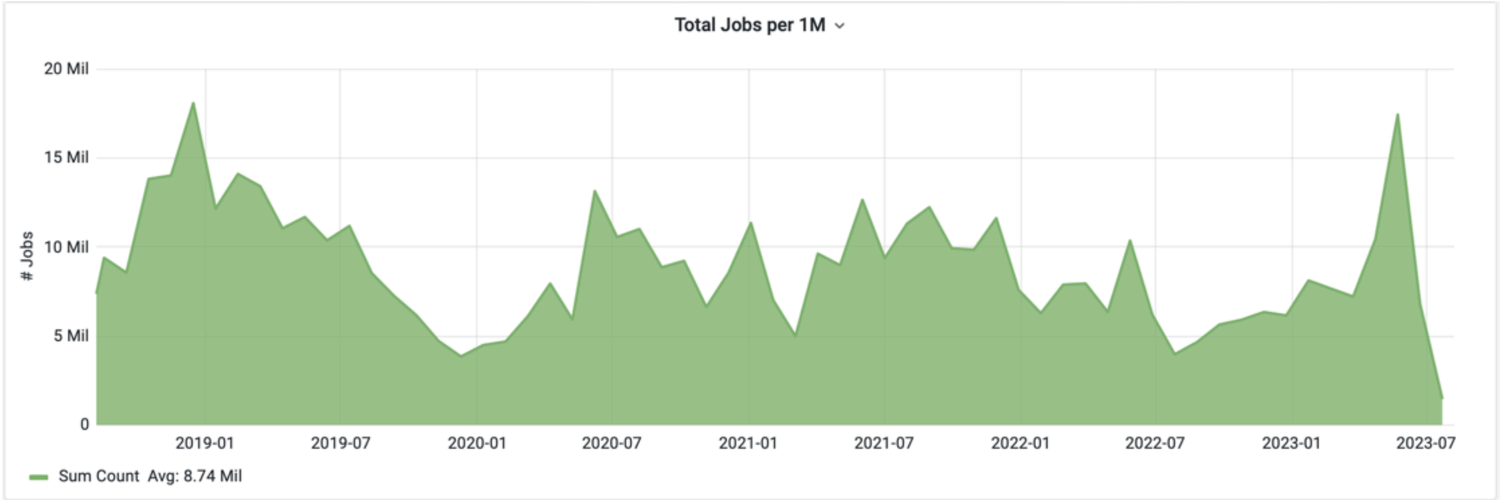

## Using containers:

- Admin starts containers
- SW makes EP, etc. (same)

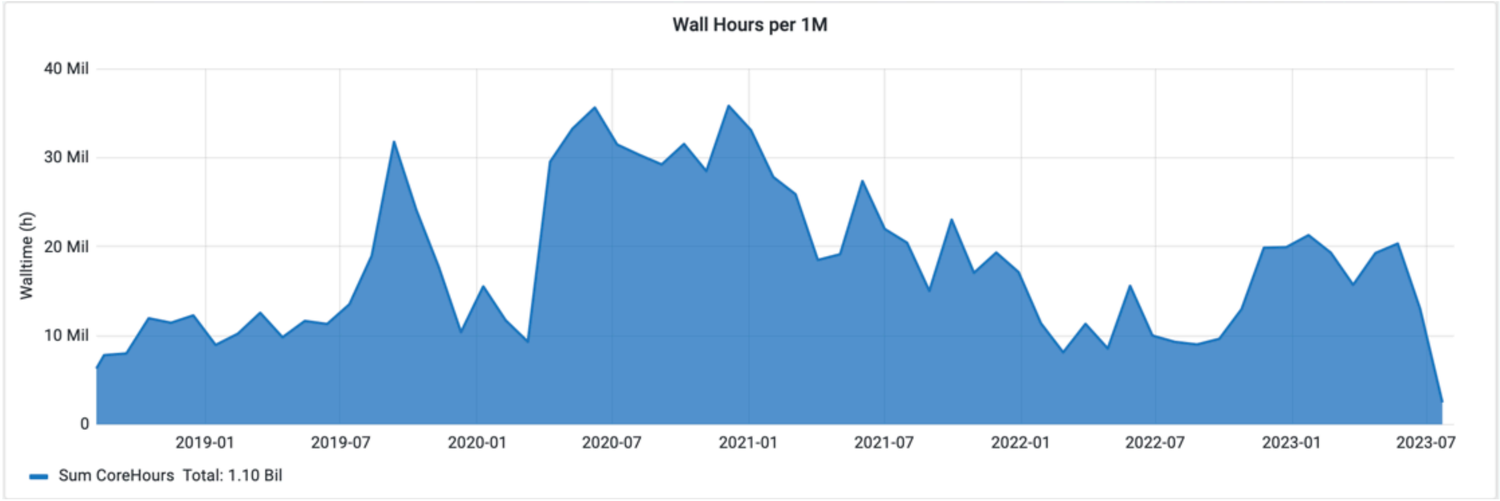
# OSPool Contributors *(United States)*

# OSPool Usage

## Jobs




## Hours




# Using OSPool

# OSPool Is HTCondor

- OSPool is an HTCondor pool:
  You have condor_q, condor_submit, DAGMan, etc.
- OSPool bonus features!
  - More capacity (usually) than a typical local system
  - Some storage on Access Point (Data lecture, Wed.)
  - Some special resources, like GPUs (AI/ML topic, Thu.)
- How does OSPool differ from local one? ***Variety***

# Varied Hardware

- Wider variety of CPUs (type, speed), memory, …
- Request what you need in submit files (**`request_cpus`**, **`request_memory`**, **`request_disk`**)
- Some specific hardware may be specified; search for documentation or contact us
  - Often in submit-file **`requirements`** expression
  - Example: GPU needs (ask us!)

# Varied OSs and Software

- Varied Operating Systems
  - All Linux, *mostly* recent, but lots of variation
  - Changes to CentOS 8 => new variants (e.g., Alma)
  - Software on the Access Point **probably won't exist** on Execution Points! (e.g., specific Python version)
- Your software
  - Never assume your software is on Execution Points
  - The Software lecture (later today) is on this topic!

# Varied Access to Data

- No shared filesystem
  - Unlike some local clusters with shared filesystems
  - Thus, files must be transferred to Execution Points
- There are many ways to handle data
  - Data lecture is Wednesday morning

# Varied Policies

- Individual sites/clusters have their own policies
  - Example: Whether to kill jobs that exceed memory
  - Example: Maximum run-time of a job (or its glidein)
- If possible, set requirements for what you need
  - But this does not help with, e.g., maximum run-time
- Generally, try to make “OSG-sized” jobs (see next)

# What Makes a Good OSG Job?

| | Ideal Jobs!<br>(up to 10,000 cores across Jobs, per user!) | Still Very Advantageous! | Less-so, but maybe |
|---|---|---|---|
| **Cores**<br>(GPUs) | **1**<br>(1; non-specific type) | **<8**<br>(1; specific GPU type) | **>8 (or MPI)**<br>(multiple) |
| **Walltime** | **<10 hrs***<br>*or checkpointable | **<20 hrs***<br>*or checkpointable | **>20 hrs** |
| **RAM** | **<few GB** | **<10s GB** | **>10s GB** |
| **Input** | **<500 MB** | **<10 GB** | **>10 GB** |
| **Output** | **<1 GB** | **<10 GB** | **>10 GB** |
| **Software** | *'portable' (pre-compiled binaries, transferable, containerizable, etc.)* | *most other than* → | *Licensed software; non-Linux* |

# More OSG Tips – Security

- Computer security is hard — read the headlines!
- OSG does its best, but no system is perfect
- Some suggestions:
  - Use strong, distinct passwords for each account
  - Do not share your account
  - Avoid world-writable directories and files
  - Avoid sensitive software and data (no HIPAA!)
  - Do not try to work around security barriers; contact us to help meet your goals in a safe way

# Acknowledgements

# *You* Can Acknowledge OSG!

If you publish or present results that benefitted from using OSG services, please acknowledge us!

https://osg-htc.org/acknowledging

# Acknowledgements


- OSG team, especially Christina Koch; in past years: Brian Lin, Mats Rynge, and Jason Patton
- This work was supported by the National Science Foundation under Cooperative Agreement OAC-2030508

# A Few Suggestions

- **Exercises**
  - Today, some exercises will specify less, so try to use what you learned yesterday — first, from memory, if possible, then look things up
  - Use Slack! There are staff online who can help, too
- **Consultations**
  - Please consider signing up for a consultation, if you haven't already
  - If the slots fill up, we will try to make more!

# Demo, Part II