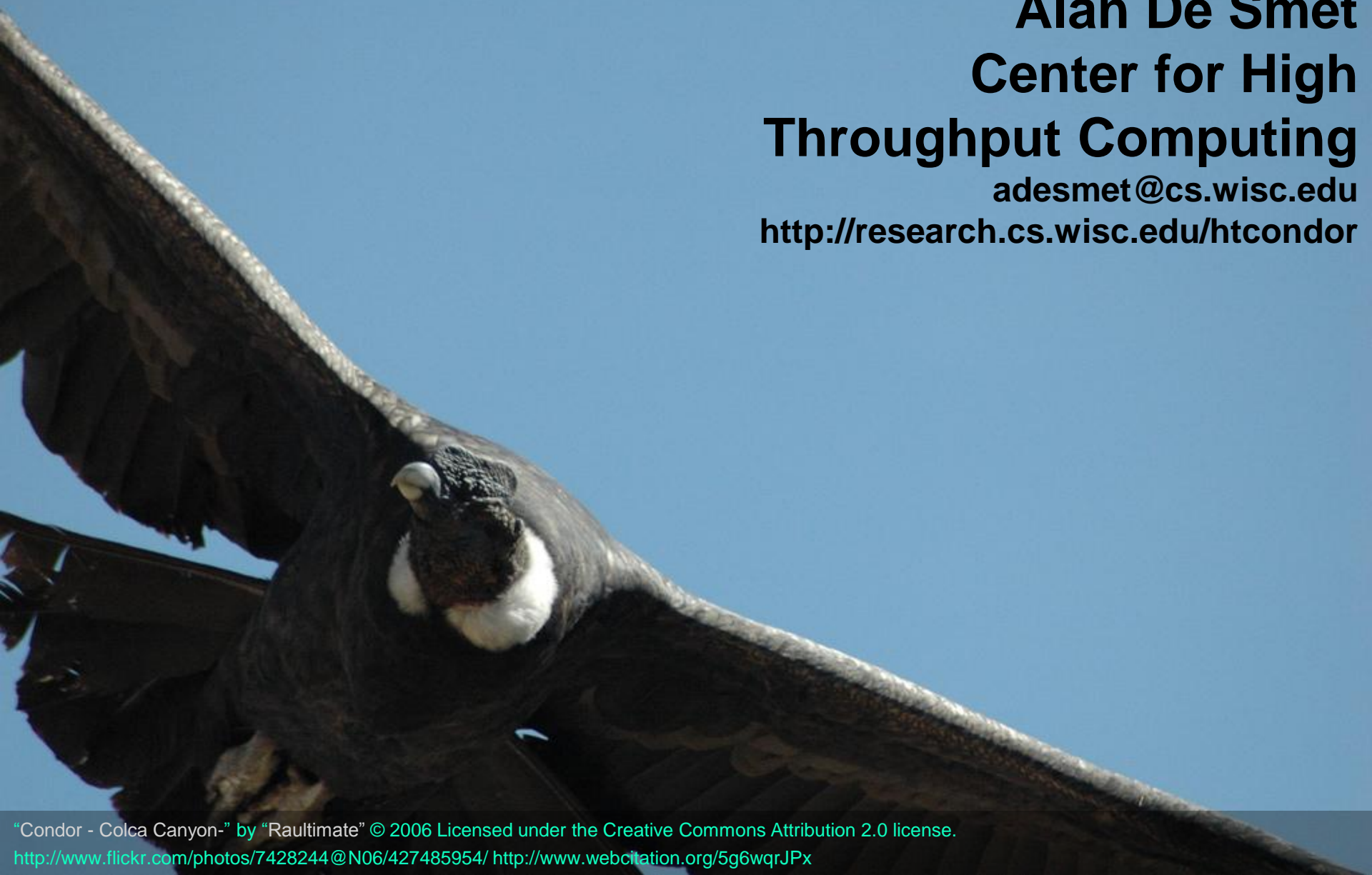


# ***Administrating HTCondor***

**Alan De Smet**  
**Center for High**  
**Throughput Computing**  
adesmet@cs.wisc.edu  
<http://research.cs.wisc.edu/htcondor>



# The next 60 minutes...

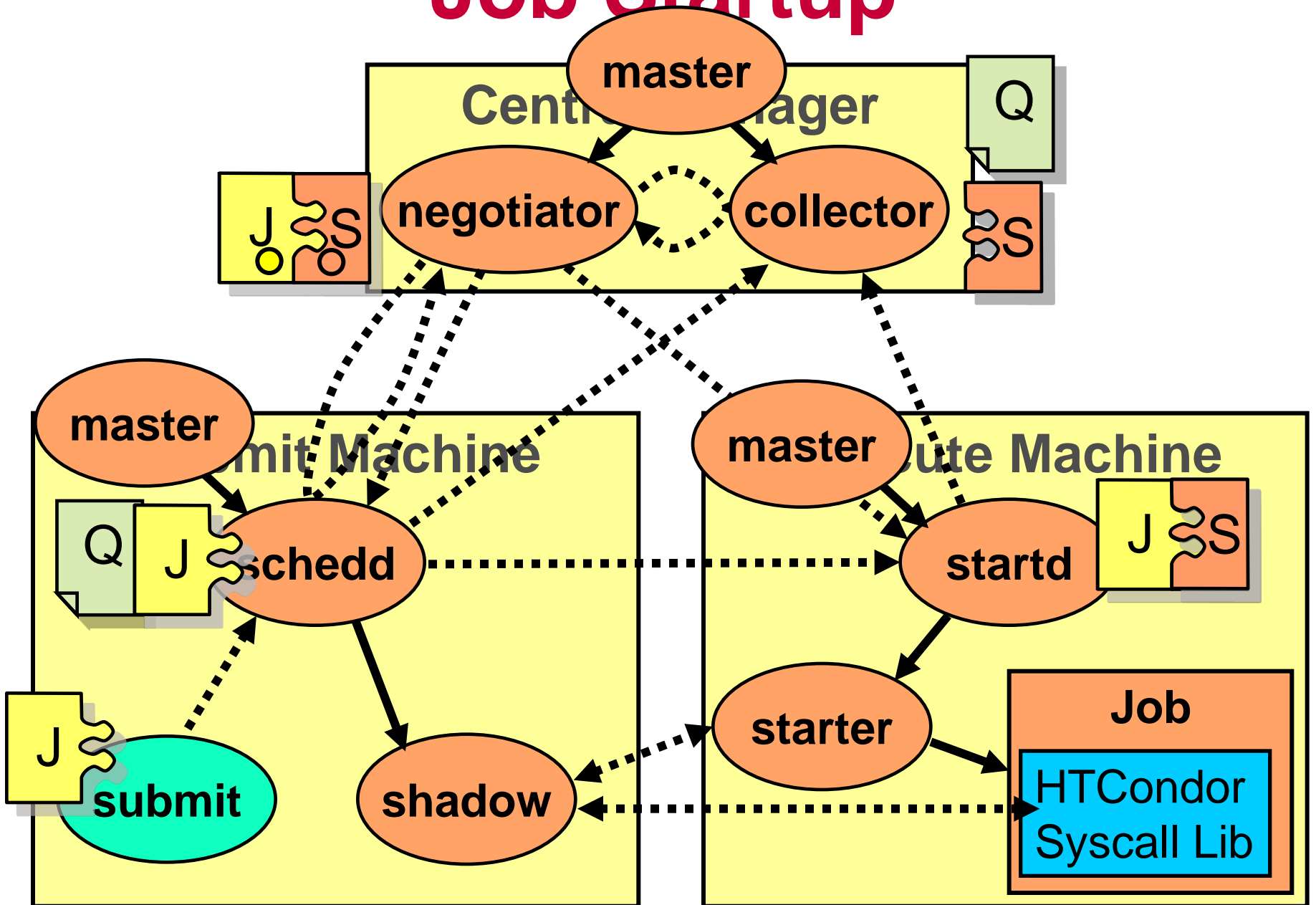
- › HTCondor  
Daemons & Job  
Startup
- › Configuration Files
- › Security, briefly
- › Policy Expressions
  - Startd (Machine)
  - Negotiator
- › Priorities
- › Useful Tools
- › Log Files
- › Debugging Jobs

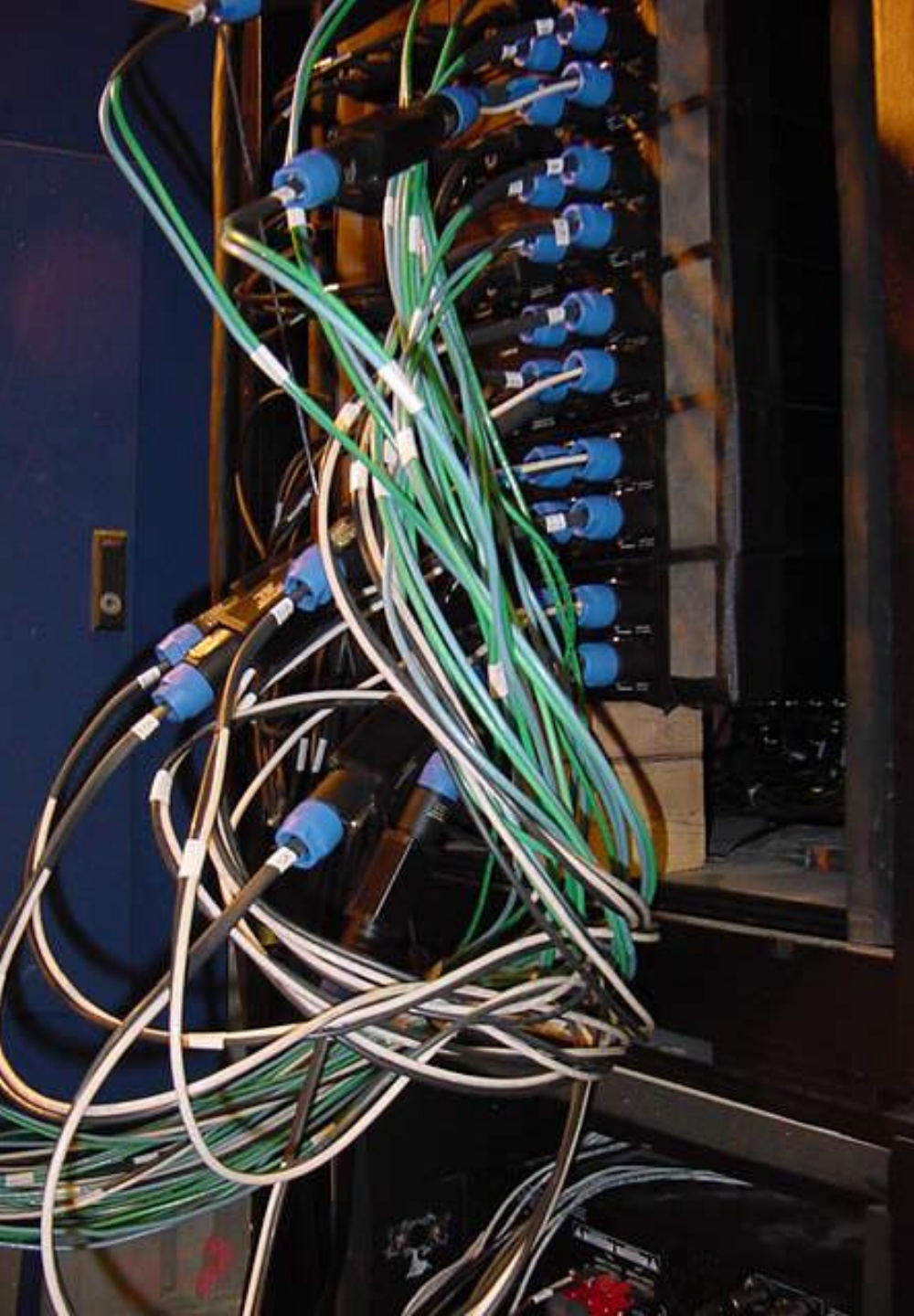


# ***Daemons & Job Startup***

“LUNAR Launch” by Steve Jurvertson (“jurvetson”) © 2006  
Licensed under the Creative Commons Attribution 2.0 license.  
<http://www.flickr.com/photos/jurvetson/114406979/>  
<http://www.webcitation.org/5XIfTI6tX>

# Job Startup





# ***Configuration Files***

“amp wiring” by “fbz\_” © 2005

Licensed under the Creative Commons Attribution 2.0 license

<http://www.flickr.com/photos/fbz/114422787/>

# Configuration File

- › Found either in file pointed to with the `CONDOR_CONFIG` environment variable, `/etc/condor/condor_config`, or `~condor/condor_config`
- › All settings can be in this one file
- › Might want to share between all machines (NFS, automated copies, Wallaby, etc)

# Other Configuration Files

## > LOCAL\_CONFIG\_FILE setting

- Comma separated, processed in order

```
LOCAL_CONFIG_FILE = \  
/var/condor/config.local, \  
/var/condor/policy.local, \  
/shared/condor/config.$(HOSTNAME), \  
/shared/condor/config.$(OPSYS)
```

# Configuration File Syntax

```
# I'm a comment!
```

```
CREATE_CORE_FILES=TRUE
```

```
MAX_JOBS_RUNNING = 50
```

```
# HTCondor ignores case:
```

```
log=/var/log/condor
```

```
# Long entries:
```

```
collector_host=condor.cs.wisc.edu, \  
secondary.cs.wisc.edu
```



# Configuration File Macros

- › You reference other macros (settings) with:
  - **A** = \$(B)
  - **SCHEDD** = \$(SBIN) /condor\_schedd
- › Can create additional macros for organizational purposes

# Configuration File Macros

- › Can append to macros:

**A**=abc

**A**=\$ (A) , def

- › Don't let macros recursively define each other!

**A**=\$ (B)

**B**=\$ (A)

# Configuration File Macros

- › Later macros in a file overwrite earlier ones
  - B will evaluate to 2:

**A=1**

**B=\$ (A)**

**A=2**

# Macros and Expressions Gotcha

- › These are simple replacement macros
- › Put parentheses around expressions

**TEN**=5+5

**HUNDRED**=\$ (TEN) \* \$ (TEN)

- HUNDRED becomes 5+5\*5+5 or 35!

**TEN**=(5+5)

**HUNDRED**=( \$ (TEN) \* \$ (TEN) )

- ((5+5)\*(5+5)) = 100



# ***Security, briefly***

"Padlock" by Peter Ford © 2005

Licensed under the Creative Commons Attribution 2.0 license

<http://www.flickr.com/photos/peterf/72583027/>

<http://www.webcitation.org/5XliBcsUg>

# HTCondor Security

- › Strong authentication of users and daemons
- › Encryption over the network
- › Integrity checking over the network



"locks-masterlocks.jpg" by Brian De Smet, © 2005  
Used with permission.

<http://www.fief.org/sysadmin/blosxom.cgi/2005/07/21#locks>

# Minimal Security Settings

› You *must* set **ALLOW\_WRITE**, or nothing works

› Simplest setting:

**ALLOW\_WRITE=\***

- *Extremely insecure!*

› A bit better:

**ALLOW\_WRITE= \**  
**\*.cs.wisc.edu**

“Bank Security Guard” by “Brad & Sabrina” © 2006  
Licensed under the Creative Commons Attribution 2.0 license

<http://www.flickr.com/photos/madaboutshanghai/184665954/> <http://www.webcitation.org/5XlhUAFuY>



# More on Security

- › Zach's talk, next!
- › Chapter 3.6, "Security," in the HTCondor Manual
- › [htcondor-admin@cs.wisc.edu](mailto:htcondor-admin@cs.wisc.edu)



"Zach Miller" by Alan De Smet





# *Policy*

“Don't even think about it” by Kat “tyger\_lyllie” © 2005  
Licensed under the Creative Commons Attribution 2.0 license  
[http://www.flickr.com/photos/tyger\\_lyllie/59207292/](http://www.flickr.com/photos/tyger_lyllie/59207292/)  
<http://www.webcitation.org/5Xlh5mYGS>

# Policy

- › Who gets to run jobs, when?

# Policy Expressions

- › Specified in `condor_config`
  - Ends up slot ClassAd
- › Policy evaluates both a slot ClassAd and a job ClassAd together
  - Policy can reference items in either ClassAd (See manual for list)
- › Can reference `condor_config` macros:  
`$(MACRONAME)`

# Slots vs Machines

- › Machine – An individual computer, managed by one startd
- › Slot – A place to run a job, managed by one starter. A machine may have many slots
- › The start advertises each slot
  - The ClassAd is a “Machine” ad for historical reasons

# Slot Policy Expressions

- › **START**
- › **RANK**
- › **SUSPEND**
- › **CONTINUE**
- › **PREEMPT**
- › **KILL**

# START

- › START is the primary policy
- › When FALSE the slot enters the Owner state and will not run jobs
- › Acts as the Requirements expression for the slot, the job must satisfy START
  - Can reference job ClassAd values including Owner and ImageSize

# RANK

- › Indicates which jobs a slot prefers
  - Jobs can also specify a rank
- › Floating point number
  - Larger numbers are higher ranked
  - Typically evaluate attributes in the Job ClassAd
  - Typically use + instead of &&

# RANK

- › Often used to give priority to owner of a particular group of machines
- › Claimed slots still advertise looking for higher ranked job to preempt the current job



# SUSPEND and CONTINUE

- › When SUSPEND becomes true, the job is suspended
- › When CONTINUE becomes true a suspended job is released



# PREEMPT and KILL

- › When PREEMPT becomes true, the job will be politely shut down
  - Vanilla universe jobs get SIGTERM
    - Or user requested signal
  - Standard universe jobs checkpoint
- › When KILL becomes true, the job is SIGKILLed
  - Checkpointing is aborted if started

# Minimal Settings

› Always runs jobs

**START** = True

**RANK** =

**SUSPEND** = False

**CONTINUE** = True

**PREEMPT** = False

**KILL** = False



"Lonely at the top" by Guyon Moree ("gumuz") © 2005

Licensed under the Creative Commons Attribution 2.0 license

<http://www.flickr.com/photos/gumuz/7340411/> <http://www.webcitation.org/5X1h8s0Kl>



# Policy Configuration

- › I am adding nodes to the Cluster... *but the Chemistry Department has priority on these nodes*

# New Settings for the Chemistry nodes

## › Prefer Chemistry jobs

**START** = True

**RANK** = *Department == "Chemistry"*

**SUSPEND** = False

**CONTINUE** = True

**PREEMPT** = False

**KILL** = False

# Submit file with Custom Attribute

- › Prefix an entry with “+” to add to job

ClassAd

`Executable = charm-run`

`Universe = standard`

`+Department = "Chemistry"`

`queue`

# What if “Department” not specified?

**START** = True

**RANK** = *Department* *=?=* "Chemistry"

**SUSPEND** = False

**CONTINUE** = True

**PREEMPT** = False

**KILL** = False

# More Complex RANK

- › Give the machine's owners (adesmet and roy) highest priority, followed by the Chemistry department, followed by the Physics department, followed by everyone else.
  - Can use automatic **Owner** attribute in job attribute to identify adesmet and roy



# More Complex RANK

```
IsOwner = (Owner == "adesmet" \  
  || Owner == "roy")  
IsChem = (Department == "Chemistry")  
IsPhys = (Department == "Physics")  
RANK = $(IsOwner)*20 + $(IsChem)*10 \  
  + $(IsPhys)
```



# Policy Configuration

- › I have an unhealthy fixation with PBS so... *kill jobs after 12 hours, except Physics jobs get 24 hours.*

# Useful Attributes

## > CurrentTime

- Current time, in Unix epoch time (seconds since midnight Jan 1, 1970)

## > EnteredCurrentActivity

- When did HTCondor enter the current activity, in Unix epoch time

# Configuration

```
ActivityTimer = \  
    (CurrentTime - EnteredCurrentActivity)  
HOUR = (60*60)  
HALFDAY = ($ (HOUR) *12)  
FULLDAY = ($ (HOUR) *24)  
PREEMPT = \  
    ($ (IsPhys) && ($ (ActivityTimer) > $FULLDAY)) \  
    || \  
    (!$ (IsPhys) && ($ (ActivityTimer) > $HALFDAY))  
KILL = $ (PREEMPT)
```



# Policy Configuration

- › The cluster is okay, but... *HTCondor can only use the desktops when they would otherwise be idle*

# Defining Idle

- › One possible definition:
  - No keyboard or mouse activity for 5 minutes
  - Load average below 0.3

# Desktops should

- › **START** jobs when the machine becomes idle
- › **SUSPEND** jobs as soon as activity is detected
- › **PREEMPT** jobs if the activity continues for 5 minutes or more
- › **KILL** jobs if they take more than 5 minutes to preempt

# Useful Attributes

## > LoadAvg

- Current load average

## > CondorLoadAvg

- Current load average generated by HTCondor

## > KeyboardIdle

- Seconds since last keyboard or mouse activity



# Macros in Configuration Files

```
NonCondorLoadAvg = (LoadAvg - CondorLoadAvg)
BgndLoad = 0.3
CPU_Busy = ($(NonCondorLoadAvg) >= $(BgndLoad))
CPU_Idle = (!$ (CPU_Busy))
KeyboardBusy = (KeyboardIdle < 10)
KeyboardIsIdle = (KeyboardIdle > 300)
MachineBusy = ($(CPU_Busy) || $(KeyboardBusy))
```

# Desktop Machine Policy

**START** = \$(CPU\_Idle) && \$(KeyboardIsIdle)

**SUSPEND** = \$(MachineBusy)

**CONTINUE** = \$(CPU\_Idle) && KeyboardIdle > 120

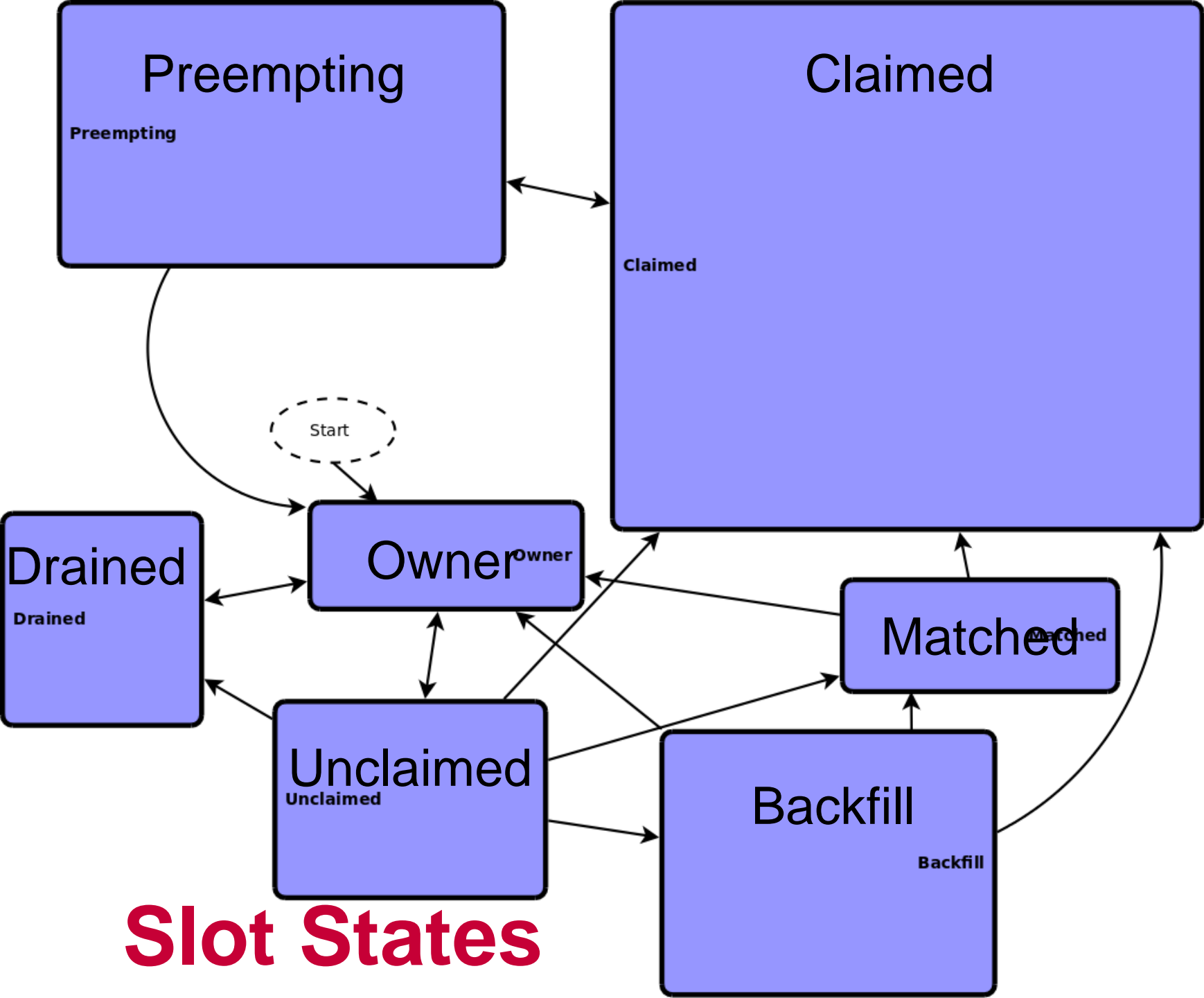
**PREEMPT** = (Activity == "Suspended") && \  
\$(ActivityTimer) > 300

**KILL** = \$(ActivityTimer) > 300

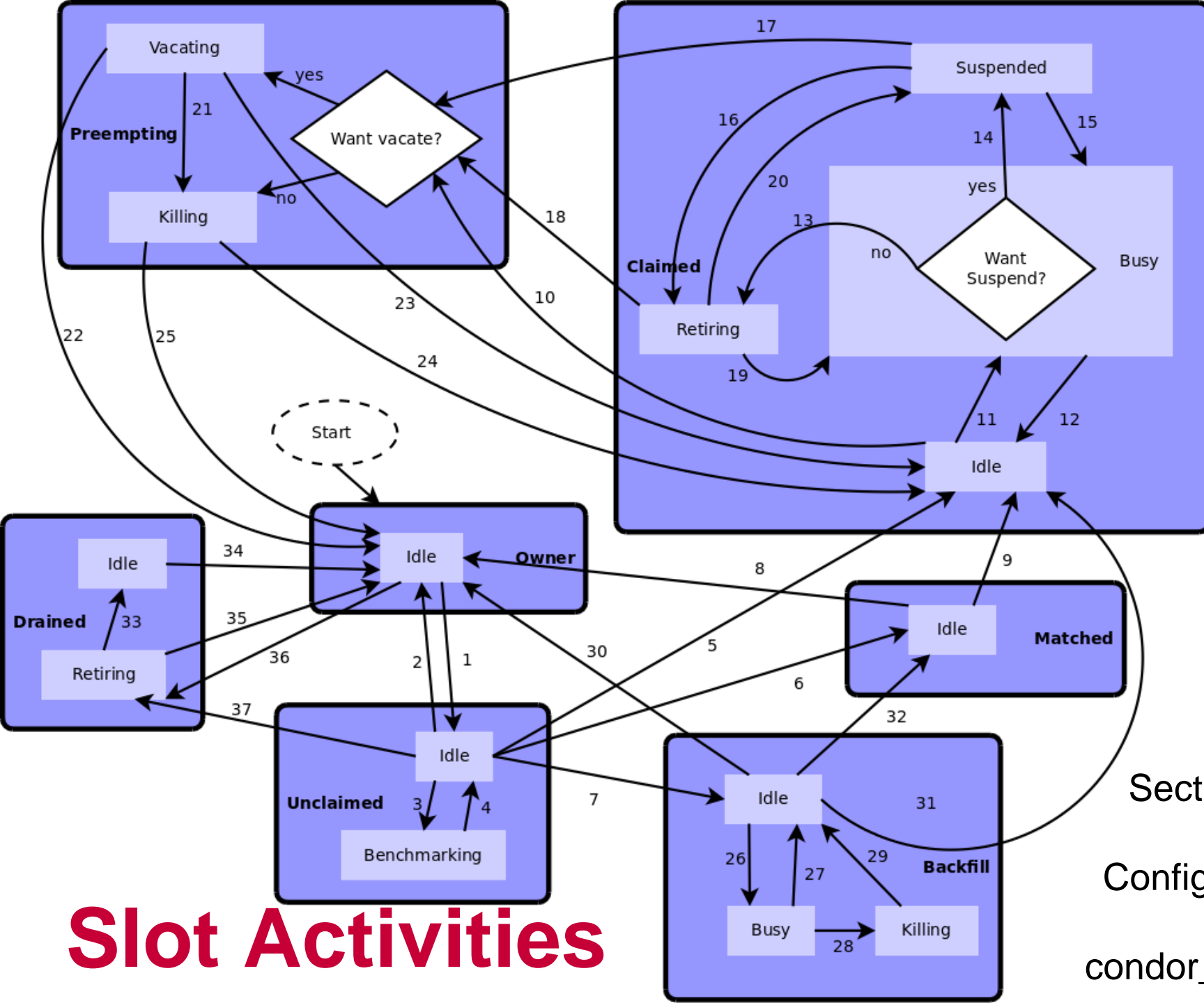
# ***Mission Accomplished***

Smiles and  
kittens for  
everyone!





# Slot States



# Slot Activities

Section 3.5:  
Policy  
Configuration  
for the  
condor\_startd)

# Custom Slot Attributes

- › Can add attributes to a slot's ClassAd, typically done in the local configuration file

**INSTRUCTIONAL=TRUE**

**NETWORK\_SPEED=1000**

**STARTD\_EXPRS=INSTRUCTIONAL,  
NETWORK\_SPEED**

# Custom Slot Attributes

- › Jobs can now specify Rank and Requirements using new attributes:  
`Requirements = INSTRUCTIONAL!=TRUE`  
`Rank = NETWORK_SPEED`
- › Dynamic attributes are available; see `STARTD_CRON_*` settings in the manual

# Further Machine Policy Information

- › For further information, see section 3.5 “Policy Configuration for the condor\_startd” in the HTCondor manual
- › htcondor-users mailing list  
<http://research.cs.wisc.edu/htcondor/mail-lists/>
- › htcondor-admin@cs.wisc.edu



# *Priorities*



# Job Priority

- › Set with `condor_prio`
- › Users can set priority of their own jobs
- › Integers, larger numbers are higher priority
- › Only impacts order between jobs for a single user on a single schedd
- › A tool for users to sort their own jobs

# User Priority

- › Determines allocation of machines to waiting users
- › View with `condor_userprio`
- › Inversely related to machines allocated (lower is better priority)
  - A user with priority of 10 will be able to claim twice as many machines as a user with priority 20

# User Priority

- › Effective User Priority is determined by multiplying two components
  - Real Priority
  - Priority Factor

# Real Priority

- › Based on actual usage
- › Defaults to 0.5
- › Approaches actual number of machines used over time
  - Configuration setting **PRIORITY\_HALFLIFE**

# Priority Factor

- › Assigned by administrator
  - Set with `condor_userprio`
- › Defaults to 1 (`DEFAULT_PRIO_FACTOR`)

# Negotiator Policy Expressions

- › **PREEMPTION\_REQUIREMENTS** and **PREEMPTION\_RANK**
- › Evaluated when `condor_negotiator` considers replacing a lower priority job with a higher priority job
- › Completely unrelated to the **PREEMPT** expression

# PREEMPTION\_REQUIREMENTS

- › If false will not preempt machine
  - Typically used to avoid pool thrashing
  - Typically use:
    - `RemoteUserPrio` – Priority of user of currently running job (higher is worse)
    - `SubmittorPrio` – Priority of user of higher priority idle job (higher is worse)
- › **PREEMPTION\_REQUIREMENTS=FALSE**



# PREEMPTION\_REQUIREMENTS

- › Only replace jobs running for at least one hour and 20% lower priority

```
StateTimer = \  
    (CurrentTime - EnteredCurrentState)  
HOUR = (60*60)  
PREEMPTION_REQUIREMENTS = \  
    $(StateTimer) > (1 * $(HOUR)) \  
    && RemoteUserPrio > SubmitterPrio * 1.2
```

# PREEMPTION\_RANK

- › Picks which already claimed machine to reclaim
- › Strongly prefer preempting jobs with a large (bad) priority and a small image size

```
PREEMPTION_RANK = \  
(RemoteUserPrio * 1000000) \  
- ImageSize
```

# Accounting Groups

- › Manage priorities across groups of users and jobs
- › Can guarantee minimum numbers of computers for groups (quotas)
- › Supports hierarchies
- › Anyone can join any group

# *Tools*



# condor\_config\_val

› Find current configuration values

```
% condor_config_val MASTER_LOG  
/var/condor/logs/MasterLog  
% cd `condor_config_val LOG`
```

# condor\_config\_val -v

› Can identify source

```
% condor_config_val -v CONDOR_HOST
```

```
CONDOR_HOST: condor.cs.wisc.edu
```

```
Defined in
```

```
`/etc/condor_config.hosts', line 6
```

# condor\_config\_val -config

> What configuration files are being used?

```
% condor_config_val -config
```

Config source:

```
    /var/home/condor/condor_config
```

Local config sources:

```
    /unsup/condor/etc/condor_config.hosts
```

```
    /unsup/condor/etc/condor_config.global
```

```
    /unsup/condor/etc/condor_config.policy
```

```
    /unsup/condor-test/etc/hosts/puffin.local
```

# condor\_fetchlog

- › Retrieve logs remotely

```
condor_fetchlog beak.cs.wisc.edu  
Master
```



# Querying daemons

## `condor_status`

- › Queries the collector for information about daemons in your pool
- › Defaults to finding `condor_startds`
- › `condor_status -schedd` summarizes all job queues
- › `condor_status -master` returns list of all `condor_masters`

# condor\_status

- › -long displays the full ClassAd
- › Optionally specify a machine name to limit results to a single host

```
condor_status -l  
node4.cs.wisc.edu
```

# condor\_status -constraint

- › Only return ClassAds that match an expression you specify
- › Show me idle machines with 1GB or more memory

- `condor_status -constraint 'Memory >= 1024 && Activity == "Idle"'`

# condor\_status

## -format

- › Controls format of output
- › Useful for writing scripts
- › Uses C printf style formats
  - One field per argument

"slanting" by Stefano Mortellaro ("fazen") © 2005  
Licensed under the Creative Commons Attribution 2.0 license  
<http://www.flickr.com/photos/fazen/17200735/>  
<http://www.webcitation.org/5XlhNWC7Y>



# condor\_status -format

› Census of systems in your pool:

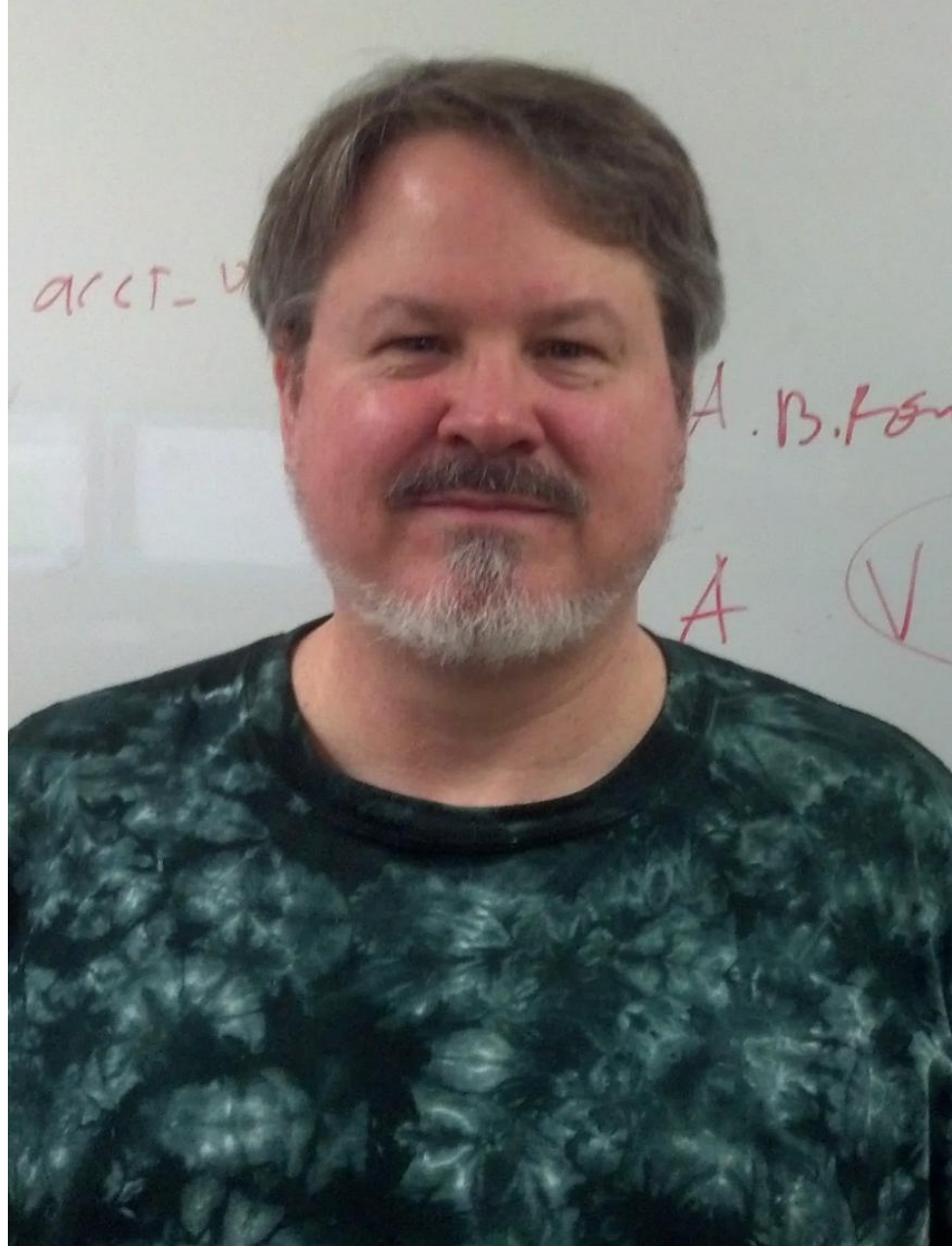
```
% condor_status -format '%s '
Arch -format '%s\n' OpSys | sort
| uniq -c
    797 INTEL LINUX
    118 INTEL WINDOWS
    108 X86_64 LINUX
     6 X86_64 OSX
```

# Examining Queues `condor_q`

- › View the job queue
- › The `-long` option is useful to see the entire ClassAd for a given job
- › supports `-constraint` and `-format`
- › Can view job queues on remote machines with the `-name` option

# condor\_q -analyze and -better-analyze

- › **condor\_q** will try to figure out why the job isn't running
- › Good at determining that no machine matches the job Requirements expressions
- › See John's talk this afternoon!



# *Log Files*





# HTCondor's Log Files

- › HTCondor maintains one log file per daemon
- › Can increase verbosity of logs on a per daemon basis
  - SHADOW\_DEBUG, SCHEDD\_DEBUG, and others
  - Space separated list

# Useful Debug Levels

- › **D\_FULLDEBUG** dramatically increases information logged
  - Does not include other debug levels!
- › **D\_COMMAND** adds information about about commands received

```
SHADOW_DEBUG = \  
    D_FULLDEBUG D_COMMAND
```

# Log Rotation

- › Log files are automatically rolled over when a size limit is reached
  - Only one old version is kept
  - Defaults to 1,000,000 bytes
  - Rolls over quickly with `D_FULLDEBUG`
  - `MAX_*_LOG`, one setting per daemon
    - `MAX_SHADOW_LOG`, `MAX_SCHEDD_LOG`, and others

# HTCondor's Log Files

- › Many log files entries primarily useful to HTCondor developers
  - Especially if `D_FULLDEBUG` is on
  - Minor errors are often logged but corrected
  - Take them with a grain of salt
  - `htcondor-admin@cs.wisc.edu`

# *Debugging Jobs*



# Debugging Jobs: condor\_q

- › Examine the job with `condor_q`
  - especially `-analyze`, `-better-analyze`, `-machine`, and `-long`
  - Compare with `condor_status -long` for a machine you expected to match
  - Did I mention John's talk?

# Debugging Jobs: User Log

- › Examine the job's user log
  - Can find with:  

```
condor_q -format '%s\n' UserLog 17.0
```
  - Set with “log” in the submit file
  - You can set **EVENT\_LOG** to get a unified log for all jobs under a schedd
- › Contains the life history of the job
- › Often contains details on problems

# Debugging Jobs: ShadowLog

- › Examine **ShadowLog** on the submit machine
  - Note any machines the job tried to execute on
  - There is often an “ERROR” entry that can give a good indication of what failed



# Debugging Jobs: Matching Problems

- › No **ShadowLog** entries? Possible problem matching the job.
  - Examine **ScheddLog** on the submit machine
  - Examine **NegotiatorLog** on the central manager

# Debugging Jobs: Remote Problems

- › **ShadowLog** entries suggest an error but aren't specific?
  - Examine **StartLog** and **StarterLog** on the execute machine

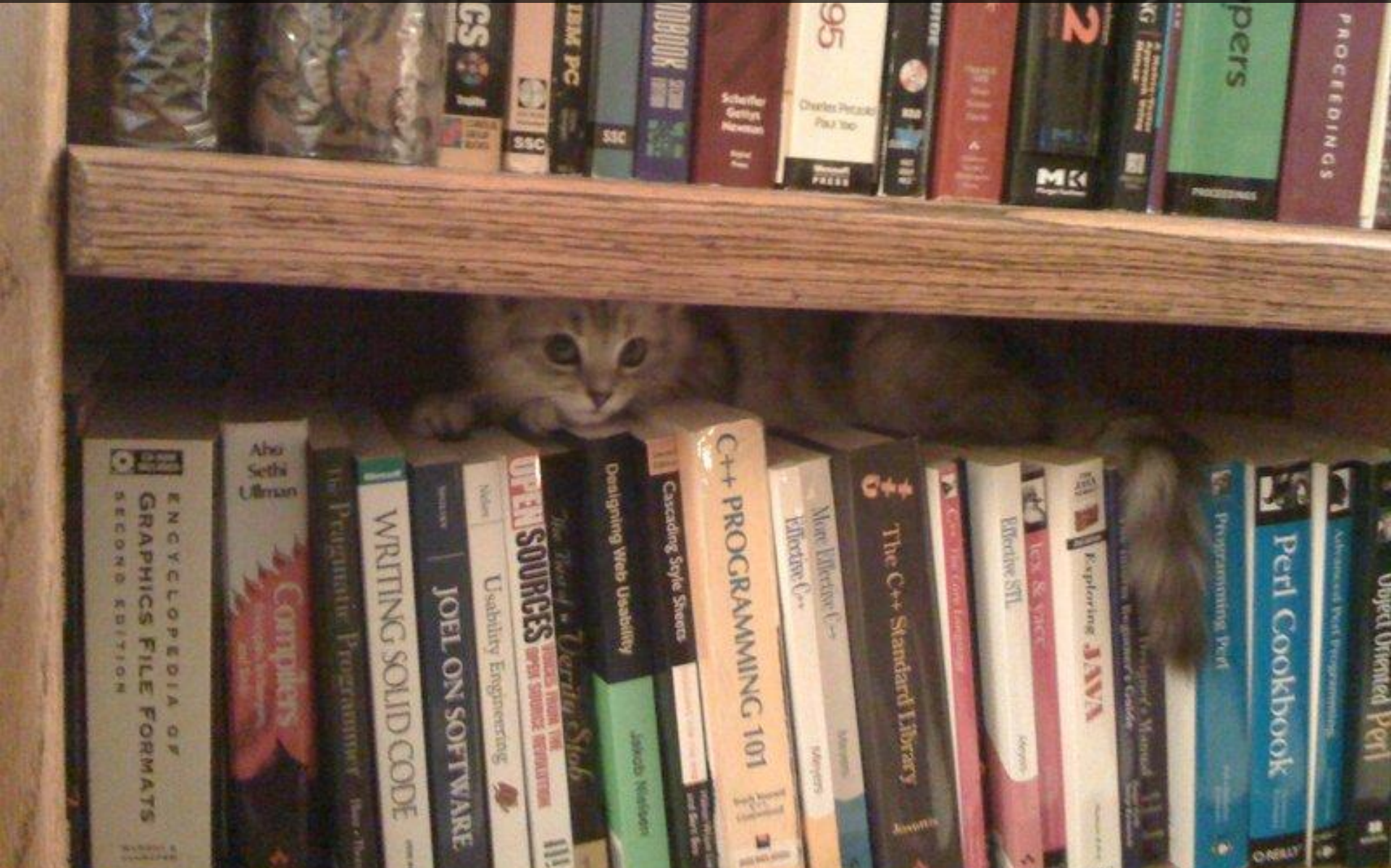
# Debugging Jobs: Reading Log Files

- › Condor logs will note the job ID each entry is for
  - Useful if multiple jobs are being processed simultaneously
  - grepping for the job ID will make it easy to find relevant entries

# Debugging Jobs: What Next?

- › If necessary add “**D\_FULLDEBUG**  
**D\_COMMAND**” to **DEBUG\_DAEMONNAME**  
setting for additional log information
- › Increase **MAX\_DAEMONNAME\_LOG** if logs  
are rolling over too quickly
- › If all else fails, email us
  - [htcondor-admin@cs.wisc.edu](mailto:htcondor-admin@cs.wisc.edu)

# *More Information*



# More Information

- › Staff here at HTCondor Week
- › HTCondor Manual
- › htcondor-users mailing list  
<http://research.cs.wisc.edu/htcondor/mail-lists/>
- › htcondor-admin  
[htcondor-admin@cs.wisc.edu](mailto:htcondor-admin@cs.wisc.edu)



***Thank You!***



**Any questions?**