Administrating Condor

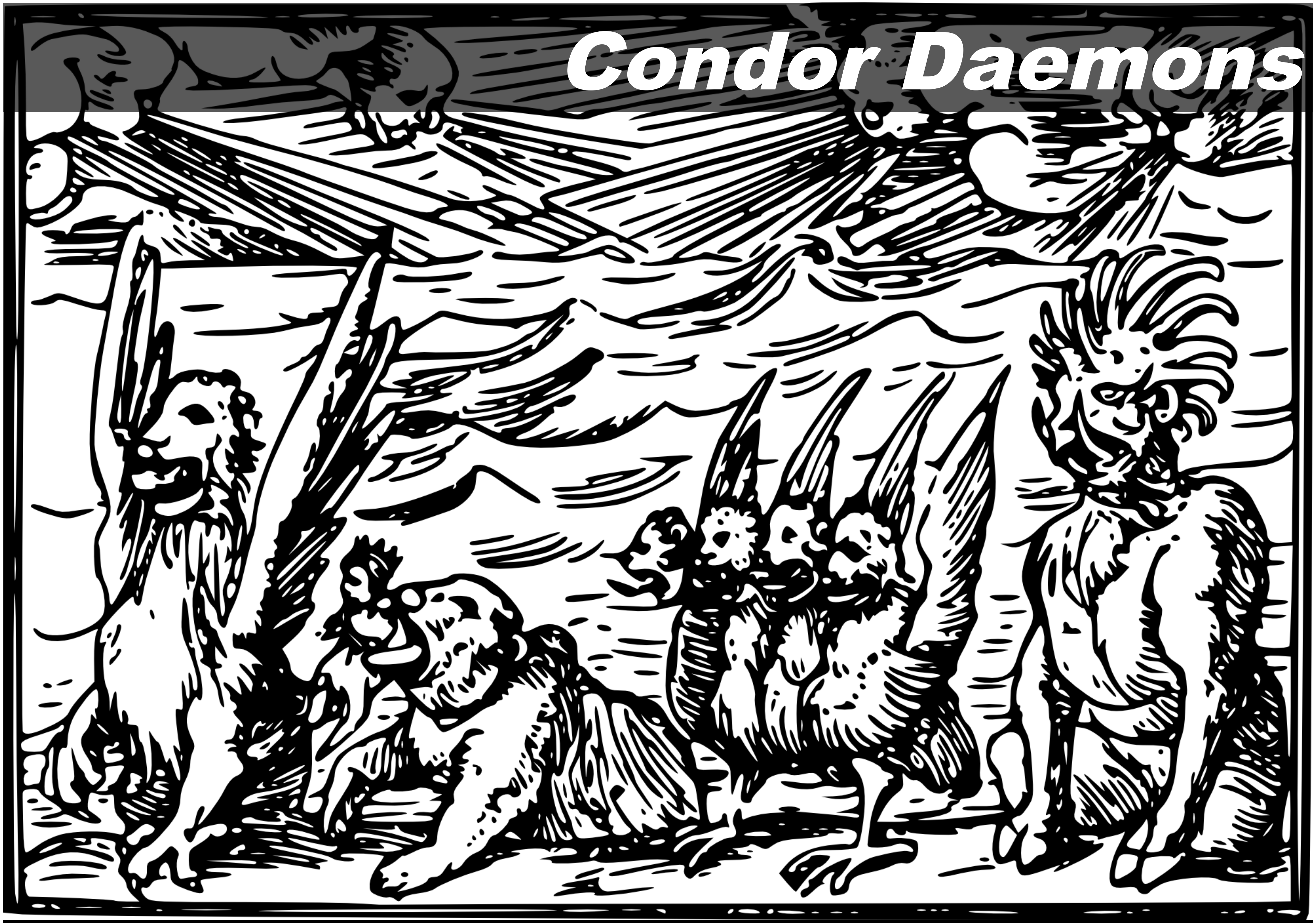
Alan De Smet
Condor Project
adesmet@cs.wisc.edu
http://www.cs.wisc.edu/condor

“Condor - Colca Canyon-” by “Raultimate” © 2006 Licensed under the Creative Commons Attribution 2.0 license.
http://www.flickr.com/photos/7428244@N06/427485954/ http://www.webcitation.org/5g6wqrJPx

The next 90 minutes…

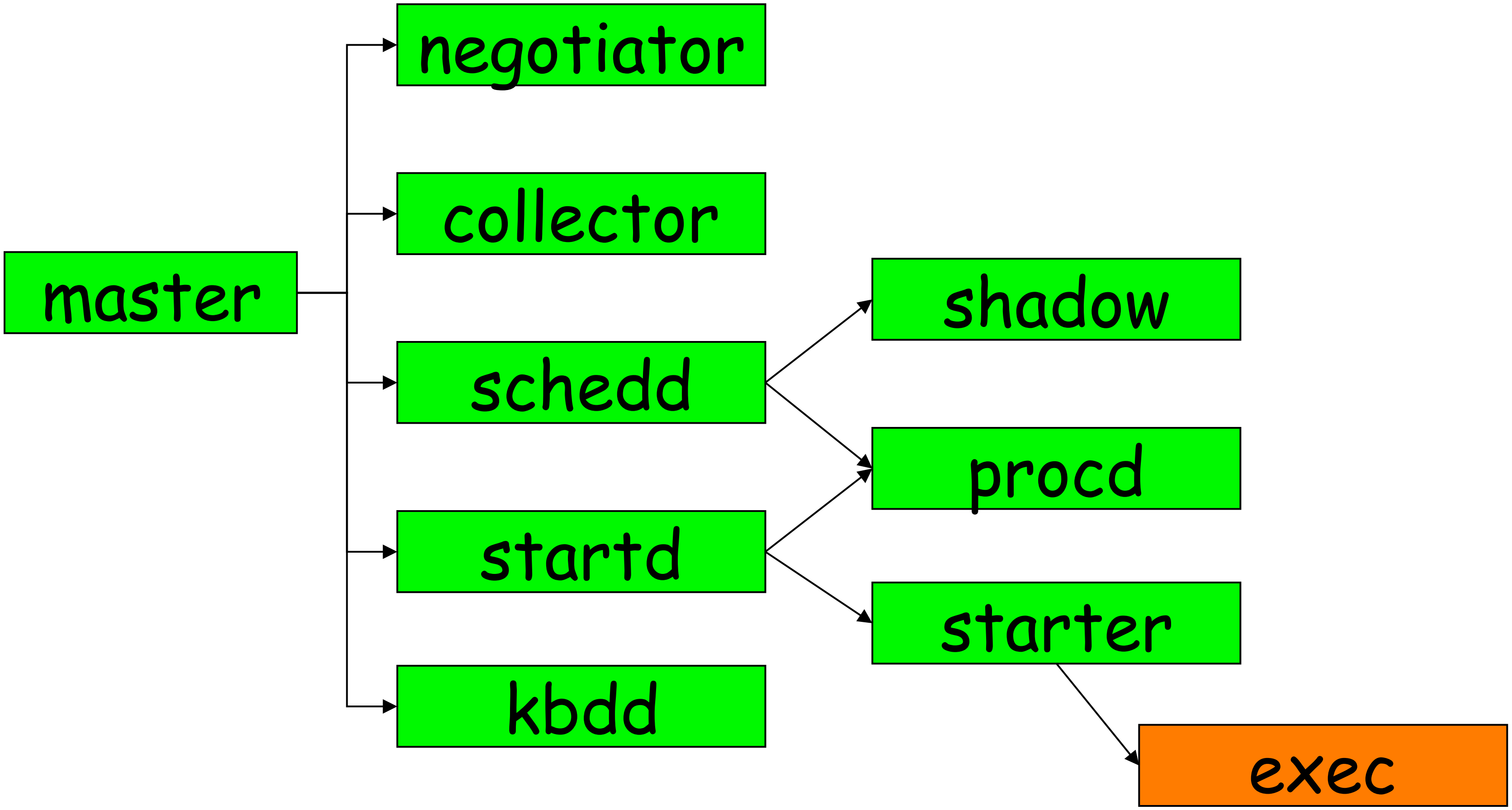
- Condor Daemons
  - Job Startup
- Configuration Files
- ClassAds
- Policy Expressions
  - Startd (Machine)
  - Negotiator
- Priorities
- Security
- Useful Tools
- Log Files
- Debugging Jobs

Condor Daemons



Title unknown, by Hans Holbein the Younger, from *Historiarum Veteris Testamenti icones*, 1543

Condor Daemons



condor_master

- You start it, it starts up the other Condor daemons
- If a daemon exits unexpectedly, restarts deamon and emails administrator
- If a daemon binary is updated (timestamp changed), restarts the daemon

`condor_master`

- Provides access to many remote administration commands:
  - `condor_reconfig`, `condor_restart`, `condor_off`, `condor_on`, etc.
- Default server for many other commands:
  - `condor_config_val`, etc.

condor_master

- Periodically runs **condor_preen** to clean up any files Condor might have left on the machine
  - Emails you notification of deleted files
  - Backup behavior, the other daemons clean up after themselves

`condor_procd`

- Tracks processes
- Automatically started as needed
  - No **`DAEMON_LIST`** entry necessary
  - Behind the scenes
- Part of privilege separation security enhancements

“IMG 0960” by Eva Schiffer © 2008 Used with permission http://www.digitalchangeling.com/pictures/ourCats2008/january2008/IMG_0960.html

condor_startd

- Represents a machine willing to run jobs to the Condor pool
- Run on any machine you want to run jobs on
- Enforces the wishes of the machine owner (the owner's "policy")

condor_startd

- Starts, stops, suspends jobs
- Spawns the appropriate **condor_starter**, depending on the type of job
- Provides other administrative commands (for example, **condor_vacate**)
- Aided by **condor_kbdd**

condor_starter

- Spawned by the **condor_startd**
  - Don't add to **DAEMON_LIST**
- Handles all the details of starting and managing the job
  - Transfer job's binary to execute machine
  - Send back exit status
  - Etc.

condor_starter

- One per running job
- The default configuration is willing to run one job per CPU

condor_kbdd

- Monitors physical keyboard and mouse so the **condor_startd** can make decisions based on local usage.

condor_schedd

- Represents jobs to the Condor pool
- Maintains persistent queue of jobs
  - Queue is not strictly first-in-first-out (priority based)
  - Each machine running **condor_schedd** maintains its own independent queue
- Run on any machine you want to submit jobs from

condor_schedd

- Responsible for contacting available machines and spawning waiting jobs
  - When told to by condor_negotiator
- Services most user commands:
  - condor_submit, condor_rm, condor_q

condor_shadow

- Represents job on the submit machine
- Spawned by **condor_schedd**
  - Don't add to **DAEMON_LIST**
- Services requests from standard universe jobs for remote system calls
  - including all file I/O
- Makes decisions on behalf of the job
  - for example: where to store the checkpoint file

condor_shadow Impact

- One **condor_shadow** running on submit machine for each actively running Condor job
- Minimal load on submit machine
  - Usually blocked waiting for requests from the job or doing I/O
  - Relatively small memory footprint
  - Can throttle, see **MAX_JOBS_RUNNING** and **SHADOW_RENICE_INCREMENT** in the manual

condor_exec.exe

- A running job.
- When user executable binaries are transferred to the execution side, they are renamed **condor_exec.exe**.

condor_collector

- Collects information from all other Condor daemons in the pool
- condor_collector
- Each daemon sends a periodic update called a ClassAd to the collector
  - Old ClassAds removed after a time out
- Services queries for information:
  - Queries from other Condor daemons
  - Queries from users (**condor_status**)

`condor_negotiator`

- Performs matchmaking in Condor
  - Pulls list of available machines and job queues from `condor_collector`
  - Matches jobs with available machines
  - Both the job and the machine must satisfy each other's requirements (2-way matching)
- Handles user priorities

Condor Daemons

- You only have to run the daemons for the services you need to provide
- **`DAEMON_LIST`** is a comma separated list of daemons to start
  - **`DAEMON_LIST=MASTER,SCHEDD,STARTD`**

Central Manager

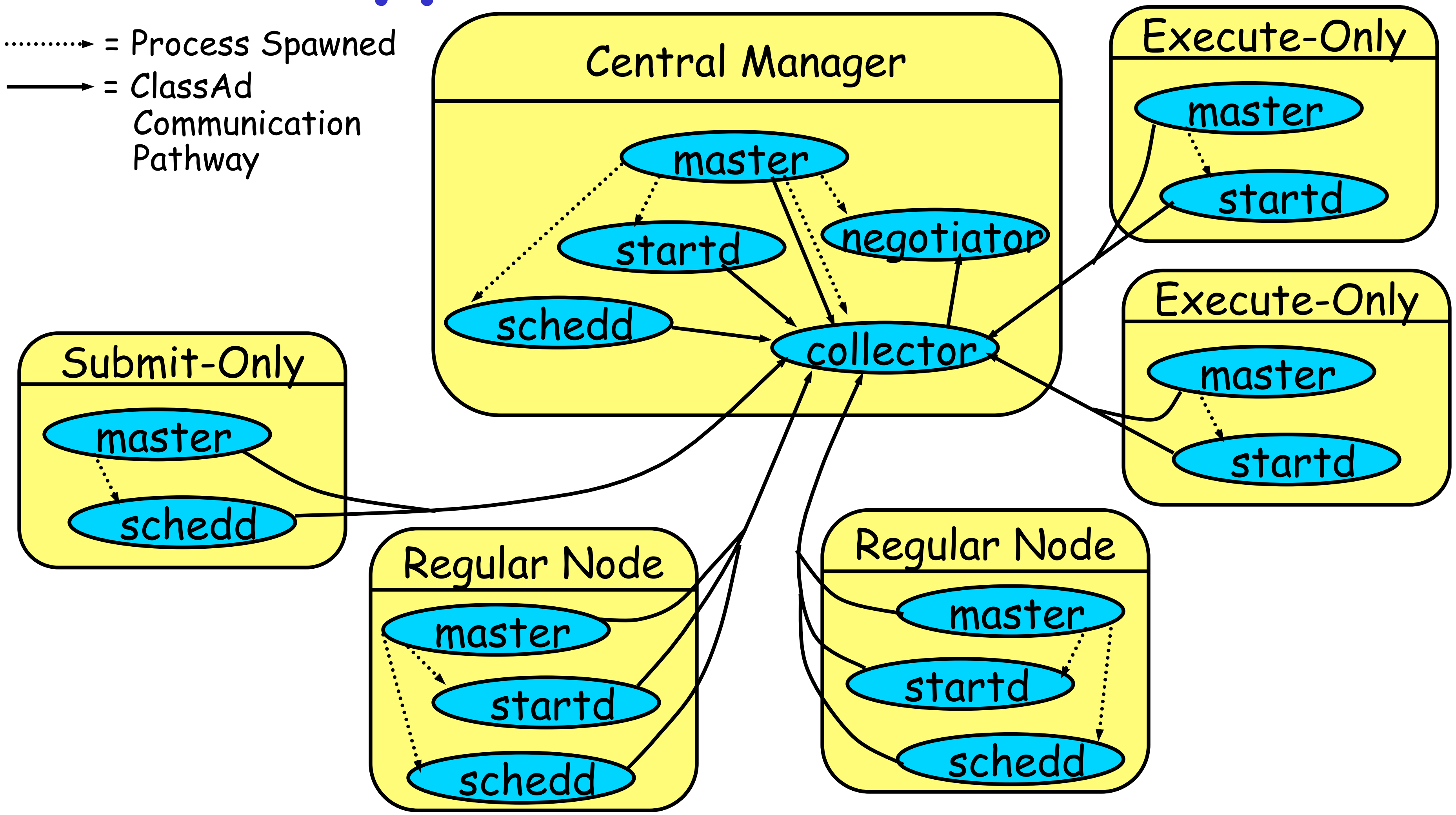
- The Central Manager is the machine running the collector and negotiator

```
DAEMON_LIST = MASTER,
  COLLECTOR, NEGOTIATOR
```

- Defines a Condor pool.

```
CONDOR_HOST =
  centralmanager.example.com
```

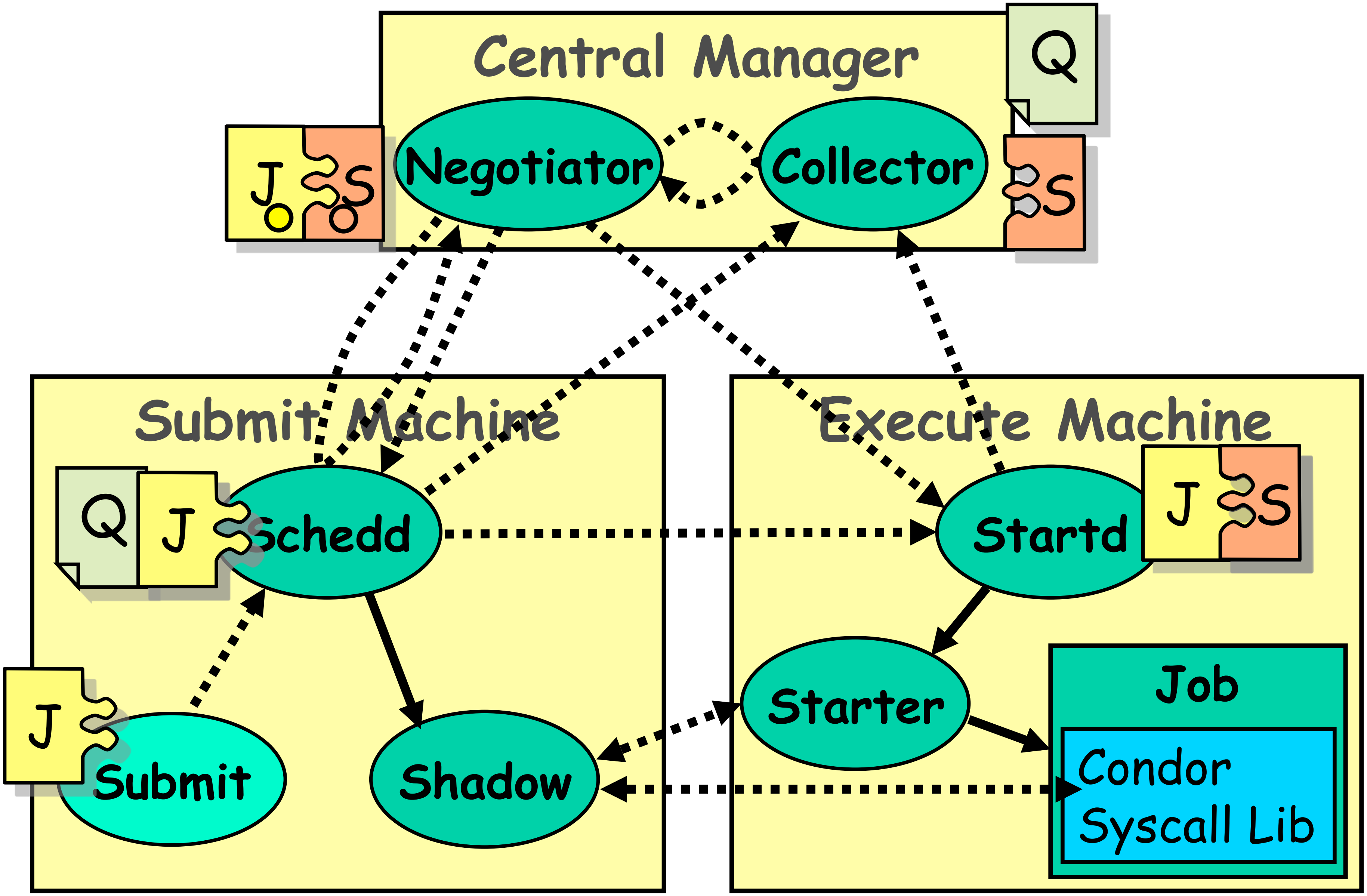
Typical Condor Pool

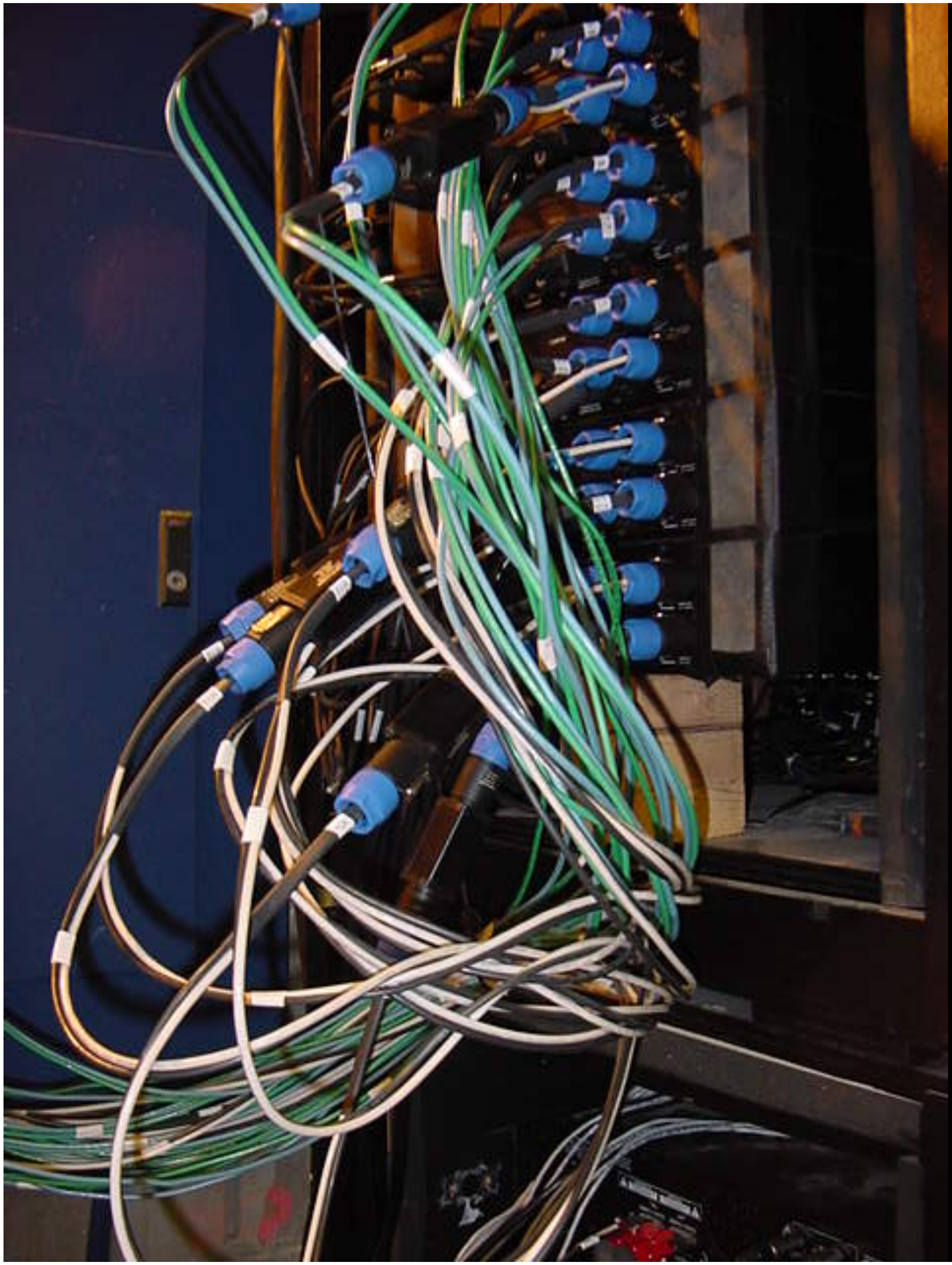


Job Startup

“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/5XIfTl6tX

Job Startup


Central Manager
Q
J
S
Negotiator
Collector
S
Submit Machine
Execute Machine
Q
J
Schedd
Startd
J
S
J
Submit
Shadow
Starter
Job
Condor
Syscall Lib



Configuration Files

“amp wiring” by “fbz_” © 2005
Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/fbz/114422787/

Global 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 file
- "Global" on assumption it's shared between machines. NFS, automated copies, etc.

Other Configuration Files

- You can configure a number of other shared configuration files:
  - Organize common settings (for example, all policy expressions)
  - Platform-specific configuration files
  - Machine specific settings
    - Local policy for a particular machine's owner
    - Different daemons to run. For example, the Central Manager

Other Configuration Files

- `LOCAL_CONFIG_FILE` macro
  - Comma separated, processed in order

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

Per-Machine Configuration Files

- Can be on local disk of each machine

```
/var/adm/condor/condor_config.local
```

- Can be in a shared directory
  - Use `$(HOSTNAME)` which expands to the machine's name

```
/shared/condor/config.$(HOSTNAME)
/shared/condor/hosts/$(HOSTNAME)/
 config.local
```

Per-Platform Configuration Files

- Use macros like $(OPSYS) which expand to the operating system

```
/shared/condor/config.$(OPSYS)
```

- $(OPSYS) will expand into entries like LINUX, WINNT51, SOLARIS28
- See “Pre-Defined Macros” in the Manual for a list of options

Configuration File Syntax

- # at start of line is a comment
  - not allowed in names, confuses Condor.
- \ at the end of line is a line-continuation
  - Both lines are treated as one big entry
  - Works in comments!

```
# This comment eats the next line \
EXAMPLE_SETTING=TRUE
```

Configuration File Macros

- Macros have the form:
  - `Attribute_Name = value`
    - Names are case insensitive
    - Values are case sensitive
- You reference other macros with:
  - `A = $(B)`
- 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

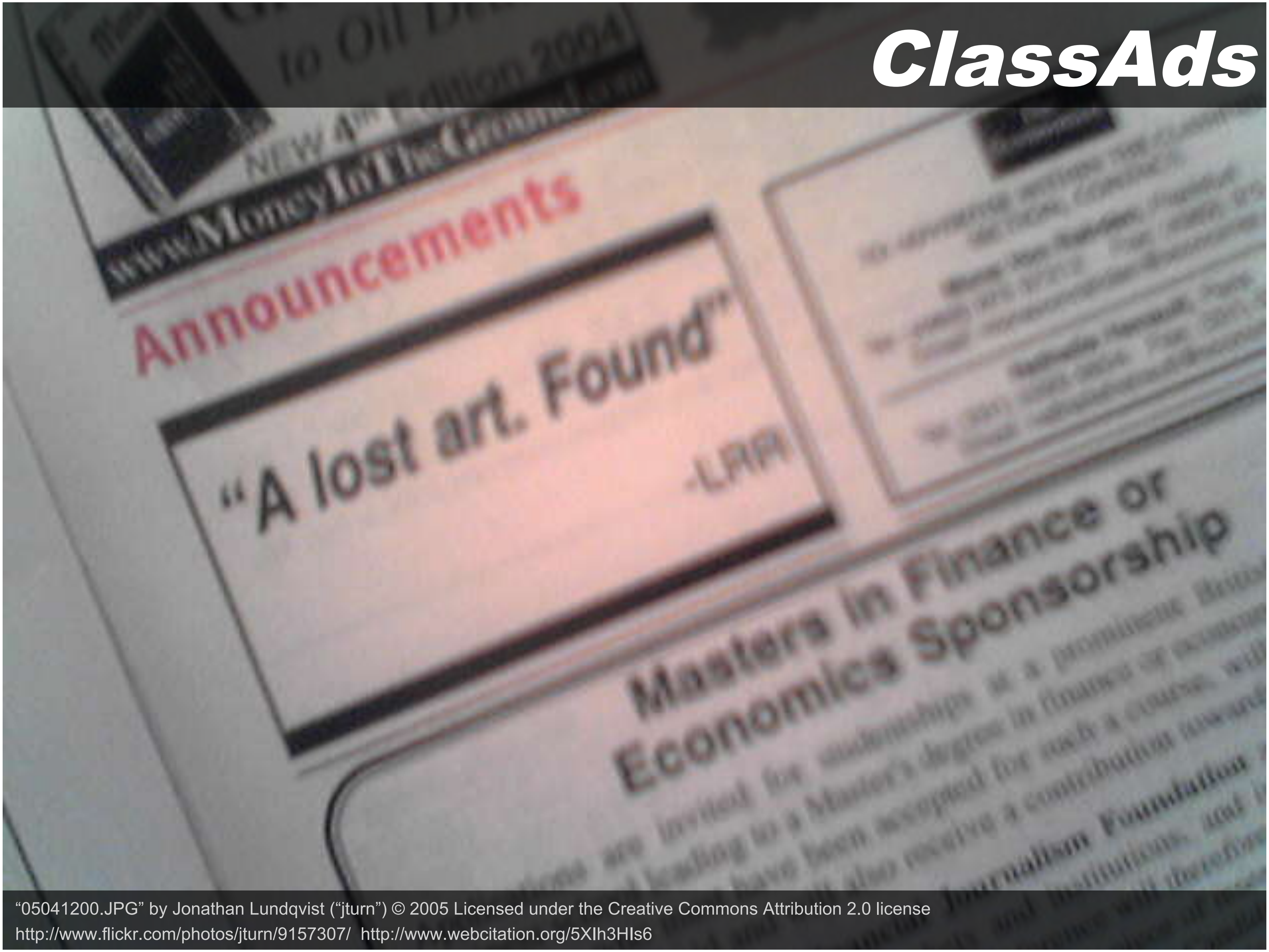
```
TEN=5+5
HUNDRED=$(TEN)*$(TEN)
```

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

```
TEN=(5+5)
HUNDRED=($(TEN)*$(TEN))
```

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

ClassAds



www.MoneyInTheGround.com

Announcements

"A lost art. Found"

-LARR

Masters in Finance or Economics Sponsorship

“05041200.JPG” by Jonathan Lundqvist (“jturn”) © 2005 Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/jturn/9157307/ http://www.webcitation.org/5XIh3HIs6

ClassAds

- "Classified Advertisements"
- Set of key-value pairs

```
MyType = "Machine"
TargetType = "Job"
Name = "slot1@puffin.cs.wisc.edu"
Rank = 0.000000
MyCurrentTime = 1271097865
IsInstructional = FALSE
```

ClassAds

- Values can be expressions

```
Price=Gallons*PerGallonCost
Gallons=9.1232
PerGallonCost=2.499
```

ClassAds

- Can be matched against each other
  - Requirements and Rank
    - MY.name – Looks for “name” in local ClassAd
    - TARGET.name – Looks for “name” in the other ClassAd
    - Name – Looks for “name” in the local ClassAd, then the other ClassAd

ClassAd matching

```
MyType = "GasPump"
Requirements = TARGET.Credit >
   (TARGET.GallonsNeeded *
  MY.PricePerGallon)
PricePerGallon = 2.99
Octane = 93
```

```
MyType = "Car"
Requirements = Octane > 87
GallonsNeeded = 9
Credit = 35.50
Rank = Octane
```

ClassAd Expressions

- Some configuration file macros specify expressions for the Machine's ClassAd
  - Notably START, RANK, SUSPEND, CONTINUE, PREEMPT, KILL
- Can contain a mixture of macros and ClassAd references

ClassAd Expressions

- +, -, *, /, <, <=,>, >=, ==, !=, &&, and || all work as expected
- TRUE==1 and FALSE==0 (guaranteed)
  - (3 == (2+1)) is identical to 1
  - (TRUE*30) is identical to 30
  - (3 == 1) is identical to 0

Special Values: UNDEFINED and ERROR

- Special values
- Passed through most operators
  - Anything == UNDEFINED is UNDEFINED
- && and || eliminate if possible.
  - UNDEFINED && FALSE is FALSE
  - UNDEFINED && TRUE is UNDEFINED

ClassAd Expressions: =?= and =!=

- =?= and =!= are similar to == and !=
- =?= tests if operands have the same type and the same value.
  - `10 == UNDEFINED` -> UNDEFINED
  - `UNDEFINED == UNDEFINED` -> UNDEFINED
  - `10 =?= UNDEFINED` -> FALSE
  - `UNDEFINED =?= UNDEFINED` -> TRUE
- =!= inverts =?=

ClassAd Functions

- ClassAds offer a variety of useful functions for string manipulation, date formatting, list management, and more.

ClassAd Expressions

- Further information: Section 4.1, “Condor's ClassAd Mechanism,” in the Condor Manual.

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.webcitation.org/5XIh5mYGS

Policy

- Allows machine owners to specify job priorities, restrict access, and implement other local policies

Policy Expressions

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

Machine (Startd) Policy Expressions

- `START`
- `RANK`
- `SUSPEND`
- `CONTINUE`
- `PREEMPT`
- `KILL`

START

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

RANK

- Indicates which jobs a machine 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 machines 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

"DSC03753" by Eva Schiffer © 2008 Used with permission http://www.digitalchangeling.com/pictures/ourCats2008/january2008/DSC03753.html

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/5XIh8s0kI

Policy Configuration

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

“I R BIZNESS CAT” by “VMOS” © 2007
Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/vmos/2078227291/ http://www.webcitation.org/5XIff1deZ

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 =!= UNDEFINED
  && 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 =!= UNDEFINED
  && Department == "Chemistry")
IsPhys =(Department =!= UNDEFINED
  && Department == "Physics")
RANK = $(IsOwner)*20 + $(IsChem)
  *10 + $(IsPhys)
```

Policy Configuration

- Cluster is okay, but... *Condor can only use the desktops when they would otherwise be idle*

"I R BIZNESS CAT" by "VMOS" © 2007
Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/vmos/2078227291/ http://www.webcitation.org/5XIff1deZ

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 Condor
- **`KeyboardIdle`**
  - Seconds since last keyboard or mouse activity

Useful Attributes

- `CurrentTime`
  - Current time, in Unix epoch time (seconds since midnight Jan 1, 1970)
- `EnteredCurrentActivity`
  - When did Condor enter the current activity, in Unix epoch time

Macros in Configuration Files

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

Desktop Machine Policy

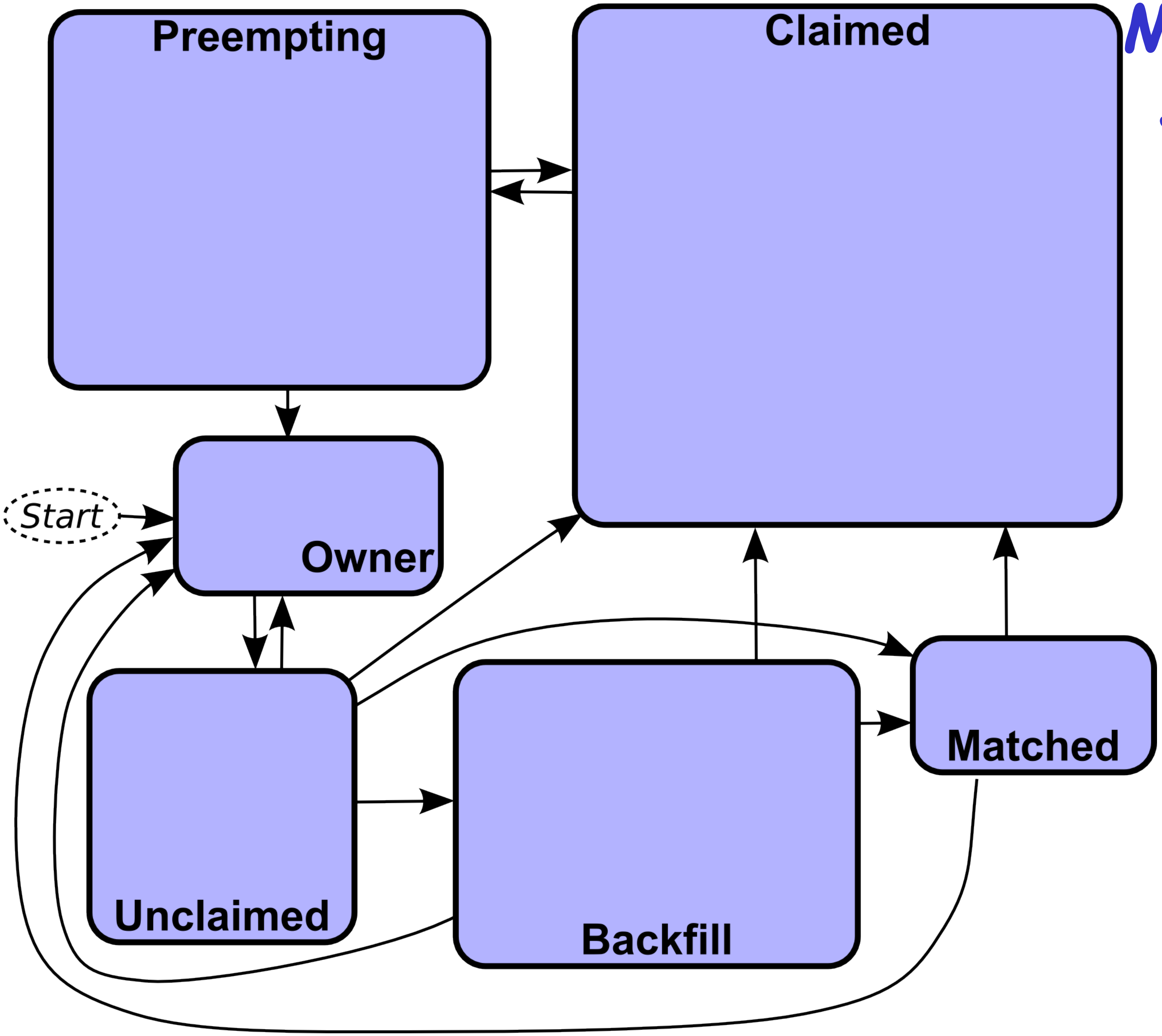
```
START = $(CPU_Idle) && $(KeyboardIsIdle)
SUSPEND   = $(MachineBusy)
CONTINUE = $(CPU_Idle) && KeyboardIdle > 120
PREEMPT   = (Activity == "Suspended") && \
            $(ActivityTimer) > 300
KILL  = $(ActivityTimer) > 300
```

Mission Accomplished

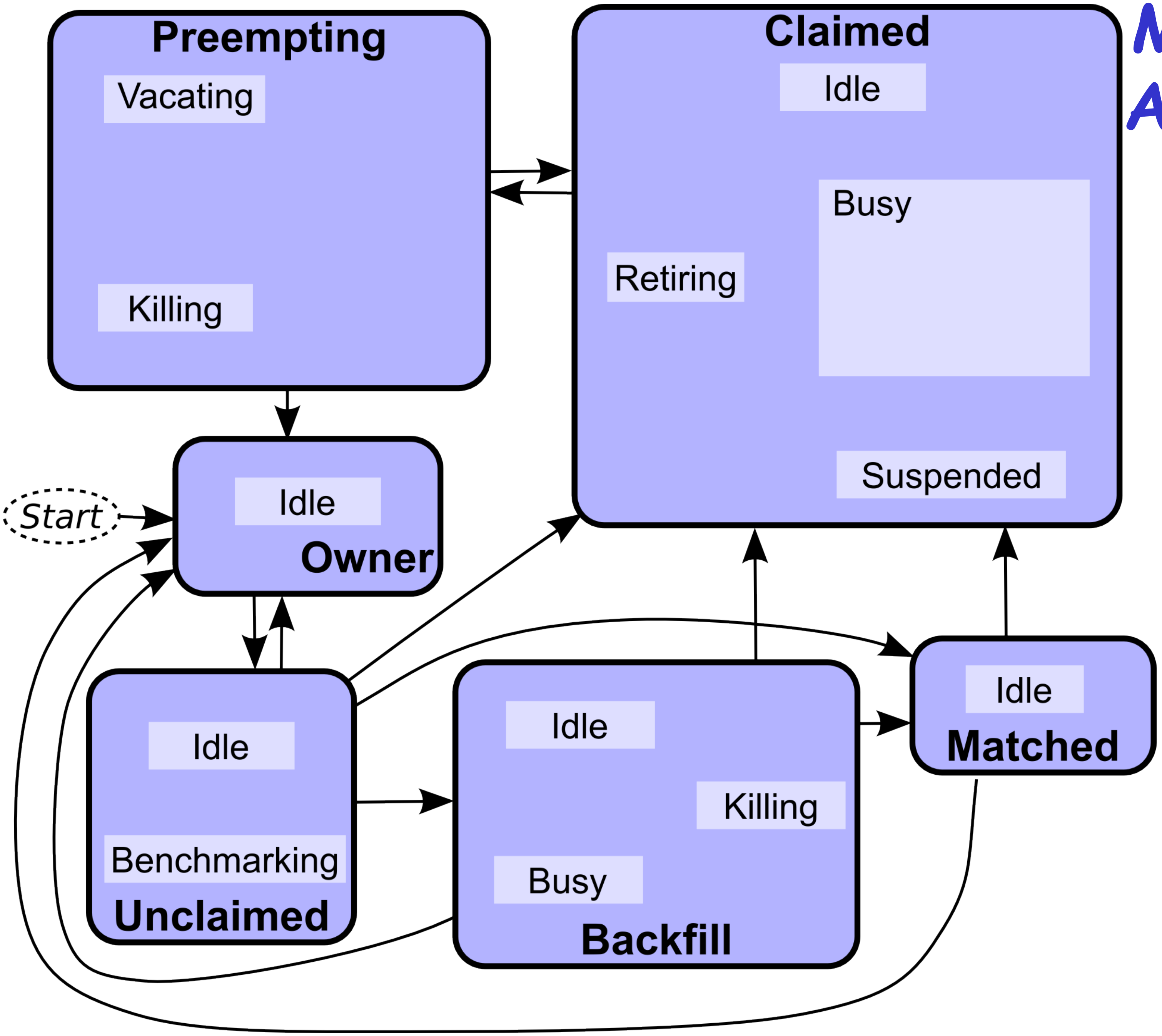
Smiles and kittens for everyone!

“Autumn and Blue Eyes” by Paul Lewis (“PJLewis”) © 2005 Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/pjlewis/46134047/ http://www.webcitation.org/5XIhBzDR2

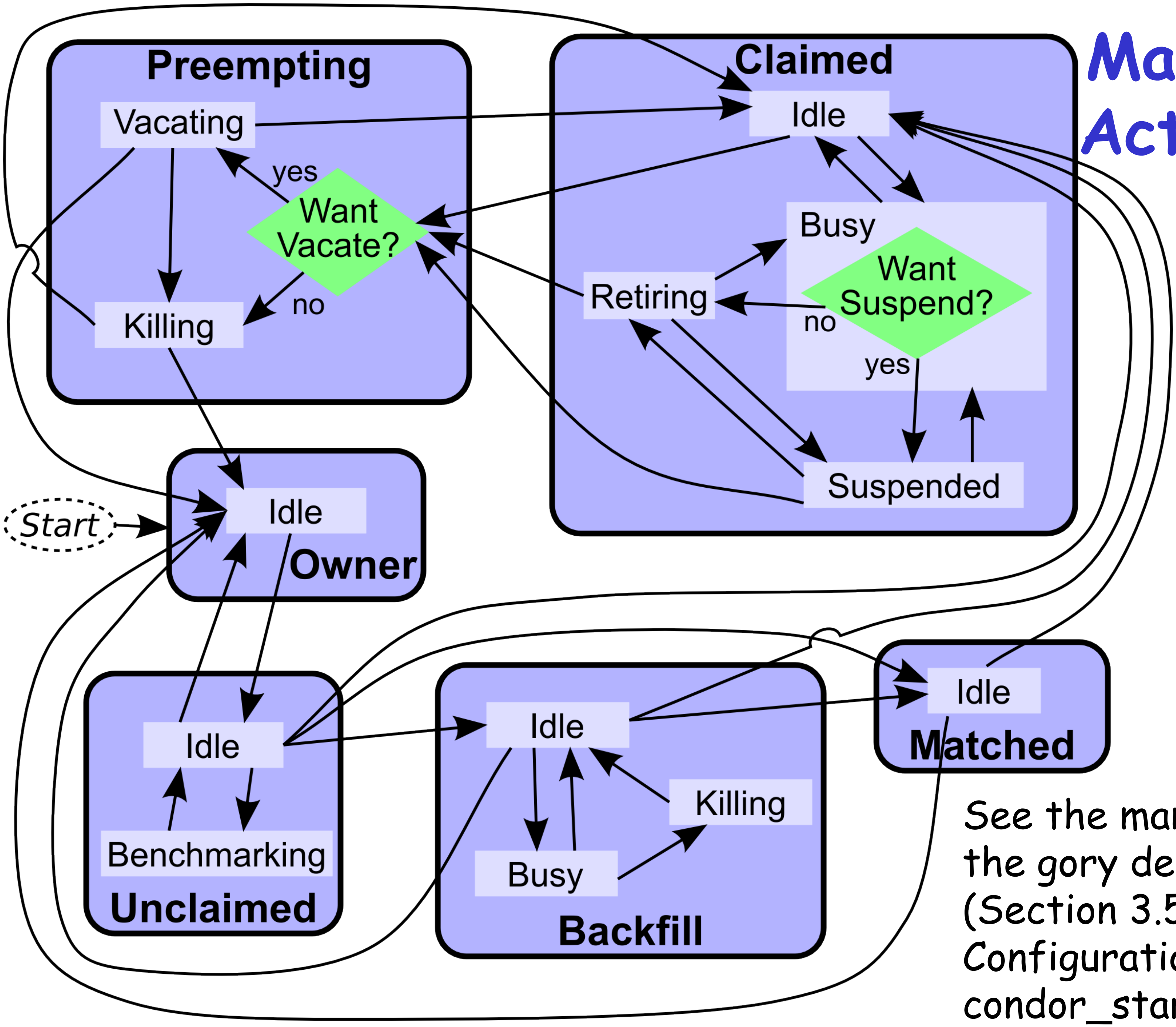
Machine States



Machine Activities



Machine Activities



See the manual for the gory details. (Section 3.5: Policy Configuration for the condor_startd)

Custom Machine Attributes

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

```
INSTRUCTIONAL=TRUE
NETWORK_SPEED=1000
STARTD_EXPRS=INSTRUCTIONAL,
  NETWORK_SPEED
```

Custom Machine Attributes

- Jobs can now specify Rank and Requirements using new attributes:

```
Requirements = (INSTRUCTIONAL=?
  =UNDEFINED ||
  INSTRUCTIONAL==FALSE)
Rank = NETWORK_SPEED
```

- Dynamic attributes are available; see **STARTD_CRON_*** settings in the manual

Custom Machine Attributes

- We can move some or all of our policy macros into the ClassAd:

```
IsOwner = (Owner == "adesmet"\
   || Owner == "roy")
STARTD_EXPRS = IsOwner
RANK = IsOwner
# Instead of RANK=$(IsOwner)
```

Further Machine Policy Information

- For further information, see section 3.5 “Policy Configuration for the condor_startd” in the Condor manual
- condor-users mailing list
  http://www.cs.wisc.edu/condor/mail-lists/
- condor-admin@cs.wisc.edu

Priorities

"IMG_2476" by "Joanne and Matt" © 2006 Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/joanne_matt/97737986/ http://www.webcitation.org/5XIieCxq4

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`)
- Nice users default to 1,000,000 (`NICE_USER_PRIO_FACTOR`)
  - Used for true bottom feeding jobs
  - Add "`nice_user=true`" to your submit file

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

- 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 > SubmittorPrio * 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
```

Security

"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/5XIiBcsUg

Condor 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/5XIhUAfuY

Security Features

- You need to turn the advanced security features on

```
SEC_DEFAULT_AUTHENTICATION=REQUIRED
SEC_DEFAULT_ENCRYPTION    =REQUIRED
SEC_DEFAULT_INTEGRITY     =REQUIRED
```

- Can set on a per security level basis, see the manual.

Security Levels: A Subset

- READ
  - querying information
  - `condor_status`, `condor_q`, etc
- WRITE
  - updating information
  - `condor_submit`, adding nodes to a pool, sending ClassAds to the collector, etc
  - Includes READ

Security Levels: A Subset

- ADMINISTRATOR
  - Administrative commands
  - `condor_on`, `condor_off`, `condor_reconfig`, `condor_restart`, etc.
  - Includes READ and WRITE

Security Levels: A Subset

- DAEMON
  - Daemon to daemon communications
  - Includes READ and WRITE
- NEGOTIATOR
  - `condor_negotiator` to other daemons
  - Includes READ

Specifying User Identities

- Canonical form (shortcuts exist): username@domain.com/hostname.com
- adesmet@cs.wisc.edu/puffin.cs.wisc.edu
- Can use * wildcard
- Hostname can be hostname or IP address with optional netmask
  - 192.168.12.1/255.255.192.0
  - 192.168.12.1/18

Setting Up Security

- List who you ALLOW access to
  - **ALLOW_WRITE=…**
- If not ALLOWed, then defaults to DENY access
- Can also DENY people
  - **DENY_WRITE=…**
  - Warning: If you set DENY_* but not a matching ALLOW_* expression, access defaults to ALLOW.

Setting Up Security

- Can define values that effect all daemons:
  - **ALLOW_WRITE, DENY_READ, ALLOW_ADMINISTRATOR**, etc.
- Can define daemon-specific settings:
  - **ALLOW_READ_SCHEDD, DENY_WRITE_COLLECTOR**, etc.

Example Filters

- Allow anyone from wisc.edu:

```
ALLOW_READ=*@wisc.edu/*.wisc.edu
```

- Allow any authenticated local user:

```
ALLOW_READ=*/*.wisc.edu
```

- Allow specific user/machine

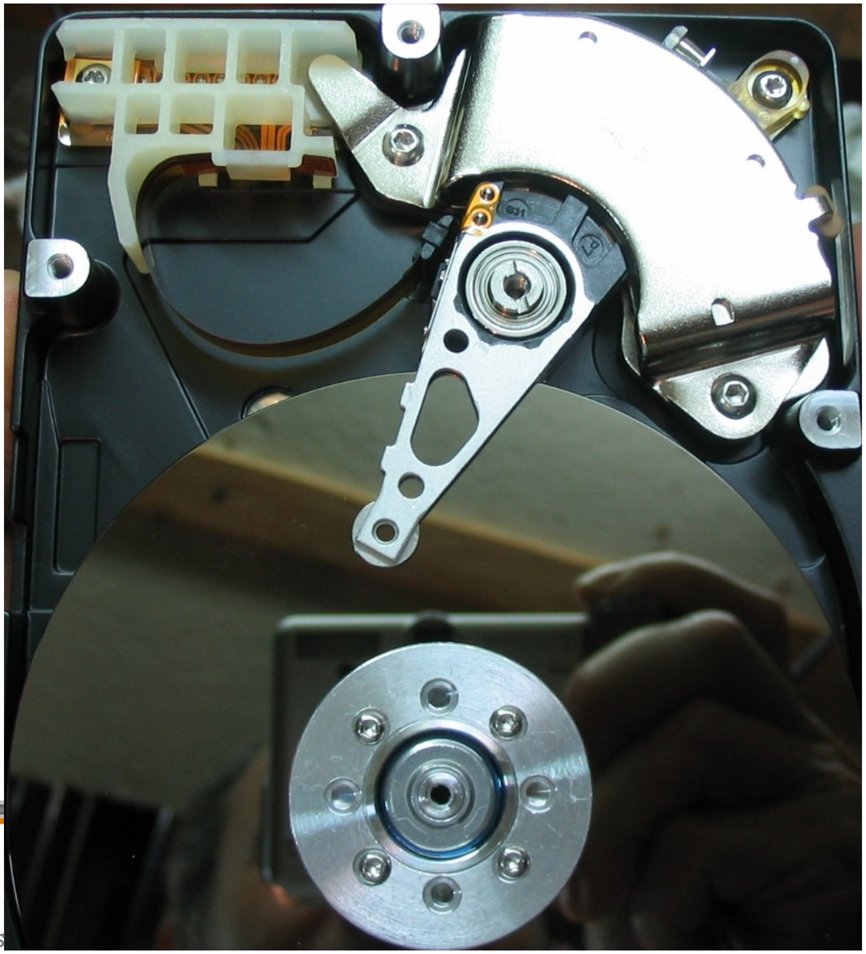
```
ALLOW_NEGOTIATOR= \
  daemon@wisc.edu/condor.wisc.edu
```

AUTHENTICATION_METHODS

- How to authenticate users and daemons?
  - FS – Local file system
  - SSL – Public key encryption
  - PASSWORD – Shared secret
  - ANONYMOUS
  - NTSSPI – Microsoft Windows
  - Kerberos
  - GSI – Globus/Grid Security Infrastructure
  - CLAIMTOBE - Insecure
  - FS_REMOTE - Network file system

FS: File System

- Checks that the user can create a directory owned by the user.
  - Only works on local machine
  - Assumes filesystem is trustworthy
- Everyone should use
- It just works!



"Hard drive" by Robbie Sproule © 2005

http://www.flickr.com/photos/robbie1/73032053/ http://www.webcitation.org/5XQVcvsyYs

PASSWORD

- Shared secret encryption file
- Only suitable for daemon-to-daemon communications
- Simple

SSL

- Public key encryption system
- Daemons and users have X.509 certificates
- All Condor daemons in pool can share one certificate
- Map file transforms X.509 distinguished name into an identity
  - You'll need to create this map file. See "3.6.4 The Unified Map File for Authentication" in the manual.

NTSSPI
Microsoft Windows

- Only works on Windows
- Insecure encryption and integrity checks

ANONYMOUS

- ANONYMOUS - A sort of "guest" user
  - CONDOR_ANONYMOUS_USER
  - Insecure encryption and integrity checks

Kerberos and GSI

- Complex to set up
- Useful if you already use one of these systems

Example Security Configuration

- Use SSL authentication for between machine connections
- Use SSL or FS authentication on a single machine

Example Security Configuration

```
# Turn on all security:
SEC_DEFAULT_AUTHENTICATION=REQUIRED
SEC_DEFAULT_ENCRYPTION=REQUIRED
SEC_DEFAULT_INTEGRITY=REQUIRED
```

Example Security Configuration

```
# Require authentication
SEC_DEFAULT_AUTHENTICATION_METHODS =
  FS, SSL
```

- Requires giving your daemons an X.509 certificates
- You will also need a map file

Example Security Configuration

```
ALLOW_READ = *
ALLOW_WRITE= *@wisc.edu/*.wisc.edu
DENY_WRITE = abuser@*.wisc.edu/*
ALLOW_ADMINISTRATOR =
  admin@wisc.edu/*.wisc.edu,
  *@wisc.edu/$(CONDOR_HOST)
```

Example Security Configuration

```
ALLOW_DAEMON = daemon@wisc.edu/
  *.wisc.edu
ALLOW_NEGOTIATOR =
  daemon@wisc.edu/$(CONDOR_HOST)
```

Users without Certificates

- Using **FS** authentication users can submit jobs and check the local queue
- **condor_q -analyze** and **condor_status** won't work for normal users without an X.509 certificate
  - Requires **READ** access to **condor_collector**
- How to let anyone read any daemon? **ANONYMOUS** authentication

Allow Any User Read Access

- `SEC_READ_AUTHENTIATION_METHODS = FS, SSL, ANONYMOUS`

- The "`ALLOW_READ = *`" handles the rest. We could more explicitly match against "`CONDOR_ANONYMOUS_USER/*`" if we wanted.

Old Condor Security

- HOSTALLOW_* and HOSTDENY_*
- Deprecated
- Security is entirely based on IP addresses and host names
- No encryption or integrity checking

More on Security

- Chapter 3.6, "Security," in the Condor Manual
- condor-admin@cs.wisc.edu
- Capture the wily Zach Miller

"Zach Miller" by Alan De Smet

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/5XIhNWC7Y

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 WINNT50
     108 SUN4u SOLARIS28
       6 SUN4x SOLARIS28
```

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 -format

- Census of jobs per user

```
% condor_q -format '%8s ' Owner
  -format '%s\n' Cmd | sort |
  uniq -c
   64 adesmet /scratch/submit/a.out
    2 adesmet /home/bin/run_events
    4   smith /nfs/sim1/em2d3d
    4   smith /nfs/sim2/em2d3d
```

`condor_q -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

condor_q -analyze

- Typical results:

```
% condor_q -analyze 471216
471216.000:  Run analysis summary.  Of 820 machines,
    458 are rejected by your job's requirements
     25 reject your job because of their own requirements
      0 match, but are serving users with a better priority in the pool
      4 match, but reject the job for unknown reasons
      6 match, but will not currently preempt their existing job
    327 are available to run your job
        Last successful match: Sun Apr 27 14:32:07 2008
```

`condor_q -better-analyze`

- Breaks down the job's requirements and suggests modifications
- Entirely replaces **`-analyze`** as of 7.5.1

condor_q -better-analyze

- (Heavily truncated output)

```
The Requirements expression for your job is:
( ( target.Arch == "SUN4u" ) && ( target.OpSys ==
    "WINNT50" ) && [snip]
Condition                          Machines  Suggestion
1 (target.Disk > 100000000)    0      MODIFY TO 14223201
2 (target.Memory > 10000)      0      MODIFY TO 2047
3 (target.Arch == "SUN4u")     106
4 (target.OpSys == "WINNT50") 110     MOD TO "SOLARIS28"

Conflicts: conditions: 3, 4
```

Log Files

“Ready for the Winter” by Anna “bcmom” © 2005 Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/bcmom/59207805/ http://www.webcitation.org/5XIhRO8L8

Condor's Log Files

- Condor 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

Condor's Log Files

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

Debugging Jobs

“Wanna buy a Beetle?” by “Kevin” © 2006 Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/kevincollins/89538633/ http://www.webcitation.org/5XIiMyhpp

Debugging Jobs: condor_q

- Examine the job with `condor_q`
  - especially `-long` and `-analyze`
  - Compare with `condor_status -long` for a machine you expected to match

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
  - condor-admin@cs.wisc.edu

More Information

“IMG 0915” by Eva Schiffer © 2008 Used with permission
http://www.digitalchangeling.com/pictures/ourCats2008/january2008/IMG_0915.html

More Information

- Condor staff here at Condor Week
- Condor Manual
- condor-users mailing list

  http://www.cs.wisc.edu/condor/mail-lists/
- condor-admin

  condor-admin@cs.wisc.edu

"Condor Manual" by Alan De Smet
(Actual first page of the 7.0.1 manual on about 700 pages of other output. The actual 7.0.1 manual is about 860 pages.)

Thank You!

Any questions?

“My mouse” by “MysterFaery” © 2006 Licensed under the Creative Commons Attribution 2.0 license
http://www.flickr.com/photos/mysteryfaery/294253525/ http://www.webcitation.org/5XIi6HRCM