



#### HTCondor Administration Basics

Greg Thain Center for High Throughput Computing

#### Overview

- > HTCondor Architecture Overview
- > Classads, briefly
- Configuration and other nightmares
- > Setting up a personal condor
- > Setting up distributed condor
- > Minor topics





#### **Two Big HTCondor Abstractions**

#### > Jobs



#### > Machines



**H**Cond



#### Life cycle of HTCondor Job





COMPUTING

#### "Submit Side"



#### "Execute Side"



# The submit side

- Submit side managed by 1 condor\_schedd process
- And one shadow per running job
  - condor\_shadow process
- The Schedd is a database

- Submit points can be performance bottleneck
- Usually a handful per pool





#### "Submit Side"



### In the Beginning...

```
universe = vanilla
executable = compute
request memory = 70M
arguments = $ (ProcID)
should transfer input = yes
output = out.$(ProcID)
error = error.$(ProcId)
+IsVerySpecialJob = true
Queue
```

#### HTCondor Submit file





# From submit to schedd

```
universe = vanilla
executable = compute
request_memory = 70M
arguments = $(ProcID)
should_transfer_input = yes
output = out.$(ProcID)
error = error.$(ProcId)
+IsVerySpecialJob = true
Queue
```

JobUniverse = 5 Cmd = "compute" Args = "0" RequestMemory = 70000000 Requirements = Opsys == "Li.. DiskUsage = 0 Output = "out.0" IsVerySpecialJob = true

condor\_submit submit\_file IsVerySpecial Submit file in, Job classad out Sends to schedd man condor\_submit for full details Other ways to talk to schedd

Python bindings, SOAP, wrappers (like DAGman)





# Condor\_schedd holds all jobs

One pool, Many schedds

condor\_submit –name chooses Owner Attribute: need authentication Schedd also called "q" not actually a queue

JobUniverse = 5Owner = "gthain" JobStatus = 1NumJobStarts = 5Cmd = "compute" Args = "0"RequestMemory = 7000000Requirements = Opsys == "Li. DiskUsage = 0Output = "out.0"IsVerySpecialJob = true



# Condor\_schedd has all jobs

- > In memory (big)
  - condor\_q expensive
- > And on disk

CENTER FOR

- Fsync's often
- Monitor with linux
- > Attributes in manual
- > condor\_q -l job.id

ROUGHPUT

• e.g. condor\_q -l 5.0

```
JobUniverse = 5
Owner = "gthain"
JobStatus = 1
NumJobStarts = 5
Cmd = "compute"
Args = "0"
RequestMemory = 7000000
Requirements = Opsys == "Li..
DiskUsage = 0
Output = "out.0"
IsVerySpecialJob = true
```



# What if I don't like those Attributes?

- > Write a wrapper to condor\_submit
- > SUBMIT\_ATTRS
- > condor\_qedit
- > +Notation
- Schedd transforms

DUGHPUT



# ClassAds: The *lingua franca* of HTCondor

CLASS		IFIEDS	
	ACHYOUR PO 20. HOUSES FOR SALE ARA DAMANSARA ARA	ROSPECTS FIR	20 20. HOUSES FOR SALE The ADDAR BUTWARY WINDOW THE ADDAR BUTWARY W



#### **Classads for people admins**





#### What are ClassAds?

ClassAds is a language for objects (jobs and machines) to

- Express attributes about themselves
- Express what they require/desire in a "match" (similar to personal classified ads)

Structure : Set of attribute name/value pairs, where the value can be a literal or an expression. Semi-structured, no fixed schema.





#### Example

#### Pet Ad

CENTER FOR

COMPUTING

Type = "Dog"
Requirements =
 DogLover =?= True
Color = "Brown"
Price = 75
Sex = "Male"
AgeWeeks = 8
Breed = "Saint Bernard"
Size = "Very Large"
Weight = 27

'HROUGHPUT

#### Buyer Ad AcctBalance = 100 DogLover = True Requirements = (Type == "Dog") && (TARGET.Price <= MY.AcctBalance) && ( Size == "Large" ||

```
Size == "Very Large" )
```

#### Rank =

```
100* (Breed == "Saint
Bernard") - Price
```



#### **ClassAd Values**

- > Literals
  - Strings ( "RedHat6" ), integers, floats, boolean (true/false), ...
- > Expressions
  - Similar look to C/C++ or Java : operators, references, functions
  - References: to other attributes in the same ad, or attributes in an ad that is a candidate for a match
  - Operators: +, -, \*, /, <, <=,>, >=, ==, !=, &&, and || all work as expected
  - Built-in Functions: if/then/else, string manipulation, regular expression pattern matching, list operations, dates, randomization, math (ceil, floor, quantize,...), time functions, eval, ...





#### **Four-valued logic**

- > ClassAd Boolean expressions can return four values:
  - True
  - False

CENTER FOR

OMPUTING

- Undefined (a reference can't be found)
- Error (Can't be evaluated)
- > Undefined enables explicit policy statements *in the absence of data* (common across administrative domains)
- Special meta-equals ( =?= ) and meta-not-equals (=!=) will never return Undefined

```
HasBeer = True [
GoodPub1 = HasBeer == True GoodPub2 = HasBeer =?= True ]
```

```
GoodPub1 = HasBeer == True
GoodPub2 = HasBeer =?= True
```



# **ClassAd Types**

> HTCondor has many types of ClassAds

- A "Job Ad" represents a job to Condor
- A "Machine Ad" represents a computing resource
- Others types of ads represent other instances of other services (daemons), users, accounting records.





# **The Magic of Matchmaking**

- > Two ClassAds can be matched via special attributes: Requirements and Rank
- > Two ads match if both their Requirements expressions evaluate to True
- Rank evaluates to a float where higher is preferred; specifies which match is desired if several ads meet the Requirements.
- > Scoping of attribute references when matching
  - MY.name Value for attribute "name" in local ClassAd
  - TARGET.name Value for attribute "name" in match candidate ClassAd
  - Name Looks for "name" in the local ClassAd, then the candidate ClassAd





#### Example

#### Pet Ad

CENTER FOR

COMPUTING

Type = "Dog"
Requirements =
 DogLover =?= True
Color = "Brown"
Price = 75
Sex = "Male"
AgeWeeks = 8
Breed = "Saint Bernard"
Size = "Very Large"
Weight = 27

'HROUGHPUT

#### Buyer Ad AcctBalance = 100 DogLover = True Requirements = (Type == "Dog") &&

(TARGET.Price <=

```
MY.AcctBalance) &&
```

```
( Size == "Large" ||
```

```
Size == "Very Large" )
```

```
Rank =
```

```
100* (Breed == "Saint
Bernard") - Price
```



#### **Back to configuration...**





# **Configuration File**

- > (Almost) all configure is in files, "root" CONDOR\_CONFIG env var /etc/condor/condor\_config
- > This file points to others
- > All daemons share same configuration
- Might want to share between all machines (NFS, automated copies, puppet, etc)





### **Configuration File Syntax**

- # I'm a comment!
- CREATE CORE FILES=TRUE
- $MAX_JOBS_RUNNING = 50$
- # HTCondor ignores case:
- log=/var/log/condor
- # Long entries:

ROUGHPUT

**FNTER FOR** 

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





# **Config file defaults**

- > CONDOR\_CONFIG "root" config file:
  - /etc/condor/condor\_config
- > Local config file:
  - /etc/condor/condor\_config.local
- Config directory
  - /etc/condor/config.d





### **Config file recommendations**

- > For "system" condor, use default
  - Global config file read-only
    - /etc/condor/condor\_config
  - All changes in config.d small snippets
    - /etc/condor/config.d/05some\_example
  - All files begin with 2 digit numbers
- > Personal condors elsewhere





# condor\_config\_val

- > condor\_config\_val [-v] <KNOB\_NAME>
  - Queries config files
- > condor\_config\_val -dump
- > Environment overrides:
- > export \_condor\_KNOB\_NAME=value
  - Over rules all others (so be careful)





#### condor\_reconfig

- > Daemons long-lived
  - Only re-read config files on condor\_reconfig command
  - Some knobs don't obey re-config, require restart
    - DAEMON\_LIST, NETWORK\_INTERFACE
- > condor\_restart





#### Got all that?





# **Configuration of Submit side**

- > Not much policy to be configured in schedd
- > Mainly scalability and security
- > MAX\_JOBS\_RUNNING
- > JOB\_START\_DELAY
- > MAX\_CONCURRENT\_DOWNLOADS
- > MAX\_JOBS\_SUBMITTED





#### **The Execute Side**

#### Primarily managed by condor\_startd process

With one condor\_starter per running jobs

Sandboxes the jobs

Usually many per pool (support 10s of thousands)







#### Startd also has a classad

#### > Condor creates it

- From interrogating the machine
- And the config file
- And sends it to the collector
- > condor\_status [-I]
  - Shows the ad
- > condor\_status –direct daemon
  - Goes to the startd




#### **Condor\_status –I machine**

```
OpSys = "LINUX"
CustomGregAttribute = "BLUE"
OpSysAndVer = "RedHat6"
TotalDisk = 12349004
Requirements = ( START )
UidDomain = "cheesee.cs.wisc.edu"
Arch = "X86 64"
StartdIpAddr = "<128.105.14.141:36713>"
RecentDaemonCoreDutyCycle = 0.000021
Disk = 12349004
Name = "slot1@chevre.cs.wisc.edu"
State = "Unclaimed"
Start = true
Cpus = 32
```





# **One Startd, Many slots**

- > HTCondor treats multicore as independent slots
- > Slots: static vs. partitionable
- > Startd can be configured to:
  - Only run jobs based on machine state
  - Only run jobs based on Resources (GPUs)
  - Preempt or Evict jobs based on policy
  - 0





## **3 types of slots**

- > Static (e.g. the usual kind)
- > Partitionable (e.g. leftovers)
- > Dynamic (usableable ones)
  - Dynamically created
  - But once created, static



#### How to configure

NUM\_SLOTS = 1
NUM\_SLOTS\_TYPE\_1 = 1
SLOT\_TYPE\_1 = cpus=100%
SLOT\_TYPE\_1\_PARTITIONABLE = true





# **Configuration of startd**

- > Mostly policy,
- > Several directory parameters
- > EXECUTE where the sandbox is
- > COLLECTOR\_HOST where the cm is
- > CLAIM\_WORKLIFE
  - How long to reuse a claim for different jobs





#### The "Middle" side

- There's also a "Middle", the Central Manager:
  - A condor\_negotiator
    - Provisions machines to schedds
  - A condor\_collector
    - Central nameservice: like LDAP
    - condor\_status queries this
- > Please don't call this "Master node" or head
- > Not the bottleneck you may think: stateless





### **Responsibilities of CM**

- > Pool-wide scheduling policy resides here
- > Scheduling of one user vs another
- > Definition of groups of users
- > Definition of preemption
- > Whole talk on this this pm.





## Defrag daemon

- > Optional, but usually on the central manager
  - One daemon defragments whole pool
- > Scan pool, try to fully defrag some startds
- > Only looks at partitionable machines
- > Admin picks some % of pool that can be "whole"



#### The condor\_master

- > Every condor machine needs a master
- > Like "systemd", or "init"

- > Starts daemons, restarts crashed daemons
- > Tunes machine for condor





#### **Quick Review of Daemons**

condor\_master: runs on all machine, always

condor\_schedd: runs on submit machine condor\_shadow: one per job condor\_startd: runs on execute machine condor\_starter: one per job condor\_negotiator/condor\_collector one per pool









#### **Process View: Central Manager**



#### **Condor Installation Basics**





# Let's Install HTCondor

- > Either with tarball
  - tar xvf htcondor-8.6.11-redhat6
- > Or native packages

wget

http://research.cs.wisc.edu/htcondor/yum/repo.d/h
tcondor-stable-rhel6.repo

get <u>http://research.cs.wisc.edu/htcondor/yum/RPM-</u> <u>GPG-KEY-HTCondor</u>

rpm -import RPM\_GPG-KEY-HTCondor

Yum install htcondor





## http://htcondorproject.org

💽 HTCondor - Home - Google Chron 👅 🔞 HICondor - Home 🛛 🗙 🕅

#### C (i) research.cs.wisc.edu/htcondor/

#### HICONDOR High Throughput Computing

Home | News | Download | Publications | Contact Us

Google Custom Search

#### **Computing with HTCondor™**

Our goal is to develop, implement, deploy, and evaluate mechanisms and policies that support <u>High Throughput Computing (HTC)</u> on large collections of distributively owned computing resources. Guided by both the technological and sociological challenges of such a computing environment, the <u>Center for High Throughput Computing</u> at UW-Madison has been building the open source <u>HTCondor distributed computing</u> software (pronounced "aitch-tee-condor") and related technologies to enable scientists and engineers to increase their computing throughput.

Note: The HTCondor software was known as 'Condor' from 1988 <u>until its</u> <u>name changed</u> in 2012. If you are looking for Phoenix Software International's software development and library management system for z/VSE or z/OS, click <u>here</u>.

#### Latest News RSS

- > HTCondor powers Marshfield Clinic project on disease genetics May 16, 2018
- HTCondor 8.7.8 released! May 10, 2018
- > HTCondor 8.6.11 released! May 10, 2018
- » HTCondor 8.7.7 released! March 13, 2018
- > HTCondor 8.6.10 released! March 13, 2018
- > HTCondor Week 2018 Registration Open February 28, 2018

More News »

#### HTCondor Week 2018

GH THROUGHPUT

CENTER FOR

COMPUTING

Our annual <u>HTCondor Week</u> user conference will be held in beautiful Madison, WI on May 21-24, 2018, and will feature tutorials and talks from HTCondor developers, administrators, and users. Visit the website for details. Also note the HEDIX



#### **Version Number Scheme**

#### > Major.minor.release

- If minor is even (a.b.c): Stable series
  - Very stable, mostly bug fixes
  - Current: 8.6
  - Examples: 8.2.5, 8.0.3
    - 8.6.0 coming soon to a repo near you
- If minor is odd (a.b.c): Developer series
  - New features, may have some bugs
  - Current: 8.7
  - Examples: 8.3.2,
    - 8.5.5 almost released



#### **The Guarantee**

- > All minor releases in a stable series interoperate
  - E.g. can have pool with 8.4.0, 8.4.1, etc.
  - But not WITHIN A MACHINE:
    - Only across machines
- > The Reality
  - We work really hard to do better
    - 8.4 with 8.2 with 8.5, etc.
    - Part of HTC ideal: can never upgrade in lock-step





#### Let's Make a Pool

- > First need to configure HTCondor
- > 1100+ knobs and parameters!
- > Don't need to set all of them...





#### **Default file locations**

BIN = /usr/bin SBIN = /usr/sbin LOG = /var/condor/log SPOOL = /var/lib/condor/spool EXECUTE = /var/lib/condor/execute CONDOR CONFIG = /etc/condor/condor config





## Let's make a pool!

- > "Personal Condor"
  - All on one machine:
    - submit side IS execute side
  - Jobs always run
- > Use defaults where ever possible
- Very handy for debugging and learning





### Minimum knob settings

#### Role

What daemons run on this machine

#### CONDOR\_HOST

• Where the central manager is

#### Security settings

• Who can do what to whom?





### **Other interesting knobs**

#### LOG = /var/log/condor Where daemons write debugging info SPOOL = /var/spool/condor Where the schedd stores jobs and data EXECUTE = /var/condor/execute Where the startd runs jobs





# Minimum knobs for personal Condor

> In /etc/condor/config.d/50PC.config

- # All daemons local
- Use ROLE : Personal

#### CONDOR\_HOST = localhost ALLOW\_WRITE = localhost





#### **Does it Work?**

\$ condor\_status
Error: communication error
CEDAR:6001:Failed to connect to <128.105.14.141:4210>

\$ condor\_submit
ERROR: Can't find address of local schedd

\$ condor\_q

Error:

Extra Info: You probably saw this error because the condor\_schedd is not running on the machine you are trying to query...







#### \$ ps auxww | grep [Cc]ondor







# **Starting Condor**

- > condor\_master –f
- > service start condor





<pre>\$ ps auxww   grep [Cc]ondor \$</pre>											
condor	19534 5	0380	Ss	11:19	0:00 condor_master						
root	19535 23	1692	S	11:19	0:00 condor_procd -A …						
condor	19557	69656	Ss	11:19	0:00 condor_collector -f						
condor	19559	51272	Ss	11:19	0:00 condor_startd -f						
condor	19560	71012	Ss	11:19	0:00 condor_schedd -f						
condor	19561	50888	Ss	11:19	0:00 condor_negotiator -f						

#### Notice the UID of the daemons





#### Quick test to see it works

<pre>\$ condor_status # Wait a few minutes</pre>												
\$ condor_status												
Name	OpSys	Arch	State	Activity	LoadAv	Mem						
slot1@chevre.cs.wi	LINUX	X86_64	Unclaimed	Idle	0.190	20480						
slot2@chevre.cs.wi	LINUX	X86_64	Unclaimed	Idle	0.000	20480						
slot3@chevre.cs.wi	LINUX	X86_64	Unclaimed	Idle	0.000	20480						
slot4@chevre.cs.wi	LINUX	X86_64	Unclaimed	Idle	0.000	20480						
-bash-4.1\$ condor_q												
Submitter: gthain@chevre.cs.wisc.edu : <128.105.14.141:35019> :												
chevre.cs.wisc.edu												

ID OWNER SUBMITTED RUN TIME ST PRI SIZE CMD

0 jobs; 0 completed, 0 removed, 0 idle, 0 running, 0 held, 0 suspended \$ condor restart # just to be sure...





### Some Useful Startd Knobs

>NUM\_CPUS = X

• How many cores condor thinks there are > MEMORY = M

- How much memory (in Mb) there is
- >STARTD\_CRON\_...
  - Set of knobs to run scripts and insert attributes into startd ad (See Manual for full details).





# **Brief Diversion into daemon logs**

- > Each daemon logs mysterious info to file
- > \$(LOG)/DaemonNameLog
- > Default:
  - /var/log/condor/SchedLog
  - /var/log/condor/MatchLog
  - /var/log/condor/StarterLog.slotX
- > Experts-only view of condor





#### Let's make a "real" pool

- > Distributed machines makes it hard
  - Different policies on each machines
  - Different owners
  - Scale





## **Most Simple Distributed Pool**

- > Requirements:
  - No firewall
  - Full DNS everywhere (forward and backward)
  - We've got root on all machines
- > HTCondor doesn't require any of these
  - (but easier with them)





# What UID should jobs run as?

#### > Three Options (all require root):

- Nobody UID
  - Safest from the machine's perspective
- The submitting User
  - Most useful from the user's perspective
  - May be required if shared filesystem exists
- A "Slot User"
  - Bespoke UID per slot
  - Good combination of isolation and utility





#### **UID\_DOMAIN SETTINGS**

UID\_DOMAIN = \
same\_string\_on\_submit
TRUST\_UID\_DOMAIN = true
SOFT\_UID\_DOMAIN = true

If UID\_DOMAINs match, jobs run as user, otherwise "nobody"





#### **Slot User**

SLOT1\_USER = slot1 SLOT2\_USER = slot2

STARTER\_ALOW\_RUNAS\_OWNER = false
EXECUTE\_LOGIN\_IS\_DEDICATED=true

Job will run as slotX Unix user




## FILESYSTEM\_DOMAIN

- > HTCondor can work with NFS
  - But how does it know what nodes have it?
- > WhenSubmitter & Execute nodes share
  - FILESYSTEM\_DOMAIN values

- e.g FILESYSTEM\_DOMAIN = domain.name

- > Or, submit file can always transfer with
  - should\_transfer\_files = yes
- > If jobs always idle, first thing to check





## **3 Separate machines**

> Central Manager

> Execute Machine

> Submit Machine





## **Central Manager**

Use ROLE : CentralManager CONDOR\_HOST = cm.cs.wisc.edu ALLOW\_WRITE = \*.cs.wisc.edu





## **Submit Machine**

Use ROLE : submit CONDOR\_HOST = cm.cs.wisc.edu ALLOW\_WRITE = \*.cs.wisc.edu UID\_DOMAIN = cs.wisc.edu FILESYSTEM DOMAIN = cs.wisc.edu





### **Execute Machine**

Use ROLE : Execute CONDOR HOST = cm.cs.wisc.edu ALLOW WRITE = \*.cs.wisc.edu UID DOMAIN = cs.wisc.edu FILESYSTEM DOMAIN = cs.wisc.edu # default is #FILESYSTEM DOMAIN=\$(FULL HOSTNAME)





## Now Start them all up

- > Does order matter?
  - Somewhat: start CM first
- > How to check:
- > Every Daemon has classad in collector
  - condor\_status -schedd
  - condor\_status -negotiator
  - condor\_status -any





## condor\_status -any

МуТуре	TargetType	Name
Collector	None	Test <u>Pool@cm.cs.wisc.edu</u>
Negotiator	None	cm.cs.wisc.edu
DaemonMaster	None	cm.cs.wisc.edu
Scheduler	None	submit.cs.wisc.edu
DaemonMaster	None	submit.cs.wisc.edu
DaemonMaster	None	wn.cs.wisc.edu
Machine	Job	slot1@wn.cs.wisc.edu
Machine	Job	slot2@wn.cs.wisc.edu
Machine	Job	slot3@wn.cs.wisc.edu
Machine	Job	slot4@wn.cs.wisc.edu



# **Debugging the pool**

> condor\_q / condor\_status

> condor\_ping ALL -name machine

- > Or
- > condor\_ping ALL –addr '<127.0.0.1:9618>'





# What if a job is always idle?

- > Check userlog may be preempted often
- > run condor\_q -better-analyze job\_id





#### Whew!





# Speeds, Feeds, Rules of Thumb

- > HTCondor scales to 100,000s of machines
  - With a lot of work
  - Contact us, see wiki page



•



## **Without Heroics:**

- > Your Mileage may vary:
  - Shared File System vs. File Transfer
  - WAN vs. LAN
  - Strong encryption vs none
  - Good autoclustering
- > A single schedd can run at 50 Hz
- > Schedd needs 500k RAM for running job
  - 50k per idle jobs

#### > Collector can hold tens of thousands of ads



### **Tools for admins**





## condor\_off

- > Three kinds for submit and execute
- > -fast:
  - Kill all jobs immediate, and exit
- > -gracefull
  - Give all jobs 10 minutes to leave, then kill
- > -peaceful
  - Wait forever for all jobs to exit





#### condor\_restart

> Restarts all daemons on a given machine

> Can be run remotely – if admin priv allows





### condor\_status

- > -collector
- > -submitter
- > -negotiator
- > -schedd
- > -master



## condor\_userprio

- > Condor\_userprio –allusers
  - Whole talk on this,





## condor\_fetchlog

> Remotely pulls a log file from remote machine

> condor\_fetchlog execute\_machine STARTD





## Thank you -- For more info

> http://htcondorproject.org

- > Detail talks today...
- > htcondor-users email list

> Talk to us!



