
Rolling
Upgrades
(without rolling over)
Greg Thain
CHTC

What are we trying to do?

- Move all nodes of a pool from version n → n + 1
    - (or from version n + 1 → n)
- WITH minimum interruption to jobs, users, etc.

CHTC

3 potential players involved

HTCondor Software Suite

Native packaging

[● ◀] systemd

CHTC

[●◀] systemd

- HTCondor does use systemd
  - For starting the master
  - But not until dependent systems are running
  - Creating cgroups and environment for the master
  - Restarting the master when it crashes

- But not for updates
  - During an update, the master re-execs itself (via condor_restart –master)
  - Gets a new executable, but keeps it's old pid
  - Systemd doesn't see the re-exec

CHTC

3 potential players involved

HTCondor Software Suite

Native packaging

[● ◀] systemd

CHTC

Role of the native packaging tools

Native packaging

- Replaces all old files with new (binaries, libs, etc)
- And then runs a script to restart the new system
- The restart script for .rpm/.deb, etc. just does
  - Nothing (of significance – does not run condor_restart)
  - This is hard-coded in the package, you can't change it

CHTC

3 potential players involved

HTCondor Software Suite

Native packaging

[● ◀] systemd

CHTC

The HTCSS role

after the native packaging tool updates the files, we are in condor-land

What does condor_master do?

It Depends!

There's a knob for that!

There's a knob for that!

```
# How often the condor_master stats the timestamp of
daemons
MASTER_CHECK_NEW_EXEC_INTERVAL = 300
# When the master detects new files, wait this long
# before action
MASTER_NEW_BINARY_DELAY = 120  # in second
# When the master detects new binaries, how to restart
MASTER_NEW_BINARY_RESTART = GRACEFUL | FAST | NEVER

DEFAULT_MASTER_SHUTDOWN_SCRIPT =
```

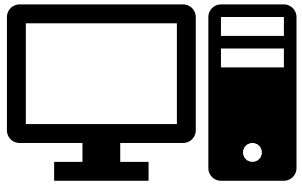
These are all under a site's control, and are part of site policies, not hard-coded into HTCSS or the packaging

How does the CHTC set these?

CHTC

It Depends!

It depends on the **role** of the node



Access Point

Central Manager

Execution Point

The Central Manager role

Central Manager

CM is **stateless**

Every daemon re-updates at least every five minutes

Just what to restart quickly

```
# When the master detects new binaries, how to restart
MASTER_NEW_BINARY_RESTART = FAST


# When the master detects new files, wait this long
# before action
MASTER_NEW_BINARY_DELAY = 5
```

The Access Point role

Access Point

AP is **stateful**, but can reconnect

If we can restart and reconnect to a running job, job continues running

So, we also want **fast** here (new in HTCSS v24…)

```
# When the master detects new binaries, how to restart
MASTER_NEW_BINARY_RESTART = FAST


# When the master detects new files, wait this long
# before action
MASTER_NEW_BINARY_DELAY = 5
```

The Execution Point role

Execution Point

In CHTC, we have two subtypes of Execution Point

FOR THE PURPOSES of UPGRADES (!)

"Big Memory"

- Really means long-running jobs
- Big memory strongly correlates to job runtime

"Normal"

- Everything else
- Max 24 hours job runtime

The "normal" Execution Point role

Access Point
(normal)

EP running jobs, many of which can't checkpoint
Want to give the jobs their promised time to complete
Functionally, like a drain, but without the explicit drain command
If we can restart and reconnect to a running job, job continues running
So, we also want **graceful** here

```
# When the master detects new binaries, how to restart
MASTER_NEW_BINARY_RESTART = GRACEFUL

SHUTDOWN_GRACEFUL_TIMEOUT = 18 * 3600  # how much grace

# When the master detects new files, wait this long

# before action

MASTER_NEW_BINARY_DELAY = 5
```

The "big mem" Execution Point role

Access Point
(big memory)

EP running jobs, many of which can't checkpoint
Want to manually check when the jobs are completed
So we explicitly run "condor_drain" by hand on these machines

```
# When the master detects new binaries, how to restart
MASTER_NEW_BINARY_RESTART = NEVER
```

```
# run the following on each big mem EP

$ condor_restart -drain -reason Upgrade
```

One warning about this state

- When draining or graceful
  - Package manager has upgraded the binaries on disk
  - But the old ones are still running

- $ condor_version will show the new one

- Perhaps this could lead to problems, but we haven't seen any yet

3 potential players involved

HTCondor Software Suite

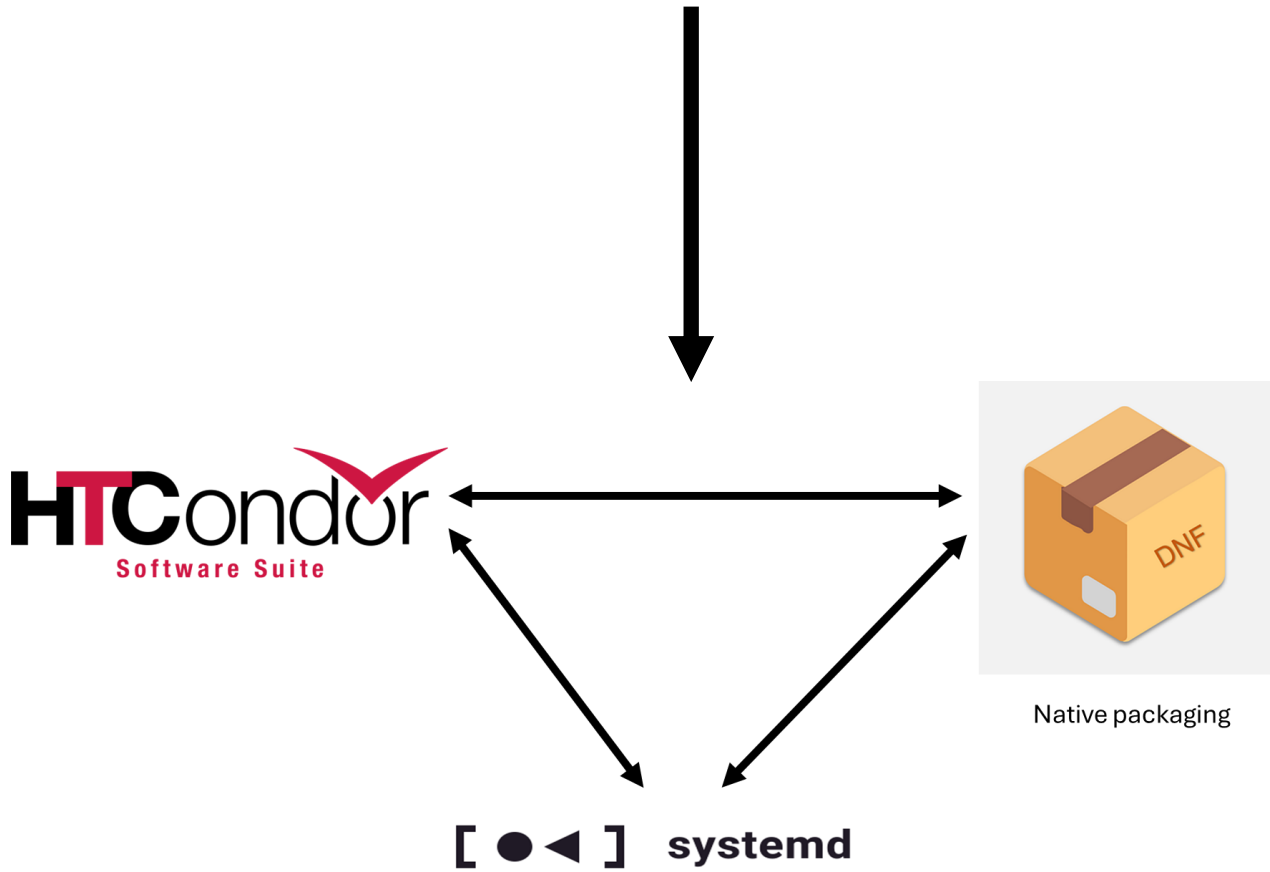
Native packaging

[● ◀] systemd

CHTC

perforce

Puppet



Native packaging

What we tell puppet to do

- Day 1
  - Update CM, most APs, kick off upgrade of 25% of Execution Points
- Day 2
  - Update very busy APs, kick off upgrade of 25% of Execution Points
- Day 3 and 4
  - Kick off upgrade of 25% of Execution Points

Do users not notice?

- We essentially have a rolling drain over the week
  - The first 20 minutes after restart, Regular EPs will only accept jobs that request at least 4 cores or half the disk on the EP
  - The first 20 minutes after restart, High Memory EPs will only accept jobs that request 100 Gbyte or more of memory
  - Users with these hard to match jobs get preferential treatment during an upgrade (may have to reset expectations)
  -

Repositories

(for 23.0, 23.x, 24.0, 24.x)

- **Snapshot**: Latest build that passed unit tests → OSG development
  - Tested daily with OSG VMU tests (integration tests with HTCondor-CE)
  - Build number increments
- **Alpha**: Release Candidate in Testing on CHTC
  - Current and Previous version release candidates
  - Build number increments
- **Beta**: Release Candidate in Testing on OSPool → OSG testing
  - Current and Previous version release candidates
  - Patch number increments
- **Release**: General Availability → OSG release
  - All versions (trimmed back to latest version when out of support)

CHTC

But before you start any upgrade…

But before you start any upgrade…

Run

$ condor_upgrade_check

Thank you and Questions?

This work is supported by NSF under Cooperative Agreement OAC-2030508 as part of the PATh Project. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.