Job Isolation

Greg Thain
Center for High Throughput Computing
University of Wisconsin - Madison

Outline

- Why put contain jobs?

- Ersatz HTCondor containment
- Docker containers
- Singularity containers

3 Protections

1) Protect the machine from the job.

2) Protect the job from the machine.

3) Protect one job from another.

The ideal container

- Allows nesting
- Need not require root
- Can’t be broken out of
- Portable to all OSes
- Allows full management:
  - Creation // Destruction
  - Monitoring
  - Limiting

Resources a job can (ab)use

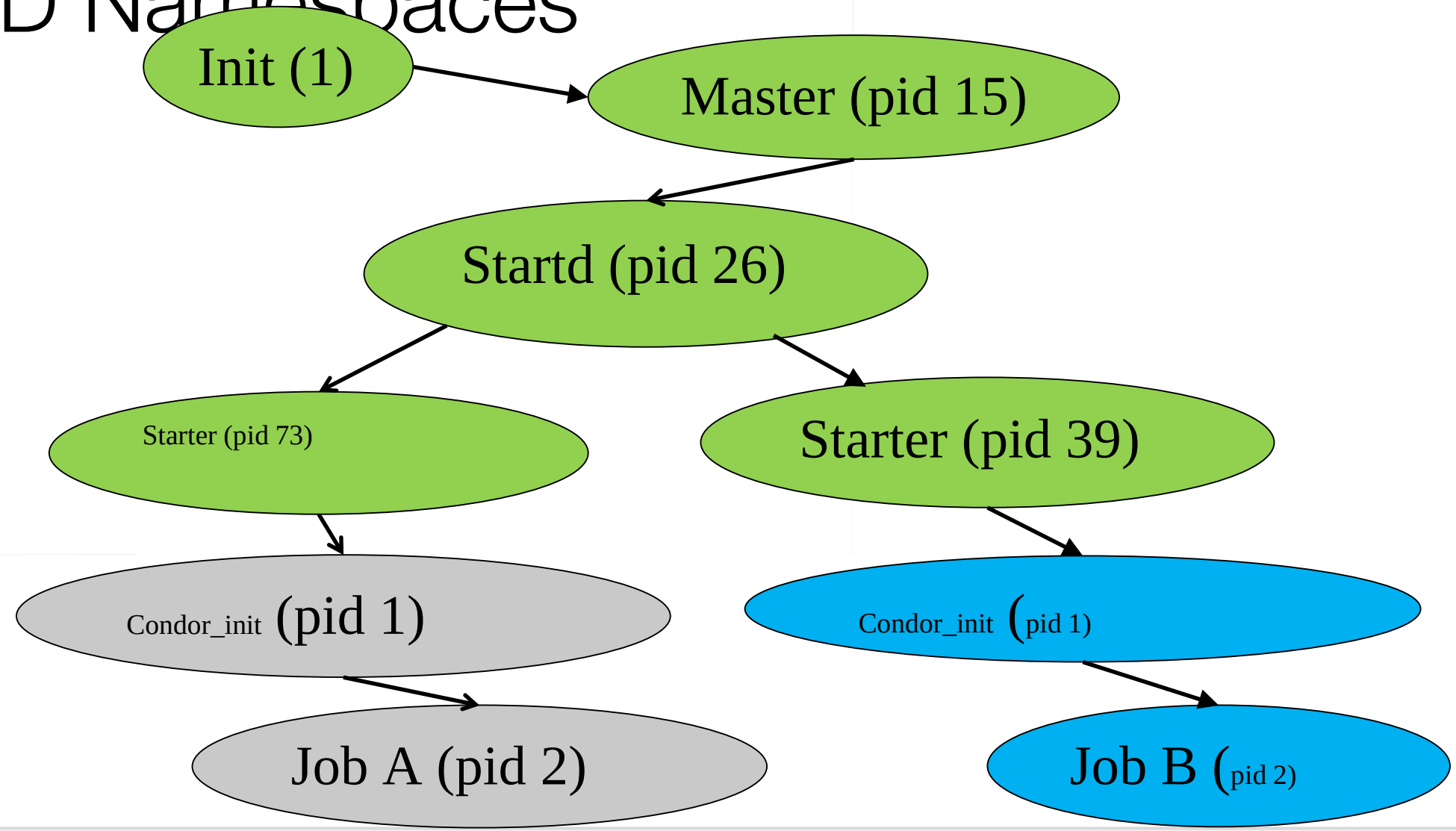
- CPU
- Memory
- Disk
- Network
- Signals
- L1-2-3 cache

HTCondor’s containment

PID namespaces

- You can’t kill what you can’t see
- Requirements:
  - RHEL 6 or later
  - `USE_PID_NAMESPACES = true`
    - (off by default)
  - Must be root (in HTCondor's implementation – future work here)

PID Namespaces



MOUNT_UNDER_SCRATCH

- Or, “Shared subtrees”
- Goal: protect /tmp from shared jobs
- Requires
  - Condor 8.0+
  - RHEL 5
  - HTCondor must be running as root
  - MOUNT_UNDER_SCRATCH = /tmp,/var/tmp

MOUNT_UNDER_SCRATCH

```
MOUNT_UNDER_SCRATCH=/tmp,/var/tmp
```

Each job sees private /tmp, /var/tmp

Downsides:

- No sharing of files in /tmp

Control Groups v1
aka “cgroups”

- Two basic kernel abstractions:

1) nested groups of processes

2) “controllers” which limit resources

Control Cgroup setup

- Implemented as filesystem
  - Mounted on /sys/fs/cgroup,
  - Groups are ***per controller***
    - E.g. /sys/fs/cgroup/memory/my_group
    - /sys/fs/cgroup/cpu/my_group
  - Interesting contents of virtual groups:
    - /sys/fs/cgroup/memory/my_group/tasks
  - Condor default is
    - /sys/fs/cgroup/<controller>/htcondor
  - Compare with systemd's slices

Cgroup controllers

- Cpu
  - Allows fractional cpu limits
- Memory
  - Need to limit swap also or else…
- … any many others

Enabling cgroups

- Requires:
  - RHEL7
  - HTCondor 8.0+
  - Rootly condor

- And… condor_master takes care of the rest

Cgroups with HTCondor

- Starter puts each job into own cgroup
  - Named exec_dir + job id
- Procd monitors
  - Procd kills atomically all procd within a job
- CPUS attr * 100 > cpu.shares
- MEMORY attr into memory controller
- CGROUP_MEMORY_LIMIT_POLICY
  - Hard or soft
  - Job goes on hold with specific message

Enter cgroup v2

- Lots of problems with v1:
- Independent controllers are difficult to reason about
- Some controllers don't work well – i.e. "soft" vs "hard" memory
- El9 gets v2, not backwardly compatible!
- Hope to get this in by end of year!

New in 10: STARTD_ENFORCE_DISK_LIMITS

- The scratch dir is created at runtime, and sized by
  - RequestDisk
- If job uses more than RequestDisk, goes on hold

- Cleaning up scratch dir is quick – one system call
- Monitoring scratch dir is quick – one system call

What backs the file system?

- Either "thinpool" lvm blocks
  - Requires SysAdmin to set up lvm
  - THINPOOL_NAME = htcondor_lv
  - THINPOOL_VOLUME_GROUP_NAME = htcondor_vg
- Or "thick pool"
  - No other knobs needed

Questions?

Thank you!

Thank you and questions

Thank you – Questions?

This work is supported by the NSF under Cooperative Agreement OAC-2030508. Any options, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.