CERN

HTCondor – Production Experience

Ben Jones – IT-CM-IS

Background

- HTCondor introduced as new production batch service
- Replaces LSF, a proprietary product, with an Open Source product
- HTCondor now has more than double the capacity of LSF
  - 100k+ cores in HTCondor
  - 46k cores in LSF
- We haven’t started reducing LSF in anger (yet!)

Big picture - Scale

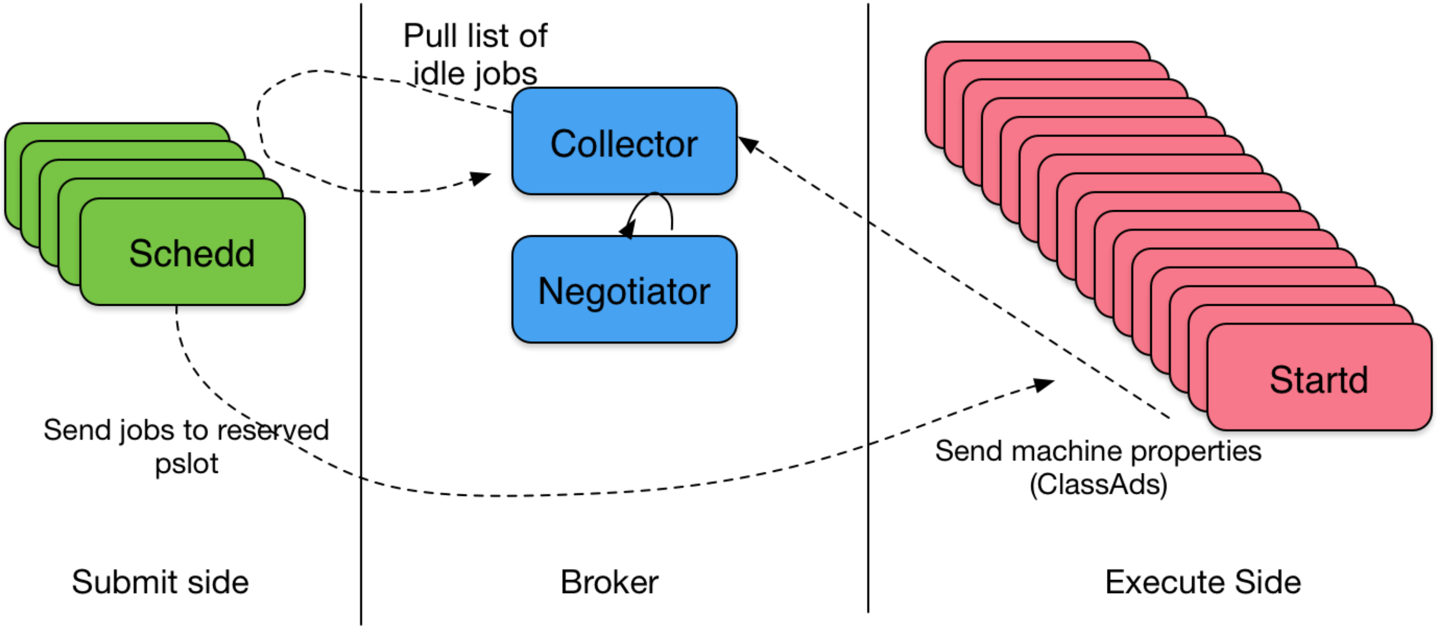
- LSF has a fixed maximum capacity of ~5k hosts
  - Due to limitation, LSF worker nodes are bigger (typically 16 core)
  - We know that “virtualisation overhead” for 16 core is ~3% whereas it’s negligible for 8 core
- HTCondor 100k+ cores with 8/10 core machines would be impossible with LSF
- CMS global pool bigger HTCondor scale, but we have different requirements (local kerberos submissions)

Reported Issues from wiki

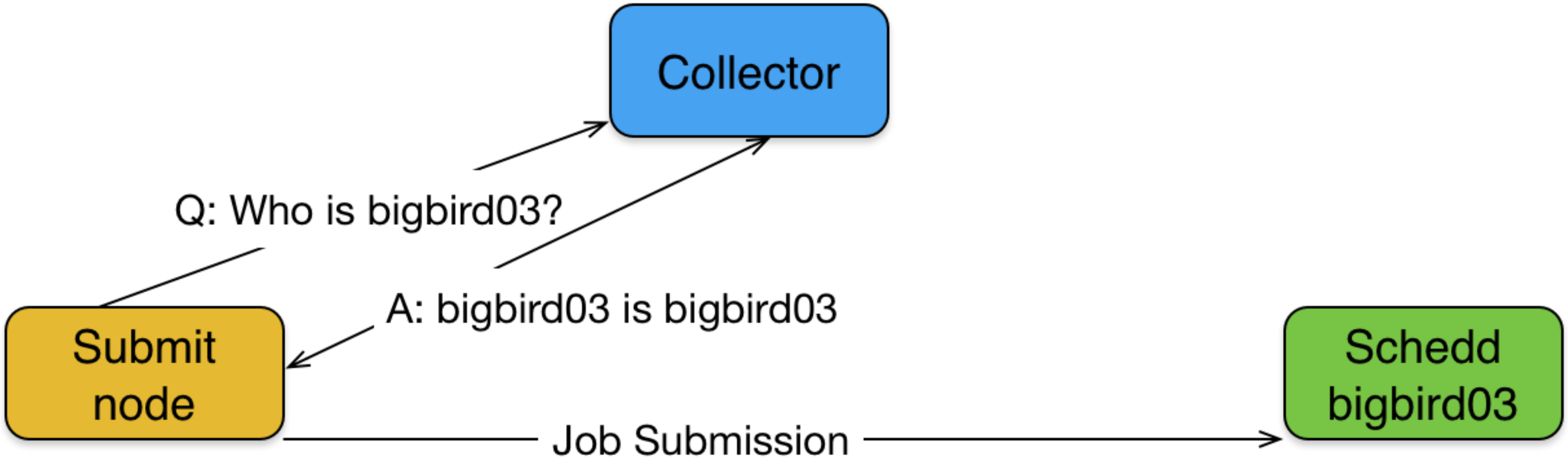
- Scheduler errors:
  - Scheduler not answering or taking long time to answer, submission fails with "ERROR: Can't find address of local schedd"
  - Jobs put on hold for node errors copying files
- Job removal issues
  - Jobs disappearing from the queue short after the expected completion without being explicitly removed
- I/O Issues
  - Submissions taking long time in particular in combination with data written in EOS(worse) or AFS(better, but still visible)
  - Jobs having a large variation in completion time when involving I/O with EOS(worse) or AFS(better, but still visible)
  - Jobs failing rate in the order of 10% when involving I/O with EOS(worse) or AFS(better, but still visible)

How HTCondor works…
[as it relates to these issues]

Symmetric job matching



Submission requires Collector



- Why query the Name if it’s the same as the Schedd fqdn?
    - Because it isn’t always – HA schedds publish names

Other submission tasks

- An AP-Req is generated at time of condor_submit
- The AP-Req is transmitted to the Schedd
- The AP-Req is turned into a kerberos & AFS token
- The Schedd writes to 3 files: log, stdout, stderr
  - In the typical case, AFS tokens are needed for this

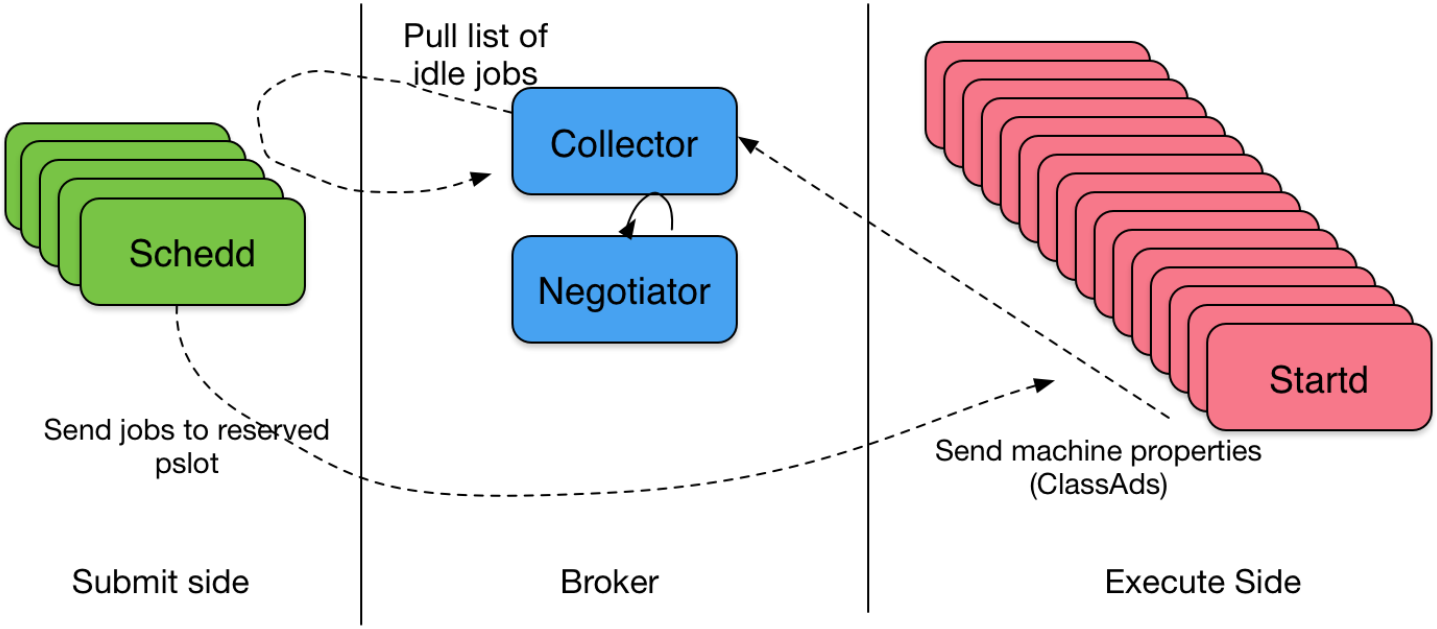
ERROR: Can't find address of local schedd

- The Schedd registers with the Collector
- condor_submit queries the Collector for the schedd
- There are therefore two potential issues:
  - The Schedd is too busy to update the Collector
  - The Collector is too busy to respond to the query
- The good news is we’ve been fixing both!

“The scheduler is being upgraded”

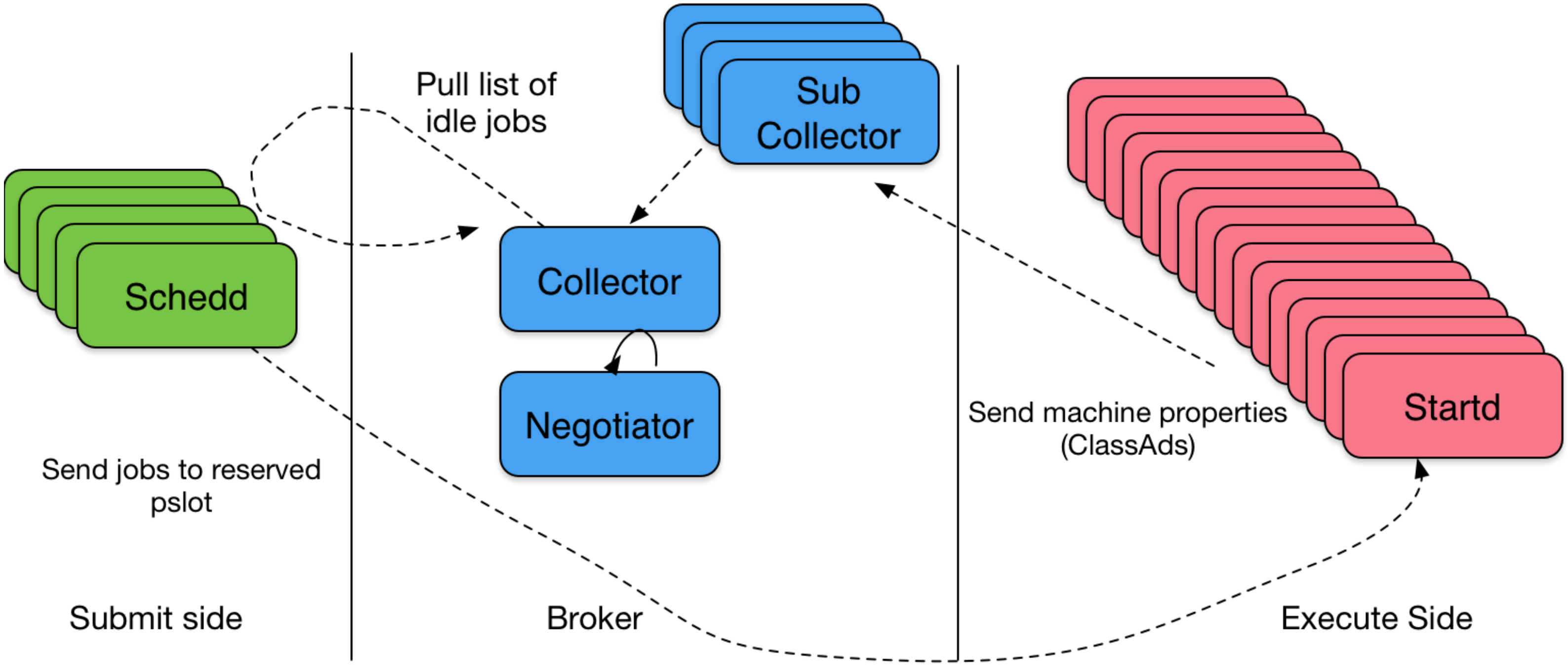
- The Scheduler and the Collector have been upgraded
- Scheduler has fixes for AFS token handling
- Collector has fixes to prioritize queries from infrastructure
  - CMS Global pool encountered same issue (though with smaller pool)
  - Development release (8.7.*) contains relevant `COLLECTOR_QUERY_WORKERS_RESERVE_FOR_HIGH_PRIO`

Collector Bottleneck



~14K Startd
…and increasing!

Split the Collectors



Splitting infrastructure

- Moving to sub collectors has reduced the times when the Collector is too busy to reply with the name of the schedd
- Still work to do! The next step is to scale out the Negotiator
  - Negotiator does the matching of jobs to machines
  - Long negotiation cycle also affects the Collector
  - Splitting pool between two negotiators

Jobs on hold for file transfer

- Where source of file (for eg: executable) is in AFS, we require tokens
- Two infrastructure reasons for transfer failures:
  - token expires before execution, and ngauth error in reacquisition – should be rare due to retries
  - AFS token lost by Schedd
    - AFS (as opposed to kerberos) token is stored in kernel keychain,not on disk. Previous HTCondor version could occasionally lose the keychain – fixed in version installed Aug 10

Jobs deleted shortly after submission

- Reasons we put jobs on hold / kill:
  - MaxRuntime expires
  - Job exceeds memory and there is memory pressure
- MaxRuntime set either by:
  - +MaxRuntime = <int>
  - +JobFlavour = ”testmatch”
- Memory slightly more complicated
  - 1 cpu = 2gb RAM
  - We rewrite requests that exceed this ratio
  - An unset value should be ”1” but we have seen cases where this isn’t the case: hence workaround to specify “request_memory = 2000”

I/O Issues

- Many of the reported issues aren’t HTCondor specific (EOS/AFS are the same speed on any batch system)
- 10% error rate reported issue due to the credential refresh on the worker nodes.
- Scaling out the ngauth service required config changes & process restarts on worker nodes to support alias

Addenda

- -spool to condor_submit means that file transfer mechanism is used between submit machine & schedd
  - Useful if you don’t need 30k files written to your AFS homedir!
- htcondor on the desktop
  - not supported by LXBATCH but encouraged by EOS?
    - True that we don’t support desktop. Any interest in a Docker?
    - The IT department continues to support and recommend LXPLUS
- “For scheduler problems log into a different LXPLUS machine”
  - This won’t help – you are mapped to a schedd (also why it might not help on a desktop either)

Questions?