Qube 6.10-1 Release Notes

##

@RELEASE: 6.10-1

##
==== CL 20025 ====

@FIX: extremely inaccurate cumulative cpu time for agenda items

JIRA: QUBE-3375
ZD: 18841

==== CL 20014 ====
@FIX: worker is added to the worker_dim dimension table as many times as there are expired entries for that same worker

==== CL 19931 ====
@FIX:Fix relative movie paths in images_to_move.py

==== CL 19897 ====
@FIX: auto_remove worker flag missing from worker config dialogs

==== CL 19834 ====
@FIX: supe/worker RPMs should be installable onto a system with core of the same major.minor mode installed

JIRA: QUBE-3332

==== CL 19478 ====
@FIX: workers are always "auto-remove"d, even if "auto_remove" is not set in worker_flags.

ZD: 18512
JIRA: QUBE-3174

==== CL 19475 ====
@FIX: issue where instances would be stuck in "QB_PREEMPT_MODE_FAIL", causing the supervisor to tell instances to "wait and retry later" in response to retryWork() indefinitely.

Issue was caused when the preemptJobNetwork() routine determines that the
instance has started but has NOT yet started working on an agenda item, in
which case it would mark the QB_PREEMPT_MODE_FAIL in order to interrupt
(i.e. aggressively preempt) the instance; However, the interrupt was not
being triggered properly.

Issue was apparently introduced in CL19126.

==== CL 19436 ====
@FIX: "down" workers not always detected properly

JIRA: QUBE-3155
ZD: 18425

==== CL 19425 ====
@FIX: issue when supe thread doesn't hear back from worker during a dispatch. Related to CL19243.

Also fixed an issue (probably harmless) where an extra call to queue.releaseJob() was sometimes made in the findSubjobAndReserveJob() method.

==== CL 19263 ====
@FIX: log directories for jobs submitted after the utility has been started but before the orphaned log removal is begun are erroneously removed

==== CL 19258 ====
@FIX: not running --use-frm when first-pass repair fails when message has different line-endings than OS X

==== CL 19243 ====
@FIX: add code to avoid mixed-up job instance status when worker-supervisor communications are dropped during job dispatch on an intermittently unreliable network

It was found that network hiccups can cause a worker to not respond to the
supervisor during the dispatch of a job instance, but still start running
the instance anyway. The worker would send the "running" instance report to
the supervisor, which is processed by a separate thread, which updates the
DB, causing a status mix-up.

Added code to detect such situations, and allowed the system to let the job
run (instead of force-removing it from the duty table) on the worker in

question.

Also added error-checking code on the worker side-- if worker detects that
it couldn't respond to the supe for a dispatch order, it will give up on
that job and release resources that it had just reserved for it.

ZD: 17868

==== CL 19236 ====
@FIX: jobs submitted by non-admin user without a specified priority attempt to submit at priority -1

JIRA: QUBE-3015

==== CL 19209 ====
@FIX: "down" workers would not be detected properly by the supervisor even when the supervisor_heartbeat_timeout expired.

ZD: 18057
JIRA: QUBE-3018

==== CL 19178 ====
@FIX: timing issue causing workers to get stuck with job instances.

Issue was seen on a very busy farm with intermittently drops in network
communications, when many supe threads would try to dispatch a single
instance at the same time.

ZD: 17868

==== CL 19163 ====
@FIX: fix an issue where a worker can sometimes get stuck with a job instance that it's not running any longer

* Issue was seen when job instances are migrated and there are intermittent
networking issues between the supe and worker causing job updates to NO
come thur in an expected, orderly fashion.

ZD: 17868

==== CL 19126 ====
@FIX: on a network with intermittent worker-supe commnuication issues, bad timing can cause job instances to get stuck in "running" state

* In a bunch of routines that handle job-command executions (i.e., migrate,
kill, etc.) in QbSupervisorCommand, add code to do one last check when a
worker is unreachable, to see if the instance still belongs to the worker
before updating the instance on DB. It was found that, since a thread
dealing with down workers can spend quite a long time, sometimes
instances that a worker was processing can be moved off of it and the DB
updated by another thread (for example, assigned and running on another
worker)-- the check is designed to prevent our thread from overwriting
such updates.

ZD: 17868

==== CL 19121 ====

@FIX: job instances cane get into an odd state when dispatch routine doesn't hear back from the worker ("found dead").

Networking hiccups can cause this communication drop, which in turn may
cause job instances to be "stuck" in the running state on a worker, and be
unkillable.

ZD: 17868

==== CL 19118 ====
@FIX: Systemctl unit files for worker and supervisor not installed into correct location

==== CL 19109 ====
@FIX: optimize job cleanup script
@CHANGE: only scan log directories if log removal necessary
@CHANGE: removal of large number of orphaned log directories does not require skipping sanity checks

==== CL 18985 ====
@FIX: 'No database selected' MySQL error when removing ghost jobs
ZD: 17882

==== CL 18351 ====
@CHANGE: background helper thread improvements

* limit the number of workers that are potentially recontacted by the background helper routine to 50 per iteration.

* background thread exits and refreshes after running for approximately 1 hour, as opposed to 24 hours

ZD: 17124