Testing Red Hat Kernels

Present and Future

Michael Hofmann
Israel Santana Alemán
Bruno Goncalves
Tales Lelo da Aparecida

Overview

- introduction
- the present
- the future

introduction

about CKI

the "other" kernel QE team

- CKI: Continuous Kernel Integration
  - home page and documentation: https://cki-project.org
  - code: https://gitlab.com/cki-project
  - internal Slack: #team-kernel-cki
  - mixed team of ~10 people: 4 QE, 4 Dev, 1 manager, 1 tech lead
- mission:
  - what: prevent bugs from being merged into kernel trees
  - how: shift kernel testing as far left as possible

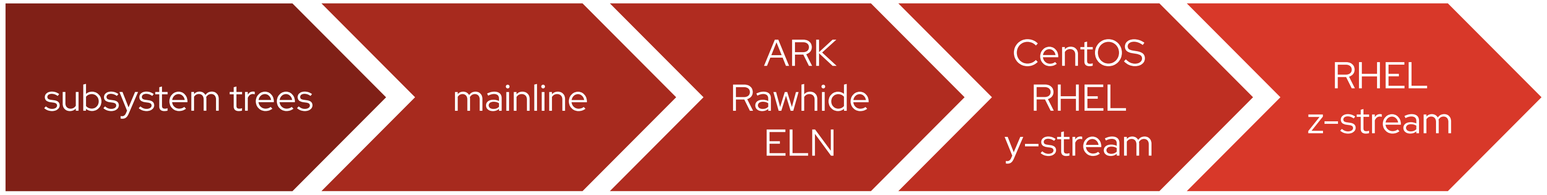
mission and reality

- what we do:
  - provide CI-as-a-service for src-git RH kernel devel workflow
  - test upstream git trees
  - host internal kernel-related infrastructure
- main "product": RH kernel development workflow GitLab CI pipeline
  - provide a fast inner development loop via GitLab merge requests (MRs)
  - build (AWS): ~300 hours/workday
  - test (Beaker): we don't want to know, but one of the biggest users

the kernel is special

no obviously it is (really!)

- "interesting" code flow

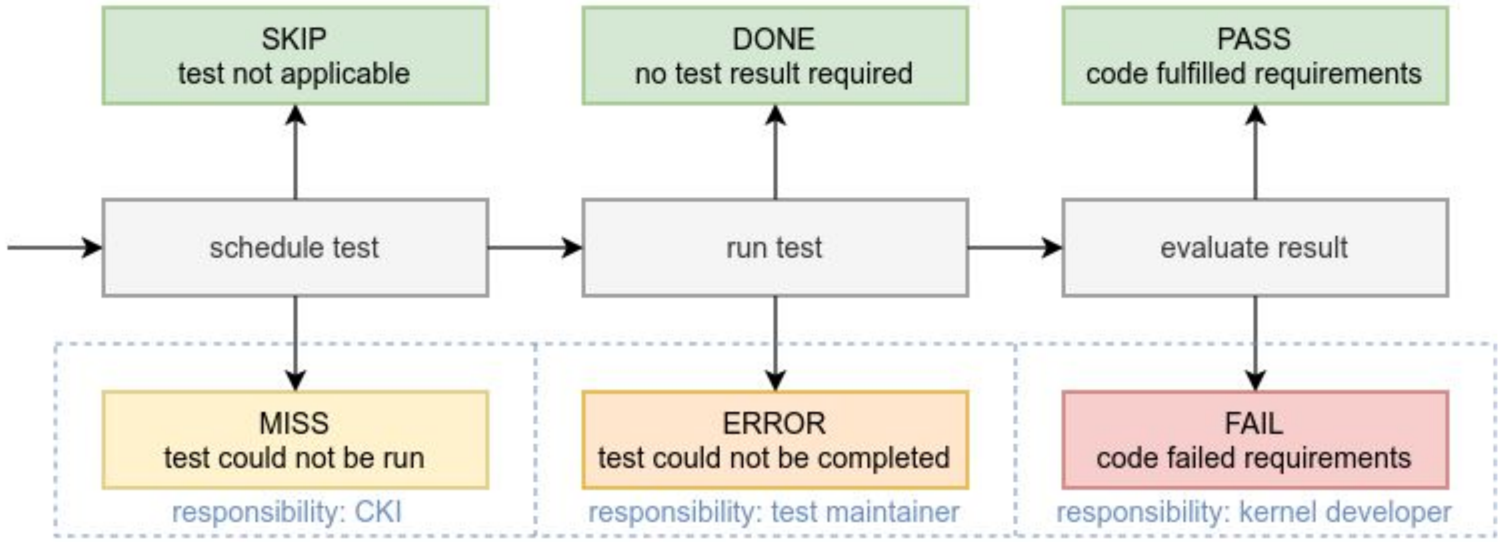


  - upstream subsystem trees and mainline
  - Always Ready Kernel (ARK) for Rawhide and RHEL+1 (ELN)
  - CentOS Stream and RHEL y-streams and z-streams
- dozens of separate test projects/frameworks
  - nearly all of them live outside the kernel tree
- testing on VMs is not good enough

reasons for test troubles

whose fault is it

- testing kernels on real hardware is annoyingly hard
- Blame Allocation Matrix for a test run:



what to do with "real" test failures

anybody said "waiving"?

- what to do depends on the reason behind the test failure
  - the MR or RPM is broken: block the change, and get it fixed
  - already present before: track it (Jira), fix asynchronously
- while waiting for asynchronous fixes:
  - selective waiving of failing tests until issue is fixed
  - automated via deterministic "known issue detection"
  - regular expressions and log files

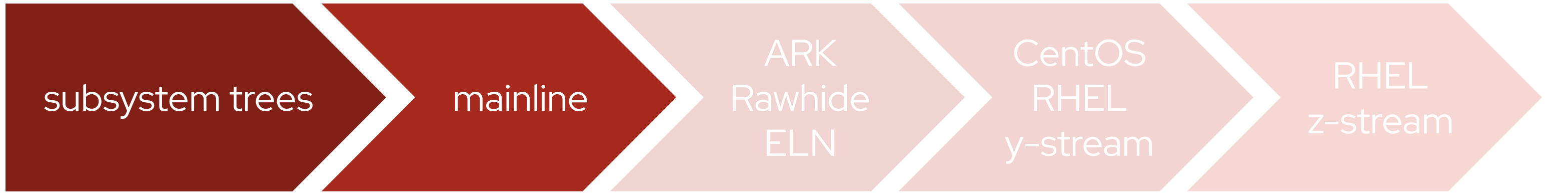
[shift] [kernel testing] [as far left as possible]

text and meaning

- shift:
  - **add** additional testing on the left
  - **keep** testing to the right to catch weird integration issues
- kernel testing:
  - **which** parts of QE test plans to run **where** on the left
- how far left is constrained by buy-in from two groups of people:
  - **developer** buy-in for caring about test results
  - **QE** buy-in for maintaining test code and checking test results
  - shifting to the left needs to happen **step-wise**

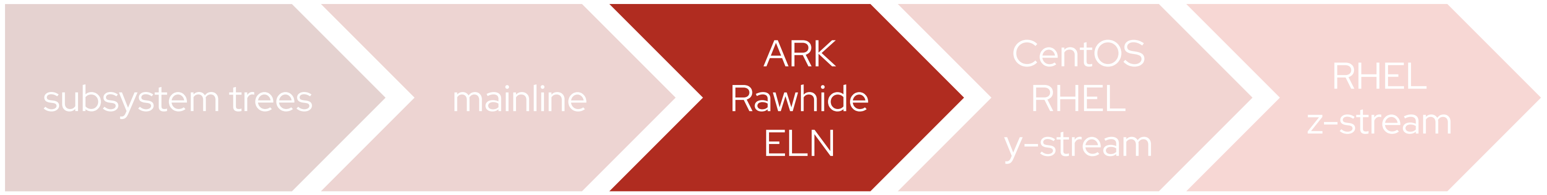
the present

CKI testing: upstream



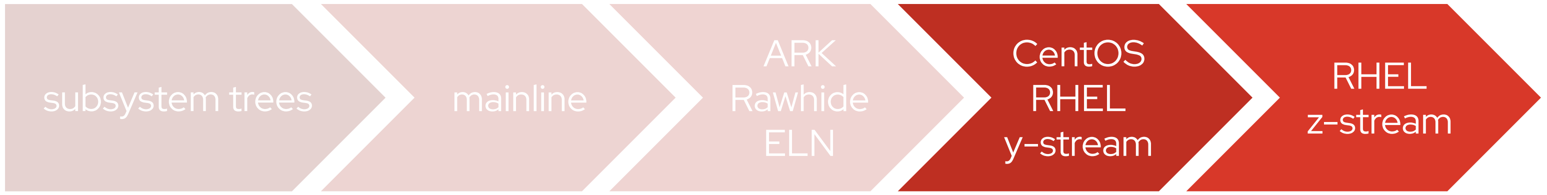
- for subsystem/mainline git trees referenced in pipeline-data
- running tests indexed by test sets in kpet-db (`sets`)
- results available in Web GUI of DataWarehouse
- test summaries reported via email

CKI testing: Rawhide/ELN/Fedora



- Always Ready Kernel (ARK) for Rawhide and RHEL+1 (ELN)
  - closely tracks mainline, separate Fedora release branches
- **src-git**: CI pipelines in kernel-ark MRs for Rawhide/ELN
  - stay ready for next Fedora release and major RHEL cycle
- **dist-git**: Fedora/Rawhide/ELN Koji builds tested for `kt1` test set
  - no gating, but test summaries reported via email

RHEL: three levels of testing



- **src-git**: before a change is merged
  - inner feedback loop for kernel development workflow (KWF)
  - find issues caused by code changes in the MR
  - stable subset of QE-maintained kernel tests specific to code change
- **dist-git**: before a kernel RPM is tagged into integration compose
  - prevent breaking of the compose because of integration issues
- **composes**: regression testing of complete composes
  - find and track regressions and weird issues in Jira

the future

harmonize CKI and kernel QE workflows

- integration of QE pipelines into kernel development workflow
  - selectively trigger QE Jenkins pipelines in MRs
  - feed test results back into MRs, and allow to gate on them
- enable consistent automatic waiving
  - for both CKI and QE Jenkins pipelines
  - at the src-git, RPM and compose level
  - needs all test results and logs in DataWarehouse

CKI test audit for shared Devel/QE understanding

- developer viewpoint:
  - run only tests specific to subsystem under change
  - only what developers would run locally on their machine
- QE viewpoint:
  - run all tests likely to catch issues
  - e.g. xfstests configured for cifs on all networking changes…
- management goals:
  - points of contact/approval for Devel/QE for each test case run
  - accountability for current state/changes of what to run when

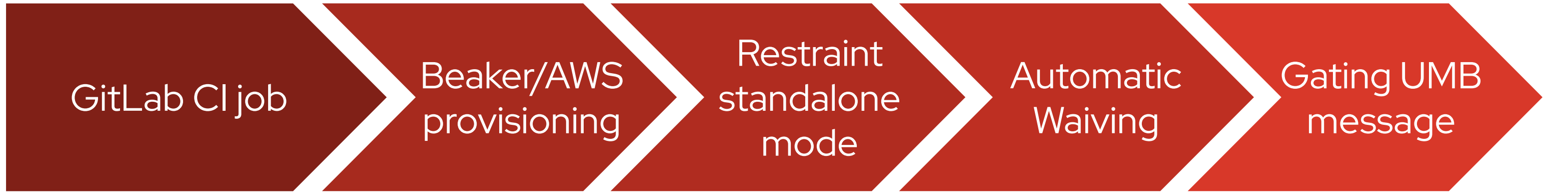
adopt Shared OS Testing Infrastructure

- in a nutshell: all tests should run via Testing Farm
- Koji/Brew RPM testing/gating via static tmt test plans in dist-git
- testing of merge requests via dynamic tmt test plans
- more ideas:
  - reverse dependency testing?
  - QE S3 artifact storage, ReportPortal, Polarion, …

Testing Farm: how to get there

make sure nobody notices the surgery on the low-level plumbing

- currently:

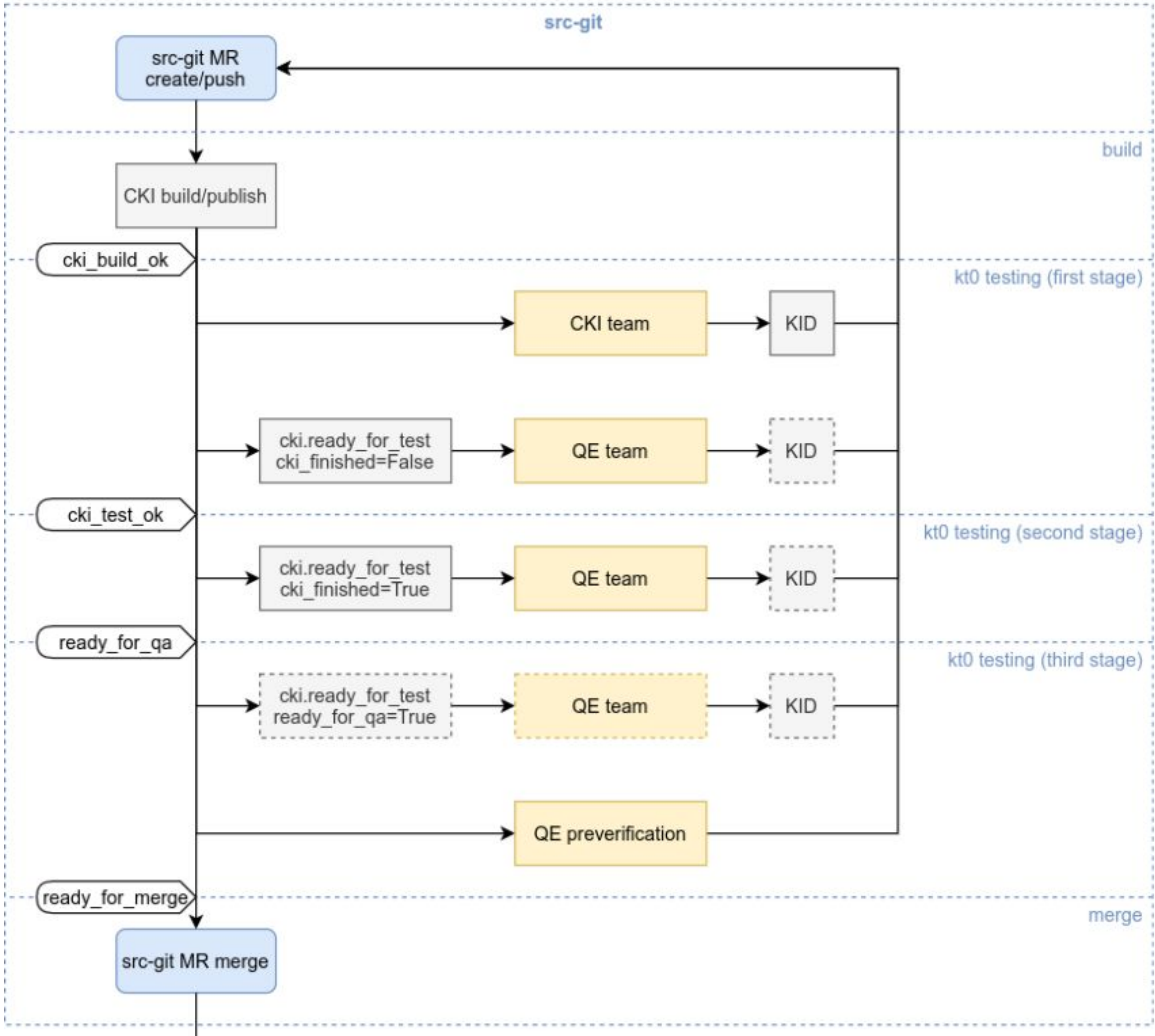


- stepwise migration to Testing Farm:
  - provision machines via Testing Farm using both Beaker + VMs
  - run restraint-based tests via equivalent tmt test plan
  - automatic waiving after results are available in OSCI dashboard

🤗 Question time 🤗

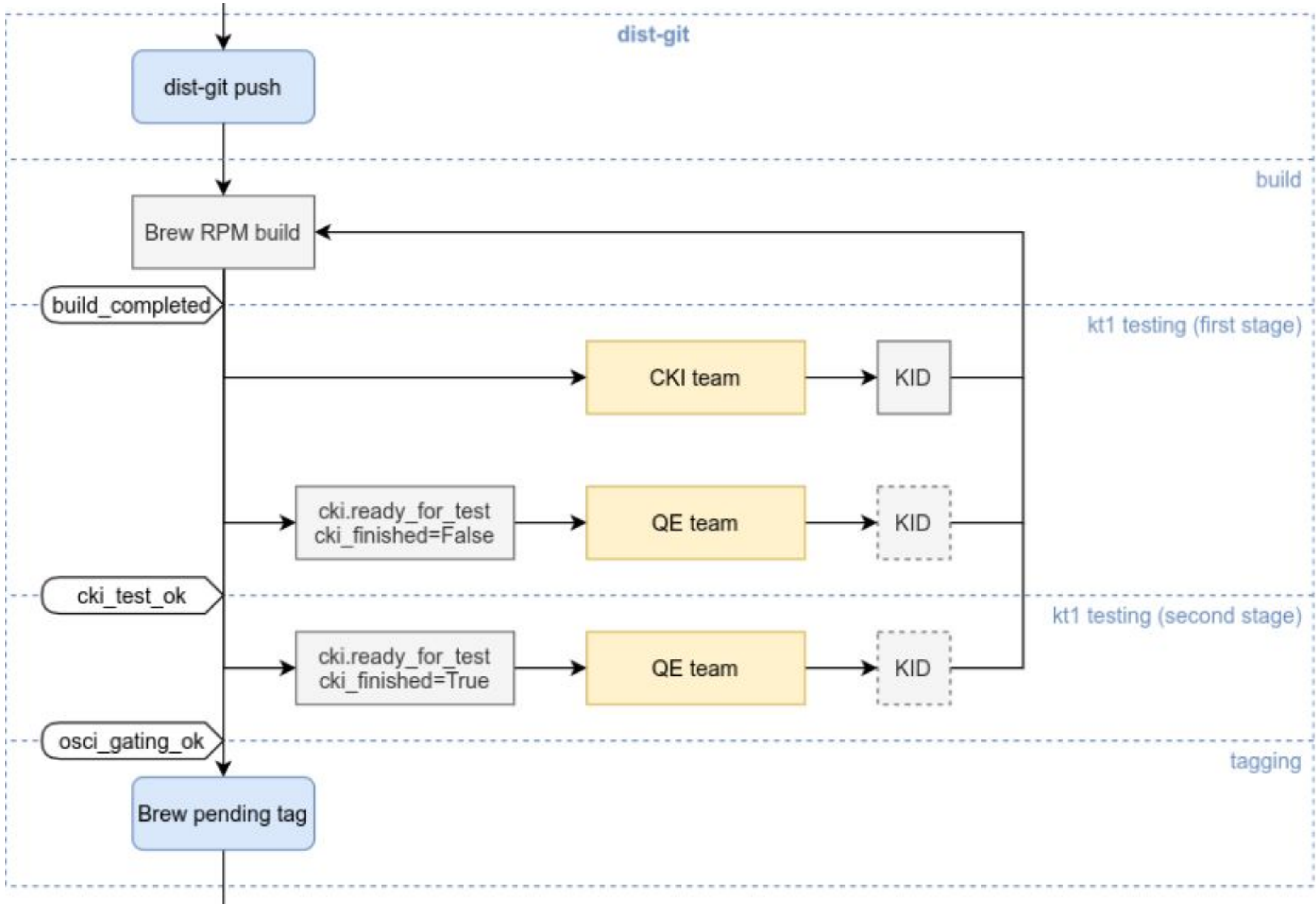
RHEL: src-git testing

- current testing providers:
  - CKI
  - LNST (manual)
- current QE workflow support:
  - UMB triggers (but unused)
- missing QE workflow support:
  - known issue detection (KID)
  - feeding results into MRs


src-git
src-git MR create/push
build
CKI build/publish
cki_build_ok
kt0 testing (first stage)
CKI team
KID
cki.ready_for_test cki_finished=False
QE team
KID
cki_test_ok
kt0 testing (second stage)
cki.ready_for_test cki_finished=True
QE team
KID
ready_for_qa
kt0 testing (third stage)
cki.ready_for_test ready_for_qa=True
QE team
KID
QE preverification
ready_for_merge
merge
src-git MR merge

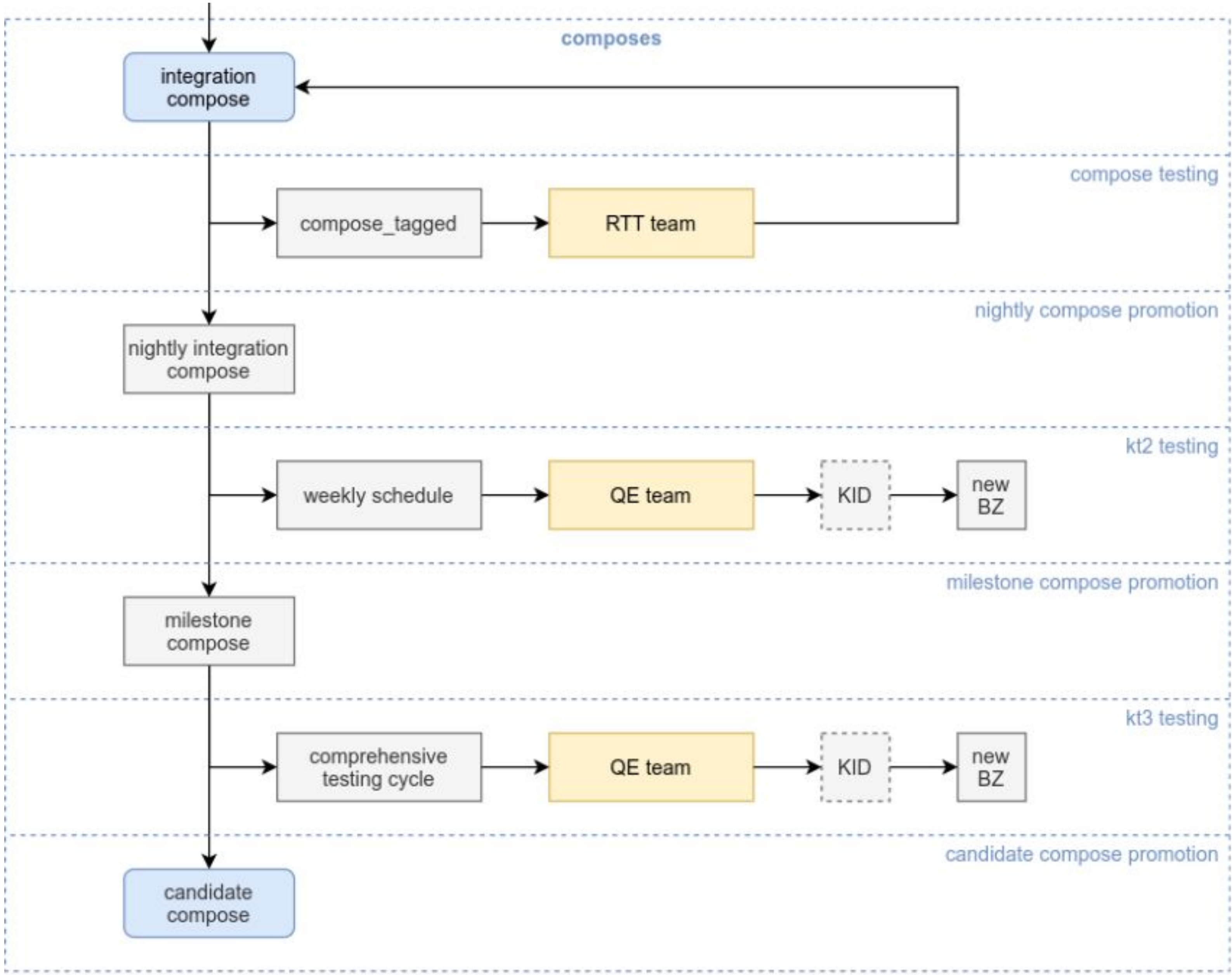
RHEL: dist-git testing

- current testing providers:
  - CKI (gating)
  - cloud boot (gating)
  - some QE (not gating)
- existing QE workflow support:
  - UMB triggers (by Brew/CKI)
- missing QE workflow support:
  - known issue detection
  - most QE testing is not gating



RHEL: compose testing

- current testing providers:
  - RTT qualification
  - QE teams
- missing QE workflow support:
  - known issue detection


composes
integration compose
compose testing
compose_tagged
RTT team
nightly compose promotion
nightly integration compose
kt2 testing
weekly schedule
QE team
KID
new BZ
milestone compose promotion
milestone compose
kt3 testing
comprehensive testing cycle
QE team
KID
new BZ
candidate compose promotion
candidate compose