These are the Red Hat Kernel Engineering Guidelines and 'Best Practices' for commits and Merge Requests (MRs) Titles and Descriptions for the Red Hat Linux Kernels, aka “Commit Rules”. Following these instructions will result in your changesets passing Red Hat's commit verification webhooks, and reduce the number of questions from reviewers, submaintainers, and maintainers about your proposed changes. If you have questions about this document or the procedure used to submit changes, file an issue on the GitLab project that you are contributing to.

Following the upstream kernel patch guidelines will satisfy the general construction of commits, but there are additional rules specific to Red Hat to assist in changeset tracking, accounting, and reviewing. Other than the differences noted in this document, following the upstream process is generally accepted as good practice.

This document assumes that the reader is comfortable with the concepts of git, GitLab, commit and patch generation, and has a general knowledge of the upstream linux.git's Documentation/SubmittingPatches. The explanation of these concepts are not within the scope of this document.

Red Hat uses GitLab to manage the patch workflow, and all changes must be submitted as GitLab merge requests. Contributors can use the GitLab UI to submit merge requests. Red Hat recommends using the 'lab' and 'bichon' command line interface open source tools when working with merge requests in the RHEL kernel trees.

Example Commit Log

The following is an example of a valid commit title & description. The first column lists the sections below which describe in detail what is needed:

1 subsystem: include missing kernel.h header to fix compile

2a Bugzilla: <bugzilla reference>
2c Upstream Status: <if not linux.git provide upstream tree, or mark as RHEL specific>
2e CVE: <CVE number if required>
2f Conflicts: <if required>

3 commit 7ff4beb311dfab4f18ff2cd64f78scd9296a39a
3 Author: John Upstream <john.upstream@someupstreamplace.com>
3 Date: Mon Mar 28 14:29:19 2011 -0700
subsystem: include missing kernel.h header to fix compile
This is needed to compile this module. It was missing from the original commit.

Signed-off-by: John Upstream <john.upstream@someupstreamplace.com>
Cc:: Jane Upstream <jane@someotherplaceupstream.com>

v2: Some information about the changes between v1 and v2

Signed-off-by: Shadow Man <shadow.man@redhat.com>

Example Merge Request

The following is an example of a valid Merge Request title and description. The first column lists the sections below which describe in detail what is needed:

subsystem: a general explanation of the changeset

Bugzilla: <bugzilla reference>
Tested: <testing information>
Upstream Status: <if not linux.git provide upstream tree, or mark as RHEL specific>
Depends: <bugzilla reference if required>
CVE: <CVE number if required>
Conflicts: <if required>

An explanation of what the problem the changeset resolves, how the changeset resolves the problem, additional testing details, etc.

v2: Some information about the changes between v1 and v2

Signed-off-by: Shadow Man <shadow.man@redhat.com>
Cc: Fedora Friend <fedora.friend@redhat.com>

Section Descriptions

The following sections provide the format that developers should follow in order to help both engineers review changesets and maintainers process commits and MRs as quickly and as efficiently as possible.
1. Title

For commits, use the upstream commit title. In general, for Merge Requests, use the upstream format of `<arch,subsystem>: <summary>`. Copying upstream commit subject lines is acceptable and commonplace; however, the title should attempt to capture what the commit does and not what the commit fixes. Not all upstream summaries are sufficient; for example “treewide” commits may be altered to indicate the specific section of code they are modifying.

Commits that are unique to RHEL (commonly referred to as RHEL-only commits) in a changeset must have a unique commit description that identifies it from other commits. Duplicate commit titles usually cause confusion for reviewers and maintainers alike. As is the case with upstream, the title must contain a descriptive explanation of the entire changeset.

2. Description

In order to handle and prioritize the large number of MRs submitted to the Red Hat Kernel project, the description must contain metadata necessary to assist in integrating changesets into official kernel releases. The metadata section contains a required bugzilla reference, testing details, and some additional metadata that may be required depending on the circumstances surrounding the commit or MR.

   a. Bugzilla Reference (required)

Each commit in a changeset and the MR description must reference at least one Red Hat bugzilla URL on a line beginning with “Bugzilla: https://bugzilla.redhat.com/”. If the MR resolves multiple bugzillas, the description must include only one bugzilla URL per “Bugzilla:” line. **Bugzilla IDs (ie, just a number) are not acceptable.**

Changesets that are lacking a bugzilla number will not be committed and will be immediately flagged by the project’s webhooks. Every changeset needs to be coordinated with other groups at Red Hat (management, QE, Documentation, etc.) and the coordination is handled through bugzilla flags and state. If the bugzilla flags and state are appropriately set on a bugzilla, then and only then, will the associated changeset be considered for inclusion into the project by the maintainers.

Good examples for bugzilla URLs in the description:

   Bugzilla: https://bugzilla.redhat.com/show_bug.cgi?id=783023
   Bugzilla: https://bugzilla.redhat.com/783023
b. Testing Details (required)

The MR description must include information about how the changeset was tested. This information can include links to test results in a bugzilla or a text description of the results.

c. Upstream Status (required)

If the upstream source tree is not linux.git, the description must include a URL for the upstream source tree. Commits with an identified source must also specify a commit sha1 in their description.

Linus' linux.git is considered to be the tree referred to when the term "upstream" is used. If the changeset is not in the linux.git tree, please specify the URL of the source tree, a URL to the patch posting on a mailing list, or explain why the changeset is unique to RHEL.

Non-upstream change tracking is useful for Engineering teams monitoring current, future, and update releases, therefore changesets that are not in an upstream tree and are unique to RHEL must have upstream status in the format "RHELX.Y.Z only". "RHEL only" must be used for changesets that must be applied to future versions of RHEL. Changes that will only be applied to a specific major release must have status "RHELX only", and changesets in a minor release that are temporary workarounds must be specified in the format "RHELX.Y only". Temporary workaround changesets can also include additional information about future work in parenthesis. Unique z-stream fixes must be specified with upstream status as "RHELX.Y.Z only".

Changes that have been sent upstream but have not been included in an upstream repo must have status “Posted” followed by a link to the upstream post, merge request, or pull request.

Good examples of Upstream Status for changesets that are not in Linus' linux.git:

- Upstream Status: https://git.kernel.org/pub/scm/linux/kernel/git/next/linux-next.git
- Upstream Status: RHEL only
- Upstream Status: RHEL7 only
- Upstream Status: RHEL8.1 only (to be removed in BZ 123456)
- Upstream Status: Posted https://lore.kernel.org/lkml/87ft4djtyp.fsf@nanos.tec.linutronix.de/

d. Changeset Dependencies (optional)

If the changeset has dependencies on other bugzillas, the MR description must include a mention of those bugzillas. Use one 'Depends:' line for each dependent bugzilla. Follow the above 'Bugzilla:' guidelines for 'Depends:' lines (ie. Bugzillas must be specified as URLs and not IDs). Missing dependencies and malformed 'Depends:' lines will be identified by the project’s webhooks.
The maintainers understand that conflicts with other patches will happen. When possible, please coordinate with others before posting. If a commit or MR has an explicit dependency on another MR, be careful to include that dependency so the maintainers can coordinate their integration properly. Make sure all dependent patches are posted before submitting an MR.

e. CVE Reference (optional)

If the changeset resolves a bugzilla that has an associated CVE number, the MR description must include the CVE reference ID. Use one “CVE:” line for each CVE that the changeset resolves, and include the number as described in the bugzilla, e.g "CVE-YYYY-XXXXX".

f. Differences from the upstream patch (optional)

If the commit differs from the original upstream commit, a short description of the differences should be included on a line that begins with “Conflicts:”. Simple patch conflicts like fuzz do not need to be documented.

3. Commit-Specific Description Information

Only commits and single-commit MRs must include the information in this section.

A detailed changelog is required for all changes made to the Red Hat Kernel sources. The maintainers and reviewers do not allow empty commits or MR descriptions.

Commits and single-commit MRs must include the commit hash, author, upstream commit date, upstream title, and upstream commit message in the MR description. An example of the format of the output is provided in the introduction section above. The Red Hat recommended 'git-backport' command replicates the format. Contributors using 'git-cherry-pick' must use the git-commit command’s --reset-author option to avoid false attribution of commits, and modify email ‘Cc:’ lines [3b].

Reviewers prefer to have the entire commit message as it appears upstream, including Cc’s and sign-offs. 'git show'-style indentation of the original upstream commit message helps distinguish the commit message from any additional comments that are added. Many reviewers use the upstream commit message and sign-off details as a measure of how reliable a commit is.

Do not include local commit IDs in changeset submissions. These only confuse reviewers, as reviewers don’t have access to the originating trees. MRs that include local commit IDs will be blocked by webhooks.
a. “commit” sha1 format (required)

Red Hat follows an “upstream-first” philosophy, so commits with an Upstream Status that identify them as originating from an external tree must specify the git sha1 ID. Red Hat Kernel project’s tools parse the commit sha1 for analysis, so the format of the git sha1 ID is strictly enforced.

The upstream sha1 ID must be referenced using one of these three formats (pick one that suits better your workflow):

commit <full sha1 hash>
commit <full sha1 hash> (<branch name>)
(cherry picked from commit <full sha1 hash>)

Note that it starts at the first column (no whitespace at front is allowed) and is followed by a new line. To be precise, the following Perl regex must be able to match:

^commit ([0-9a-f]{40})( \(.*\))?$|^(cherry picked from commit ([0-9a-f]{40}))$\n
b. Email Bridge & Upstream contributors (required)

Upstream contributors Cc’s, Signed-off-by:, and other tags must be shifted right by 4 spaces to avoid the email bridge erroneously cc’ing non-redhat.com email addresses.

4. Version Information (if necessary)

If the changeset has been updated, the Merge Request Description, and the updated commits must contain information on the changes.

5. DCO & Signed-off-by Tag

A "Signed-off-by:" tag on each commit of a changeset is required on all commits and MR descriptions. This sign-off must be completed using your redhat.com email address.

The Linux Kernel Community has required an acknowledgement of the Developer’s Certificate of Origin (DCO), https://www.developercertificate.org, for many years. The DCO is a way of handling contributions to an open source project in which each contribution is associated with a certification signifying that the contributor has the right to submit the contribution under the applicable open source license of the project.

Contributions made to the Red Hat kernel must acknowledge the DCO with the use of a "Signed-off-by:" tag from your redhat.com email address. The tag must start at the beginning of a new line. For example,
Signed-off-by: Shadow Man <shadow.man@redhat.com>

is NOT acceptable. The proper tag usage requires the Sign-off start at the beginning of a new line,

Signed-off-by: Shadow Man <shadow.man@redhat.com>

Contributors must add a Signed-off-by: tag regardless of having signed-off on the original upstream commit.

6. Internal Contributors

Internal “Cc:” or additional “Signed-off-by” tags can be added at the end of the merge request description. The GitLab Email Bridge will cc email addresses in the MR description.