Everything you ever wanted to know about Linux -stable releases

Rules on what kind of patches are accepted, and which ones are not, into the "-stable" tree:

- It must be obviously correct and tested.
- It cannot be bigger than 100 lines, with context.
- It must fix only one thing.
- It must fix a real bug that bothers people (not a, "This could be a problem..." type thing).
- It must fix a problem that causes a build error (but not for things marked CONFIG_BROKEN), an oops, a hang, data corruption, a real security issue, or some "oh, that's not good" issue. In short, something critical.
- Serious issues as reported by a user of a distribution kernel may also be considered if they fix a notable performance or interactivity issue. As these fixes are not as obvious and have a higher risk of a subtle regression they should only be submitted by a distribution kernel maintainer and include an addendum linking to a bugzilla entry if it exists and additional information on the user-visible impact.
- New device IDs and quirks are also accepted.
- No "theoretical race condition" issues, unless an explanation of how the race can be exploited is also provided.
- It cannot contain any "trivial" fixes in it (spelling changes, whitespace cleanups, etc).
- It must follow the :ref:`Documentation/process/submitting-patches.rst <submittingpatches>` rules.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst`, line 29);** ***backlink***
>
> Unknown interpreted text role "ref".

- It or an equivalent fix must already exist in Linus' tree (upstream).

Procedure for submitting patches to the -stable tree

> **Note**
>
> Security patches should not be handled (solely) by the -stable review process but should follow the procedures in :ref:`Documentation/admin-guide/security-bugs.rst <securitybugs>`.
>
> > **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst`, line 40);** ***backlink***
> >
> > Unknown interpreted text role "ref".

For all other submissions, choose one of the following procedures

Option 1

To have the patch automatically included in the stable tree, add the tag

> **System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst`, line 54)**
>
> Cannot analyze code. No Pygments lexer found for "none".
>
> ```
> .. code-block:: none
>
>      Cc: stable@vger.kernel.org
> ```

in the sign-off area. Once the patch is merged it will be applied to the stable tree without anything else needing to be done by the author or subsystem maintainer.

Option 2

After the patch has been merged to Linus' tree, send an email to stable@vger.kernel.org containing the subject of the patch, the commit ID, why you think it should be applied, and what kernel version you wish it to be applied to.

Option 3

Send the patch, after verifying that it follows the above rules, to stable@vger.kernel.org. You must note the upstream commit ID in the changelog of your submission, as well as the kernel version you wish it to be applied to.

:ref:`option_1` is **strongly** preferred, is the easiest and most common. :ref:`option_2` and :ref:`option_3` are more useful if the patch isn't deemed worthy at the time it is applied to a public git tree (for instance, because it deserves more regression testing first). :ref:`option_3` is especially useful if the original upstream patch needs to be backported (for example the backport needs some special handling due to e.g. API changes).

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process)stable-kernel-rules.rst`, line 82);** ***backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process)stable-kernel-rules.rst`, line 82);** ***backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process)stable-kernel-rules.rst`, line 82);** ***backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process)stable-kernel-rules.rst`, line 82);** ***backlink***
>
> Unknown interpreted text role "ref".

Note that for :ref:`option_3`, if the patch deviates from the original upstream patch (for example because it had to be backported) this must be very clearly documented and justified in the patch description.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process)stable-kernel-rules.rst`, line 89);** ***backlink***
>
> Unknown interpreted text role "ref".

The upstream commit ID must be specified with a separate line above the commit text, like this:

> **System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process)stable-kernel-rules.rst`, line 96)**
>
> Cannot analyze code. No Pygments lexer found for "none".
>
> ```
> .. code-block:: none
>
>     commit <sha1> upstream.
> ```

Additionally, some patches submitted via :ref:`option_1` may have additional patch prerequisites which can be cherry-picked. This can be specified in the following format in the sign-off area:

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-`

master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst, line 100); *backlink*

Unknown interpreted text role "ref".

System Message: WARNING/2 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst, line 104)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

     Cc: <stable@vger.kernel.org> # 3.3.x: a1f84a3: sched: Check for idle
     Cc: <stable@vger.kernel.org> # 3.3.x: 1b9508f: sched: Rate-limit newidle
     Cc: <stable@vger.kernel.org> # 3.3.x: fd21073: sched: Fix affinity logic
     Cc: <stable@vger.kernel.org> # 3.3.x
     Signed-off-by: Ingo Molnar <mingo@elte.hu>
```

The tag sequence has the meaning of:

System Message: WARNING/2 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst, line 114)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

     git cherry-pick a1f84a3
     git cherry-pick 1b9508f
     git cherry-pick fd21073
     git cherry-pick <this commit>
```

Also, some patches may have kernel version prerequisites. This can be specified in the following format in the sign-off area:

System Message: WARNING/2 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst, line 124)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

     Cc: <stable@vger.kernel.org> # 3.3.x
```

The tag has the meaning of:

System Message: WARNING/2 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\process\(linux-master) (Documentation) (process) stable-kernel-rules.rst, line 130)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

     git cherry-pick <this commit>
```

For each "-stable" tree starting with the specified version.

Following the submission:

- The sender will receive an ACK when the patch has been accepted into the queue, or a NAK if the patch is rejected. This response might take a few days, according to the developer's schedules.
- If accepted, the patch will be added to the -stable queue, for review by other developers and by the relevant subsystem maintainer.

Review cycle

- When the -stable maintainers decide for a review cycle, the patches will be sent to the review committee, and the maintainer of the affected area of the patch (unless the submitter is the maintainer of the area) and CC: to the linux-kernel mailing list.
- The review committee has 48 hours in which to ACK or NAK the patch.
- If the patch is rejected by a member of the committee, or linux-kernel members object to the patch, bringing up issues that the maintainers and members did not realize, the patch will be dropped from the queue.
- The ACKed patches will be posted again as part of release candidate (-rc) to be tested by developers and testers.
- Usually only one -rc release is made, however if there are any outstanding issues, some patches may be modified or dropped or additional patches may be queued. Additional -rc releases are then released and tested until no issues are found.
- Responding to the -rc releases can be done on the mailing list by sending a "Tested-by:" email with any testing information desired. The "Tested-by:" tags will be collected and added to the release commit.
- At the end of the review cycle, the new -stable release will be released containing all the queued and tested patches.
- Security patches will be accepted into the -stable tree directly from the security kernel team, and not go through the normal review cycle. Contact the kernel security team for more details on this procedure.

Trees

- The queues of patches, for both completed versions and in progress versions can be found at:

  https://git.kernel.org/pub/scm/linux/kernel/git/stable/stable-queue.git

- The finalized and tagged releases of all stable kernels can be found in separate branches per version at:

  https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git

- The release candidate of all stable kernel versions can be found at:

  https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable-rc.git/

  > **Warning**
  >
  > The -stable-rc tree is a snapshot in time of the stable-queue tree and will change frequently, hence will be rebased often. It should only be used for testing purposes (e.g. to be consumed by CI systems).

Review committee

- This is made up of a number of kernel developers who have volunteered for this task, and a few that haven't.