Linux kernel patch format

Most Linux kernel submissions are merged into the kernel source code repository by script. These instructions describe the proper format for emailed kernel patch submissions, to ensure that submittors and maintainers waste a minimum amount of time on these details.

1. Email subject format

The email subject has a precise format, communicating several pieces of information.

[PATCH $version $n/$total] $subsystem: one-line summary

- "[PATCH": constant prefix
- $version: kernel version against which this patch was generated
- $n/$total: when a change encompasses more than one patch, $n indicates the order of this patch in the series, and $total indicates the total number of patches in the series.
- $subsystem: area of the kernel, or device driver, to which this patch applies.
- one-line summary: summarizes the change this patch makes. This summary is copied *directly* into the SCM (i.e. git) changelog, so make sure that your summary is descriptive. Ensure that $subsystem uniquely identifies the subsystem or driver being modified. **"update to latest CVS"** or **"fix bug in probe"** is not specific enough about *what portion of the code* is being modified.
- Keep your overall subject line under 65 characters or so.

The scripts will strip all the text inside the brackets ("[PATCH ...]"), and replace it "[PATCH]".

Example:

```
[PATCH 2.6.9-rc1-mm2 1/2] 8139cp: fix PCI DAC mode
```

The "$n/$total" may be omitted if there is only one patch in the series. Writing "1/1" is not necessary.

2. Email body contents: description

At the beginning of your email, use as many lines as you wish to describe the patch. This text is copied *directly* into the SCM (i.e. git) changelog.

Include comments and other data (such as diffstat) you don't wish to be in the kernel changelog **following a "---" terminator line**. The terminator must be on a line by itself.

3. Email body contents: patch

Sub-Rule Number One: Patches must be reviewable in a standard Linux email client. This means in particular, **no compression** and **no base64 encoding**. Attachments are discouraged, but some corporate

mail systems provide no other way to send patches.

Sub-Rule Number Two: Patch must be apply-able by a script that has no knowledge of [MIME] encoding. You must make sure your mailer does *not* escape standard US-ASCII characters, wrap long lines, or encode plaintext patches in base64 (or any other encoding).

Sub-Rule Number Three: Patch must be rooted one level above a standard kernel source tree. i.e.

```
--- a/drivers/scsi/megaraid/megaraid_mm.c       2004-08-31 04:05:10 -04:00
+++ b/drivers/scsi/megaraid/megaraid_mm.c       2004-08-31 04:05:10 -04:00
```

or in other words, the patch must be apply-able using

```
patch -sp1 < foo.patch
```

4. One patch per email

This cannot be stressed enough. Even when you are resending a change for the 5th time, resist the urge to attach 20 patches to a single email. If you do send multiple emails, make sure the second and subsequent emails are sent as replies to the first, to keep them all together in a thread.

5. Sign your work

The sign-off is a simple line at the end of the explanation for the patch, which certifies that you wrote it or otherwise have the right to pass it on as a open-source patch. The rules are pretty simple: if you can certify the below:

```
Developer's Certificate of Origin 1.1

By making a contribution to this project, I certify that:

(a) The contribution was created in whole or in part by me and I
    have the right to submit it under the open source license
    indicated in the file; or

(b) The contribution is based upon previous work that, to the best
    of my knowledge, is covered under an appropriate open source
    license and I have the right under that license to submit that
    work with modifications, whether created in whole or in part
    by me, under the same open source license (unless I am
    permitted to submit under a different license), as indicated
    in the file; or

(c) The contribution was provided directly to me by some other
    person who certified (a), (b) or (c) and I have not modified
    it.

(d) I understand and agree that this project and the contribution
    are public and that a record of the contribution (including all
    personal information I submit with it, including my sign-off) is
    maintained indefinitely and may be redistributed consistent with
    this project or the open source license(s) involved.
```

then you just add a line at the end of your patch description, saying

```
Signed-off-by: Random J Developer <random@developer.org>
```

6. Avoid attachments and MIME

Attachments make it more difficult to review and comment on your patches. MIME (the mechanism by which files are attached to an email) has historically created a problem for patch import scripts. Some unfortunate email programs insist upon base64 encoding for all attachments, which completely shrouds the patch to some scripts and mailers.

7. Follow these instructions even when resending

Quite often, when a patch receives comments, the patch author will (deep in an email thread) include a revised version of their patch but omit the email subject one-line summary, and overall patch description. This isn't script friendly, and requires the patch description to be hand-edited.

For more details, read Documentation/SubmittingPatches and Documentation/SubmittingDrivers in the kernel source tree.
