Handling messy pull-request diffstats

Subsystem maintainers routinely use `git request-pull` as part of the process of sending work upstream. Normally, the result includes a nice diffstat that shows which files will be touched and how much of each will be changed. Occasionally, though, a repository with a relatively complicated development history will yield a massive diffstat containing a great deal of unrelated work. The result looks ugly and obscures what the pull request is actually doing. This document describes what is happening and how to fix things up; it is derived from The Wisdom of Linus Torvalds, found in Linus1 and Linus2.

A Git development history proceeds as a series of commits. In a simplified manner, mainline kernel development looks like this:

```
... vM --- vN-rc1 --- vN-rc2 --- vN-rc3 --- ... --- vN-rc7 --- vN
```

If one wants to see what has changed between two points, a command like this will do the job:

```
$ git diff --stat --summary vN-rc2..vN-rc3
```

Here, there are two clear points in the history; Git will essentially "subtract" the beginning point from the end point and display the resulting differences. The requested operation is unambiguous and easy enough to understand.

When a subsystem maintainer creates a branch and commits changes to it, the result in the simplest case is a history that looks like:

```
... vM --- vN-rc1 --- vN-rc2 --- vN-rc3 --- ... --- vN-rc7 --- vN
                       |
                       +-- c1 --- c2 --- ... --- cN
```

If that maintainer now uses `git diff` to see what has changed between the mainline branch (let's call it "linus") and cN, there are still two clear endpoints, and the result is as expected. So a pull request generated with `git request-pull` will also be as expected. But now consider a slightly more complex development history:

```
... vM --- vN-rc1 --- vN-rc2 --- vN-rc3 --- ... --- vN-rc7 --- vN
             |         |
             |         +-- c1 --- c2 --- ... --- cN
             |                    /
             +-- x1 --- x2 --- x3
```

Our maintainer has created one branch at vN-rc1 and another at vN-rc2; the two were then subsequently merged into c2. Now a pull request generated for cN may end up being messy indeed, and developers often end up wondering why.

What is happening here is that there are no longer two clear end points for the `git diff` operation to use. The development culminating in cN started in two different places; to generate the diffstat, `git diff` ends up having pick one of them and hoping for the best. If the diffstat starts at vN-rc1, it may end up including all of the changes between there and the second origin end point (vN-rc2), which is certainly not what our maintainer had in mind. With all of that extra junk in the diffstat, it may be impossible to tell what actually happened in the changes leading up to cN.

Maintainers often try to resolve this problem by, for example, rebasing the branch or performing another merge with the linus branch, then recreating the pull request. This approach tends not to lead to joy at the receiving end of that pull request; rebasing and/or merging just before pushing upstream is a well-known way to get a grumpy response.

So what is to be done? The best response when confronted with this situation is to indeed to do a merge with the branch you intend your work to be pulled into, but to do it privately, as if it were the source of shame. Create a new, throwaway branch and do the merge there:

```
... vM --- vN-rc1 --- vN-rc2 --- vN-rc3 --- ... --- vN-rc7 --- vN
             |         |                                      |
             |         +-- c1 --- c2 --- ... --- cN           |
             |                    /               |           |
             +-- x1 --- x2 --- x3                +-----------+-- TEMP
```

The merge operation resolves all of the complications resulting from the multiple beginning points, yielding a coherent result that contains only the differences from the mainline branch. Now it will be possible to generate a diffstat with the desired information:

```
$ git diff -C --stat --summary linus..TEMP
```

Save the output from this command, then simply delete the TEMP branch; definitely do not expose it to the outside world. Take the saved diffstat output and edit it into the messy pull request, yielding a result that shows what is really going on. That request can then be sent upstream.