

## Re: Documentation of kernel messages (Summary)

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-06/msg10509.html>

---

- *From:* Rob Landley <[rob@xxxxxxxxxxx](mailto:rob@xxxxxxxxxxx)>
  - *Date:* Mon, 25 Jun 2007 11:44:45 -0400
- 

On Monday 25 June 2007 09:48:41 Michael Holzheu wrote:

Hi all,

Any idea, how to proceed with this topic? Do you think that any of the suggested solutions for documentation / translation of kernel messages will have a chance to be included in the kernel?

Personally? No to the second question, which renders the first "do it yourself outside of the tree".

Just a guess, and I don't speak for anyone else here, but I think most of us are waiting to see how long it takes you to lose interest.

I tried to summarize the outcome of this discussion...

There are two main issues:

- \* Translate messages
- \* Document messages

I'm somewhat sympathetic to translation (although I tend to think of it as a distro's job, and still wonder why IBM is pushing this directly rather than trying to bounce it off Red Hat or somebody). But you could put a sed filter around dmesg to give you swahili output entirely in userspace. Have you shown the simple approach to be insufficient yet?

Is "document messages" the same problem as "translate messages", or are these two different problems conflated together?

I find "document messages" to be a horrible idea conceptually, because I think the messages should `_be_` documentation. If the message is unclear it should be clarified. "There's too much garbage on the floor, we should laminate it so we can't smell it anymore." Er, no...

## Re: Documentation of kernel messages (Summary)

Three possible solutions have been suggested:

1. Use message numbers maintained by driver owners
2. Automatically create hashes for printk
3. Use printk format string as message identifier

#2 has the advantage that you can do it without buy-in from the community, and without touching the code. (Of course it doesn't necessarily help you if the printk string is "%s:%s".)

But I'm still convinced you could tackle over 90% of the problem from userspace, without touching the kernel.

Pros and Cons of the different Message ID methods:

=====

Message numbers maintained by driver owners:

- + Unique IDs
- + Can be kept stable between different kernel versions
- + Efficient way of matching printk to Documentation or translation
- Difficult to maintain

Will at best only ever be done for some drivers, not all. But that's probably inescapable no matter what your approach is.

printk hashes:

- Additional preprocessor step needed

Only for you guys. Those of us building our own kernels and NOT translating them or producing "the big book of explanations of kernel messages for IBM customers who don't understand them" shouldn't have to run this preprocessor step.

Pros and Cons of external documentation (vs. internal documentation):

- =====
- + Developers have less work to do
  - + Kernel sources and patches have less lines of code, since message descriptions are not contained in the kernel sources.
  - + You can support multiple languages
  - Hard to keep printks and descriptions up-to-date. If using hashes, each typo fix in a printk breaks the link to your documentation.
  - Since the developer knows the meaning of a printk best, he probably would be the right person to write the description.

## Re: Documentation of kernel messages (Summary)

Imposing additional requirements on developers isn't going to work unless you reject patches from developers who don't follow your procedures and submit the appropriately filled-out forms with each patch for processing by the central bureaucracy.

Signed-off-by works because A) Linus enforces it, B) The overhead's very low, C) attribution is already highly good thing within our community values, D) it annoys SCO.

The javadoc function comments in the source don't have complete coverage and probably never will (although they're getting better). The premise of doing something similar for messages is that a translating dmesg in userspace can't necessarily get complete coverage, so you need a solution modeled on something that doesn't have complete coverage years after its introduction...

– For every Distribution or vanilla kernel you probably need separate external message databases.

"If you're resigned to doing that anyway you don't actually need him to lie there all the time, do you?" – Ford Prefect

If you need an external message database what's wrong with doing it entirely in userspace via a sed-based translating dmesg?

Rob

--

"One of my most productive days was throwing away 1000 lines of code."

– Ken Thompson.

–

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>