## Kernel Traffic #85 For 18 Sep

By Zack Brown

### Table Of Contents



### Introduction

Thanks go to Rhet Turnbull for catching a bad link to the Open Group in last week's introduction. Looks like I used a .com instead of a .org there... it's fixed now. Thanks, Rhet!

Thanks also go to Peter H. Ruegg, who noticed that I will sometimes end a paragraph using  within a multi-paragraph <li>, that I did not begin with a . Yoikes! That slipped right past my radar. I'll try to do better on that. Thanks a million, Peter!

Jamie Guinan caught me spelling 'subthread' as 'ubhread' last issue, which...well, that's just weird. Anyway, thanks Jamie!

I also got some replies to the ongoing "is the kernel closed?" discussion in the KT introduction, but the intro of KT is not the place to carry on that debate. Folks interested in this should voice their concerns on linux-kernel, and I'll cover the discussion. ;-)

**Mailing List Stats For This Week**

We looked at 1178 posts in 4527K.

There were 352 different contributors. 178 (50%) posted more than once. 126 (35%) posted last week too.

The top posters of the week were:

- 57 posts in 144K by Alan Cox <alan@lxorguk.ukuu.org.uk>
- 41 posts in 140K by Alexander Viro <viro@math.psu.edu>
- 37 posts in 148K by Linus Torvalds <torvalds@transmeta.com>
- 30 posts in 76K by "David S. Miller" <davem@redhat.com>
- 27 posts in 140K by Andrea Arcangeli <andrea@suse.de>
- 27 posts in 81K by Rik van Riel <riel@conectiva.com.br>

**1. Speeding Up Laptop Suspension**

**27 Aug - 5 Sep (18 posts): Suggestion for laptop**

**suspension**

Richard M. Stallman [*] offered a technical report and suggestion:

> **On my laptop, suspending to disk is slow. It takes a whole minute, and another minute to unsuspend--if all of core is in use. But it is much faster when I do it not too long after the system was booted, when it has not yet managed to use up most of core.**
>
> **This suggests that it could be nice to make the kernel empty its caches, and mark all that core "unused", when a suspend is about to happen. I don't know the details of how suspension works, so I am not sure this is possible, but I think there are hooks for doing something in the kernel before a suspend.**

Alex Buell [*] couldn't help himself, and quipped, **"Not to be sarcastic but this is the *best* suggestion I've seen so far from the long list of suggestions you've made in the past. Praise be to Allah, you may be saved."** Daniel Phillips [*] replied more soberly to Richard, **"There are other good reasons why users should be able to tell the kernel to shrink its caches. For example, throwing away the cache to force actual device accesses would help determine if low-level file and device I/O are working correctly. An example for normal users: you might be aware that some big job that filled up the caches is now finished, and so the caches can be shrunk immediately instead of piecemeal as other applications try to get memory. As user, you are sometimes in a much better position to know when to shrink the caches than the OS is."** He went on to give some implementation suggestions, and added that a lot of the code was already in the kernel, so this new patch should be fairly easy to write. But Stephen C. Tweedie [*] replied that controlling the cache should really be on a per-file basis rather than per-device, and explained, **"You want to be able to do things like open a file O_RSYNC and know that the kernel will read from physical media instead of from cache, or to open it O_DIRECT and know that the data will not be cached at all. There are so many uses for that sort of thing that trying to fix it by adding block device or system-wide**

**virtual memory calls is really not going to address the needs adequately."**

There was no reply to that, but Theodore Y. Ts'o [*] also replied to Richard, saying he'd experienced similar behavior on his Sony Vaio 505TX, though he hadn't had time to really investigate it. He said it would be a nice project for someone interested in it, though he cautioned that the code might end up having to be very hardware- or BIOS-dependent. Some discussion followed from this, and at one point Pavel Machek [*] recommended, **"You *dont* need bios support to do suspend-to-disk, as swsusp shows. That is generic across all machines. I believe improving swsusp (it is pretty slow these days, partly because it trashes all caches: it does so by applying artifical memory pressure) is much better project than trying to find out what value do you need in order for BIOS to compress ram."**

Stephen Rothwell [*] also replied to Richard's initial post, and posted some code to zero out free memory. On his machine, this reduced suspend time from around 30 seconds to 10. But in a later post, he added that it had only been an experiment, and that **"I agree with others that swsusp is probably the way to go (especially for machines with ACPI but no S4BIOS support), but I thought I might be able to provide Richard with a quick and (only slightly) dirty solution to his immediate problem."** Richard Gooch [*] tried his own RAM-filling program on a Dell Inspiron 3200, and the suspend time dropped from 47 seconds to 20. Richard S. replied that this didn't change anything on his laptop, though he added, **"But this particular obscure model of laptop is not important. The thing is to handle most laptops, to make suspending faster for most users, and to build it in by default so that it works "out of the box" on most machines."** Pavel again recommended work be done on 'swsusp', and Richard G. said he felt Richard S.'s suggestion shouldn't be the default. He suggested instead, **"What would be nicer is if when you suspend, you record the cached block numbers (and then flush+clear the caches), and on resume kick off a daemon/kernel thread which touches those blocks, bringing them back into the caches. Rate-limit the daemon so that "urgent" (i.e. what the user needs *now*) I/O is only marginally affected. And since that daemon has time**

**to repopulate, it can re-order the I/O to avoid head seeks."**

**2. VM Patches Shaping Up, But Still Not Ready**

**1 Sep - 11 Sep (116 posts): 2.4.0-test8-pre1 is quite bad**

Under 2.4.0-test8-pre1, Tigran Aivazian [*] tried copying a 2G file from one place to another on the same filesystem, and found that **"It starts swapping like mad and generally behaves indecently, despite the huge 1024M of RAM it has."** Rik van Riel [*] gave a link to a patch (http://www.surriel.com/patches/2.4.0-t8p1-vmpatch2) and remarked, **"I'm working on these issues and seem to be pretty close to having fixed most of them now..."** Mark Hahn [*] reported that the patch looked like a major improvement, and there was a bit of discussion. Then (slightly elsewhere) Bert Hubert [*] asked Linus Torvalds [*] to merge the patch into the next test kernel, in the interests of widening the group of people reporting bugs and submitting patches for it. This made sense to Rik, who added that he definitely wanted to have the new VM in place *before* 2.4; Bert had expressed the idea that the only drawback to merging the patches now would be instability for people currently using the 2.4 test patches on production systems, but Rik replied, **"Not really. I'm not aware of any bug with my VM that doesn't occur in the standard VM too."**

Linus asked what had happened to a particular bug they'd been seeing earlier with the new patches, which, he said, **"certainly didn't happen with the standard VM."** Rik replied that he hadn't been able to trigger it with his latest patches. He said that it **"seems to be defanged for now and isn't very high priority to me..."** But Linus replied, **"I'd like to know what it was. Last I heard, it was still the case of "pages just off the freelist had some bits set that they shouldn't have". That makes me nervous."** Bert replied that maybe this was a reason to merge the patch, so many more folks would have a better chance of uncovering the bug.

Rik and Linus argued briefly over whether the bug was worth searching for or had even existed in the first place (Rik had booby-trapped the code and found nothing, but Linus wasn't convinced), and there was apparently some private email

discussing what the bug might be, and there followed a long technical discussion on the list.

**3. Positive Reports On The Latest VM Patches**

**2 Sep - 5 Sep (14 posts): Rik van Riel's VM patch**

John Levon reported that Rik van Riel [*]'s latest Virtual Memory patch to test8-pre1 seemed to really improve things. He hadn't done any statistics, but the system seemed smooth, and kept very good interactivity even under extremely heavy load. Bill Huey [*] replied exuberantly, **"Yes, it kicks butt and it finally (just about) removes the final Linux kernel showstopper for recent kernels. ;-)"** Byron Stanoszek [*] added while dancing around, **"This patch is plain awesome. It really sped up my 586 test machine (very noticible when compiling XFree86.. which knocked off about a half hour of compilation time), and there isn't a [noticable] memory leak like in the old VM system before."**

Jeff V. Merkey [*] suggested benchmarking the system in order to have some numbers to throw at SCO when their numbers hit the street. For more on SCO's claims, see Issue #84, Section #1 (**23 Aug:** "Posix Threads (pthreads) In Linux").

Elsewhere, Rik said:

> **there are 4 major issues left in the VM area:**
>
> 1. **system hangs under load with 0 lowmem free (but still some high memory free) [not much details on this one yet]**
>
> 2. **dirty bits can get lost, try_to_swap_out() and other places have a race with the hardware**
>
>    **[from mm/vmscan.c, line 60 has a race with the /hardware/]**
>
>    ```
>    55  if (pte_young(pte)) {
>    56          /*
>    57           * Transfer the "accessed" bit from the pa
>    58           * tables to the global page map.
>    59           */
>    60          set_pte(page_table, pte_mkold(pte));
>    61          SetPageReferenced(page);
>    ```

```
62          goto out_failed;
63  }
```

3. it appears something can corrupt page->count or delete a page from the cache while the page is locked [tripped up by my VM patch?]
4. the innd data corruption bug

**4. 2.4.0-test8-pre2: Long-Time (Over A Year) Filesystem Corruption Bug Uncovered**

**3 Sep (22 posts): test8-pre2 fs corruption?**

Mohammad A. Haque reported filesystem corruption with test8-pre2. Thomas Molina [*] and others also saw the corruption, though some folks only saw it while reading the 'bugtraq' mailing list. At one point, Alexander Viro [*] reported:

> **Fun... OK, folks. There was a bug in ext2 (and damn next to every other local fs) since 2.3.7. Sometimes (depending on phase of moon, pagecache contents and a bunch of other shit) truncate() leaves the tail of last block not zeroed. Looks like it surfaced from time to time, but had been mostly unnoticed. Something in the last kernels made it show up more often.**
>
> **I dearly hope that patch I've posted today will fix the bloody thing for good. Because if it will not we've got something else to deal with.**
>
> **As for the bug itself - look at the truncate_inode_pages() (part that does zero-out of the partial page) and ask yourself: WTF will happen if pagecache has nothing for that file? Had been there for more than a year...**

Discussion continued, and it was discovered that a message to the 'bugtraq' list had triggered a bug in pine, which accounted for some of the problems folks had reported. The kernel bug turned out to be real, however. Alexander and others posted patches, and the thread petered out.

### 5. 2.4.0-test8-pre4: Filesystem Corruption Bug And Rolled Up Newspaper

**4 Sep - 7 Sep (16 posts): test8-pre4: innd fixed?**

Linus Torvalds [*] announced test8-pre4 (pre3 had been "internal only" between him and Alexander Viro [*], trying to track down the recent filesystem corruption problems. Linus said:

> **Could people who have seen the innd active list file corruption thing please try out linux-2.4.0-test8-pre4? Despite some reports, it was _not_ fixed in pre2, and pre3 was a internal-only test to fix the remaining issue with Al Viro.**
>
> **pre4 finally passes all my truncate() tests, and the code looks good. The only thing left to verify is whether innd really is happy, or whether the inactive file corruption may have been due to something else altogether. I'm personally fairly optimistic, but this needs to be verified so that everybody can either breathe a sigh of relief or go looking for "the final bug".**
>
> **Note that reiserfs is reported to have the same bug, and pre4 won't have fixed that - only ext2. The infrastructure is all in place so that the reiserfs fix is probably a one-liner, but for now I just want to hear from people who have their news-spools and active file on an ext2 filesystem.**

Mohammad A. Haque asked if this was the same problem from Section #4. Linus replied that at least part of that problem had been due to a buggy 'pine' and funny mail headers; Alexander Viro also replied to Mohammad:

> **Hell knows. Let me put it that way: one long-standing bug definitely had kicked the bucket. The question being: had any other crap survived?**
>
> **In other words, if you can reproduce fs corruption on clean ext2 (after forced fsck if you've used it with earlier kernels) - tell.**

Nearby, Simon Kirby [*] reported mailbox corruption when exiting 'mutt'. Martin Costabel reported an identical problem, and André Dahlqvist reported the same thing on test7, as did Horst von Brand [*]. Alexander took knife and fork to the code and managed to reproduce the corruption in just under an hour and a half. He found and fixed one bug, and requested, **"folks, _any_ additional eyes are more than welcome. I tried to make the structure of code as straightforward as possible, so it should be readable..."**

### 6. 2.4.0-test8-pre5: More Swipes At Filesystem Corruption Bug

**5 Sep - 11 Sep (5 posts): linux-2.4.0-test8-pre5**

Linus Torvalds [*] announced 2.4.0-test8-pre5, and explained:

> **This entry in the changelog says it all:**
>
> **- truncate. Guess what? We threw away the key to the clue-box**
>
> **Most of the other stuff is cleanups or reasonably straightforward fixes. The truncate thread that's been going through the last few pre-releases is the big thing, and the one that has caused heartache.**
>
> **Oh, well. It really is fixed now. Everybody involved has been forced to wear the brown paper bag for the next month or so, but we cut out small holes so that we can see where we are going and not bump into things and make even more of a spectacle of ourselves.**
>
> **And the really is fixed this time. Cross my heart and hope to die.?I can only say that "truncate" has always been the nastiest operation of them all.**
>
> **The truncate bug, btw, has been there for roughly 14 months. We just first made it easier to trigger, and then messed up about three times when we tried to fix it. Oh, well.**

> **Read my lips. No more fscks. And thanks to everybody who did.**

He included the full ChangeLog:

- **pre5**
  1. **truncate. Guess what? We threw away the key to the clue-box.**
  2. **simplify signal notification. And remember the spinlock.**
  3. **VIA ide driver update (well, rewrite - the old one was buggy and broken)**
  4. **network driver fixes (not checking for oom etc)**
  5. **USB serial driver SMP locking fixes**
  6. **fix memory leak on failed USB configuration queries**
  7. **USB initialization using proper "init()" calls.**
  8. **dvd capacity bug fix and other cdrom driver cleanups**
  9. **sis5513 IDE chipset update**
  10. **do_fork() - add "stack-top" for ia64 (and potentially other architectures that may care)**
  11. **devfs support for LVM**
  12. **quota transfer miscount fix**
  13. **x86 checksum/copy prefetch**
  14. **NFS sillydelete fix**
  15. **mark_buffer_dirty() doesn't actually use the second argument. Delete it.**
  16. **SCSI communications device - no need to complain about it**
  17. **SCSI WP test fix (all pages, not just the first one)**

Peter Samuelson [*] replied with a bug in the VIA update (item 3 from the ChangeLog), and Vojtech Pavlik [*] said he'd send a patch for that to Linus soon.

**7. Still Swatting Filesystem Corruption Bug(s)**

**6 Sep (30 posts): Still ext2-corruption in test8-pre5 (incl.**

**OOPS)**

Udo A. Steinberg reported filesystem corruption with test8-pre5, and posted an oops. There was a bit of discussion, and at one point Linus Torvalds [*] posted a patch and said:

> **How about this patch?**
>
> **NOTE NOTE NOTE! I'm on my way home now to be a family man, so I've not actually tested it AT ALL. You have been warned.**
>
> **The basic approach should be ok, and it avoids using the write path at all because it isn't actually needed. The truncate() case is, in the end, much simpler than writing, exactly because we don't need to allocate any new blocks etc.**
>
> **We just grab the page, populate it with buffers if required, and find the one buffer that we need to clear out. We clear it out and mark it dirty. End of story.**
>
> **NOTE: Udo, because I haven't actually tested this (it may not actually compile etc small details), you probably shouldn't actually test this out as-is unless you are _really_ daring and don't mind fixing up after me. It's more a "this is how it should work" kind of thing.**
>
> **Al? Mind giving it a quick look?**

Alexander liked the patch, though he had some specific objections. Mohammad A . Haque tested it and was unable to reproduce the corruption. However, Tim Waugh [*] posted an oops, to which Linus replied:

> **This one I cannot explain.**
>
> **It's a bh that is NULL, but it's a new case completely. It looks like you have a 1kB blocksize, no? It furthermore looks like the page only had two buffers on it, and accessing the fourth one blows up (accessing the third one gets NULL, which is why it only blows up on the fourth one).**

> **But it _has_ to have four buffers on it. Two buffers just aren't enough to cover a 4kB page.**
>
> **Uh.**
>
> **DUH!**
>
> **I found the bug.**
>
> **It shouldn't be "bh->b_next". That's just the next buffer on the hash chain.**
>
> **It should be "bh->b_this_page".**
>
> **Just change block_truncate_page() to use b_this_page instead of b_next.**
>
> **The "good news" is that me testing this would neve rhave found it, because all my filesystems are 4kB blocks, so the "next bh" case never triggers at all.**
>
> **Thanks, and THIS time it really is fixed. I mean, how many times can we get it wrong? At some point, we just have to run out of really bad ideas..**

Tim said it seemed fixed this time, and Rik van Riel [*] remarked sardonically, **"I'm really really really looking forward to the first kernel since 2.3.7 that doesn't have the potential to eat my files..."**

### 8. 2.4.0-test8-pre6: Filesystem Corruption Bug Seems Dead

**6 Sep - 9 Sep (18 posts): Linux-2.4.0-test8-pre6**

Linus Torvalds [*] announced:

> **Yeah. Maybe we fixed truncate, and maybe we didn't. I've thought that we fixed it now several times, and I was always wrong. Time for some reverse phychology:**
>
> **I'm sure this one doesn't fix the truncate bug either.**

> **But I have this ugly feeling that I'm coming down with the same flu that everybody else in my family had the last week, so I'd better release this before I start puking on my keyboard.**

He also posted the full ChangeLog:

- **pre6:**
  1. **trunate - the never-ending story. Makes me feel like a long Kurosawa movie. But in this one the hero _will_ survive, or my name isn't Maxwell.**
  2. **SCSI tape driver potential memory leak.**
  3. **XMM FP handler bug fix: we really must not change the FP error mask on exceptions. People care.**

Udo A. Steinberg reported no corruption so far, but promised to be really vicious over the next few days. Elsewhere, under the Subject: test8-pre6 file corruption and oops, Kenneth Johansson thought he'd hit the bug again, but it turned out he was just using an old 'System.map' file.

**9. Non-GPLed Drivers**

**6 Sep - 7 Sep (5 posts): Is it OK to release non-GPL network driver with source?**

Dave Allen [*] posted:

> **My company is currently working on a linux network driver (I'm sorry, but I can't disclose which company or the nature of the driver right now). However, recent discussions on this list have made me grow concerned about licensing problems with the GPL.**
>
> **The source code for the driver _is_ going to be available, but it will not be GPL'd. There are no patches to the kernel involved. I understand that there should be no problems, but the use of inline functions in the kernel header files makes the situation a bit more complicated. The compiled**

> **binary code then does contain GPL'd code, so would it not then be considered a "derivative work"?**
>
> **If indeed this is a violation of the GPL, is there any way around this by releasing only the source code (even though it isn't GPL'd)? I mean, the compiled binary code does contain GPL'd code, but the source code does not. Is it OK to distribute this?**
>
> **So, I'm asking the experts here for opinions or insights.**

Alan Cox [*] replied, **"Wrong list. Seriously. If you want to know about copyright law ask your company lawyer."** And Linus Torvalds [*] replied to Dave:

> **Note that whenever it's not GPL'd, all the module restrictions kick in. So it's going to be "legal" the same way any binary only module is "legal" - assuming all the nasty requirements are met. For something as simple (from a conceptual standpoint, not necessarily an implementation standpoint) as a network driver, doing that is not likely to be a big problem.**
>
> **It obviously cannot be linked into the kernel, but as a loadable module it's ok as long as it uses the standard interfaces and nothing more.**
>
> **And sure, having source available might make it easier for people to help you: it can't become part of the standard kernel, and as such it will never be supported, but that's true of binary-only modules too.**
>
> **I wouldn't recommend it, but I don't see that it would be an insurmountable problem.**

There was no reply.

**10. Kernel Debugging Documentation**

**7 Sep (3 posts): Kernel Debugging Documentation**

Andrea Arcangeli [*] announced:

> **Back in May I wrote a quite estensive documentation about all the possible/best ways to debug the Linux Kernel for a talk/tranining that I did in San Jose in May. I find now the time to clean it up and to upload since I think it could result useful to everybody dealing with kernel developement.**
>
> **ftp://ftp.suse.com/pub/people/andrea/talks/english/2000/kdebug-may-2000-20000907.tar.gz (ftp://ftp.suse.com/pub/people/andrea/talks/english/2000/kdebug-may-2000-20000907.tar.gz)**
>
> **It addresses MCORE/LKCD/KDB/KGDB/NMI-watchdog/TRACER/IKD/PRINTK/BUG/OOPS/KMSGDUMP and many other issues (simptom/realbug as well).**
>
> **They were the digital slides for the talk, so while writing them I expected to ingegrate them with speach, but they should be readable also standalone.**
>
> **They're written in MGP (MagiPoint, not that I like it too much but kpresenter wasn't that powerful at that time). A postscript is included into the tarball as well (they should be easily readable with `gv` with antialiasing enabled).**
>
> **I'd say it's a _must_ read for any kernel developer (feel free to announce it on other places as well if you think it's good idea of course).**
>
> **Hope you find them useful, have fun.**
>
> **If you have patches for the document send them to me and I'll integrate them. Kurt just sent me a 1000 lines patch to correct my english errors in the original version :))**

There was not much discussion, but folks were generally really impressed.

### 11. Linus And Others Concerned For The Future Of GPL

**8 Sep - 9 Sep (5 posts): Linux-2.4.0-test8**

Linus Torvalds [*] announced the latest test release in preparation for 2.4, and added:

> **The only one of any note that I'd like to point out directly is the clarification in the COPYING file, making it clear that it's only _that_ particular version of the GPL that is valid for the kernel. This should not come as any surprise, as that's the same license that has been there since 0.12 or so, but I thought I'd make that explicit.**
>
> **Why? There's been some discussions of a GPL v3 which would limit licensing to certain "well-behaved" parties, and I'm not sure I'd agree with such restrictions - and the GPL itself allows for "any version" so I wanted to make this part unambigious as far as my personal code is concerned.**
>
> **The reason I wanted to mention that particular issue here explicitly (rather than as just a one-liner in the changelog) is that code written by others is obviously under _their_ discretion, and not limited by my personal foibles, fears and misgivings.**
>
> **If anybody wants to explicitly state that their code will be valid under any version of the GPL (current or future - whatever they may look like), please send patches to say so for the code in question. If you've used the FSF boiler-place copyright notice, you already have this in place (it says "v2 or later" - the FSF itself doesn't recommend v1 any more).**
>
> **(Me, I'm taking the careful "wait and see" approach. I don't know if a GPL v3 is imminent, and I don't know if the issues discussed will even _become_ real issues, so you might as well consider me a paranoid, if careful, bastard).**

Alan Cox [*] remarked, **"I think an appropriate concern. The future GPL is constrained by the GPLv2 clause 9 to be 'similar in spirit...'. You also dont ever have to take any code that specifies GPLv3 or later,"** and added, **"Every line of code I wrote is under the GPLv2 or later (except those bits I contributed that were BSD non advertising derived and which I left the BSD license on)."**

Jamie Lokier [*] said his code was "version 2 or later" as well, and went on:

> **Linus, nobody can ever force GPLv3 upon you. If you don't like GPLv3 when you actually see one, then you can restrict the kernel to GPLv2 only and refuse contributions that don't honour that.**
>
> **It would be a right pain to be unable to cut & paste between kernel code and GPLv3 user space code, except for kernel versions prior to test8. This may well affect you.**
>
> **I wouldn't be surprised if GPLv3 simply clarifies things. Clearer legal language, clearer on dynamic linking etc.**
>
> **If you end up happy with GPLv3 after all, (and you might; it may suit you better than the current one), it will be very difficult to retroactively change post-test8 kernels to allow GPLv3 after all.**
>
> **Remember, the cabal aren't writing any new license, RMS and his lawyer are. You may ignore speculation about the contents on slashdot, gnu.misc.discuss etc.**
>
> **... But these are the most important reasons not to use "GPLv2 only" yet:**
>
> 1. **If you accept a patch now to your "GPLv2 only" kernel, *you* won't have permission to retroactively permit "GPLv2 or later" on the patched code. You'd have to contact all the major contributors or remove the patched code.**

2. **If the GPLv3 is accepted in the long term and is applied to some good user space code, or code from another OS, *you* won't be able to incorporate that code into your "GPLv2 only" kernel. Even if you like GPLv3.**
3. **Similarly, *others* won't be able to take parts of the kernel and incorporate them into their GPLv3 user space code, or code for another OS.**
4. **You didn't write most of the device drivers. I think they should be treated as a public resource, available to other OS designers as well as user space, under the terms of their original authors. Which, we can best assume, is GPLv2 _or later_. Many of the device drivers may well outlive the Linux kernel that we know today.**

**For these reasons I think you should hold off the "GPLv2 only" declaration until such time as you actually see an officially published GPLv3. All speculation you've header about GPLv3's contents so far is just hot air.**

**It's good that you raised your concerns though.**

In reply to Jamie's idea that Linus could choose not to accept code that used version 3, Linus replied:

**As I already explained to Alan in a private email, it's not about me accepting other peoples GPLv3 code.**

**It's about the fact that when I chose the GPL, I did it because I wanted the source-code to be free and unencumbered. Forever. Whether I maintained that code or not. I didn't want my code to have any extra rules and regulations - the GPLv2 is already quite complex enough, but it is, in my opinion the "minimum required" complexity. So it suited and continues to suit my needs and opinions admirably.**

**The fact that I still maintain my own code and can**

> **choose to ignore other peoples code doesn't change that feeling. I want _my_ code to be out there, freely available, and unencumbered by any extra restrictions, forever and ever. Regardless of whether it is I who maintain it or not. I obviously won't be able to maintain it _forever_, after all.**
>
> **And I'd hate to see my code become part of something I don't like.**
>
> **Thus the (current) limitation to v2. And only, obviously, for code _I_ wrote and hold the copyright to.**

And to Jamie's statement that version 3 would probably just clarify its language without changing any of its basic meanings, Linus replied:

> **I hope so. And yes, in the end I _believe_ so. It's just that I used to believe so without even thinking about it. Last week some people opened my eyes to the fact that I can't just take it for granted.**
>
> **In the likely case that the GPL v3 is fine, I will license all my code under "v2-v3". Maybe it even clarifies the issue of "similar in spirit", so that I wouldn't need to worry at all about what the FSF considers "similar" ever again, and I can stop worrying altogether.**
>
> **I'm not against a new version of the GPL - I'm just spooked by some of the things that have been discussed that _might_ be part of a new version of the GPL. Things that would mean that if I didn't limit my code to the v2, future code maintainers might use my code with restrictions or other things that I never agreed to.**
>
> **Quite frankly, I'm probably just jumpy. And part of this is very much pre-emptive: making sure that the FSF knows that whatever they do, they have to take other peoples feelings into account too, and not just make a new version of the GPL on**

**their own whims.**

**So don't read _too_ much into this. It just means that the FSF does not have a blanket permission to expand upon the GPL as far as my personal code is concerned. Nothing more.**

Jamie noted that any attempt to relicense portions of the kernel under "version 2 or 3" would be very difficult due to the number of contributors. There was no reply.

We Hope You Enjoy Kernel Traffic