LINUXCARE SUPPORT FOR THE REVOLUTION

Your Account

ABOUT US | PROFESSIONAL SERVICES | LINUXCARE UNIVERSITY | LINUXCARE LABS | TECH

Home

Kernel Traffic #86 For 25 Sep

By Zack Brown

Table Of Contents

Introduction

Thanks go to Rasmus Andersen for catching me calling a 2G file a 2M file in Issue #85, Section #2 (**1 Sep:** "VM Patches Shaping Up, But Still Not Ready"). Thanks, Rasmus!

Mailing List Stats For This Week

We looked at 1337 posts in 5607K.

There were 395 different contributors. 200 (50%) posted more than once. 142 (35%) posted last week too.

The top posters of the week were:

- 79 posts in 253K by Alan Cox <alan@lxorguk.ukuu.org.uk>
- 37 posts in 147K by "Jeff V. Merkey" <jmerkey@timpanogas.com>
- 37 posts in 123K by Jamie Lokier <lk@tantalophile.demon.co.uk>
- 29 posts in 99K by Jes Sorensen <jes@linuxcare.com>
- 28 posts in 92K by Alexander Viro <viro@math.psu.edu>

1. Reaching For Full Preemption

6 Sep - 12 Sep (9 posts): [patch]2.4.0-test6 "spinlock" preemption patch

George Anzinger [*] posted a big patch to make the kernel nearly fully preemptive. According to design, the patch would be non-premptive only when handling interrupts, doing bottom-half processing, or when holding a spinlock, writelock or readlock. He also gave a link to a file describing the patch in more detail (ftp://ftp.mvista.com/pub/Real-Time/2.4.0-test6/preempt.txt). Latencies for context switching had been measured as high as 12 ms, though the patch developers were working on speeding up those places. Andrew Morton [*] replied:

> **From memory, there are three long-lived spinlocks which affect latency. Try applying http://www.uow.edu.au/~andrewm/linux/low-latency.patch (http://www.uow.edu.au/~andrewm/linux/low-latency.patch)**
>
> **Also, review Ingo's ll-patch. He may have picked up on some extra ones.http://www.redhat.com/~mingo/lowlatency-patches/ (http://www.redhat.com/~mingo/lowlatency-patches/)**

Andrea Arcangeli [*] also replied to George, with some sample code that would deadlock under George's patch. A quick 'grep' through the kernel sources by Andrea revealed a number of places that used the same construction and would deadlock in the same way. Rik van Riel [*] suggested George's sponsor (MontaVista) would be welcome to clean up all those places in the code, and Andrea added that the cleanup would be a good idea even without the preemption patch; and went on, **"Those cleanups can start in the 2.4.x timeframe but I'd rather not depend on them during 2.4.x to have a stable kernel. (2.5.x looks a better time to change such an ancient API) This is just my humble opinion of course."** George was very happy to hear about the areas needing fixing, and mentioned that this was what he and the other patch-developers had had in mind as well.

2. Module Directory Structure Changes

6 Sep - 12 Sep (13 posts): modules directory

Andrew Park noticed that the recent 2.4 test kernels installed modules in a different directory than previous versions. He asked if he should change 'conf.modules' so that modules could be found at boot, and Mohammad A. Haque suggested just installing the latest 'modutils'. Keith Owens [*] told Andrew, **"Why don't you look in linux/Documentation/Changes? That file exist precisely to stop repeated questions like this on the linux kernel developers list."** Simon Huggins pointed out that the 'Changes' file only listed the needed versions of packages, but didn't include information about *when* the upgrade became necessary. He suggested a regular step-by-step changelog-style file would be a lot easier to read and would encourage folks to refer back to it.

3. Patches For Big RAM, Better VM, Raw IO, SMP Performance, Large Files, And Other Stuff

6 Sep - 12 Sep (14 posts): 2.2.18pre2aa2 and patches for 2.2.18pre3

Andrea Arcangeli [*] posted a patch against 2.2.18pre2, and listed:

- **Support for 4Gigabyte of RAM on IA32 (me and Gerhard Wichert)**
- **Support for 2T of RAM on alpha (me)**
- **Improved VM for high end machines with enough ram and doing heavy I/O under high memory pressure plus fixes for the MAP_SHARED oom by killing kpiod. (me)**
- **RAW-IO (doable with bigmem enabled too). (Stephen C. Tweedie)**
- **SMP scheduler improvements. (me and partly from 2.3.x contributed by Ingo Molnar)**
- **LFS (>2G file on 32bit architectures)**
- **misc fixes**

Matthew Hawkins found the SMP parts of the patch really great, and had been using each of Andrea's patches for a while. He advocated putting the patches into the main kernel; Marcelo Tosatti [*] replied, **"Andrea VM patches will be included in 2.2.18. They are not included yet because Alan does not want two untested changes together in the same kernel (2.2.18pre3 has ext2 patches)."** But Alan Cox [*] replied, **"We'll see,"** and added, **"Right now I dont want to mix VM changes with the ext2 fixups and the large amount of driver work."** Elsewhere, Roeland Th. Jansen and others also said the patches were golden, and Alan said, **"I am slowly trickling them in as I get points I can be sure I can pin down problems the cause reliably. Andrea's patches tend to be far reaching in their effects and not always 100% obviously correct. Its the nature of fixing bugs in that kind of code."**

4. Best Compiler Versions For 2.2 and 2.4

6 Sep - 12 Sep (10 posts): Compiler warnings

David Gomez got a lot of warnings compiling 2.2.17 with 'gcc' 2.96, and Jakub

Jelinek [*], David Woodhouse [*] and Alan Cox [*] all warned him to use a different 'gcc' (Jakub recommended 2.4). As David put it, **"You cannot safely compile even 2.4 kernels with gcc-2.96 on any platform, as far as I'm aware. It's an insane thing to do. Use a sensible compiler."** And Alan put it, **"The kernel isnt ready for 2.96."** Out of curiosity, Keith Owens [*] asked which compiler Alan recommended for IA64 kernels, but Alan replied, **"Im not familiar with the needs for the IA64 tree."** Jakub and Jes Sorensen [*] agreed that 'gcc' 2.96 *would* be the thing to use in that case. Jes clarified, **"The discussion in question here was about 2.2.x anyway and there is no support for ia64 in that kernel."**

5. /proc API

9 Sep - 12 Sep (8 posts): How to put something in /proc

Giuliano Pochini [*] wanted to keep track of some kernel variables by creating a file in '/proc', and asked how to do it. Davide Libenzi [*] told him to search for examples of 'proc_register()' in the kernel sources, but Alexander Viro [*] closed the book with:

> **_Don't_**
>
> **proc_register() is dead. Use create_proc_read_entry() instead.**
>
> **Folks, support of the static procfs entries is gone and it will not be back. Any initializer for struct proc_dir_entry is a LARTable offense. So is kmalloc(sizeof(struct proc_dir_entry),...) and its ilk. You do it - you suffer.**

6. NFS In 2.2?

10 Sep - 18 Sep (57 posts): Linux 2.2.18pre4

Alan Cox [*] announced 2.2.18pre4, and Steven N. Hirsch [*] asked with a sigh, **"Alan, are the NFS client/server patches EVER going to make it into the base kernel?"** Alan replied, **"I still hope so but there is a maximum sane rate of change and its important to change stuff piece by piece. 2.2.18pre4 isnt the right place to change NFS."** Albert D. Cahalan [*] pointed out, **"Over on the freebsd-questions mailing list you can see desperate people trying to convert Linux systems over to that other OS to escape Linux 2.2.xx NFS. This is kind of serious, you know?"** Alan replied, **"Shrug. So you want me to make it worse by shipping unproven code in a way I can't test it?"** He added:

> **FreeBSD people flow steadily to Linux too. Personally I don't actually care See if FreeBSD is the better OS for a given job then use it for that. If Linux is the better OS use that.**
>
> **2.2 is the stable branch, my job is to keep it stable. That means I have a permanent queue of neat things I really want to apply but can't right now.**

Rik van Riel [*] suggested, **"Why not use the NFS patch that almost all Linux distributions have been shipping with for months?"** Alan answered, **"Most of those patches are in I believe. Whats needed next is to jump to the major revision."**

KT covered this discussion way back in Issue #34, Section #15 (**31 Aug 1999:** "NFS In The Linus Tree"). By Issue #61, Section #14 (**21 Mar:** "Status Of NFSv3") the patches had still not gone in, and by Issue #71, Section #6 (**27 May:** "Troubles Getting NFS Fixes Into 2.2.x") developers were on their hands and knees.

7. Threading: The Saga Continues

11 Sep - 13 Sep (15 posts): [BUG] threaded processes get stuck in rt_sigsuspend/fillonedir/exit_notify

David Ford [*] noticed that in the recent 2.4 kernels, 'kill -9' would leave zombie processes. Ulrich Drepper [*] replied:

> **The thread group changes broke the signal handling in linuxthreads. The CLONE_SIGHAND is now also used to enable thread groups but since linuxthreads already used CLONE_SIGHAND and is not prepared for thread groups all hell breaks loose.**
>
> **I've told Linus several times about this problems but he puts out one test release after the other without this fixed.**

(Later, Ulrich added to David, **"This is completely unrelated. The fix for your problem is to change the CLONE_SIGHAND flag back to it's original behavior. Changing linuxthreads to take advantage of the new kernel functionality is on a different plate."**)

Ray Bryant volunteered to work on fixing linuxthreads if there were any documentation, but Ulrich replied, **""Fixing" alone won't cut it. I've started a rewrite and send Linus more comments about what is needed but not even got a reply. Seems the short interest span is already over."** Elsewhere, David asked what would be involved in fixing it, and Theodore Y. Ts'o [*] replied:

> **I suspect it's going to require a kernel developer who is willing (and has the time) to tackle this as their own project, preferably someone who is trusted by Linus for having "good taste".**
>
> **The only problem is that the set of people who are (a) kernel developers, (b) understand Posix threads in all of their ugly, gory detail, and (c) have survived with their sense of good taste intact seems to be relatively small.**

David figured the new stuff should be backed out until the whole thing could be brought back to working order, and Ted replied, **"I didn't realize things had changed that broke the old threading model. Did Linus do more than add support for the new thread groups? I didn't think any that had changed that would break the old LinuxThread programs."** David replied that *something* had changed, since all his threaded apps had become highly unstable with recent 2.4 kernels; and Ulrich explained, **"First he introduces CLONE_THREAD (or how it was called). This was fine. But in pre2 ore pre3 he unified CLONE_SIGHAND and CLONE_THREAD under the new name CLONE_SIGNAL which makes perfect if CLONE_SIGHAND would be used. But it is. Simply undo this change, separate the two flags."** Mark Kettenis [*] replied, **"This has been fixed in test8 (final). CLONE_SIGHAND and CLONE_THREAD are seperate bits again, and CLONE_SIGNAL is both of them or'ed together."** David reported that something else must have been broken as well in that case, since he was still seeing problems; but there was no reply.

For more on the ins and outs of this issue, see Issue #62, Section #4 (**24 Mar:** "POSIX Threads; Philosophy Of Kernel Development") and Issue #84, Section #1 (**23 Aug:** "Posix Threads (pthreads) In Linux")

8. Backporting The IDE Patch

12 Sep (5 posts): Linux 2.2.18pre6

Alan Cox [*] announced 2.2.18pre6, and Andre Hedrick [*] replied:

> **I see you parse-pieces for the 2.4 code as backports into 2.2 of the larger ide-patch. What is it going to take to just accept it?**
>
> **The ali-driver is straight out of 2.4 or ide-patch with parts of cmd646 taken to actively tune the drive and chipsets.**
>
> **If it is the issue of the taskfile now in my patch, I will pull it if it will prevent me from having to carry this monster around.**
>
> **I am considering the pull of the backport because I do not have time to do both directions.**

Alan replied, **"I'd prefer to do it bit by bit, driver by driver without touching the core and picking the important and stable drivers first."** He added:

> **If you can hand the backport on then I think that would be a good idea. Getting 2.4 stable is far far more important than the 2.2 backport, and getting taskfile ready for 2.5 likewise.**
>
> **You seem to have a new via maintainer perhaps you can interest him ?**
>
> **For the continuation, see Section #10**

9. Using Netware Elevator Ideas In Linux

12 Sep - 14 Sep (20 posts): elevator code

Rik van Riel [*] asked Jeff V. Merkey [*], **"since I vaguely remember an email from you describing how you spent <large amount of time> tweaking and changing the disk IO elevator in Netware, and since we might want to improve the Linux elevator sort a bit, could you give us some hints on what to do and where not to waste our time? ;)"** Jeff replied (quoted in full):

Here's a complete description of the internal architecture of the NetWare elevator and async disk I/O subsystem.

ARCHITECTURE OF THE NETWARE ELEVATOR AND DISK SUBSYSTEM

The NetWare Disk subsystem is called the "NetWare Media Manager (MM)" and supports all elevator, async I/O, and mirroring/hotfixing for the NetWare operating system. It is logically defined as four separate layers between the server file cache to the disk drivers as follows:

- **Layer 1**
  NetWare File Cache/LRU
- **Layer 2**
  MM Mirroring/Hotfixing/Segmentation Layer
- **Layer 3**
  MM Elevator/Disk Heads
- **Layer 4**
  NetWare Disk Drivers

LAYER 1

In NetWare, all free memory in the entire system is owned by the file cache, and the NetWare memory manager and VM agent actually sits on top of the file cache. Although Linux is not implemented the same way, this would be akin to Linux using the buffer cache set to a block size of 4K as the "page pool" for the VM, kmalloc(), etc. The advantage to this approach for Netware is clear since it's a file server. All the available memory is used as LRU file cache, and applications alloc and free from the file cache in NetWare if they need memory. All memory allocation requests actually get free pages from the file cache itself, which alleviates the need to balance LRU memory and app memory in the server when one starves the other, which is a problem Linux has to deal with today. In NetWare, the problem does not exist -- it's an LRU

policy how much memory apps can get access to. The file cache is hard coded to 4K blocks in IA32, and other architectures NetWare has been ported to like SPARC, MIPS, etc. emulate this 4K block model. This is why NWFS on Linux uses the IO_BLOCK_SIZE defined set to 4K. It's also why the NWFS LRU is hard coded to 4K blocks to emulate this architecture.

The file cache is divided into four pools. Memory below 1MB (for DOS and 20 bit DMA buffers and RPC call structures support), memory below 16MB (for 24 bit DMA disk drivers that need buffers below 16MB), non-Movable Cache memory (memory that has been pinned to page tables, gdt tables, OS Data and OS Code), and movable Cache memory (for app memory for code and data, file data, VM and stack memory, etc.). Movable Cache memory is used to cache file and data blocks for the system.

When the NetWare file cache submits an I/O request to the mirroring agent, however, these requests are always translated from 4K blocks into sector relative offsets and the file cache in NetWare always views a volume as a logical entity starting a sector 0....n and translates 4K blocks into sector relative LOGICAL volume offset. For example, if a file mapped to block 2 on volume, this would translate into 2 * sectors_per_block(8) or a logical LBA offset of 16, which is identical to how NT and W2K also maps NTFS volume I/O. It was done this way to allow the File Cache the ability to perform I/O to the granularity of 1 sector up to the size of the disk (though it typically uses a cluster size of 64K as max for native NetWare).

LAYER 2

The MM Mirroring Agent is responsible for dupping I/O requests, and translating logical volume LBAs into physical disk LBAs. NetWare striping, mirroring, and segmentation are performed at this layer, and it is at this layer that remapping of I/O LBA offsets to a particular device occurs. In NetWare, disk I/O requests are generated by the File Cache as logical volume LBA's, and passed to this layer. NetWare I/O does not enforce fixed block sizes as is done by Linux. In NetWare, a MM Disk I/O request is very simple and consists of <disk #, LBA start, number of sectors, buffer pointer, callback function>. There is no block aligned enforcement as exists in Linux and the MM interface allows disk requests to start anywhere on the device and can be any length up to the size of the disk. At this layer, NetWare maintains a map of mirrored and segmented devices, and if it detects that a device has one or more mirrored partners, the I/O request gets dupped and mapped for each mirror that exists. The same appplies to volume segments. This layer performs the remapping of I/O requests for volume

segments that may span disks. This layer should be viewed a a mapping agent and does little else. The exceptions to this statement relate to hotfixing and remirroring. This layer owns the remirror daemon, and if it detects a mirrored partition is out of sync, it spawns the remirror process which re-syncs partitions that are not in-sync, such as a disk that failed or a new member being added to a previously existing mirror group. NetWare alows up to 8 partitions to be mirrored with each other.

LAYER 3

The layer beneath the mirroring agent is the MM elevator. The elevator in NetWare is structured as a series of an A(queue) and B(de-queue) queues for each disk that resides in the server. At this point, things get a little complicated to understand, however. The design decision to use two queues for each disk was chosen to solve the problem of elevator starvation. In Linux, merging of I/O requests occurs at this layer. In NetWare, oddly enough, the original implementation in 286 NetWare was very similiar, however, a newer model was adopted based on lessons learned in live customer accounts. In NetWare, requests are merged at A) the boundry between the File Cache and the I/O subsystem, and B) in the drivers themselves and NOT THE ELEVATOR.

Disk Drivers in NetWare use two functions to service requests, GetRequest() and PutRequest(). When an I/O request is submitted via PutRequest(), it is first placed on the disk's A queue and NetWare drivers provide a poll() type function that gets called by PutRequest() when an I/O request is first posted. If the driver is already busy, it returns a BUSY status and at this point the thread that inserted the request returns to the caller. If the driver returns an idle status from poll(), the driver poll() function will call GetRequest(). GetRequest() will first check the B queue for a chain of disk requests, and if the B queue is empty, it will take the entire list on the A queue, move to the B queue, zero the heads on the A queue, and allow the driver to take one or more of the I/O requests on the B queue. Disk completion interrupts will cause the disk Drivers to call the callback post routine in each I/O request the disk has completed, then the next thing the driver will do is call GetRequest() until the B queue is completely empty. Once the B queue is empty of requests, GetRequest() will again check the A queue for new requests and move the list to the B queue and start processing the I/O request chains all over again. If an incoming I/O is posted with PutRequest() and the driver poll() returns BUSY and is servicing a chain of I/O's on the B list, it will just drop the request on the A queue, elevator index it in the chain and return.

This mechamisn completely avoids the problem of elevator

starvation by using an alternating A and B list. There are some optimizations that occur here that allow the disk drivers to merge requests. In NetWare, the disk drivers are designed as "smart" drivers, and it is inside the drivers themselves that merging occurs. This is different than Linux. Linux disk drivers are fairly "dumb" drivers by comparison. The design decision to do it this way was based on study of different vendor cards, and many of the boards themselves were found to have more "hints" about how the intelligent merging of I/O requests should happen based on the interface designs and subtle features that each hardware vendor could instrument to make this process more efficient and provide faster I/O bandwidth.

The elevator maintains a special set of linkages in the I/O request chain, and if requests were found to be continguous, a special set of links were instrumented that would give the drivers "hints" that a chain of one or more I/O requests could be merged. The B queue was always assumed to be owned exclusively by the disk driver, and the drivers calls to GetRequest() were the actual mechanism that moved and manipulated the I/O chains on the A and B queues. NetWare disk drives that used this optimization would typically take the entire B queue in one GetReuest() call if they were linked as contiguous, and issue a single I/O operation in the hardware underneath. We tried doing what Linux does and merge the requests above the drivers, but found that just giving the driver the chain and allowing the driver to decide produced higher performance numbers for certain boards, reduced processing overhead, and memory usage. The NetWare File Cache also merged requests as well, since MM I/O requests are not fixed blocks like Linux, but varible length sector requests. If a 64K cluster needed to be written, it was always sent as a single 64K I/O request.

LAYER 4

Disk drivers in NetWare are described above. One important difference is that in Linux disk I/O must be explicitly inititiated with a call to run the tq_disk routine. In NetWare, submission of requests would kick the A and B queues into the driver automatically without needing this extra step. I understand why Linux did it this way -- to allow the elevator to fill up. The NetWare model's use of an A and B list circumvented the need for an external kicker by using this method.

I hope this explains some of it. Feel free to ask for more info. People at times think I don't get it in Linux, but the fact is I do, I just know it's pointless to argue or debate about stuff with folks since there's a lot of passion involved, and insulting people's code will

just get a shotgun blast directed this way. I see Linux hitting a lot of the same obstacles and problems I saw a decade ago working on NetWare, and it's a temptation jump in and add my two cents, but I know unless folks find out for themselves, they will ignore me a lot. I remember going to NetScape in 1995 with Ty Mattingly (Ray Noorda's righ hand man and a personal friend of Bill Gates), and sitting around a table with a bunch of 26 year old millionaires while Ty tried to explain to them how Microsoft was going to crush them into oblivion, and they discounted every word he said. Netscape doesn't exist today.

Everyone was generally impressed with this elevator model, and Martin Dalecki [*], Horst von Brand [*], Andrea Arcangeli [*], Andi Kleen [*] and Hans Reiser [*] discussed various details with Jeff.

10. New Maintainer For ATA Backport

12 Sep - 13 Sep (3 posts): 2.2 Backport Terminated ...

Continuing from Section #8, Andre Hedrick [*] announced he had no time to keep up the ATA backport to 2.2, and gave his current To Do list:

- **Finish Stablity of 2.4.0**
- **Finish CASCADE for 2.4.0 and introduction @ ALS 8 drives per channel, or 16 drives per card.**
- **Finish ORBS Castlewood for 2.4.0**
- **Finish ARCO-IDE RAID for 2.4.0**
- **Finish ONION-BUG fix without TASKFILE IOCTL for 2.4.0**
- **Parse 48-bit LBA for 2.4.0/2.5.0**
- **Finish TASKFILE for 2.5.0**
- **Prep for SerialATA SuperSets for 2.5.0**
- **Finish Drive Certification Model and test Suite.**
- **Stop pissing Linus off on a regular basis ;-)**

Bartlomiej Zolnierkiewicz [*] volunteered to maintain the backport, maybe even on a regular basis.

11. Linus To Adopt Patch Queue System!!!

13 Sep - 15 Sep (62 posts): Proposal: Linux Kernel Patch Management System

Daniel Quinlan [*] announced:

Here is a proposal to improve the kernel development process. It was co-written by Sebastian Kuzminsky, Linus Torvalds, Theodore Ts'o, and myself. We are posting the proposal here for public review and comment. Seb is the guy writing on the software; he's nearly done the initial version.

Description of the problem:

1. **There is no system that archives and tracks Linux kernel patches.**
2. **There isn't a good way of marking that a particular patch is believed to address a particular problem on the TODO list. (Patches should be tied to the TODO items.)**
3. **There is no archival system and no easy way to determine which patches have made it into the kernel.**

Proposal:

1. **Developers submit all Linux kernel patches to kernel-patches@kernel.org (not in place yet, so don't start sending patches).**
2. **Each patch will conform to a standardized, but simple, text format, which looks something like this (exact details to be determined):**

```
-------
To: kernel-patches@kernel.org
Subject: this is a short description of the patch

<tag 1>: <whatever>
<tag 2>: <line one>
    <additional lines are indented>
<more tags>

<patch>
-------
```

Required tags:

> **"Version" - the base kernel version. For example, "2.4.0-test8-pre1". The web page will list valid version strings.**
>
> **"Description" - a detailed description of the patch.**

Optional tags:

> **"Fixes" - followed by one or more bug numbers (tracked by tytso for now). For example, "T0001" might be tytso bug number 0001.**
>
> **"Obsoletes" - followed by one or more kernel-patch identifiers. For example, "KP7555".**

"Requires" - followed by one or more kernel-patch identifiers. For example, "KP7555".

"Cc" - followed by one or more email addresses to be carbon-copied on success.

"Flags" - followed by one or more supported flags. For example, "experimental" (that is, don't submit anything to Linus).

The tags are basically in RFC 822 format, but are placed in the body of the email. (Additional lines are preceeded by whitespace, tags are followed by a colon, etc.)

Linus wants the body of patches to be in text format and not MIME-encoded or uuencoded.

3. A robot will process all patches for correctness (mostly, does it apply?) with the possibility of additional tests later such as compilation tests, regression tests, etc.
4. If the robot likes the patch, it will create a unique identifier (i.e. "KP7562") and prepend a log entry to the body of the patch:

```
--- linux/Documentation/patch-log Tue Aug 29 17:24:37 2000
+++ linux/Documentation/patch-log Tue Aug 29 17:24:44 2000
@@ -1 +1,3 @@
+Applied patch KP7562 (2000/08/30)
+ Synopsis: <short description> (<author>)
+
```

(Yes, this prepend patch will always successfully apply.)

Good patches are sent to the linux-kernel mailing list which is also where additional discussion about these patches can take place. All patches (good and bad) will be logged and there will be a web interface to access the patch database.

We had some amount of discussion about whether a separate mailing list would be a good idea, but we ruled the idea out because fragmenting the kernel-related discussion would have negative effects on development. If it becomes a problem, we can always separate it later.

If the patch is long, the actual body of the patch won't be included in the email to the discussion mailing list, just a URL to the patch.

Also, information about each applied patch can be retrieved

from the patch web site (using the identifier).

5. **If the robot doesn't like the patch, it will send the patch back to the submitter with a failure report.**
6. **When and if Linus applies an entire patch, the patch-log will be updated with a record of the changes. If Linus applies a partial patch, then he will remove (or edit) the patch-log section of the patch.**

Notes:

- **The web site will document version strings that will work with the server. Patches against unsupported versions will probably not work and should be rejected, sent to somewhere else, etc.**
- **This system can be put into place quickly.**
- **Linus should reject most out-of-band patches if this is to work.**

Future features?

- **PGP signing of patches**
- **conversion of uuencoded patches to text format for people with broken mailers**

Folks took this in stride, though it addressed one of the most bitter debates in Linux kernel development. A number of folks have been truly angry at Linus for not adopting something like this long ago. And many heartfelt proposals have seemed at times to fall on deaf ears over the past few years. Since this time the proposal came from Linus, it seems much more likely that he will eventually adopt it or something similar.

There were a bunch of suggestions from various folks. Someone suggested 'bugzilla' as a way to track fixes instead of using Theodore Y. Ts'o [*]'s numbering system. Ted replied:

It's an open question whether it's less work for me to read through all of linux-kernel, looking for bug reports, and filing them myself, OR run something like bugzilla, and then have to winnow out all of the bogus/bullshit patches which people submit --- and then have to scan l-k anyway, since a lot of people won't bother to use the formal web submission tool.

If everyone used it, and it was integrated into a full software development process that included a software control system, sure; that's what bugzilla was designed around, and it's not bad at doing what it was designed to do. But given the somewhat chaotic development process used by the kernel, simply throwing in bugzilla without putting in the rest of the changes necessary to really make it work well is probably a bad idea. And I don't think we have the mandate from the developers and from Linus to make that kind of major change to how the Linux kernel is developed.

Elsewhere, Mitchell Blank Jr [*] opined that the original proposal would only work if Linus would refuse to look at patches not sent through the queue. Alan Cox [*] replied, **"If that attitude is taken the large numbers of patches will never make 2.4 proper."** And Ted added:

> **I've never asked for that attitude. If only 50% of the patches go through this system, my job will be made *much* easier.**
>
> **That being said, the existing system isn't perfect. I can tell you that there are a large number of "obvious" bug fixes which haven't been making it into 2.4 proper. I do try to keep track of patches, although in a much more informal way than actual bug reports. Some of them are very simple error checking fixes which got submitted a month or two ago, got ignored by Linus, and are pretty much forgotten. If the patch author doesn't aggressively submit (in my personal experience I've had to sometime retransmit three or more times), the patch can get dropped, even if an outside observer thinks that it's an obviously good patch.**
>
> **I have an RMAIL folder containing about 120 patches that fall in this category. Some of the patches may still require some fixing, and some of the patches may have since been applied (although every so often I sweep through the folder and try to eliminate those that have already been accepted). My guess is that percentage of "good patches" that haven't been accepted is probably 50-75%. If anyone would like to go through that list and retransmit some of the more obviously correct ones to Linus, let me know, and I'll send you that RMAIL folder.....**

Elsewhere, Alexander Viro [*] replied to the original proposal, **"Sigh... You know, some stuff is security-sensitive. Dunno about other folks, but in such situations I prefer to send the patches OOB to relevant maintainers. And they often go through several rewrites before they go into the tree. Having descriptions of _all_ pending patches in publicly accessible place may become an interesting problem."** Daniel replied, **"We could add a "security" or "private" flag that would do the right thing. I haven't asked Linus about how he wants security-sensitive stuff handled (OOB or with the kernel-patch system but not public)."**

Elsewhere, David S. Miller [*] proposed setting up an 'rsync' daemon to make Linus' tree publically available on an ongoing basis. He explained:

> **I don't want to sift through a log file on some web site etc. to find out what he's applied. Very simply, (drumroll please) I want to run diff. :-) With a whole "his tree vs. my tree" diff I also can immediately spot all sorts of problems, bugs, and conflicts that might otherwise go undetected until a later time. It seems to me that one of the purposes of this log+patch_archive is to be able to**

> **know what is in his tree. Why be so indirect about this? Just show us exactly what is there in a unified format, ie. the live sources themselves.**
>
> **With the "apply log" and patch application checker robot setup, there will always be a period of limbo where it's really difficult to put together a clean diff against a tree which really represents the current state of affairs in Linus's live sources. An rsync type setup would allow me to send a clean patch at just about any time.**
>
> **True, if Linus does not accept out of band patches, this is unlikely to be a real problem except for the issue how much indirection happens here to produce "a tree near-exact to Linus's". But Linus actually writes code too, is he going to be required to submit his patches to this site? I hope not :-)**
>
> **This kind of sillyness is why I think the live rsync, or some equivalent, is what is really needed at least for high bandwidth patch submitters like myself.**

Ted replied that David's idea was really orthogonal to the proposal, but was a good idea nevertheless. Daniel agreed it was not really related, though it could be useful. David added that Daniel's proposal **"has the potential to make more work for those of us who do our patch submissions effectively already."** Ted replied:

> **How can we simplify things? Part of the design of this new proposal was to change as little as possible from the existing setup (people's habits are hard to change), and yet to make my life and Linus's life much easier. In the long run, it will make your life easier, to the extent that having an up-to-date bug list is easier, and because then I won't have to continually pester people about whether certain bugs have been fixed.**
>
> **Right now, having to paw through diff files to see when Linus has applied a particular patch (add grumble about lack of a source code control system) is really not fun. Alan did it for a while, and burned out, and I can tell you, I can't really blame him --- it's a lot of work.**
>
> **Is it really that hard to annotate the patch with a bit of information, and then send it off to a different mailing address instead of sending it directly to Linus and the l-k list?**
>
> **What can we do to make things simpler on developers? Certainly this isn't going to work unless the developers use it, and that means we need to keep things as easy as possible for the patch submitters.**

The discussion went in several different directions at this point. Randy Dunlap [*]

suggested (peripherally) that Ted break his "To Do" list into sections maintained by actual code maintainers, and Ted said he'd happily let any code maintainers take over the relevant sections of his To Do list. Elsewhere, Richard Gooch [*] suggested (also peripherally) patching the 'patch' program to optionally send email to the author of the diff being applied. Russell King [*] replied that a wrapper for 'patch' would do as well. Richard asked Linus if this were a possibility, but there was no reply. Elsewhere, David replied to Ted's question, saying he had enough respect for Ted to give the proposal a try regardless of his own reservations. But he added, **"I contend that some of us will feel that they are being punished for doing a good job thus far without this patch robot."**

Elsewhere but also in reply to Ted's question, Rogier Wolff [*] compared the whole proposal to BitKeeper (http://www.bitkeeper.com/). Ted replied, **"What we've implemented is a very small subset of the sort of features that bitkeeper has. The problem with bitkeeper is that it's **so** different from CVS that it takes time to learn --- I spent a day getting my head wrapped around it, and I still wouldn't call myself an expert; that's what's necessary to really get a good feel for how it works and why it's so nice. The problem, though, is that bitkeeper is only useful if a large number of other developers use it, and given its non-OSS license, it's not clear it will get that critical mass. Personally, I have no problem with the license. But if there are enough other people who are license fanatics who do have a problem with it, then bitkeeper loses a lot of value for me. If Linus were willing to dictate from high that we were going to use bitkeeper, and that all patches had to come in as bitkeeper changelogs, then that might get us critical mass. If he doesn't do that, though, my big concern is whether or not it'll be able to garner enough critical mass for it to be worth the trouble for kernel developers to want to spend time learning it."** The rest of the discussion centered around the relative merits of BitKeeper.

12. VM Patches Looking Good

14 Sep - 15 Sep (10 posts): [PATCH *] VM patch for 2.4.0-test8

Rik van Riel [*] announced:

> **The new VM patch seems has received a major amount of code cleanup, performance tuning and stability improvement over the last few days and is now almost production quality, with the following 4 items left for 2.4:**
>
> - **improve streaming IO performance**
> - **out of memory handling**
> - **integrate Ben LaHaise's readahead on the VMA level (and make drop_behind() work for that) .. fixes kswapd cpu eating**
> - **(maybe) make drop_behind() work better for some cases**
> - **testing, testing, testing, testing ...**
>
> **The post-2.4 TODO list contains these items:**

- **physical page based aging (reduce kswapd cpu use more and do better/more fair page aging)**
- **much much better IO clustering (neatly abstracted away?)**
- **page->mapping->flush() callback for journaling and network filesystems (maybe later in 2.4)**
- **thrashing control (like process suspension?)**

The new VM already seems to be more stable under load than the old VM and tuning has taken it so far that I'm already running into bottle necks in /other/ places (eg. the elevator code) when putting the system under rediculously heavy load...

I haven't had much time to do things like dbench and tiobench testing though, which is why I'm sending this email and asking the enthousiast benchmarkers to give the patch a try and tell me about the results.

Oh, and please don't restrict yourself to just the synthetic benchmarks. The VM is there to give the best results for applications that have something like a working set and has not been tuned yet to give good performance for benchmarks (which seem to run very much different from any application I've ever seen).

David S. Miller [*] caught a bug in some of the Least Recently Used (LRU) page tests, and Rik replied, **"This bug certainly explains some of the performance things I've seen in the stress test last night..."** He added, **"the box /survived/ a stress test that would get programs killed on quite a few "stable" kernels we've been shipping lately. ;)"** He made a new patch incorporating Dave's fix, and said, **"Unless somebody else manages to find a bug in this patch, this will be the last patch at this feature level and the next patch will contain a new feature. The new feature in question will be either the out of memory killer, or Ben LaHaise's readahead-on-VMA-level code."**

13. Possible Microsoft Litigation Over NTFS Support In Linux

14 Sep (5 posts): Re: Microsoft NT/W2K/Linux NTFS Repair/Debug Tool

Jeff V. Merkey [*] announced, **"Microsoft has threatened us with litigation due to our support of Linux NTFS development, and we have dissolved our NTFS licensing agreements with Microsoft in response to their demands that cease to support Linux development. Microsoft demanded that we delete any and all NTFS tools we had been providing to customers based on their intellectual property. As a result of this, we can no longer provide this tool in the United States."** Andre Hedrick [*] asked, **"Wait they attacked you after the request for cross over support?"** Jeff replied:

Yes Andre, they did, they accussed me of knowingly conspiring with Linus to provide full NTFS on Linux based on the email you and I sent to them. The agreements they signed with us were very liberal, and allowed us to create any tools and NTFS stuff we wanted, there were no non-competes, or anything to stop us from providing stuff on Linux. What they alleged was a belief that since we were going open source and supporting Linux NTFS users, they believed it was impossible for me to keep a "chinese wall" in place in my head between their IP and Linux IP. A very valid example of the legal theory of the "doctrine of inevitable disclosure".

But I must admit, in fact what's going on here is that by pulling the open source NDS for W2K off the table, I renigged on a "faustian" agreement to open source NDS on W2K. This was compounded by the fact that we released the MANOS sources with a complete NT/W2K PE and DLL loader (which we wrote "clean room", one of their old tricks). They found this very irritating. They were quite unconcerned about our NTFS work on Linux until we posted MANOS and announced an Open Source NetWare.

We've started our "clean room" NTFS core and I've spent some late nights working on it, and we doubt they will take any action since we dissolved the agreement. The last thing they need is for me to take the stand and testify just what kind of deals they offered to get us to leave Novell in 1997 and divide the NetWare markets by using the "Linux IP Laundry-Mat" to launder Novell's NDS for their consumption (Oh! Look what we found on the internet and downloaded today!).

NDS would be a useless wart on the rump of Linux. It's for managing large numbers of file and print servers, not internet/intranet servers like Linux. Linux already has vastly superior internet directory capabilities.

Andre replied to Jeff's first paragraph:

Wait, this was a proposal of mine to MicroSoft to grant permission development in a clean room model that only used white papers or other stuff that could be extracted passively.

I alos pointed out that this simple act of allowing open development of a public NTFS would help them blow holes in the DOJ monopoly issue.

Jeff replied:

The way they took this was that we had changed sides in the war,

> **since I was perceived to be approaching them with you. Here's what they said about you,**
>
> **"... We are concernd about the veracity of your associates. Despite the representations they have made to you, we have not been taking GPL code from Linux and using internally at Microsoft. This approach by these Linux people is little more than an attempt to [blackmail Microsoft] with unsubstanciated rumors. We see no benefit whatsoever to provided NTFS R/W capabilities on Linux ..."**
>
> **Not very nice to be sure. I know that black and white markings (like a penguin) are in style right now, but white and black stripes are not ! :-)**

But he concluded, **"I have the ability to litigate against them. They know this and I doubt will go any further than to bluster and threaten."** End Of Thread (tm).

14. 2.0.39: The Saga Continues

15 Sep (3 posts): [ANNOUNCEMENT] pre-patch-2.0.39final

David Weinehall [*] announced 2.0.39final, and said, **"Haha! This is final 2.0.39, whether you like it or not! Noone screamed too loudly about pre8, so it can't be that bad. Oh, and don't bicker and argue about the devfs files. It's not devfs itself, only headers. They won't disturb your kernel one bit. They might simplify backporting of drivers, though."** Alan Cox [*] replied:

> **Whine bicker bitch moan complain ;)**
>
> **I'll go over it in detail early next week**

And David speculated:

> **You know, one thing that seems to be a general truth is that people don't report problems until you hold the gun to their head, and count down from 10. Usually, they speak up at 2 or 1. It's probably due to seeing too many Hollywood movies.**
>
> **To prove this, I got a bugreport today. A bug that seems to have been introduced in pre4, which was released 25th of April. One would expect that bug to have been reported sooner, right? Hell no. Why report the bug when you can fall back to pre3 and be happy?! :^)**
>
> **Oh well. The backdraw with announcing a final is that it only works once or twice, then the people who don't experience bugs and want the proper final version get fed up.**

EOT.

15. New VM Goes In 2.4: The Saga Heats Up

15 Sep - 16 Sep (3 posts): test9-pre1

Linus Torvalds [*] announced 2.4.0-test9-pre1, and said, **"Ok, the new MM balancing has been getting good reviews, and no longer has any known issues. Integrated. We'd have needed to do it sooner or later anyway (just see what happened in 2.2.x), the sooner we do it the less traumatic it will end up being."** Bert Hubert [*] let out a cry of joy, and Rik van Riel [*] added:

> **Note that the big performance tuning has only just begun...**
>
> **Now that the number of users of the new VM code has increased tenfold overnight, I'm sure I'll get a lot more reports of workloads where performance isn't right yet...**
>
> **(even though most workloads should see increased performance)**
>
> **It's almost a shame I'll be at Linux Kongress next week and only sporadically connected to the net ;)**

End of thread.

16. NFS Patches In 2.2; Yes, 2.2

16 Sep - 8 Sep (3 posts): Linux 2.2.18pre9

After Section #6, it didn't look too promising for NFS in the stable tree, yet in this thread Alan Cox [*] announced 2.2.18pre9, which included the changelog entry: "NFSv3 support and NFS updates (Trond Myklebust [*] and co)". Henning P. Schmiedehausen [*] replied, **"Biggest understatement this side of Linux 2.4. :-) Cool. Will test."** eot.

We Hope You Enjoy Kernel Traffic