conference reports

This issue's reports are on the Linux 2.5 Kernel Developers Summit

OUR THANKS TO THE SUMMARIZER:

Rik Farrow, with thanks to La Monte Yarroll and Chris Mason for sharing their notes.

For additional information on the Linux 2.5 Kernel Developers Summit, see the following sites:

<http://lwn.net/2001/features/KernelSummit/>

<http://cgi.zdnet.com/slink?91362:12284618>

<http://www.osdn.com/conferences/kernel/>

Linux 2.5 Kernel Developers Summit

San Jose, California March 30-31, 2001

Summarized by Rik Farrow

The purpose of this workshop was to provide a forum for discussion of changes to be made in the 2.5 release of Linux (a trademark of Linus Torvalds). I assume that many people reading this will be familiar with Linux, and I will attempt to explain things that might be unfamiliar to others. That said, the odd-numbered releases, like 2.3 and now 2.5, are development releases where the intent is to try out new features or make large changes to the kernel. The even-numbered releases are considered the stable releases.

I got my first impression of the people attending the conference at the Thursday night reception. There was only one woman out of the 65 registered attendees, but other than that, this appeared very similar to any other USENIX event. The real difference is that few of these people were system administrators, and most were kernel hackers. I walked around asking people what their focus area in the kernel was. That question turned out to be hard to answer, even though I could make some guesses by looking at the Kernel Developer's mailing list traffic summary: *<http://kt.zork.net/kernel-traffic/latest.html>*. For example, Rik van Riel, originally from the Netherlands but now working for Conectiva in Brazil, focuses on virtual memory and memory management (VM and MM). I also met one or two "kernel janitors," programmers who clean up kernel code, remove defunct code, etc.

I was also pleased to discover that this meeting, put on by USENIX and OSDN (*<http://www.osdn.com>*) and sponsored by IBM, EMC, and AMD, was the first opportunity for many of these people to meet in person. Perhaps this is not so amazing given the distributed nature of Linux development, but I certainly thought that, in all of this time, someone would have brought this group together before.

Another difference appeared when the first session started on Friday morning. The conference room was set up with circular tables, each with power strips for laptops, and only a few attendees were not using a laptop. USENIX had provided Aeronet wireless setup via the hotel's T1 link, and people were busy typing and compiling. Chris Mason of OSDN noticed that Dave Miller had written a utility to modulate the speed of the CPU fans based upon the temperature reading from his motherboard. Another person whipped up a quick program to test an assertion made by the first presenter about degraded performance in the 2.4 release.

Requirements for a High Performance Database

Lance Larsh, Oracle Corporation

If you thought that having big business make suggestions about improving the Linux kernel would be poorly received, you would be wrong. In fact, I could tell that attendees were mostly receptive, as they want Linux to become a better commercial OS platform.

Larsh began by explaining a little about how Oracle works. He also told us that databases like to do raw I/O, and that this is not much of a benefit in the current Linux implementation. For example, even if a continuous batch of sectors is to be written, the kernel breaks it up into smaller batches and adds a buffer header to each sector. Another problem had to do with the elevator algorithm, which sorts and merges requests based on their physical location on a hard drive (spindle). Another problem involved io_request_lock, a global lock that Larsh suggested should be per device unless global synchronization was really required.

Larsh also suggested that Linux do away with the elevator algorithm and let the hardware do the work. Linus Torvalds asked if Larsh had tried setting some elvtune parameter to one, and Larsh said he hadn't. One thing that became clear to me was that most of the Linux kernel developers were software guys (something that Andre Hedrick really made a point of later). Modern hard drives reorder the physical location of tracks on the fly based on the current location of the heads, so using any elevator algorithm makes little sense.

Oracle also has problems with the memory model used by Linux. Some IA32 (Intel x86) -based systems can have in excess of 4GB of RAM, but Linux device drivers handle this by using a bounce buffer to copy data to a region below 1GB, losing performance to the copy. Asynchronous I/O was also a problem, as is the use of the O_SYNC and O_DSYNC flags. Quite a lively debate started at this point, with one participant saying that O_DSYNC was the default in 2.4.

Then Larsh dropped a bombshell. He reported that an SMP system with SCSI drives was 10 to 15 times slower, measured with iozone, in 2.4 than in 2.2. This effect does not show up with IDE drives.

Oracle would also like support for large page sizes. Richard Henderson said that this would also benefit scientific applications. This topic came up again on Saturday during van Riel's presentation about MM. In general, Oracle wants things standardized across as many OS platforms as possible, in the same way, for example, that shared memory is.

This session went 17 minutes over schedule and ended with an exchange between Linus, Larsh, and Alan Cox. Linus suggested fast semaphores for scheduling rather than spin locks, and Cox suggested that the user space spin locks work like Mozilla. "What Mozilla is doing is counting how many times you spin on locks before giving up, assuming that some other process has been locked but is not scheduled."

Ted Ts'o, who moderated the event, called a break at that point. Breaks were always 30 minutes, giving ample time for discussion.

SCTP

La Monte H.P. Yarroll, Motorola

La Monte Yarroll described a new protocol that will be peer to UDP and TCP (layer four for OSI fans). SCTP stands for Stream Control Transmission Protocol (RFC2960) and has several design goals:

- Sequenced delivery of user messages within multiple streams
- Network-level fault tolerance through support of multi-homing at either or both ends of an association
- MTU set at layer four to prevent fragmentation at the IP layer
- Optional bundling of multiple messages within the same packet

SCTP (<*http://www.cis.ohio-state.edu/cgibin/rfc/rfc2960.html*>) is very long (134 pages), but Yarroll stated that there are already 24 implementations that inter-operate. Essentially, SCTP combines the reliability of TCP with the ability to send messages, even multiple message streams, over the same connection. It has some built-in fail-over because it supports the concept of multi-homing: that is, a single server can listen at multiple interfaces, and if one path quits responding, it can resume the same connection using a different interface and presumably another path.

Someone asked if SCTP can do load balancing, and Yarroll responded that this is an open research issue with no known solution. "First, slag one network, move over to another one, slag that one, move back, and so on," quipped Yarroll. Someone else asked whether SCTP was any better than TCP in connection setup and breakdown. Yarroll reported that SCTP is somewhat better, as only the third packet used in setup can carry data, and that multiple streams can use the same connection. Also, there are no bitwise flags, and all options are word-aligned.

Someone else asked if there is any talk of moving part of the protocol into hardware. Yarroll answered, "It is a dream. There are a lot of properties that should make SCTP hardware implementable." Ted Ts'o pointed out that fiber channels are very expensive, and SCSI over SCTP would be a viable option.

During the break, Stephen Tweedie, the next presenter, moved toward the front and Linus intercepted him at the table where I was sitting. Soon, Ben LaHaise joined in a spirited discussion about zero copy writes. Zero copy writes avoid the performance hit of a memory to memory copy, and Linus shared his skepticism about how it is being implemented. My impression was of a professor with not a lot of seniority arguing with his grad students and other professors. At one point, Linus said something that I thought was very revealing: "We don't want to wind up like Windows NT with lots of subtle bugs. We want stability over performance."

Block Device Layer

Stephen Tweedie, RedHat

Ts'o introduced this session by joking that it was completely uncontroversial and not relevant to the kernel. This was a fitting beginning.

Tweedie began by discussing scalability issues. These include large numbers of devices, large devices (current 2TB limit), 512-byte block size being a problem for large disks, and related SCSI issues, like large numbers of logical units. He jumped next to robustness. Currently, information from the SCSI layer does not pass to higher layers, so a one-bit error could result in a RAID disk being taken offline.

Broaching the issue of under-performance, Tweedie stated that the kernel cannot pass in single I/Os that are contiguous on disk but discontinuous in memory. “Each I/O merge is a scan of up to 8,192 requests!” Tweedie said. Scheduling could be made more fair by scheduling per spindle rather than for host bus adapter.

Device naming has been and will continue to be an issue. There are worldwide namings in fiber channel, SCSI naming by probe order, and the same with IDE devices. Ts’o mentioned that in some cases you will not be able to enumerate all devices at boot time, and sixteen bits for dev_t (that holds the minor device number) will not be enough.

Jens Axbone mentioned that he had done some tests with Andre Hedrick where they moved the io_request_locks within the device layer and got better performance. In the future, elevator scans should only occur once, when either inserting or scanning, not twice as happens now. Because of writeback caching on ATA, you have to do a write-back flush even if you disable the cache on the drive. Don Duggans quipped, “The placebo bit.”

Tweedie agreed that each driver should maintain its own queue. Someone said that we build devices that look like 36 logical devices but are really hundreds of spindles. We would prefer that you can disable part of the elevator. Tweedie responded, “That can be done, but you will still want us to merge requests, just not order them.”

ADVANCED FILE SYSTEM INTEGRATION

Stephen Lord, SGI

Lord discussed some advanced features of XFS, suggesting that some of them could be moved into the kernel as part of the VFS interface. The key ideas surrounded the notion of delayed writes. Instead of scheduling a write of data immediately to disk, in delayed writes only the space for the data is reserved, and the actual write is done later. As most programs will continue to write data, delayed writes cause disk writes to become batched, and temporary files may never need to be written to disk at all.

After some comments in the Q&A about the need for Linux to scale larger, Dave Miller asked if it makes sense for anyone to use 128 CPU systems. Lord answered that SGI’s customers were buying them. Miller suggested that the latency issues could be reduced by using clusters of eight CPU machines. Lord replied that clusters can fail because of variable bandwidth needed by different parts of the application. Others seemed to think that smaller scale clusters might be the future. Miller brought up NUMA (non-uniform memory access) architectures and wondered if it is necessary to scale up to so many CPUs.

The discussion shifted back to talk about delayed I/O and what happens if insufficient space is reserved for file metadata. Lord replied that SGI has been successfully doing this for years.

ILLUMINATING THE NETDRIVER API . . . ONE MORE TIME

Jamal Hadi Salim, Znyx Networks (also Robert Olsson and Alexy Kutsnetsov)

Hadi Salim exposed a serious problem with existing Linux kernels: under extreme network loads, the system collapses. A comparison between a FreeBSD system and a Linux system showed that the BSD kernel could handle up to 70kpps (kilopackets per second) but that Linux folds at only 24kpps. Keeping in mind that a T1 filled to capacity with 64 byte packets is about 25kpps, this is not good. Furthermore, on SMP systems, performance is worse. This behavior also starves other network interfaces that are not overloaded.

When Andy Grover asked why BSD did so much better, someone replied, “They lied!”

Hadi Salim, working with Olsson and Kutsnetsov, had experimented with a solution that appears to provide a tenfold increase in performance. Their goals were to reduce interrupts on overload, drop packets early on overload, remove or reduce unfairness, and balance latency and throughput, without requiring specific hardware solutions (e.g., Tulip chip).

An obvious solution occurred to me (too bad no one asked me about this earlier). Each packet arriving at the network interface generates an interrupt, and it is this interrupt processing that soon overwhelms the kernel. Serial card designers of the early nineties had already figured this out, and the fastest drivers used polling instead of interrupts. Hadi Salim’s solution is similar in that interrupts are disabled when the load increases and only re-enabled when the DMA ring assigned to the device is empty. If more packets arrive when the DMA ring is full, they are dropped without any further processing. Using 2.4.0 beta 10 they could get 200kpps peak using commodity hardware that supports DMA and polling.

Larry McVoy, who seems to have worked for all the UNIX workstation vendors at some point, remarked, “Rob Warnick at SGI did an enormous amount of good work in this area, forced because they [SGI] had to work with slow processors. If you do this stuff right, you just pass the packets right up, but if you get a blast, then you start to do interrupt collapsing.” When Miller remarked that this would clean up the drivers a lot, there was a round of applause.

There was more discussion about the effect of this solution on servers (better performance), latency (less latency), and older drivers. Hadi Salim said that the system reverts to old behavior when the driver does not support polling. The session ended with everyone feeling good about this solution, which is likely to find its way into the 2.5 kernel.

Linux Hot Plug

Johannes Erdfelt, VA Linux; Greg Kroah-Hartman, WireX

Erdfelt started off by saying that they don't have a solution to this problem. His own area of expertise is USB, particularly hubs and hot plug needs to deal with SCSI, Firewire, PCMCIA/Cardbus, and hotswap PCI in addition to USB devices. The problems include name and file permissions: that is, the device name presented to the user should not change and neither should the permissions associated with that device. Since not all of these devices have anything resembling a serial number or UUID, identifying them can be a problem.

Other issues include what to do when there are multiple drivers (e.g., using a parallel port) and how to notify applications (hey, a joystick just appeared!).

Existing solutions include cardmgr for PCMCIA (which works because it is not very complicated [laughter] compared to USB, scsidev, or devfs/devfsd). Using a script, /sbin/hotplug, works but does not help with naming. At this point, Erdfelt asked for suggestions.

Someone immediately asked why devfs (think procfs) doesn't work, and Erdfelt answered that it doesn't solve the naming problem. Ts'o pointed out that in the PCMCIA world, shell scripts could handle things pretty well, but the SCSI devices would be more difficult. Linus said that /sbin/hotplug worked well for him. He thought that vendors could be relied on to provide scripts for all the hot-pluggable devices and that would solve 99% of the problem. "A script is a valid answer for a geek's machine; it can be edited, and RedHat can do it for every possible device."

Peter Anvin said that he is responsible for assigning device numbers. And right now, we are running out of character numbers. He just assigned number 228 out of 225, and suggested that the discussion (or flame war) be done offline in the 10 p.m. BoF. This did not stop the discussion, which continued until the end of the session. Anvin made an interesting point when he mentioned that the Japanese wanted Japanese device names rather than English ones.

Reception

The reception drew most of the attendees, even some people who had said earlier that they couldn't stay. I wound up talking to Peter Loscocco of the NSA about his part in a project to add real security to the Linux kernel and missed the BoFs. Well, I could have attended since they went until midnight, but I packed it in. There were also BoFs after the conference finished. You can read about the BoFs, and get another perspective on the conference, at LWN: <*http://lwn.net/2001/features/KernelSummit/*>.

Kernel kbuild

Keith Owens, OC Software; and Eric Raymond

Nine o'clock Saturday morning started off pretty quietly, no surprise there. The topic concerned a proposed replacement for the kernel configuration tools in the present distribution. I was happy to hear this, but not everyone else was. The present system presents a text-based menu that does not let you go backward, and a GUI-based one that also fails to satisfy me.

Raymond is not too fond of it either. "I've been examining the existing kernel configuration system, and I have about concluded that the best favor we could do everybody involved with it is to take it out behind the barn and shoot it through the head." Owens and Raymond set out to build a much improved system, and they did.

The new system (CML2) makes it impossible to generate an invalid configuration. The new system uses a Python engine to enforce a set of rules to do this. This prompted concern from Jes Sorenson, who is apparently porting Linux to IA64. "Can I make it work without Python?" he asked. Many people pointed out that the work could be done on the cross-compiling system and that Python was not required on the target system.

There was actually a fair amount of resistance to this effort. Raymond pointed out that the new version uses 40% less code, runs faster, and has features the old tool does not. At one point, the discussion veered into arguing about the kernel symbol table. But it ended on a good note, with Raymond saying, "Down the road, I want to make configuration so easy that your Aunt Tilly could do it." This was followed by enthusiastic applause.

Future VM Work

Rik van Riel, Conectiva S.A.

Van Riel had a detailed presentation covering a lot of difficult topics in VM, his own specialty. Occasionally he wound up listening to the discussion that raged about contentious issues, such as page size.

Van Riel began by discussing memory balancing, deciding which pages to steal and when. The current implementation scans the process page tables and steals whatever it can. Better methods would involve keeping track of working sets, providing processes with a defined amount of physical memory. Another approach would use reverse mapping, keeping track of which processes have a mapping from a virtual page to a physical page. Doing this would add eight bytes to the page table entry, doubling its size.

Once again, moving to a larger page size was suggested. Linus pointed out that early versions of his kernel did just that, but that it would make no sense to do this "just for Oracle." Van Riel explained that there would be benefits for other users, such as smaller page tables and a reduction in page faults. Large pages are not suitable for all uses, so the ideal would be to mix large and small pages.

Shared page tables would also help in the case where processes share memory. Again, this is also something that Oracle asked for, but it would involve constraints (like having to be aligned on a 4MB boundary on some architectures). Another layer of locking would also be needed, leading to the possibility of deadlocks.

The Linux kernel still has some problems with deadlocks: for example, it needs to allocate memory to free some pages. Thrashing can also occur; Van Riel suggested suspending processes when the system begins to thrash, in order to stabilize the load, and gradually releasing each process as the system recovers.

Mandatory Access Controls / SE Linux

Peter Loscocco, NSA

I thought that this proposal was well received, although Loscocco told me later that he was disappointed that he didn't get more support. You can get more details, or the code if you like, at <*http://www.nsa.gov/selinux/*>. Loscocco also told me that in the eighties, he was responsible for the ARPANET IMP that passed through the NSA site, and that stories about how the NSA was copying all the data on the ARPANET were really funny. Eventually, the NSF moved the IMP somewhere else because of difficulties in keeping it up and running. But that's another story.

SE Linux uses a large kernel patch to add Mandatory Access Controls (MAC) to a standard Linux kernel. MAC means that permissions are no longer discretionary, and that only a security administrator can change them. The changes to the kernel are pervasive and fine-grained, meaning that it is possible to tie down everything.

One important point is that SE Linux is not an Orange Book system. The system has not been evaluated, although it has evolved from earlier work such as DTOS within MACH, and Flask with Fluke (see a paper presented at a USENIX Security Symposium). SE Linux uses Type Enforcement, which goes way beyond Orange Book designs. Type Enforcement includes the usual subject (user) and object (resource) found in earlier models but adds the program to the equation. With Type Enforcement, you can specify that a user can write to `/etc/shadow` but only when running the `passwd` program.

The design goals of SE Linux include development of:

- Separation policies to keep data separate from some other part of the system
- Containment policies to restrict Web server access to authorized data
- Integrity policies to protect applications from modification
- Invocation policies to control the chain of processing

The current implementation focuses on enforcement, not policy. Loscocco said that the sample implementation does come with some policy models that could be used to nail down a Web server, for example.

This session ended with Linus saying that he liked the code he had seen, but he was aware that there were many security projects in progress right now, and that he did not want to have to choose between them. If a function call could replace each section of code the NSA design had added, then it might be acceptable. People who did not want to use the security could replace this with a function that simply returns.

Asynchronous I/O for Linux

Ben LaHaise, RedHat Canada Ltd.

Asynchronous I/O would allow programs to write data to some device and then continue without waiting for the operation to complete. LaHaise pointed out that this would be useful in event-based applications when you want to avoid using thread-based I/O (8KB overhead per thread per each request, and there could be tens or thousands of requests). It also makes efficient use of raw I/O, fits well with zero copy, and has lower overhead for daemons than select/poll system calls.

Asynchronous I/O, as proposed by LaHaise, would add four new system calls:

`_submit_ios()` allows a process to fire off an asynchronous operation.
`_io_cancel()` is for canceling outstanding operations.
`_io_getevents()` gets information on completed operations.
`_io_wait()` allows a process to wait for the completion of a specific operation, essentially turning it synchronous.

Linus, who obviously was already familiar with the proposal, interrupted LaHaise at this point: "You have already added four system calls, why have a new device?" The device in question would be named `/dev/aio` and be used to arrange for a section of mapped memory that the calling process could use to detect I/O completion. Linus was not at all happy with this use of mapped memory and continued to ask if LaHaise really thought that `mmap` was necessary.

Toward the end of his defense of `mmap`, Ulrich Drepper, defender of the system call API, asked, "Why do you want to add all this support to character devices? It is completely wrong to do this." LaHaise pointed out that POSIX AIO doesn't do a very good job of the semantics, and that his approach will really make a difference with sockets, particularly UDP. Drepper asked if you can have AIO for `fsync()`, and LaHaise responded that he had code that could currently do that.

This project is not finished yet, and is currently about a ~3000 line patch. Work to be done included adding network sockets, limiting memory pinning, and writing some documentation.

Linux and Power Management, ACPI

Andy Grover, Intel

Grover started out by explaining the reasons why PC power management is important. He suggested some obvious things, like green PCs and mobile PCs. When he mentioned that servers might also be able to do this to reduce power when sitting idly in large clusters, Linus reacted. It was obvious this is something he wanted.

In the older scheme, currently supported by the Linux kernel, APM handles power management. The trouble with that is that APM is an obsolete interface, and that the BIOS keeps track of APM so that it is not managed by the OS. The replacement for APM is the Advanced Configuration and Power Interface (ACPI), conceived by Intel, Microsoft, and Toshiba, which allows OS-directed power management.

Grover went on to describe five soft power levels and four hard (device) levels, ranging from fully on to power off. Grover felt that implementing soft power level three would be the best target for 2.5. S3 permits idle devices to be disabled and has better power saving than levels one or two and quicker wakeup than level four (sleep). Enabling level three is a prerequisite for doing level four.

Grover explained that implementing level three means that it must be possible to shut off devices, and then to turn them all on again while restoring sufficient state for them to return to working condition. This implies having a device manager that knows enough about all the devices to shut them down or reinitialize them.

A lively discussion began at this point, with different people wondering how to make this work. Linus pointed out that the PCI device has extra, unused fields in its structure that would allow it to be extended to handle any device, and build an internal device tree. He had actually considered adding the changes in earlier, but the patches are very large, and adding large patches disturbs people. Ts'o pointed out that 2.5 might be the ideal time to do this.

The following Web sites have more info:
<http://developer.intel.com/technology/iapc/acpi>;
<http://phobos.fachschaften.tu-muenchen.de/acpi> (for ACPI 4 Linux)

BitKeeper

Larry McVoy, BitMover Inc.

Before he got started, McVoy suggested that we thank Ted Ts'o for making the developers summit possible. Then, he had a short rant about scaling up the Linux kernel, suggesting that SMP clusters, with one kernel monitoring every four kernels, was the way to go.

McVoy was really there to talk about BitKeeper, a source code control system. Subversion, another CVS system, had been the topic of a Friday night BoF, which McVoy could not attend. According to McVoy, as well as Victor Yodaiken, BitKeeper had the only GUI interface that would really work for kernel hackers. He went on to describe various features, as well as demonstrating them for the kernel hackers.

"BitKeeper is a peer-to-peer system. You get revision control files; we merge revision histories. We can sync sideways, rather than go up and then down, which is what we did when I worked at Sun (and I wrote their systems). You have to keep track of revisions, but we compress them."

Listening to McVoy, and the reaction to his demonstration, made it appear that BitKeeper really would work well in the world of Linux. There are some problems: for example, getting more checked out than you want if several modules have been modified since the last time you peered. Raymond complained when he found that the only editor bindings

Summit group photo. See the Linux Weekly News *for a color version with ids of the participants <http://lwn.net/2001/features/KernelSummit/>* [photo: J. Corbet/LWN.net] were for vi/vim (Raymond is an emacs fan).

In the end, McVoy said that Alan Cox says we are about 75% to getting wto the way that Alan works now. BitKeeper is a product but is free for noncommercial use. I enjoyed listening to McVoy, and it was obvious that lots of attendees were willing to listen to an experienced "old-timer" (McVoy is about 40 years old).

BUGTRACKING OPEN SESSION

Ted Ts'o took the helm for this final official session. The focus was on bugtracking, which appears (to me) to be very lame at this point. The current mechanism is that someone posts a bug to the kernel developers mailing list, and then someone (usually Alan Cox) notices the bug, saves it, and perhaps dispatches it to the person who manages that portion of code. Ts'o said he tried this for a while and found it very difficult to do.

Ts'o also made "a modest proposal." Rather than have long periods between releases, which leads to a last-minute rush of code submissions, he proposed that a date be announced for a feature freeze. While there was serious discussion of this, no date for 2.5 was announced.

Suggestions for improving bugtracking included using a database or trying Bugzilla. Another suggestion was to use RedHat's sendbug script, which collects information about the system and includes it in the bug report, something that turns out to be really important when trying to support many CPU architectures, disk subsystems, etc.

When Ts'o ended the discussion it was 6:35 p.m. on Saturday night. The room broke into six large groups who just kept on talking.

Perhaps the workshop should have been three days long.