Linux Kernel 2.6: New Features - I

Jerry Cooperstein

Axian Inc

coop@axian.com

@ OGI 10/01/02

Linux Kernel 2.6
(or 3.0?)

- Feature Freeze: Halloween 2002
- Better Performance, Especially on **SMP**
- Better Scalability
- Better I/O Subsystem, New Filesystems
- Many New Hardware Drivers
- New Platforms
- Many Features Tested as 2.4 Patches

Latest Status

- Guillaume Boissiere maintains a status report, updated weekly at:

http://kernelnewbies.org/status/latest.html

Linux Kernel History:
Release Dates, Lines of Code

- 0.01: 09/1991 7.5 K
- 1.0: 03/1994 158 K
- 1.2: 03/1995 277 K
- 2.0: 07/1996 649 K
- 2.2: 01/1999 1536 K
- 2.4: 01/2001 2888 K
- 2.6: ??/2003 ~4200 K

New Features: General

- Preemptable Kernel
- O(1) Scheduler
- New Kernel Device Structure (**`kdev_t`**)
- Improved Posix Threading Support (NGPT and NPTL)
- New Driver Model & Unified Device Structure

New Features: General

- Faster Internal Clock Frequency
- Paring Down the BKL (Big Kernel Lock)
- Better in Place Kernel Debugging
- Smarter IRQ Balancing
- ACPI Improvements
- Software Suspend to Disk and RAM

New Features: General

- Support for USB 2.0
- ALSA (Advanced Linux Sound Architecture)
- LSM (Linux Security Module)
- Hardware Sensors Driver (lm-sensors)

New Features: Architectures

- AMD 64-bit (x86-64)
- PowerPC 64-bit (ppc64)
- User Mode Linux (UML)

New Features: Journalling Filesytems

- Ext3, ReiserFS (already in 2.4)
- JFS (IBM)
- XFS (SGI)

New Features: I/O Layer

- Rewrite of Block I/O Layer (BIO)
- Rewrite of Buffer Layer
- Asynchronous I/O
- IDE Layer Update
- ACL Support (Access Control List)
- New NTFS Driver

New Features: Networking

- NFS v4
- Zero-Copy NFS
- TCP Segmentation Offload
- SCTP Support
  (Stream Control Transmission Protocol)
- Bluetooth Support (not experimental)
- NAPI (Network Interrupt Mitigation)

Preemptable Kernel

- Robert Love **www.kernel.org/pub/linux/kernel/people/rml** (original patch from MontaVista)
- Old: No Kernel Preemption
  - Execution in kernel mode interrupted only by explicit yields, sleeps, and IRQ's
- New: Kernel May Be Preempted
  - New process may be swapped in after servicing an interrupt

Preemptable Kernel

- **OLD:**
  - Kernel executing code for process A
  - Services interrupt
  - Returns to Process A
- **NEW:**
  - Kernel executing code for Process A
  - Services interrupt
  - Returns to process A, B, C, …..

Preemptable Kernel

- Modify spinlocks for Preemption:

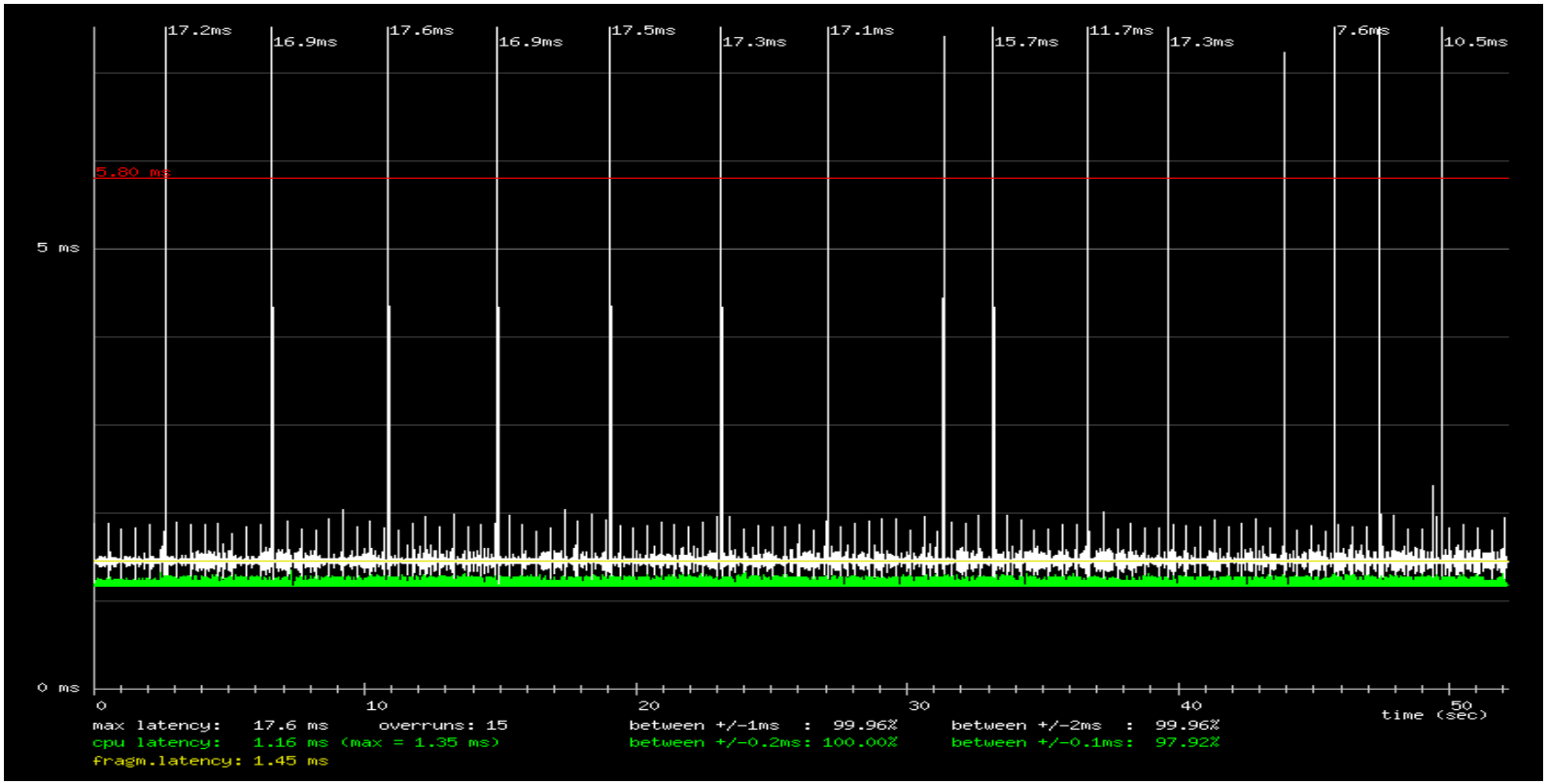
```
spin_lock(lock)          ->
     preempt_disable()
     _raw_spin_lock(lock)
spin_unlock(lock)        ->
     _raw_spin_unlock(lock)
     preempt_enable()
```

- **`preempt_disable(), preempt_enable()`**
  - increment or decrement a counter, and introduce a memory **`barrier()`**call

Preemptable Kernel

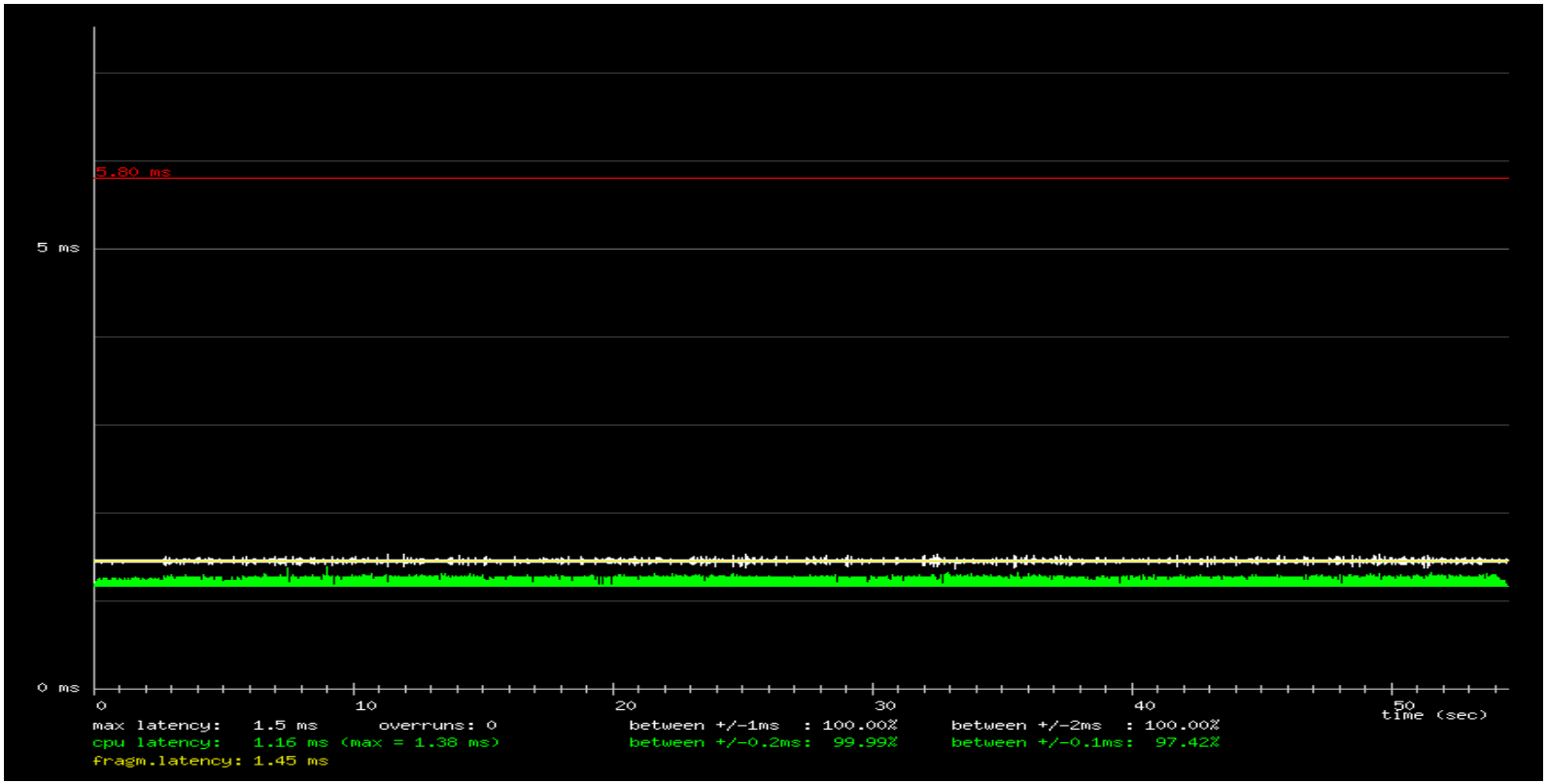
- Kernel must be fully re-entrant
- UP systems will show SMP problems (good for debugging)
- System latency greatly reduced

Latencies with Preemption Off



Linux Magazine, May 01, 2002

Latencies with Preemption On



Linux Magazine, May 01, 2002

O(1) Scheduler

- Ingo Molnar
  (www.kernel.org/pub/linux/kernel/people/mingo)
- Old: Scan all processes, CPU's: O(N)
- New: Maintain 2 queues per CPU - active and expired processes
- Sort by priority - no search: O(1)
- Real time processes share one queue

O(1) Scheduler

- Already included in Red Hat 7.3
- Much better scalability with SMP
- Some (Old) Results:

```
  01/21/2002

Here are some results from running VolanoMark on different
versions of O(1)-scheduler based on 2.4.17.
Volcanomark is a Java(TM) chatroom benchmark: multiple rooms, where for
   each room, every input from a client generates a write to every
   other client (think broadcast storm).
Partha Narayanan, partha@us.ibm.com
```

O(1) Scheduler

VolanoMark 2.1.2 Loopback test,
8-way 700MHZ Pentium III,
1GB Kernel,
IBM JVM 1.3. (build cx 130 -20010626)
Throughput in msg/sec

KERNEL	UP	4-way	8-way
2.4.17	11005	15894	11595
2.4.17 + D2 patch	10606	23300	29726
2.4.17 + G1 patch	10415	23038	31098
2.4.17 + H6 patch	10914	22270	32300
2.4.17 + H7 patch	11018	23427	31674
2.4.17 + J2 patch	13015	23071	33259

Partha Narayanan, partha@us.ibm.com

New Kernel Device Structure (**`kdev_t`**)

- 16-bit **`dev_t`** seen by **`knod(),stat()`:**
  - 8-bit major No. (driver)
  - 8-bit minor No. (instance, mode, device)
- **`kdev_t`** is new internal kernel structure
- For now just:

```
struct{ushort major, minor}
```

New Kernel Device Structure (**kdev_t**)

- Eventually will have more information:
  - device, block and sector sizes
  - name, flags, methods jump table, etc.
- Eventually:
  - 20-bit major numbers
  - 12-bit minor numbers

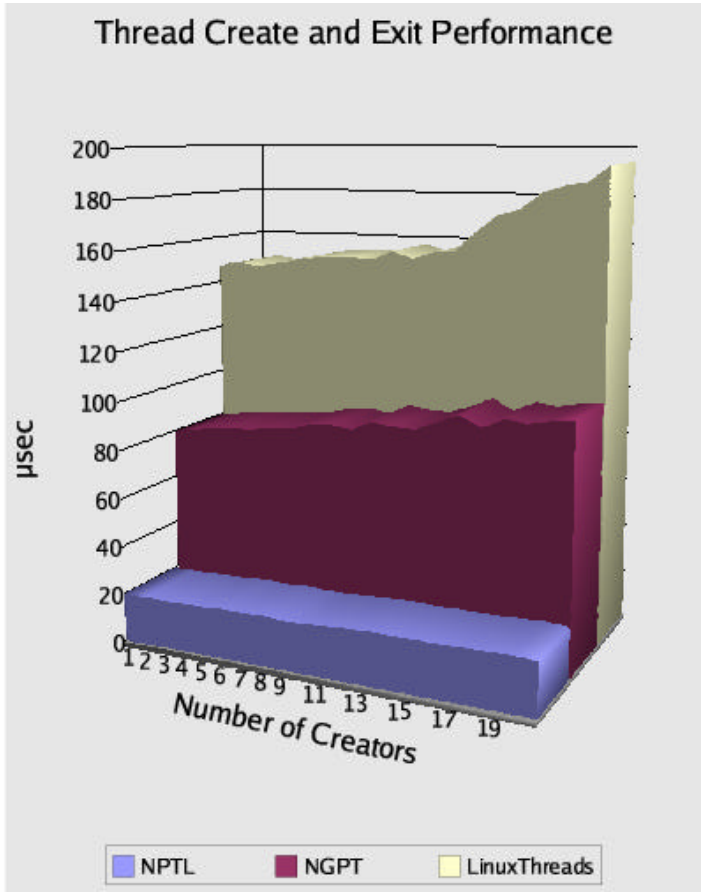
Improved Posix Threading Support

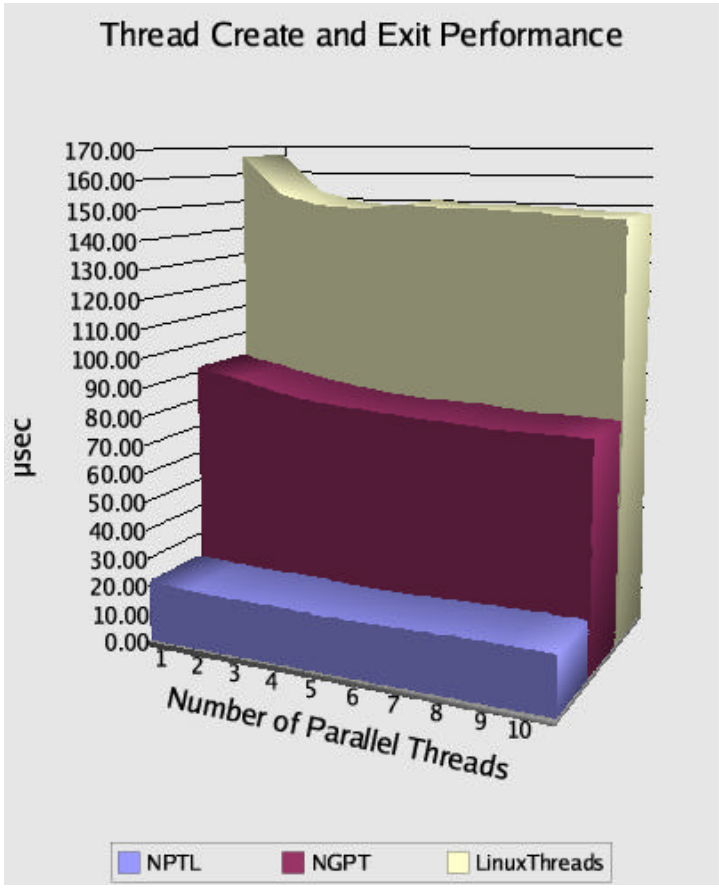
- Next Generation Posix Threads (NGPT)
- Drop in Replacement for LinuxThreads
- Better POSIX Compliance
- Better performance on SMP
- Mostly user space; needed kernel mods
- **http://www.124.ibm.com/developerworks /oss/pthreads**

Improved Posix Threading Support

- Native Posix Thread Library (NPTL)
- Ingo Molnar and Ulrich Drepper
- 1:1 pure kernel thread model (NGTP is M:N, M user threads per N kernel threads)
- Requires 2.536+K + gcc 3.2 + glibc 2.3
- **http://people.redhat.com/drepper/nptl-design.pdf**

Improved Posix Threading





- http://people.redhat.com/drepper/perf-s-100000-pro{par}.pdf

New Driver Model & Unified Device Structure

- Treat all devices in a unified fashion
- **driverfs** virtual filesystem can be mounted and seen as a tree by bus
- New API:

```
void device_init_dev(struct device *)
struct device *device_alloc();
int device_register(struct device *);
void get_device    (struct device *);
```

New Driver Model & Unified Device Structure

```
void put_device   (struct device *);
int valid_device  (struct device *);
void lock_device  (struct device *);
void unlock_device(struct device *);
```

- Initializing, Exiting:

```
int device_driver_init();
void device_driver_exit();
```

New Driver Model & Unified Device Structure

- New data structures:

```
struct device{
    struct list_head bus_list;
    struct iobus     *parent;
    char             name[DEVICE_NAME_SIZE];
    char             bus_id[BUS_ID_SIZE];
    struct           driver_dir_entry *dir;
    spinlock_t       lock
```

New Driver Model & Unified Device Structure

```
    atomic_t        refcount;
    struct device_driver *driver;
    void            *driver_data;
    void            *platform_data;
    u32             current_state;
    unsigned char *saved_state;
}
```

New Driver Model & Unified Device Structure

- New data structure:

```
struct device_driver{
    int (*probe)  (struct device *dev);
    int (*remove) (struct device *dev);
    int (*suspend)(struct device *dev,
                   u32 state, u32 level);
    int (*resume) (struct device *dev,
                   u32 level);
}
```

Faster Internal Clock Frequency

- Raise HZ to 1000
  - More frequent switching
  - Better interactive response

```
/* Internal kernel timer frequency */
 # define HZ         1000
/*.. some user interfaces are in ticks"*/
# define USER_HZ    100
/* like times() */
# define CLOCKS_PER_SEC  (USER_HZ)
```

Paring Down the BKL (Big Kernel Lock)

- Finer-grain control over locking
- Better scalability, SMP
- Tedious task
- Prone to intermittent errors
- Will take a long time to complete
- BKL will remain in limited role

BKL Removal (Example)

- **OLD:**
```
lock_kernel();
… critical code;
unlock_kernel();
```
- **NEW:**
```
spin_lock(&lock_A);
… critical code
spin_unlock(&lock_A);
spin_lock(&lock_B);
… critical code;
spin_unlock(&lock_B);
```

Better in Place Kernel Debugging

`CONFIG_PREEMPT=y` **`CONFIG_DEBUG_SLAB=y`**

`CONFIG_DEBUG=y` **`CONFIG_DEBUG_SPINLOCK=y`**

`CONFIG_KALLSYMS=y`

- Check if sleeping in atomic code
- No need to run **`ksymoops`**
- Poison memory on freeing
- Also possibly: LTT (Linux Trace Toolkit) and DProbes (Dynamic Probes)

ACPI Improvements

- Based on Intel ACPI CA (Component Architecture)
- Derived from Intel “generic” Unix implementation
- Still weak on end user support
- Increases kernel size by ~70 KB
- ACPI4Linux project: http://acpi.sourceforge.net

Software Suspend

- Save machine state in swap partition by hitting **`Sysrq-d,`** or a shutdown option
- Restore on reboot
- Entirely in software; no APM needed
- Be careful using this, its experimental!
- **http://falcon.sch.bme.hu/~seasons/linux/swusp.html**

Smarter IRQ Balancing

- Better control of what CPU's interrupts are handled on.
- Can set the CPU affinity for IRQ's by writing mask to:

  `/proc/irq/IRQ#/smp_affinity`

- Better Load Balancing

USB 2.0 (Universal Serial Bus)

- Replacing USB 1.1
- High speed: 480 Mbit/sec (up from 12)
- **`usbdevfs`** -> **`usbfs`** (special filesystem)
- Split transactions (start and complete phases)
- **http://www.linux-usb.org/usb2.html**

ALSA (Advanced Linux Sound Architecture)

- Audio, MIDI functionality, replaces OSS (Open Sound System)
- SMP and thread-safe
- Many cards, Professional to Consumer
- User space library for applications
- **http://www.alsa-project.org**

LSM (Linux Security Modules)

- General kernel framework for supporting access control modules
- Security fields added to kernel structs
- Hooks in kernel to do access control
- **`security_ops`** jump table (fine control)
- **http://lsm.immunix.org**

Hardware Sensor Drivers (lm_sensors)

- Monitor system environmental info:
  - power status
  - voltages
  - temperature, etc.
- Works with I2C, SMB busses
- Many different chip sets supported
- **http://www.lm-sensors.nu**

AMD 64-bit (x86-64)

- “Hammer”
- Supports IA32 binaries
- 4 KB pages with 4 level Page Tables
- Linux has 3 levels: leads to 40 bits per user process (1 TB)
- **http://www.x86-64.org**

PowerPC 64-bit (ppc64)

- Used in IBM iSeries and pSeries
- Full 64-bit and 32-bit addressing.
- **http://www.penguinppc64.org**

User Mode Linux (UML)

- Virtual machine, not real hardware
- Can be used for:
  - Kernel development and debugging
  - User space debugging
  - Trying new distributions, kernels, filesystems
  - Commercial hosting (ASP)
- **http://user-mode-linux.sourceforge.net**

Upcoming Linux Seminars

- Linux Kernel 2.6 - New Features II
  - **October 15 @ OGI**
- Linux Network Programming
  - **January 7 @ OGI**

Upcoming Linux Programming Classes at Axian

- RHD236 (Linux Kernel Internals)
  - **November 4 - 8**
- Linux Kernel Network Programming ***(New)***
  - **December 2 - 6**

Upcoming Linux Programming Classes at OGI

- Linux Kernel Internals
  - **December 16 - 20**
- Linux Device Drivers
  - **November 11 - 15**
- Linux Kernel Network Programming ***(New)***
  - **February 10 - 14**

Upcoming PLUG Meeting

- Thursday, Oct. 3, 7PM at PSU, Smith Memorial Center Room 294/296
- Presentation:

  Overview of LVM

  (Logical Volume Manager)

  Cooper Stevenson
- **`http://pdxLinux.org`**