Why userspace (still) sucks.

Trainwrecks, Gore and other programming tragedies.

Dave Jones <davej@redhat.com>
Red Hat

Motivation

Reducing boot time, increase idle time.

Reducing boot time.

Doing readahead whilst idle.

Profiling

kernel patch to log every stat()/open()

Reducing boot time.

Enhanced kernel patch to monitor all I/O (deletes too etc).

Later used systemtap

Reducing noise.

Disable readahead.

Watching the horror show.

155234 line log.

Boot up.

79576 files were stat()’d

26769 were open()’d

1382 commands were exec’d.

Shutdown

23246 files were stat()’d

8724 files were open()’d.

Magnanimous disclaimer.

Let the gorefest begin.

WARNING:

Those of a sensitive disposition may want to look away.

HAL. "If it’s a file, I’ll open it".

HAL. "If it’s a file, I’ll open it".

Re-read and reparsed *dozens* of XML files during startup.

- In some cases, it did this 54 times per XML file.
- Performed no caching.
- Read a bunch of files describing devices that were not even present.

Accounted for a total of 1918 open()’s, and 7106 stat()’s

HAL. "I’m sorry Dave".

Polls the cd drive every 2 seconds even when no user is logged in.

Turns out to be a really bad idea on big boxen.

CUPS

Read in ppd files describing every printer known to man...

CUPS

Read in ppd files describing every printer known to man...
... Even though there was not even a printer connected.

Responsible for around 2500 stat()’s, and around 500 open()’s

About that no printer..

About that no printer..

"The Hewlett-Packard Linux Imaging and Printing Project provides drivers for HP printers and multi-function peripherals."

hplip wakes up every half second for a select() timeout..

.. socket events will wake it up anyway.

Xorg

How not to probe for PCI devices.

How not to probe for PCI devices.

- Scans through /proc/bus/pci/ in order
- Seemed to *guess* at random bus numbers
- Tried to open those devices in /proc/bus/pci/
- Sequentially probed for devices on busses 0xf6 through 0xfb
  - (even though they didn’t exist)
- Retried entries that it had already attempted to scan
  - regardless of whether they succeeded or not.

Xorg (cont-d).

X really likes to stat & reopen lots of files it had already opened.

like libGLcore.so.

irqbalance.

Wakes up every 10 seconds to re-balance interrupts.

irqbalance.

Wakes up every 10 seconds to re-balance interrupts.

re-balanced interrupts where no IRQs had ever occurred.

irqbalance.

Wakes up every 10 seconds to re-balance interrupts.

re-balanced interrupts where no IRQs had ever occurred.

Also re-balanced interrupts where an IRQ had not occurred in some time.

A two line change saved a few dozen syscalls.

irqbalance.

Wakes up every 10 seconds to re-balance interrupts.

re-balanced interrupts where no IRQs had ever occurred.

Also re-balanced interrupts where an IRQ had not occurred in some time.

A two line change saved a few dozen syscalls.

Reopens fd’s.

A brief history of file notification.

A brief history of file notification.

Apps spin in a loop polling files.

A brief history of file notification.

Apps spin in a loop polling files.

SGI ’fam’ daemon for IRIX

A brief history of file notification.

Apps spin in a loop polling files.

SGI ’fam’ daemon for IRIX

Linux dnotify

A brief history of file notification.

Apps spin in a loop polling files.

SGI ’fam’ daemon for IRIX

Linux dnotify

Linux port of fam

A brief history of file notification.

Apps spin in a loop polling files.

SGI ’fam’ daemon for IRIX

Linux dnotify

Linux port of fam

inotify

A brief history of file notification.

Apps spin in a loop polling files.

SGI ’fam’ daemon for IRIX

Linux dnotify

Linux port of fam

inotify

gamin

A brief history of file notification.

<insert bong hit here>

gamin

It’s like inotify never happened..

gamin

It’s like inotify never happened..

This causes an incredibly high number of context switches/sec

nautilus

Spends lots of time stat’ing

- $HOME/Templates,
- /usr/share/applications,
- $HOME/.local/share/applications

even though they had not changed.

INOTIFY!

nautilus

Spends lots of time stat’ing

- $HOME/Templates,
- /usr/share/applications,
- $HOME/.local/share/applications

even though they had not changed.

INOTIFY!

- Well, almost...

Madness with fonts

apps stat()/open() fonts, and never actually use them.

Make the problem worse to highlight it..

Madness with fonts

apps stat()/open() fonts, and never actually use them.

Make the problem worse to highlight it..
.. by moving 6000 TTF's to $HOME/.fonts

Madness with fonts

apps stat()/open() fonts, and never actually use them.

Make the problem worse to highlight it..
.. by moving 6000 TTF’s to $HOME/.fonts

The log file almost doubled in size.

font madness.

- gnome-session stat()’d 2473 and open()’d 2434 ttf’s.

font madness.

- gnome-session stat()’d 2473 and open()’d 2434 ttf’s.
- metacity open()’d another 238.

font madness.

- gnome-session stat()’d 2473 and open()’d 2434 ttf’s.
- metacity open()’d another 238.
- Just to be on the safe side, wnck-applet open()’s another 349 too.

font madness.

- gnome-session stat()’d 2473 and open()’d 2434 ttf’s.
- metacity open()’d another 238.
- Just to be on the safe side, wnck-applet open()’s another 349 too.
- Nautilus decides it does not want to be left out of the fun, and open()s up another 301.

font madness.

- gnome-session stat()’d 2473 and open()’d 2434 ttf’s.
- metacity open()’d another 238.
- Just to be on the safe side, wnck-applet open()’s another 349 too.
- Nautilus decides it does not want to be left out of the fun, and open()s up another 301.
- mixer_applet rounds things off by open()ing up 860 ttfs.

font madness.

- gnome-session stat()’d 2473 and open()’d 2434 ttf’s.
- metacity open()’d another 238.
- Just to be on the safe side, wnck-applet open()’s another 349 too.
- Nautilus decides it does not want to be left out of the fun, and open()s up another 301.
- mixer_applet rounds things off by open()ing up 860 ttfs.
- X font server.
  - Was rebuilding the font cache every time it booted, even if no changes had occurred in the fonts directories.

Bugs from the twilight zone..

mixer_applet2 did a stat on libgstffmpegcolorspace.so

Bugs from the twilight zone..

mixer_applet2 did a stat on libgstffmpegcolorspace.so

Also wakes up every 100 ms to check if the volume changed.
gstreamer has a ’volume-changed’ signal for this.

Fun with timers.

Fun with timers.

```
peer_check_expire        181 crond
dst_run_gc               194 syslogd
rt_check_expire          251 auditd
process_timeout          334 hald
it_real_fn               410 automount
process_timeout          437 kjournald
process_timeout         1260
it_real_fn              1564 rpc.idmapd
commit_timeout          1574
wb_timer_fn             1615 init
process_timeout         1652 sendmail
process_timeout         1653
process_timeout         1833
neigh_periodic_timer    1931
process_timeout         2218 hald-addon-stor
process_timeout         3492 cpuspeed
delayed_work_timer_fn   4447
process_timeout         7620 watchdog/0
it_real_fn              7965 Xorg
process_timeout        13269 gdmgreeter
process_timeout        15607 python
cursor_timer_handler   34096
i8042_timer_func       35437
rh_timer_func          52912
```

(Secret: The kernel sucks too)

USB: Every 256ms, a timer was firing in the USB code.

(Secret: The kernel sucks too)

keyboard controller.

i8042 code polls the keyboard controller at HZ/20 to see if someone has hot-plugged a PS2 keyboard/mouse or not.

(Secret: The kernel sucks too)

Cursor blinking.

- Hilariously, at HZ/5 we wake up to blink the cursor.
- Even if we are running X, and not sat at a VT

- And it used to be worse..

gnome-terminal whilst idle

- Even when cursor blinking is turned off.

Xorg

- X uses interval timers to wake up a *lot*

- Even if it was not the currently active VT.

- Ironically, this is due to X using its ’smart scheduler’
- -dumbsched ’fixes’ it.

- Historically, gettimeofday was slow.
- And interval timers were fast.. I guess?

Sendmail

Wakes up every 5 seconds, even when it’s totally unused/idle.
(It wakes up quicker if a signal comes in about actual mail).

Network Manager

nm-applet wakes up every second to redraw its icon.

Even if the daemon isn’t running.
Even if the state hasn’t changed.

gnome power manager.

"gnome powermanager makes the system eat more power"

gnome power manager.

"gnome powermanager makes the system eat more power"

- Asks the X server twice per second what the current video mode is to find out if it changed.
- Totally unnecessary: X sends apps an event when the mode changes

Battery applet.

battery applet polls every second

Even if running on AC

gnome-screensaver

Polls X every second instead of using the X screensaver API..
.. to find out the mouse position, to detect user activity.

Totally unnecessary, X has an entire API for this!

gnome-screensaver

Polls X every second instead of using the X screensaver API..
.. to find out the mouse position, to detect user activity.

Totally unnecessary, X has an entire API for this!

- Has some weird smart card event monitoring voodoo.
- Polls several times a second.

gpm

gpm wakes up every 2 seconds to poll the console, even in runlevel 5

.. to wait for X to go away.

pcscd - smart card daemon

a daemon for dealing with smart card readers.

Scans all USB devices once per second to find smart card readers.

```
Pid 2224 pcscd run 301599+ ms...
  reads /dev/bus/usb/001/001 301 times, once every 1000 ms avg 43 bytes
  reads /dev/bus/usb/002/001 301 times, once every 1000 ms avg 43 bytes
  reads /dev/bus/usb/003/001 301 times, once every 1000 ms avg 43 bytes
  reads /dev/bus/usb/004/001 301 times, once every 1000 ms avg 43 bytes
  reads /dev/bus/usb/004/002 301 times, once every 1000 ms avg 57 bytes
  reads /dev/bus/usb/005/001 301 times, once every 1000 ms avg 43 bytes
  reads /dev/bus/usb/005/002 301 times, once every 1000 ms avg 43 bytes
  reads /dev/bus/usb/005/005 301 times, once every 1000 ms avg 52 bytes
  reads /dev/bus/usb/005/006 301 times, once every 1000 ms avg 52 bytes
  reads /dev/bus/usb/005/008 301 times, once every 1000 ms avg 50 bytes
```

pcscd - smart card daemon

The pcsc daemon wakes up once per second for no reason at all.

has a 1 second select loop time out, but all it does on timeout is loop again.

the pcsc-lite library has a 200msec polling loop that gets called from gdm-binary.

xenstored - Xen metadata database

xenstored is ’special’.

Its implementation of transactions requires it to take a complete copy of the database backend file everytime it starts & ends a transaction.

xenstored - Xen metadata database

xenstored is ’special’.

Its implementation of transactions requires it to take a complete copy of the database backend file everytime it starts & ends a transaction.

XenD performs about 16 transactions every time you ask for info about a domain

xenstored - Xen metadata database

xenstored is ’special’.

Its implementation of transactions requires it to take a complete copy of the database backend file everytime it starts & ends a transaction.

XenD performs about 16 transactions every time you ask for info about a domain

if you monitor 20 guest domains once a second, this translates into xenstored doing 80 MB of disk I/O every second.

nautilus-cd-burner wakes up every 2.5 secs

```
/usr/libexec/mapping-daemon
Process 2977 attached - interrupt to quit
poll([{fd=4, events=POLLIN}, {fd=3, events=POLLIN}], 2, 144) = 0
poll([{fd=4, events=POLLIN}, {fd=3, events=POLLIN}], 2, 5000) = 0
poll([{fd=4, events=POLLIN}, {fd=3, events=POLLIN}], 2, 5000) = 0
poll([{fd=4, events=POLLIN}, {fd=3, events=POLLIN}], 2, 5000) = 0
poll([{fd=4, events=POLLIN}, {fd=3, events=POLLIN}], 2, 5000) = 0
```

Its just a lame timeout to clean up when not in use:

- g_timeout_add (5000, &cleanup_timeout, NULL);

Tools

- you don’t *need* to use kernel patches.
- strace, ltrace, strace-account
- oprofile
- systemtap scripts

Tools

Measuring context switches.

```
vmstat 10
```

and check the "cs" column.

Context switches

idle runlevel 3 this should be between 10 and 20
gnome (FC6) ~400
gnome+assorted hacks ~35-40

(More on this in Ryan Lortie's talk)

Spare stuff...

autofs daemon

automount wakes the system up every second, even when it’s not in use at all

/* We often start several automounters at the same time. Add some
  randomness so we don’t all expire at the same time. */

make all timers fire at the same time instead.

dhcdbd

dhcdbd wakes up 5 times per second needlessly
Why does it have its own homegrown mainloop implementation anyway,
instead of using glib’s mainloop, which doesn’t have this flaw ?

ipw3945d

Another reason binary blobs suck.

```
 Pid 1888 ipw3945d run 301538+ ms...
   reads /sys/bus/pci/drivers/ipw3945/0000:03:00.0/cmd 2954 times, once every 102
ms avg 3008 bytes
   writes /sys/bus/pci/drivers/ipw3945/0000:03:00.0/cmd 9 times, once every 29654
ms avg 20.3333333333333 bytes
```

This activity doesn’t appear to be at all related to level of network I/O,
pretty much a constant rate of access regardless of what the system is doing.

Random detritus..

pam_timestamp_check wakes up every 5 seconds to poll if a file or two have changed.
Ideally, it would just use inotify for this rather than polling.

stickynotes applet wakes up every 100ms
to manually poke in the X event queue. It should just use an event filter

gnome-settings-daemon wakes up for every keypress
It probably has the same problem gtk had a while ago, it selects for some
xkb events, without making sure to unselect all the others:

clock applet updates every second even when seconds aren't showing
the clock applet updates itself every second, even when the clock isn't
configured to show seconds.