Bootchart 2
what you see under the hood ...
Nov 2010

Michael Meeks
michael.meeks@novell.com

“Stand at the crossroads and look; ask for the ancient paths, ask where the good way is, and walk in it, and you will find rest for your souls...” - Jeremiah 6:16
Bootchart 1
Bootchart 1 – it rocked

- **http://www.bootchart.org**
  - Ziga Mahkovec <ziga.mahkovec@klika.si>
  - Met Owen Taylor's challenge ...
  - showed us for the first time what was going on
- We noticed a lot was wrong in booting
  - and started to fix it.
  - boot times of 1 minute+ - common
  - years of accumulated bug fixes in booting:
    > *Yeah, that would be faster, but what about NFS root, with an NTP server, and remote syslogging!?* - I need to fix that fast!
  - the problems were **so** prevalent, this was no issue
Bootchart 1

- Poor resolution – 25 pixels per second, and lower res data collection.

- Processes appear to take no time, when we know they are busy: eg. boot.udev, udevadm

- Many processes appear to take no CPU time, even at startup when they are linking.

- Bootchart 1 - less truthful even than gdb!
Bootchart 1 – other issues ...

- Initial version rendered using **Java**
  - Not ubiquitous on Linux, requires compilation
- Enter: pybootchartgui
  - [http://code.google.com/p/pybootchartgui](http://code.google.com/p/pybootchartgui)
  - Anders Norgaard & Henning Niss' blow for freedom
  - Hackable: python / cairo rendering – to SVG, PNG
- Initial version data collection written in **shell**
  - while true; do cat /proc/*/stat > log-file; done
- Bootchart-collector:
  - Scott James Remnant 's contribution ...
  - re-write in C for faster collection
  - still using low-res /proc/<task>/stat data
Bootchart 2
for the world of the
ten second boot
Bootchart 2 – a new approach

- I re-wrote bootchart-collector:
  - Use kernel's taskstats interface: **ns** accurate time accounting for processes.
  - Interface baroque, unpleasant, and inefficient
    - **But** – available in all shipping kernels (unless you turn it off -go Moblin!)
  - allows us to say: “which process used how much CPU”
  - Uses PROC_EVENTS to get real process parentage
    - remove shocking pacct nonsense

- Integrates & improves pybootchartgui
  - better coupling with the collector - key.
New, and differently broken bootchart!
Which in bulk has lots of little lines:

Who is a naughty process then?

Killed feature to add alpha transparency based on %age of CPU used
Cumulative CPU usage, by process

- banshee-1 - 6072(ms) (19.10%)
- mutter - 2008(ms) (6.32%)
- init - 1764(ms) (5.55%)
- X - 1528(ms) (4.81%)
- moblin-panel-ap - 1276(ms) (4.01%)
- dbus-daemon - 1268(ms) (3.99%)
- moblin-panel-my - 1140(ms) (3.59%)
- network-manager - 1104(ms) (3.47%)
- bluetooth-panel - 992(ms) (3.12%)
- mono - 944(ms) (2.97%)
- gnome-settings- - 928(ms) (2.92%)
- empathy - 916(ms) (2.88%)
- hald - 820(ms) (2.58%)
- moblin-panel-pe - 796(ms) (2.50%)
- empathy - 780(ms) (2.45%)
Spot the SSD ...

- Interleave 'sleep' (or CPU) and 'read'
Any ideas what is happening here for many seconds?
The icing – kernel boot-charting

Crippled by software resume

Delayed module init
More detailed kernel charting:

- Pioneered by Arjan van de Ven – parsing `dmesg`
  - Kernel: "\texttt{initcall\_debug printk.time=y quiet}"
- Now its own tab in `pybootchartgui -i`
Earlier booting ...

- rdinit=bootchartd
  - Charting from the earliest initrd ...

More tools and tricks:

- A deeper dive:
  - The process is slow but where !?
  - `prctl(PR_SET_NAME, "HelloMu", 0, 0, 0);`
**Missed tricks / future fun**

- **Use:** `/proc/<pid>/schedstat`
  - Auke Koke's (nice) bootchart does this

- **Or** better kernel interfaces:
  - Polling `/proc/<pid>` is for lamers – very little changes each timeslice
  - Using taskstats – for-each-thread, for-each-cycle is even worse
  - We need to use `sched_switch` kernel tracing to reduce thrash

- **More truthfulness:**
  - Scale output to remove bootchart side-effects
  - Back propagate I/O delays in the rendering

- **Better GUI**
  - Interactive / remote rendering …
  - Graph swap / I/O delay inside process bars
  - Expand / collapse process trees interactively
Other helpful tooling ...

• In-kernel tool for better svg rendering:
  - dmesg | perl scripts/bootchart.pl > foo.svg

• Timechart – even more detailed CPU foo

• Various magic scripts for systemtap
  - http://git.fedoraproject.org/git/?p=tuned.git
  - http://git.fedoraproject.org/git/?p=tuned.git;a=blob_plain;f=doc/README.utils;hb=HEAD
    > Various scripts of goodness
    > What processes are taking what time
Conclusion / Q&A

- Bootchart2 reaches places other boot charts cannot.
  - [http://github.com/mmeeks/bootchart](http://github.com/mmeeks/bootchart)
  - Plenty more to do there, grab me afterward
    - python hackers? **Package Me!**

- Getting better data is a pre-requisite for optimising
  - Never optimize without profiling
  - Never optimise without submitting a botchart2 patch :-)

- Thanks – to all the people that did it [mostly not me]
  - Particularly Riccardo Magliocchetti (co maintainer)

Oh, that my words were recorded, that they were written on a scroll, that they were inscribed with an iron tool on lead, or engraved in rock for ever! I know that my Redeemer lives, and that in the end he will stand upon the earth. And though this body has been destroyed yet in my flesh I will see God, I myself will see him, with my own eyes - I and not another. How my heart yearns within me. - Job 19: 23-27