Learning Kernel Hacking from Clever People - A grab bag of hints & tips

CELF Embedded Linux Conference – US

16th April 2008

Hugh Blemings

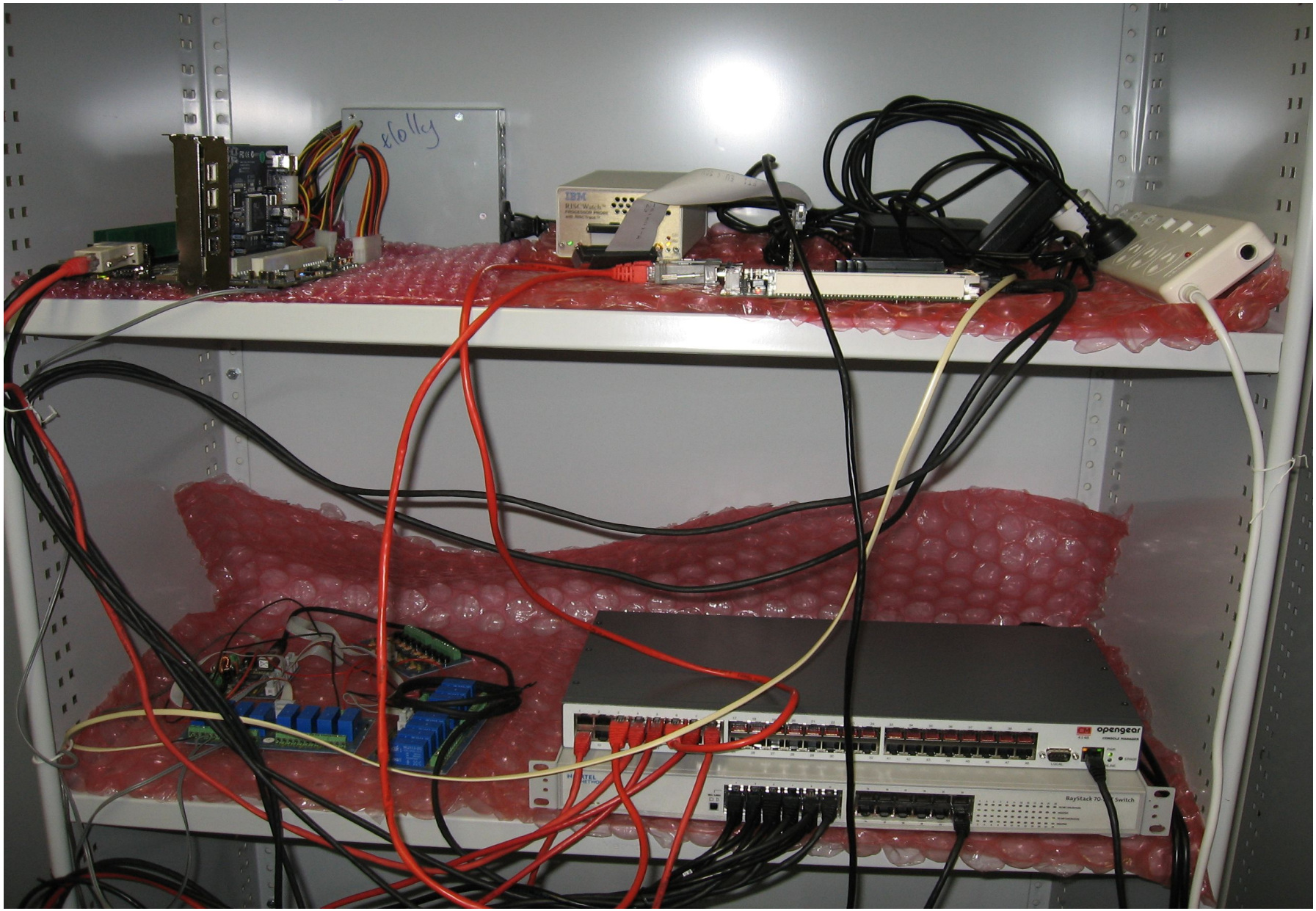
Introduction

- Who am I ?
  - Working on Free Software since mid '90s
  - My profession since 1997 :)
    - With OzLabs at IBM Linux Technology Centre since 2001
  - Mostly work on PowerPC “stuff”, particularly Linux
- What is this talk about ?
  - The tips and tricks used by experienced kernel/FOSS developers
    - I'm *not* an expert, but am fortunate to work with and know a number of them!
  - Based on own experiences with Taishan board port and suggestions solicited from colleagues
- Questions welcome at any time

Topics

- Hardware setup
- Early bringup
- Getting the kernel running
- Working with kernel source
- Building kernels
- Random datapoints & quotes

Hardware Setup



Hardware Setup (continued)

Hardware Setup (continued)

- Remote reset
- Remote power control
- Serial concentrators
- Ethernet hub / switches with “monitoring” port
- Local subnets / tftp servers
  - Sometime firmware has broken tftp/dhcp support
- In circuit debuggers
- Oscilloscope / LEDs

Early Bringup

- Simulators
- New hardware can have interesting “Features”
- Chicken Switches & Broken Things
- Ignoring firmware
  - Do some/all device initialisation yourself ?
- In memory console
- Device numbering fun
  - Linux thinks: eth0 and eth1
  - Hardware thinks: EMAC2 and EMAC3
  - PCB is labelled: Ethernet I and Ethernet II
  - You're not plugged in to any of them ?

Getting the kernel running

- Get it into memory
  - If necessary hack your bootloader to get something going
  - jtag or ICE trickery
- Some Gotchas
  - Kernel will re-initialise serial ports
    - Can lead you astray if divisors wrong or other registers incorrect
  - Don't forget `debug` on kernel command line...
  - Watch the right serial port...

Working with kernel source

- Getting a current tree
  - `git pull`
  - Matt Mackall's “ketchup”
- Visualising source tree & commits
  - `gitk`
- Finding bugs w/bisection search
  - `git bisect start`
  - `git bisect good v2.6.24`
  - `git bisect bad v2.6.25`
    - build resulting kernel
      - `git bisect good` or `git bisect bad` as appropriate
    - repeat
  - `git bisect reset`

Working with kernel source (continued)

- Quilt is wonderful
  - Modest learning curve time well spent
  - Tutorial would be a whole session in itself
- Keep work as patches
  - Makes tracking mainline much easier
  - Takes a bit of getting used to, but well worth effort
- Navigating tree
  - etags (emacs) or ctags (vi/vim)
- Generic diff visualisation
  - `dirdiff`

Building kernels

- You're going to do this a lot, so make it quick
  - Buy a honkin' big compile box
    - IBM p595 64 way POWER6
    - About 3k/s
  - Or borrow cycles
    - `ccache`
    - `distcc`
    - `ccontrol`
- Object code in separate tree to kernel source
  - `make O=../some-other-dir [...]`
    - Nice for keeping build cruft out of your source tree

Random Datapoints

- A neat conversion utility
  - `iprint`
- Inspect bit fields with ease
  - `bitfield`
- bash stuff
  - Curly brace expansion

    ```
    gcc -o foo{,.c}
    ```
  - Command line completion

    ```
    /etc/bash.completion
    ```
  - Time saving

    ```
    #for-really-long-command-lines
    ```

Random Datapoints (continued)

- Man pages to read include
  - `lsof`
  - `x11vnc`
  - `screen -x`
- A Swiss Army knife
  - `LD_PRELOAD=/path/to/some/hack/i/made.so`

Random Quotes

- “When you have a hard problem, find a smart person to help you solve it”
  - paulus (paraphrased)
- “There are many cases where you don't need to check preconditions for things you're trying to do. Simplest example is opening a file. No need to `fstat(3)` a file to see if it exists before trying `fopen(3)`. `fopen(3)` will tell you if the file isn't there. Moreover, checking first can cause a race condition, especially if the file might disappear between the `fstat(3)` and the `fopen(3)`.”
  - Tridge (paraphrased)
- “Don't send email when drunk”
  - gkh (not paraphrased :)

Close

- Questions ?
- Thankyou to
  - Stephen Rothwell, Jeremy Kerr, Paul Mackerras, Tridge, Greg Kroah-Hartman, Michael Neuling, Ronnie Sahlberg, Matt Mackall, David Gibson and Martin Schwenke
- Contributions welcome
- Links will be in online version

Close

- Questions ?
- Thankyou to
  - Stephen Rothwell, Jeremy Kerr, Paul Mackerras, Tridge, Greg Kroah-Hartman, Michael Neuling, Ronnie Sahlberg, Matt Mackall, David Gibson and Martin Schwenke
- Contributions welcome
- Links will be in online version