Intro to Kernel Hacking - finding things to work on.

Tobin C. Harding
me@tobin.cc

Slides:
http://tobin.cc/kernel-dev-finding-work.pdf

Primary key fingerprint:
28A5 A3AF D5AB 4545 BC0E A244 0312 492D 661E C3B8

Who I am and why I’m doing this talk

- I'm just some guy from someplace who likes hacking on the kernel. ~~I'm not being paid to be here~~, my opinions are my own.

Who I am and why I’m doing this talk

- I'm just some guy from someplace who likes hacking on the kernel. ~~I'm not being paid to be here~~, my opinions are my own.
- I have less than a year of full time kernel development experience.

Who I am and why I’m doing this talk

- I'm just some guy from someplace who likes hacking on the kernel. ~~I'm not being paid to be here~~, my opinions are my own.
- I have less than a year of full time kernel development experience.
- During that year I hit a wall and was stuck with nothing [kernel] to do for 2 months.

Intended audience

- This talk is targeted at wannabe kernel hackers, hopefully some of whom have a couple of patches mainlined already.

Intended audience

- This talk is targeted at wannabe kernel hackers, hopefully some of whom have a couple of patches mainlined already.
- The aim of this talk is to provide motivation/guidance to the target audience and also to get feed back from those more experienced.

Motivation

Some reasons why you might **not** want to do kernel development:

- It’s hard.

Motivation

Some reasons why you might **not** want to do kernel development:

- It’s hard.
- Some kernel developers are *prickly*.

Motivation

Some reasons why you might **not** want to do kernel development:

- It’s hard.
- Some kernel developers are *prickly*.
- It’s not *cool*, we still use email.

Motivation

Some reasons why you might **not** want to do kernel development:

- It’s hard.
- Some kernel developers are *prickly*.
- It’s not *cool*, we still use email.
- It can be lonely (if you get stuck working on something nobody cares about).

Motivation

So, why do **you** want to do kernel development?

- What motivates you to get up in the morning and stand/sit in front of your keyboard?

Motivation

So, why do **you** want to do kernel development?

- What motivates you to get up in the morning and stand/sit in front of your keyboard?
- How much time have you got to devote to this? It could easily be the best part of a year (full time) before you are *really* doing anything useful.

Motivation

Some reasons why you might **want** to do kernel development:

Motivation

Some reasons why you might **want** to do kernel development:

- It’s cool, we still use email.

Motivation

Some reasons why you might **want** to do kernel development:

- It’s cool, we still use email.
- If you are the sort of person who likes programming, systems programming, systems programming in C, open source systems programming in C ...

Motivation

Some reasons why you might **want** to do kernel development:

- It’s cool, we still use email.
- If you are the sort of person who likes programming, systems programming, systems programming in C, open source systems programming in C ..
- Most kernel developers are polite, well mannered, and extremely generous with their time. Many will go out of their way to help you if you ask the correct questions in the correct manner and are seen to be putting in effort.

Motivation

Basically, if this sort of work is meaningful to you on an individual level and you feel you have the mental, financial, and temporal resources then you should definitely pursue it - for the right person kernel hacking is a **seriously** satisfying way to pass your days.

Skills - social

'The main issues faced by any large software system are not technical issues but people issues' – unknown (possibly paraphrased)

- Get some people skills, read some books (How to win friends and influence people, more aptly named: 'How not to be a dick').
- Pay attention, have 'beginners mind'. Everything is done in the open so observe how others act and imitate.

Skills - technical

- Learn C **well**:
  - **The C Programming Language** - Brian W. Kernighan and Dennis M. Ritchie.
  - **The Linux Programming Interface** - Michael Kerrisk.

  Unless you have written a **lot** of C then do yourself a favour and complete the above books including all exercises.

- Learn OS theory (and Linux in particular):
  - **Operating System Concepts** - Avi Silberschatz, Peter Baer Galvin, Greg Gagne.
  - **Linux Device Drivers** - Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman
  - **Linux Kernel Development** - Robert Love.

Good news

- There is unlimited amounts of work to do in the kernel.

Good news

- There is unlimited amounts of work to do in the kernel.
- The kernel community is very appreciative and welcoming of any *beneficial* help.

Guidelines

- Don't break anything.

Guidelines

- Don't break anything.
- Be exceedingly respectful of other peoples time.

Guidelines

- Don't break anything.
- Be exceedingly respectful of other peoples time.
- Work on what interests you.

Guidelines

- Don't break anything.
- Be exceedingly respectful of other peoples time.
- Work on what interests you.
- Focus on interesting **tractable** problems.

One method of progression

- Do your first patch.
- Do your first patch set.
- Do a bunch of checkpatch cleanup patches in staging.
- Get some real hardware and write a driver.
  - Pick hardware with a driver currently in staging.
  - Pick hardware that has a similar driver intree.

  (Please do not patch any kernel directory outside of staging until you have a bunch of patches merged (IMO at least 100)).

  then ...

Moving on from Staging

This talk hinges on three points:

- You can't/shouldn't do cleanups outside of staging.

Moving on from Staging

This talk hinges on three points:

- You can't/shouldn't do cleanups outside of staging.
- You can't change code when you don't understand it.

Moving on from Staging

This talk hinges on three points:

- You can't/shouldn't do cleanups outside of staging.
- You can't change code when you don't understand it.
- Even when you find something that needs doing, most likely it will not get accepted. Or not easily.

This is because the kernel is based on trust.

You get patches in the kernel when people trust that you will be around to fix the bugs you introduce.

However

- Kernel hackers tend to have more things to work on than they have time.
- You can convince such a hacker to give you some of their work.

But hang on!

Giving work to newbies takes effort too

But hang on!

Giving work to newbies takes effort too

- It takes time to explain it.

But hang on!

Giving work to newbies takes effort too

- It takes time to explain it.
- Reviewing patches takes time, especially newbie patches.

But hang on!

Giving work to newbies takes effort too

- It takes time to explain it.
- Reviewing patches takes time, especially newbie patches.
- They have a nasty habit of pestering you when they either can't do it or they do it wrong.

Get some people skills

If you can convince a kernel hacker to spend some of their time on you *they* benefit also

Get some people skills

If you can convince a kernel hacker to spend some of their time on you *they* benefit also

- It's nice when things that you want done get done by other people.

Get some people skills

If you can convince a kernel hacker to spend some of their time on you *they* benefit also

- It's nice when things that you want done get done by other people.
- It's nice when people respect you, asking someones guidance is a sign of respect.

Get some people skills

If you can convince a kernel hacker to spend some of their time on you *they* benefit also

- It's nice when things that you want done get done by other people.
- It's nice when people respect you, asking someones guidance is a sign of respect.
- You learn things by teaching, hackers tend to like learning things.

You need to convince them, and yourself, that you are worth the effort

You need to convince them, and yourself, that you are worth the effort

- You have the base skill set (see above).

You need to convince them, and yourself, that you are worth the effort

- You have the base skill set (see above).
- You are putting in the effort.

You need to convince them, and yourself, that you are worth the effort

- You have the base skill set (see above).
- You are putting in the effort.
- You pay attention, no one likes saying things twice.

You need to convince them, and yourself, that you are worth the effort

- You have the base skill set (see above).
- You are putting in the effort.
- You pay attention, no one likes saying things twice.
- You are pleasant to interact with (you heard the bit about people skills right?).

You don't always need to ask directly. Often while talking, kernel developers may mention some thing they have been meaning to do, or they wish someone would do. Make a note of it, have a go at it, if you can get part way started that is enough to show 3 of the 4 points above. They will most likely step in and guide you on the things you are stuck on.

Talk to real kernel developers

- Go to conferences.
- Lurk on the mailing lists (kernel newbies and driver dev list).

If anyone mentions anything that you feel you can do or you are interested in researching then do it. This is a gift economy, try to give without taking. Do stuff that helps others while demanding as little as possible of them in return.

Summing up

Finding things to work on in the kernel may be not so much about the technical aspects of finding something to work on but rather the social aspects. Be patient, while at first it seems impossible to find something to do, very soon you will be swamped in tasks. There is unlimited amounts of work to do and no rush, take your time, do what you do meticulously. Expect that your changes will not get accepted, or if they do, not for a good while. You may have more success if you separate your patches from your ego. Or follow the concept of working not for the fruit of your labour but for the labour itself. Finally, when you wake up one morning and a patch set you put in has 20 responses to it or the first time you get an email from <insert kernel god here>, it will all be worth it. Money can't buy that feeling. Some of the best programmers in the world work on the kernel, and a bunch of them are very pleasant to work with.

Thanks, enjoy yourself!