Linux Kernel Hacking 101

Kelley Nielsen
salticidoftheearth.com

Talk website:
http://tinyurl.com/m97b5r5

The process of kernel hacking is a

CYCLE

The Creative Cycle

- Code your changes
- Send in your patch
- Gather feedback
- Repeat

The Creative Cycle

- Find a contribution to make
  - Read stored communications
  - Gain experience
  -

The Creative Cycle

- Find a contribution to make
  - Read stored communications
  -
  -

The Creative Cycle

- Find a contribution to make
  - Read stored communications
  - Gain experience
  -

The Creative Cycle

- Find a contribution to make
  - Read stored communications
  - Gain experience
  - Ask

The Creative Cycle

- Code your contribution
- Prepare and send the patchset

The Creative Cycle

- Gather feedback
  -
  -
  -
  -

The Creative Cycle

- Gather feedback
  - Testing results
  -
  -
  -

The Creative Cycle

- Gather feedback
  - Testing results
  - Mentoring and guidance
  -
  -

The Creative Cycle

- Gather feedback
  - Testing results
  - Mentoring and guidance
  - Discussion of strategies
  -

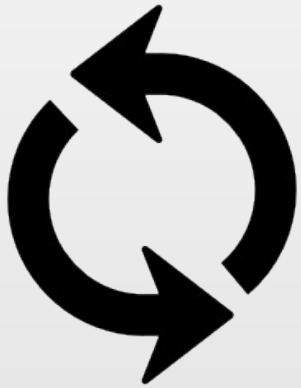
The Creative Cycle

- Gather feedback
  - Testing results
  - Mentoring and guidance
  - Discussion of strategies
  - General suggestions

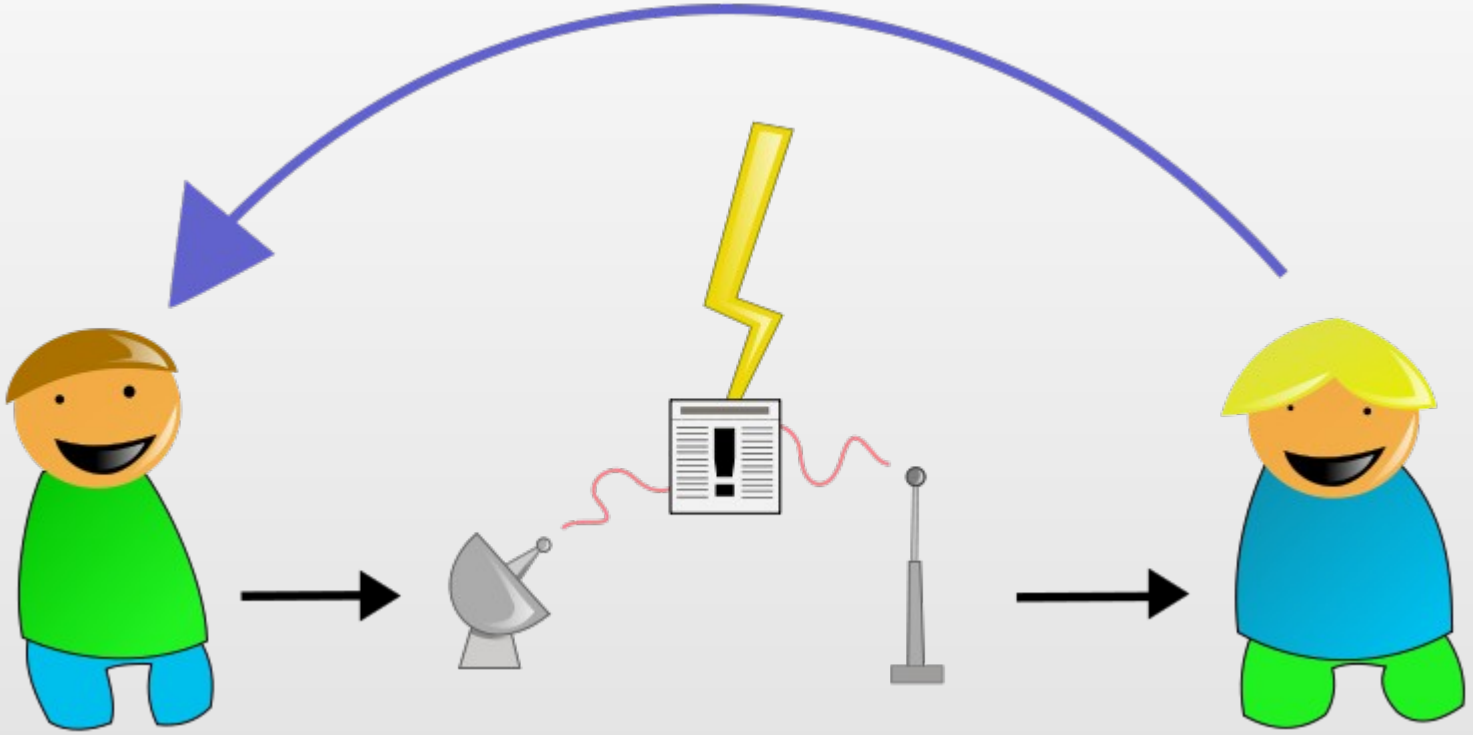
The Creative Cycle

- Incorporate changes
  - Use feedback to improve
  - If your contribution is rejected, find another
- Regenerate and resend your patchset

Repeat the previous two slides until
Your patchset is accepted…
Then repeat with a new patchset…



Communicating with team members and community



Communicating

- Mailing lists, private email
-
-
-

Communicating

- Mailing lists, private email
- IRC
-
-

Communicating

- Mailing lists, private email
- IRC
- Conferences
-

Communicating

- Mailing lists, private email
- IRC
- Conferences
- Linux Weekly News

Coding your changes

Coding your changes

- Plan
- Code
- Compile & run
- Test & debug

Coding your changes

- Plan
  - Trace existing paths of execution
  - Find examples similar to your goal
  -
  -

Coding your changes

- Plan
  - Trace existing paths of execution
  - Find examples similar to your goal
  -
  -

Coding your changes

- Plan
  - Trace existing paths of execution
  - Find examples similar to your goal
  - Learn which lines are relevant
  -

Coding your changes

- Plan
  - Trace existing paths of execution
  - Find examples similar to your goal
  - Learn which lines are relevant
  - Your friends are grep and the lxr ident search

Coding your changes

- Hack your code
  - Language is C
  - Follow kernel coding style
  - Gain skill with your editor

Coding your changes

- Compile and run
  - This involves installation and rebooting!
  - ne
  -

Coding your changes

- Compile and run
  - This involves installation and rebooting!
  - Use a virtual machine or dedicated machine
  -

Coding your changes

- Compile and run
  - This involves installation and rebooting!
  - Use a virtual machine or dedicated machine
  - Maintain a known working and a test kernel

Coding your changes

- Test and debug
  - Your friend is printk()
  -
  -

Coding your changes

- Test and debug
  - Your friend is printk()
  - Find messages in /var/log/kern.log
  -

Coding your changes

- Test and debug
  - Your friend is printk()
  - Find messages in /var/log/kern.log
  - Create testing scenarios

Generating and Delivering Your Patch

Delivering Your Patch

- Git commit strategy
- Git branching strategy
- Composing the commit message
- The patch

Delivering Your Patch

- Git commit strategy
  - Make small incremental changes
  - Make changes in logical order
  - Each change should be encapsulated
  -
  -

Delivering Your Patch

- Git commit strategy
  - Make small incremental changes
  - Make changes in logical order
  - Each change should be encapsulated
  - Be prepared to alter previous commits
  -

Delivering Your Patch

- Git commit strategy
  - Make small incremental changes
  - Make changes in logical order
  - Each change should be encapsulated
  - Be prepared to alter previous commits
  -

Delivering Your Patch

- Git commit strategy
  - Make small incremental changes
  - Make changes in logical order
  - Each change should be encapsulated
  - Be prepared to alter previous commits
  -

Delivering Your Patch

- Git commit strategy
  - Make small incremental changes
  - Make changes in logical order
  - Each change should be encapsulated
  - Be prepared to alter previous commits
  - Be prepared for upstream changes as you update your sources

Delivering Your Patch

- Git branching strategy
  - Keep master branch unmodified
  - Create a working branch
  -
  -

Delivering Your Patch

- Git branching strategy
  - Keep master branch unmodified
  - Create a working branch
  -
  -

Delivering Your Patch

- Git branching strategy
  - Keep master branch unmodified
  - Create a working branch
  - Use a throwaway branch to squash commits
  -

Delivering Your Patch

- Git branching strategy
  - Keep master branch unmodified
  - Create a working branch
  - Use a throwaway branch to squash commits
  - If this method doesn’t work for you, try another

Delivering Your Patch

- Composing the commit message
  - Work as hard on this as you do on coding!
  -
  -
  -

Delivering Your Patch

- Composing the commit message
  - Work as hard on this as you do on coding!
  - Read existing examples
  -
  -

Delivering Your Patch

- Composing the commit message
  - Work as hard on this as you do on coding!
  - Read existing examples
  - Be concise, yet thorough
  -

Delivering Your Patch

- Composing the commit message
  - Work as hard on this as you do on coding!
  - Read existing examples
  - Be concise, yet thorough
  - Use imperative language

Delivering Your Patch

- The patch itself
  - Run checkpatch.pl on the files you change
  -
  -
  -

Delivering Your Patch

- The patch itself
  - Run checkpatch.pl on the files you change
  - Generate patch with a git command
  -
  -

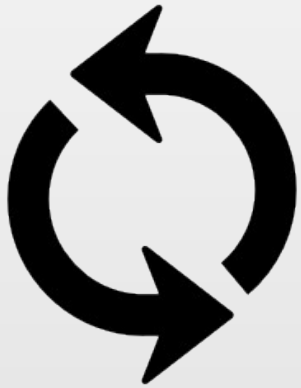
Delivering Your Patch

- The patch itself
  - Run checkpatch.pl on the files you change
  - Generate patch with a git command
  - Cc everyone involved with the code
  -

Delivering Your Patch

- The patch itself
  - Run checkpatch.pl on the files you change
  - Generate patch with a git command
  - Cc everyone involved with the code
  - Send using plain text mail

Now wait for your feedback,
And do it all again…



Happy Hacking!

Talk website:
http://tinyurl.com/m97b5r5

Resources

- Kernel.org git repositories https://git.kernel.org/cgit/
- Linux Kernel Newbies http://kernelnewbies.org/
- Outreachy First Patch Tutorial http://kernelnewbies.org/Outreachyfirstpatch
- The Eudyptula Challenge http://eudyptula-challenge.org/
- KVM Installation https://help.ubuntu.com/community/KVM/Installation
- Linux Kernel Mailing List https://lkml.org/
- Various Linux related mailing lists at Gmane http://gmane.org/find.php?list=kernel
- Linux Foundation events http://events.linuxfoundation.org/
- Linux Weekly News http://lwn.net/
- #irchelp http://www.irchelp.org/
- Irssi – The Client of the Future http://www.irssi.org/
- Wikipedia entry for the grep command http://en.wikipedia.org/wiki/Grep
- Linux Cross Reference at Free Electrons http://lxr.free-electrons.com/
- Linux Kernel Coding Style (pdf) https://computing.llnl.gov/linux/slurm/coding_style.pdf
- Vim the editor http://www.vim.org/
- Pro Git http://git-scm.com/book https://www.gitbook.io/book/gitbookio/progit
- The Mutt E-Mail Client http://www.mutt.org/
- Video of this talk from ScaLE 13x https://www.youtube.com/watch?v=WUU-gI3iYnY

Image Credits

- Arrows Circle by Freepik CC BY 3.0
- Communication shannon-weaver2 by Einar Faanes CC BY-SA 3.0
- Xaric screen shot by Triddle BSD License http://commons.wikimedia.org/wiki/File:Xaric_screen_shot.jpg
- Coding All Night Long by Snatcherdudette http://snatcherdudette.deviantart.com/art/Coding-all-night-long-183815498
- Music present by Marta Crowe Creative Commons Attribution 2.0 Generic
- Linux Foundation Logo is in the public domain
- Linux “Tux” Logo is in the public domain