Contributing to Linux Kernel

Vaishali Thakkar (Oracle)

Vaishali.thakkar@oracle.com

https://vthakkar1994.wordpress.com/

https://github.com/vthakkar1994/Linux-Kernel-Workshop

What you need

- Git
- Source-code of linux-next/staging-testing
- Your favorite text-editor [Vim]
- Mail transport client esmtp
- Mail client mutt or git send-email for sending patches
- Instructions available at:

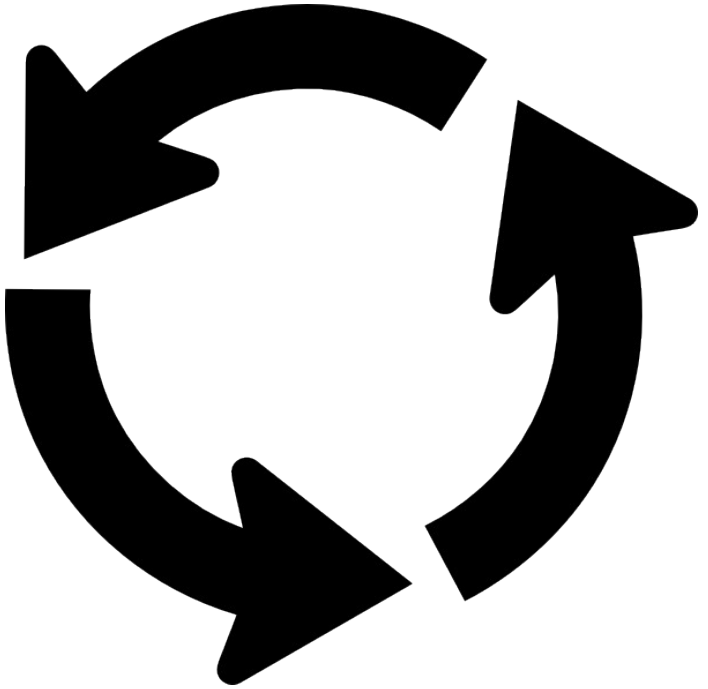
https://github.com/vthakkar1994/Linux-Kernel-Workshop

What we are going to cover

- Work flow cycle of the Linux kernel
- Linux kernel communication and coding style
- Git basics
- Understanding patch best practices with generating and mailing first patch
- What after sending first patch?

The process of the kernel hacking is a

CYCLE



The creative cycle

- Code your changes

The creative cycle

- Code your changes
- Send in your patch

The creative cycle

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

The creative cycle

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

Communication style

- Mailing lists – scripts/get_maintainer.pl

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
  - Consistent communictaion style

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
  - ✔ Consistent communication style
    - Respond inline – Say NO to top-posting

From: Kludge Crufty <example@email.com>

Subject: Design decisions for next release

On Fri, Sep 12, 2014 at 03:00:56PM -0700, Baz Quux wrote:

> On Fri, 12 September 2014 at 02:30:17PM -0700, Foo Bar wrote:

>

> I think we should do X.

 I think we should do Y.

Kludge

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
  - ✔ Consistent communication style
  - ✔ Respond inline
  - – Make sure to use one of the standard email clients listed in Documentation/email-clients.txt.
    1. Mutt
       - sudo apt-get install mutt
    2. git send-email
       - sudo apt-get install git-email
    3. Thunder bird

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
  - ✔ Consistent communication style
  - ✔ Respond inline
  - ✔ Use standard email clients
    - Do NOT use the gmail web interface

      WHY: It line-wraps the mail

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
  - ✔ Consistent communication style
  - ✔ Respond inline
  - ✔ Use standard email clients
  - ✗ Do NOT use the gmail web interface
    - DO NOT use outlook

      WHY: It mangles patches (turns tabs into spaces).

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
  - ✔ Consistent communication style
  - ✔ Respond inline
  - ✔ Use standard email clients
  - ✗ Do NOT use the gmail web interface
  - ✗ DO NOT use outlook
  - ✗ Don't include quotes in your signature

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
- Avoid sending private mails
    - Likely to be missed
    - Not encouraged by developers/maintainers

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
- Avoid sending private mails
- IRC Ettiquetts
  - NEVER SHOUT! Using all capital letters is the same as screaming

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
- Avoid sending private mails
- IRC Ettiquetts
  - ✗ NEVER SHOUT! Using all capital letters is the same as screaming
  - Be considerate. When you are asking for help, being rude or pushy will rarely get you an answer to your question.

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
- Avoid sending private mails
- IRC Ettiquetts
  - ✗ NEVER SHOUT! Using all capital letters is the same as screaming
  - ✔ Be considerate. When you are asking for help, being rude or pushy will rarely get you an answer to your question.
  - ✔ Be patient. People have their full-time jobs as well.

Communication style

- Mailing lists – scripts/get_maintainer.pl
- Responding to emails
- Avoid sending private mails
- IRC Ettiquetts
- Remember, anything you post on the internet is there FOREVER.

Coding Style

- Documentation/codingstyle

Coding Style

- Documentation/codingstyle
- Use Tabs

Coding Style

- Documentation/codingstyle
- Use Tabs
- All tabs are 8 characters

    - 'set tabstop=8'

Coding Style

- Documentation/codingstyle
- Use Tabs
- All tabs are 8 characters
- 80 character line limit

Coding Style

- Documentation/codingstyle
- Use Tabs
- All tabs are 8 characters
- 80 character line limit
- Run scripts/checkpatch.pl before sending any patch

- Open source
- Created by Linux in 2005 to work on Linux kernel
- Fastest version control system
- Installation: 'sudo apt-get install git'
- Setting up .gitconfig

Basic commands

- Git clone

  git clone git://git.kernel.org/pub/scm/linux/kernel/git/gregkh/staging.git

- Git branch

  git checkout -b first-patch

- Git status

Creating and emailing first patch

- Run scripts/checkpatch.pl

  scripts/checkpatch.pl –file –terse directory/file.c

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code
- Compile the change

  - make directory/file.o

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code
- Compile the change
- Create the patch

  - git add directory/file.c

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code
- Compile the change
- Create the patch
  - ✔ git add directory/file.c

    - git commit -s -v

    - Adds signed-off by line and shows the diff you are committing

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code
- Compile the change
- Create the patch
  - ✔ git add directory/file.c
  - ✔ git commit -s -v
    - git format-patch -o /tmp/ HEAD^
    - -o flag specifies where to put the patch

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code
- Compile the change
- Create the patch
  - ✔ git add directory/file.c
  - ✔ git commit -s -v
  - ✔ git format-patch -o /tmp/ HEAD^
    - Getting the list of maintainers
    - scripts/get_maintainer.pl /tmp/xxx.patch

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code
- Compile the change
- Create the patch
  - ✔ git add directory/file.c
  - ✔ git commit -s -v
  - ✔ git format-patch -o /tmp/ HEAD^
  - ✔ Getting the list of maintainers

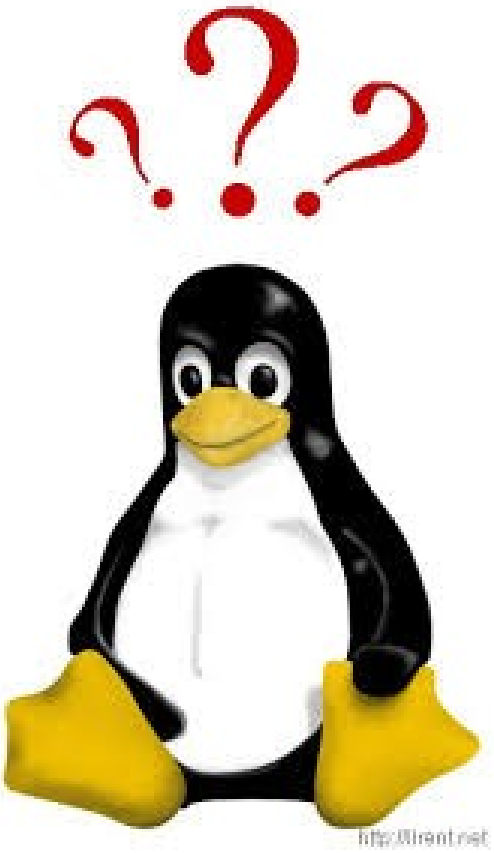
    - Sending your patch with git send-email or mutt

    - git send-email /tmp/xxx.patch | mutt -H /tmp/xxx.patch

Creating and emailing first patch

- Run scripts/checkpatch.pl
- Pick one warning and change the code
- Compile the change
- Create the patch
  - ✔ git add directory/file.c
  - ✔ git commit -s -v
  - ✔ git format-patch -o /tmp/ HEAD^
  - ✔ Getting the list of maintainers
  - ✔ Sending your patch with mutt or git send-email

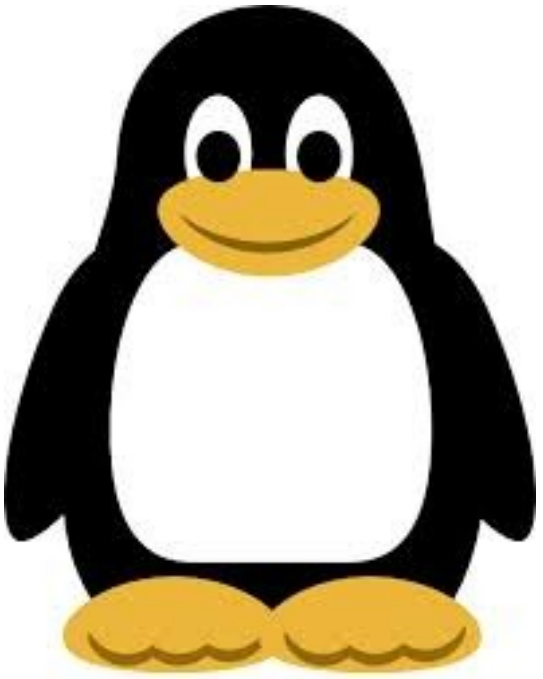
Questions



What after sending first patch?

- Repeat the work-flow cycle
- Use bug finding tools [sparse, smatch, coccinelle etc]
- Work on API functions
- Work on easy to fix issues of y2038 problem, devm functions etc
- Work on drivers or topics of your interest in the linux-kernel

Welcome to the Linux Kernel community



Thank You