Your First Linux Kernel Patch

http://people.redhat.com/mwhitehe/KernelPatchingRPI.pdf

Take frequent stretch breaks

“The brain can only absorb what the butt can endure”

- Captain Robert “Bubba” Hagg

A Reason to be Persistent

- https://lkml.org/lkml/2004/12/20/255

  ‘So at one level I absolutely _hate_ trivial patches: they take time and effort to merge, and individually the patch itself is often not really obviously "worth it". But at the same time, I think the trivial patches are among the most important ones - exactly because they are the "entry" patches for every new developer.’

  - Linus Torvalds

Login to your System

- You should have already done these steps:

```
# This takes a long time over the network
git clone git://git.kernel.org/pub/scm/linux/kernel/git/davem/net-next.git

cd net-next

# This is computationally intensive
git branch  tutorial-devel ; git checkout tutorial-devel
```

Use tmux if needed to share a login

- On first login:
  - tmux -S /tmp/my_session_name
  - chmod 777 /tmp/my_session_name
- On remaining logins:
  - tmux -S /tmp/my_session_name attach

Multiple Developers?

- The primary developer uses the “Signed-off-by:” line provided by “--signoff” flag passed to git (shown later).
- Other participants should add lines in the commit message to get credit.

  Reviewed-by:

  Tested-by:

  Suggested-by:

  Reported-by:

Configure yourself in git

- Create your identity (goes in ~/.gitconfig)

```
git config --global user.email “myemail@rpi.edu”
git config --global user.name “My Name”
git config --global sendemail.smtpencryption tls
git config --global sendemail.smtpserver mail.rpi.edu
git config --global sendemail.smtpuser myemail
git config --global sendemail.smtpserverport 587
```

Test email

echo “Subject: testing” > ~/test.email

git send-email -to tedheadster@gmail.com ~/test.email

For the netdev mailing list, see what other people are submitting

- The netdev kernel mailing list is one of a few that use the ‘patchwork’ software to track patches semi-automatically. It is run by Dave Miller
- Look at existing entries and copy their style

http://patchwork.ozlabs.org/project/netdev/list/

printk()

- Presently there are many examples of printk(“my message %d\n”, int_arg);
- The accepted standard is to use printk() with a leading argument.
- printk(KERN_INFO “my message %d\n”, int_arg);

printk() levels

```
KERN_DEBUG
KERN_INFO
KERN_NOTICE
KERN_WARNING
KERN_ERR
KERN_CRIT
KERN_ALERT
KERN_EMERGENCY
```

Edit code and commit it

- Do a ‘cd net-next/drivers/net/ethernet’
- Edit code
- As of 10/15/2013 I see 9557 printk statements that need patches.
- When happy, do a ‘git commit -a’. Note: the first line becomes your Subject line in the email, so choose it well.
- If you are working in a group, secondary authors should be at bottom of commit with “Reviewed-by: First Last <email@email.com>” to get credit.

Go forth and code

Generate the patch

- Do a ‘cd ~/net-next’
- Generate patch with

  git format-patch --subject-prefix

  “PATCH net-next” --signoff master
- A file is created starting with ‘0001-’

Check your style

- The Linux kernel has a style checking included
- scripts/checkpatch.pl
- Use it!

Send a test email of the patch

- Email the patch to an alternate personal email as a test, because it automatically CC's your address in ~/.gitconfig.

```
user@linux:~$ git send-email -to
another_one@email.com 0001-patchfile
```

Questions on what we did?