Contributing to Linux kernel

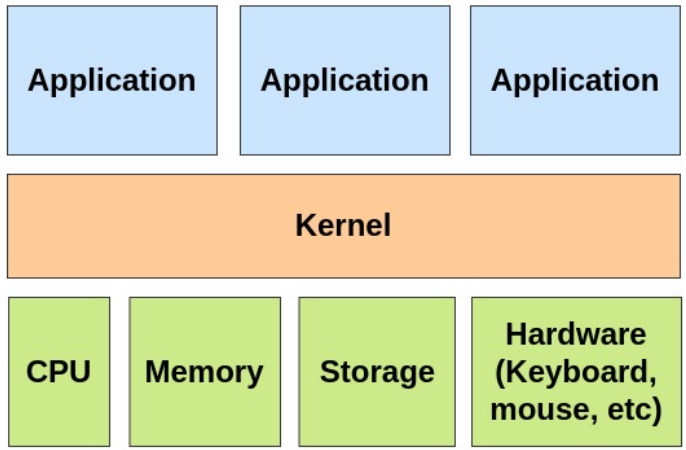
miniDebConf 2023

Villupuram

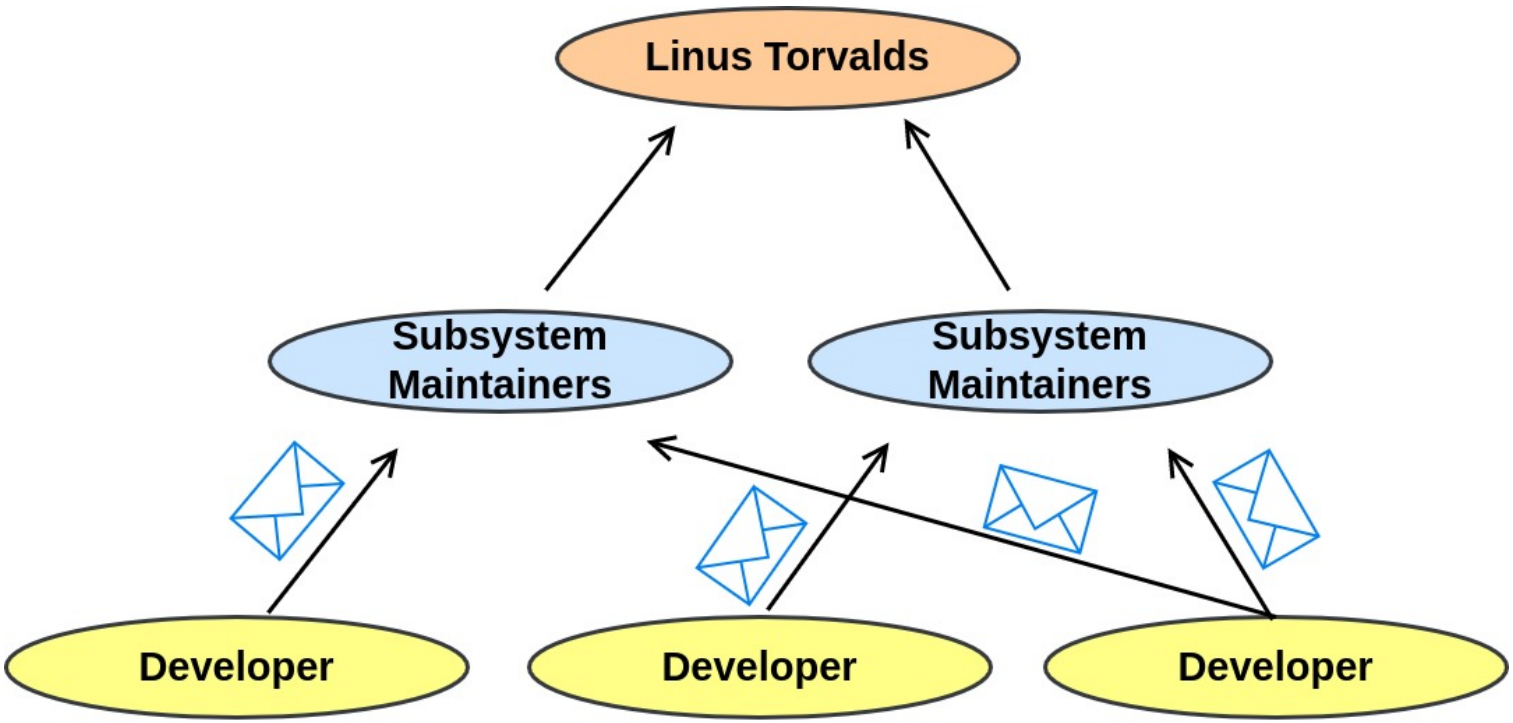
https://nihaal.me

Linux Kernel

- Kernel
  - Interface between hardware and software
  - Resource manager
- Linux powers
  - Top 500 Supercomputers
  - Desktops
  - Single board computers (Raspberry Pi, Beagle Bone)
  - Android
- Actively developed in the open
  - By developers across the globe
  - Supported by multiple companies



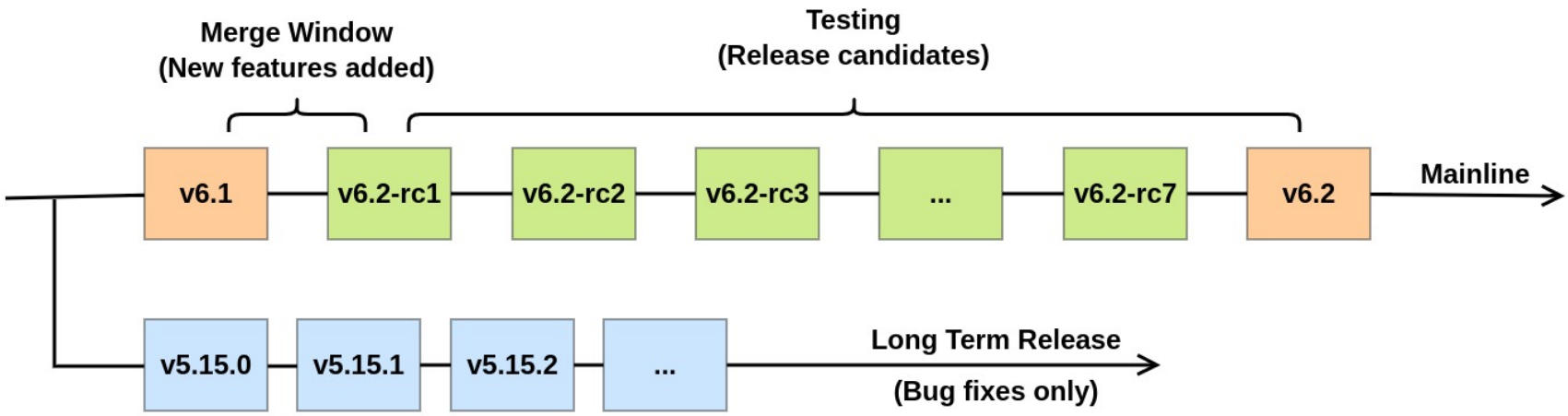
Linux development workflow



Patch

- A single code change (commit)
- Contains
  1. Subject line
  2. Description of the change
  3. Diff

Kernel versions



- Each subsystem also has a next tree
- Linux-next combines latest commits from all next trees

Ways to start contributing

1. Testing
2. Documentation
3. Cleaning up staging drivers

Why contribute?

- To give back
- To learn Git, C, how Operating Systems work
- To build a career
  - Open source contribution
  - Linux kernel developer jobs

1. Testing

- Test *Linux-next* on your local system
  - Download source code
  - Add linux-next remote tree and fetch tags
  - Checkout to the latest next tag
  - Compile and install

  `https://lists.kernelnewbies.org/pipermail/kernelnewbies/2017-April/017765.html`
- Report any bugs or warnings to the respective mailing lists

2. Documentation

1. Fix Sphinx build warnings (kernel doc comments)
   `https://lwn.net/Articles/810404/`
2. Updating documentation to reflect actual code
3. Documenting sysfs attributes ABI
   - Under *Documentation/ABI*

```
What:       /sys/class/backlight/<backlight>/brightness
Date:       April 2005
KernelVersion:  2.6.12
Contact:    Richard Purdie <rpurdie@rpsys.net>
Description:
        Control the brightness for this <backlight>. Values
        are between 0 and max_brightness. This file will also
        show the brightness level stored in the driver, which
        may not be the actual brightness (see actual_brightness).
Users:      HAL
```

   - `https://aishpant.dev/blog/outreachy-recap/`

3. Cleaning up staging drivers

- Staging drivers
  - Under *drivers/staging* in the kernel source tree
  - Drivers not ready for production
  - Beginners are encouraged to contribute
- Clean up?
  - Conform to kernel coding style (using checkpatch.pl)
  - Remove unused functions, temporary variables
  - TODO files
- How to send your first patch?
  - `https://kernelnewbies.org/FirstKernelPatch`

3. Cleaning up staging drivers

1. Set up and configure tools
   - git
   - Editor
   - git send-email
   - Mail client

3. Cleaning up staging drivers

2. Download Linux source code and checkout to the right tree

- Download source

```
$ git clone
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
$ git remote add staging
https://git.kernel.org/pub/scm/linux/kernel/git/gregkh/staging.git/
$ git checkout -b development staging/staging-next
```

- Compile the kernel with CambridgeUS configuration

```
$ zcat /proc/config.gz > .config
$ make -j16
```

Avoid

- Working on the wrong tree
  - Always use Subsystem-specific *next* tree
  - Check “t:” entry in MAINTAINERS file

3. Cleaning up staging drivers

3. Clean up

- Use Checkpatch (or other tools) to find warnings to fix

```
$ ./scripts/checkpatch.pl -f drivers/staging/somedriver/somefile.c
```

- Some checkpatch warnings are to be ignored.
  Read `https://kernelnewbies.org/CheckpatchTips`
- Fix the warning

4. Commit and generate patch

```
$ git add drivers/staging/somedriver/somefile.c
$ git commit -s -v
$ git format-patch -1
```

Remember

Each patch must make only one change and be easy to review

3. Cleaning up staging drivers

Follow correct patch format

- Correct subject line prefix
- Explain why your changes are needed
- Signed-off-by line

5. Test your patch

- Make sure it compiles. *Never break the build*.
- Test with hardware if available
- Ensure it doesn't introduce new warnings
  - Run *scripts/checkpatch.pl*
  - Compile with C=1 (Sparse check)

3. Cleaning up staging drivers

6. Send your patch

- Find whom to send to
  - Run *scripts/get_maintainers.pl*

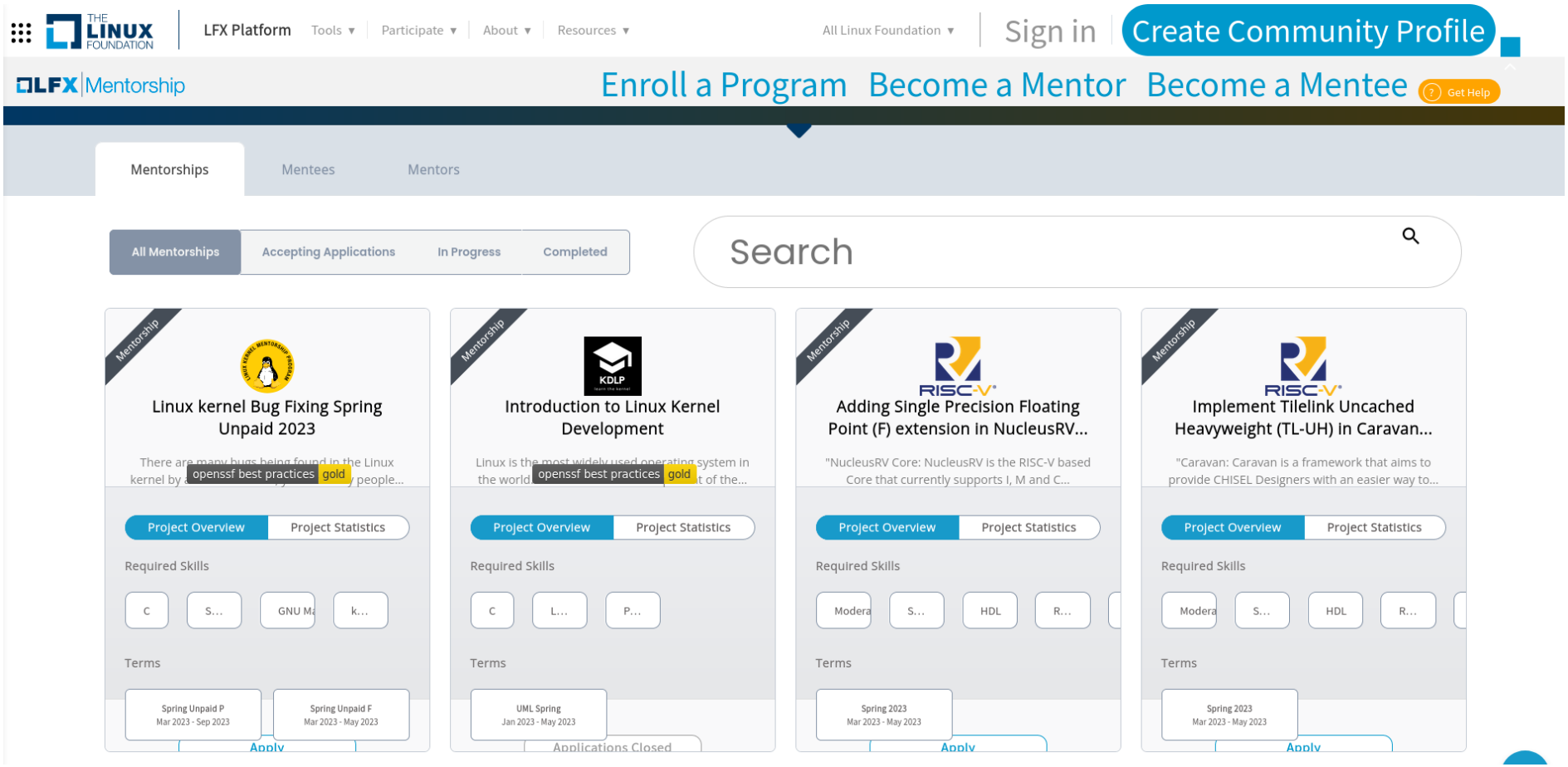
7. Respond to review comments and resend

- When sending revision, explain changes from previous version
- Follow mailing list etiquette
  - Plaintext mails
  - No top posting
  - No attachments (use pastebin)

What next? Moving into development

1. Find and fix bugs
   - Static checkers: Sparse, Coccinelle, Smatch
   - Syzbot bugs
2. Keep learning
   - A Beginner's Guide to Linux Kernel Development (LFD103)
   - LF live mentorship sessions
   - Linux weekly news (`https://lwn.net`)
3. Do an internship
   - LFX mentorship
   - Outreachy
   - Collabora, Red Hat, etc
4. Pick a subsystem and get deep into it
   - Watch mailling list conversations
   - Read related documentation
   - Fix bugs in that subsystem

LFX bug fixing internship is now accepting applications



Conclusion

- Takeaways
  1. You can contribute to the kernel
  2. Send your first patch today!
     `https://kernelnewbies.org/FirstKernelPatch`
  3. Apply for Linux kernel bug fixing internship
     `https://mentorship.lfx.linuxfoundation.org/project/f629822d-978e-437d-a617-28a02f428ee7`
- Happy hacking! நன்றி