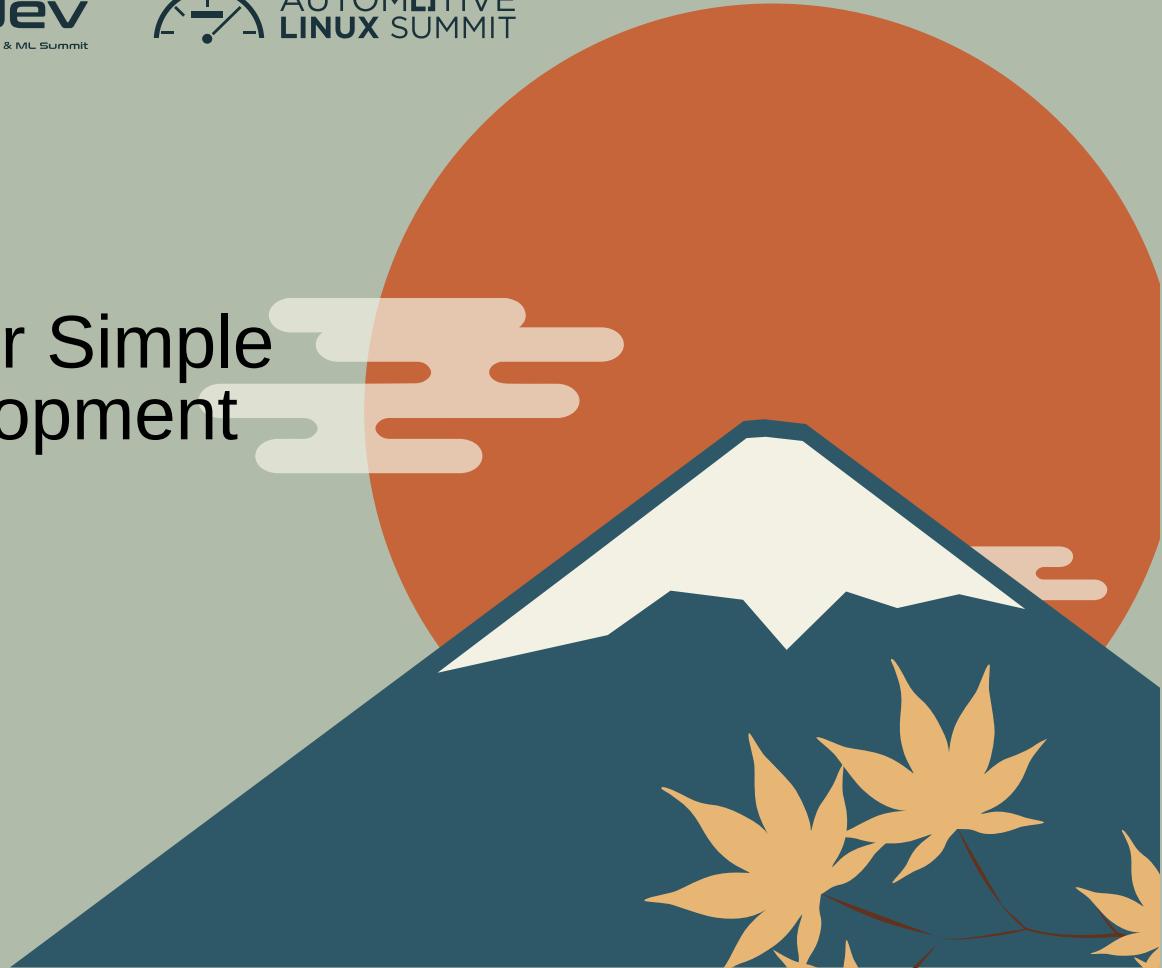




# hkml: Mailing Tool for Simple Linux Kernel Development

SeongJae Park (SJ)  
[<sj@kernel.org>](mailto:sj@kernel.org)  
[<sjpark@crusoe.ai>](mailto:sjpark@crusoe.ai)



# About The Speaker: SJ

- Kernel Programmer, maintaining DAMON
- Working for crusoe.ai (we're hiring!)
- Was working for Meta when this talk was submitted
  - And don't know how to update talk intro on the webpage (sorry if it confused you)
  - All opinions are speaker's own, though

hkml: Mailing Tool for Simple Linux Kernel Development - SeongJae (SJ) Park, ~~Meta~~ crusoe.ai

# Table of Contents

- hkml: Mailing Tool for Simple Linux Kernel Development (40 minutes)
  - Linux Kernel Development Process and Pain Points (7 minutes)
  - hkml: Hackers' mailing tool (3 minutes)
  - hkml live demo (20 minutes)
  - QnA (10 minutes)

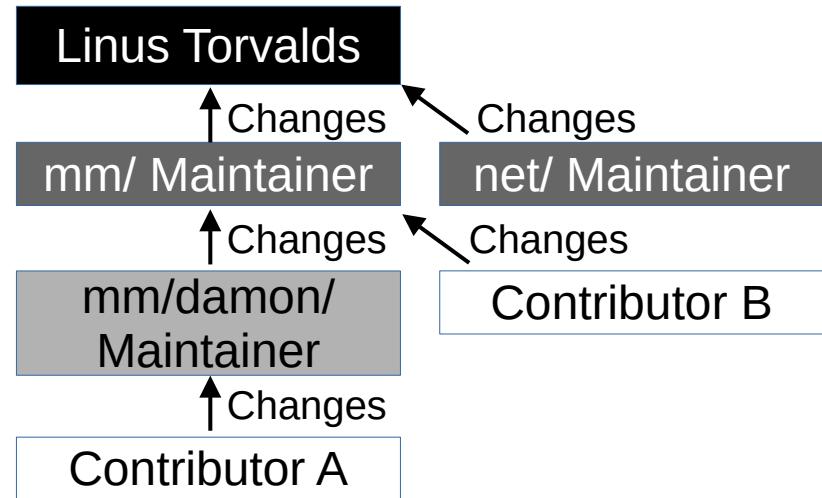
# Linux Kernel Development Process and Its Pain Points

# Linux Kernel Development Statistics

- Survived from the test of time, since 1991
- Being used nearly everywhere
- Keeping and even accelerating development speed
  - ~2,000 developers, ~15,000 commits, per ~9 weeks
- Most successful and active open source software

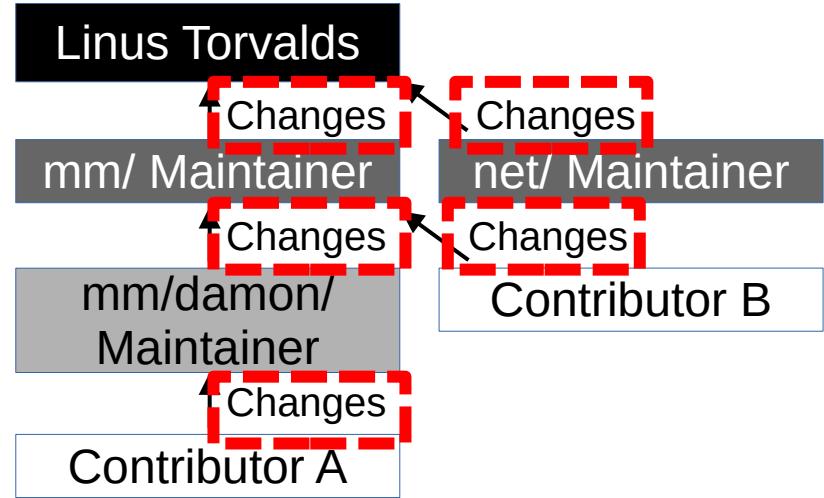
# Kernel Development Process, 1000 ft View

- Kernel: A set of (recursive) subsystems
- One maintainer team per subsystem
- Contributors send changes to maintainers
- Maintainers send changes to upper level maintainer
- Distributed, scalable, simple



# Individual Code Change Process, 100 ft View

- Make local changes
- Convert changes into human-readable form
- Send it to the relevant people
- Discuss about the proposal with the people
- Repeat until accept/reject
- Straightforward



# Making Change Request, 10 ft View

- Standardized tool ('git') and resources available
  - Make local changes
    - 'git commit'
  - Convert changes into human-readable form
    - 'git format-patch' or 'git request-pull'
- Make local changes
  - Convert changes into human-readable form
  - Send it to the relevant people
  - Discuss about the proposal with the people
  - Repeat until accept/reject
  - Straightforward

# Sending Change Request, 10 ft View

- Send it as a plain text email
- Standardized tool ('git') makes this semi-automated and doable
- Finding relevant people
  - Maintainers, reviewers, and mailing list of the subsystem
  - MAINTAINERS file and `get_maintainer.pl` can be used
- Sending the patch or pull request
  - `'git send-email'`
- Make local changes
- Convert changes into human-readable form
- Send it to the relevant people
- Discuss about the proposal with the people
- Repeat until accept/reject
- Straightforward

# Change Request Discussion, 10 ft View

- How to reply?
  - Bring your own mailing tool  
(Note: Gmail is sub-optimum)
- How to read others' mails?
  - Subscribe to mailing list, or
  - Search mailing list archives
- How to convince others?
  - Out of the scope of this talk
- Make local changes
- Convert changes into human-readable form
- Send it to the relevant people
- Discuss about the proposal with the people
- Repeat until accept/reject
- Straightforward

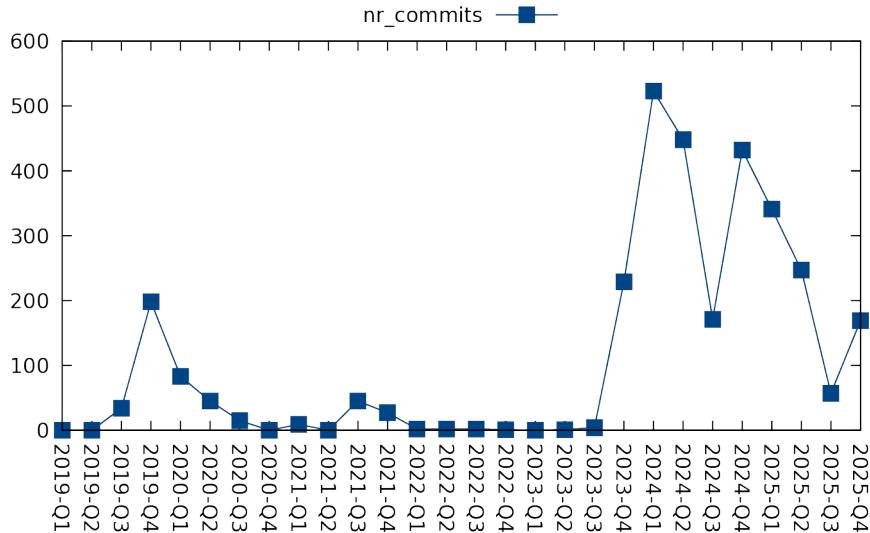
# Change Request Discussion Pain Points

- Finding proper email client
  - There are more than two ways to do that
    - Kernel official documentation introduces 15 tools
- Reading other's email
  - Subscribing doesn't work for busy subsystems (hundreds of mails per day)
  - Public archives (lore.kernel.org) lack comfy of private inbox
    - No reply button, difficult personalization
- Pain points for some people including beginners

# hkml: Hackers' Mailing Tool

# Evolution Story of hkml

- 2019: Public-inbox based hack
  - Supported only listing and reading mails
  - Only for scratching the developer's itch
- 2021: DAMON maintenance mailing tool
  - Extended for formatting and sending mails
  - Still a hack for personal usages
  - Was unable to recommend to others
- 2024: Linux kernel developers mailing tool
  - Extended for mbox support, interactive UI, and more
  - Officially **committed** to support any Linux kernel developers including DAMON community



# hkml: Mailing tool for Mails-driven development

- Design goals: minimum setup and resources
- Support reading public-inbox archives and mbox files
- Highly optimized for Linux kernel development
- Known users: DAMON maintainer and a few cool folks
- Available at  
<https://github.com/sjp38/hackermail>



This QR code is generated from  
<https://www.qr-code-generator.com>

# hkml Demo Time

# Setup

- `git clone https://github.com/sjp38/hackermail`
- That's it!

# Reading Mails from Mailing Lists

- `hkml list <list name, e.g., damon>`
- Show the mails of the list with an interactive UI
- The interactive UI supports most works
  - Replying, Finding patches to review, etc
  - Press ‘?’ for shortcut keys
  - Press ‘m’ for menus

# Reading Mails of a Thread on Mailing List

- `hkml list <msgid or lore.kernel.org link>`
  - e.g.,

```
'html list \
https://lore.kernel.org/20251125015841.76180-1-sj@kernel.org', or
'hkml list 20251125015841.76180-1-sj@kernel.org'
```

# Reading Mails Not Sent to Mailing Lists

- Save the mail[s] as an mbox file
  - Modern email clients including Gmail support this
- `hkml list <mbox file>`

# Tagging Mails

- From the list, open menu (press ‘m’) → ‘manage tags’
- To read mails of a tag,
  - `hkml list <tag name>`

# Replying to Mails

- From the list, open menu (press ‘m’) → ‘reply’
  - ‘hkml’ will open a text editor for writing/sending the reply
  - The mail will be tagged as ‘drafts’ or ‘sent’ depend on your following action
  - ‘hkml’ find previous draft and let you continue writing it

# Writing and Sending a Mail

- ‘hkml write’ from the terminal
  - Works similar to replying feature

# Downloading Patches

- From the list, open menu → ‘handle as patches’
  - → ‘export patch[es]’
- Save the patches on the user-specified path
- Collect {Reviewed,Acked,Tested}-by: tags in replies
- Merge cover letter into first patch’s commit message (mm style) if requested

# Testing Patches

- From the list, open menu → ‘handle as patches’
  - → ‘check patch[es]’
  - Download the patches and run checkpatch.pl

# Applying Patches

- From the list, open menu → ‘handle as patches’
  - → ‘apply patch[es]’
- Download the patches and apply on the current tree
- Make a merge commit having cover letter’s description as its commit message, if requested

# Formatting and Sending Patches

- From terminal, ‘hkml patch format <commits>’
  - Setup CV with commit message of
    - first commit’s parent, or merge commit of given commits
  - Add recipients based on `get_maintainer.pl`
    - Only coverletter gets all recipients
  - Run `checkpatch.pl` for each patch
  - Give user a moment to review subjects and reviewers
  - Finally send the patches
  - User can abort the process at each step

# hkml Advanced Features Demo (Depending on time, this might be skipped)

# Remote tags synchronization

- `hkml sync --remote <your private git repo>`

# Mails Monitoring

- `hkml monitor add . . .`
- `hkml monitor start`
  - Periodically run ‘`hkml list`’ with specified filter/decoration options
    - Only new mails are git-fetched; no overhead to `lore.kernel.org`
  - Send the output as an email to user

# And more hidden features

- Flexible mails filtering
- Flexible mails display options
- Refer to USAGE.md
- Suggest/contribute your features

# QnA

- Feel free to use
  - [sj@kernel.org](mailto:sj@kernel.org)
  - <https://github.com/sjp38/hackermail/issues>



This QR code is generated from  
<https://www.qr-code-generator.com>