CE Workgroup

Status of Embedded Linux

October 2016

Tim Bird

Architecture Group Chair

LF Core Embedded Linux Project

Nature of this talk…

- Quick overview of lots of embedded topics
- A springboard for further research
  - If you hear something interesting, you have something to search for
- Sorry… this is going to be *very fast*
  - I know your brain is already full
  - Slides available after the event

Nature of this talk…

- Quick overview of lots of embedded topics
- A springboard for further research
  - If you hear something interesting, you have something to search for
- Sorry… this is going to be *very fast*
  - I know your brain is already full
  - Slides available after the event

Outline

Kernel Versions
Technology Areas
CE Workgroup Projects
Other Stuff
Resources

Outline

Kernel Versions

Technology Areas

CE Workgroup Projects

Other Stuff

Resources

Kernel Versions

- Linux v4.3 – 1 Nov 2015 – 63 days
- Linux v4.4 – 10 Jan 2016 – 70 days
- Linux v4.5 – 13 Mar 2016 – 63 days
- Linux v4.6 – 15 May 2016 – 63 days
- Linux v4.7 – 24 July 2016 – 70 days
- Linux v4.8 – 2 Oct 2016 – 70 days
  - My prediction was off by 7 days
- Greg KH already announced 4.9 as next LTS
  - Predicted on December 4 or 11

Linux v4.4

- LightNVM feature
  - Take control of low-level SSD features
    - Will talk about this later
- Perf can build and load eBPF files
- Arm64 can have 16K pages
- Broadcom VC4 GPU (raspberry pi)
- Devfreq cooling – thermal management
- Various PWM drivers

Linux v4.5

- ARM multiplatform hits an important milestone
  - Major patch including lots of minor platforms
  - Many v6 and v7 platforms are now supported
  - What's the big deal?
    - Can now build a single ARM kernel image that works on multiple platforms (or platform configurations)
    - Opposite of embedded
- Not much else specific to embedded
  - Well, continued mainlining of drivers for SoC features

Linux 4.6

- GPIO subsystem rework
- scripts/dtc/dtx_diff
  - Compare device trees in a number of formats
- Improved page-poisoning
  - Separate from debug, can set poison value to 0 (to clear pages after free for security reasons)

Linux 4.7

- Schedutil frequency governor
  - Use the load calculated by the scheduler instead of the average load over past little while
  - See http://lwn.net/Articles/682391/
- VFS layer can iterate through directories in parallel
- Ability to attach BPF programs to tracepoints
- Ftrace histogram triggers
  - Can tell tracer to accumulate events into buckets and give results, via the sysfs interface
- Android sync_file feature moved from staging

Sync_file

- Allows for explicit fencing for buffers by userspace
- How it works:
  - Producer driver sends the fence related to the buffer to userspace via a sync_file
  - An intermediary (e.g. a compositor) passes these fenced fds to DRM in an atomic commit
  - Consumer will not use the buffer for anything before the fence(s) signals
- This avoids a lot of waiting
- See Documentation/sync_file.txt

Linux 4.8

- New kernel documentation system
- New pseudo-random number generator
  - See https://lwn.net/Articles/686033/
- ARM64 support for kexec and kprobes
- New timer wheel implementation
  - https://lwn.net/Articles/646950/
  - Better performance:
    - No more cascade operations
    - Quick determination of next timout
  - Long timeouts have reduced resolution
  - Automatically coalesces longer timeouts

Linux 4.9 (predicted)

- Virtually mapped kernel stacks
  - http://lwn.net/Articles/692953/
  - Allows to detect stack overruns
  - Cleans up kernel code
  - Faster process creation
  - on x86, for now

Observations

- Embedded-specific, non-driver features:
  - Nothing lately for:
    - Boot-up time
    - System size
    - Embedded filesystems (are these done now?)
    - Embedded security
  - Work progressing on:
    - Power management
    - Some real-time stuff (e.g. timer wheel stuff)
    - Solid state storage
    - GPU drivers
- There is lots of processor support, and lots of device drivers for embedded

Things to watch (from past)

- Kernel tinification!
- RT-preempt
- Persistent memory
  - (NVM = Non-Volatile Memory)
- SoC mainlining progress

Things to watch (status)

- Kernel tinification! (stalled)
- RT-preempt (only 10K lines left?!)
- Persistent memory (in progress)
  - Good talk about issues:
    - “Making use of persistent memory”
      - http://lwn.net/Articles/674752/
- SoC mainlining progress (slow progress)

Outline

Kernel Versions

Technology Areas

CE Workgroup Projects

Other Stuff

Resources

Bootup Time

- Mostly old news…
- XIP on x86
  - See https://lwn.net/Articles/637532/
- Asynchronous probing
  - Discussed at last kernel summit
- Reduction in probe deferral
  - Explicit probe ordering can be used to get a specific subsystem (like display) up sooner
  - The “On-demand probing” patches were NAKed
  - Need to measure effect on overall boot time

Bootup Time (cont.)

- No talks at ELC or ELCE this year
  - But boot time is NOT a solved problem
  - Boot time issues are unique per platform, and reductions tend not to be mainlinable
    - e.g. remove stuff not needed
- Some good previous talks:
  - ELCE 2014 - *12 Lessons Learnt in Boot Time Reduction* by Andrew Murray
  - ELC 2015 - *Fastboot Tools and Techniques* by John Mehaffey
- Lots of problems persist in user space
  - I’m looking at you, Android

Device Tree

- Device Tree Overlays
  - Seems to be working as intended
  - Session at ELC 2016 by Pantellis on making overlays independent of the base board
    - Should allow add-on boards to be used with different platforms
- Device Tree validation
  - Schema for binding language, validator for bindings and for device tree data
  - This work stalled
- Updated Device Tree specification has been discussed
  - Want to update material and make it more available

Graphics

- Vulkan API from Khronos Group
  - Alternative to Direct3D or OpenGL
  - Reduce CPU overhead for CPU/GPU operations
  - AMD announced plans to open source the driver (but Intel and Valve already working it)
  - Version 1.0 is now available
  - Nvidia now supports it
- Qt license change
  - From LGPL 2.0 to LGPL 3.0
  - Companies scrambling to find alternative
    - GPL/LGPL 3.0 is undesirable for CE products

GPUs and OSS support

- Integrated GPUs
  - AMD, Intel, Nvidea, Qualcomm:Adreno
- GPU IP suppliers
  - ARM:Mali, Imagination:PowerVR, Vivante
- GPU support
  - Freedreno – Adreno (good progress)
  - ??? – for PowerVR (no progress)
  - Etnaviv – for Vivante (good progress)
  - Nouveau – for Nvidia (not sure of status)
  - Lima – for Mali (no progress)

File Systems

- Proposals for UBIFS handling of MLC NAND
  - Lots of complexity due to MLC characteristics
  - See “NAND Support: (New?) Challenges for the MTD/NAND Subsystem” – Boris Brezillon (at ELC)
- Rise of black-box, block-based storage
  - A lot of embedded has switched to eMMC, so is using regular block storage instead of raw NAND
  - IO scheduling for solid state storage
  - LightNVM (next page)

(new) LightNVM

- Framework for holding SSD parameters
- Allows kernel to manage flash translation layer
- SSDs have weird (black-box) FTL implementations
  - Are often optimized for FAT filesystems
  - Recent drives allow direct access to blocks
- See http://lwn.net/Articles/641247/
  - “The host primarily handles data placement, I/O scheduling, and garbage collection and leaves everything else to the SSD controller”

Networking

- Bluetooth:
  - Bluetooth 4.2 has better security, faster speeds
  - 6lowpan integration
  - Working on mesh networking
- New protocols for IOT
  - Thread – Nest's low-power IP stack
  - Others (Sigfox, LoRaWan, etc.)

Real Time – RT-preempt

- Linux Foundation Real-Time Linux Collaborative project
  - Thomas Gleixner is a Linux Foundation fellow
  - Executive Summary: More stuff going upstream
  - RT Summit here this week
- Latest RT-preempt is for 4.8 kernel
  - Tends to follow LTS releases
  - See https://www.kernel.org/pub/linux/kernel/projects/rt/

Real Time - other

- Xenomai 3.0.1
  - Uses Cobalt RT core
  - Supports both dual-kernel and single-kernel configurations (using RT-preempt)
  - See xenomai.org
- Some RT talks
  - ELCE 2015 – Practical Real-Time Linux – by Arnout Vandecappele
  - Presentation on Xenomai at ELC 2016

Security

- "Making your own security modules" – Casey Schaufler
  - http://lwn.net/Articles/674949/
  - Promote experimentation by giving tips on how to write your own security modules
- New project for kernel security hardening:
  - http://kernsec.org/wiki/index.php/Kernel_Self_Protection_Project
  - Want to address classes of problems, instead of individual bugfixes

System Size

- Kernel tinification project appears stalled
  - Tiny repository removed from linux-next
  - No activity in one year!
- Single-user patches
  - Gets rid of users and groups
  - Saves about 25K
  - http://lwn.net/Articles/631853/
  - Mainlined in kernel v4.1
- Removal of kernel command-line parsing
  - Not mainlined

System Size (cont.)

- Intel X86 XIP patches
  - See https://lwn.net/Articles/637532/
- Nicolas Pitre has done work recently on support for gcc --gc-sections
  - Lighter-weight option similar to LTO
  - See his talk from Linaro Connect
    - http://connect.linaro.org/resource/sfo15/sfo15-bfo2-reducing-the-arm-linux-kernel-size-without-losing-your-mind/
- Vitaly Wool also doing stuff

Testing

- Kselftest
- Fuego - LTSI Test Project
- Kernelci.org
- LAVA V2
- Lots of automated testing talks at ELC and ELCE 2016

kselftest

- Inside kernel source tree
  - Makefile target: ‘make kselftest’
- Ability to install tests mainlined in kernel v4.1
  - Cross-build now supported
    - I didn’t have time to test this myself
  - http://lwn.net/Articles/628625/
- See “Linux Kernel Selftest Framework BoFs – Quality Control for New Releases” – Shuah Khan (at ELC)
- See http://lwn.net/Articles/608959/

Fuego - LTSI test project

- Available now
  - https://bitbucket.org/tbird20d/fuego/
- Big focus on documentation (wiki)
  - http://bird.org/fuego
- Working on lots of issues:
  - Command line tool
  - Device dictionary
  - Test packaging
  - Easier toolchain integration

Kernelci.org

- Place to get free build/boot testing for your board
  - “ci” = continuous integration
  - Builds 126 trees continuously, then reports any errors
- http://kernelci.org
- ELC and ELCE 2016 – by Kevin Hilman

- The most successful public, distributed build and test system for Linux, in the world!

Toolchains

- Khem Raj added support to the Yocto Project for Clang (LLVM)
  - Builds all but about 45 packages
  - He has a mini-distro with kernel, musl, toybox, built with clang
- Presentations on Clang at ELC and ELCE 2016

Tracing

- eBPF to be used for dynamic tracing
  - Perf supports eBPF (in 4.4)
    - eBPF = extended Berkeley Packet Filter
- New tracefs filesystem
  - No longer part of debugfs
  - But all (psuedo) dirs and files the same
- Histograms (not mainlined yet?)

Miscellaneous

- Next LTS kernel version:
  - 4.9
  - This is the earliest it's been announced!
    - We haven't even opened the 4.9 merge window
- Non-Linux announcements

Lots of non-Linux in IOT

- Magenta – RTOS by Google
  - Fuchsia OS - Some attributes of Android
  - Based on LK
  - BSD license
- Zephyr – RTOS from Wind River
  - Apache 2 license
  - Minimal size – as small as 8K
    - Highly configurable
  - NoMMU
  - Networking: WiFi, Bluetooth, NFC
- Linaro recently announced support for Zephyr !!

Outline

Kernel Versions

Technology Areas

CE Workgroup Projects

Other Stuff

Resources

Projects and initiatives

- Shared Embedded Distribution
- Device Mainlining
- LTSI
- Fuego
- eLinux wiki

Shared Embedded Distribution

- Goals
  - Create an industry-supported distribution of embedded Linux
    - Main goal is very long term support (15 years)
- Status
  - Toshiba has created Yocto layer meta-Debian
  - Presented at ELCE, ELC, and LCJ
- Next steps
  - Get more companies collaborating on the project

Device Mainlining

- Goal is to study obstacles to mainlining, and work to reduce obstacles
- Projects:
  - Mobile phone source analysis
    - Phone kernels have between 1.1 and 3.1 million lines of code out-of-tree
  - Upstream analysis tools:
    - https://github.com/tbird20d/upstream-analysis-tools
  - Patch submission tool
    - Make it easier for people with deficient e-mail clients (ie corporate-mandated Outlook)
- See http://elinux.org/CE_Workgroup_Device_Mainlining_Project

Long Term Support Initiative

- LTSI 4.1 is latest kernel
- Many presentations available on status
- Latest project push is testing facility
  - See previous page on Fuego test framework
- Kernel divergence measurement tool
  - Hisao Munakata Presentation at ELC 2016

eLinux wiki

- http://elinux.org
  - Web site dedicated to information for embedded Linux developers
    - The wikipedia of embedded linux!
- Hundreds of pages covering numerous topic areas: bootup time, realtime, security, power management, flash filesystem, toolchain, editors
- Lots of pages in last few years about low-cost development boards
- Please use and add to site

Outline

Kernel Versions
Technology Areas
CE Workgroup Projects
Other Stuff
Resources

Resources

- LWN.net
  - http://lwn.net/
  - If you are not subscribed, please do so
- Kernel Newbies
  - http://kernelnewbies.org/Linux_[34].?
- eLinux wiki - http://elinux.org/
  - Especially http://elinux.org/Events for slides
- Celinux-dev mailing list

Outline

Kernel Versions
Technology Areas
CE Workgroup Projects
Other Stuff
Resources

Special: High-level Status Review

High Level Status Review

- Trends from the last few years
- Generalization vs. specialization
- Fragmentation…

First – overall status

- We're doing great
- Linux in over 1.5 billion objects
- Next little rant is about a few things…
- It's looking like Linux won't be on those 9 billion leaf nodes in the IOT
  - It'll be in the gateway, but many of the individual sensor nodes will be something else
- Most disturbingly – it may not be the first OS running on a cereal box

Cereal box Linux

Likely first application…

What's needed

- $1.10 cost of goods
  - .40 CPU, flash, memory
  - .50 display
  - .05 input device
  - .15 battery
- Where are we?
  - $5.00 CPU, flash, memory
  - Others?

Trends from the last few years

- Decrease in the number of talks on a few key topics
- Slowdown in kernel submissions in some areas
  - Slowdown of (apparent) progress in some areas
- OSS RTOS fragmentation
  - Rise of permissive licenses (non-GPL)

Traditional embedded Linux topics

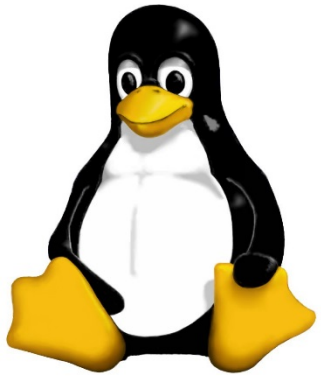
- Boot time
- System size
- File systems
- Power management
- Real time
- Security

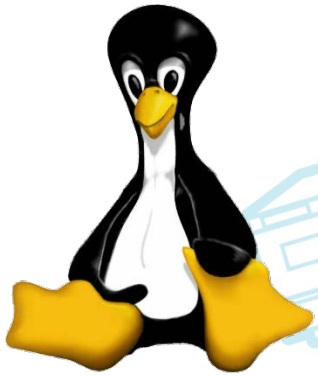
Trend 1 – slowdown in progress

- Boot time – can get fast boot times, but nothing is submitted upstream
  - Part of problem is that many boot time reductions are specializations of the kernel, which are rejected upstream
    - Results are not repeatable
  - Really just bloat by another name
- System Size – exact same thing
  - Requires difficult manual effort

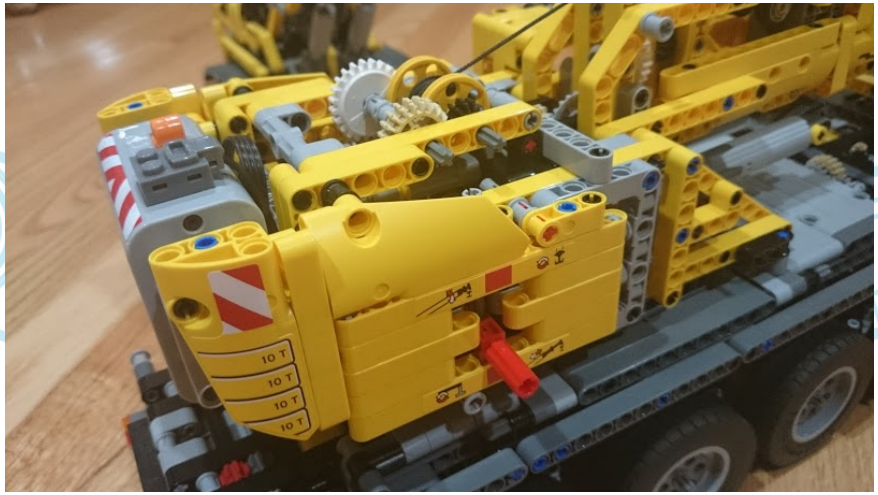
Generalization vs. specialization

- Upstream wants generalization
- Reducing boot time means taking stuff out
  - This is a specialization of Linux
- Andrew Murray's presentation on boot time
  - From ELC 2014
- Subtractive vs. Additive engineering





Subtractive vs. Additive Engineering



Subtractive Engineering

Linux

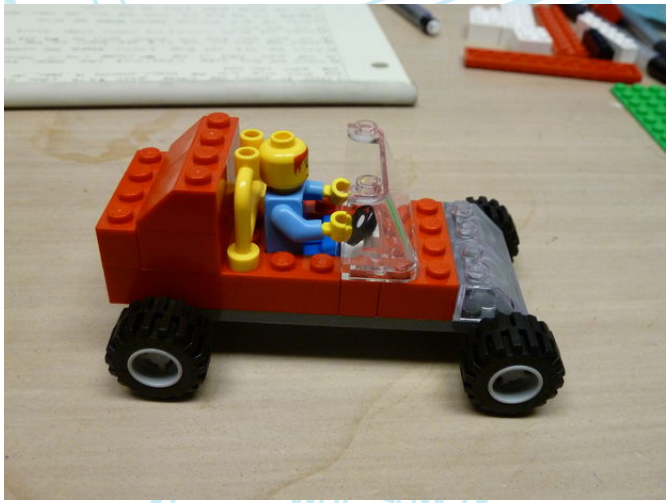
Franken-Linux

Resistance to specialization

- Device tree is moving the kernel towards more generalization
  - Drivers written to handle all possible IP block configurations, across multiple CPUS
  - Used to be handled with config options (compile-time)
- Patches for some system specializations will not be accepted upstream
  - E.g. network stack
    - Don’t want functionality regressions
  - Don’t want 40,000 kernel config items
    - Already have 13,000

Different Approaches

RTOS



In-house custom OS

Sony's Nuttx experience

- Project to support an audio player
- Easier to add a bunch of stuff to Nuttx than to trim down Linux
  - Sony worked on: Board support, C++ support, ELF loading customizations, openOCD support, power management, ADB (debug agent)

Trend 2 - IoT RTOS fragmentation

- In the IoT space, fragmentation is a problem:
  - Many choices (Linux, Nuttx, Free RTOS, Contiki, Zephyr, mbed, and more)
  - Different licenses
    - Can’t share code between different systems
- Fragmentation means less sharing
  - I worry about the board support not being able to be used by different communities
  - It’s hard enough to mainline work to one upstream
    - E.g. Sony’s ELF loading customizations

The paradox of embedded

- My keynote from ELC 2014
- Paradox is this:
  - You want generalized software to expand the set of people who will collaborate on it
  - Embedded wants to deeply specialize software for efficiency (cost, performance, power)
  - Specialized software loses “community effect” of shared testing and collaboration
- How do you balance this tension?

Problems that I see

- Absence of a shared embedded Linux distribution means there's no place to share non-kernel technology
  - System-wide optimizations
  - Feedback-directed optimization (FDO)
    - E.g. Function coalescing for page fault reduction on bootup
  - Security enhancements (eg Tizen stuff)
- There is no upstream for certain test stuff
  - Lots of frameworks - fragmentation
  - Test collateral is not being shared
    - Test configurations, results, board descriptions

More sharing

- We need more sharing of:
  - Distribution package work
    - Security fixes
    - Package updates
    - Integration and test
  - Test experience
    - Not just tests, but test results, board access methods and other test collateral
    - E.g. What should the result of bonnie be on a beaglebone black? What LTP test failures can I ignore?

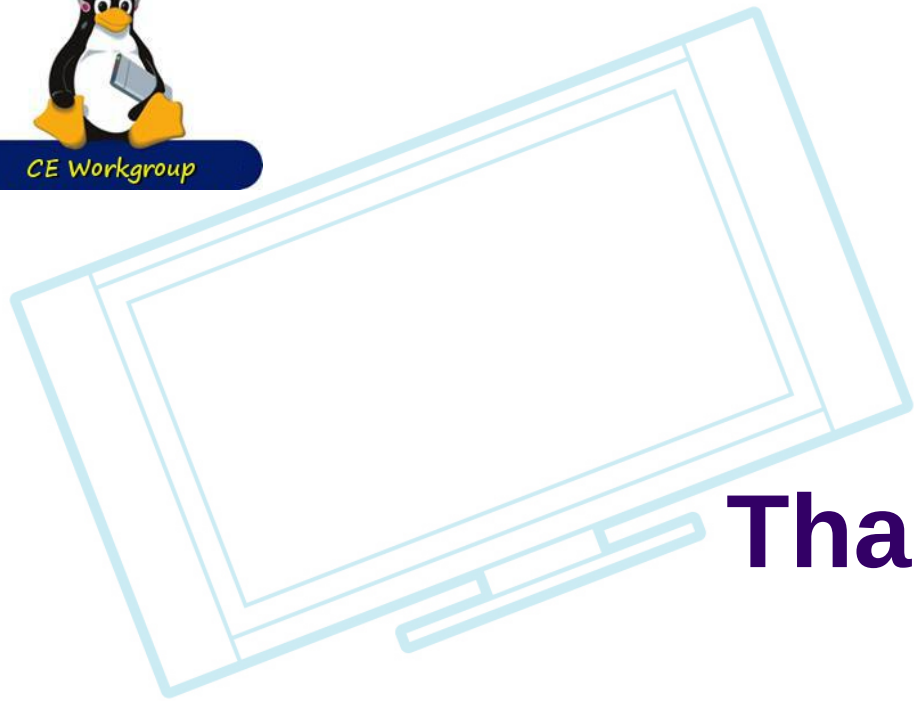
Recommendations

- Use automation to overcome reduction in community effect
  - That's why I'm personally focused on test
- Work hard to reduce fragmentation
  - Maybe dual-license code (GPL and BSD)
    - BSD code can be used by anyone
  - Look at other projects and find commonality!
- Keep working on upstreaming

Let’s keep working together to make Linux the best OS possible…

Let’s keep working together to make Linux the best OS possible…

besides that, it’s fun!



Thanks!

Ksummit-discuss 2016 issues

- Some of the most interesting discussions occur on ksummit-discuss mailing list
- A few big areas of discussion:
  - Stable workflow
  - Kernel testing
  - Git-series
  - GPL defense issues
  - Backport trees (and SoC mainlining issues)

KDSL: Stable workflow

- Some vendors don't trust the stable trees, and developers want to improve the quality of them
  - Some patches in stable caused regressions
  - Some users cherry-pick patches instead of basing product on whole tree
- Suggestions to improve quality:
  - Have more review of stable patches
    - But sub-system maintainers are already overworked
  - Have more testing of stable patches
    - This became a long thread about kernel testing
  - Identify commit that patch fixes
    - To allow for easier back-porting to multiple stable trees

KSDL: Kernel testing

- Would like to see new kernel code submitted with unit tests
  - Recommendation only – not required
- Kselftests is a good start, but needs more features
  - Examples, documentation, test interface, standard logging and reporting
- Need to declare test requirements
  - Test should return “not supported” instead of failure, if hardware is missing or requirements are not met
- Important to measure actual bugs caught

KSDL: Git-series

- Lots of discussions (in different threads) about how to use git to manage patches
- Josh Triplett introduced git-series, to maintain a set of patches in git
  - Similarities to quilt or stgit
    - Maintain patches as first-class objects, with meta-data and versioning for each patch

KSDL: GPL defense issues

- Big thread about when to enforce the GPL with lawsuit
  - See http://lwn.net/Articles/698452/
- Linus and Greg KH very reluctant to use lawsuits
  - It drives companies away instead of helping them join the community
- Busybox cited as example of legal enforcement that damaged the project
- Lots of disagreement on some issues

GPL defense issues (cont.)

- When to sue?
  - Can the threat make companies behave better?
  - If you never sue, is there any threat?
- What is objective?
  - Getting useful code (good for developers)
    - Bad actors don't have good code
  - Allowing end-users to get code for their devices (good for users)
    - End users don't have copyrights, and have no standing to sue
- Who can decide to sue?
  - Small group of copyright holders, or the SFC, or an individual?
    - Problem is if other stakeholders don't want to sue

KSDL: Backport trees (and SoC mainlining)

- Long thread about value of trees with backported features
  - LSTI and LSK (Linaro Stable Kernel)
- Some people don’t like these trees
  - Reduces incentive to work with upstream
  - Loses shared testing with upstream
- Industry reality is that backporting is inevitable
  - Mobile vendors are not on upstream due to product cycle issues and legacy out-of-tree code
- Backport trees try to find balance in quality between back-porting to in-house tree and forward-porting patches to upstream

Ksummit-discuss issues

- And much more…
  - See https://lists.linuxfoundation.org/pipermail/ksummit-discuss/
- Many topics were put onto the kernel summit agenda
- Kernel summit planned for end of October, in Santa Fe, New Mexico