Linux Plumbers Conference 2019

Report of Contributions

Contribution ID: **22** Type: **not specified**

Utilizing tools made for ”Big Data” to analyse Ftrace data - making it fast and easy

Tuesday, 10 September 2019 15:45 (45 minutes)

Tools based on low level tracing tend to generate large amounts of data, typically outputted in some kind of text or binary format. On the other hand the predefined data analysis features of those tools are often useless when it comes to solving a nontrivial or very user-specific problem. This is when the possibility to make sophisticated analysis via scripting can be extremely useful.

Fast and easy scripting inside the tracing data is possible if we take advantage of the already existing infrastructure, originally developed for the purposes of the “Big Data” and ML industries. A PoC interface for accessing Ftrace data in Python (via NumPy arrays) will be demonstrated, together with few examples of analysis scripts. Currently the prototype of the interface is implemented as an extension of KernelShark. This is a work in progress, and we hope to receive advice from experts in the field to make sure the end result works seamlessly for them.

I agree to abide by the anti-harassment policy

Primary author: KARADZHOV, Yordan (VMware)

Presenter: KARADZHOV, Yordan (VMware)

Session Classification: LPC Refereed Track

Contribution ID: **26** Type: **not specified**

pidfds: Process file descriptors on Linux

Wednesday, 11 September 2019 12:00 (45 minutes)

Traditionally processes are identified globally via process identifiers (PIDs). Due to how pid allocation works the kernel is free to recycle PIDs once a process has been reaped. As such, PIDs do not allow another process to maintain a private, stable reference on a process. On systems under pressure it is thus possible that a PID is recycled without other (non-parent) processes being aware of it. This becomes rather problematic when (non-parent) processes are in charge of managing other processes as is the case for system managers or userspace implementations of OOM killers.

Over the last months we have been working on solving these and other problems by introducing pidfds – process file descriptors. Among other nice properties, the allow callers to maintain a private, stable reference on a process.

In this talk we will look at challenges we faced and the different approaches people pushed for. We will see what already has been implement and pushed upstream, look into various implementation details and outline what we have planned for the future.

I agree to abide by the anti-harassment policy

I confirm that I am already registered for LPC 2019

Primary author: Mr BRAUNER, Christian

Presenter: Mr BRAUNER, Christian

Session Classification: LPC Refereed Track

Contribution ID: **34** Type: **not specified**

Red Hat joins CI party, brings cookies

Monday, 9 September 2019 17:00 (45 minutes)

For the past couple of years the CKI (”cookie”) project at Red Hat has been transforming the way the company tests kernels, going from staged testing to continuous integration. We’ve been testing patches posted to internal maillists, responding with our results, and last year we started testing stable queues maintained by Greg KH, posting results to the “stable” maillist.

Now we’d like to expand our efforts to more upstream maillists, and join forces with CI systems already out there. We’ll introduce you to the way our CI works, which tests we run, our extensive park of hardware, and how we report results. We’d like to hear what you need from a CI system, and how we can improve. We’d like to invite you to cooperation, both long-term, and right there, at a hackfest organized during the conference.

Naturally, real cookies will make an appearance.

I agree to abide by the anti-harassment policy

Yes

Primary authors: KONDRASHOV, Nikolai (Red Hat); KABATOVA, Veronika (Red Hat)

Presenters: KONDRASHOV, Nikolai (Red Hat); KABATOVA, Veronika (Red Hat)

Session Classification: LPC Refereed Track

Contribution ID: **38** Type: **not specified**

FPGAs 101: A Software Engineer's Adventure into Hardware Development

FPGAs are becoming more pervasive because they've gone down in price, and process improvements allow substantial designs to fit on commoditized hardware. Furthermore, processors are shipping with embedded FPGAs, making it an interesting target for scaled deployments and hobbyists alike. It's likely that in the foreseeable future, many platforms you use daily will have an FPGA embedded. As such, it's important for open source, and in particular, the Linux community at large to really start to treat these platforms like all other readily available hardware.

As a professional software developer for the last 14 years, I found the toolchain to be surprisingly complex, and the design process extremely obtuse. This talk is aimed at people who want to learn more about the process and possibly contribute to making an open source toolchain that can work on a variety of platforms.

The talk will start with some basic information about what an FPGA is, and an overview of the tooling pipeline as it exists today. Following that, we will explore what's currently available for hackers who want to begin exploring designs on FPGAs. I will go on to discuss my experience with proprietary tools vs. open tools and attempt to provide a quantitative comparison of what's currently available in open source with the proprietary tools provided by vendors. I will then go on to detail the current challenges, and what will be needed to have a robust FPGA ecosystem available within Linux.

As the title states, by the end of the talk, you should have a fairly decent 101 level understanding of FPGA development on Linux.

I agree to abide by the anti-harassment policy

Yes

Primary author: WIDAWSKY, Ben

Presenter: WIDAWSKY, Ben

Session Classification: LPC Refereed Track

Contribution ID: **39** Type: **not specified**

Linux kernel fastboot on the way

Monday, 9 September 2019 15:45 (45 minutes)

Linux kernel fastboot is critical for all kinds of platforms: from embedded/smartphone to desktop/cloud, and it has been hugely improved over years. But, is it all done? Not yet!

This topic will first share the optimizations done for our platform, which cut the kernel (inside a VM) bootime from 3000ms to 300ms, and then list the future potential optimization points.

Here are our optimizations:
1. really enable device drivers' asynchronous probing, like i915 to improve boot parallelization
2. deferred memory init leveraging memory hotplug feature
3. Optimize rootfs mounting (including storage driver and mounting)
4. kernel modules and configs optimization
5. reduce the hypervisor cost
6. tools for profiling/analyzing

Potential optimizations spots for future, which needs discussion and collaboration from the whole community:
1. how to make maximal use of multi-core and effectively distribute boot tasks to each core
2. smp init for each CPU core costs about 8ms, a big burden for large systems
3. force highest cpufreq as early as possible (kernel decompress time)
4. devices enumeration for firmware (like ACPI) set to be parallel
5. in-kernel deferred memory init (for 4GB+ platform)
6. user space optimization like systemd

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr TANG, Feng

Presenter: Mr TANG, Feng

Session Classification: LPC Refereed Track

Contribution ID: **42** Type: **not specified**

Interrupt Message Store: A scalable interrupt mechanism for the cloud

Tuesday, 10 September 2019 15:45 (45 minutes)

With virtualization being the key to the success of cloud computing, Intel's
introduction of the Scalable IO Virtualization (SIOV) aims to further the cause by shifting the creation of assignable virtual devices from hardware to a more software assisted approach. Using SIOV, a device resource can be mapped directly to guest or other user space drivers for near native DMA (Direct Memory Access) performance. This flexible composition of direct assignable devices a.k.a. Assignable Device Interfaces (ADIs) is device specific and light weight, thus making them highly scalable. SIOV enables simpler device designs by unchaining hardware from costly PCI standard and can help address limitations associated with direct device assignment.

Until now, message signaled interrupts (MSI and MSI-X) were the de facto standard for device interrupt mechanism and could support up to 2048 interrupts per device. But now with SIOV, there is a need to support a large number of ADIs (>2048), through a matching scalable interrupt management mechanism to service these ADIs.

Interrupt message storage (IMS) is conceived as a scalable albeit device specific interrupt mechanism to meet such a demand. It allows non-PCI standard storage and enumeration of MSI address/data pair to reduce hardware overhead and achieve scalability. The size, location, and storage format for IMS is device-specific; some devices may implement IMS as on-device storage, while other devices may implement IMS in host memory.

Also, one of the limitations with the current Linux MSI-X code is that PCIe device MSI-x enablement and allocation is static. i.e. device driver gets only one chance to enable MSI-X vectors, usually during probe. With IMS, we aim to make the vector negotiation with the OS dynamic, deferring vector allocation to post probe phase, where actual demand information is available.

Through this presentation, the audience can view the internals of the complex
and ever evolving Linux interrupt subsystem and understand how IMS can fit into the maze of interrupt domains, chips, remapping etc. Also, an initial IMS Linux implementation will be presented with highlights on some of the unique implementation challenges. We will conclude by demonstrating a test case using the SIOV enabled device as an example for a complete view of IMS in a scalable virtualization environment.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: DEY, Megha

Presenter: DEY, Megha

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **44** Type: **not specified**

Core scheduling

Monday, 9 September 2019 15:00 (45 minutes)

There have been two different approaches proposposed on the LKML over the past year on core scheduling. One was the coscheduling approach by Jan Schönherr, originally posted at https://lkml.org/lkml/2018/9/7/1521 and the next version posted at https://lkml.org/lkml/2018/10/19/859

Upstream chose a different route and decided to modify CFS, and only do "core-scheduling". Vineeth picked up the patches from Peter Zijlstra. This is a discussion on how we can further that work, especially when there are security implications such as L1TF and MDS, which make important this work to go upstream.

Aubrey Li will talk about Core scheduling: Fixing when fast instructions go slow

Keeping system utilization high is important both to keep costs down and to keep energy efficiency up. That often means tightly packing compute jobs and using the latest processor features. However, these approaches can be at odds when a new processor feature like AVX512 is used. The performance of latency critical jobs can be reduced by 10% if co-located with deep learning training jobs. These jobs use AVX512 instructions to accelerate wide vector operations. Whenever a core executes AVX512 instructions, the core automatically reduces its frequency. This can lead to a significant overall performance loss for a non-AVX512 job on the same core. In this presentation, we will discuss how to preserve performance while still allowing AVX512-based acceleration.

AVX512 task detection
- From user space, PMU events can be used but it's expensive.
- In the kernel, I proposed to expose process AVX512 usage elapsed time as a heuristic hint.
- Discuss an interface for tasks in cgroup.

AVX512 task isolation
- Discuss kernel space solution, if the recent proposal core scheduling can be leveraged for isolation.
- Discuss user space solution, if user space job scheduler is better than kernel scheduler

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: Mr LI, Aubrey; SCHÖNHERR, Jan; REIS, Hugo; REMANAN PILLAI, Vineeth

Presenters: Mr LI, Aubrey; SCHÖNHERR, Jan; REIS, Hugo; REMANAN PILLAI, Vineeth

Session Classification: Scheduler MC

Contribution ID: **55** Type: **not specified**

What does remote attestation buy you?

Monday, 9 September 2019 15:00 (45 minutes)

TPM remote attestation (a mechanism allowing remote sites to ask a computer to prove what software it booted) was an object of fear in the open source community in the 2000s, a potential existential threat to Linux's ability to interact with the free internet. These concerns have largely not been realised, and now there's increasing interest in ways we can use remote attestation to improve security while avoiding privacy concerns or attacks on user freedom.

More modern uses of remote attestation include simplifying deployment of machines to remote locations, easy recovery of systems with nothing more than a network connection, automatic issuance of machine identity tokens, trust-based access control to sensitive resources and more. We've released a full implementation, so this presentation will discuss how it can be tied in to various layers of the Linux stack in ways that give us new functionality without sacrificing security or freedom.

I agree to abide by the anti-harassment policy

Yes

Primary author: GARRETT, Matthew (Google)

Presenter: GARRETT, Matthew (Google)

Session Classification: LPC Refereed Track

Contribution ID: **58** Type: **not specified**

Kernel Address Space Isolation

Tuesday, 10 September 2019 12:45 (45 minutes)

Recent vulnerabilities like L1 Terminal Fault (L1TF) and Microarchitectural Data Sampling (MDS) have shown that the cpu hyper-threading architecture is very prone to leaking data with speculative execution attacks.

Address space separation is a proven technology to prevent side channel vulnerabilities when speculative execution attacks are used. It has, in particular, been successfully used to fix the Meltdown vulnerability with the implementation of Kernel Page Table Isolation (KPTI).

Kernel Address Space Isolation aims to use address spaces to isolate some parts of the kernel to prevent leaking sensitive data under speculative execution attacks.

A particularly good example is KVM. When running KVM, a guest VM can use speculative execution attacks to leak data from the sibling hyper-thread, thus potentially accessing data from the host kernel, from the hypervisor or from another VM, as soon as they run on the same hyper-thread.

If KVM can be run in an address space containing no sensitive data, and separated from the full kernel address space, then KVM would be immune from leaking secrets no matter on which cpu it is running, and no matter what is running on the sibling hyper-threads.

A first proposal to implement KVM Address Space Isolation has recently been submitted and got some good feedback and discussions:

https://lkml.org/lkml/2019/5/13/515

This presentation would show progress and challenges faced while implementing KVM Address Space Isolation. It also looks forward to discuss the possibility to have a more generic kernel address space isolation framework (not limited to KVM), and how it can be interfaced with the current memory management subsystem in particular.

MERGED with:

Address space isolation has been used to protect the kernel from the userspace and userspace programs from each other since the invention of the virtual memory.

Assuming that kernel bugs and therefore vulnerabilities are inevitable it might be worth isolating parts of the kernel to minimize damage that these vulnerabilities can cause.

Recently we've implemented a proof-of-concept for "system call isolation (SCI)" mechanism that allows running a system call with significantly reduced page tables. In our model, the accesses to a significant part of the kernel memory generate page faults, thus giving the "core kernel" an opportunity to inspect the access and refuse it on a pre-defined policy.

Our first target for the system call isolation was an attempt to prevent ROP gadget execution [1], and despite its weakness it makes a ROP attack harder to execute and as a nice side effect SCI can be used as Spectre mitigation.

Another topic of interest is a marriage between namespaces and address spaces. For instance, the kernel objects that belong to a particular

network namespace can be considered as private data and they should not be mapped in other network namespaces.

This data separation greatly reduces the ability of a tenant in one namespace to exfiltrate data from a tenant in a different namespace via a kernel exploit because the data is no longer mapped in the global shared kernel address space.

We believe it would be helpful to discuss the general idea of address space isolation inside the kernel, both from the technical aspect of how it can be achieved simply and efficiently and from the isolation aspect of what actual security guarantees it usefully provides.

[1] https://lore.kernel.org/lkml/1556228754-12996-1-git-send-email-rppt@linux.ibm.com/

I agree to abide by the anti-harassment policy

Yes

Primary author: CHARTRE, Alexandre (Oracle)

Presenters: CHARTRE, Alexandre (Oracle); BOTTOMLEY, James (IBM); RAPOPORT, Mike (IBM); NIDER, Joel (IBM Research)

Session Classification: LPC Refereed Track

Contribution ID: **59** Type: **not specified**

Decoupling ZRAM from a specific backend

Wednesday, 11 September 2019 15:00 (45 minutes)

ZRAM is a compressed RAM based block device implementation which has gotten a lot of use recently primarily in the Android world. ZRAM consists of the block device front-end, compressor back-end and memory allocator back-end. Compressor back-end is accessed via a common API, and therefore it is easy with ZRAM to select the particular compression algorithm that fits your special purpose. As opposed to
that, selecting a memory allocator back-end for ZRAM is still not possible because ZRAM is using zsmalloc API directly.

With that said, zsmalloc is not the only kernel allocator for storing compressed objects. There also are zbud (up to 2 objects per page) and z3fold (up to 3 objects per page). Designed to store only integral number of objects per page, these two have deterministic behavior with low I/O latencies. Compression ratio suffers for these two of course – by much for zbud and not so much for z3fold.

Still z3fold might be a better choice as a backend for ZRAM when compression ratio is not as important as keeping latencies low. As a z3fold primary author I keep getting questions when it will be available for use with ZRAM, and keep answering that it has to be a result of a wider consensus. To get closer to this, apart from
zsmalloc / z3fold comparisons, this talk will describe in detail how the existing zpool API should be extended to match ZRAM requirements and whether there is a performance penalty here as this introduces a level of indirection.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: WOOL, Vitaly

Presenter: WOOL, Vitaly

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **60** Type: **not specified**

libtrace - making libraries of our tracing tools

Monday, 9 September 2019 12:22 (22 minutes)

I would like to discuss how to implement a series of libraries for all the tracing tools that are out there, and have a repository that at least points to them. From libftrace, libperf, libdtrace to liblttng and libbabletrace.

I agree to abide by the anti-harassment policy

Yes

Primary author: ROSTEDT, Steven

Presenter: ROSTEDT, Steven

Session Classification: Tracing MC

Contribution ID: **63** Type: **not specified**

Integration of PM-runtime with System-wide Power Management

Tuesday, 10 September 2019 12:00 (45 minutes)

There are two flavors of power management supported by the Linux kernel: system-wide PM based on transitions of the entire system into sleep states and working-state PM focused on controlling individual components when the system as a whole is working. PM-runtime is part of working-state PM concerned about the opportunity to put devices into low-power states when they are not in use.

Since both PM-runtime and system-wide PM act on devices in a similar way (that is, they both put devices into low-power states and possibly enable them to generate wakeup signals), optimizations related to the handling of already suspended devices can be made, at least in principle. In particular:
* It should be possible to avoid resuming devices already suspended by runtime PM during system-wide PM transitions to sleep states.
* It should be possible to leave devices suspended during system-wide PM transitions to sleep states in PM-runtime suspend while resuming the system from those states.
* It should be possible to re-use PM-runtime callbacks in device drivers for the handling of system-wide PM.

These optimizations are done by some drivers, but making them work in general turns out to be a hard problem. They are achieved in different ways by different drivers and some of them are in effect only in specific platform configurations. Moreover, there are no general guidelines or recipes that driver writers can follow in order to arrange for these optimizations to take place. In an attempt to start a discussion on approaching this problem space more consistently, I will give an overview of it, describe the solutions proposed and used so far and suggest some changes that may help to improve the situation.

I agree to abide by the anti-harassment policy

Yes

Primary author: WYSOCKI, Rafael (Intel Open Source Technology Center)

Presenter: WYSOCKI, Rafael (Intel Open Source Technology Center)

Session Classification: LPC Refereed Track

Contribution ID: **64** Type: **not specified**

Linux Gen-Z Sub-system

Wednesday, 11 September 2019 10:45 (45 minutes)

Gen-Z Linux Sub-system

Discuss design choices for a Gen-Z kernel sub-system and the challenges of supporting the Gen-Z interconnect in Linux.

Gen-Z is a fabric interconnect that connects a broad range of devices from CPUs, memory, I/O, and switches to other computers and all of their devices. It scales from two components in an enclosure to an exascale mesh. The Gen-Z consortium has over 70 member companies and the first version of the specification was published in 2018. Past history for new interconnects suggests we will see actual hardware products two years after the first specification - in 2020. We propose to add support for a Gen-Z kernel sub-system, a Gen-Z component device driver environment, and user space management applications.

A Gen-Z sub-system needs support for these Gen-Z features:

- Registration and enumeration services that are similar to existing sub-systems like PCI.
- Gen-Z Memory Management Unit (ZMMU) provides memory mapping and access to fabric addresses. The Gen-Z sub-system can provide services to track PTE entries for the two types of ZMMU's in the specification: page grid and page table based.
- Region Keys (R-Keys) - Each ZMMU page can have R-Keys used to validate page access authorization. The Gen-Z sub-system needs to provide APIs for tracking, freeing, and validating R-Keys.
- Process Address Space Identifier (PASID) - ZMMU requester and responder Page Table Entries (PTEs) contain a PASID. The Gen-Z sub-system needs to provide APIs for tracking PASIDs.
- Data mover - Transmit and receive data movers are optional elements in bridges and other Gen-Z components. The Gen-Z sub-system can provide a user space interface to a RDMA driver that uses a Gen-Z data mover. For example, a libfabric Gen-Z provider implementation can use a RDMA driver to access data mover queues.
- UUIDs - Components are identified by UUIDs. The Gen-Z sub-system provides interfaces for tracking UUIDs of local and remote components. A Gen-Z driver binds to a UUID similarly to how a PCI driver binds to a vendor/device id.
- Interrupt handling - Interrupt request packets in Gen-Z trigger local interrupts. Local components such as bridges and data movers can also be sources of interrupts.

We will discuss our proposed design for the Gen-Z sub-system illustrated in the following block diagram:

Indico rendering error

```
Could not include image: Cannot read image data. Maybe not an image file?
```

Gen-Z fabric management is global to the fabric. The operating system may not know what components on the fabric are assigned to it; the fabric manager decides which components belong to the operating system. Although user space discovery/management is unusual for Linux, it will

allow the Gen-Z sub-system to focus on the mechanism of component management rather than the policy choices a fabric manager must make.

To support user space discovery/management, the Gen-Z sub-system needs interfaces for management services:

- Fabric managers need read/write access to component control space in order to do fabric discovery and configuration. We propose using /sys files for each control structure and table.
- User space Gen-Z managers need notification of management events/interrupts from the Gen-Z fabric. We propose using poll on the bridges' device files to communicate events.
- Local management services pass fabric discovery events from user space to the kernel. Our proposed design uses generic Netlink messages for communication of these component add/remove/modify events.

We are leveraging our experience with writing Linux bridge drivers for three different Gen-Z hardware bridges in the design of the Gen-Z Linux sub-system. Most recently, we wrote the DOE Exascale PathForward project's bridge driver with data movers (https://github.com/HewlettPackard/zhpe-driver). We wrote drivers for the Gen-Z Consortium's demonstration card that supports a block device and a NIC as well as a driver for the bridge in HPE's "The Machine" that is a precursor to Gen-Z.

From our work so far, here are questions we would like feedback on:

- We intend to expose control space in /sys so that user space fabric managers can work. We ask for feedback on the proposed hierarchy and mechanisms.
- Gen-Z uses PASIDs and the sub-system could use generic PASID interfaces. Any interest in this elsewhere in the kernel?
- We have need of generic IOMMU interfaces since Gen-Z ZMMU needs to interface with the IOMMU in a platform independent way. Any interest in this elsewhere in the kernel? We saw some patch sets along these lines.
- We intend to use generic NetLink for communication between user space and the kernel. Any thoughts on that decision?
- Gen-Z maps huge address spaces from remote components, and to get good performance those mappings need huge pages. Currently, the kernel does not support this use case. We would like to discuss how best to handle these huge mappings.
- We wrote a parser for the Gen-Z specification's control structure that generates C structures with bitfields. In general, we know the Linux kernel frowns on bitfields. Are bitfields ok in this context?

I agree to abide by the anti-harassment policy

Yes

Primary authors: HULL, Jim (Hewlett Packard Enterprise); DALL, Betty (HPE); PACKARD, Keith (Hewlett Packard Enterprise)

Presenters: HULL, Jim (Hewlett Packard Enterprise); DALL, Betty (HPE); PACKARD, Keith (Hewlett Packard Enterprise)

Session Classification: LPC Refereed Track

Contribution ID: **65** Type: **not specified**

Scaling performance profiling infrastructure for data centers

Monday, 9 September 2019 12:00 (45 minutes)

Understanding Application performance and utilization characteristics is critically important for cloud-based computing infrastructure. Minor improvements in predictability and performance of tasks can result in large savings. Google runs all workloads inside containers and as such, cgroup performance monitoring is heavily utilized for profiling. We rely on two approaches built on Linux performance monitoring infrastructure to provide task, machine, and fleet performance views and trends. A sampling approach collect metrics across the machine and try to attribute it back to cgroups while a counting approach tracks when a cgroup is scheduled and maintains state per cgroup. There are number of trade-offs associated with both approaches. We will present an overview and associated use-cases for both approaches at Google.

As the servers have gotten bigger, number of cores and containers on a machine have grown significantly. With the bigger scale, interference is a bigger problem for multi-tenant machines and performance profiling becomes even more critical. However, we have hit multiple issues in scaling the underlying Linux performance monitoring infrastructure to provide fresh and accurate data for our fleet. The performance profiling has to deal with the following issues:

- **Interference**: To be tolerated by workloads, monitoring overhead should be minimal - usually below 2%, some latency-sensitive workloads are certainly even less tolerant than that. As we gain more introspection into our workloads, we end up having to use more and more events, to pinpoint certain bottlenecks. That unavoidably incurs event multiplexing as the number of core hardware counters is very limited compared to containers profiled and number of events monitored. Adding counters is not free in hardware and similarly in the kernel as more work registers must be saved and restored on context switches which can cause jitters for applications being profiled.
- **Accuracy**: Sampling at machine level reduces some of the associated costs, but attributing the counters back to containers is lossy and we see a large drop in accuracy of profiling. The attribution gets progressively worse as we move to bigger machines with large number of threads. The attribution errors severely limit the granularity of performance improvements and degradations we can measure in our fleet.
- **Kernel overheads**: Perf_events event multiplexing is a complex and expensive algorithm that is especially taxing when run in cgroup mode. As implemented, scheduling of cgroup events is bound by the number of cgroup events per-cpu and not the number of counters, unlike regular per-cpu monitoring. To get a consistent view of activity on a server, Google needs to periodically count events per-cgroup. Cgroup monitoring is preferred over per-thread monitoring because Google workloads tend to use an extensive number of threads, so that would be prohibitively expensive to use. We have explored ways to avoid these scaling issues and make event multiplexing faster.
- **User-space overheads**: The bigger the machines, the larger the volume of profiling data generated. Google relies extensively on the perf record tool to collect profiles. There are significant user-space overheads to merge the per-cpu profiles and post-process for attribution. As we look to make perf-record multi-threaded for scalability, data collection and merging becomes yet another challenge.
- **Symbolization overheads** : Perf tools rely on /proc/PID/maps to understand process mappings and to symbolize samples. The parsing and scanning of /proc/PID/maps is time-consuming with large overheads. It is also riddled with race conditions as processes are created and destroyed during parsing.

These are some of the challenges we have encountered while using perf_events and the perf tool at scale. To continue to make this infrastructure popular, it needs to adapt to new hardware and data-center realities fast now. We are planning to share our findings and optimizations followed by an open discussion on how to best solve these challenges.

I agree to abide by the anti-harassment policy

Yes

Primary authors: JNAGAL, Rohit; ERANIAN, Stephane (Google Inc); ROGERS, Ian (Google Inc)

Presenters: JNAGAL, Rohit; ERANIAN, Stephane (Google Inc); ROGERS, Ian (Google Inc)

Session Classification: LPC Refereed Track

Contribution ID: **66** Type: **not specified**

New hardware with modern I2C address conflicts

Wednesday, 11 September 2019 10:45 (45 minutes)

For some time now, special camera setups exist having features which are challenging for I2C address layouts as we know them in Linux: a) a high-speed serial link which can embed I2C communication (e.g. GMSL or FPD-Link III) and b) the ability to reprogram the client addresses of the I2C devices on the camera.

The use case for these cameras is to run multiple of them in parallel, and not just a single one. To be easily pluggable, they don't have a way to configure the I2C addresses they need. They use initially all the same I2C addresses and rely on software to reprogram them and sort out that problem.

The really tricky thing is now that they are connected to the same serial high speed link. As a result, all the clients with initially equal addresses sit (more or less, depending on the link) on the same I2C bus as well and need to be carefully reprogrammed one-by-one to a unique address.

The camera setup above is the primary example we are facing right now. Some early implementations for GMSL and FPD-Link exist with different approaches to map the I2C topology. However, there might be other hardware facing very similar problems. We definitely want to have you in the room.

An introductory talk gives a few more details of current implementations, and explains the current problems in abstracting all this. From there on, we hope to have gathered enough highly interested people for discussion, opinions, and brainstorming. The goal is, of course, to enhance the I2C core to provide reasonable support for such scenarios which will be beneficial for all users like these high speed links.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: SANG, Wolfram

Presenter: SANG, Wolfram

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **67** Type: **not specified**

Touch but don't look: Running the kernel in execute only memory

Monday, 9 September 2019 10:45 (45 minutes)

Execute only memory can protect from attacks that involve reading executable code. This feature already exists on some CPUs and is enabled for userspace.

This talk will explain how we are working on creating a virtualized "not-readable" permission bit for guest page tables for x86 and the impact to the kernel. This bit can be used to create execute-only memory for userspace programs as done on other architectures, but newly also kernel text itself. This project has a working POC, but requires extra care being taking in the kernel going forward around certain code patterns in order for the kernel to run in execute only. This will be the main "call to action" of the talk.

The talk will cover three areas:

-Benefits of execute only memory

As was covered in the talk last year by Kristen Accardi, execute only memory can protect code diversification schemes like KASLR, ASLR, and especially fined grained ASLR. This would be a brief summary and will also touch on some attacks that involve reading kernel text

-How we are implementing this across QEMU, KVM, and the guest Linux Kernel.

The solution is sort of novel and interesting it itself, but most of the talk will be about kernel impact of this feature on not the hypervisor implementation. The gist of the solution involves pretending to the guest that the CPU has one less physical address bit than it actually does, so what looks to the guest like a reserved bit looks to the CPU like a physical address bit. Our proposed new KVM APIs can allow userspace VMMs to duplicate memory such that this bit selects from differently permission-ed copies of the same guest physical memory. Intel EPT has the ability to create execute only guest physical memory, so by having the second half of the memory as execute only, we can make a bit that can mark guest virtual memory as execute only.

-Proposed APIs for using execute only memory in userspace and changes and restrictions required to the Linux kernel in order for it to map its own executable code as execute only.

Our POC required making surprisingly few changes to the Linux kernel, however there were impacts especially around features that involve modifying or mapping new executable code. Long term, however, supporting this feature fully would involve the community agreeing that going forward, code patterns that violate execute only memory would not be allowed in the kernel.

I agree to abide by the anti-harassment policy

Yes

Primary author: EDGECOMBE, Rick (Intel)

Presenter: EDGECOMBE, Rick (Intel)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **68** Type: **not specified**

Upstream Graphics: Too little, too late

Monday, 9 September 2019 15:00 (45 minutes)

DRM is merging new drivers at a brisk pace, and with lima and panfrost to support ARM Mali GPUs the last obvious gap in not yet reverse-engineered hardware is getting closed. Plus new features, more contributors, more patches - in general upstream graphics is as healthy as it's never been before.

Time for some celebratory drinks, except this talk will be none of that. Now that we've achieved the goal of supporting all things graphics in upstream, the struggles didn't disappear. The promised land of "Upstream First" is leaving a rather sour aftertaste.

This talk will go through all the ways companies and teams have tried to ship graphics drivers using upstream, and how they all go wrong.

It will, unfortunately, not present solutions.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: VETTER, Daniel (Intel)

Presenter: VETTER, Daniel (Intel)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **69** Type: **not specified**

Malloc for everyone and beyond NUMA

Wednesday, 11 September 2019 12:45 (45 minutes)

With heterogeneous computing, program's data (range of virtual addresses) have to move to different physical memory during the lifetime of an application to keep it local to compute unit (CPU, GPU, FPGA, …). NUMA have been the model used so far but it has assumptions that do not work with all the memory type we now have. This presentation will explore the various types of memory and how we can expose and use them through unified API.

I agree to abide by the anti-harassment policy

Yes

Primary author: GLISSE, Jerome (Red Hat)

Presenter: GLISSE, Jerome (Red Hat)

Session Classification: LPC Refereed Track

Contribution ID: **70** Type: **not specified**

Killing the mmap_sem's contention

Tuesday, 10 September 2019 12:00 (45 minutes)

Big systems are becoming more common these days. Having thousands of CPUs is no more a dream and some applications are attempting to spread over all these CPUs by creating threads.
This leads to contention on the mm->mmap_sem which is protecting the memory layout shared by these threads.
There were multiple attempts to get rid of the mmap_sem's contention or the mmap_sem itself, Speculative Page Fault, RangeLock, Scalable Address Spaces Using RCU Balanced Trees…
Unfortunately, these attempts didn't last enough to reach the upstream state. One the reason could be the major impact they are implying on the MM code or that they are only addressing part of the overall picture (SPF).
Last discussions at the LSF/MM summit were not leading to an agreement on a solution (see LWN coverage).
This topic is presenting one of emerging solution which didn't get the time to be proposed at the last LSF/MM. It is based on discussion some folks had at the end of the summit, trying to brainstorm a way to move to a split lock mechanism, as it was done for the PTE locking, removing the mm->page_table_lock.
Currently, this work is still in progress and some deviations on the original design are expected to happen, so kind of split lock is the current option but this may change in the meantime.
This topic is linked to the use of a Maple Tree to replace both the VMA RB tree and the VMA double linked list. Matthew Wilcox and Liam R. Howlett are working on.

I agree to abide by the anti-harassment policy

Yes

Primary authors: Mr GLISSE, Jérôme; Mr DUFOUR, Laurent

Presenters: Mr GLISSE, Jérôme; Mr DUFOUR, Laurent

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **71** Type: **not specified**

LAG and hardware offload to support RDMA and IO virtualized interfaces

Monday, 9 September 2019 15:45 (45 minutes)

Link Aggregation (LAG) is traditionally served by bonding driver. Linux bonding driver supports all LAG modes on almost any LAN drivers - in the software. However modern hardware features like SR-IOV-based virtualization and state full offloads such as RDMA are currently not well supported by this model. One of possible options to solve that is to implement LAG functionality entirely in NIC's hardware or firmware. In our presentation we present another approach, where LAG functionality for state full offloads such as RDMA and IO virtualization is implemented mostly in software, with very limited support from existing Hardware and firmware. A concept that should make the solution more generic without complicating the HW any further.

The presentation is focused on 3 areas: implementation of active-backup mode for RDMA and virtual functions, usage of RX hash value to implement flow-based active-active mode and new active-active mode for virtual functions.

Proposed implementation of the active-backup mode for RDMA is done in RDMA and LAN drivers. An application continues using direct HW support for RDMA. LAN driver (with the help of RDMA driver) observes notifications from the bonding driver and accordingly controls low-level TX scheduling and RX rules for RDMA queues. The same mechanism can be used to transparently redirect network virtual functions from active to backup. We further explore the use of RX hash to implement active-active mode.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: Mr KASHYAP, Vivek (Intel); Ms SINGHAI JAIN, Anjali (Intel); Dr UMINSKI, Piotr (Intel)

Presenters: Mr KASHYAP, Vivek (Intel); Ms SINGHAI JAIN, Anjali (Intel); Dr UMINSKI, Piotr (Intel)

Session Classification: Networking Summit Track

Contribution ID: **73** Type: **not specified**

Challenges of the RDMA subsystem

Monday, 9 September 2019 17:45 (45 minutes)

The RDMA subsystem in Linux (drivers/infiniband) is now becoming widely used and deployed outside its traditional use case of HPC. This wider deployment is creating demand for new interactions with the rest of the kernel and many of these topics are challenging.

This talk will include a brief overview of RDMA technology followed by an examination & discussion of the main areas where the subsystem has presented challenges in Linux:

Very complex user API. An overview of the current design, and some reflection on historical poor choices

The DMA from user space programming model and the challenge matching that to the DMA API in Linux

Development of user space drivers along with kernel drivers

Delegation of security decisions to HW

Interaction with file systems, DAX, and the page cache for long term DMA

Inter-operation with GPU, DMABUF, VFIO and other direct DMA subsystems

Growing breadth of networking functionality and overlap with netdev, virtio, and nvme

Fragmentation of wire protocols and resulting HW designs

Placing high performance as paramount and how this results in HW restrictions limiting the architecture and APIs of the subsystem

The advent of new general computation acceleration hardware is seeing new drivers proposed for Linux that have many similar properties to RDMA. These emerging drivers are likely to face these same challenges and can benefit from lessons learned.

RDMA has been a successful mini-conference at the last three LPC events, and this talk is intended to complement the proposed RDMA micro-conference this year. This longer more general topic is intended to engage people unfamiliar with the RDMA subsystem and the detailed topics that would be included in the RDMA track.

The main goal would be to help others in the kernel community have more background for RDMA and its role when making decisions. In part this proposal is motivated by the number of times I heard the word 'RDMA' mentioned at LSF/MM. Often as some opaque consumer of some feature.

Jason Gunthorpe is a Sr. Principal Engineer at Mellanox and has been the co-maintainer for the RDMA subsystem for the last year and a half. He has 20 years' experience working with the Linux kernel and in RDMA and InfiniBand technologies.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr GUNTHORPE, Jason (Mellanox Technologies)

Presenter: Mr GUNTHORPE, Jason (Mellanox Technologies)

Session Classification: LPC Refereed Track

Contribution ID: **74** Type: **not specified**

Reflections on kernel quality, development process and testing

Wednesday, 11 September 2019 12:00 (45 minutes)

In this talk Dmitry will highlight some of the areas for improvement related to release quality, security, and developer experience and productivity. Then try to show that the existing processes, approaches and tools poorly cope with the current scale and rate of change and don't provide adequate quality and developer experience. Lastly Dmitry will advocate that only pervasive changes to the process, tooling and testing approaches can significantly improve the situation.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: VYUKOV, Dmitry (Google)

Presenter: VYUKOV, Dmitry (Google)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **75** Type: **not specified**

Formal verification made easy (and fast)!

Tuesday, 10 September 2019 17:45 (45 minutes)

Linux is complex, and formal verification has been gaining more and more attention because independent "asserts" in the code can be ambiguous and not cover all the desired points. Formal models aim to avoid such problems of natural language, but the problem is that "formal modeling and verification" sound complex. Things have been changing.

What if I say it is possible to verify Linux behavior using a formal method?

- Yes! We already have some models; people have been talking about it, but they seem to be very specific (Memory, Real-time…).

What if I say it is possible to model many Linux subsystems, to auto-generate code from the model, to run the model on-the-fly, and that this can be as efficient as just tracing?

- No way!

Yes! It is! It is hard to believe, I know.

In this talk, the author will present a methodology based on events and state (automata), and how to model Linux' complex behaviors with small and intuitive models. Then, how to transform the model into efficient C code, that can be loaded into the kernel on-the-fly to verify Linux! Experiments have also shown that this can be as efficient as tracing (sometimes even better)!

This methodology can be applied on many the kernel subsystems, and the idea of this talk is also to discuss how to proceed towards a more formally verified Linux!

I agree to abide by the anti-harassment policy

Yes

Primary author: BRISTOT DE OLIVEIRA, Daniel (Red Hat, Inc.)

Presenter: BRISTOT DE OLIVEIRA, Daniel (Red Hat, Inc.)

Session Classification: LPC Refereed Track

Contribution ID: **77** Type: **not specified**

Enabling TPM based system security features

Tuesday, 10 September 2019 15:00 (45 minutes)

Nowadays all consumer PC/laptop devices contain TPM2.0 security chip (due to Windows hardware requirements). Also servers and embedded devices increasingly carry these TPMs. It provides several security functions to the system and the user, such as smartcard-like secure keystore and key operations, secure secret storage, bruteforce-protected access control, etc.

These capabilities can be used in a multitude of scenarios and use cases, including disk encryption, device authentication, user authentication, network authentication, etc. of desktops/laptops, servers, IoTs, mobiles, etc.
Utilizing the TPM requires several layers of software; the driver (inside the kernel), tpm middleware (a TSS implementation), security middleware (e.g. pkcs11), applications (e.g. ssh).

This talk first gives an architectural overview of the hard-/software components involved in typical use cases. Then we will dive into a set of concrete use cases and on different ways in which they can be built up; these use cases will be related to device/user authentication around pkcs11 and openssl implementations.

The talk will end with a list of software and works in progress for introducing TPM functionality to core applications. Finally, a list of potential projects for extending the utilization of the TPM in core software is presented. This latter list shall then drive the discussion on which software is missing or which software has cotributors attending that would like to include such features or which software is currently missing on the list. The current lists of core software are available and updated at https://tpm2-software.github.io/software

Keywords: core libraries, device support, security, tpm, tss

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr FUCHS, Andreas (Fraunhofer SIT)

Presenter: Mr FUCHS, Andreas (Fraunhofer SIT)

Session Classification: LPC Refereed Track

Contribution ID: **79** Type: **not specified**

Application-specific accelerators

Wednesday, 11 September 2019 12:00 (45 minutes)

Application-specific accelerators are going to start showing up in larger numbers in the times ahead. Today there's often no suitable subsystem for them to aggregate into, and the first of them have landed under drivers/misc for the time being.

The goal of this BoF is to introduce and discuss the ground rules for a new drivers/accel subsystem, how it fits in with other subsystems, and expectations of contributions in the short and medium term.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: JOHANSSON, Olof

Presenter: JOHANSSON, Olof

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **81** Type: **not specified**

Reworking of KVA allocator in Linux kernel

Monday, 9 September 2019 10:00 (45 minutes)

Hello.

I would like to give a talk about KVA allocator in the kernel and about improvements i have done.

See below the presentation:

ftp://vps418301.ovh.net/incoming/Reworking_of_KVA_allocator_in_Linux_kernel.pdf

Thank you in advance!

–
Vlad Rezki

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr REZKI, Uladzislau

Presenter: Mr REZKI, Uladzislau

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **82** Type: **not specified**

Update on Task Migration at Google Using CRIU

Tuesday, 10 September 2019 16:00 (30 minutes)

Over the last year we have worked on expanding the task migration using CRIU in Google. The talk will discuss how in some cases the kernel interfaces are lacking for the purpose of migration:

- Lack of support for reading rseq configuration which means that it requires userspace support to migrate users of rseq properly.
- Lack of support for reading what cgroup events the users have registered for.
- Many kernel C/R interfaces are protected by CAP_SYS_ADMIN which we deemed unsafe to have for the migrator agent - CAP_RESTORE could be the solution.

We will discuss new challenges which we have encountered while developing the migration technology further:

- The lack of clean error classification in CRIU forced us to parse the migration logs.
- Lack of support for some less often used kernel features in CRIU (e.g. O_PATH, PR_SET_CHILD_SUBREAPER).
- Migrating containers while also changing the IP of the container is hard but in many cases could be done with little effort on the library or user side.
- We have finalized streaming migration support on our side and in the process we have realized that the hitless migration is infeasible for our latency sensitive users.

I agree to abide by the anti-harassment policy

Yes

Primary author: YURTSEVER, Kamil (Google)

Presenter: YURTSEVER, Kamil (Google)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **83** Type: **not specified**

Inline Encryption Support

Monday, 9 September 2019 17:00 (45 minutes)

Storage hardware with built-in "inline" encryption support is becoming increasingly common, especially on mobile SoCs running Android; it's also now part of the UFS and eMMC standards. These devices en/decrypt data between the application processor and disk without generating disk latency or cpu overhead. Inline encryption hardware can be programmed to hold multiple encryption keys simultaneously and can be dynamically reprogrammed to use any of these programmed encryption keys to en/decrypt a particular request. This makes this new class of storage ideal for supporting fscrypt (file-based encryption). Unfortunately, there isn't currently a unified approach for supporting inline encryption hardware in the Linux kernel.

We've sent out an RFC patchset to add support for inline encryption to the block subsystem, UFS driver, f2fs, and fscrypt
(https://www.spinics.net/lists/linux-block/msg40330.html).
We'll discuss our approach including:

- How the filesystem communicates an encryption key to inline encryption hardware for each struct bio it submits.
- How to add support for inline encryption to storage drivers.
- Support for layered devices like device mapper.
- A software crypto fallback.
- How this work can make future encryption tasks cleaner - like metadata encryption, file-based encryption on removable storage and the possibility of unifying how fscrypt, dm-crypt, and eCryptfs implement encryption.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: TANGIRALA, Satya

Presenter: TANGIRALA, Satya

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **84** Type: **not specified**

Wayland

Monday, 9 September 2019 15:45 (45 minutes)

Wayland is getting close to being ready for day 2 day generic desktop use, close but there still are many small issues to tackle, see e.g. :
https://hansdegoede.livejournal.com/21944.html
https://hansdegoede.livejournal.com/22212.html

The purpose of this microconference is to get people together to discuss the various open issues, try to come up with solutions for some of them and possibly implement some of them.

Expected audience

Anyone who is present at plumbers and is interested in furthering Wayland support.

Expected Topics:
-Discussion about allowing apps run by other users to connect through Wayland, e.g. apps run by sudo
-Should apps (games) be able to change the monitor resolution, should this be a Wayland protocol extension or a portal
-Getting the compositor out of the way for fullscreen games (unredirect support)
-Unified API for monitor configuration à la xrandr to allow commandline configuration of monitor settings?
-More to be added based on CfP for this microconference

Possible speakers/participants which I know plan to be present at plumbers are Alberto Ruiz, Benjamin Berg, Christian Kellner and me.

I also expect Benjamin and or Christian to be willing to co-host the Microconf with me, but I still need to ask them.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: DE GOEDE, Hans (Red Hat)

Presenter: DE GOEDE, Hans (Red Hat)

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **85** Type: **not specified**

Traffic footprint characterization of workloads using BPF

Wednesday, 11 September 2019 10:45 (45 minutes)

Application workloads are becoming increasingly diverse in terms of their network resource requirements and performance characteristics. As opposed to long running monoliths deployed in virtual machines, containerized workloads can be as short lived as few seconds. Today, container orchestrators that schedule these workloads primarily consider their CPU and memory resource requirements since they can easily be quantified. However, network resources characterization isn't as straight forward. Ineffective scheduling of containerized workloads, which could be throughput intensive or latency sensitive, can lead to adverse network performance. Hence, I propose characterizing and learning network footprints of applications running in a cluster, which can be used while scheduling them in containers/VMs such that their network performance can be improved.

There is a well-known network issue, which is achieving low latency for mice flows (those that send relatively small amounts of data) by separating them from the elephant flows (those that send a lot of data). I've written an eBPF program in C that runs at various hook points in the Linux connection tracking (aka conntrack) kernel functions in order to detect network elephant flow, and attribute them to the container or VM, where the flows ingress or egress from. The agent that loads this eBPF program from user space runs in every host in a cluster. It then feeds this learnt information to a container (or VM) scheduling system such that they can use this information proactively, while scheduling containerized workloads with light network footprint (e.g., microservices, functions) and heavy network footprint (e.g., data analytics, data computational applications) on the same cluster, in order to improve their latency and throughput, respectively.

eBPF facilitates running the programs with minimal CPU overhead, in a pluggable, tunable and safe manner, and without having to change any kernel code. It's also worthwhile to discuss how the workload's learnt network footprint can be used for dynamically allocating or tuning Linux network resources like bandwidth, vcpu/vhost-net allocation, receive-side scaling (RSS) queue mappings, etc.
I'll submit a paper with the (working) source code snippets and details if the talk is accepted.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: GHAG, Aditi (VMware)

Presenter: GHAG, Aditi (VMware)

Session Classification: Networking Summit Track

Contribution ID: **88** Type: **not specified**

Efficient Userspace Optimistic Spinning Locks

Wednesday, 11 September 2019 15:00 (45 minutes)

The most commonly used simple locking functions provided by the pthread library are pthread_mutex and pthread_rwlock. They are sleeping locks and so do suffer from unpredictable wakeup latency limiting locking throughput.

Userspace spinning locks can potentially offer better locking throughput, but they also suffer other drawbacks like lock holder preemption which will waste valuable CPU time for those lock spinning CPUs. Another spinning lock problem is contention on the lock cacheline when a large number of CPUs are spinning on it.

This talk presents a hybrid spinning/sleeping lock where a lock waiter can choose to spin in userspace or in the kernel waiting for the lock holder to release the lock. While spinning in the kernel, the lock waiters will queue up so that only the one at the queue head will be spinning on the lock reducing lock cacheline contention. If the lock holder is not running, the kernel lock waiters will go to sleep too so as not to waste valuable CPU cycles. The state of kernel lock spinners will be reflected in the value of lock. Thus userspace spinners can
monitor the lock state and determine the best way forward.

This new type of hybrid spinning/sleeping locks combine the best attributes of sleeping and spinning locks. It is especially useful for applications that need to run on large NUMA systems where potentially a large number of CPUs may be pounding on a given lock.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr LONG, Waiman (Red Hat)

Presenter: Mr LONG, Waiman (Red Hat)

Session Classification: LPC Refereed Track

Contribution ID: **92** Type: **not specified**

Maple Tree

Monday, 9 September 2019 12:00 (45 minutes)

The Red-Black tree and Radix tree are used in many places in the kernel to store ranges. Both of these trees have drawbacks when used for ranges. The Red-Black tree requires writing your own insertion & search code. It is also designed with the assumption that memory accesses are cheap, which is no longer true. The Radix tree performs acceptably well when ranges are aligned to a power of 2, but has awful worst-case performance.

The Maple tree is a fast, cache efficient tree with a simple API. It supports contiguous ranges efficiently, while suffering only minor penalties for discontiguous ranges. Single entries are also supported as a range of length one.

The Maple tree can optionally track free ranges to allow for more efficient allocation. In order to allow it to be used as the basis for the page cache, it will need support for search marks as well as handling reclamation of shadow entries. Other potential users of the Maple tree want more than the three search marks currently supported by the Radix tree.

We want to discuss requirements with potential users of the Maple tree, and to present development since the last Plumbers conference where the broad outlines of the tree were first presented.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr HOWLETT, Liam (Oracle)

Co-author: Mr WILCOX, Matthew (colleague)

Presenter: Mr HOWLETT, Liam (Oracle)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **93** Type: **not specified**

Civil communication in practice: What does it mean to you as an open source developer?

Tuesday, 10 September 2019 17:45 (45 minutes)

Code review is a collaborative activity involving sentiments and emotions that can affect developers' productivity, creativity, and contribution satisfaction. Discussions in a code review environment in open source could get spirited at times as people from diverse backgrounds and interests are part of it. As a consequence, open source communities have become introspective and started to think about the extent to which the differences in communication styles during code reviews can actually affect the morale of the community. Even though many open source projects have started to establish a code of conduct formalizing ground rules for communication between participants with the goal to make everyone comfortable in contributing to the open source project, we still have a need to understand how communication and feelings surrounding it happen in practice.

To address those needs, we propose a BOF with the Linux Community. The goal is to do a short survey focusing on analyzing e-mails from the Linux Kernel Mailing List (LKML) to understand the differences in communication styles and how they impact the Linux community. As a result of this BOF, we will be able to provide valuable information to help communities write their guidelines for code reviews or tools to improve communication in a code review environment.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: FERREIRA, Isabella (Polytechnique Montréal)

Co-authors: STEWART, Kate (Linux Foundation); KHAN, Shuah (The Linux Foundation); GERMAN, Daniel (University of Victoria); ADAMS, Bram (Polytechnique Montréal)

Presenters: FERREIRA, Isabella (Polytechnique Montréal); STEWART, Kate (Linux Foundation); KHAN, Shuah (The Linux Foundation); GERMAN, Daniel (University of Victoria); ADAMS, Bram (Polytechnique Montréal)

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **97** Type: **not specified**

Moving the Linux ABI to userspace

Wednesday, 11 September 2019 10:00 (45 minutes)

The ABI between Linux and user software mostly sits at the user/privileged boundary, although many architectures extend this with a small amount of special-case code that sits in userspace, such as in special pages or shared libraries (vDSOs) mapped into each user process 1 that user code can call into.

The reasons for this are a bit arbitrary: system interface libraries such as glibc and Bionic are maintained as separate projects from the kernel, by different people. The privileged/unprivileged boundary is the de facto demarcation point between projects, because by design only kernel code can run privileged.

Because Linux's user/privileged boundary and ABI are welded together in this way though, the Linux ABI is forced to evolve (or prevented from doing so) for reasons that have little to do with functionality, such as backwards compatibility for superseded interfaces, and optimisations (e.g., vDSO gettimeofday(), getcpu() etc.).

Moving implementation of pieces of kernel functionality between privileged space and userspace is currently hard due to the resulting ABI breaks, yet moving functionality into userspace (e.g., into the vDSO) has some interesting potential use cases, such as:

- Allowing the user/privileged boundary to evolve independently of the kernel ABI.
- Providing a way to push obsolete, deprecated, redundant and/or regrettable syscalls out of the kernel proper.
- Making it easier for userspace to refine its own ABI personality: so things like libc, fakeroot etc., can catch and reimplement syscalls in a transparent way.
- Migrating to a unified library-style ABI instead of relying on a patchwork of bare syscalls, vDSO etc., but without the risk of competing or incompatible implementations.

Migrating a vDSO function to be implemented in privileged space is straightforward: a stub function can be left in the vDSO for old userspace callers to use: the stub just makes the appropriate syscall.

The converse is harder, and requires syscall trapping or filtering mechanisms such as BPF or ptrace.

This presentation will describe some approaches to reflecting syscalls back to userspace, and how feasible they look.

Things I aim to cover:

- What mechanisms can be used?
- How expensive are they, and what breaks?
- What's the likely overhead of doing *all* syscalls through a vDSO or similar?

1 e.g.,
Documentation/ABI/stable/vdso
Documentation/arm/kernel_user_helpers.txt

I agree to abide by the anti-harassment policy

Yes

Primary author: MARTIN, Dave (ARM Limited)

Presenter: MARTIN, Dave (ARM Limited)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **98** Type: **not specified**

Maintaining out of tree patches over the long term

Tuesday, 10 September 2019 10:45 (45 minutes)

The PREEMPT_RT patchset is the longest existing large patchset living outside the Linux kernel. Over the years, the realtime developers had to maintain several stable kernel versions of the patchset. This talk will present the lessons learned from this experience, including workflow, tooling and release management that has proven over time to scale. The workflow deals with upstream changes and changes to the patchset itself. Now that the PREEMPT_RT patchset is about the be merged upstream, we want to share our toolset and methods with others who may be able to benefit from our experience.

This talk is for people who want to maintain an external patchset with stable releases.

I agree to abide by the anti-harassment policy

Yes

Primary authors: WAGNER, Daniel; BRISTOT DE OLIVEIRA, Daniel (Red Hat, Inc.); ROSTEDT, Steven; ZANUSSI, Tom; KACUR, John

Presenters: WAGNER, Daniel; BRISTOT DE OLIVEIRA, Daniel (Red Hat, Inc.); ROSTEDT, Steven; ZANUSSI, Tom; KACUR, John

Session Classification: LPC Refereed Track

Contribution ID: **99** Type: **not specified**

oomd2 and beyond: a year of improvements

Monday, 9 September 2019 10:00 (45 minutes)

Running out of memory on a host is a particularly nasty scenario. In the Linux kernel, if memory is being overcommitted, it results in the kernel out-of-memory (OOM) killer kicking in. Perhaps surprisingly, the kernel does not often handle this well. oomd builds on top of recent kernel development to effectively implement OOM killing in userspace. This results in a faster, more predictable, and more accurate handling of OOM scenarios.

oomd has gained a number of new features and interesting deployments in the last year. The most notable feature is a complete redesign of the control plane which enables arbitrary but "gotcha"-free configurations. In this talk, Daniel Xu will cover past, present, future, and path-not-taken development plans along with experiences gained from overseeing large deployments of oomd.

I agree to abide by the anti-harassment policy

Yes

Primary author: XU, Daniel (Facebook)

Presenter: XU, Daniel (Facebook)

Session Classification: LPC Refereed Track

Contribution ID: **100** Type: **not specified**

Finding more DRAM

Wednesday, 11 September 2019 10:00 (45 minutes)

The demand of DRAM across different platforms is increasing but the cost is not decreasing. Thus DRAM is a major factor of the total cost across all kinds of devices like mobile, desktop or servers. In this talk we will be presenting the work we are doing at Google, applicable to Android, Chrome OS and data center servers, on extracting more memory out of running applications without impacting performance.

The key is to proactively reclaim idle memory from the running applications. For the Android and Chrome OS, the user space controller can provide hints of the idle memory at the applications level while the servers running multiple workloads, an idle memory tracking mechanism is needed. With such hints the kernel can proactively reclaim memory given that estimated refault cost is not high. Using in-memory compression or second tier memory, the refault cost can be reduced drastically.

We have developed and deployed the proactive reclaim and idle memory tracking across Google data centers 1. Defining idle memory as memory not accessed in the last 2 mins, we found 32% idle memory across data centers and we were able to reclaim 30% of this idle memory, while not impacting the performance. This results in 3x cheaper memory for our data centers. 98% of the applications spend only around 0.1% of their CPU on memory compression and decompression. Also the idle memory tracking on average takes less than 11% of a single logical CPU.

The cost of proactive reclaim and idle memory tracking is reasonable for the data centers cost of ownership of memory, however, it imposes challenges for power constrained devices based on Android and Chrome OS. These devices run diverse applications e.g. Chrome OS can run Android and Linux in a VM. To that end, we are working on making idle memory tracking and proactive reclaim feasible for such devices. Henceforth, we are interested and would like to initiate discussion on making proactive reclaim useful for other use-cases as well.

1 Software-Defined Far Memory in Warehouse-Scale Computers, ACM ASPLOS 2019.

I agree to abide by the anti-harassment policy

Yes

Primary authors: BUTT, Shakeel (Google); BAGHDASARYAN, Suren (Google); ZHAO, Yu (Google)

Presenters: BUTT, Shakeel (Google); BAGHDASARYAN, Suren (Google); ZHAO, Yu (Google)

Session Classification: LPC Refereed Track

Contribution ID: **102** Type: **not specified**

BPF is eating the world, don't you see?

Tuesday, 10 September 2019 10:00 (45 minutes)

The BPF VM in the kernel is being used in ever more scenarios where running a restricted, validated program in kernel space provides a super powerful mix of flexibility and performance which is transforming how a kernel work.

That creates challenges for developers, sysadmins and support engineers, having tools for observing what BPF programs are doing in the system is critical.

A lot has been done recently in improving tooling such as perf and bpftool to help with that, trying to make BPF fully supported for profiling, annotating, tracing, debugging.

But not all perf tools can be used with JITed BPF programs right now, areas that need work, such as putting probes and collecting variable contents as well as further utilizing BTF for annotation are areas that require interactions with developers to gather insights for further improvements so as to have the full perf toolchest available for use with BPF programs.

These recent advances and this quest for feedback about what to do next should be the topic of this talk.

I agree to abide by the anti-harassment policy

Yes

Primary author: CARVALHO DE MELO, Arnaldo (Red Hat Inc.)

Presenter: CARVALHO DE MELO, Arnaldo (Red Hat Inc.)

Session Classification: LPC Refereed Track

Contribution ID: **103** Type: **not specified**

Memory management bits in arch/*

Tuesday, 10 September 2019 10:00 (45 minutes)

There is a lot of similar and duplicated code in architecture specific bits of memory management.

For instance, most architectures have

```
\#define PGALLOC\_GFP (GFP\_KERNEL | \_\_GFP\_ZERO)
```

for allocating page table pages and many of them use similar, if not identical, implementation of pte_alloc_one*().

But that's only the tip of the iceberg.

There are several early_alloc() or similarily called routines that do

```
if (slab\_is\_available())
    return kzalloc();
else
    return memblock\_alloc();
```

Some other trivial examples are free_initmem(), free_initrd_mem() which were nearly identical accross many architectures until very recently.

More complex cases are per-cpu initialization, passing of memory topology to the generic MM, reservation of crash kernel, mmap of vdso etc. They are not really duplicated, but still are very similar in at least several architectures.

While factoring out the common code is an obvious step to take, I believe there is also room for refining arch <-> mm interface to avoid adding extra HAVE_ARCH_NO_BOOTMEM^w^wWHAT_NOT and then searching for the ways to get rid of them.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: RAPOPORT, Mike (IBM)

Presenter: RAPOPORT, Mike (IBM)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **104** Type: **not specified**

replacing mmap_sem with finer grained locks

Tuesday, 10 September 2019 10:45 (45 minutes)

In the linux kernel, most operations affecting a process's address space are protected by by mmap_sem (a per-process read-write semaphore).

This simple design is increasingly a problem for multi-threaded applications, and often causes threads that operate on separate parts of their address space to end up blocking on each other due to false sharing issues - mmap_sem only supports locking the entire address space at once, so it can't take into consideration that the operations are not overlapping.

I would like to discuss:
1- The sort of blocking issues that are seen today due to the current mmap_sem design;
2- mmap_sem mitigations that have been introduced over time, and have kept the situation bearable but not fundamentally solved the issue;
3- try to discuss from first principles how the MM data structures and locking mechanisms would have to evolve to support finer grained MM locking, and how to progressively migrate the current MM codebase towards such a finer grained MM locking scheme;
4- (hopefully) present early results with a fine grained MM locking prototype.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LESPINASSE, Michel (Google)

Presenter: LESPINASSE, Michel (Google)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **107** Type: **not specified**

printk: Why is it so complicated?

Monday, 9 September 2019 12:45 (45 minutes)

The printk() function has a long history of issues and has undergone many iterations to improve performance and reliability. Yet it is still not an acceptable solution to reliably allow the kernel to send detailed information to the user. And these problems are even magnified when using a real-time system. So why is printk() so complicated and why are we having such a hard time finding a good solution?

This talk will briefly cover the history of printk() and why the recent major rework was necessary. It will go through the details of the rework and why we believe it solves many of the issues. And it will present the issues still not solved (such as fully synchronous console writing), why these issues are particularly complex and controversial, and review some of the proposed solutions for moving forward.

This talk may be of particular interest to developers with experience or interest in lockless ring buffers, memory barriers, and NMI-safe synchronization.

I agree to abide by the anti-harassment policy

Yes

Primary author: OGNESS, John (Linutronix GmbH)

Presenter: OGNESS, John (Linutronix GmbH)

Session Classification: LPC Refereed Track

Contribution ID: **108** Type: **not specified**

The list is our process: An analysis of the kernel's email-based development process

Monday, 9 September 2019 12:45 (45 minutes)

Implementing safety-critical systems usually requires adhering to meticulously defined development processes that specify how code is supposed to be developed, integrated and reviewed, driven by the assumption that a disciplined approach leads to reliably high quality. While known to produce code that can satisfy the highest quality standards, Linux kernel development does not follow such strict patterns, although it is certainly far from a random process. But how can we ensure the quality of a mostly informal approach?

Our work aims at identifying core properties, strengths and weaknesses in the development process by tracking the evolution of components from initial submissions on mailing lists to the final merged contributions.

This talk will:

- introduce heuristics to identify the evolution of patches on the mailing list and match patch emails against their included git commit counterparts.
- present our publicly available data set of kernel-related email available, curated large-scale data set from more than 200 kernel-related mailing lists

We discuss observations and insights and we draw, ranging form simpler questions like how long the average time from the first version of a patch submission to its final inclusion is, down to a categorisation and analysis of off-list patches and ignored patches.

We particularly seek interaction with experts from the community to discuss benefits and limitations of our approach. We will show how we would like to make this information available in the patchwork tool, and present prototypes of tools and development process analyses that that we would like to refine so that they are useful to Linux kernel developers and maintainers in their every day work. We hope this work can contribute to a future kernel maintainers handbook.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: Mr RAMSAUER, Ralf (OTH Regensburg); Prof. MAUERER, Wolfgang (OTH Regensburg); BULWAHN, Lukas (BMW AG)

Presenters: Mr RAMSAUER, Ralf (OTH Regensburg); Prof. MAUERER, Wolfgang (OTH Regensburg); BULWAHN, Lukas (BMW AG)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **109** Type: **not specified**

KUnit - Unit Testing for the Linux Kernel

Wednesday, 11 September 2019 10:45 (45 minutes)

KUnit is a new lightweight unit testing and mocking framework for the Linux kernel. Unlike Autotest and kselftest, KUnit is a true unit testing framework; it does not require installing the kernel on a test machine or in a VM (however, KUnit still allows you to run tests on test machines or in VMs if you want) and does not require tests to be written in userspace running on a host kernel. You can read more about KUnit in this LWN article.

In the first half of the talk we will provide background on what unit testing is, why we think it is important for the Linux kernel, how KUnit provides a viable unit testing library implementation, and offer a brief demonstration of how it might be used.

In the second half of the talk we will talk about the future. We will talk about KUnit's roadmap, the challenges that KUnit is facing, how to structure the Linux kernel testing paradigm, and how KUnit fits into it.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: HIGGINS, Brendan (Google LLC)

Presenter: HIGGINS, Brendan (Google LLC)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **110** Type: **not specified**

drgn: Programmable Debugging

Monday, 9 September 2019 10:00 (22 minutes)

drgn (https://github.com/osandov/drgn) is a programmable debugger that makes it easy to introspect and debug state in the kernel. With drgn, it's possible to explore and analyze data structures with the full power of Python. See the LWN coverage of the presentation at LSF/MM: https://lwn.net/Articles/789641/. This presentation will demonstrate the capabilities of drgn, discuss future plans, and explore ways that the kernel and surrounding ecosystem can make introspection easier and more powerful.

I agree to abide by the anti-harassment policy

Yes

Primary author: SANDOVAL, Omar

Presenter: SANDOVAL, Omar

Session Classification: Tracing MC

Contribution ID: **111** Type: **not specified**

Tracing Data Access Pattern with Bounded Overhead and Best-effort Accuracy

Tuesday, 10 September 2019 15:00 (45 minutes)

Background

Memory pressure is inevitable in many environments. A decade size survey1 of DRAM to CPU ratio in virtual machines and physical machines for data centers implies that the pressure will be even more common and severe. As an answer to this problem, heterogeneous memory systems utilizing recently evolved memory devices such as non-volatile memory along with the DRAM are rising.

Nevertheless, because such devices are not only denser and cheaper but also obviously slower than DRAM, more optimal memory management is required.
For this reason, various novel approaches[2,3,4] have proposed and discussed.
One common goal of general memory management mechanisms including such approaches is placing each data object in proper location according to its data access pattern. Thus, knowing the data access pattern of given workloads is key.

The Linux kernel is utilizing pseudo-LRU scheme for the purpose but it sacrificed too much accuracy for low overhead. Some approaches[3,5,6] track the access pattern based on page table access bit but such a technique could incur arbitrarily high overhead[3,6] or low accuracy[5] as the size of target workloads grows.

Our solution

We are developing a data access pattern profiling technique and tools that allow users to control the upper bound of profiling overhead while providing a best-effort quality of the result regardless of the size of the target workload.
Basically, the solution is implemented on page table access bit sampling, which is widely used from other approaches[5]. In this approach, users can control the upper bound of the profiling overhead by setting the total number of the sampling regions.

What differentiates ours from the others is its adaptive classification of sampling regions. The algorithm adaptively merges and splits each sampling region so that every data item in each region to have a similar data access pattern. In this way, our mechanism can minimize the number of sampling regions while maximizing the profiling accuracy.

Implementation

We implemented the mechanism as a kernel module that interacts with userspace via the ‘debugfs’ interface. We also provide userspace tools that help the use of the interface and visualization of the profiled results.

Expected users

We believe our mechanism could be used by both kernel space and user space.

In kernel space, the aforementioned heterogeneous memory management approaches[2,3] could directly use this mechanism for efficient and effective data access pattern exploitation. Furthermore, this can be used by many traditional memory management subsystems that relying on the kernel's pseudo-LRU or naive assumptions. For example, selection of victim pages for page cache eviction or swap, pages to be promoted or demoted to or from huge pages (THP), target pages to compact nearby could use this.

In userspace, system administrators or application programmers could use this tool to quickly analyze their workloads. The result can be used for a various way. Administrators might use the result to know a performance-effective working set size of their workloads and operate their system more efficiently.
Programmers would optimize their programs using madvise()-like system calls[5] to give data access pattern hints to the system.

Evaluations

We applied this resulting tools to more than twenty of various realistic workloads including scientific, machine learning, and big data applications and confirmed that it provides effective and efficient profiling. For the confirmation, we visualized the output and compared with manual code review. We also evaluated its usefulness by manually estimating the performance-effective working set size and optimizing with madvise() system calls. Our performance-effective working set size was similar to that we found using time-consuming repetitive experiments and the optimization improved the performance under memory pressure situation up to 2x.

Future plans

We are planning to open source this tool and submit the patchset to LKML for upstream merge.

Expected results of this talk

We hope this talk to help discussions about the effective and efficient way to get data access pattern and how to use the data from memory management systems.
Also, we would like to hear back kernel core developers' comments for upstreaming of this tool.

References

1 Nitu, Vlad, et al. "Welcome to zombieland: practical and energy-efficient memory disaggregation in a datacenter." Proceedings of the Thirteenth EuroSys Conference. ACM, 2018.
[2] "NUMA nodes for persistent-memory management." https://lwn.net/Articles/787418/
[3] "Proactively reclaiming idle memory." https://lwn.net/Articles/787611/
[4] "[RFCv2 0/6] introduce memory hinting API for external process." https://lore.kernel.org/lkml/20190531064313.193437-1-minchan@kernel.org/T/#u
[5] "Cache Modeling and Optimization using Miniature Simulations." https://www.usenix.org/conference/atc17/technical-sessions/presentation/waldspurger
[6] "Idle Page Tracking." https://www.kernel.org/doc/html/latest/admin-guide/mm/idle_page_tracking.html

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: PARK, SeongJae; Mr LEE, Yunjae (Seoul National University); Prof. YEOM, Heon Y. (Seoul National University)

Presenter: PARK, SeongJae

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **112** Type: **not specified**

CPU controller on a single runqueue

Tuesday, 10 September 2019 17:00 (45 minutes)

The cgroups CPU controller in the Linux scheduler is implemented using hierarchical runqueues, which introduces a lot of complexity, and incurs a large overhead with frequently scheduling workloads. This presentation is about a new design for the cgroups CPU controller, which uses just one runqueue, and instead scales the vruntime by the inverse of the task priority. The goal is to make people familiar with the new design, so they know what is going on, and do not need to spend a month examining kernel/sched/fair.c to figure things out.

I agree to abide by the anti-harassment policy

Yes

Primary author: VAN RIEL, Rik (Facebook)

Presenter: VAN RIEL, Rik (Facebook)

Session Classification: LPC Refereed Track

Contribution ID: **114** Type: **not specified**

Core Scheduling: Taming Hyper-Threads to be secure

Monday, 9 September 2019 10:45 (45 minutes)

Last couple of years, we have witnessed an onslaught of vulnerabilities in the design and architecture of cpus. It is interesting and surprising to note that the vulnerabilities are mainly targeting the features designed to improve the performance of cpus - most notable being the hyperthreading(smt). While some of the vulnerabilities could be mitigated in software and cpu microcodes, couple of others didn't have any satisfiable mitigation other than making sure that smt is off and every context switch needed to flush the cache to clear the data used by the task that is being switched out. Turning smt off is not a viable alternative to many production scenarios like cloud environment where you lose a considerable amount of computing power by turning off smt. To address this, there have been community efforts to keep smt on while trying to make sure that non-trusting applications are never run concurrently in the hyperthreads of the core, they have been widely called as core scheduling.

This talk is about the development, testing and profiling efforts of core scheduling in the community. There were multiple proof of concepts - while differing in the design, ultimately trying to make sure that only mutually trusted applications run concurrently on the core. We discuss the design, implementation and performance of the POCs. We also discuss the profiling attempts to understand the correctness and performance of the patches - various powerful kernel features that we leveraged to get the most time sensitive data from the kernel to understand the effect of scheduler with the core scheduling feature. We plan to conclude with a brief discussion of the future directions of core scheduling.

The core idea about core scheduling is to have smt on and make sure that only trusted applications run concurrently on siblings of a core. If there are no group of trusting applications runnable on the core, we need to make sure that remaining siblings should idle while applications run in isolation on the core. This should also consider the performance aspects of the system. Theoretically it is impossible to reach the same level of performance where the cores are allowed to any runnable applications. But if the performance of core scheduling is worse than or same as the smt off situation, we do not gain anything from this feature other than the added complexity in the scheduler. So the idea is to achieve a considerable boost in performance compared to smt-off for the majority of production workloads.

Security boundary is another aspect of critical importance in core scheduling. What should be considered as a trust boundary? Should it be at the user/group level, process level or thread level? Should kernel be considered trusty by applications or vice-versa? With virtualization and nested virtualization in picture, this gets even more complicated. But answers to most of these questions are environment and workload dependent and hence these are implemented as policies rather than hardcoding in the code. And then arises the question - how the policies should be implemented? Kernel has a variety of mechanisms to implement these kind of policies and the proof of concepts posted upstream mainly uses cgroups. This talk also discusses other viable options for implementing the policies.

I agree to abide by the anti-harassment policy

Yes

Primary authors: DESFOSSEZ, Julien (DigitalOcean); REMANAN PILLAI, Vineeth

Presenters: DESFOSSEZ, Julien (DigitalOcean); REMANAN PILLAI, Vineeth

Session Classification: LPC Refereed Track

Contribution ID: **115** Type: **not specified**

Linux Kernel VxLan with Multicast Routing for flood handling

Monday, 9 September 2019 10:00 (45 minutes)

The Linux kernel VxLan driver supports two ways of handling flooded traffic to multiple remote VxLan termination end points (VTEPS):
(a) Head end replication: where the VxLan driver sends a copy of the packet to each participating remote VTEPs
(b) Use of multicast routing to forward to participating remote VTEPs

(b) is generally preferred for both hardware and software VTEP deployments because it scales better. The kernel VxLan driver supports (b) with static config today. One has to specify the multicast group with the outgoing uplink interface for VxLan multicast replication to work. This is mostly ok for deployments where VTEPs are deployed on the host/hypervisor. When deploying Linux VTEPs on the Top-Of-the-Rack (TOR) switches in a data center CLOS network, it is impossible to configure the outgoing interface statically. Typically a multicast routing protocol like PIM is used to dynamically calculate multicast trees and install forwarding paths for multicast traffic.

In this talk we will cover:
- Vxlan Multicast deployment scenarios with Vxlan VTEPs at the TOR switches
- Current challenges with integrating Vxlan Multicast replication in a dynamic multicast routing environment
- Solutions to these challenges: (a) Patches to fix routing of locally generated multicast packets (need for ip_mr_output) (b) Patches to VxLan driver to allow multicast replication without a static outgoing interface
- Scale
- Futures on VxLan deployments in multicast environment

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: PRABHU, Roopa (Roopa)

Presenter: PRABHU, Roopa (Roopa)

Session Classification: Networking Summit Track

Contribution ID: **116** Type: **not specified**

RISC-V Container

Compared to VM, container technology has been always argued for the security. We might need to discuss how to fit current container implementation into RISC-V arch in such a area. And RISC-V has not had any particular hardware considerations like Intel SGX and even AMD, however we can go far as we can and get some feedback to RISC-V foundation.

I agree to abide by the anti-harassment policy

Yes

Primary author: CHEN, Tiejun (VMware)

Presenter: CHEN, Tiejun (VMware)

Session Classification: RISC-V MC

Contribution ID: **118** Type: **not specified**

Securing Container Runtimes with openat2 and libpathrs

Tuesday, 10 September 2019 18:00 (30 minutes)

Userspace has (for a long time) needed a mechanism to restrict path resolution. Obvious examples are those of FTP servers, Web Servers, archiving utilities, and now container runtimes. While the fundamental issue with privileged container runtimes opening paths within an untrusted rootfs was known about for many years, the recent CVEs (CVE-2018-15664 and CVE-2019-10152 being the most recent) to that effect has brought more light to the issue.

This is an update on the work briefly discussed during LPC 2018, complete with redesigned patches and a new userspace library that will allow for backwards-compatibility on older kernels that don't have openat2(2) support. In addition, the patchset now has new semantics for "magic links" (nd_jump_link-style "symlinks") that will protect against several file descriptor re-opening attacks (such as CVE-2016-9962 and CVE-2019-5736) that have affected all sorts of container runtimes and other programs. It also provides the ability for userspace to further restrict the re-opening capabilities of O_PATH descriptors.

In order to facilitate easier (safe) use of this interface, a new userspace library (libpathrs) has been developed which makes use of the new openat2(2) interfaces while also having userspace emulation of openat2(RESOLVE_IN_ROOT) for older kernels. The long-term goal is to switch the vast majority of userspace programs that deal with potentially-untrusted directory trees to use libpathrs and thus avoid all of these potential attacks.

The important parts of this work (and its upstream status) will be outlined and then discussion will open up on what outstanding issues might remain.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr SARAI, Aleksa (SUSE LLC)

Presenter: Mr SARAI, Aleksa (SUSE LLC)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **120** Type: **not specified**

Having one, unified eBPF network packet filter, no more, no less.

Monday, 9 September 2019 17:00 (45 minutes)

For long time, The kernel have contained two mechanisms with similar packet filtering functionality: tc filter (with chains) and iptables/nftables.

As eBPF is starting to take over, once again we seem to have two mechanisms with similar functionality: BPFilter and the newly suggested OVS-eBPF datapath (on top on tc).

As we move to using eBPF, I'd like to discuss the possibility of uniting those two functionalities, both the BPFilter and OVS-eBPF path, into a single one and let go of all the duplicate code.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr SHATTAH, Guy

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **122** Type: **not specified**

Improving Route Scalability with Nexthop Objects

Wednesday, 11 September 2019 12:00 (45 minutes)

Route entries in a FIB tend to be very redundant with respect to nexthop configuration with many routes using the same gateway, device and potentially encapsulations such as MPLS. The legacy API for inserting routes into the kernel requires the nexthop data to be included with each route specification leading to duplicate processing verifying the nexthop data, an effect that is magnified as the number of paths in the route increases (e.g., ECMP).

A new API was recently committed to the kernel for managing nexthops as separate objects from routes. The nexthop API allows nexthops to be created first and then routes can be added referencing the nexthop object. This API allows routes to be managed with less overhead (e.g., dramatically reducing the time to insert routes) and enables new capabilities such as atomically updating a nexthop configuration without touching the route entries using it.

This talk will discuss the nexthop feature touching on the kernel side implementation, reviewing the userspace API and what to expect for notifications, performance improvements and potential follow on features. While the nexthop API is motivated by Linux as a NOS, it is useful for other networking deployments as well such as routing on the host and XDP.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: AHERN, David

Presenter: AHERN, David

Session Classification: Networking Summit Track

Contribution ID: **123** Type: **not specified**

The ieee802154 and 6lowpan Kernel Subsystems

Monday, 9 September 2019 17:30 (30 minutes)

This talk will put the spotlight on the linux-wpan project, which brings IEEE 802.15.4 and 6LoWPAN support to the Linux Kernel. Designed for low-power devices these protocols are ideal for the use in some IoT applications. Over the last years IEEE 802.15.4 support has slowly found its way into the mainline kernel. The 6LoWPAN code is shared with the Bluetooth stack and the ieee802154 subsystem itself is growing in functionality.

The talk will give an overview of the implemented functionality in the ieee802154 and 6lowpan subsystems and their use from userspace for the data (socket) and control (netlink) planes. It will describe the current hardware support, header compression techniques used in 6lowpan and areas where the stack is currently limited. We will close with a comparison of linux-wpan against other IEEE 802.15.4 stacks (Zephyr, RIOT, OpenThread).

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr SCHMIDT, Stefan

Presenter: Mr SCHMIDT, Stefan

Session Classification: You, Me, and IoT MC

Contribution ID: **124** Type: **not specified**

Kernel documentation

Tuesday, 10 September 2019 17:00 (45 minutes)

What could be more fun than talking about kernel documentation? Things we could get into:

- The state of the RST transition, what remains to be done, whether it's all just useless churn that makes the documentation worse, etc.
- Things we'd like to improve in the documentation toolchain.
- Overall organization of Documentation/ and moving docs when the need arises. It seems I end up fighting about this more than just about anything else, but I think it's important to organize our docs for the convenience of the people using them.
- The ultimate vision for kernel docs (for now). RST conversion and imposing some organization are important, but they will not, themselves, give us a coherent set of documentation. What can we do to have documentation that is useful, current, and maintainable, rather than the dusty attic we have now?

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: CORBET, Jonathan (Linux Plumbers Conference)

Presenter: CORBET, Jonathan (Linux Plumbers Conference)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **125** Type: **not specified**

Introduce an implementation of IOMMU in linux-riscv

Monday, 9 September 2019 10:45 (30 minutes)

IOMMU is a very popluar equipment for both embed and server virtualization area. In the topic we'll focus on embed area and shared virtual address.

Firstly, we'll talk about the value of IOMMU for the embed system and what the benefit we could get from IOMMU in our cost-down embed system.

Secondly, Guo will share the experience on the IOMMU implementation, eg: How to keep the same asid with CPU and IOMMU in hardware. How to share CPU's page table with IOMMU for user space address.

Lastly, let's have a free discussion on riscv mmu, iommu and SVA related issues.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr GUO, Ren (c-sky.com (belong to Alibaba.com))

Co-author: Mr MAO, Han (c-sky.com (belong to Alibaba.com))

Presenters: Mr GUO, Ren (c-sky.com (belong to Alibaba.com)); Mr MAO, Han (c-sky.com (belong to Alibaba.com))

Session Classification: RISC-V MC

Contribution ID: **126** Type: **not specified**

Introduce an implementation of perf trace in riscv system

Monday, 9 September 2019 11:15 (15 minutes)

RISC-V trace spec draft have defined some trace format, we'll share our implementation of linux perf trace based on the spec. How to deal with SMP perf issues, how to verify our design in qemu, demonstrate a demo of perf trace with riscv-qemu.

Lastly, let's discuss perf issues from PMU to trace, any riscv perf topic.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr REN, Guo

Co-author: Mr MAO, Han (c-sky.com (belong to Alibaba.com))

Presenters: Mr REN, Guo; Mr MAO, Han (c-sky.com (belong to Alibaba.com))

Session Classification: RISC-V MC

Contribution ID: **127** Type: **not specified**

Secure Image-less Container Migration

Tuesday, 10 September 2019 16:45 (15 minutes)

Container runtimes, engines and orchestrators provide a production-grade, robust, high-performing, but also relatively self-managing, self-healing infrastructure using innovative open-source technologies.

CRIU allows the running state of containerised applications to be preserved as a collection of files that can be used to create an equivalent copy of the applications at a later time, and possibly on a different system.

However, for a live migration mechanism to be effective it is very important to minimize the downtime of these applications without compromising security. Therefore, in this talk we discuss new features of CRIU that enable seamless live migration based on direct communication mechanism between source and destination nodes, in order to avoid the generation of intermediate image files and to keep only necessary state information cached in memory.

I agree to abide by the anti-harassment policy

Yes

Primary authors: Mr STOYANOV, Radostin (University of Aberdeen); Dr KOLLINGBAUM, Martin (University of Aberdeen)

Presenters: Mr STOYANOV, Radostin (University of Aberdeen); Dr KOLLINGBAUM, Martin (University of Aberdeen)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **130** Type: **not specified**

Fixing the Linux boot process in RISC-V

Monday, 9 September 2019 10:25 (20 minutes)

RISC-V now has better support for open source boot loaders like U-Boot and coreboot compared to last year. As a result of this developers can use the same boot loaders to boot Linux on RISC-V as they do in other architectures, but there's more work to be done. We will discuss the current state of the boot flow and pending issues.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: PATRA, ATISH (Western Digital)

Presenter: PATRA, ATISH (Western Digital)

Session Classification: RISC-V MC

Contribution ID: **132** Type: **not specified**

RISC-V Platform Specification Progress

Monday, 9 September 2019 10:00 (25 minutes)

The RISC-V UNIX-Class platform specification working group started in May and aims to have a first release by the end of the year. This talk will discuss where we are and where we're going.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: DABBELT, Palmer (SiFive); PATRA, ATISH (Western Digital)

Presenters: DABBELT, Palmer (SiFive); PATRA, ATISH (Western Digital)

Session Classification: RISC-V MC

Contribution ID: **133** Type: **not specified**

Csky Intro - what's the meaning of a new arch for linux

Wednesday, 11 September 2019 10:00 (45 minutes)

The csky architecture officially merged the main line in linux-4.20. Before that, eight architectures have just been removed from the main line. Many people ask what is the meaning of csky upstream? Also includes our colleagues. Here, we will give some examples to introduce the progress of the csky architecture in the past six months and the value and significance of linux-csky. This is an open discussion about the csky architecture and any questions are welcomed.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: Mr REN, Guo; Mr MAO, Han (c-sky.com (belong to Alibaba.com))

Presenters: Mr REN, Guo; Mr MAO, Han (c-sky.com (belong to Alibaba.com))

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **134** Type: **not specified**

Analyzing changes to the binary interface exposed by the Kernel to its modules

Tuesday, 10 September 2019 10:00 (30 minutes)

Operating system distributors often face challenges that are somewhat different from that of upstream kernel developers. For instance, some kernel updates often need to stay at least binary compatible with modules that might be “out of tree” for some time.

In that context, being able to automatically detect and analyze changes to the binary interface exposed by the kernel to its module does have some noticeable value.

The Libabigail framework is capable of analyzing ELF binaries along with their accompanying debug info in the DWARF format, detect and report changes in types, functions, variables and ELF symbols. It has historically supported that for user space shared libraries and application so we worked to make it understand the Linux kernel binaries.

In this presentation, we are going to present the current support of ABI analysis for Linux Kernel binaries, especially the kind of information that Libabigail consumes from DWARF and thus what it would need from an alternative debug info format.

We hope the presentation will lead to discussions on topics revolving around what it would take to adapt Libabigail to the emerging alternate debug info formats and if that would make sense at all.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr SEKETELI, Dodji (Red Hat)

Presenter: Mr SEKETELI, Dodji (Red Hat)

Session Classification: Toolchains MC

Contribution ID: **135** Type: **not specified**

Cgroup v1/v2 Abstraction Layer

Tuesday, 10 September 2019 19:10 (20 minutes)

Abstract

We have cgroup v1 users who want to switch to cgroup v2, but there currently isn't an upstream migration story for them. (Previous LPC talks have focused on the issues of migrating from v1 to v2, but no substantial upstream solution has come to fruition.)

The goal of this talk is to discuss the cgroup v1 to v2 migration path and gauge community interest in a cgroup v1/v2 abstraction layer.

Problem Statement

Several Oracle products have very, very long product lifetimes and are designed to run on a wide range of Linux kernels and systemd versions. These products are encountering difficulties as cgroups continues to grow and change. Older kernels only support v1, but v2 is the future in newer kernels with v1 effectively in maintenance mode. Newer versions of systemd have started to abstract the cgroup interface, but upgrading older systems to newer versions of systemd is often not feasible. Ultimately, long-lifespan products are spending an increasing and inordinate amount of time and effort managing their cgroup interfaces.

There is interest within Oracle to create a cgroup abstraction layer that will allow long-lived products to utilize the most advanced cgroups features available on every supported system. Ideally these products will be able to rely upon a library to abstract away the low-level cgroup implementation details on that system.

Audience

Anyone interested in cgroups

Why Should the Audience Attend and/or Care

- We would like to develop a cgroups abstraction layer in the next year or so. We would love to collaborate with others to build and design a solution that can help the entire community
- Do other people/companies have interest in an abstraction layer? We want to hear other use cases and needs to better serve as many people as possible
- Is there already something out there that we can utilize and build on?
- Given the wide array of users and use cases, the library will likely need to have bindings for today's most popular languages - python, go java, etc.
- There are a multitude of API possibilities. What level(s) of abstraction are of interest to the community? e.g.
  GiveMeCpus(cgname=foo, cpu_count=2, exclusive=True, numa_aligned=True, …)
  CgroupCreate(cgname=foo, secure_from_sidechannel=True, …)

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: HROMATKA, Tom

Presenter: HROMATKA, Tom

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **137** Type: **not specified**

Early HPC uses cases for RISC V

Monday, 9 September 2019 12:00 (15 minutes)

The current main uses cases of RISC V center on embedded uses and small configurations. However, RISC V seems to be also a useful platform to do High Performance Computing and may be able to deliver custom solutions that can go well beyond what the traditional processor vendors can offer. There are already efforts underway to use ARM for that purpose but those approaches are constrained by limits placed on that platform through licensing. It is natural to expect a move to RISC V there as well.

This talk is looking at use cases in HPC such as to create custom compute solutions replacing GPUs and numerous vector processing extensions of typical processors. HPC users often feel constrained by the limits on the implementations provided to them and are hopeful that RISC V will offer a heretofore unavailable flexibility for them.

Other further use cases may be customizing access to newer forms of memory (such as HBM, Persistent memory, DDR5/6 and other approaches) as well as providing implementations of fast packed processing for High Speed Networks (such as Infiniband, NVlink and Ethernet). The problem of line rate processing at 100Gbps and higher may actually require the development of custom processors to have a reasonable way to process data at these speeds.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LAMETER, Christopher (Jump Trading LLC)

Presenter: LAMETER, Christopher (Jump Trading LLC)

Session Classification: RISC-V MC

Contribution ID: **139** Type: **not specified**

TAB Elections

Monday, 9 September 2019 18:30 (1 hour)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk

Contribution ID: **140** Type: **not specified**

Closing Plenary

Wednesday, 11 September 2019 18:45 (1 hour)

Contribution ID: **141** Type: **not specified**

Closing Party

Wednesday, 11 September 2019 20:00 (3 hours)

Busses will be leaving from the Corinthia Hotel lobby from 19:30

I agree to abide by the anti-harassment policy

I confirm that I am already registered for LPC 2019

Contribution ID: **146** Type: **not specified**

Upstream kernel CI

Monday, 9 September 2019 17:45 (45 minutes)

Testing the upstream kernel is not an easy task. The burden is still largely put on developers, although several projects are now covering parts of it such as 0-day, LKFT, CKI, Coccinelle, syzkaller and kernelci.org. While they all tend to have their own speciality, they also face a lot of similar challenges.

This BoF is to give an opportunity to exchange ideas and bring together people from the upstream kernel testing community. Are there ways to share kernel builds, platforms or code between projects to remove duplication of efforts? Which open tools are you using, and is there a need for anything new? Which areas of the kernel are suffering the most from a lack of test coverage? How does one even power up a dev board in a lab?

Tackling these problems requires a lot of energy. Last but not least and thanks to Collabora, attendees will be offered food (if the venue permits it)!

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: TUCKER, Guillaume (Collabora Limited)

Presenter: TUCKER, Guillaume (Collabora Limited)

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **148** Type: **not specified**

Dealing with complex test suites

Tuesday, 10 September 2019 10:35 (20 minutes)

Boot testing is already hard to do well on a wide variety of hardware. However it is only scratching the surface of the kernel code base. To take projects such as Kernel CI to the next level and increase coverage, functional tests are becoming the next big thing on the list. Large test suites that run close to the hardware are very hard to tame. Some projects such as ezbench could become very helpful outside of its initial territory that is Intel graphics. But to start with, let us try to define the problem space and take a look at the state of the art in this area to then come up with ideas that apply to upstream kernel functional testing.

I agree to abide by the anti-harassment policy

Yes

Primary author: TUCKER, Guillaume (Collabora Limited)

Presenter: TUCKER, Guillaume (Collabora Limited)

Session Classification: Testing and Fuzzing MC

Contribution ID: **149** Type: **not specified**

RISC-V hypervisor implementation

Monday, 9 September 2019 12:15 (45 minutes)

The RISC-V hypervisor extension is carefully designed to be compliant with both Type-1 and Type-2 hypervisors. We have ported Xvisor (Type-1) and KVM (Type-2) for RISC-V architecture. In this session, we share our experience porting these hypervisors and also discuss future work on RISC-V hypervisors.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr PATEL, Anup (Western Digital)

Presenter: Mr PATEL, Anup (Western Digital)

Session Classification: RISC-V MC

Contribution ID: **150** Type: **not specified**

RISC-V Hypervisor ISA Emulation

Monday, 9 September 2019 12:15 (45 minutes)

This presentation discusses the work done to add the RISC-V Hypervisor Extension support to QEMU. This allows everyone to use QEMU as a development platform for porting Hypervisors to RISC-V. This can be seen by the recent effort to port KVM to RISC-V.

This presentation will discuss how the RISC-V Hypervisor extension works and how it is different to other common architectures Hypervisor support. It will talk about how the extension was implemented in QEMU and problems that were identified with the draft specification in the process. Finally it will conclude with the current upstream status and any pending work related to both QEMU and the RISC-V Hypervisor specification in general, including current Hypervisor project porting status.

We are also looking for feedback on existing issues in the RISC-V Hypervisor specification and possible solutions. This will help in making a more software friendly and robust specification.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr FRANCIS, Alistair

Presenter: Mr FRANCIS, Alistair

Session Classification: RISC-V MC

Contribution ID: **151** Type: **not specified**

Kernel Runtime Security Instrumentation (KRSI)

Wednesday, 11 September 2019 18:00 (20 minutes)

Existing Linux Security Modules can only be extended by modifying and rebuilding the kernel, making it difficult to react to new threats. The Kernel Runtime Security Instrumentation project (KRSI) (prototype code) aims to help this by providing an LSM that allows eBPF programs to be added to security hooks.

The talk discusses the need for such an LSM (with representative use cases) and compares it to some existing alternatives, such as Landlock, a separate custom LSM, kprobes+eBPF etc. The second half of the talk outlines the proposed design and interfaces, and includes a live demo.

KRSI is an LSM that:

- Allows the attachment of eBPF programs to security hooks.
- Provides a good ecosystem of safe eBPF helper functions specifically written with security and auditing features in mind.

This enables the development of a new class of userspace security products that:

- Reduce the overhead of building and updating the kernel/LSM when a new security vulnerability is discovered.
- Allows the system owners to choose the format in which the data is audit logged. Provide flexibility w.r.t granularity of auditing needed and add new auditing without needing to re-build or update the LSM/Kernel (in contrast to the existing audit framework)

The intended audience for this talk would be:

- Security-focused kernel engineers
- Engineers building user-space security products on Linux.
- Security Engineers and Admins who care about the time required to deploy security software to detect and prevent a new class of malicious activity.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr SINGH, KP

Presenter: Mr SINGH, KP

Session Classification: BPF MC

Contribution ID: **153** Type: **not specified**

Fighting uninitialized memory in the kernel

Tuesday, 10 September 2019 11:15 (15 minutes)

During the last two years, KMSAN (a detector of uses of uninitialized memory based on compiler instrumentation) has found more than a hundred bugs in the upstream kernel.
We'll discuss the current status of the tool, some of its findings and implementation challenges. Ideally, I'd like to get more people to look at the code, as finding bugs in particular subsystems may require deeper knowledge of those subsystems.
Another thing that'll be covered is the new stack and heap initialization features that will hopefully prevent most of the bugs related to uninitialized memory in the kernel.

I agree to abide by the anti-harassment policy

Yes

Primary author: POTAPENKO, Alexander (Google)

Presenter: POTAPENKO, Alexander (Google)

Session Classification: Testing and Fuzzing MC

Contribution ID: **154** Type: **not specified**

Linux Perf advancements for compute intensive and server systems

Tuesday, 10 September 2019 12:00 (45 minutes)

Modern server and compute intesive systems are naturally built around several top performance CPUs with large amount of cores and equipped by shared memory that spans a number of NUMA domains. Compute intensive workloads usually implement highly parallel CPU bound cyclic codes performing mathematics calculations that reference data located in the shared memory. Performance observability and profiling of these workloads on such systems have unique characteristics and impose specific requirements on software performance tools. The requirements include tools CPU scalability, coping with high rate and volume of collected performance data as well as NUMA awareness. In order to fulfill that requirements a number of extensions have been implemented in Linux Perf tool that are currently a part of the Linux kernel source tree:
https://marc.info/?l=linux-kernel&m=154149439404555&w=2,
https://marc.info/?l=linux-kernel&m=154149439404555&w=2,
https://marc.info/?l=linux-kernel&m=155293062518459&w=2 .

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: BUDANKOV, Alexey

Presenter: BUDANKOV, Alexey

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **155** Type: **not specified**

Multipath TCP Upstreaming

Monday, 9 September 2019 12:00 (45 minutes)

Multipath TCP (MPTCP) is an increasingly popular protocol that members of the kernel community are actively working to upstream. A Linux kernel fork implementing the protocol has been developed and maintained since March 2009. While there are some large MPTCP deployments using this custom kernel, an upstream implementation will make the protocol available on Linux devices of all flavors.

MPTCP is closely coupled with TCP, but an implementation does not need to interfere with operation of normal TCP connections. Our roadmap for MPTCP in Linux begins with the server use case, where connections and additional TCP subflows are generally initiated by peer devices. This will start with RFC 6824 compliance, but with a minimal feature set to limit the code footprint for initial review and testing.

The MPTCP upstreaming community has shared a RFC patch set on the netdev list that shows our progress and how we plan to build around the TCP stack. We'll share our roadmap for how this patch set will evolve before final submission, and discuss how this first step will differ from the forked implementation.

Once we have merged our baseline code, we have plans to continue development of more advanced features for managing subflow creation (path management), scheduling outgoing packets across TCP subflows, and other capabilities important for client devices that initiate connections. This includes making use of a userspace path manager, which has an alpha release available already. In future kernel releases we will make use of additional TCP features and optimize MPTCP performance as we get more feedback from kernel users.

Both the communication and the code are public and open. You can find us at mptcp@lists.01.org and https://is.gd/mptcp_upstream

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: MARTINEAU, Mat (Intel); BAERTS, Matthieu (Tessares)

Presenters: MARTINEAU, Mat (Intel); BAERTS, Matthieu (Tessares)

Session Classification: Networking Summit Track

Contribution ID: **156** Type: **not specified**

bpftrace

Monday, 9 September 2019 12:44 (22 minutes)

bpftrace is a high level tracing language running on top of BPF: https://github.com/iovisor/bpftrace

We'll talk about important updates from the past year, including improved tracing providers and new language features, and we'll also discuss future plans for the project.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr ROBERTSON, Alastair (Yellowbrick)

Presenter: Mr ROBERTSON, Alastair (Yellowbrick)

Session Classification: Tracing MC

Contribution ID: **158** Type: **not specified**

Security feature parity between GCC and Clang

Tuesday, 10 September 2019 11:00 (30 minutes)

There are many security features common to both GCC and Clang, but there is a growing set of features that are missing from GCC and present in Clang, missing from Clang and present in GCC, or missing in both. This session seeks to enumerate and discuss these areas, with the eye toward finding next steps forward (or at least elevating development priority).

Potential areas of focus:
- LTO (especially link speed)
- forward-edge CFI (software and hardware support)
- backward-edge CFI (software and hardware support)
- stack variable auto-initialization
- caller-saved register wipe on function return
- integer overflow detection
- stack clash protection
- implicit fall-through
- memory tagging

I agree to abide by the anti-harassment policy

Yes

Primary author: COOK, Kees (Google)

Presenter: COOK, Kees (Google)

Session Classification: Toolchains MC

Contribution ID: **161** Type: **not specified**

RCU internals and usage

Wednesday, 11 September 2019 15:00 (45 minutes)

This session will focus on answering questions on the internals and the usage of Linux-kernel RCU. However, questions regarding details of the RCU-related patches in the -rt patchset will be deferred to other venues, given that this topic consumed the entire time in the 2018 informal RCU BoF session.

This is not intended to be a tutorial on RCU basics, though a separate session on this topic might be offered if there is sufficient interest.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: MCKENNEY, Paul (IBM Linux Technology Center)

Presenter: MCKENNEY, Paul (IBM Linux Technology Center)

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **163** Type: **not specified**

Greybus for IoT

Monday, 9 September 2019 15:00 (30 minutes)

Greybus is an RPC like protocol on top UniPro bus that has been designed for the Project ARA. This goal of that project was to develop a modular smartphone. Greybus gives the ability to the host to control remotely the buses (such as i2c or spi) of the modules.
Although Project ARA has been aborted, Greybus has been merged to Linux kernel, and it is still maintained by the community.
Greybus has been designed for modular smartphones, but there are many others pertinents use cases for it:
- IoT, to let a Linux base station directly control sensors, and avoid writing complex firmware for the modules
- USB, to control peripherals on the board using existing Linux drivers
- To control system-on-chip hardware peripherals managed by a small core, with messages sent from a larger CPU This approach would be more generic that writing a custom protocol on top of RPMSG
The intent of this talk is to briefly present Greybus, how we could use it for general purpose, and talk about the work in progress, that would make it possible.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr BAILON, Alexandre (BayLibre)

Presenter: Mr BAILON, Alexandre (BayLibre)

Session Classification: You, Me, and IoT MC

Contribution ID: **165** Type: **not specified**

Beyond per-CPU atomics and rseq syscall: subset of eBPF bytecode for the do_on_cpu syscall

Wednesday, 11 September 2019 17:40 (20 minutes)

The Restartable Sequences system call [1,2,3,4] introduced in Linux 4.18 has limitations which can be solved by introducing a bytecode interpreter running in inter-processor interrupt context which accesses user-space data.

This discussion is about the subset of the eBPF bytecode and context needed by this interpreter, and extensions of that bytecode to cover load-acquire and store-conditional memory accesses, as well as memory barrier instructions. The fact that the interpreter needs to allow loading data from userspace (tainted data), which can then be used as address for loads and stores, as well as conditional branches source register, will also be discussed.

1 “PerCpu Atomics” http://www.linuxplumbersconf.org/2013/ocw/system/presentations/1695/original/LPC%20-%20PerCpu%20Atomics.pdf
[2] “Restartable sequences” https://lwn.net/Articles/650333/
[3] “Restartable sequences restarted” https://lwn.net/Articles/697979/
[4] “Restartable sequences and ops vectors” https://lwn.net/Articles/737662/

I agree to abide by the anti-harassment policy

Yes

Primary author: DESNOYERS, Mathieu (EfficiOS Inc.)

Presenter: DESNOYERS, Mathieu (EfficiOS Inc.)

Session Classification: BPF MC

Contribution ID: **166** Type: **not specified**

Wrapping system calls in glibc

Tuesday, 10 September 2019 10:30 (30 minutes)

The glibc project decided a while back that it wants to add wrappers for system calls which are useful for general application usage. However, that doesn't mean that all those missing system calls are added immediately.

System call wrappers still need documentation in the manual, which can be difficult in areas where there is no consensus how to describe the desired semantics (e.g., in the area of concurrency). Copyright assignment to the FSF is needed for both the code and the manual update, but can usually be performed electronically these days, and is reasonably straightforward. On top of that, the glibc project is seriously constrained by available reviewer bandwidth.

Some more specific notes:

Emulation of the system call is not required. It has been historically very problematic. The only thing that has not come back to bite us is checking if a new flag argument is zero and call the old, equivalent system call instead in this case.

Wrapper names should be architecture-independent if at all possible. Sharing system call names as much as possible between architectures in the UAPI headers helps with that.

Mutiplexing system calls are difficult to wrap, particularly if the types and number of arguments vary. Previous attempts to use varargs for this have led to bugs. For example, open/openat would not pass down the mode flag for O_TMPFILE initially, or cannot be called with a non-variadic prototype/function pointer on some architectures. We wouldn't want to wrap ipc or socketcall (even if they had not been superseded), and may wrap futex as separate functions.

We strongly prefer if a system call that is not inherently architecture-specific (e.g., some new VFS functionality) is enabled for all architectures in the same kernel release.

When it comes to exposing the system call, we prefer to use ssize_t or size_t for buffer sizes (even if the kernel uses int or unsigned int), purely for documentation purposes. Flag arguments should not be long int because it is unclear whether in the future more than 32 flags will be added on 64-bit architectures. Except for pthread_* functions, error reporting is based on errno and special return values.

Passing file offsets through off64_t * arguments is fine with us. Otherwise, off64_t parameter passing tends to vary too much.

If constants and types related to a particular system call are defined in a separate header which does not contain much else, we can include that from the glibc headers if available. As result, new kernel flags will become available to application developers immediately once they install newer kernel headers. This may not work for multiplexing system calls, of course, even if we wrap the multiplexer.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LEVIN, Dmitry (BaseALT)

Co-author: WEIMER, Florian

Presenters: LEVIN, Dmitry (BaseALT); WEIMER, Florian; ROZYCKI, Maciej W.

Session Classification: Toolchains MC

Contribution ID: **168** Type: **not specified**

Discussion about IBNBD/IBTRS Upstreaming: Action Items.

Wednesday, 11 September 2019 12:00 (30 minutes)

We are going through upstreaming IBNBD/IBTRS 5th iterations, the latest effort is here: https://lwn.net/Articles/791690/.

We would like to discuss in an open round about the unique features of the driver and the library, whether and how they are beneficial for the RDMA eco-system and what should be the next steps in order to get them upstream.

A face to face discussion about action items will smooth the path.

I agree to abide by the anti-harassment policy

Yes

Primary authors: Mr WANG, Jinpu (1 & 1 IONOS Cloud GmbH); Mr KIPNIS, Danil (1 & 1 IONOS Cloud GmbH)

Presenters: Mr WANG, Jinpu (1 & 1 IONOS Cloud GmbH); Mr KIPNIS, Danil (1 & 1 IONOS Cloud GmbH)

Session Classification: RDMA MC

Contribution ID: **170** Type: **not specified**

Over the Air (OTA) Updates: State of the Union? Democratize?

Monday, 9 September 2019 15:30 (30 minutes)

IoT applications, be they Autonomous Cars 1 or Health Care or Smart Home or Factory Automation, the IoT devices (sensors and actuators), gateways, and cloud/datacenter endpoints need software and/or firmware updates, to fix security issues, patch bugs, and/or release new features. IoT with its numerous remote devices and gateways presents a large attack surface, making the application of security patches as they become available especially important. Let us review key OTA Update requirements, available open source solutions, and how to ease adoption through the introduction of a standard API, one that abstracts the complexities and trade-offs. The underlying implementation would be selected based on the application needs and where in IoT architectural stack a given node lies (device/edge/cloud).

Most of us are familiar with OTA in the context of our mobile phones. A large proportion of Tesla's success and customer confidence stems from its OTA update support [2], for example a braking distance issue fix, accepting only signed updates, and rolling out new features.

What are key OTA requirements [3]?
1) Ability to upgrade the bootloader, kernel, root filesystem, firmware, applications, device specific data.
2) Robust - Never "brick" the device.
3) Atomic - success or failure, nothing partial with undefined behavior.
4) Automated – not requiring human interaction during the process
5) Auditable – logs – what got updated
6) Preserve user data (customizations etc)
7) Signed, accepting updates only from trusted sources.
8) Secure communication channel.

Note: We shall drop the bootloader in item 1 because it is a transient power-on, process is rarely a source of runtime bugs.

What are OTA implementation considerations [4]?
Inline or Shadow Partition?
OTA is not easy and there are many implementation options with their respective trade-offs. Should the update happen in-place or use a shadow partition to copy over? What size should the partition be? The shadow partition approach is certainly safer just in case there is a power glitch at either the recipient or server node, or a network glitch or some other error condition that could corrupt an in-place update. Roll-back in case of a corrupted update is easy with shadow partitions because the original boot image is intact.
Block-based or file update?
The former is a complete image, easy to verify with a version number and hash signature, making it simpler to manage across a fleet of devices. The latter is more concise but should issues arise in the application of the patch, the system could become inoperable.
Trusted Source?
Can the update payload be trusted? Is it signed by a trusted entity? This requires the nodes have the public key and certificate of the trusted entity.
Where can updates be obtained?
Perhaps a vendor specific site. Possibly even a public site if open source.
Update Push or Pull? Frequency?
Should nodes poll for updates or should a management application push updates to nodes?
Secure transmission?
Is the transmission channel secured using TLS/HTTPS or over a VPN?

What open source projects address OTA?
The projects below vary in their robustness, network bandwidth needs, and the types of hardware they support.

OSTree [6,7]
Provides a git like approach to version control for Linux operating systems that does an atomic complete filesystem update. The userspace solution can operate either standalone or be layered with a package manager for a hybrid solution.

Balena.io [8]
balenaOS Yocto Linux based host OS that comes packaged with balenaEngine, a lightweight docker-compatible container engine. A device supervisor runs in its own container and allows pulling new code even if the application code crashes.

SWUpdate [9,10]
SWUpdate is a Linux Update agent with the goal to provide an efficient and safe way to update an embedded system. SWUpdate supports local and remote updates, multiple update strategies and it can be well integrated in the Yocto build system by adding the meta-swupdate layer. Supports updating FPGAs and Microcontrollers.

Swupd [11]
swupd is an operating system software manager and update program that operates at a file-level to enable verifiable integrity and update efficiency.

Mender.io [12]
An open source update manager for embedded devices based on the client-server model with security and robustness.

How can we make OTA Update Easier?

Linux is the King/Queen of IoT, be it on small form factor highly resource constrained devices or on server class gateways. The OTA implementation depends upon the node, whose selection depends upon the demands of the use case. What if we could abstract away the nuances of the implementation and ease consumption, along the lines of Libvirt [13] for virtual machines that abstracts away for Cloud orchestrators machine architecture (ARM, X86) and hypervisor implementation (KVM/Xen/ESXi/HyperV/ACRN). What if we introduce "update" akin to reboot, with configuration and action sub-commands?

```
update config source <source-url>
update config key  <add|delete|update> <name> <public-key>
update config schedule <monthly|hourly|minute> <integer>
update config log <log path> // defaults to syslog
update config verify [true|false] // verifies signature publickey
update config boot-retry-limit <integer>
update config secure [true|false]  // mutual authentication [14]
update [--secure [true|false]] [--verify [true|false]] [--source <source-url>] [--now] [--noreb
   // values specified override config settings
   // typically reboot after update
```

Should no update implementation exist, these methods should gracefully fail reporting an error to the default log location. An update implementation when installed overwrites the default methods, which typically report "Not implemented. Consider installing X, Y or Z"

Future Enhancements:
We defer for the future supporting more secure WiFi access for IoT and OTA such as wpa3 [15]. Also in this vein is use of secure storage media such as self-encrypting-drives and read-only memory [16].

References:

1. https://www.slideshare.net/leonanavi/software-over-the-air-sota-for-automotive-grade-linux-agl
2. https://electrek.co/2017/07/17/tesla-fleet-hack-elon-musk/

3. https://mender.io/learn/whitepapers/_resources/Software%20Updates.pdf
4. https://www.embedded.com/design/operating-systems/4461019/OTA-updates-for-Embedded-Linux–part-1—–Fundamentals-and-implementation
5. https://elinux.org/Secure_OTA_Update
6. https://ostree.readthedocs.io/en/latest/manual/introduction/
7. https://samthursfield.wordpress.com/2014/01/08/os-level-version-control/
8. https://www.balena.io/what-is-balena/
9. https://github.com/sbabic/swupdate
10. http://events17.linuxfoundation.org/sites/events/files/slides/ELC2017_SWUpdate.pdf
11. https://clearlinux.org/documentation/clear-linux/concepts/swupd-about
12. https://mender.io
13. https://libvirt.org
14. https://searchsecurity.techtarget.com/definition/mutual-authentication
15. https://www.linux.com/news/wpa3-how-and-why-wi-fi-standard-matters
16. https://openiotelcna2017.sched.com/event/9J5i/surviving-in-the-wilderness-integrity-protection-and-system-update-patrick-ohly-intel-gmbh

I agree to abide by the anti-harassment policy

Yes

Primary author: Dr BHANDARU, Malini (VMware)

Presenter: Dr BHANDARU, Malini (VMware)

Session Classification: You, Me, and IoT MC

Contribution ID: **171** Type: **not specified**

User interfaces for per-group default domain type

Monday, 9 September 2019 15:00 (25 minutes)

This topic will discuss 1) why do we need per-group default domain type, 2) how it solves the problems in the real IOMMU driver, and 3) the user interfaces.

I agree to abide by the anti-harassment policy

Yes

Primary author: LU, Baolu

Presenter: LU, Baolu

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **172** Type: **not specified**

Upstream 1st: Tools and workflows for multi kernel version juggling of short term fixes, long term support, board enablement and features with the upstream kernel

Monday, 9 September 2019 10:00 (20 minutes)

Having maintained a distribution agnostic reference kernel (Yocto), an operating system vendor kernel (Wind River) and finally a semi-conductor kernel (Xilinx), there are a lot of obvious workflows and tools that are used to deliver kernels and support them after release.

The less than obvious workflows (and tools) are often related to distro kernel tree maintenance and balancing the needs of short term fixes (often security related), with a model that allows long term support, all in trees that may be carrying specific features or board support that are destined for upstream eventually. Many methods to juggle these demands are ad-hoc or specific to the various distros.

If a tree is not (somewhat) history clean, and patch history is not tracked over time, moving to a new kernel version, understanding why a change was made or debugging a problem are made much harder.

All the competing demands are coupled with the need to have development supported with the goal of getting changes into the mainline kernel. Understanding the technical solutions (tools), workflows (tools + social) and how to support the community at large to reduce everyone's workload is often given limited time. Stepping back and looking at the different solutions that maintainers are using may highlight common patterns and opportunities to collaborate/standardize on various techniques. Less-than-ideal solutions are also valuable as lessons learned and are worth sharing.

I agree to abide by the anti-harassment policy

Yes

Primary author: ASHFIELD, Bruce (Xilinx)

Presenter: ASHFIELD, Bruce (Xilinx)

Session Classification: Distribution Kernels MC

Contribution ID: **174** Type: **not specified**

Distros and Syzkaller - Why bother?

Monday, 9 September 2019 13:00 (30 minutes)

Syzkaller is run on Upstream and Stable trees. When paired with KASAN it has proven its usefulness uncovering large numbers of Out-of-Bounds (OOB) and Use-after-free (UAF) bugs. These results are readily available on the syzbot dashboard. What do distros gain by running Syzkaller?

Distros regularly add features to their kernels, fix bugs and add third party drivers. Syzkaller testing focused on these changes and additions can uncover bugs and detect regressions.

Syzkaller can be part of a distro's continuous integration (CI) strategy. Dedicated Syzkaller CI servers can be running the distro's next release candidate, only being halted and restarted as features, bug fixes or third party drivers are added.

How can distros collaborate? There are many third party drivers common to all distros. Distros can collaborate on the Syzkaller testing framework for these drivers. Likewise for features that are going Upstream.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr KENNEDY, George

Session Classification: Distribution Kernels MC

Contribution ID: **175** Type: **not specified**

RCU configuration, operation, and upcoming changes for real-time workloads

Wednesday, 11 September 2019 10:30 (30 minutes)

RCU has changed a surprising amount over the past few years, what with elimination of many RCU Kconfig options in favor of kernel boot parameters, RCU flavor consolidation, ongoing work on speeding up RCU's handling of offloaded callbacks, and newly started work on providing warnings when RCU's callback handling is overloaded. These changes affect how RCU behaves, and in some cases in ways that affect realtime usage. This talk will summarize the changes relevant to realtime, and outline how this affects configuration and tuning of RCU.

I agree to abide by the anti-harassment policy

Yes

Primary author: MCKENNEY, Paul (IBM Linux Technology Center)

Presenter: MCKENNEY, Paul (IBM Linux Technology Center)

Session Classification: Real Time MC

Contribution ID: **177** Type: **not specified**

PCI Resources assignment policies

Monday, 9 September 2019 17:35 (25 minutes)

This is meant to be a rather open discussion on PCI resource assignment policies. I plan to discuss a bit what the different arch/platforms do today, how I've tried to consolidate it, then we can debate the pro/cons of the different approaches and decide where to go from there.

I agree to abide by the anti-harassment policy

Yes

Primary author: HERRENSCHMIDT, Benjamin (Amazon AWS)

Presenter: HERRENSCHMIDT, Benjamin (Amazon AWS)

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **178** Type: **not specified**

Making SCHED_DEADLINE safe for kernel kthreads

Monday, 9 September 2019 16:00 (30 minutes)

Dmitry Vyukov's testing work identified some (ab)uses of sched_setattr() that can result in SCHED_DEADLINE tasks starving RCU's kthreads for extended time periods, not millisecond, not seconds, not minutes, not even hours, but days. Given that RCU CPU stall warnings are issued whenever an RCU grace period fails to complete within a few tens of seconds, the system did not suffer silently. Although one could argue that people should avoid abusing sched_setattr(), people are human and humans make mistakes. Responding to simple mistakes with RCU CPU stall warnings is all well and good, but a more severe case could OOM the system, which is a particularly unhelpful error message.

It would be better if the system were capable of operating reasonably despite such abuse. Several approaches have been suggested.

First, sched_setattr() could recognize parameter settings that put kthreads at risk and refuse to honor those settings. This approach of course requires that we identify precisely what combinations of sched_setattr() parameters settings are risky, especially given that there are likely to be parameter settings that are both risky and highly useful.

Second, in theory, RCU could detect this situation and take the "dueling banjos" approach of increasing its priority as needed to get the CPU time that its kthreads need to operate correctly. However, the required amount of CPU time can vary greatly depending on the workload. Furthermore, non-RCU kthreads also need some amount of CPU time, and replicating "dueling banjos" across all such Linux-kernel subsystems seems both wasteful and error-prone. Finally, experience has shown that setting RCU's kthreads to real-time priorities significantly harms performance by increasing context-switch rates.

Third, stress testing could be limited to non-risky regimes, such that kthreads get CPU time every 5-40 seconds, depending on configuration and experience. People needing risky parameter settings could then test the settings that they actually need, and also take responsibility for ensuring that kthreads get the CPU time that they need. (This of course includes per-CPU kthreads!)

Fourth, bandwidth throttling could treat tasks in other scheduling classes as an aggregate group having a reasonable aggregate deadline and CPU budget. This has the advantage of allowing "abusive" testing to proceed, which allows people requiring risky parameter settings to rely on this testing. Additionally, it avoids complex progress checking and priority setting on the part of many kthreads throughout the system. However, if this was an easy choice, the SCHED_DEADLINE developers would likely have selected it. For example, it is necessary to determine what might be a "reasonable" aggregate deadline and CPU budget. Reserving 5% seems quite generous, and RCU's grace-period kthread would optimally like a deadline in the milliseconds, but would do reasonably well with many tens of milliseconds, and absolutely needs a few seconds. However, for CONFIG_RCU_NOCB_CPU=y, the RCU's callback-offload kthreads might well need a full CPU each! (This happens when the CPU being offloaded generates a high rate of callbacks.)

The goal of this proposal is therefore to generate face-to-face discussion, hopefully resulting in a good and sufficient solution to this problem.

I agree to abide by the anti-harassment policy

Yes

Primary author: MCKENNEY, Paul (IBM Linux Technology Center)

Presenter: MCKENNEY, Paul (IBM Linux Technology Center)

Session Classification: Scheduler MC

Contribution ID: **179** Type: **not specified**

Present state of Linux DRM Subsystem Interfaces

Today every modern multimedia supported SoC's comprises of variety of display controller interfaces bounded with LCD panels or bridges and a GPU, for providing feasible display acceleration.

The Linux kernel handle all these display controller interfaces with associated panels, bridges via DRM subsystem, but it becomes a daunting task for many of the display users to make use of this DRM stack due to lack of technical documentation and guidelines for their vendor specific panels with bounded display controllers.

So, this talk will address those issues and challenges by starting a brief explanation of Linux DRM subsystem with associated display controller interfaces like HDMI, RGB, LVDS and DSI. After that the talk will cover the key factors that required while bringing up vendor defined solutions to make use of mainline DRM subsystem.

This talk makes use of real time challenges that have been observed while working with Allwinner Display controllers with variety of associated LCD panels, bridges which are validated via ARM Mali GPU.

I agree to abide by the anti-harassment policy

Yes

Primary author: TEKI, Jagan

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **182** Type: **not specified**

Monitoring and Stabilizing the In-Kernel ABI

Monday, 9 September 2019 11:00 (30 minutes)

The Kernel's API and ABI exposed to Kernel modules is not something that is usually maintained in upstream. Deliberately. In fact, the ability to break APIs and ABIs can greatly benefit the development. Good reasons for that have been stated multiple times. See e.g. Documentation/process/stable-api-nonsense.rst.
The reality for distributions might look different though. Especially - but not exclusively - enterprise distributions aim to guarantee ABI stability for the lifetime of their released kernels while constantly consuming upstream patches to improve stability and security for said kernels. Their customers rely on both: upstream fixes and the ability to use the released kernels with out-of-tree modules that are compiled and linked against the stable ABI.

In this talk I will give a brief overview about how this very same requirement applies to the Kernels that are part of the Android distribution. The methods presented here are reasonable measures to reduce the complexity of the problem by addressing issues introduced by ABI influencing factors like build toolchain, configurations, etc.

While we focus on Android Kernels, the tools and mechanisms are generally useful for Kernel distributors that aim for a similar level of stability. I will talk about the tools we use (like e.g. libabigail), how we automate compliance checking and eventually enforce ABI stability.

I agree to abide by the anti-harassment policy

Yes

Primary author: MAENNICH, Matthias (Google)

Presenter: MAENNICH, Matthias (Google)

Session Classification: Distribution Kernels MC

Contribution ID: **183** Type: **not specified**

Real time softirq mainlining

Wednesday, 11 September 2019 12:00 (30 minutes)

Which Real Time softirq implementation do we want for mainline?

_ Vector-Lock based? (depend on sleeping spinlocks machinery)
_ Vector masking based?
_ Other?

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: WEISBECKER, Frederic (Suse)

Presenter: WEISBECKER, Frederic (Suse)

Session Classification: Real Time MC

Contribution ID: **184** Type: **not specified**

Full dynticks / isolation for Real Time

Wednesday, 11 September 2019 12:30 (30 minutes)

_ What is needed upstream for real time support of Full Dynticks and isolation?
_ Specific requests?

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: WEISBECKER, Frederic

Presenter: WEISBECKER, Frederic

Session Classification: Real Time MC

Contribution ID: **185** Type: **not specified**

Status of Dual Stage SMMUv3 integration

Monday, 9 September 2019 15:30 (25 minutes)

Since August 2018 I have been working on SMMUv3 nested stage integration at IOMMU/VFIO levels, to allow virtual SMMUv3/VFIO integration.

This shares some APIs with the Intel and ARM SVA series (cache invalidation, fault reporting) but also introduces some specific ones to pass information about guest stage 1 configuration and MSI bindings.

In this session I would like to discuss the upstream status and get a chance to clarify open points. This is also an opportunity to synchronize about the VFIO fault reporting requirements for recoverable errors.

I agree to abide by the anti-harassment policy

Yes

Primary author: AUGER, Eric (Red Hat)

Presenter: AUGER, Eric (Red Hat)

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **186** Type: **not specified**

A trace-cmd front end interface to ftrace histogram, triggers and synthetic events.

Monday, 9 September 2019 11:06 (22 minutes)

Ftrace histograms, based on triggers and synthetic events were implemented few years ago by Tom Zanussi. They are very powerful instrument for analyzing the kernel internals, using ftrace events, but its user interface is very complex and hard to use. This proposal is to discuss possible ways to define more easy to use and intuitive interface to this feature, using trace-cmd application.

I agree to abide by the anti-harassment policy

Yes

Primary author: STOYANOV, Tzvetomir

Presenter: STOYANOV, Tzvetomir

Session Classification: Tracing MC

Contribution ID: **187** Type: **not specified**

CPU Idle Time Management Improvements

Tuesday, 10 September 2019 18:15 (25 minutes)

There are some improvements in the CPU idle time management to be made, like switching over to using time in nanoseconds (64-bit), reducing overhead and some governor modifications (including possible deprecation of the menu governor) which need to be discussed.

I agree to abide by the anti-harassment policy

Yes

Primary author: WYSOCKI, Rafael (Intel Open Source Technology Center)

Presenter: WYSOCKI, Rafael (Intel Open Source Technology Center)

Session Classification: Power Management and Thermal Control MC

Contribution ID: **188** Type: **not specified**

CRIU and the PID dance

Tuesday, 10 September 2019 15:10 (20 minutes)

CRIU only restores processes with the same PID the processes used to have during checkpointing. As there is no interface to create a process with a certain PID like fork_with_pid() CRIU does the **PID dance** to restore the process with the same PID as before checkpointing.

The PID dance consists of open()ing /proc/sys/kernel/ns_last_pid, write()ing PID-1 to /proc/sys/kernel/ns_last_pid and close()ing it. Then CRIU does a clone() and a getpid() to see if the clone() resulted in the desired PID. If the PID does not match, CRIU aborts the restore.

This PID dance is slow, racy and requires CAP_SYS_ADMIN.

Fortunately the newly introduced clone3() offers the possibility to be extended to support clone3() with a certain/desired PID. There are currently (July 2019) discussions how to extend clone3() to be able to use it with a certain PID. By the time LPC has started these patches will probably be already posted. With these patches it should be possible to solve the problems that the PID dance is slow and racy.

Which leaves the problem of CAP_SYS_ADMIN. This is a problem for CRIU because it is the major reason why CRIU needs to be run as root during restore. If the root and CAP_SYS_ADMIN requirement could be somehow relaxed it would solve the problems for people running CRIU as non-root for container migration as reported during last year's LPC and it would also open up easy CRIU usage in areas like HPC with MPI based checkpointing and restoring running as non-root.

In this talk we want to give some background how and why CRIU does the PID dance, we want to present our changes based on clone3() to be able to create a process with a certain PID. Then we would like to get feedback from the community if a rootless restore is important and how to relax the CAP_SYS_ADMIN requirement and how this relaxation could be implemented.

I agree to abide by the anti-harassment policy

Yes

Primary author: REBER, Adrian (Red Hat)

Presenter: REBER, Adrian (Red Hat)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **191** Type: **not specified**

Time series of thread profiles in production

Wednesday, 11 September 2019 12:27 (15 minutes)

At MongoDB, we implemented an eBPF tool to collect and display a complete time-series view of information about all threads whether they are on- or off-CPU. This allows us to inspect where the database server spends its time, both in userspace and in kernel. Its minimal overhead allows to deploy it in production.

This can be an effective method to collect diagnostic information in the field and surface a specific workload which is bound by a syscall. It would be interesting to hear what solution other vendors use to profile in production.

I agree to abide by the anti-harassment policy

Yes

Primary author: AHMAD, Josef (MongoDB Inc.)

Presenter: AHMAD, Josef (MongoDB Inc.)

Session Classification: Databases MC

Contribution ID: **193** Type: **not specified**

Using Greybus, mikroBus and PocketBeagle to consolidate kernel IoT sensor/actuator development

Monday, 9 September 2019 18:00 (30 minutes)

Many "*drivers*" for IoT sensors and actuators live outside kernel space through efforts that seek to provide abstractions not sufficiently handled in the kernel today. This is resulting in great code fragmentation that can be resolved by better understanding the developer needs and communicating an achievable collaborative approach. Pushing the interface to these devices off to userspace is not the Linux-way.

We'll look at the problems projects like MRAA/UPM, Adafruit_Blinka and numerous other projects from IoT tooling and breakout board providers are seeking to solve outside the kernel. These include providing libraries that support a very broad array of sensors, that help build understanding of the sensors themselves, make it easy to augment sensor parameters, and, at least for Adafruit_Blinka, include running the same interface code on microcontrollers.
Using these userspace libraries also aid in rapid prototyping by avoiding the step of configuring the kernel to use these sensors on non-probable busses.

Using Greybus, it is possible to, in a more flexible and secure way than device tree overlays, add IoT sensors in a rapid-prototyping fashion. See

Using mikroBus, it is possible to collaborate across a large number of embedded Linux development platforms across a large number of IoT sensors and actuators. This is at least partially thanks to the almost 700 different available Click boards and a number of add-on daughter boards for embedded Linux development boards to interface to them.

I agree to abide by the anti-harassment policy

Yes

Primary authors: KRIDNER, Jason (BeagleBoard.org); FUSTINI, Drew (OSH Park)

Presenters: KRIDNER, Jason (BeagleBoard.org); FUSTINI, Drew (OSH Park)

Session Classification: You, Me, and IoT MC

Contribution ID: **194** Type: **not specified**

Implementing NTB controller using PCIe endpoint

Monday, 9 September 2019 18:05 (15 minutes)

A PCI-Express non-transparent bridge (NTB) is a point-to-point PCIe bus connecting 2 host systems. NTB functionality can be achieved in a platform having 2 endpoint instances. Here each of the endpoint instance will be connected to an independent host and the hosts can communicate with each other using endpoint as a bridge. The endpoint framework and the "new" NTB EP function driver should configure the endpoint instances in such a way that the transactions from one endpoint is routed to the other endpoint instance. The host will see the connected endpoint as an NTB port and the existing NTB tools (ntb_pingpong, ntb_perf) in Linux kernel could be used.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr I, Kishon Vijay Abraham

Presenter: Mr I, Kishon Vijay Abraham

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **195** Type: **not specified**

Architecture considerations for vfio/iommu handling

Monday, 9 September 2019 16:30 (15 minutes)

While x86 is probably the most prominent platform for vfio/iommu development and usage, other architectures also see quite a bit of movement. These architectures are similar to x86 in some parts and quite different in others; therefore, sometimes issues come up that may be surprising to folks mostly working on more common platforms.

For example, PCI on s390 is using special instructions. QEMU needs to fill in 'real' values for some memory-layout values for devices passed via vfio and needs a way to retrieve them.

Other architectures (e.g. ARM) may also have some unusual requirements not obvious to people not working on those platforms. It seems beneficial to at least raise awareness of those issues so that we don't end up with interfaces/designs that are hard to implement or not sufficient on less common platforms.

I agree to abide by the anti-harassment policy

Yes

Primary author: HUCK, Cornelia

Presenter: HUCK, Cornelia

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **198** Type: **not specified**

Multiple thermal zones representation

Tuesday, 10 September 2019 15:00 (25 minutes)

The current design of the thermal framework forces the usage of a governor with a thermal zone thus limiting the scope of the decisions.
The question of the multiple thermal zones representation and how they are handled by a governor was put several times on the table but without a clear consensus.
In order to go forward in this area, this MC topic proposes a simple design with a hierarchical thermal zones representation and how they can be managed by a governor. The design keeps the compatibility with the current flat representation.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LEZCANO, Daniel (Linaro)

Presenter: LEZCANO, Daniel (Linaro)

Session Classification: Power Management and Thermal Control MC

Contribution ID: **199** Type: **not specified**

CFS load balance rework

Monday, 9 September 2019 17:00 (30 minutes)

The cfs load_balance has became more and more complex over the years and has reached the point where policy can't be explained sometimes. Furthermore, available metrics have evolved and load balance doesn't always take full advantage of it to calculate the imbalance. It's probably the good time to do a rework of the load balance code as proposed in this patchset:
https://lkml.org/lkml/2019/7/19/594
In addition to this patchset , we could discuss the next evolution that could be done on the load_balance

I agree to abide by the anti-harassment policy

Yes

Primary author: GUITTOT, Vincent (Linaro)

Presenter: GUITTOT, Vincent (Linaro)

Session Classification: Scheduler MC

Contribution ID: **201** Type: **not specified**

C-state latency measurement infrastructure

Tuesday, 10 September 2019 17:50 (25 minutes)

We in Intel developed instrumentation for measuring C-state wake latency. The instrumentation, which we call “waltr” (WAke up Latency Tracer) consists of user-space and kernel modules parts.

In principle, waltr works by scheduling delayed interrupts and measuring the wake latency close to the x86 ‘mwait’ x86 instruction. This requires an external device equipped with high precision clock and capable of delayed interrupts. We have been
using the Intel i210 Ethernet adapter for these purposes. But theoretically this
could be a completely different device, e.g., a GFX card.

The C-state latency measurement instrumentation should be very useful for the open-source community and we would like to upstream the kernel parts of it. We are seeking for feedback on how to properly modify the kernel in a maintainable and reusable way, to benefit everyone.

Here are few examples for the dilemmas have.
* How do we design a framework for compliant devices like the i210 adapter?
* What would be the right user-space API for the delayed interrupts provider?
* How do we take snapshots of C-state counters and deliver them to user-space?

I am asking for a 20-30 minutes time-slot. And I am hoping to talk to people more about this in hallway discussions.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: BITYUTSKIY, Artem

Presenter: BITYUTSKIY, Artem

Session Classification: Power Management and Thermal Control MC

Contribution ID: **202** Type: **not specified**

Syscall overhead from Spectre/Meltdown fixes

Wednesday, 11 September 2019 13:12 (10 minutes)

users are very worry about any kind of overhead due kernel patches applied to solve Intel CPU issues (Spectre/Meltdown/etc.) – what others are observing? what kind of workloads / test cases do you use for evaluation?

I agree to abide by the anti-harassment policy

Yes

Primary author: KRAVTCHUK, Dimitri

Presenter: KRAVTCHUK, Dimitri

Session Classification: Databases MC

Contribution ID: **203** Type: **not specified**

New InnoDB REDO log design and MT sync challenges

Wednesday, 11 September 2019 12:42 (15 minutes)

since MySQL 8.0 we have a newly redesigned lock-free REDO log implementation. However, this development involved several questions about overall efficiency around MT communications and synchronization. Curiously spinning on CPU showed to be the most efficient on low load.. – but any plans to implement “generic” MT framework for more efficient execution of any MT apps ?

I agree to abide by the anti-harassment policy

Yes

Primary authors: Mr OLCHAWA, Pawel; KRAVTCHUK, Dimitri

Presenter: Mr OLCHAWA, Pawel

Session Classification: Databases MC

Contribution ID: **204** Type: **not specified**

Regressions due CPU cache issues and missed visibility in Linux/kernel instrumentation

Wednesday, 11 September 2019 13:12 (10 minutes)

all MT apps are extremely sensible to CPU cache issues, and MySQL/InnoDB is part of them.. Several times we observed significant regressions (up to 40% and more) due CPU cache miss or simple cache sync due concurrent access to the same variable by several threads, and all “perf” CPU related stats did not show any difference.. Any plans to address it with more deep CPU stats instrumentation?

I agree to abide by the anti-harassment policy

Yes

Primary authors: Mr OLCHAWA, Pawel; KRAVTCHUK, Dimitri

Presenter: Mr OLCHAWA, Pawel

Session Classification: Databases MC

Contribution ID: **205** Type: **not specified**

io_uring - excitement - looking for feedback & potential issues

Wednesday, 11 September 2019 10:05 (15 minutes)

many devs are excited about the progress reported on this new stuff, but is it followed / considered by kernel devs.? what kind of gain to expect? any potential issues or feedback to share?

I agree to abide by the anti-harassment policy

Yes

Primary author: KRAVTCHUK, Dimitri

Presenter: KRAVTCHUK, Dimitri

Session Classification: Databases MC

Contribution ID: **206** Type: **not specified**

Filesystem atomic writes / O_ATOMIC

Wednesday, 11 September 2019 10:40 (15 minutes)

seems like the patches proposed by Fusion-io devs for general O_ATOMIC support within Linux kernel are in stand-by since 6 years.. – any plans to address it ?.. What is the main reason to not guarantee atomicy of O_DIRECT writes on flash drives? – seems like most of flash storage vendors are able to provide atomic writes support on HW level, and just SW level (kernel/FS/etc.) is missed.. The main benefit for MySQL/InnoDB is to get a rid of "double write" to protect from data corruption (partially written pages) – so, every page is written twice, increasing IO write traffic + doubling page write latency + reducing by half flash drive life expectation..

I agree to abide by the anti-harassment policy

Yes

Primary author: KRAVTCHUK, Dimitri

Presenter: KRAVTCHUK, Dimitri

Session Classification: Databases MC

Contribution ID: **207** Type: **not specified**

MySQL @EXT4 performance impacts with latest Linux kernels

Wednesday, 11 September 2019 10:55 (20 minutes)

since newer kernels (4.14, 5.1, ..) we are observing 50% regression on MySQL IO-bound workloads using EXT4 comparing to the same results on the same HW, but running kernel 3.x or 4.1. Unfortunately we have absolutely no explanation for this regression right now and looking for any available FS layer instrumentation/visibility to understand what is the root problem for such a regression and how it can be by-passed from MySQL code (or fixed if the problem is in EXT4)..

(more details are expected up to conference date)

I agree to abide by the anti-harassment policy

Yes

Primary author: KRAVTCHUK, Dimitri

Presenter: KRAVTCHUK, Dimitri

Session Classification: Databases MC

Contribution ID: **208** Type: **not specified**

MySQL @XFS

Wednesday, 11 September 2019 11:15 (15 minutes)

historically XFS was always showing lower performance comparing to EXT4 on most of IO-bound workloads used for MySQL/InnoDB benchmark testing.. However, since the new kernels & XFS arrived, we observed significantly better results on XFS now -vs- EXT4 particularly when InnoDB “double write” is enabled. From the other side, for our big surprise, XFS was doing worse if “double write” was disabled (which is nonsense, because how overall performance can be worse if we do twice less IO writes on the same IO-bound workload?) – fortunately we found a workaround to by-pass this issue, but still lacking deep understanding of the problem and observation/ visibility details from XFS layer).. – all is looking like a kind of IO starvation, but how it can be detected on time and ahead?..

(more details are expected up to conference date)

I agree to abide by the anti-harassment policy

Yes

Primary author: KRAVTCHUK, Dimitri

Presenter: KRAVTCHUK, Dimitri

Session Classification: Databases MC

Contribution ID: **209** Type: **not specified**

IP port -vs- UNIX socket difference on - IP stack is 20-30% slower on MySQL

Wednesday, 11 September 2019 12:57 (15 minutes)

MySQL is allowing user sessions connections via IP port and UNIX socket on Linux systems. However, curiously connecting via UNIX socket is delivering up to 30% higher performance comparing to IP local port (loopback).. – any reason for this? and be "loopback" code improved to match the same level of efficiency as UNIX socket? can the same improvements make over all IP stack to be more efficient?

I agree to abide by the anti-harassment policy

Yes

Primary author: KRAVTCHUK, Dimitri

Presenter: KRAVTCHUK, Dimitri

Session Classification: Databases MC

Contribution ID: **210** Type: **not specified**

IP / UNIX Socket Backlog

Wednesday, 11 September 2019 12:57 (15 minutes)

there is “backlog” option used in MySQL for both IP and UNIX sockets, but seems like it has a significant overhead on heavy connect/disconnect activity workloads (e.g. like most of Web apps which are doing “connect; SQL query; disconnect”) – any explanation/ reason for this? can it be improved?

I agree to abide by the anti-harassment policy

Yes

Primary author: KRAVTCHUK, Dimitri

Presenter: KRAVTCHUK, Dimitri

Session Classification: Databases MC

Contribution ID: **212** Type: **not specified**

Optional or reduced PCI BARs

Monday, 9 September 2019 17:05 (25 minutes)

Modern PCI graphics devices may contain several gigabytes of memory mapped in its BAR. This trend is continuing into storage with NVMe devices containing large Controller Memory Buffers and Persistent Memory Regions.

Some PCI hierarchies are resource constrained and cannot fit as many devices as desired. In NVMe's case, it's preferable to enumerate and attach all devices rather than use the entire memory window for one or two devices with large, optional BARs.

Current PCI core architecture will prevent a PCI device from being enabled if any of the BARs are unset. This proposal is about a way to hint at the PCI layer that some BARs are optional and could be omitted or reduced (by limiting it at the bridge window) in order to keep such devices enabled.

I agree to abide by the anti-harassment policy

Yes

Primary author: DERRICK, Jonathan

Presenter: DERRICK, Jonathan

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **214** Type: **not specified**

GWP-ASAN

Tuesday, 10 September 2019 10:55 (20 minutes)

In this talk Dmitry will introduce the idea of GWP-ASAN, a sampling tool that finds use-after-free and heap-buffer-overflows bugs in production environments. GWP-ASan supplements the normal slab allocator and chooses random allocations to 'sample'. These sampled allocations are placed into a special guarded pool, which is based upon the traditional 'Electric Fence Malloc Debugger' idea. Dmitry will share experiences of using such tool in user-space and speculate about how useful such tool would be for kernel.

I agree to abide by the anti-harassment policy

Yes

Primary author: VYUKOV, Dmitry (Google)

Presenter: VYUKOV, Dmitry (Google)

Session Classification: Testing and Fuzzing MC

Contribution ID: **215** Type: **not specified**

syzbot: update and open problems

Tuesday, 10 September 2019 12:00 (20 minutes)

In this talk, Dmitry will share updates on syzkaller/syzbot since last year: USB fuzzing, bisection, memory leaks. Talk about open problems: testability of kernel components; test coverage; syzbot process.

I agree to abide by the anti-harassment policy

Yes

Primary author: VYUKOV, Dmitry (Google)

Presenter: VYUKOV, Dmitry (Google)

Session Classification: Testing and Fuzzing MC

Contribution ID: **217** Type: **not specified**

Improving producer-consumer type workload performance

Tuesday, 10 September 2019 16:15 (25 minutes)

When each CPU core can independently control its performance states, then there is performance loss on some benchmarks compared to the case when there are no independent performance states. There are couple of options to indicate to the cpufreq drivers when a producer thread wakes a consumer thread: One sending some hints like we do for IO boost or give boost PELT utilization. But there is a challenge in cleanly identifying a producer/consumer relationship in scheduler code. There are several ways a thread can wait and get signaled to wake in Linux.
They don't end up in one place in scheduler code to cleanly implement. I experimented a case where futex are used between producer and consumers, where a hint is passed when cpufreq drivers to give small boost.
The idea here is to discuss:
- Shall we solve this problem?
- How to unify wait and wake up functions?
- Is it better to give a hint or boost PELT utilization of the consumer?

I agree to abide by the anti-harassment policy

Yes

Primary author: PANDRUVADA, Srinivas

Presenter: PANDRUVADA, Srinivas

Session Classification: Power Management and Thermal Control MC

Contribution ID: **220** Type: **not specified**

Proxy Execution

Monday, 9 September 2019 15:45 (15 minutes)

Proxy execution can be considered as a generalization of the real-time priority inheritance mechanism. With proxy execution a task can run using the context of some other task that is "willing" to let the first task run as this improves performace for both. With this topic I'd like to detail about progress that has been made after the initial RFC posting on LKML and discuss about open problems and questions.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LELLI, Juri (Red Hat)

Presenter: LELLI, Juri (Red Hat)

Session Classification: Scheduler MC

Contribution ID: **221** Type: **not specified**

Real-Time Container

Wednesday, 11 September 2019 11:00 (1 minute)

I'd like to review if-how we can build real-time container. It should include but not limited these topics here,

1. Understanding container Scheduling
2. Test and evaluations
3. Possible factors related to latency issues
4. discussions like tracing containers-leveled metrics
5. tips
6. etc.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: CHEN, Tiejun (VMware)

Presenter: CHEN, Tiejun (VMware)

Session Classification: Real Time MC

Contribution ID: **222** Type: **not specified**

Task latency-nice

Monday, 9 September 2019 18:15 (15 minutes)

Currently there is no user control on how much time scheduler should spend searching for CPUs when scheduling a task. It is hardcoded logic based on some heuristics that doesn't work well in many cases. e.g. very short running tasks. Provide a new latency-nice property user can set for a task (similar to nice value) that controls the search time and also potentially the preemption logic. Also discuss best interfaces to have this (potentially Cgroups).

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: MAZUMDAR, Subhra

Presenter: MAZUMDAR, Subhra

Session Classification: Scheduler MC

Contribution ID: **223** Type: **not specified**

Taking suspend/resume validation to the next level

Tuesday, 10 September 2019 17:25 (25 minutes)

At LPC 2015, we introduced analyze_suspend, a new open source tool to show where the time goes during Linux suspend/resume. Now called "sleepgraph", it has evolved in a number of ways over the last four years. Most importantly, it is now the core of a framework that we use for suspend/resume endurance testing.

Endurance testing has allowed us to identify, track, report and sometimes fix issues that developers used to dismiss as "unreproducible".

But to improve Linux suspend/resume quality further, we need more people testing different machines and reporting bugs. This is an appeal for ideas how the power of the broader open source community can be harnessed to improve Linux suspend/resume quality.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: BROWN, Len (Intel Open Source Technology Center)

Presenter: BROWN, Len (Intel Open Source Technology Center)

Session Classification: Power Management and Thermal Control MC

Contribution ID: **224** Type: **not specified**

Core Scheduling for RT

Wednesday, 11 September 2019 10:00 (30 minutes)

Recently speculative execution techniques have shown that an untrusted application can steal data from another one when both share the same core. To avoid such problems users have to disable SMT, causing non-negligible performance impact. Core-scheduling tries to mitigate the performance problem by allowing trusted applications to run concurrently on siblings of a core while avoiding two untrusted applications to share the same core.

However, this has a number of ramifications and applications for Real-Time schedulers too. For instance, the Admission Control of SCHED_DEADLINE depends on the number of CPUs, but with core scheduling, the number of CPUs available is a dynamic function. OTOH Real-Time workloads often want SMT disabled for determinism, and core-scheduling gives the capability for a single task to claim an entire core.

So I propose discussing the impact and possibilities of core-scheduling for Real-Time.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: ZIJLSTRA, Peter (Intel OTC)

Presenter: ZIJLSTRA, Peter (Intel OTC)

Session Classification: Real Time MC

Contribution ID: **225** Type: **not specified**

PREEMPT_RT: status and Q&A

Wednesday, 11 September 2019 13:00 (30 minutes)

In this talk, Thomas Gleixner will present the status of the PREEMPT_RT, along with a section of questions and answers regarding the upstream work and the future of the project.

I agree to abide by the anti-harassment policy

Yes

Primary author: GLEIXNER, Thomas

Presenter: GLEIXNER, Thomas

Session Classification: Real Time MC

Contribution ID: **227** Type: **not specified**

Use IOMMU to prevent DMA attacks from Thunderbolt devices

Monday, 9 September 2019 18:25 (15 minutes)

The Thunderbolt vulnerabilities are public and have a nice name as Thunderclap (https://thunderclap.io/) nowadays. This topic will introduce what kind of vulnerabilities we have identified with Linux and how we are fixing them.

I agree to abide by the anti-harassment policy

Yes

Primary author: LU, Baolu

Presenter: LU, Baolu

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **228** Type: **not specified**

PASID Management in Linux

Monday, 9 September 2019 16:00 (25 minutes)

PASID (Process Address Space ID) is a PCIe capability that enables sharing of a single device across multiple isolated address domains. It has been becoming a hot term in I/O technology evolution. e.g. it is foundation of SVM and SIOV. Combined with the usages of PASID and the configuration difference due to architecture difference across vendors, it brings an interesting topic on PASID management in Linux. Especially regards to software complexity for VM live migration support in cloud. This talk will review the PASID usages and configuration methods, then elaborate the gaps for PASID management. Finally propose a solution and start talks with peers.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr LIU, Yi (Intel)

Co-authors: Mr JACOB, Pan (Intel); Mr LU, Baolu (Intel); Mr TIAN, Kevin (Intel); RAJ, Ashok

Presenter: Mr JACOB, Pan (Intel)

Session Classification: VFIO/IOMMU/PCI MC

Contribution ID: **229** Type: **not specified**

Ethernet Cable Diagnostic using Netlink Ethtool API

Tuesday, 10 September 2019 17:45 (45 minutes)

Many Ethernet PHYs contain hardware to perform diagnostics of the Ethernet cable. Breaks in the cable and shorts within a twisted pair or to other pairs can be detected, and an estimate to the length along the cable to the fault can be made. The talk will explain, at a high level, how such diagnostics work, sending pulses down the cables and looking for reflections. There is no standardization on such diagnostics, and what information the PHY reports varies between vendors. The ongoing work to allow ethtool to make use of a netlink socket makes the ethtool API much more flexiable. This flexibility has been used to provide a generic API to request a PHY performs diagnostics tests and to report the results. Some aspects of this API will be discussed, using the Marvell PHYs as examples. The talk aims to spread knowledge on this work and encourage driver writers to implement diagnostics for other PHYs.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LUNN, Andrew

Presenter: LUNN, Andrew

Session Classification: Networking Summit Track

Contribution ID: **230** Type: **not specified**

TurboSched: Core capacity Computation and other challenges

Monday, 9 September 2019 18:00 (15 minutes)

Turbosched is a proposed scheduler enhancement that aims to sustain turbo frequencies for a longer duration by explicitly marking small tasks that are known to be jitters and pack them on a smaller number of cores. This ensures that the other cores will remain idle, and the energy thus saved can be used by CPU intensive tasks for sustaining higher frequencies for a longer duration.

The current TurboSched RFCv4 (https://lkml.org/lkml/2019/7/25/296) has some challenges:

- **Core Capacity Computation**: Spare core capacity defines the upper bound for task packing above which jitter tasks should not be packed further into a core, else it hurts the performance of the other tasks running on that core. To achieve this we need a mechanism to compute the capacity of the cores in terms of its active SMT threads. But the computation of CPU Capacity itself if arguable and non-reliable in case of CPU hotplug events. This makes the TurboSched to have unexpected behavior in case of hotplugs or in presence of asymmetric CPU capacities. The discussion also involves the use of other parameters like nr_running with utilization to decide upper bound for task packing.
- **Interface**: There are multiple approaches to mark a small-task as a jitter. A cgroup based approach is favorable to the distros as it is a well-understood interface requiring minimal modification for the existing tools. However, the kernel community has expressed objection to this interface since whether a task is jitter or not is a task-attribute and not a task-group attribute. Further, a task being a jitter is not a resource-partition problem, which is what cgroup aims to solve. The other approach would be to define this via a sched_attribute which can be updated via an existing syscall. Finally, we can support both the approaches as discussed on LWN https://lwn.net/Articles/792471/
- **Limiting the Search Domain for packing**: On systems with a large number of CPUs, searching all the CPUs where the small-tasks should be packed can be expensive in the task-wakeup path. Hence we should limit the
domain of CPUs over which the search is conducted. In the current implementation, TurboSched uses the DIE domain to pack tasks on PowerPC, but certain architectures might prefer the LLC or the NUMA domains. Thus we need to discuss a unified way of describing the search domain which can work across all architectures.

This topic is a continuation from the OSPM talk and aims to mitigate these problems generic across architectures.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: SHAH, Parth

Presenter: SHAH, Parth

Session Classification: Scheduler MC

Contribution ID: **231** Type: **not specified**

Using Yocto to build a distro and maintain a kernel tree

Monday, 9 September 2019 10:20 (20 minutes)

We'd like to spend a few minutes to provide some background around how we're using Yocto to produce kernel builds as well as bigger images that contain userspace as well, and then try to address some of the issues we're seeing with this process.

There are a few topics we'd like to discuss with the room:

- Using a single kernel branch for multiple, very different projects?
- Working with kernel config fragments?
- Reproducible kernel builds/cloning sources?
- Is there anything saner than cve-check for pointing out known security vulnerabilities?

I agree to abide by the anti-harassment policy

Yes

Primary authors: RAJARAM, Senthil; LEVIN, Sasha

Presenters: RAJARAM, Senthil; LEVIN, Sasha

Session Classification: Distribution Kernels MC

Contribution ID: **232** Type: **not specified**

Map batch processing

Wednesday, 11 September 2019 18:20 (20 minutes)

bcc community has long discussed that batch
dump, lookup and delete will help its typical
use case, periodically retrieving and deleting
all samples in the kernel. Without batch APIs,
bcc typically does
iterate through all keys (get_next_key API)
get (key, value) pairs
iterate through all keys to delete them

Also, Brian Vazquez
has proposed BPF_MAP_DUMP command to dump
more than one entry per syscall call.
https://www.spinics.net/lists/netdev/msg583538.html

This discussion will propose new bpf subcommands
for map batch processing, e.g., batching
get_next_key/lookup/update/delete/lookup_and_delete.
discuss its pros and cons etc.

Looks the subject has been discussed actively in the mailing list.
If the discussion reached its maturity, we may not need to discuss
in the conference.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: SONG, Yonghong

Presenter: SONG, Yonghong

Session Classification: BPF MC

Contribution ID: **233** Type: **not specified**

BPF Debugging

Wednesday, 11 September 2019 15:23 (22 minutes)

Debugging BPF program logic is hard these days.
Developers typically write their programs and
then checking map values or perf_event outputs
make sense or not. For tricky issues, temporary
maps or bpf_trace_printk are used so developer
can get more insight about what happens. But
this requires possibly multiple rounds of
modifying sources, recompilation and redeployment, etc.

This discussion surrounds creating bpf debugging
tool, bdb (bpf debugger) similar naming after gdb/lldb.
This tool should try to do what gdb for ELF execution.
- specify breakpoints at source/xlated/jitted level
- retrieve data for registers, stacks and globals(maps)
and presented at both register and variable level.
- different conditions to retrieve data, e.g.,
running 100 times, only if this variable == 1.
this will require kernel to live patch bpf codes.
- modifying data (register, stack slot, globals)?
how does this interact with verifier to ensure safety.
- this will leverage BTF and existing test_run framework.
- production debugging vs. qemu debugging
qemu debugging may be truely single-step.

I agree to abide by the anti-harassment policy

Yes

Primary author: SONG, Yonghong

Presenter: SONG, Yonghong

Session Classification: BPF MC

Contribution ID: **235** Type: **not specified**

Bringing BPF developer experience to the next level

Wednesday, 11 September 2019 15:00 (23 minutes)

The way BPF application developers build applications is constantly improving. There are still rough corners, as well as (as of yet) fundamentally inconvenient developer workflows involved (e.g., on-the-fly compilation). The ultimate goal of BPF application development is to provide experience as straightforward and simple as a typical user-land application.

We'll discuss major pain points with BPF developer experience today and present motivation for solving them. Libbpf and BTF type info integration are at the center of the puzzle that's being put together to provide a powerful and yet less error-prone solution:
- BPF CO-RE and how it is addressing adapting to ever-changing kernel and facilitates safe and efficient kernel introspection;
- consistent and safer APIs to load/attach/work with BPF programs;
- declarative and more powerful ways to define and initialize BPF maps;
- providing and standardizing BPF-side helper library for all BPF code needs.

I agree to abide by the anti-harassment policy

Yes

Primary author: NAKRYIKO, Andrii (Facebook)

Presenter: NAKRYIKO, Andrii (Facebook)

Session Classification: BPF MC

Contribution ID: **236** Type: **not specified**

Implementing LoRa, FSK and further LPWAN interfaces

Monday, 9 September 2019 16:00 (30 minutes)

This talk will give an overview of LoRa and related wireless technologies and their role in IoT infrastructure. An initial RFC for a socket interface had been submitted last summer as proof of concept - a linux-lora.git staging tree and linux-lpwan mailing list have been in use for collaboratively iterating on patches towards a mergeable proposal. Open topics include abandoning PF_LORA in favor of PF_PACKET and how to layer PF_LORAWAN on top of LoRa and FSK; on the driver side the LoRa gateway chipset SX1301/SX1308 has run into problems with clk/spi/reset, and no solution for expanding from DT to ACPI has been found yet; adding protocol families and testing them on a large range of devices has not been easy, and while many of these wireless technologies share design principles they have so far been unable to share any code on the PHY layer. 6LoWPAN and SCHC are candidates for higher-level soft-MACs. 3D-UNB is one of multiple candidates for getting similar treatment to LoRa.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr FÄRBER, Andreas (SUSE)

Presenter: Mr FÄRBER, Andreas (SUSE)

Session Classification: You, Me, and IoT MC

Contribution ID: **237** Type: **not specified**

Eliminating WrapFS hackery in Android with ExtFUSE (eBPF/FUSE)

Tuesday, 10 September 2019 17:00 (15 minutes)

This work proposes to adopt Extended FUSE (ExtFUSE) framework for improving the performance of Android SDCard FUSE daemon, thereby eliminating a need for out-of-tree WrapFS hackery in the Android kernel.

ExtFUSE leverages eBPF framework for developing extensible FUSE file systems. It allows FUSE daemon in Android to register "thin" eBPF handlers that can serve metadata as well as data I/O file system requests right in the kernel to improve performance. Our evaluation with Android SDCardFS under ExtFUSE shows about 90% improvement in app launch latency with less than thousand lines of eBPF code in the kernel. In the presentation, I will share my findings and progress made to get feedback from the Android kernel developers.

Overall, this work benefits millions of Android devices that are currently running out-of-tree WrapFS-based code in the kernel for emulating FAT functionality and enforcing custom security checks.

I agree to abide by the anti-harassment policy

Yes

Primary author: BIJLANI, Ashish (Georgia Institute of Technology)

Session Classification: Android MC

Contribution ID: **238** Type: **not specified**

Being Kernel Maintainer at Oracle - Lessons & Challenges.

Linux kernel maintenance is widely spoken topic at many conferences. Yet, it has it's own complex share of problems which are unique to maintainers, sub-systems and Organizations.

Oracle has a very Open and challenging environment but with access to a lot of information and knowledge about our customer's products and strategies, it can very tricky for a kernel maintainer especially the challenges it brings and also the need to keep a keen eye on internal and public discussions.

In this talk, I would like to share my experiences of being a Kernel Maintainer at Oracle. The topics that would be covered are:
- How we maintain the kernel(UEK)?
- How does the Kernel stay up-to-date (stable fixes)?
- How we handle KABI breakages and updates.
- How do we handle back-ports and Security fixes(CVE's).

In addition to the above, we would also like to talk about our Upstream tracking project which essentially helps developers to keep their work up-to-date with mainline.

I agree to abide by the anti-harassment policy

Yes

Primary author: PAIS, Allen

Session Classification: Distribution Kernels MC

Contribution ID: **239** Type: **not specified**

BPF packet capture helpers, libbpf interfaces

Monday, 9 September 2019 10:45 (45 minutes)

Packet capture is useful from a general debugging standpoint, and is useful in particular in debugging BPF programs that do packet processing. For general debugging, being able to initiate arbitrary packet capture from kprobes and tracepoints is highly valuable (e.g. what do the packets that reach kfree_skb() - representing error codepaths - look like?). Arbitrary packet capture is distinct from the traditional concept of pre-defined hooks, and gives much more flexibility in probing system behaviour. For packet-processing BPF programs, packet capture can be useful for doing things such as debugging checksum errors. The intent of this proposal is to help drive discussion around how to ease use of such features in BPF programs, namely:

- should additional BPF helper(s) be provided to format packet data suitable for libpcap interpretation?
- should libbpf provide interfaces for retrieving packet capture data?
- should interfaces be provided for pushing filters?

Note that while there has been some work in this area already, such as

https://new.blog.cloudflare.com/xdpcap/

…it seems like such efforts would be made much simpler if APIs were provided.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: MAGUIRE, Alan (Oracle)

Presenter: MAGUIRE, Alan (Oracle)

Session Classification: Networking Summit Track

Contribution ID: **240** Type: **not specified**

Performance guarantees under thermal pressure

Tuesday, 10 September 2019 15:25 (25 minutes)

Performance capping due to thermal limitations is common scenario particularly in mobile systems. Today user-space has no information about what level of performance that can be expected worst case and SCHED_DEADLINE can admit reservations which are impossible to fulfill.
The purpose of the this topic is to discuss what level guarantees the kernel should provide. Should the kernel have a platform specific or tunable sustained performance level?

I agree to abide by the anti-harassment policy

Yes

Primary author: RASMUSSEN, Morten (Arm)

Presenter: RASMUSSEN, Morten (Arm)

Session Classification: Power Management and Thermal Control MC

Contribution ID: **241** Type: **not specified**

RDMA, File Systems, and DAX

Wednesday, 11 September 2019 11:00 (30 minutes)

For almost 2 years now the use of RDMA with DAX filesystems has been disabled due to the incompatibilities of RDMA and the file system page handling.

A general consensus has emerged from many conferences and email threads on a path to support RDMA directly to persistent memory which is managed by a filesystem.

This talk will present the work done since LSFmm to support RDMA and FS DAX.

Specifically this work requires exclusive layout lease grants to obtain pins.
Fails truncate operations on file pages which have been given pins. And supports recovery by admins by allowing them to identify offending processes holding these pins.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr WEINY, Ira

Presenter: Mr WEINY, Ira

Session Classification: RDMA MC

Contribution ID: **242** Type: **not specified**

Future ipv4 unicast extensions

Tuesday, 10 September 2019 12:45 (45 minutes)

IPv4's success story was in carrying unicast packets worldwide.
Service sites still need IPv4 addresses for everything, since the majority of Internet client nodes don't yet have IPv6 addresses. IPv4 addresses now cost 15 to 20 dollars apiece (times the size of your network!) and the price is rising.

The IPv4 address space includes hundreds of millions of addresses reserved for obscure (the ranges 0/8, and 127/16), or obsolete (225/8-231/8) reasons, or for "future use" (240/4 - otherwise known as class E). Instead of leaving these IP addresses unused, we have started an effort to make them usable, generally. This work stalled out 10 years ago, because IPv6 was going to be universally deployed by now, and reliance on IPv4 was expected to be much lower than it in fact still is.

We have been reporting bugs and sending patches to various vendors. For Linux, we have patches accepted in the kernel and patches pending for the distributions, routing daemons, and userland tools. Slowly but surely, we are decontaminating these IP addresses so they can be used in the near future.

Many routers already handle many of these addresses, or can easily be configured to do so, and so we are working to expand unicast treatment of these addresses in routers and other OSes. We plan an authorized experiment to route some of these addresses globally, monitor their reachability from different parts of the Internet, and talk to ISPs who are not yet treating them as unicast to update their networks.

Wouldn't it be a better world with a few hundred million more IPv4 addresses in it?

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Dr TÄHT, Dave (Bufferbloat.net)

Presenter: Dr TÄHT, Dave (Bufferbloat.net)

Session Classification: Networking Summit Track

Contribution ID: **244** Type: **not specified**

RISCV NOMMU/M-Mode Linux

Monday, 9 September 2019 13:15 (15 minutes)

This presentation will discuss the work ongoing to implement Linux kernel support for RISCV hardware lacking a memory management unit (MMU). A side effect of this work is also the ability to execute the kernel directly in M-Mode and how this is implemented while keeping most of the architecture code unmodified. The presentation will include examples of testing environment builds, discuss the support state of userspace toolchains and C libraries and will present the direct application of this work to a real hardware platform (Kendryte K210 SoC).

I agree to abide by the anti-harassment policy

Yes

Primary author: LE MOAL, Damien (Western Digital)

Presenter: LE MOAL, Damien (Western Digital)

Session Classification: RISC-V MC

Contribution ID: **245** Type: **not specified**

Task-centric thermal management

Tuesday, 10 September 2019 15:50 (25 minutes)

Thermally unsustainable compute demand is in most systems controlled by reducing performance through disabling performance states on specific CPUs or other devices in the system. It provides an efficient method to ensure the system doesn't overheat, however, it doesn't take the actual workload into account which could be better served if the performance caps were applied differently.
The intention with this topic is to discuss the idea of controlling tasks, i.e. compute demand (potentially from user-space), instead of controlling devices directly.

I agree to abide by the anti-harassment policy

Yes

Primary author: RASMUSSEN, Morten (Arm)

Presenter: RASMUSSEN, Morten (Arm)

Session Classification: Power Management and Thermal Control MC

Contribution ID: **246** Type: **not specified**

Printing in Linux as of today

Tuesday, 10 September 2019 10:00 (20 minutes)

Today's is a scenario when we can not think of having either a mobile phone or a laptop or a tablet. With the progress of technology and having all these handheld devices, we have been able to get many of our documents digitized. However, whatever advancements we see in this space of documentation, it is still very hard to find someone who did not have the need to print or scan a hard copy. Even today a critical agreement gets signed over a hard copy so do most of our banking documents or promo advertisements in a supermarket.

The OpenPrinting (OP) organization works on the development of new printing architectures, technologies, printing infrastructure, and interface standards for Linux and UNIX-style operating systems. OP collaborates with the IEEE-ISTO Printer Working Group (PWG) on IPP projects.
We maintains cups-filters which allows CUPS to be used on any Unix-based (non-macOS) system. OpenPrinting maintains the Foomatic database which is a database-driven system for integrating free software printer drivers with CUPS under Unix. It supports every free software printer driver known to us and every printer known to work with these drivers.

OpenPrinting has been doing a commendable job in improving the way the world prints on a UNIX based system. The projects that we maintain are taken up by almost all the Linux distributions and most recently Google Chrome OS. It is also used by most of the printer manufacturers to support printing. Today it is very hard to think about printing in these OSs without the involvement of OpenPrinting. We have been successful in implementing the driverless printing following the IPP standards proposed by the PWG. Because of that, today someone can think of printing from a Linux box by just connecting a printer over network or USB. Now using a printer has become as simple as using a thumb drive.

A short showcase on printing in Linux.

I agree to abide by the anti-harassment policy

Presenters: BASU, Aveek; KAMPPETER, Till

Session Classification: Open Printing MC

Contribution ID: **249** Type: **not specified**

GUP and ZONE_DEVICE pages

Wednesday, 11 September 2019 10:00 (1 hour)

P2P
- Suggestion with VFIO (Don)
- RDMA as the importer, VFIO as the exporter

get_user_pages() and friends
- Discussion on future GUP, required to support P2P
- GUP to SGL?
- Non struct page based GUP

hmm_range_fault()
- Integrating RDMA ODP with HMM
- 'DMA fault' for ZONE_DEVICE pages

I agree to abide by the anti-harassment policy

I confirm that I am already registered for LPC 2019

Presenters: DUTILE, Don (Red Hat); GUNTHORPE, Jason (Mellanox Technologies); HUBBARD, John (NVIDIA)

Session Classification: RDMA MC

Contribution ID: **250** Type: **not specified**

GUP for P2P

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Session Classification: RDMA MC

Contribution ID: **251** Type: **not specified**

Shared IB Objects

Wednesday, 11 September 2019 12:30 (30 minutes)

Consider a case of a server with a huge amount of memory and thousands of processes are using it to serve clients requests.

In such a case, the HCA will have to manage thousands of MRs which will compete for caches and address translation entities.

The way to improve performance is to allow sharing of IB objects between processes. One process will create several MRs and share them.

This will reduce the number of address translation entries and cache miss dramatically.

This talk will cover the implementation of a Shared Object mechanism.

I agree to abide by the anti-harassment policy

Presenter: SHAIA, Yuval (Oracle)

Session Classification: RDMA MC

Contribution ID: **252** Type: **not specified**

Improving RDMA performance through the use of contiguous memory and larger pages for files.

Wednesday, 11 September 2019 13:00 (30 minutes)

As memory sizes grow so do the sizes of the data transferred between RDMA devices. Generally, the Operating system needs to keep track of the state of each of its pieces of memory and that is on Intel x86 a page of 4 KB. This is also connected to hardware providing memory management features such as the processor page tables as well as the MMU features of the RDMA NIC.

The overhead of the operating system increases as the number of these pages reaches ever higher orders of magnitude. I.e. for 4GB of data one needs 1 million of these page descriptors. Each page descriptor is a 64-byte cache line and thus a 4GB operation requires 64MB of cache lines to be managed.

A lot of efforts on optimization of I/O focuses on avoiding touching these page descriptors through the use of larger contiguous memory or larger page sizes. This talk gives an overview of the current methods in use to avoid these lowdowns and the work in progress to improve the situation and make it less of an effort to avoid these issues.

I agree to abide by the anti-harassment policy

Presenter: LAMETER, Christopher (Jump Trading LLC)

Session Classification: RDMA MC

Contribution ID: **254** Type: **not specified**

reference Integrity measurements for TPM2 security policy

Wednesday, 11 September 2019 17:20 (20 minutes)

Firmware on commodity PCs have used the TPM to store integrity measurements from security relevant components as part of the boot process for some time. Grub2 has recently merged patches that extend this integrity measurement chain through to the launching of the OS kernel. Collecting and storing these measurements in the TPM is a necessary precondition for implementing authorization policy based on the state of the system, but this alone is insufficient.

This talk will begin by discussing the current state of boot-time integrity measurement collection in UEFI firmware and Grub2. We'll then present a notional use-case implementing security controls based on TPM2 policy mechanisms while describing the plumbing required to enable interaction with the TPM2 device. The remainder of this talk will then discuss the existing gaps in software and tooling required to implement work-flows for managing configuration of the relevant security controls across system install and update operations.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: TRICCA, Philip (Intel)

Presenter: TRICCA, Philip (Intel)

Session Classification: System Boot and Security MC

Contribution ID: **255** Type: **not specified**

DMABUF Developments

Tuesday, 10 September 2019 18:00 (15 minutes)

To discuss recent developments and directions with DMABUF:
* DMABUF Heaps/ION destaging
* Better DMABUF ownership state machine documentation
* DMABUF cache maintenance optimizations
* Kernel graphics buffer idea

I agree to abide by the anti-harassment policy

Yes

Primary authors: SEMWAL, Sumit; STULTZ (IN ABSENTIA), John

Presenters: SEMWAL, Sumit; STULTZ (IN ABSENTIA), John

Session Classification: Android MC

Contribution ID: **256** Type: **not specified**

Common Print Dialog Backends

Tuesday, 10 September 2019 10:20 (30 minutes)

The OpenPrinting project "Common Print Dialog Backends" provides a D-Bus interface to separate the print dialog GUI from the communication with the actual printing system (CUPS, Google Cloud Print, e.t.c.) having each printing system being supported with a backend and these GUI-independent backends working with all print dialogs (GTK/GNOME, Qt/KDE, LibreOffice, e.t.c.). This allows for easily updating all print dialogs when something in a print technology changes, as only the appropriate backend needs to get updated. Also new print technologies can get easily introduced by adding a new backend.
For quickly getting this concept into the Linux distributions we need these important tasks to be done.
The CUPS backend tells the print dialog only about printer-specific user-settable options, not about general options implemented in CUPS or cups-filters and so being available for all print queues. These are options like N-up, reverse order, selected pages, e.t.c. as they are only common for CUPS and not necessarily available with other print technologies like Google Cloud Print, they should get reported to the print dialog by the CUPS backend.
A print dialog should allow to print into a (PDF) file. This should be implemented in a new print dialog backend. [DONE: https://github.com/OpenPrinting/cpdb-backend-file]
As it will take time until GTK4 with its new print dialog is out, we should get support for the new Common Print Dialog Backends concept for the current GTK3 dialog. As this dialog has its own backend concept one simply would need an "adapter" backend to get from the old concept to the new, common concept.
[Qt print dialog integration]

I agree to abide by the anti-harassment policy

I confirm that I am already registered for LPC 2019

Presenters: PATIBANDLA, Rithvik; KAMPPETER, Till

Session Classification: Open Printing MC

Contribution ID: **257** Type: **not specified**

Working with SANE to make IPP scanning a reality

Tuesday, 10 September 2019 10:50 (40 minutes)

Printing at today's date has progressed a lot and the world is already utilising the benefits of driverless printing. In today's scenario it is very hard to think of a printer without a scanner. But unfortunately a technology like driverless scanning has yet to see the light of the day. In today's date you cannot think of using a scanner without a scanner driver. We want to discuss more on this and what needs to be done to get rid of this problem.

Version 2.0 and newer of the Internet Printing Protocol (IPP) supports polling the full set of capabilities of a printer and if the printer supports a known Page Description Language (PDL), like PWG Raster, Apple Raster, PCLm, or PDF, it is possible to print without printer-model-specific software (driver) or data (PPD file), so-called "driverless" printing. This concept was introduced for printing from smartphones and IoT devices which do not hold a large collection of printer drivers. Driverless printing is already fully supported under Linux. Standards following this scheme are IPP Everywhere, Apple AirPrint, Mopria, and Wi-Fi Direct Print. As there are many multi-function devices (printer/scanner/copier all-in-one) which use the IPP, the Printing Working Group (PWG) has also worked out a standard for IPP-based scanning, "driverless" scanning, to also allow scanning from a wide range of client devices, independent of which operating systems they are running.
Conventional scanners are supported under Linux via the SANE (Scanner Access Now Easy) system and require drivers specific to the different scanner models. Most of them are written based on reverse-engineering due to lack of support by the scanner manufacturers. To get driverless scanning working with the software the users are used to the best solution is to write a SANE module for driverless IPP scanning. This module will then automatically support all IPP scanners, thousands of scanners where many of them do not yet exist.
Another application for driverless IPP scanning is sharing local scanners which are accessed with SANE. Instead of the SANE frontend being a UI, either command line or graphical, it could be a daemon which emulates an IPP scanner on the network, executing the client's scan requests on the local scanner.
This way the client only needs to support IPP scanning, no driver for the actual scanner is needed and the client can be of any operating system or device type, including mobile phones, tablets, IoT, e.t.c.

I agree to abide by the anti-harassment policy

I confirm that I am already registered for LPC 2019

Presenter: BASU, Aveek

Session Classification: Open Printing MC

Contribution ID: **258** Type: **not specified**

Printer/Scanner Applications - The new format for printer and scanner drivers

Tuesday, 10 September 2019 12:00 (30 minutes)

The upstream author of CUPS has deprecated the classic way to implement printer drivers, describing the printer's capabilities in PPD (PostScript Printer Description) files and providing filters to convert standard PDLs (Page Description Languages) into the printer's own, often proprietary data format. With the background of PostScript not being the standard PDL any more, most modern (even the cheapest) printers being IPP driverless printers (using standard PDLs and printer's capabilities can get polled from the printer via IPP), and modern systems using sandboxed application packaging (Snappy, Flatpak, e.t.c.) the new Printer Application concept got introduced.
A Printer Application is a (simple) daemon emulating a driverless IPP printer (can be in the local network but also simply on localhost). Like a physical printer this daemon advertises itself via DNS-SD, takes get-printer-attributes IPP requests and answers with printer capability info so that the client can create a local print queue pointing to it, takes print jobs, converts them to the physical printer's data format and sends them off to the printer.
This way the client "sees" a driverless IPP printer and the Printer Application is the printer driver (printer-model-specific software to make the printer work). So with the driver being connected to the system's printing stack only via IP and no consisting of files spread into directories of the printing stack, both the printing stack and the driver can be in separate, sandboxed applications, provided as sandboxed packages in the app stores of the appropriate packaging systems (Snappy, Flatpak, e.t.c.). And this allows the driver not depending on a specific operating system distribution any more. A printer manufacturer only needs to make a driver "for Snappy", not for Ubuntu Desktop/Server, Ubuntu Core, Red Hat, SUSE, e.t.c. making development and testing much easier and cheaper.
And one can even go further: As the Printer Working Group (PWG) also has created an IPP driverless scanning standard, we can create Scanner Applications emulating a driverless IPP scanner and internally using scanner drivers, like SANE, to communicate with the scanner, allowing the same form of OS-distribution-independent sandboxed driver packages for ANY scanner, especially also stand-alone scanners without printing engine.
For multi-function printers one could also have a combined Printer/Scanner application. Any such Printer and/or Scanner Application can even provide an IPP System Service interface to allow configuring the driver without need of specialized GUI applications on the client.
We have a Google Summer of Code student working on a framework for Printer Applications, to convert classic printer drivers into Printer Applications to kick off the new standard.
In this session we will present the new format, its integration into real life systems, problems we got into during the work with our student, and how to present it to hardware manufacturers as the new way to go.

I agree to abide by the anti-harassment policy

Presenter: KAMPPETER, Till

Session Classification: Open Printing MC

Contribution ID: **259** Type: **not specified**

The Future of Printer Setup Tools - IPP Driverless Printing and IPP System Service

Tuesday, 10 September 2019 12:30 (30 minutes)

Very common in the daily life of computer users are printer setup tools, these GUI applications where you configure a queue for a new printer which you want to use. You select the printer from auto-detected ones and choose a driver for it, nowadays it gets rather common that the driver is selected automatically. You also set option defaults, like Letter/A4, print quality, …
With the advent of driverless IPP printers and automatic setup of network printers the classic printer setup tool gets less important. Especially one sees this on smartphones and tablets which do not even have a printer setup tool and option settings and default printers are selected in the print dialogs.
But this does not mean that the time of printer setup tools is over, especially in larger networks they can help getting an overview of the available printers, controlling tools like cups-browsed (or perhaps also the print dialog backends?) to make the user's print dialogs only showing the relevant ones or to create printer clusters.
Also the printers itself could be configured with a printer setup tool when they support the new IPP System Service standard, an interface which allows remote administration of IPP network printers, similar to what you can do with the printer's web interface but with a standardized client GUI.
In this session we will talk about new possibilities in printer setup tools and their implementation.
Ideas are:
Client GUI for IPP System Service - Administration of network printers
Configuring cups-browsed - GUI for printer list filtering, printer clustering, …
Configuring Common Print Dialog Backends
More ideas are naturally welcome.

I agree to abide by the anti-harassment policy

Presenter: KAMPPETER, Till

Session Classification: Open Printing MC

Contribution ID: **260** Type: **not specified**

3D Printing without the use of any slicer.

Tuesday, 10 September 2019 13:00 (30 minutes)

Currently to print an stl model in a 3D printer the same needs to be sliced first into a gcode to be understandable by a 3D printing software. In Linux we do not have any filter that can convert a stl code to a gcode. First we plan to discuss on what is the current scenario and then what can we do to fit in Linux.

Presenter: BASU, Aveek

Session Classification: Open Printing MC

Contribution ID: **261** Type: **not specified**

KernelCI applied to distributions

Monday, 9 September 2019 12:00 (30 minutes)

While kernelci.org as a project is dedicated to testing the upstream Linux kernel, the same KernelCI software may be reused for alternative purposes. One typical example is distribution kernels, which often track a stable branch but also carry some extra patches and a specific configuration. Aside from covering a particular downstream branch, having a separate KernelCI instance also makes it possible to add specific tests that cover user-space functionality.

A key aspect of KernelCI however is that the moving part remains the kernel revision. It is in theory possible to cover a full OS image with moving parts in user-space too, but that is not something it was originally designed for - hence an interesting subject for discussion.

I agree to abide by the anti-harassment policy

Yes

Primary author: TUCKER, Guillaume (Collabora Limited)

Presenter: TUCKER, Guillaume (Collabora Limited)

Session Classification: Distribution Kernels MC

Contribution ID: **262** Type: **not specified**

Making the Kubernetes Service Abstraction Scale using eBPF

Tuesday, 10 September 2019 15:00 (45 minutes)

In this talk, we will present a scalable re-implementation of the Kubernetes service abstraction with the help of eBPF. We will discuss recent changes in the kernel which made the implementation possible, and some changes in the future which would simplify the implementation.

Kubernetes is an open-source container orchestration multi-component distributed system. It provides mechanisms for deploying, maintaining and scaling applications running in containers across a multi-host cluster. Its smallest scheduling unit is called a pod. A pod consists of multiple co-located containers. Each pod has its own network namespace and is addressed by an unique IP address in a cluster. Network connectivity to and among pods is handled by an external plugin.

Multiple pods which provide the same functionality can be grouped into services. Each service is reachable within a cluster via its virtual IP address allocated by Kubernetes. Also, a service can be exposed to outside of a cluster via the public IP address of a cluster host IP address and a port which is allocated by Kubernetes. Each request sent to a service is load-balanced to any of its pods.

Kube-proxy is a Kubernetes component which is responsible for the service abstraction implementation. The default implementation is based on Netfilter's iptables. For each service and its pods it creates couple rules in the nat table which do a load-balancing to pods. For example, for the "nginx" service which virtual IP address is 10.107.41.178 and which is running two pods with IP addresses 10.217.1.154 and 10.217.1.159 the following relevant iptables rules are created:

<pre> <code> -A KUBE-SERVICES -d 10.107.41.178/32 -p tcp -m comment –comment "default/nginx: cluster IP" -m tcp –dport 80 -j KUBE-SVC-253L2MOZ6TC5FE7P

-A KUBE-SVC-253L2MOZ6TC5FE7P -m statistic –mode random –probability 0.50000000000 -j KUBE-SEP-PCCJCD7AQBIZDZ2N -A KUBE-SVC-253L2MOZ6TC5FE7P -j KUBE-SEP-UFVSO22B5A7KHVMO

-A KUBE-SEP-PCCJCD7AQBIZDZ2N -s 10.217.1.154/32 -j KUBE-MARK-MASQ -A KUBE-SEP-PCCJCD7AQBIZDZ2N -p tcp -m tcp -j DNAT –to-destination 10.217.1.154:80 -A KUBE-SEP-UFVSO22B5A7KHVMO -s 10.217.1.159/32 -j KUBE-MARK-MASQ -A KUBE-SEP-UFVSO22B5A7KHVMO -p tcp -m tcp -j DNAT –to-destination 10.217.1.159:80 </code> </pre>

It has been demonstrated \[1\]\[2\]\[3\] that kube-proxy due to its foundational technologies (Netfilter, iptables) is one of the major pain points when running Kubernetes at large scale from performance, reliability, and operations perspective.

Cilium is an open-source networking and security plugin for container orchestration systems, such as Kubernetes. Unlike the majority of such networking plugins, it heavily relies on eBPF technology which lets one to dynamically reprogram the kernel.

The most recent Cilium v1.6 release brings the implementation in eBPF of the Kubernetes service abstraction. This allows one to run a Kubernetes cluster without kube-proxy. Thus, it makes Kubernetes no longer dependent on Netfilter/iptables. This improves scalability and reliability of a Kubernetes cluster.

No Kubernetes knowledge is required. The talk might be relevant for those who are interested in container networking with eBPF (loadbalancing, NAT).

\[1\]: https://sched.co/MPch
\[2\]: https://bit.ly/2xKk2pr
\[3\]: https://bit.ly/2WU7BCN

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: Mr DANIEL, Borkmann (Cilium); Mr MARTYNAS, Pumputis (Cilium)

Presenters: Mr DANIEL, Borkmann (Cilium); Mr MARTYNAS, Pumputis (Cilium)

Session Classification: Networking Summit Track

Contribution ID: **263** Type: **not specified**

Using the new mount API with containers

Tuesday, 10 September 2019 17:00 (30 minutes)

The Linux kernel has recently acquired a new API for creating mounts. This allows a greater range of parameter and parameter values to be specified, including, in the future, container-relevant information such as the namespaces that a mount should use.

Future developments of this API also need to work out how to deal with upcalling from the kernel to gain parameters not directly supplied, such as DNS records, automount configurations or configuration overrides, whilst preventing namespacing violations through the upcall.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr HOWELLS, David (Red Hat)

Presenter: Mr HOWELLS, David (Red Hat)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **264** Type: **not specified**

Device power management based on platform firmware

Tuesday, 10 September 2019 17:00 (25 minutes)

Continuing the attempts to reducing fragmentation in power management on ARM platforms, there are discussions if something similar to ACPI can be done.i.e. device centric power management.

Currently, a device has power, performance, reset, and clock domains associated with it. SCMI provides interface to deal with these domains directly. This was simpler approach to start with the SCMI specification to keep the OSPM related changes minimal. So for a given device it's power, performance, reset, clock,…etc domains need to be known and appropriate requests should be made on those domains when needed. Since this list seem to ever growing on ARM platforms, like pinmux, gpio, iomux,…etc, the current approach is not sustainable for long.

Instead of this, there's a thought on making these device centric and drive it.
So OSPM need not care which power/perf/reset/clock domain it belongs. All the details are abstracted from OSPM completely.

This talk is to discuss and understand where how to drive this platform firmware based device power management from Linux kernel. Which existing subsystem to reuse ?

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr HOLLA, Sudeep (ARM)

Presenter: Mr HOLLA, Sudeep (ARM)

Session Classification: Power Management and Thermal Control MC

Contribution ID: **265** Type: **not specified**

Using kernel keyrings with containers

Tuesday, 10 September 2019 18:40 (30 minutes)

The kernel contains a keyrings facility for handling tokens for filesystems and other kernel services to use. These are frequently disabled for container environments, however, because they were not made namespace aware by the authors of the user-namespace and others.

Unfortunately, this lack prevents various things from working inside containers. To get around this, keys are now being tagged with a namespace tag that allows keys operating in different namespaces to coexist in the same keyring and restrictions have been placed on joining session keyrings across namespaces.

This still isn't sufficient to make them truly useful here. Intended future developments include: granting a permit to use a key to a container; adding per-container keyrings; request-key upcall namespacing.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr HOWELLS, David (Red Hat)

Presenter: Mr HOWELLS, David (Red Hat)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **266** Type: **not specified**

Seccomp Syscall Interception

Tuesday, 10 September 2019 15:45 (15 minutes)

Recently the kernel landed seccomp support for SECCOMP_RET_USER_NOTIF which enables a process (watchee) to retrieve a fd for its seccomp filter. This fd can then be handed to another (usually more privileged) process (watcher). The watcher will then be able to receive seccomp messages about the syscalls having been performed by the watchee.

We have integrated this feature into userspace and currently make heavy use of this to intercept mknod() syscalls in user namespaces aka in containers.
If the mknod() syscall matches a device in a pre-determined whitelist the privileged watcher will perform the mknod syscall in lieu of the unprivileged watchee and report back to the watchee on the success or failure of its attempt. If the syscall does not match a device in a whitelist we simply report an error.

This talk is going to show how this works and what limitations we run into and what future improvements we plan on doing in the kernel.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr BRAUNER, Christian

Presenter: Mr BRAUNER, Christian

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **267** Type: **not specified**

Address Space Isolation for Container Security

Tuesday, 10 September 2019 15:30 (15 minutes)

Containers are generally percieved less secure than virtual machines. Without going into a theological argument about the actual state of the affairs, we suggest to explore the possibility of using address space isolation inside the kernel to make containers even more secure.

Assuming that kernel bugs and therefore vulnerabilities are inevitable it is worth isolating parts of the kernel to minimize damage that these vulnerabilities can cause.

One way to create such isolation is to assign an address space to the Linux namespaces, so that tasks running in namespace A have different view of kernel memory mappings than the tasks running in namespace B.

For instance, by keeping all the objects in a network namespace private, we can achieve levels of isolation equivalent to running a separated network stack.

Another possible usecase is isolating address spaces for different user namespaces.

Beside marrying namespaces with address spaces we also considering implementaiton of isolated memory mappings using mmap()/madvise() so that a region of the caller's memory would be hidden from the rest of the system.

We are going to give a short update on current status of our research and we are going to discuss implications of the address space isolation and possible future directions:

- What are the trade-offs between letting user-space to control the isolation or keeping the control completely in-kernel.
- What should be user-visible interface for address space management? Does it need to be on/off switch at kernel command line or do we need runtime knobs for that? Or maybe even "address space namespace" or "address space cgroup"?
- How can we evaluate the security improvements beyond empiric obvservation that when less code and data are mapped, there are less vulnerabilities exposed?

I agree to abide by the anti-harassment policy

Yes

Primary authors: RAPOPORT, Mike; BOTTOMLEY, James (IBM)

Presenters: RAPOPORT, Mike; BOTTOMLEY, James (IBM)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **268** Type: **not specified**

Compact C Type Format Support in the GNU toolchain

Tuesday, 10 September 2019 12:30 (30 minutes)

A brief introduction to CTF and its recent addition to the GNU toolchain: what is it for, what's there now, what improvements are planned, and why you might want to use this stuff rather than DWARF.

What cool things might we be able to do now that C programs can inspect their own types cheaply? What cool things might we be able to do if we extend this to other languages, so C programs could introspect into other languages' type systems?

A particular focus of interest will be finding out how CTF could help BTF, and vice versa: they are doing similar but slightly different things, and surely the two schemes could cooperate to the benefit of both.

I agree to abide by the anti-harassment policy

Yes

Primary authors: ALCOCK, Nick (Oracle Corporation); BHAGAT, Indu (Oracle Corporation)

Presenters: ALCOCK, Nick (Oracle Corporation); BHAGAT, Indu (Oracle Corporation)

Session Classification: Toolchains MC

Contribution ID: **269** Type: **not specified**

A pure Go BPF library

Wednesday, 11 September 2019 15:45 (22 minutes)

At the LSF/MM eBPF track, we discussed the necessity of a common Go library to interact with BPF. Since then, Cilium and Cloudflare have worked out a proposal to upstream parts of github.com/newtools/ebpf and github.com/cilium/cilium/pkg/bpf into a new common library.

Our goal is to create a native Go library instead of a CGO wrapper of C libbpf. This provides superior performance, debuggability and ease of deployment. The focus will be on supporting long-running daemons interacting with the kernel, such as Cilium or Cloudflare's L4 load balancer.

We'd like to present this proposal to the wider BPF community and solicit feedback. We'll cover the goals and guiding principles we've set ourselves and our initial roadmap.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: STRINGER, Joe (Isovalent / Cilium); BAUER, Lorenz (Cloudflare); PUMPUTIS, Martynas

Presenters: STRINGER, Joe (Isovalent / Cilium); BAUER, Lorenz (Cloudflare); PUMPUTIS, Martynas

Session Classification: BPF MC

Contribution ID: **270** Type: **not specified**

Programmable socket lookup with BPF

Monday, 9 September 2019 12:45 (45 minutes)

At Netconf 2019 we have presented a BPF-based alternative to steering packets into sockets with iptables and TPROXY extension. A mechanism which is of interest to us because it allows (1) services to share a port number when their IP address ranges don't overlap, and (2) reverse proxies to listen on all available port numbers.

The solution adds a new BPF program type BPF_INET_LOOKUP, which is invoked during the socket lookup. The BPF program is able to steer SKBs by overwriting the key used for listening socket lookup. The attach point is associated with a network namespace.

Since then, we have been reworking the solution to follow the existing pattern of using maps of socket references for redirecting packets, that is REUSEPORT_SOCKARRAY, SOCKMAP, or XSKMAP. We expect to publish the next version of BPF_INET_LOOKUP RFC patch set, which addresses the feedback from Netconf, in August.

During LPC 2019 BPF Microconference we would like to briefly recap on how BPF-driven socket lookup compares to classic bind()-based dispatch, TPROXY packet steering, and socket dispatch on TC ingress currently in development by Cilium.

Next we would like discuss low-level implementation challenges. How to best ensure that packet delivery to connected UDP sockets remains unaffected? Can a BPF_INET_LOOKUP program co-exist with reuseport groups? Is there a possibility of code sharing with REUSEPORT_SOCKARRAY implementation?

Following the implementation discussion, we will touch on performance aspects, that is what is the observed cost of running BPF during socket lookup both in SYN flood and UDP flood scenarios.

Finally, we want to go into the usability of user-space API. Redirection with a BPF map of sockets raises a question who populates the map, and if existing network applications like NGINX need to be modified in any way to receive traffic steered with this new mechanism.

The desired outcome of the discussion is to identify steps needed to graduate the patch set from an RFC series to a ready-for-review submission.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: SITNICKI, Jakub (Cloudflare); BAUER, Lorenz (Cloudflare); MAJKOWSKI, Marek (Cloudflare)

Presenters: SITNICKI, Jakub (Cloudflare); BAUER, Lorenz (Cloudflare); MAJKOWSKI, Marek (Cloudflare)

Session Classification: Networking Summit Track

Contribution ID: **271** Type: **not specified**

IoT from the point of view of view of a generic and enterprise distribution

Monday, 9 September 2019 17:00 (30 minutes)

Having been focused on IoT in Fedora for Red Hat for 3 years and the wider Arm and embedded ecosystem for a lot longer and dealing with customers that are looking to prototype large scale IoT deployments for a range of use cases while using a distribution similar to what they use in their data centre but with IoT use cases, increased security I have a bunch of war wounds and ideas about the things that work, the things that need work and the things that don't work.

The core pieces are there but there's bits missing or are incomplete, covering gpio and sensors, bluetooth and various wireless technologies through to security such as secure boot, TPM2s and IMA what are the technologies that users and customers are asking for and how can they be improved in Linux to make it easier for generic but IoT focused distros that need to address wide use cases in as generic a means as possible?

This talk will cover the technologies being used and what makes it hard for end users to consume them in order to aide discussion. How we can take things that in some cases are developed on a single device running a single variant of Linux and how we can improve the overall ecosystem on Linux.

I agree to abide by the anti-harassment policy

Yes

Primary author: ROBINSON, Peter (Red Hat)

Presenter: ROBINSON, Peter (Red Hat)

Session Classification: You, Me, and IoT MC

Contribution ID: **273** Type: **not specified**

All about Kselftest

Tuesday, 10 September 2019 12:50 (40 minutes)

Kselftest started out as an effort to enable a developer-focused regression test framework in the kernel to ensure the quality of new kernel releases. Today it is an integral part of the Linux Kernel development process to qualify Linux mainline and stable release candidates.

Shuah will go over the Kselftest framework, how to write tests that work well with the framework for effective reporting of results. In addition, Shuah will discuss how the framework is tailored for developers as well as users to serve their individual and unique needs and discuss future plans.

I agree to abide by the anti-harassment policy

Yes

Primary authors: KHAN, Shuah (The Linux Foundation); RUE, Dan

Co-author: ROXELL, Anders

Presenters: KHAN, Shuah (The Linux Foundation); ROXELL, Anders; RUE, Dan

Session Classification: Testing and Fuzzing MC

Contribution ID: **274** Type: **not specified**

SwitchDev offload optimizations

Monday, 9 September 2019 17:45 (45 minutes)

Linux has a nice SW bridge implementation which provides most of the classic Ethernet switching features. DSA and SwitchDev frameworks allow us to represent HW switch devices in Linux and potentially offload the SW forwarding to HW.

But the offloading facilities are not perfect, and there seem to be room for further improvements:

- Limiting the flooding of L2-Multicast traffic. IGMP snooping can limit the flooding of L3 traffic, but L2-Multicast traffic are always flooded.
- Today all bridge slave interfaces are put into promiscuous mode to allow learning/flooding. But if the bridge is offloaded with HW capable of doing learning/learning, then this should not be necessary.
- When not put into promiscuous mode, the struct net_device structure has a list of multicast addresses which should be received by the interface. But when VLAN sub-interfaces are created, the VLAN information is lost when addresses are installed in the mc list.
- The assumption in the bridge code is that all multicast frames goes to the CPU. But what would it actually take only to request the needed multicast frames to the CPU?
- Challenges in adding new redundancy and protection protocols to the kernel, and how to offload such protocols to HW.

The intend with the talk is to present some of the issues we are facing in adding DSA/SwitchDev drivers for existing and near-time future HW. I will have few solutions to present, but will give our thoughts on how it may be solved. Hopefully with will result in good discussions and input from the audience.

Background information: I'm working on a SwitchDev driver for a yet to be released HW Ethernet switch. It will be a TSN switch targeting industrial networks, with HW accelerators to implement redundancy protocols. CPU power are very limited, and latency are extremely important, which is why it is important for us to improve the HW offload facilities.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr NIELSEN, Allan

Presenter: Mr NIELSEN, Allan

Session Classification: Networking Summit Track

Contribution ID: **275** Type: **not specified**

Unifying trace processing ecosystems with Babeltrace

Monday, 9 September 2019 12:00 (22 minutes)

Babeltrace started out as the reference implementation of a Common Trace Format (CTF) reader. As the project evolved, many trace manipulation use-cases (merging, trimming, filtering, conversion, analysis, etc.) emerged and were implemented either as part of the Babeltrace project, on top of its APIs or through custom tools.

Today, as more tracers emerged, each using their own trace format, the tracing ecosystem has become fragmented making tools exclusive to certain tracers. The newest version of Babeltrace aims at bridging the gap between the various tracing ecosystems by making it easy to implement trace processing tools over an agnostic trace IR.

The discussion will aim at identifying the work needed to accommodate the various tracers and their associated tooling (scripts, graphical viewers, etc.) over the next releases.

I agree to abide by the anti-harassment policy

Yes

Primary author: GALARNEAU, Jérémie (EfficiOS/LTTng/Babeltrace)

Presenter: GALARNEAU, Jérémie (EfficiOS/LTTng/Babeltrace)

Session Classification: Tracing MC

Contribution ID: **276** Type: **not specified**

TrenchBoot - how to nicely boot system with Intel TXT and AMD SVM

Wednesday, 11 September 2019 16:05 (25 minutes)

TrenchBoot is a cross-community OSS integration project for hardware-rooted, late launch integrity of open and proprietary systems. It provides a general purpose, open-source DRTM kernel for measured system launch and attestation of device integrity to trust-centric access infrastructure. TrenchBoot closes the the measurement gap and reduces the need to trust system firmware. This talk will introduce TrenchBoot architecture and recent work within Oracle to launch the Linux kernel directly with Intel TXT or AMD SVM Secure Launch. It will propose mechanisms for integrating a Linux distro into a TrenchBoot system launch. DRTM-enabled capabilities for client, server and embedded platforms will be presented for consideration by the Linux community.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: KIPER, Daniel

Presenter: KIPER, Daniel

Session Classification: System Boot and Security MC

Contribution ID: **277** Type: **not specified**

disk write barriers

Wednesday, 11 September 2019 10:20 (20 minutes)

for example, for a write-ahead logging, one needs to guarantee that writes to log are completed before the corresponding data pages are written. fsync() on the log file does this, but it is an overkill for this.

I agree to abide by the anti-harassment policy

Yes

Primary author: GOLUBCHIK, Sergei

Presenter: GOLUBCHIK, Sergei

Session Classification: Databases MC

Contribution ID: **279** Type: **not specified**

BPF Tracing Tools: New Observability for Performance Analysis

Monday, 9 September 2019 13:06 (22 minutes)

Many new BPF tracing tools are about to be published, deepening our view of kernel internals on production systems. This session will summarize what has been done and what will be next with BPF tracing, discussing the challenges with taking kernel and application analysis further, and the potential kernel changes needed.

I agree to abide by the anti-harassment policy

Yes

Primary author: GREGG, Brendan (Netflix)

Presenter: GREGG, Brendan (Netflix)

Session Classification: Tracing MC

Contribution ID: **281** Type: **not specified**

An Evaluation of Host Bandwidth Manager

Wednesday, 11 September 2019 12:45 (45 minutes)

Host Bandwidth Manager (HBM) is a BPF based framework for managing per-cgroupv2 egress and ingress bandwidths in order to provide a better experience to workloads/services coexisting within a host. In particular, HBM allows us to divide a host's egress and ingress bandwidth among workloads residing in different v2 cgroups. Note that although sample BPF programs are included in the BPF patches, one can easily use different algorithms for managing bandwidth.

This talk presents an evaluation of HBM and associated BPF programs. It explores the performance of various approaches to bandwidth management for TCP flows that use Cubic, Cubic with ECN or DCTCP for their congestion control. For evaluating performance, we consider how well flows can utilize the allocated bandwidth, how many packets are dropped by HBM, increases to RTTs due to queueing, RPC size fairness, as well as RPC latencies. This evaluation is done independently for egress and ingress. In addition, we explore the use of HBM for protecting against incast congestion by also using HBM on the root v2 cgroup.

Our testing shows that HBM, with the appropriate BPF program, is very effective at managing egress bandwidths regardless of which TCP congestion control algorithm is used, preventing flows from exceeding the allocated bandwidth while allowing them to use most of their allocation. Not surprisingly, effectively managing ingress bandwidth requires ECN, and preferably DCTCP. Finally, we show that using HBM is very effective at preventing packet losses due to incast congestion, as long as we are willing to sacrifice some ingress bandwidth.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: BRAKMO, Lawrence (Facebook)

Presenter: BRAKMO, Lawrence (Facebook)

Session Classification: Networking Summit Track

Contribution ID: **282** Type: **not specified**

Kernel Boot Time Tracing

Monday, 9 September 2019 10:22 (22 minutes)

Tracing kernel boot is useful when we chase a bug in device and machine initialization, boot performance issue etc. Ftrace already supports to enable basic tracing features in kernel cmdline. However, since the cmdline is very limited and too simple, it is hard to enable complex features which are recently introduced, e.g. multiple kprobe events, trigger actions, and event histogram.
To solve this limitation, I introduce a boot time tracing feature on new structured kernel cmdline, which allows us to write complex tracing features in treed key-value style text file.
In this talk, I would like to discuss how this solves the boot time tracing, and the syntax of tracing subsystem for this structured kernel cmdline.

I agree to abide by the anti-harassment policy

Yes

Primary author: HIRAMATSU, Masami (Linaro Ltd.)

Presenter: HIRAMATSU, Masami (Linaro Ltd.)

Session Classification: Tracing MC

Contribution ID: **283** Type: **not specified**

Sharing PMU counters across compatible perf events

Monday, 9 September 2019 10:44 (22 minutes)

Hardware PMU counters are limited resources. When there are more perf events than the available hardware counters, it is necessary to use time multiplexing, and the perf events could not run 100% of time.

On the other hand, different perf events may measure the same metric, e.g., instructions. We call these perf events "compatible perf events". Technically, one hardware counter could serve multiple compatible events at the same time. However, current perf implementation doesn't allow compatible events to share hardware counters.

There are efforts to enable sharing of compatible perf events. To the best of our knowledge, the latest attempt was https://lkml.org/lkml/2019/2/26/823. Unfortunately, we haven't make much progress on this front.

At Facebook we are investing on user space sharing of compatible performance counters to reduce the need for time multiplexing and the cost of context switch when monitoring the same events in several threads and cgroups. A kernel solution would be preferable.

In the Tracing MC, we would like to discuss how we can enable PMU sharing compatible perf events. This topic may open other discussions in perf subsystem. We think this would be a fun section.

I agree to abide by the anti-harassment policy

Yes

Primary authors: LIU, Song; CARRILLO CISNEROS, David (Facebook)

Presenters: LIU, Song; CARRILLO CISNEROS, David (Facebook)

Session Classification: Tracing MC

Contribution ID: **284** Type: **not specified**

Secure and Trusted boot in OpenBMC

Wednesday, 11 September 2019 15:00 (20 minutes)

The OpenBMC project has brought modern Linux to the firmware in your new server. A missing piece of this is ensuring the firmware is the image you expect it to be running.

The next generation of BMC hardware will allow a hardware root of trust to secure the boot chain. This talk will present the a proposed design for trusted boot in OpenBMC.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: STANLEY, Joel (IBM)

Presenter: STANLEY, Joel (IBM)

Session Classification: System Boot and Security MC

Contribution ID: **287** Type: **not specified**

XDP: the Distro View

Tuesday, 10 September 2019 10:00 (1 hour)

It goes without saying that XDP is wanted more and more by everyone. Of course, the Linux distributions want to bring to users what they want and need. Even better if it can be delivered in a polished package with as few surprises as possible: receiving bug reports stemming from users' misunderstanding and from their wrong expectations does not make good experience neither for the users nor for the distro developers.

XDP presents interesting challenges to distros: from the initial enablement (what config options to choose) and security considerations, through user supportability (packets "mysteriously" disappearing, tcpdump not seeing everything), through future extension (what happens after XDP is embraced by different tools, some of those being part of the distro, how that should interact with users' XDP programs?), to more high level questions, such as user perception ("how comes my super-important use case cannot be implemented using XDP?").

Some of those challenges are long solved, some are in progress or have good workarounds, some of them are yet unsolved. Some of those are solely the distro's responsibility, some of them need to be addressed upstream. The talk will present the challenges of enabling XDP in a distro. While it will also mention the solved ones, its main focus are the problems currently unsolved or in progress. We'll present some ideas and welcome discussion about possible solutions using the current infrastructure and about future directions.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: BENC, Jiri (Red Hat)

Presenters: BENC, Jiri (Red Hat); Dr HØILAND-JØRGENSEN, Toke (RedHat); BROUER, Jesper Dangaard (Red Hat)

Session Classification: Networking Summit Track

Contribution ID: **288** Type: **not specified**

IO: Durability, Errors and Documentation

Wednesday, 11 September 2019 12:07 (20 minutes)

Postgres (and many other databases) have, until fairly recently, assumed that IO errors would a) be reliably signalled by fsync/fdatasync/… b) repeating an fsync after a failure would either result in another failure, or the IO operations would succeed.

That turned out not to be true: See also https://lwn.net/Articles/752063/

While a few improvements have been made, both in postgres and linux, the situation is still pretty bad.

From my point of view, a large part of the problem is that linux does not document what error and durability behaviour userspace can expect from certain operations.

Problematic areas for the kernel:
- The regular behaviour of durability fs related syscalls are not documented. One extreme example of that is sync_file_range (look at the warning section of the manpage)
- FS behaviour when encountering IO errors is poorly, if at all, documented. For example: there still is no documentation about the error behaviour of fsync, ext4's errors= operation reads as if it applied to all IO errors, but only applies to metadata errors.
- There is very little consistency for error behaviour between filesystems. To the degree that XFS will return different data after writeback failed than ext4.
- There is no usable interface to query / be notified of IO errors
- the rapid development of thin provisioned storage has increased the likelihood of IO errors drastically, as large parts of the IO stack treat out-of-space on the block level as an IO error

It seems worthwhile to work together to at least partially clean this up.

I agree to abide by the anti-harassment policy

Yes

Primary authors: FREUND, Andres (EnterpriseDB / PostgreSQL); Mr VONDRA, Tomas (Postgresql)

Presenters: FREUND, Andres (EnterpriseDB / PostgreSQL); Mr VONDRA, Tomas (Postgresql)

Session Classification: Databases MC

Contribution ID: **289** Type: **not specified**

Live patch services

Wednesday, 11 September 2019 18:15 (15 minutes)

Discussion about current live patch services and how we can make it more open and flexible. How we can make more open source distributions use or make their own live patch services. What we are still missing? and what we can share?

I agree to abide by the anti-harassment policy

Yes

Primary author: FERRAZZI, Alice

Presenter: FERRAZZI, Alice

Session Classification: Live Patching MC

Contribution ID: **290** Type: **not specified**

Automatically testing distribution kernel packages

Monday, 9 September 2019 12:30 (30 minutes)

Provide better kernel packages to the distribution users, is a really hot topic in distributions, as the kernel package is the fundamental part of the distribution.
One of the way to provide a better quality kernel is to implement a quality control by using automated tests.
Each distributions are probably using different tools and tests suits.
Let's share our knowledge and which tools are using.

Which Continuous integrations tools are better to use? (buildbot, jenkins)
What kernel tools are better to use for testing (lpt, kselftest)

I agree to abide by the anti-harassment policy

Yes

Primary author: FERRAZZI, Alice

Presenter: FERRAZZI, Alice

Session Classification: Distribution Kernels MC

Contribution ID: **291** Type: **not specified**

XDP bulk packet processing

Monday, 9 September 2019 15:00 (45 minutes)

It is well known that batching can often improve software performance. This is mainly because it utilizes the instruction cache in a more efficient way. From the networking perspective, the size of driver's packet processing pipeline is larger than the sizes of instruction caches. Even though NAPI batches packets over the full stack and driver execution, they are processed one by one by many large sub systems in the processing path. Initially this was raised by Jesper Brouer. With Edward Cree's listifying SKBs idea, the first implementation results look promising. How can we take this a step further and apply this technique to the XDP processing pipeline?

To do that, the proposition is to back down from preparing xdp_buff struct one-by-one, passing it to XDP program and then acting on it, but instead we would prepare in driver an array of XDP buffers to be processed. Then, we would have only a single call per NAPI budget to XDP program, which would give us back a list of actions that driver needs to take. Furthermore, the number of indirect function calls, gets reduced, as driver gets to jited BPF program via indirect function call.

In this talk I would like to present the proof-of-concept of described idea, which was yielding around 20% better XDP performance for dropping packets with touching headers memory (modified xdp1 from linux kernel's bpf samples).

However, the main focus of this presentation should be a discussion about a proper, generic implementation, which should take place after showing out the POC, instead of the current POC. I would like to consider implementation details, such as:
- would it be better to provide an additional BPF verifier logic, that when properly instrumented (make use of prologue/epilogue?), would emit BPF instructions responsible for looping over XDP program, or should we have the loop within the XDP programs?
- the mentioned POC has a whole new NAPI clean Rx interrupt routine; what should we do to make it more generic in order to make driver changes smaller?
- How about batching the XDP actions? Do all the drops first, then Tx/redirect, then the passes. Would that pay off?

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: FIJAŁKOWSKI, Maciej

Presenter: FIJAŁKOWSKI, Maciej

Session Classification: Networking Summit Track

Contribution ID: **292** Type: **not specified**

TPM 2.0 Linux sysfs interface

Wednesday, 11 September 2019 18:05 (25 minutes)

At the time of writing this paper the Linux kernel supported TPM 1.2 functionalities in sysfs. To these functionalities we include:

$ls/sys/devices/pnp0/00:04/tpm/tpm0activecapsdeviceenabledpcrsppisubsystemtimeoutscanceldevdurationsow$
ls /sys/devices/pnp0/00:04/tpm/tpm0/ppi
request response tcg_operations transition_action version vs_operations

We would expect the same or similar level of support for TPM 2.0. At least kernel should be able to request localities, change PCR banks, list PCRs, extend PCRs, clear TPM, take ownership. For now, the TPM2.0 is unusable in any way. Despite enabling all TPM options in the kernel configuration. There is a TPM 2.0 software stack, however, it has many dependencies and has to be compiled by anyone who would like to utilize TPM2.0 (packages in package managers was broken for certain distros at the time of writing the document).

Additionally, Linux has Integrity Measurement Architecture which utilizes TPM to attest the rootfs whether it has been maliciously modified. However, the only supported TPM is the one in version 1.2. Enabling it is as simple as adding a single kernel cmdline parameter: ima_tcb and defining a policy. However, it will only work with TPM 1.2 tools like tpm-tools, trousers.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: Mr PIOTR, Król (3mdeb Embedded Systems Consulting); Mr MICHAŁ, Żygowski (3mdeb Embedded Systems Consulting)

Presenters: Mr PIOTR, Król (3mdeb Embedded Systems Consulting); Mr MICHAŁ, Żygowski (3mdeb Embedded Systems Consulting)

Session Classification: System Boot and Security MC

Contribution ID: **293** Type: **not specified**

Non-UEFI-aware measured boot using coreboot, GRUB and TPM2.0

Wednesday, 11 September 2019 17:40 (25 minutes)

The main issue in using TPM2.0 in such measured boot solution is that at the moment of writing this abstract neither Trusted Grub, nor Linux kernel has TPM2.0 implementation. There are of course implementations based on UEFI systems, where bootloaders can utilize TCG EFI protocol to handle TPM. However other non-UEFI based solutions suffer from lack of TPM2.0 drivers in the bootloaders. Taking, for example, coreboot with vboot and measured mode the chain of trust ends on at verifying and measuring the MBR code. This limits the trusted boot technology for firmware solutions that do not base on UEFI specification.

As TPM2.0 is already supported in coreboot, the next stage would be enabling it in GRUB2. As a matter of fact that TPM1.2 has already been enabled in its derivative, Trusted GRUB2.0, but we consider it much unsatisfying.

Chain of trust:

coreboot + payload -(chain cuts here)-> Trusted GRUB -> kernel

Establishing a chain of trust will make SRTM (Static Root of Trust for Measurement) based on coreboot fully featured. As security solutions are used more and more widely it will help coreboot to stay up to date with all the competitor's proprietary solutions.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: KRÓL, Piotr (3mdeb Embedded Systems Consulting); Mr MICHAŁ, Żygowski (3mdeb Embedded Systems Consulting)

Presenters: KRÓL, Piotr (3mdeb Embedded Systems Consulting); Mr MICHAŁ, Żygowski (3mdeb Embedded Systems Consulting)

Session Classification: System Boot and Security MC

Contribution ID: **294** Type: **not specified**

eBPF support in the GNU Toolchain

Tuesday, 10 September 2019 13:00 (30 minutes)

This proposal covers the ongoing effort about adding eBPF support to the GNU Toolchain.

Binutils support is already upstream 1. This includes a CGEN cpu description, assembler, disassembler and linker. A GCC backend will be submitted for inclusion upstream before September.

Both the binutils and GCC ports will be briefly described, and then a list of points will be discussed with the kernel community, and also with the llvm people present.

The main goals of the sessions are:
1) to ensure the port is useful to the eBPF community and
2) to agree on ABI (with the kernel) and interoperability (with llvm.)

1 https://sourceware.org/ml/binutils/2019-05/msg00306.html

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr MARCHESI, Jose E. (Oracle Inc, GNU Project)

Presenter: Mr MARCHESI, Jose E. (Oracle Inc, GNU Project)

Session Classification: Toolchains MC

Contribution ID: **295** Type: **not specified**

Collaboration/unification around unit testing frameworks

Tuesday, 10 September 2019 12:20 (30 minutes)

From the initial reactions and interest I have seen wrt. KTF (http://heim.ifi.uio.no/~knuto/ktf/, https://github.com/oracle/ktf) and the discussions on LKML around KUnit (https://lkml.org/lkml/2018/11/29/82), it seems there's a general belief that some form of unit test framework like these can be a good addition to the tools and infrastructure already available in the kernel.

It seems however that different people have different notions about what and how such a framework should ideally look, and what features belong there. I'd like to see if we can bring that discussion forward by focusing on some of these items, where people seem to have quite differing views depending on where they come from. Here is a non extensive list of some topics that seems to pop up when this gets discussed:

- "Purity" of unit testing - what constitutes a "unit" in the kernel?
- Testing kernel code - user space vs kernel space? (both useful)
- Immediate development/debugging requirements vs longer term needs
- Driver/hardware interaction testing?
- "Neat"-factor
- ease of use
- Network testing (more than 1 kernel involved)
- How to best integrate with existing test infrastructure in the kernel
- Unification and simpliciation options …

I'd like to make a short intro into this, and hopefully we can have some good exchange based on that.

I agree to abide by the anti-harassment policy

Yes

Primary author: Dr OMANG, Knut (Oracle)

Presenter: Dr OMANG, Knut (Oracle)

Session Classification: Testing and Fuzzing MC

Contribution ID: **296** Type: **not specified**

Seamless transparent encryption with BPF and Cilium

Tuesday, 10 September 2019 17:00 (45 minutes)

Providing encryption in dynamic environments where nodes are added and removed on-the-fly and services spin-up and are then torn-down frequently, such as Kubernetes, has numerous challenges. Cilium, an open source software package for providing and transparently securing network connectivity, leverages BPF and the Linux encryption capabilities to provide L3/L7 encryption and authentication at the node and service layers. Giving users the ability to apply encryption either to entire nodes or on specified services. Once configured through a high level feature flag (–enable-encrypt-l3, –enable-encrypt-l7) the management is transparent to the user. Cilium will manage and ensure traffic is encrypted allowing for auditing of encrypted/unencrypted flows via a monitoring interface to ensure compliance.

In this talk we will show how Cilium accomplishes this in the Linux datapath and control plane. As well as discuss how Cilium with Linux and BPF fits into the evolving encryption standards and frameworks such as IPsec, mTLS, Secure Production Identity Framework For Everyone (SPIFFE), and Istio. Looking forward we propose a set of extensions to the Linux kernel, specifically to the BPF infrastructure, to ease the adoption and improve the efficiency of these protocols. Specifically, we will look at a series of BPF helpers, possible hardware support, scaling to thousands of nodes, and transparently enforcing policy on encrypted sessions.

Finally to show this is not mere slide-ware we will show a demo Cilium implementing transparent encryption.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr FASTABEND, John (Isovalent)

Presenter: Mr FASTABEND, John (Isovalent)

Session Classification: Networking Summit Track

Contribution ID: **297** Type: **not specified**

TPM2 Security in the face of bus interposers

Wednesday, 11 September 2019 17:00 (20 minutes)

TPM2 introduced a plain text authorization scheme with the idea that the system using the TPM should now whether the transport was secure. The presence of interposers on the bus, either as physical devices

https://www.nccgroup.trust/us/our-research/tpm-genie/

Or as compromised pre-boot firmware make this threat a reality. A NULL seed based scheme has been proposed for Linux

https://lore.kernel.org/linux-integrity/1540193596.3202.7.camel@HansenPartnership.com/

we should discuss if this is the best we can do and if it is how should we extend it to the layers below that use the TPM (like UEFI and grub).

I agree to abide by the anti-harassment policy

Yes

Primary author: BOTTOMLEY, James (IBM)

Presenter: BOTTOMLEY, James (IBM)

Session Classification: System Boot and Security MC

Contribution ID: **298** Type: **not specified**

Can we agree on what needs to happen to get shiftfs upstream

Tuesday, 10 September 2019 17:30 (30 minutes)

Since Canonical is now shipping it I think we can all agree it solves a problem and we just need to get the patches into shape for upstream submission. Can we discuss a pathway for doing that.

I agree to abide by the anti-harassment policy

Yes

Primary authors: BOTTOMLEY, James (IBM); BRAUNER, Christian; Mr FORSHEE, Seth (Canonical)

Presenters: BOTTOMLEY, James (IBM); BRAUNER, Christian; Mr FORSHEE, Seth (Canonical)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **299** Type: **not specified**

Formal Methods for the Linux Kernel

Tuesday, 10 September 2019 15:45 (45 minutes)

This BoF session aims to bring together Linux kernel developers who have an interest in formal methods (or formal methods experts with an interest in kernel development). Topics for discussion:

- A poll of formal methods currently used in the context of the Linux kernel: SPIN, TLA+, CBMC, herd, plain English etc.
- High level design specification vs. low level algorithm modelling. What properties people seek to verify?
- Bridging the gap between formal models and the actual code: built-in run-time verification (e.g. lockdep), CBMC-based kernel self-tests, event trace analysis. Any other suggestions?
- How to encourage wider adoption of formal methods by kernel developers (e.g. help reduce the ramp-up time)
- Potential for a consolidated repository of formal specs (or in-kernel directory)

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: MARINAS, Catalin

Presenter: MARINAS, Catalin

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **300** Type: **not specified**

CRIU: Reworking vDSO proxification, syscall restart

Tuesday, 10 September 2019 19:30 (20 minutes)

We have a number of unsolved time and vdso related issues in CRIU.

- Syscall restart: if a task Checkpoint interrupted a syscall, on restore CRIU blindely starts again the syscall (executing SYSCALL/SYSENTER/INT80/etc instruction with the original regset). It works OKish, but not with time blocking syscalls i.e., poll(), nanosleep(), futex() and etc. For this purpose, Glibc and vDSO use restart_syscall(). Which won't work in CRIU as kernel is not aware of interrupted syscall. To solve those issues I suggest to extend PTRACE_GET_SYSCALL_INFO with information from task_struct->restart_block. This way on restore criu will be able to adjust syscall arguments on application Restore.
- vDSO proxification. There is a chance that between Checkpoint and Restore events vDSO code may change. That may be in example, migration to another node or updating the kernel on the very same node. The old vDSO code can't be used anymore as vvar physical page can be missing [migration to an older kernel] or it may have different offsets. CRIU deals with that by mmaping old vdso code and patching entries with jumps to a new vdso. That's far from being perfect: the original application could have being Checkpointed while executing vdso code, but luckily we haven't got any reports about crashes on restore so far! Addressing this problem, we could add symbol table to vvar and got/plt tables to vdso, allowing CRIU to do linker job on restore by patching relocations on older vdso to newer vvar. The other approach would be making proxification process more correct: we could single-step application on Checkpoint from bytes that might be patched on Restore (JUMP_PATCH_SIZE). But additional trouble would be signals which may have being delivered while application was executing the very same bytes. That can be solved probably with hijacking SA_RESTORER..

I agree to abide by the anti-harassment policy

Yes

Primary authors: SAFONOV, Dmitry; VAGIN, Andrei

Presenters: SAFONOV, Dmitry; VAGIN, Andrei

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **301** Type: **not specified**

Scheduler domains and cache bandwidth

Monday, 9 September 2019 17:45 (15 minutes)

The Linux Kernel scheduler represents a system's topology by the means of scheduler domains. In the common case, these domains map to the cache topology of the system.

The Cavium ThunderX is an ARMv8-A 2-node NUMA system, each node containing 48 CPUs (no hyperthreading). Each CPU has its own L1 cache, and CPUs within the same node will share a same L2 cache.

Running some memory-intensive tasks on this system shows that, within a given NUMA node, there are "socklets" of CPUs. Executing those tasks (which involve the L2 cache) on CPUs of the same "socklet" leads to a reduction of per-task memory bandwidth.
On the other hand, running those same tasks on CPUs of different "socklets" (but still within the same node) does not lead to such a memory bandwidth reduction.

While not truly equivalent to sub-NUMA clustering, such a system could benefit from a more fragmented scheduler domain representation, i.e. grouping these "socklets" in different domains.

This talk will be an opportunity to discuss ways for the scheduler to leverage this topology characteristic and potentially change the way scheduler domains are built.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: SCHNEIDER, Valentin (Arm Ltd)

Presenter: SCHNEIDER, Valentin (Arm Ltd)

Session Classification: Scheduler MC

Contribution ID: **303** Type: **not specified**

Making Networking Queues a First Class Citizen in the Kernel

Tuesday, 10 September 2019 15:45 (45 minutes)

XDP (the eXpress Data Path) is a new method in Linux to process packets at L2 and L3 with really high performance. XDP has already been deployed for use cases involving ingress packet filtering, or transmission back through the ingress interface, are already well supported today. However, as we expand the use cases that involve the XDP_REDIRECT action, e.g., to send packets to other devices, or zero-copy them to userspace sockets, it becomes challenging to retain the high performance of the simpler operating modes.

One of the keys to get good performance for these advanced use cases, is effective use of dedicated hardware queues (on both Rx and Tx), as this makes it possible to split traffic over multiple CPUs, with no synchronization overhead in the fast path. The problem with using hardware queues like this is that they are a constrained resource, but are hidden from the rest of the kernel: Currently, each driver allocates queues according to its own whims, and users have little or no control over how the queues are used or configured.

In this presentation we discuss an abstraction that makes it possible to keep track of queues in a vendor-neutral way: We implement a new submodule in the Linux networking core that drivers can register their queues to. Other pieces of code can then allocate and free individual queues (or sets of them) satisfying certain properties (e.g., “a Tx/Rx pair”, or “one queue per core”). This submodule also makes sure that the queues get IDs that are hardware independent, so that they can easily be used by other components. We show how this could be exposed to userspace, and how it can interact with the existing REDIRECT primitives, such as device maps.

Finally if there is time, we would like to discuss a related problem: often a userspace program wants to express its configuration not in terms of queue IDs, but in terms of a set of packets it wants to process (e.g., by specifying an IP address). So how do we change user space APIs that use queue IDs to be able to use something more meaningful such as properties of the packet flow that a user wants? To solve this second problem, we propose to introduce a new bind option in AF_XDP that takes a simple description of the traffic that is desired (e.g. “VLAN ID 2”, “IP address fc00:dead:cafe::1”, or “all traffic on a netdev”). This hides queue IDs from userspace, but will use the new queue logic internally to allocate and configure an appropriate queue.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary authors: KARLSSON, Magnus (Intel); TÖPEL, Björn (Intel); DANGAARD BROUER, Jesper (RedHat); HÖILAND-JÖRGENSEN, Toke (RedHat); KICINSKI, Jakub (Netronome); MIKITYANSKIY, Maxim (Mellanox)

Presenters: KARLSSON, Magnus (Intel); TÖPEL, Björn (Intel); DANGAARD BROUER, Jesper (RedHat); HÖILAND-JÖRGENSEN, Toke (RedHat); KICINSKI, Jakub (Netronome); MIKITYANSKIY, Maxim (Mellanox)

Session Classification: Networking Summit Track

Contribution ID: **304** Type: **not specified**

Soft Affinity

Wednesday, 11 September 2019 15:45 (45 minutes)

When multiple instances of workloads are consolidated in same host it is good practice to partition them for best performance. For e.g give a NUMA node parition to each instance. Currently Linux kernel provides two interfaces to hard parition: sched_setaffinity system call or cpuset.cpus cgroup. But this doesn't allow one instance to burst out of its partition and use available CPUs from other partitions when they are idle. Running all instances free range without any affinity, on the other hand, suffers from cache coherence overhead across sockets (NUMA nodes) when all instances are busy. To achieve the best of both worlds introduce new Soft Affinity feature that allows the scheduler to chose a preferred set of CPUs when they are idle but burst out of it and use the allowed set if they are all busy.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: MAZUMDAR, Subhra

Presenter: MAZUMDAR, Subhra

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **305** Type: **not specified**

Making it easier for distros to package kernel source

Monday, 9 September 2019 10:40 (20 minutes)

Every distro has to package the kernel tree using their own unique package files. Some parts of the process are built-in to the kernel source and are easy: build, install, and headers. Some parts are not: configs, devel package, userspace tools package, tests, distro versioning, changelogs, custom patches, etc.

This discussion revolves around some of the issues and difficulties a distro maintainer faces when packaging the kernel source code. What changes can we agree to push upstream to make our lives easier.

Further, discuss possibilities of plugging in distro packaging into the kernel source tree (through external means or internal hooks). This allows developers to quickly build (from a common devel env) a particular distro-like kernel for proper testing.

Sample topics include:
* config maintainence for distros
* top-level Makefile hooks for distros
* make devel_install -like command
* distro versioning

I agree to abide by the anti-harassment policy

Yes

Primary author: ZICKUS, Don (Red Hat)

Presenter: ZICKUS, Don (Red Hat)

Session Classification: Distribution Kernels MC

Contribution ID: **306** Type: **not specified**

libcamera: Unifying camera support on all Linux systems

Tuesday, 10 September 2019 16:00 (15 minutes)

The libcamera project was started at the end of 2018 to unify camera support on all Linux systems (regular Linux distributions, Chrome OS and Android). In 9 months it has produced an Android Camera HAL implementing the LIMITED profile for Chrome OS, and work is in progress to implement the FULL profile. Two platforms are currently supported (Intel IPU3 and Rockchip ISP), with work on additional platforms ongoing.

First-class Android support doesn't only depend on the effort put on libcamera, but requires cooperation with the Android community and industry. In particular, libcamera has reached a point where it needs to discuss the following topics:

- Feedback from the Android community on the overall architecture
- Feedback from SoC vendors on the device-specific interfaces and device support in general
- Next development steps for libcamera to support the LEVEL 3 profile
- Contribution of libcamera to Project Treble and integration in AOSP
- Future of the Android Camera HAL API and feedback from libcamera team

Discussions regarding the shortcomings of the Linux kernel APIs for Android camera support, and how to address them, is also on-topic as libcamera suffers from the same issues.

As the Linux Plumbers Conference will gather developers from the Google Android teams, from the Android community, from the Linux kernel media community and from the libcamera project, we strongly believe this is a unique occasion to design the future of camera support in Linux systems all together.

I agree to abide by the anti-harassment policy

Yes

Primary author: PINCHART, Laurent (Ideas on Board Oy)

Presenter: PINCHART, Laurent (Ideas on Board Oy)

Session Classification: Android MC

Contribution ID: **308** Type: **not specified**

Life at a Networking Vendor – Keeping up with the Joneses

Tuesday, 10 September 2019 12:00 (45 minutes)

Working for a networking hardware vendor can be an extremely rewarding experience for a kernel developer. The rate at which new features are accepted in the kernel also provides lots of motivation to develop new features that showcase hardware capabilities. This could be done by adding new support for dataplane offloads via cls flower, netfilter, or switchdev (if we still think it exists!). In-driver support for pre-SKB packet processing via XDP and AF_XDP also provide a chance for developers to search for new software optimizations in their driver receive and transmit path.

In addition to thinking about what is happening upstream, developers at hardware vendors regularly find themselves managing internal and external expectations from those responsible for developing features that are not always exclusive to the Linux kernel. This could range from frameworks like DPDK and VPP that run on Linux or completely different OSes/stacks to functionality that is available without software interaction.

There is no quicker way to develop new features and resolve issues than to have direct contact with hardware and firmware developers. The goal of this talk will be to share some experiences balancing the expectations of customers and partners along with those of the community.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: GOSPODAREK, Andy (Broadcom)

Presenter: GOSPODAREK, Andy (Broadcom)

Session Classification: Networking Summit Track

Contribution ID: **310** Type: **not specified**

Monitoring and Stabilizing the In-Kernel ABI

Tuesday, 10 September 2019 15:15 (15 minutes)

The Kernel's API and ABI exposed to Kernel modules is not something that is usually maintained in upstream. Deliberately. In fact, the ability to break APIs and ABIs can greatly benefit the development. Good reasons for that have been stated multiple times. See e.g. Documentation/process/stable-api-nonsense.rst.
The reality for distributions might look different though. Especially - but not exclusively - enterprise distributions aim to guarantee ABI stability for the lifetime of their released kernels while constantly consuming upstream patches to improve stability and security for said kernels. Their customers rely on both: upstream fixes and the ability to use the released kernels with out-of-tree modules that are compiled and linked against the stable ABI.

In this talk I will give a brief overview about how this very same requirement applies to the Kernels that are part of the Android distribution. The methods presented here are reasonable measures to reduce the complexity of the problem by addressing issues introduced by ABI influencing factors like build toolchain, configurations, etc.

While we focus on Android Kernels, the tools and mechanisms are generally useful for Kernel distributors that aim for a similar level of stability. I will talk about the tools we use (like e.g. libabigail), how we automate compliance checking and eventually enforce ABI stability.

I agree to abide by the anti-harassment policy

Yes

Primary author: MÄNNICH, Matthias (Google)

Presenter: MÄNNICH, Matthias (Google)

Session Classification: Android MC

Contribution ID: **311** Type: **not specified**

DRM/KMS for Android, adoption and upstreaming

Tuesday, 10 September 2019 18:15 (15 minutes)

A short update on the status of DRM/KMS ecosystem adoption and how Google is improving verification of the DRM display drivers in Android devices.

I agree to abide by the anti-harassment policy

Yes

Primary author: DELVA, Alistair (Google)

Presenter: DELVA, Alistair (Google)

Session Classification: Android MC

Contribution ID: **312** Type: **not specified**

Android Virtualization (esp. Camera, DRM)

Tuesday, 10 September 2019 15:45 (15 minutes)

An update on how we plan to enable multimedia testing on our 'cuttlefish' virtual platform. Overview of missing components for graphics virtualization.

I agree to abide by the anti-harassment policy

Yes

Primary author: DELVA, Alistair (Google)

Presenter: DELVA, Alistair (Google)

Session Classification: Android MC

Contribution ID: **313** Type: **not specified**

Handling memory pressure on Android

Tuesday, 10 September 2019 17:45 (15 minutes)

Topic will discuss how Android framework utilizes new kernel features to better handle memory pressure. This includes app compaction, new kill strategies and improved process tracking using pidfds.

I agree to abide by the anti-harassment policy

Yes

Primary author: BAGHDASARYAN, Suren (Google)

Presenter: BAGHDASARYAN, Suren (Google)

Session Classification: Android MC

Contribution ID: **314** Type: **not specified**

UEFI and TianoCore update

Wednesday, 11 September 2019 15:20 (20 minutes)

The UEFI forum is rolling out a new “code first” process, to be available for both UEFI and ACPI specifications, in order to speed up time between initial definition and upstream support.

The UEFI self-certification testsuite (SCT) has been open sourced.

UEFI interface implementation in U-Boot now sufficient for GRUB use (and more) across multiple distributions..

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LINDHOLM, Leif (Linaro, TianoCore, GRUB)

Presenter: LINDHOLM, Leif (Linaro, TianoCore, GRUB)

Session Classification: System Boot and Security MC

Contribution ID: **315** Type: **not specified**

scheduler: uclamp usage on Android

Tuesday, 10 September 2019 18:30 (15 minutes)

Android has been using an out-of-tree schedtune cgroup controller for task performance boosting of time-sensitive processes. Introduction of utilization clamping (uclamp) feature in the Linux kernel opens up an opportunity to adopt an upstream mechanism for achieving this goal. The talk will present our plans on adopting uclamp in Android.

I agree to abide by the anti-harassment policy

Yes

Primary author: BAGHDASARYAN, Suren (Google)

Presenter: BAGHDASARYAN, Suren (Google)

Session Classification: Android MC

Contribution ID: **316** Type: **not specified**

Do we need CAP_BPF_ADMIN?

Wednesday, 11 September 2019 16:07 (23 minutes)

Currently, most BPF functionality requires CAP_SYS_ADMIN or CAP_NET_ADMIN. However, in many cases, CAP_SYS_ADMIN/CAP_NET_ADMIN gives the user more than enough permissions. For example, tracing users need to load BPF programs and access BPF maps, so they need CAP_SYS_ADMIN. However, they don't need to modify the system, so CAP_SYS_ADMIN adds significant risk.

To better control BPF functionality, this is time to think about CAP_BPF_ADMIN (or even multiple CAP_BPF_*s). In this BPF MC, we would like to discuss whether we need CAP_BPF_ADMIN, and what CAP_BPF_ADMIN would look like. We will present survey of major BPF use cases, and identify use cases that may benefit from a new CAP. Then, we will discuss which syscalls/commands should be gated by the new CAP. We expect constructive discussions between the BPF folks and security folks.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: LIU, Song

Presenter: LIU, Song

Session Classification: BPF MC

Contribution ID: **318** Type: **not specified**

netfilter hardware offloads

Monday, 9 September 2019 17:00 (45 minutes)

With the advent of the the flow rule and flow block API, ethtool_rx, netfilter and tc can share the same infrastructure to represent hardware offloads.

This presentation discusses the reuse of the existing infrastructure originally implemented by tc, such as the netdev_ops->ndo_setup_tc() interface and the TC_SETUP_CLSFLOWER classifier.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: Mr NEIRA, Pablo

Presenter: Mr NEIRA, Pablo

Session Classification: Networking Summit Track

Contribution ID: **319** Type: **not specified**

kernelCI: testing a broad variety of hardware

Tuesday, 10 September 2019 10:00 (35 minutes)

kernelCI: testing a broad variety of hardware

The Linux kernel runs on an extremely wide range of hardware, but with the rapid pace of kernel development, it's difficult to ensure the full range of supported hardware is adequately tested.

The kernelCI project is a small, but growing project, focused on testing the core kernel on diverse set of architectures, boards and compilers using distributed labs to test hardware anywhere on the planet.

The goal of this presentation is to give a very brief overview of the project, and discuss the near-term future goals and plans.

Recently added:
- support for clang-build kernels
- more arches: ARC, RISC-V, MIPS

The future:
- official Linux Foundation project launching
- more tests: subsystem-focused test suites
- more labs with more hardware
- scaling of infrastructure
- better reporting

I agree to abide by the anti-harassment policy

Yes

Primary authors: HILMAN, Kevin (BayLibre); TUCKER, Guillaume (Collabora Limited)

Presenters: HILMAN, Kevin (BayLibre); TUCKER, Guillaume (Collabora Limited)

Session Classification: Testing and Fuzzing MC

Contribution ID: **320** Type: **not specified**

How we're using ebpf in Android networking

Tuesday, 10 September 2019 17:15 (15 minutes)

A short update on eBPF in Android networking:

- how we're using ebpf in Android P on 4.9+ for statistics collection and Q on 4.9+ for xlat464 offload, with a focus on the sorts of problems we've run into
- where we'd like to go, ie. future plans with regard to xlat464/forwarding/nat offload and XDP.

I agree to abide by the anti-harassment policy

Yes

Primary authors: ŻENCZYKOWSKI, Maciej (Google); COLITTI, Lorenzo (Google)

Session Classification: Android MC

Contribution ID: **321** Type: **not specified**

Kernel Debugging Tools

Monday, 9 September 2019 15:00 (45 minutes)

For many years developers have leveraged gdb or crash to look at kernel crash dumps on linux. Although those tools have served us well, it can sometimes be difficult to navigate the crash dump to find the information you really need. In this talk, we would like to present some new tools that make it easier to debug kernel crash dumps and enhance kernel developer's ability to root causes problems the first time they happen. We will present information about the following tools:

- crash-python
- drgn
- sdb

In addition, we're looking for interested members to join the kernel debugging community to continue to build on these tools, provide feedback, and help generate ideas on how we can make kernel crash dump debugging simpler.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: WILSON, George (Delphix)

Presenters: WILSON, George (Delphix); SANDOVAL, Omar; DIMITROPOULOS, Serapheim

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **322** Type: **not specified**

Scaling container policy management with kernel features

Wednesday, 11 September 2019 10:00 (45 minutes)

Cilium is an open source project which implements the Container Network Interface (CNI) to provide networking and security functions in modern application environments. The primary focus of the Cilium community recently has been on scaling these functions to support thousands of nodes and hundreds of thousands of containers. Such environments impose a high rate of churn as containers and nodes appear and leave the cluster. For each change, the networking plugin needs to handle the incoming events and ensure that policy is in sync with network configuration state. This creates a strong incentive to efficiently interpret and map down cluster events into the required Linux networking configuration to minimize the window during which there are discrepancies between the desired and realized state in the cluster—something that is made possible through eBPF and other kernel features.

Cilium realizes these policy and container events through the use of many aspects of the networking stack, from rules to routes, tc to socket hooks, skb->mark to the skb->cb. Modelling the changes to datapath state involves a non-trivial amount of work in the userspace daemon to structure the desired state from external entities and allow incremental adjustments to be made, keeping the amount of work required to handle an event proportional to its impact on the kernel configuration. Some aspects of datapath configuration such as the implementation of L7 policy have gone through multiple iterations, which provides a window for us to explore the past, present and future of transparent proxies.

This talk will discuss the container policy model used by Cilium to apply whitelist filtering of requests at layers 3, 4 and 7; memoization techniques used to cache intermediate policy computation artifacts; and impacts on dataplane design and kernel features when considering large container based deployments with high rates of change in cluster state.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: STRINGER, Joe (Cilium.io)

Presenter: STRINGER, Joe (Cilium.io)

Session Classification: Networking Summit Track

Contribution ID: **323** Type: **not specified**

Using SCEV to establish pre and post-conditions over BPF code

Wednesday, 11 September 2019 17:20 (20 minutes)

Currently, the BPF verifier has to "execute" code at least once and then it can prune branches when it detects the state is the same. In this session we would like to cover a technique called Scalar Evolution (SCEV) which is used by LLVM and GCC to perform optimization passes such as identifying and promoting induction variables and do worst case trip analysis over loops. At its most basic usage SCEV finds the start value of variables, the variables stride and the variables ending value over a block of code. Building a SCEV pass into the BPF verifier would allow us to create a set of pre and post conditions over blocks of BPF codes.

We see this as potentially useful to avoid "executing" loops in the verifier and instead allowing the verifier to check pre-conditions before entering the loop. And additionally establishing pre and post conditions on function calls to avoid having to execute the verifier on functions repeatedly. We suspect this will likely be necessary to support shared libraries for example.

The goal of the session will be to do a brief introduction to SCEV. Provide a demonstration of some early prototype work that can build pre and post conditions over blocks of BPF code. Then discuss next steps for possible inclusion.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr FASTABEND, John (Isovalent)

Presenter: Mr FASTABEND, John (Isovalent)

Session Classification: BPF MC

Contribution ID: **325** Type: **not specified**

Solving issues associated with modules and supplier-consumer dependencies

Tuesday, 10 September 2019 15:30 (15 minutes)

GKI or any ARM64 Linux distro needs a single ARM64 kernel that works across all SoCs. But having a single ARM64 kernel that works across all SoCs has a lot of hurdles. One of them, is getting all the SoC specific devices to be handed off cleanly from the bootloader to the kernel even when all their drivers are loaded as modules. Getting this to work correctly involves proper ordering of events like module loading, device initialization and device boot state clean up. This discussion is about the work that's being done in the upstream kernel to automate and facilitate the proper ordering of these events.

I agree to abide by the anti-harassment policy

Yes

Primary author: KANNAN, Saravana (Google)

Presenter: KANNAN, Saravana (Google)

Session Classification: Android MC

Contribution ID: **326** Type: **not specified**

Linaro Kernel Functional Testing (LKFT): functional testing of android common kernels

Tuesday, 10 September 2019 17:30 (15 minutes)

As part of the Android Microconference:

Linux Kernel Functional Test is a system to detect kernel regressions across the range of mainline, LTS and Android Common kernels. It is able to run a variety of operating systems from Linux to Android across an array of systems under test. You're probably thinking in terms of standard test suites like CTS, VTS, LTP, kselftest and so on and you're be right. We'll talk about how things have been going over the past year and some of the challenges face when testing at scale.

The 'F' in LKFT is for Functional, and during this interactive session we will explore how to continue to make strides beyond pass/fail tests. Kernel regressions aren't just an option that once worked now is failing. They also include degradation in performance. The session will explore the recent add to LKFT involving the Energy Aware Scheduler (EAS) with boards that have power probes on hardware. Last we'll talk about audio and some things we've been exploring with testing the audio stack on Android.

I agree to abide by the anti-harassment policy

Yes

Primary author: GALL, Tom (Linaro)

Presenter: GALL, Tom (Linaro)

Session Classification: Android MC

Contribution ID: **327** Type: **not specified**

flattening the hierarchy discussion

Monday, 9 September 2019 17:30 (15 minutes)

There is a presentation in the refereed track on flattening the CPU controller runqueue hierarchy, but it may be useful to have a discussion on the same topic in the scheduler microconference.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: VAN RIEL, Rik (Facebook)

Presenter: VAN RIEL, Rik (Facebook)

Session Classification: Scheduler MC

Contribution ID: **328** Type: **not specified**

Linux in Safety Critical Systems

Tuesday, 10 September 2019 15:00 (45 minutes)

It looks like there may well be enough critical folks present to have a good BOF about safety and linux. Topics can include safety processes and methodologies, tooling to support analysis, security update concerns, etc. Basically, if you're interested in using Linux in safety critical systems come join, and we'll see where the conversation goes.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: STEWART, Kate (Linux Foundation)

Co-author: BULWAHN, Lukas (BMW AG)

Presenters: STEWART, Kate (Linux Foundation); BULWAHN, Lukas (BMW AG)

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **329** Type: **not specified**

Update on objtool - Power

Wednesday, 11 September 2019 16:10 (10 minutes)

A quick update on the objtool port on Power, what is the current state and what more needs to be done. Also, discuss how do we integrate it upstream.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr BABULAL, Kamalesh

Presenter: Mr BABULAL, Kamalesh

Session Classification: Live Patching MC

Contribution ID: **330** Type: **not specified**

Reuse host JIT back-end as offload back-end

Wednesday, 11 September 2019 17:00 (20 minutes)

eBPF offload is a powerful feature on modern SmartNICs used to accelerate XDP or TC based BPF. The current kernel eBPF offload infrastructure was introduced for the Netronome NFP based SmartNICs, these were based around a proprietary ISA and had some specific verifier requirements.

In the near future this may be joined by SmartNICs using public ISA's such as RISC-V and Arm which also happen to be used as host CPUs. This talk will discuss the implications of reusing these ISAs and other back-end features for offload to a sea of cores as well as how much of a host CPU back-ends can be reused and what additional infrastructure may be needed. As an example we will use the current work on a many core RISC-V processor ongoing within.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr WANG, JIONG (Netronome Systems)

Presenter: Mr WANG, JIONG (Netronome Systems)

Session Classification: BPF MC

Contribution ID: **331** Type: **not specified**

Making Livepatching Infrastructure Better

Wednesday, 11 September 2019 18:00 (15 minutes)

Currently testing/stressing of livepatching infrastructure is limited to the creation of livepatching module for the reported CVE/Security issues. Continuous testing of the infrastructure is required, it can be achieved by randomly selecting the patch(s) posted over kernel mailing list to improve and fix the bugs seen in the infrastructure. I would like to discuss the in house framework used for testing livepatch. The discussion would help to understand/provide feedback on how/what should be tested. The improvements which can be made to improve the testing coverage.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr BABULAL, Kamalesh

Presenter: Mr BABULAL, Kamalesh

Session Classification: Live Patching MC

Contribution ID: **332** Type: **not specified**

SGX upstreaming status and challenges

Wednesday, 11 September 2019 15:40 (25 minutes)

The presentation gives an overview of what has been implemented in the SGX patch set and what there is still left to do. The presentation goes through the known blockers for upstreaming. In particular, access control related issues will be discussed.

I agree to abide by the anti-harassment policy

Yes

Primary author: SAKKINEN, Jarkko

Presenter: SAKKINEN, Jarkko

Session Classification: System Boot and Security MC

Contribution ID: **333** Type: **not specified**

Generic Kernel Image (GKI) progress

Tuesday, 10 September 2019 15:00 (15 minutes)

A year ago at Linux Plumbers, we talked about a generic Android kernel that boots and runs reasonably well on any Android device. This talk shares the progress we've made so far on many fronts. A summary of those work streams, problems we discovered along the way and our plans for them. We will talk about our short term goals and long term vision to get Android device kernels as close to the mainline as possible.

I agree to abide by the anti-harassment policy

Yes

Primary author: PATIL, Sandeep (Google)

Presenter: PATIL, Sandeep (Google)

Session Classification: Android MC

Contribution ID: **335** Type: **not specified**

Emulated storage features (eg sdcardfs)

Tuesday, 10 September 2019 16:15 (15 minutes)

Update and discussion of emulated storage on Android

I agree to abide by the anti-harassment policy

Yes

Primary author: ROSENBERG, Daniel (Google)

Presenter: ROSENBERG, Daniel (Google)

Session Classification: Android MC

Contribution ID: **336** Type: **not specified**

What SQLite Devs Wish Linux Filesystem Devs Knew About SQLite

Wednesday, 11 September 2019 12:00 (7 minutes)

(1) SQLite is the most widely used database in the world. There are probably in excess of 300 billion active SQLite databases on Linux devices. SQLite is a significant client of the Linux filesystem - perhaps the largest single non-streaming client, especially on small devices such as phones.

(2) Unlike other relational database engines, SQLite tends to live out on the edge of the network, not in the datacenter.

(3) An SQLite database is a single ordinary file in the filesystem. The database file format is well-defined and stable. The US Library of Congress designates SQLite database files as a recommended format for long-term archive storage of structured data.

(4) SQLite is not a client/server database. SQLite is a library. The application makes a function call that contains SQL text and SQLite translates that SQL into a sequence of filesystem operations that implement the desired operation, all within the same thread. There is no messaging and no IPC. There is no server process that hangs around to coordinate access to the database file.

(5) SQLite does not get to choose a filesystem type or mount options. It has to make due with whatever is at hand. Therefore, SQLite really wants to be able to discover filesystem properties at run-time, so that it can tune its behavior for maximum performance and reliability.

(6) Diagrams showing how SQLite creates the illusion of atomic commit on a non-atomic filesystem.

I agree to abide by the anti-harassment policy

Yes

Primary author: Dr HIPP, Richard (SQLite)

Presenter: Dr HIPP, Richard (SQLite)

Session Classification: Databases MC

Contribution ID: **337** Type: **not specified**

Mathematizing the latency

Wednesday, 11 September 2019 11:01 (30 minutes)

We know that reducing the sections with preemption and IRQ disabled reduces the latency, also that IRQs influences on it, but some cases are hard to catch. For example, in the old jump label update, there was a burst of IPIs causing latency spikes. Such non-periodic behavior is hard to mathematize. As a side effect, this adds pessimism to "possible formulas" that tries to define the worst-case latency, mainly regarding IRQs. Daniel would like to discuss his idea about the possible approaches to this problem, without adding unpractical pessimism.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Primary author: BRISTOT DE OLIVEIRA, Daniel (Red Hat, Inc.)

Presenter: BRISTOT DE OLIVEIRA, Daniel (Red Hat, Inc.)

Session Classification: Real Time MC

Contribution ID: **338** Type: **not specified**

What happened in kernel live patching over the last year

Wednesday, 11 September 2019 15:00 (10 minutes)

A short summary of a development in kernel live patching over the last year. There have been many improvements since LPC in Vancouver, but there are still some outstanding issues. Not all attendees might closely follow live-patching mailing list and therefore the talk should be a good starting point for the microconference.

I agree to abide by the anti-harassment policy

Yes

Primary author: BENEŠ, Miroslav

Presenter: BENEŠ, Miroslav

Session Classification: Live Patching MC

Contribution ID: **339** Type: **not specified**

API for state changes made by callbacks

Wednesday, 11 September 2019 17:00 (30 minutes)

The discussion should focus on an API for handling state of changes made by callbacks. It was already discussed as a global state handling at the last LPC in Vancouver. New ideas have occurred since then. The discussion should also include patch versioning, stickiness and transition reversal.

Patches submitted upstream so far:
https://www.spinics.net/lists/live-patching/msg05063.html
http://lore.kernel.org/r/20190719074034.29761-1-pmladek@suse.com

I agree to abide by the anti-harassment policy

Yes

Primary author: MLÁDEK, Petr

Presenter: MLÁDEK, Petr

Session Classification: Live Patching MC

Contribution ID: **340** Type: **not specified**

Rethinking late module patching

Wednesday, 11 September 2019 15:10 (30 minutes)

Current livepatch implementation supports late patching of modules when they are loaded (and unpatching when unloaded). It has caused headaches and LPC microconference is a good opportunity to discuss the future of the feature. There were attempt to deny the module removal. Introduction of patch module dependencies could also simplify the code and issue a lot. On the other hand, such solutions could make livepatch less flexible. It is necessary to weigh their advantages and downsides properly.

I agree to abide by the anti-harassment policy

Yes

Primary author: BENEŠ, Miroslav

Presenter: BENEŠ, Miroslav

Session Classification: Live Patching MC

Contribution ID: **341** Type: **not specified**

Source-based livepatch creation tooling

Wednesday, 11 September 2019 15:40 (30 minutes)

At last year's Live Patching MC, an approach to automating source based live patch creation had been proposed. The implementation made good progress since then, in particular an initial release of the "klp-ccp" utility has been published (https://github.com/SUSE/klp-ccp) recently. Its purpose is to handle the transformation of patched kernel parts into self-contained live patch source code files.

However, klp-ccp is only part of a larger pipeline and in working further towards fully automated live patch creation, it's worth to discuss how the individual pieces are best glued together.

Among the open questions are:

- Can klp-ccp and klp-convert make use of the same source of information for resolving symbols to instances from target kernel?
- Can we perhaps introduce some convention for accessing the IPA optimization reports created by GCC's -fdump-ipa-clones?
- Can we introduce some mechanism for obtaining the original kernel compilation's compiler flags each?

I agree to abide by the anti-harassment policy

Yes

Primary author: STANGE, Nicolai (SUSE)

Presenter: STANGE, Nicolai (SUSE)

Session Classification: Live Patching MC

Contribution ID: **342** Type: **not specified**

klp-convert and livepatch relocations

Wednesday, 11 September 2019 17:30 (30 minutes)

The kernel already supports special livepatch relocation types enable several interesting livepatch modules use cases:

- Access to symbols outside of normal C scoping rules
- Deferred access to yet-to-be loaded kernel module symbols
- Support for architecture-specific special sections like altinstructions and paravirt instructions

Although the kernel supports loading livepatch modules with these features, there remains no easy in-kernel means of creating such relocation types. The klp-convert patchset adds this functionality to the kernel build system, reducing dependencies on out-of-tree livepatch build mechanisms.

Talk about the current state of the klp-convert patchset: what has been implemented, what is being worked on, and what issues are still outstanding.

I agree to abide by the anti-harassment policy

Yes

Primary author: LAWRENCE, Joe (Red Hat)

Presenter: LAWRENCE, Joe (Red Hat)

Session Classification: Live Patching MC

Contribution ID: **343** Type: **not specified**

Do we need a Livepatch Developers Guide?

Wednesday, 11 September 2019 16:20 (10 minutes)

Over the past few years, kernel engineers have been busy implementing livepatch support features (the consistency model, atomic replace, shadow variables, etc.) to increase potential livepatch patch coverage. At the same time, more and more vendors have adopted livepatching to solve continuous uptime/update problems.

As the livepatch feature set grows and matures and demand for livepatch patches rise, developers will be seeking guidance and best practices when writing livepatch patches. The kpatch project addressed this with a "Patch Author Guide" on its github site. This document covers several common patch writing FAQ and techniques, but it is currently very kpatch-specific.

The kernel livepatch subsystem has some great technical documentation in Documentation/livepatch. Talk about the state of those docs and whether they are approachable and complete enough for livepatch patch authors. Do we need a wholesale "Patch Author Guide" or can we adopt some of the same FAQ and technique details from the kpatch guide.

I agree to abide by the anti-harassment policy

Yes

Primary author: LAWRENCE, Joe (Red Hat)

Presenter: LAWRENCE, Joe (Red Hat)

Session Classification: Live Patching MC

Contribution ID: **344** Type: **not specified**

Taking RISC-V to the Datacenter

Monday, 9 September 2019 13:00 (15 minutes)

What's it going to take to allow us to make the benefits of the RISC-V architecture available in centralized computing systems? Are there some things we need to be working on right now to pave the way for future success here? How can the state of the ARM architecture help us understand this problem?

This presentation will explore the technical decisions made in designing a data-center scale ARM server. Then, highlight the technical and product differences between x86 and ARM systems, and show where RISC-V is heading in relation to those. Finally, describe a few places where focusing the RISC-V Linux architecture in a particular direction may help enable datacenter-class machines while still allowing the existing embedded Linux roadmap to succeed.

I agree to abide by the anti-harassment policy

Yes

Primary author: PACKARD, Keith (SiFive)

Presenter: PACKARD, Keith (SiFive)

Session Classification: RISC-V MC

Contribution ID: **353** Type: **not specified**

Update on the LLVM port of the Linux Kernel

Tuesday, 10 September 2019 12:00 (30 minutes)

This topic will cover how the LLVM port of the linux kernel is going, where it's being used, and some of the pain points still plaguing those efforts. The issues the kernel port is having almost always are the same issues that other projects have porting from gcc to clang.

A lot of updates have been made to both the kernel and to llvm/clang which are making both projects better.

I agree to abide by the anti-harassment policy

Yes

Primary author: WEBSTER, Behan

Presenter: WEBSTER, Behan

Session Classification: Toolchains MC

Contribution ID: **354** Type: **not specified**

Opening session

Tuesday, 10 September 2019 15:00 (10 minutes)

Presenter: GRABER, Stéphane (Canonical Ltd.)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **355** Type: **not specified**

Closing session

Tuesday, 10 September 2019 19:50 (10 minutes)

Presenter: GRABER, Stéphane (Canonical Ltd.)

Session Classification: Containers and Checkpoint/Restore MC

Contribution ID: **356** Type: **not specified**

Power Management and Thermal Control BoF Sessions

Tuesday, 10 September 2019 19:00 (1 hour)

Session Classification: Power Management and Thermal Control MC

Contribution ID: **358** Type: **not specified**

Welcome

Presenters: BORKMANN, Daniel (Cilium.io); MILLER, David (Red Hat Inc.)

Session Classification: Networking Summit Track

Contribution ID: **359** Type: **not specified**

Open Session

Wednesday, 11 September 2019 10:00 (5 minutes)

Quick introduction of people. Frame discussion. Will be quick I promise.

I agree to abide by the anti-harassment policy

I confirm that I am already registered for LPC 2019

Primary author: BLACK, Daniel (IBM)

Presenter: BLACK, Daniel (IBM)

Session Classification: Databases MC

Contribution ID: **360** Type: **not specified**

Break

Session Classification: Databases MC

Contribution ID: **361** Type: **not specified**

Conclusion

Wednesday, 11 September 2019 13:22 (8 minutes)

From discussions to code. Where it goes from here?

Primary author: BLACK, Daniel (IBM)

Presenter: BLACK, Daniel (IBM)

Session Classification: Databases MC

Contribution ID: **362** Type: **not specified**

Improving Buffered I/O

Tuesday, 10 September 2019 10:45 (45 minutes)

What I'd like to get to is to discuss that buffered IO basically sucks for databases with high throughput, and direct IO sucks for databases that aren't individually well tuned, and is not adaptive to memory pressure at all.

Buffered IO is slow, until recently only synchronous, has double buffering issues and writeback is hard to control.

Direct IO requires that the application's equivalent of the page-cache is well tuned for the workload - but most installations don't have DBAs to do so, and in a lot of environments it's unrealistic to give all databases the peak required memory. In contrast to that the kernel page-cache adapts reasonably to changing workloads, caching data for the applications that need it most.

Input both from the developers of other databases and from the kernel side would be very welcome.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Yes

Primary authors: Mr VONDRA, Tomas (Postgresql); FREUND, Andres (EnterpriseDB / PostgreSQL)

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **363** Type: **not specified**

Discussions on kselftest

Wednesday, 11 September 2019 12:45 (45 minutes)

Presenter: KAHN, Shuah

Session Classification: Kernel Summit Track

Contribution ID: **364** Type: **not specified**

Deep Argument Inspection and Seccomp

Monday, 9 September 2019 15:45 (45 minutes)

Presenter: BRAUNER, Christian

Session Classification: Kernel Summit Track

Contribution ID: **365** Type: **not specified**

PCI microconference follow-up

Wednesday, 11 September 2019 12:45 (45 minutes)

Discussion around topics related
to PCI specifications and microconference follow up

- Root complex integrated endpoints
- Native host controllers link management
- VFIO/IOMMU/PCI follow up

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Yes

Primary author: PIERALISI, Lorenzo

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **366** Type: **not specified**

Persistent Memory as Memory

Tuesday, 10 September 2019 17:00 (45 minutes)

Discussion of using Persistent Memory as first- (or second-) class memory.

Google has a successful prototype of a software-managed "Transparent" mode for 3dXPoint / AEP memory, but we're working on re-designing this into something that is more supportable and at least partially upstreamable.

We want to open a discussion of how we can represent this "swap"-like use of AEP sensibly.

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Yes

Primary author: ADAMS, Jonathan (Google)

Presenter: ADAMS, Jonathan (Google)

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **367** Type: **not specified**

Tracing MC follow-up BoF

Tuesday, 10 September 2019 12:45 (45 minutes)

Follow up on the tracing microconference

Topics to be discussed:
- Perf related events
- Histogram sql syntaxes

I agree to abide by the anti-harassment policy

Yes

I confirm that I am already registered for LPC 2019

Yes

Primary author: ROSTEDT, Steven

Session Classification: Birds of a feather (BoF)

Track Classification: Birds of a Feather (BoF)

Contribution ID: **369** Type: **not specified**

Welcome Reception

Monday, 9 September 2019 19:00 (2 hours)

Sete Colinas Terrace

19:00-21:00

Same location as Lunch.

I agree to abide by the anti-harassment policy

I confirm that I am already registered for LPC 2019

Contribution ID: **370** Type: **not specified**

Bus service for Evening Party

Wednesday, 11 September 2019 19:30 (30 minutes)

Buses will start circulating at 7:30PM.

Last return bus is at 11PM

Session Classification: Networking Summit Track

Contribution ID: **371** Type: **not specified**

Closing Party @ Centro Cultural de Belém (CCB)

Wednesday, 11 September 2019 20:00 (2h 55m)

Closing Party will be held at the Centro Cultural de Belém (CCB). Accessible by bus starting from the entrance (upstairs) behind the LPC registration desk.

Last return bus: 11PM

Session Classification: Networking Summit Track

Contribution ID: **372** Type: **not specified**

Last Bus service - 11PM

Wednesday, 11 September 2019 22:55 (5 minutes)

Session Classification: Networking Summit Track

Contribution ID: **373** Type: **not specified**

Closing Plenary (Floriana I/II/III)

Wednesday, 11 September 2019 18:45 (45 minutes)

Session Classification: Networking Summit Track

Contribution ID: **374** Type: **not specified**

ARM v8.5 Memory Tagging Extension

Tuesday, 10 September 2019 18:45 (15 minutes)

What is MTE and why we do need to add the support for the Linux Userspace? Memory Tagging is an ARMv8.5 extension and provides architectural support for run-time detection of various classes of memory errors. It can be used to aid with software debugging to eliminate vulnerabilities before they can be exploited (i.e. bounds violations, use-after-free,use-after-return, use-out-of-scope and use-before-initialisation).

What does MTE support for a Linux Userspace application mean? We can divide this topic in two main parts: userspace awareness (initialization, relaxation of the ABI, paging support for the tags, swapping) and userspace debugging (enable tagging in the userspace memory allocator).

The presentation will briefly introduce the MTE concepts trying to put them in the context of what is required for the Linux OS support. It will focus then on the enablement of the ARMv8.5 extension in the userspace trying to analyze the challenges that we faced during the endeavor: memory alignment, tags management, memory impact, etc.

Primary author: FRASCINO, Vincenzo (ARM)

Presenter: FRASCINO, Vincenzo (ARM)

Session Classification: Android MC