Linux Plumbers Conference
2020

Report of Contributions
Data-race detection in the Linux kernel

Monday, 24 August 2020 10:00 (45 minutes)

In this talk, we will discuss data-race detection in the Linux kernel. The talk starts by briefly providing background on data races, how they relate to the Linux-kernel Memory Consistency Model (LKMM), and why concurrency bugs can be so subtle and hard to diagnose (with a few examples). Following that, we will discuss past attempts at data-race detectors for the Linux kernel and why they never reached production quality to make it into the mainline Linux kernel. We argue that a key piece to the puzzle is the design of the data-race detector: it needs to be as non-intrusive as possible, simple, scalable, seamlessly evolve with the kernel, and favor false negatives over false positives. Following that, we will discuss the Kernel Concurrency Sanitizer (KCSAN) and its design and some implementation details. Our story also shows that a good baseline design only gets us so far, and most important was early community feedback and iterating. We also discuss how KCSAN goes even further, and can help detect concurrency bugs that are not data races.

Tentative Outline:
- Background
  - What are data races?
  - Concurrency bugs are subtle: some examples
- Data-race detection in the Linux kernel
  - Past attempts and why they never made it upstream
  - What is a reasonable design for the kernel?
  - The Kernel Concurrency Sanitizer (KCSAN)
  - Design
  - Implementation
  - Early community feedback and iterate!
- Beyond data races
  - Concurrency bugs that are not data races
  - How KCSAN can help find more bugs
- Conclusion

Keywords: testing, developer tools, concurrency, bug detection, data races

References: https://lwn.net/Articles/816850/, https://lwn.net/Articles/816854/

I agree to abide by the anti-harassment policy

I agree

Primary author: ELVER, Marco (Google)
Presenter: ELVER, Marco (Google)
Session Classification: LPC Refereed Track

Track Classification: LPC Refereed Track (Closed)
Core Scheduling: Taming Hyper-Threads to be secure

Monday, 24 August 2020 09:00 (45 minutes)

The core idea behind core scheduling is to have SMT (Simultaneous Multi Threading) on and make sure that only trusted applications run concurrently on the hardware threads of a core. If there is no group of trusting applications runnable on the core, we need to make sure that remaining hardware threads are idle while applications run in isolation on the core. While doing so, we should also consider the performance aspects of the system. Theoretically it is impossible to reach the same level of performance where all hardware threads are allowed to run any runnable application. But if the performance of core scheduling is worse than or the same as that without SMT, we do not gain anything from this feature other than added complexity in the scheduler. So the idea is to achieve a considerable boost in performance compared to SMT turned off for the majority of production workloads.

This talk is continuation of the core scheduling talk and micro-conference at LPC 2019. We would like to discuss the progress made in the last year and the newly identified use-cases of this feature.

Progress has been made in the performance aspects of core scheduling. Couple of patches addressing the load balancing issues with core scheduling, have improved the performance. And stability issues in v5 have been addressed as well.

One area of criticism was that the patches were not addressing all cases where untrusted tasks can run in parallel. Interrupts are one scenario where the kernel runs on a cpu in parallel with a user task on the sibling. While two user tasks running on the core could be trusted, when an interrupt arrives on one cpu, the situation changes. Kernel starts running in interrupt context and the kernel cannot trust the user task running on the other sibling cpu. A prototype fix has been developed to fix this case. One gap that still exists is the syscall boundary. Addressing the syscall issue would be a big hit to performance, and we would like to discuss possible ways to fix it without hurting performance.

Lastly, we would also like to discuss the APIs for exposing this feature to userland. As of now, we use CPU controller CGroups. During the last LPC, we had discussed this in the presentation, but we had not decided on any final APIs yet. ChromeOS has a prototype which uses prctl(2) to enable the core scheduling feature. We would like to discuss possible approaches suitable for all use cases to use the core scheduling feature.

I agree to abide by the anti-harassment policy

I agree

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

Co-author: FERNANDES, Joel

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

Session Classification: LPC Refereed Track
**Track Classification:**  LPC Refereed Track (Closed)
Morello and the challenges of a capability-based ABI

Tuesday, 25 August 2020 09:00 (45 minutes)

The Morello project is an experimental branch of the Arm architecture for evaluating the deployment and impact of capability-based security. This experimental ISA extension builds on concepts from the CHERI project from Cambridge University.

As experimentation with Morello on Linux is underway, this talk will focus on the pure-capability execution environment, where all pointers are represented as 128-bit capabilities with tight bounds and limited permissions. After a brief introduction to the Morello architecture, we will outline the main challenges to overcome for the kernel to support a pure-capability userspace. Beyond the immediate issue of adding a syscall ABI where all pointers are 128 bits wide, the kernel is expected to honour the restrictions associated with user capability pointers when it dereferences them, in order to prevent the confused deputy problem.

These challenges can be approached in multiple ways, with different trade-offs between robustness, maintainability, and invasiveness. We will attempt at covering a few of these approaches, in the hope of generating useful discussions with the community.

I agree to abide by the anti-harassment policy

I agree

Primary author: BRODSKY, Kevin (Arm)
Presenter: BRODSKY, Kevin (Arm)
Session Classification: LPC Refereed Track
Track Classification: LPC Refereed Track (Closed)
A theorem for the RT scheduling latency (and a measuring tool too!)

Friday, 28 August 2020 07:00 (45 minutes)

Defining Linux as an RTOS might be risky when we are outside of the kernel community. We know how and why it works, but we have to admit that the black-box approach used by cyclicstest to measure the PREEMPT_RT’s primary metric, the scheduling latency, might not be enough for trying to convince other communities about the properties of the kernel-rt.

In the real-time theory, a common approach is the categorization of a system as a set of independent variables and equations that describe its integrated timing behavior. Two years ago, Daniel presented a model that could explain the relationship between the kernel events and the latency, and last year he showed a way to observe such events efficiently. Still, the final touch, the definition of the bound for the scheduling latency of the PREEMPT_RT using an approach accepted by the theoretical community was missing. Yes, it was.

Closing the trilogy, Daniel will present the theorem that defines the scheduling latency bound, and how it can be efficiently measured, not only as a single value but as the composition of the variables that can influence the latency. He will also present a proof-of-concept tool that measures the latency. In addition to the analysis, the tool can also be used in the definition of the root cause of latency spikes, which is another practical problem faced by PREEMPT_RT developers and users. However, discussions about how to make the tool more developers-friendly are still needed, and that is the goal of this talk.

The results presented in this talk was published at the ECRTS 2020, a top-tier academic conference about real-time systems, with reference to the discussions made in the previous edition of the Linux Plumbers.

I agree to abide by the anti-harassment policy

I agree

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

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

Session Classification: LPC Refereed Track

Track Classification: LPC Refereed Track (Closed)
Graphical user sessions have been plagued with various performance related issues. Sometimes these are simply bugs, but often enough issues arise because workstations are loaded with other tasks. In this case a high memory, IO or CPU use may cause severe latency issues for graphical sessions. In the past, people have tried various ways to improve the situation, from running without swap to heuristically detecting low memory situations and triggering the OOM. These techniques may help in certain conditions but also have their limitations.

GNOME and other desktops (currently KDE) are moving towards managing all applications using systemd. This change in architecture also means that every application is placed into a separate cgroup. These can be grouped to separate applications from essential services and they can also be adjusted dynamically to ensure that interactive applications have the resources they need. Examples of possible interventions are allocating more CPU weight to the currently focused application, creating memory and IO latency guarantees for essential services (compositor) or running oomd to kill applications when there is memory pressure.

The talk will look at what GNOME (and KDE) currently does in this regard and how well it is working at at this point so far. This may show areas where further improvements in the stack are desirable.

I agree to abide by the anti-harassment policy

I agree

Primary author: BERG, Benjamin
Presenter: BERG, Benjamin
Session Classification: LPC Refereed Track
Track Classification: LPC Refereed Track (Closed)
First investigations about Kernel Address Space Isolation (ASI) were presented at LPC last year as a way to mitigate some CPU hyper-threading data leaks possible with speculative execution attacks (like L1 Terminal Fault (L1TF) and Microarchitectural Data Sampling (MDS)). In particular, Kernel Address Space Isolation aims to provide a separate kernel address space for KVM when running virtual machines, in order to protect against a malicious guest VM attacking the host kernel using speculative execution attacks.

https://www.linuxplumbersconf.org/event/4/contributions/277/

At that time, a first proposal for implementing KVM Address Space Isolation was available. Since then, new proposals have been submitted. The implementation has become much more robust and it now provides a more generic framework which can be used to implement KVM ASI but also Kernel Page Table Isolation (KPTI).

Currently, RFC version 4 of Kernel Address Space Isolation is available. The proposal is divided into three parts:

- Part I: ASI Infrastructure and PTI
  https://lore.kernel.org/lkml/20200504144939.11318-1-alexandre.chartre@oracle.com/

- Part II: Decorated Page-Table
  https://lore.kernel.org/lkml/20200504145810.11882-1-alexandre.chartre@oracle.com/

- Part III: ASI Test Driver and CLI
  https://lore.kernel.org/lkml/20200504150235.12171-1-alexandre.chartre@oracle.com/

This presentation will show progress and evolution of the Kernel Address Space Isolation project, detail the kernel ASI framework and how it is used to implement KPTI and KVM ASI. It also looks forward to discuss possible ways to integrate the project upstream, concerns about making changes in some of the nastiest corners of the x86, and kernel page table management improvement, in particular page table creation and population.

I agree to abide by the anti-harassment policy

I agree

Primary author: CHARTRE, Alexandre (Oracle)
Presenter: CHARTRE, Alexandre (Oracle)
Session Classification: LPC Refereed Track
Track Classification: LPC Refereed Track (Closed)
Configuring a kernel for safety critical applications

Monday, 24 August 2020 08:00 (45 minutes)

For security there are various projects which provide guidelines on how to configure a secure kernel - e.g., Linux Self Protection Project. In addition there are security enhancements which have been added to the Linux kernel by various groups - e.g., grsecurity or PAX security patch. We are looking to define appropriate guidelines for safety enhancements to the Linux kernel. The session will focus on the following:

1. Define the use cases (primarily in automotive domain) and the need for safety features.
2. Define criteria for safe kernel configurations.
3. Define a preliminary proposal for a serious workgroup to define requirements for relevant safety enhancements.

Note that the emphasis is 100% technical, and not related in any way to safety assessment processes. I will come with an initial set of proposals, to be discussed and for follow up.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Dr COPPERMAN, Elana (Mobileye)
Presenter:      Dr COPPERMAN, Elana (Mobileye)
Session Classification:  LPC Refereed Track

Track Classification:  LPC Refereed Track (Closed)
SoC support lifecycle in the kernel

Wednesday, 26 August 2020 08:00 (45 minutes)

The world of system-on-chip computing has changed drastically over the past years with the current state being much more diverse as the industry keeps moving to 64-bit processors, to little-endian addressing, to larger memory capacities, and to a small number of instruction set architectures.

In this presentation, I discuss how and why these changes happen, and how we can find a balance between keeping older technologies working for those that rely on them, and identifying code that has reached the end of its useful life and should better get removed.

I agree to abide by the anti-harassment policy

I agree

Primary author: BERGMANN, Arnd (Linaro)
Presenter: BERGMANN, Arnd (Linaro)
Session Classification: Kernel Summit
Track Classification: Kernel Summit
seccomp feature development

Wednesday, 26 August 2020 09:00 (45 minutes)

As outlined in https://lore.kernel.org/lkml/202005181120.971232B7B@keescook/ the topics include:

- fd passing
- deep argument inspection
- changing structure sizes
- syscall bitmasks

Specifically, seccomp needs to grow the ability to inspect Extensible Argument syscalls, which requires that it inspect userspace memory without Time-of-Check/Time-of-Use races and without double-copying. Additionally, since the structures can grow and be nested, there needs to be a way to deal with flattening the arguments into a linear buffer that can be examined by seccomp’s BPF dialect. All of this also needs to be handled by the USER_NOTIF implementation. Finally, fd passing needs to be finished, and there needs to be an exploration of syscall bitmasks to augment the existing filters to gain back some performance.

I agree to abide by the anti-harassment policy

I agree

Primary author: COOK, Kees (Google)

Presenter: COOK, Kees (Google)

Session Classification: Kernel Summit

Track Classification: Kernel Summit
DAMON: Data Access Monitoring Framework for Fun and Memory Management Optimizations

Wednesday, 26 August 2020 10:00 (45 minutes)

Background

In an ideal world, memory management provides the optimal placement of data objects under accurate predictions of future data access. Current practical implementations, however, rely on coarse information and heuristics to keep the instrumentation overhead minimal. A number of memory management optimization works were therefore proposed, based on the finer-grained access information. Lots of those, however, incur high data access pattern instrumentation overhead, especially when the target workload is huge. A few of the others were able to keep the overhead small by inventing efficient instrumentation mechanisms for their use case, but such mechanisms are usually applicable to their use cases only.

We can list up below four requirements for the data access information instrumentation that must be fulfilled to allow adoption into a wide range of production environments:

- **Accuracy.** The instrumented information should be useful for DRAM level memory management. Cache-level accuracy would not highly required, though.
- **Light-weight overhead.** The instrumentation overhead should be low enough to be applied online while making no impact on the performance of the main workload.
- **Scalability.** The upper-bound of the instrumentation overhead should be controllable regardless of the size of target workloads, to be adopted in general environments that could have huge workloads.
- **Generality.** The mechanism should be widely applicable.

DAMON: Data Access MONitor

DAMON is a data access monitoring framework subsystem for the Linux kernel that designed to mitigate this problem. The core mechanisms of DAMON called ‘region based sampling’ and ‘adaptive regions adjustment’ make it fulfill the requirements. Moreover, its general design and flexible interface allow not only the kernel code but also the user space can use it.

Using this framework, therefore, the kernel’s core memory management mechanisms including reclamation and THP can be optimized for better memory management. The memory management optimization works that incurring high instrumentation overhead will be able to have another try. In user space, meanwhile, users who have some special workloads will be able to write personalized tools or applications for deeper understanding and specialized optimizations of their systems.

In addition to the basic monitoring, DAMON also provides a feature dedicated to semi-automated memory management optimizations, called DAMON-based Operation Schemes (DAMOS). Using this feature, the DAMON users can implement complex data access aware optimizations in only a few lines of human-readable schemes descriptions.
Overhead and Performance

We evaluated DAMON’s overhead, monitoring quality, and usefulness using 25 realistic workloads on my QEMU/KVM based virtual machine.

DAMON is lightweight. It increases system memory usage by only -0.39% and consumes less than 1% CPU time in the typical case. It slows target workloads down by only 0.63%.

DAMON is accurate and useful for memory management optimizations. An experimental DAMON-based operation scheme for THP removes 69.43% of THP memory overhead while preserving 37.11% of THP speedup. Another experimental DAMON-based reclamation scheme reduces 89.30% of residential sets and 22.40% of system memory footprint while incurring only 1.98% runtime overhead in the best case.

Current Status of The Project

Development of DAMON started in 2019, and several iterations were presented in academic papers[1,2,3], the kernel summit of last year[4], and an LWN article[4]. The source code is available[6] for use and modification, the patchsets[7] are periodically being posted for review.

Agenda

I will briefly introduce DAMON and share how it has evolved since last year’s kernel summit talk. I will introduce some new features, including the DAMON-based operation schemes. There will be a live demonstration and I will show performance evaluation results. I will outline plans and the roadmap of this project, leading to a Q&A session to collect feedback with a view on getting it ready for general use and upstream inclusion.


3 Yunjae Lee, Yunhee Kim, and Heon. Y. Yeom, Lightweight Memory Tracing for Hot Data Identification, In Cluster computing, 2020. (Accepted but not published yet)


6 https://github.com/sjp38/linux/tree/damon/master

7 https://lore.kernel.org/linux-mm/20200525091512.30391-1-sjpark@amazon.com/

I agree to abide by the anti-harassment policy

I agree

Primary author: Dr PARK, SeongJae (Amazon)

Presenter: Dr PARK, SeongJae (Amazon)
Session Classification:  Kernel Summit

Track Classification:  Kernel Summit
Recent changes in the kernel memory accounting (or how to reduce the kernel memory footprint by ~40%)

Tuesday, 25 August 2020 10:00 (45 minutes)

Not a long time ago memcg accounting used the same approach for all types of pages. Each charged page had a pointer at the memory cgroup in the struct page. And it held a single reference to the memory cgroup, so that the memory cgroup structure was pinned in the memory by all charged pages.

This approach was simple and nice, but it didn’t work well for some kernel objects, which are often shared between memory cgroups. E.g. an inode or a denty can outlive the original memory cgroup by far, because it can be actively used by someone else. Because there was no mechanism for the ownership change, the original memory cgroup was pinned in the memory, so that only a very heavy memory pressure could get rid of it. This lead to a so called dying memory cgroups problem: an accumulation of dying memory cgroups with uptime.

It has been solved by switching to an indirect scheme, where slab pages didn’t reference the memory cgroup directly, but used a memcg pointer in the corresponding slab cache instead. The trick was that the pointer can be atomically swapped to the parent memory cgroup. In combination with slab caches reference counters it allowed to solve the dying memcg problem, but made the corresponding code even more complex: dynamic creation and destruction of per-memcg slab caches required a tricky coordination between multiple objects with different life cycles.

And the resulting approach still had a serious flow: each memory cgroup had its own set of slab caches and corresponding slab pages. On a modern system with many memory cgroups it resulted in a poor slab utilization, which varied around 50% in my case. This made the accounting quite expensive: it almost doubled the kernel memory footprint.

To solve this problem the accounting has to be moved from a page level to an object level. If individual slab objects can be effectively accounted on individual level, there is no more need to create per-memcg slab caches. A single set of slab caches and slab pages can be used by all memory cgroups, which brings the slab utilization back to >90% and saves ~40% of total kernel memory. To keep the reparenting working and not reintroduce the dying memcg problem, an intermediate accounting vessel called obj_cgroup is introduced. Of course, some memory has to be used to store an objcg pointer for each slab object, but it’s by far smaller than consequences of a poor slab utilization. The proposed new slab controller implements a per-object accounting approach. It has been used on the Facebook production hosts for several months and brought significant memory savings (in a range of 1 GB per host and more) without any known regressions.

The object-level approach can be used to add an effective accounting of objects, which are by their nature not page-based: e.g. percpu memory. Each percpu allocation is scattered over multiple pages, but if it’s small, it takes only a small portion of each page. Accounting such objects was nearly impossible on a per-page basis (duplicating chunk infrastructure will result in a terrible overhead), but with a per-object approach it’s quite simple. Patchset implements it. Percpu memory is getting more and more used as a way to solve the contention problem on a multi-CPU system. Cgroups internals and bpf maps seem to be biggest users at this time, but likely new use cases will be added. It can easily take hundreds on MBs on a host, so if it’s not account edit creates an issue in container memory isolation.

Links:
1. https://lore.kernel.org/linux-mm/20200527223404.1008856-1-guro@fb.com/
2. https://lore.kernel.org/linux-mm/20200528232508.1132382-1-guro@fb.com/
I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr GUSHCHIN, Roman (Facebook)
Presenter:  Mr GUSHCHIN, Roman (Facebook)
Session Classification:  LPC Refereed Track

Track Classification:  LPC Refereed Track (Closed)
Opening session

Monday, 24 August 2020 07:00 (5 minutes)

I agree to abide by the anti-harassment policy

Primary author:  GRABER, Stéphane (Canonical Ltd.)
Presenter:  GRABER, Stéphane (Canonical Ltd.)
Session Classification:  Containers and Checkpoint/Restore MC
Track Classification:  Containers and Checkpoint/Restore MC
> Overlayfs new features

*Monday, 24 August 2020 08:10 (30 minutes)*

Containers are by far the biggest use case for overlayfs. Yet, there seems to be very little cross talk between overlayfs and containers mailing lists.

This talk is going to present some opt-in overlayfs features that were added in recent years (redirect_dir, index, nfs_export, xino, metacopy).

Most of those features have not been enabled by most container runtimes, because of various reasons:

- Requires more development in userspace (image migration)
- Unrelated runtime bugs (mount leaks)
- Mismatch for containers needs
- Lack of promotion

This talk is about giving the opportunity to container runtime developers to better understand what they may get from overlayfs.

This talk is not about containers wish list from overlayfs, because usersns overlayfs mount needs 45 minutes on its own...

I agree to abide by the anti-harassment policy

I agree

**Primary author:** GOLDSTEIN, Amir (CTERA Networks)

**Presenter:** GOLDSTEIN, Amir (CTERA Networks)

**Session Classification:** Containers and Checkpoint/Restore MC

**Track Classification:** Containers and Checkpoint/Restore MC
Extensible Syscalls

Thursday, 27 August 2020 07:00 (45 minutes)

Most Linux syscall design conventions have been established through trial and error. One well-known example is the missing flag argument in a range of syscalls that triggered the addition of a revised version of these syscalls. Nowadays, adding a flag argument to keep syscalls extensible is an accepted convention recorded in our kernel docs.

In this session we’d like to propose and discuss a few simple conventions that have proven useful over time and a few new ones that were just established recently with the addition of new in-kernel apis. Ideally these conventions would be added to the kernel docs and maintainers encouraged to use them as guidance when new syscalls are added.

We believe that these conventions can lead to a more consistent (and possibly more pleasant) uapi going forward making programming on Linux easier for userspace. They hopefully also prevent new syscalls running into various design pitfalls that have lead to quirky or cumbersome apis and (security) bugs.

Topics we’d like to discuss include the use of structs versioned by size in syscalls such as openat2(), sched_{set,get}_attr(), and clone3() and the associated api that we added last year, whether new syscalls should be allowed to use nested pointers in general and specifically with an eye on being conveniently filterable by seccomp, the convention to always use unsigned int as the type for register-based flag arguments instead of the current potpourri of types, naming conventions when revised versions of syscalls are added, and - ideally a uniform way - how to test whether a syscall supports a given feature.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  BRAUNER, Christian (Canonical);  SARAI, Aleksa (SUSE LLC)

Presenters:  BRAUNER, Christian (Canonical);  SARAI, Aleksa (SUSE LLC)

Session Classification:  Kernel Summit

Track Classification:  Kernel Summit
Kernel documentation

Thursday, 27 August 2020 08:00 (45 minutes)

The long process of converting the kernel’s documentation into RST is finally coming to an end…what has that bought us? We have gone from a chaotic pile of incomplete, crufty, and un-integrated docs to a slightly better organized pile of incomplete, crufty, slightly better integrated docs. Plus we have the infrastructure to make something better from here.

What are the next steps for kernel documentation? What would we really like our docs to look like, and how might we find the resources to get them to that point? What sorts of improvements to the build infrastructure would be useful? I’ll come with some ideas (some of which you’ve certainly heard before) but will be more interested in listening.

I agree to abide by the anti-harassment policy

I agree

Primary author:  CORBET, Jonathan (Linux Plumbers Conference)

Presenter:  CORBET, Jonathan (Linux Plumbers Conference)

Session Classification:  Kernel Summit

Track Classification:  Kernel Summit
Write once, herd everywhere

Tuesday, 25 August 2020 07:00 (45 minutes)

With Linux Kernel Memory Model introduced into kernel, litmus tests have been proven to be a powerful tool to analyze and design parallel code. More and more C litmus tests are written, some of which are merged into Linux mainline.

Actually the herd tool behind LKMM have models for most of mainstream architectures: litmus tests in asm code are supported. So in theory, we can verify a litmus test in different versions (C and asm code), and this will help us on 1) verifying the correct of LKMM and 2) test the implementation of parallel primitives in a particular architecture, by comparing the results of exploring the state spaces of different versions of litmus tests.

This topic will present some work to make it possible to translate between litmus tests (mostly C to asm code). The work provides an interface for architecture maintainers to provide their rules for the litmus translation, in this way, we can verify the consistency between LKMM and the implementation of parallel primitives, and this could also help new architectures to provide parallel primitives consistent with LKMM.

This topic will introduce the overview of the translation and hopefully some discussion will be made during or after the topic on the interface.

I agree to abide by the anti-harassment policy

I agree

Primary author:    FENG, Boqun
Presenter:         FENG, Boqun
Session Classification:  LPC Refereed Track
Track Classification:   LPC Refereed Track (Closed)
Restricted kernel address spaces

Thursday, 27 August 2020 09:00 (45 minutes)

This proposal is recycled from the one I’ve suggested to LSF/MM/BPF [0]. Unfortunately, LSF/MM/BPF was cancelled, but I think it is still relevant.

Restricted mappings in the kernel mode may improve mitigation of hardware speculation vulnerabilities and minimize the damage exploitable kernel bugs can cause.

There are several ongoing efforts to use restricted address spaces in Linux kernel for various use cases:
* speculation vulnerabilities mitigation in KVM [1]
* support for memory areas with more restrictive protection that the defaults (“secret”, or “protected” memory) [2] [3] [4]
* hardening of the Linux containers [ no reference yet :) ]

Last year we had vague ideas and possible directions, this year we have several real challenges and design decisions we’d like to discuss:

• “Secret” memory userspace APIs

Should such API follow “native” MM interfaces like mmap(), mprotect(), madvise() or it would be better to use a file descriptor, e.g. like memfd-create does?

MM “native” APIs would require VM_something flag and probably a page flag or page_ext. With file-descriptor VM_SPECIAL and custom implementation of .mmap() and .fault() would suffice. On the other hand, mmap() and mprotect() seem better fit semantically and they could be more easily adopted by the userspace.

• Direct/linear map fragmentation

Whenever we want to drop some mappings from the direct map or even change the protection bits for some memory area, the gigantic and huge pages that comprise the direct map need to be broken and there’s no THP for the kernel page tables to collapse them back. Moreover, the existing API defined in <asm/setmemory.h> by several architectures don’t really presume it would be widely used.

For the “secret” memory use-case the fragmentation can be minimized by caching large pages, use them to satisfy smaller “secret” allocations and then collapse them back once the “secret” memory is freed. Another possibility is to pre-allocate physical memory at boot time.

Yet another idea is to make page allocator aware of the direct map layout.

• Kernel page table management

Currently we presume that only one kernel page table exists (well, mostly) and the page table abstraction is required only for the user page
tables. As such, we presume that `page table == struct mm_struct` and the
mm_struct is used all over by the operations that manage the page tables.

The management of the restricted address space in the kernel requires
ability to create, update and remove kernel contexts the same way we do
for the userspace.

One way is to overload the mm_struct, like EFI and text poking did. But
it is quite an overkill, because most of the mm_struct contains
information required to manage user mappings.

My suggestion is to introduce a first class abstraction for the page
table and then it could be used in the same way for user and kernel
context management. For now I have a very basic POC that slitted several
fields from the mm_struct into a new `struct pg_table` [5]. This new
abstraction can be used e.g. by PTI implementation of the page table
cloning and the KVM ASI work.

[0] https://lore.kernel.org/linux-mm/20200206165900.GD17499@linux.ibm.com/
[1] https://lore.kernel.org/lkml/20200504145810.11882-1-alexandre.chartre@oracle.com
[2] https://lore.kernel.org/lkml/20190612170834.14855-1-mhellenb@amazon.de/
[3] https://lore.kernel.org/lkml/20200130162340.GA14232@rapoport-lnx/
[5] https://git.kernel.org/pub/scm/linux/kernel/git/rppt/linux.git/log/?h=pg_table/v0.0

I agree to abide by the anti-harassment policy

I agree

Primary author:  RAPOPORT, Mike (IBM)
Presenter:  RAPOPORT, Mike (IBM)
Session Classification:  Kernel Summit
Track Classification:  Kernel Summit
Inline Encryption Support and new related features

Thursday, 27 August 2020 10:00 (45 minutes)

I gave a talk about file based encryption and the proposed inner workings of inline encryption at last year’s LPC. Since then, the patchset has gone through almost 10 revisions, and the block layer patches have been merged a little while ago into Linux v5.8 (and the remaining patches are being targeted for the v5.9 release). There have been many changes in the design and implementation over the past 10 revisions, some of which are likely worth going over.

An older version of the implementation has also been checked into Android for more than half a year now, and new changes and features have been proposed and implemented on top of the base inline encryption patchset, and are currently being maintained out of tree in Android like:

1. hardware wrapped key support
2. device mapper support
3. UFS crypto variant operations
4. eMMC inline encryption support
5. direct I/O support for fscrypt
6. metadata encryption.

These are all features we’d like to see upstreamed soon. I’d like to talk about and discuss some of these features and what we’d like to propose upstream for them.

I agree to abide by the anti-harassment policy

I agree

Primary author: TANGIRALA, Satya
Presenter: TANGIRALA, Satya
Session Classification: Kernel Summit
Track Classification: Kernel Summit
BoF: RCU Implementation

Wednesday, 26 August 2020 10:00 (45 minutes)

This is a gathering to discuss Linux-kernel RCU internals.

The exact topics depend on all of you, the attendees. In 2018, the focus was entirely on the interaction between RCU and the -rt tree. In 2019, the main gathering had me developing a trivial implementation of RCU on a whiteboard, coding-interview style, complete with immediate feedback on the inevitable bugs.

Come (virtually!) and see what is in store in 2020!

I agree to abide by the anti-harassment policy

I agree

Primary author: MCKENNEY, Paul (Facebook)
Presenter: MCKENNEY, Paul (Facebook)
Session Classification: BOFs Session
Track Classification: Birds of a Feather (BoF)
BoF: upstream drivers for open source FPGA SoC peripherals

Tuesday, 25 August 2020 07:00 (45 minutes)

There are active open source projects such as LiteX which have developed IP (e.g. chip-level hardware design) needed for building an open source SoC. The common workflow is that this SoC would be synthesized into a bitstream and loaded into a FPGA. (Aside: there is also the possibility of using these IP modules in an ASIC, but the scenario of supporting fixed-in-silicon hardware peripherals is already well-established in Linux).

The scenario of an open source SoC in a FPGA raises a question:

What is the best trade-off between complexity in the hardware peripheral IP and the software drivers?

Open source SoC design is done in a Hardware Description Language (HDL) with Verilog, VHDL, SystemVerilog or even newer languages (Chisel, SpinalHDL, Migen). This means we have the source and toolchain necessary to regenerate the design.

LiteX is a good example of an open source SoC framework where it provides IP for common peripherals like DRAM controller, Ethernet, PCIe, SATA, SD Card, Video and more. A key design decision for these peripherals are Control and Status Registers (CSR). The hardware design and the software drivers must agree on the structure of these CSRs.

The Linux kernel drivers for LiteX are currently being developed out-of-tree. A sub-project called Linux-on-LiteX-Vexriscv combines the Vexrisc core (32-bit RISC-V), LiteX modules, and a build system which results in a FPGA bitstream, kernel and rootfs.

There is an long-running effort led by Mateusz Holenko of Antmicro to land the LiteX drivers upstream starting with the LiteX SoC controller and LiteUART serial driver. Recently, support for Microwatt, a POWER-based core from IBM, was been added to LiteX and Benjamin Herrenschmidt has rekindled discussion of how best structure the LiteX CSRs and driver code for upstream. In addition, an experienced Linux graphics developer, Marin Perens, has jumped into the scene with a LiteDIP: "Plug-and-play LiteX-based IP blocks enabling the creation of generic Linux drivers. Design your FPGA-based SoC with them and get a (potentially upstream-able) driver for it instantly!"

Martin has blog posts that dives further into the issues I’ve tried to describe above: “FPGA: Why So Few Open Source Drivers for Open Hardware?” [7]

I think this BoF will be useful in accelerating the discussion that is happening on different mailing lists and hopefully bringing us closer to consensus.

[5] https://groups.google.com/d/msg/linux-litex/TJLLcsuBibY/3vP8_7nGAwAJ

I agree to abide by the anti-harassment policy
I agree

**Primary author:** Mr FUSTINI, Drew (BeagleBoard.org Foundation)

**Presenter:** Mr FUSTINI, Drew (BeagleBoard.org Foundation)

**Session Classification:** BOFs Session

**Track Classification:** Birds of a Feather (BoF)
kdevops: bringing devops to kernel development

Wednesday, 26 August 2020 09:10 (20 minutes)

Doing kernel development is fun, but setting up your throw away systems to do kernel development or testing is not so much fun, and it can be tedious and time consuming. For instance, setting up a full filesystems test lab can sometimes take weeks, at best.

kdevops was released with the motivation of reducing the amount of time and to avoid the complexity involved to set systems up from scratch for Linux kernel development and testing.

Throw away systems for kernel development can also vary. Some users may wish to use KVM, others may want to use OS X and Virtualbox. Some may want to use cloud environments, and the APIs for each of these vary. And what Linux distribution you use can also vary.

kdevops takes advantage of a few devops technologies which aims at making some of this abstract both local virtualization solutions and cloud environments. Solutions used include: vagrant, terraform and ansible.

I agree to abide by the anti-harassment policy

I agree

Primary author: CHAMBERLAIN, Luis (State Street)
Presenter: CHAMBERLAIN, Luis (State Street)
Session Classification: Testing and Fuzzing MC
Track Classification: Testing and Fuzzing MC
Currently RT developers maintain a series of RT releases based off various stable versions which add the RT patches on top. Once the RT patches have been merged into mainline the baseline stable releases will have the patches however we need to figure out how the testing will be handled, currently the stable maintainers rely on other people and organizations to do the bulk of their testing. We also need to ensure that the stable maintainers are OK with accepting fixes for RT specific issues like unbounded latencies, currently that seems OK with the stable rules as they are applied but it’s not clearly OK for the documented rules.

What are our plans here?

I agree to abide by the anti-harassment policy

I agree

**Primary author:** BROWN, Mark  
**Presenter:** BROWN, Mark  
**Session Classification:** Real-time MC  
**Track Classification:** Real-time MC
BoF: Show off your pets!

Tuesday, 25 August 2020 08:00 (45 minutes)

It’s not an evening social but pets are good. Stop by and show off your pets on video camera!

I agree to abide by the anti-harassment policy

I agree

Primary author:  ABBOTT, Laura
Presenter:  ABBOTT, Laura
Session Classification:  BOFs Session
Track Classification:  Birds of a Feather (BoF)
This introduces a working proof-of-concept alternative to RDMA, implementing a zero-copy DMA transfer between the NIC and GPU, while still performing the protocol processing on the host CPU. A normal NIC/host memory implementation is also presented.

By offloading most of the data transfer from the CPU, while not needing to reimplement the protocol stack, this should provide a balance between high performance and feature flexibility.

This presentation would cover the changes needed across the kernel; mm support, networking queues, skb handling, protocol delivery, and a proposed interface for zero-copy RX of data which is not directly accessible by the host CPU. It would also solicit input for further API design ideas in this area.

A paper is planned. This proposal was originally submitted for the main track and was recommended for the networking track instead.

I agree to abide by the anti-harassment policy

I agree

**Primary author:** LEMON, Jonathan (Facebook)

**Presenter:** LEMON, Jonathan (Facebook)

**Session Classification:** Networking and BPF Summit

**Track Classification:** Networking & BPF Summit
System Firmware and Device Firmware Updates using Unified Extensible Firmware Interface (UEFI) Capsules

Thursday, 27 August 2020 07:35 (20 minutes)

Firmware is responsible for low-level platform initialization, establishing root-of-trust, and loading the operating system (OS). Signed UEFI Capsules define an OS-agnostic process for verified firmware updates, utilizing the root-of-trust established by firmware. The open source FmpDevicePkg in TianoCore provides a simple method to update system firmware images and device firmware images using UEFI Capsules and the Firmware Management Protocol (FMP).

This session describes the EFI Development Kit II (EDK II) capsule implementation, implementing FMP using FmpDevicePkg, creating Signed UEFI Capsules using open source tools, and an update workflow based on the Linux Vendor Firmware Service (fwupd.org).

I agree to abide by the anti-harassment policy

I agree

Primary author:  HSIUNG, Harry (Intel)
Presenter:  HSIUNG, Harry (Intel)
Session Classification:  System Boot and Security MC
Track Classification:  System Boot and Security MC
futex2: A New Interface

Monday, 24 August 2020 07:30 (25 minutes)

After a renewed interest in futex from several groups who are trying to extend the interface (i.e. futex wait multiple, futex swap, variable-sized futexes), alongside failed attempts to solve long-standing issues that cannot be solved under the current interface, Thomas Gleixner is convinced a new implementation of futex is necessary. This topic will collect feedback on the work being done to design this new interface and discuss next steps to get this effort upstream.

I agree to abide by the anti-harassment policy

I agree

**Primary author:** ALMEIDA, André (Collabora)

**Presenter:** ALMEIDA, André (Collabora)

**Session Classification:** Real-time MC

**Track Classification:** Real-time MC
CRIU mounts migration: problems and solutions

Monday, 24 August 2020 07:25 (20 minutes)

OpenVZ and Virtuozzo containers use CRIU as the core technology for container migration in production. And Virtuozzo containers are slightly different thing to what most people would imagine containers today. They are “system containers” which is the one with full systemd inside, the one you would enter via ssh, the one which is an analogy to a virtual machine where the user gets root access inside and can do almost everything like on the hardware node with Linux.

This difference between application and system containers brings a lot of complex problems when it comes to container migration of the system containers. Lets consider the mounts problem. The user inside a container can explicitly or implicitly (by systemd, docker or some other means) create multiple different mount namespaces and mounts in them. And if we migrate the container, the user inside does not expect their mounts to change. So we need to checkpoint and restore them.

In this talk I would share main problems I’ve faced when I tried to improve the correctness of our current mount restore algorithm in CRIU and I would show new “mounts-v2” algorithm which tries to cover much more cases than the previous one. To achieve this we need at least one kernel patch and maybe more to come.

I would like to restart the discussion on bind mounts across namespaces at the point it had stopped a while ago. I hope we can reach a consensus about the kernel modifications required to solve the problem of checkpoint/restore of complex mounts. And I really hope for some useful advice on how to further improve the new algorithm.

[1] https://lore.kernel.org/lkm/1485214628-23812-1-git-send-email-avagin@openvz.org/

Here are links to mounts-v2 implementation in Virtuozzo criu:

I agree to abide by the anti-harassment policy

I agree

Primary author: TIKHOMIROV, Pavel
Presenter: TIKHOMIROV, Pavel
Session Classification: Containers and Checkpoint/Restore MC
Track Classification: Containers and Checkpoint/Restore MC
Opening session

I agree to abide by the anti-harassment policy

I agree

Primary authors: RAPOPORT, Mike (IBM); BERGMANN, Arnd (Linaro)

Session Classification: linux/arch/* MC

Track Classification: linux/arch/* MC
Fast checkpointing with criu-image-streamer

Monday, 24 August 2020 09:10 (20 minutes)

New cloud offerings such as Google preemtible VMs are up to 5x cheaper than regular machines. These VMs come with tight eviction deadlines (~30secs). This introduces a new goal: How can we evacuate an application from a machine as fast as possible?

Note that this problem is different from live migration, which aims at minimizing application downtime.

To do fast checkpointing, we developed criu-image-streamer. It enables streaming of images to and from CRIU during checkpoint/restore with low overhead.

The talk will cover the criu-image-streamer architecture, and shows the Linux mechanisms used to achieve checkpointing rates of 15GB/s and load-balance the checkpointed image output on an array of UNIX pipes.

The criu-image-streamer tool is open-source and can be found at https://github.com/Checkpoint-restore/criu-image-streamer

I agree to abide by the anti-harassment policy

I agree

Primary author: VIENNOT, Nicolas (Two Sigma)
Presenter: VIENNOT, Nicolas (Two Sigma)
Session Classification: Containers and Checkpoint/Restore MC
Track Classification: Containers and Checkpoint/Restore MC
FastFreeze: Unprivileged checkpoint/restore for containerized applications

Monday, 24 August 2020 07:45 (15 minutes)

CRIU is not easy to use for the average user. What to do with the file system? How and where to store images?

We developed an easy-to-use checkpoint/restore tool that uses the CRIU engine. It provides the following features:
  * It does not require root access to operate. Only an empty container (e.g. kubernetes) is required
  * Provides time virtualization, critical when migrating (java) applications across different machines
  * Provides CPUID virtualization, essential when migrating applications across an heterogeneous cluster
  * Handles file system checkpoint/restore
  * Fast image upload/download from Google Storage or AWS S3
  * Image compression
  * Production metrics

The talk will do a overview of these different components, and present the current state of rootless CRIU.
I will be covering the introduction of a new kernel capability, CAP_CHECKPOINT_RESTORE, proposed by Adrian Reber.

The tool that I will be presenting will be open-sourced before the talk.

I agree to abide by the anti-harassment policy

I agree

Primary author: VIENNOT, Nicolas (Two Sigma)
Presenter: VIENNOT, Nicolas (Two Sigma)
Session Classification: Containers and Checkpoint/Restore MC
Track Classification: Containers and Checkpoint/Restore MC
Multiple XDP programs on a single interface - status and next steps

Wednesday, 26 August 2020 07:00 (45 minutes)

At last year’s LPC I presented a proposal for how to attach multiple XDP programs to a single interface and have them run in sequence. In this presentation I will follow up on that, and present the current status and next steps on this feature.

Briefly, the solution we ended up with was a bit different from what I envisioned at the last LPC: We now rely on the new ‘freplace’ functionality in BPF which allows a BPF program to replace a function in another BPF program. This makes it possible to implement the dispatcher logic in BPF, which is now part of the ‘libxdp’ library in the xdp-tools package.

In this presentation I will explain how this works under the covers, what it takes for an application to support this mode of operation, and discuss how we can ensure compatibility between applications, whether or not they use libxdp itself. I am also hoping to solicit feedback on the solution in general, including any possible deficiencies or possible improvements.

I agree to abide by the anti-harassment policy

I agree

Primary author:  HØILAND-JØRGENSEN, Toke (Red Hat)
Presenter:  HØILAND-JØRGENSEN, Toke (Red Hat)
Session Classification:  Networking and BPF Summit
Track Classification:  Networking & BPF Summit
NUMA topology limitations

Tuesday, 25 August 2020 08:30 (30 minutes)

Recent experiments on more "creative" hardware have shown that the NUMA topology code has some unwritten assumptions which can be broken relatively easily. While the pictured topology may be considered questionable, somewhat saner topologies can trigger the same class of issues, which can be tested via e.g. QEMU.

The idea would be to point out said limitations, discuss if / how much we really care and potential ways forward.

Note: I plan to have an RFC on the list highlighting the issues in the above link, along with simplified QEMU reproducers, in a few weeks' time.

I agree to abide by the anti-harassment policy

I agree

Primary author: SCHNEIDER, Valentin (Arm Ltd)
Presenter: SCHNEIDER, Valentin (Arm Ltd)
Session Classification: Scheduler MC
Track Classification: Scheduler MC
Checkpoint-restoring containers with Docker inside

Monday, 24 August 2020 08:40 (20 minutes)

CRIU is the most advanced Checkpoint-Restore project on Linux.

But even with CRIU at the moment it is not feasible to checkpoint - restore all possible topologies of processes and namespaces. Even relatively simple case of a process tree with two UTS/IPC namespaces is not supported by CRIU, not mentioning more complex cases like a process tree with more than one PID namespaces.

In OpenVZ and Virtuozzo versions of CRIU these problems were partially solved with introduction of the support for nested PID namespaces, several IPC/UTS namespaces (with respect to USER namespaces) and overlayfs mounts.

These improvements allow us to get basic support of checkpoint-restoring OpenVZ system containers with Docker containers inside.

We have already prepared several upstream kernel patches [4].

I agree to abide by the anti-harassment policy

I agree

Primary authors: MIKHALITSYN, Alexander (Virtuozzo); TIKHOMIROV, Pavel

Presenters: MIKHALITSYN, Alexander (Virtuozzo); TIKHOMIROV, Pavel

Session Classification: Containers and Checkpoint/Restore MC

Track Classification: Containers and Checkpoint/Restore MC
openat2 landed in Linux 5.6, but unfortunately (though it does make it easier to implement safer container runtimes) there are still quite a few remaining tricks that attackers can use to attack container runtimes. This talk will give a quick overview of the remaining issues, some proposals for how we might fix them, and how libpaths will make use of them. In addition, a brief update on libpaths will be given.

Examples of attacks include:

- Fake /proc mounts.
- Bind-mounting on top of magic-links (such as /proc/$pid/attr/exec).

I agree to abide by the anti-harassment policy

I agree
Looking forward on proxy execution

Tuesday, 25 August 2020 10:10 (30 minutes)

I mentioned at last OSPM [1] how proxy execution could improve scheduling on big.LITTLE systems, but that obviously cannot happen until the bases of proxy execution work properly.

I’ve been given the green light to spend some time on proxy execution, so this would be an opportunity for me to present the current state of things (some grey areas here as I’m still investigating bugs right now), and discuss some points that need to be addressed to make forward progress.

I agree to abide by the anti-harassment policy

I agree

Primary author: SCHNEIDER, Valentin (Arm Ltd)
Presenter: SCHNEIDER, Valentin (Arm Ltd)
Session Classification: Scheduler MC
Track Classification: Scheduler MC
Virtualization is coming to automotive and helping advance the industry. Google is working on a reference VM platform for Android Automotive OS based on virtio and open standards.

Our work in this space builds on the cuttlefish virtual platform and adds support for new devices, including audio, sensors and vehicle bus access.

The session will focus on our design goals and choices, how we extended cuttlefish into ‘trout’ for auto, and the team’s vision going forward.

I agree to abide by the anti-harassment policy

I agree

Primary authors: GRANATA, Enrico (Google LLC); DELVA, Alistair (Google)

Presenters: GRANATA, Enrico (Google LLC); DELVA, Alistair (Google)

Session Classification: Android MC

Track Classification: Android MC
Thread Control Block (TCB) is a data structure in the Linux kernel which contains thread-specific information needed to manage it. The Thread Control Block acts as a library of information about the threads in the system. TCB is being manipulated by the kernel constantly, while the thread is being executed and while it is switched out. Assuring the integrity of the TCB is critical to achieve safe thread life cycle management in Linux.

As part of making TCB management safe, several tasks will need to be performed:

1. Analysis of the TCB • What kind of information is stored in TCB. For example: o All flags (unless there is a specific justification for an exception). o Namespace Cgroups information o Signal handlers o MMU list o Security fields o Dependencies on LSMs (e.g., in_execve or brk_randomized) o stack_canary o seccomp related data o stack pointer o parent pointer o child/sibling lists o PI data structures o RT mutexes o futex list o NUMA balancing fields o tlbflush_unmap_batch data

• What is the criticality of this information to the thread execution (Categorization to critical/non critical, etc.). For example:
  o Parent pointer
  o Signal handlers
• Identify the safety critical part(s) of the TCB

1. Analysis of the possible failure modes • What possible faults might be caused by the kernel, that will influence the TCB. For example:
  o Altering of data during context switch out
  o Corruption of data while thread is not running (e.g. due to bit flip)

2. Propose solutions for protecting the TCB – Examples:
• Kernel configurations on kernel space code – Protect the kernel space code and data by using kernel self protection mechanisms (e.g., enable CONFIG_HARDENED_USERCOPY, or disable CONFIG_DEVKMEM)
  • CRC the safety critical data after switch out
  • Allocate RO block and store immutable safety critical data in that block

I agree to abide by the anti-harassment policy

I agree

Primary authors:  Dr COPPERMAN, Elana (Mobileye);  Mr DAVIDOVICH, Rafi (Mobileye)

Presenters:  Dr COPPERMAN, Elana (Mobileye);  Mr DAVIDOVICH, Rafi (Mobileye)

Session Classification:  Kernel Dependability & Assurance MC

Track Classification:  Kernel Dependability & Assurance MC
Unifying Test Reporting with KernelCI

Wednesday, 26 August 2020 10:00 (30 minutes)

A year ago, the Linux Foundation KernelCI project embarked on a new effort: unifying reporting from all upstream kernel testing systems.

Our aim is to develop a new generic interface that can be used by any test system to submit results into a common database. This allows sending a single report email for each kernel revision being tested, backed by a single web dashboard collating the results, no matter how many or which systems contributed.

In the same way that the Linux kernel has a great number of contributors and is being used in a great number of ways, the long-term goal of KernelCI is to match that scale with an open testing philosophy.

We’ve been developing a report schema, a submission protocol, and a prototype implementation, focusing on making it easy to both start submitting results, and to accommodate requirements from new participants.

Come and see what we’ve achieved so far, what the schema is like, how you can start reporting, subscribe to results, and play a part in further development.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  KONDRASHOV, Nikolai (Red Hat);  TUCKER, Guillaume (Collabora)

Presenters:  KONDRASHOV, Nikolai (Red Hat);  TUCKER, Guillaume (Collabora)

Session Classification:  Testing and Fuzzing MC

Track Classification:  Testing and Fuzzing MC
scheduler fairness

Tuesday, 25 August 2020 08:00 (30 minutes)

scheduler fails to provide the same runtime to tasks when the system can’t be balanced like 9 running tasks on 8 CPUs. This talk will come back on the different proposal made during OSPM and discuss the way to move forward

I agree to abide by the anti-harassment policy

I agree

Primary author:  GUITTOT, Vincent (Linaro)
Presenter:  GUITTOT, Vincent (Linaro)
Session Classification:  Scheduler MC
Track Classification:  Scheduler MC
Kprobes Jump Optimized for more Archs

Tuesday, 25 August 2020 07:25 (25 minutes)

Since "Kprobes jump optimization" was introduced by Masami Hiramatsu in 2009, Only x86, arm32, powerpc64 have supported it. It seems that architecture met obstacles to implement the feature.

In this talk, let’s compare x86, arm32, powerpc64 OPTKPROBES’ feature, and find out the limitation of them. Then let’s talk about how to implement kprobes jump Optimized for new archs (riscv & csky).

In the end, the talk will give out some advice to ISA hardware design to help implementing the feature of kprobes jump optimized.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr GUO, Ren
Presenter: Mr GUO, Ren
Session Classification: linux/arch/*/ MC
Track Classification: linux/arch/*/ MC
Standards for device-side test artifacts

Wednesday, 26 August 2020 07:45 (20 minutes)

This session will involve a discussion around a proposal for standards for device-side test artifacts. Currently there are no standards (that the author is aware of) for where tests should be placed in a device under test, or how test frameworks should discover, interact with, and collect results from test artifacts.

Tim will propose adding some new directories to the FileSystem Hierarchy Standard to specify that:

* test code and data should go under /usr/test
* a test wrapper function, called “{testname}-run” should be placed in /usr/test/bin
* test output should be placed in /var/test (or maybe /var/log/test)
* with name “{testname}-output-[datestamp].[appropriate-extension]”
* a user account called “test” should be created, with well known pid 88
* a group called “test” should be created, with well known gid 88
* the directories and files above should be owned by ’test:’

This would allow end users and automated tools to find and easily execute any tests that are packaged with a system. It also designates a place in the filesystem where tests can be placed. Having separate locations for test artifacts allows for different mounting or storage decisions for those locations in the filesystem. This could be beneficial since tests might not be part of production releases, or test artifacts might only be applied to a device temporarily.

This is intended to be a discussion among automated testing and distribution developers, to see if this is something useful going forward, and to plan next steps.

I agree to abide by the anti-harassment policy

I agree

Primary author:  BIRD, Tim (Sony)
Presenter:  BIRD, Tim (Sony)
Session Classification:  Testing and Fuzzing MC
Track Classification:  Testing and Fuzzing MC
RISC-V Linux Tracing (K/Uprobe)

Wednesday, 26 August 2020 08:00 (30 minutes)

Linux Tracing contains a board list of kernel features (ftrace, perf, bpf, k/uprobe) and it will be the bottleneck of the user debugging experience without them. So tracing micro-conference was held in the 2018 & 2019 Linux plumber conference and it’s also a hot-pot topic of Linux today. But as a newborn architecture, what’s the status of RISC-V Linux tracing? Ready to use?

Many new features of RISC-V Linux have been developed recently and some are related to tracing. e.g: k/uprobe is the basic infrastructure of Linux dynamic tracing that other architectures have implemented, and RISC-V Linux k/uprobe’s patchset has been proposed since November 2018 (more than 1 year past way). The work blocked the many Linux tools (such as: systemtap, trace-cmd, perf probe, ...)

Now, k/uprobe has finally been completed with several developers’ effort, and we’ll give DEMOs of “trace-cmd & perf probe ...” in the talk to enhance people’s confidence in RISC-V Linux debugging.

In the end, let’s talk about how to improve k/uprobe from the ISA view:
The single-step trap exception is an ancient technology that has been supported by many CPU architectures, but RISC-V ISA does not support this feature. It seems that the designers of RISC-V feel that the single-step exception feature can be completely replaced by inserting a breakpoint instruction. Is this true? Here, Introduce a new improved hw mechanism to solve the shortcomings of the traditional single-step exception for Linux tracing (k/uprobe) arch implementation.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr REN, Guo
Presenter: Mr REN, Guo
Session Classification: RISC-V MC
Track Classification: RISC-V MC
The way to d_path helper

Tuesday, 25 August 2020 07:00 (45 minutes)

The d_path is eBPF tracing helper, that returns string with full path for given ‘struct path’ object and was requested long time ago by many people.

Along the way of implementing it, other features had to be added to the verifier:

- compile time BTF IDs resolving
  This allows using of kernel objects BTF IDs without resolving them in runtime and saves few cycles on resolving during kernel startup and introducing single interface for accessing such IDs

- allow to pass BTF ID + offset as helper argument
  This allows to pass an argument to helper, which is defined via parent BTF object + offset, like for bpf_d_path (added in following changes):

```c
SEC("fentry/filp_close")
int BPF_PROG(prog_close, struct file file, void *id)
{
...
ret = bpf_d_path(file->f_path, ...
```

In this talk I’ll show implementation details of d_path helper and details of both aforementioned features and why they are important for d_path helper.

I agree to abide by the anti-harassment policy

I agree

**Primary author:** OLSA, Jiri

**Presenter:** OLSA, Jiri

**Session Classification:** Networking and BPF Summit

**Track Classification:** Networking & BPF Summit
ASI: Efficiently Mitigating Speculative Execution Attacks with Address Space Isolation

Thursday, 27 August 2020 08:10 (20 minutes)

Speculative execution attacks, such as L1TF, MDS, LVI pose significant security risk to hypervisors and VMs. A complete mitigation for these attacks requires very frequent flushing of buffers (e.g., L1D cache) and halting of sibling cores. The performance cost of such mitigations is unacceptable in realistic scenarios. We are developing a high-performance security-enhancing mechanism to defeat speculative attack which we dub Address Space Isolation (ASI). In essence, ASI is an alternative way to manage virtual memory for hypervisors, providing very strong security guarantees at a minimal performance cost. In the talk, we will discuss the motivation for this technique as well as initial results we have.

I agree to abide by the anti-harassment policy

I agree

Primary author: Dr WEISSE, Ofir (Google)
Presenter: Dr WEISSE, Ofir (Google)
Session Classification: System Boot and Security MC
Track Classification: System Boot and Security MC
Speculative execution attacks, such as L1TF, MDS, LVI pose significant security risk to hypervisors and VMs. A complete mitigation for these attacks requires very frequent flushing of buffers (e.g., L1D cache) and halting of sibling cores. The performance cost of such mitigations is unacceptable in realistic scenarios. We are developing a high-performance security-enhancing mechanism to defeat speculative attack which we dub Address Space Isolation (ASI). In essence, ASI is an alternative way to manage virtual memory for hypervisors, providing very strong security guarantees at a minimal performance cost. In the talk, we will discuss the motivation for this technique as well as initial results we have.

I agree to abide by the anti-harassment policy

I agree

**Primary author:**  WEISSE, Ofir (Google)

**Presenter:**  WEISSE, Ofir (Google)

**Session Classification:**  BOFs Session

**Track Classification:**  Birds of a Feather (BoF)
Per Thread Queues (PTQ)

Wednesday, 26 August 2020 07:45 (45 minutes)

In this talk we introduce Per Thread Queues (PTQ). PTQ is a type of network packet steering that allows application threads to be assigned dedicated network queues for both transmit and receive. This facility provides highly granular traffic isolation between applications and can also help facilitate high performance when combined with other techniques such as busy polling. PTQ extends both XPS and aRFS.

A key concept of PTQ is “global queues”. These are a device independent, abstract representation of network queues. Global queues are as their name implies, they can be treated as managed resource across not only a system, but also across the a data center similar to how other resources are managed across a datacenter (memory, CPU, network priority, etc.). User defined semantics and QoS characteristics can be added to global queues. For instance, queue #233 in the datacenter might refer to a queue with QoS properties specific to handling video. Ultimately in the data path, a global queue is resolved to a real device queue that provides the semantics and QoS associated with the global queue. This resolution happens per a device specific mapping functions that maps a global queue to a device queue.

Threads may be assigned a global queue for both transmit and receive. The assignment comes from pools of transmit and receive queues configured in a cgroup. When a thread starts in a cgroup, the queue pools of the cgroup are consulted. If a queue pool is configured, the kernel assigns a queue to the thread (either a TX queue, RX queue, or both). The assigned queues are stored in the threads task structure. To transmit, the mapped device queue for the assigned transmit queue is used in lieu of XPS queue selection; for receive, the mapped device queue for the assigned receive queue is programmed into the device via ndo_rx_flow_steer.

This talk will cover the design, implementation, and configuration of PTQ. Additionally, we will present performance numbers and discuss some of the many ways that this work can be further enhanced.

I agree to abide by the anti-harassment policy

I agree

Primary author: HERBERT, Tom
Presenter: HERBERT, Tom
Session Classification: Networking and BPF Summit
Track Classification: Networking & BPF Summit
Kubernetes service load-balancing at scale with BPF & XDP

Thursday, 27 August 2020 07:00 (45 minutes)

With the incredible pace of containerisation in enterprises, the combination of Linux and Kubernetes as an orchestration base layer is often considered as the "cloud OS". In this talk we provide a deep dive on Kubernetes's service abstraction and related to it the path of getting external network traffic into one’s cluster.

With this understanding in mind, we then discuss issues and shortcomings of the existing kube-proxy implementation in Kubernetes for larger scale and high churn environments and how it can be replaced entirely with the help of Cilium by utilising BPF and XDP. Cilium’s service load-balancing architecture consists of two main components, that is, BPF at the socket layer for handling East-West traffic and BPF at the driver layer for processing the North-South traffic path.

Given XDP has only recently been added to Cilium in order to accelerate service load-balancing, we’ll discuss our path towards implementing the latter, lessons learned, provide a detailed performance analysis compared to kube-proxy in terms of forwarding cost as well as CPU consumption, and future extensions on kernel side.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  BORKMANN, Daniel (Cilium.io); PUMPUTIS, Martynas (Cilium)

Presenters:  BORKMANN, Daniel (Cilium.io); PUMPUTIS, Martynas (Cilium)

Session Classification:  Networking and BPF Summit

Track Classification:  Networking & BPF Summit
Planning code obsolescence

Tuesday, 25 August 2020 07:00 (25 minutes)

The majority of the code in the kernel deals with hardware that was made a long time ago, and we are regularly discussing which of those bits are still needed. In some cases (e.g. 20+ year old RISC workstation support), there are hobbyists that take care of maintainership despite there being no commercial interest. In other cases (e.g. x.25 networking) it turned out that there are very long-lived products that are actively supported on new kernels.

When I removed support for eight instruction set architectures in 2018, those were the ones that no longer had any users of mainline kernels, and removing them allowed later cleanup of cross-architecture code that would have been much harder before.

I propose adding a Documentation file that keeps track of any notable kernel feature that could be classified as "obsolete", and listing e.g. following properties:

- Kconfig symbol controlling the feature
- How long we expect to keep it as a minimum
- Known use cases, or other reasons this needs to stay
- Latest kernel in which it was known to have worked
- Contact information for known users (mailing list, personal email)
- Other features that may depend on this
- Possible benefits of eventually removing it

With that information, my hope is that it becomes easier to plan when some code can be removed after the last users have stopped upgrading their kernels, while also preventing code from being removed that is actually still in active use.

In the discussion at the linux/arch/* MC, I would hope to answer these questions:

- Do other developers find this useful to have?
- Where should the information be kept (Documentation/*, Kconfig, MAINTAINERS, wiki.kernel.org, ...)
- Which information should be part of an entry?
- What granularity should this be applied to – only high-level features like CPU architectures and subsystems, or individual drivers and machines?

I agree to abide by the anti-harassment policy

I agree

Primary author: BERGMANN, Arnd (Linaro)

Presenter: BERGMANN, Arnd (Linaro)
Session Classification: linux/arch/* MC

Track Classification: linux/arch/* MC
4G/4G memory split on 32-bit architectures

Tuesday, 25 August 2020 09:25 (25 minutes)

On 32-bit Linux machines, the 4GB of virtual memory are usually split between 3GB address space for user processes and a little under 1GB directly mapped physical memory.

While kernels can address more physical memory than what is directly mapped, this requires the “highmem” feature that is likely going away in the long run, while there are still systems using 32-bit ARM Linux with 2GB or more that should get kernel updates for many years to come.

As an alternative to highmem, we are proposing a new way to split the available virtual memory, giving 3.75GB of address space to both user space and to the linear physical memory mapping.

In this presentation, we discuss the state of those patches and the trade-offs we found for performance, security and compatibility with existing systems.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  Mr WALLEIJ, Linus (Arm); BERGMANN, Arnd (Linaro); Mr MOHAMMED, Afzal

Presenters:  Mr WALLEIJ, Linus (Arm); BERGMANN, Arnd (Linaro)

Session Classification:  linux/arch/* MC

Track Classification:  linux/arch/* MC
Traceloop and BPF

Monday, 24 August 2020 07:00 (45 minutes)

We will present traceloop, a tool for tracing system calls in cgroups or in containers using in-kernel Berkeley Packet Filter (BPF) programs.

Many people use the “strace” tool to synchronously trace system calls using ptrace. Traceloop similarly traces system calls but with low overhead (no context switches) and asynchronously in the background, using BPF and tracing per cgroup. We will show how it is integrated with Kubernetes via Inspektor Gadget.

Traceloop’s traces are recorded in perf ring buffers (BPF_MAP_TYPE_PERF_EVENT_ARRAY) configured to be overwritable like a flight recorder. As opposed to “strace”, the tracing is permanently enabled on Kubernetes pods but rarely read, only on-demand, for example in case of a crash.

We will present both past limitations with their workarounds, and how new BPF features can improve traceloop. This includes:

- Lack of bpf_get_current_cgroup_id() on Linux < 4.18 and systems not using cgroup-v2. Workaround using the mount namespace id.
- New BPF programs can only be inserted in a PROG_ARRAY map from userspace, making synchronous updates more complicated.
- BPF ringbuffer to replace BPF perf ringbuffer to improve memory usage.

I agree to abide by the anti-harassment policy

I agree

Primary authors: CREQUY, Alban (Kinvolk); LÜKE, Kai (Kinvolk)

Presenters: CREQUY, Alban (Kinvolk); LÜKE, Kai (Kinvolk)

Session Classification: Networking and BPF Summit

Track Classification: Networking & BPF Summit
The original latency nice proposal, a per-task parameter that reduced wakeup latency by short circuiting idle core/cpu searches in the wakeup path, was made over a year ago. Upstream discussion ultimately identified multiple seemingly related proposals, "Per-task vruntime wakeup bonus", "Small background task packing" and "Skip energy aware task placement". The scheduler maintainers asked the authors of the above to explore the perceived commonality and whether a single per-task parameter (formerly known as latency nice) can adequately and sensically control the intended uses. A stated constraint is that concepts like “latency nice” must be consistent with the general understanding of “nice” to include range and the direction of niceness.

A framework for evaluation was created and the four proposals are currently under discussion on the mailing list.

The goal of this proposal is to have a discussion about the main use-cases identified so far and agree on which make sense and how to progress them.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  Mr BELLASI, Patrick; HYSER, chris; SHAH, Parth; EGGEMANN, Dietmar; WANG, Xi (Google)

Presenters:  Mr BELLASI, Patrick; HYSER, chris; SHAH, Parth; EGGEMANN, Dietmar; WANG, Xi (Google)

Session Classification:  Scheduler MC

Track Classification:  Scheduler MC
Evaluation of tail call costs in eBPF

Monday, 24 August 2020 09:00 (45 minutes)

We would like to present results of an estimation of tail calls costs between eBPF programs. This was carried out for two kernel versions, 5.4 and 5.5. The latter introduces an optimization to remove the retropine mitigating spectre flaws, in certain conditions. The numbers come from 2 benchmarks, executed over our eBPF software stack. The first one uses the in-kernel testing BPF_PROG_TEST_RUN. The second one uses kprobes, network namespaces and iperf3 to get figures from a production-like environment. The conditions to trigger the optimization from kernel 5.5 were met in both cases, resulting in a drop of the cost of one tail call from 20-30 ns to less than 10 ns.

More recent techniques to estimate CPU time cost of eBPF programs would be covered, as well as other improvements to the measurement system. At Cloudflare we have production deployment of eBPF programs with multiple tail calls. Thus, estimating and limiting the cost of these is important from a business perspective. As a result, examples of strategies used or considered to limit costs associated with tail calls would be outlined in the presentation too.

The desired outcome from the discussion is to get feedback on the methods deployed, both for benchmarks and to limit tail calls.

As this work is part of an internship for a master thesis, a paper would be written with the relevant elements of the thesis.

This would be a relatively short presentation, 20 minutes long, including questions.

I agree to abide by the anti-harassment policy

I agree

Primary author:  JOLY, Clément (Cloudflare)

Presenters:  JOLY, Clément (Cloudflare); SERMAN, François

Session Classification:  Networking and BPF Summit

Track Classification:  Networking & BPF Summit
Networking Androids

Android Networking - update for 2020:
- what are our pain points wrt. kernel & networking in general,
- progress on upstreaming Android Common Kernel networking code,
- and the unknown depths of non-common vendor changes,
- how we’re using bpf,
- how it’s working,
- what’s not working,
- how it’s better then writing kernel code,
- why it’s so much worse,
- etc...

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr ZENCZYKOWSKI, Maciej (Google, Inc.)
Presenter: Mr ZENCZYKOWSKI, Maciej (Google, Inc.)
Session Classification: Networking and BPF Summit
Track Classification: Networking & BPF Summit
Multidimensional fair-share rate limiting in BPF

Tuesday, 25 August 2020 09:00 (45 minutes)

As UDP does not have flood attack protections such as SYN cookies, we developed a novel fair-share rate limiter in unprivileged BPF, designed for a UDP reverse proxy, that is capable of applying rate limits to specific traffic streams while minimizing the impact on others. To achieve this, we base our work on Hierarchical Heavy Hitters, which proposes a method to group packets on source and destination IP address, and we are able to substantially simplify the algorithm for our rate-limiting use case in order to allow for an implementation in BPF.

We further extend the concept of a hierarchy from IPs addresses to ports, providing us with precise rate limits based on the 4-tuple.

Our approach is capable of rate limiting floods originating from single addresses, subnets but also reflection attacks, and applies limits as specific as possible. To verify it’s performance we evaluated the approach against different simulated scenarios.

The outcome of this project is a single library that can be activated on any UDP socket and provides a flood protection out of the box.

I agree to abide by the anti-harassment policy

I agree

Primary author: OTTEN, Jonas (Cloudflare)
Co-author: BAUER, Lorenz (Cloudflare)
Presenters: OTTEN, Jonas (Cloudflare); BAUER, Lorenz (Cloudflare)
Session Classification: Networking and BPF Summit
Track Classification: Networking & BPF Summit
Linux has a new ‘lockdown’ security mode where changes to the running kernel requires verification with a cryptographic signature and restrictions to accesses to kernel memory that may leak to userspace.

Lockdown’s ‘integrity’ mode requires just the signature, while in ‘confidentiality’ mode in addition to requiring a signature the system can’t leak confidential information to userspace.

Work needs to be done to add cryptographic signatures for eBPF bytecode. The signature be then passed to the kernel via sys_bpf() reusing the kernel module signing infrastructure.

The main eBPF loader, libbpf, may perform relocations on the received bytecode for things like CO-RE (Compile Once, Run Everywhere), thus tampering with the signature made with the original bytecode.

It is thus needed to move such modifications to the signed bytecode from libbpf to the kernel, so that it may be done after the signature is verified.

This presentation is intended to provide a problem statement, some ideas being discussed, provide a reading list, and to foster awareness about this security feature so that BPF can be used in environments where ‘lockdown’ mode is required.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr MELO, Arnaldo (Red Hat Inc.)
Presenter:  Mr MELO, Arnaldo (Red Hat Inc.)
Session Classification:  Networking and BPF Summit
Track Classification:  Networking & BPF Summit
Two significant parts of interaction between architectures and the generic MM are memory model (flat, discontigmem, sparsemem) and memory detection and initialization.

SPARSEMEM was designed as replacement for DISCONTIGMEM, but although sparse memory model was stable and robust for long time, there are still several architectures that require DISCONTIGMEM and the conversion is not as trivial as one might think. I’d like to discuss the trade-offs and challenges involved in this transition in a hope to remove the complexity associated with maintenance of both models.

While the necessity to support extra memory model translates into code complexity and maintenance burden, the lack of consistency in memory detection and initialization among the architecture may cause exposure of run time bugs. Moreover, absence of a generic abstraction for physical memory layout makes every architecture to reinvent the wheel and, for example, we have e820 with numa_meminfo on x86, memblock on ARM/ARM64 and memblock with device tree in PowerPC. I believe that reaching a consensus about a generic data structure that will describe physical memory layout, including bank extents, NUMA nodes span, availability of mirroring and hotplug, would be beneficial to all architectures.

I agree to abide by the anti-harassment policy

I agree

Primary author:  RAPOPORT, Mike (IBM)
Presenter:  RAPOPORT, Mike (IBM)
Session Classification:  linux/arch/* MC
Track Classification:  linux/arch/* MC
Continuous Integration for mainline Real-Time Linux

Monday, 24 August 2020 09:00 (25 minutes)

Soon, the Real-Time Linux project will have its PREEMPT_RT patches in mainline Linux. One part of the Real-Time Linux collaboration project is its continuous integration system CI-RT (https://github.com/ci-rt) with one known lab running (https://ci-rt.linutronix.de).

In this talk, a possible way how to run the existing CI-RT tests on mainline Linux will be presented. Additionally, possible real-time test introduction for other, wider-spread test frameworks like Kernel CI will be discussed so that real-time regressions can be found as soon as possible and the awareness of Linux’s real-time capabilities and their implications for development is raised amongst kernel hackers. Also, this aims at testing with a larger hardware variety in other labs.

The audience is invited to participate in a moderated discussion on the talk’s topic and is encouraged to bring up any additional ideas on it.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr GERMANN, Bastian
Presenter: Mr GERMANN, Bastian
Session Classification: Real-time MC
Track Classification: Real-time MC
A programmable Qdisc with eBPF

Wednesday, 26 August 2020 09:00 (45 minutes)

Today we have a few dozens of Qdisc’s available in Linux kernel, offering various algorithms to schedule network packets. You can change the parameters of each Qdisc, but you cannot change the core algorithm of a given Qdisc. A programmable Qdisc offers a way to customize your own scheduling algorithms without writing a Qdisc kernel module from scratch. With eBPF emerges across the Linux network stack, it is time to explore how to integrate eBPF with Qdisc’s.

Unlike the existing eBPF TC filter and action, a programmable Qdisc is much more complicated, because we have to think about how to store skb’s and what we can offer for users to program. More importantly, a hierarchical Qdisc is even harder while could offer more flexibility.

We will examine the latest eBPF functionalities and packet scheduler architecture, discuss those challenges with possible solutions for a programmable Qdisc with eBPF.

I agree to abide by the anti-harassment policy

I agree

Primary author: WANG, Cong
Presenter: WANG, Cong
Session Classification: Networking and BPF Summit
Track Classification: Networking & BPF Summit
The BPF LSM or Kernel Runtime Security Instrumentation (KRSI) aims to provide an extensible LSM by allowing privileged users to attach eBPF programs to security hooks to dynamically implement MAC and Audit Policies.

KRSI was introduced in LSS-US 2019 and has since then had multiple interesting updates and triggered some meaningful discussions. The talk provides an update on:

- Progress in the mainline kernel, the ongoing discussions, and a recap of the interesting discussions that were resolved.
- New infrastructure merged into BPF to support the BPF LSM use-case.
- Some optimisations that can improve the performance characteristics of the currently existing LSM framework which would not only benefit KRSI but also all other LSMs.

The talk showcases how the design has evolved over time and what trade-offs were considered and what’s upcoming after the initial patches are merged.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr SINGH, KP
Presenter: Mr SINGH, KP
Session Classification: Networking and BPF Summit
Track Classification: Networking & BPF Summit
syzkaller/sanitizers status update

Wednesday, 26 August 2020 07:15 (30 minutes)

syzkaller is an open-source coverage-guided OS kernel fuzzer used to continuously test the Linux kernel. To date syzkaller has found 3000+ bugs in the upstream kernel. The kernel sanitizers are a family of dynamic bug finding tools (KASAN, KMSAN, KCSAN) that detect various types of bugs in the kernel.

In this talk Dmitry will give an overview of new developments in the past year for syzkaller and sanitizers and share some stats for kernel bugs and syzkaller contributions. Then Dmitry will outline the testing process of the syzkaller itself and some nice features that the kernel testing process could borrow. The talk concludes with future work for syzkaller/sanitizers.

I agree to abide by the anti-harassment policy

I agree

Primary author:  VYUKOV, Dmitry (Google)
Presenter:  VYUKOV, Dmitry (Google)
Session Classification:  Testing and Fuzzing MC
Track Classification:  Testing and Fuzzing MC
Right-sizing is hard, resizable BPF maps for optimum map size

Thursday, 27 August 2020 09:00 (45 minutes)

Right-sizing BPF maps is hard. By allocating for a worse case scenario we build large maps consuming large chunks of memory for a corner case that may never occur. Alternatively, we may try to allocate for the normal case choosing to ignore or fail in the corner cases. But, for programs running across many different workloads and system parameters its difficult to even decide what a normal case looks like. For a few maps we may consider using the BPF_F_NO_PREALLOC flag, but here we are penalized at allocation time and still need to charge our memory limits to match our max memory usage.

For a concrete example, consider a sockhash map. This map allows users to insert sockets into a map to build load balancers, socket hashing, policy, etc. but, how do we know how many sockets will exist in a system. What do we do when we overrun the table?

In this talk we propose a notion of resizable maps. The kernel already supports resizable arrays and resizable hash tables giving us a solid grounding to extend the underlying data structures of similar maps in BPF. Additionally, we also have the advantage of allowing the BPF programmer to tell us when to grow these maps to avoid hard-coded heuristics.

We will provide two concrete examples where the above has proven useful. First, using the sockmap and sockhash tables noted above. This way we can issue a bpf_grow_map() indicating to the BPF map code more slots should be allocated if possible. We can decide using BPF program logic where to put this low-water mark. Finally, we will also illustrate how using resizable arrays can ensure the system doesn’t run out of slots for the associated data in an example program. This has become a particularly difficult problem to solve with the current implementations where worse case can be severe, requiring 10x or more entries than the normal case. With the addition of resizable maps we expect many of the issues with right-sizing can be eliminated.

I agree to abide by the anti-harassment policy

I agree

Primary author: FASTABEND, John (Isovalent)
Presenter: FASTABEND, John (Isovalent)
Session Classification: Networking and BPF Summit
Track Classification: Networking & BPF Summit
How we built Magic Transit

Thursday, 27 August 2020 09:45 (45 minutes)

In this talk we will present Magic Transit, Cloudflare’s layer 3 DDoS protection service, as a case study in building a network product from the standard Linux networking stack. Linux provided us with flexibility and isolation that allowed us to stand up this product and on-board more than fifty customers within a year of conceptualization. Cloudflare runs all of our services on every server on our edge, and Magic Transit is not an exception to that rule - one of our biggest design challenges was working a layer 3 product into a networking environment tuned for proxy and server products. We’ll cover how we built Magic Transit, what worked really well, and what challenges we encountered along the way.

Magic Transit is largely implemented as a “configurator”, that is our software manages the network setup, and lets the kernel do the heavy lifting with network namespaces, policy routing and netfilter to safely direct and scrub IP traffic for our customers. This design allows drop-in integration with our DDoS protection systems, and our proxying and L7 products, and in a way that our operations team was familiar with. These benefits do not come without their caveats; specifically route placement/reporting inconsistencies, quirks revolving around icmp packets being generated from within a namespace when fragmentation occurs, problems stemming from conntrack and a mystery around offload... Finally we’ll touch on our future plans to migrate our web of namespaces to a Rust service that makes use of ebpf/xdp.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  Mr HEINE, Erich (Cloudflare);  Mr JONES, Connor (Cloudflare)

Presenters:  Mr HEINE, Erich (Cloudflare);  Mr JONES, Connor (Cloudflare)

Session Classification:  Networking and BPF Summit

Track Classification:  Networking & BPF Summit
Packet mark in the Cloud Native world

Monday, 24 August 2020 07:45 (45 minutes)

The 32-bit “mark” associated with the skb has served as a metadata exchange format for Linux networking subsystems since the beginning of the century. Over that time, the interpretation and reuse of the field has grown to encapsulate a wide range of networking use cases, expanding to touch everything from iptables, tc, xfrm, openvswitch, sockets, routing, to eBPF. In recent years, more than a dozen network control applications have been written in the Cloud Native space alone, many of which are using the packet mark in different ways to solve networking problems. The kernel facilities define no specific semantics to these bits, which leaves it up to these applications to co-ordinate to avoid incompatible mark usage.

This talk will explore use cases for sharing metadata between Linux subsystems in light of recent containerization trends, including but not limited to: application identity, firewalling, ip masquerade, network isolation, service proxying and transparent encryption. Beyond that, Cilium’s particular usage will be discussed with approaches used to mitigate conflicts due to the inevitable overload of the mark.

I agree to abide by the anti-harassment policy

I agree

Primary author: STRINGER, Joe (Cilium.io)
Presenter: STRINGER, Joe (Cilium.io)
Session Classification: Networking and BPF Summit
Track Classification: Networking & BPF Summit
xen-netfront and virtio_net XDP offloading

Monday, 24 August 2020 09:45 (45 minutes)

In the proposed talk I would like to discuss the opportunity to create a core for XDP program offloading from a guest to a host. The main goal here is to increase packet processing speed.

There was an attempt to merge offloading for virtio-net but the work is in progress.
After addition XDP processing to the xen-netfront driver the similar Xen task has to be solved as well.
vmxnet3 driver currently doesn’t support XDP processing but after adding it the same problem has to be solved there.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr KIRJANOV, Denis
Presenter:  Mr KIRJANOV, Denis
Session Classification:  Networking and BPF Summit
Track Classification:  Networking & BPF Summit
Eliminating bugs in BPF JITs using automated formal verification

Friday, 28 August 2020 07:00 (45 minutes)

This talk will present our ongoing efforts of using formal verification to eliminate bugs in BPF JITs in the Linux kernel. Formal verification rules out classes of bugs by mechanically proving that an implementation adheres to an abstract specification of its desired behavior.

We have used our automated verification framework, Serval, to find 30+ new bugs in JITs for the x86-32, x86-64, arm32, arm64, and riscv64 architectures. We have also used Serval to develop a new BPF JIT for riscv32, RISC-V compressed instruction support for riscv64, and new optimizations in existing JITs.

The talk will roughly consist of the following parts:

- A report of the bugs we have found and fixed via verification, and why they escaped selftests.
- A description of how the automated formal verification works, including a specification of JIT correctness and a proof strategy for automated verification.
- A discussion of future directions to make BPF JITs more amenable to formal verification.

The following links to a list of our patches in the kernel, as well as the code for the verification tool and a guide of how to run it:

https://github.com/uw-unsat/serval-bpf

I agree to abide by the anti-harassment policy

I agree

Primary author: NELSON, Luke (University of Washington)

Presenter: NELSON, Luke (University of Washington)

Session Classification: Networking and BPF Summit

Track Classification: Networking & BPF Summit
Userspace OVS with HW Offload and AF_XDP

Friday, 28 August 2020 09:00 (45 minutes)

OVS has two major datapaths: 1) the Linux kernel datapath, which shipped with Linux distributions and 2) the userspace datapath, which usually coupled with DPDK library as packet I/O interface, and called OVS-DPDK. Recent OVS also supports two offload mechanisms: the TC-flower for the kernel datapath, and the DPDK rte_flow for the userspace datapath. The tc-flower API with kernel datapath seems to be more feature-rich, with the support for connection tracking. However, the userspace datapath is in general faster than the kernel datapath, due to more packet processing optimizations.

With the introduction of AF_XDP to OVS, the userspace datapath can process packets at high rate without requiring DPDK library. AF_XDP socket creates a fast packet channel to the OVS userspace datapath and shows similar performance compared to using DPDK. In this case, the AF_XDP socket with OVS userspace datapath enables a couple of new ideas. First, unlike OVS-DPDK, with AF_XDP, the userspace datapath can enable TC-flower offload, because the device driver is still running in the kernel. Second, when considering flows which can’t be offloaded to the hardware, ex: L7 processing, these flows can be redirected to OVS userspace datapath using AF_XDP socket, which is faster than processing in kernel. And finally, users can implement new features using a custom XDP program attached to the device, when flows can’t be offloaded due to lack of hardware support.

In summary, with this architecture, we hope that a flow can be processed in the following sequences:
1) In hardware with tc-flower API. This shows best performance with the latest hardware. And if not capable,
2) In XDP. This shows second to the hardware performance, with the flexibility for new features and with eBPF verifier’s safety guarantee. And if not capable,
3) In OVS userspace datapath. This shows the best software switching performance.

Moving forward, we hope to unify the two extreme deployment scenarios; the high performance NFV cases using OVS-DPDK, and the enterprise hypervisor use cases using OVS kernel module, by just using the OVS userspace datapath with AF_XDP. Currently we are exploring the feasibility of this design and limitations. We hope that by presenting this idea, we can get feedback from the community.

I agree to abide by the anti-harassment policy

I agree

Primary author:  TU, William (VMware)
Presenter:  TU, William (VMware)
Session Classification:  Networking and BPF Summit
Track Classification:  Networking & BPF Summit
BPF extensible network: TCP header option, CC, and socket local storage

Friday, 28 August 2020 07:45 (45 minutes)

This talk will discuss some recent works that extend the TCP stack with BPF: TCP header option, TCP Congestion Control (CC), and socket local storage.

Hopefully the talk can end with getting ideas/desires on which part of the stack can practically be realized in BPF.

I agree to abide by the anti-harassment policy

I agree

Primary author: LAU, MARTIN
Presenter: LAU, MARTIN
Session Classification: Networking and BPF Summit

Track Classification: Networking & BPF Summit
A Realtime Tour Through BPF

Monday, 24 August 2020 07:05 (25 minutes)

Injecting large quantities of preempt-disabled code pretty much anywhere in a realtime Linux kernel at runtime. What is not to like?

This discussion will open with a review of recent changes to BPF, including the new ability for at least some BPF programs to be preemptible. It will continue with an overview of BPF use cases, which is hoped to set the stage for a discussion on how realtime and BPF can better live together.

I agree to abide by the anti-harassment policy

I agree

Primary author: MCKENNEY, Paul (Facebook)
Presenter: MCKENNEY, Paul (Facebook)
Session Classification: Real-time MC
Track Classification: Real-time MC
Virtio based communication between RC<->EP and between HOSTS connected to NTB

Wednesday, 26 August 2020 10:40 (20 minutes)

Existing Linux endpoint only supports pci-epf-test for communication between RootComplex and EndPoint systems (Both running Linux). While pci-epf-test is good enough for "testing" communication between RootComplex and Endpoint, additional development based on pci-epf-test was required for implementing any real use-cases.

This paper proposes to use existing Virtio infrastructure in Kernel used for
1. Communication between HOST and GUEST systems in Virtualization context
2. Communication between different cores in an SoC
to be used for RC<->EP communication and for communication between HOSTS connected to NTB.

Using the proposed mechanism, existing Virtio based drivers like rpmsg, net, scsi, blk etc.. could be made to be used for RootComplex and Endpoint communication.

The same mechanism can also be extended to be used for communication between HOSTS connected to NTB. Here instead of existing ntb_transport, virtio transport should be used.

The first RFC [1] posted garnered quite a bit of interest among the community and various approaches for designing it was discussed.

In this paper, Kishon will provide high-level view of how virtio could be used for RC<->EP communication and also discuss the various design approaches, with pros and cons of each approach and accelerate getting the community alignment of the overall design.


I agree to abide by the anti-harassment policy

I agree

Primary author: Mr I, Kishon Vijay Abraham
Presenter: Mr I, Kishon Vijay Abraham
Session Classification: VFIO/IOMMU/PCI MC
Track Classification: VFIO/IOMMU/PCI MC
Unify vDSOs across multiple architectures

Tuesday, 25 August 2020 08:25 (25 minutes)

vDSO (virtual dynamic shared object) is a mechanism that the Linux kernel provides as an alternative to system calls to reduce, where meaningful, the costs in terms of cycles. This is possible because certain syscalls like gettimeofday() do not write any data and return one or more values that are provided by the kernel, which makes calling them directly as a library function relatively safe.

Even if the mechanism is pretty much standard, every architecture in the last few years ended up implementing its own vDSO library in the architectural code. The purpose of this presentation is to examine the approach adopted from Linux 5.2 that identifies the commonalities between the architectures and tries to consolidate the common code paths in a unified vDSO library.

The presentation will start with a generic introduction to the vDSO concepts, it will proceed to cover some of the design choices, implementation details and issues encountered during the unification and it will conclude with an analysis of the possible future development (e.g. addition of new architectures, new syscalls conversions, new possible features, etc.).

I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr FRASCINO, Vincenzo
Presenter:  Mr FRASCINO, Vincenzo
Session Classification:  linux/arch/* MC
Track Classification:  linux/arch/* MC
BoF: Synchronizing timestamps of trace events between host and guest VM

Wednesday, 26 August 2020 08:00 (45 minutes)

Synchronization of kernel trace event timestamps between host and guest VM is a key requirement for analyzing the interaction between host and guest kernels. The task is not trivial, although both kernels run on the same physical hardware. There is a non-linear scaling of the guest clock, implemented intentionally by the hypervisor in order to simplify live guest migration to another host.

I’ll describe in short our progress on this task, using a PTP-like algorithm for calculating trace events timestamp offset. Any new ideas, comments, suggestions are highly welcomed.

I agree to abide by the anti-harassment policy

I agree

Primary author: STOYANOV, Tzvetomir
Presenter: STOYANOV, Tzvetomir
Session Classification: BOFs Session
Track Classification: Birds of a Feather (BoF)
fw_devlink is a new feature that got into upstream in the past year. It adds device links between devices by parsing the firmware. This talk will provide a quick refresher of what it does (talked a bit about it in LPC 2019) and what changes got in since then.

Also, provide a refresher on how sync_state() works and what it is and can be used for.

I agree to abide by the anti-harassment policy

I agree

Primary author:  KANNAN, Saravana
Presenter:  KANNAN, Saravana
Session Classification:  Android MC
Track Classification:  Android MC
IEEE802154 and RPLD updates

Thursday, 27 August 2020 09:20 (40 minutes)

This session will give an update on what happened in the IEEE802154 and 6lowpan subsystems since the last LPC IoT microconf. In addition it will present the newly added non-storing mode of our RPL Linux implementation, rpld.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr SCHMIDT, Stefan

Session Classification: You, Me, and IoT Two MC

Track Classification: You, Me, and IoT Two MC
Linux Kernel dependability - Proactive & reactive thinking

Tuesday, 25 August 2020 10:00 (30 minutes)

Let’s discuss proactive and reactive approaches to Linux Kernel dependability. We all care about keeping our data safe and systems secure. We counter security attacks using fuzzers and other test tools to identify vulnerabilities and hardening the code base.

How can we ensure we aren’t introducing new problems?

Regression testing and continuous fuzzing helps in finding regressions and new problems as code evolves and new features get added. All of these efforts are focused on finding and fixing existing problems.

Could we do more in understanding common design and coding mistakes to avoid and/or minimize introducing vulnerabilities. Could we be proactive in detecting and mitigating common weaknesses.

In this talk, we will discuss available detection and mitigation methods in the Linux Kernel to counter important Common Weaknesses Enumeration Categories such as Memory Buffer Errors and go over gaps if any.

I agree to abide by the anti-harassment policy

I agree

Primary author: KHAN, Shuah (The Linux Foundation)
Presenter: KHAN, Shuah (The Linux Foundation)
Session Classification: Kernel Dependability & Assurance MC
Track Classification: Kernel Dependability & Assurance MC
How do we kick our RT habit?

Monday, 24 August 2020 07:55 (25 minutes)

Inside our large database application setup, we have a few critical processes. Some of the functions include, heartbeat (for the cluster), monitoring what was happening (to debug in case a cluster does go down) amongst others.

Elaborating on a single example, if the heartbeat process doesn’t run when it should, the cluster could remove the node, and then the node would have to shutdown, which would then need the monitoring process to do more work to identify why we failed.

Clearly a database consumes a lot of CPU, and so these critical processes became RT and have been RT for a long time. With containers coming in, and RT cgroups being sub-optimal, maybe it is time to revisit this decision. We have some observations. Are these RT processes? Maybe not in the strict academic RT sense, but these are critical, time sensitive processes (with a deterministic function). Or does SCHED_OTHER need to be fixed for a clearly SCHED_OTHER problem?

Helmets advised for this discussion!

I agree to abide by the anti-harassment policy

I agree

Primary authors:  GIANI, Dhaval (Oracle);  SANGAPPA, Prakash (Oracle)

Presenters:  GIANI, Dhaval (Oracle);  SANGAPPA, Prakash (Oracle)

Session Classification:  Real-time MC

Track Classification:  Real-time MC
BoF: Android MC BoF

Wednesday, 26 August 2020 19:00 (4 hours)

This is a placeholder for the Android MC follow-up BoF that should be scheduled to run 48 to 72 hours after the Android MC.

I agree to abide by the anti-harassment policy

I agree

Primary authors: STULTZ, John (Linaro); KJOS, Todd (Google); IYER, Lina; SEMWAL, Sumit; YAGHMOUR, Karim (Opersys inc.)

Presenters: STULTZ, John (Linaro); KJOS, Todd (Google); IYER, Lina; SEMWAL, Sumit; YAGHMOUR, Karim (Opersys inc.)

Session Classification: BOFs Session

Track Classification: Birds of a Feather (BoF)
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

I agree

**Primary author:**  GLEIXNER, Thomas

**Presenter:**  GLEIXNER, Thomas

**Session Classification:**  Real-time MC

**Track Classification:**  Real-time MC
LLVM BOF

Monday, 24 August 2020 09:00 (45 minutes)

Come join us to work through issues specific to building the Linux kernel with LLVM. In addition to our Micro Conference, let’s carve out time to follow up on unresolved topics from our meetup in February:

- Improving our home page
- Status of each architecture
- Call to action / how to get started / Evangelism
- Improving Documentation/
- Maintainer model
- Minimum supported versions of LLVM
- s390 virtualized testing
- Follow ups to Rust in Kernel MC session


I agree to abide by the anti-harassment policy

I agree

Primary authors: DESAULNIERS, Nick (Google); WEBSTER, Behan (Converse in Code Inc.)

Presenters: DESAULNIERS, Nick (Google); WEBSTER, Behan (Converse in Code Inc.)

Session Classification: BOFs Session

Track Classification: Birds of a Feather (BoF)
Improving Kernel Builds with TuxMake and TuxBuild

Thursday, 27 August 2020 10:15 (15 minutes)

Reproducing build errors reported to a mailing list is a pain. How much time do we collectively spending asking "What kernel config did you use?", "What compiler?" and "What architecture"?

What if we could version and distribute build environments similarly to how we version Linux source code?

TuxMake is a tool that provides portable and repeatable Linux kernel builds across a variety of architectures, toolchains, kernel configurations, and make targets. Critically, it supports docker natively so that build environments are portable and builds are fully repeatable. TuxMake provides Docker images with cross build toolchains for a comprehensive set of supported architectures.

TuxMake provides both a command line tool and a Python API. With each build, you can specify the target architecture; which compiler to use; whether to use ccache, sccache, or doing a clean build; which targets to build; which kernel predefined configuration to start from, and which additional configurations to apply on top of that. You can pass arbitrary environment variables, also control the build concurrency. TuxMake is then responsible for running all the necessary commands to build a kernel to your specification, collect artifacts, logs, and extract metadata from the build environment.

TuxMake is in its early development stages, and is being designed to be extensible.

TuxBuild is a highly scalable and parallel Linux kernel building service. It consists of a REST API and a command-line client which can perform individual or pre-defined sets of builds. All builds happen on-demand, in parallel, and are easy to use both interactively and from a CI system.

TuxBuild solves the problem of build capacity and build automation, and allows kernel developers to perform more builds, more quickly, and more easily.

TuxMake is open source software, and TuxBuild is a private build service provided by Linaro.

More information about TuxMake and TuxBuild can be found at https://gitlab.com/Linaro/tuxmake and https://gitlab.com/Linaro/tuxbuild.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  RUE, Dan; TERCEIRO, Antonio (Linaro)

Presenters:  RUE, Dan; TERCEIRO, Antonio (Linaro)

Session Classification:  LLVM MC
Track Classification: LLVM MC
Understanding Linux Lists

Tuesday, 25 August 2020 07:10 (30 minutes)

Understanding the Linux kernel source code requires understanding the role played by different entities. An interesting example is the case of structures of type list_head. Some are actually heads of lists. Others are inlined inside of list elements. Documentation about which are which, and which heads are connected to which elements, is not systematic. We have developed a tool, Liliput, that takes into account how list_head structures are used to reconstruct this information. We have used the tool to find a few bugs, as well as to uncover some interesting list programming paradigms.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  LAWALL, Julia (Inria); VOLANSCHI, Nic (Inira)

Presenters:  LAWALL, Julia (Inria); VOLANSCHI, Nic (Inira)

Session Classification:  Kernel Dependability & Assurance MC

Track Classification:  Kernel Dependability & Assurance MC
Core Scheduling feature Upstreaming Plans

Tuesday, 25 August 2020 07:00 (50 minutes)

As a follow up to the OSPM discussion, we would like to discuss the upstreaming plans. As per the OSPM, the work left to be done were:
1. Documentation.
2. Cross cpu, vruntime comparison logic for CFS tasks.
3. Kernel protection from sibling during Syscall and interrupts.
4. Load balancing fixes.
5. API and usage.
6. Hotplug fixes.
7. Other fixes and code cleanup.

Now, v6 is released and we have made good progress. Documentation is mostly done and code has been cleaned up as per discussion at OSPM. Kernel protection from siblings during syscall and interrupts is also complete (pending posting and review). Hotplug fixes are also ready (pending posting and review). We need to work on vruntime comparison and load balancing fixes. Also API needs to be finalized.

The plan that we propose is to have a phased upstreaming approach. The code now could be considered almost feature complete, but with some known bugs. We propose to upstream the current code after a thorough review and then work on the known bugs aiming to get them in shortly there after:
1. Vruntime comparison.
2. Load balancing fixes.
3. Uperf regression reported by Aubrey (ksoftirqd getting force-idled).

The feature will be default-disabled on SMT systems and will be marked as experimental until all these known issues are fixed.

API is the other major thing. We have couple of different API proposals during OSPM/in mailing list, but did not reach a consensus:
1. Coresched specific cgroup.
2. prctl/sched_setattr.
4. Auto tagging based on process properties(user, group, VM etc).
5. Trusted cookie value (We can make 0 as default, and auto tag everything on fork).

The current API of cpu cgroups might not be worth upstreaming. We could either have the first phase go in without any API(not usable without out of tree patch) and then get API in soon after, or have a simple auto tagging interface(all tasks/processes under a separate tag, etc) in the first phase.

So, we propose 4 sessions:
1. Discuss vruntime priority comparison.
2. Discuss load balancer issue.
3. Discuss API.
4. Discuss upstreaming.

I agree to abide by the anti-harassment policy

I agree
**Primary authors:** DESFOSSEZ, Julien (DigitalOcean); FERNANDES, Joel; REMANAN PILLAI, Vineeth (DigitalOcean)

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

**Session Classification:** Scheduler MC

**Track Classification:** Scheduler MC
Safety in processes CPU execution state

Tuesday, 25 August 2020 08:00 (20 minutes)

A process running a safety critical function needs to be free from any interference. One source of this interference comes from are interruptions to the program flow from either synchronous events like system calls, or asynchronous events such as interrupts.

This talk details the sources of such events; the hazards that are associated with them, and some of the ways in which these may be mitigated. It will also go into some of the complexities of a modern processor such as an x86, showing what is considered to be the execution state and the issues surrounding monitoring the program flow.

We will show a mitigation developed to detect any changes in the execution state of a given process and discuss the limitations, performance and the issues raised during the development of the feature.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  DOOKS, Ben;  Mr PETERSOHN, Jens

Presenters:  DOOKS, Ben;  Mr PETERSOHN, Jens

Session Classification:  Kernel Dependability & Assurance MC

Track Classification:  Kernel Dependability & Assurance MC
Passing and retrieving information from bootloader and firmware

Thursday, 27 August 2020 10:25 (25 minutes)

Each operating system relies on the information exposed to it by the firmware. It consists of various data like memory map, device structure (either ACPI or devicetree), firmware version, vendor, etc. But passing information from operating system bootloader has been neglected for many years. In this presentation, we will mainly focus on retrieving information from firmware and bootloader by Linux kernel with a special focus on bootloader log and DRTM TPM event log.

I agree to abide by the anti-harassment policy

I agree

Primary authors: Mr KIPER, Daniel (Oracle); Mr ŻYGOWSKI, Michał (3mdeb Embedded Systems Consulting)

Presenters: Mr KIPER, Daniel (Oracle); Mr ŻYGOWSKI, Michał (3mdeb Embedded Systems Consulting)

Session Classification: System Boot and Security MC

Track Classification: System Boot and Security MC
LinuxBoot Ready is not ready: making linuxboot systems work

Thursday, 27 August 2020 08:35 (20 minutes)

A broad collection of companies are now using LinuxBoot for their firmware. They are still running into kexec issues involving drivers that don’t correctly shut down, start up, or still need the BIOS to set magic, undocumented bits.

We have to be able to mark drivers and associated code as "LinuxBoot Ready." This might be done in Kconfig with an option that would only present those drivers know to work with kexec.

But what does "work with" mean?

The goal of this talk is to discuss where LinuxBoot is now in use; what problems have been seen; and how we can deal with them.

I agree to abide by the anti-harassment policy

I agree

Primary author: MINNICH, ronald (Google)

Presenter: MINNICH, ronald (Google)

Session Classification: System Boot and Security MC

Track Classification: System Boot and Security MC
CFS flat runqueue v2

Tuesday, 25 August 2020 10:40 (30 minutes)

Last year I presented an approach to flatten the hierarchical runqueues used with the CPU controller in CFS, and Paul Turner came up with what we thought at the time were some insurmountable problems.

However, it looks like one relatively small change in how and when vruntime is accounted, and what is done with tasks that cannot have all of their delta exec runtime converted into vruntime at once, should resolve the corner cases that were present in last year’s code.

I hope to use this presentation and discussion session to ascertain whether that is indeed the case :)

I agree to abide by the anti-harassment policy

I agree

Primary author: VAN RIEL, Rik (Facebook)
Presenter: VAN RIEL, Rik (Facebook)
Session Classification: Scheduler MC
Track Classification: Scheduler MC
mikroBUS Driver for Add-on Boards

Thursday, 27 August 2020 07:05 (40 minutes)

mikroBUS is an add-on board socket standard by MikroElektronika that can be freely used by anyone following the guidelines. The mikroBUS standard includes SPI, I2C, UART, PWM, ADC, GPIO and power (3.3V and 5V) connections to interface common embedded peripherals, there are more than 750 add-on boards ranging from wireless connectivity boards to human-machine interface sensors which conform to the mikroBUS standard, out of which more than 140 boards already have device driver support in the Linux kernel. Today, the most straightforward method for loading these device drivers is to provide device-tree overlay fragments at boot time, this method suffers from the need to maintain a large out-of-tree database for which there is a need to maintain a separate overlay for every mikroBUS add-on board for every mikroBUS socket and also for targets that do not support dynamic loading of overlays, this method requires at-least a single reboot to enable the support in a potentially error-prone way.

The mikroBUS driver tries to solve the problem by introducing a new pseudo-bus driver (pseudo-bus since there is no actual bus controller involved) which enables the mikroBUS as a probeable bus such that the kernel can discover the device(s) on the bus at boot time, this is done by storing the add-on board device driver-specific information on a non-volatile storage accessible over one of the buses (currently on the mikroBUS I2C bus, subject to change) on the mikroBUS port. The format for describing the device driver-specific information is an extension to the Greybus manifest, the choice of using the Greybus manifest for the purpose is not entirely coincidental; there is ongoing work to evaluate ways to add mikroBUS sockets and devices via Greybus expansion and the manifest format can describe the device driver-specific data in a fairly good manner. With more than 100 clicks with tested support now, the mikroBUS driver makes use of the Unified Properties API and GPIO lookup tables for passing named properties and named GPIOs to device drivers. There are already several Linux platforms with mikroBUS sockets and the mikroBUS driver helps to reduce the time to develop and debug support for various mikroBUS add-on boards. Further, it opens up the possibility for support under dynamically instantiated busses such as Greybus.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr M A, Vaishnav (Beagleboard.org)

Session Classification:  You, Me, and IoT Two MC

Track Classification:  You, Me, and IoT Two MC
Native Booting using NVMe over Ethernet Fabrics

Thursday, 27 August 2020 09:00 (20 minutes)

NVMe over Fabrics™ (NVMe-oF™) lacks a native capability for boot from Ethernet. We will introduce a joint model to address boot from NVMe-oF/TCP, its impact to the kernel and the entire ecosystem, and collect feedback from the Linux community. This architectural model is being designed for standardization by the appropriate committees (e.g., NVM Express™ or UEFI™ Forum).

I agree to abide by the anti-harassment policy

I agree

Primary authors: FARLEY, Doug (Dell EMC); SZUBOWICZ, Lenny (Red Hat)

Presenters: FARLEY, Doug (Dell EMC); SZUBOWICZ, Lenny (Red Hat)

Session Classification: System Boot and Security MC

Track Classification: System Boot and Security MC
Secure boot without UEFI: booting VMs on Power

Thursday, 27 August 2020 07:10 (20 minutes)

Much of the Secure and Trusted Boot ecosystem is built around UEFI. However, not all platforms implement UEFI, including IBM’s Power machines.

In this talk, I present a proposal for secure boot of virtual machines on Power. This is an important use case, as many Power machines ship with a firmware hypervisor, and all user workloads run as virtual machines or “Logical Partitions” (LPARs).

Linux Virtual Machines on Power boot via an OpenFirmware (IEEE1275) implementation which is loaded by the hypervisor. The OpenFirmware implementation then loads grub from disk, and grub then loads Linux. To secure this, we propose to:

- Teach grub how to verify Linux-module-style "appended signatures". Distro kernels for Power are already signed with these signatures for use with the OpenPower ‘host’ secure boot scheme.
- Sign grub itself with an appended signature, allowing firmware to verify grub.

We’re really interested in feedback on our approach. We have it working internally and are preparing it for upstreaming, so now is the ideal time for us to get community input and answer any questions on the overall design and high-level implementation decisions.

I agree to abide by the anti-harassment policy

I agree

Primary author:  AXTENS, Daniel (IBM)
Presenter:  AXTENS, Daniel (IBM)
Session Classification:  System Boot and Security MC
Track Classification:  System Boot and Security MC
Advanced Applications of DRTM with TrenchBoot SecureLaunch for Linux

Thursday, 27 August 2020 10:00 (20 minutes)

The TrenchBoot Project has put forth an RFC for adding direct support to Linux for x86 DRTM. Many people are familiar with the early launch capability implemented by Intel’s tboot, but there has also been academic work on live relaunch, e.g. Jon McCune’s Flicker. SecureLaunch was designed to support a range of launch integrity capabilities. This discussion will review a subset of solutions that can be implemented using DRTM, along with roadmap candidates for SecureLaunch feature development.

I agree to abide by the anti-harassment policy

I agree

Primary author: SMITH, Daniel (Apertus Solutions, LLC)
Presenter: SMITH, Daniel (Apertus Solutions, LLC)
Session Classification: System Boot and Security MC
Track Classification: System Boot and Security MC
Q&A: GCC Steering Committee, GLIBC, GDB, Binutils Stewards

Wednesday, 26 August 2020 07:00 (25 minutes)

Question and Answer session and general discussion with members of the GCC Steering Committee, GLIBC Stewards, GDB Stewards, Binutils Stewards, and GNU Toolchain Fund Trustees.

I agree to abide by the anti-harassment policy

I agree

Primary author: EDELSOHN, David (IBM Research)
Presenter: EDELSOHN, David (IBM Research)
Session Classification: GNU Tools Track

Track Classification: GNU Tools Track
Designing and Packaging Printer/Scanner Drivers as Printer Application Snaps.

Friday, 28 August 2020 09:25 (30 minutes)

At the time of the Linux Plumbers 2020 taking place we have all the tools to create printer and scanner drivers in the new architecture: PAPPL, the Printer Application library gives us most of the always needed code for a standard-conforming IPP-printer-emulating Printer Application, cups-filters provides additional data format conversion code, and snapcraft creates the sandboxed Snap packages. Here we will present and discuss the workflow of designing and creating the drivers in the form of a Printer (and Scanner) Application and making a Snap ("snapping") it. The outcome of this session will also used in our Google Season of Docs project of creating a Printer/Scanner driver design and packaging tutorial.

I agree to abide by the anti-harassment policy

I agree

Primary author: KAMPPETER, Till (OpenPrinting / Canonical)
Presenter: KAMPPETER, Till (OpenPrinting / Canonical)
Session Classification: Open Printing MC
Track Classification: Open Printing MC
BoF: GDB

Monday, 24 August 2020 07:00 (25 minutes)

GDB BoF, for GDB developers to meet and discuss any topic about the GDB development process. Some proposed discussion topics are:

- The moving of gdbsupport and gdbserver, is anything left? Is there anything more to moved from the gdb to gdbsupport?
- Replacing of macros with a more C++-like API (like what has been started from the type system). Other C++-ification.
- Feedback on the new version numbering scheme.
- Large changes that people would like to pre-announce.
- Unsure how to approach the task of contributing an upstream port? This would be a good time to ask.

But really this is about what you want to discuss, so don’t hesitate to propose more topics. Please notify the moderator (Simon Marchi) in advance if possible, just so we can get a good overview of what people want to talk about.

I agree to abide by the anti-harassment policy

I agree

Primary author: MARCHI, Simon (EfficiOS)
Presenter: MARCHI, Simon (EfficiOS)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
IPPS Standards Landscape

Friday, 28 August 2020 10:00 (30 minutes)

- History of Internet Printing Protocol
  - IETF and PWG
- Recent IPP standards
  - IPP Everywhere
  - IPP System Service
  - IPP Transaction-based Printing Extensions
  - IPP 3D Printing Extensions
- Current IPP standards updates in progress
  - IPP Production Printing Extensions
  - IPP Enterprise Printing Extensions
  - IPP Driverless Printing Extensions
  - IPP Encrypted Jobs and Documents
  - Job Accounting with IPP
- Future IPP standards directions
  - Cloud Registration updates for IPP System Service
  - IPP 3D updates for additional technologies/materials
- Conclusions

I agree to abide by the anti-harassment policy

I agree

Primary author: MCDONALD, Ira (High North Inc / IEEE-ISTO PWG Secretary / IPP WG Co-Chair)

Presenter: MCDONALD, Ira (High North Inc / IEEE-ISTO PWG Secretary / IPP WG Co-Chair)

Session Classification: Open Printing MC

Track Classification: Open Printing MC
Printer Applications - The future of Printing in Linux.

Friday, 28 August 2020 07:10 (45 minutes)

Printer Applications replace CUPS printer drivers, solving numerous packaging, distribution, and support issues in the Linux printing environment. This session will provide some history, current developments, and future work that is needed to complete the transition from printer driver to printer application.

I agree to abide by the anti-harassment policy
I agree

Primary author: SWEET, Michael (Lakeside Robotics Corporation)
Presenter: SWEET, Michael (Lakeside Robotics Corporation)
Session Classification: Open Printing MC
Track Classification: Open Printing MC
3D Printing.

Friday, 28 August 2020 08:00 (45 minutes)

3D printing continues to be a hot topic, with both vendors and standards organizations competing to see who will determine how it will be used. This session will talk a little about the history of 3D printing, provide an overview of current standards efforts, and finally talk about the software and infrastructure that is needed on Linux to make 3D printing more accessible.

I agree to abide by the anti-harassment policy

I agree

Primary author:  SWEET, Michael (Lakeside Robotics Corporation)
Presenter:  SWEET, Michael (Lakeside Robotics Corporation)
Session Classification:  Open Printing MC
Track Classification:  Open Printing MC
Hot-adding a PCI device requires gaps in the address space for new BARs, and extra bus numbers if this is a bridge. Usually these resources are reserved not by the kernel, but by BIOS, bootloader, firmware.

If a bridge have windows not big enough (or fragmented too much) for newly requested BARs, it is still may be possible to allocate a memory region for new BARs, if at least some working BARs can be moved after pausing the drivers supporting this feature.

This approach is also useful if a BIOS don’t allocate all requested BARs, leaving some (for example, SR_IOV) unassigned, without gaps for bridge windows to extend. And it can help in allocating large (gigabytes in size) BARs.

Second (and optional) part is re-enumerating the buses allows hot-adding large switches in the middle of an existing PCIe tree, but it’s problematic point is renaming entries in /sys/bus/pci and /proc/bus/pci.

I agree to abide by the anti-harassment policy

I agree

Primary author: MIROSHNICHENKO, Sergei (Yadro)
Presenter: MIROSHNICHENKO, Sergei (Yadro)
Session Classification: VFIO/IOMMU/PCI MC
Track Classification: VFIO/IOMMU/PCI MC
BoF: IPE (Integrity Policy Enforcement) LSM merge discussion

Tuesday, 25 August 2020 09:00 (45 minutes)

Gather stakeholders from security, block, and VFS to discuss potential merging of the IPE LSM vs. integration with IMA.

Background:

- IPE: https://microsoft.github.io/ipe/
- Mailing list thread: https://lore.kernel.org/linux-security-module/20200802143143.GB20261@amd/T/#mc30b4a8fa55

I agree to abide by the anti-harassment policy

I agree

Primary authors: MORRIS, James; ZOHAR, Mimi (IBM)

Presenters: MORRIS, James; ZOHAR, Mimi (IBM)

Session Classification: BOFs Session

Track Classification: Birds of a Feather (BoF)
Why RISC-V Is Not Nearly Boring Enough

Wednesday, 26 August 2020 07:05 (30 minutes)

When RISC-V grows up, it wants to be a wildly successful computing platform. Being an ISA is fun but being the world’s fastest supercomputer would be really cool.

So how do we get there? By being dead boring. If I have an operating system to install on a platform built around the RISC-V ISA, the install MUST work out of the box – no mucking about with strange boot loaders, or grabbing odd bits of firmware and kernel patches. To do that means standardizing what a RISC-V platform looks like so that an OEM knows exactly what must be built, and so that an operating system knows what exactly what hardware and firmware it will find.

And let’s just say that right now, the RISC-V Platform Specification has a long way to go to. An OEM can only guess at what needs to be built; an OS can only run by using a lot of fiddly bits. These are some of my thoughts on what needs to be done:

. A clear vision
. A clear process
. A clear – and complete – specification

I agree to abide by the anti-harassment policy

I agree

Primary author:  STONE, Albert (Red Hat)
Presenter:  STONE, Albert (Red Hat)
Session Classification:  RISC-V MC
Track Classification:  RISC-V MC
Sane-airscan: the future of Linux driverless scanning

Friday, 28 August 2020 08:50 (30 minutes)

The driverless scanning came to Linux, allowing thousands of compatible devices, produced by many vendors, to just work. Alexander Pevzner, the author of sane-airscan SANE backend, will speak about present state and about perspectives.

I agree to abide by the anti-harassment policy

I agree

Primary author: PEVZNER, Alexander
Presenter: PEVZNER, Alexander
Session Classification: Open Printing MC
Track Classification: Open Printing MC
A Ridiculously Short Intro into Device Attestation

Thursday, 27 August 2020 09:35 (20 minutes)

A Ridiculously Short Intro into Device Attestation

Dimitar Tomov, Design First, ES
Ian Oliver, Nokia Bell Labs, FI

Very practical look at how to use a TPM and perform device attestation. A system can have trusted qualities instead of being 100% trusted. Cross-referencing different types of attestation data can provide evidence for trusted qualities. The decision of whether a device is trusted is not responsibility of the attestor and verifier - these just gather and check the evidence. Example use cases of Time Attestation.

Intro

Use of Trusted Platform Modules (TPM), Measured Boot and [Remote] Attestation can provide significant security benefits to, arguably, the most sensitive and critical parts of a system, particularly the firmware and initial boot. However, the verification of attestation claims can be daunting and complex.

In this presentation, we briefly describe what measurements are and can be taken, how these are reported by a TPM. What the TPM attest structures contain and how this information can be better understood in terms of device identity, configuration parameters, temporal aspects etc.

We conclude with a short demonstration (example as presentation platform allows) of attestation of trustable devices (servers, IoT, etc) focusing on certain temporal and device identity aspects.

I agree to abide by the anti-harassment policy

I agree

Primary authors: Mr TOMOV, Dimitar (DesignFirst); Mr OLIVER, Ian (Nokia Bell Labs)

Presenters: Mr TOMOV, Dimitar (DesignFirst); Mr OLIVER, Ian (Nokia Bell Labs)

Session Classification: System Boot and Security MC

Track Classification: System Boot and Security MC
Identifying Sources of OS Noise

Monday, 24 August 2020 10:10 (25 minutes)

This topic focuses on identifying sources of operating system "noise", primarily for polling mode latency-sensitive applications on Linux. What do we mean by operating system noise? We mean things external to an application that can affect execution of the application in a negative way, usually meaning a delay in execution causing missed deadlines. The intent here is to identify the most common noise generators and stimulate discussion on techniques for mitigating them.

Noise is not a new topic for Linux and especially the Linux PREEMPT_RT community, but over the years the performance parameters have changed. Instead of a single system being deployed to run a single realtime application with max latency thresholds of 100 microseconds, we now see one system with hundreds of cores deployed to service a mix of realtime and non-realtime applications. Some of the realtime application thresholds are in the low-double digit microsecond range. As tolerances decrease, the acceptable ceiling for noise must also decrease. A delay of 15μs might have been acceptable when the max latency was 100μs, but when max latency is 20μs, 15μs is entirely unacceptable. We need to come up with ways to wall-off these low-latency applications and protect them from sources of noise.

I agree to abide by the anti-harassment policy

I agree

Primary authors: WILLIAMS, Clark (Red Hat); LELLI, Juri (Red Hat)

Presenters: WILLIAMS, Clark (Red Hat); LELLI, Juri (Red Hat)

Session Classification: Real-time MC

Track Classification: Real-time MC
System call wrappers for glibc

Friday, 28 August 2020 09:35 (45 minutes)

Most programmers prefer to call system calls via functions from their C library of choice, rather than using the generic syscall function or custom inline-assembler sequences wrapping a system call instruction. This means that it is desirable to add C library support for new system calls, so that they become more widely usable.

This talk covers glibc-specific requirements for adding new system call wrappers to the GNU C Library (glibc), namely code, tests, documentation, patch review, and copyright assignment (not necessarily in that order). Developers can help out with some of the steps even if they are not familiar with glibc procedures or have reservations about the copyright assignment process.

I plan to describe the avoidable pitfalls we have encountered repeatedly over the years, such as tricky calling conventions and argument types, or multiplexing system calls with polymorphic types. The ever-present temptation of emulating system calls in userspace is demonstrated with examples.

Finally, I want to raise the issue of transition to new system call interfaces which are a superset of existing system calls, and the open problems related to container run-times and sandboxes with seccomp filters—and the emergence of non-Linux implementations of the Linux system call API.

The intended audience for this talk are developers who want to help with getting system call wrappers added to glibc, and kernel developers who define new system calls or review such patches.

I agree to abide by the anti-harassment policy

I agree

Primary author: WEIMER, Florian (Red Hat)

Presenter: WEIMER, Florian (Red Hat)

Session Classification: GNU Toolchain MC

Track Classification: GNU Toolchain MC
Various static analysis tools have been used for many years in the kernel development; even more, some static analysis tools have dedicatedly been developed in the realm of the kernel community.

While with the introduction of the first static analysis tools, some relevant kernel bugs were found and fixed, the repeated execution of those static analysis tools on recent kernels suffer from a large set of false positives compared to the really relevant findings that would require attention and fixing.

So making use of these results in the long term requires to track the false positives. Most efforts using static analysis tools and tracking false positives have been done by single individuals in the community. For single individuals doing this with a long history of following the kernel development with a specific tool in mind, some simple light non-distributed solutions might be sufficient for tracking false positives.

However for anyone that would like to involve in following these static analysis findings or for a larger open group to continuously assess findings more technology and organisational setup is needed.

I would like to discuss if we see a critical mass for running some static analysis tools, maintaining a database of false positive findings of static analysis tools collaboratively, what is the technical setup required to maintain those findings, and what are the organisational steps that should be taken towards establishing such a collaborative effort.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr BULWAHN, Lukas
Presenter:  Mr BULWAHN, Lukas
Session Classification:  Kernel Dependability & Assurance MC
Track Classification:  Kernel Dependability & Assurance MC
Kselftest running in test rings - Where are we?

Wednesday, 26 August 2020 08:05 (20 minutes)

Kselftest is a developer test suite which has evolved to run in test rings, and by distributions. This evolution hasn't been an easy one.

In this talk, Shuah shares what it takes to get Kselftest running in test rings such as Kernel CI. She will go over the changes necessary to run Kselftests to fully support relocatable builds and enable integration into test rings.

The primary goal is discussion on existing problems and blockers to run Kselftest in Kernel CI.

I agree to abide by the anti-harassment policy

I agree

Primary author: KHAN, Shuah (The Linux Foundation)
Presenter: KHAN, Shuah (The Linux Foundation)
Session Classification: Testing and Fuzzing MC
Track Classification: Testing and Fuzzing MC
LTO, PGO, and AutoFDO in the kernel

Newer compiler optimization techniques stand to improve the runtime performance of Linux kernels. These techniques analyze more of a program (Link Time Optimization aka "LTO") or make use of profiling information to improve code layout (Profile Guided Optimization "PGO" and Automatic Feedback Directed Optimization "AutoFDO"). Now that Google is shipping all three in various kernel distributions, let's take a look at the tradeoffs and path towards upstreaming these patch series.

I agree to abide by the anti-harassment policy

I agree

Primary authors: TOLVANEN, Sami (Google); WENDLING, Bill (Google); DESAULNIERS, Nick (Google)

Presenters: TOLVANEN, Sami (Google); WENDLING, Bill (Google); DESAULNIERS, Nick (Google)

Session Classification: LLVM MC

Track Classification: LLVM MC
BoF: Speed vs accuracy for math library optimization

Thursday, 27 August 2020 07:30 (25 minutes)

Math library developers sometimes can trade slight loss of accuracy for significant performance gains or slight loss of performance for significant accuracy gains. This BoF is to review some recent and coming libm/libgcc changes and share ideas on how to decide where to draw the line for loss of performance vs improved accuracy and vice-versa.

I agree to abide by the anti-harassment policy

I agree

Primary author:  MCGEHEARTY, Patrick (Oracle)
Presenter:  MCGEHEARTY, Patrick (Oracle)
Session Classification:  GNU Tools Track

Track Classification:  GNU Tools Track
Thermal: Use of running average of temperature for thermal thresholds

Friday, 28 August 2020 08:30 (25 minutes)

In the current thermal core, occasional spikes can cause thermal shutdowns or any associated processing. There are several reports in bug databases. Instead of each thermal driver coming up with its own mechanism, the thermal core can optionally use running average for threshold processing.

I agree to abide by the anti-harassment policy

I agree

Primary author: PANDRUVDADA, Srinivas
Presenter: PANDRUVDADA, Srinivas
Session Classification: Power Management and Thermal Control MC
Track Classification: Power Management and Thermal Control MC
Remote offline of a CPU through Hardware Feedback Interface to increase system TDP

Friday, 28 August 2020 07:50 (25 minutes)

Intel Hardware provides guidance to the Operating System (OS) scheduler to perform optimal workload scheduling through a hardware feedback interface structure in memory. Via this interface Hardware can also provide recommendation to the OS to not schedule any software threads on a CPU, so essentially offline a CPU remotely. There are three methods to implement this, each has its own advantages and disadvantages. Discuss the best method to implement this feature.

I agree to abide by the anti-harassment policy

I agree

Primary author: PANDRUVADA, Srinivas
Presenter: PANDRUVADA, Srinivas
Session Classification: Power Management and Thermal Control MC
Track Classification: Power Management and Thermal Control MC
Criteria of using VFIO mdev (vs. userspace DMA)

Wednesday, 26 August 2020 07:00 (20 minutes)

VFIO mdev provides a framework for subdevice assignment and reuses existing VFIO uAPI to handle common passthrough-related requirements. However, subdevice (e.g. ADI defined in Intel Scalable IOV) might not be a PCI endpoint (e.g. just a work queue), thus requires some degree of emulation/mediation in kernel to fit into VFIO device API. Then there is a concern on putting emulation in kernel and how to judge abuse of mdev framework by simply using it as an easy path to hook into virtualization stack. An associated open is about differentiating mdev from userspace DMA framework (such as uacce), and whether building passthrough features on top of userspace DMA framework is a better choice than using mdev.

I agree to abide by the anti-harassment policy

I agree

Primary author:  TIAN, Kevin (Intel)
Presenter:  RAJ, Ashok
Session Classification:  VFIO/IOMMU/PCI MC
Track Classification:  VFIO/IOMMU/PCI MC
Cross-architecture collaboration panel

Tuesday, 25 August 2020 08:00 (25 minutes)

Open discussion about the ways to improve collaboration between developers working on different architectures.

I agree to abide by the anti-harassment policy

I agree

Primary author: RAPOPORT, Mike (IBM)
Session Classification: linux/arch/* MC
Track Classification: linux/arch/* MC
Towards Learning From Linux Kernel Configurations’ Failures with Clang

Thursday, 27 August 2020 10:00 (15 minutes)

The Linux kernel offers more than ten thousands configuration options that can be combined to build an almost infinite number of kernel variants. Developers and contributors spend significant effort and computational resources to continuously track and hopefully fix configurations that lead to build failures. In this talk, we report on our endeavor to develop an infrastructure, called TuxML, able to build any kernel configuration and learn what could explain or even prevent configurations’ failures. We will present some insights over 300K+ configurations coming from different releases/versions of the kernel. Our results show that TuxML can accurately cluster failures, automatically trace the responsible configuration options, and learn by itself to avoid unnecessary and costly builds.

In the last part of the talk, we will discuss the applicability of TuxML as well as the open challenges when building in the large kernel configurations with Clang. We believe there is potential to better understand problematic cases (through clustering and statistical learning) and such insights can drive the improvement of Clang-based building of Linux.

I agree to abide by the anti-harassment policy

I agree

Primary author: Prof. ACHER, Mathieu (University of Rennes 1)
Presenter: Prof. ACHER, Mathieu (University of Rennes 1)
Session Classification: LLVM MC
Track Classification: LLVM MC
Lightning Talk: Teaching GraalVM DWARFishe: Debugging Native Java using gdb

Or is it DWARVish? Whatever, GraalVM Native implements compilation of a complete suite of Java application classes to a single, complete, native ELF image. It’s much like how a C/C++ program gets compiled. Well, except that the image contains nothing to explain how the bits were derived from source types and methods or where those elements were defined. Oh and the generated code is heavily inlined and optimized (think gcc -O2/3). Plus many JDK runtime classes and methods get substituted with lightweight replacements. So, a debugging nightmare.

Anyway, we have resolved the debug problem much like how you do with C/C++ by generating DWARF records to accompany and explain the program bits. So far, we have file and line number resolution, breakpoints, single stepping & stack backtraces. We’re now working on type names and layouts and type, location & liveness info for heap-mapped values/objects, parameters and local vars. I’ll explain how we obtain the necessary input from the Java compiler, how we model it as DWARF records and how we test it for correctness using objdump and gdb itself.

By that point I will probably need to stop to take a breath

I agree to abide by the anti-harassment policy

I agree

Primary author: DINN, Andrew (Red hat)
Presenter: DINN, Andrew (Red hat)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Tutorial: GNU poke, what is new in 2020

Monday, 24 August 2020 09:30 (55 minutes)

It’s been almost a year since the nascent GNU poke \(^1\) got first introduced to the public at the GNU Tools Cauldron 2019 in Montreal. We have been hacking a lot during these turbulence months and poke is maturing fast and approaching a first official release, scheduled for late summer.

In this talk we will first do a quick introduction to the program for the benefit of the folk still unfamiliar with it. Then we will show (and demonstrate) the many new features introduced during this last year: full support for union types, styled output, struct constructors, methods and pretty-printers, integral structs, the machine-interface, support for Poke scripts, and many more. Finally, we will be tackling some practical matters (what we call “Applied Pokology”\(^2\)) useful for toolchain developers, such as how to write binary utilities in Poke, how to best implement typical C data structures in Poke type descriptions, and our plans to integrate poke with other toolchain components such as GDB.

<\b> About GNU poke</b>

GNU poke is an interactive, extensible editor for binary data. Not limited to editing basic entities such as bits and bytes, it provides a full-fledged procedural, interactive programming language designed to describe data structures and to operate on them.

\(^1\) [http://www.jemarch.net/poke](http://www.jemarch.net/poke)

\(^2\) [http://www.jemarch.net/pokology](http://www.jemarch.net/pokology)

I agree to abide by the anti-harassment policy

I agree

---

**Primary author:** MARCHESI, Jose E. (GNU Project, Oracle Inc.)

**Presenter:** MARCHESI, Jose E. (GNU Project, Oracle Inc.)

**Session Classification:** GNU Tools Track

**Track Classification:** GNU Tools Track
GCC’s -fanalyzer option

I’ll be talking about the -fanalyzer static analysis option I added in GCC 10; give an overview of the internal implementation, its current strengths and limitations, on how I’m reworking it for GCC 11, and ideas for future directions.

I agree to abide by the anti-harassment policy

I agree

Primary author: MALCOLM, David (Red Hat)
Presenter: MALCOLM, David (Red Hat)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Lightning Talk: Fuzzing glibc’s iconv program

Tuesday, 25 August 2020 08:30 (10 minutes)

A while back, I found myself triaging an iconv bug report that found hangs in the program when run with certain inputs. Not knowing a lot about iconv internals, I wrote a rudimentary fuzzer to investigate the problem, which caught over 160 different input combinations that led to hangs and a clear pattern hinting at the cause.

In this short talk, I’ll share my experiences with fuzzing iconv and eventually cleaning up some of the iconv front-end with a patch.

I agree to abide by the anti-harassment policy

I agree

Primary author: SHANKAR, Arjun (Red Hat)
Presenter: SHANKAR, Arjun (Red Hat)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Energy Model evolution possibilities

Friday, 28 August 2020 07:00 (25 minutes)

The Energy Model (EM) framework aims to provide information about energy consumption of a given performance domain. The power values stored for each performance level are used during calculation in Energy Aware Scheduler (EAS) or in thermal framework for the CPUfreq cooling device. Recently the EM has been extended to support other devices than CPUs (like GPUs, DSP, etc). It opens new possibilities to use the EM framework and the first proposed is the Devfreq cooling device. Another one is to use EM together with CPU utilization signal maintained by the task scheduler to estimate the energy consumption in the CPU cooling device. Furthermore, the EM could help to control the capping (in thermal or in powercap frameworks) in a more generic way. This presentation will discuss the new use cases and the proposed design, as well as existing obstacles and corner cases.

I agree to abide by the anti-harassment policy

I agree

Primary author:  LUBA, Lukasz
Presenter:  LUBA, Lukasz
Session Classification:  Power Management and Thermal Control MC
Track Classification:  Power Management and Thermal Control MC
Lightning Talk: Accelerating machine learning workloads using new GCC built ins

Wednesday, 26 August 2020 08:00 (10 minutes)

Basic Linear Algebra Subprograms (BLAS) are used everywhere in machine learning and deep learning applications today. OpenBLAS is an optimized BLAS open source library used widely in AI workloads that implement algebraic operations for specific processor types. This talk covers recent optimization in the OpenBLAS library for the POWER10 processor. As part of this optimization, assembly code for matrix multiplication kernels in OpenBLAS is converted to C code using new compiler builtins. A sample optimization for matrix multiplication for POWER hardware in OpenBLAS will be used to explain how builtins are used and show the impact of application performance.

I agree to abide by the anti-harassment policy

I agree

Primary author: S, Rajalakshmi
Presenter: S, Rajalakshmi
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Update on the BPF support in the GNU Toolchain

Wednesday, 26 August 2020 09:30 (25 minutes)

In 2019 Oracle contributed support for the eBPF (as of late renamed to just BPF) in-kernel virtual architecture to binutils and GCC. Since then we have continued working on the port, and recently sent a patch series upstream adding support for GDB and the GNU simulator.

This talk will describe this later work and other current developments, such as the gradual introduction of xbpf, a variant of BPF that removes most of the many restrictions in BPF, originally conceived as a way to ease the debugging of the port itself and of BPF programs, but that can also be leveraged in non-kernel contexts that could benefit from a fully-toolchain-supported virtual architecture.

I agree to abide by the anti-harassment policy
I agree

Primary author: MARCHESI, Jose E. (GNU Project, Oracle Inc.)
Presenter: MARCHESI, Jose E. (GNU Project, Oracle Inc.)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
New frontiers in CTF linking: type deduplication

Tuesday, 25 August 2020 09:30 (25 minutes)

Last year we introduced support for the Compact C Type Format (CTF) to the GNU toolchain and presented at the last Cauldron.

Back then, the binutils side was only doing slow, non-deduplicating linking and format dumping, but things have moved on. The libctf library and ld in binutils has gained the ability to properly deduplicate CTF: output CTF in linked ELF objects is now often smaller than the CTF in any input object file. The performance hit of deduplication is usually in the noise or at least no more than a second or two (and there are still some easy performance wins to pick).

The libctf API has also improved somewhat, with support for a number of missing features, improved error reporting, and a much-improved way to iterate over things in the CTF world.

This talk will provide an overview of the novel type deduplication algorithm used to reduce the size of CTF, with occasional diversions into the API improvements where necessary, and (inevitably) discussion of upcoming work in the area, solicitations of advice from others working on similar things, etc.

I agree to abide by the anti-harassment policy

I agree

Primary author: ALCOCK, Nick (Oracle Corporation)
Presenter: ALCOCK, Nick (Oracle Corporation)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
We had a panel led discussion at last year’s GNU Tools Cauldron and more recently at the FOSDEM LLVM Developer’s room on improving cooperation between GNU and LLVM projects. This year we are proposing an open format BoF, particularly because we believe that being part of LPC and a virtual conference we may have more LLVM and GNU developers in the same (virtual) room.

At both previous sessions we have explored the issues, but struggled to come up with concrete actions to improve cooperation. This BoF will attempt to find concrete actions that can be taken.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Dr BENNETT, Jeremy (Embecosm)
Presenter:  Dr BENNETT, Jeremy (Embecosm)
Session Classification:  GNU Tools Track

Track Classification:  GNU Tools Track
BoF: Binutils

Tuesday, 25 August 2020 07:00 (25 minutes)

A BoF meeting for folks interested in the GNU Binutils.
Possible topics for discussion:
* Should GOLD be dropped?
* Automatic changelog generation.
* Configuring without support for old binary formats (eg ihex, srec, tekhex, verilog)

I agree to abide by the anti-harassment policy
I agree

Primary author:  CLIFTON, Nick
Presenter:  CLIFTON, Nick
Session Classification:  GNU Tools Track
Track Classification:  GNU Tools Track
Lightning talk: RISC-V Bitmanip optimizations

Thursday, 27 August 2020 08:00 (10 minutes)

Support for the bit manipulation extension to RISC-V is currently out-of-tree and represents work by Jim Wilson at SiFive, Claire Wolf at Symbiotic EDA and Maxim Blinov at Embecosm. Since last year, I have been working on additional optimizations for the bit manipulation extension, which I shall present.

I agree to abide by the anti-harassment policy

I agree

Primary author: BLINOV, Maxim (Embecosm)
Presenter: BLINOV, Maxim (Embecosm)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Enable Intel CET in Linux OS

Thursday, 27 August 2020 09:30 (25 minutes)

This is a follow up report of Intel CET enabling in Linux OS. I will update the current status of Intel CET with binutils, glibc, GCC, LLVM and Linux kernel as well as Linux distributions. CET presentation can be downloaded from here.

I agree to abide by the anti-harassment policy

I agree

Primary author:  LU, H.J. (Intel)
Presenter:  LU, H.J. (Intel)
Session Classification:  GNU Tools Track
Track Classification:  GNU Tools Track
BoF: KernelCI Unified Reporting in Action

Thursday, 27 August 2020 07:00 (45 minutes)

See the Kernel CI’s new Unified Reporting in action: from multi-CI submission, through common dashboards and notification subscription, to report emails.

Explore and discuss the report schema and protocol. Learn how to send testing results, using your own, or example data. Help us accommodate your reporting requirements in the schema, database, dashboards and emails.

Bootstrap automatic sending of your system’s results to the common database, with our help. Discuss future development, dive into implementation details, explore and hack on the code, together with the development team.

I agree to abide by the anti-harassment policy

I agree

Primary author: KONDRASHOV, Nikolai (Red Hat)
Presenter: KONDRASHOV, Nikolai (Red Hat)
Session Classification: BOFs Session
Track Classification: Birds of a Feather (BoF)
The Light-Weight JIT Compiler Project

Monday, 24 August 2020 08:30 (25 minutes)

Recently CRuby got a JIT based on GCC or Clang. Experience with use of the CRuby JIT confirmed the known fact that GCC does not fit well for all JIT usage scenarios. Ruby needs a light-weight JIT compiler used as a tier 1 compiler or as a single JIT compiler. This talk will cover experience of GCC usage for CRuby JIT and drawbacks of GCC as a tier 1 JIT compiler. This talk also will cover the light-weight JIT compiler project motivations, current and possible future states of the project.

I agree to abide by the anti-harassment policy

I agree

Primary author:  MAKAROV, Vladimir
Presenter:  MAKAROV, Vladimir
Session Classification:  GNU Tools Track
Track Classification:  GNU Tools Track
ION/DMABUF-Heaps Transition & DMABUF cache handling

Monday, 24 August 2020 08:30 (15 minutes)

Covers issues and TODOs for the transition from ION to the upstream DMA-BUF Heaps infrastructure.

Also will discuss thoughts on DMA-BUF cache handling, following up from LWN articles here:
https://lwn.net/Articles/822052/
https://lwn.net/Articles/822521/

I agree to abide by the anti-harassment policy

I agree

Primary author: STULTZ, John (Linaro)
Presenter: STULTZ, John (Linaro)
Session Classification: Android MC
Track Classification: Android MC
Partial Cache Flushing w/ DMA-BUFs

Monday, 24 August 2020 08:45 (15 minutes)

Covering patches used in the Android Common tree to provide partial cache flushes for DMA-BUFs, and what issues and blockers need to be resolved for this functionality to move upstream.

I agree to abide by the anti-harassment policy

I agree

Primary author: VALSARAJU, Hridya
Presenter: VALSARAJU, Hridya
Session Classification: Android MC
Track Classification: Android MC
Update on libcamera in AOSP

Monday, 24 August 2020 09:00 (15 minutes)

Major new features since last year’s LPC include

- RAW capture
- GStreamer element
- V4L2 compatibility layer (allowing V4L2 application to use libcamera transparently through LD_PRELOAD)
- Support for new platforms
- Raspberry Pi (with a fully open-source implementation of the image processing algorithms)
- i.MX7

We have continued working on the Android HAL implementation to pass the Android camera test suite (CTS), and will resume discussions with Android devs once we complete this. There’s also been interest from multiple SoC vendors who are interested in supporting libcamera and are working to port their camera stack to it.

I agree to abide by the anti-harassment policy

I agree

Primary author: PINCHART, Laurent (Ideas on Board Oy)
Presenter: PINCHART, Laurent (Ideas on Board Oy)
Session Classification: Android MC
Track Classification: Android MC
Will cover outstanding and recently upstreamed patches from Android Common Kernel that are needed for Android to function.

Including a brief overview of Android Common tree, device specific changes, the vma-naming patch and inline encryption functionality.

I agree to abide by the anti-harassment policy

I agree

Primary authors: SEMWAL, Sumit; TANGIRALA, Satya

Presenters: SEMWAL, Sumit; TANGIRALA, Satya

Session Classification: Android MC

Track Classification: Android MC
Incremental Filesystem

Overview of the new Incremental Filesystem in the Android Common Kernel and discussion on issues or blockers to getting the functionality upstream.

I agree to abide by the anti-harassment policy

I agree

Primary author: LAWRENCE, Paul
Presenter: LAWRENCE, Paul
Session Classification: Android MC
Track Classification: Android MC
Android Upstreaming TODOs (dm-user)

Monday, 24 August 2020 09:45 (15 minutes)

Covering outstanding up-streaming efforts for patches in the Android Common tree:

Specifically:
- dm-user

I agree to abide by the anti-harassment policy

I agree

Primary authors:  ANDERSON, David; LAWRENCE, Paul

Presenters:  ANDERSON, David; LAWRENCE, Paul; DABBELT, Palmer (Google)

Session Classification:  Android MC

Track Classification:  Android MC
Making RISC-V EBBR compatible

Wednesday, 26 August 2020 07:35 (25 minutes)

There are ongoing efforts to add UEFI support for RISC-V Linux kernel. As a result, RISC-V can be fully EBBR compatible. We will discuss the current progress and what’s the best approach to make that happen.

I agree to abide by the anti-harassment policy

I agree

Primary author: PATRA, ATISH (Western Digital)
Presenter: PATRA, ATISH (Western Digital)
Session Classification: RISC-V MC
Track Classification: RISC-V MC
RISC-V 32-bit glibc port

Wednesday, 26 August 2020 10:30 (30 minutes)

This will include details about the 64-bit time_t problem and how RV32 is going to be the first 32-bit architecture with a 64-bit time_t. What still needs to be done for 32-bit support? How do we get this merged? We will also like to discuss the plan to test and maintain it once it is merged.

I agree to abide by the anti-harassment policy

I agree

Primary author: FRANCIS, Alistair
Presenter: FRANCIS, Alistair
Session Classification: RISC-V MC
Track Classification: RISC-V MC
The hypervisor extension v0.5 is already available in the latest Qemu and v0.6.1 patches are already in the mailing list. The kvm patches has been on the mailing list and waiting to be merged. We will discuss the ongoing designs for nested hypervisor implementation.

I agree to abide by the anti-harassment policy

I agree

Primary author:  PATEL, Anup (Western Digital)
Presenter:  PATEL, Anup (Western Digital)
Session Classification:  RISC-V MC
Track Classification:  RISC-V MC
An introduction of vector ISA support in RISCV Linux

Wednesday, 26 August 2020 09:15 (30 minutes)

We will talk about the implementation of vector support in Linux kernel, how user space can get its layout or size and the future work for Linux kernel and glibc.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  HU, Greentime (SiFive);  CHEN, Vincent (SiFive)

Presenters:  HU, Greentime (SiFive);  CHEN, Vincent (SiFive)

Session Classification:  RISC-V MC

Track Classification:  RISC-V MC
Kludging The editor with The compiler

Thursday, 27 August 2020 08:30 (25 minutes)

Emacs Lisp (Elisp) is the Lisp dialect used by the Emacs text editor family. GNU Emacs can currently execute Elisp code either interpreted or byte-interpreted after it has been compiled to byte-code. In this presentation I’ll discuss the libgccjit based Elisp compiler implementation being integrated in Emacs. Though still a work in progress, this implementation is able to bootstrap a functional Emacs and compile all Emacs Elisp files, including the whole GNU Emacs Lisp Package Archive (ELPA). Native compiled Elisp shows an increase of performance ranging from ~2x up to ~40x with respect to the equivalent byte-code, measured over a set of small benchmarks.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr CORALLO, Andrea (Arm)
Presenter:  Mr CORALLO, Andrea (Arm)
Session Classification:  GNU Tools Track
Track Classification:  GNU Tools Track
Flatpak - a desktop version of containers

Flatpak is a sandboxing system targeting Linux Desktop applications. This talk will explain how flatpak uses various Linux kernel and userspace features to implement sandboxing, and compare and contrast to how it works with server-side container systems like Docker.

It will also talk about future plans and ideas in this area, including things that we can do with existing frameworks as well as things that would require new kernel or userspace features.

I agree to abide by the anti-harassment policy

I agree

Primary author: LARSSON, Alexander (Red Hat)
Presenter: LARSSON, Alexander (Red Hat)
Session Classification: Application Ecosystem MC
Track Classification: Application Ecosystem MC
Linux RISC-V Kernel Policy for Draft Specs

Wednesday, 26 August 2020 10:00 (30 minutes)

The Linux RISC-V Kernel has adopted a policy to accept patches only for frozen/ratified RISC-V specs. This was done to align with RISC-V spec development process of the RISC-V Foundation and avoid maintenance burden. Considering the time taken by RISC-V spec development process, is there a better policy which Linux RISC-V Kernel can adopt??

Projects such as QEMU RISC-V and OpenSBI have been accepting patches for draft specs without any issues. The policy adopted by these projects is as follows:
1) Features/functionality pertaining to draft spec will not be enabled by default
2) Backward-compatibility will not be maintained for features/functionality pertaining to draft spec

This talk is a place-holder for discussing above described Linux RISC-V Kernel Policy on draft specs.

I agree to abide by the anti-harassment policy

I agree

Primary authors: FRANCIS, Alistair; PATEL, Anup (Western Digital)
Co-authors: PATRA, ATISH (Western Digital); PATRA, ATISH (Western Digital)
Presenter: FRANCIS, Alistair
Session Classification: RISC-V MC
Track Classification: RISC-V MC
Lightning Talk: elfutils debuginfod http-server progress: clients and servers

Monday, 24 August 2020 08:00 (10 minutes)

We will recap the elfutils debuginfod server from last year. It has been integrated into a number of consumers, learned to handle a bunch of distro packaging formats, and some public servers are already online.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  EIGLER, Frank;  MEREY, Aaron (Red Hat)

Presenters:   EIGLER, Frank;  MEREY, Aaron (Red Hat)

Session Classification:  GNU Tools Track

Track Classification:  GNU Tools Track
Project Ranger Update

Monday, 24 August 2020 09:00 (25 minutes)

The Ranger project was introduced at the GNU tools Cauldron last year. This project provides GCC with enhanced ranges and an on-demand range query API. By the time the conference is on, we expect to have the majority of the code in trunk and available for other passes to utilize.

In this update, we will:

- Cover what has changed since last fall.
- Describe current functionality, including the API that is available for use.
- Plans going forward / Whats in the pipe.

I agree to abide by the anti-harassment policy

I agree

Primary author: MACLEOD, Andrew (Red Hat)
Co-author: HERNANDEZ, Aldy (Red Hat)
Presenters: HERNANDEZ, Aldy (Red Hat); MACLEOD, Andrew (Red Hat)
Session Classification: GNU Tools Track

Track Classification: GNU Tools Track
Lightning Talk: The challenges of GNU tool chain support for CORE-V

Thursday, 27 August 2020 08:15 (10 minutes)

CORE-V is a family of 32- and 64-bit cores based on the RISC-V architecture, being developed by the Open Hardware Group, a consortium of 50+ companies, universities and other organizations. It is based on the family of RISC-V cores originally developed under the PULP project at ETH Zürich and the University of Bologna.

PULP cores already have an out-of-tree GNU tool chain, but it is based on GCC of 2017, and as would be expected is developed as a research compiler to experiment with different extensions to the core. This talk will explore the challenges of getting from this tool chain to an up to date GNU tool chain, in-tree. The areas to be explored include

- migrating from a 2017 code base (still a lot of C) to the 2020 code base (C++)
- retrospectively adding tests for 2,700 new instruction variants and their associated compiler optimizations
- upstreaming extensions which, while present in manufactured silicon and products, are not yet approved by the RISC-V Foundation

I agree to abide by the anti-harassment policy

I agree

Primary authors: Dr BENNETT, Jeremy (Embecosm); Dr BLACKMORE, Craig (Embecosm)

Presenters: Dr BENNETT, Jeremy (Embecosm); Dr BLACKMORE, Craig (Embecosm)

Session Classification: GNU Tools Track

Track Classification: GNU Tools Track
State of flow-based diagnostics in GCC

Thursday, 27 August 2020 09:00 (25 minutes)

GCC has a robust set of diagnostics based on control- and data-flow analysis. They are able to detect many kinds of bugs primarily related to invalid accesses. In this talk I will give an overview of the latest state of some of these diagnostics and sketch out my ideas for future enhancements in this area.

I agree to abide by the anti-harassment policy

I agree

Primary author:  Mr SEBOR, Martin (Red Hat)

Presenter:  Mr SEBOR, Martin (Red Hat)

Session Classification:  GNU Tools Track

Track Classification:  GNU Tools Track
The usage of PREEMPT_RT in safety-critical systems: what do we need to do?

Monday, 24 August 2020 09:25 (25 minutes)

This session shall shed some light on what needs to be done to use PREEMPT_RT in safety-critical systems.

For a structured discussion, this session first introduces:

- different types of assumed example systems, and the criticality of the real-time property in those systems,
- derived real-time requirements towards the kernel and hardware, and
- different general strategies described in safety standards to show real-time requirements to be met.

This short introduction then guides discussion among the audience through the various aspects and dimensions of the challenges and potential feasibility of addressing the question what needs to be done to use PREEMPT_RT in safety-critical systems.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr BULWAHN, Lukas
Presenter: Mr BULWAHN, Lukas
Session Classification: Real-time MC
Track Classification: Real-time MC
BoF: DWARF5/DWARF64

Monday, 24 August 2020 07:30 (25 minutes)

Can we switch to DWARF5 by default for GCC11? Which benefits does that bring? Which features work, which don’t (LTO/early-debug, Split-Dwarf, debug-types, debug_{[pub]}names, etc.). Which DWARF consumers support DWARF5 (which don’t) and which features can be enabled by default?

Additionally some larger applications are hitting the limits of 32bit offsets on some arches. Should we introduce a -fdwarf(32|64) switch, so users can generate DWARF32 or DWARF64? And/Or are there other ways to reduce the offset size limits that we should explore?

I’ll provide an overview and preliminary answers/patches for the above questions and we can discuss what the (new) defaults should be and which other DWARF5/DWARF64 questions/topics should be answered and/or worked on.

I agree to abide by the anti-harassment policy

I agree

Primary author:  WIELAARD, Mark
Presenter:  WIELAARD, Mark
Session Classification:  GNU Tools Track
Track Classification:  GNU Tools Track
Android Bootloader Consolidation

Quick overview of the negative costs of each SoC vendor having to update their bootloader to track Android boot flow requirements that change almost yearly, and what might be done to avoid this duplicative effort, that doesn’t bring much value to vendors.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  Mr PROTSENKO, Sam (Software Engineer); STULTZ, John (Linaro)
Presenters:  Mr PROTSENKO, Sam (Software Engineer); STULTZ, John (Linaro)
Session Classification:  Android MC
Track Classification:  Android MC
Improving SEPolicy Development Experience

Monday, 24 August 2020 10:15 (15 minutes)

Despite having capabilities to specify access control at a high level of granularity, SEPolicy is typically added in the development process as an afterthought; to accord the same permissions to a given set of processes that were developed with no regard to access restrictions. On Android - where SEPolicy operates in mandatory access control (MAC) mode - OEMs typically rely on tools such as audit2allow to help speed-up the development process and end up with scenarios where vendor and system applications are given more privileges than necessary for correctness. In these cases instead of utilizing SEPolicy to implement a security blueprint, rules are modified to pass restrictions. On Android, abuse due to vendors granting excessive permissions is prevented by neverallow checks and xTS requirements. However tests such as xTS are done at the end of the cycle versus at the beginning of vendor/OEM application design.

This talk focuses on the tools lacking for SEPolicy development, the approach with which such tools may be developed and shares our experience in developing tools to analyze and model SE-Policy.

I agree to abide by the anti-harassment policy

I agree

Primary author:  CHALLAKERE, Nagaravind (Microsoft)

Presenters:  CHALLAKERE, Nagaravind (Microsoft); CATTELL, Shaylin

Session Classification:  Android MC

Track Classification:  Android MC
Protected KVM: Memory protection of KVM guests in Android

Monday, 24 August 2020 10:30 (15 minutes)

This talk outlines a proposal to re-factor and extend the arm64/KVM implementation in order to enable the execution of guest VMs in memory carveouts protected from the host kernel, as well as potential use-cases in the Android world. Using this architecture, we intend to remove the host kernel from the Trusted Computing Base, hence protecting guest secrets, such as private user data, against attacks targeting the host.

I agree to abide by the anti-harassment policy

I agree

Presenter:  PERRET, Quentin (Google)

Session Classification:  Android MC

Track Classification:  Android MC
Integrating open source packages into the AOSP

Monday, 24 August 2020 11:00 (15 minutes)

Discussion on the difficulties with adding and maintaining open source projects in the Android build system. Why is it so complex? Could the Android build system be more open source friendly?

I agree to abide by the anti-harassment policy

I agree

Primary authors: PINCHART, Laurent (Ideas on Board Oy); YAGHMOUR, Karim (Opersys inc.); STULTZ, John (Linaro)

Presenters: PINCHART, Laurent (Ideas on Board Oy); YAGHMOUR, Karim (Opersys inc.); STULTZ, John (Linaro)

Session Classification: Android MC

Track Classification: Android MC
Multiple CI efforts to provide coverage of the Linux kernel are now building and providing results of builds with Clang (KernelCI, 0day bot, Linaro toolchain team and tuxbuild team, Clang Built Linux). Let’s all meet to discuss what’s working, what can be improved, the current status of builds of various architectures, and what the future direction of testing the various LLVM utilities might look like.

I agree to abide by the anti-harassment policy

I agree

Primary author: DESAULNIERS, Nick (Google)
Presenter: DESAULNIERS, Nick (Google)
Session Classification: LLVM MC
Track Classification: LLVM MC
The switch to an online event required a lot of scrambling by the Linux Plumbers Conference organizing committee. This is a session to talk about how we did it — what technologies were involved, where the challenges were, what is available to a group organizing a conference for nearly 1000 people using only free software. Come to talk about what we did, to learn about running an online event of your own, or just to ask questions about the whole process.

I agree to abide by the anti-harassment policy

I agree

Primary author:  CORBET, Jonathan (Linux Plumbers Conference)

Presenter:  CORBET, Jonathan (Linux Plumbers Conference)

Session Classification:  BOFs Session

Track Classification:  Birds of a Feather (BoF)
KUnit - One Year Later

Wednesday, 26 August 2020 08:40 (30 minutes)

Last year I presented a talk titled "KUnit - Unit Testing for the Linux Kernel" in which we presented the proposed KUnit unit testing framework. We discussed how it worked; why it was needed; and what we were planning on doing with it.

One year later, KUnit is now upstream and we have learned a lot. In this talk I intend to discuss what we have accomplished since our talk last year, what we learned, why things were different from what we expected, and what we are planning on doing going forward - most notably, new features - (and hopefully get some input from the audience).

Some specific topics we hope to cover include:

- Issues with our communication on how to use KUnit and what it is good for.
- Successes and failures collaborating with other kernel testing projects.
- Proposed features such as mocking and driver fuzzing.

I agree to abide by the anti-harassment policy

I agree

Primary author:  HIGGINS, Brendan (Google LLC)
Presenter:  HIGGINS, Brendan (Google LLC)
Session Classification:  Testing and Fuzzing MC
Track Classification:  Testing and Fuzzing MC
Lightning Talk: AMD GCN Update

Wednesday, 26 August 2020 08:15 (10 minutes)

A quick overview of the project status, roadmap, and a few interesting features of the port.

I agree to abide by the anti-harassment policy

I agree

Primary author:  STUBBS, Andrew (Mentor Graphics / CodeSourcery)
Presenter:  STUBBS, Andrew (Mentor Graphics / CodeSourcery)
Session Classification:  GNU Tools Track

Track Classification:  GNU Tools Track
BPF in the GNU toolchain and the Linux kernel

Friday, 28 August 2020 07:00 (45 minutes)

In 2019 Oracle contributed support for the eBPF (as of late renamed to just BPF) in-kernel virtual architecture to binutils and GCC. Since then we have continued working on the port, and recently sent a patch series upstream adding support for GDB and the GNU simulator.

After a brief description of the recent work done in this field, a set of points will be brought for discussion with the kernel hackers.

I agree to abide by the anti-harassment policy

I agree

Primary author: MARCHESI, Jose E. (GNU Project, Oracle Inc.)
Presenter: MARCHESI, Jose E. (GNU Project, Oracle Inc.)
Session Classification: GNU Toolchain MC
Track Classification: GNU Toolchain MC
To complete the driverless support for IPP network multi-function devices there is also IPP Fax Out, the standard for sending faxes, as print jobs, through the fax functionality of the device.

The fax support is provided by an additional printing channel with its own URI (ending with “/ipp/faxout” instead of “/ipp/print”) and printing to this channel makes the document being faxed.

It naturally requires supplying the phone number as an IPP attribute, but otherwise it is exactly like printing, if polling this URI for capabilities you get the fax-specific “printer” capabilities and options, to be used for fax jobs.

Current devices have this functionality ready available and we will show how we make it available for desktop applications and discuss possible alternatives.

I agree to abide by the anti-harassment policy

I agree

Primary author:  BASU, Aveek
Presenter:  BASU, Aveek
Session Classification:  Open Printing MC
Track Classification:  Open Printing MC
BoF: OpenMP, OpenACC & Offloading

Thursday, 27 August 2020 07:00 (25 minutes)

BoF to discuss topics related to concurrency and offloading work onto accelerators. On the OpenMP side, in particular the implementation of the missing OpenMP 5.0 (soon: 5.1) features.

Especially for offloading with OpenACC/OpenMP, optimizing the performance and in particular restricting the amount and frequency of data transfers is crucial and involves topics like value propagations, cloning, loop parallelizations, and memory management - including pinning, asynchronous operations and unified memory. And with offloading code and GPU offloading becoming ubiquitous, deployment and keeping pace with supporting consumer and high-end hardware updates is a challenge.

Related topics and trends can also be discussed, be it base language concurrency features, offloading without using OpenMP/OpenACC, other accelerators.

I agree to abide by the anti-harassment policy

I agree

Primary authors: BURNUS, Tobias (Mentor, A Siemens Business); JELINEK, Jakub (Red Hat)

Presenters: BURNUS, Tobias (Mentor, A Siemens Business); JELINEK, Jakub (Red Hat)

Session Classification: GNU Tools Track

Track Classification: GNU Tools Track
Last year we introduced support for the Compact C Type Format (CTF) into the GNU toolchain. We have since improved the linking of CTF so that types are properly deduplicated: the work is done by libctf on ld’s behalf so that other programs can do what ld does. With the aid of a few dozen lines of makefile changes and a 300-odd line program using libctf, we can now produce a fully deduplicated description of all types in the kernel, with types specific to single modules localized appropriately. A recent kernel (with a 3000-module enterprise configuration) comes to about 7MiB of types, after compression, of which about half is core-kernel stuff and the rest are types only used by single modules (that users can often avoid loading).

We plan to do more changes to improve CTF in ways the kernel team might find useful (representing static functions’ types is planned, as well as further space reductions), but I don’t want to make this up entirely on my own, so I thought I should ask what people need.

One obviously essential piece not present yet is turning CTF into BTF in the first place. Directly translating CTF into BTF and vice versa as I proposed last year is possible, but BTF is such a moving target that I fear we might have trouble keeping up (we can hardly release binutils as often as the kernel is released, and nobody upgrades the two in sync anyway).

But bidirectional conversion straight from CTF<->BTF might not actually be necessary for the kernel to exploit CTF: emitting C source code corresponding to CTF is definitely possible, and this might be just as useful: this is already doable with BTF, of course, so this might serve as a bidirectional gateway that requires less chasing. At the very least going from CTF -> BTF rather than from DWARF -> BTF would speed up compiles and make them take much less disk space (a recent test using an enterprise kernel showed a space saving by generating CTF instead of DWARF of around ten gigabytes). But there could be other advantages, too (among other things CTF is much easier to change than DWARF at present).

Does anyone have any other ideas of things I might do to make your lives easier? A CTF file format bump is happening in the near future, so now is the time to propose new stuff. I want to take some of the burden of the more boring parts of BTF off you and drop it into binutils where you can forget about it, if possible.

I agree to abide by the anti-harassment policy

I agree

Primary author: ALCOCK, Nick (Oracle Corporation)
Presenter: ALCOCK, Nick (Oracle Corporation)
Session Classification: GNU Toolchain MC
Track Classification: GNU Toolchain MC
The implementation of C++ modules in GCC and other compilers may pose some constraints on the kind of preprocessor and language constructs glibc headers can use (and the kernel headers they require). With this BoF, we hope to coordinate this a bit between GCC and glibc, so that we do not have to put hacks into the compiler or rely on the fixincludes mechanism (which is incompatible with glibc updates).

I agree to abide by the anti-harassment policy
I agree

Primary authors: WEIMER, Florian (redhat); SIDWELL, Nathan
Presenters: WEIMER, Florian (redhat); SIDWELL, Nathan
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Following the Linux Kernel Defence Map

*Tuesday, 25 August 2020 09:20 (30 minutes)*

Linux kernel security is a very complex topic. To learn it, I created a [Linux Kernel Defence Map](#) showing the relationships between:

- Vulnerability classes
- Exploitation techniques
- Bug detection mechanisms
- Defence technologies

These kernel defence technologies have the corresponding Kconfig options. A lot of them are not enabled by the major Linux distributions.

So I created a [kconfig-hardened-check](#) tool that can help to examine security-related options in your Linux kernel config.

In this short talk we will follow the [Linux Kernel Defence Map](#) and explore the [kconfig-hardened-check](#) tool.

I agree to abide by the anti-harassment policy

I agree

**Primary author:** POPOV, Alexander

**Presenter:** POPOV, Alexander

**Session Classification:** Kernel Dependability & Assurance MC

**Track Classification:** Kernel Dependability & Assurance MC
“Asm goto with outputs” is a clang extension of the GNU “asm goto” feature. As the name implies, it allows asm goto to have outputs on the default branch (outputs on indirect branches aren’t supported). In this talk, we discuss the benefits of this feature, its implementation and design limits, and how the clang and gcc communities can work together on future GNU C extensions.

I agree to abide by the anti-harassment policy

I agree

Primary author:  WENDLING, Bill (Google)
Presenter:  WENDLING, Bill (Google)
Session Classification:  LLVM MC
Track Classification:  LLVM MC
Security Features Update and Comparison

Friday, 28 August 2020 08:40 (45 minutes)

Compare the status of GCC and Clang security features, and provide a time to discuss the progress on current work (e.g. auto-variable-initialization, caller-saved register clearing). More work is needed on sanitizers (e.g. bounds checking, arithmetic overflow handling) and Control Flow Integrity.

I agree to abide by the anti-harassment policy

I agree

Primary author: COOK, Kees (Google)
Presenter: COOK, Kees (Google)
Session Classification: GNU Toolchain MC
Track Classification: GNU Toolchain MC
ld.so in the 2020’s

Tuesday, 25 August 2020 09:00 (25 minutes)

Since dynamic libraries have become universal, the runtime linker loader has been a critical but often times overlooked component of the OS. The general design and many implementation details were solidified back in the 1990’s and addressed issues that were facing OS designers and software developers back then. The computing environment is quite different in the second decade of the 21st century and the demands on the runtime linker loader are now quite different. This talk uses case studies drawn from nearly 20 years of experience working at Red Hat supporting the HPC community to illustrate some of the current challenges facing this often times overlooked but critical piece of technology.

I agree to abide by the anti-harassment policy

I agree

Primary author: WOODARD, Ben (Red Hat Inc)
Presenter: WOODARD, Ben (Red Hat Inc)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
BoF - Git Users & Contributors

Git is a fundamental tool in the open source ecosystem - a widely used source control system, it’s relied on by many packages which Linux relies on, and by the Linux kernel itself. However, it has its share of obstacles for both users and contributors, in the form of steep learning curves, exclusive language, and more. To help address these obstacles, the Git project wants to hear from users! Core contributors to Git will chat with BoF attendees to discover what’s important to different types of users, as well as to communicate new Git features which attendees may find useful.

An incomplete list of topics to discuss:

- How-to/Q&A - ask Git contributors for tips and solutions
- Concerns about Git’s inclusiveness - in what ways does Git exclude you?
- Concerns about Git’s usability - what blocks you and your colleagues from getting the most out of Git?
- Code lifecycle - how can Git better support your continuous integration/continuous deployment and code review workflows?
- Feature requests - what would you like to see Git learn to do in the future?
- Project contribution - how can you start contributing to Git?

I agree to abide by the anti-harassment policy

I agree

Primary authors: SHAFFER, Emily (Google); NIEDER, Jonathan (Google); TAN, Jonathan (Google); STEADMON, Josh (Google)

Presenters: SHAFFER, Emily (Google); NIEDER, Jonathan (Google); TAN, Jonathan (Google); STEADMON, Josh (Google)

Session Classification: BOFs Session

Track Classification: Birds of a Feather (BoF)
Avoiding Security Flaws

Tuesday, 25 August 2020 10:30 (30 minutes)

At the end of the day, “security flaws” are just a special case of “regular” bugs, so anything that helps avoid bugs will also help with reducing the incidence of security flaws. This explores the approaches taken to avoiding bugs generally and security flaws in particular.

Find and fix bugs before they are released. This is fundamentally a matter of testing. Whether that’s done via unit testing, functional testing, regression testing, or fuzzing, there are a few basic dependencies:
- code coverage (how do you know which code got tested?)
- deterministic failure (hard to fix a bug if it can’t be reproduced)
- disable randomization during debugging
- always initialize memory allocation contents

Limit userspace behaviors to avoid hitting bugs (if you can’t reach a bug, you can’t trip over it), mainly via attack surface reduction:
- DAC (everyone understands uids, and file permissions)
- MAC (LSMs: SELinux, AppArmor, etc)
- seccomp (syscall limitations)
- Yama (ptrace limitations)

And most importantly, generalize any work done to fix bugs. Instead of fixing the same kind of bug over and over, focus on removing entire classes of bugs.
- redesign APIs that were easy to misuse (avoid shooting yourself in the foot)
- remove features that only causes problems (e.g. %n in format strings)
- create detection systems that catch a bug before it happens (e.g. saturate reference counters)

I agree to abide by the anti-harassment policy

I agree

Primary author: COOK, Kees (Google)
Presenter: COOK, Kees (Google)
Session Classification: Kernel Dependability & Assurance MC
Track Classification: Kernel Dependability & Assurance MC
Compile times with Clang

Thursday, 27 August 2020 08:50 (20 minutes)

In this talk we will discuss clang-built kernel compile times, current work to improve compiler performance and recommendations to reduce build times regardless of toolchain.

We will present our findings alongside several metrics of compiler performance, including:

- Comparative timing breakdowns between toolchains
- Linux perf profiling on clang builds of the kernel
- Perfetto traces on clang builds of the kernel

I agree to abide by the anti-harassment policy

I agree

Primary authors: HUCKLEBERRY, Nathan; CHANCELLOR, Nathan

Presenters: HUCKLEBERRY, Nathan; CHANCELLOR, Nathan

Session Classification: LLVM MC

Track Classification: LLVM MC
Clang-tidy and Clang-format

Thursday, 27 August 2020 09:10 (20 minutes)

Clang is a production C compiler (part of LLVM) that provides APIs for C code parsing, formatting, custom compiler warnings, static analysis, etc. This framework has spawned widely used tools like clang-format and clang-tidy. These tools can be easily tailored for particular codebases like the Linux kernel.

This talk shows how to run clang-format, clang-tidy (including writing custom checks), and scan-build to help everyday Linux kernel development, using the kernel support we landed.

Furthermore, we will seek feedback on how we can incorporate these tools into wider kernel dev/CI workflows, as well as what kinds of static analyses we should seek to develop in the future.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  HUCKLEBERRY, Nathan; OJEDA, Miguel
Presenters:  HUCKLEBERRY, Nathan; OJEDA, Miguel
Session Classification:  LLVM MC
Track Classification:  LLVM MC
Exploring Profile Guided Optimization of the Linux Kernel

Wednesday, 26 August 2020 10:00 (25 minutes)

Exploring Profile Guided Optimization of Linux Kernel

Author

ian Bearman is the former team lead supporting GCC and GNU developer tools for Linux at Microsoft. Nearly 20 years of experience in code generation, optimization, and developer tools.

Abstract

The Gnu/Linux Tools Team at Microsoft spent some time this year looking at using profile guided optimization in GCC to optimize the Linux kernel. As part of this plan we looked into enabling Link Time Optimization as well. Though we were only able to demonstrate small wins, I would like to share our experience through this process and share experience with LTO and PGO on other non-Linux operating systems.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr BEARMAN, ian (Microsoft)
Presenter: Mr BEARMAN, ian (Microsoft)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Lightning Talk: Linking LTO and Make

Tuesday, 25 August 2020 08:45 (10 minutes)

A brief status update on John’s progress regarding his GOSC project to parallelizing LTO during the build phase using Make.

I agree to abide by the anti-harassment policy

I agree

Primary author:  RAVI, John (North Carolina State University)

Presenter:  RAVI, John (North Carolina State University)

Session Classification:  GNU Tools Track

Track Classification:  GNU Tools Track
Barriers to in-tree Rust

Thursday, 27 August 2020 07:35 (30 minutes)

What would it take to have in-tree support for writing kernel code in Rust? What should Kbuild integration look like? What APIs should be the initial priorities to expose in Rust? Let's figure out if any other other questions remain (e.g., can we safely link against GCC-built kernels, and do we need to) about how to get in-tree support for Rust.

Rust is a systems programming language that is particularly well-suited to the kernel: it is a "better C" in a way that matches the kernel's needs (no GC, kernel-style OO, etc.) Rust can also be of significant benefit for security - safe Rust protects against entire classes of vulnerabilities such as use-after-frees, buffer overflows, and use of uninitialized memory, which form a large percent of kernel vulnerabilities.

(This session will not be an intro to the Rust language. See last year's Linux Security Summit NA talk "Linux Kernel Modules in Rust" video / slides for an overview of Rust for kernel hackers and a demo of Rust modules.)

I agree to abide by the anti-harassment policy

I agree

Primary authors: BAUBLITZ, John; DESAULNIERS, Nick (Google); GAYNOR, Alex; THOMAS, Geoffrey; TRIPLETT, Josh; OJEDA, Miguel

Presenters: BAUBLITZ, John; DESAULNIERS, Nick (Google); GAYNOR, Alex; THOMAS, Geoffrey; TRIPLETT, Josh; OJEDA, Miguel

Session Classification: LLVM MC

Track Classification: LLVM MC
Performance improvements in power-sharing scenarios

Friday, 28 August 2020 09:20 (25 minutes)

There are use cases in which the processor shares power budget with some other data-processing devices, like a GPU. In those cases it may be possible to improve the performance of the system by limiting the maximum frequency of CPUs. We will discuss possible ways to utilize this observation in the Linux kernel.

See https://lore.kernel.org/linux-pm/20200428032258.2518-1-currojerez@riseup.net/ for one possible approach to this problem.

I agree to abide by the anti-harassment policy

I agree

Primary author: WYSOCKI, Rafael (Intel Open Source Technology Center)
Co-author: JEREZ, Francisco
Presenters: WYSOCKI, Rafael (Intel Open Source Technology Center); JEREZ, Francisco
Session Classification: Power Management and Thermal Control MC
Track Classification: Power Management and Thermal Control MC
Over time computers get more and more complicated and there are more and more dependencies between devices in them which affect power management. We will discuss issues arising from that and possible ways to address them.

See https://lore.kernel.org/linux-pm/20200624103247.7115-1-daniel.baluta@oss.nxp.com/T/#mbe0060ea9b225073d63ae3ff8b for a patch series submission related to that problem space.

I agree to abide by the anti-harassment policy

I agree

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

**Co-author:** BALUTA, Daniel (University POLITEHNICA of Bucharest)

**Presenters:** WYSOCKI, Rafael (Intel Open Source Technology Center); BALUTA, Daniel (University POLITEHNICA of Bucharest)

**Session Classification:** Power Management and Thermal Control MC

**Track Classification:** Power Management and Thermal Control MC
Functioning temperature range - Warming devices

The thermal framework is only designed to detect and handle hotspot, not coldspot. Some systems need to increase their performance state or leak power to warm some devices which are getting too cold (outdoor devices when night comes). The logic is the mirror of managing hot spots.

I agree to abide by the anti-harassment policy

I agree

Primary author: GOPINATH, Thara (Linaro Inc)
Co-author: LEZCANO, Daniel (Linaro)
Presenters: GOPINATH, Thara (Linaro Inc); LEZCANO, Daniel (Linaro)
Session Classification: Power Management and Thermal Control MC
Track Classification: Power Management and Thermal Control MC
An ever-increasing number of embedded devices need fine grain control on their performance in order to limit the power consumption. There are three primary reasons for this: to increase the battery life, to protect the components and to control the temperature.

Due to the increasing complexity of SoCs, we’re now seeing lots of thermal sensors on the die to quickly detect hot spots and allow the OS to take steps to mitigate these events - either through better scheduling, frequency throttling, idle injection or other similar techniques.

Mobile devices are even more interested in managing power consumption because, depending upon the situation or the workload, the performance places higher or lower priority on certain components in regards to others. One example is virtual reality where a hotspot on the graphics can lead to a performance throttling on the GPU resulting in frame drops and a dizziness feeling for the user. Another example is the ratio between the cost in energy for a specific performance state vs a benefit not noticeable for the user, like saving milliseconds when rendering a web page. And last but not least, a battery low situation where we want to guarantee a longer duration before shutdown can create a unique prioritization scheme.

This non-exhaustive list of examples shows there is a need to act dynamically on the devices’ power from the userspace who has full knowledge of the running application. In order to catch unique scenarios and tune the system at runtime, the solution today leverages a thermal daemon monitoring the temperature of different devices and trying to anticipate where to reduce the power consumption, given the application is running. The thermal daemon turns the different “knobs” here and there, in every place where it is possible to act on the power. One of these places is the thermal framework which exports an API via sysfs to manually set the level of the performance state for a given device declared as a passive cooling device.

The powercap provides all the infrastructure to export the power consumption and set the power limit. The combination of the energy model and the powercap framework will offer an unified access to the power management of the different devices.

I agree to abide by the anti-harassment policy

I agree

Primary author: LEZCANO, Daniel (Linaro)
Presenter: LEZCANO, Daniel (Linaro)
Session Classification: Power Management and Thermal Control MC
Track Classification: Power Management and Thermal Control MC
Negotiating DMA-BUF Heaps

Thursday, 27 August 2020 08:00 (45 minutes)

With the introduction of DMA-BUF Heaps, the kernel has introduced a fairly generic API for applications and drivers to request memory that can be used for DMA operations.

Currently, two DMA-BUF Heaps backends (system and CMA) are available and a bunch of others are being explored and proposed for mainline inclusion.

However, the current design seems to imply applications know beforehand which heap is suitable for its needs. While this might play well for system-specific applications, it doesn’t offer a generic solution for generic, system-agnostic applications.

The goal of this BoF is to discuss what are the expectations for a DMA-BUF Heap generic negotiation interface that can be used by in-kernel and applications consumers.

In addition to this, we’d like to discuss the future of DMA-BUF heaps, are they meant to be used by current allocators, such as GEM/TTM and Videobuf2?

I agree to abide by the anti-harassment policy

I agree

Primary author: GARCIA, Ezequiel (Collabora, Ltd.)
Presenter: GARCIA, Ezequiel (Collabora, Ltd.)
Session Classification: BOFs Session
Track Classification: Birds of a Feather (BoF)
A Look Inside Mutter / GNOME Shell

Friday, 28 August 2020 07:45 (45 minutes)

Mutter is a Wayland compositor and X11 compositing window manager based on the Clutter toolkit. GNOME Shell is GNOME’s signature desktop, and is built on top of Mutter.

In this presentation, I’ll start with a quick overview of various aspects of Mutter internals, such as:

- The different abstractions layers for rendering the scene graph (Clutter, Cogl, Graphene)
- GBM/EGL native renderer
- Ongoing transition to atomic KMS
- Usage of hardware planes, and the challenges of assigning planes
- Nesting compositors with an X11 / Wayland hybrid
- DMA-BUF based screencasts with PipeWire

After that, I’ll cover ongoing changes, as well as future plans. Some of these topics are:

- Experimenting with libliftoff
- The path to atomic modesetting
- Plane assignment
- Support for DMA-BUF modifiers in PipeWire
- Usage of modern OpenGL features in Cogl (UBO)
- Vulkan API on a compositor

Ideally, we will be able to create a proof-of-concept branch of Mutter using libliftoff; a proof-of-concept branch of PipeWire with better DMA-BUF support; and understand what’s missing / what’s feasible to implement Vulkan-based rendering.

I agree to abide by the anti-harassment policy

I agree

Primary author:  STAVRACAS NETO, Georges Basile (Endless OS Foundation)
Presenter:  STAVRACAS NETO, Georges Basile (Endless OS Foundation)
Session Classification:  Application Ecosystem MC
Track Classification:  Application Ecosystem MC
Print-Scan-Fax in Linux.

Friday, 28 August 2020 07:00 (10 minutes)

This session is all about Print-Scan-Fax in Linux where we stand as of date. This is to discuss on the problem areas and what we look ahead in the future.

I agree to abide by the anti-harassment policy

I agree

Primary author: BASU, Aveek
Presenter: BASU, Aveek
Session Classification: Open Printing MC
Track Classification: Open Printing MC
The KernelCI project has been increasingly in the spotlight since it joined the Linux Foundation in October 2019. In addition to having a strong set of founding members, it has also started growing a healthy ecosystem. While still relatively small in size compared to the object under test that is the Linux kernel, as a relatively young project it is showing some very positive signs. Its roots are getting stronger, and it looks like it will keep bearing more fruit every year.

Extending its scope to collate kernel test results from other systems such as 0-Day and Syzbot, getting a bigger compute capacity thanks to cloud resources donated by Microsoft and Google, ramping up functional testing capabilities across the board, supporting KUnit developers to integrate it in the KernelCI framework and getting more and more diverse contributors are all strong examples.

By continuing this trend, KernelCI will also keep increasing its impact on the Linux kernel code quality and development workflows. Ultimately, it will need to be owned by the kernel community in order to truly succeed. Now is the time to engage more with maintainers, developers and many others to make it all happen in a collective effort.

I agree to abide by the anti-harassment policy

I agree

Primary author:  TUCKER, Guillaume (Collabora)
Presenter:  TUCKER, Guillaume (Collabora)
Session Classification:  Testing and Fuzzing MC
Track Classification:  Testing and Fuzzing MC
Renode - a flexible simulator for CI in complex embedded systems

Thursday, 27 August 2020 08:40 (40 minutes)

Renode is an instruction set simulator with a flexible platform definition language and plug-and-play SoC component library that can be used to compose virtual hardware setups. It allows users to simulate complex systems, including multi-node wired and wireless networked systems, offering automated testing and rich debugging capabilities. It includes support of numerous development boards, SoCs, CPUs and peripherals, as well as provides a number of other features such as Verilator co-simulation, state saving and loading, event hooks, performance metrics and detailed logs, allowing the user to perform architecture exploration as well as prototyping, development and testing of complex systems.

Renode enables development with and around Linux through its support of various architectures and configurations, such as RISC-V, Arm and the recently added POWER ISA. RISC-V, with the weight of the open hardware movement behind it, is actively supported in Renode which offers demos and definitions for a variety of platforms, including Linux capable ones like Kendryte, LiteX/VexRiscv, HiFive Unleashed and PolarFire SoC. The recently released Renode 1.10 comes with support for the RISC-V flagship PolarFire SoC Icicle Kit - the first mass-produced Linux-capable RISC-V implementation. We are going to show how you can run an unmodified Yocto-based Linux BSP on top of a virtual Icicle board, even if you don’t have access to a real one yet.

I agree to abide by the anti-harassment policy

I agree

Primary author: GIELDA, Michael (Antmicro)

Session Classification: You, Me, and IoT Two MC

Track Classification: You, Me, and IoT Two MC
Assessing kernel system call correctness by testing

Tuesday, 25 August 2020 08:30 (30 minutes)

Key question: Can system calls be regarded as independent and consequently tested individually rather than in some form of use-case specific call sequence?

The kernel has a set of asynchronously operated state machines, e.g., RCU, buddy-system, rate-limits of all sorts, that cause a repeated identical system call to take different paths in consecutive invocations. The model thus is that the result of a system call is effected by two aspects:

1. the formal input, i.e., parameters to the system call, and
2. the kernel’s global system space.

As the global system state space is modified by all active processes, the "global system state” input is uncontrolled (and assumed to be uncontrollable) whereas the formal input, i.e., the arguments pass to system_call_X(), is assumed to be held constant.

In that case, the assumed path variability is assumed to be causally related to the code being conditioned in part on the global system state. To now judge the correctness of the system_call_X() implementation, the repeated tests need to be conducted while allowing the system state space to freely roam around.

In practical terms, if we have two processes, i.e., process A calling fd = open(...); ret = read(fd,...), and process B, calling other system calls, X, Y, Z, etc., be it on the same or different cores, do we expect the execution path of the read() to causally depend on the order or unrelated calls concurrently being executed on the system?

This is relevant for dependability as:
If calls may be treated as independent, then assessment of correctness can be done by repeated testing of individual calls while exercising some background load of arbitrary type. If this assumption is invalid due to the design of the kernel, then assessment of correctness is only possible by testing permutations of call sets.

We would like to discuss: What arguments would you see in favor of “calls are independent” or to bolster the claim of “calls are non-independent”?

I agree to abide by the anti-harassment policy

I agree

Primary authors: Dr PETERSOHN, Jens (Elektrobit Automotive GmbH); Prof. MC GUIRE, Nicholas (OpenTech)

Presenters: Dr PETERSOHN, Jens (Elektrobit Automotive GmbH); Prof. MC GUIRE, Nicholas (OpenTech)

Session Classification: Kernel Dependability & Assurance MC

Track Classification: Kernel Dependability & Assurance MC
Generic functionality for system call and trap entry and exit

Tuesday, 25 August 2020 09:00 (25 minutes)

The system call entry and exit code is needlessly duplicated and different in all architectures. The work carried after the real low level ASM bits should not be different across architectures as well as the code that handles the pending work before returning from a system call to user space. Likewise, the interrupt and exception handling has to establish the state for various kernel subsystems like lockdep, RCU and tracing and there is no good reason to have twenty-some similar and pointlessly different implementations.

A common infrastructure for kernel entry handling was merged in v5.9 release cycle and for now it is only used by x86.

Let’s discuss how this infrastructure is adopted by other architectures.

I agree to abide by the anti-harassment policy

I agree

Primary author: GLEIXNER, Thomas
Presenter: GLEIXNER, Thomas
Session Classification: linux/arch/*/ MC
Track Classification: linux/arch/*/ MC
Using Linux, Zephyr, & Greybus for IoT

Thursday, 27 August 2020 10:15 (40 minutes)

We provide a gentle introduction to Greybus, its integration into the Zephyr RTOS, and how Linux uses the Greybus application layer protocol to control peripherals attached to wireless micros. There are a lot of technologies at play, so it’s important to give some attention to each. Details of the software architecture will be provided, as well as a guide to help developers wire up and speak Greybus with their own sensors and boards.

The second half of the talk will involve some demonstrations on readily available dev kits such as the nRF52840 from Nordic Semi and the CC13552R SensorTag from Texas Instruments. The configuration and build process will be shown and hopefully we will highlight some of Zephyr’s many features along the way. Demos will use the IEEE 802.15.4 and BLE physical layers (both of which use 6LowPAN and IPv6 in layers 2 and 3). We will use Greybus to toggle some GPIO and to read data from I2C sensors.

Lastly, we will list the open problems on the roadmap to completion. Work needs to be done within the Linux kernel, within the Zephyr ecosystem, within the Zephyr kernel, as well as in the Linux userspace. Some of the open problems include

- Authentication and Encryption in Greybus
- Automatic Joining and Rejoining of devices
- Additional Device Tree Bindings Greybus in Zephyr

I agree to abide by the anti-harassment policy

I agree

Primary author:  FRIEDT, Christopher (Friedt Professional Engineering Services)

Session Classification:  You, Me, and IoT Two MC

Track Classification:  You, Me, and IoT Two MC
Introduction to Kernel Dependability & Assurance MC

Tuesday, 25 August 2020 07:00 (10 minutes)

We will shortly describe the overall topic of Kernel Dependability & Assurance MC and where we see how the topics in the MC agenda fit to this larger table. If there is a bit of time, we can align among speakers and the audience this common understanding on the large scope of the two terms, dependability and assurance.

Primary authors: STEWART, Kate (Linux Foundation); BULWAHN, Lukas (BMW AG); KHAN, Shuah (The Linux Foundation)

Presenters: STEWART, Kate (Linux Foundation); BULWAHN, Lukas (BMW AG); KHAN, Shuah (The Linux Foundation)

Session Classification: Kernel Dependability & Assurance MC

Track Classification: Kernel Dependability & Assurance MC
How to measure kernel testing success.

Wednesday, 26 August 2020 10:30 (30 minutes)

Over the years, more services are contributing to the testing of kernel patches and git trees. These services include Intel’s 0-day, Google’s Syzkaller, KernelCI and Red Hat’s CKI. Combined with all the manual testing done by users, the linux kernel should be rock solid! But it isn’t.

Every service and tester is committed to stabilizing the linux kernel, but there is duplication and redundant testing that makes the testing effort inefficient.

How do we know new tests are filling in the kernel gaps? How do we know each service isn’t running the same test on the same hardware? How do we measure this work towards the goal of stabilizing the linux kernel?

Is functional testing good enough?  
Is fuzzing good enough?  
Is code coverage good enough?  
How to incorporate workload testing?  
How to leverage the unified kernel testing data (kcidb)?

This talk is an open discussion about those problems and how to address them. I encourage maintainers to bring ideas on how to qualify their subsystem as stable.

By the end of the talk, a core set of measurables should be defined and trackable on kernelci.org with clear gaps that testers can help fill in.

I agree to abide by the anti-harassment policy

I agree

Primary author:  ZICKUS, Don (Red Hat)

Presenter:  ZICKUS, Don (Red Hat)

Session Classification:  Testing and Fuzzing MC

Track Classification:  Testing and Fuzzing MC
Using the Thread Networking Protocol for IoT Applications with embedded Linux

Thursday, 27 August 2020 07:45 (40 minutes)

The IoT landscape has many competing protocols and technologies for enabling communication between sensor End Nodes, Embedded Linux Edge devices, and ultimately cloud resources. One such technology is the Thread Network Protocol, an IPv6 based, Meshing, 802.15.4 protocol that allows for on and off mesh device-to-device, and device-to-cloud communication.

This talk aims to give a brief introduction to Thread, the advantages to using Thread instead of generic Linux IEEE 802.15.4 WPAN, and identify the challenges encountered while bringing up a Thread Border Router using Buildroot.

We will use the freely available OpenThread project released by Google, and show the use of standard mechanisms (DHCP, DNS, UDP and CoAP) to allow for Thread End Nodes to discovery our Thread Border Router on the Mesh Network, and server resources on the off mesh local network.

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr MAGYAR, Michael (MMB Networks)

Session Classification: You, Me, and IoT Two MC

Track Classification: You, Me, and IoT Two MC
GKI compatibility in Android R, how did it go?

Monday, 24 August 2020 07:05 (15 minutes)

Summary of how GKI efforts for Android R went and what are the next steps.

I agree to abide by the anti-harassment policy

I agree

Primary authors:  KJOS, Todd (Google);  DELVA, Alistair (Google);  MUCKLE, Steve

Presenters:  KJOS, Todd (Google);  DELVA, Alistair (Google);  MUCKLE, Steve

Session Classification:  Android MC

Track Classification:  Android MC
Ecosystem experience with GKI & v2

Monday, 24 August 2020 07:20 (15 minutes)

Short panel discussion covering different vendors experience with GKI and planned next steps.

I agree to abide by the anti-harassment policy

I agree

Primary authors: IYER, Lina; STULTZ, John (Linaro); ZHANG, Pete

Presenters: IYER, Lina; STULTZ, John (Linaro); ZHANG, Pete

Session Classification: Android MC

Track Classification: Android MC
Update on GKI KMI enforcement tools

Monday, 24 August 2020 07:35 (15 minutes)

Covering ABI monitoring, what has happened with libabigail in the past year and what remains to be done

I agree to abide by the anti-harassment policy

I agree

Primary author: MÄNNICH, Matthias (Google)
Presenter: MÄNNICH, Matthias (Google)
Session Classification: Android MC
Track Classification: Android MC
Upstreaming debt from GKI work

Monday, 24 August 2020 07:50 (15 minutes)

Overview and discussion on upstreaming efforts connected with GKI work.

I agree to abide by the anti-harassment policy

I agree

Primary author:  SEMWAL, Sumit
Presenter:  SEMWAL, Sumit
Session Classification:  Android MC
Track Classification:  Android MC
Intro

Monday, 24 August 2020 07:00 (5 minutes)

**Presenter:** YAGHMOUR, Karim (Opersys inc.)

**Session Classification:** Android MC

**Track Classification:** Android MC
Introduction

Wednesday, 26 August 2020 07:00 (5 minutes)

I agree to abide by the anti-harassment policy

Presenter: DABBELT, Palmer (Google)
Session Classification: RISC-V MC
Track Classification: RISC-V MC
Suspend/Resume Quality, and Performance

sleepgraph is an open source tool in the pm-graph project: https://01.org/pm-graph

sleepgraph has helped us improve both Linux suspend/resume quality and performance over the last few years.

In this session we will review the capabilities of the tool, so that you will be able to run it and understand its results. We will also highlight some of the areas where it shows we can improve Linux.

I agree to abide by the anti-harassment policy

I agree

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
Track Classification:  Power Management and Thermal Control MC
Plasma on Mobile devices

Friday, 28 August 2020 08:30 (45 minutes)

KDE, previously known as the one of the Desktop Environment evolved into one of the largest Free and Open-source software community at one point. Currently one of the projects supported by community is Plasma Mobile: open-source user interface and ecosystem running on top of the Linux distribution.

This talk aims to talk about journey of the Plasma Mobile, how it evolved into what it is today, and it’s future.

- Initial development of the Plasma Mobile
- Basic architecture details
- Advantages to KDE community
- Application ecosystem and development
- Future for Plasma Mobile

I agree to abide by the anti-harassment policy
I agree

Primary author: SHAH, Bhushan
Presenter: SHAH, Bhushan
Session Classification: Application Ecosystem MC
Track Classification: Application Ecosystem MC
For better or worse, the Linux kernel relies heavily on hardware ordering guarantees concerning dependencies between memory access instructions as a way to provide efficient, portable implementations of concurrent algorithms. In spite of the lack of C language support, preserving source-level dependencies through to the generated CPU instructions is achieved through a delicate balance of volatile casts, magic compiler flags and sheer luck.

Wouldn’t it be nice if we could do better?

This talk will briefly introduce the problem space (and aim to define some basic terminology to avoid people talking past each other) before opening up to discussion. Some questions to start us off:

- What does Linux currently rely on?
- How can we enforce dependencies at the source level?
- How can we detect broken dependencies and/or insert memory barriers?
- Are annotations a non-starter?
- Does LTO make things worse and why?
- Just how expensive are memory barriers?
- Can we strike a balance between "optimising compiler" and "portable assembler"?

I agree to abide by the anti-harassment policy

I agree

Primary authors: DEACON, Will; DEACON, Will; ZIJLSTRA, Peter (Intel OTC); MCKENNEY, Paul (Facebook)

Presenters: DEACON, Will; ZIJLSTRA, Peter (Intel OTC); MCKENNEY, Paul (Facebook)

Session Classification: LLVM MC

Track Classification: LLVM MC
Welcome

Monday, 24 August 2020 07:00 (5 minutes)

Welcome to the Real-time Microconference.

Session Classification: Real-time MC

Track Classification: Real-time MC
Enhancements to IOMMU and VFIO User APIs for guest SVA

Wednesday, 26 August 2020 07:20 (20 minutes)

IOMMU UAPIs was partially merged to support basic guest Shared Virtual Address (SVA) functionalities such as cache invalidation, bind guest page tables, and page request service. These initial patches defined UAPI data structures without the transport mechanics and specifics for future extensions.

To bridge these gaps, new patchsets are being developed by Yi L Liu and Jacob Pan to address the following:

1. Define the roles between IOMMU and VFIO UAPI, allow IOMMU core to directly handle user pointers
2. Add sanity checking of UAPI data based on argsz, flags
3. Added a new UAPI for reporting domain nesting info
4. Document UAPI design and provide examples of interactions with VFIO

- In separate patchset, more to follow suit.

Currently, at its version 7 with many design choices reviewed and suggested by Alex Williamson, Eric Auger, and Christoph Hellwig, Yi and Jacob are trying to close on the patchset At LPC 2020.

I agree to abide by the anti-harassment policy

I agree

Primary authors: PAN, Jacob; Mr LIU, Yi

Presenters: PAN, Jacob; Mr LIU, Yi

Session Classification: VFIO/IOMMU/PCI MC

Track Classification: VFIO/IOMMU/PCI MC
IOASID API extensions for Intel Scalable IOV usages

Wednesday, 26 August 2020 07:40 (20 minutes)

As it currently stands in the mainline kernel, IOASID is a generic kernel service that provides PCIe PASID or ARM SMMU sub-stream ID allocations. On VT-d and Intel's Scalable IO Virtualization(SIOV) platforms, IOASID core serves a particularly important role as its usage spans the following dimensions:
- bare metal and guest SVM
- A slew of in-kernel users consists of VFIO, IOMMU, mm, VDCM*, KVM

To fulfill the requirements of SIOV, we are proposing adding the following functionalities:
1. Extend IOASID set to support permission checking, token sharing, quota management
2. Add reference counting for life cycle management
3. Add per IOASID set the private ID for non-identity guest-host PASID mappings
4. Add notifiers to keep IOASID users synchronized on state change events, e.g. FREE, BIND/UNBIND, etc.

At LPC 2020. We are trying to get a consensus on the principles of these API extensions. If time permits, we would like to walk through the life-cycle of an IOASID on Intel’s SIOV enabled platforms. Kernel documentation will be included in the patchset submission.

I agree to abide by the anti-harassment policy

I agree

Primary authors: LIU, Yi; PAN, Jacob
Presenters: LIU, Yi; PAN, Jacob
Session Classification: VFIO/IOMMU/PCI MC
Track Classification: VFIO/IOMMU/PCI MC
The Linux kernel has lacked support for RCEC AER handling until now. Several patches have been submitted to address this gap. The purpose of this discussion is to ensure various cases for use of RCEC in native and non-native modes (sometimes referred to as firmware-first) are addressed.

https://lore.kernel.org/linux-pci/20200812164659.1118946-1-sean.v.kelley@intel.com/

I agree to abide by the anti-harassment policy

I agree

Primary author: Mr KELLEY, Sean (Intel Corp.)
Presenter: Mr KELLEY, Sean (Intel Corp.)
Session Classification: VFIO/IOMMU/PCI MC
Track Classification: VFIO/IOMMU/PCI MC
Welcome / Intro

Welcome, Overview and platform audio/debug

**Presenters:** HILMAN, Kevin (BayLibre); LEVIN, Sasha

**Session Classification:** Testing and Fuzzing MC

**Track Classification:** Testing and Fuzzing MC
Isolated dynamic user namespaces

Monday, 24 August 2020 09:30 (20 minutes)

We would like to discuss a proposal for more advanced in-kernel idmap isolation.

I agree to abide by the anti-harassment policy

I agree

Primary authors: GRABER, Stéphane (Canonical Ltd.); BRAUNER, Christian (Canonical)

Presenters: GRABER, Stéphane (Canonical Ltd.); BRAUNER, Christian (Canonical)

Session Classification: Containers and Checkpoint/Restore MC

Track Classification: Containers and Checkpoint/Restore MC
This is a first brainstorm around building a sensible, better capability model on top of pidfds.

I agree to abide by the anti-harassment policy

I agree

Primary author:  BRAUNER, Christian (Canonical)
Presenter:  BRAUNER, Christian (Canonical)
Session Classification:  Containers and Checkpoint/Restore MC
Track Classification:  Containers and Checkpoint/Restore MC
Contribution ID: 249

Welcome

Thursday, 27 August 2020 07:00 (5 minutes)

Presenters: WEBSTER, Behan (Converse in Code Inc.); DESAULNIERS, Nick (Google)

Session Classification: LLVM MC

Track Classification: LLVM MC
Allowing device drivers to enable PCI capabilities vs IOMMU

Wednesday, 26 August 2020 09:35 (15 minutes)

Current implementation allows IOMMU to automatically enable certain PCI features that require IOMMU co-ordination. For various reasons to ensure ordering etc. But new use cases such as Scalable IOV, and also a way to quirk behavior due to bugs could be managed on the device vs adding a certain quirk table and such. Provides more control to support new requirements from modern devices such as devices that support SIOV.

I agree to abide by the anti-harassment policy
I agree

Primary authors: RAJ, Ashok; LU, Baolu
Presenters: RAJ, Ashok; LU, Baolu
Session Classification: VFIO/IOMMU/PCI MC
Track Classification: VFIO/IOMMU/PCI MC
dma-iommu conversion work for the Intel VT-d driver

Wednesday, 26 August 2020 10:05 (20 minutes)

We can remove a lot of duplicated code from the Intel IOMMU driver by using the generic dma-iommu path for IO virtual address handling.

We have two main issues preventing us from merging this work. The intel i915 gpu driver doesn’t handle scatter gather lists correctly and we need to work on a generic copy of the Intel IOMMU driver’s bounce buffer code for untrusted devices.

This micro conference will be a great opportunity to get together with the relevant people in a (virtual) room and discuss open issues and how to make progress on that work so that it can eventually be merged.

I agree to abide by the anti-harassment policy

I agree

Primary author: MURPHY, Thomas
Presenter: MURPHY, Thomas
Session Classification: VFIO/IOMMU/PCI MC
Track Classification: VFIO/IOMMU/PCI MC
The Intel Volume Management Device (VMD) behaves similar to a PCI-to-PCI bridge that changes the subdevice’s requester ID to VMD’s. VMD also remaps subdevice MSI/X into its own MSI/X mapping table. Because of the requester ID factor, the VMD device and subdevice domain fall under a single IOMMU group.

VMD is being integrated more and more into Intel chipsets and the desire to assign individual subdevices is only going to become more of an outstanding problem as time goes on. The existing model of assignment of the whole IOMMU group to a VM is problematic to VMD subdevices, as well as any expectation surrounding interrupt remapping.

VFIO/IOMMU may need an (unsafe) DMA remapping provider-consumer relationship to assigning individual subdevice DMA contexts. To handle MSI/X, the guest may need to avoid using it in the first place, or have VMD in the host deliver the interrupts.

I agree to abide by the anti-harassment policy

I agree

Primary author: DERRICK, Jonathan
Presenter: DERRICK, Jonathan
Session Classification: VFIO/IOMMU/PCI MC
Track Classification: VFIO/IOMMU/PCI MC
BoF: The GNU C Library

Tuesday, 25 August 2020 07:30 (25 minutes)

The GNU C Library is used as the C library in the GNU systems and most systems with the Linux kernel. The library is primarily designed to be a portable and high performance C library. It follows all relevant standards including ISO C11 and POSIX.1-2008. It is also internationalized and has one of the most complete internationalization interfaces known.

This BoF aims to bring together developers of other components that have dependencies on glibc and glibc developers to talk about the following topics:
* What is the state of 64-bit time_t? glibc? kernel?
* What is the state of the RV32 port?
* Planning for glibc 2.33 and what work needs to be done between August 2020 and January 2021.
* Planning for glibc 2.34 and what work needs to be done between January 2021 and July 2021.
... and more.

I agree to abide by the anti-harassment policy

I agree

Primary author: O’DONELL, Carlos (Red Hat)
Presenter: O’DONELL, Carlos (Red Hat)
Session Classification: GNU Tools Track
Track Classification: GNU Tools Track
Linux gained a new process creation system call clone3() in 2019 for the 5.3 release. It provides a superset and hopefully cleaner semantics than legacy clone().

I'd like to discuss a few things related to it:

- How to expose this safely to other libraries: various libraries in userspace want to make use of it to get access to new features such as CLONE_INTO_CGROUP (notably systemd for this one) and others. What are the adoption blockers? Can we sanely deal with deadlocking issues due to atfork handlers? Should we even expose the separate stack to userspace?

- Improving the stack handling: the legacy clone syscall exposes a stack (and on some architectures a stack size) argument to userspace. clone3() does this too because we didn’t want to regress any use-cases so legacy-clone() callers could migrate to clone3(). There’s a few differences though. clone3() requires a stack size argument to be passed and it doesn’t require userspace to know in which direction the stack grows. Each architecture will do the right thing in the kernel instead. However, it still seems that we require userspace to do too much. When I look at what each clone() implementation is doing in the glibc source code in pure assembly my head starts spinning. How can we make this easier? Can we come up with a scheme that makes it almost trivial to use the stack argument in userspace?
Security related flags: a common GCC/LLVM view

Wednesday, 26 August 2020 08:30 (25 minutes)

Both GCC and LLVM toolchains provide a wide range of security-related flags. Some are dedicated to finding bugs statically, some provide low-cost runtime protection and other rely on heavy instrumentations. Although many flags are shared between the two toolchains, some are unique, and there implementation may differ. This talk aims at providing a broad overview and status on these security-related aspects.

I agree to abide by the anti-harassment policy

I agree

Primary author: GUELTON, Serge

Presenter: GUELTON, Serge

Session Classification: GNU Tools Track

Track Classification: GNU Tools Track
DTrace on Linux

Thursday, 27 August 2020 09:00 (45 minutes)

DTrace on Linux has existed for many years now, but it depended on rather invasive kernel modifications. With the emergence of tracing facilities in the Linux kernel, such as BPF, perf, tracepoints, … a re-implementation of the well-known DTrace tool (and D language) is possible without extensive kernel modifications.

The re-implementation of DTrace has been ongoing and has made significant progress in the past 12 months. The BoF session will give a brief overview of the work that has been done, with highlights of the techniques used. The bulk of the session is aimed at discussing the work that remains to be done and to brainstorm ways to do it.

References: https://github.com/oracle/dtrace-utils/tree/2.0-branch-dev
Wiki: https://github.com/oracle/dtrace-utils/wiki
Mailing list: https://oss.oracle.com/mailman/listinfo/dtrace-devel

I agree to abide by the anti-harassment policy

I agree

Primary author:  VAN HEES, Kris (Oracle USA)
Presenter:  VAN HEES, Kris (Oracle USA)
Session Classification:  BOFs Session
Track Classification:  Birds of a Feather (BoF)
containers and mountinfo woes

This summarizes my (not-so-good) experience wrt using the kernel API exposed as /proc/*/mount{s,info} in various container projects (docker, runc, aufs, cri-o, cilium etc.), and outlines various problems with this API and its (ab)use.

Mountinfo API is quite adequate for 10s of mounts (systems with no containers). With containers, each one adds a few mounts, and there might be thousands of containers – so we now have 10.000s of mounts, for which mountinfo is just not working any more.

The following issues are illustrated with examples from real code and/or real bugs.

(1) Some major problems with the current mountinfo API are:

- it is slow (since there is no way to get information about a specific mount point, or a specific subset of mounts – it’s all or nothing); in my experience, it takes up to 0.1s to read mountinfo on a loaded system;
- it is text-based (so everyone writes their own parser, and many of them are slow and/or incorrect);
- it is racy (there is a mount but it can’t be found) – and this leads to actual bugs.

(2) In addition to the above issues, there are cases when mountinfo is abused by userspace developers (most can be fixed). Those would not cause issues if mountinfo is fast – alas currently it’s not the case.

- checking if a mount(2) has succeeded (not needed at all);
- checking if a mount is already there before calling mount(2):
  - not needed in many cases;
  - can be done using two stat(2) syscalls – for real fs;
  - unavoidable with bind mounts;
- checking if a mount is there before calling umount(2) (not needed at all);
- checking if umount(2) succeeded (not needed);
- finding mount root of a specified directory (an alternative approach is to traverse the directory tree up calling stat(2) until dev is no longer matches);
- parsing mountinfo multiple times in a loop ((runc did it 50 to 100+ times for a simple runc run call);

(3) So, we are in a desperate need of a new API.

Here are the typical use cases:
- check if a directory is a mount point (including or excluding bind mounts);
- find all mounts under a given path;
- get some info about a particular mount (same as mountinfo currently provides, e.g. propagation flags or Root directory aka field 4);
- ...
Introduction

Thursday, 27 August 2020 07:00 (5 minutes)

A brief overview of the presenters and topics.

**Presenters:** FRIEDT, Christopher (Friedt Professional Engineering Services); FUSTINI, Drew (BeagleBoard.org Foundation); KRIDNER, Jason (Texas Instruments and BeagleBoard.org Foundation)

**Session Classification:** You, Me, and IoT Two MC
**Untrusted/External devices management**

*Wednesday, 26 August 2020 08:15 (25 minutes)*

Location v/s Trust

- Currently firmware can mark ports as external-facing (and thus indicates any devices downstream that port are external). PCI & IOMMU subsystem treats external devices as untrusted (ATS is not allowed, sets up bounce buffers, and uses "strict" iommu).
- We should separate "Location" from "Trust". (Not all internal devices may be trustworthy).
- Location of a device should be exposed to the user space as a read only property. (E.g. use case: user may want to keep statistics about external devices plugged, and differentiate it from internal devices).
- It is OK if we want to treat external devices as untrusted (as current). But we should expose the pci_dev->untrusted property of the device to userspace (to allow it to implement any special policies it may want to implement for untrusted devices).
- Ideally userspace should also be able to change the pdev->untrusted attribute (i.e. be able to choose which devices to treat as trusted vs untrusted). This is a harder problem to solve as pdev->untrusted is used in the boot path by IOMMU code (i.e. before userspace comes up).

**I agree to abide by the anti-harassment policy**

I agree

**Primary author:** Mr JAIN, Rajat (Google)

**Presenter:** Mr JAIN, Rajat (Google)

**Session Classification:** VFIO/IOMMU/PCI MC

**Track Classification:** LPC Microconference (Closed)
BoF: Core Scheduling API

Wednesday, 26 August 2020 07:00 (45 minutes)

The goal of this discussion is to bring some of the discussions from LKML to a room where some of us can get together, figure out what interface makes sense for initial merging into upstream, what future goals are. Are cgroups the way forward, or should coreschedfs be a thing?

We want to follow this up with

Core Scheduling: Cross CPU priority comparison

Core Scheduling load balancing has been one of the corner cases that still has to be resolved. While we have the attention of the core group, we want to try to resolve this issue, and get to a point where we can have fixes and have a clear path ahead to merging.

I agree to abide by the anti-harassment policy

Presenter: GIANI, Dhaval (Oracle)
Session Classification: BOFs Session
Track Classification: Birds of a Feather (BoF)
Contribution ID: 263

Type: not specified

**Introduction**

*Thursday, 27 August 2020 07:00 (5 minutes)*

**Session Classification:** System Boot and Security MC
Contribution ID: 264

**BoF: Extensible Syscalls - Checking for Supported Features**

*Friday, 28 August 2020 07:00 (45 minutes)*

Based on the kernel summit talk “Extensible Syscalls” we want to continue the discussions around checking for supported features in syscalls. The were various proposals in the room that would be interesting to discuss in detail and come to a conclusion what would work best!

I agree to abide by the anti-harassment policy

I agree

**Presenters:** SARAI, Aleksa (SUSE LLC); BRAUNER, Christian (Canonical)

**Session Classification:** BOFs Session
BoF: refcount_t conversions

Friday, 28 August 2020 08:00 (45 minutes)

Many reference counters in the kernel are still atomic_t. There are Coccinelle scripts to find these, there are older patches sent to the list that were ignored, and new instances have been added. Let’s try to get this work finished up.
https://github.com/KSPP/linux/issues/104

I agree to abide by the anti-harassment policy

I agree

Presenter: COOK, Kees (Google)
Session Classification: BOFs Session
BoF: Show off your beer!

Friday, 28 August 2020 11:00 (4 hours)

This is a place for a post-conference gathering to celebrate the end of a long week. Hang out with members of the program committee, speakers, and attendees, lift a glass of whatever is appropriate for your time zone, and enjoy one last BBB experience before we all disperse again. No presentations, no slides.

I agree to abide by the anti-harassment policy

Presenter: CORBET, Jonathan (Linux Plumbers Conference)

Session Classification: BOFs Session

Track Classification: Birds of a Feather (BoF)
BoF: Improving Diversity

Friday, 28 August 2020 10:00 (45 minutes)

Abstract:
There have been numerous initiatives to increase the diversity of contributors to the Linux kernel over the years, and there has been a steady increase in the relative % of contributors as well as absolute numbers. This BOF will review some of the historical data on gender diversity from the recently released kernel history report. Challenge to brainstorm on is how to shift the relative %higher, what’s critical mass, etc.

Improving diversity is not just limited to gender, and if participants want to start to discuss how we can improve outreach and inclusion to other groups, that would be great.

I agree to abide by the anti-harassment policy

I agree

Presenters: STEWART, Kate (Linux Foundation); KHAN, Shuah (The Linux Foundation)

Session Classification: BOFs Session