Contribution ID: **6** Type: **not specified**

Performance and Scalability Microconference

There are a few wide-ranging, active projects to enhance performance and scalability in both the Linux kernel and userspace. One purpose of this forum is for developers from these different projects to meet and collaborate - not only kernel developers but also researchers doing more experimental work. Another purpose is to cover topics that are too broad for more specific sessions. The structure will be similar to what was followed the previous years, including topics such as synchronization primitives, bottlenecks in memory management, testing/validation, lockless algorithms, and RCU.

Here are some examples of potential topics, though the attendees will ultimately decide the agenda:

- Seamless hypervisor update with IOMMU-type-agnostic, directly-attached devices and virtual functions. The goal is to update the host kernel while minimizing guest downtime to avoid disruption in guest applications (e.g. timeouts). This work does not yet address VFIO devices. A discussion between relevant projects and developers would be beneficial, including Jason Zeng (VMM Fast Restart), Anthony Yznaga (PKRAM), and Pavel Tatashin, who has been researching this topic and upstreaming optimizations facilitating the broader project. We also need some MM folks because the approach will involve passing more kernel memory state from one kernel to another via kexec or warm reboots.
- Performance characteristics of RT spinlocks. This is in line with upstreaming the remaining locking bits from the -RT patchset. While this is mostly for an RT-specific track, some performance characteristics that would land in rtmutex are worth discussing, such as lateral stealing and top-waiter optimistic spinning. No complete series has been posted yet, but this is something core folks are aware of. Series
- Accounting CPU-intensive kernel threads in the CPU controller via remote charging. There is no general way to account kernel thread CPU cycles spent on behalf of a cgroup. Use cases include async memory reclaim, helper threads in multithreaded padata jobs, unbound workqueues (writeback, dm-crypt, btrfs), and net rx. Initial requirement, Discussion 1, Discussion 2
- Design discussion and performance characteristics of Maple Tree. v1 series, lwn article
- mmap_sem contention in procfs. There have been no discussions upstream about this (nor are there solutions yet), but it has been discussed offline frequently and there is machinery to reproduce the issue quickly here. The overall issue is that a thread reading procfs can block other threads trying to change the address space and have latency issues. The ideal way of addressing this is for readers not to have to take the mmap_sem at all, but instead rely on still having a valid VMA (memory) when freeing under RCU, along with the maple tree infrastructure.
- futex2: Among other things, attempts to tackle the performance limitations of the single NUMA node hash table, making operations from remote nodes more expensive. Series
- Batching optimizations in the internals of get_user_pages() and put_user_pages(), and changing or adding interfaces to the same to enable more pages to be pinned at once. There is no consensus on what these interfaces should look like. Discussion
- NUMA-aware spinlocks (series now in v14 upstream, still with outstanding issues). series, lwn article
- Fast kdump for embedded devices. The problems are outlined in this post, but there has been no discussion so far.

Attendees:

- Alex Kogan (interested)
- Andre Almeida (interested)
- Anthony Yznaga (interested)
- Jason Zeng (response pending)
- Paul McKenney (interested)
- Peter Zijlstra (response pending)
- Waiman Long (interested)

These are the folks we have at least contacted so far, and we plan to reach out to more people soon.

Daniel, Pavel, and Ying Huang organized the last Performance and Scalability Uconf in 2018, with this schedule.

Here are some of the outcomes from the event:

- With feedback from the audience, Daniel Jordan was able to get the first steps of his project, formerly known as ktask to parallize CPU-intensive work, merged in mainline as part of the padata parallel execution mechanism. Deferred struct page init is now parallelized on x86 systems. Patch 1/8, Patch 6/8, padata documentation
- During Boqun Feng's topic on workqueues and CPU hotplug, the audience concluded that the problem under discussion was actually not an issue and a stale comment had misled the community, so a follow-on patch removed part of the initial fix. Initial fix, Follow-on patch
- In response to a question from the audience, Mike Kravetz proposed aligning addresses returned from mmap(MAP_ANONYMOUS) calls of at least THP size on THP boundaries, resulting in an RFC patch discussion.

The previous Performance and Scalability Uconf was held in 2015 and organized by Davidlohr Bueso:

- Blog post
- Wiki page
- LKML announcement

I agree to abide by the anti-harassment policy

I agree

Primary authors: BUESO, Davidlohr (SUSE Labs); JORDAN, Daniel; TATASHIN, Pavel

Session Classification: Performance and Scalability MC

Track Classification: Performance and Scalability MC