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

Scaling performance profiling infrastructure for data centers

Monday 9 September 2019 12:00 (45 minutes)

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

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

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

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

I agree to abide by the anti-harassment policy

Yes

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

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

Session Classification: LPC Refereed Track