Contribution ID: **266** Type: **not specified**

user-space only uprobes - could (a BPF-based) vDSO help?

Thursday 19 September 2024 17:00 (30 minutes)

There have been many improvements in reducing the overhead associated with user-space probes (uprobes) such as using system calls instead of traps to fire the probe. However it remains the fact that there is significant overhead associated with uprobe firing. Add to this that in many tracing cases, the predicate associated with the probe is negative;

- is execname == “foo”?
- is pid == 1234?

etc. In such cases we still need to trap or syscall into the kernel to run the BPF program that evaulates those predicates. However, with a combination of a memory-mapped BPF map (storing the “foo” or the 1234 for comparison) and a vDSO-like experience (where tgids, uids etc are cached in a memory-mapped map such that no syscall is required to retrieve them), many such predicates could potentially be evaluated fully in userspace. This would mean that in the negative predicate case a trap/syscall would not be required, limiting overhead for uprobe attachment to cases where in-kernel execution is required. We would need a way of JITing to a userspace-only program along with a mode of attachment that worked system-wide for userspace programs. Many of the helpers would be JITed to a vDSO retrieval (e.g. bpf_get_current_pid_tgid()).

Exploring BPF-based vDSO in its own right is also interesting, since BPF can overcome some of the issues with simple caching of values (since it can catch events that invalidate cached values and update them), but the added selling point of facilitating user-space only tracing makes this a potentially interesting area for exploration with the community.

Primary author: MAGUIRE, Alan (Oracle)

Presenter: MAGUIRE, Alan (Oracle)

Session Classification: eBPF Track

Track Classification: eBPF Track