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

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. eg: 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, tracec-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