Contribution ID: **429** Type: **not specified**

Challenges with Kernels built with LTO

We have been using CONFIG_LTO_CLANG_THIN in our production kernel for a few years. While delivering non-trivial performance improvements, kernels with CONFIG_LTO_CLANG_THIN enabled also bring challenges to our work:
LTO causes confusion for tracing users. With LTO, the compiler is more likely to do selective inlining, i.e., inline a kernel function at some call sites, but not some others. When such kernel functions are being traced, the user can easily miss a lot of events without realizing it.
LTO makes kernel live patching more difficult, especially with the kpatch-build toolchain. With LTO, individual object files are LLVM IR, kpatch-build cannot easily perform binary diff on these object files. LTO also makes it more challenging to verify the correctness of a live patch.
In this talk, we will share more information about these issues, and discuss with the community about potential ways to address these issues.

Primary authors: LIU, Song (Meta); SONG, Yonghong

Presenters: LIU, Song (Meta); SONG, Yonghong

Session Classification: Toolchains MC

Track Classification: Toolchains MC