Contribution ID: **178** Type: **not specified**

Kernel ABI Monitoring and Toolchain Support

Wednesday 14 September 2022 12:45 (45 minutes)

The new CTF(Compact C Type Format) supported in libabigail is able to extract a corpus representation for the debug information in Kernel binary and its modules, i.e, entire Kernel release (kernel + modules). Using CTF reader improvements the time to extract and build the corpus compared with DWARF reader, for example, extracting ABI information from the Linux kernel takes up to ~4.5times less time, this was done using a Kernel compiled by GCC, nowadays LLVM doesn't support binaries generation with CTF debug info, would be nice to have this.

But what about of the modules inserted (loaded) at runtime in the Kernel image?. To make the comparison it uses kABI scripts this is useful among other things to load modules with compatible kABI, this mechanism allows modules to be used with a different kernel version that of the kernel for which it was built. So what of using a single notion of ABI (libabigail) also for the modules loader?

Since we add support for CTF in libabigail, is needed the patch for building the Kernel with CTF enabled in the Kernel upstream configuration. Also some GCC attributes that affect the ABI and are used by kernel hackers like noreturn, interrupt, etc. are not represented in DWARF/CTF debug format and therefore they are not present in the corpus.

A stricter conformance to DWARF standards would be nice, full DWARF 5 support, getting things like ARM64 ABI extensions (e.g., for HWASAN) into things like elfutils at the same time as the compile-link toolchain, more consistency between Clang and GCC debug info for the same sources, the same for Clang and Clang with full LTO. And an extending ABI monitoring coverage beyond just architecture, symbols and types / dealing with header constants, macros and more

The interest in discussing ways to standardize ABI and type information in a way that it can be embedded into binaries in a less ambiguous way. In other words, what can we do to not rely entirely on intermediate formats like CTF or DWARF to make sense of an ABI? Maybe CTF is already a good starting point, yet some additions are needed (e.g. other language features like for C++)?

I agree to abide by the anti-harassment policy

Yes

Primary authors: Mr SEKETELI, Dodji; Mr PROCIDA, Giuliano; Mr MARTINEZ, Guillermo E.; Mr MÄNNICH, Matthias

Presenters: Mr SEKETELI, Dodji; Mr PROCIDA, Giuliano; Mr MARTINEZ, Guillermo E.; Mr MÄNNICH, Matthias

Session Classification: Toolchains

Track Classification: Toolchains Track