Contribution ID: **316** Type: **not specified**

Supply true signatures to vmlinux BTF

I would like to discuss my ongoing work to supply true signatures for available functions in kallsyms. The change will be in clang, pahole, libbpf and kernel.

For clang, the goal is to add additional functions to dwarf where these functions are not in current dwarf or their signatures have changed. The example includes like
- original func 'void foo(int a, int b)' becomes 'void foo(int b)', so a new 'void foo(int b)' will be encoded in dwarf.
- functions like bar.llvm.<hash> at lto mode.
- functions with struct/union arguments and these arguments will become two (or more?) separate arguments (e.g., 16 byte struct will become two actual arguments for x86_64), or become a pointer to the struct/union.

In pahole, the new functions will be added to vmlinux.BTF.
If an existing func signature changed, add the signature-changed func to vmlinux.BTF and discard the old one.
The goal is to represent actual function signatures in vmlinux.

With new vmlinux.BTF, we can have true signatures available to users for fentry etc. We can also handle functions like bar(…) (in vmlinux.BTF bar.llvm.<hash>) in libbpf properly (e.g., through regex match etc.). gcc functions like bar.isra, tar.constprop, etc. can be handled as well if true signature is available.

Primary author: SONG, Yonghong

Presenter: SONG, Yonghong

Session Classification: eBPF Track

Track Classification: eBPF Track