Source-based livepatch creation tooling

Linux Plumbers Conference 2019 – Live Patching MC

Nicolai Stange
Live patching developer
nstange@suse.de

klp-ccp – C Copy&Paste

```
klp-ccp -o lp.c                                          \
        --compiler=x86_64-gcc-4.8.1                      \
        --patched-functions=foo,bar                      \
        --pol-cmd-can-externalize-fun=my-pol-cef.sh      \
        ...                                              \
  -- ORIGINAL GCC CMDLINE
```

1. Preprocess, parse, evaluate,
2. build closure of patched functions,
3. depreprocess, write output.

https://github.com/SUSE/klp-ccp

Open questions: Integration / Best practices

- Obtaining the original GCC command line
  - `make -sn foo/bar.o` from configured kernel sources?
  - Store away `*.cmd` files from original kernel build?
  - `make 2>&1 | tee make.out` during kernel build?
- Externalizability of (unmodified) `static` functions
  - Almost always optional and a size optimization. Exception: escaped function pointers.
  - `readelf -sW <obj>`. Pitfall: incompatible definitions from different compilation units. See `encode_string()` from `nfsv4.ko`: `nfs4xdr.c` vs. `callback_xdr.c`.
  - GCC's IPA clones dump of limited use, dead code elimination not recorded.
  - Relate source location information with DWARF (`DW_TAG_subprogram`'s `DW_AT_name`, `DW_AT_decl_file`, `DW_AT_decl_line`, `DW_AT_low_pc`)? Problem: IPA clones indistinguishable from "proper" out-of-line instances of inlined functions.
- Would it make sense to establish common conventions for accessing objects, DWARF and IPA clone dumps?

Example: DWARF entry for IPA clone

```
0000000000003c60 FUNC LOCAL DEFAULT vmx_l1d_flush.isra.47
```

```
< 1><0x00029758>  DW_TAG_subprogram
DW_AT_name                vmx_l1d_flush
DW_AT_decl_file           0x00000009 ../arch/x86/kvm/vmx.c
DW_AT_decl_line           0x000023f5
DW_AT_prototyped          yes(1)
DW_AT_inline              DW_INL_inlined
```

```
< 1><0x0004557b>  DW_TAG_subprogram
DW_AT_abstract_origin     <0x00029758>
DW_AT_low_pc              0x00003c60
DW_AT_high_pc             <offset-from-lowpc>100
DW_AT_frame_base          DW_OP_call_frame_cfa
DW_AT_GNU_all_call_sites  yes(1)
```