Kbuild support for klp-relocation generation

Lukáš Hruška <lhrska@suse.cz>
Kernel Live Patching Developer
Motivation

- LP needs to reference:
  - global symbols (even exported in case of modules)
  - non-included local symbols
- Unexported `kallsyms_lookup_name()`
- Module loader - Livepatch module ELF format for relocations
  - requires `.ko` modification - not possible from src / `objcopy`
  - `modpost` modification required (skip unresolved symbols in LP module)
LP relocations compared to Userspace relocations

Userspace relocations

Section: \textit{.rela.section\_name}
- \textit{.rela} - prefix for all relocation sections
- \textit{.section\_name} - name of section to which those relocation entries are related

Section Entry: \textit{offset \ldots symbol\_name}
- \textit{offset} - the location in the binary that needs to be updated
- \textit{symbol\_name} - the name of the symbol that this relocation needs to find during linking
LP relocations compared to Userspace relocations

**LP relocations**

**Section:** 
- `.klp.rela.objname.section_name`  
- SHF_RELA_LIVEPATCH flag
  
  - `.klp` - prefix indicating this section is managed by kernel livepatching
  
  - `.rela + .section_name` - same as Userspace
  
  - `.objname` - name of object LPed by this module

**Section Entry:** 
- `... .klp.sym.objname.symbol_name,symbol_pos`
  
  - `.klp.sym` - prefix indicating this symbol is managed by kernel livepatching
  
  - `.objname` - name of object where to look for this symbol
  
  - `.symbol_name` - same as Userspace
  
  - `.sympos` - index of symbol in case of existing multiple symbols with the same name
**klp-convert**

- RFC in 2016 (Josh Poimboeuf)
  - macro `KLP_MODULE_RELOC` "moving" `klp_module_reloc` structure to specific section
  - `modpost` skipping all unresolved symbols on modules tagged by "livepatch" in `modinfo`
- v2 in 2019
  - tool can automatically resolve reloc symbols
  - in case of multiple symbols with same name print their details and position
  - macro `KLP_SYMPOS` creates `klp_module_reloc` record with specified symbol position
- v3-v7 did not introduce some new functionalities
  - bug fixes, edge-cases, styling
Minimal version
Based on v7 patchset

- Removed automated .klp.rela records creation
- macro KLP_RELOC_SYMBOL rename the tagged variable to this format:
  .klp.sym.rela.lp_object.sym_object.sym_name,sympos
  - .klp.sym.rela - prefix which modpost then only allows as unresolved symbol
  - .lp_object - name of object LPed by this module
  - .sym_object - name of object where to look for this symbol
  - .sym_name - name of symbol that the module loader need to find
  - .sympos - index of symbol in case of existing multiple symbols with the same name
Minimal version

Example

```
extern char *saved_command_line \
    KLP_RELOC_SYMBOL(vmlinux, vmlinux, saved_command_line, 0);
```

```
$> readelf -r -W <compiled livepatch module>
Relocation section '.rela.text' at offset 0x32e60 contains 10 entries:
    Offset   Info   Type               Symbol's Value  Symbol's Name + Addend
[...]
0000000000000068 0000003c00000002 R_X86_64_PC32
0000000000000000 .klp.sym.rela.vmlinux.vmlinux.saved_command_line,0 - 4
[...]
```

```
$> readelf -r -W <livepatch_module_proceed_by_klp_convert>
Relocation section '.klp.rela.vmlinux.text' at offset 0x5cb60 contains 1 entry:
    Offset   Info   Type               Symbol's Value  Symbol's Name + Addend
0000000000000068 0000003c00000002 R_X86_64_PC32
0000000000000000 .klp.sym.vmlinux.saved_command_line,0 - 4
```
Discussion

- Why still not upstreamed?
- Is minimal version enough for everyone?
- Future of already created selftests?
- Should klp-convert be part of kbuild?
Thank you