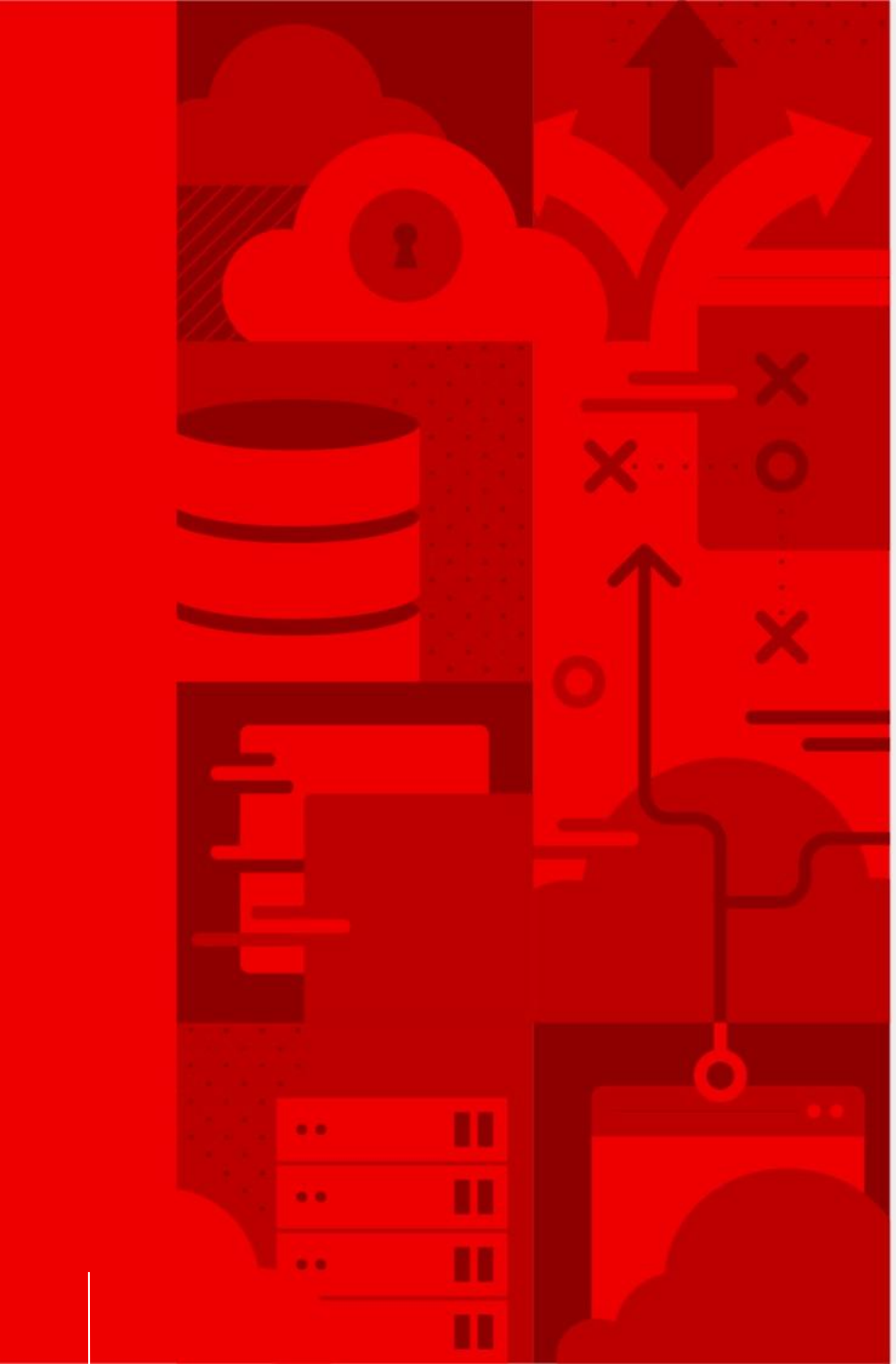


Integrating kas-alias into kernel build: Overcoming Challenges with Non-Invasive Modifications

Alessandro Carminati
Principal Software Engineer

Red Hat

What's kas_alias

- `kas_alias` is yet another script in the `scripts/` directory
- It provides the alias adding service for duplicate symbols for both `vmlinux` image and modules.
- The alias are entries in the main `kallsyms` table or module symbols table that duplicates a line, changing the symbol name by adding `@source_file_line_num`.

```
~ # cat /proc/kallsyms | grep " name_show"
ffffcaa2bb4f01c8 t name_show
ffffcaa2bb4f01c8 t name_show@kernel_irq_irqdesc_c_264
ffffcaa2bb9c1a30 t name_show
ffffcaa2bb9c1a30 t name_show@drivers_pnp_card_c_186
ffffcaa2bbac4754 t name_show
ffffcaa2bbac4754 t name_show@drivers_regulator_core_c_678
ffffcaa2bc025e2c t name_show
ffffcaa2bc025e2c t name_show@drivers_fpga_fpga_mgr_c_618
ffffcaa2a052102c t name_show    [hello]
ffffcaa2a052102c t name_show@hello_hello_c_8    [hello]
ffffcaa2a051955c t name_show    [rpmsg_char]
ffffcaa2a051955c t name_show@drivers_rpmsg_rpmsg_char_c_365     [rpmsg_char]
```

State as version 7

- ➕ Provides the alias service for the `vmlinux` image
- ➕ Provides the alias service for the in-tree modules
- ➕ Export symbol statistics
- ➕ Provides the alias service for later builds and/or out-of-tree modules using the exported file
- ➖ The Makefile machinery I chose to implement the modules part, is at very best, controversial
- ➖ It does nothing to address the mangled duplicate symbols from LLVM monolithic LTO builds
- ➖ Duplicate symbols (name and the rest) from headers inclusion, still have duplicate names
- ➖ Names duplicates from C file inclusion
- ➖ If a module introduces a duplicate for a symbol that was unique in the tree build, this symbol in the module will have the alias, the old build does not

Makefile pipeline issue

Issue Statement:

- Add aliases to necessary modules by extending the `Makefile.modfinal` pipeline with actions added to module link commands.

Implementation Details:

- Prepare the `kas_alias` command in `Makefile.modfinal`.
- Integrate `kas_alias` into the linker sequence.

Current Process:

- `kas_alias` modifies the object inline and uses a backup to handle rebuilds.
- `Makefile.modfinal` modifies the `%.o` files directly.

Proposal:

- Update `kas_alias` to generate a new `.o.kas` file for each module, rather than altering the original file.
- Depending on configuration, `Makefile.modfinal` will produce the module from either `module.o.kas` (aliases enabled) or `module.o` (aliases disabled).

https://t.ly/dZx6D

LTO Symbols

Issue Statement:

- LTO kernel builds in **monolithic** mode avoid duplicate symbols, but identifying **mangled symbols** with numeric suffixes can be challenging.

Details:

- Monolithic LTO with LLVM **nearly eliminates function duplicates from headers**, but **different functions with the same name still exis**t, identified by **mangled names**.
- These symbols aren't flagged as duplicates since they're technically distinct, but:
  - Tracing a symbol's origin is difficult.
  - The numeric suffix is used only during linking, no use after it.
- Duplicate symbols from the same compiler's unit follow the same mangling scheme.

Proposal:

- Remove the numeric suffix and treat symbols as if they have none.

```
$ aarch64-linux-gnu-nm -n
build_aarch64_llvm_mLTO/vmlinux | grep "
name_show[^_]*"
ffffffc080130138 t name_show
ffffffc0809f8f58 t name_show.49514
ffffffc080b186e8 t name_show.56508
ffffffc080d78290 t name_show.76351
ffffffc080d79bb8 t name_show.76393
ffffffc080d81f48 t name_show.76692
ffffffc080d98938 t name_show.77366
ffffffc080deae08 t name_show.80196
ffffffc080e9fdf0 t name_show.87066
ffffffc080ea09c0 t name_show.87087
ffffffc080ea3f18 t name_show.87260
ffffffc080ea6b58 t name_show.87316
ffffffc080eadf48 t name_show.87596
ffffffc081178040 t name_show.101380
ffffffc081283710 t name_show.105581
```

Red Hat

C file inclusion

Issue Statement:

- When a C file includes another C file, debug information for symbols remains unchanged, even if macros modify the symbol.

Proposal:

- Adjust debug information using the `#line` directive, with the preprocessor performing a simple calculation to maintain line number consistency.

Note:

- Although preprocessor math isn't a real concept, workarounds using preprocessor hacks can address this issue. It's not elegant, but it's effective.

https://t.ly/3YFlB

```
// ====== inc.h ======
#define _X_INC_0 1
#define _X_INC_1 2
#define _X_INC_2 3
…

# define INC_LINE(x) INC_LINE_CONCAT(_X_INC_, x)
# define INC_LINE_CONCAT(a, b) a ## b

// ====== use.c ======
#include <misc/inc.h>

#ifndef ELF_COMPAT
#define ELF_COMPAT 0
#else
#line INC_LINE(__LINE__) "fs/binfmt_elf.c:compat_binfmt_elf.c"
 #endif
```

Later builds issue

Issue:

- When new kernel modules are built **after the initial build**, it can introduce symbols that were previously unique but now become duplicates, leading to **inconsistent aliasing** between the original build and the new module.

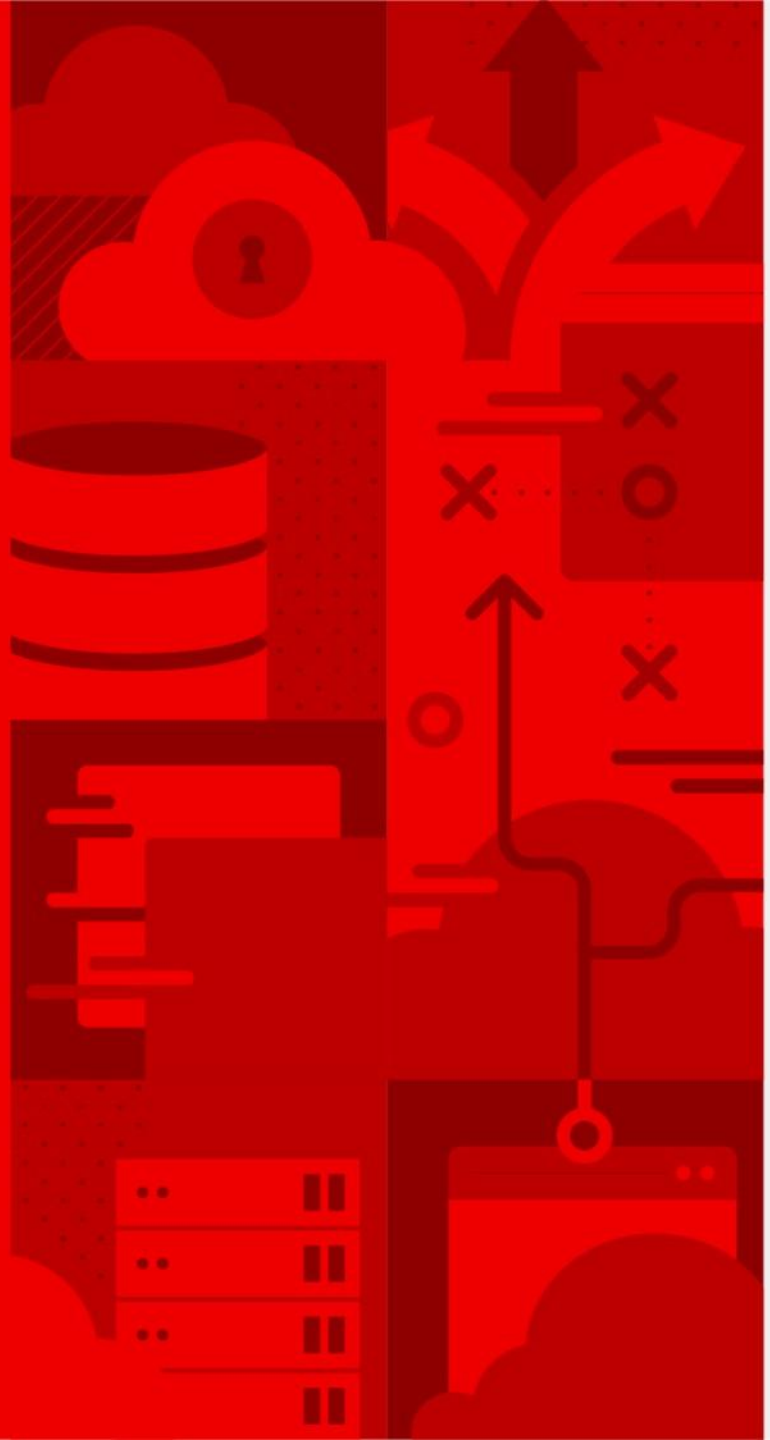
Options to solve this:

- **Hire a fortune-teller**. Not very professional, thought.
- **Accept the current situation** and manage new duplicates manually when they arise.
- **Add aliases to all symbols**, whether they are duplicates or not, to maintain consistency across builds, and pay the price of an insane amount of useless aliases.

The Alias strategy

Issue Statement:

- Assumed community acceptance of alias strategy as a possible solution
- Mixed feedback on the mailing list regarding the strategy's value
- Some comments referred to the approach as "ugly" (subjective feedback)
- Technical concerns: decorated symbols cannot be used between versions
- Seeking feedback from the community (or today’s group) on whether to continue the effort



Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat