type-based slab allocation: `kmalloc_obj` family

Kees Cook <kees@kernel.org>

Linux Plumbers Conference, mm Microconf

2025-12-10

https://outflux.net/slides/2025/lpc/kmalloc_obj.pdf

Goal: reason about allocations based on type info

Currently the allocator has no visibility into type information which would be useful:

- alignment (for better memory layout, or hardening options, e.g. randomize offset within a larger allocation space)
- type/name/identifier (type assignment validation, various hardening and performance improvements: grouping allocations by type, etc)

v5 series posted (with Subject typo, whoops)

https://lore.kernel.org/all/20251203233036.3212363-1-kees@kernel.org/

Old:

```
ptr = kmalloc(sizeof(*ptr), gfp);
ptr = kmalloc(sizeof(struct something), gfp);
ptr = kzalloc(sizeof(*ptr), gfp);
ptr = kmalloc_array(count, sizeof(*ptr), gfp);
ptr = kcalloc(count, sizeof(*ptr), gfp);
```

New:

```
ptr = kmalloc_obj(*ptr, gfp);
ptr = kmalloc_obj(struct something, gfp);
ptr = kzalloc_obj(*ptr, gfp);
ptr = kmalloc_objs(*ptr, count, gfp);
ptr = kzalloc_objs(*ptr, count, gfp);
```

And even non-type-redundant declarations:

```
     auto ptr = kmalloc_obj(struct something, gfp);    /* Linus approved(tm) ;) */
```

Also does flexible array allocation via `kmalloc_flex`

https://lore.kernel.org/all/20251203233036.3212363-4-kees@kernel.org/

```
struct something {
     int counter;
     struct info flex_member[] __counted_by(counter);
} *ptr;

Old:

     ptr = kmalloc(struct_size(ptr, flex_member, count), gfp);

New:

     ptr = kmalloc_flex(*ptr, flex_member, count, gfp);
```

This will do also initialization of the `counter` member, if it was annotated with `counted_by` without needing to add that member name to the macro (through the magic of `__builtin_counted_by_ref`) or a no-op if unannotated.

Next steps, thoughts, feedback?

- Acceptable? (I've simplified/improved it in v5 based on Linus's feedback)

- Need to handle `devm_kmalloc` family too. (I didn't do that yet since the simple API has been getting redesigned a few times already...)