Linux
Plumbers
Conference 2022
>> Dublin, Ireland / September 12-14, 2022

Code tagging, and low overhead kernel memory allocation tracking

kent.overstreet@gmail.com
surenb@google.com

Code tagging framework

Codetag: generic structure to record a code location

Created at compile time for each instrumented code location

Placed into an array of codetags by the linker

Gets embedded into an application-specific structure that records additional information

Framework provides functions to traverse codetags for each application. Application-specific structure can be accessed from a codetag using **container_of**

Codetags from dynamically loaded modules are supported.

Memory allocation tracking

Specifies **alloc_tag** application-specific structure to record allocation call count and total memory size allocated at that location.

Implements debugfs interface to access counter values for each code location that allocates memory.

Provides:

- Low performance and memory overhead
- Memory allocation tracking from different kernel allocators (page, slab, vmalloc)
- Tracking allocations in the kernel core and dynamically loaded modules

Linux Plumbers Conference 2022

>> Dublin, Ireland / September 12-14, 2022

Sample output

```
# sort -hr /sys/kernel/debug/alloc_tags|head
 153MiB  8599 mm/slub.c:1826 module:slub func:alloc_slab_page
6.08MiB  49 mm/slab_common.c:950 module:slab_common func:_kmalloc_order
5.09MiB  6335 mm/memcontrol.c:2814 module:memcontrol func:alloc_slab_obj_exts
4.54MiB  78 mm/page_alloc.c:5777 module:page_alloc func:alloc_pages_exact
1.32MiB  338 include/asm-generic/pgalloc.h:63 module:pgtable func:__pte_alloc_one
1.16MiB  603 fs/xfs/xfs_log_priv.h:700 module:xfs func:xlog_kvmalloc
1.00MiB  256 mm/swap_cgroup.c:48 module:swap_cgroup func:swap_cgroup_prepare
 734KiB  5380 fs/xfs/kmem.c:20 module:xfs func:kmem_alloc
 640KiB  160 kernel/rcu/tree.c:3184 module:tree func:fill_page_cache_func
 640KiB  160 drivers/char/virtio_console.c:452 module:virtio_console func:alloc_buf
```

Selective context capture support

User can enable context capture for the location with the potential leak

```
# echo "file include/asm-generic/pgalloc.h line 63 enable" > /sys/kernel/debug/alloc_tags.ctx
# cat /sys/kernel/debug/alloc_tags.ctx
  920KiB      230 include/asm-generic/pgalloc.h:63 module:pgtable func:__pte_alloc_one
    size: 4096
    pid: 1474
    tgid: 1474
    comm: bash
    ts: 175332940994
    call stack:
         pte_alloc_one+0xfe/0x130
         __pte_alloc+0x22/0xb0
         copy_page_range+0x842/0x1640
         dup_mm+0x42d/0x580
         copy_process+0xfb1/0x1ac0
```

More fun with code tagging

Fault Injection

- dynamic_fault(class)
  Returns true if a fault has been injected, and the code path should fail

Example fault injection points, after hooking memory allocation paths:

```
fs/xfs/libxfs/xfs_iext_tree.c:606 module:xfs func:xfs_iext_realloc_rootclass:memory disabled "
fs/xfs/libxfs/xfs_inode_fork.c:503 module:xfs func:xfs_idata_reallocclass:memory disabled "
fs/xfs/libxfs/xfs_inode_fork.c:399 module:xfs func:xfs_iroot_reallocclass:memory disabled "
fs/xfs/xfs_buf.c:373 module:xfs func:xfs_buf_alloc_pagesclass:memory disabled "
fs/xfs/xfs_iops.c:497 module:xfs func:xfs_vn_get_linkclass:memory disabled "
fs/xfs/xfs_mount.c:85 module:xfs func:xfs_uuid_mountclass:memory disabled "
```

Linux Plumbers Conference 2022

>> Dublin, Ireland / September 12-14, 2022

Latency Tracing

- code_tag_time_stats_start()
- code_tag_time_stats_end()

Example debugfs output, after hooking prepare_to_wait() and finish_wait():

```
fs/xfs/xfs_extent_busy.c:589 module:xfs func:xfs_extent_busy_flush
count:        61
rate:         0/sec
frequency:    19 sec
avg duration:   632 us
max duration:   2 ms
quantiles (us): 274 288 288 296 296 296 296 336 336 336 336 336 336 336 336
```

Better error codes

Ever have to look through thousands of lines when something is returning -EINVAL and didn't log an error message?

- ERR(errcode)
  Returns a unique error code, convertible to original error with error_class()
- errname() returns an error string that includes file and line number where the error originated

Example output:

```
VFS: Cannot open root device "sda" or unknown-block(8,0): error -EINVAL at fs/ext4/super.c:4387
```

Linux Plumbers Conference 2022

>> Dublin, Ireland / September 12-14, 2022

Status and Future work

RFC posted at: https://lore.kernel.org/all/20220830214919.53220-1-surenb@google.com

Tree for testing: https://github.com/surenbaghdasaryan/linux/tree/alloc_tags_rfc

Future work:

Instrument more allocators:

- vmalloc
- per-cpu allocations
- any others we missed?

Other code tagging application ideas?