# Kernel Pwn Cheat Sheet



## Kernel version

```
commit 09688c0166e76ce2fb85e86b9d99be8b0084cdf9 (HEAD -> master, tag: v5.17-rc8,
origin/master, origin/HEAD)
Author: Linus Torvalds <torvalds@linux-foundation.org>
Date:   Sun Mar 13 13:23:37 2022 -0700

    Linux 5.17-rc8
```

## Kernel config

| config | memo |
| --- | --- |
| CONFIG_KALLSYMS | /proc/sys/kernel/kptr_restrict |

| | |
|---|---|
| CONFIG_USERFAULTFD | /proc/sys/vm/unprivileged_userfaultfd |
| CONFIG_STATIC_USERMODEHELPER | |
| CONFIG_SLUB | default allocator |
| CONFIG_SLAB | |
| CONFIG_SLAB_FREELIST_RANDOM | |
| CONFIG_SLAB_FREELIST_HARDENED | |
| CONFIG_FG_KASLR | |
| CONFIG_BPF | /proc/sys/kernel/unprivileged_bpf_disabled |
| CONFIG_SMP | multi-processor |
| CONFIG_HAVE_STACKPROTECTOR | cannary |
| CONFIG_RANDOMIZE_BASE | kaslr |
| CONFIG_HARDENED_USERCOPY | prevent copying over object size |

## Process management

### task_struct

- task_struct
  - thread_info
    - `syscall_work`
  - cred
  - `tasks`
    - init_task
      - init_cred
  - `comm`
    - `prctl(PR_SET_NAME, name);`

### current

- current
  - get_current
    - current_task
      - DECLARE_PER_CPU
        - DECLARE_PER_CPU_SECTION
          - __PCPU_ATTRS
            - *case CONFIG_SMP*
              - PER_CPU_BASE_SECTION
    - this_cpu_read_stable
      - __pcpu_size_call_return
        - this_cpu_read_stable_8
          - percpu_stable_op

        - *case CONFIG_SMP*
            - `movq %%gs:%P[var], %[val]` where `var = &current_task`

- start_kernel
    - setup_per_cpu_areas
        - *case CONFIG_SMP*
            - per_cpu_offset
            - `__per_cpu_offset[cpu] = pcpu_base_addr - __per_cpu_start + pcpu_unit_offsets[cpu]`
        - switch_to_new_gdt
            - load_percpu_segment
                - cpu_kernelmode_gs_base
                    - fixed_percpu_data
                        - DECLARE_PER_CPU_FIRST
                        - fixed_percpu_data
                    - per_cpu
                        - *case CONFIG_SMP*
                            - per_cpu_ptr
                                - SHIFT_PERCPU_PTR
                                    - RELOC_HIDE
                    - *case CONFIG_SMP*
                        - `gs = &fixed_percpu_data.gs_base + __per_cpu_offset[cpu]`

## Syscall

- entry_SYSCALL_64
    - pt_regs
        - `pt_regs` may be use for stack pivoting
    - do_syscall_64
        - `add_random_kstack_offset();`
        - syscall_enter_from_user_mode
            - __syscall_enter_from_user_work
                - syscall_trace_enter
                    - `SYSCALL_WORK_SECCOMP`
        - do_syscall_x64
    - swapgs_restore_regs_and_return_to_usermode

## Memory allocator

### kmem_cache

- *case CONFIG_SLUB*
    - kmem_cache

        - kmem_cache_cpu
            - `freelist`
            - slab
                - `slab_cache`
                - `freelist`
        - `offset`
        - `random`
        - kmem_cache_node
- *case CONFIG_SLAB*
    - kmem_cache
        - array_cache
            - `entry`
        - kmem_cache_node
            - `shared`

### __kmem_cache_create

- *case CONFIG_SLUB*
    - __kmem_cache_create
        - kmem_cache_open
            - calculate_order
            - calculate_sizes
                - oo_make
                    - order_objects

- *case CONFIG_SLAB*
    - __kmem_cache_create
        - set_objfreelist_slab_cache
            - calculate_slab_order

### kmalloc

- kmalloc
    - kmalloc_index
        - __kmalloc_index
            - *case CONFIG_SLUB*
                - `#define KMALLOC_MIN_SIZE 8`
            - *case CONFIG_SLAB*
                - `#define KMALLOC_MIN_SIZE 32`
    - kmalloc_caches
    - kmalloc_type
        - `#define GFP_KERNEL_ACCOUNT (GFP_KERNEL | __GFP_ACCOUNT)`
        - `GFP_KERNEL` → `KMALLOC_NORMAL`
        - `GFP_KERNEL_ACCOUNT` → `KMALLOC_CGROUP`
    - *case CONFIG_SLUB*
        - kmem_cache_alloc_trace

        - slab_alloc
            - slab_alloc_node
                - __slab_alloc
                    - ___slab_alloc
                        - `slab = c->slab = slub_percpu_partial(c);`
                        - new_slab
                            - allocate_slab
                                - alloc_slab_page
                                    - `folio = (struct folio *)alloc_pages(flags, order);`
                                - shuffle_freelist
                - get_freepointer_safe
                    - freelist_ptr
                    - `get_freepointer_safe(cache, object) = (object + cache->offset) ^ *(object + cache->offset) ^ cache->random`
    - *case CONFIG_SLAB*
        - kmem_cache_alloc_trace
            - slab_alloc
                - __do_cache_alloc
                    - ___cache_alloc
                        - cache_alloc_refill
                    - ___cache_alloc_node
                        - cache_grow_begin
                            - kmem_getpages
                                - __alloc_pages_node
                                    - `return __alloc_pages(gfp_mask, order, nid, NULL);`
                            - cache_init_objs
                                - shuffle_freelist

### kfree

- *case CONFIG_SLUB*
    - kfree
        - virt_to_folio
            - virt_to_page
                - __pa
                    - __phys_addr
                        - __phys_addr_nodebug

                - `x - __START_KERNEL_map + __START_KERNEL_map - PAGE_OFFSET`
                - `#define __PAGE_OFFSET page_offset_base`
            - pfn_to_page
                - __pfn_to_page
                    - vmemmap
                        - VMEMMAP_START
                            - `vmemmap_base`
        - page_folio
            - _compound_head
                - pageflags
    - folio_slab
    - slab_free
        - do_slab_free
            - `likely(slab == c->slab)` → `likely(slab == slab->slab_cache->cpu_slab->slab)`
            - set_freepointer
                - `BUG_ON(object == fp);`
            - __slab_free
                - `put_cpu_partial(s, slab, 1);`

- *case CONFIG_SLAB*
    - kfree
        - __cache_free
            - cache_flusharray
            - __free_one
                - `WARN_ON_ONCE(ac->avail > 0 && ac->entry[ac->avail - 1] == objp)`

## Physmem

- page tables
    - `page_offset_base`
        - heap base address (by kmalloc) and it is mapped to `/dev/mem`
        - `secondary_startup_64` can be found at `page_offset_base + offset`
    - `vmalloc_base`
    - `vmemmap_base`
        - base address of pages

## Paging

- `CR3`, `Page Global Directory`, `Page Upper Directory`, `Page Middle Directory`, `Page Table Entry` are used

- each register or variable holds an encoded pointer, not a raw pointer
- the 12~51 bits of each register or valiable indicates the base address of the next directory
- see 5.3.3 4-Kbyte Page Translation / AMD64 Architecture Programmer's Manual, Volume 2 for details
- last byte of `Page Global Directory(PML4E)` often be 0x67(0b01100111)

## Copy from/to user

- copy_from_user
  - check_copy_size
    - *case CONFIG_HARDENED_USERCOPY*
      - check_object_size
        - __check_object_size
          - check_heap_object
            - *case CONFIG_HARDENED_USERCOPY*
              - *case CONFIG_SLUB*
                - __check_heap_object
              - *case CONFIG_SLAB*
                - __check_heap_object
              - *otherwise*
                - __check_heap_object
              - check_page_span
    - *otherwise*
      - check_object_size
- copy_to_user

## Symbol

- EXPORT_SYMBOL
  - _EXPORT_SYMBOL
    - __EXPORT_SYMBOL
      - __cond_export_sym
        - ___cond_export_sym
          - __cond_export_sym_1
            - ___EXPORT_SYMBOL
              - __KSYMTAB_ENTRY
                - RO_DATA
- kernel_symbol_value
  - offset_to_ptr

## Snippet

- gain root privileges
  - (kernel) `commit_creds(prepare_kernel_cred(NULL));`
- break out of namespaces

- (kernel) `switch_task_namespaces(find_task_by_vpid(1), init_nsproxy);`
- (user) `setns(open("/proc/1/ns/mnt", O_RDONLY), 0);`
- (user) `setns(open("/proc/1/ns/pid", O_RDONLY), 0);`
- (user) `setns(open("/proc/1/ns/net", O_RDONLY), 0);`

## Structures

| structure | size | flag (v5.14+) | memo |
|---|---|---|---|
| ldt_struct | 16 | GFP_KERNEL_ACCOUNT | |
| shm_file_data | 32 | GFP_KERNEL | |
| seq_operations | 32 | GFP_KERNEL_ACCOUNT | /proc/self/stat |
| msg_msg | 48 ~ 4096 | GFP_KERNEL_ACCOUNT | |
| msg_msgseg | 8 ~ 4096 | GFP_KERNEL_ACCOUNT | |
| subprocess_info | 96 | GFP_KERNEL | `socket(22, AF_INET, 0);` |
| timerfd_ctx | 216 | GFP_KERNEL | |
| pipe_buffer | 640 = 40 x 16 | GFP_KERNEL_ACCOUNT | |
| tty_struct | 696 | GFP_KERNEL | /dev/ptmx |
| setxattr | 0 ~ | GFP_KERNEL | |
| sk_buff | 320 ~ | GFP_KERNEL_ACCOUNT | |

### ldt_struct

- modify_ldt
  - write_ldt
    - `#define LDT_ENTRIES 8192`
    - `#define LDT_ENTRY_SIZE 8`
    - alloc_ldt_struct
  - read_ldt
    - desc_struct
    - `copy_to_user`
      - `copy_to_user` won't panic the kernel when accessing wrong address

### shm_file_data

- shmat
  - do_shmat

### seq_operations

- proc_stat_init
  - stat_proc_ops
- stat_open
  - single_open_size

        - single_open
            - `start = single_start`
            - `next = single_next`
            - `stop = single_stop`
            - `show = show`

- seq_read_iter
    - `m->op->start`

## msg_msg, msg_msgseg

- msg_queue
    - `q_messages` → `msg_msg`
- msgsnd
    - ksys_msgsnd
        - do_msgsnd
            - load_msg
                - alloc_msg

- msgrcv
    - ksys_msgrcv
        - do_msgrcv
            - `#define MSG_COPY 040000`
            - copy_msg

## subprocess_info

- socket
    - __sys_socket
        - sock_create
            - __sock_create
                - __request_module
                    - call_modprobe
                        - call_usermodehelper_setup

## timerfd_ctx

- timerfd_create
- timerfd_release
    - `kfree_rcu`

## pipe_buffer

- pipe, pipe2
    - do_pipe2
        - do_pipe_flags
            - create_pipe_files
                - get_pipe_inode
                    - alloc_pipe_info
                        - `#define PIPE_DEF_BUFFERS 16`

        - pipefifo_fops

- pipe_write
  - `buf->ops = &anon_pipe_buf_ops;`
- pipe_release
  - put_pipe_info
    - free_pipe_info
      - pipe_buf_release
        - `ops->release`

## tty_struct

- unix98_pty_init
  - tty_default_fops
    - tty_fops
- ptmx_open
  - tty_init_dev
    - alloc_tty_struct
- tty_ioctl
  - tty_paranoia_check
    - `#define TTY_MAGIC 0x5401`
  - tty_pair_get_tty
  - `tty->ops->ioctl`

### setxattr

- setxattr
  - path_setxattr
    - setxattr
      - `vfs_setxattr` may fail, but `kvmalloc` and `kvfree` complete successfully

## sk_buff

- socketpair
  - __sys_socketpair
    - sock_create
      - __sock_create
        - *case PF_UNIX*
          - unix_family_ops
            - unix_create
              - *case SOCK_DGRAM*
                - unix_dgram_ops
              - unix_create1
                - `sk->sk_allocation = GFP_KERNEL_ACCOUNT;`

- unix_dgram_sendmsg
  - sock_alloc_send_pskb
    - alloc_skb_with_frags
      - alloc_skb
        - __alloc_skb
          - `struct skb_shared_info` is at the end of `data`

## Variables

| variable | memo |
| --- | --- |
| modprobe_path | /proc/sys/kernel/modprobe |
| core_pattern | /proc/sys/kernel/core_pattern |
| n_tty_ops | (read) `scanf`, (ioctl) `fgets` |

### modprobe_path

- execve
  - do_execve
    - do_execveat_common
      - bprm_execve
        - exec_binprm
          - search_binary_handler
            - __request_module
              - call_modprobe
                - call_usermodehelper_setup
                - call_usermodehelper_exec

### core_pattern

- do_coredump
  - format_corename
  - call_usermodehelper_setup
  - call_usermodehelper_exec

### poweroff_cmd

- orderly_poweroff
  - poweroff_work_func
    - __orderly_poweroff
      - run_cmd
        - call_usermodehelper
          - call_usermodehelper_setup
          - call_usermodehelper_exec

### n_tty_ops

- tty_struct
  - tty_ldisc
- n_tty_init
  - tty_register_ldisc