

# Kernel Pwn Cheat Sheet

- [Kernel version](#)
- [Kernel config](#)
- [Process management](#)
  - [task\\_struct](#)
  - [current](#)
- [Syscall](#)
- [Memory allocator](#)
  - [kmem\\_cache](#)
  - [kmalloc](#)
  - [kfree](#)
- [Phymem](#)
- [Paging](#)
- [Copy from/to user](#)
- [Symbol](#)
- [Snippet](#)
- [Structures](#)
  - [ldt\\_struct](#)
  - [shm\\_file\\_data](#)
  - [seg\\_operations](#)
  - [msg\\_msg, msg\\_msgseg](#)
  - [subprocess\\_info](#)
  - [timerfd\\_ctx](#)
  - [pipe\\_buffer](#)
  - [tty\\_struct](#)
  - [setxattr](#)
  - [sk\\_buff](#)
- [Variables](#)
  - [modprobe\\_path](#)
  - [core\\_pattern](#)
  - [poweroff\\_cmd](#)
  - [n\\_tty\\_ops](#)

## 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\\_kernelmodes\\_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](#)
  - [swaps\\_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

## kmalloc

- [kmalloc](#)
  - [kmalloc\\_index](#)
    - [\\_\\_kmalloc\\_index](#)
      - case *CONFIG\_SLUB*
        - #define KMAALLOC\_MIN\_SIZE 8
      - case *CONFIG\_SLAB*
        - #define KMAALLOC\_MIN\_SIZE 32
  - [kmalloc\\_caches](#)
  - [kmalloc\\_type](#)
    - #define GFP\_KERNEL\_ACCOUNT (GFP\_KERNEL | \_\_GFP\_ACCOUNT)
    - GFP\_KERNEL → KMAALLOC\_NORMAL
    - GFP\_KERNEL\_ACCOUNT → KMAALLOC\_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](#)
                  - [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](#)
          - [cache\\_init\\_objs](#)
            - [shuffle\\_freelist](#)

## kfree

- *case CONFIG\_SLUB*
    - [kfree](#)
      - [slab\\_free](#)
        - [do\\_slab\\_free](#)
          - `likely(slab == c->slab) → likely(slab == slab->slab_cache->cpu_slab->slab)`
          - [\\_\\_slab\\_free](#)
            - [set\\_freepointer](#)
              - `BUG_ON(object == fp);`
          - `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)`

## Physem

- [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`

## 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 variable 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         | <code>socket(22, AF_INET, 0);</code> |
| 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`

## **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   |
|----------------------------|--|
| <code>modprobe_path</code> | <code>/proc/sys/kernel/modprobe</code>                 |
| <code>core_pattern</code>  | <code>/proc/sys/kernel/core_pattern</code>             |
| <code>n_tty_ops</code>     | (read) <code>scanf</code> , (ioctl) <code>fgets</code> |

### [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](#)