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

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

Syscall

Memory allocator

kmem_cache

- *case CONFIG_SLAB*
  - [kmem_cache](#)
    - [array_cache](#)
      - `entry`
    - [kmem_cache_node](#)
      - `shared`

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](#)
              - [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);`

- *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`
- [page](#)
- [vmalloc_to_page](#)
- [page_to_virt](#)
  - `page_to_virt(page) = page_offset_base + (((page - vmemmap_base) / 64) << 12)`
  - [__va](#)
    - [PAGE_OFFSET](#)
      - [__PAGE_OFFSET](#)
        - `page_offset_base`
  - [PFN_PHYS](#)
    - [PAGE_SHIFT](#)
  - [page_to_pfn](#)
    - [__page_to_pfn](#)
      - [vmemmap](#)
        - [VMEMMAP_START](#)
          - `vmemmap_base`

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](#)
        - [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](#)

- [seq_read_iter](#)
  - `m->op->start`

[msg_msg, msg_msgseg](#)

- [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
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**

n_tty_ops

- **tty_struct**
  - **tty_ldisc**
- **n_tty_init**
  - **tty_register_ldisc**