PB173 – Ovladače jádra – Linux

XI.

Jiří Slabý

ITI, Fakulta Informatiky

7. 12. 2010

Část I

Komunikace mezi procesy

Komunikace mezi procesy

LDD3 část kap. 6 (zastaralá)

- Čekání na událost
- Buzení procesů

Jednoduchá komunikace

- 2 procesy (producent-konzument)
  - A: čeká na nějakou hotovou práci
  - B: udělá nějakou práci a oznámí A dokončení
  - A: pokračuje

API

- **`linux/completion.h`**, `struct completion`
- `DECLARE_COMPLETION`$_S$, `init_completion`$_D$
- Čekání: `wait_for_completion`, `wait_for_completion_interruptible` (retval)
- Dokončení: `complete`, `complete_all`

```
static DECLARE_COMPLETION(my_comp);
static char *str;
...
```

A

```
wait_for_completion(&my_comp);
printk(KERN_DEBUG "%s\n", str);
```

B

```
strcpy(str, "Ahoj");
complete(&my_comp);
```

Úkol

Použití `completion`

1. Doplnit do pb173/11/events `completion`
2. Číst se bude, až někdo dokončí zápis
   - Čekání v `read` je možné přerušit signálem (`wait_for_completion_interruptible`)
3. Spustit 2 instance `cat /dev/my_name`
4. Pozorovat, co se děje
5. Spustit nekolikrát `echo XXX > /dev/my_name`

Složitější komunikace

- Čekání na více místech na jinou událost
- 3 procesy
  - A: plní buffer
  - B: čeká na 100B
  - C: čeká na 200B

API

- **`linux/wait.h`**, `wait_queue_head_t`
- `DECLARE_WAIT_QUEUE_HEAD`$_S$, `init_waitqueue_head`$_D$
- Čekání: `wait_event`, `wait_event_interruptible`
- Dokončení: `wake_up`, `wake_up_all`
- `completion` je obalená `wait_queue`

Složitější komunikace – příklad

```
static DECLARE_WAIT_QUEUE_HEAD(my_wait);
static atomic_t my_cnt = ATOMIC_INIT(0);
static char *str;
...
A
  while (1) {
    str[atomic_inc_return(&my_cnt) - 1] = 'A';
    wake_up_all(&my_wait);
    msleep(10);
  }

B
  wait_event(my_wait, atomic_read(&my_cnt) > 100);
  printk(KERN_DEBUG "%s\n", str);

C
  if (wait_event_interruptible(my_wait, atomic_read(&my_cnt) > 200)) {
    printk(KERN_WARNING "interrupted\n");
    return;
  }
  printk(KERN_DEBUG "%s\n", str);
```

Úkol

Použití `wait_queue`

1. Místo `completion`, použít v pb173/11/events `wait_queue`
2. Číst se bude, až někdo zápíše více než 5 znaků

Scheduler

- Celé je to instruování plánovače
- Ve skutečnosti je `wait_event` a `wake_up`:
  - Nastavení stavu procesu: `set_current_state`
  - Uspání procesu: `schedule`
  - Probuzení procesu: `wake_up_process`
  - Popř. s obsluhou signálů: `signal_pending`
  - **`linux/sched.h`**
- `completion` a `wait_queue` ulehčuje práci
  - Pamatováním si, koho vzbudit
  - Případnou kontrolou signálů
  - Případnou kontrolou vypršení timeoutu
  - `completion` navíc řeší předání čítače (kolik procesů vzbudit)

Část II

Hackovací techniky

Hackovací techniky

Z pohledu jádra

- Zabít stroj (DoS)
- Zneužít stroj

Zabít stroj

- Jednodušší
  - Stačí provést něco zlého
    - Předat špatný ukazatel
    - Vytvořit příliš procesů
    - Naalokovat hodně paměti
  - Jádro udělá dereferenci a spadne
- Nejčastější druhy
  - DDoS
  - Paket smrti
  - Neočekávaný vstup od uživatele

Zneužít stroj

- K získání administrátorských oprávnění
- Složitější
  - Musí se předat „divný” ukazatel
  - Jádro udělá dereferenci a spustí náš kód
- Obtížná část:
  - Zjistit, jak se dá díra zneužít
  - Vyrobit správná data, která podstrčit
  - Nepřepsat toho příliš mnoho, aby nedošlo k pádu

Co s root účtem

- Vyrobit si e-mail relay pro spamy
- Upravit jakoukoliv službu
- Ukrást peníze/vědomosti (vláda, armáda)
- Skrýt se v systému (rootkit) a sbírat data
  - Změnit tabulku systémových volání
    - `ls /proc` a `ps` neuvidí proces
    - Napsat vlastní readdir s patřičným filtrem
    - Najít tabulku např. přes /proc/kallsyms
    - Změnit tabulku a volat původní readdir
  - Podobně pro přerušení
    - Stisky klávesnice (hesla) apod.

Úkol

Buffer overflow příklad

1. Prostudujte pb173/11/hack
   - Uživatelský proces: user
   - Jaderný modul: kern
2. Přeložit a vložit modul
3. Nastavit práva /dev/my_name na 666
4. Pod běžným uživatelem spustit pb173:
   - `pb173 0 /dev/my_name` (nezneužít díru)
   - `pb173 1 /dev/my_name` (zneužít díru)
5. Opravit díru ve `write`
   - `pb173 1 /dev/my_name` nesmí skončit s úspěchem