PB173 – Binární programování Linux

VII. C bez libc

Jiri Slaby

Fakulta informatiky
Masarykova univerzita

4. 11. 2014

C a `libc`

- `libc` se stará o komunikaci s OS (jádrem)
  - Jádro poskytuje pouze „základní” funkcionalitu
- `libc` poskytuje spoustu nadstaveb
  - Soubory: `fopen`, `fread`, `fwrite`, `fclose`, `...`
  - Síť: `getaddrinfo`, `ntoh*`, `hton*`
  - Více vláken (`libpthread`): `pthread_*` (včetně zamykání)
  - A spoustu dalšího (celý POSIX), ...

Bez `libc` nic z toho nemáme

Jak teď ale např. otevřu soubor?

Komunikace s jádrem

- Sdílená paměť
  - Jádro zapisuje, uživatel čte a naopak
  - V jednom z dalších cvičení
- Systémová volání
  - Podobné jako zavolání funkce
  - Ale program a jádro běží v jiných kontextech
  - Přepnutí kontextu
    - Uložení a změna stavu procesoru
    - Relativně drahá operace
  - Uvidíme dnes dále

Systémová volání

- Volání funkce
- Instrukcí procesoru
  - Závislé na architektuře
  - Softwarové přerušení (x86_32: číslo `0x80`)
  - Speciální instrukce (x86_32: `sysenter`, x86_64: `syscall`)
  - **Demo:** implementace `syscall` z `glibc`
- Systémová volání jsou očíslovaná
  - Do jednoho registru číslo
  - Čísla `__NR_*` definovaná jádrem (pro každou architekturu)
  - `sys/syscall.h`
  - Např. (x86_64): `__NR_write` (1), `__NR_exit` (60)

Úkol

`libc` o úroveň níže

1. Zavolejte systémová volání
   - `write` s nějakým textem
   - `exit`
2. Pomocí funkce `syscall` z `libc`
   - Viz `man 2 syscall`
   - První argument: číslo systémového volání
   - Další argumenty: odpovídají už danému volání
   - Např. `syscall(__NR_kill, -1, SIGKILL)`
3. Přeložte a vyzkoušejte

C bez `libc`

- Žádné hlavičkové soubory z `libc`
- Jen to, co poskytuje jádro (`/usr/include/linux/`)
- Při linkování: `gcc -nostdlib ...`
  - `gcc` stále může vkládat volání `memset` apod.

Volací konvence

Parametr	Registry			
	Systémová volání		Uživatelský prostor	
	i386	x86_64	i386	x86_64
1.	EBX	RDI	EAX	RDI
2.	ECX	RSI	EDX	RSI
3.	EDX	RDX	Zásobník	RDX
4.	ESI	R10	Zásobník	RCX
5.	EDI	R8	Zásobník	R8
6.	EBP	R9	Zásobník	R9
7.	*Jen 6 parametrů*		Zásobník	Zásobník
8.			Zásobník	Zásobník
Návratová	EAX	RAX	EAX	RAX

Pozn.: i386 a `-mregparm`

Úkol

Systémová volání bez `libc`

1. Otevřete a projděte si `pb173-bin/07/`
2. Zavolejte
   - `fork`
   - Z potomka: `write`
   - Z rodiče: `read` 16 bytů a jejich `write`
   - Z obou potom: `exit`
3. Přeložte a spusťte
   - `gcc -nostdlib ...`

Data z jádra

- Jádro předává
  - Parametry programu
  - Proměnné prostředí
  - Rozšiřující vektor
- Vše na zásobníku
  - Formát zásobníku je pevně daný

Formát zásobníku

Počet parametrů	`long`
1. parametr	`char *`
...	`char *`
m-tý parametr	`char *`
Konec parametrů	`0UL`
1. proměnná prostředí	`char *`
...	`char *`
n-tá proměnná prostředí	`char *`
Konec proměnných prostředí	`0UL`
Rozšiřující vektor 1	`struct { long type, val; }`
...	`struct { long type, val; }`
Rozšiřující vektor o	`struct { long type, val; }`
Konec rozšiřujících vektorů	`{AT_NULL, ? }`

Rozšiřující vektor

Typ záznamů

```
struct {
  long type;
  long value;
};
```

- Typy: `AT_NULL`, `AT_ENTRY`, `AT_RANDOM`, ...
- Pole struktur zakončuje typ `AT_NULL`

Úkol

Zjištění názvu programu a platformy

1. Vypište první parametr programu
   - Tj. název samotného programu
2. Dále iterujte přes zásobník až k rozšiřujícímu vektoru
3. Tam najděte typ `AT_PLATFORM`
4. Vypište hodnotu jako řetězec
5. Přeložte (bez `libc`) a spusťte

Virtuální systémová volání

`vsyscall`

- Jen na některých architekturách
- Neprobíhá přepnutí kontextu
- Speciální stránka(y) s kódem namapovaným jádrem
- Podpora jen 3 funkcí
  - `gettimeofday`: aktuální čas (`0xffffffffff600000`)
  - `time`: čas v sekundách od Epochy (`0xffffffffff600400`)
  - `getcpu`: číslo CPU a NUMA uzlu (`0xffffffffff600800`)
- **Demo:** `/proc/self/maps`

Úkol

Volání `time` z `vsyscall`

1. Zavolejte adresu s `time`
2. Vypište hodnotu
3. Přeložte a spusťte několikrát

Virtuální dynamická knihovna

`vdso`

- Speciální knihovna přilinkovaná jádrem
- Podpora také závislá na architektuře
- Podobná `vsyscall`, ale více flexibilní
  - Navíc obsahuje jen `clock_gettime`
- **Demo:** `ldd` a `readelf`
- Knihovna v ELF formátu
  - Adresa začátku v auxv: `getauxval(AT_SYSINFO_EHDR)`
  - Dále se přečte ELF pomocí `libelf`
  - Ukázkový kód: `Documentation/vDSO/parse_vdso.c`

Úkol: domácí úkol