Област вежби: Linux руковаоци

ТУТОРИЈАЛ О РУКОВАОЦИМА У/И УРЕЂАЈА НА LINUX ОПЕРАТИВНОМ СИСТЕМУ

Предуслови:

- Rpi2 рачунар (нису потреби додатаци),
- Преводиоц *GCC* освежен на верзију 4.7 или новију,
- Преузето и подешено језгро Linux оперативног система на Raspberry Pi уређају, према опису из документа „УВОД - Raspberry Pi рачунар“,
- Подешен мрежни приступ на један од начина представљених у документу „УВОД - Raspberry Pi рачунар“ уколико се ради преко мреже. Ако се Rpi2 рачунар користи као самосталан рачунар овај захтев се може занемарити,
- Познавање језика Це и материјала из вежби „УВОД У КОНКУРЕНТНО ПРОГРАМИРАЊЕ“ и „СИНХРОНИЗАЦИЈА И СИГНАЛИЗАЦИЈА ПРОГРАМСКИХ НИТИ“.

Увод

У овој вежби дат је кратак тутроријал који се односи на прављење руковаоца У/И уређаја на Linux оперативном систему. Туторијал се заснива на одабраним репрезентативним и на посебно написаним примерима који за циљ имају да демонстрирају различите начине писања, као и различите функционалности руковаоца У/И уређаја на Linux ОС. Више информација о писању руковаоца и о самим примерима доступно је у [1], [2], [3] и [4].

Примери

Туторијал чини осам независних пројеката помоћу којих се праве руковаоци У/И уређајима, односно објекти Linux језгра. Такође, приложен је и један пројекат који омогућује прављење корисничког програма који служи за тестирање функционалности једног од модула.

Доступни су следећи пројекти:

1. Пројекат „nothing“ дефинише празан руковалац. То је празан шаблон који се може користити за дефинисање произвољног руковаоца.
2. Пројекат „hello1“ дефинише једноставан руковалац који исписује поздравну поруку.

3. Пројекат „hello2“ дефинише једноставан руковалац који демонстрира употребу *module_init()* и *module_exit()* макроа.
4. Пројекат „hello3“ дефинише једноставан руковаоц који демонстрира употребу *__init*, *__initdata* и *__exit* макроа.
5. Пројекат „hello4“ дефинише једноставан руковалац који демонстрира дефинисање документације руковаоца.
6. Пројекат „hello5“ дефинише једноставан руковалац који демонстрира прослеђивање параметара приликом регистровања руковаоца у Linux језгро.
7. Пројекат „chardev“ дефинише једноставан руковалац који омогућује рад са уређајем знаковног типа. Имплементирано је читање из уређаја (*cat* команда).
8. Пројекат „memory“ дефинше руковалац задате меморијске зоне, која је представљена кроз апстракцију уређаја знаковног типа. У руковаоцу је имплементирано читање и писање (*cat* и *echo* команда, респективно).
9. Пројекат „memory_test_app“ дефинише једноставан кориснички програм, који се користи за тестирање „memory” уређаја. Он демонстрира писање и читање низа знакова из уређаја унутар корисничког програма.

Упутство

У даљем тексту се налазе упутства са одговарајућим командама за превођење, инсталирање и тестирање сваког од руковаоца. Уз пројекте је приложена и текстуална датотека „Readme.txt“ у којој се налазе детаљнија упутства.

Приликом тестирања модула најбоље је отворити два (sudo) терминала. Пријава као sudo корисника се омогућује "sudo -i" командом, након чега је потребно унети лозинку корисника. У једном терминалу је потребно позиционирати се у директоријум у којем се налази модул (*.ko датотека). У другом је потребно омогућити праћење порука које се исписују унутар модула (printk). Ови логови се не виде у стандардном (user space) терминалу већ их је потребно исчитавати из kernel log-a. Kernel log је доступан помоћу команде "cat /proc/kmsg" (блокира терминал и прати све нове логове) или преко "dmesg" (само излиста све поруке). Логовање се врши у том (другом) терминалу.

Примери nothing, hello1, hello2, hello3

- модул се региструје (insert-yje) у језгро командом "insmod *.ko",
- модул се брише/извлачи из језгра командом "rmmod *".

Пример hello4

- информације о модулу су доступне командом: "modinfo *.ko" (нпр. modinfo hello4.ko).

Пример hello5

- "modinfo hello5.ko" даје информације о улазним променљивама које се могу слати приликом позива,
- регистровање hello5.ko модула са свим параметрима:

  "insmod ./hello5/hello5.ko myShort=234 myInt=12345 myLong=9123456 myString="test" myIntArray=7,8",
- наравно, могућа је њихова комбинација или изостављање (користи се default вредност),
- преко "modinfo" команде се добијају информације о улазним аргументима.

Пример chardev

- додавање новог "dev" уређаја: "mknod /dev/<device name> c <major#> <minor#>",
- брисање: "rm /dev/<device name>",
- у chardev примеру он аутоматски тражи од ОС да му додели major#:

  register_chrdev(0, DEVICE_NAME, &fops);

  minor број се у примеру не користи,
- у chardev примеру је продржано само исчитавање: "cat /dev/chardev",
- писање у уређај је додато али није имплементирано (дато је у memory примеру),
- приликом покушаја писања у уређај руковалац избаци поруку "Sorry, this operation isn't supported".

Пример memory

Memory уређај је нешто сложенији. Уз њега иде и једноставан програм "memory_test_app" који тестира функционалност уређаја. Идеја јесте да се дода нови уређај '/dev/memory' који у потпуности омогућује читање и писање у уређај. Читање и писање је исто као и код chardev уређаја.

- у овом примеру се експлицитно дефинише major број (60). Такође је потребно додати уређај у "dev" folder: "mknod /dev/memory c 60 0",
- memory_test_app програм отвара "/dev/memory" уређај и врши једноставно писање и читање из уређаја,
- уколико се у memory_test_app закоментарише писање, могуће је кобиновати писање из конзоле (echo) а читање из програма и обрнуто.

Референце

[1] Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman, „Linux Device Drivers, 3rd Edition“, O'Reilly.

[2] Daniel P. Bovet, Marco Cesati, „Understanding the Linux Kernel, 3rd Edition“, O'Reilly.

[3] Peter Jay Salzman, Michael Burian, Ori Pomerantz, „The Linux Kernel Module Programming Guide“.

[4] Xavier Calbet, „Writing device drivers in Linux: A brief tutorial“, Free Software Magazine.