Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 3
по операционным системам

Студент Колганов О.С.

Группа ИУ7 — 62Б

Преподаватель Рязанова Н.Ю.

Москва.
2020 г.

Задание 1

Реализовать загружаемый модуль ядра, который при загрузке записывает в системный журнал информацию о запущенных процессах. Модуль должен собираться при помощи Make-файла.

Загружаемый модуль должен содержать:

- Указание лицензии GPL
- Указание автора

Текст программы (ps_module.c):

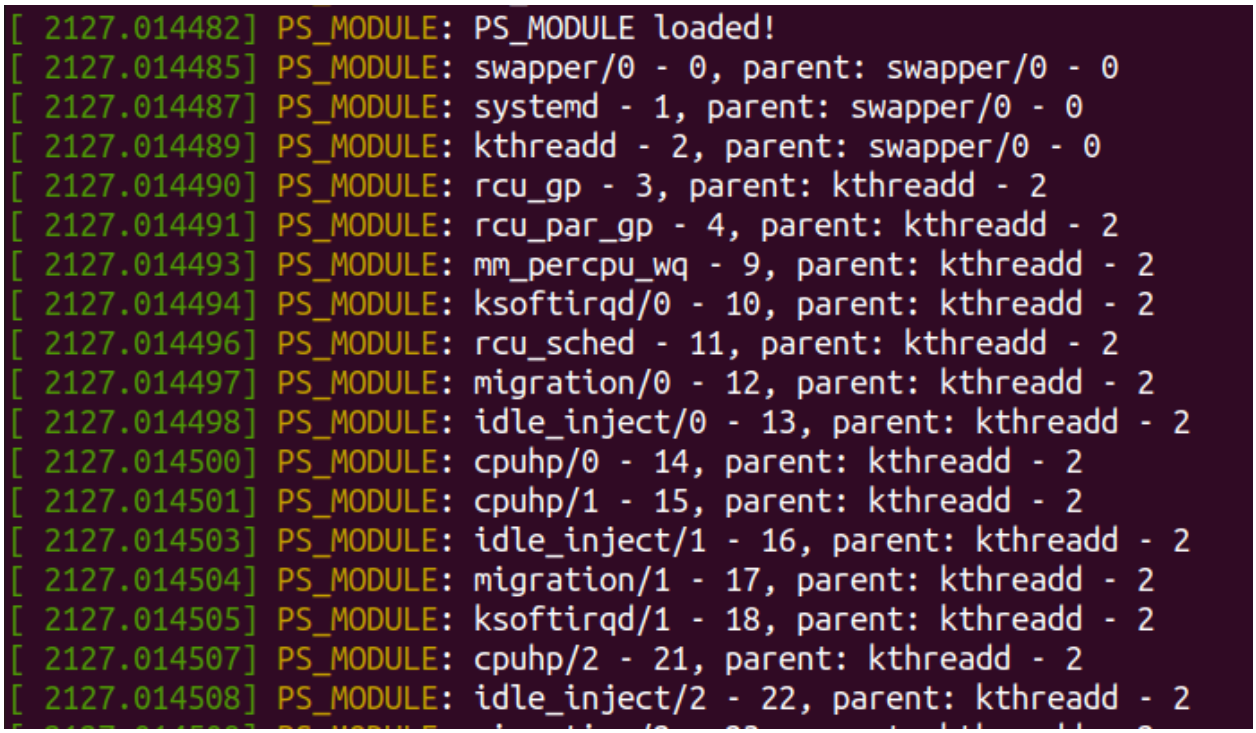
```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/init_task.h>

MODULE_LICENSE("GPL");
MODULE_AUTHOR("Moxxx1e");

static int __init my_module_init(void) {
	printk(KERN_INFO "PS_MODULE: PS_MODULE loaded!\n");

	struct task_struct *task =  &init_task;
	do {
		printk(KERN_INFO "PS_MODULE: %s - %d, parent: %s - %d\n",
task->comm, task->pid, task->parent->comm, task->parent->pid);
	} while ((task = next_task(task)) != &init_task);

	printk(KERN_INFO "PS_MODULE: current: %s - %d, parent: %s - %d\
n", current->comm, current->pid, current->parent->comm, current-
>parent->pid);

	return 0;
}

static void __exit my_module_exit(void) {
	printk(KERN_INFO "PS_MODULE: PS_MODULE unloaded!\n");
```

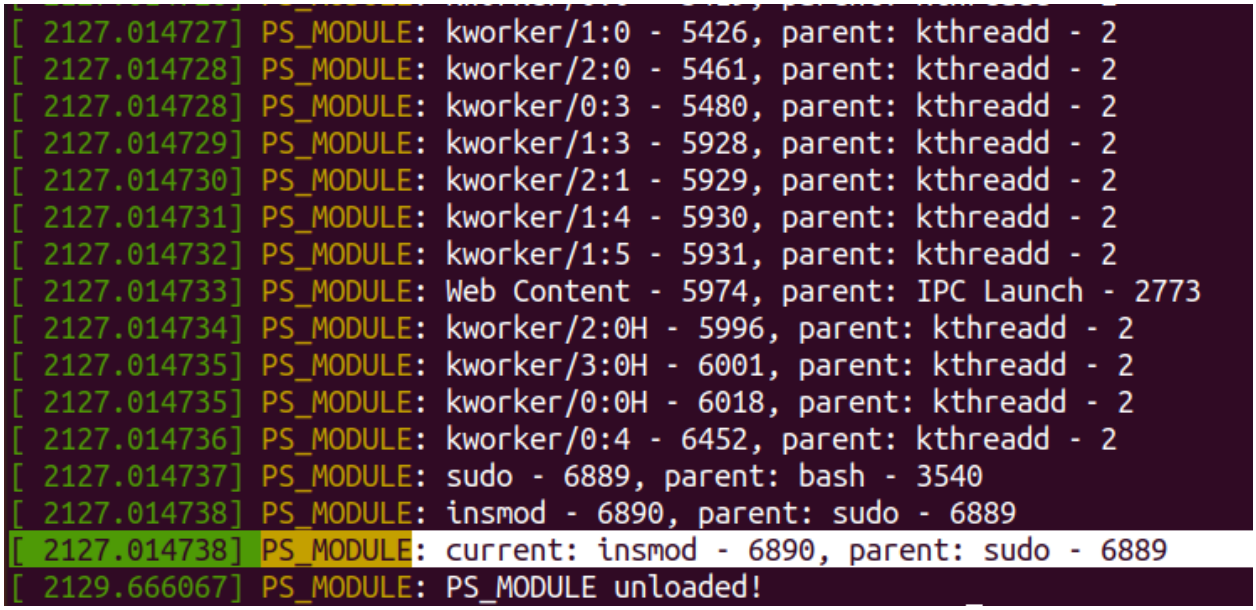
```
}


module_init(my_module_init);
module_exit(my_module_exit);
```

Makefile:

```
CONFIG_MODULE_SIG=n

ifneq ($(KERNELRELEASE),)
	obj-m := ps_module.o

else
	CURRENT = $(shell uname -r)
	KDIR = /lib/modules/$(CURRENT)/build
	PWD = $(shell pwd)

default:
	sudo $(MAKE) -C $(KDIR) M=$(PWD) modules
	sudo make clean

clean:
	rm -rf .tmp_versions
	rm .ps_module.*
	rm *.o
	rm *.mod.c
	rm *.symvers
	rm *.order

endif
```

Демонстрация работы программы





Задание 2

Реализовать три загружаемых модуля ядра:

- Вызываемый модуль md1
- Вызывающий модуль md2
- «Отладочный» модуль md3

Каждый загружаемый модуль должен содержать:

- Указание лицензии GPL

- Указание автора

Загружаемые модули должны собираться при помощи Make-файла (сборка командой make). **Вызов каждой функции модуля должен сопровождаться записью в системный журнал** информации, какая функция какого модуля была вызвана.

Модуль md1

Модуль md1 демонстрирует возможность создания экспортируемых данных и функций. Данный модуль ядра должен содержать:

- Экспортируемые строковые (char *) и численные (int) данные.
- Экспортируемые функции возвращающие строковые и числовые значения.

Например:

  - Функция, возвращающая в зависимости от переданного целочисленного параметра различные строки (на усмотрение студента);
  - Функция, производящая подсчет факториала переданного целочисленного параметра;
  - Функция возвращающая 0;

Текст программы (md1.c):

```
#include <linux/kernel.h>
#include <linux/module.h>


MODULE_LICENSE("GPL");
MODULE_AUTHOR("Moxxx1e");


extern int factorial(int number)
{
    int i, ans;


    printk(KERN_INFO "MD1: factorial function called!");
    ans = number;
    for (i = number - 1; i > 1; i--)
        ans *= i;
```

```
    return (ans == 0) ? (1) : (ans);
}
EXPORT_SYMBOL(factorial);

extern int return_zero(void)
{
    printk(KERN_INFO "MD1: return_zero function called!");
    return 0;
}
EXPORT_SYMBOL(return_zero);

const int hw_code = 1;
char* hw_message = "Hello, world!";
const int gb_code = 2;
char* gb_message = "Good by!";
char* def_message = "The quick brown fox jumps over the lazy dog.";

EXPORT_SYMBOL(hw_code);
EXPORT_SYMBOL(gb_code);
EXPORT_SYMBOL(hw_message);
EXPORT_SYMBOL(gb_message);
EXPORT_SYMBOL(def_message);

extern char* switch_string(int number)
{
    printk(KERN_INFO "MD1: switch_string function called!");
    switch (number) {
    case hw_code:
        return hw_message;
    case gb_code:
        return gb_message;
```

```
    default:
        return def_message;
    }
}
EXPORT_SYMBOL(switch_string);

static int __init my_module_init(void)
{
    printk(KERN_INFO "MD1: module loaded\n");
    return 0;
}

static void __exit my_module_exit(void)
{
    printk(KERN_INFO "MD1: module unloaded\n");
}

module_init(my_module_init);
module_exit(my_module_exit);
```

Модуль md2

Модуль md2 демонстрирует использование данных и функций экспортируемых первым модулем (md1).

Данный модуль должен при загрузке:

- Вызывать все экспортированные модулем md1 процедуры и вывести в системный журнал возвращаемые ими значения с указанием имени вызванной процедуры.
- Вывести в системный журнал все экспортированные модулем md1 данные.

Текст программы (md2.c):

```
#include <linux/kernel.h>
```

```
#include <linux/module.h>

MODULE_LICENSE("GPL");
MODULE_AUTHOR("Moxxx1e");

extern const int hw_code;
extern char* hw_message;
extern const int gb_code;
extern char* gb_message;
extern char* def_message;

extern int factorial(int number);
extern char* switch_string(int number);
extern int return_zero(void);
static int __init my_module_init(void)
{
    printk(KERN_INFO "MD2: module loaded\n");

    printk(KERN_INFO "MD2: export variables:\n"
                     "hw_code: %d, hw_message: %s\n"
                     "gb_code: %d, gb_message: %s\n"
                     "def_message: %s\n",
        hw_code, hw_message,
        gb_code, gb_message, def_message);

    printk(KERN_INFO "MD2: export functions:\n"
                     "factorial(5): %d\n"
                     "return_zero(): %d\n"
                     "switch_string(1): %s",
        factorial(5), return_zero(), switch_string(1));

    return 0;
```

```
}


static void __exit my_module_exit(void)
{
    printk(KERN_INFO "MD2: module unloaded\n");
}


module_init(my_module_init);
module_exit(my_module_exit);
```

Модуль md3

Модуль md3 демонстрирует сценарий некорректного завершения установки модуля, и возможность использования загружаемого модуля в качестве функции выполняемой в пространстве ядре.

Процедура инициализации этого загружаемого модуля должна возвращать ненулевое значение и выводить в системный журнал данные и возвращаемые значения экспортированных модулем md1 процедур (аналогично md2).

Данный модуль включен в работу для проработки вопросов, связанных с отладкой модулей ядра.

Текст программы:

```
#include <linux/kernel.h>
#include <linux/module.h>


MODULE_LICENSE("GPL");
MODULE_AUTHOR("Moxxx1e");


extern const int hw_code;
extern char* hw_message;
extern const int gb_code;
```

```
extern char* gb_message;
extern char* def_message;

extern int factorial(int number);
extern char* switch_string(int number);
extern int return_zero(void);

static int __init my_module_init(void)
{
    printk(KERN_INFO "MD3: module loaded\n");

    printk(KERN_INFO "MD3: export variables:\n"
                     "hw_code: %d, hw_message: %s\n"
                     "gb_code: %d, gb_message: %s\n"
                     "def_message: %s\n",
        hw_code, hw_message,
        gb_code, gb_message, def_message);

    printk(KERN_INFO "MD3: export functions:\n"
                     "factorial(5): %d\n"
                     "return_zero(): %d"
                     "switch_string(1): %s",
        factorial(5), return_zero(), switch_string(1));

    return -1;
}

static void __exit my_module_exit(void)
{
    printk(KERN_INFO "MD3: module unloaded\n");
}
```

```
module_init(my_module_init);
module_exit(my_module_exit);
```

Makefile:

```
CONFIG_MODULE_SIG=n

ifneq ($(KERNELRELEASE),)
	obj-m := md1.o md2.o md3.o

else
	CURRENT = $(shell uname -r)
	KDIR = /lib/modules/$(CURRENT)/build
	PWD = $(shell pwd)

default:
	sudo $(MAKE) -C $(KDIR) M=$(PWD) modules
	sudo make clean

clean:
	rm -rf .tmp_versions
	rm .md1.*
	rm .md2.*
	rm .md3.*
	rm *.o
	rm *.mod.c
	rm *.symvers
	rm *.order

endif
```

Сборка модулей:

```
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ make
sudo make -C /lib/modules/5.3.0-53-generic/build M=/home/oleg/Документы/BMSTU/OS/OS/lab_03 modules
make: вход в каталог «/usr/src/linux-headers-5.3.0-53-generic»
  CC [M]  /home/oleg/Документы/BMSTU/OS/OS/lab_03/md1.o
  CC [M]  /home/oleg/Документы/BMSTU/OS/OS/lab_03/md2.o
  CC [M]  /home/oleg/Документы/BMSTU/OS/OS/lab_03/md3.o
  Building modules, stage 2.
  MODPOST 3 modules
  CC      /home/oleg/Документы/BMSTU/OS/OS/lab_03/md1.mod.o
  LD [M]  /home/oleg/Документы/BMSTU/OS/OS/lab_03/md1.ko
  CC      /home/oleg/Документы/BMSTU/OS/OS/lab_03/md2.mod.o
  LD [M]  /home/oleg/Документы/BMSTU/OS/OS/lab_03/md2.ko
  CC      /home/oleg/Документы/BMSTU/OS/OS/lab_03/md3.mod.o
  LD [M]  /home/oleg/Документы/BMSTU/OS/OS/lab_03/md3.ko
make: выход из каталога «/usr/src/linux-headers-5.3.0-53-generic»
sudo make clean
rm -rf .tmp_versions
rm .md1.*
rm .md2.*
rm .md3.*
rm *.o
rm *.mod.c
rm *.symvers
rm *.order
```

Загрузка 1 и 2 модулей:

```
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo insmod md1.ko
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ lsmod | grep md1
md1                    16384  0
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo insmod md2.ko
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ lsmod | grep md2
md2                    16384  0
md1                    16384  1 md2
```

Выгрузка модулей:

```
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo rmmod md1.ko
rmmod: ERROR: Module md1 is in use by: md2
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo rmmod md2.ko
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo rmmod md1.ko
```

Демонстрация работы модулей

Как показано на скриншоте, модуль md2 вызвал экспортированные модулем md1 процедуры (factorial, switch_string, return_zero), вывел в системный журнал возвращаемые ими значения.

Также модуль md2 вывел все экспортированные модулем md1 данные (hw_code, hw_message, gb_code, gb_message и def_message).

```
[ 3006.466790] MD1: module loaded
[ 3008.244969] MD2: module loaded
[ 3008.244975] MD2: export variables:
               hw_code: 1, hw_message: Hello, world!
               gb_code: 2, gb_message: Good by!
               def_message: The quick brown fox jumps over the lazy dog.
[ 3008.244976] MD1: switch_string function called!
[ 3008.244977] MD1: return_zero function called!
[ 3008.244978] MD1: factorial function called!
[ 3008.244980] MD2: export functions:
               factorial(5): 120
               return_zero(): 0
               switch_string(1): Hello, world!
[ 3014.048509] MD2: module unloaded
[ 3015.838552] MD1: module unloaded
```

Загрузка 1 и 3 (отладочного) модуля.
На скриншоте продемонстрирован сценарий некорректного завершения установки модуля.

```
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo insmod md1.ko
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo insmod md3.ko
insmod: ERROR: could not insert module md3.ko: Operation not permitted
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo rmmod md1.ko
```

Загрузка 2 модуля без предварительной загрузки 1 модуля:

```
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ sudo insmod md2.ko
insmod: ERROR: could not insert module md2.ko: Unknown symbol in module
oleg@Moxxx1e:~/Документы/BMSTU/OS/OS/lab_03$ dmesg | tail -8
[ 1799.244185] md2: Unknown symbol switch_string (err -2)
[ 1799.244240] md2: Unknown symbol factorial (err -2)
[ 1799.244288] md2: Unknown symbol hw_code (err -2)
[ 1799.244335] md2: Unknown symbol gb_code (err -2)
[ 1799.244376] md2: Unknown symbol gb_message (err -2)
[ 1799.244420] md2: Unknown symbol def_message (err -2)
[ 1799.244461] md2: Unknown symbol hw_message (err -2)
[ 1799.244505] md2: Unknown symbol return_zero (err -2)
```