

Міністерство освіти та науки України
Національний технічний університет України
“Київський політехнічний інститут ім. Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки

Лабораторна робота №6

з дисципліни “Технології програмування для комп'ютерних систем – 3”

Виконав:
студент групи ІВ-91мн
Гончаренко Олександр

Київ 2020 р.

Завдання

Завдання розраховане на вже виконане завдання #5, і полягає у модифікації реалізації того завдання.

Зауваження:

- I. Подивіться адреси завантаження своїх модулів на `sysfs` у каталозі поруч з параметрами модуля: `/sys/module/<modulename>/sections/`
 - A. Файли з адресами мають такі ж назви, як було названо секції, тобто починаються з крапки (`.text`, `.init.text`, ...), врахуйте це.
- II. Будь-ласка, додатково ознайомтесь з `debugfs`.
- III. Докладніше у додатку *Debugging Techniques*.
- IV. Для довідки:
 - . `$KDIR/Documentation/admin-guide/dynamic-debug-howto.rst`
 - A. `$KDIR/Documentation/filesystems/debugfs.txt`
 - B. `$KDIR/Documentation/filesystems/proc.txt`
 - C. `$KDIR/Documentation/filesystems/sysfs.txt`
 - D. `$KDIR/Documentation/admin-guide/sysfs-rules.rst`

Завдання:

Для *Basic* потрібно виконати одне з двох завдань на вибір.

Для *Advanced* потрібно виконати обидва.

Завдання Basic 1:

- I. Додайте `BUG_ON()` замість друку повідомлення та повернення `-EINVAL` для неприпустимого значення параметра.
- II. Додайте примусове внесення помилки “начебто `kmalloc()` повернув 0” під час формування елемента списку для якогось повідомлення (останнього із серії, 5-го, ... — на ваш вибір).
- III. Модифікуйте `Makefile` аналогічно *appendix1*.
- IV. Отримайте обидва повідомлення, роздивіться їх та для одного з них виконайте пошук місця аварії аналогічно *appendix1*.
 - A. Зауважте, що при виконанні `BUG_ON()` модуль буде “зайнятий”, і ви не зможете виконати `rmmod`.

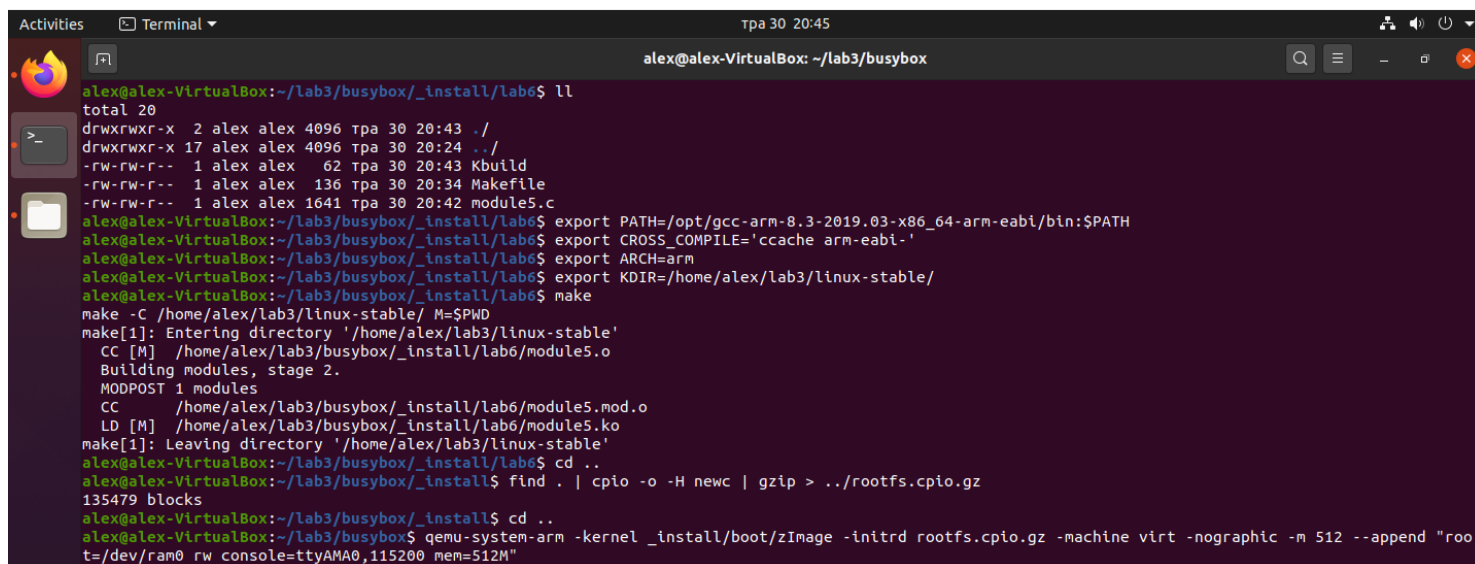
Завдання Basic 2:

- I. Упевніться у відсутності каталогу: `/sys/kernel/debug/dynamic_debug`
 - A. Це означає вимкнену опцію `CONFIG_DYNAMIC_DEBUG` (якщо збиралося по методичці, то не повинно бути).
- II. Замініть у функції `exit` модуля `hello` (`hello1`) друк вмісту списку на `pr_debug` і додайте два виклики `pr_debug` до та після друку списку.
- III. Перевірте залежність друку повідомлень від `#define DEBUG` на початку файлу.
- IV. Перезберіть ядро з увімкненим `CONFIG_DYNAMIC_DEBUG`, замініть його на `nfs`.
 - . Перезберіть модуль.

V. Аналогічно показаному в *appendix2*, поекспериментуйте з друком з прапорцями `r`, `f`, `m`, а також зі встановленням їх для всього модуля та для окремих рядків.

Послідовність виконання роботи

1. Створюємо модуль. Для цього у файловій системі створюємо директорію `lab6`, у яку додаємо усі необхідні файли для роботи з модулем. Виконуємо команди `export` для збирання ядра, а також `export KDIR=/home/alex/lab6/linux-stable/`. Виконуємо команду `make` для збирання модулю. Створимо архів `CPIO` для `rootfs` та заархівуємо його за допомогою `GZip`:

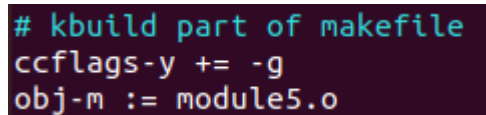


```
alex@alex-VirtualBox: ~/lab3/busybox/_install/lab6$ ll
total 20
drwxrwxr-x  2 alex alex 4096 тра 30 20:43 ./
drwxrwxr-x 17 alex alex 4096 тра 30 20:24 ../
-rw-rw-r--  1 alex alex   62 тра 30 20:43 Kbuild
-rw-rw-r--  1 alex alex  136 тра 30 20:34 Makefile
-rw-rw-r--  1 alex alex 1641 тра 30 20:42 module5.c
alex@alex-VirtualBox:~/lab3/busybox/_install/lab6$ export PATH=/opt/gcc-arm-8.3-2019.03-x86_64-arm-eabi/bin:$PATH
alex@alex-VirtualBox:~/lab3/busybox/_install/lab6$ export CROSS_COMPILE='ccache arm-eabi-'
alex@alex-VirtualBox:~/lab3/busybox/_install/lab6$ export ARCH=arm
alex@alex-VirtualBox:~/lab3/busybox/_install/lab6$ export KDIR=/home/alex/lab3/linux-stable/
alex@alex-VirtualBox:~/lab3/busybox/_install/lab6$ make
make -C /home/alex/lab3/linux-stable/ M=$PWD
make[1]: Entering directory '/home/alex/lab3/linux-stable'
CC [M] /home/alex/lab3/busybox/_install/lab6/module5.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/alex/lab3/busybox/_install/lab6/module5.mod.o
LD [M] /home/alex/lab3/busybox/_install/lab6/module5.ko
make[1]: Leaving directory '/home/alex/lab3/linux-stable'
alex@alex-VirtualBox:~/lab3/busybox/_install/lab6$ cd ..
alex@alex-VirtualBox:~/lab3/busybox/_install$ find . | cpio -o -H newc | gzip > ../rootfs.cpio.gz
135479 blocks
alex@alex-VirtualBox:~/lab3/busybox/_install$ cd ..
alex@alex-VirtualBox:~/lab3/busybox$ qemu-system-arm -kernel _install/boot/zImage -initrd rootfs.cpio.gz -machine virt -nographic -m 512 --append "root=/dev/ram0 rw console=ttyAMA0,115200 mem=512M"
```

2. Виконання завдання Basic1:

Замінімо виведення повідомлення та повернення `-EINVAL` для неприпустимого значення параметра викликом функції `BUG_ON()`.

Додаємо навмисне внесення помилки під час формування останнього елементу списку. До `Kbuild` додаємо прапорець `-g`:



```
# kbuild part of makefile
ccflags-y += -g
obj-m := module5.o
```

Нижче наведено змінений фрагмент коду:

```
Activities Terminal alex@alex-Virtu
#define DEBUG
#include <linux/init.h>
#include <linux/module.h>
#include <linux/printk.h>
#include <linux/ktime.h>
#include <linux/slab.h>

LIST_HEAD(lab5_list_head);

struct time_keeper {
    ktime_t time_before;
    ktime_t time_after;
    struct list_head time_list;
};

static void print_text(unsigned int repeats)
{
    unsigned int repeat;
    struct time_keeper *ptr;
    for (repeat = 0; repeat < repeats; repeat++) {
        if (repeat == repeat - 1)
            ptr = 0;
        else
            ptr = kmalloc(sizeof(*ptr), GFP_KERNEL);
        ptr->time_before = ktime_get();
        ptr->time_after = ktime_get();
        pr_info("Hello, world!\n");
        list_add(&ptr->time_list, &lab5_list_head);
    }
}

static unsigned int repeats = 1;

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "How many hello to print");

static int __init module5_init(void)
{
    BUG_ON(repeats > 10);
    if (repeats >= 5 && repeats <= 10)
        pr_warn("Repeatition from 5 to 10 times\n");
    if (repeats == 0)
        pr_warn("No repeatition\n");
    print_text(repeats);
    return 0;
}
```

3. Протестуємо роботу модуля:

Можна побачити, що завантаження модулю зі значенням параметру, який є більшим за 10 (у прикладі `repeats = 12`), призводить до виконання макросу `BUG_ON`.

```
/ # cd lab6/
/lab6 # modinfo module5.ko
filename:      module5.ko
author:       AlexHoncharenko
description:   Training to debug modules
license:      Dual BSD/GPL
parm:         repeats:How many hello to print
depends:
vermagic:     4.19.120 SMP mod_unload ARMv7 p2v8
```

```
Activities Terminal tpa 30 20:48
alex@alex-VirtualBox: ~/lab3/busybox

/lab6 # insmod module5.ko repeats=11
[ 62.822253] module5: loading out-of-tree module taints kernel.
[ 62.877010] -----[ cut here ]-----
[ 62.884863] kernel BUG at /home/alex/lab3/busybox/_install/lab6/module5.c:39!
[ 62.895554] Internal error: Oops - BUG: 0 [#1] SMP ARM
[ 62.897440] Modules linked in: module5(0+)
[ 62.901073] CPU: 0 PID: 62 Comm: insmod Tainted: G          0      4.19.120 #1
[ 62.907079] Hardware name: Generic DT based system
[ 62.910205] PC is at module5_init+0x18/0x1000 [module5]
[ 62.922178] LR is at do_one_initcall+0x54/0x208
[ 62.922758] pc : [<bf005018>]   lr : [<c0302d4c>]   psr: 200f0013
[ 62.925050] sp : c9435db0 ip : c94172c0 fp : 00000000
[ 62.926249] r10: bf002040 r9 : c1604c48 r8 : 00000000
[ 62.927483] r7 : bf005000 r6 : fffffe00 r5 : c1604c48 r4 : bf002000
[ 62.928879] r3 : 0000000b r2 : 6dc6440a r1 : 00003d26 r0 : 00000000
[ 62.930542] Flags: nzCv IRQs on FIQs on Mode SVC_32 ISA ARM Segment none
[ 62.932271] Control: 10c5387d Table: 493e006a DAC: 00000051
[ 62.934039] Process insmod (pid: 62, stack limit = 0x(ptrval))
[ 62.935762] Stack: (0xc9435db0 to 0xc9436000)
[ 62.942797] 5da0:                                c1788000 c1604c48 fffffe00 bf005000
[ 62.946147] 5dc0: 00000000 c1604c48 bf002040 c0302d4c 00000000 c035c13c 00210d00 00000000
[ 62.949399] 5de0: c1604c48 c93c5600 c9435de4 6dc6440a 00000000 e097afff ffe00000 fffff000
[ 62.952492] 5e00: 8040003f c93c5200 dbd00a60 6dc6440a dbd00a60 c93c5600 bf002040 6dc6440a
[ 62.960797] 5e20: bf002040 00000002 c9417240 00000002 c9417180 c03d2400 00000001 c03d474c
[ 62.962319] 5e40: c9435f30 c9435f30 00000002 c9417140 00000002 c03d4768 bf00204c 00007fff
[ 62.966931] 5e60: bf002040 c03d1658 00000001 c03d0f6c bf002088 bf001150 bf00222c bf002170
[ 62.970283] 5e80: c0f089d4 c1356810 c121dcac c121dcb8 c121dd10 c1604c48 c1608ec4 c9373680
[ 62.978387] 5ea0: ffffff00 e0800000 c9373680 c93c5600 00000000 00000000 00000000 00000000
[ 62.981527] 5ec0: 00000000 00000000 6e72656b 00006c65 00000000 00000000 00000000 00000000
[ 62.984779] 5ee0: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
[ 62.987763] 5f00: 00000000 6dc6440a 00000080 00001bc4 00000000 e0979bc4 0012ce1c c1604c48
[ 62.995880] 5f20: 0011b1f8 fffffe00 00000051 c03d4bac e09682c2 e0968400 e0968000 00011bc4
[ 62.999097] 5f40: e0979444 e0979278 e09757d8 00003000 00003040 00000000 00000000 00000000
[ 63.003135] 5f60: 00001720 0000002d 0000002e 00000018 00000000 00000010 00000000 6dc6440a
[ 63.004670] 5f80: 000f411f 0011b1f8 b6f81950 00011bc4 00000080 c0301204 c9434000 00000080
[ 63.010622] 5fa0: 000f411f c0301000 0011b1f8 b6f81950 0011b258 00011bc4 0011b1f8 00000000
[ 63.014410] 5fc0: 0011b1f8 b6f81950 00011bc4 00000080 00000001 bef53e80 001086c5 000f411f
[ 63.017672] 5fe0: bef53b38 bef53b28 0003b270 b6e3b1b0 600f0010 0011b258 00000000 00000000
[ 63.026121] [<bf005018>] (module5_init [module5]) from [<c0302d4c>] (do_one_initcall+0x54/0x208)
[ 63.036039] [<c0302d4c>] (do_one_initcall) from [<c03d2400>] (do_init_module+0x64/0x214)
[ 63.038759] [<c03d2400>] (do_init_module) from [<c03d4768>] (load_module+0x2150/0x243c)
[ 63.040631] [<c03d4768>] (load_module) from [<c03d4bac>] (sys_init_module+0x158/0x18c)
[ 63.045960] [<c03d4bac>] (sys_init_module) from [<c0301000>] (ret_fast_syscall+0x0/0x54)
[ 63.046939] Exception stack(0xc9435fa8 to 0xc9435ff0)
[ 63.047927] 5fa0:                                0011b1f8 b6f81950 0011b258 00011bc4 0011b1f8 00000000
[ 63.055128] 5fc0: 0011b1f8 b6f81950 00011bc4 00000080 00000001 bef53e80 001086c5 000f411f
[ 63.057393] 5fe0: bef53b38 bef53b28 0003b270 b6e3b1b0
[ 63.060628] Code: e34b4f00 e5943000 e353000a 9a000000 (e7f001f2)
[ 63.067008] ---[ end trace d96997be976862a4 ]---
```

За допомогою утиліти objdump можна побачити, що значення PC та рядку з BUG_ON є ідентичними.

```
Activities  Terminal  тра 30 20:51
alex@alex-VirtualBox: ~/lab3/busybox/_install/
swapoff: can't open '/etc/fstab': No such file or directory
The system is going down NOW!
Sent SIGTERM to all processes
Sent SIGKILL to all processes
Requesting system poweroff
[ 187.875819] reboot: Power down
alex@alex-VirtualBox:~/lab3/busybox$ cd _install/lab6/
alex@alex-VirtualBox:~/lab3/busybox/_install/lab6$ arm-eabi-objdump -dS module5.ko

module5.ko:      file format elf32-littlearm

Disassembly of section .init.text:

00000000 <init_module>:

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "How many hello to print");

static int __init module5_init(void)
{
    0:  e92d47f0      push    {r4, r5, r6, r7, r8, r9, sl, lr}
      BUG_ON(repeats > 10);
    4:  e3004000      movw    r4, #0
    8:  e3404000      movt    r4, #0
   c:  e5943000      ldr     r3, [r4]
  10:  e353000a      cmp     r3, #10
  14:  9a000000      bls     1c <init_module+0x1c>
  18:  e7f001f2      .word   0xe7f001f2
}
```

Якщо ж вести значення параметру менше за 10, то при завантаженні модуля (з параметром, наприклад, repeats=7), побачимо null pointer dereference.


```

/lab6 # insmod module5.ko repeats=7
[ 70.632603] module5: loading out-of-tree module taints kernel.
[ 70.638332] Repeation from 5 to 10 times
[ 70.638930] Hello there!
[ 70.639309] Hello there!
[ 70.639642] Hello there!
[ 70.639992] Hello there!
[ 70.640320] Hello there!
[ 70.640648] Hello there!
[ 70.641080] Unhandled fault: page domain fault (0x81b) at 0x00000000
[ 70.641941] pgd = (ptrval)
[ 70.642347] [00000000] *pgd=56e67835, *pte=00000000, *ppte=00000000
[ 70.643484] Internal error: : 81b [#1] SMP ARM
[ 70.644215] Modules linked in: module5(O+)
[ 70.644977] CPU: 0 PID: 64 Comm: insmod Tainted: G          0      4.19.114 #1
[ 70.645909] Hardware name: Generic DT based system
[ 70.646902] PC is at module4_init+0xa0/0x1000 [module5]
[ 70.647212] LR is at 0x1/
[ 70.647355] pc : [<bf0050a0>]      lr : [<00000017>]      psr: 000f0013
[ 70.647508] sp : d6e69db0   ip : 40000000   fp : 00000000
[ 70.647638] r10: 006000c0   r9 : 00000007   r8 : c135834c
[ 70.647772] r7 : bf0010bc   r6 : 00000007   r5 : 00000000   r4 : bf002000
[ 70.647930] r3 : 00000010   r2 : b8000000   r1 : 00000010   r0 : 708db130
[ 70.648142] Flags: nzcv IRQs on FIQs on Mode SVC_32 ISA ARM Segment none
[ 70.648322] Control: 10c5387d Table: 56e6006a DAC: 00000051
[ 70.648483] Process insmod (pid: 64, stack limit = 0x(ptrval))
[ 70.648653] Stack: (0xd6e69db0 to 0xd6e6a000)
[ 70.648855] 9da0:                                c1788000 c1604c48 fffffe00 bf005000
[ 70.649147] 9dc0: 00000000 c1604c48 bf002040 c0302d4c 00000000 c035c10c 00210d00 00000000
[ 70.649414] 9de0: c1604c48 d6dfb280 d6e69de4 6dc64400 00000000 e0c93fff ffe00000 fffff000
[ 70.649675] 9e00: 8040003f d6dfb3c0 dbef4d60 6dc64400 dbef4d60 d6dfb280 bf002040 6dc64400
[ 70.649956] 9e20: bf002040 00000002 d6e599c0 00000002 d6e59900 c03d232c 00000001 c03d4678
[ 70.650225] 9e40: d6e69f30 d6e69f30 00000002 d6e598c0 00000002 c03d4694 bf00204c 00007fff
[ 70.650485] 9e60: bf002040 c03d1584 00000001 c03d0e98 bf002088 bf00110c bf00222c bf002170
[ 70.650745] 9e80: c0f089bc c1356640 c121db9c c121dba8 c121dc00 c1604c48 c1608ec4 d6e0d180
[ 70.651017] 9ea0: fffff000 e0800000 d6e0d180 d6dfb280 00000000 00000000 00000000 00000000
[ 70.651283] 9ec0: 00000000 00000000 6e72656b 00006c65 00000000 00000000 00000000 00000000
[ 70.651544] 9ee0: 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
[ 70.651804] 9f00: 00000000 6dc64400 00000080 00001b20 00000000 e0c92b20 0012cd78 c1604c48
[ 70.652064] 9f20: 0011b1f8 fffffe00 00000051 c03d4ad8 e0c812b6 e0c813c0 e0c81000 00011b20
[ 70.652324] 9f40: e0c923a0 e0c921d4 e0c8e760 00003000 00003040 00000000 00000000 00000000
[ 70.652584] 9f60: 000016f4 0000002d 0000002e 00000018 00000000 00000010 00000000 6dc64400
[ 70.652844] 9f80: 000f411e 0011b1f8 b6f50950 00011b20 00000080 c0301204 d6e68000 00000080
[ 70.653104] 9fa0: 000f411e c0301000 0011b1f8 b6f50950 0011b258 00011b20 0011b1f8 00000000
[ 70.653364] 9fc0: 0011b1f8 b6f50950 00011b20 00000080 00000001 beba6e80 001086c4 000f411e
[ 70.653624] 9fe0: beba6b38 beba6b28 0003b270 b6e0a1b0 600f0010 0011b258 00000000 00000000
[ 70.653884] [<bf0050a0>] (module4_init [module5]) from [<c0302d4c>] (do_one_initcall+0x54/0x208)
[ 70.654144] [<c0302d4c>] (do_one_initcall) from [<c03d232c>] (do_init_module+0x64/0x214)
[ 70.654404] [<c03d232c>] (do_init_module) from [<c03d4694>] (load_module+0x2150/0x243c)
[ 70.654664] [<c03d4694>] (load_module) from [<c03d4ad8>] (sys_init_module+0x158/0x18c)
[ 70.654924] [<c03d4ad8>] (sys_init_module) from [<c0301000>] (ret_fast_syscall+0x0/0x54)
[ 70.655184] Exception stack(0xd6e69fa8 to 0xd6e69ff0)
[ 70.655444] 9fa0:                                0011b1f8 b6f50950 0011b258 00011b20 0011b1f8 00000000
[ 70.655704] 9fc0: 0011b1f8 b6f50950 00011b20 00000080 00000001 beba6e80 001086c4 000f411e
[ 70.655964] 9fe0: beba6b38 beba6b28 0003b270 b6e0a1b0
[ 70.656224] Code: eb51afed e1a05000 eb4ed016 e2866001 (e1c500f0)
[ 70.656484] ---[ end trace 0ec960796a5946e6 ]---
Segmentation fault

```

Як результат виконання objdump отримали таке:

```

        if (!index)
            return ZERO_SIZE_PTR;

        return kmem_cache_alloc_trace(kmalloc_caches[index],
50:  e3008000    movw    r8, #0
54:  e3a0a0c0    mov     sl, #192          ; 0xc0
        pr_info("Hello, world!\n");
58:  e3007000    movw    r7, #0
5c:  e3408000    movt    r8, #0
60:  e340a060    movt    sl, #96 ; 0x60
64:  e3407000    movt    r7, #0
        for (repeat = 0; repeat < repeats; repeat++) {
68:  e3a06000    mov     r6, #0
6c:  e1590006    cmp     r9, r6
70:  0a000017    beq     d4 <init_module+0xd4>
        if (repeat == repeats - 1)
74:  e2493001    sub     r3, r9, #1
78:  e1530006    cmp     r3, r6
            ptr = 0;
7c:  03a05000    moveq   r5, #0
            if (repeat == repeats - 1)
80:  0a000004    beq     98 <init_module+0x98>
84:  e3a02018    mov     r2, #24
88:  e1a0100a    mov     r1, sl
8c:  e5980018    ldr     r0, [r8, #24]
90:  ebfffffe    bl      0 <kmem_cache_alloc_trace>
94:  e1a05000    mov     r5, r0
            ptr->time_before = ktime_get();
98:  ebfffffe    bl      0 <ktime_get>
            for (repeat = 0; repeat < repeats; repeat++) {
9c:  e2866001    add     r6, r6, #1
            ptr->time_before = ktime_get();
a0:  e1c500f0    strd    r0, [r5]
            pr_info("Hello, world!\n");
a4:  e1a00007    mov     r0, r7
a8:  ebfffffe    bl      0 <prinfo>
            ptr->time_after = ktime_get();
ac:  ebfffffe    bl      0 <ktime_get>

```

4. Виконання завдання Basic2:

У функції exit модуля друк вмісту списку змінимо на pr_debug і додамо два виклики pr_debug до та після друку списку.

Для того, аби побачити зміни при завантаженні та вивантаженні модуля необхідно додати #define DEBUG на початку файлу та аби рівень логування був 8, аби виводилися debugповідомлення.

Встановлюємо параметр CONFIG_DYNAMIC_DEBUG у ~/lab3/linux-stable/fragments/bbb.cfg та перезбираємо ядро.

```

# --- Networking ---
CONFIG_BRIDGE=y
CONFIG_DYNAMIC_DEBUG=y

```



```
Activities Terminal tpa 30 21:01 alex@alex-VirtualBox: ~/lab3/linux-stable
alex@alex-VirtualBox:~/lab3/linux-stable$ ./scripts/kconfig/merge_config.sh arch/arm/configs/multi_v7_defconfig fragments/bbb.cfg
Using arch/arm/configs/multi_v7_defconfig as base
Merging fragments/bbb.cfg
Value of CONFIG_AM335X_PHY_USB is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_AM335X_PHY_USB=m
New value: CONFIG_AM335X_PHY_USB=y

Value of CONFIG_USB_MUSB_TUSB6010 is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_USB_MUSB_TUSB6010=m
New value: CONFIG_USB_MUSB_TUSB6010=y

Value of CONFIG_USB_MUSB_OMAP2PLUS is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_USB_MUSB_OMAP2PLUS=m
New value: CONFIG_USB_MUSB_OMAP2PLUS=y

Value of CONFIG_USB_MUSB_HDRC is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_USB_MUSB_HDRC=m
New value: CONFIG_USB_MUSB_HDRC=y

Value of CONFIG_USB_MUSB_DSPS is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_USB_MUSB_DSPS=m
New value: CONFIG_USB_MUSB_DSPS=y

Value of CONFIG_USB_MUSB_AM35X is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_USB_MUSB_AM35X=m
New value: CONFIG_USB_MUSB_AM35X=y

Value of CONFIG_USB_CONFIGFS is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_USB_CONFIGFS=m
New value: CONFIG_USB_CONFIGFS=y

Value of CONFIG_NOP_USB_XCEIV is redefined by fragment fragments/bbb.cfg:
Previous value: CONFIG_NOP_USB_XCEIV=m
New value: CONFIG_NOP_USB_XCEIV=y

scripts/kconfig/conf --alldefconfig Kconfig
#
# configuration written to .config
#
alex@alex-VirtualBox:~/lab3/linux-stable$
```

З'явився каталог `/sys/kernel/debug/dynamic_debug`:

```
alex@alex-VirtualBox:~/lab3/linux-stable$ sudo ls -la /sys/kernel/debug/dynamic_debug
[sudo] password for alex:
total 0
drwxr-xr-x  2 root root 0 tpa 30 20:07 .
drwx----- 36 root root 0 tpa 30 20:07 ..
-rw-r--r--  1 root root 0 tpa 30 20:07 control
alex@alex-VirtualBox:~/lab3/linux-stable$
```

Нижче наведено змінений фрагмент коду:



```
Activities Terminal ▾ Тра 30 21:03
alex@alex-VirtualBox: ~/lab3/b

else
    ptr = kmalloc(sizeof(*ptr), GFP_KERNEL);
    ptr->time_before = ktime_get();
    ptr->time_after = ktime_get();
    pr_info("Hello, world!\n");
    list_add(&ptr->time_list, &lab5_list_head);
}

static unsigned int repeats = 1;

module_param(repeats, uint, 0444);
MODULE_PARM_DESC(repeats, "How many hello to print");

static int __init module5_init(void)
{
    BUG_ON(repeats > 10);
    if (repeats >= 5 && repeats <= 10)
        pr_warn("Repeation from 5 to 10 times\n");
    if (repeats == 0)
        pr_warn("No repeatition\n");
    print_text(repeats);
    return 0;
}

static void __exit module5_exit(void)
{
    struct list_head *p;
    struct list_head *n;
    struct time_keeper *curr;
    pr_info("Module 5 exit\n");
    → pr_debug("Before printing of the list\n");
    list_for_each_safe(p, n, &lab5_list_head) {
        curr = list_entry(p, struct time_keeper, time_list);
        → pr_debug("Time needed for printing is: %lld(ns).\n",
                    curr->time_after - curr->time_before);
        list_del(p);
        kfree(curr);
    }
    → pr_debug("After printing of the list\n");
}

module_init(module5_init);
module_exit(module5_exit);

MODULE_AUTHOR("AlexHoncharenko");
MODULE_DESCRIPTION("Training to debug modules");
MODULE_LICENSE("Dual BSD/GPL");
```

Змінюючи прапорці у `/sys/kernel/debug/dynamic_debug/control` можна змінювати формат повідомлення для всього модулю та для окремих рядків.

```
/lab6 # echo 8 > /proc/sys/kernel/printk
/lab6 # insmod module5.ko repeats=3
[ 153.173200] module5: loading out-of-tree module taints kernel.
[ 153.179912] Hello there!
[ 153.180438] Hello there!
[ 153.180764] Hello there!
/lab6 # cat /sys/kernel/debug/dynamic_debug/control | grep module5
/home/alex/lab3/busybox/_install/lab6/module5.c:59 [module5]module5_exit =p "Before printing of the list\012"
/home/alex/lab3/busybox/_install/lab6/module5.c:63 [module5]module5_exit =p "Time needed for printing is: %lld(ns).\012"
/home/alex/lab3/busybox/_install/lab6/module5.c:67 [module5]module5_exit =p "After printing of the list\012"
/lab6 # echo 'file module5.c line 59 +mf' > /sys/kernel/debug/dynamic_debug/control
/lab6 # echo 'file module5.c line 59 +mf' > /sys/kernel/debug/dynamic_debug/control [ 272.478981] random: fast init done
/lab6 # echo 'file module5.c line 63 -p' > /sys/kernel/debug/dynamic_debug/control
/lab6 # echo 'file module5.c line 67 =pl' > /sys/kernel/debug/dynamic_debug/control
/lab6 # cat /sys/kernel/debug/dynamic_debug/control | grep module5
/home/alex/lab3/busybox/_install/lab6/module5.c:59 [module5]module5_exit =pmf "Before printing of the list\012"
/home/alex/lab3/busybox/_install/lab6/module5.c:63 [module5]module5_exit =_ "Time needed for printing is: %lld(ns).\012"
/home/alex/lab3/busybox/_install/lab6/module5.c:67 [module5]module5_exit =pl "After printing of the list\012"
/lab6 # rmmod module5
[ 339.839005] Module 5 exit
[ 339.839675] module5:module5_exit: Before printing of the list
[ 339.840490] 67: After printing of the list
```