



OPERATING
SYSTEM (CS3431)

LINUX KERNEL MODULES

2.0

26

TABLE OF CONTENTS

01

Creating Makefile

02

Makefile Code

03

Creating File For Kernel Structures

04

Programming Simple.c

05

make Command

06

Loading Kernel Module

07

DMESG Command

08

Removing Kernel Module

INTRODUCTION

In this project, you will learn how to create a kernel module and load it into the Linux kernel. The project can be completed using the Linux virtual machine that is available with this text. Although you may use an editor to write these C programs, you will have to use the terminal application to compile the programs, and you will have to enter commands on the command line to manage the modules in the kernel. As you'll discover, the advantage of developing kernel modules is that it is a relatively easy method of interacting with the kernel, thus allowing you to write programs that directly invoke kernel functions. It is important for you to keep in mind that you are indeed writing kernel code that directly interacts with the kernel. That normally means that any errors in the code could crash the system! However, since you will be using a virtual machine, any failures will at worst only require rebooting the system.

OUR GROUP MEMBERS

Saif-Ur-Rehman (BCS211119)

Roshail Azhar (BCS211130)

Kashan Mazhar (BCS211139)

Submitted to: Sir Siraj Rathore



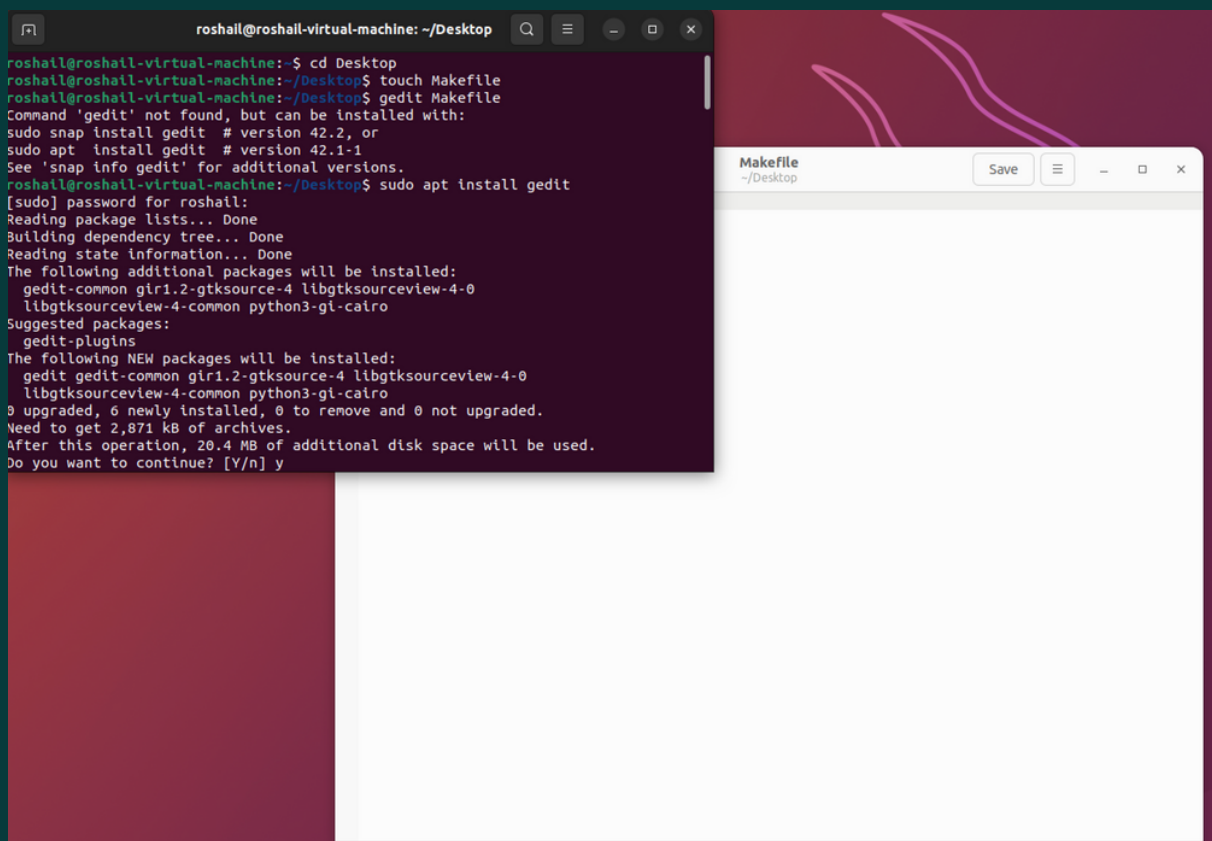
**This Project is
related to kernel
modules and data
structures.**

CREATING KERNEL MODULES & BUILDING KERNEL DATA STRUCTURES

The first part of this project involves following a series of steps for creating and inserting a module into the Linux kernel.

01 — Creating Make File

Go to terminal and type command: `$ touch Makefile`
This will create your Makefile



The screenshot shows a terminal window on the left and a file editor on the right. The terminal window has a title bar that reads "roshail@roshail-virtual-machine: ~/Desktop". The terminal output shows the following commands and their results:

```
roshail@roshail-virtual-machine:~$ cd Desktop
roshail@roshail-virtual-machine:~/Desktop$ touch Makefile
roshail@roshail-virtual-machine:~/Desktop$ gedit Makefile
Command 'gedit' not found, but can be installed with:
sudo snap install gedit # version 42.2, or
sudo apt install gedit # version 42.1-1
See 'snap info gedit' for additional versions.
roshail@roshail-virtual-machine:~/Desktop$ sudo apt install gedit
[sudo] password for roshail:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  gedit-common gir1.2-gtksource-4 libgtksourceview-4-0
  libgtksourceview-4-common python3-gi-cairo
Suggested packages:
  gedit-plugins
The following NEW packages will be installed:
  gedit gedit-common gir1.2-gtksource-4 libgtksourceview-4-0
  libgtksourceview-4-common python3-gi-cairo
0 upgraded, 6 newly installed, 0 to remove and 0 not upgraded.
Need to get 2,871 kB of archives.
After this operation, 20.4 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

The file editor on the right has a title bar that reads "Makefile ~/Desktop". It shows a blank file with a "Save" button and standard window controls.

02 — Make File Code

Code:

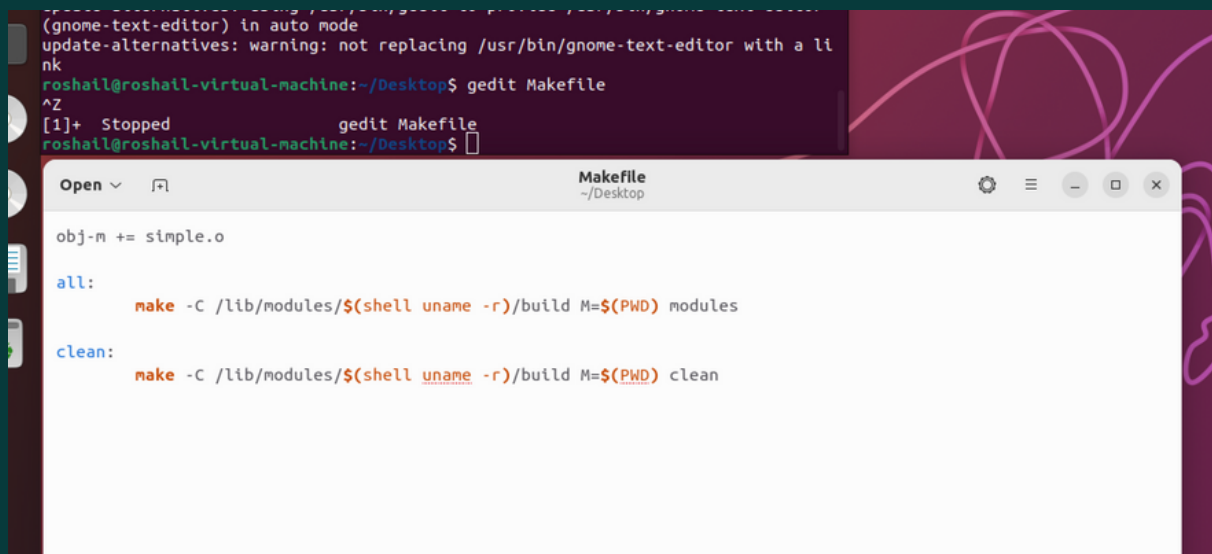
```
obj-m += simple.o
```

```
all:
```

```
make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
```

```
clean:
```

```
make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```



The screenshot shows a terminal window at the top with the following commands and output:

```
(gnome-text-editor) in auto mode
update-alternatives: warning: not replacing /usr/bin/gnome-text-editor with a li
nk
roshail@roshail-virtual-machine:~/Desktop$ gedit Makefile
^Z
[1]+  Stopped                  gedit Makefile
roshail@roshail-virtual-machine:~/Desktop$
```

Below the terminal is a text editor window titled "Makefile" at the path "~/Desktop". It contains the following code:

```
obj-m += simple.o

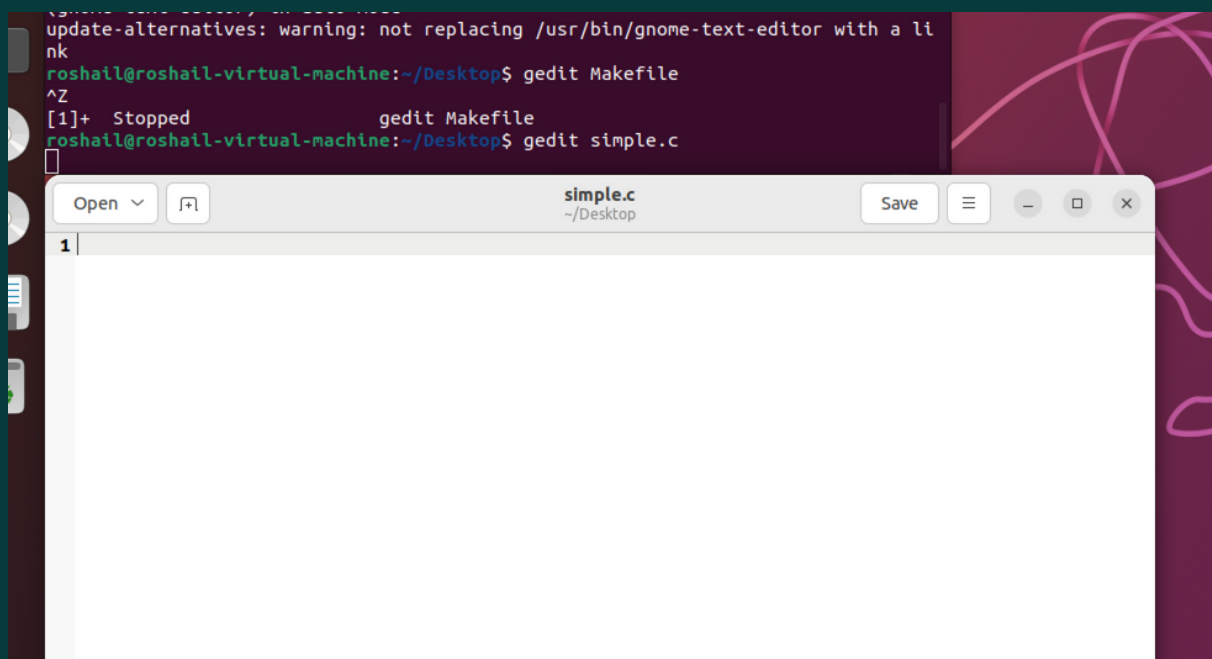
all:
    make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:
    make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

03 — Creating File For Kernel Structure

Go to terminal and type command: `$ gedit simple.c`

This will create your `simple.c` file



The screenshot shows a terminal window at the top with the following commands and output:

```
update-alternatives: warning: not replacing /usr/bin/gnome-text-editor with a li
nk
roshail@roshail-virtual-machine:~/Desktop$ gedit Makefile
^Z
[1]+  Stopped                  gedit Makefile
roshail@roshail-virtual-machine:~/Desktop$ gedit simple.c

```

Below the terminal is a text editor window titled "simple.c" at the path "~/Desktop". It is currently empty, with the cursor at line 1.

04 — Programming Simple.c

This part of this project involves modifying the kernel module so that it uses the kernel linked-list data structure.

```
update-alternatives: warning: not replacing /usr/bin/gnome-text-editor with a li
nk
roshail@roshail-virtual-machine:~/Desktop$ gedit Makefile
^Z
[1]+  Stopped                  gedit Makefile
roshail@roshail-virtual-machine:~/Desktop$ gedit simple.c
[1]

simple.c
~/Desktop
Save

1 #include <linux/init.h>
2 #include <linux/module.h>
3 #include <linux/kernel.h>
4 #include <linux/list.h>
5 #include <linux/slab.h>
6
7 struct birthday
8 {
9     int month;
10    int day;
11    int year;
12    char *name;
13    struct list_head list;
14 };
15
16 /**
17  * The following defines and initializes a list_head object named birthday_list
18  */
19 static LIST_HEAD(birthday_list);
20 struct birthday *person;
21 struct birthday *next;
22
23
24 int simple_init(void)
```

```
Setting 24 int simple_init(void)
Setting 25 {
Setting 26 printk(KERN_INFO "****Loading Module***\n");
Process 27
Process 28 person = kmalloc(sizeof(*person), GFP_KERNEL);
Process 29 person->name = "Saif";
Process 30 person->day = 29;
Process 31 person->month = 7;
Process 32 person->year = 2001;
Process 33 INIT_LIST_HEAD(&person->list);
Setting 34 list_add_tail(&person->list, &birthday_list);
update-35
gnome-36 person = kmalloc(sizeof(*person), GFP_KERNEL);
update-37 person->name = "Roshail";
nk      38 person->day = 8;
roshail 39 person->month = 11;
^Z      40 person->year = 2000;
[1]+ St 41 INIT_LIST_HEAD(&person->list);
roshail 42 list_add_tail(&person->list, &birthday_list);
[1]
43
44 person = kmalloc(sizeof(*person), GFP_KERNEL);
45 person->name = "Zoha";
46 person->day = 10;
47 person->month = 12;
48 person->year = 1998;
49 INIT_LIST_HEAD(&person->list);
50 list_add_tail(&person->list, &birthday_list);
51
52 person = kmalloc(sizeof(*person), GFP_KERNEL);
53 person->name = "Kashan Sunny Mazhar";
54 person->day = 13;
55 person->month = 4;
56 person->year = 2001;
57 INIT_LIST_HEAD(&person->list);
58 list_add_tail(&person->list, &birthday_list);
59
60 person = kmalloc(sizeof(*person), GFP_KERNEL);
61 person->name = "Sufyan";
62 person->day = 23;
63 person->month = 1;
64 person->year = 1999;
65 INIT_LIST_HEAD(&person->list);
66 list_add_tail(&person->list, &birthday_list);
67
68 person = kmalloc(sizeof(*person), GFP_KERNEL);
```

04 — Continued

```
update-1 67
rk        68 person = kmalloc(sizeof(*person), GFP_KERNEL);
-oshall  69 person->name = "Hanza Slave";
vZ        70 person->day = 23;
[1]+ St   71 person->month = 1;
-oshall  72 person->year = 2001;
]         73 INIT_LIST_HEAD(&person->list);
         74 list_add_tail(&person->list, &birthday_list);
         75
         76 list_for_each_entry(person, &birthday_list, list) {
         77 printk(KERN_INFO "Name: %s Birthday: Month: %d Day: %d Year: %d", person->name, person->month, person->day, person->year);
         78 }
         79
         80 return 0;
         81 }
         82
         83 void simple_exit(void)
         84 {
         85 printk(KERN_INFO "****Removing Module***\n");
         86 list_for_each_entry_safe(person, next, &birthday_list, list) {
         87 printk(KERN_INFO "Removing %s %d %d %d", person->name, person->month, person->day, person->year);
         88 list_del(&person->list);
         89 kfree(person);
         90 }
         91 }
         92
         93 module_init(simple_init);
         94 module_exit(simple_exit);
         95
         96 MODULE_LICENSE("GPL");
         97 MODULE_DESCRIPTION("Simple Module");
         98 MODULE_AUTHOR("SGG");
```

C Tab Width: 8 Ln 48, Col 20 INS

Code:

```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/list.h>
#include <linux/slab.h>
```

```
struct birthday
{
    int month;
    int day;
    int year;
    char *name;
    struct list_head list;
};
```

```
/**
 * The following defines and initializes a list_head object named
 * birthday_list
 */
static LIST_HEAD(birthday_list);
struct birthday *person;
struct birthday *next;
```


04 — Continued

```
int simple_init(void)
{
    printk(KERN_INFO "***Loading Module**\n");

    person = kmalloc(sizeof(*person), GFP_KERNEL);
    person->name = "Saif";
    person->day = 29;
    person->month = 7;
    person->year = 2001;
    INIT_LIST_HEAD(&person->list);
    list_add_tail(&person->list, &birthday_list);

    person = kmalloc(sizeof(*person), GFP_KERNEL);
    person->name = "Kashan Mazhar";
    person->day = 11;
    person->month = 3;
    person->year = 2000;
    INIT_LIST_HEAD(&person->list);
    list_add_tail(&person->list, &birthday_list);

    person = kmalloc(sizeof(*person), GFP_KERNEL);
    person->name = "Roshail";
    person->day = 18;
    person->month = 12;
    person->year = 2001;
    INIT_LIST_HEAD(&person->list);
    list_add_tail(&person->list, &birthday_list);

    person = kmalloc(sizeof(*person), GFP_KERNEL);
    person->name = "Zoha";
    person->day = 13;
    person->month = 4;
    person->year = 1999;
    INIT_LIST_HEAD(&person->list);
    list_add_tail(&person->list, &birthday_list);

    person = kmalloc(sizeof(*person), GFP_KERNEL);
    person->name = "Sufyan";
    person->day = 23;
    person->month = 1;
    person->year = 2002;
    INIT_LIST_HEAD(&person->list);
    list_add_tail(&person->list, &birthday_list);
```

04 — Continued

```
list_for_each_entry(person, &birthday_list, list) {
    printk(KERN_INFO "Name: %s Birthday: Month: %d Day: %d Year:
%d", person->name, person->month, person->day, person->year);
}

return 0;
}

void simple_exit(void)
{
    printk(KERN_INFO "***Removing Module**\n");
    list_for_each_entry_safe(person, next, &birthday_list, list) {
        printk(KERN_INFO "Removing %s %d %d %d", person->name, person-
>month, person->day, person->year);
        list_del(&person->list);
        kfree(person);
    }
}

module_init(simple_init);
module_exit(simple_exit);

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Simple Module");
MODULE_AUTHOR("SGG");
```

04 — Continued

Code Synopsis:

Initially, we define a struct containing the elements that are to be inserted in the linked list. The following C struct defines birthdays:

```
struct birthday { int day; int month; int year; struct list head list; }
```

Inserting Elements into the Linked List

We declare a list head object, which we use as a reference to the head of the list by using the LIST_HEAD() macro

```
static LIST_HEAD(birthday list);
```

This macro defines and initializes the variable birthday list, which is of type struct list_head.

We create and initialize instances of struct birthday as follows:

```
struct birthday *person;  
  
person = kmalloc(sizeof(*person), GFP_KERNEL);  
person->day = 2; person->month = 8;  
person->year = 1995;  
INIT_LIST_HEAD(&person->list);
```

Traversing the Linked List:

- A pointer to the structure being iterated over
- A pointer to the head of the list being iterated over
- The name of the variable containing the list head structure

Removing Elements from the Linked List

Removing elements from the list involves using the list_del() macro, which is passed a pointer to struct list_head

```
list_del(struct list_head *element)
```

This removes element from the list while maintaining the structure of the remainder of the list

04 — Continued

Code Synopsis Continued:

Additional Kernel Structure Requirements:

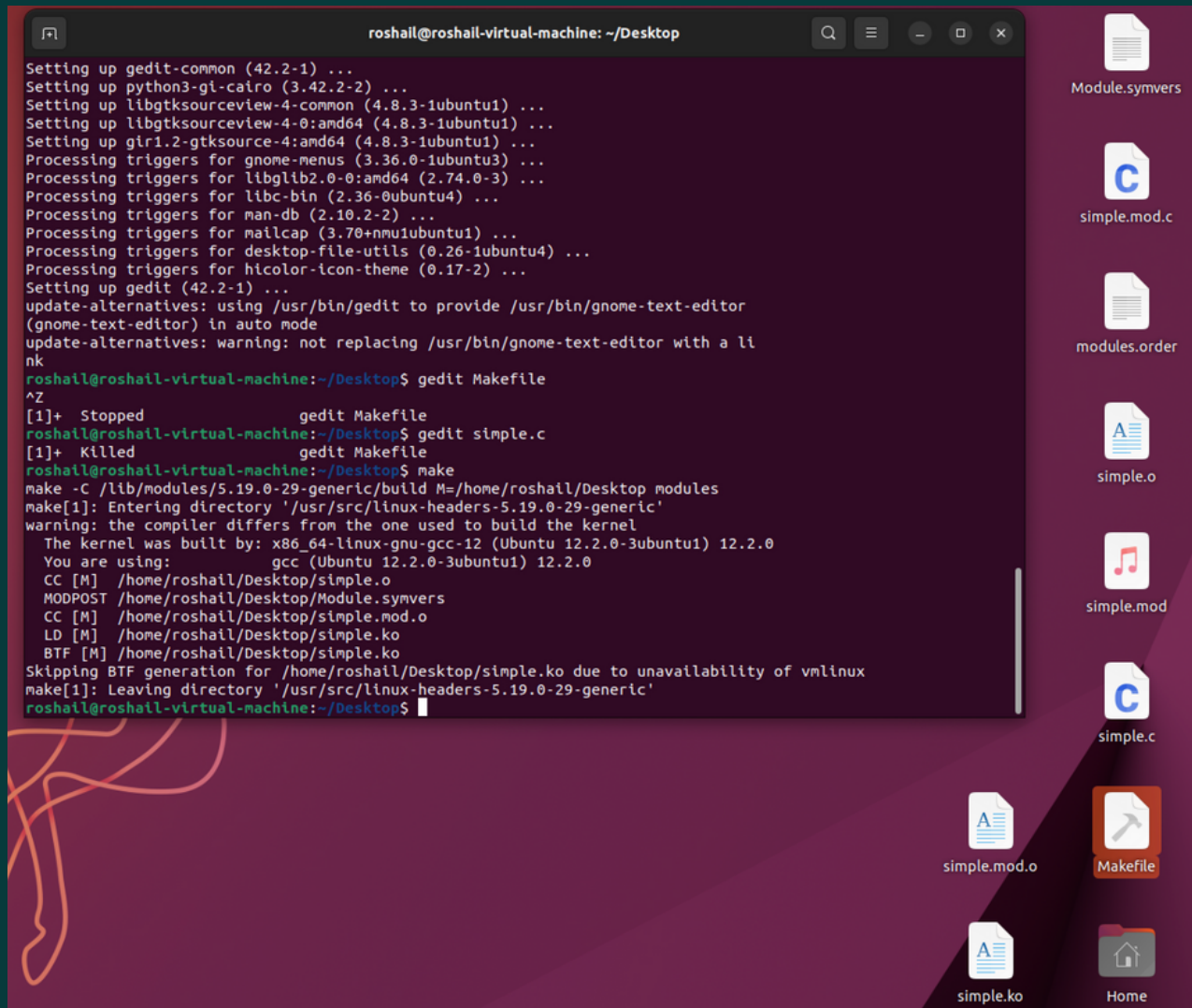
In the module entry point, we created a linked list containing five struct birthday elements. Traverse the linked list and output its contents to the kernel log buffer. Invoke the dmesg command to ensure the list is properly constructed once the kernel module has been loaded. In the module exit point, delete the elements from the linked list and return the free memory back to the kernel. Again, invoke the dmesg command to check that the list has been removed once the kernel module has been unloaded.

```
roshail@roshail-virtual-machine:~/Desktop$ sudo rmmod simple
roshail@roshail-virtual-machine:~/Desktop$ dmesg
[ 4913.717122] Removing Hamza Slave 1 23 2001
[ 4936.958704] ***Loading Module***
[ 4936.958708] Name: Saif Birthday: Month: 7 Day: 29 Year: 2001
[ 4936.958711] Name: Roshail Birthday: Month: 11 Day: 8 Year: 2000
[ 4936.958713] Name: Zoha Birthday: Month: 12 Day: 18 Year: 1998
[ 4936.958714] Name: Kashan Sunny Mazhar Birthday: Month: 4 Day: 13 Year: 2001
[ 4936.958716] Name: Sufyan Birthday: Month: 1 Day: 23 Year: 1999
[ 4936.958717] Name: Hamza Slave Birthday: Month: 1 Day: 23 Year: 2001
[ 4992.756014] ***Removing Module***
[ 4992.756018] Removing Saif 7 29 2001
[ 4992.756021] Removing Roshail 11 8 2000
[ 4992.756023] Removing Zoha 12 18 1998
[ 4992.756024] Removing Kashan Sunny Mazhar 4 13 2001
[ 4992.756026] Removing Sufyan 1 23 1999
roshail@roshail-virtual-machine:~/Desktop$
```

*Please Refer to Heading 8 for the complete understanding of the above image

05 — make Command

The compilation produces several files. The file `simple.ko` represents the compiled kernel module. The following step illustrates inserting this module into the Linux kernel.



The screenshot shows a terminal window titled "roshail@roshail-virtual-machine: ~/Desktop" with the following output:

```
Setting up gedit-common (42.2-1) ...
Setting up python3-gi-cairo (3.42.2-2) ...
Setting up libgtksourceview-4-common (4.8.3-1ubuntu1) ...
Setting up libgtksourceview-4-0:amd64 (4.8.3-1ubuntu1) ...
Setting up gir1.2-gtksource-4:amd64 (4.8.3-1ubuntu1) ...
Processing triggers for gnome-menus (3.36.0-1ubuntu3) ...
Processing triggers for libglib2.0-0:amd64 (2.74.0-3) ...
Processing triggers for libc-bin (2.36-0ubuntu4) ...
Processing triggers for man-db (2.10.2-2) ...
Processing triggers for mailcap (3.70+nmu1ubuntu1) ...
Processing triggers for desktop-file-utils (0.26-1ubuntu4) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Setting up gedit (42.2-1) ...
update-alternatives: using /usr/bin/gedit to provide /usr/bin/gnome-text-editor
(gnome-text-editor) in auto mode
update-alternatives: warning: not replacing /usr/bin/gnome-text-editor with a li
nk
roshail@roshail-virtual-machine:~/Desktop$ gedit Makefile
^Z
[1]+  Stopped                  gedit Makefile
roshail@roshail-virtual-machine:~/Desktop$ gedit simple.c
[1]+  Killed                   gedit Makefile
roshail@roshail-virtual-machine:~/Desktop$ make
make -C /lib/modules/5.19.0-29-generic/build M=/home/roshail/Desktop modules
make[1]: Entering directory '/usr/src/linux-headers-5.19.0-29-generic'
warning: the compiler differs from the one used to build the kernel
The kernel was built by: x86_64-linux-gnu-gcc-12 (Ubuntu 12.2.0-3ubuntu1) 12.2.0
You are using:          gcc (Ubuntu 12.2.0-3ubuntu1) 12.2.0
CC [M] /home/roshail/Desktop/simple.o
MODPOST /home/roshail/Desktop/Module.symvers
CC [M] /home/roshail/Desktop/simple.mod.o
LD [M] /home/roshail/Desktop/simple.ko
BTF [M] /home/roshail/Desktop/simple.ko
Skipping BTF generation for /home/roshail/Desktop/simple.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-5.19.0-29-generic'
roshail@roshail-virtual-machine:~/Desktop$
```

The desktop environment shows several files and folders: `Module.symvers`, `simple.mod.c`, `modules.order`, `simple.o`, `simple.mod`, `simple.c`, `simple.mod.o`, `simple.ko`, `Makefile`, and a `Home` folder. A red scribble is visible on the left side of the desktop background.

06 — Loading Kernel Module

Kernel module loaded using the insmod command, which is run as follows:

To check whether the module has loaded, enter the lsmod command and search for the module simple. Recall that the module entry point is invoked when the module is inserted into the kernel

```
roshail@roshail-virtual-machine:~/Desktop$ sudo insmod simple.ko
roshail@roshail-virtual-machine:~/Desktop$ lsmod
```

Module	Size	Used by
simple	16384	0
isofs	53248	2
bnep	28672	2
intel_rapl_msr	20480	0
intel_rapl_common	40960	1 intel_rapl_msr
crct10dif_pclmul	16384	1
ghash_clmulni_intel	16384	0
btush	61440	0

07 — dmesg

To check the contents of this message in the kernel log buffer, enter the command

```
roshail@roshail-virtual-machine:~/Desktop$
roshail@roshail-virtual-machine:~/Desktop$ sudo insmod simple.ko
roshail@roshail-virtual-machine:~/Desktop$ dmesg
```

```
[ 4913.717122] Removing Hamza Slave 1 23 2001
[ 4936.958704] ***Loading Module***
[ 4936.958708] Name: Saif Birthday: Month: 7 Day: 29 Year: 2001
[ 4936.958711] Name: Roshail Birthday: Month: 11 Day: 8 Year: 2000
[ 4936.958713] Name: Zoha Birthday: Month: 12 Day: 18 Year: 1998
[ 4936.958714] Name: Kashan Sunny Mazhar Birthday: Month: 4 Day: 13 Year: 2001
[ 4936.958716] Name: Sufyan Birthday: Month: 1 Day: 23 Year: 1999
roshail@roshail-virtual-machine:~/Desktop$
```

08 — Removing Module

Removing the kernel module involves invoking the `rmmod` command

Be sure to check with the `dmesg` command to ensure the module has been removed. Because the kernel log buffer can fill up quickly, it often makes sense to clear the buffer periodically. This can be accomplished as follows:

```
sudo dmesg -c
```

```
roshail@roshail-virtual-machine:~/Desktop$ sudo rmmod simple
roshail@roshail-virtual-machine:~/Desktop$ dmesg
[ 4913.717122] Removing Hamza Slave 1 23 2001
[ 4936.958704] ***Loading Module***
[ 4936.958708] Name: Saif Birthday: Month: 7 Day: 29 Year: 2001
[ 4936.958711] Name: Roshail Birthday: Month: 11 Day: 8 Year: 2000
[ 4936.958713] Name: Zoha Birthday: Month: 12 Day: 18 Year: 1998
[ 4936.958714] Name: Kashan Sunny Mazhar Birthday: Month: 4 Day: 13 Year: 2001
[ 4936.958716] Name: Sufyan Birthday: Month: 1 Day: 23 Year: 1999
[ 4936.958717] Name: Hamza Slave Birthday: Month: 1 Day: 23 Year: 2001
[ 4992.756014] ***Removing Module***
[ 4992.756018] Removing Saif 7 29 2001
[ 4992.756021] Removing Roshail 11 8 2000
[ 4992.756023] Removing Zoha 12 18 1998
[ 4992.756024] Removing Kashan Sunny Mazhar 4 13 2001
[ 4992.756026] Removing Sufyan 1 23 1999
roshail@roshail-virtual-machine:~/Desktop$
```