Report

Video link:
https://youtu.be/X0HfHoXzptA

Question:

1. What is a static kernel module? What is a dynamic kernel module? What is the other name of a dynamic kernel module? What are the differences between system calls and dynamic kernel modules (mention at least 3)?

   Static kernel module: Kernel modules compiled as base kernel

   Dynamic kernel module(Loadable kernel module): Kernel modules compiled separately and loaded as users demand

   Difference:

   - Necessity to rebuild the whole kernel
   - If available at anytime or not
   - Necessity to reboot after loading in the kernel

2. What are the commands **insmod**, **rmmod** and **modinfo** for? How do you use them? (Write how would you use them with a module named **dummyModule.ko**).

   Insmod: install the module into the kernel(ex, sudo insmod dummyModule.ko)

   Rmmod:unload the module from the kernel(ex, sudo rmmod dummyModule)

   Modinfo:show the information of the module(including author, license, parameters…)(ex, sudo modinfo dummyModule.ko)

3. Write the usage (parameters, what data type they are and what do they do) of the following commands:

   A. module_init

      function to be run at kernel boot time or module insertion

   B. module_exit

      function to be run at kernel shutdown or module unloading

   C. MODULE_LICENSE

      Will place the license in the .ko file, to give authority to the kernel modification, or will return “kernel tainted”

   D. module_param

      module_param(paramname, datatype, permission)

      to input the information of the parameters while installing the module

   E. MODULE_PARM_DESC

To explain the parameter

4. What do the following terminal commands mean (explain what they do and what does the -x mean in each case):
   A. cat

      concatenate, allow us to create single, multiple files, view contain of file, concatenate files and redirect output in terminal or files

   B. ls -l

      show all the files under the current folder

      -l, with more information, including size, created date, owner, and group it belong to

   C. dmesg -wH

      show information in kernel space

      -w, follow, wait at the state to keep printing out information in kernel space

      -H, enable user-friendly features

   D. lsmod

      show the list of modules in the kernel

   E. lsmod | grep

      with the specific module name typed after grep can directly show the module and its info

5. There is a 0644 in the line

   module_param(studentId, int, 0644);

   inside **paramsModule.c** (Section 1.3). What does 0644 mean?

   Right to access the corresponding files in sysfs,

6. What happens if the initialization function of the module returns -1? What type of error do you get?

   The program cancels loading the module and stops executing

   Error: operation not permitted

7. In Section 1.3 – step 6, **modinfo** shows the information of some variables inside the module but two of them are not displayed. Why is it?

   As the two variables, dummyStudentId, dummySecretValue, don’t input themselves through module_param into the kernel while installing module, nor do they explain themselves with MODULE_PARM_DESC

8. What is the /sys/module folder for?

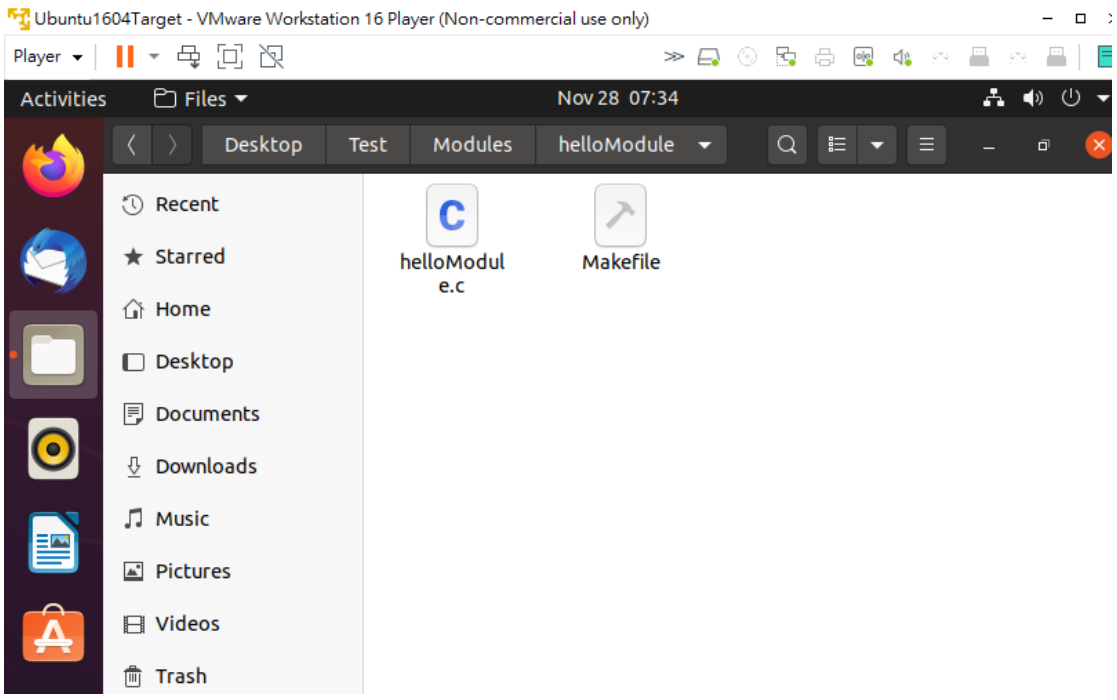
   The folder consists of the modules in the kernel, all dynamic modules will be in here, but static modules will only be in here if it has a version or at least one parameter

9. In Section 1.3 (paramsModule.c), the variable **charparameter** is of type **charp**. What is charp?

   It’s the special datatype for kernel parameter, assigned to string datatype

Screenshot 1:

Save the c program and the Makefile in the same folder, so later when we call “make” command we can execute the command stored in Makefile

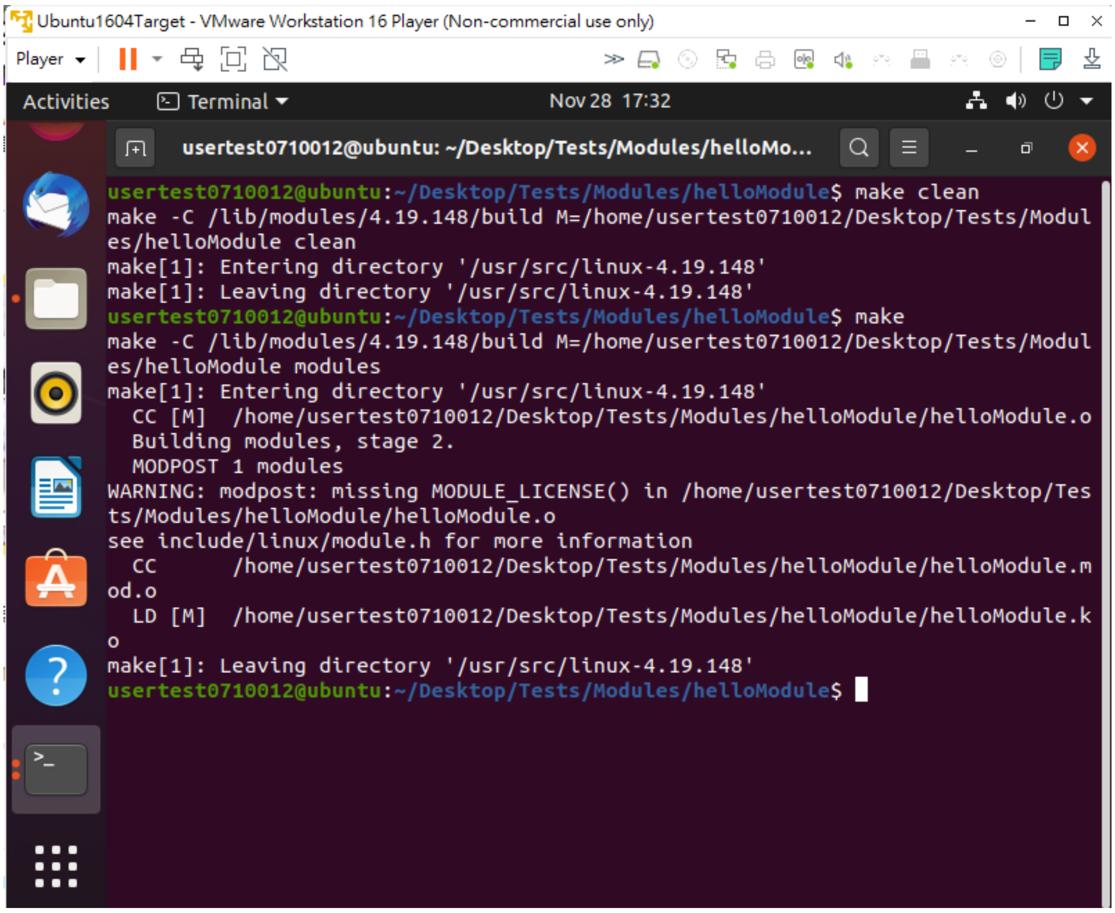


Screenshot 2:

Call “make clean” to delete the former .exe and other file, to prepare the right environment for make

Call “make” to execute the Makefile and compile the c program into .o file, and to .exe

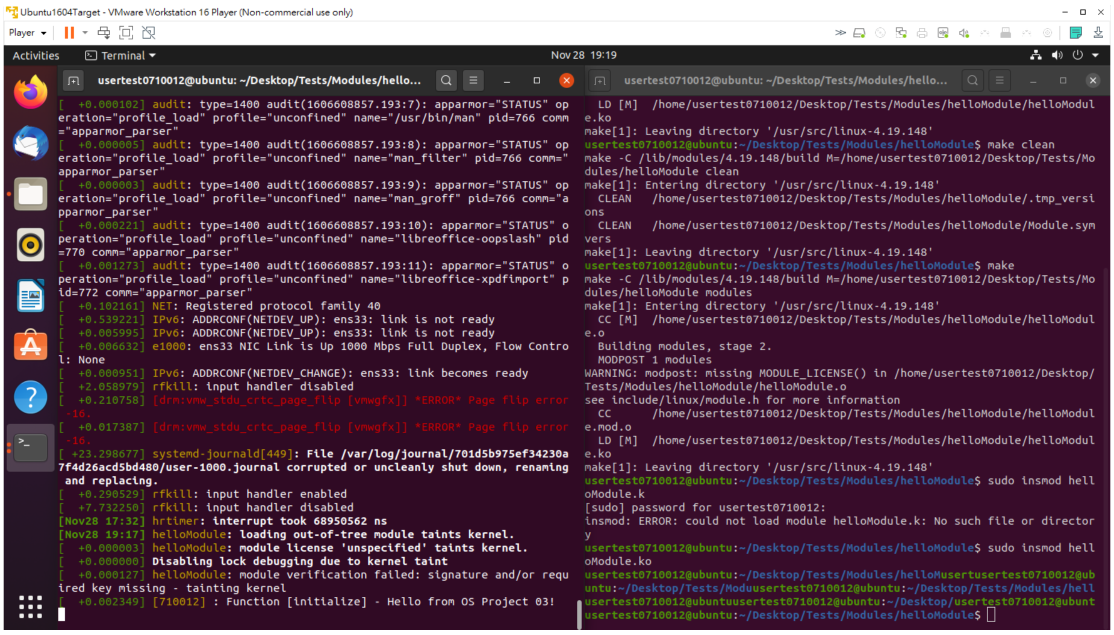
The module will be loaded into the kernel after we run the mentioned command



```
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ make clean
make -C /lib/modules/4.19.148/build M=/home/usertest0710012/Desktop/Tests/Modul
es/helloModule clean
make[1]: Entering directory '/usr/src/linux-4.19.148'
make[1]: Leaving directory '/usr/src/linux-4.19.148'
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ make
make -C /lib/modules/4.19.148/build M=/home/usertest0710012/Desktop/Tests/Modul
es/helloModule modules
make[1]: Entering directory '/usr/src/linux-4.19.148'
  CC [M]  /home/usertest0710012/Desktop/Tests/Modules/helloModule/helloModule.o
  Building modules, stage 2.
  MODPOST 1 modules
WARNING: modpost: missing MODULE_LICENSE() in /home/usertest0710012/Desktop/Tes
ts/Modules/helloModule/helloModule.o
see include/linux/module.h for more information
  CC      /home/usertest0710012/Desktop/Tests/Modules/helloModule/helloModule.m
od.o
  LD [M]  /home/usertest0710012/Desktop/Tests/Modules/helloModule/helloModule.k
o
make[1]: Leaving directory '/usr/src/linux-4.19.148'
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ 
```

Screenshot 3:

When we install the module into the kernel, it will run the initialization in the c program, and print out the message written into the kernel space

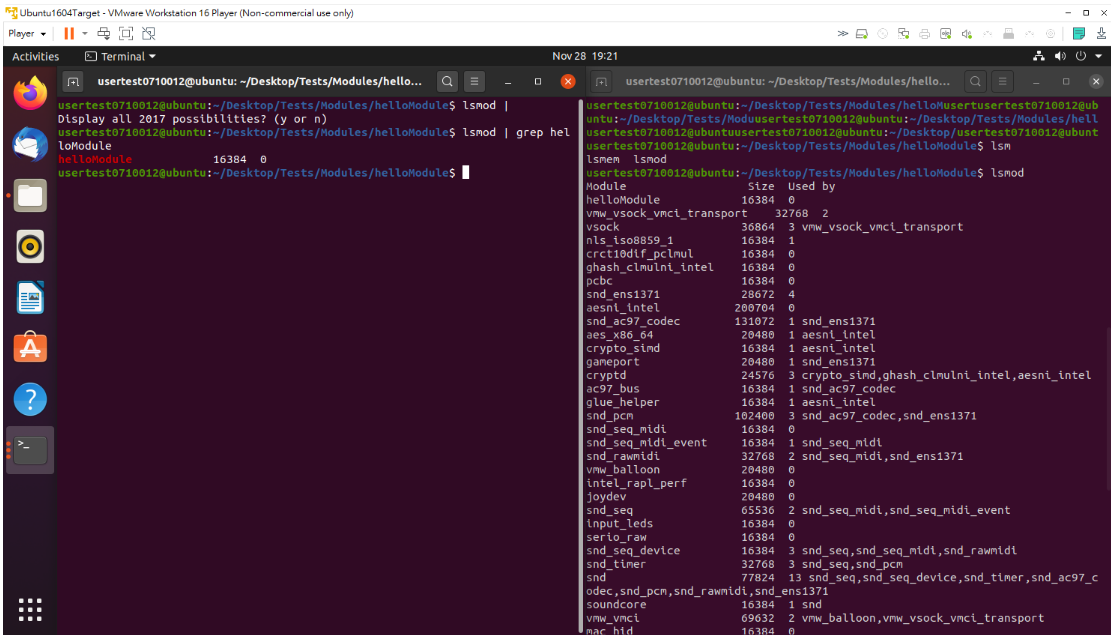


```
[  +0.000102] audit: type=1400 audit(1606608857.193:7): apparmor="STATUS" op
eration="profile_load" profile="unconfined" name="/usr/bin/man" pid=766 comm
="apparmor_parser"
[  +0.000005] audit: type=1400 audit(1606608857.193:8): apparmor="STATUS" op
eration="profile_load" profile="unconfined" name="man_filter" pid=766 comm="
apparmor_parser"
[  +0.000003] audit: type=1400 audit(1606608857.193:9): apparmor="STATUS" op
eration="profile_load" profile="unconfined" name="man_groff" pid=766 comm="a
pparmor_parser"
[  +0.000221] audit: type=1400 audit(1606608857.193:10): apparmor="STATUS" o
peration="profile_load" profile="unconfined" name="libreoffice-oopslash" pid
=770 comm="apparmor_parser"
[  +0.001273] audit: type=1400 audit(1606608857.193:11): apparmor="STATUS" o
peration="profile_load" profile="unconfined" name="libreoffice-xpdfimport" p
id=772 comm="apparmor_parser"
[  +0.102161] NET: Registered protocol family 40
[  +0.539221] IPv6: ADDRCONF(NETDEV_UP): ens33: link is not ready
[  +0.005995] IPv6: ADDRCONF(NETDEV_UP): ens33: link is not ready
[  +0.006632] e1000: ens33 NIC Link is Up 1000 Mbps Full Duplex, Flow Contro
l: None
[  +0.000951] IPv6: ADDRCONF(NETDEV_CHANGE): ens33: link becomes ready
[  +2.058979] rfkill: input handler disabled
[  +0.210758] [drm:vmw_stdu_crtc_page_flip [vmwgfx]] *ERROR* Page flip error
 -16.
[  +0.017387] [drm:vmw_stdu_crtc_page_flip [vmwgfx]] *ERROR* Page flip error
 -16.
[ +23.298677] systemd-journald[449]: File /var/log/journal/701d5b975ef34230a
7f4d26acd5bd480/user-1000.journal corrupted or uncleanly shut down, renaming
 and replacing.
[  +0.290529] rfkill: input handler enabled
[  +7.732250] rfkill: input handler disabled
[Nov28 17:32] hrtimer: interrupt took 68950562 ns
[Nov28 19:17] helloModule: loading out-of-tree module taints kernel.
[  +0.000003] helloModule: module license 'unspecified' taints kernel.
[  +0.000000] Disabling lock debugging due to kernel taint
[  +0.000127] helloModule: module verification failed: signature and/or requ
ired key missing - tainting kernel
[  +0.002349] [710012] : Function [initialize] - Hello from OS Project 03!
```

```
  LD [M]  /home/usertest0710012/Desktop/Tests/Modules/helloModule/helloModul
e.ko
make[1]: Leaving directory '/usr/src/linux-4.19.148'
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ make clean
make -C /lib/modules/4.19.148/build M=/home/usertest0710012/Desktop/Tests/Mo
dules/helloModule clean
make[1]: Entering directory '/usr/src/linux-4.19.148'
  CLEAN   /home/usertest0710012/Desktop/Tests/Modules/helloModule/.tmp_versi
ons
  CLEAN   /home/usertest0710012/Desktop/Tests/Modules/helloModule/Module.sym
vers
make[1]: Leaving directory '/usr/src/linux-4.19.148'
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ make
make -C /lib/modules/4.19.148/build M=/home/usertest0710012/Desktop/Tests/Mo
dules/helloModule modules
make[1]: Entering directory '/usr/src/linux-4.19.148'
  CC [M]  /home/usertest0710012/Desktop/Tests/Modules/helloModule/helloModul
e.o
  Building modules, stage 2.
  MODPOST 1 modules
WARNING: modpost: missing MODULE_LICENSE() in /home/usertest0710012/Desktop/
Tests/Modules/helloModule/helloModule.o
see include/linux/module.h for more information
  CC      /home/usertest0710012/Desktop/Tests/Modules/helloModule/helloModul
e.mod.o
  LD [M]  /home/usertest0710012/Desktop/Tests/Modules/helloModule/helloModul
e.ko
make[1]: Leaving directory '/usr/src/linux-4.19.148'
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ sudo insmod hell
oModule.k
[sudo] password for usertest0710012:
insmod: ERROR: could not load module helloModule.k: No such file or director
y
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ sudo insmod hell
oModule.ko
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloMusertusertest0710012@ub
untu:~/Desktop/Tests/Moduusertest0710012@ubuntu:~/Desktop/Tests/Modules/hell
usertest0710012@ubuntuusertest0710012@ubuntu:~/Desktop/usertest0710012@ubunt
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ 
```

Screenshot 4:

After we installed the module into the kernel, we can find the module in the module list(can directly find the module and its info by calling “| grep and the module’s name”)

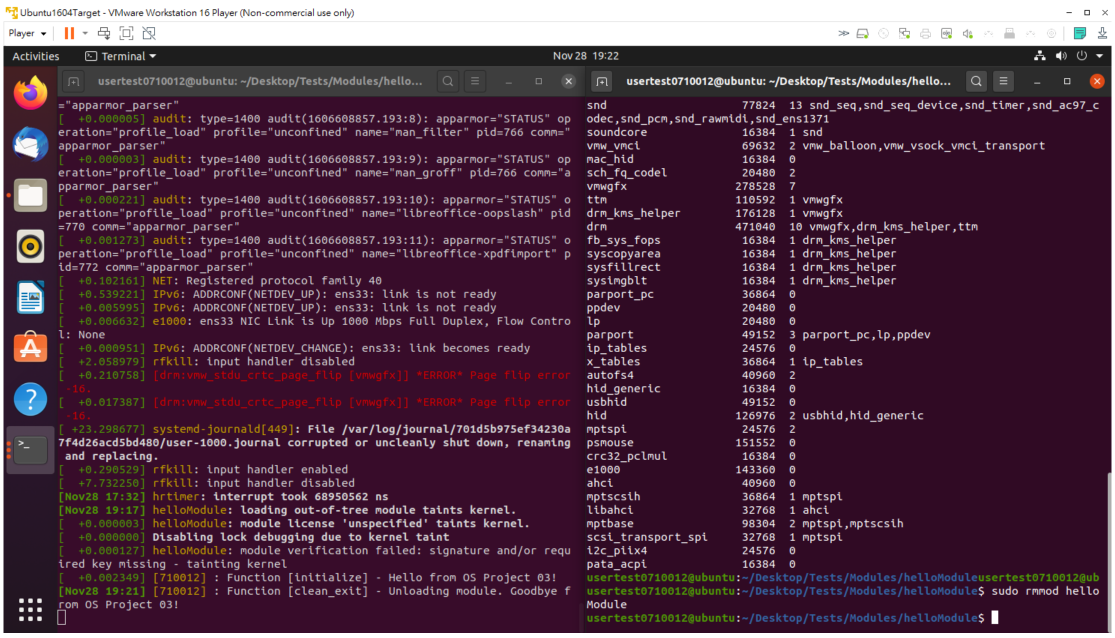


```
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ lsmod |
Display all 2017 possibilities? (y or n)
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ lsmod | grep hel
loModule
helloModule            16384  0
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ 
```

```
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloMusertusertest0710012@ub
untu:~/Desktop/Tests/Moduusertest0710012@ubuntu:~/Desktop/Tests/Modules/hell
usertest0710012@ubuntuusertest0710012@ubuntu:~/Desktop/usertest0710012@ubunt
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ lsm
lsmem  lsmod
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ lsmod
Module                  Size  Used by
helloModule            16384  0
vmw_vsock_vmci_transport    32768  2
vsock                  36864  3 vmw_vsock_vmci_transport
nls_iso8859_1          16384  1
crct10dif_pclmul       16384  0
ghash_clmulni_intel    16384  0
pcbc                   16384  0
snd_ens1371            28672  4
aesni_intel           200704  0
snd_ac97_codec        131072  1 snd_ens1371
aes_x86_64             20480  1 aesni_intel
crypto_simd            16384  1 aesni_intel
gameport               20480  1 snd_ens1371
cryptd                 24576  3 crypto_simd,ghash_clmulni_intel,aesni_intel
ac97_bus               16384  1 snd_ac97_codec
glue_helper            16384  1 aesni_intel
snd_pcm               102400  3 snd_ac97_codec,snd_ens1371
snd_seq_midi           16384  0
snd_seq_midi_event     16384  1 snd_seq_midi
snd_rawmidi            32768  2 snd_seq_midi,snd_ens1371
vmw_balloon            20480  0
intel_rapl_perf        16384  0
joydev                 20480  0
snd_seq                65536  2 snd_seq_midi,snd_seq_midi_event
input_leds             16384  0
serio_raw              16384  0
snd_seq_device         16384  3 snd_seq,snd_seq_midi,snd_rawmidi
snd_timer              32768  3 snd_seq,snd_pcm
snd                    77824  13 snd_seq,snd_seq_device,snd_timer,snd_ac97_c
odec,snd_pcm,snd_rawmidi,snd_ens1371
soundcore              16384  1 snd
vmw_vmci               69632  2 vmw_balloon,vmw_vsock_vmci_transport
mac_hid                16384  0
```

Screenshot 5:

After we unload the module from the kernel, the c program will automatically run the clean_exit function, and print out the message in the kernel space

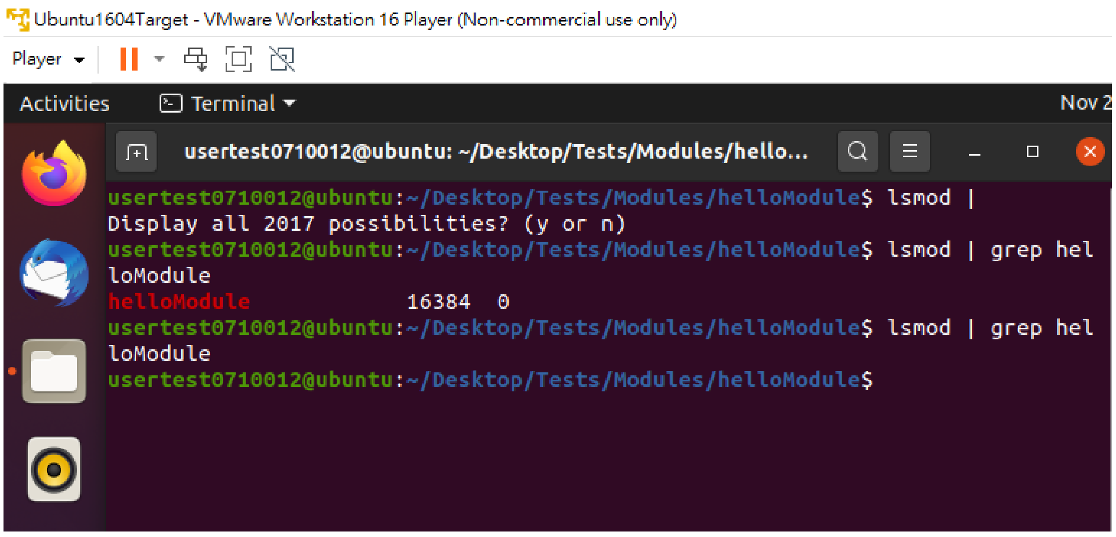


```
="apparmor_parser"
[  +0.000005] audit: type=1400 audit(1606608857.193:8): apparmor="STATUS" op
eration="profile_load" profile="unconfined" name="man_filter" pid=766 comm="
apparmor_parser"
[  +0.000003] audit: type=1400 audit(1606608857.193:9): apparmor="STATUS" op
eration="profile_load" profile="unconfined" name="man_groff" pid=766 comm="a
pparmor_parser"
[  +0.000221] audit: type=1400 audit(1606608857.193:10): apparmor="STATUS" o
peration="profile_load" profile="unconfined" name="libreoffice-oopslash" pid
=770 comm="apparmor_parser"
[  +0.001273] audit: type=1400 audit(1606608857.193:11): apparmor="STATUS" o
peration="profile_load" profile="unconfined" name="libreoffice-xpdfimport" p
id=772 comm="apparmor_parser"
[  +0.102161] NET: Registered protocol family 40
[  +0.539221] IPv6: ADDRCONF(NETDEV_UP): ens33: link is not ready
[  +0.005995] IPv6: ADDRCONF(NETDEV_UP): ens33: link is not ready
[  +0.006632] e1000: ens33 NIC Link is Up 1000 Mbps Full Duplex, Flow Contro
l: None
[  +0.000951] IPv6: ADDRCONF(NETDEV_CHANGE): ens33: link becomes ready
[  +2.058979] rfkill: input handler disabled
[  +0.210758] [drm:vmw_stdu_crtc_page_flip [vmwgfx]] *ERROR* Page flip error
 -16.
[  +0.017387] [drm:vmw_stdu_crtc_page_flip [vmwgfx]] *ERROR* Page flip error
 -16.
[ +23.298677] systemd-journald[449]: File /var/log/journal/701d5b975ef34230a
7f4d26acd5bd480/user-1000.journal corrupted or uncleanly shut down, renaming
 and replacing.
[  +0.290529] rfkill: input handler enabled
[  +7.732250] rfkill: input handler disabled
[Nov28 17:32] hrtimer: interrupt took 68950562 ns
[Nov28 19:17] helloModule: loading out-of-tree module taints kernel.
[  +0.000003] helloModule: module license 'unspecified' taints kernel.
[  +0.000000] Disabling lock debugging due to kernel taint
[  +0.000127] helloModule: module verification failed: signature and/or requ
ired key missing - tainting kernel
[  +0.002349] [710012] : Function [initialize] - Hello from OS Project 03!
[Nov28 19:21] [710012] : Function [clean_exit] - Unloading module. Goodbye f
rom OS Project 03!
```

```
snd                    77824  13 snd_seq,snd_seq_device,snd_timer,snd_ac97_c
odec,snd_pcm,snd_rawmidi,snd_ens1371
soundcore              16384  1 snd
vmw_vmci               69632  2 vmw_balloon,vmw_vsock_vmci_transport
mac_hid                16384  0
sch_fq_codel           20480  2
vmwgfx                278528  7
ttm                   110592  1 vmwgfx
drm_kms_helper        176128  1 vmwgfx
drm                   471040  10 vmwgfx,drm_kms_helper,ttm
fb_sys_fops            16384  1 drm_kms_helper
syscopyarea            16384  1 drm_kms_helper
sysfillrect            16384  1 drm_kms_helper
sysimgblt              16384  1 drm_kms_helper
parport_pc             36864  0
ppdev                  20480  0
lp                     20480  0
parport                49152  3 parport_pc,lp,ppdev
ip_tables              24576  0
x_tables               36864  1 ip_tables
autofs4                40960  2
hid_generic            16384  0
usbhid                 49152  0
hid                   126976  2 usbhid,hid_generic
mptspi                 24576  2
psmouse               151552  0
crc32_pclmul           16384  0
e1000                 143360  0
ahci                   40960  0
mptscsih               36864  1 mptspi
libahci                32768  1 ahci
mptbase                98304  2 mptspi,mptscsih
scsi_transport_spi     32768  1 mptspi
i2c_piix4              24576  0
pata_acpi              16384  0
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModuleusertest0710012@ub
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ sudo rmmod hello
Module
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ 
```

Screenshot 6:

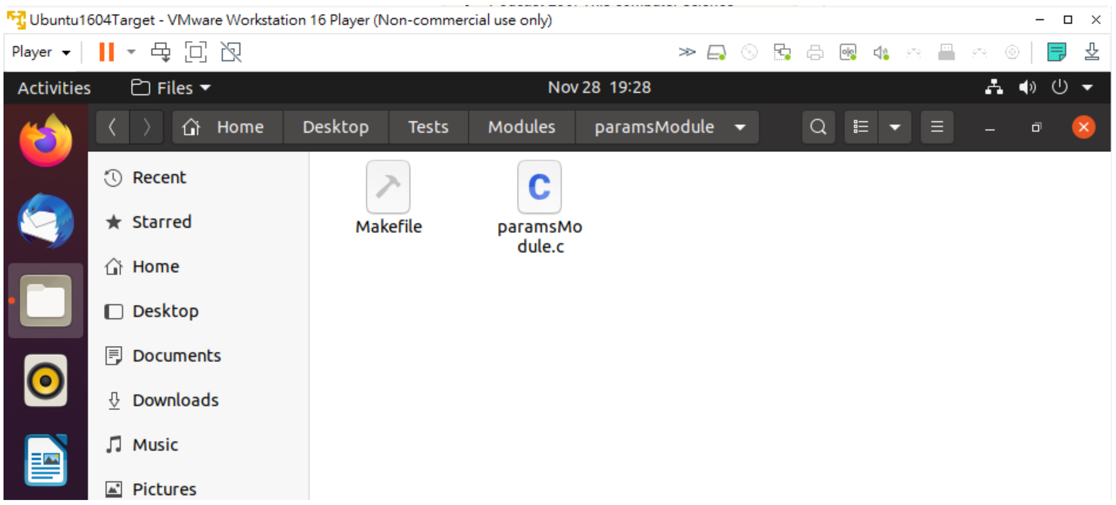
After unloading the module, the module no longer exist in the module list, therefore, it cannot be found even by calling “| grep” command


Ubuntu1604Target - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
Nov 2
usertest0710012@ubuntu: ~/Desktop/Tests/Modules/hello...

```
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ lsmod |
Display all 2017 possibilities? (y or n)
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ lsmod | grep hel
loModule
helloModule            16384  0
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$ lsmod | grep hel
loModule
usertest0710012@ubuntu:~/Desktop/Tests/Modules/helloModule$
```

Screenshot 7:

Same as the helloModule, we save the c program and the Makefile in the same folder



Screenshot 8:

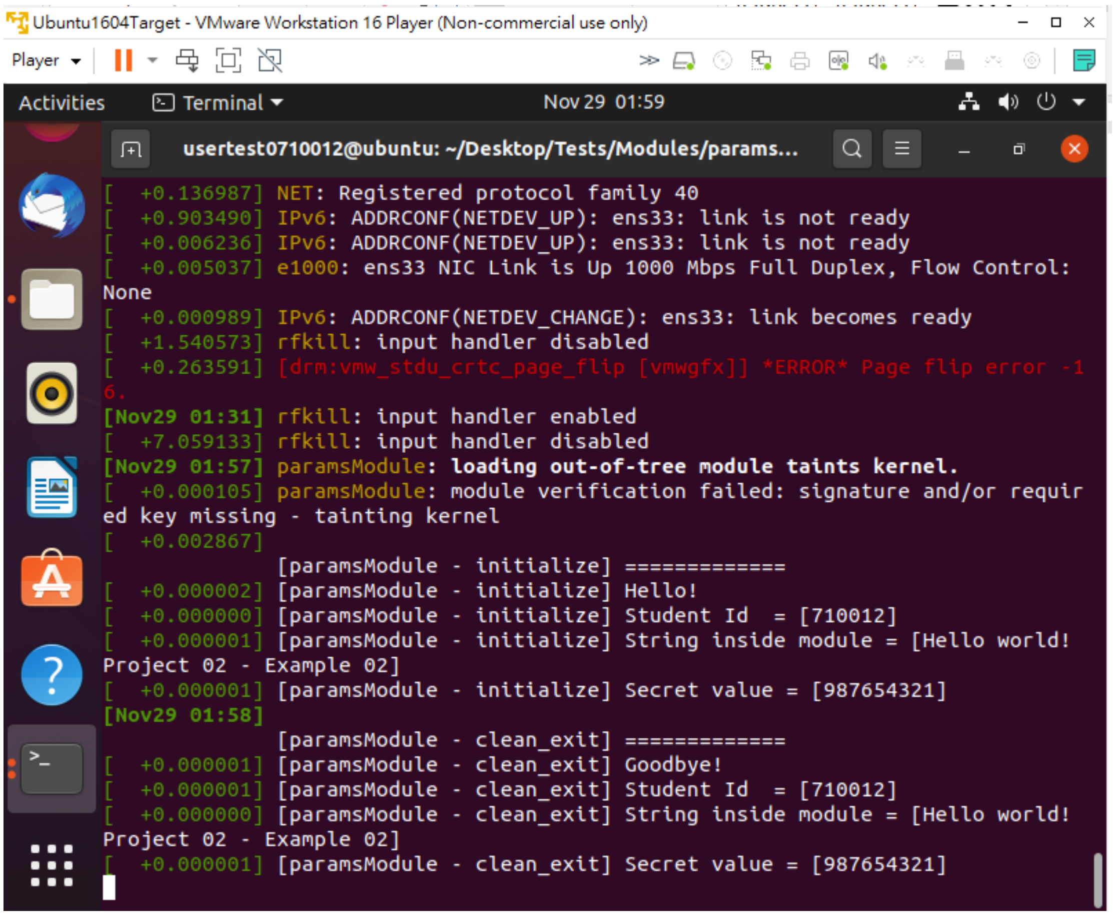
If we simply install the module, the value showed in the kernel space will be the default value, which is

Hello!

Student Id = [710012]

String inside module = [Hello world! Project 02 - Example 02]

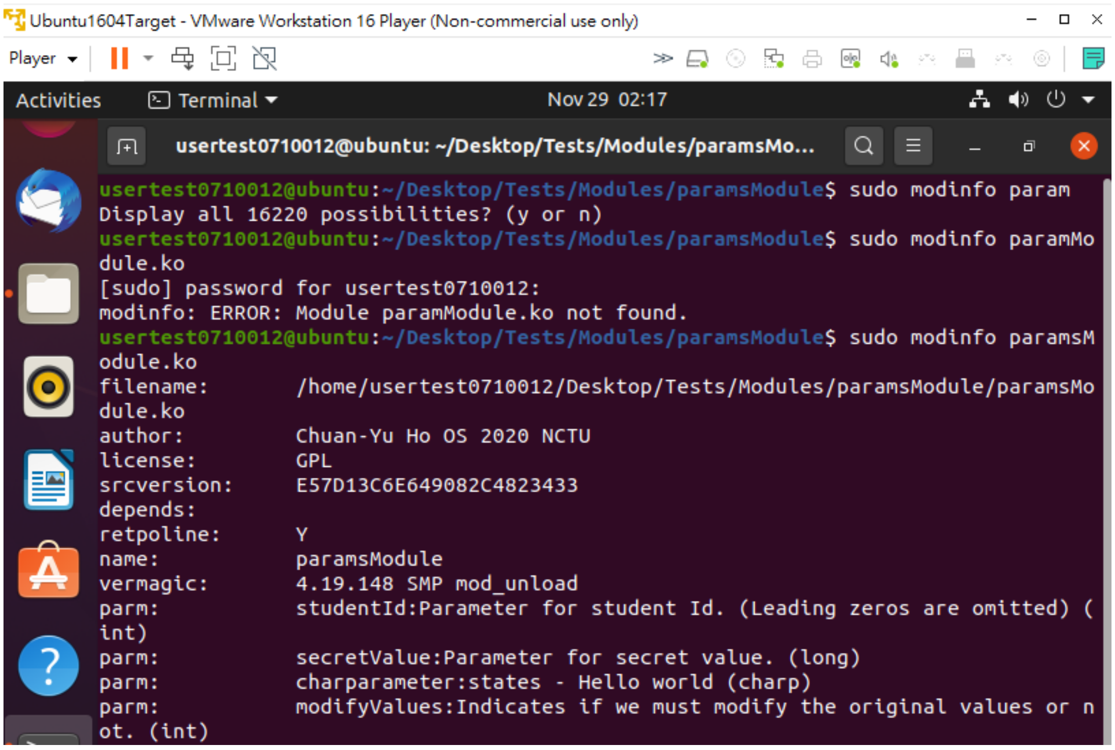
Secret value = [987654321]


Ubuntu1604Target - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
Nov 29 01:59
usertest0710012@ubuntu: ~/Desktop/Tests/Modules/params...

```
[   +0.136987] NET: Registered protocol family 40
[   +0.903490] IPv6: ADDRCONF(NETDEV_UP): ens33: link is not ready
[   +0.006236] IPv6: ADDRCONF(NETDEV_UP): ens33: link is not ready
[   +0.005037] e1000: ens33 NIC Link is Up 1000 Mbps Full Duplex, Flow Control:
None
[   +0.000989] IPv6: ADDRCONF(NETDEV_CHANGE): ens33: link becomes ready
[   +1.540573] rfkill: input handler disabled
[   +0.263591] [drm:vmw_stdu_crtc_page_flip [vmwgfx]] *ERROR* Page flip error -1
6.
[Nov29 01:31] rfkill: input handler enabled
[   +7.059133] rfkill: input handler disabled
[Nov29 01:57] paramsModule: loading out-of-tree module taints kernel.
[   +0.000105] paramsModule: module verification failed: signature and/or requir
ed key missing - tainting kernel
[   +0.002867]
               [paramsModule - initialize] =============
[   +0.000002] [paramsModule - initialize] Hello!
[   +0.000000] [paramsModule - initialize] Student Id  = [710012]
[   +0.000001] [paramsModule - initialize] String inside module = [Hello world!
Project 02 - Example 02]
[   +0.000001] [paramsModule - initialize] Secret value = [987654321]
[Nov29 01:58]
               [paramsModule - clean_exit] =============
[   +0.000001] [paramsModule - clean_exit] Goodbye!
[   +0.000001] [paramsModule - clean_exit] Student Id  = [710012]
[   +0.000000] [paramsModule - clean_exit] String inside module = [Hello world!
Project 02 - Example 02]
[   +0.000001] [paramsModule - clean_exit] Secret value = [987654321]
█
```

Screenshot 9:

We load the module once again into the kernel with modifyValues changed to “1” rather than the default value “0”


Ubuntu1604Target - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
Nov 29 02:17
usertest0710012@ubuntu: ~/Desktop/Tests/Modules/paramsMo...

```
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ sudo modinfo param
Display all 16220 possibilities? (y or n)
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ sudo modinfo paramMo
dule.ko
[sudo] password for usertest0710012:
modinfo: ERROR: Module paramModule.ko not found.
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ sudo modinfo paramsM
odule.ko
filename:       /home/usertest0710012/Desktop/Tests/Modules/paramsModule/paramsMo
dule.ko
author:         Chuan-Yu Ho OS 2020 NCTU
license:        GPL
srcversion:     E57D13C6E649082C4823433
depends:
retpoline:      Y
name:           paramsModule
vermagic:       4.19.148 SMP mod_unload
parm:           studentId:Parameter for student Id. (Leading zeros are omitted) (
int)
parm:           secretValue:Parameter for secret value. (long)
parm:           charparameter:states - Hello world (charp)
parm:           modifyValues:Indicates if we must modify the original values or n
ot. (int)
```

Screenshot 10:

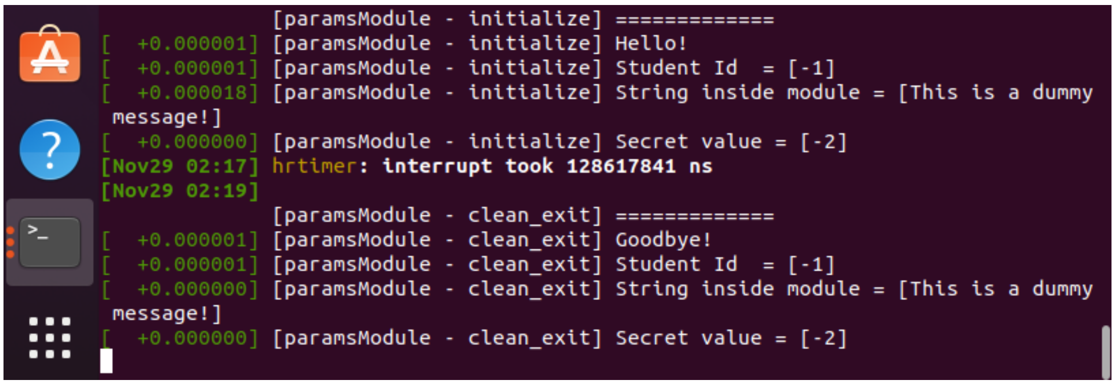
As we change modifyValues, the code enter the “if”, the value of the parameters will be changed, which becomes

Hello!

student Id   = [-1]

String inside module = [This is a dummy message!]

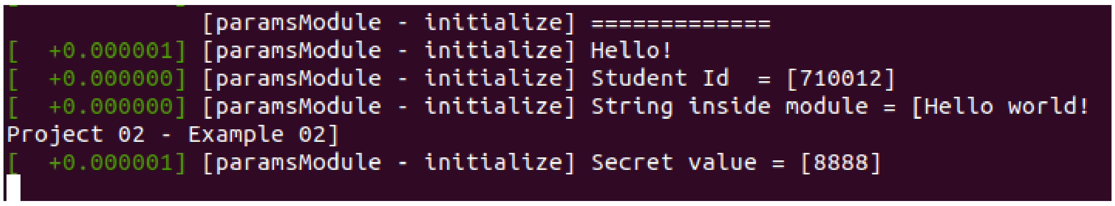
Secret value = [-2]



```
                  [paramsModule - initialize] =============
[   +0.000001] [paramsModule - initialize] Hello!
[   +0.000001] [paramsModule - initialize] Student Id  = [-1]
[   +0.000018] [paramsModule - initialize] String inside module = [This is a dummy
 message!]
[   +0.000000] [paramsModule - initialize] Secret value = [-2]
[Nov29 02:17] hrtimer: interrupt took 128617841 ns
[Nov29 02:19]
                  [paramsModule - clean_exit] =============
[   +0.000001] [paramsModule - clean_exit] Goodbye!
[   +0.000001] [paramsModule - clean_exit] Student Id  = [-1]
[   +0.000000] [paramsModule - clean_exit] String inside module = [This is a dummy
 message!]
[   +0.000000] [paramsModule - clean_exit] Secret value = [-2]
```

Screenshot 11:

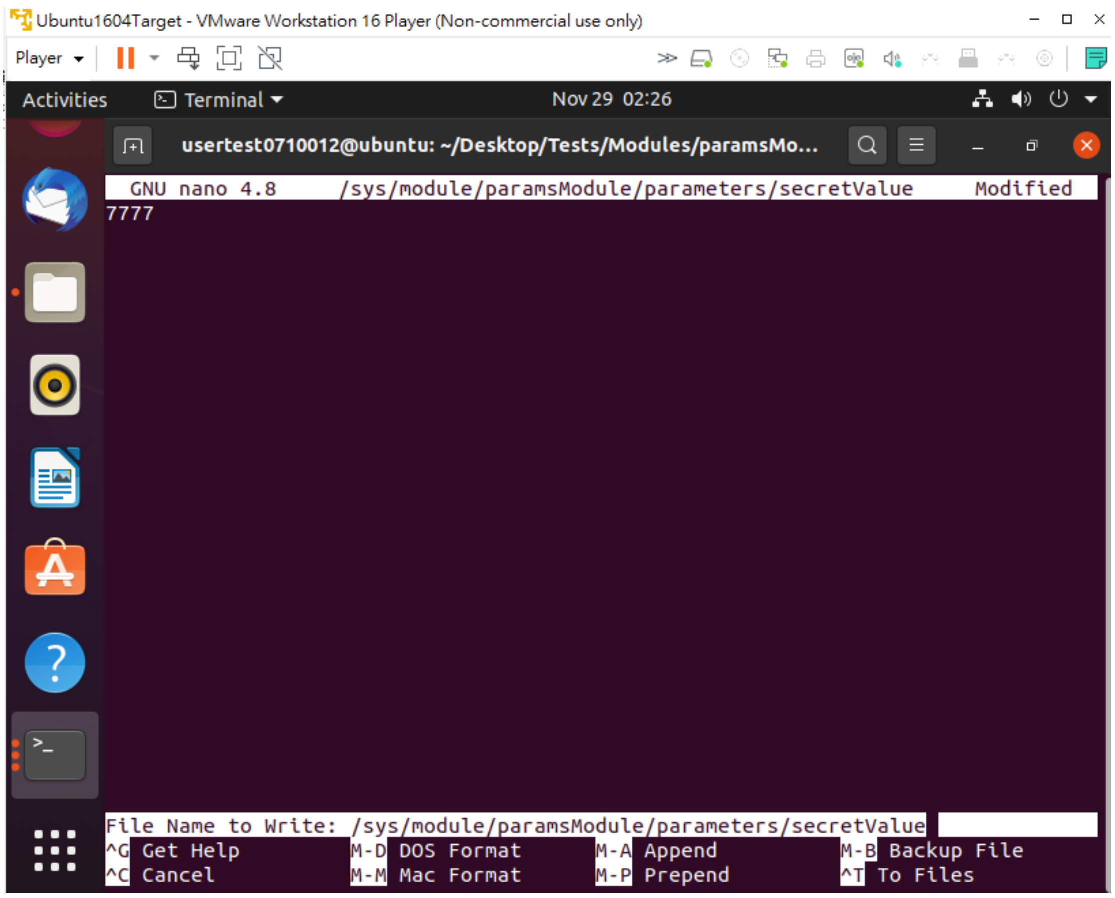
We can modify more than one parameters when installing the module, this time we install the module with the studentId changed to “710012” and secretValue changed to “8888”



```
              [paramsModule - initialize] =============
[  +0.000001] [paramsModule - initialize] Hello!
[  +0.000000] [paramsModule - initialize] Student Id  = [710012]
[  +0.000000] [paramsModule - initialize] String inside module = [Hello world!
Project 02 - Example 02]
[  +0.000001] [paramsModule - initialize] Secret value = [8888]
```

Screenshot 12:

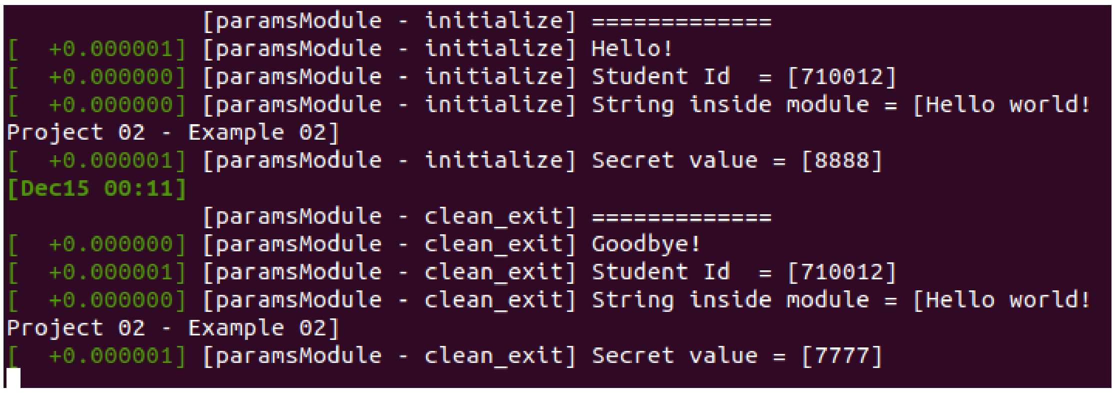
When the module is loaded into the kernel, there will appear a folder with the module’s name in /sys/module folder, and in the folder, there will be files of the parameters declared in the module, and we can change the value of them in the module value by directly change them in the file


Ubuntu1604Target - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Terminal
Nov 29 02:26
usertest0710012@ubuntu: ~/Desktop/Tests/Modules/paramsMo...

```
  GNU nano 4.8      /sys/module/paramsModule/parameters/secretValue      Modified
7777

File Name to Write: /sys/module/paramsModule/parameters/secretValue
^G Get Help          M-D DOS Format       M-A Append           M-B Backup File
^C Cancel            M-M Mac Format       M-P Prepend          ^T To Files
```

Screenshot 13:

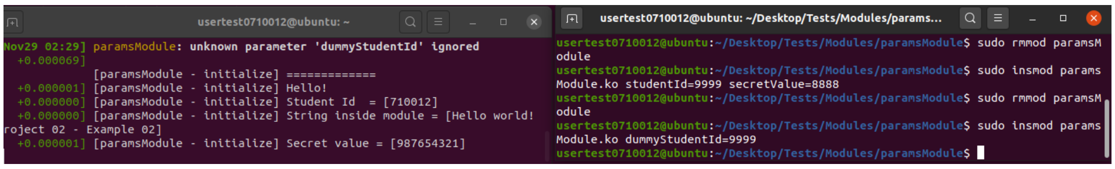
As we change the value of the parameter in the file, the value when we unload the module will be the value we changed to



```
               [paramsModule - initialize] =============
[  +0.000001] [paramsModule - initialize] Hello!
[  +0.000000] [paramsModule - initialize] Student Id  = [710012]
[  +0.000000] [paramsModule - initialize] String inside module = [Hello world!
Project 02 - Example 02]
[  +0.000001] [paramsModule - initialize] Secret value = [8888]
[Dec15 00:11]
               [paramsModule - clean_exit] =============
[  +0.000000] [paramsModule - clean_exit] Goodbye!
[  +0.000001] [paramsModule - clean_exit] Student Id  = [710012]
[  +0.000000] [paramsModule - clean_exit] String inside module = [Hello world!
Project 02 - Example 02]
[  +0.000001] [paramsModule - clean_exit] Secret value = [7777]
```

Screenshot 14:

As dummyStudentId and dummySecretValue don't have the module_param and MODULE_PARAM_DESC assigned to them, so we can't call and change their value while installing the module

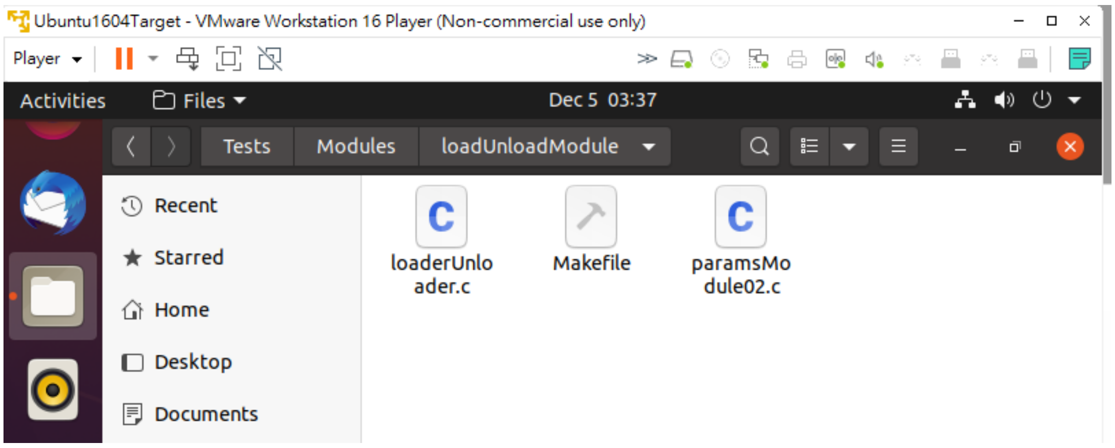

usertest0710012@ubuntu: ~
usertest0710012@ubuntu: ~/Desktop/Tests/Modules/params...

```
Nov29 02:29] paramsModule: unknown parameter 'dummyStudentId' ignored
  +0.000069]
              [paramsModule - initialize] =============
  +0.000001] [paramsModule - initialize] Hello!
  +0.000000] [paramsModule - initialize] Student Id  = [710012]
  +0.000000] [paramsModule - initialize] String inside module = [Hello world!
roject 02 - Example 02]
  +0.000001] [paramsModule - initialize] Secret value = [987654321]
```

```
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ sudo rmmod paramsM
odule
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ sudo insmod params
Module.ko studentId=9999 secretValue=8888
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ sudo rmmod paramsM
odule
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ sudo insmod params
Module.ko dummyStudentId=9999
usertest0710012@ubuntu:~/Desktop/Tests/Modules/paramsModule$ 
```

Screenshot 15:

Same as the previous modules, we save the c program and Makefile in the same folder, however, we add one more loaderUnloader c program to install and unload the module by c program automatiically



Screenshot 16:

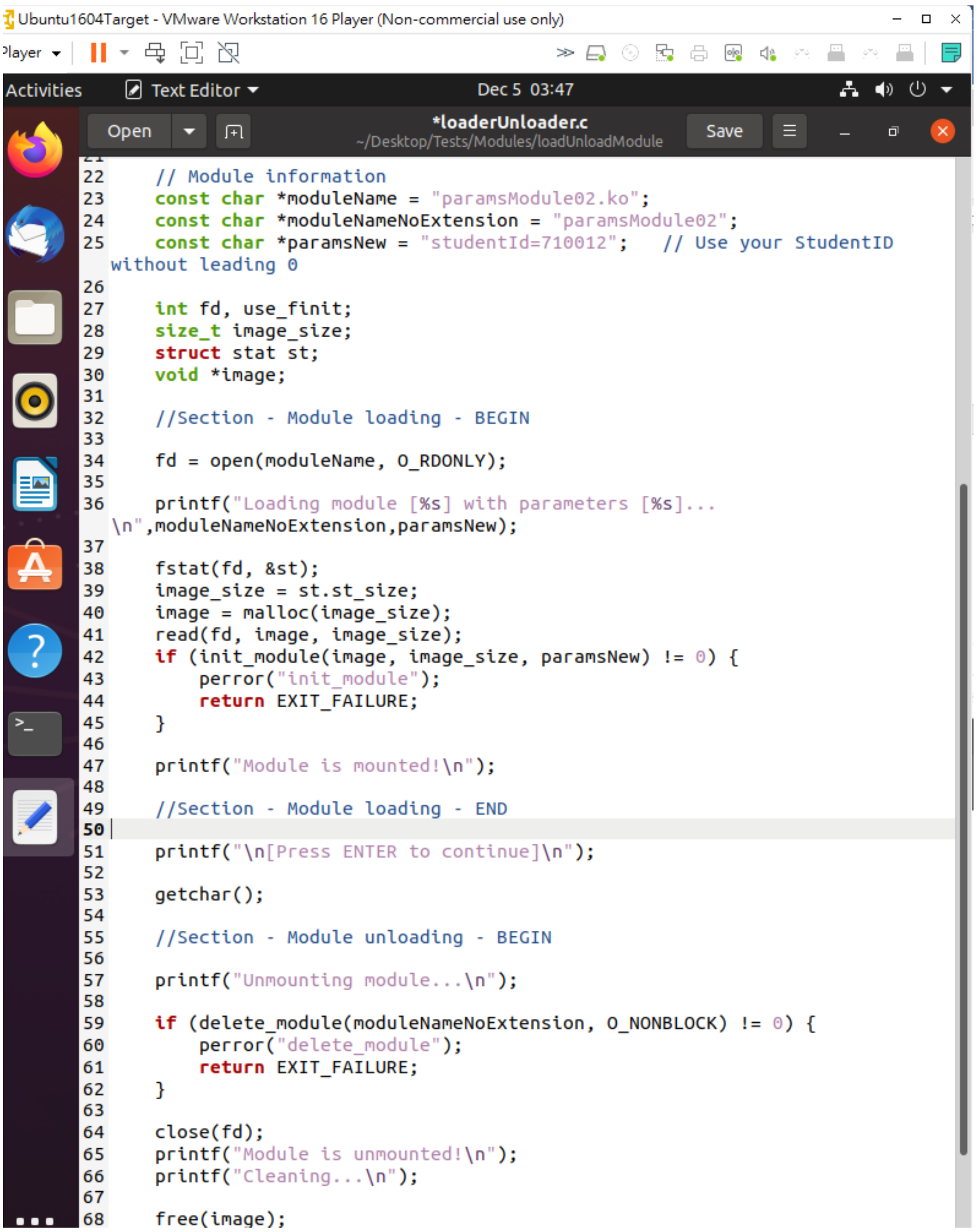
First define the load, unload function : init_module(finit_module share same purpose with init_module, another loader function), delete_module

To load the module, we call init_module and keep track of the return value, if the value is not 0, the program will counter error(perror("init_module");) and exit

Same with loading the module, we unload the module by calling delete_module and keep track of its return value also

And the paramNew string input the “studentId=710012” into the kernel along with the module(like “sudo insmod paramModule studentId=710012”) to change the value of studentId in the module

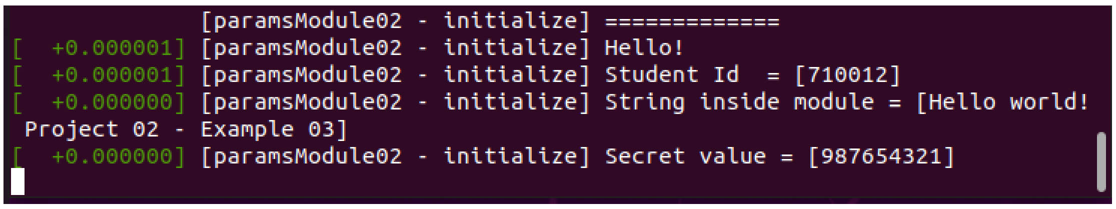
To keep the module load and unload in instant, getchar() is placed in the middle to pause the running code


Ubuntu1604Target - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
Text Editor
Dec 5 03:47
Open
*loaderUnloader.c
~/Desktop/Tests/Modules/loadUnloadModule
Save

```
    // Module information
    const char *moduleName = "paramsModule02.ko";
    const char *moduleNameNoExtension = "paramsModule02";
    const char *paramsNew = "studentId=710012";    // Use your StudentID
without leading 0

    int fd, use_finit;
    size_t image_size;
    struct stat st;
    void *image;

    //Section - Module loading - BEGIN

    fd = open(moduleName, O_RDONLY);

    printf("Loading module [%s] with parameters [%s]...
\n",moduleNameNoExtension,paramsNew);

    fstat(fd, &st);
    image_size = st.st_size;
    image = malloc(image_size);
    read(fd, image, image_size);
    if (init_module(image, image_size, paramsNew) != 0) {
        perror("init_module");
        return EXIT_FAILURE;
    }

    printf("Module is mounted!\n");

    //Section - Module loading - END

    printf("\n[Press ENTER to continue]\n");

    getchar();

    //Section - Module unloading - BEGIN

    printf("Unmounting module...\n");

    if (delete_module(moduleNameNoExtension, O_NONBLOCK) != 0) {
        perror("delete_module");
        return EXIT_FAILURE;
    }

    close(fd);
    printf("Module is unmounted!\n");
    printf("Cleaning...\n");

    free(image);
```

Screenshot 17:

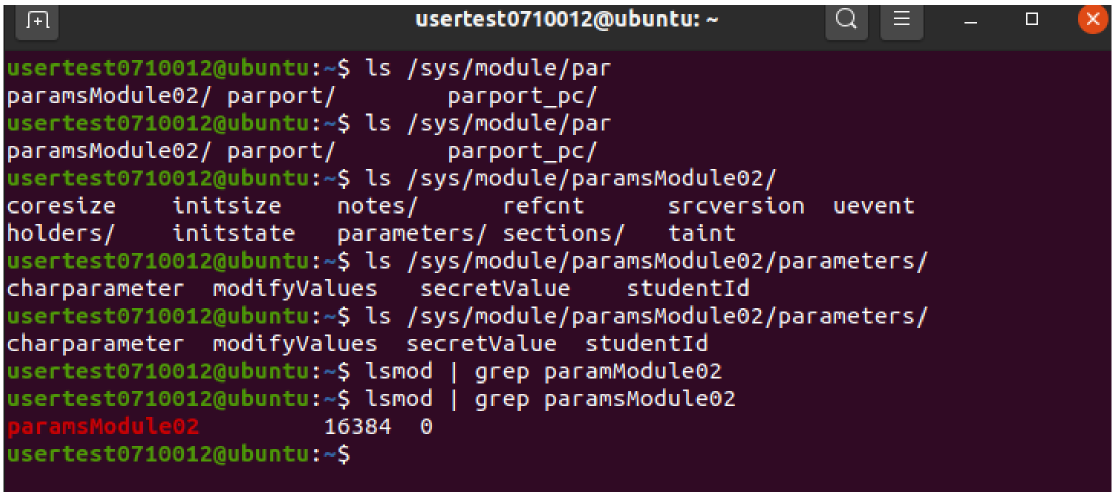
The module is installed as soon as we run the loaderUnloader program



```
                [paramsModule02 - initialize] ==============
[  +0.000001] [paramsModule02 - initialize] Hello!
[  +0.000001] [paramsModule02 - initialize] Student Id  = [710012]
[  +0.000000] [paramsModule02 - initialize] String inside module = [Hello world!
 Project 02 - Example 03]
[  +0.000000] [paramsModule02 - initialize] Secret value = [987654321]
```

Screenshot 18:

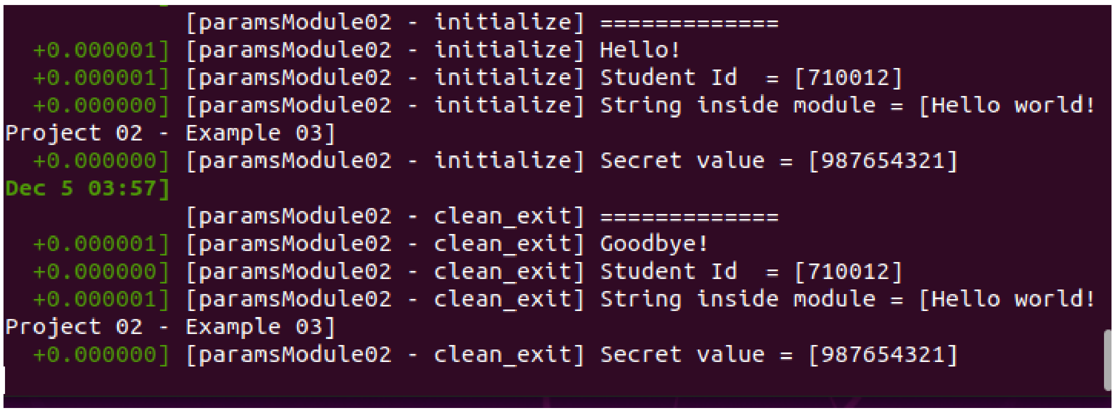
While the program is stuck in the getchar(), we can look up the module list, or directly call its module name to find the paramsModule02 module



```
usertest0710012@ubuntu:~$ ls /sys/module/par
paramsModule02/ parport/        parport_pc/
usertest0710012@ubuntu:~$ ls /sys/module/par
paramsModule02/ parport/        parport_pc/
usertest0710012@ubuntu:~$ ls /sys/module/paramsModule02/
coresize    initsize    notes/      refcnt      srcversion  uevent
holders/    initstate   parameters/ sections/   taint
usertest0710012@ubuntu:~$ ls /sys/module/paramsModule02/parameters/
charparameter  modifyValues   secretValue    studentId
usertest0710012@ubuntu:~$ ls /sys/module/paramsModule02/parameters/
charparameter  modifyValues  secretValue  studentId
usertest0710012@ubuntu:~$ lsmod | grep paramModule02
usertest0710012@ubuntu:~$ lsmod | grep paramsModule02
paramsModule02         16384  0
usertest0710012@ubuntu:~$
```

Screenshot 19:

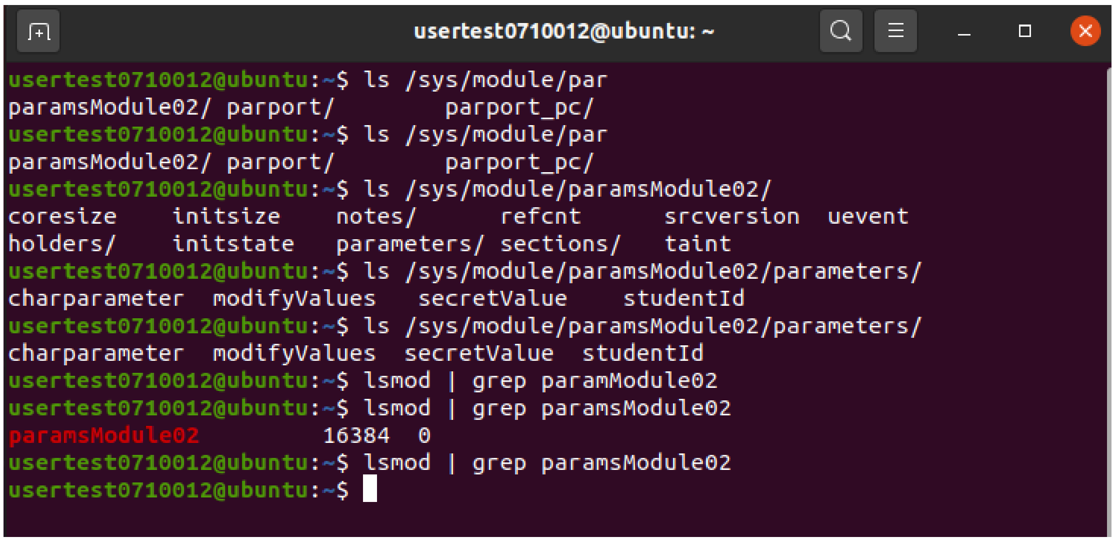
After we press enter to get through getchar(), the module is automatically unloaded



```
             [paramsModule02 - initialize] =============
  +0.000001] [paramsModule02 - initialize] Hello!
  +0.000001] [paramsModule02 - initialize] Student Id  = [710012]
  +0.000000] [paramsModule02 - initialize] String inside module = [Hello world!
Project 02 - Example 03]
  +0.000000] [paramsModule02 - initialize] Secret value = [987654321]
Dec 5 03:57]
             [paramsModule02 - clean_exit] =============
  +0.000001] [paramsModule02 - clean_exit] Goodbye!
  +0.000000] [paramsModule02 - clean_exit] Student Id  = [710012]
  +0.000001] [paramsModule02 - clean_exit] String inside module = [Hello world!
Project 02 - Example 03]
  +0.000000] [paramsModule02 - clean_exit] Secret value = [987654321]
```

Screenshot 20:

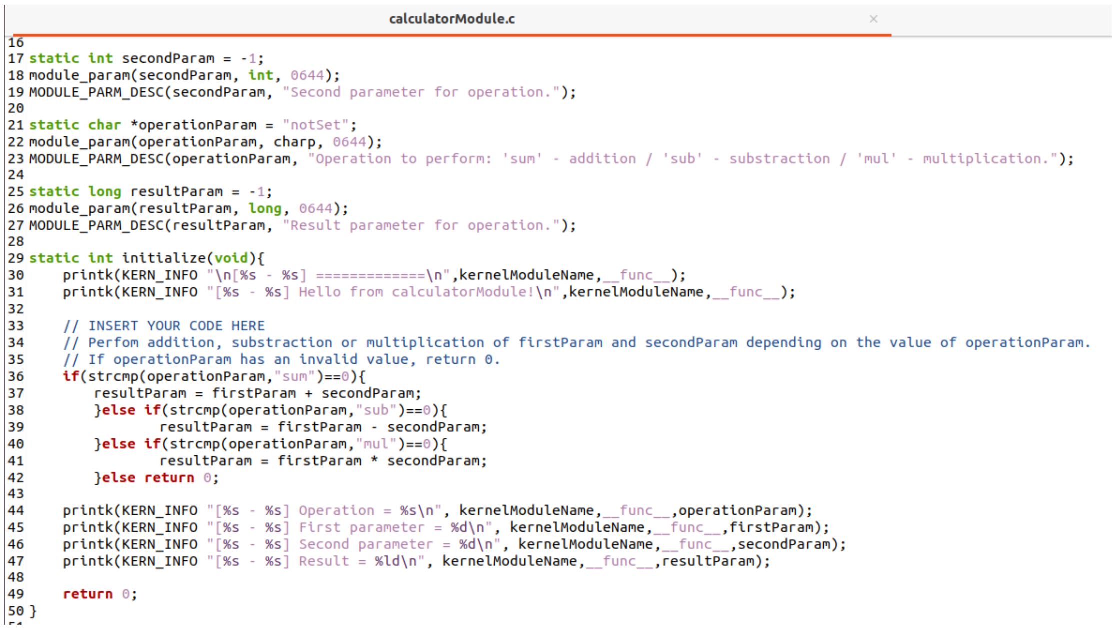
As the module is already unloaded, we cannot find it in the module list anymore



```
usertest0710012@ubuntu:~$ ls /sys/module/par
paramsModule02/ parport/        parport_pc/
usertest0710012@ubuntu:~$ ls /sys/module/par
paramsModule02/ parport/        parport_pc/
usertest0710012@ubuntu:~$ ls /sys/module/paramsModule02/
coresize    initsize    notes/     refcnt     srcversion  uevent
holders/    initstate   parameters/ sections/   taint
usertest0710012@ubuntu:~$ ls /sys/module/paramsModule02/parameters/
charparameter  modifyValues   secretValue    studentId
usertest0710012@ubuntu:~$ ls /sys/module/paramsModule02/parameters/
charparameter  modifyValues  secretValue  studentId
usertest0710012@ubuntu:~$ lsmod | grep paramModule02
usertest0710012@ubuntu:~$ lsmod | grep paramsModule02
paramsModule02         16384  0
usertest0710012@ubuntu:~$ lsmod | grep paramsModule02
usertest0710012@ubuntu:~$ 
```

Screenshot 21:

In the calculatorModule.c file(the real module loaded into kernel), we first compare the input operationParam to see if we need to add, subtract, or multiply this time, and give the answer of the computation of firstParam and secondParam to resultParam



```c

static int secondParam = -1;
module_param(secondParam, int, 0644);
MODULE_PARM_DESC(secondParam, "Second parameter for operation.");

static char *operationParam = "notSet";
module_param(operationParam, charp, 0644);
MODULE_PARM_DESC(operationParam, "Operation to perform: 'sum' - addition / 'sub' - substraction / 'mul' - multiplication.");

static long resultParam = -1;
module_param(resultParam, long, 0644);
MODULE_PARM_DESC(resultParam, "Result parameter for operation.");

static int initialize(void){
    printk(KERN_INFO "\n[%s - %s] ============\n",kernelModuleName,__func__);
    printk(KERN_INFO "[%s - %s] Hello from calculatorModule!\n",kernelModuleName,__func__);

    // INSERT YOUR CODE HERE
    // Perfom addition, substraction or multiplication of firstParam and secondParam depending on the value of operationParam.
    // If operationParam has an invalid value, return 0.
    if(strcmp(operationParam,"sum")==0){
        resultParam = firstParam + secondParam;
        }else if(strcmp(operationParam,"sub")==0){
                resultParam = firstParam - secondParam;
        }else if(strcmp(operationParam,"mul")==0){
                resultParam = firstParam * secondParam;
        }else return 0;

    printk(KERN_INFO "[%s - %s] Operation = %s\n", kernelModuleName,__func__,operationParam);
    printk(KERN_INFO "[%s - %s] First parameter = %d\n", kernelModuleName,__func__,firstParam);
    printk(KERN_INFO "[%s - %s] Second parameter = %d\n", kernelModuleName,__func__,secondParam);
    printk(KERN_INFO "[%s - %s] Result = %ld\n", kernelModuleName,__func__,resultParam);

    return 0;
}
```

Screenshot 22:

Screenshot 22 and 23 is parts in addition function, so we'll apply them to subtraction and multiplication also.

First, we make sure the module name and file we're going to load into the kernel(moduleName, moduleNameNoExtension). After that, as we have to change the value of firstParam, secondParam, and operationParam along while installing the module, we have to concatenate them into a string and assigned them to paramNew to load in the kernel along with the module

After that we install the module and the computation should be done in calculatorModule.c

```c
long addition (int input1, int input2)
{
    long result = 0;

    const char *moduleName = "calculatorModule.ko";
        const char *moduleNameNoExtension = "calculatorModule";
        char n1[100];
        char n2[100];
        //itoa(input1,n1,10);
        //itoa(input2,n2,10);
        sprintf(n1, "%d", input1);
        sprintf(n2, "%d", input2);
        char first[100]="firstParam=";
        strcat(first,n1);
        strcat(first, " secondParam=");
        strcat(first, n2);
        strcat(first, " operationParam=\"sum\"");
        const char *paramsNew = first;
        int fd, use_finit;
        size_t image_size;
        struct stat st;
        void *image;
//Section - Module loading - BEGIN
        fd = open(moduleName, O_RDONLY);
        fstat(fd, &st);
        image_size = st.st_size;
        image = malloc(image_size);
        read(fd, image, image_size);
        if (result=init_module(image, image_size, paramsNew) != 0) {
                perror("init_module");
                return EXIT_FAILURE;
        }
//Section - Module loading - END
```

Screenshot 23:

Between the time when the module is installed and hasn't been unload, there should exist a folder called "calculatorModule", and there should be a file called "resultParam" in the parameter folder also, the result after compution should be stored in the file.

We can extract the value of the result from the file and store it in the "result" parameter in calculator.c program

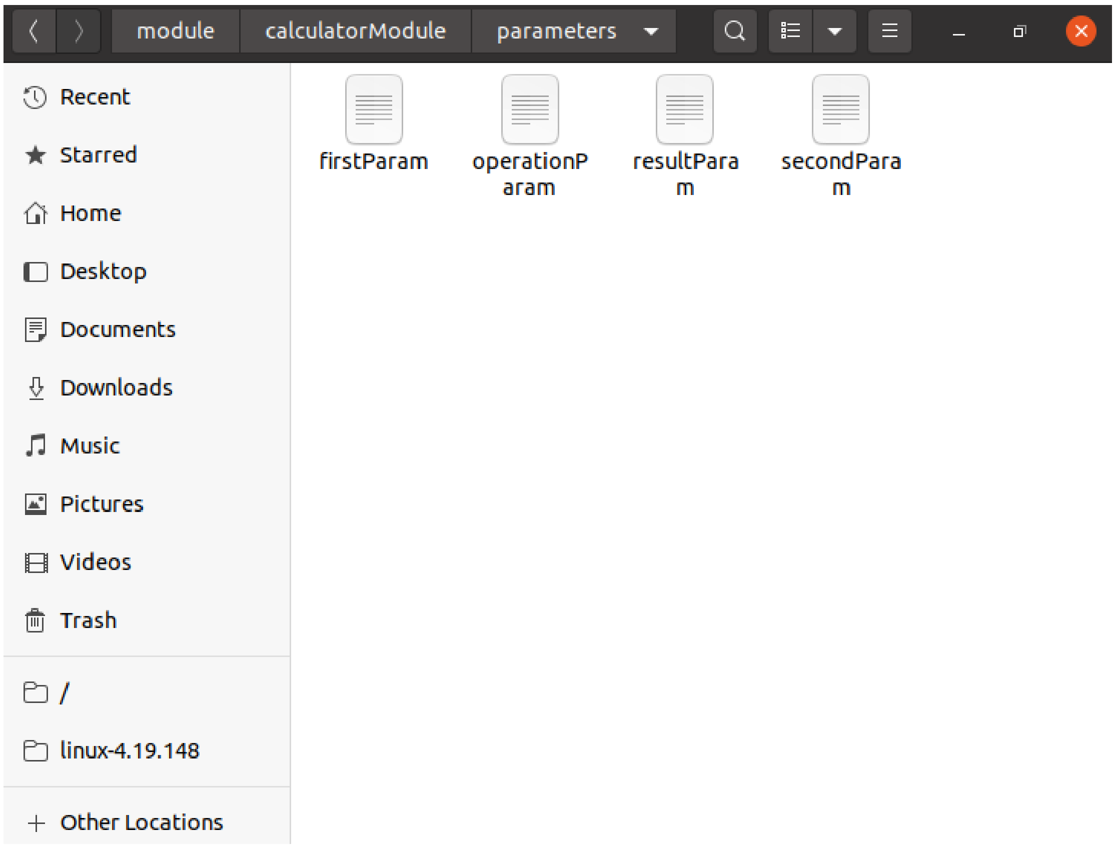
After that, we unload the module by calling delete_module

```
        FILE *fp = fopen("/sys/module/calculatorModule/parameters/resultParam","r");
        fscanf(fp,"%ld",&result);
        fclose(fp);

//Section - Module unloading - BEGIN
        if (delete_module(moduleNameNoExtension, O_NONBLOCK) != 0) {
                perror("delete_module");
                return EXIT_FAILURE;
        }
        close(fd);
        free(image);
//Section - Module unloading - END

    return result;
}
```

Screenshot 24:

Every call of operation(sum, sub, mul) will load in the calcultorModule.c module, and the following file should be created everytime, with the value of them stored in each of them.

After the operation ended, every file down below, along with the parent folder "calculaterModule" should disappear


module
calculatorModule
parameters
Recent
Starred
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
/
linux-4.19.148
Other Locations
firstParam
operationParam
resultParam
secondParam