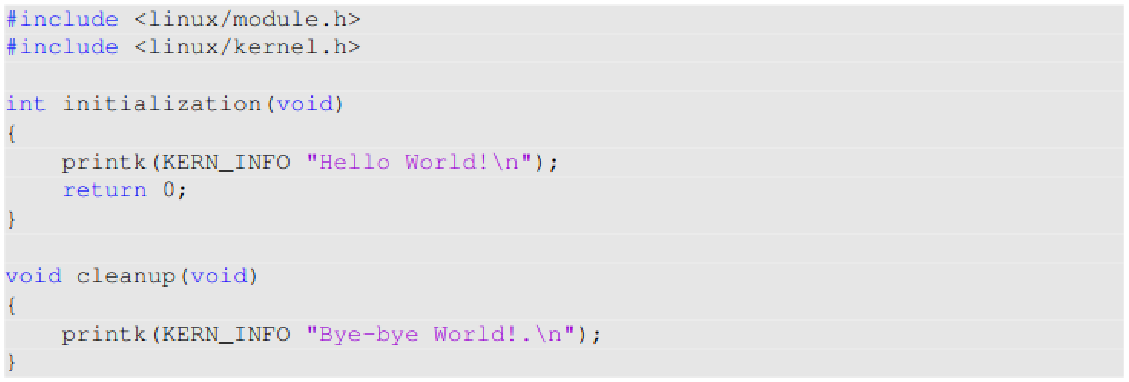
Firewall Exploration Lab

Task 1: Implementing a Simple Firewall: We will implement a simple packet filtering type of firewall which inspects each incoming and outgoing packets, and enforces the firewall policies set by the administrator.

- **Task 1.A: Implement a Simple Kernel Module:** LKM(Linux kernel Module) allows us to add a new module to the kernel at the runtime. This new module enables us to extend the functionalities of the kernel, without rebuilding the kernel or even rebooting the computer. The packet filtering part of a firewall can be implemented as an LKM. In This task we will get familiar with LKM.

  We have been provided with a simple loadable kernel module.

Listing 1: `hello.c` (included in the lab setup files)

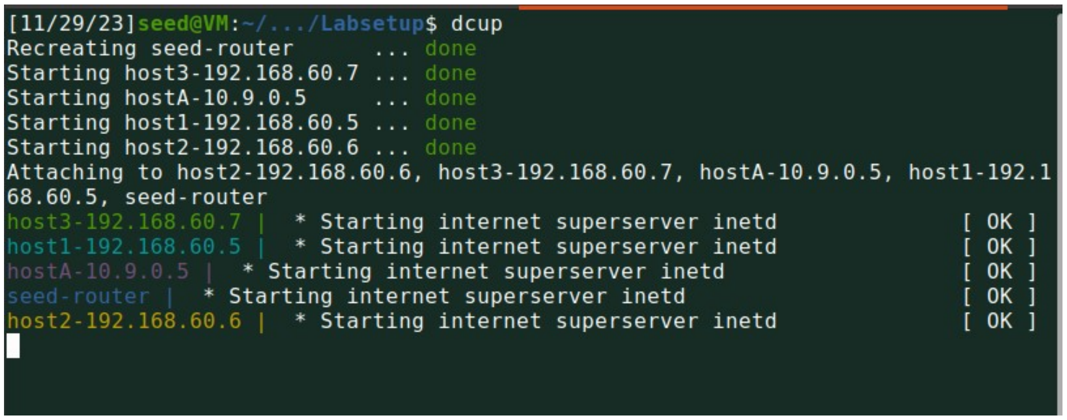


```
#include <linux/module.h>
#include <linux/kernel.h>

int initialization(void)
{
    printk(KERN_INFO "Hello World!\n");
    return 0;
}

void cleanup(void)
{
    printk(KERN_INFO "Bye-bye World!.\n");
}
```

It contains two methods, *initialization* method is executed at the time of loading the module into the kernel and *cleanup* module which is executed at the time of removal of the module from the kernel.

We are provided with a makefile, which would be used at the time of compiling our sample code.

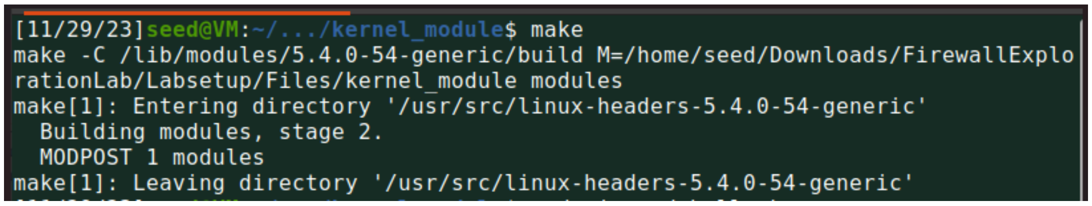
Before getting started with Task 1, I've made our virtual environment live, which is provided in the form of IaaC(docker-compose.yml).



```
[11/29/23]seed@VM:~/.../Labsetup$ dcup
Recreating seed-router       ... done
Starting host3-192.168.60.7 ... done
Starting hostA-10.9.0.5      ... done
Starting host1-192.168.60.5 ... done
Starting host2-192.168.60.6 ... done
Attaching to host2-192.168.60.6, host3-192.168.60.7, hostA-10.9.0.5, host1-192.1
68.60.5, seed-router
host3-192.168.60.7 |  * Starting internet superserver inetd                [ OK ]
host1-192.168.60.5 |  * Starting internet superserver inetd                [ OK ]
hostA-10.9.0.5 |  * Starting internet superserver inetd                    [ OK ]
seed-router |  * Starting internet superserver inetd                       [ OK ]
host2-192.168.60.6 |  * Starting internet superserver inetd                [ OK ]
```

Figure: Starting docker compose lab environment

To generate the modules navigate to **LabSetup->Files->kernel_module,** and get a command line from the directory.
Compile the .c file using *make* command



```
[11/29/23]seed@VM:~/.../kernel_module$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Downloads/FirewallExplo
rationLab/Labsetup/Files/kernel_module modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  Building modules, stage 2.
  MODPOST 1 modules
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
```

Figure: generating kernel module

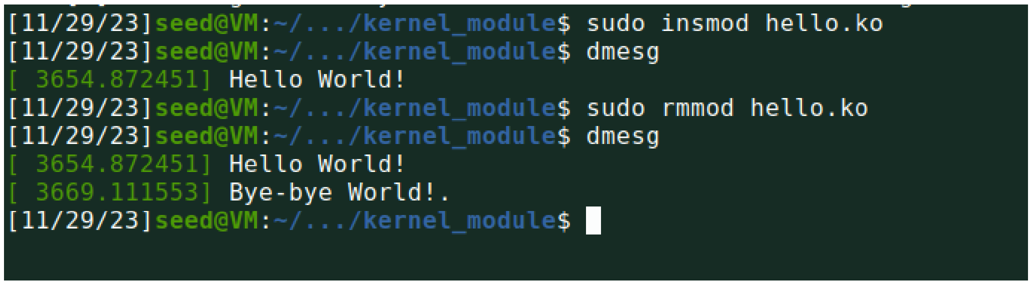
You can verify, that your module is generate by checking for file named hello.ko
Now, there are 3 commands used to to insert, remove, list kernel modules

~ sudo insmode {module_name} (Insert the module)

~ lsmod (List modules)

~ sudo rmmod {module_name} (Remove the module)

And to read the kernel and kernel module logs, we use *dmesg.*
Screenshot below shows the insertion, and removal of kernel module, and the log it generated



```
[11/29/23]seed@VM:~/.../kernel_module$ sudo insmod hello.ko
[11/29/23]seed@VM:~/.../kernel_module$ dmesg
[ 3654.872451] Hello World!
[11/29/23]seed@VM:~/.../kernel_module$ sudo rmmod hello.ko
[11/29/23]seed@VM:~/.../kernel_module$ dmesg
[ 3654.872451] Hello World!
[ 3669.111553] Bye-bye World!.
[11/29/23]seed@VM:~/.../kernel_module$ 
```

- **Task 1.B Implement a Simple Firewall Using Netfilter:**