



دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

گزارش کار آزمایشگاه آزمایشگاه سیستم‌های عامل

گزارش آزمایش شماره ۱۰
(آشنایی با درایورها)

۲۰

شماره‌ی گروه:

ارشیا یوسف‌نیا (۴۰۱۱۱۰۴۱۵)

گروه:

محمدعارف زارع زاده (۴۰۱۱۰۶۰۱۷)

دکتر بیگی

استاد درس:

تابستان ۱۴۰۴

تاریخ:

فهرست مطالب

۱	آزمایش ۱	۱
۱	۱.۱ نیازمندی‌های اجرا	۱
۱	۲.۱ توضیح کد و روند آزمایش	۱
۴	آزمایش ۲	۴
۴	۱.۲ نیازمندی‌های اجرا	۴
۴	۲.۲ توضیح کد و روند آزمایش	۴

لیست تصاویر

۱	برنامهٔ درایور آزمایش	۲
۲	دستورالعمل ساخت یا Makefile	۲
۳	نتیجهٔ ساخت و استفاده از درایور	۳
۴	برنامهٔ درایور آزمایش	۴
۵	دستورالعمل ساخت یا Makefile	۵
۶	ساخت و بارگذاری ماژول و تبادل بسته با اینترنت	۶
۷	مشاهده فایل لاگ بسته‌های دیده شده از درایور و پیام‌های کرنل	۷

۱ آزمایش ۱

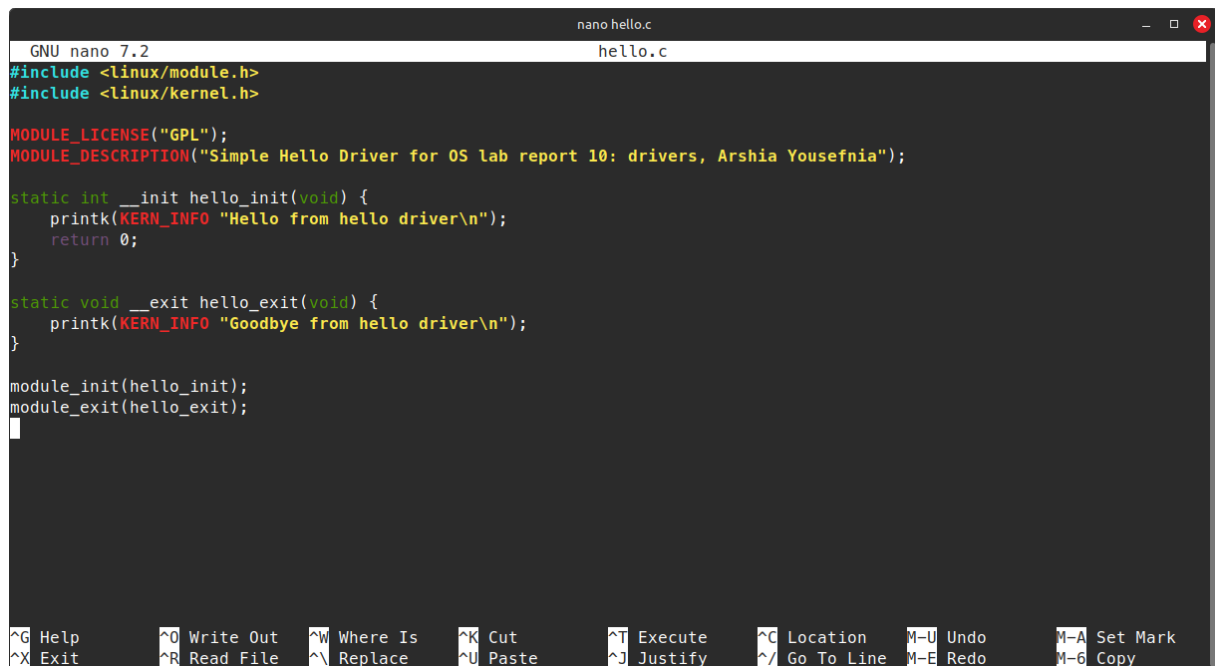
۱.۱ نیازمندی‌های اجرا

برای ساخت ماژول و بارگذاری و حذف، مراحل زیر را دنبال کنید:

```
# install required dependencies
sudo apt upgrade
sudo apt install build-essential linux-headers-$(uname -r)
# command to build a driver module
make
# commands to load and remove a driver module
sudo insmod driver.ko
sudo rmmod driver.ko
```

۲.۱ توضیح کد و روند آزمایش

در این بخش یک درایور ساده می‌نویسیم که صرفاً یک پیام برای ما چاپ کند تا از درستی کار خود آگاه شویم. برای نوشتن این درایور اتفاقاتی که باید در هنگام لود شدن و در هنگام خروج انجام شود را مشخص کنیم. در اینجا در هر دو مورد صرفاً یک لاگ می‌اندازیم. شکل ۱ این برنامه را نشان می‌دهد که تابعی که باید هنگام آغاز و خروج فراخوانی شود را مشخص می‌کند و نکاتی مثل توضیح یا لایسنس را هم مشخص می‌کند. در نهایت برای ساخت و کامپایل محتوای Makefile را مطابق شکل ۲ پر می‌کنیم تا درایور آماده نصب شود. در شکل ۳ نتیجه ساخت و نصب درایور مشاهده می‌شود. همچنین با دستور `dmesg` لاگ‌ها را بررسی می‌کنیم و پیام‌های خود را پیدا می‌کنیم. همه فایل‌ها به پیوست گزارش ارسال شده است.



```
GNU nano 7.2 hello.c
#include <linux/module.h>
#include <linux/kernel.h>

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Simple Hello Driver for OS lab report 10: drivers, Arshia Yousefnia");

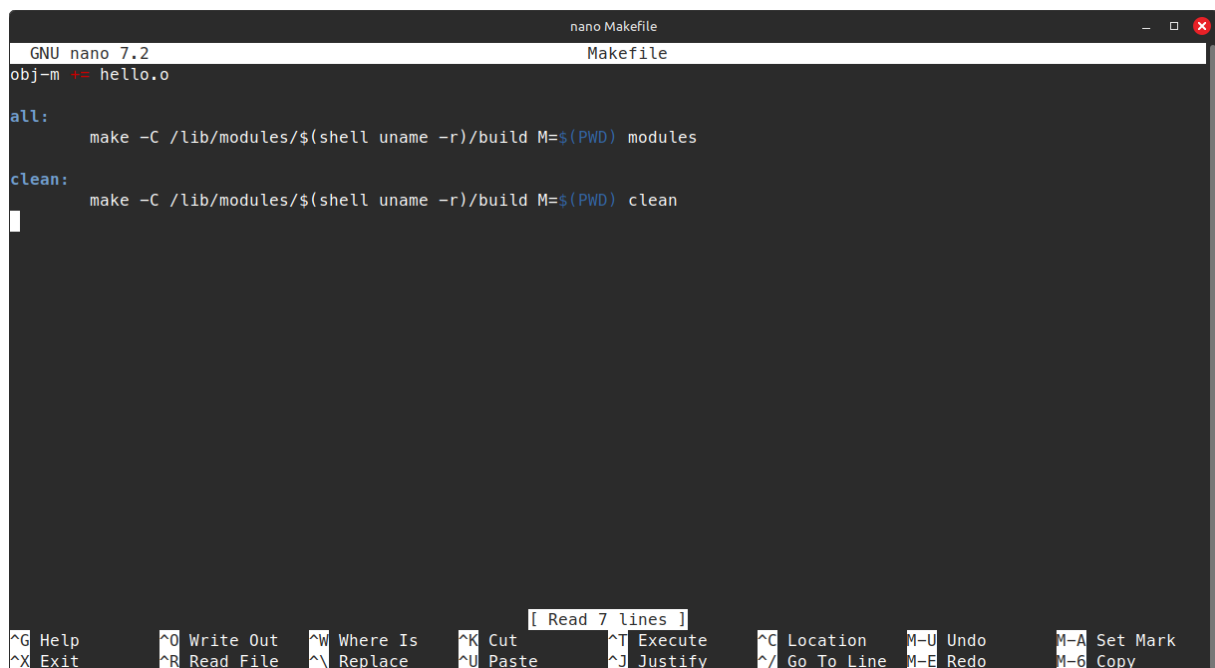
static int __init hello_init(void) {
    printk(KERN_INFO "Hello from hello driver\n");
    return 0;
}

static void __exit hello_exit(void) {
    printk(KERN_INFO "Goodbye from hello driver\n");
}

module_init(hello_init);
module_exit(hello_exit);
```

The screenshot shows a terminal window with the nano text editor. The file being edited is 'hello.c'. The code defines a kernel module with a license, a description, an initialization function that prints 'Hello from hello driver', and an exit function that prints 'Goodbye from hello driver'. The module is initialized and exits using the standard module functions.

شکل ۱: برنامه درایور آزمایش



```
GNU nano 7.2 Makefile
obj-m += hello.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 with the nano text editor. The file being edited is 'Makefile'. The Makefile defines the object file 'hello.o' and provides instructions for building the module (using 'make -C /lib/modules/\$(shell uname -r)/build M=\$(PWD) modules') and cleaning the build (using 'make -C /lib/modules/\$(shell uname -r)/build M=\$(PWD) clean').

شکل ۲: دستورالعمل ساخت یا Makefile

```
arshia@arshia-Notebook: ~/Desktop/report10-codes/hello
ls
hello.c Makefile
arshia@arshia-Notebook: ~/Desktop/report10-codes/hello
make
make -C /lib/modules/6.8.0-71-generic/build M=/home/arshia/Desktop/report10-codes/hello modules
make[1]: Entering directory '/usr/src/linux-headers-6.8.0-71-generic'
warning: the compiler differs from the one used to build the kernel
The kernel was built by: x86_64-linux-gnu-gcc-13 (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0
You are using: gcc-13 (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0
CC [M] /home/arshia/Desktop/report10-codes/hello/hello.o
MODPOST /home/arshia/Desktop/report10-codes/hello/Module.symvers
CC [M] /home/arshia/Desktop/report10-codes/hello/hello.mod.o
LD [M] /home/arshia/Desktop/report10-codes/hello/hello.ko
BTF [M] /home/arshia/Desktop/report10-codes/hello/hello.ko
Skipping BTF generation for /home/arshia/Desktop/report10-codes/hello/hello.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-6.8.0-71-generic'
arshia@arshia-Notebook: ~/Desktop/report10-codes/hello
sudo insmod hello.ko
[sudo] password for arshia:
arshia@arshia-Notebook: ~/Desktop/report10-codes/hello
sudo rmmod hello.ko
arshia@arshia-Notebook: ~/Desktop/report10-codes/hello
dmesg | tail -n 5
[ 88.735351] br-51184d51f14e: port 1(veth4e538a6) entered blocking state
[ 88.735370] br-51184d51f14e: port 1(veth4e538a6) entered forwarding state
[ 462.746875] systemd-journald[291]: /var/log/journal/cc75d8529955423f8fa94d47c8173c78/user-1000.journal: Journal file
uses a different sequence number ID, rotating.
[ 793.078040] Hello from hello driver
[ 803.292579] Goodbye from hello driver
arshia@arshia-Notebook: ~/Desktop/report10-codes/hello
```

شکل ۳: نتیجه ساخت و استفاده از درایور

۲ آزمایش ۲

۱.۲ نیازمندی‌های اجرا

مشابه آزمایش قبل:

```
# install required dependencies
sudo apt upgrade
sudo apt install build-essential linux-headers-$(uname -r)
# command to build a driver module
make
# commands to load and remove a driver module
sudo insmod driver.ko
sudo rmmod driver.ko
```

۲.۲ توضیح کد و روند آزمایش

در این آزمایش یک درایور می‌نویسیم که با استفاده از netfilter یک هوک روی ترافیک دستگاه ایجاد کند و از این طریق بتوانیم ترافیک را که با پروتکل ipv4 است رصد کنیم. این هوک لاگ ترافیک را به فایلی که در برنامه مشخص شده می‌ریزد. در آغاز به کار درایور با این هوک را در netfilter ثبت می‌کنیم و در پایان کار هم آن را برمی‌داریم. این نوع از شنود می‌تواند سربار به سیستم اضافه کند. شکل ۴ برنامه درایور را نشان می‌دهد که مطابق طرز کار netfilter یک هوک به آن اضافه کرده‌ایم. در ادامه شکل ۵ ملزومات ساخت ماژول را نشان می‌دهد. در شکل ۶ ما درایور را می‌سازیم و آن را لود می‌کنیم، سپس برای تبادل بسته از دستور ping استفاده می‌کنیم تا ترافیک داشته باشیم. در نهایت درایور را برمی‌داریم و در شکل ۷ محتوایات فایلی که برای لاگ بسته‌ها مشخص کردیم را می‌بینیم. برای اطمینان پیام‌های کرنل را هم نگاه می‌کنیم و لاگ‌های نصب و حذف درایور خود را می‌بینیم. همه فایل‌ها به پیوست گزارش ارسال شده است. یک مستند مناسب از netfilter نیز در [۱] آمده است.

```
nano netfilter.c
GNU nano 7.2 netfilter.c *
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
#include <linux/fs.h>
#include <linux/uaccess.h>

#define LOG_FILE "/var/log/net_traffic.log"

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Network Traffic Logger using Netfilter for OS lab report 10: drivers, Arshia Yousefnia");

static struct nf_hook_ops nfho;

static unsigned int hook_func(void *priv, struct sk_buff *skb, const struct nf_hook_state *state) {
    struct iphdr *iph;
    struct file *filp;
    loff_t pos = 0;
    char buf[128];

    if (!skb || !skb->network_header) return NF_ACCEPT;

    iph = ip_hdr(skb);
    if (iph) {
        snprintf(buf, sizeof(buf), "Packet: SRC=%pI4 DST=%pI4 PROTO=%u\n", &iph->saddr, &iph->daddr, iph->protocol);

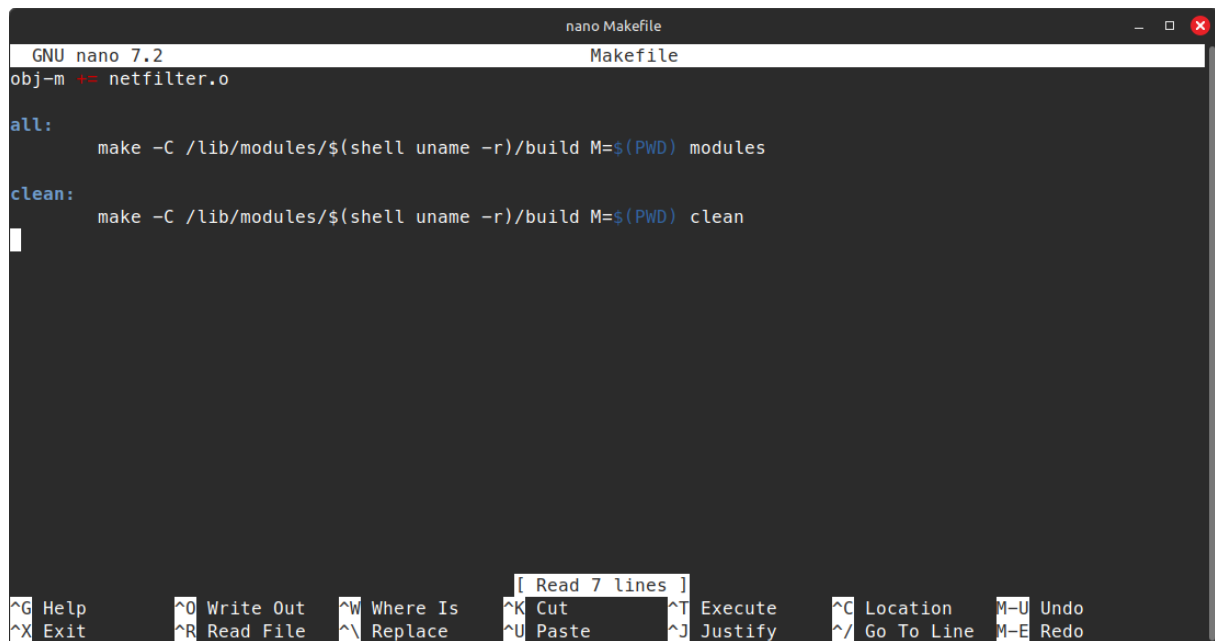
        filp = filp_open(LOG_FILE, O_WRONLY | O_APPEND | O_CREAT, 0644);
        if (!IS_ERR(filp)) {
            kernel_write(filp, buf, strlen(buf), &pos);
            filp_close(filp, NULL);
        }
    }
    return NF_ACCEPT;
}

static int __init net_init(void) {
    nfho.hook = hook_func;
    nfho.hooknum = NF_INET_PRE_ROUTING;
    nfho.pf = PF_INET;
    nfho.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &nfho);
    printk(KERN_INFO "Arshia loaded NetLogger module\n");
    return 0;
}

static void __exit net_exit(void) {
    nf_unregister_net_hook(&init_net, &nfho);
    printk(KERN_INFO "Arshia unloaded NetLogger module\n");
}

module_init(net_init);
module_exit(net_exit);
```

شکل ۴: برنامه درایور آزمایش



The image shows a terminal window titled "nano Makefile" running GNU nano 7.2. The editor is editing a file named "Makefile". The content of the file is as follows:

```
obj-m += netfilter.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 bottom of the window displays the nano editor's command shortcuts:

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute	^C Location	M-U Undo
^X Exit	^R Read File	^_\ Replace	^U Paste	^J Justify	^/_ Go To Line	M-E Redo

شکل ۵: دستورالعمل ساخت یا Makefile

```
arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
> ls
Makefile netfilter.c
arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
> make
make -C /lib/modules/6.8.0-71-generic/build M=/home/arshia/Desktop/report10-codes/netfilter modules
make[1]: Entering directory '/usr/src/linux-headers-6.8.0-71-generic'
warning: the compiler differs from the one used to build the kernel
The kernel was built by: x86_64-linux-gnu-gcc-13 (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0
You are using: gcc-13 (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0
CC [M] /home/arshia/Desktop/report10-codes/netfilter/netfilter.o
MODPOST /home/arshia/Desktop/report10-codes/netfilter/Module.symvers
CC [M] /home/arshia/Desktop/report10-codes/netfilter/netfilter.mod.o
LD [M] /home/arshia/Desktop/report10-codes/netfilter/netfilter.ko
BTF [M] /home/arshia/Desktop/report10-codes/netfilter/netfilter.ko
Skipping BTF generation for /home/arshia/Desktop/report10-codes/netfilter/netfilter.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-6.8.0-71-generic'
arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
> sudo insmod netfilter.ko
arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
> ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data:
64 bytes from 1.1.1.1: icmp_seq=1 ttl=46 time=193 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=46 time=229 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=46 time=238 ms
64 bytes from 1.1.1.1: icmp_seq=4 ttl=46 time=263 ms
64 bytes from 1.1.1.1: icmp_seq=5 ttl=46 time=284 ms
64 bytes from 1.1.1.1: icmp_seq=6 ttl=46 time=128 ms
64 bytes from 1.1.1.1: icmp_seq=7 ttl=46 time=140 ms
64 bytes from 1.1.1.1: icmp_seq=8 ttl=46 time=148 ms
64 bytes from 1.1.1.1: icmp_seq=9 ttl=46 time=136 ms
64 bytes from 1.1.1.1: icmp_seq=10 ttl=46 time=208 ms
64 bytes from 1.1.1.1: icmp_seq=11 ttl=46 time=216 ms
64 bytes from 1.1.1.1: icmp_seq=12 ttl=46 time=250 ms
64 bytes from 1.1.1.1: icmp_seq=13 ttl=46 time=260 ms
64 bytes from 1.1.1.1: icmp_seq=14 ttl=46 time=299 ms
64 bytes from 1.1.1.1: icmp_seq=15 ttl=46 time=320 ms
64 bytes from 1.1.1.1: icmp_seq=16 ttl=46 time=125 ms
^C
--- 1.1.1.1 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 15018ms
rtt min/avg/max/mdev = 124.611/214.737/319.817/62.152 ms
arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
> sudo rmmod netfilter.ko
arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
>
```

شکل ۶: ساخت و بارگذاری ماژول و تبادل بسته با اینترنت

```
arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
> cat /var/log/net_traffic.log
Packet: SRC=1.1.1.1 DST=10.68.229.225 PROTO=1
Packet: SRC=10.68.229.26 DST=10.68.229.225 PROTO=17
Packet: SRC=107.178.240.159 DST=10.68.229.225 PROTO=6
Packet: SRC=107.178.240.159 DST=10.68.229.225 PROTO=6
Packet: SRC=142.250.187.67 DST=10.68.229.225 PROTO=17
Packet: SRC=142.250.187.67 DST=10.68.229.225 PROTO=17
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=13.107.246.62 DST=10.68.229.225 PROTO=6
Packet: SRC=45.80.228.185 DST=10.68.229.225 PROTO=6
Packet: SRC=172.67.69.195 DST=10.68.229.225 PROTO=6
Packet: SRC=172.67.69.195 DST=10.68.229.225 PROTO=6
Packet: SRC=172.67.69.195 DST=10.68.229.225 PROTO=6
Packet: SRC=172.67.69.195 DST=10.68.229.225 PROTO=6
Packet: SRC=1.1.1.1 DST=10.68.229.225 PROTO=1

arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
> dmesg | tail -n 10
[ 1138.933530] ? __pfx_kthread+0x10/0x10
[ 1138.933545] ret_from_fork+0x44/0x70
[ 1138.933557] ? __pfx_kthread+0x10/0x10
[ 1138.933570] ret_from_fork_asm+0x1b/0x30
[ 1138.933589] </TASK>
[ 1138.933594] ---[ end trace 0000000000000000 ]---
[ 1138.934114] softirq: huh, entered softirq 6 TASKLET 0000000052b2e85d with preempt_count 00000100, exited with 00000000
0?
[ 1151.278713] Arshia unloaded NetLogger module
[ 1314.738815] Arshia loaded NetLogger module
[ 1376.546138] Arshia unloaded NetLogger module

arshia@arshia-Notebook: ~/Desktop/report10-codes/netfilter
```

شکل ۷: مشاهده فایل لاگ بسته‌های دیده شده از درایور و پیام‌های کرنل

[١] URL: <https://en.wikipedia.org/wiki/Netfilter>.