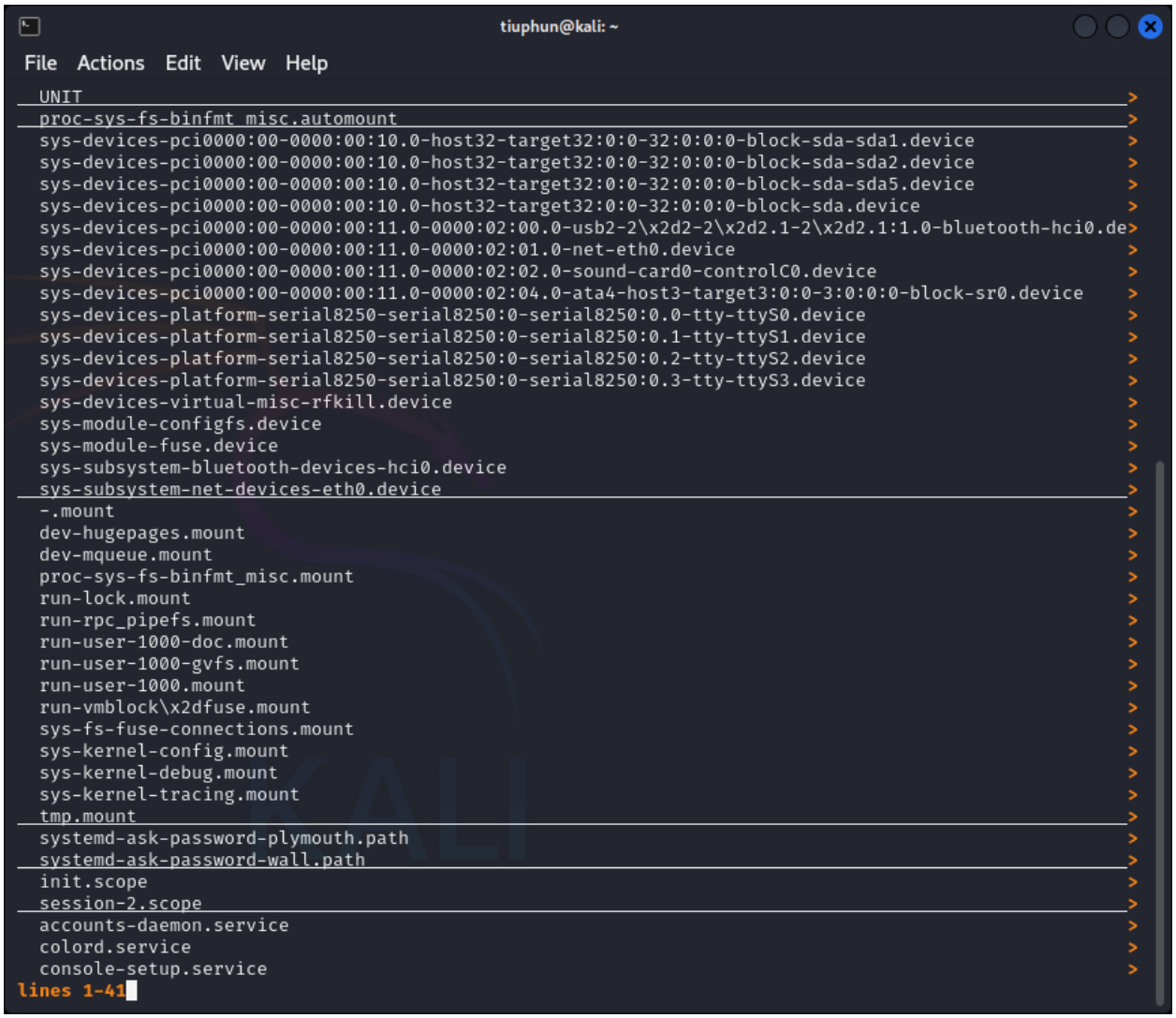
Lab 09 Report

Nguyen Tieu Phuong 20210692

FYI: All exercises are done in my Kali Linux VM, @tiuphun (local user account).

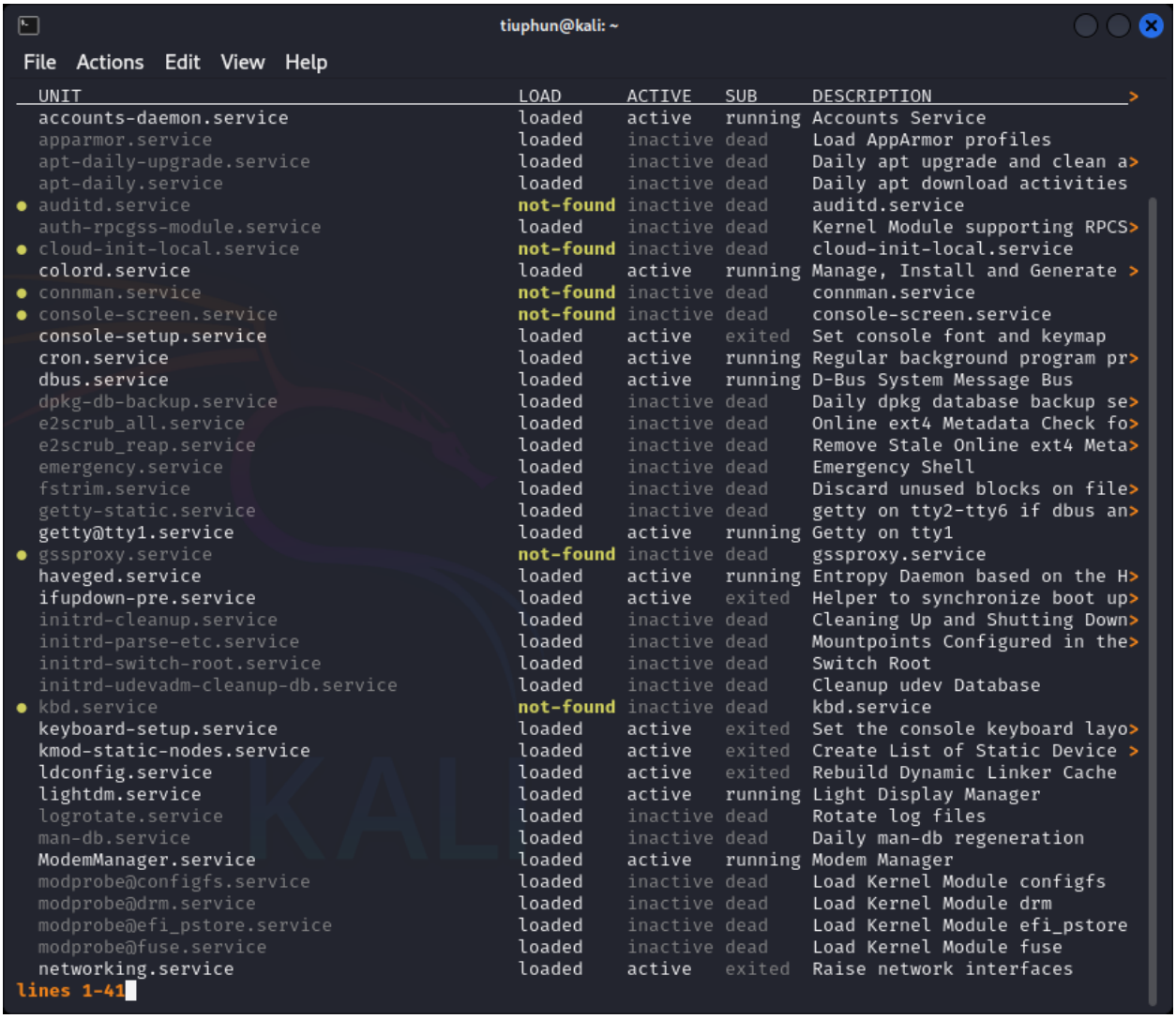
Exercise 1. Manage services using systemd:

1. Check whether systemd is available in your Linux system (systemctl command)

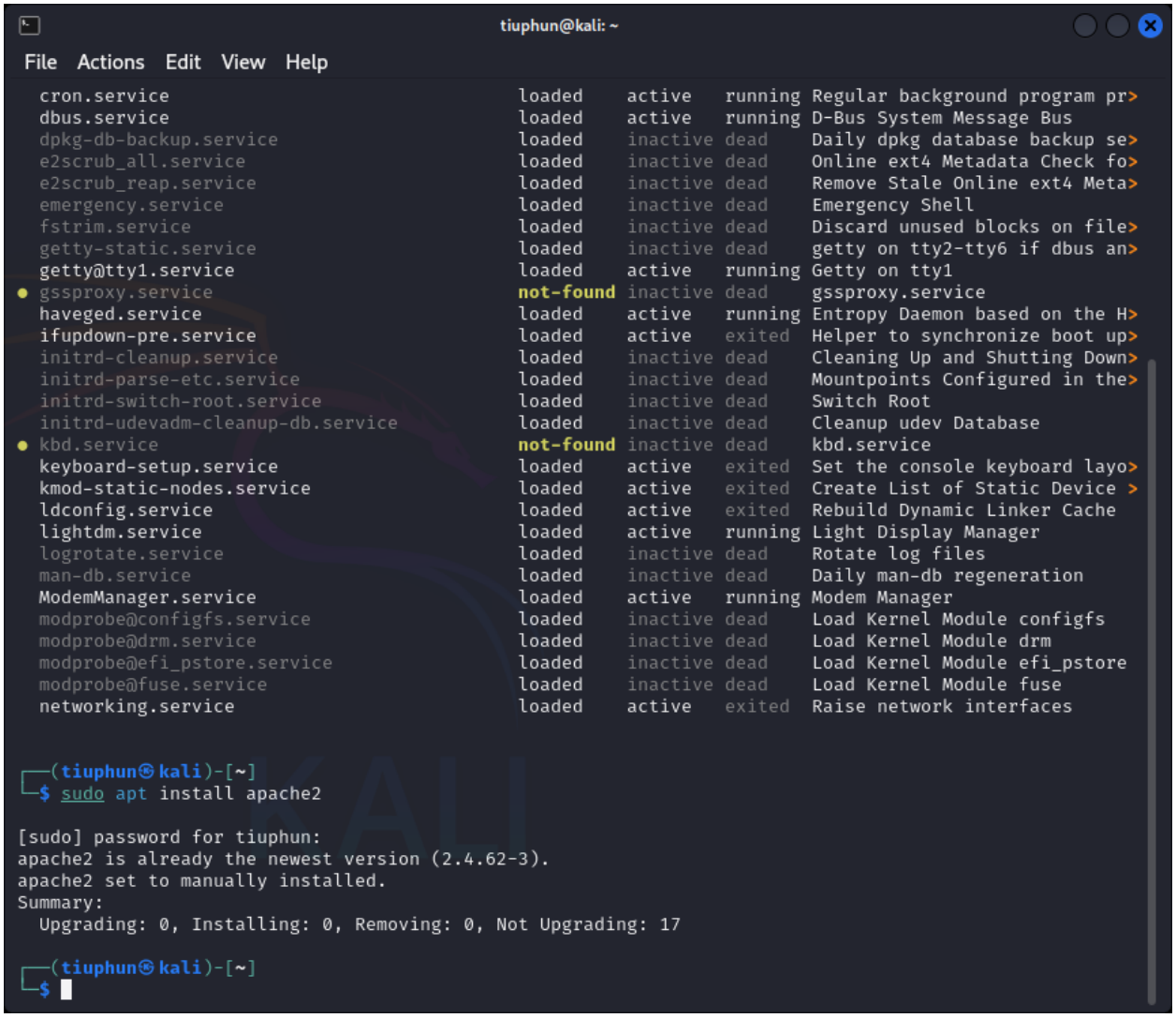


2. List all services in your Linux OS

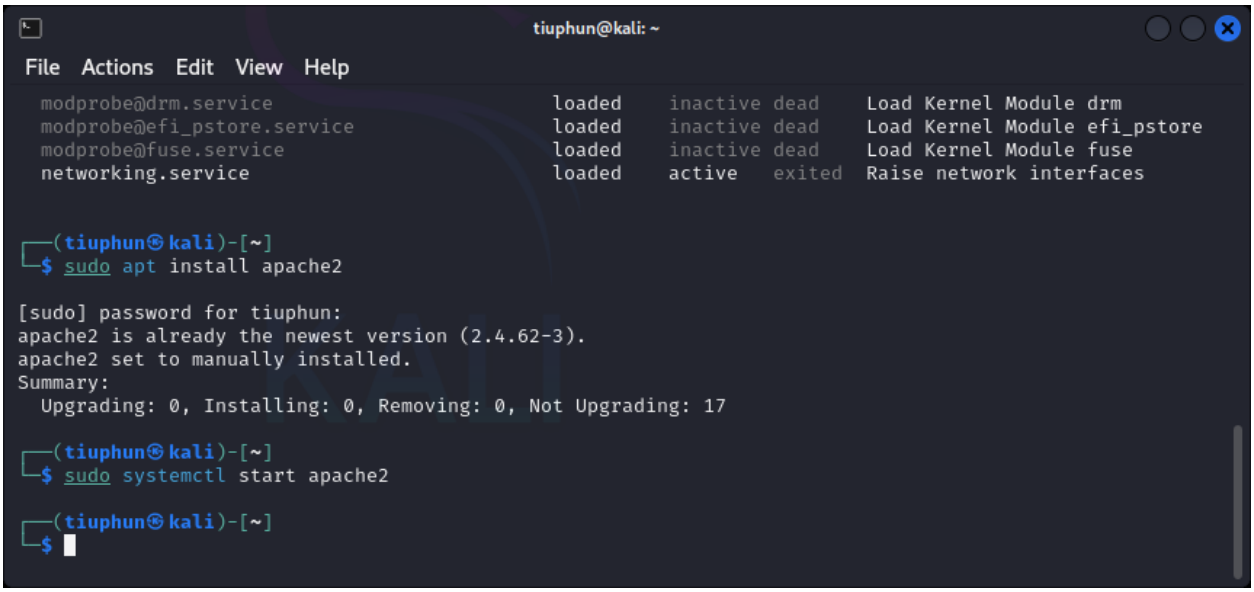
$ systemctl list-units --type=service –all

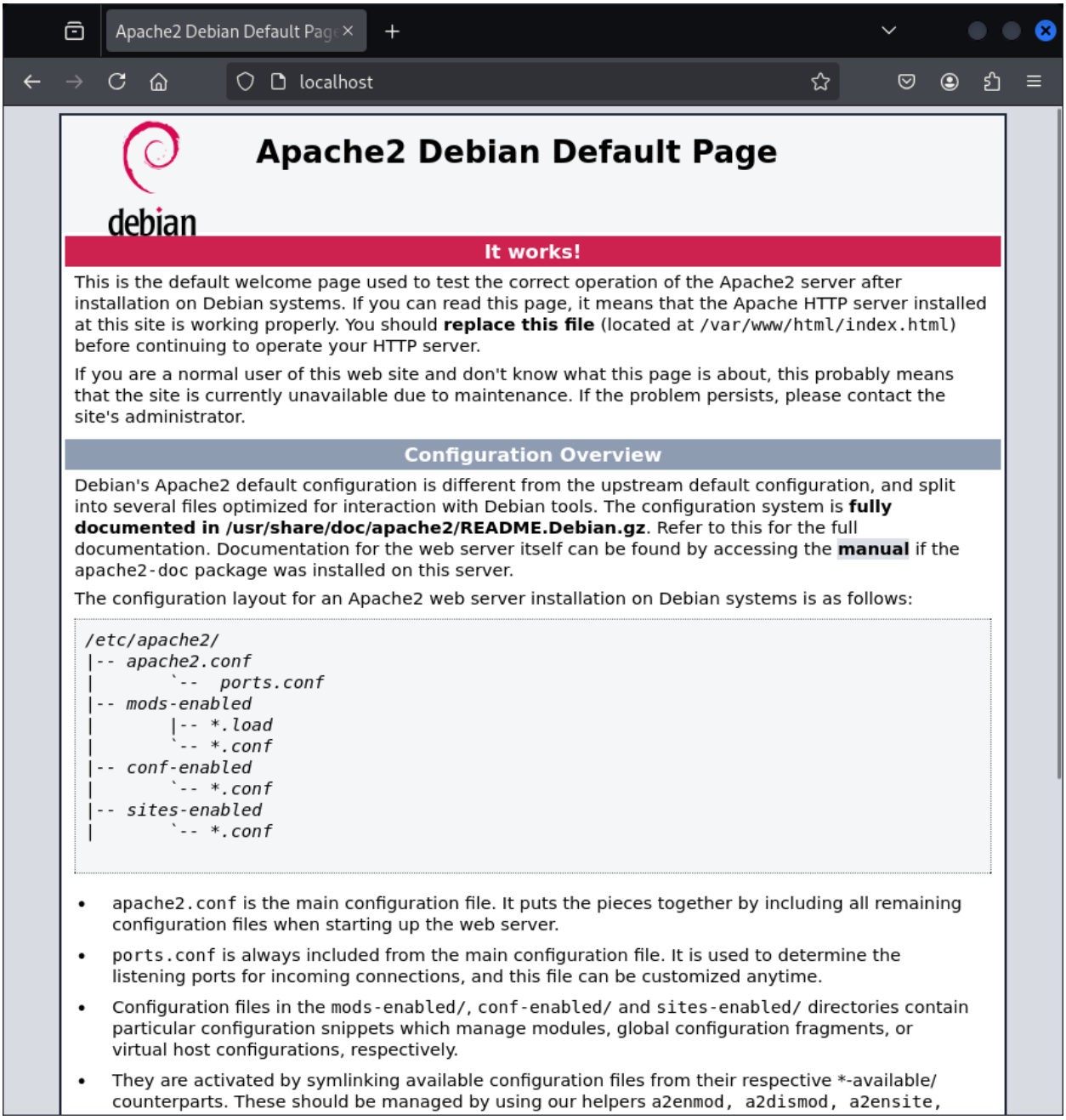


3. Check whether Apache server is installed or not (httpd). If not, please install Apache server

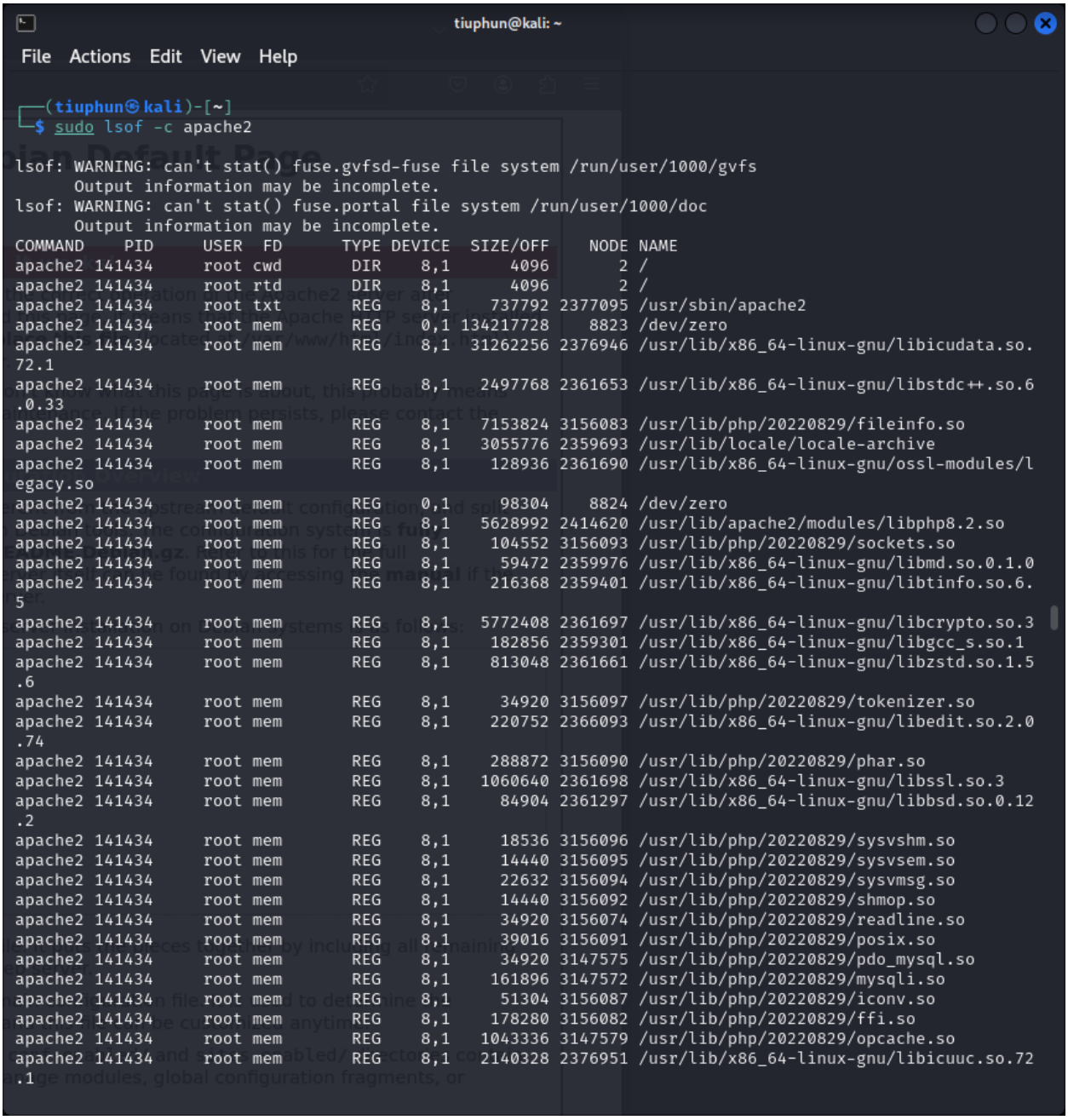


4. Start Apache server


tiuphun@kali: ~
File Actions Edit View Help
modprobe@drm.service loaded inactive dead Load Kernel Module drm
modprobe@efi_pstore.service loaded inactive dead Load Kernel Module efi_pstore
modprobe@fuse.service loaded inactive dead Load Kernel Module fuse
networking.service loaded active exited Raise network interfaces
┌──(tiuphun㉿kali)-[~]
└─$ sudo apt install apache2
[sudo] password for tiuphun:
apache2 is already the newest version (2.4.62-3).
apache2 set to manually installed.
Summary:
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 17
┌──(tiuphun㉿kali)-[~]
└─$ sudo systemctl start apache2
┌──(tiuphun㉿kali)-[~]
└─$



5. Show all using files of Apache server



6. Try to access to your web server from another computer using IP address. If we want access the web server using the domain name (test.com), which steps should we do?

- **Edit the `hosts` file on the client machine**:

On the computer trying to access the server, edit its `/etc/hosts` file to map `test.com` to the Apache server's IP address:

```
sudo nano /etc/hosts
```

Add this line (replace `SERVER_IP` with your Apache server's IP address):

```
SERVER_IP test.com
```

Save and exit (`Ctrl + O`, then `Enter`, and `Ctrl + X`).

- **Set Up Virtual Host on the Apache Server**: On the server, configure Apache to recognize `test.com`:

```
sudo nano /etc/apache2/sites-available/test.com.conf
```

Add the following content:

```
<VirtualHost *:80>
    ServerName test.com
    DocumentRoot /var/www/html
</VirtualHost>
```

Save and exit the file.

- **Enable the Virtual Host**:

```
sudo a2ensite test.com.conf
sudo systemctl reload apache2
```

- **Access the Server**: From another computer, open a browser and visit `http://test.com`.

7. Turn off the Apache server's service, try to access to the Apache server from another computer

8. Configure your system so the Apache server boots up along with your computer

9. Restart your computer using the command line. Login to your system. How could we know whether the Apache server is working or not?

We run **sudo systemctl status apache2** or go to http://localhost or from another computer go to http://server_IP

Exercise 2. Create and manage a new service

1. Create a new file server.php with the following content. Try to guess the purpose of this file? (Hint: you can do it after executing the step 3)

<?php

$sock = socket_create(AF_INET, SOCK_DGRAM, SOL_UDP);

socket_bind($sock, '0.0.0.0', 10000);

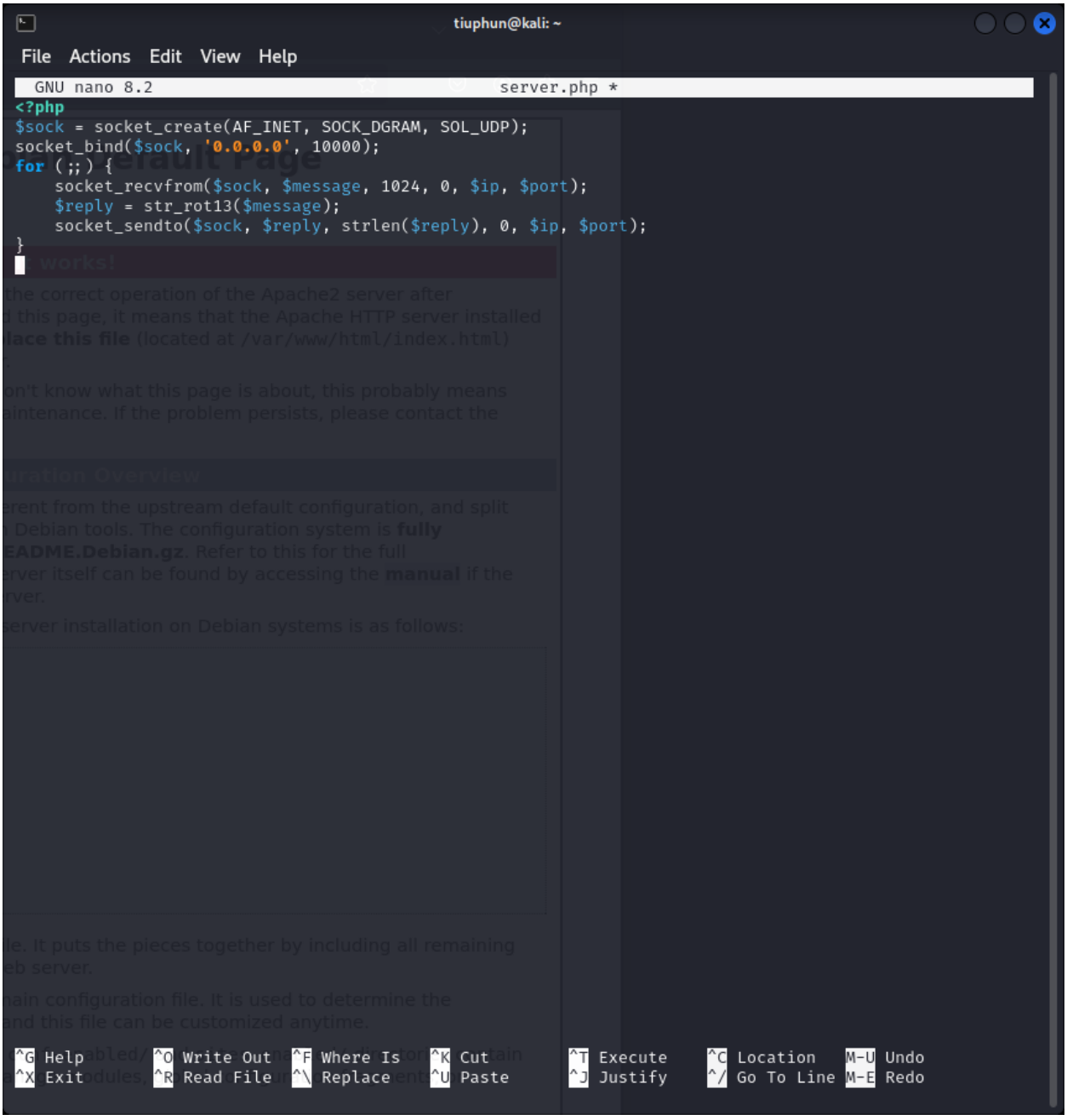
for (;;) {

socket_recvfrom($sock, $message, 1024, 0, $ip, $port);

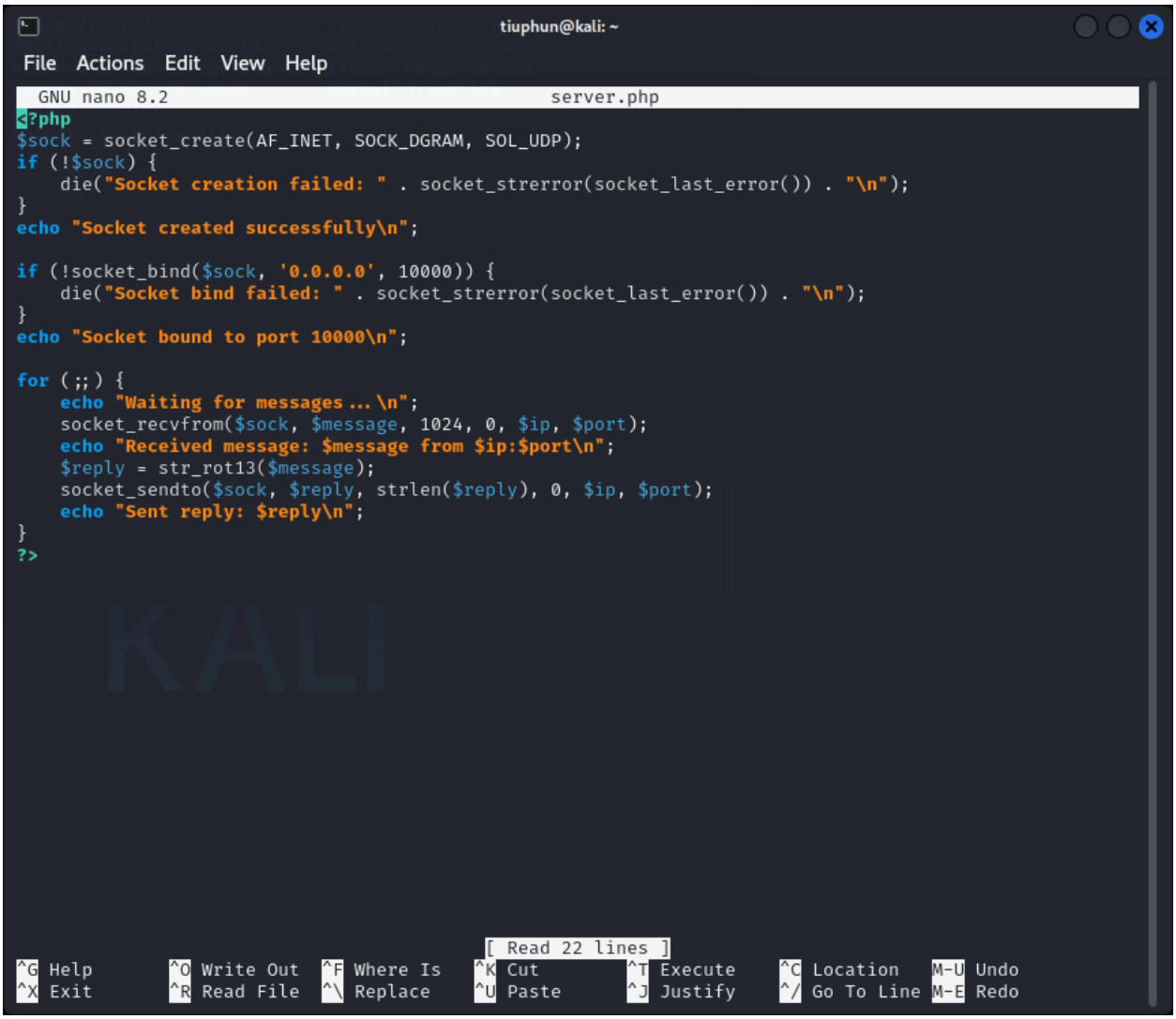
$reply = str_rot13($message);

socket_sendto($sock, $reply, strlen($reply), 0, $ip, $port);

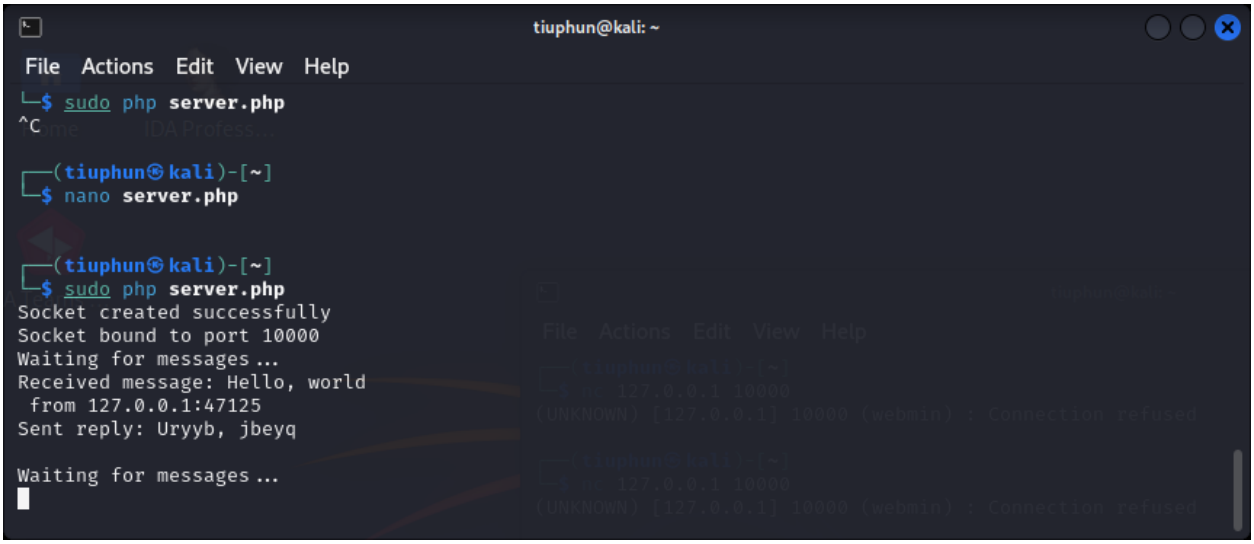
}



The script above doesn't work. I tried to fix and replace with the new code:

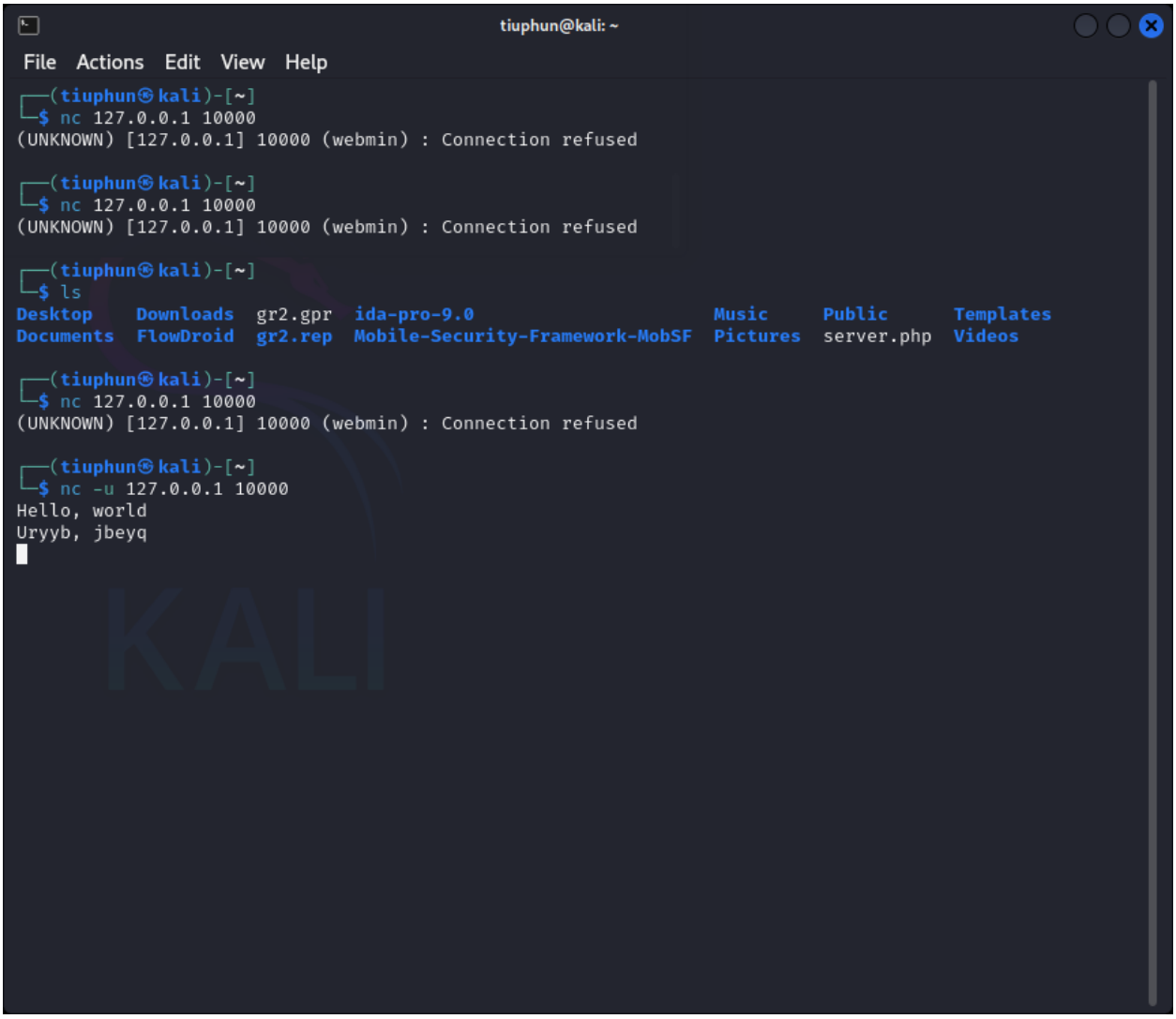


2. Run "php server.php" to start the php server. (You might need to install php server – php_cli)



3. Check the status of php server by login to another account (different terminal). Run the command “nc 127.0.0.1 10000” and then type “Hello world” after connecting with local php server. Is the printed message “Uryyb, jbeyq”?

Yes

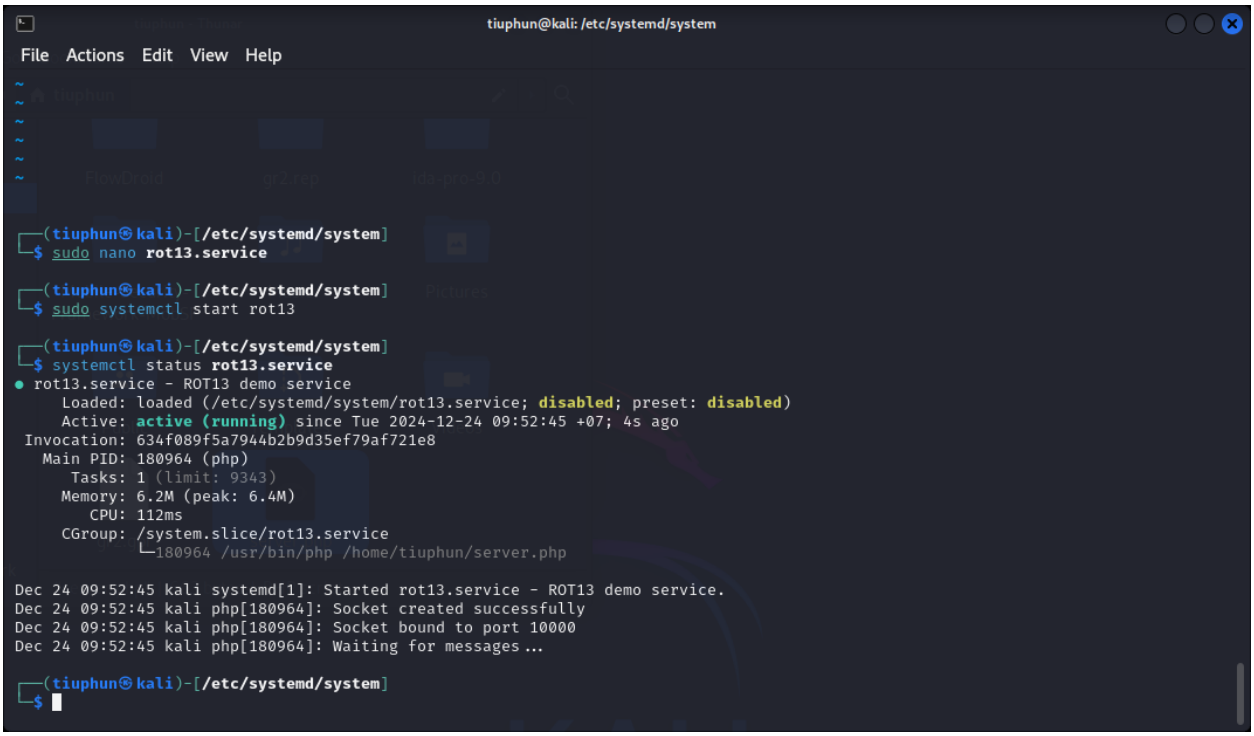


4. To convert the PHP server to a service, create a file /etc/systemd/system/rot13.service with the following content

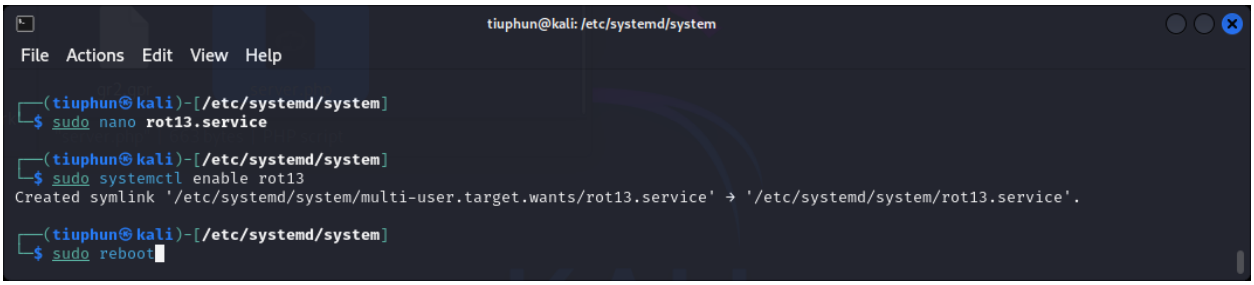
[Unit] Description=ROT13 demo service After=network.target StartLimitIntervalSec=0[Service]

Type=simple Restart=always RestartSec=1 User=centos ExecStart=/usr/bin/env php /path/to/server.php [Install] WantedBy=multi-user.target

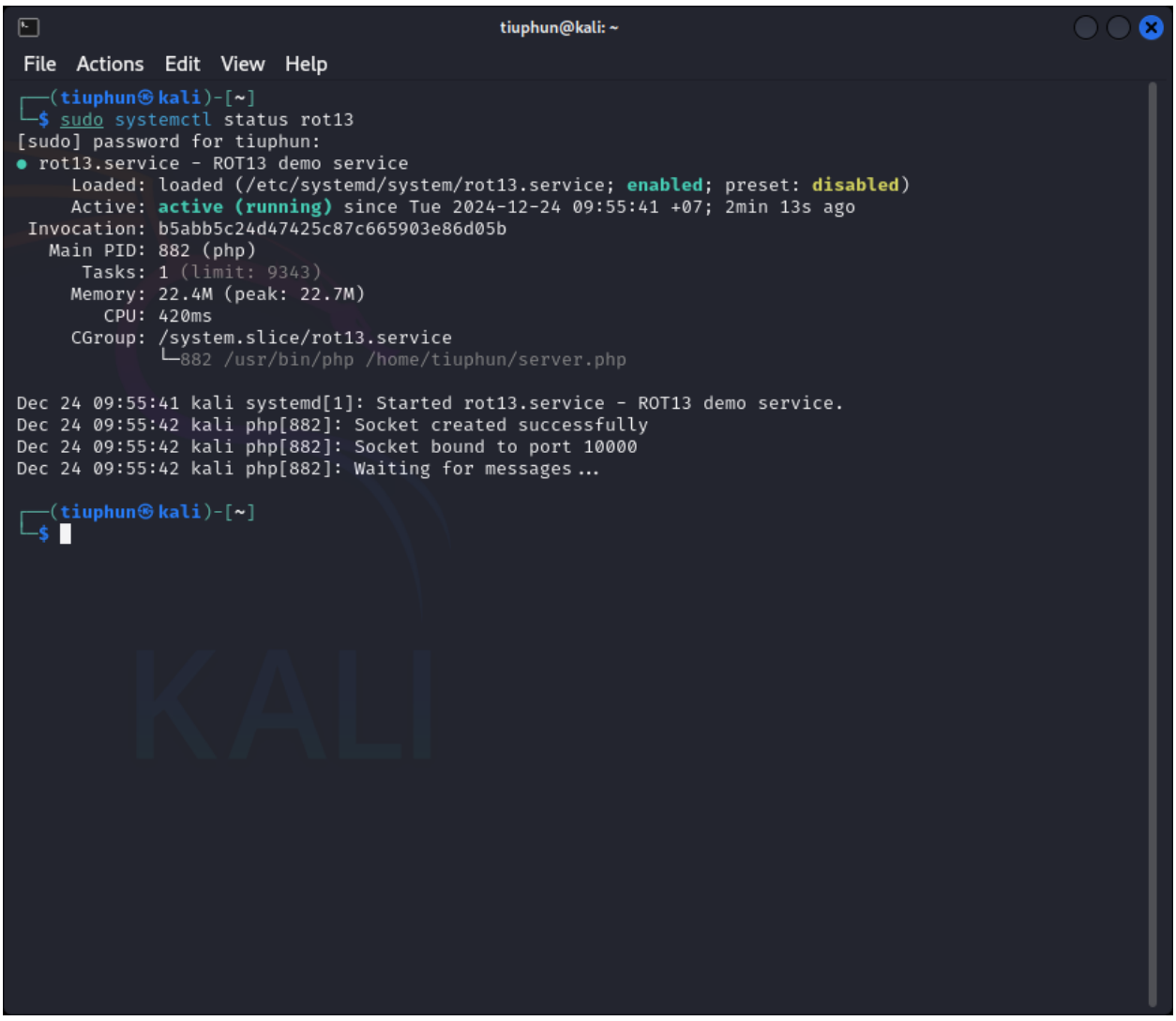
5. Try to start the service rot13 using systemctl



6. Use systemctl to configure rot13 so it can boot up along with the Linux OS



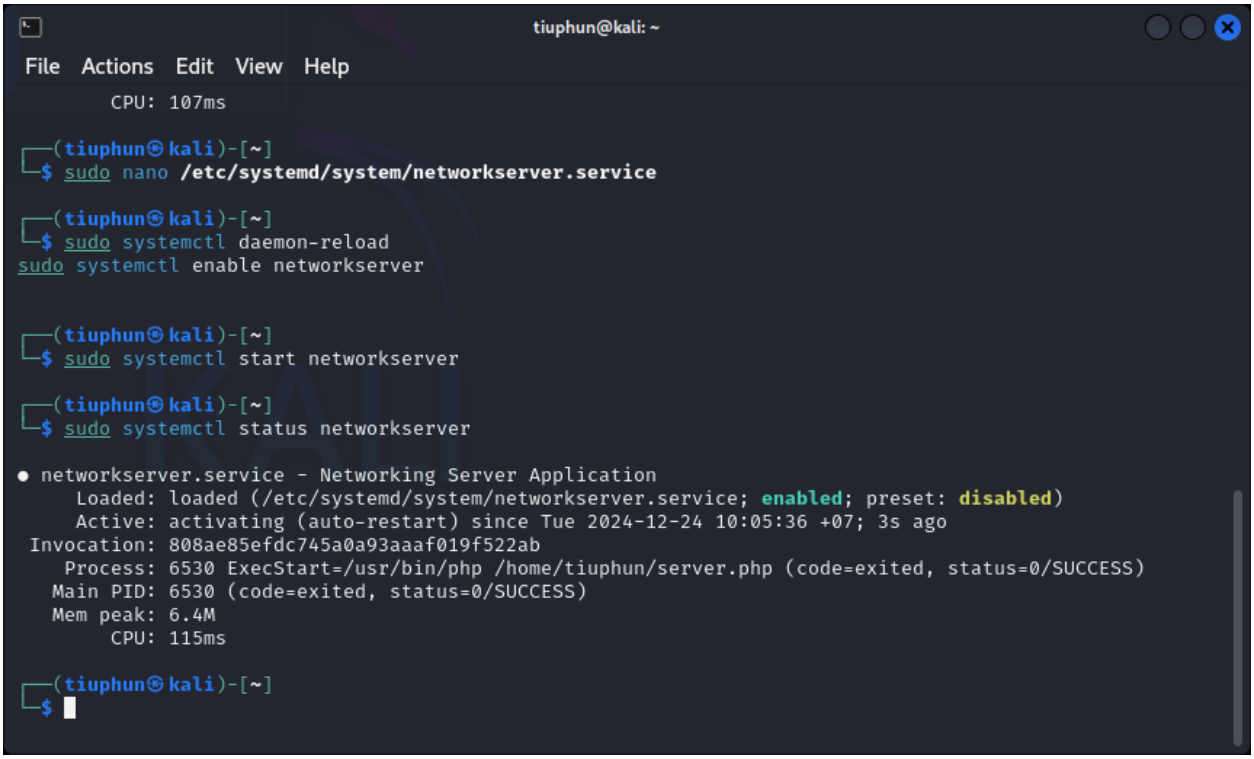
Reboot the system and verify that the service boot up with the OS:



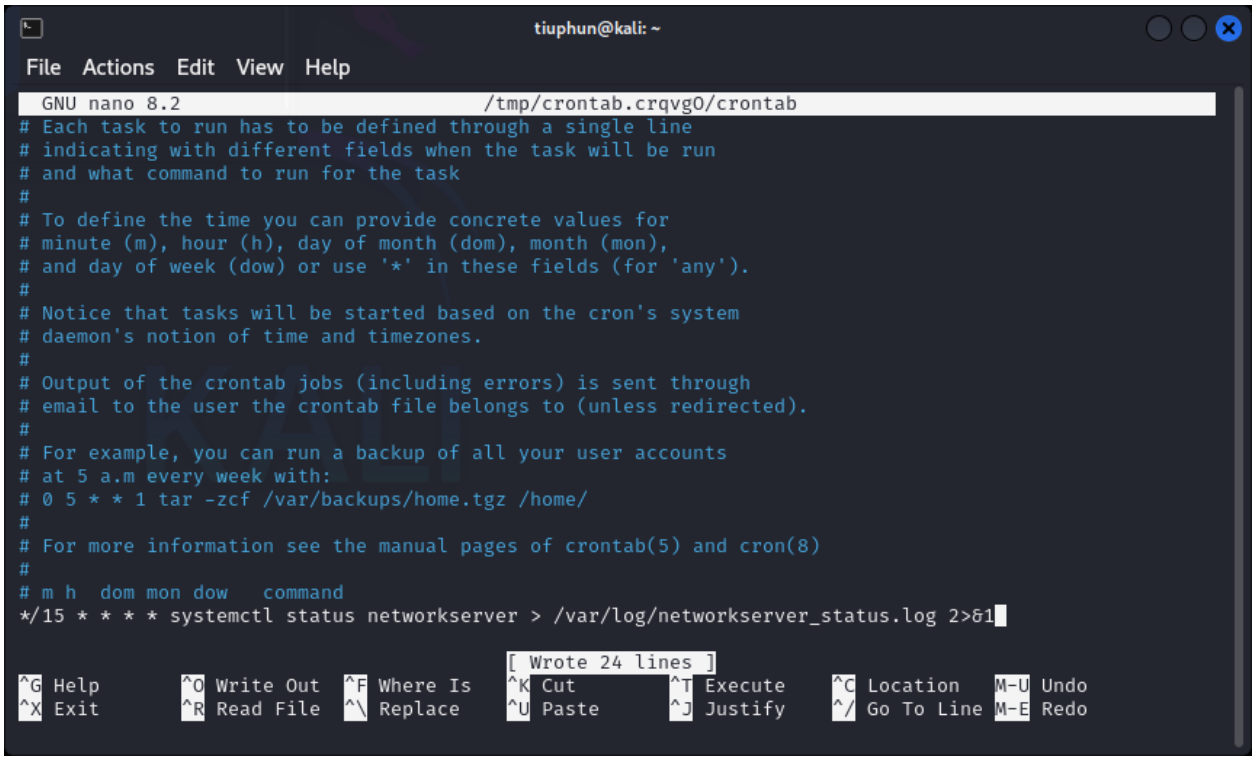
It works!

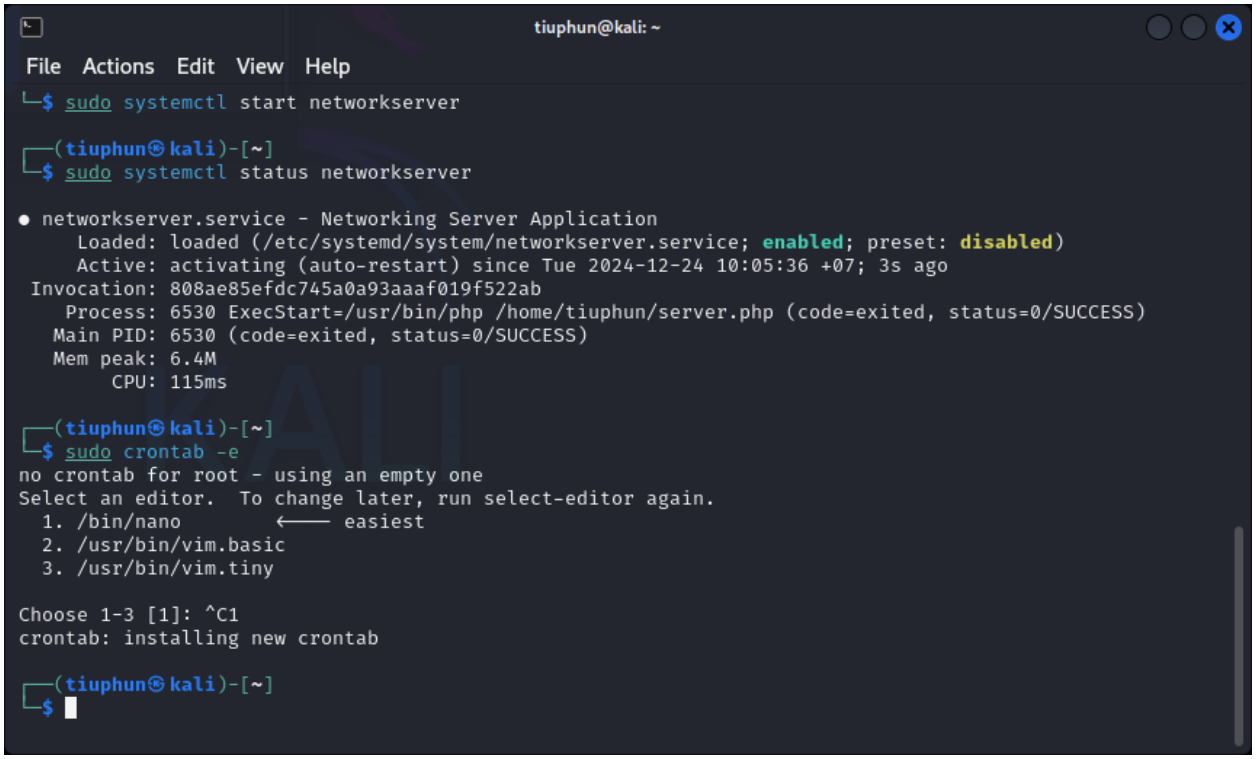
Exercise 3. Configure your networking server application

1. Configure your networking program (on server side) so it will become a Linux service. Change your system so your server will start when your computer boots up



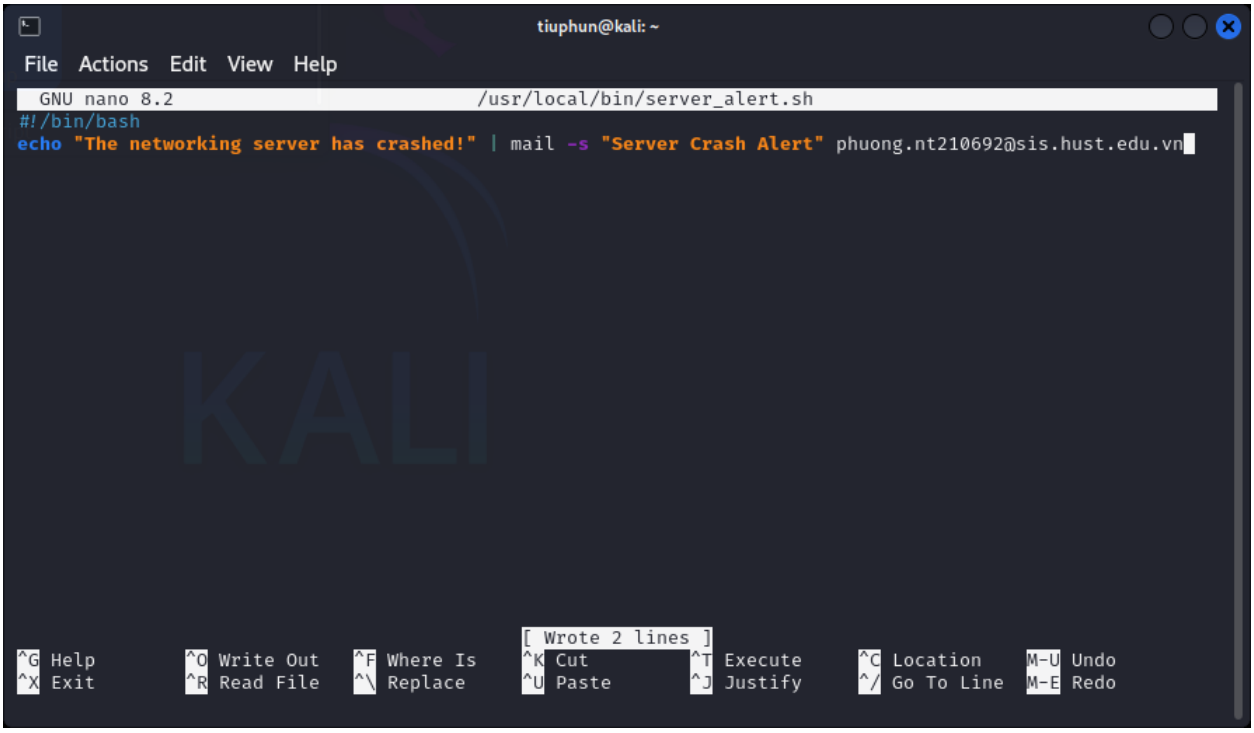
2. Regularly write down your server status to log files (For example: every 15 minutes)



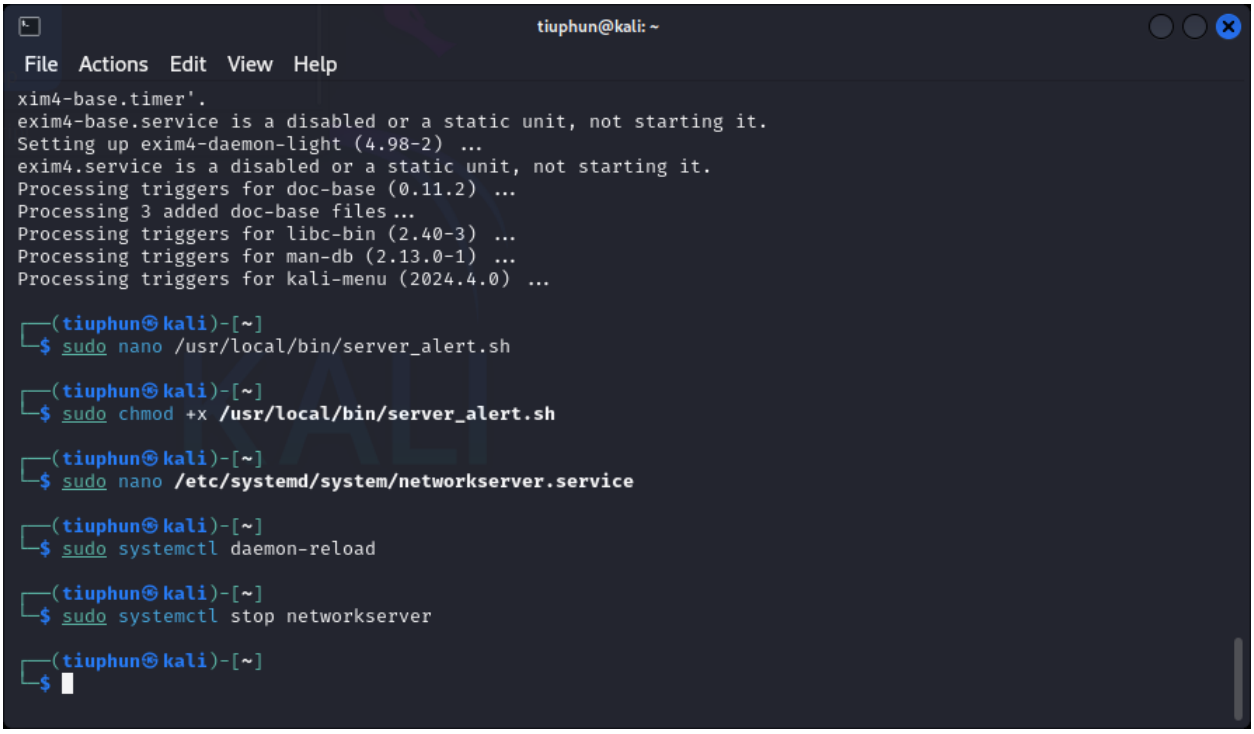


3. Configure that your PC will send you an email if your server process crashes

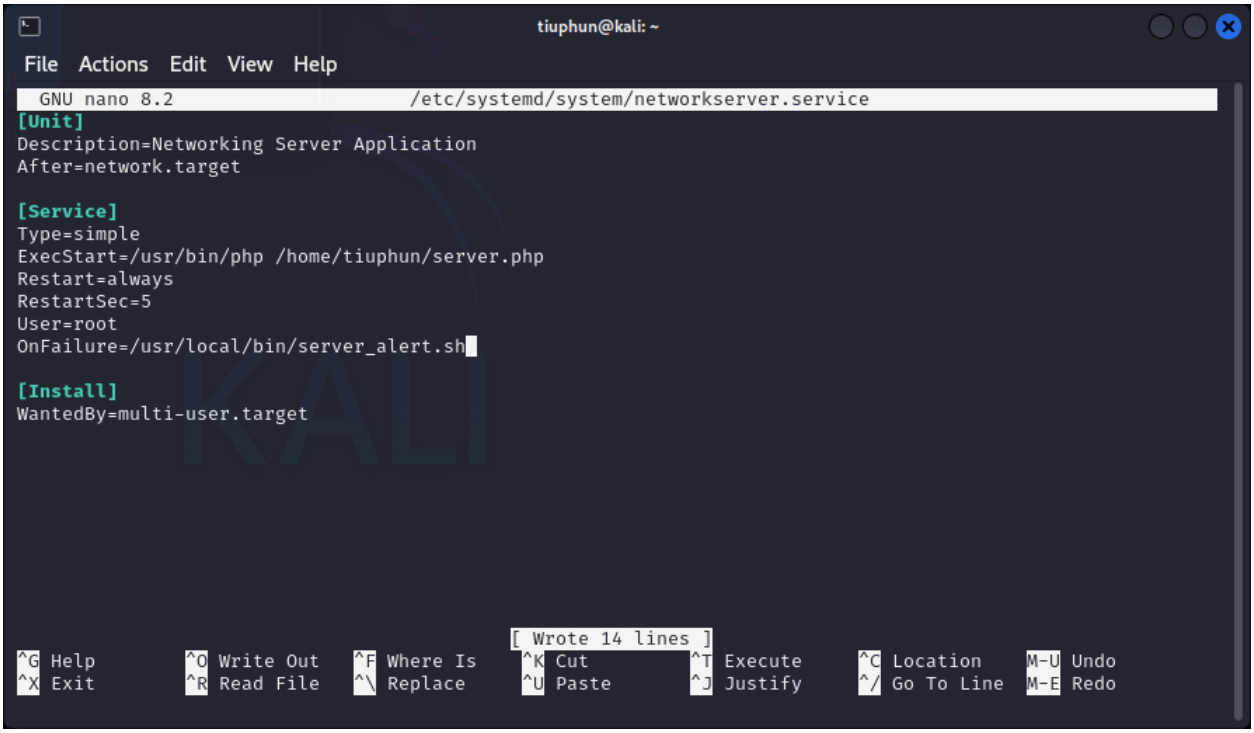
Create a new script for alert



Set run permissions



Reconfig service to run the alert script in case of failure



THE END