

# **ITAS 181 Lab 14**

## **Linux System Monitoring and Security Management**

Raj Singh

04 December 2023

## **Project Overview**

This series of projects was designed to advance my proficiency in Linux system administration, focusing on performance monitoring, network services analysis, and firewall configuration. Across Fedora and Ubuntu systems, I engaged with sysstat tools to monitor and interpret system resources, utilized nmap for service examination, and skillfully managed firewall rules through iptables and UFW. Each project reinforced the importance of precise documentation and adherence to best practices, enhancing my capabilities in securing and optimizing Linux environments. The exercises not only fortified my technical expertise but also underscored the critical nature of safeguarding system integrity and network security.

### **Project 14-1**

**Objective:** Monitor system performance using sysstat tools.

This guide is designed for learners and professionals seeking practical experience in Linux system administration. It encompasses key aspects of system management and security across Fedora and Ubuntu Linux systems. The guide is structured to provide step-by-step instructions on:

1. **Monitoring System Performance:** Utilizing tools like sysstat, top, and vmstat, learners will gain insights into CPU usage, memory, swap space, and I/O statistics, crucial for maintaining optimal system performance.
2. **Analyzing Network Services:** Through tools like nmap, participants will learn to assess active network services, understand port usage, and identify potential vulnerabilities, enhancing their network management skills.
3. **Configuring Firewalls:** The guide covers detailed processes for setting up and managing firewall rules using iptables, UFW, and Firewalld. This segment focuses on securing the system by controlling network traffic and safeguarding against unauthorized access.

Each section is crafted to build a comprehensive skill set in Linux system administration, ensuring learners are well-equipped to handle real-world scenarios in system monitoring and security management. The guide serves as a valuable resource for those aiming to fortify their expertise in Linux environments.

## **Project 14-2**

**Objective:** Monitor system performance.

The project includes a series of tasks using various command-line utilities to assess different aspects of system health.

- **Accessing the System:** You start by logging into your Fedora Linux VM via a command-line terminal.
- **Using `top`:** This utility provides real-time information on running processes, memory usage, and system load averages. You're tasked with noting down key statistics like the number of running processes, total and used memory, buffer usage, total and used swap memory, and load averages over 1, 5, and 15 minutes.
- **Exiting `top`:** You'll learn how to exit the utility.
- **Employing `free`:** This command offers details about memory and swap usage. You compare its output with the data from `top`.
- **Utilizing `vmstat`:** This tool gives additional insights into memory, swap, and system performance, compared again with previous data.
- **Checking with `uptime`:** This command helps in understanding the system's load average, which you'll compare with the data from `top`.
- **Installing `iostat`:** Here, you install and use `iostat` to identify processes consuming significant storage device I/O.
- **Exploring with `iostat`:** You actively monitor storage device I/O usage by different processes.
- **Installing `ioping`:** This step involves installing `ioping`, a tool for monitoring I/O performance.
- **Testing with `ioping`:** You use `ioping` to measure and note the average I/O values of your storage device, understanding why it's important to measure these under normal system performance.
- **Wrapping Up:** The project concludes with logging out of the shell.

```

rajgurshersingh — root@fedora:~ — ssh raj@172.16.237.137 — 84x27
[root@fedora ~]# ioping /dev/nvme0n1p3
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=1 time=2.13 ms (warmup)
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=2 time=2.77 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=3 time=1.27 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=4 time=2.26 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=5 time=1.18 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=6 time=1.91 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=7 time=1.59 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=8 time=1.43 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=9 time=701.7 us (fast)
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=10 time=1.11 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=11 time=1.23 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=12 time=1.98 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=13 time=2.16 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=14 time=968.1 us
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=15 time=1.19 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=16 time=1.93 ms
4 KiB <<< /dev/nvme0n1p3 (block device 18.4 GiB): request=17 time=1.13 ms
^C
--- /dev/nvme0n1p3 (block device 18.4 GiB) ioping statistics ---
16 requests completed in 24.8 ms, 64 KiB read, 644 iops, 2.52 MiB/s
generated 17 requests in 16.3 s, 68 KiB, 1 iops, 4.17 KiB/s
min/avg/max/mdev = 701.7 us / 1.55 ms / 2.77 ms / 543.5 us
[root@fedora ~]#

```

Figure 1 - I/O Performance Metrics Captured by ioping

## Project 14-3

**Objective:** Examine network services.

We look into examining network services on an Ubuntu Linux virtual machine. The project utilizes the nmap utility and the /etc/services file to provide a comprehensive view of the network services running on the system.

- **Starting Up:** You begin by booting your Ubuntu Linux VM and logging in as the root user.
- **Installing Nmap:** The first task is to install the nmap utility, a powerful tool for network discovery and security auditing.
- **Stopping Services:** You practice managing services by stopping the Samba (smbd.service) and Very Secure FTP (vsftpd.service) daemons.
- **Initial Nmap Scan:** Using nmap, you conduct an initial scan of the local system (127.0.0.1) to identify active services and their corresponding ports. This includes identifying the service running on port 631/tcp.
- **Investigating Services:** You further explore the services file to find the full name of the service running on port 631 by using the grep command.

- **Restarting Services:** The next step is to restart the previously stopped services (Samba and VSFTP).
- **Post-Restart Nmap Scan:** A second nmap scan follows to observe changes in open ports resulting from restarting the services.
- **Conclusion:** The project concludes with logging out of the shell.

```


root@raj:~# nmap -sS itas.ca
Starting Nmap 7.80 ( https://nmap.org ) at 2023-12-04 21:11 UTC
Nmap scan report for itas.ca (10.104.192.242)
Host is up (0.71s latency).
Other addresses for itas.ca (not scanned): 10.104.192.243 10.104.192.241 104.128.240.83 10.104.142.192 1
04.128.240.82 104.128.240.81 10.104.142.193 10.104.142.191 2002:8e19:61f5::8e19:61f5 2002:8e19:61f3::8e1
9:61f3
rDNS record for 10.104.192.242: dc2.itas.ca
Not shown: 988 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
3389/tcp  open  ms-wbt-server

Nmap done: 1 IP address (1 host up) scanned in 1.33 seconds
root@raj:~#

```

Figure 2 - Network Service Scan Results: Utilizing nmap on Ubuntu Linux

The nmap scan results shown in the image provide insights into the Ubuntu Linux virtual machine's network configuration. This configuration suggests a machine with significant integration into a Windows-based network infrastructure, potentially serving roles in domain name resolution, directory services, remote procedure calls, and secure remote access. The presence of these services indicates a multi-faceted utility of the machine and also underscores the necessity for stringent security measures given the sensitivity of the services exposed.



```
rajgurshersingh — root@raj: ~ — ssh raj@172.16.237.132 — 104x26
root@raj:~# nmap -sS google.ca
Starting Nmap 7.80 ( https://nmap.org ) at 2023-12-04 21:13 UTC
Nmap scan report for google.ca (142.251.33.67)
Host is up (0.0026s latency).
Other addresses for google.ca (not scanned): 2607:f8b0:400a:805::2003
rDNS record for 142.251.33.67: sea09s28-in-f3.1e100.net
Not shown: 998 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 4.37 seconds
root@raj:~#
```

Figure 3 - nmap Scan of Google.ca: Showing Standard HTTP and Secured HTTPS Services.

The nmap scan depicted in the image shows two open ports for google.ca: port 80, which is used for HTTP (unsecured web traffic), and port 443, used for HTTPS (secured web traffic). The presence of open ports for both HTTP and HTTPS is standard for a web server, indicating that google.ca is configured to serve web content securely, with HTTPS ensuring encrypted communications. The quick scan time suggests a responsive server, and the additional IPv6 address listed implies modern internet protocol support. The domain resolves to an IP address with a reverse DNS of sea09s28-in-f3.1e100.net, which is owned by Google, reflecting the domain's legitimate association with the company's network infrastructure.

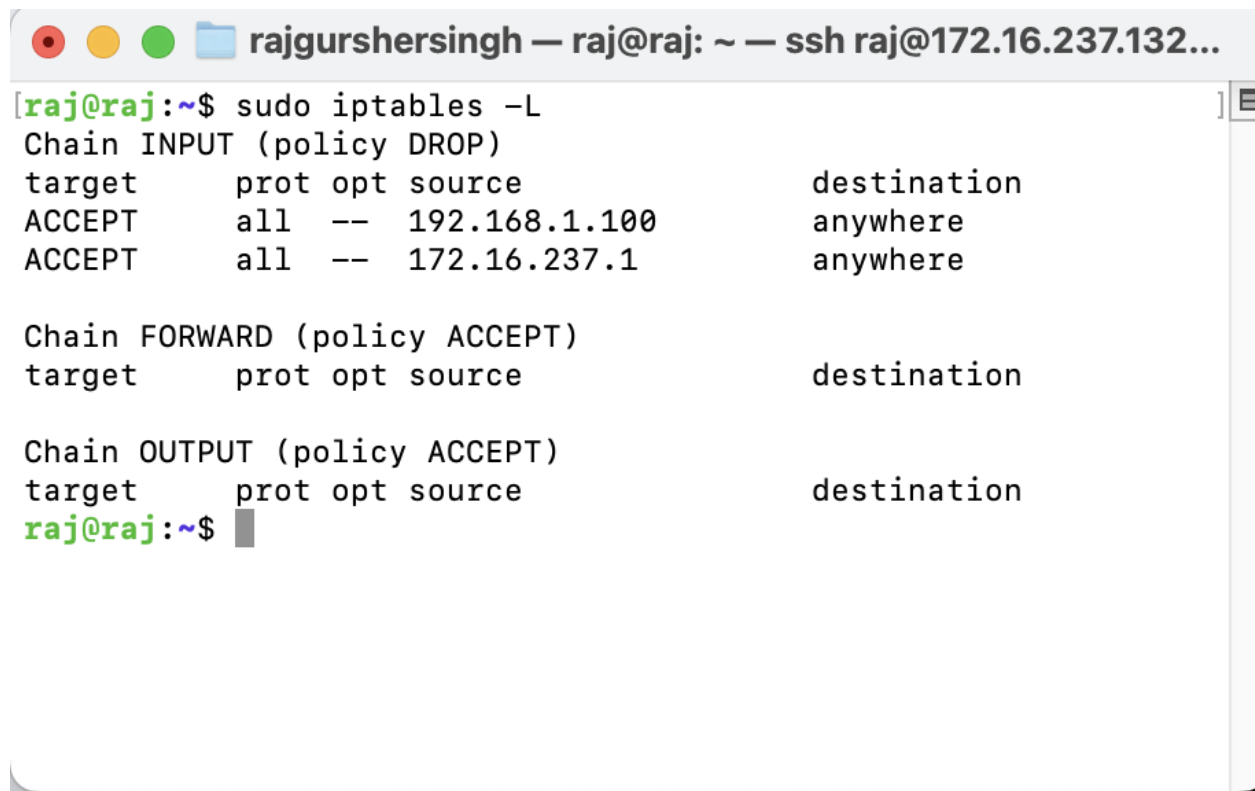
## Project 14-5

**Objective:** Gain practical experience in securing a Linux system by manipulating iptables, the user-space utility program that allows a system administrator to configure the IP packet filter rules of the Linux kernel firewall.

The tasks guide you through:

1. **Logging into Ubuntu VM:** Accessing the system as root.
2. **Inspecting Current iptables Rules:** Reviewing the default policies for INPUT, FORWARD, and OUTPUT chains.
3. **Installing Apache:** Installing the Apache web server, unless previously done.
4. **Testing Apache Access:** Checking if the default Apache page is displayed in a web browser.

5. **Modifying iptables INPUT Policy:** Changing the default INPUT chain policy to DROP to deny incoming traffic.
6. **Verifying iptables Policy Change:** Ensuring the INPUT chain policy is set to DROP.
7. **Testing Connectivity Impact:** Checking if the web page is still accessible after the policy change.
8. **Allowing Specific Traffic:** Adding a rule to accept incoming connections from a specific IP address.
9. **Checking iptables for New Rule:** Verifying the new rule addition under the INPUT chain.
10. **Re-testing Web Access:** Confirming if the web page is now accessible from the allowed IP address.
11. **Inspecting Rules with nftables:** Using nftables to list the current ruleset and examining the INPUT chain.
12. **Resetting iptables Rules:** Flushing current rules and setting the default INPUT policy to ACCEPT.
13. **Activating UFW (Uncomplicated Firewall):** Enabling UFW and observing the changes it makes to iptables.
14. **Configuring UFW Rules:** Allowing traffic for SSH, HTTP, FTP, and NFS services through UFW.
15. **Disabling UFW:** Turning UFW off temporarily.
16. **Exiting the Shell:** Logging out from the root session.

A terminal window titled 'rajgurshersingh — raj@raj: ~ — ssh raj@172.16.237.132...' displays the output of the command 'sudo iptables -L'. The output shows three chains: INPUT (policy DROP), FORWARD (policy ACCEPT), and OUTPUT (policy ACCEPT). The INPUT chain has two rules: one accepting traffic from 192.168.1.100 and another accepting traffic from 172.16.237.1, both with target 'ACCEPT' and destination 'anywhere'. The FORWARD and OUTPUT chains are currently empty.

```
[raj@raj:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source                destination
ACCEPT     all  --  192.168.1.100          anywhere
ACCEPT     all  --  172.16.237.1           anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
raj@raj:~$
```

Figure 4 - Iptables rules in action - Configuring Secure Access from only my IP

the iptables command's output reflects a carefully configured firewall setup on an Ubuntu virtual machine, illustrating the principles of restrictive access for robust security. The INPUT chain's default policy is set to DROP, actively blocking all unsolicited inbound traffic, with the exception of a specified trusted IP address (192.168.1.100) which is permitted. The FORWARD and OUTPUT chains are left open, signifying an allowance for outbound connections and internal network forwarding. This setup is a common security measure, demonstrating how iptables can be fine-tuned to grant access selectively while maintaining a defensive posture against potential network threats.