SKILLS | DevOps and Cloud Computing

Advanced Linux For DevOps

Objective

- Schedule and manage tasks using cron jobs and background processes.
- Debug system issues related to memory and CPU usage with practical Linux tools.
- Manage Linux processes using commands like ps, top, htop nice, and kill.
- Utilize essential networking tools such as ping, ifconfig/ip, netstat, and traceroute.
- Configure Linux firewalls using iptables or firewalld to allow or block traffic.
- Monitor and analyze network traffic using tools like tcpdump and iftop.

PW SKILLS

Cron jobs and their importance for task automation.

Cron Jobs

Introduction:

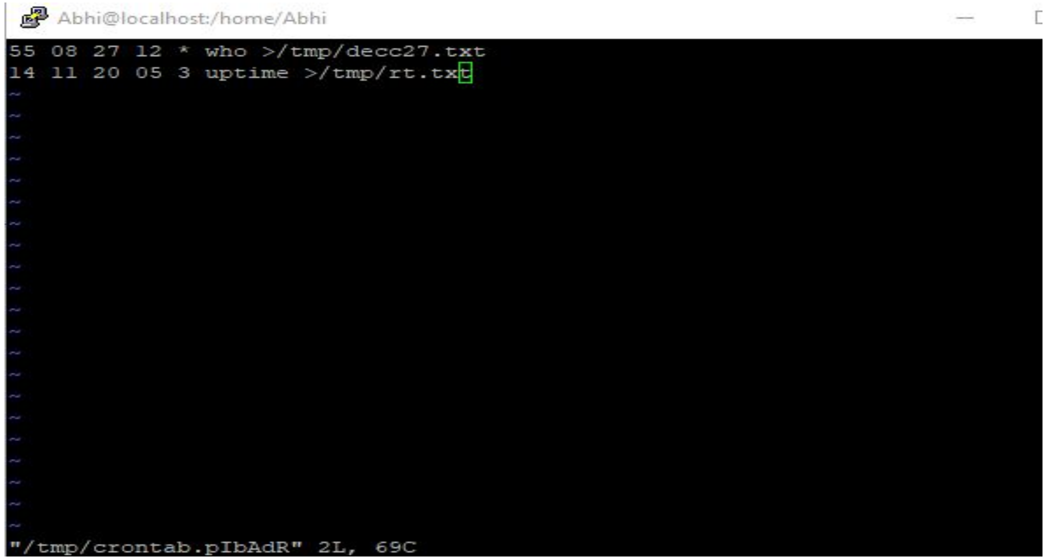
- Cron jobs are powerful tools used in Unix-like operating systems for automating repetitive tasks. Named after the Greek word "Chronos," meaning time, cron jobs allow users to schedule commands or scripts to run at specified intervals without manual intervention.

- This capability is essential for maintaining efficiency and reliability in various computing environments.

Cron Jobs

How It Works:

To edit crontab entries, use ‘crontab –e’

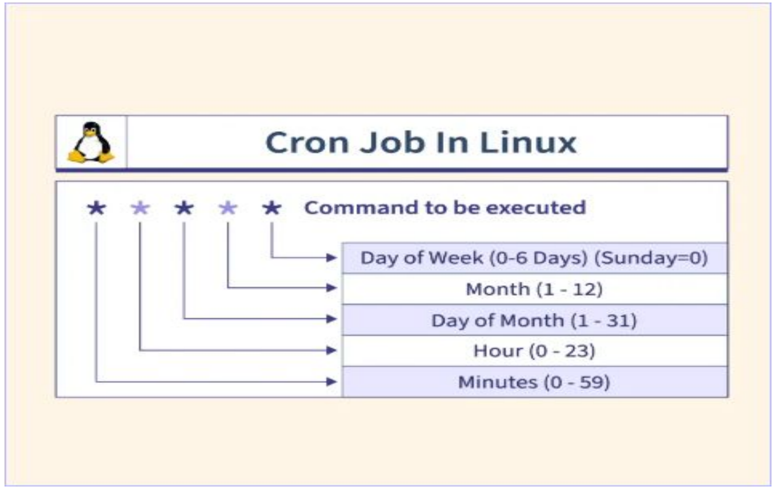


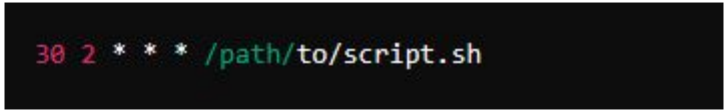
```
55 08 27 12 * who >/tmp/decc27.txt
14 11 20 05 3 uptime >/tmp/rt.txt
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/tmp/crontab.pIbAdR" 2L, 69C
```

Cron Jobs

How It Works:

Example: To run a script every day at 2:30 AM:





```
30 2 * * * /path/to/script.sh
```

Cron Jobs

Importance of Cron Jobs for Task Automation:

Cron jobs play a crucial role in task automation across various domains, particularly in system administration and web development.

Here are some key benefits:

- Task Automation
- Consistency
- Error Reduction
- Resource Management
- Enhanced Productivity

Pop Quiz

Q. Which command is used to schedule jobs in Linux?

A

schedule

B

cron

Pop Quiz

Q. Which command is used to schedule jobs in Linux?

A

schedule

B

cron

PW SKILLS

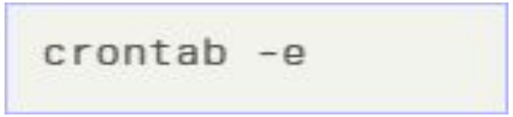
Demonstrating how to schedule tasks using crontab.

Scheduling tasks using crontab

To schedule tasks using ‘crontab’, follow these concise steps:

1. Open the Crontab file:

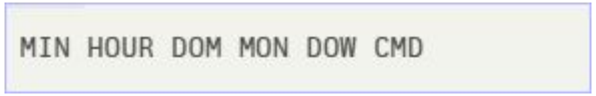
- Use the command to edit your crontab:



```
crontab -e
```

Scheduling tasks using crontab

2. Understand the Syntax:



```
MIN HOUR DOM MON DOW CMD
```

- MIN: Minute (0-59)
- HOUR: Hour (0-23)
- DOM: Day of the Month (1-31)
- MON: Month (1-12)
- DOW: Day of the Week (0-6, where 0 is Sunday)
- CMD: Command to execute

Scheduling tasks using crontab

3. Add Your Cron Job:

- For example, to run a script every day at 2 PM, add:

```
0 14 * * * /path/to/your_script.sh
```

4. Save and Exit:

- After adding your cron job, save the file and exit the editor. The cron job will now be scheduled.

Scheduling tasks using crontab

5. List Current Cron Jobs:

```
crontab -l
```

6. Remove a Cron Job:

- To delete a job, open the crontab with ‘crontab –e’, remove the specific line, and save.

Pop Quiz

Q. Which of the following entries runs a script every day at 3 AM?

A

```
0 3 * * * /path/to/script.sh
```

B

```
3 0 * * * /path/to/script.sh
```

Pop Quiz

Q. Which of the following entries runs a script every day at 3 AM?

A

```
0 3 * * * /path/to/script.sh
```

B

```
3 0 * * * /path/to/script.sh
```

PW SKILLS

Managing background processes using &, jobs, fg, and bg.

Managing Background Processes

To manage background processes in Linux, you can use several commands and techniques. Here's a brief overview:

Running Background Processes:

1. Using '&':

- To run a command in the background, append '&' at the end of the command. For example:

```
long_running_command &
```

Managing Background Processes

Managing Background Processes:

2. Using jobs:

- The ‘jobs’ command lists all background and stopped jobs along with their job IDs. Simply type:

```
jobs
```

Managing Background Processes

Managing Background Processes:

3. Bringing a Job to the Foreground:

- To bring a background job to the foreground, use the ‘fg’ command followed by the job ID or job spec. For example:

```
fg %1
```

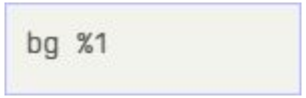
This brings job number 1 to the foreground.

Managing Background Processes

Managing Background Processes:

4. Resuming a Stopped Job in the Background:

- If you have suspended a job (using ‘CTRL+Z’), you can resume it in the background with:



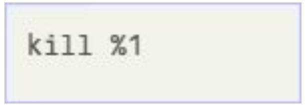
```
bg %1
```

Managing Background Processes

Managing Background Processes:

5. Killing a Background Process:

- To terminate a specific background job, use the ‘kill’ command followed by the job ID:



```
kill %1
```

Pop Quiz

Q. What happens to a background process if you close the terminal?

A

It is suspended.

B

It terminates immediately.

Pop Quiz

Q. What happens to a background process if you close the terminal?

A

It is suspended.

B

It terminates immediately.

Debugging needs for heap memory and CPU issues.

Heap Memory Issues

1. Memory Leaks:

Detection: Use tools like Valgrind to check for memory leaks by analyzing memory allocations and deallocations.

Monitoring: Regularly inspect ‘/proc/<PID>/maps’ to check the memory map of processes, which includes heap regions.

Heap Memory Issues

2. Buffer Overflows:

Detection: Tools like Electric Fence can help catch buffer overruns by placing guard pages around allocated memory.

Prevention: Implementing safe memory allocation practices and using libraries that provide bounds checking can mitigate the risk of buffer overflows.

Heap Memory Issues

3. Heap Corruption:

Detection: Utilize debugging features in the Linux kernel, such as slab debugging, which uses memory poison techniques to detect adjacent writes that corrupt allocated buffers.

Analysis: Analyze core dumps generated during crashes to pinpoint the location and cause of heap corruption.

CPU Issues

1. High CPU Usage:

Monitoring: Use tools like ‘top’ or ‘htop' to monitor processes consuming excessive CPU resources.

Profiling: Profiling tools (e.g., ‘gprof’, ‘perf’) can help identify functions or code paths that are inefficient or consuming too much CPU time.

CPU Issues

2. Thread Management:

Deadlock Detection: Monitor thread states to identify deadlocks where threads are waiting indefinitely for resources held by each other.

Performance Analysis: Use thread profiling tools to understand contention issues and optimize thread usage for better performance.

3. Garbage Collection Tuning:

For languages with automatic memory management, tuning garbage collection settings can significantly impact application performance.

Pop Quiz

Q. Which command can be used to monitor real-time CPU usage of processes in Linux?

A

free

B

top

Pop Quiz

Q. Which command can be used to monitor real-time CPU usage of processes in Linux?

A

free

B

top

PW SKILLS

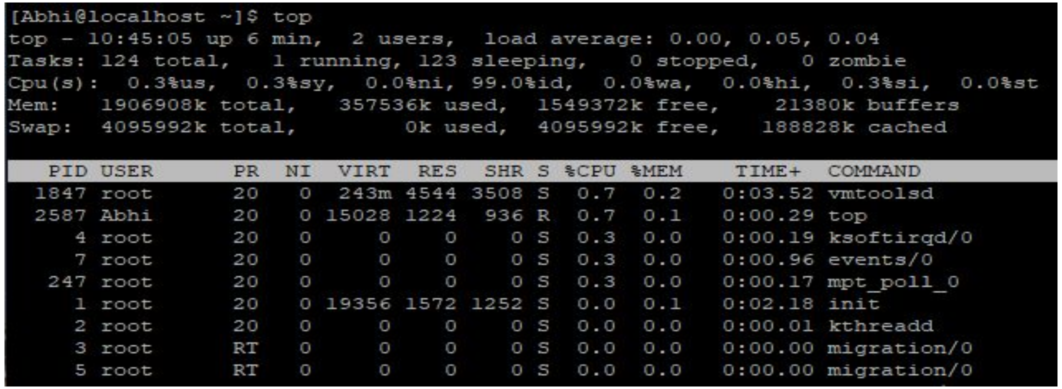
Demonstrating top & htop to identify resource-heavy processes

Top & htop

Using 'top'

- The top command provides a dynamic view of the system's processes, sorted by CPU usage. Here's how to use it:

1. Launch the command: Provides a real-time view of CPU and memory usage. Look for processes with high CPU usage.



```
[Abhi@localhost ~]$ top
top - 10:45:05 up 6 min,  2 users,  load average: 0.00, 0.05, 0.04
Tasks: 124 total,   1 running, 123 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  0.3%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.3%si,  0.0%st
Mem:   1906908k total,  357536k used,  1549372k free,   21380k buffers
Swap:  4095992k total,        0k used,  4095992k free,  188828k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1847 root      20   0  243m 4544 3508 S  0.7  0.2   0:03.52 vmtoolsd
 2587 Abhi      20   0 15028 1224  936 R  0.7  0.1   0:00.29 top
    4 root      20   0     0    0    0 S  0.3  0.0   0:00.19 ksoftirqd/0
    7 root      20   0     0    0    0 S  0.3  0.0   0:00.96 events/0
  247 root      20   0     0    0    0 S  0.3  0.0   0:00.17 mpt_poll_0
    1 root      20   0 19356 1572 1252 S  0.0  0.1   0:02.18 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.01 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
```

Top & htop

Using ‘top’

2. Understanding the Output:

- The output includes columns such as PID (Process ID), USER, %CPU, %MEM, and COMMAND.
- The %CPU column indicates the percentage of CPU usage by each process, helping you identify which processes are consuming the most resources.

3. Interacting with ‘top’:

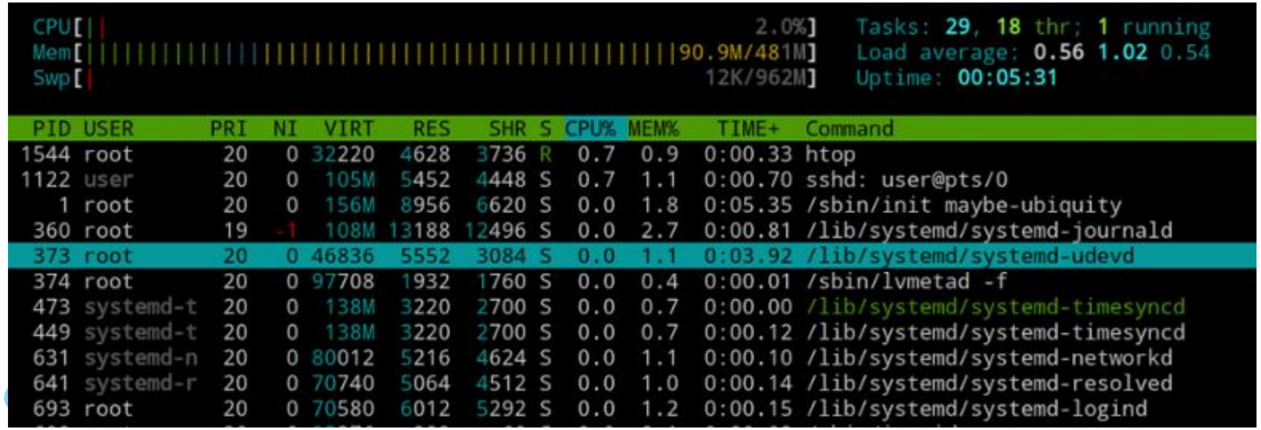
- You can refresh the display by pressing r.
- To kill a process, press k and enter the PID of the process you wish to terminate.

Top & htop

Using ‘htop’

- The htop command is an enhanced version of ‘top’, providing a more user-friendly interface and additional features:

1. Launch the command: This will open an interactive display of processes.



```
CPU[||                                                     2.0%]   Tasks: 29, 18 thr; 1 running
Mem[|||||||||||||||||||||||||||||||||||||||||||||||||90.9M/481M]   Load average: 0.56 1.02 0.54
Swp[|                                                  12K/962M]   Uptime: 00:05:31

 PID USER       PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
1544 root        20   0 32220  4628  3736 R  0.7  0.9  0:00.33 htop
1122 user        20   0  105M  5452  4448 S  0.7  1.1  0:00.70 sshd: user@pts/0
   1 root        20   0  156M  8956  6620 S  0.0  1.8  0:05.35 /sbin/init maybe-ubiquity
 360 root        19  -1  108M 13188 12496 S  0.0  2.7  0:00.81 /lib/systemd/systemd-journald
 373 root        20   0 46836  5552  3084 S  0.0  1.1  0:03.92 /lib/systemd/systemd-udevd
 374 root        20   0 97708  1932  1760 S  0.0  0.4  0:00.01 /sbin/lvmetad -f
 473 systemd-t   20   0  138M  3220  2700 S  0.0  0.7  0:00.00 /lib/systemd/systemd-timesyncd
 449 systemd-t   20   0  138M  3220  2700 S  0.0  0.7  0:00.12 /lib/systemd/systemd-timesyncd
 631 systemd-n   20   0 80012  5216  4624 S  0.0  1.1  0:00.10 /lib/systemd/systemd-networkd
 641 systemd-r   20   0 70740  5064  4512 S  0.0  1.0  0:00.14 /lib/systemd/systemd-resolved
 693 root        20   0 70580  6012  5292 S  0.0  1.2  0:00.15 /lib/systemd/systemd-logind
```

Top & htop

Using ‘htop’

2. Key features:

- Visual Representation
- Sorting Options
- Color-Coded Display

3. Interacting with ‘htop’:

- Use arrow keys to navigate through the list.
- Press ‘F9’ to kill a selected process or adjust its priority (renice).
- You can filter processes using ‘F3’ (search) and ‘F4’ (filter) options.

Top & htop

Identifying Resource-Heavy Processes

To effectively identify resource-heavy processes using either command:

- Look for processes with high values in the ‘%CPU’ and ‘%MEM’ columns.
- In ‘htop’, you can also toggle visibility of threads by pressing ‘H’, allowing you to see which threads within a process are consuming resources.
- For more detailed analysis, consider using additional commands like ‘atop’, which provides historical data on resource usage.

Pop Quiz

Q. What is one advantage of using htop over top?

A

It allows interactive sorting and filtering.

B

It requires less memory.

Pop Quiz

Q. What is one advantage of using htop over top?

A

It allows interactive sorting and filtering.

B

It requires less memory.

PW SKILLS

Linux tools like vmstat and iostat for analyzing performance metrics.

vmstat & iostat

Linux provides various tools for analyzing performance metrics, with vmstat and iostat being two of the most useful for monitoring system performance.

vmstat:

The vmstat command, short for "virtual memory statistics," is used to report information about processes, memory, paging, block I/O, traps, and CPU activity.

vmstat & iostat

Key features:

- Memory usage
- CPU activity
- I/O statistics

Usage: To run vmstat with a 1-second interval for continuous monitoring:

```
vmstat 1
```

vmstat & iostat

iostat:

The ‘iostat’ command is part of the sysstat package and focuses on input/output (I/O) statistics for devices and CPUs. It helps identify bottlenecks in disk I/O performance.

Key features:

- Disk I/O statistics
- CPU utilization

Usage: To run iostat, simply enter:

```
iostat
```

Pop Quiz

Q. Which command must be installed to use both vmstat and iostat on a Linux system?

A

procps

B

sysstat

Pop Quiz

Q. Which command must be installed to use both vmstat and iostat on a Linux system?

A

procps

B

sysstat

PW SKILLS

Benefits of Linux debugging for DevOps, including visibility and control.

Benefits of Linux debugging

Linux debugging plays a crucial role in DevOps, providing significant benefits that enhance both visibility and control over system performance and application behavior. Here are the key advantages:

Benefits of Linux Debugging for DevOps

1. Enhanced Visibility:

- Real-Time monitoring
- Comprehensive Logging

Benefits of Linux debugging

2. Improved Control:

- Error Detection and Resolution
- Automated Testing and Feedback Loops

3. Increased Efficiency:

- Streamlined Processes
- Faster Recovery from Failures

Benefits of Linux debugging

4. Better Collaboration:

- Cross-Functional Insights

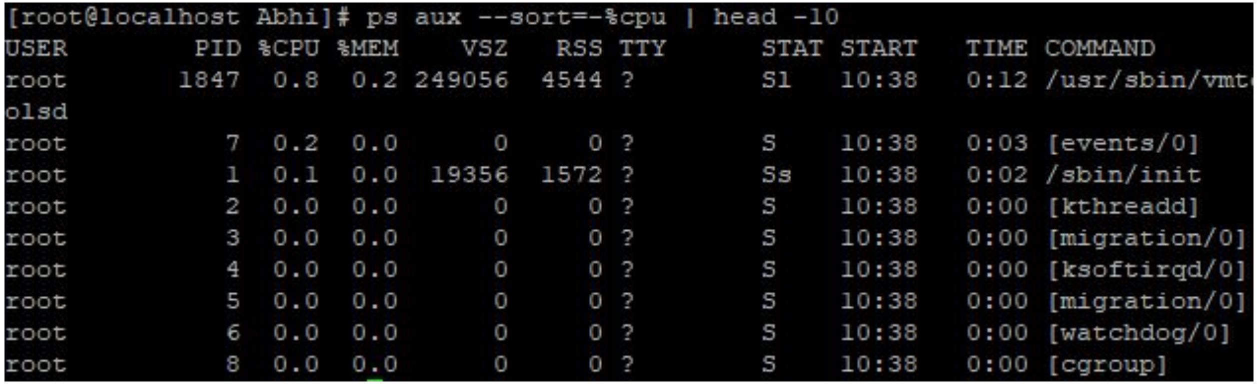
5. Risk Mitigation:

- Proactive Issue Management

Introducing process management commands

ps to list processes and explain key columns

The ps command in Linux is a powerful utility used to display information about currently running processes.



```
[root@localhost Abhi]# ps aux --sort=-%cpu | head -10
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      1847  0.8  0.2 249056  4544 ?        Sl   10:38   0:12 /usr/sbin/vmt
olsd
root         7  0.2  0.0      0     0 ?        S    10:38   0:03 [events/0]
root         1  0.1  0.0  19356  1572 ?        Ss   10:38   0:02 /sbin/init
root         2  0.0  0.0      0     0 ?        S    10:38   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    10:38   0:00 [migration/0]
root         4  0.0  0.0      0     0 ?        S    10:38   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S    10:38   0:00 [migration/0]
root         6  0.0  0.0      0     0 ?        S    10:38   0:00 [watchdog/0]
root         8  0.0  0.0      0     0 ?        S    10:38   0:00 [cgroup]
```

ps to list processes and explain key columns

Key columns in 'ps' Output:

The ps command in Linux is a powerful utility used to display information about currently running processes.

1. PID (process ID): It is essential for managing processes, such as terminating or modifying their priority.

2. TTY (Terminal): Indicates the terminal associated with the process.

3. TIME: Displays the total CPU time consumed by the process.

ps to list processes and explain key columns

4. CMD (Command): Shows the command that was used to start the process.

5. USER: The username of the account that owns the process.

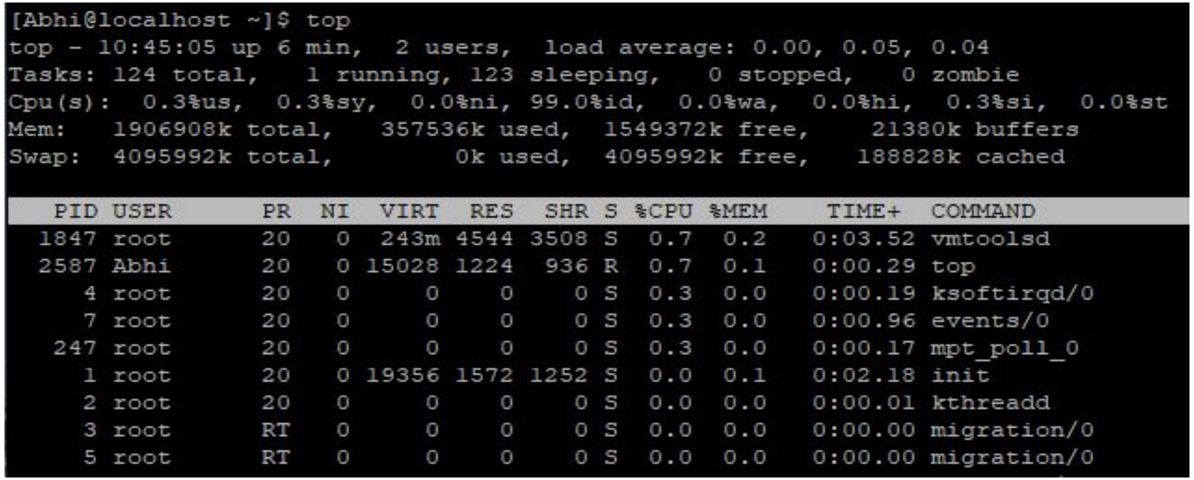
6. %CPU: Represents the percentage of CPU usage by the process.

7. %MEM: Indicates the percentage of physical memory used by the process.

top/htop for real-time process monitoring

Using 'top'

Provides a real-time view of CPU and memory usage. Look for processes with high CPU usage.



```
[Abhi@localhost ~]$ top
top - 10:45:05 up 6 min,  2 users,  load average: 0.00, 0.05, 0.04
Tasks: 124 total,   1 running, 123 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  0.3%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.3%si,  0.0%st
Mem:   1906908k total,   357536k used,  1549372k free,    21380k buffers
Swap:  4095992k total,        0k used,  4095992k free,   188828k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1847 root      20   0  243m 4544 3508 S  0.7  0.2   0:03.52 vmtoolsd
 2587 Abhi      20   0 15028 1224  936 R  0.7  0.1   0:00.29 top
    4 root      20   0     0    0    0 S  0.3  0.0   0:00.19 ksoftirqd/0
    7 root      20   0     0    0    0 S  0.3  0.0   0:00.96 events/0
  247 root      20   0     0    0    0 S  0.3  0.0   0:00.17 mpt_poll_0
    1 root      20   0 19356 1572 1252 S  0.0  0.1   0:02.18 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.01 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
```

top/htop for real-time process monitoring

Using ‘top’

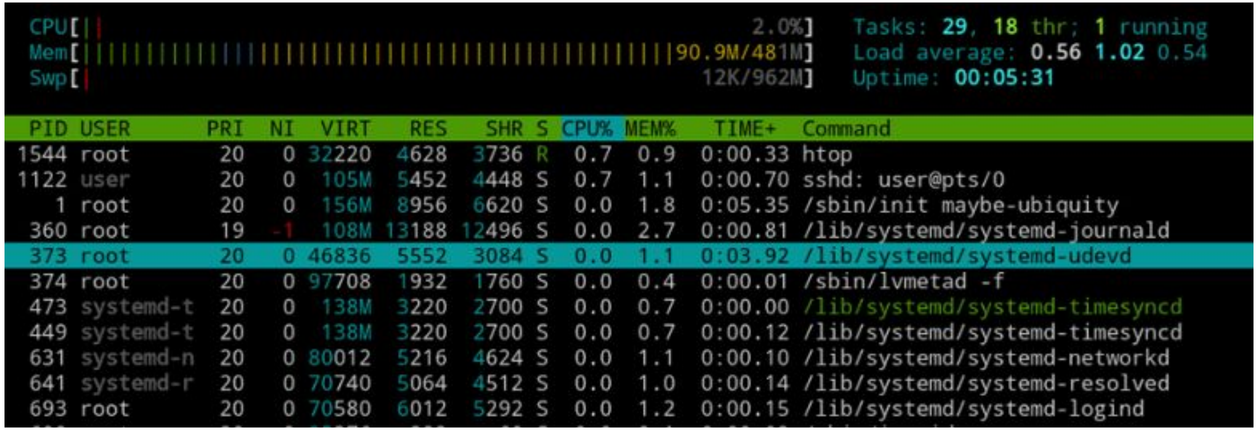
Key columns in ‘top’ output:

- **PID:** Process ID, a unique identifier for each process.
- **USER:** The username of the account that owns the process.
- **%CPU:** Percentage of CPU usage by the process.
- **%MEM:** Percentage of physical memory used by the process.
- **TIME+:** Total CPU time consumed by the process.
- **COMMAND:** The command that started the process.

top/htop for real-time process monitoring

Using ‘htop’

An enhanced version of top with a more user-friendly interface.



```
  CPU[||                                                       2.0%]   Tasks: 29, 18 thr; 1 running
  Mem[|||||||||||||||||||||||||||||||||||||||||||||||||||||90.9M/481M]   Load average: 0.56 1.02 0.54
  Swp[|                                                     12K/962M]   Uptime: 00:05:31

  PID USER       PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 1544 root        20   0 32220  4628  3736 R  0.7  0.9  0:00.33 htop
 1122 user        20   0  105M  5452  4448 S  0.7  1.1  0:00.70 sshd: user@pts/0
    1 root        20   0  156M  8956  6620 S  0.0  1.8  0:05.35 /sbin/init maybe-ubiquity
  360 root        19  -1  108M 13188 12496 S  0.0  2.7  0:00.81 /lib/systemd/systemd-journald
  373 root        20   0 46836  5552  3084 S  0.0  1.1  0:03.92 /lib/systemd/systemd-udevd
  374 root        20   0 97708  1932  1760 S  0.0  0.4  0:00.01 /sbin/lvmetad -f
  473 systemd-t   20   0  138M  3220  2700 S  0.0  0.7  0:00.00 /lib/systemd/systemd-timesyncd
  449 systemd-t   20   0  138M  3220  2700 S  0.0  0.7  0:00.12 /lib/systemd/systemd-timesyncd
  631 systemd-n   20   0 80012  5216  4624 S  0.0  1.1  0:00.10 /lib/systemd/systemd-networkd
  641 systemd-r   20   0 70740  5064  4512 S  0.0  1.0  0:00.14 /lib/systemd/systemd-resolved
  693 root        20   0 70580  6012  5292 S  0.0  1.2  0:00.15 /lib/systemd/systemd-logind
```

top/htop for real-time process monitoring

Using ‘htop’

Key Columns in ‘htop’ Output:

Similar to ‘top’, but with additional visual aids, including:

- PID, USER, %CPU, and %MEM, along with a visual representation of CPU and memory usage.

Adjust process priorities with nice and renice.

Nice Command:

- **Purpose:** The ‘nice’ command is used to start a new process with a specified priority, known as the "nice value.“

- **Nice Value:** The nice value ranges from -20 (highest priority) to 19 (lowest priority). A lower nice value increases the process's priority, allowing it to receive more CPU time.

Adjust process priorities with nice and renice.

Nice Command:

Usage:

- To start a process with a specific nice value:

```
nice -n <nice_value> <command>
```

- For example, to start a process with a higher priority:

```
nice -n -5 my_process
```

Adjust process priorities with nice and renice.

renice Command:

- **Purpose:** The ‘renice’ command modifies the priority of an already running process.

Usage:

- To change the priority of a running process by its PID:

```
renice <new_nice_value> -p <PID>
```

Adjust process priorities with nice and renice.

renice Command:

Usage:

- For example, to change the priority of a process with PID 1234 to a lower priority (higher nice value):

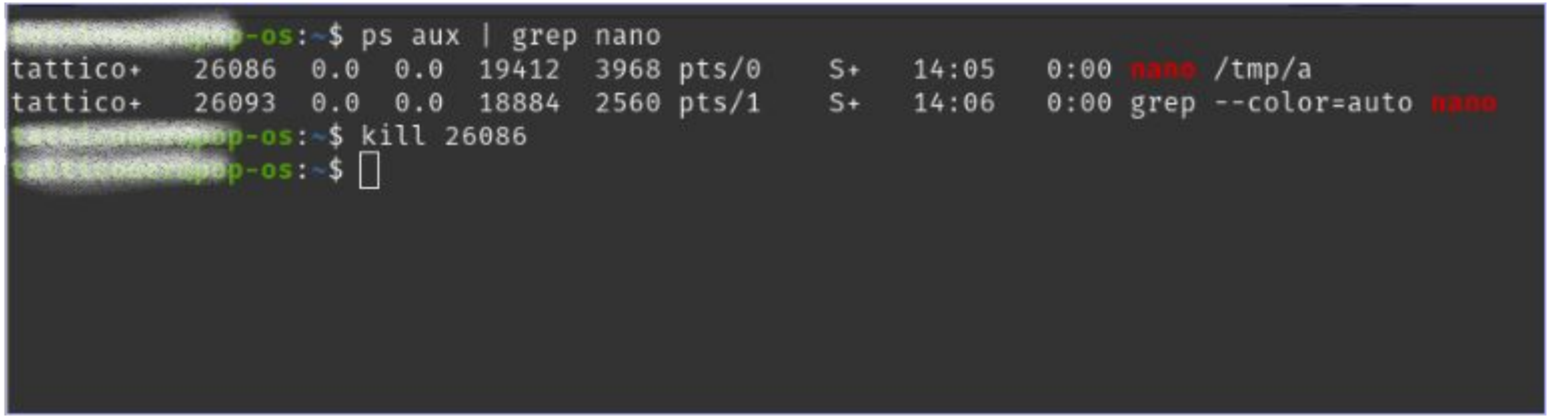
```
sudo renice 10 -p 1234
```

- You can also change the priority for all processes owned by a specific user or group:

```
renice <new_nice_value> -u <user_id>
renice <new_nice_value> -g <group_id>
```

Terminate processes using kill

Using the 'kill' Command:



```
[redacted]p-os:~$ ps aux | grep nano
tattico+   26086  0.0  0.0  19412  3968 pts/0    S+   14:05   0:00 nano /tmp/a
tattico+   26093  0.0  0.0  18884  2560 pts/1    S+   14:06   0:00 grep --color=auto nano
[redacted]p-os:~$ kill 26086
[redacted]p-os:~$ 
```

Terminate processes using kill

Using the 'kill' Command:

1. Basic Syntax:

```
kill [signal] PID
```

2. Common signals:

SIGTERM (15): Requests graceful termination, allowing the process to clean up.

```
kill 1234  # Sends SIGTERM to process with PID 1234
```

SIGKILL (9): Forces immediate termination, which does not allow the process to clean up.

```
kill -9 1234  # Forcefully kills the process
```

Terminate processes using kill

Using the ‘kill’ Command:

SIGINT (2): Interrupts the process, similar to pressing ‘Ctrl+C’.

```
kill -2 1234  # Sends SIGINT to the process
```

3. Killing Multiple Processes:

```
kill -15 1234 5678 91011  # Sends SIGTERM to multiple PIDs
```

Terminate processes using kill

Using the 'kill' Command:

4. Using pkill and killall:

- **pkill:** Kills processes by name without needing the PID.

```
pkill nginx   # Kills all instances of nginx
```

- **killall:** Terminates all instances of a process by name.

```
killall nginx  # Kills all nginx processes
```

Pop Quiz

Q. In the output of the ps command, what does the PID column represent?

A

Process Information

B

Process Identifier

Pop Quiz

Q. In the output of the ps command, what does the PID column represent?

A

Process Information

B

Process Identifier

Pop Quiz

Q. How can you adjust the priority of an already running process?

A

Using the renice command

B

Using the nice command

Pop Quiz

Q. How can you adjust the priority of an already running process?

A

Using the renice command

B

Using the nice command

PW SKILLS

Take A 5-Minute **Break!**

- **Stretch and relax**
- **Hydrate**
- **Clear your mind**
- **Be back in 5 minutes**

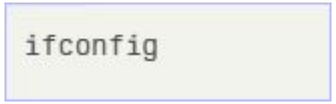
PW SKILLS

Introducing key networking tools

ifconfig/ip for interface management.

ifconfig command

- The ‘ifconfig’ command is used to view and configure network interfaces. To display all active network interfaces, simply type:



```
ifconfig
```

- This command shows details such as the interface name, IP address, netmask, broadcast address, and MAC address.

ifconfig/ip for interface management.

Common Operations:

1. Assigning an IP Address:

```
ifconfig [interface_name] [IP_address]
```

Example:

```
ifconfig eth0 192.168.1.100
```

ifconfig/ip for interface management.

2. Enabling/Disabling Interfaces:

- To enable an interface:

```
ifconfig eth0 up
```

- To disable an interface:

```
ifconfig eth0 down
```

ifconfig/ip for interface management.

3. Setting a Broadcast Address:

```
ifconfig eth0 broadcast 192.168.1.255
```

4. Viewing Specific Interface Information:

```
ifconfig eth0
```

Advanced Operations: You can also change the Maximum Transmission Unit (MTU) size:

```
ifconfig eth0 mtu 1500
```

ifconfig/ip for interface management.

ip Command

- The ‘ip’ command is part of the ‘iproute2’ package and is considered more modern and versatile than ‘ifconfig’.

```
ip addr show
```

- This command provides detailed information about each interface, including its state (up or down), IP addresses, and more.

ifconfig/ip for interface management.

Common Operations:

1. **Assigning an IP Address:**

```
ip addr add [IP_address] dev [interface_name]
```

Example:

```
ip addr add 192.168.1.100/24 dev eth0
```

ifconfig/ip for interface management.

2. Enabling/Disabling Interfaces:

- To enable an interface:

```
ip link set dev eth0 up
```

- To disable an interface:

```
ip link set dev eth0 down
```

ifconfig/ip for interface management.

3. Deleting an IP Address:

```
ip addr del [IP_address] dev [interface_name]
```

4. Viewing Specific Interface Information:

```
ip addr show dev eth0
```

Pop Quiz

Q. To assign an IP address of 192.168.1.10 to the interface eth0, which command is correct?

A

```
ifconfig 192.168.1.10 eth0
```

B

```
ifconfig eth0 192.168.1.10
```

Pop Quiz

Q. To assign an IP address of 192.168.1.10 to the interface eth0, which command is correct?

A

```
ifconfig 192.168.1.10 eth0
```

B

```
ifconfig eth0 192.168.1.10
```

Using ping to test connectivity.

The ping command is a fundamental tool used to test network connectivity between devices. It operates by sending Internet Control Message Protocol (ICMP) echo request packets to a specified IP address or hostname and waiting for a response.

Using the Ping Command

1. **Basic Syntax:**

```
ping [host_or_IP_address]
```

Using ping to test connectivity.

2. Executing the Command:

- Open your terminal (Linux or macOS) or Command Prompt (Windows).
- Type the command followed by the target address:

```
ping www.google.com
```

3. Interpreting Results:

If the connection is successful, you will see replies from the target with details such as:

- Round-trip time (latency)
- Packet loss percentage
- Example output might look like this:

```
Reply from 172.217.14.206: bytes=32 time=14ms TTL=56
```

Using ping to test connectivity.

4. Stopping the Command: On Linux and macOS, use ‘Ctrl + C’ to stop the continuous pinging, which sends packets until interrupted.

Common Options:

- **Specify Number of Pings:**

To limit the number of pings sent, use the -c option in Linux:

```
ping -c 4 www.google.com
```

- **Change Packet Size:**

You can specify the size of packets sent using the -s option:

```
ping -s 64 www.google.com
```

Pop Quiz

Q. What does a successful ping response indicate?

A

The device is reachable and responding

B

The network is congested

Pop Quiz

Q. What does a successful ping response indicate?

A

The device is reachable and responding

B

The network is congested

netstat and ss for socket statistics

The netstat and ss commands are essential tools in Linux for monitoring network connections and socket statistics.

netstat Command

The netstat (network statistics) command is a traditional tool used to display network connections, routing tables, interface statistics, and more.

Common Uses:

- Display All Connections:

```
netstat -a
```

netstat and ss for socket statistics

- **Show Listening Ports:**

```
netstat -l
```

- **List TCP Connections:**

```
netstat -at
```

- **List UDP Connections:**

```
netstat -au
```

- **Display Statistics by Protocol:**

```
netstat -s
```

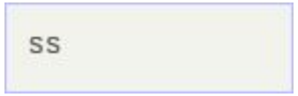
netstat and ss for socket statistics

ss command

The ss (socket statistics) command is a modern replacement for ‘netstat’, providing detailed information about socket connections and network statistics with improved speed and efficiency.

Common Uses:

- Display All Socket Connections:



```
ss
```

netstat and ss for socket statistics

- Show Listening Sockets Only:

```
ss -l
```

- Display Summary of Socket Statistics:

```
ss -s
```

- List TCP Connections:

```
ss -t
```

- List UDP Connections:

```
ss -u
```

Pop Quiz

Q. What is a key advantage of using the ss command over netstat?

A

It provides more detailed output.

B

It is slower but more accurate.

Pop Quiz

Q. What is a key advantage of using the ss command over netstat?

A

It provides more detailed output.

B

It is slower but more accurate.

Demonstrating traceroute to analyze network paths.

The traceroute command is a network diagnostic tool used to trace the route that packets take from a source to a destination over an IP network. It helps identify the path and measure the transit delays of packets across the network.

Basic Usage

- To perform a basic traceroute operation, you can use the following command:

```
traceroute [options] <hostname or IP>
```

Demonstrating traceroute to analyze network paths.

For example, to trace the route to Google, you would type:

```
traceroute www.google.com
```

Understanding Traceroute Output:

When executed, traceroute provides a detailed output showing each hop along the route. Each line typically includes:

- The hop number
- The IP address of the router or device
- Round-trip time (RTT) for each probe sent to that hop

Demonstrating traceroute to analyze network paths.

Key Options:

- -m max-ttl: Set the maximum number of hops (TTL).
- -n: Do not resolve IP addresses to domain names, which speeds up the process.
- -p port: Specify the destination port to use during the traceroute.
- -q nqueries: Set the number of probes sent to each hop.
- -4 / -6: Force the use of IPv4 or IPv6 respectively.

Demonstrating traceroute to analyze network paths.

Example Commands:

1. **Basic Traceroute:**

```
traceroute www.example.com
```

1. **Using IPv4 Only:**

```
traceroute -4 www.example.com
```

1. **Setting Maximum Hops:**

```
traceroute -m 10 www.example.com
```

1. **Disabling DNS Resolution:**

```
traceroute -n www.example.com
```

Pop Quiz

Q. Which of the following commands would display a traceroute to example.com without resolving domain names?

A

```
traceroute example.com -n
```

B

```
traceroute example.com --n
```

Pop Quiz

Q. Which of the following commands would display a traceroute to example.com without resolving domain names?

A

```
traceroute example.com -n
```

B

```
traceroute example.com --n
```

Show nslookup for DNS queries.

The nslookup command is a powerful tool used for querying Domain Name System (DNS) servers to obtain domain name or IP address mapping information. It can operate in two modes: interactive and non-interactive.

1. **Interactive Mode:**

- **To enter interactive mode, simply type:**

```
nslookup
```

- **You can then enter multiple queries, such as:**

```
www.example.com
```

Show nslookup for DNS queries.

2. Non-Interactive Mode:

- **For a single query, use:**

```
nslookup [hostname]
```

Example:

```
nslookup www.example.com
```

Show nslookup for DNS queries.

Common Commands and Options:

- **Query A Record:**

```
nslookup example.com
```

- **Check NS Records**

```
nslookup -type=ns example.com
```

Show nslookup for DNS queries.

- **Get SOA Record:**

```
nslookup -type=soa example.com
```

- **Enable Debug Mode:**

```
nslookup -debug example.com
```

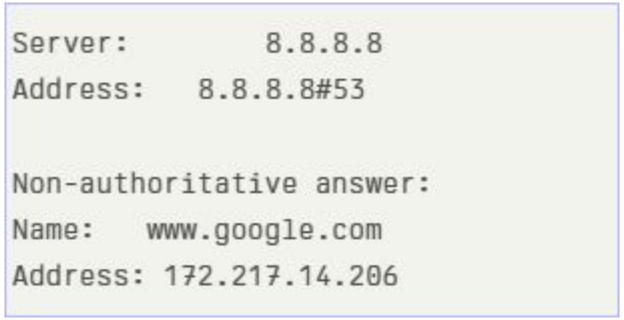
- **Specify a Different DNS Server:**

```
nslookup example.com [DNS_server]
```

Show nslookup for DNS queries.

Example Outputs:

When you run a command like nslookup www.google.com, you might see output similar to this:



```
Server:         8.8.8.8
Address:    8.8.8.8#53

Non-authoritative answer:
Name:    www.google.com
Address: 172.217.14.206
```

Pop Quiz

Q. Which mode does nslookup operate in for querying multiple hosts?

A

Only non-interactive mode

B

Both interactive and non-interactive modes

Pop Quiz

Q. Which mode does nslookup operate in for querying multiple hosts?

A

Only non-interactive mode

B

Both interactive and non-interactive modes

SKILLS

Introduction to iptables and firewalld.

iptables

What is iptables?
Iptables is a command-line utility for configuring the built-in Linux kernel firewall. It allows system administrators to define rules that control incoming and outgoing network traffic.

Providing a robust mechanism for securing Linux systems against unauthorized access and various network-based attacks.

How does it work?
Iptables operates using a set of rules organized into chains, which are evaluated for each packet that traverses the network.

firewalld

What is firewalld?
Firewalld is a dynamic firewall management tool available on Linux systems, designed to simplify the management of firewall rules. It provides an easier interface compared to iptables and supports zones, which allow administrators to define different levels of trust for network connections.

Key Features:

- Zones
- Dynamic Management
- Rich Interface

Basic concepts of allowing/blocking traffic using rules.

Allowing & Blocking Traffic

1. Understanding Firewall Rules : Firewalls use rules to control the flow of network traffic based on predefined criteria. These rules can either allow or block packets based on attributes such as source/destination IP addresses, port numbers, and protocols.

2. Rule Structure:

Firewall rules typically consist of:

- Action
- Criteria
- Chain

Allowing & Blocking Traffic

3. Default Policies: Firewalls can be configured with default policies that dictate the behavior for traffic not explicitly defined by rules:

- Accept by Default:
- Drop by Default:

4. Rule Evaluation Order: Rules are processed in a sequential manner from top to bottom.

- Place specific allow rules before broader block rules.
- Use a default rule at the end to handle unspecified traffic.

Allowing & Blocking Traffic

Common Commands:

- Allow Incoming Traffic from a Specific IP:

```
iptables -A INPUT -s 192.168.1.10 -j ACCEPT
```

- Block Incoming Traffic from a Specific IP:

```
iptables -A INPUT -s 203.0.113.51 -j DROP
```

- Allow Traffic on a Specific Port (e.g., SSH):

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

- Reject Connections with a Response:

```
iptables -A INPUT -s 203.0.113.51 -j REJECT
```

- Logging and Monitoring:

```
iptables -A INPUT -j LOG --log-prefix "Dropped Packet: "
```

How to add and remove rules using iptables.

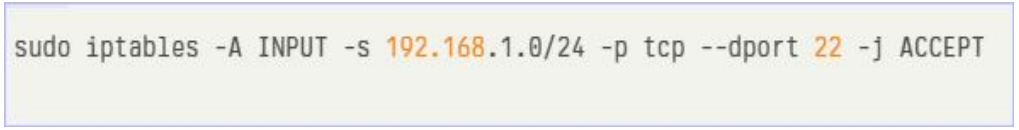
Adding & Removing rules

The iptables command is used to configure the Linux kernel's packet filtering rules. Here’s a concise guide on how to add and remove rules.

Adding Rules:

1. Append a Rule:

- To add a rule at the end of a specific chain (e.g., INPUT), use the -A option:



```
sudo iptables -A INPUT -s 192.168.1.0/24 -p tcp --dport 22 -j ACCEPT
```

- This command allows incoming SSH traffic from the specified subnet.

Adding & Removing rules

2. Insert a Rule:

- To add a rule at a specific position in the chain, use the -I option followed by the index number:

```
sudo iptables -I INPUT 1 -p tcp --dport 80 -j ACCEPT
```

- This command inserts a rule allowing HTTP traffic at the top of the INPUT chain.

3. Allow Established Connections:

```
 sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j
ACCEPT
```

Adding & Removing rules

Removing Rules:

1. Delete a Rule by Specification:
To delete a specific rule, you can use the -D option along with the same parameters used to create it:

```
sudo iptables -D INPUT -s 192.168.1.0/24 -p tcp --dport 22 -j ACCEPT
```

2. Delete a Rule by Index Number:
First, list your rules with line numbers:

```
sudo iptables -L --line-numbers
```

Adding & Removing rules

Then, delete a rule by its index number:

```
sudo iptables -D INPUT 2
```

3. Flush All Rules:

```
sudo iptables -F  # Flush all rules in all chains
```

Configuration of firewalld zones for specific traffic control

Configuration of firewalld zones

Firewalld uses zones to manage network traffic based on predefined trust levels, allowing administrators to control incoming and outgoing traffic effectively.

Key Concepts of Firewalld Zones:

1. **Predefined Zones: Public, Home, Internal, Drop, Block.**

1. **Assigning Zones to Interfaces:**

```
sudo firewall-cmd --zone=public --change-interface=eth0
```

Managing Services and Ports:

```
sudo firewall-cmd --zone=public --add-service=http --permanent
```

- To allow specific services in a zone, use:

Configuration of firewalld zones

- To open specific ports:

```
sudo firewall-cmd --zone=home --add-port=8080/tcp --permanent
```

4. Setting Default Behavior:

```
sudo firewall-cmd --zone=public --set-target=DROP
```

5. Reloading Firewalld:

```
sudo firewall-cmd --reload
```

Configuration of firewalld zones

Example Configuration Steps:

1. Assign Zone to Interface:

```
sudo firewall-cmd --zone=public --change-interface=eth0
```

1. Allow HTTP Service in Public Zone:

```
sudo firewall-cmd --zone=public --add-service=http --permanen
```

1. Open Custom Port in Home Zone:

```
sudo firewall-cmd --zone=home --add-port=8080/tcp --permanen
```

1. Set Default Target for Public Zone:

```
sudo firewall-cmd --zone=public --set-target=REJECT
```

1. Reload Firewalld:

```
sudo firewall-cmd --reload
```

tcpdump for capturing and analyzing network packets.

tcpdump

What is tcpdump?

Tcpdump is a powerful command-line packet analyzer used to capture and analyze network traffic on a system.

Key features:

- Packet capture
- Filtering capabilities
- Output options

tcpdump

Basic Usage:

1. Listing Available Interfaces:

```
sudo tcpdump -D
```

2. Capturing Packets:

```
sudo tcpdump -i eth0
```

3. Applying Filters:

To capture only HTTP traffic (TCP port 80):

```
sudo tcpdump -i eth0 tcp port 80
```

tcpdump

To capture packets from a specific IP address:

```
sudo tcpdump -i eth0 host 192.168.1.5
```

4. Saving Output to a File:

```
sudo tcpdump -i eth0 -w captured_packets.pcap
```

5. Reading Saved Capture Files:

```
tcpdump -r captured_packets.pcap
```

Pop Quiz

Q. How can you capture only HTTP traffic using tcpdump?

A

tcpdump http

B

tcpdump port 80

Pop Quiz

Q. How can you capture only HTTP traffic using tcpdump?

A

tcpdump http

B

tcpdump port 80

PW SKILLS

iftop and nload for real-time bandwidth monitoring

iftop and nload

1. Iftop

Iftop is a command-line tool that provides real-time monitoring of network bandwidth usage.

Key features:

- Real-Time Display
- Connection-Based Monitoring
- Filtering Options

iftop and nload

Installation

Debian/Ubuntu:

```
sudo apt-get install iftop
```

CentOS/RHEL:

```
sudo yum install epel-release
sudo yum install iftop
```

Usage Example: To start monitoring traffic on a specific interface (e.g., eth0):

```
sudo iftop -i eth0
```

iftop and nload

2. nload

Nload is a simpler command-line tool that visualizes incoming and outgoing traffic separately. It provides a graphical representation of bandwidth usage over time.

Key features:

- Separate Monitoring
- Total Data Transfer Stats
- User-Friendly Interface

iftop and nload

Installation

Debian/Ubuntu:

```
sudo apt-get install nload
```

CentOS/RHEL:

```
sudo yum install nload
```

Usage Example: To monitor traffic on the default interface:

```
nload
```

Pop Quiz

Q. What type of traffic does the nload command monitor?

A

Both incoming and outgoing traffic

B

Only incoming traffic

Pop Quiz

Q. What type of traffic does the nload command monitor?

A

Both incoming and outgoing traffic

B

Only incoming traffic

Practical examples to analyze network traffic and bandwidth usage.

Practical Examples

1. Wireshark:

Description: A powerful open-source packet analyzer that captures and displays network packets in real-time.

Practical Example: To capture HTTP traffic:

- Open Wireshark and select the network interface.
- Set a capture filter (e.g., tcp port 80).
- Start the capture and analyze the packets to identify slow responses or errors.

Practical Examples

2. Tcpdump

Description: A command-line packet capture tool that allows users to capture and analyze network packets.

Practical Example: To capture all traffic on a specific interface and save it to a file:

```
sudo tcpdump -i eth0 -w capture.pcap
```

Later, analyze the captured file with Wireshark or tcpdump itself:

```
tcpdump -r capture.pcap
```

Practical Examples

3. Iftop

Description: A real-time bandwidth monitoring tool that displays bandwidth usage by individual connections.

Practical Example: To monitor bandwidth usage on eth0:

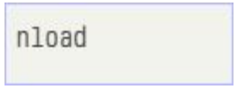
```
sudo iftop -i eth0
```

This command shows which hosts are consuming the most bandwidth.

Practical Examples

4. Nload

Description: A command-line tool that visualizes incoming and outgoing traffic separately.

Practical Example: Start monitoring network traffic: 

```
nload
```

This provides a graphical representation of total incoming and outgoing traffic.

Practical Examples

5. SolarWinds NetFlow Traffic Analyzer:

- **Description:** A comprehensive tool for monitoring and analyzing network traffic using flow data.
- **Practical Example:** Use it to set alerts for unusual traffic patterns or generate reports on bandwidth usage trends.

Practical Examples

6. PRTG Network Monitor:

- **Description:** A powerful tool that monitors network devices, bandwidth, and applications in real-time.

- **Practical Example:** Set up sensors to monitor specific interfaces or applications, allowing for detailed analysis of bandwidth usage over time.

Time for case study!

Important

- Complete the post-class assessment
- Complete assignments (if any)
- Practice the concepts and techniques taught in this session
- Review your lecture notes
- Note down questions and queries regarding this session and consult the teaching assistants

Thanks!