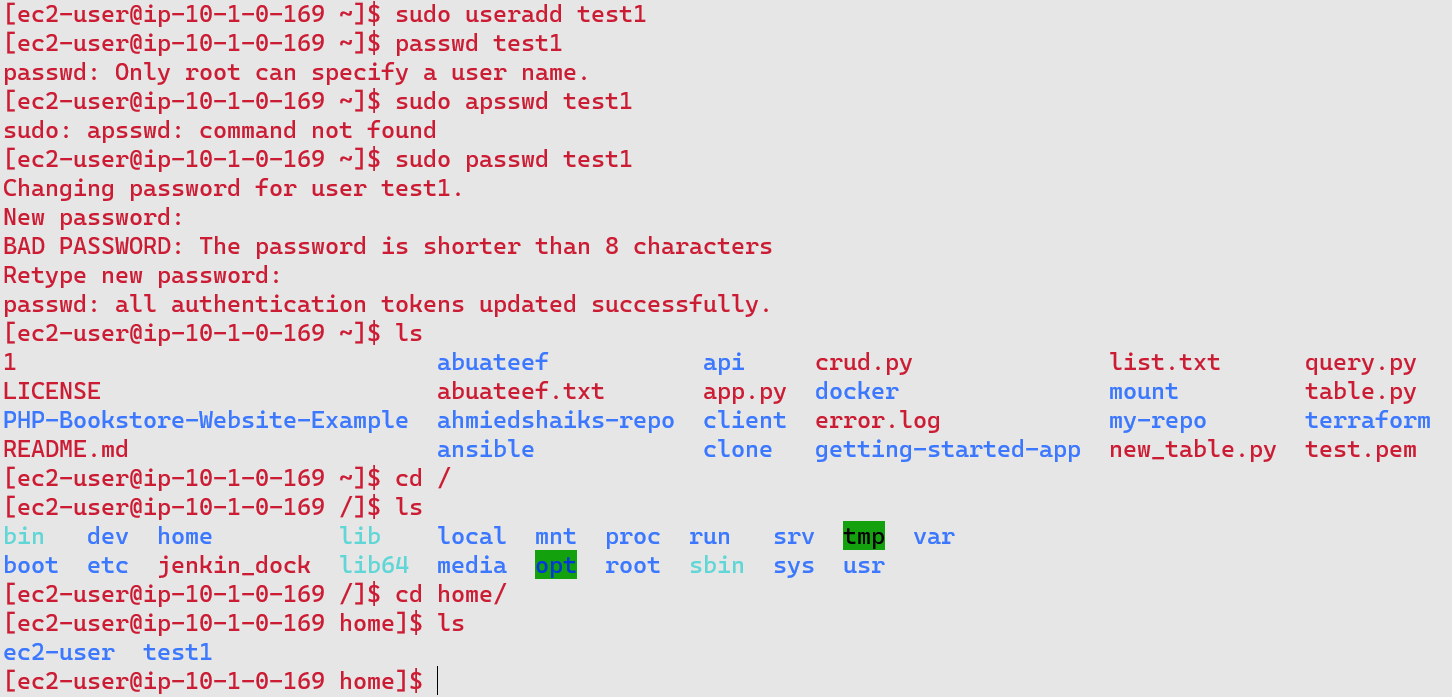
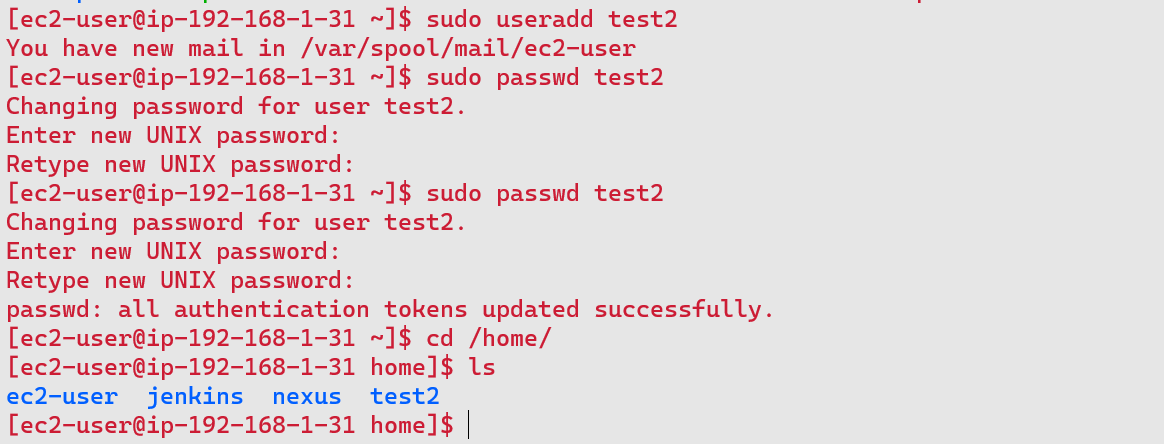
Task 1

1. Go through the Linux commands from basic to advanced level
2. provision 2 Linux instances
3. create users in both the instances i.e., user1 in one Linux and user2 in another Linux



User 1 as test1 in one machine



User 2 in as test2 in second machine

* switch to user1 and create a directory and add some files in one Linux and copy that directory to user2 in another Linux instance

Two instances with two user and copy a file and a directoty

Scp – I pem /home/ec2-user/filename ec2-user@IP-address:/path/to/destination

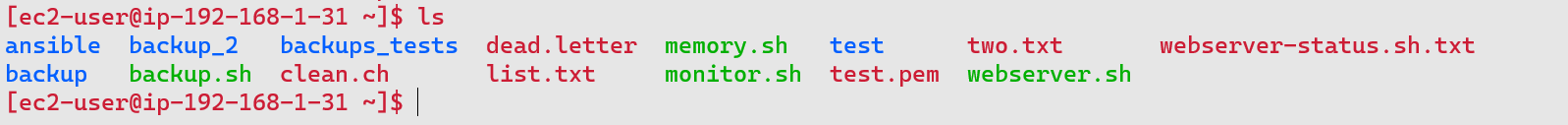
For secure copying a file

To copy a directory

From machine one to machine 2

Scp -r /path/to/directory ec2-user@ IP-address:/path/to/directory

Output.



Task 2:

Print information when root user login

Print below message to the root user when logged in.

How many users logged in currently

Current uptime

Free RAM utilization

Free root partition size

#!/bin/bash

# Count the number of users currently logged in

num\_users=$(who | wc -l)

# Get the uptime

uptime=$(uptime)

# Get the free RAM utilization

free\_ram=$(free -h | awk 'NR==2{print $4}')

# Get the free root partition size

free\_root\_partition=$(df -h / | awk 'NR==2{print $4}')

# Print the information

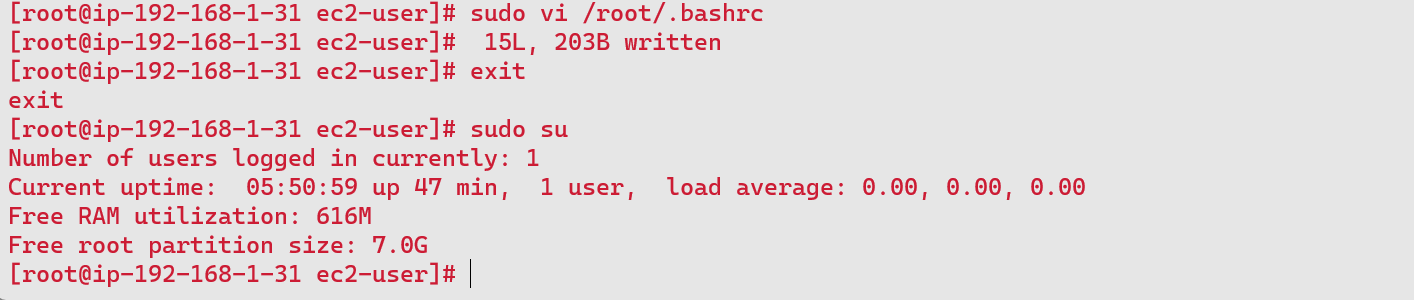
echo "Number of users logged in currently: $num\_users"

echo "Current uptime: $uptime"

echo "Free RAM utilization: $free\_ram"

echo "Free root partition size: $free\_root\_partition"

run command ./monitor.sh



Task 3: Shell script to monitor CPU usage of the server and send CPU usage notification to emails

#!/bin/bash

# Set threshold for CPU usage (in percentage)

THRESHOLD=60

LOCALUSER="ec2-user"

# Get CPU usage percentage

CPU\_USAGE=$(top -bn1 | grep "Cpu(s)" | sed "s/.\*, \*\([0-9.]\*\)%\* id.\*/\1/" | awk '{print 100 - $1}')

# Check if CPU usage exceeds threshold

if (( $(echo "$CPU\_USAGE > $THRESHOLD" | bc -l) )); then

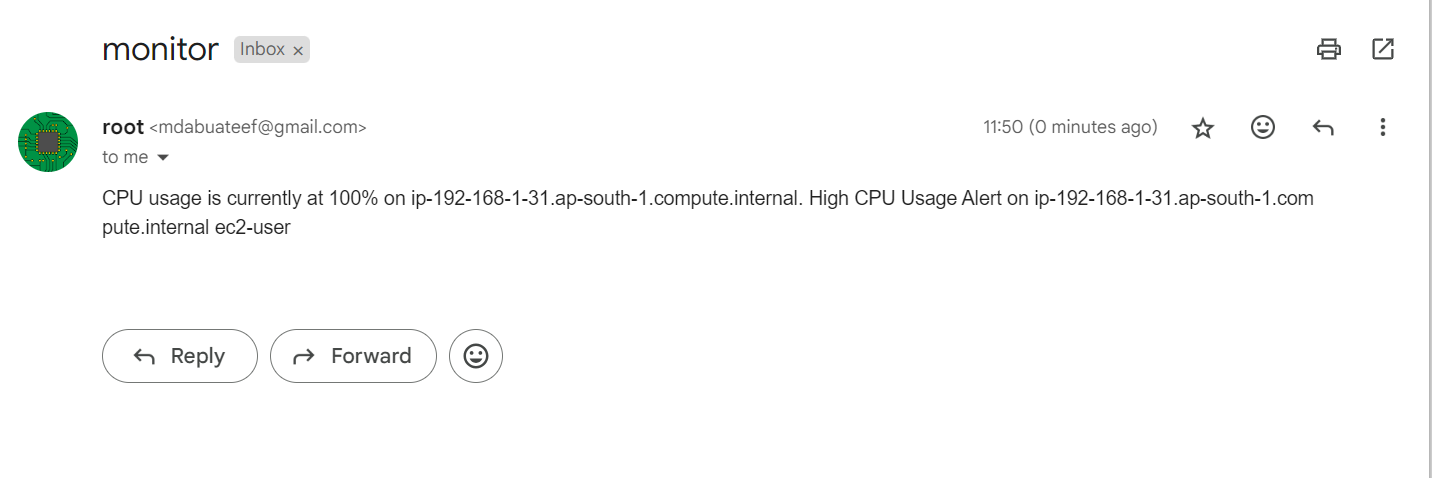
SUBJECT="High CPU Usage Alert on $(hostname)"

MESSAGE="CPU usage is currently at $CPU\_USAGE% on $(hostname)."

# Send email notification

echo "$MESSAGE" "$SUBJECT" $LOCALUSER | mail -s “minitor” mdabuateef@gmail.com

fi



Task3 -Shell script to monitor Memory usage of the server

#!/bin/bash

# Set threshold for memory usage (in percentage)

THRESHOLD=10

LOCAL\_USER="ec2-user"

# Get memory usage percentage

MEMORY\_USAGE=$(free | awk '/Mem/{printf("%.2f"), $3/$2\*100}')

# Check if memory usage exceeds threshold

if (( $(echo "$MEMORY\_USAGE > $THRESHOLD" | bc -l) )); then

SUBJECT="High Memory Usage Alert on $(hostname)"

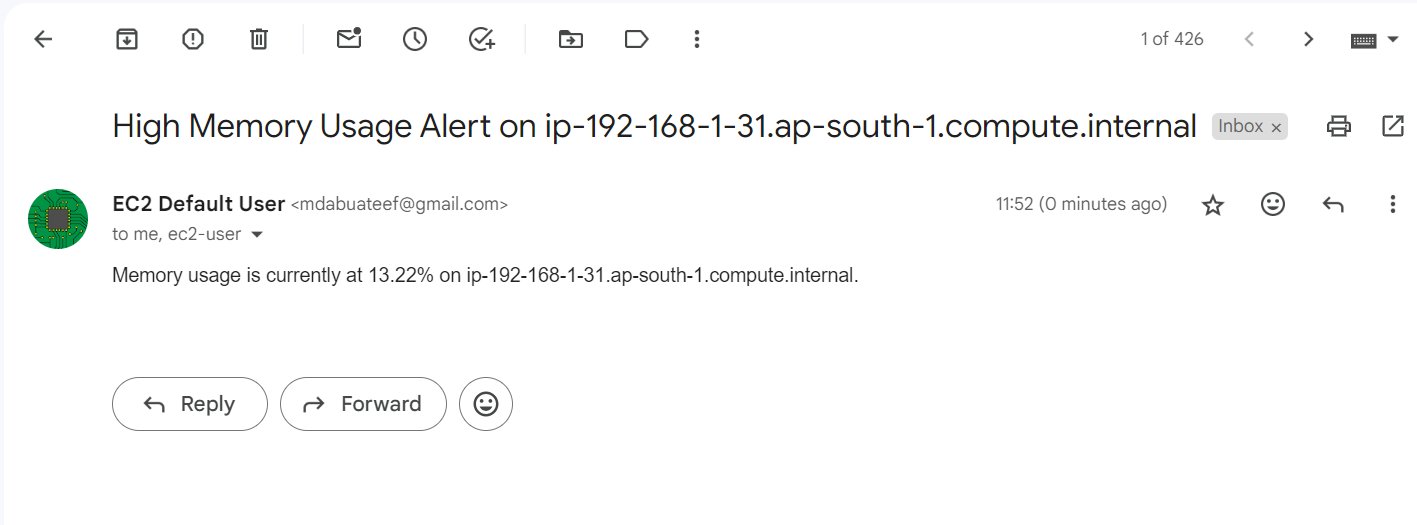
MESSAGE="Memory usage is currently at $MEMORY\_USAGE% on $(hostname)."

# Send email notification

echo "$MESSAGE" "$SUBJECT" "$LOCAL\_USER"

fi

run command ./memory.sh



Task 4: Install any sample web application like httpd, nginx etc..

Sudo yum install nginx -y

To verify nginx service using script

#!/bin/bash

# Check if Nginx configuration file exists in /etc/nginx/

if [ -f "/etc/nginx/nginx.conf" ]; then

echo "Nginx configuration file found in /etc/nginx/"

else

echo "Nginx configuration file not found in /etc/nginx/"

exit 1 # Exit with non-zero status to indicate failure

fi

# Check if Nginx log file exists in /var/log/nginx/

if [ -f "/var/log/nginx/error.log" ]; then

echo "Nginx log file found in /var/log/nginx/"

else

echo "Nginx log file not found in /var/log/nginx/"

fi

# Check if Nginx service is active

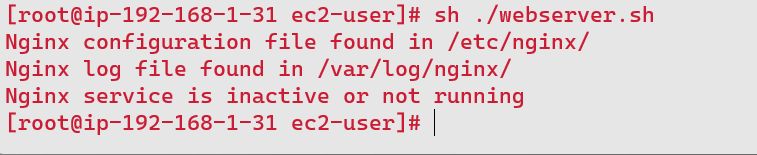
if systemctl is-active --quiet nginx; then

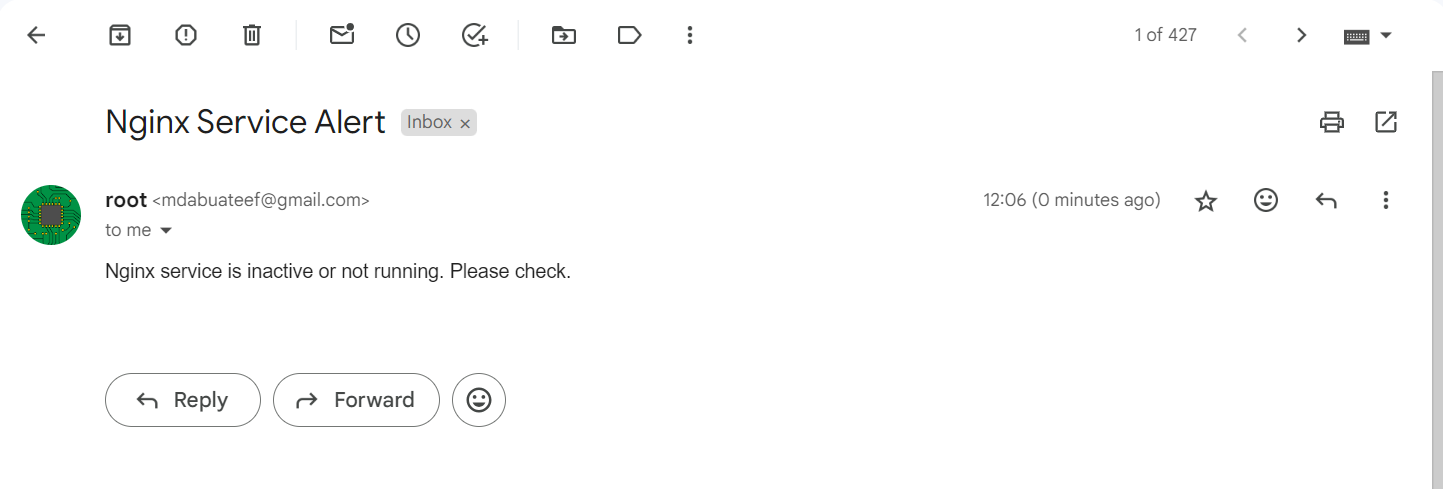
echo "Nginx service is active"

else

echo "Nginx service is inactive or not running"

fi





task 4: - Shell script to check the status of the web server if the web application is up it has to print the message as "server is up and running" if the web application is down it has to send notification to email The shell script has to run for every 5 mins

#!/bin/bash

# Function to send email notification

send\_notification() {

local recipient="mdabuateef@gmail.com" # Update with your email address

local subject="Nginx Service Alert"

local message="Nginx service is inactive or not running. Please check."

echo "$message" | mail -s "$subject" "$recipient"

}

# Check if Nginx configuration file exists in /etc/nginx/

if [ -f "/etc/nginx/nginx.conf" ]; then

echo "Nginx configuration file found in /etc/nginx/"

else

echo "Nginx configuration file not found in /etc/nginx/"

exit 1 # Exit with non-zero status to indicate failure

fi

# Check if Nginx log file exists in /var/log/nginx/

if [ -f "/var/log/nginx/error.log" ]; then

echo "Nginx log file found in /var/log/nginx/"

else

echo "Nginx log file not found in /var/log/nginx/"

fi

# Check if Nginx service is active

if systemctl is-active --quiet nginx; then

echo "Nginx service is active"

else

echo "Nginx service is inactive or not running"

# Send email notification

send\_notification

fi

save the file and open a crontab with command crontab -e

and use \*/5 \* \* \* \* ./path/to/file.sh

Task 5:

- Shell script to take backups automatically at particular period of time.

#!/bin/bash

# Define backup directory

backup\_dir="/home/ec2-user/backup\_2"

# Define source directory to be backed up

source\_dir="/home/ec2-user/backup"

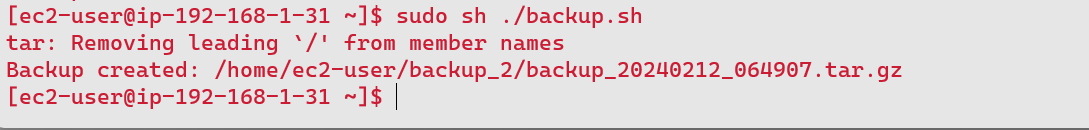
# Create backup filename with timestamp

backup\_file="$backup\_dir/backup\_$(date +'%Y%m%d\_%H%M%S').tar.gz"

# Create tar archive of source directory

tar -czf "$backup\_file" "$source\_dir" && echo "Backup created: $backup\_file" || echo "Backup failed"

save the file and open a crontab with command > crontab -e and use \* 18 \* \* \* /bin/bash ~/backup.sh



task 5: - Write a script to automatically delete files older than a week in specific directories and create a compressed archive with a timestamp

#! /bin/bash

# Define directories to clean and archive

directories=(

"/path/to/directory1"

"/path/to/directory2"

)

# Define the threshold for deletion (1 week ago)

threshold=$(date -d '1 week ago' +%s)

# Create a timestamp for the archive filename

timestamp=$(date +'%Y%m%d\_%H%M%S')

# Loop through each directory

for dir in "${directories[@]}"; do

# Find files older than the threshold and delete them

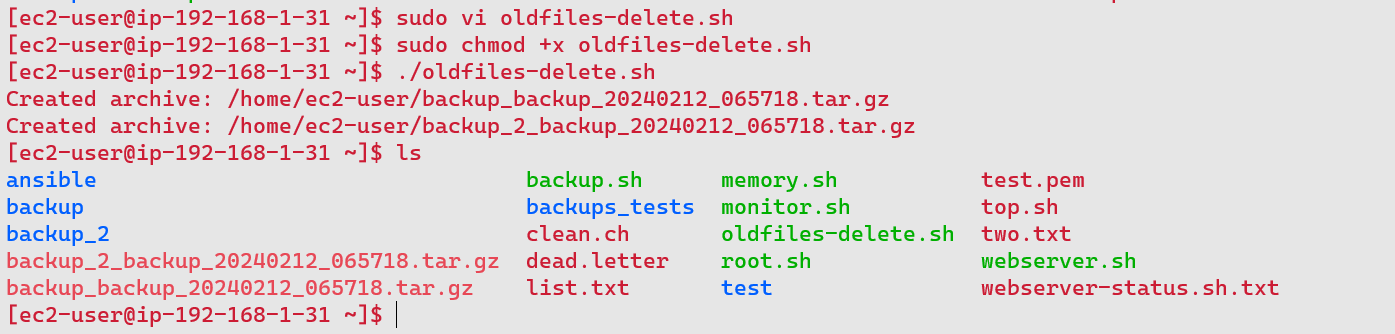
find "$dir" -type f -mtime +7 -exec rm -f {} \;

# Create a compressed archive with timestamp

archive\_file="${dir}\_backup\_$timestamp.tar.gz"

tar -czf "$archive\_file" -C "$dir" .

echo "Created archive: $archive\_file"



Done