CLOUD COMPUTING ASSIGNMENT 01

Name: Ayesha Saif

Hawk Id: asaif@hawk.iit.edu

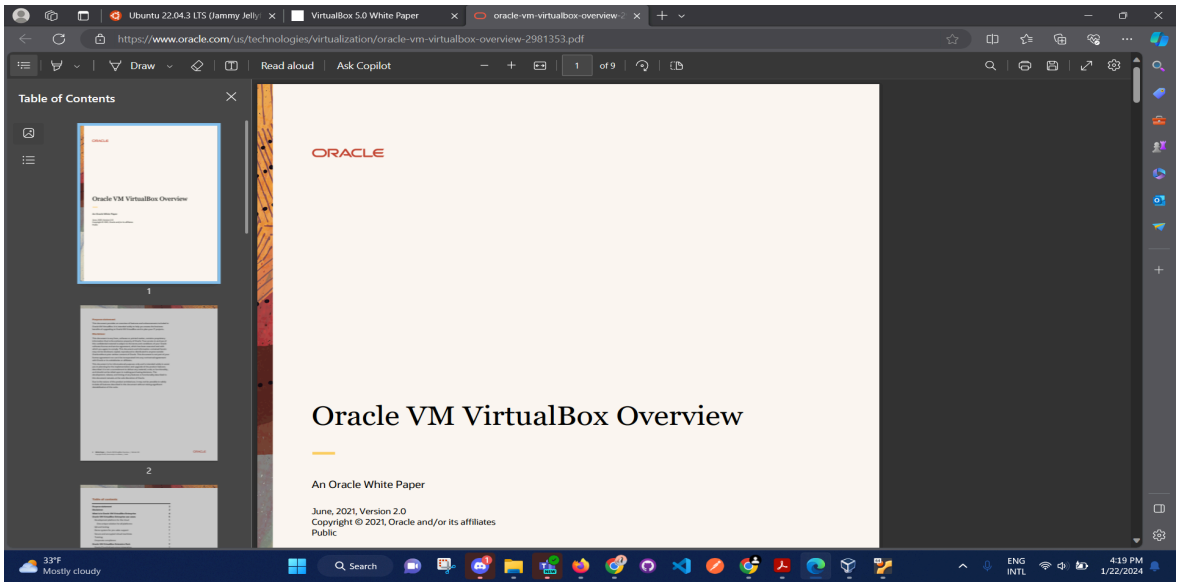
Subject: CS 553 Cloud Computing

Submission To: Prof. Ioan Raicu

CS553 Homework #1

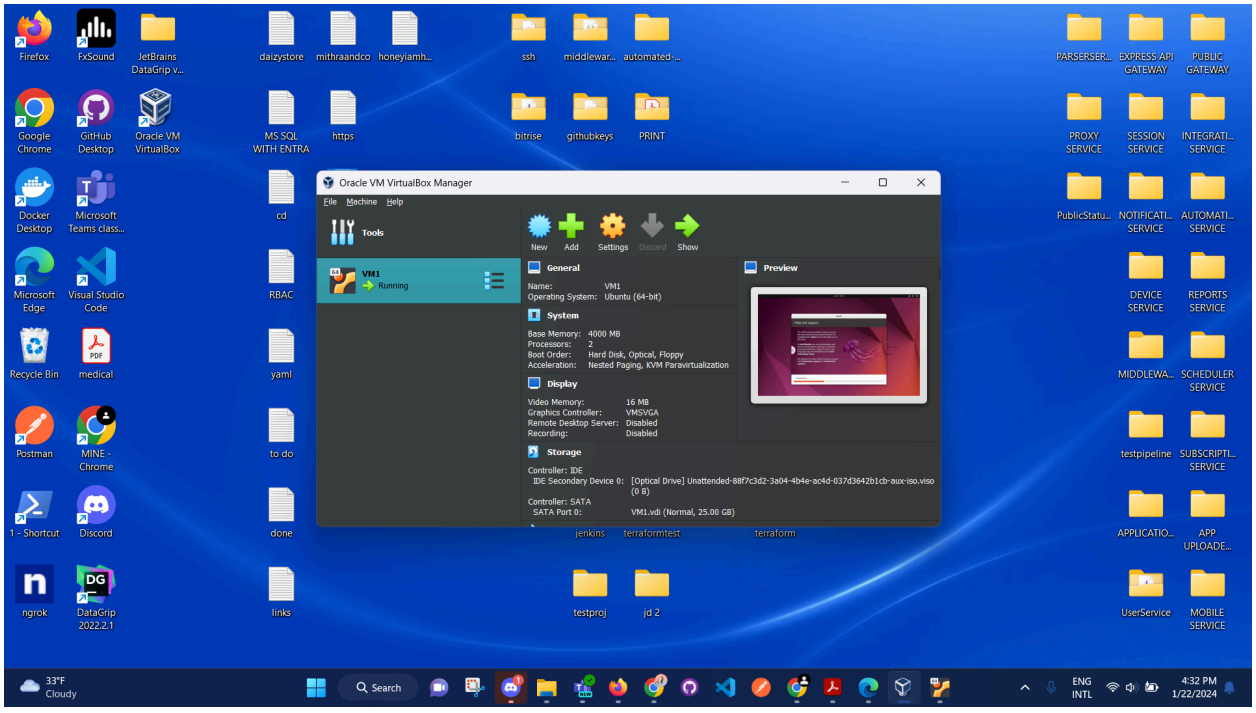
LINUX & COMPUTER SYSTEMS

1. Setup VM, Linux, and basic testing (Screenshot of each step are attached)
   a. Read Oracle VirtualBox White Paper

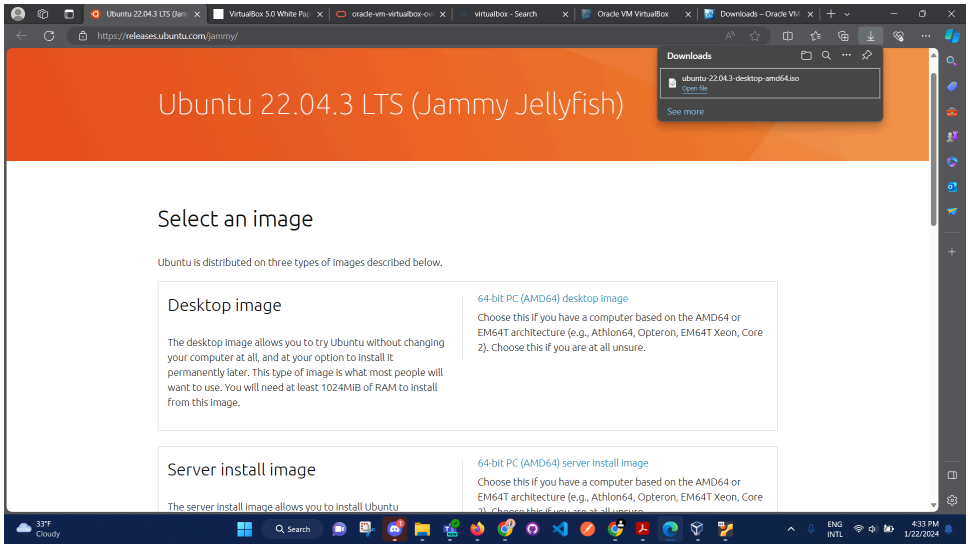


   b. Download Oracle VirtualBox 7.0

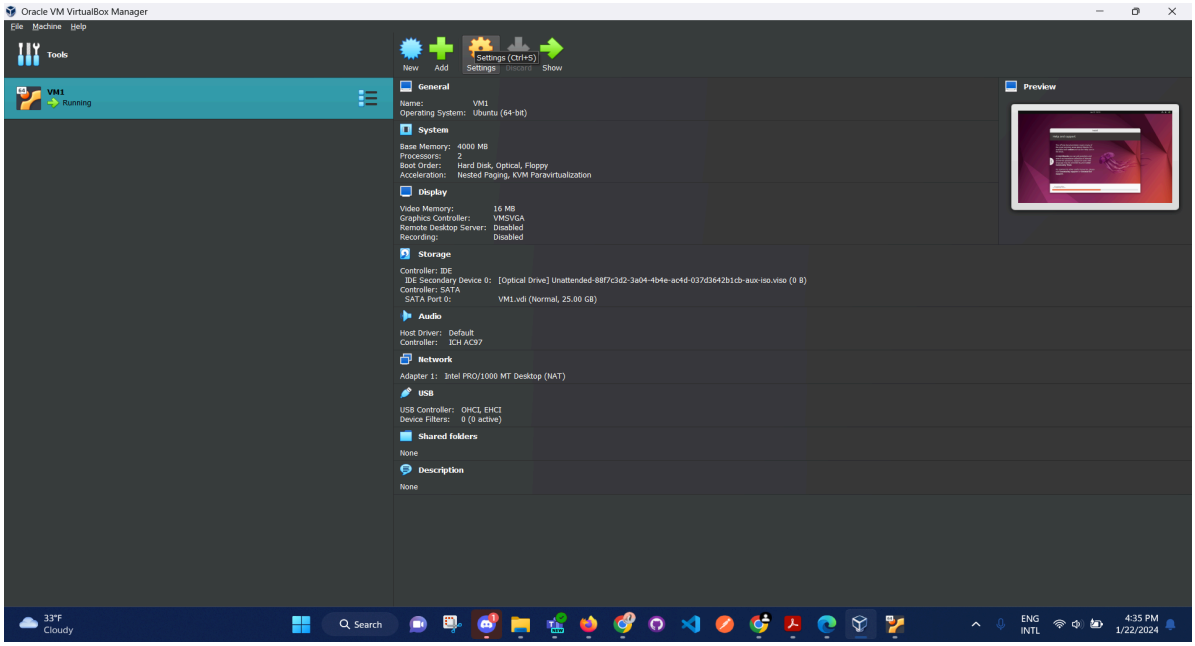
c. Install VirtualBox 7.0



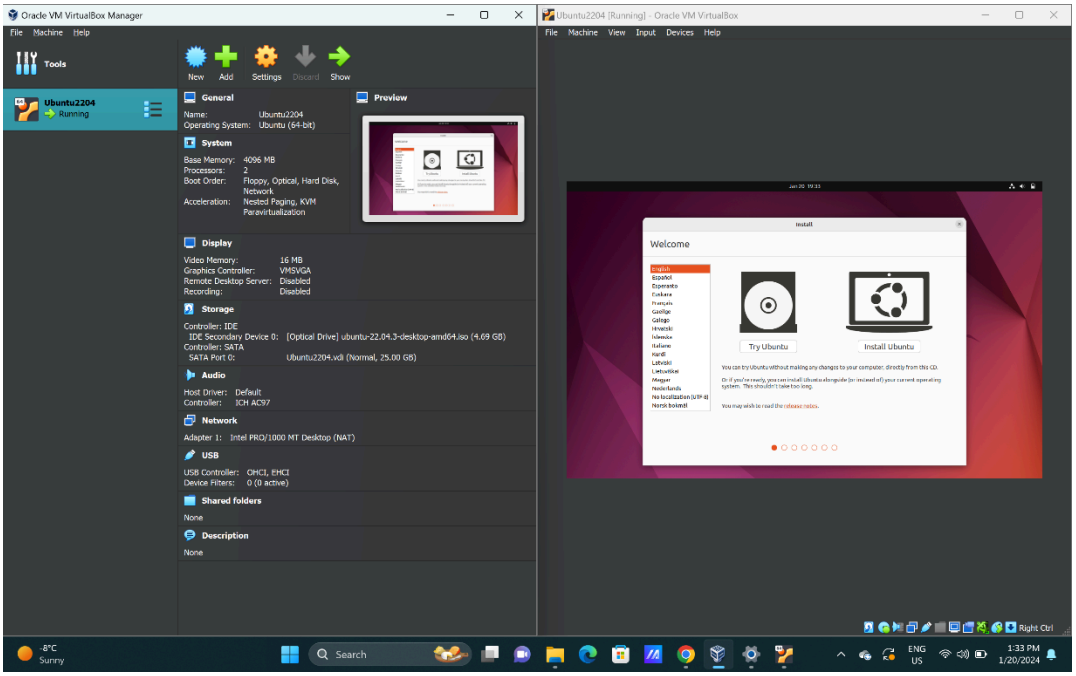
d. Download Ubuntu 22.04 Linux ISO Image



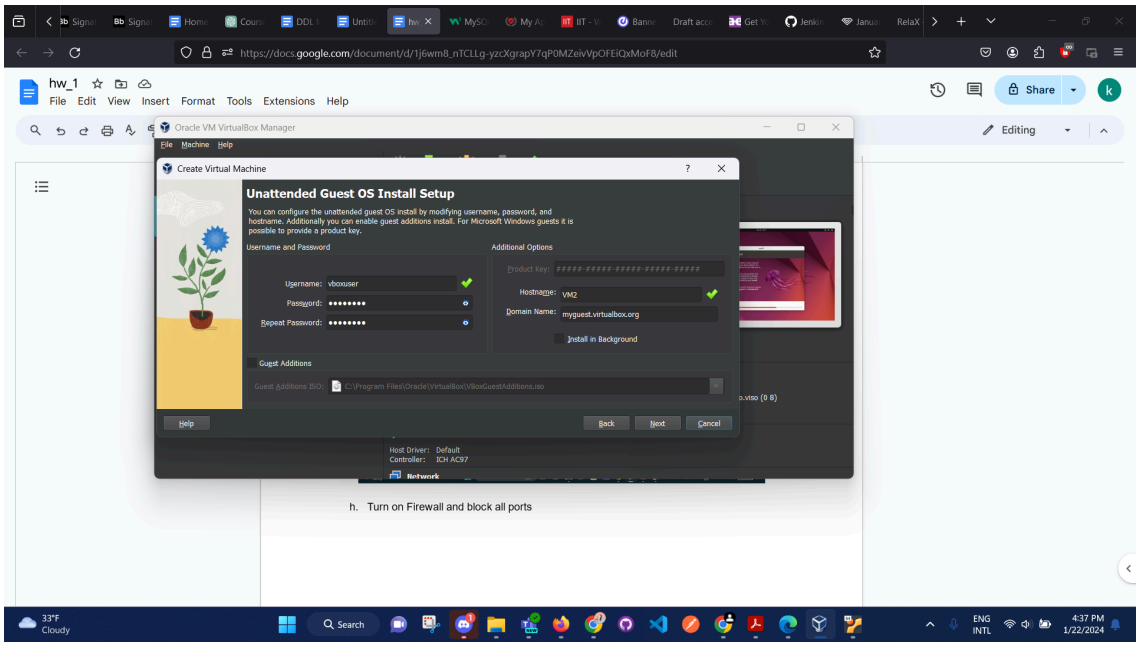
e. Create Virtual Machine (VM), to support Linux, Ubuntu, 64-bit, 4GB RAM, Virtual Disk 25GB, VDI image, dynamically allocated, 2-core, and a network interface (1GbE or Wifi) with Bridged Adapter



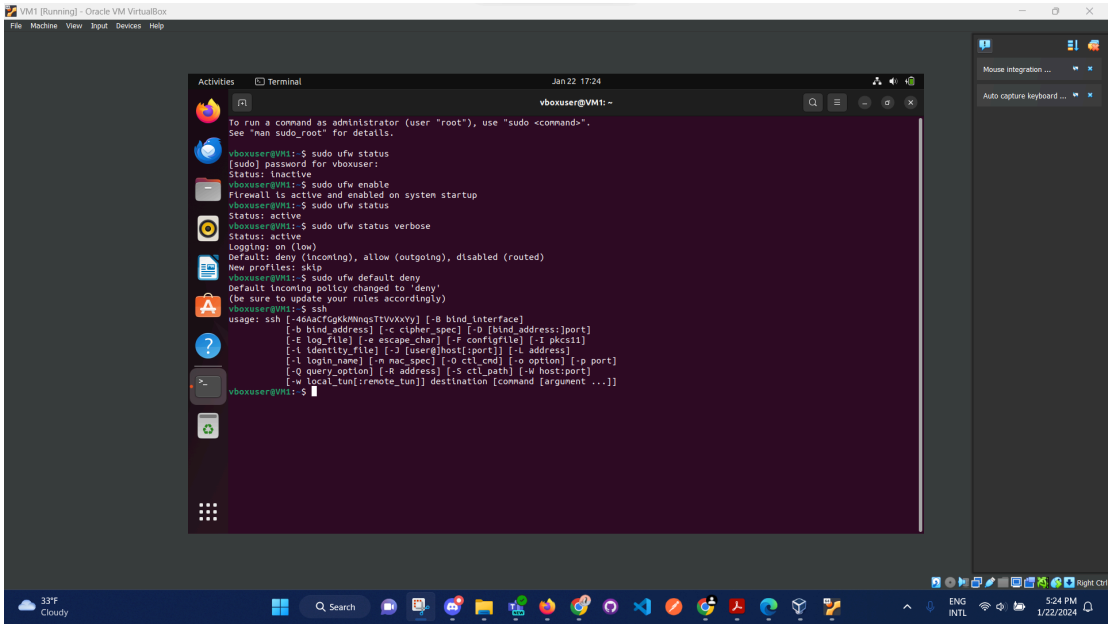
f. Install Linux from the ISO image



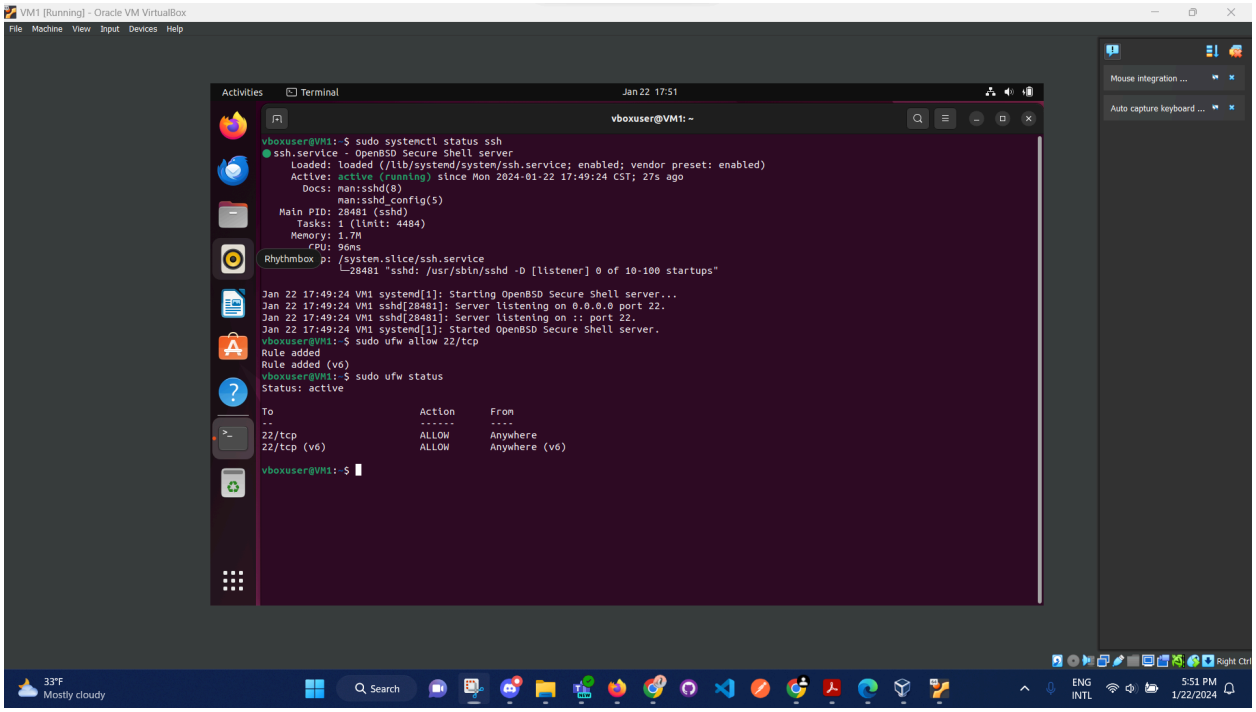
g. Create a user id and password



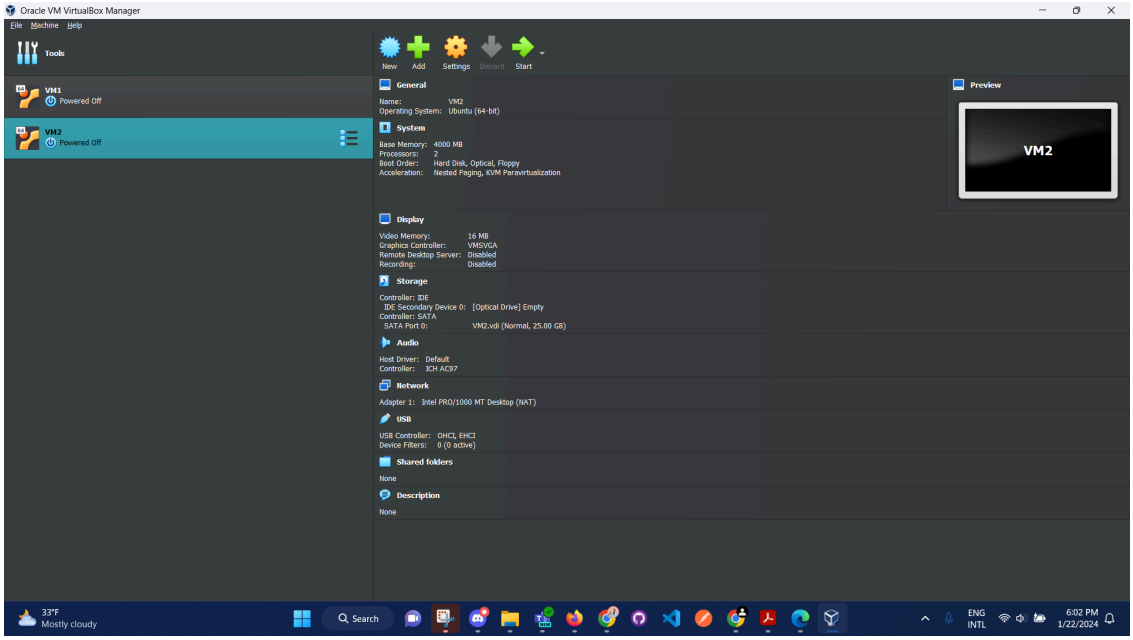
h. Turn on Firewall and block all ports



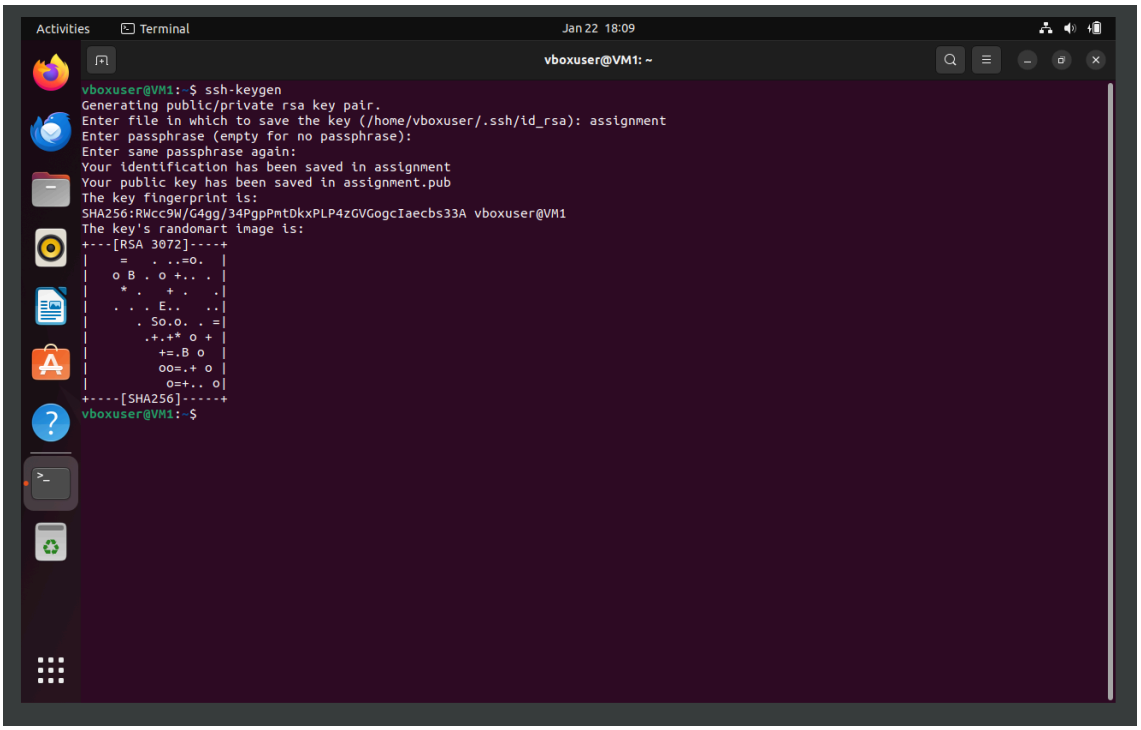
i. Enable SSH access to your new Linux installation; open SSH port in firewall

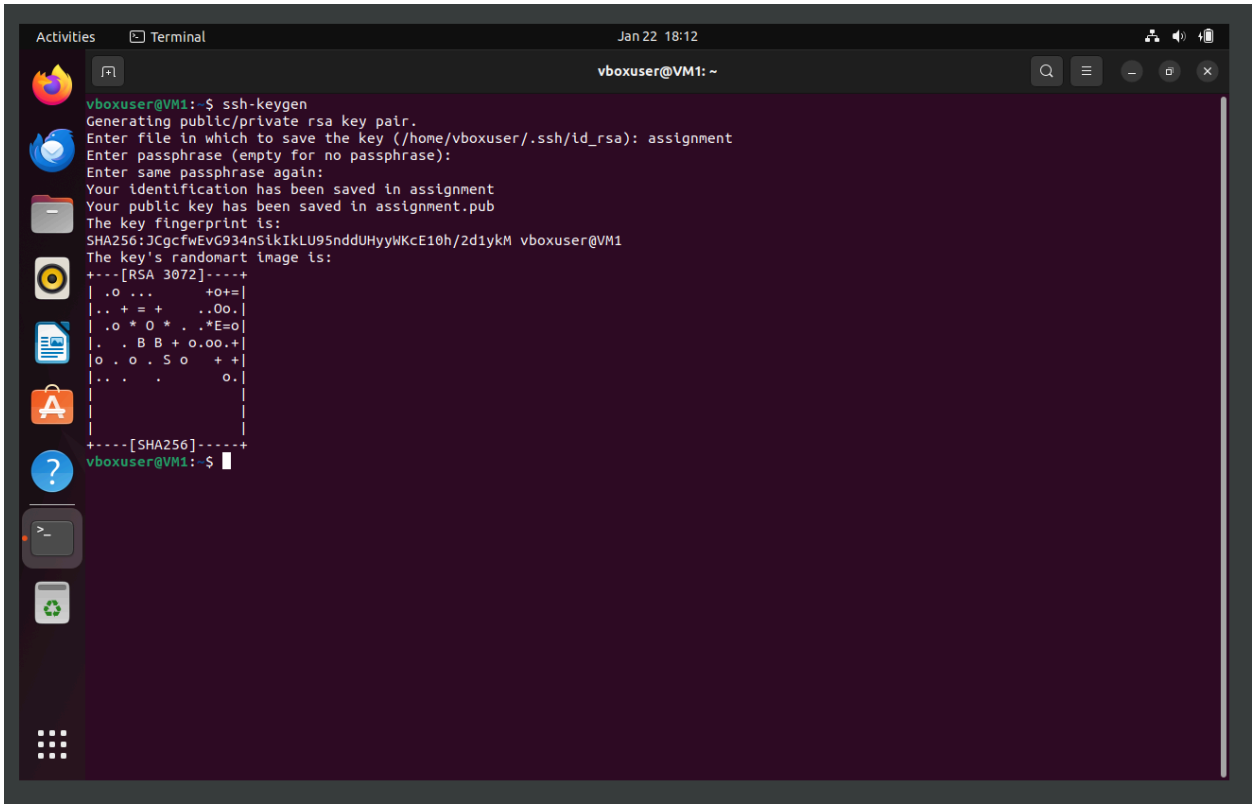


j. Repeat steps 5 through 9, and create another VM with the same specifications as the first one.

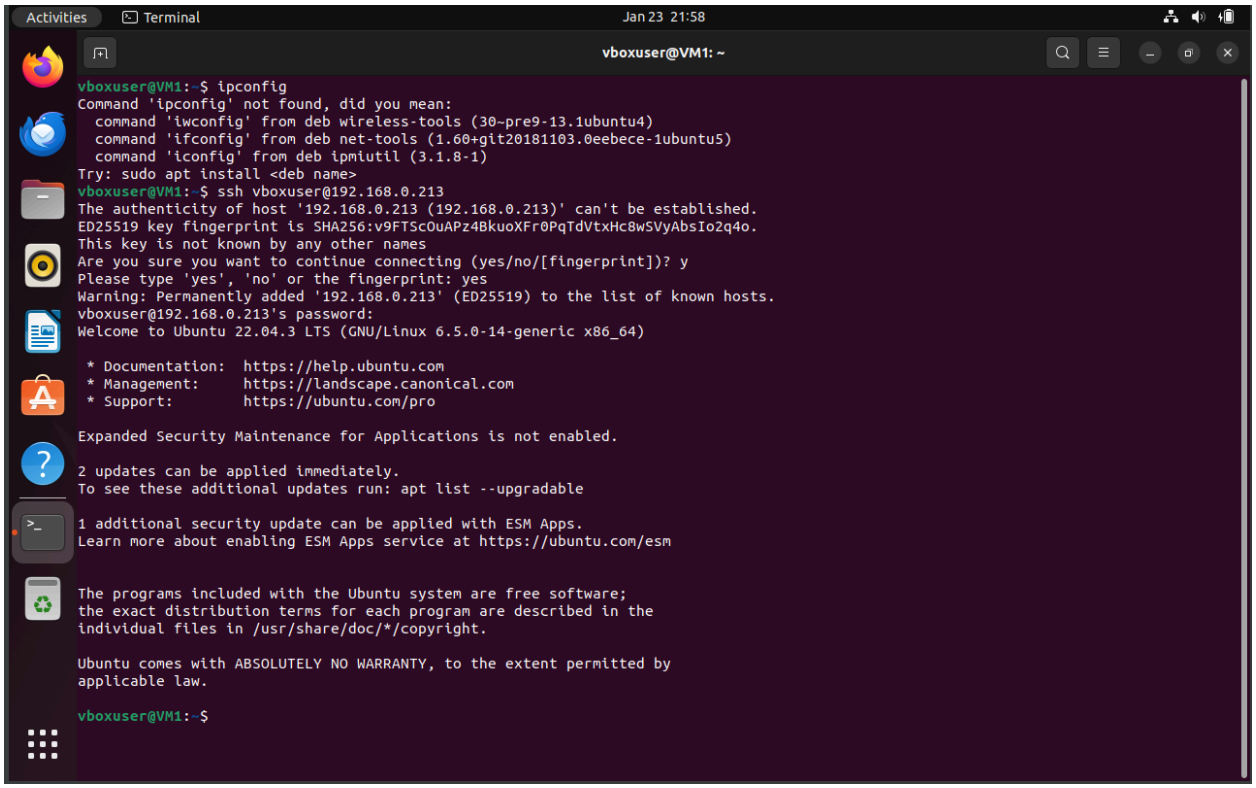


k. Create private/public keys and install them properly in both of your new VMs





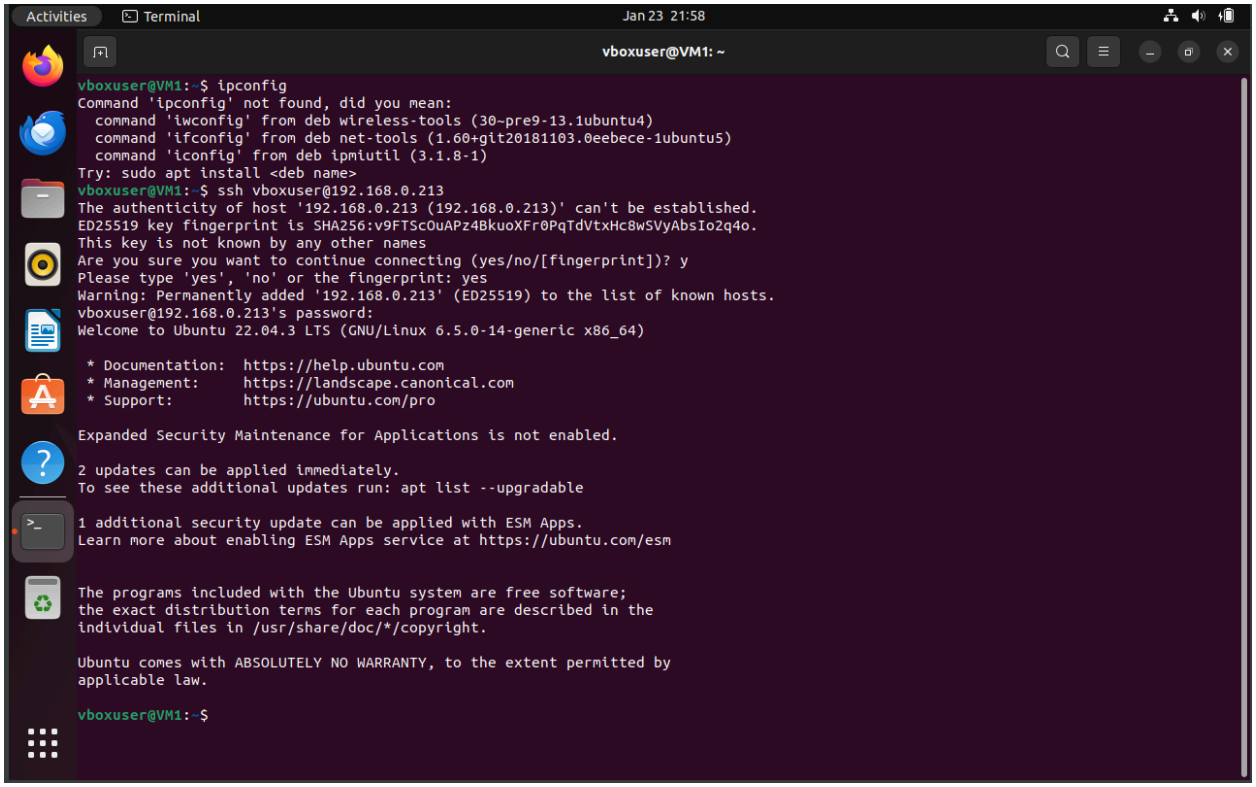
I. Test that you can connect remotely to your keys, from one VM to the other VM



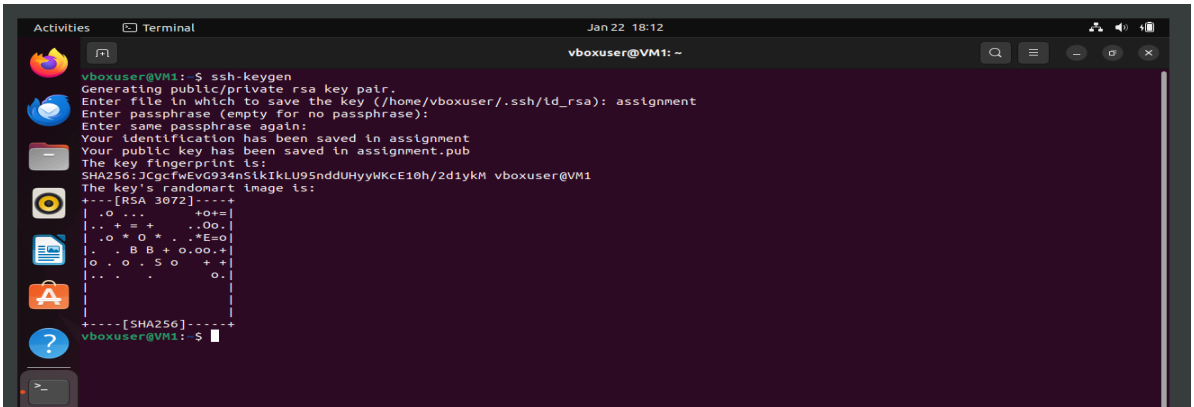
CLOUD COMPUTING ASSIGNMENT 01

2. <u>Show an example of using the following commands; take screen shots of your commands; clear screen between each command; explain in own words what these commands do:</u>

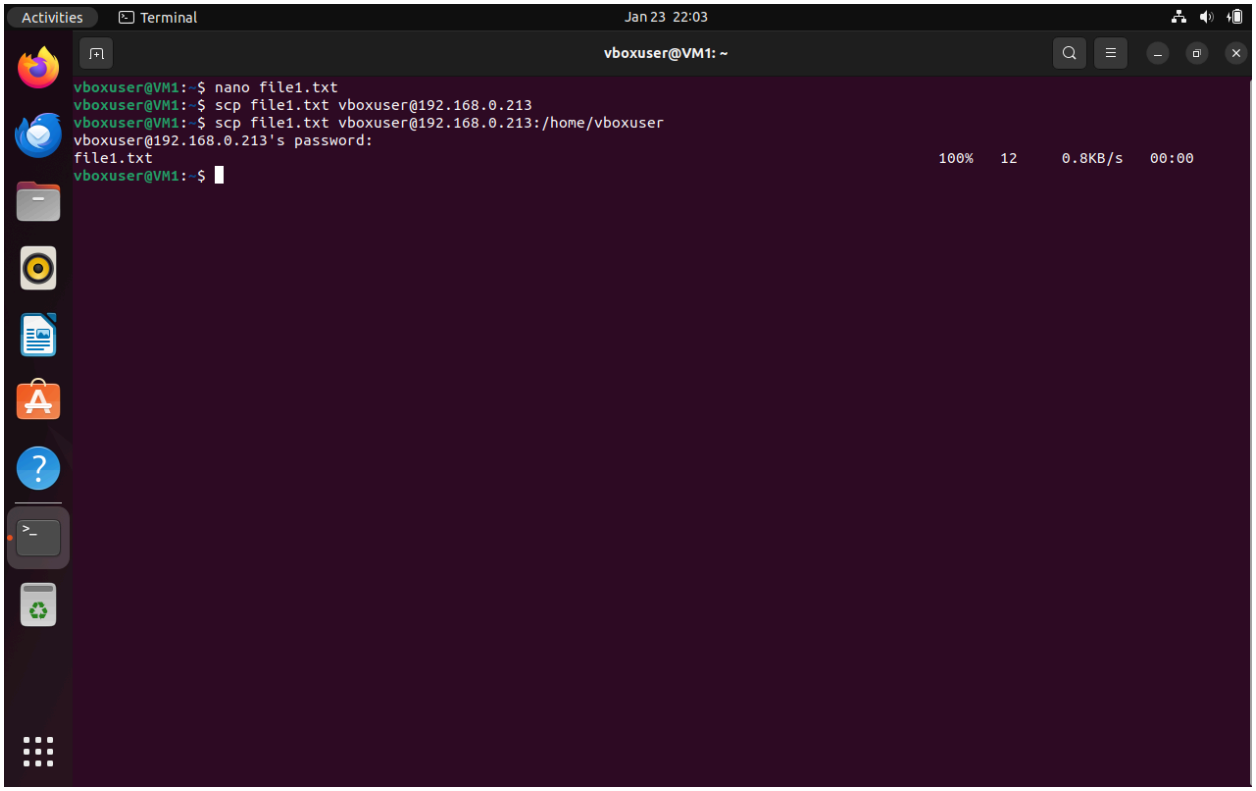
1) ssh (Secure Shell):
   Description: Connects to a remote server securely over a network.
   Example Usage: ssh username@remote_server

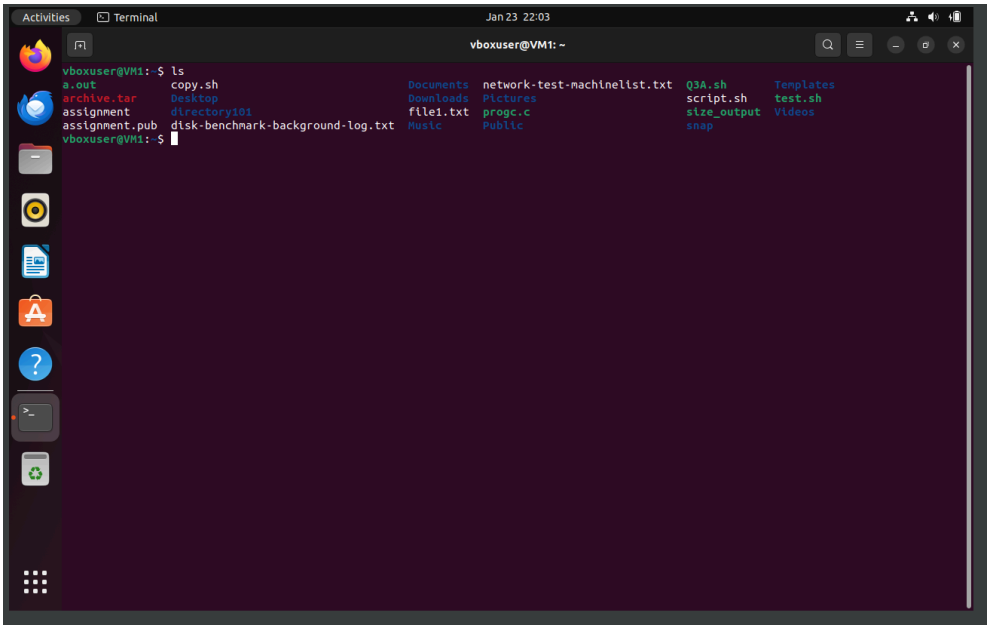


2) ssh-keygen:
   Description: Generates SSH key pairs for secure authentication.
   Example Usage: ssh-keygen -t rsa

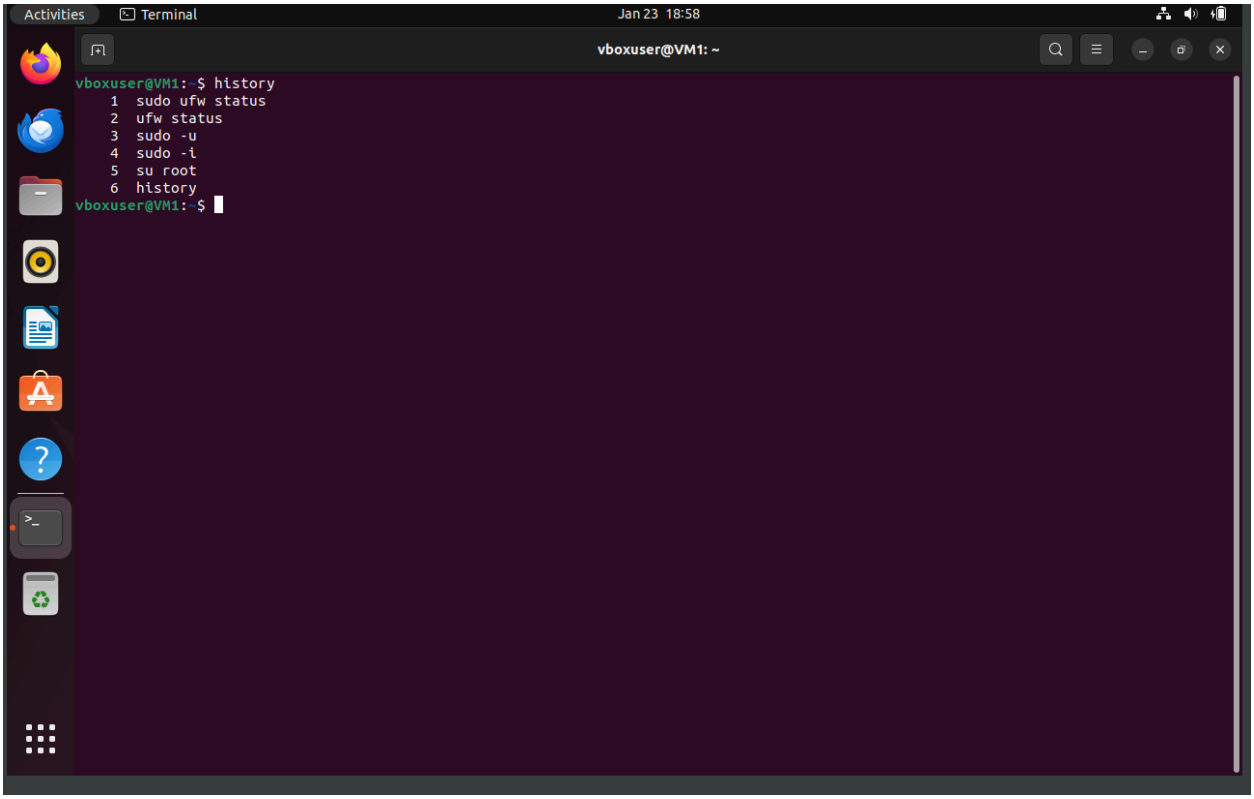


3) scp (Secure Copy):
   Description: Copies files securely between hosts on a network.
   Example Usage: scp local_file.txt username@remote_server:/path/to/destination

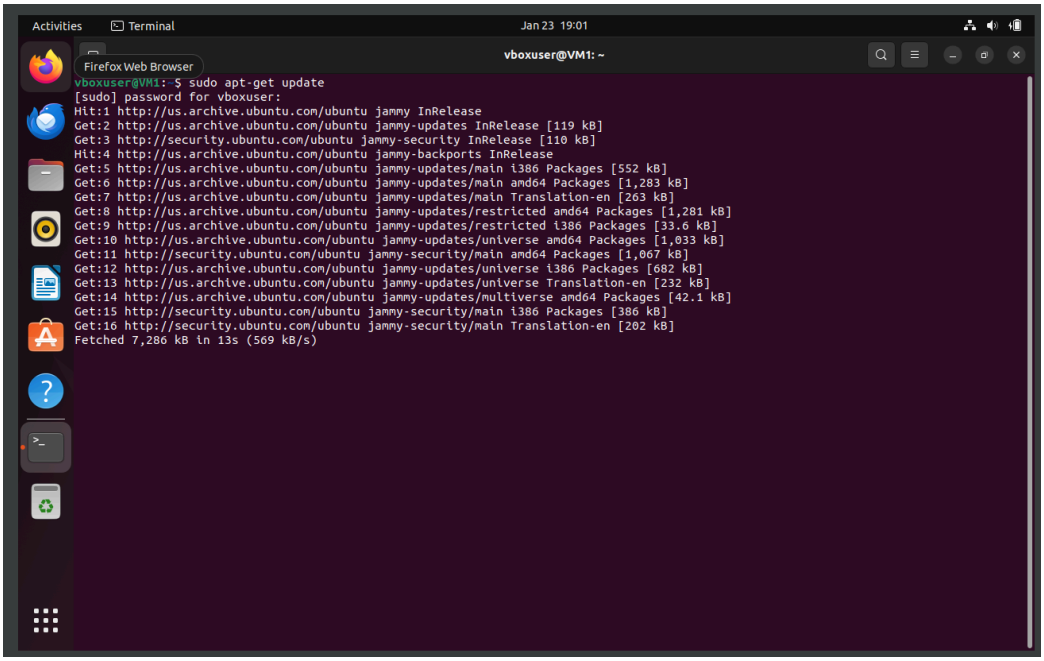




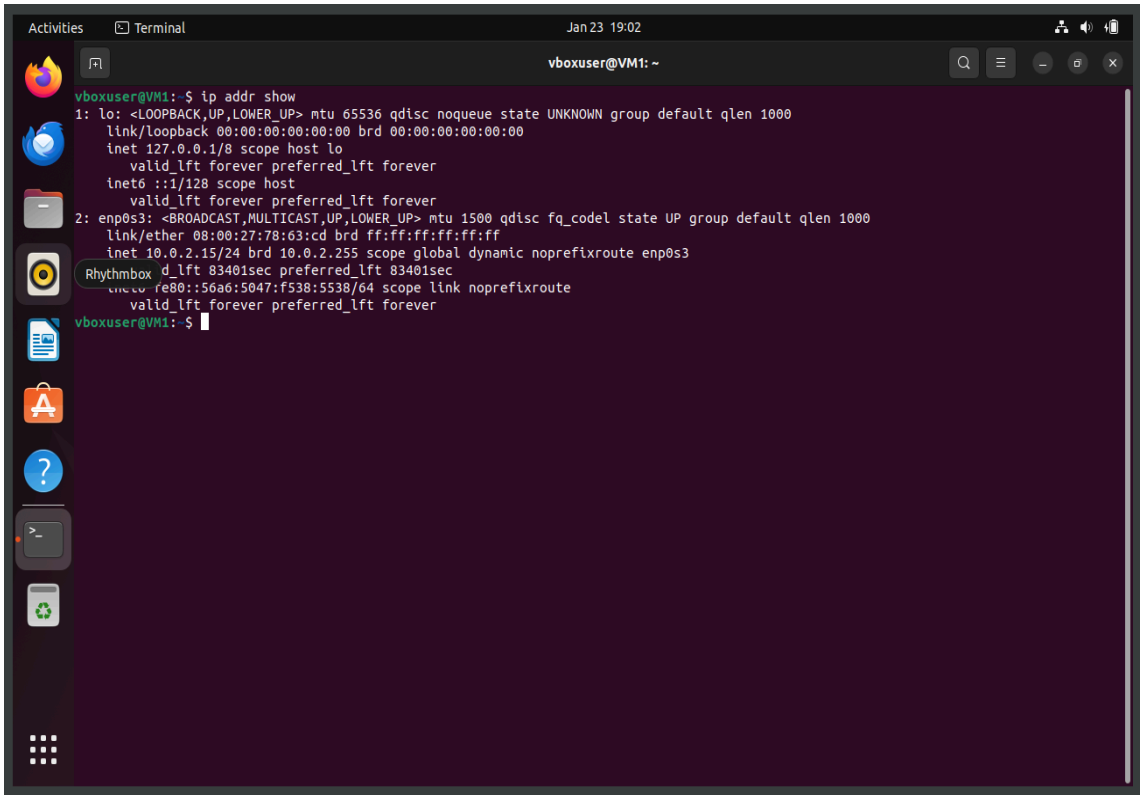
4) history:
   Description: Displays a list of previously executed commands.
   Example Usage: history



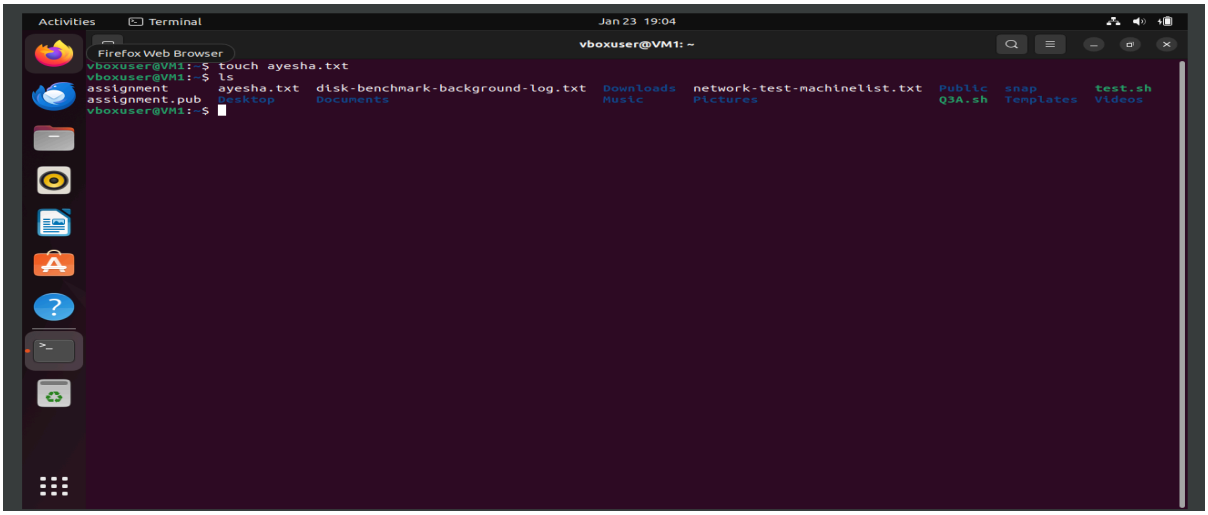
5) sudo (Superuser Do):
   Description: Executes a command with elevated privileges (as a superuser).
   Example Usage: sudo command_to_execute



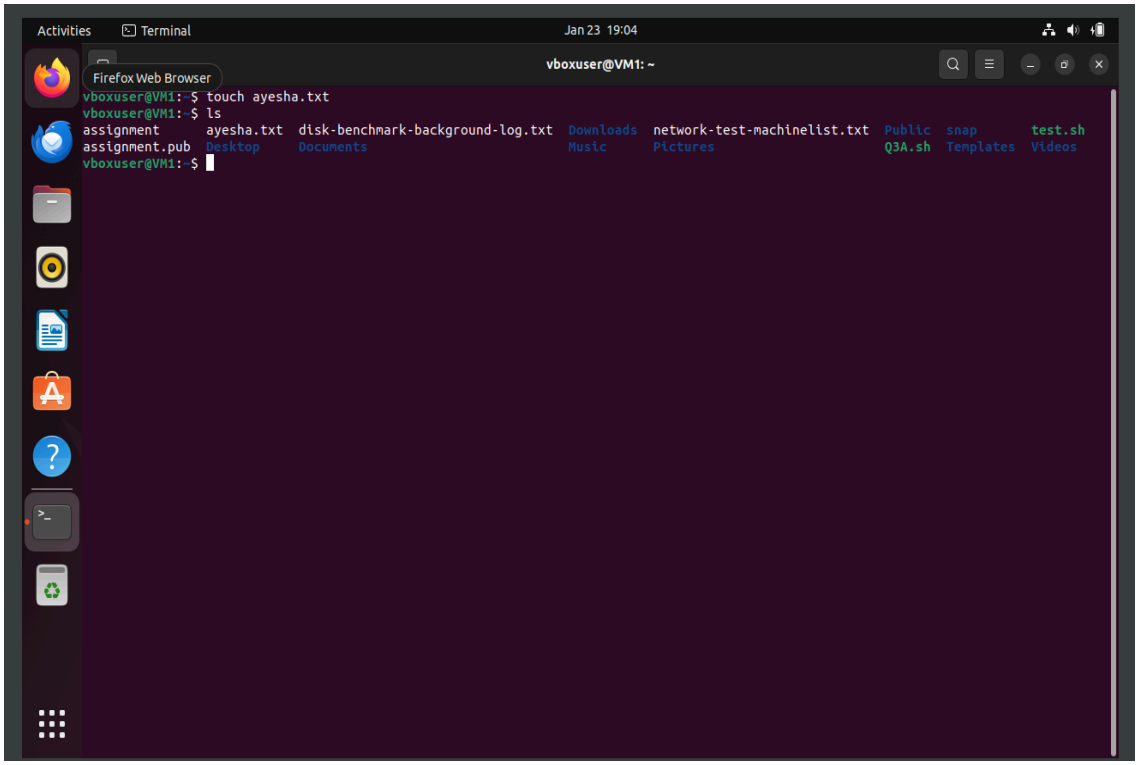
6) ip:
   Description: Shows and manipulates network interfaces.
   Example Usage: ip addr show



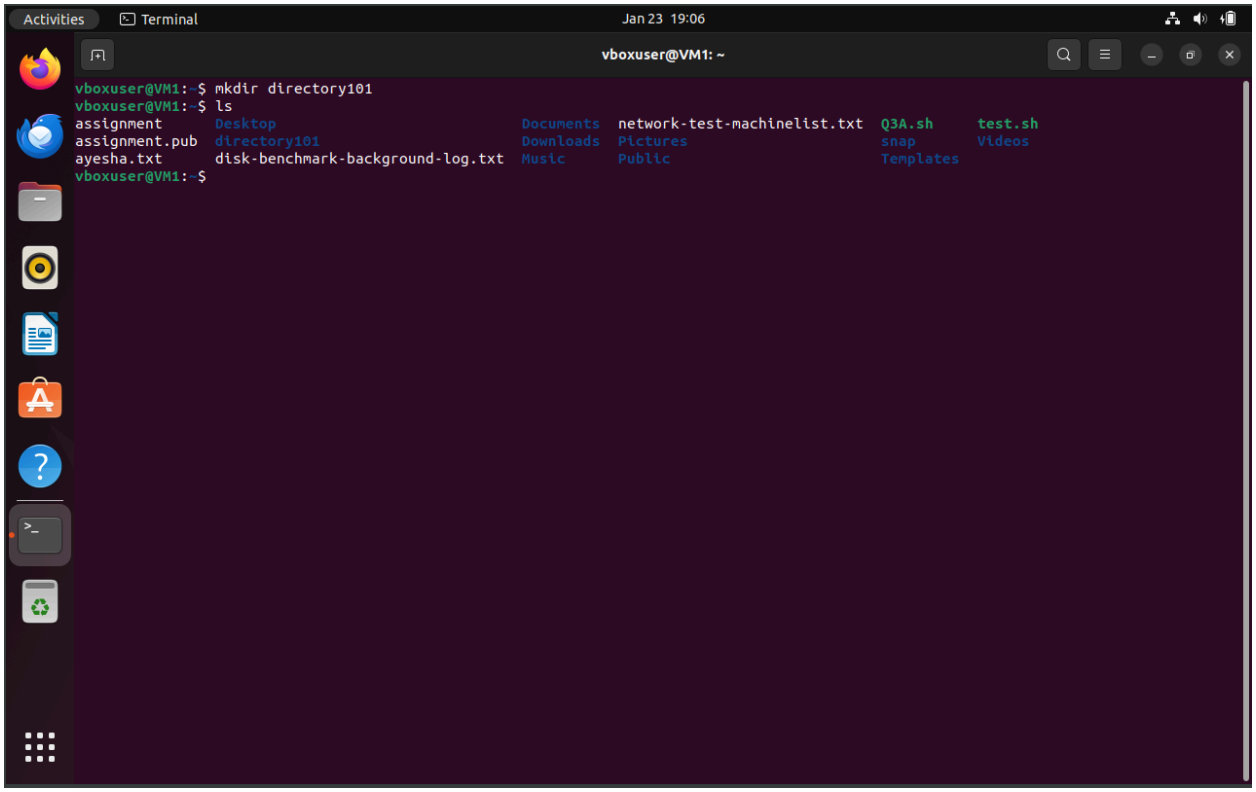
7) touch:
   Description: Creates an empty file or updates the access and modification timestamps of a file.
   Example Usage: touch filename.txt



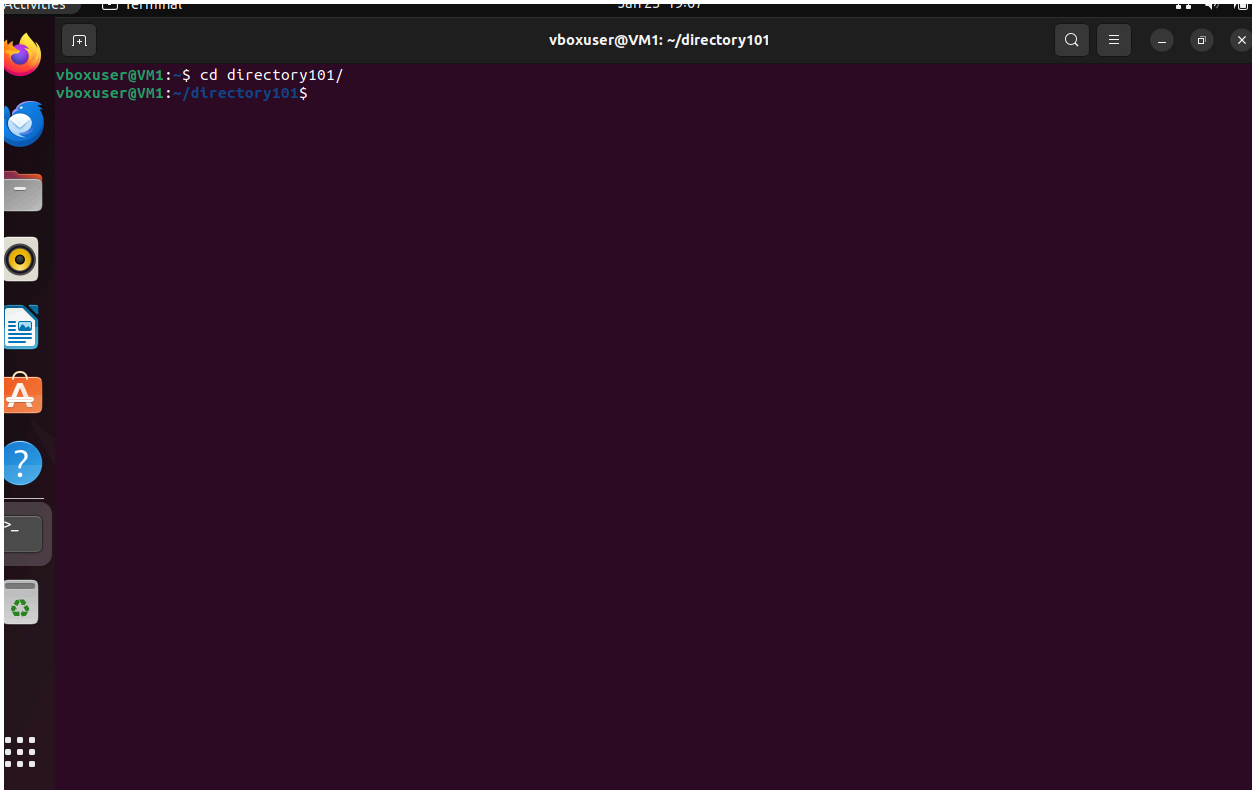
8) ls (List):
Description: Lists files and directories in the current directory.
Example Usage: ls



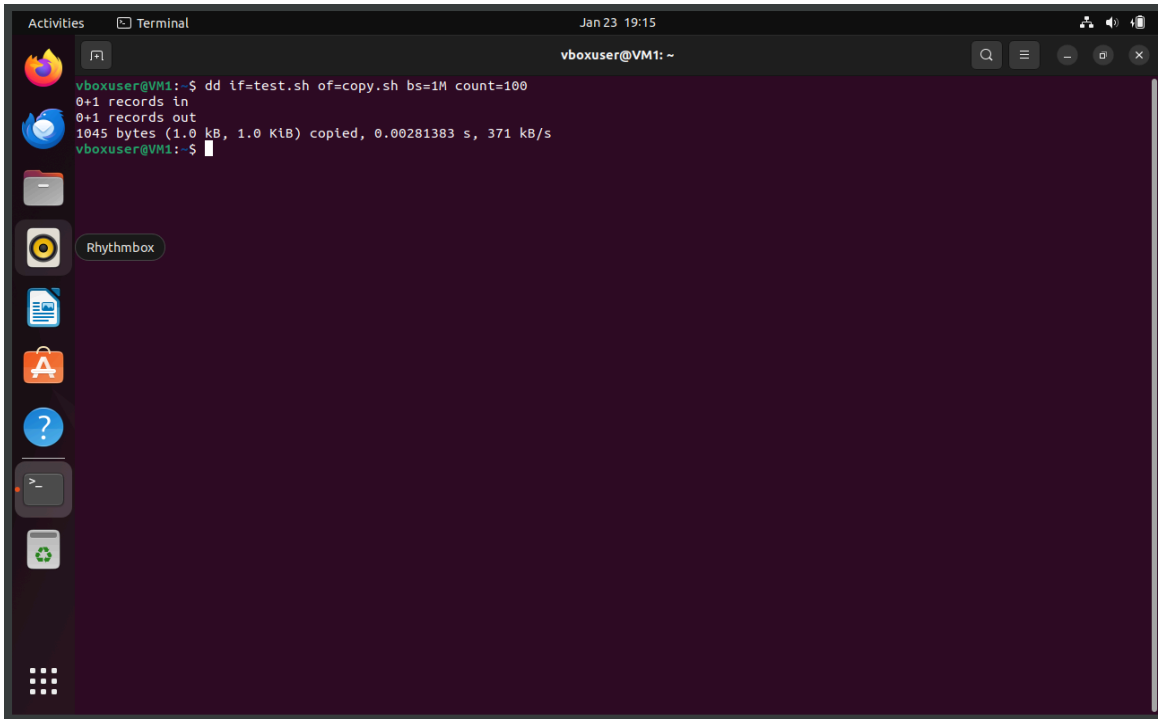
9) mkdir (Make Directory):
Description: Creates a new directory.
Example Usage: mkdir new_directory



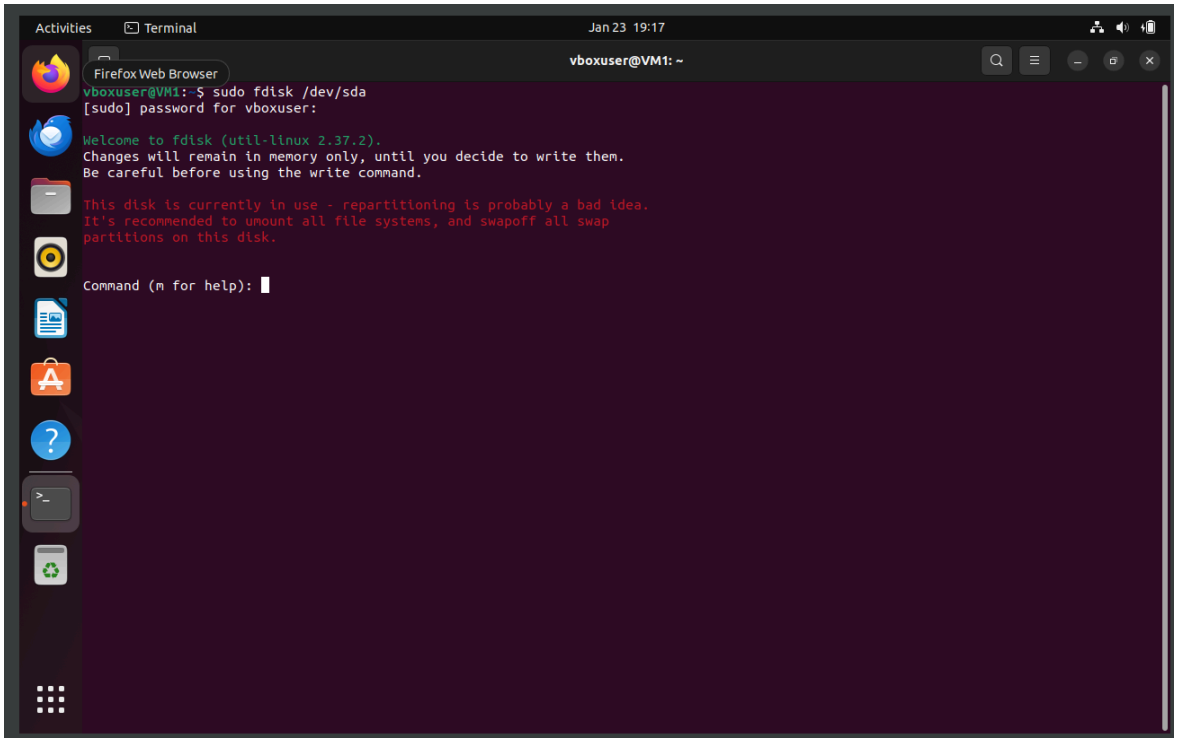
10) cd (Change Directory):
    Description: Changes the current working directory.
    Example Usage: cd /path/to/directory



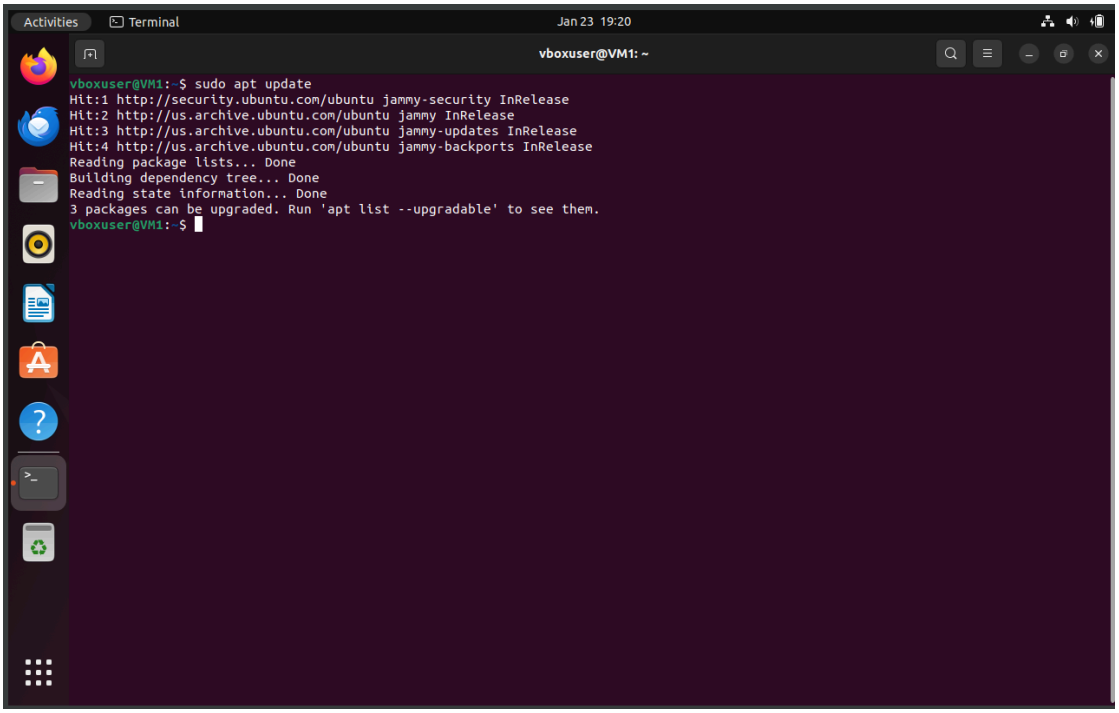
11) dd:
Description: Copies and converts data according to specified options.
Example Usage: dd if=input_file of=output_file bs=4k



12) fdisk:
Description: Manipulates disk partition tables.
Example Usage: fdisk /dev/sdX

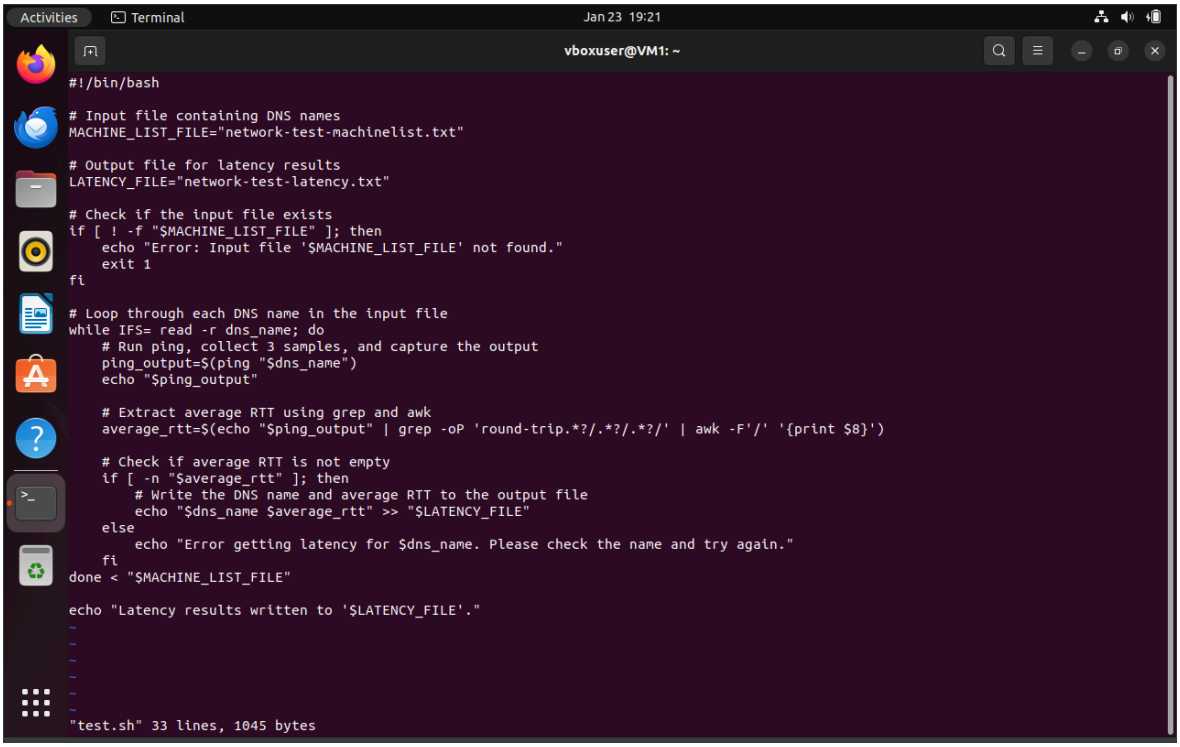


13) apt (Advanced Package Tool):
Description: Package management tool for Debian-based systems.
Example Usage: sudo apt update && sudo apt install package_name



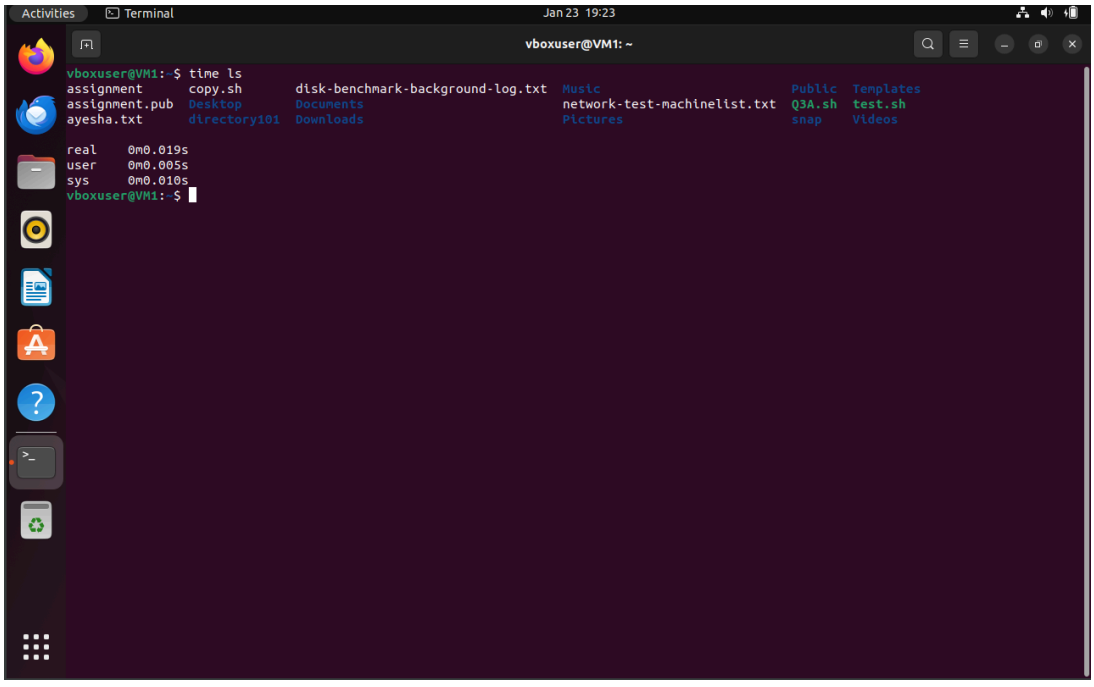
14) vi:

Description: Text editor with modes for inserting, navigating, and editing text.
Example Usage: vi filename.txt

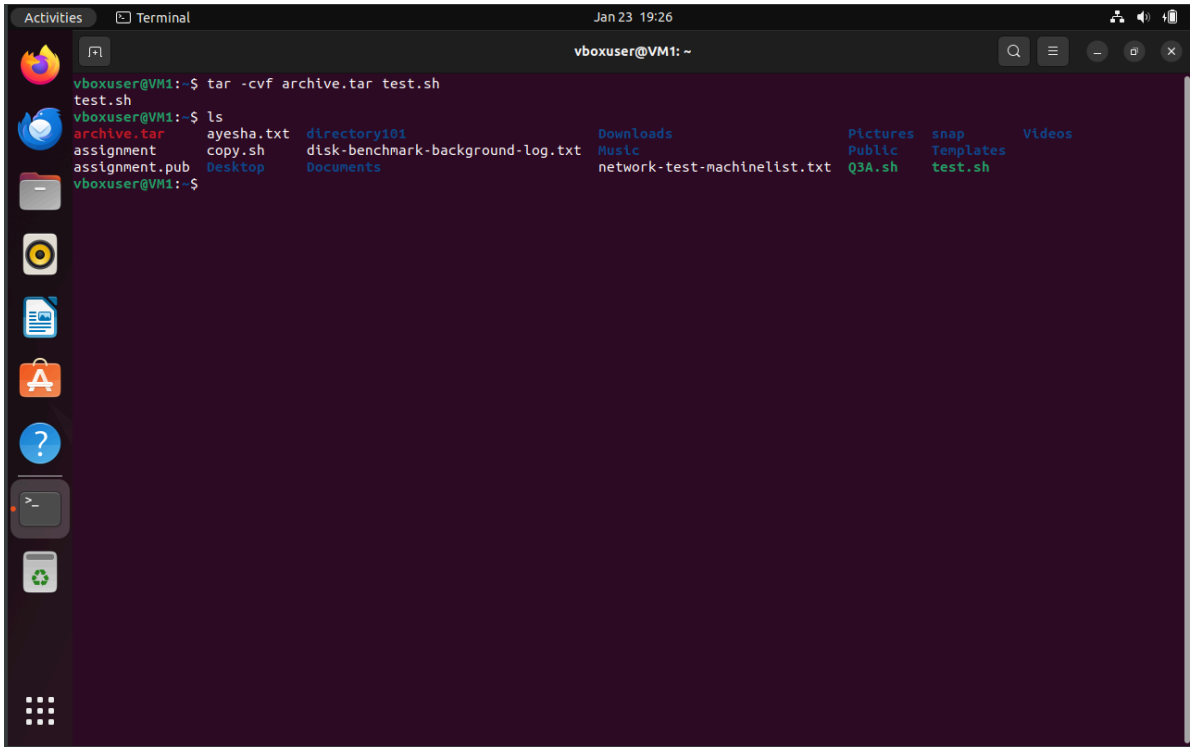


15) time:

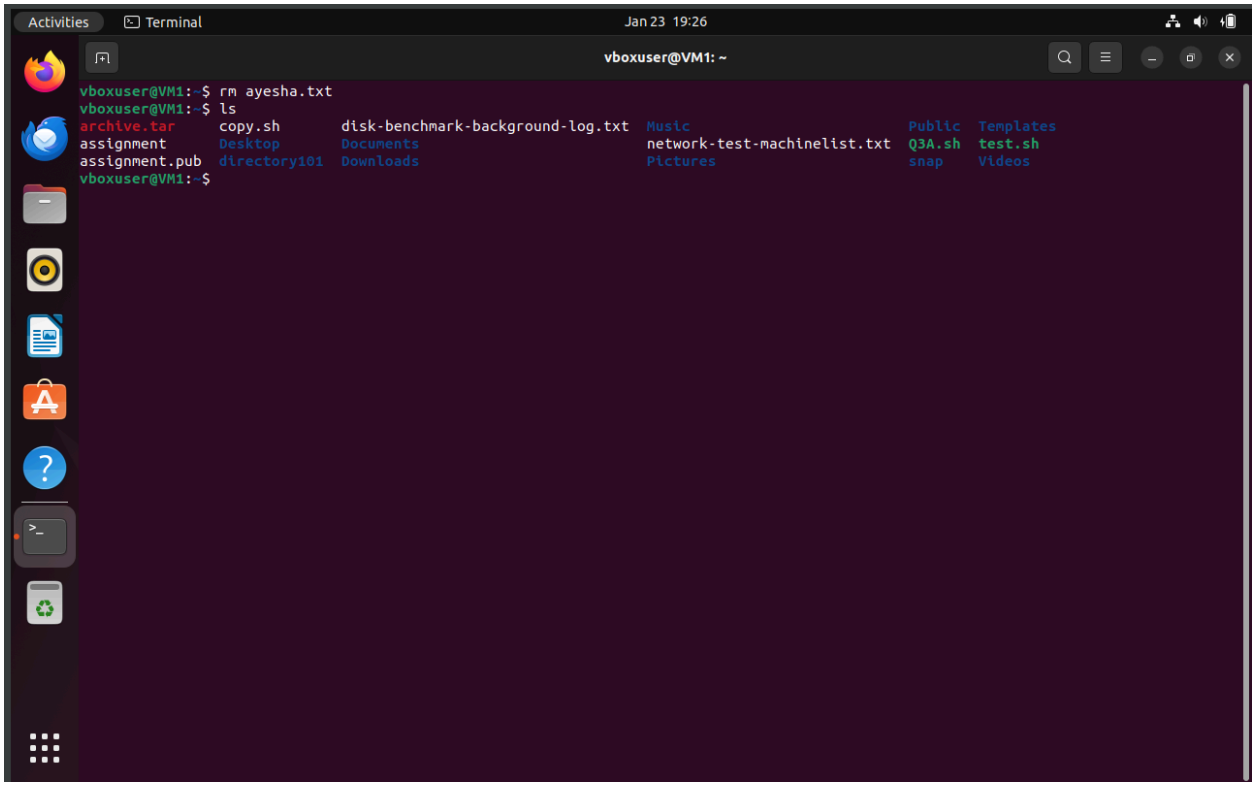
Description: Measures the execution time of a command.
Example Usage: time command_to_measure



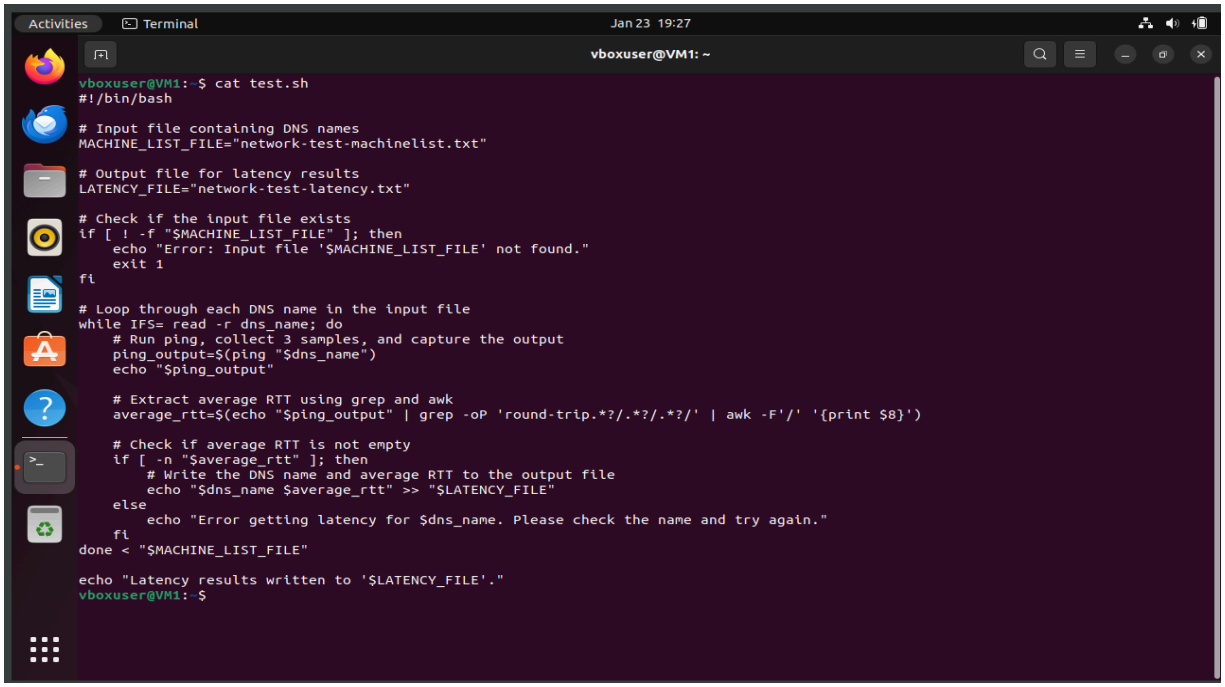
16) tar:
Description: Creates or extracts files from an archive file.
Example Usage:
To create a tar archive: tar -cvf archive.tar file1 file2.
To extract from a tar archive: tar -xvf archive.tar



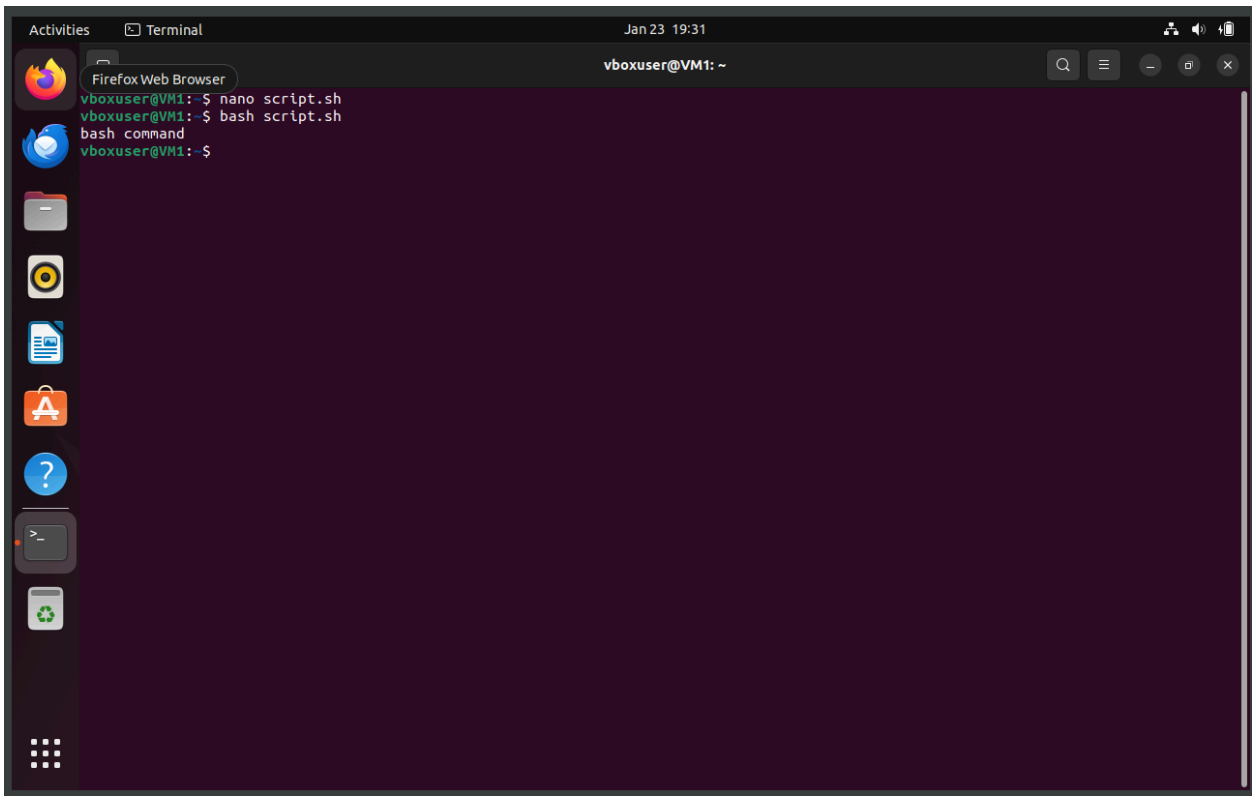
17) rm (Remove):
Description: Removes (deletes) files or directories.
Example Usage:
To remove a file: rm filename.txt
To remove a directory and its contents: rm -r directory_name



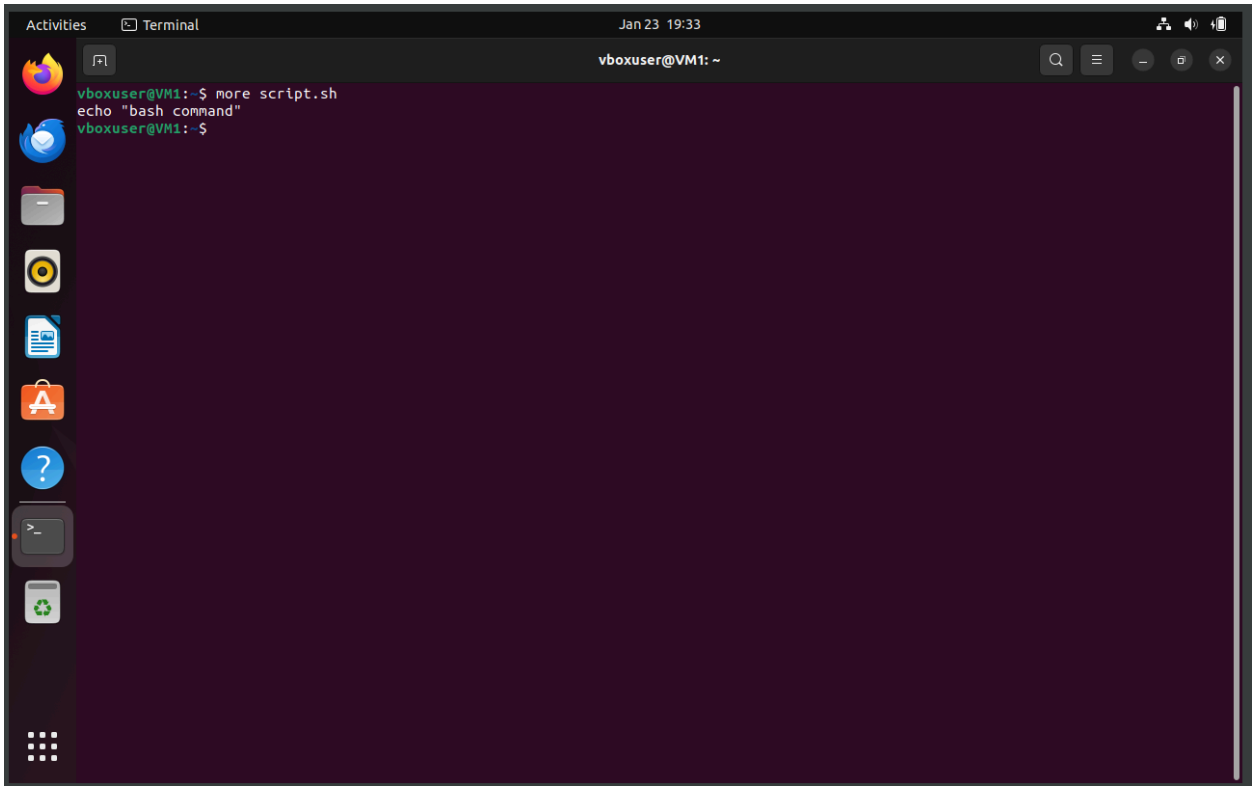
18) cat (Concatenate):
Description: Displays the content of a file or concatenates files.
Example Usage: cat filename.txt



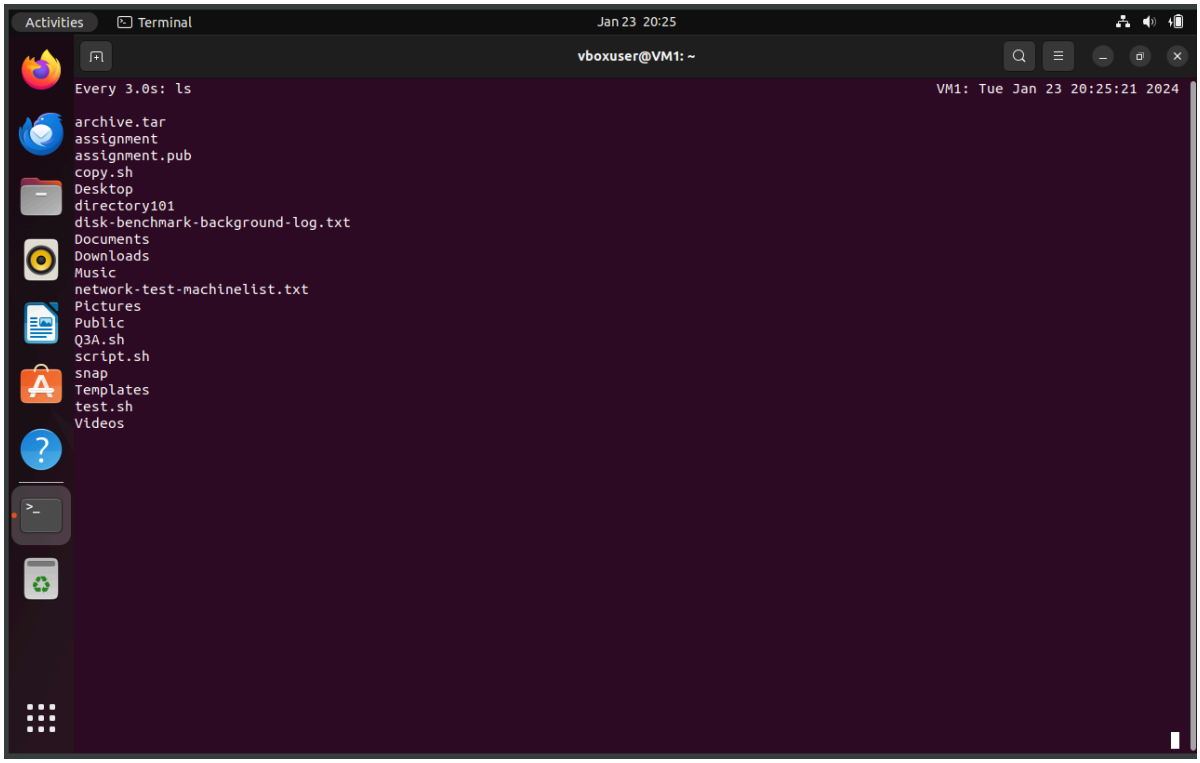
19) bash:
Description: The GNU Bourne-Again SHell, a command processor that typically runs in a text window.
Example Usage: Running a script: bash script.sh



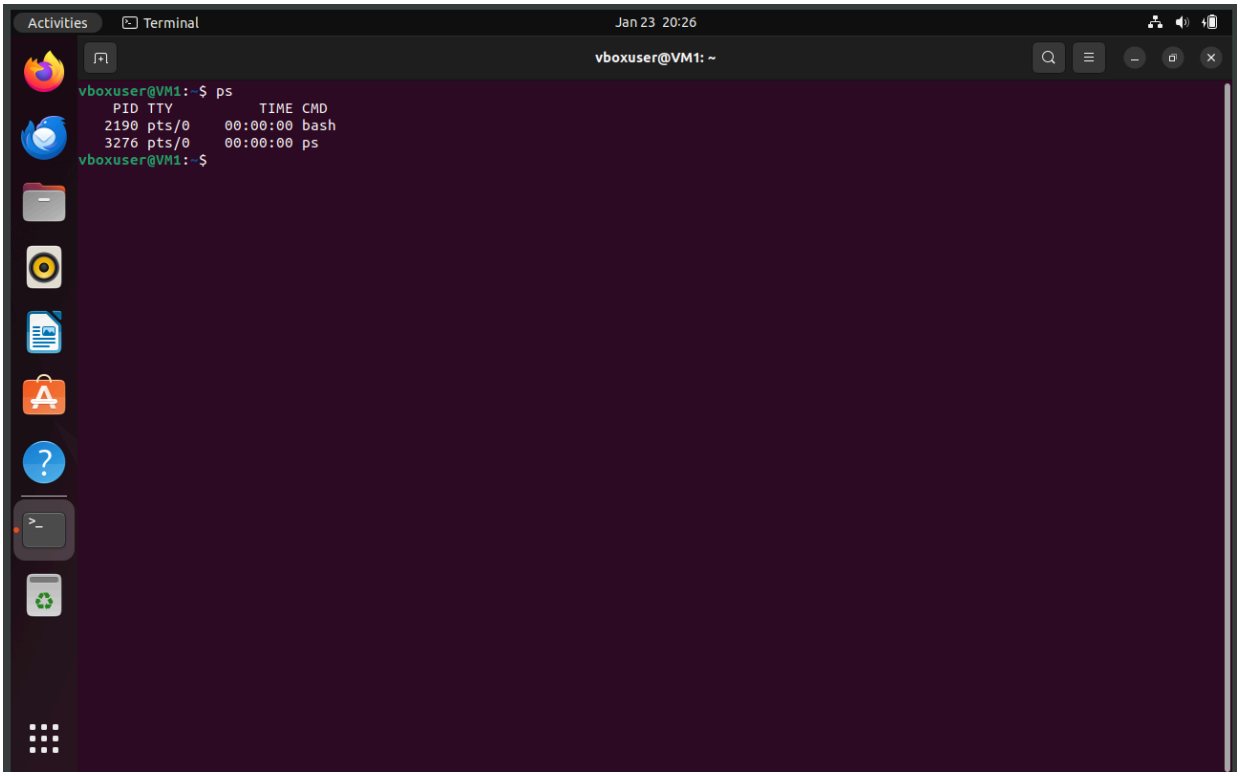
20) more:
Description: Displays the contents of a file one screen at a time.
Example Usage: more filename.txt



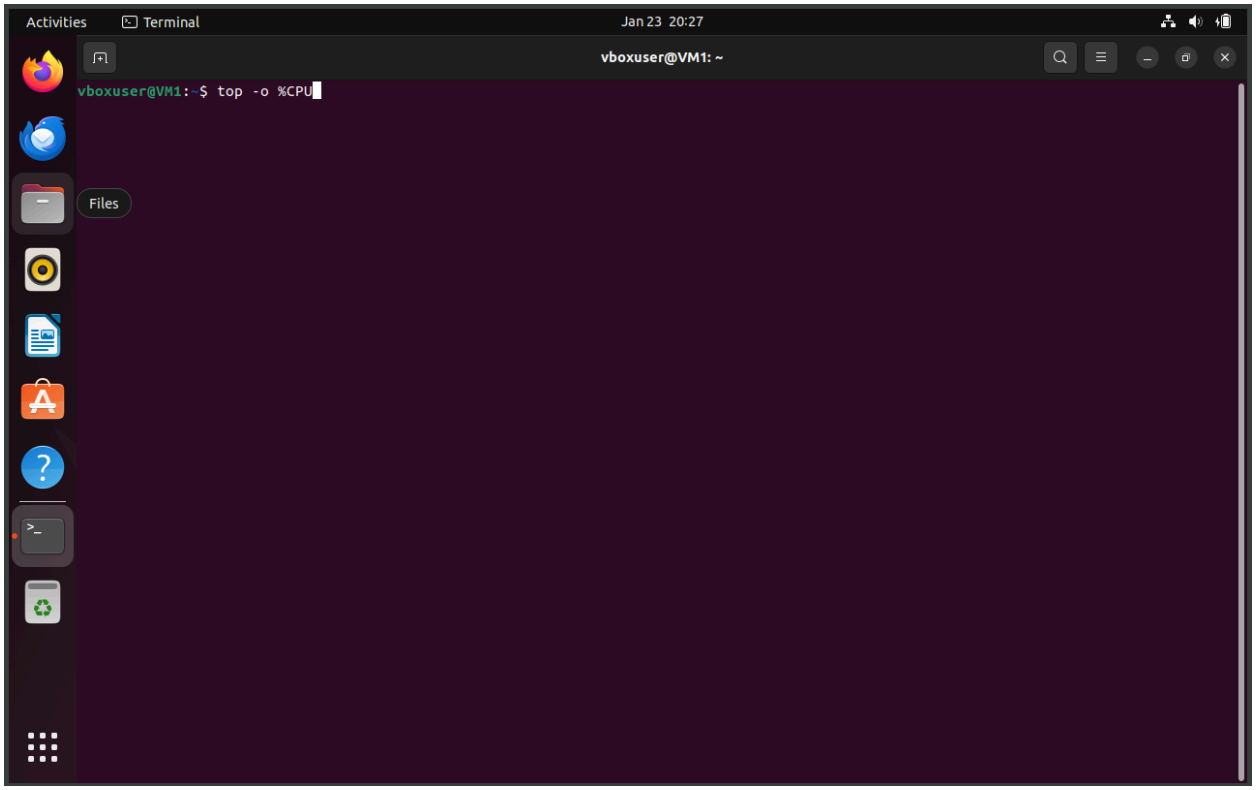
21) watch:
Description: Periodically executes a command and displays the output.
Example Usage: watch -n 5 command_to_watch

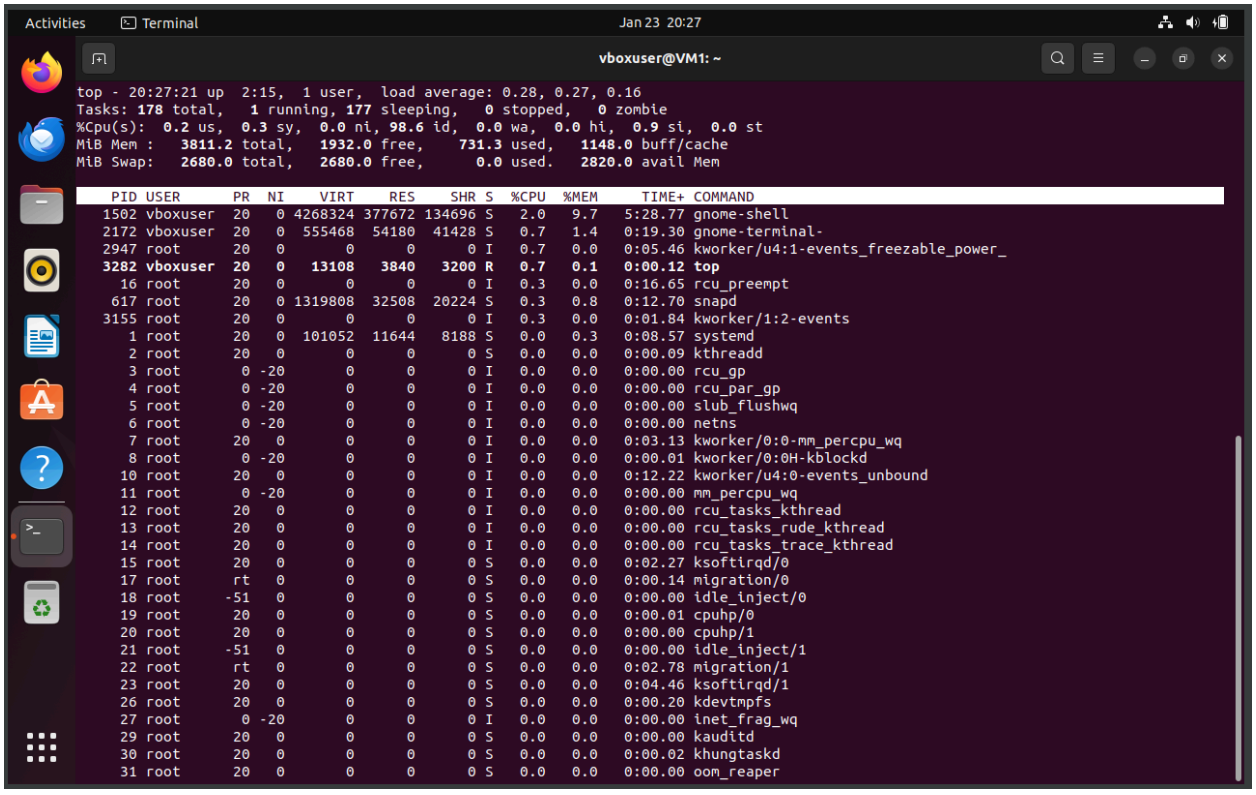


22) ps (Process Status):
Description: Displays information about processes running on the system.
Example Usage: ps aux



23) top:
Description: Displays real-time system information, including a list of processes.
Example Usage: top

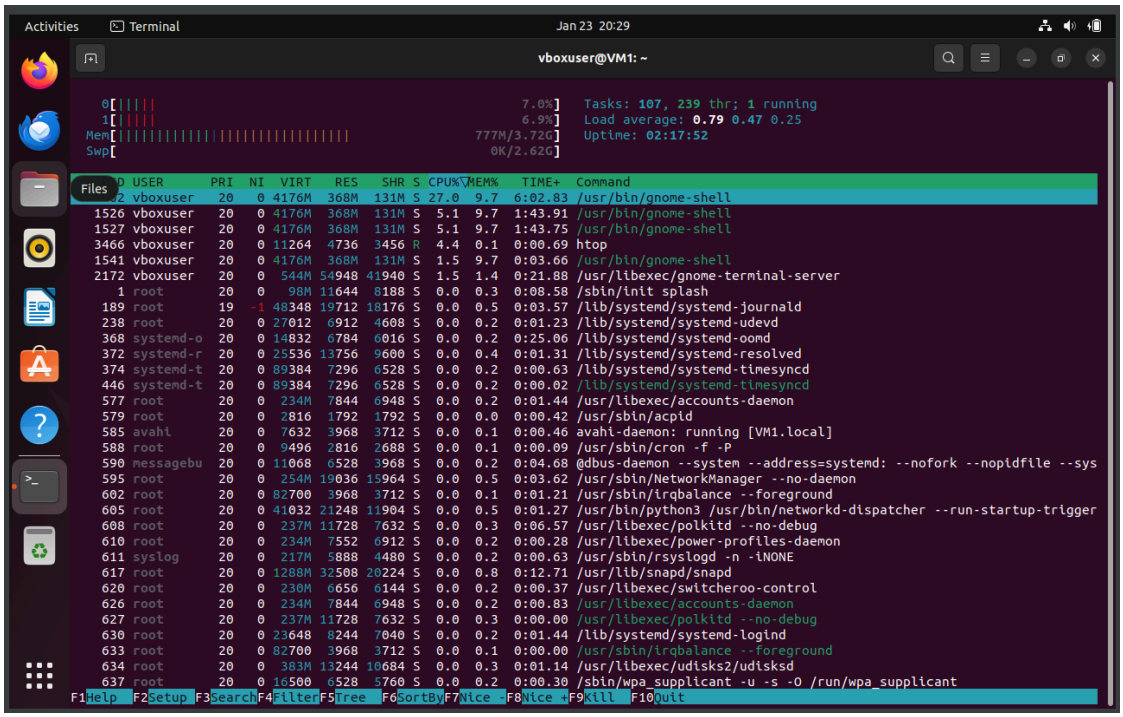

Activities
Terminal
Jan 23 20:27
vboxuser@VM1: ~
vboxuser@VM1:~$ top -o %CPU
Files

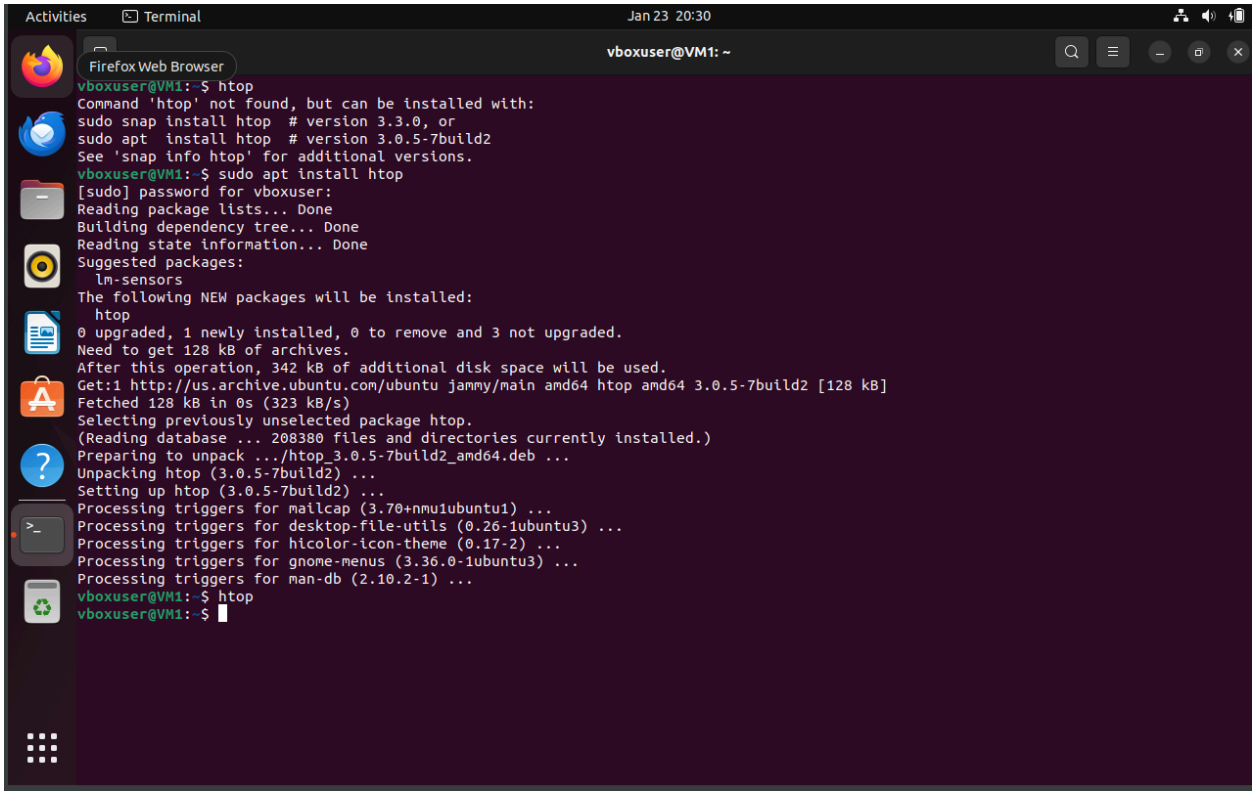

Activities
Terminal
Jan 23 20:27
vboxuser@VM1: ~
top - 20:27:21 up 2:15, 1 user, load average: 0.28, 0.27, 0.16
Tasks: 178 total, 1 running, 177 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.3 sy, 0.0 ni, 98.6 id, 0.0 wa, 0.0 hi, 0.9 si, 0.0 st
MiB Mem : 3811.2 total, 1932.0 free, 731.3 used, 1148.0 buff/cache
MiB Swap: 2680.0 total, 2680.0 free, 0.0 used. 2820.0 avail Mem
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
1502 vboxuser 20 0 4268324 377672 134696 S 2.0 9.7 5:28.77 gnome-shell
2172 vboxuser 20 0 555468 54180 41428 S 0.7 1.4 0:19.30 gnome-terminal-
2947 root 20 0 0 0 0 I 0.7 0.0 0:05.46 kworker/u4:1-events_freezable_power_
3282 vboxuser 20 0 13108 3840 3200 R 0.7 0.1 0:00.12 top
16 root 20 0 0 0 0 I 0.3 0.0 0:16.65 rcu_preempt
617 root 20 0 1319808 32508 20224 S 0.3 0.8 0:12.70 snapd
3155 root 20 0 0 0 0 I 0.3 0.0 0:01.84 kworker/1:2-events
1 root 20 0 101052 11644 8188 S 0.0 0.3 0:08.57 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.09 kthreadd
3 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_gp
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_par_gp
5 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 slub_flushwq
6 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 netns
7 root 20 0 0 0 0 I 0.0 0.0 0:03.13 kworker/0:0-mm_percpu_wq
8 root 0 -20 0 0 0 I 0.0 0.0 0:00.01 kworker/0:0H-kblockd
10 root 20 0 0 0 0 I 0.0 0.0 0:12.22 kworker/u4:0-events_unbound
11 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 mm_percpu_wq
12 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_kthread
13 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_rude_kthread
14 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_trace_kthread
15 root 20 0 0 0 0 S 0.0 0.0 0:02.27 ksoftirqd/0
17 root rt 0 0 0 0 S 0.0 0.0 0:00.14 migration/0
18 root -51 0 0 0 0 S 0.0 0.0 0:00.00 idle_inject/0
19 root 20 0 0 0 0 S 0.0 0.0 0:00.01 cpuhp/0
20 root 20 0 0 0 0 S 0.0 0.0 0:00.00 cpuhp/1
21 root -51 0 0 0 0 S 0.0 0.0 0:00.00 idle_inject/1
22 root rt 0 0 0 0 S 0.0 0.0 0:02.78 migration/1
23 root 20 0 0 0 0 S 0.0 0.0 0:04.46 ksoftirqd/1
26 root 20 0 0 0 0 S 0.0 0.0 0:00.20 kdevtmpfs
27 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 inet_frag_wq
29 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kauditd
30 root 20 0 0 0 0 S 0.0 0.0 0:00.02 khungtaskd
31 root 20 0 0 0 0 S 0.0 0.0 0:00.00 oom_reaper

24) htop:

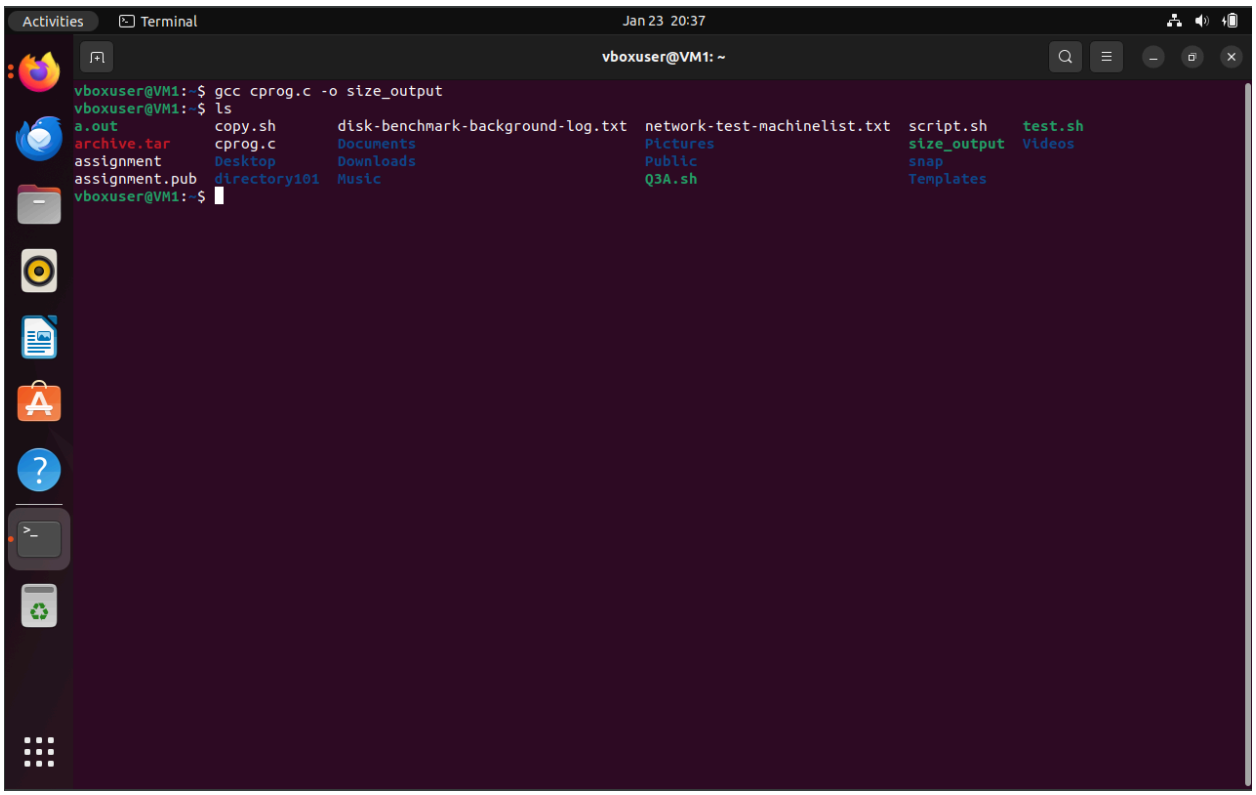
Description: An interactive process viewer and system monitor.

Example Usage: htop

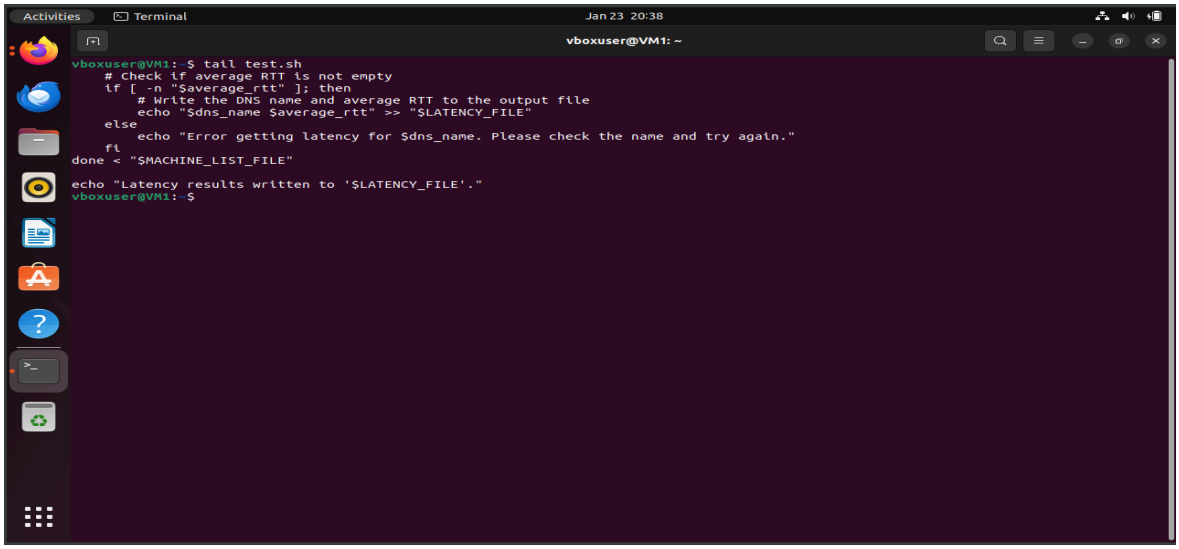




25) gcc (GNU Compiler Collection):
Description: Compiler for various programming languages, primarily C, C++, and Fortran.
Example Usage: gcc -o output_program source_code.c



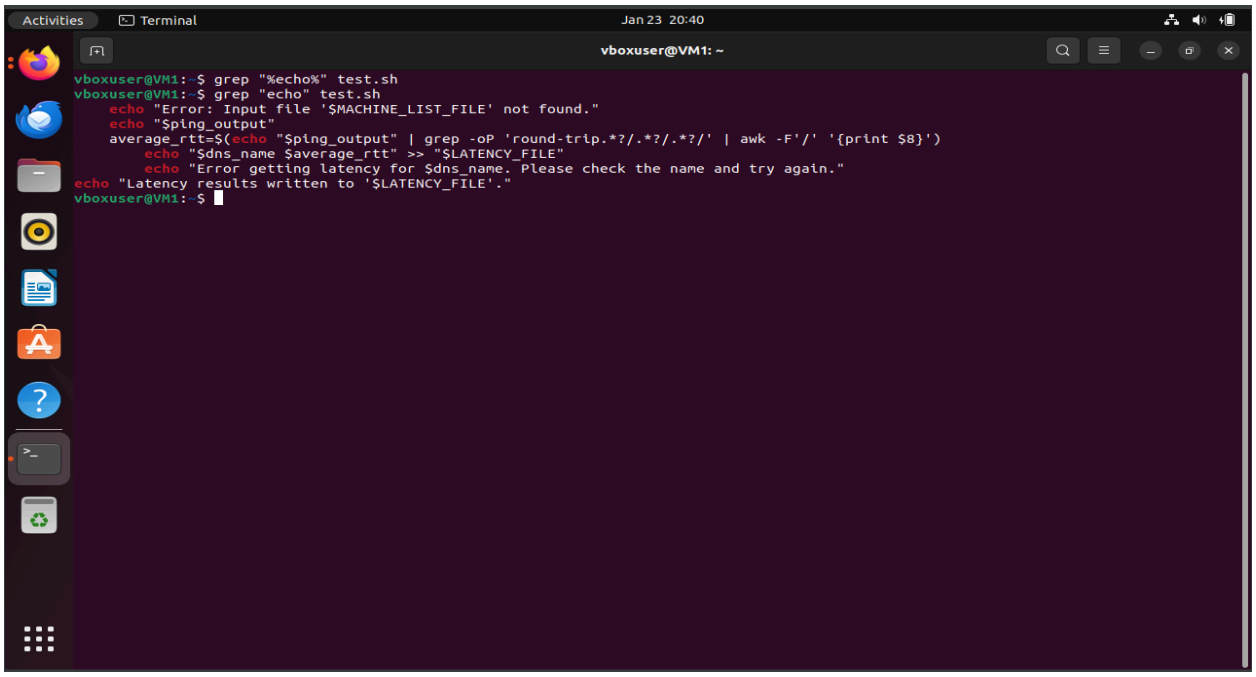
26) tail:
Description: Displays the last part of a file (by default, the last 10 lines).
Example Usage: tail filename.txt



27) grep:

Description: Searches for a specific pattern in files.

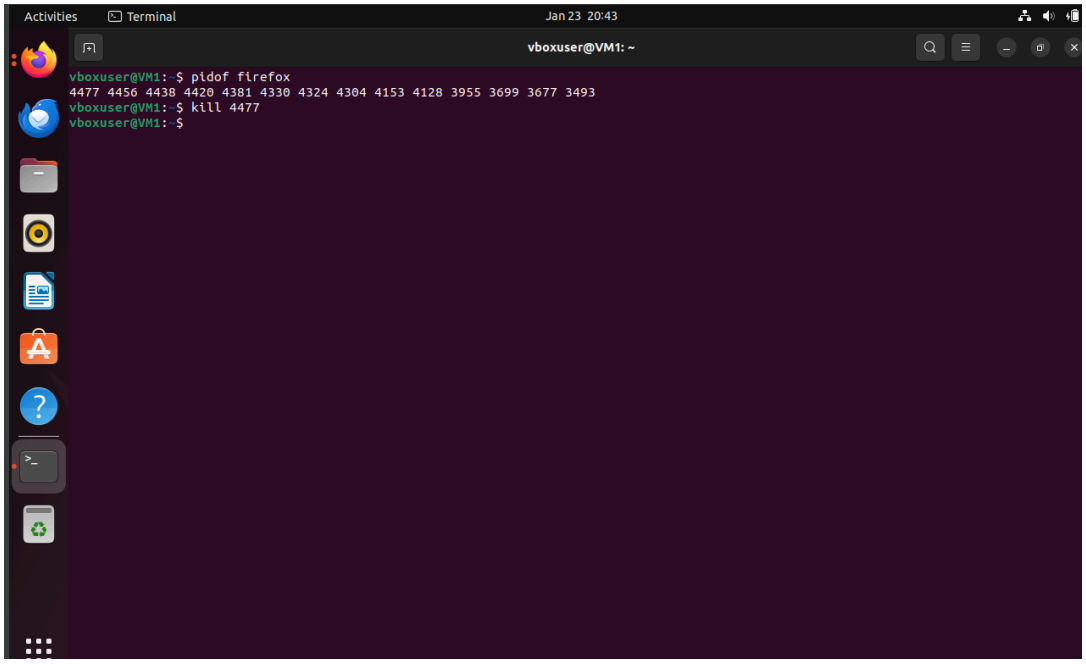
Example Usage: grep "pattern" filename.txt



28) kill:

Description: Sends a signal to terminate a process.

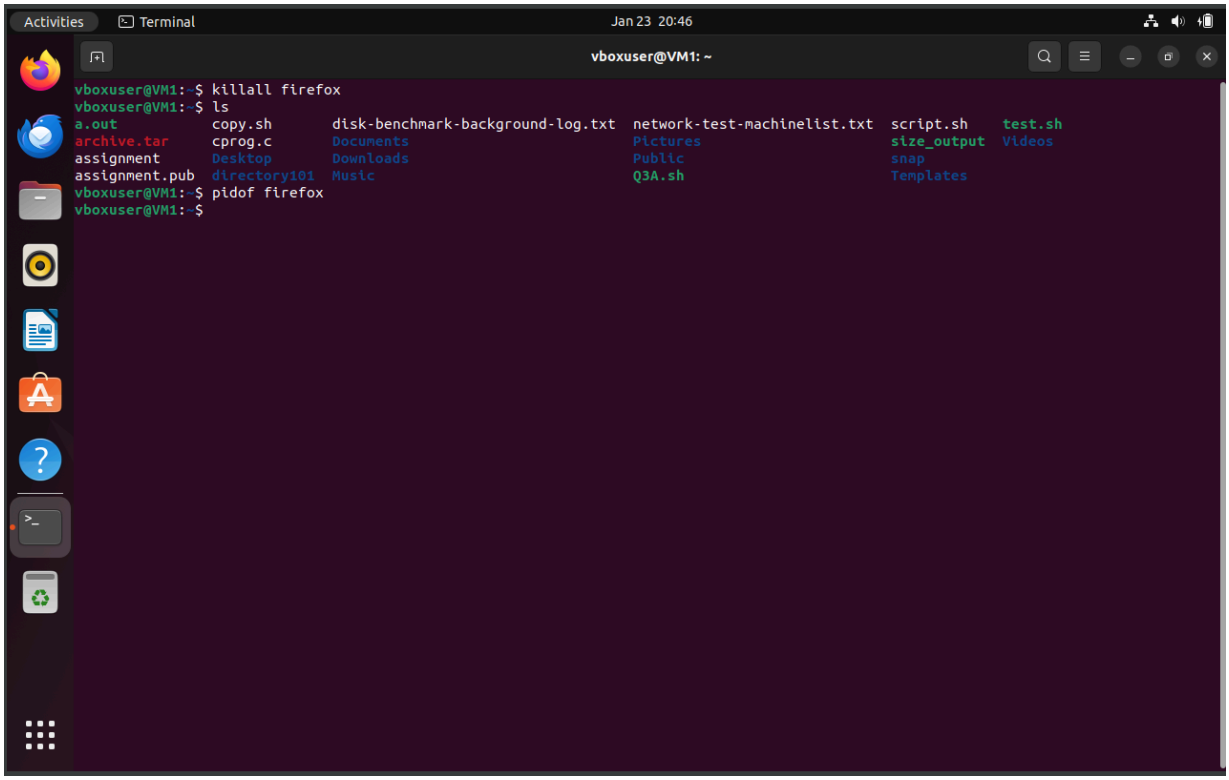
Example Usage: kill process_id



29) killall:

Description: Kills processes by name.

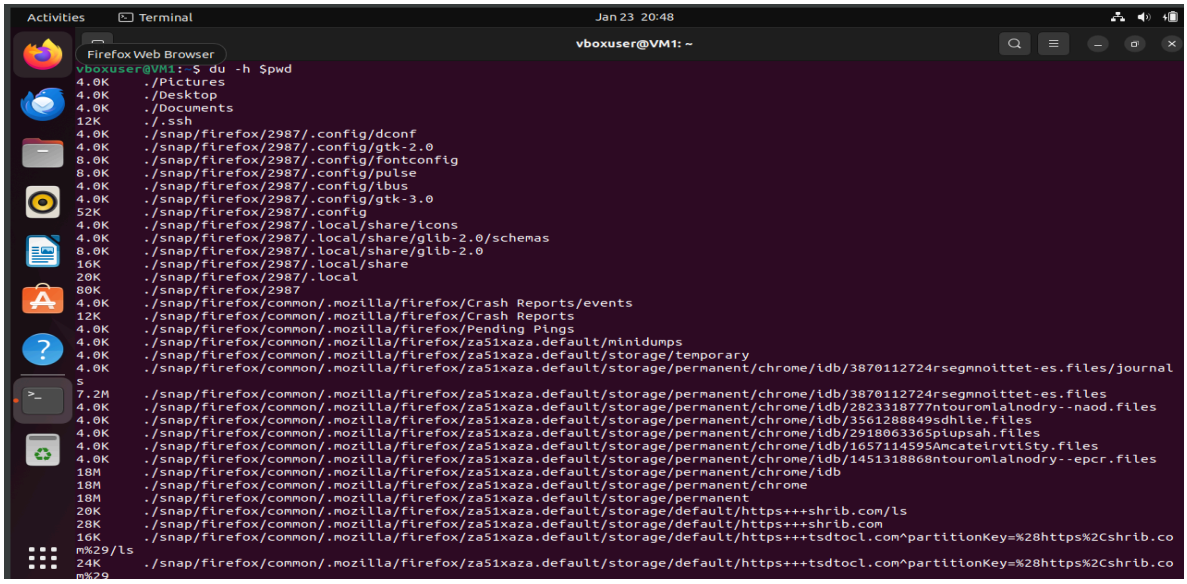
Example Usage: killall process_name



30) du: (Disk Usage)

Description: Shows the disk space used by files and directories.

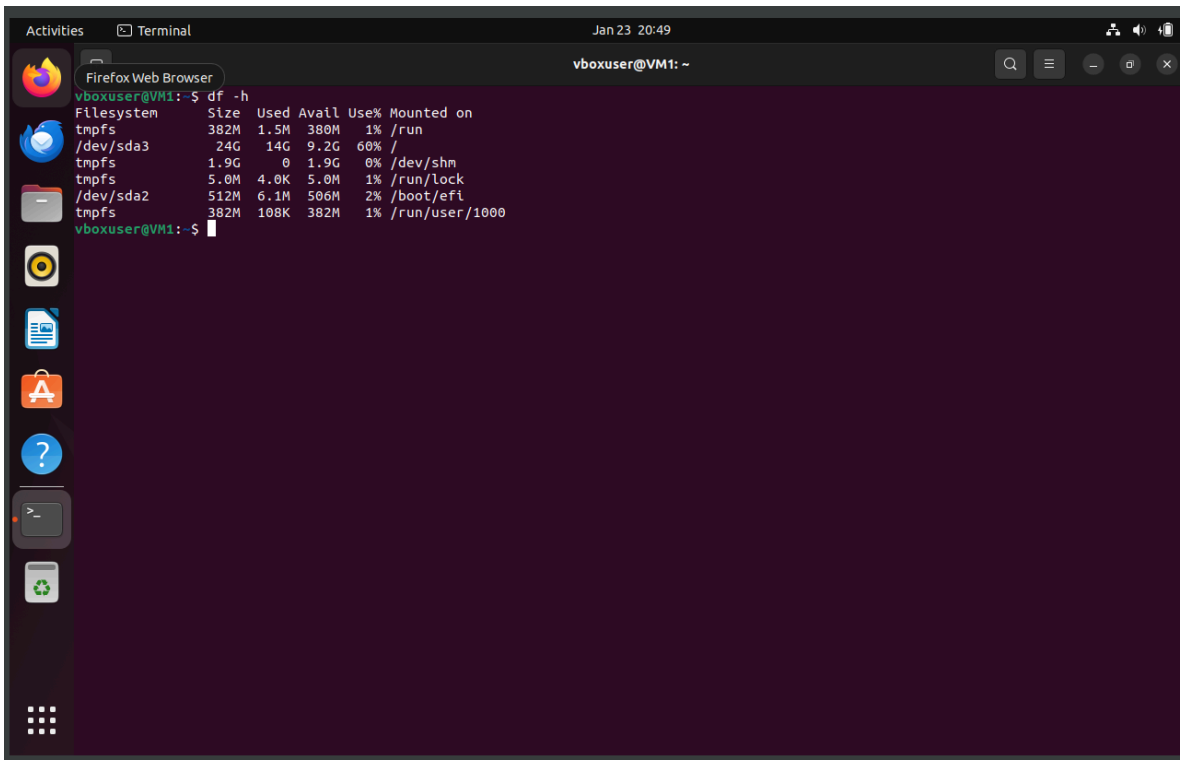
Example Usage: du -h directory_name



31) df (Disk Free):

Description: Displays the amount of disk space available on the file system.

Example Usage: df -h



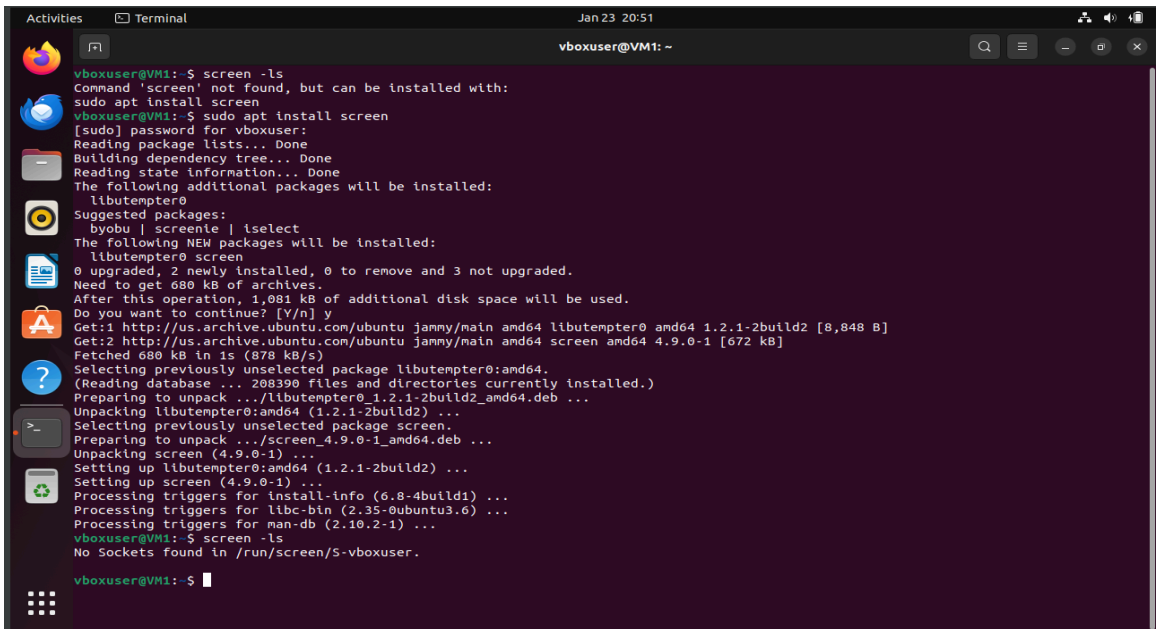
32) screen:

Description: Multiplexes a physical terminal between several processes.

Example Usage:

To start a new screen session: screen

To detach from a screen session: Press Ctrl-a followed by d



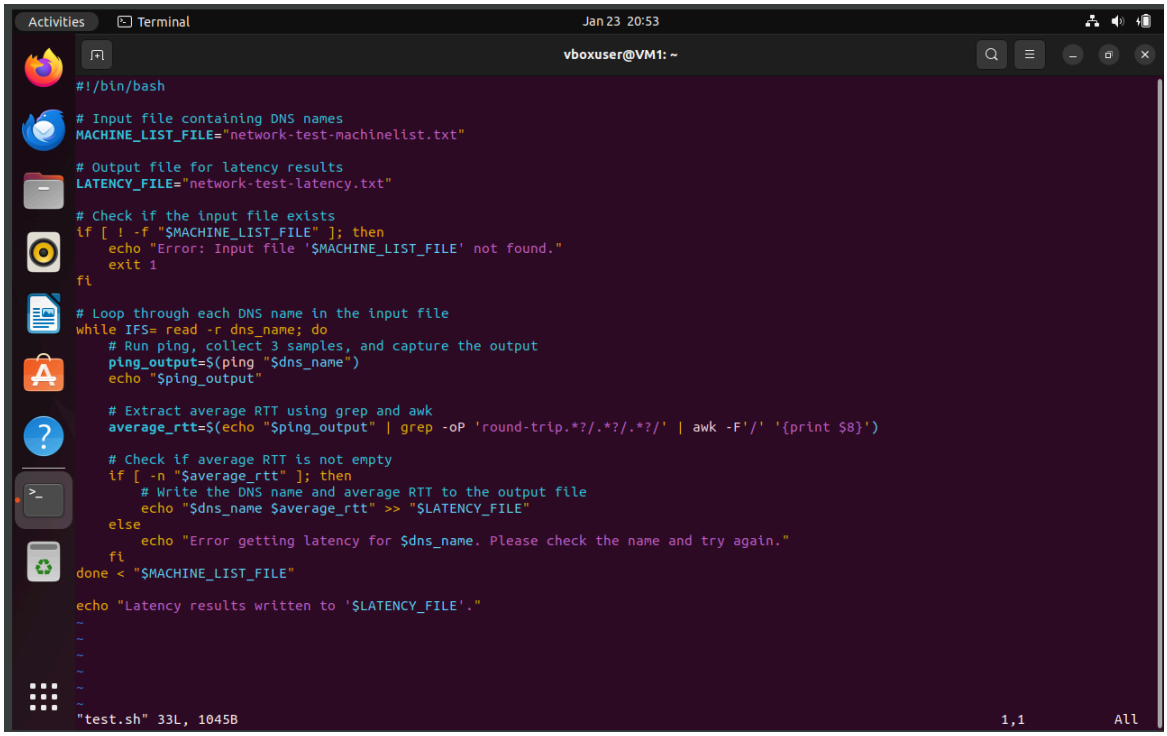
33) vim:

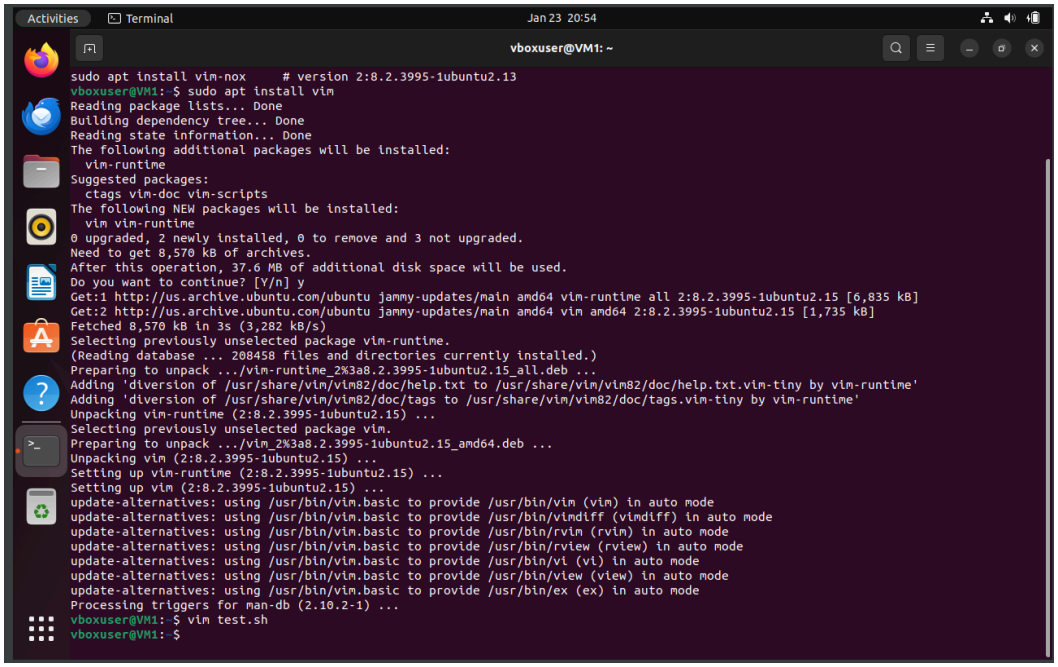
Description: A highly configurable text editor.

Example Usage: vim filename.txt

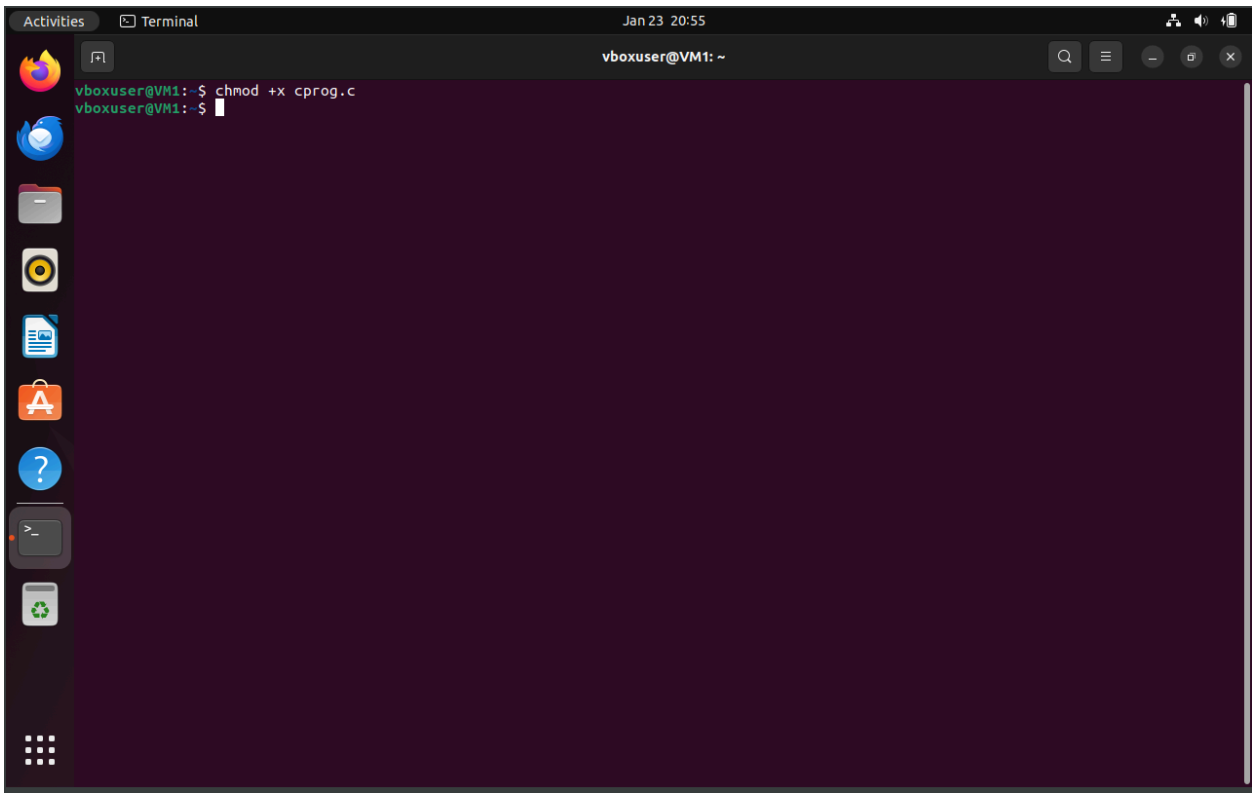
To enter editing mode, press i.

To save and exit, press Esc, then type :wq and press Enter.

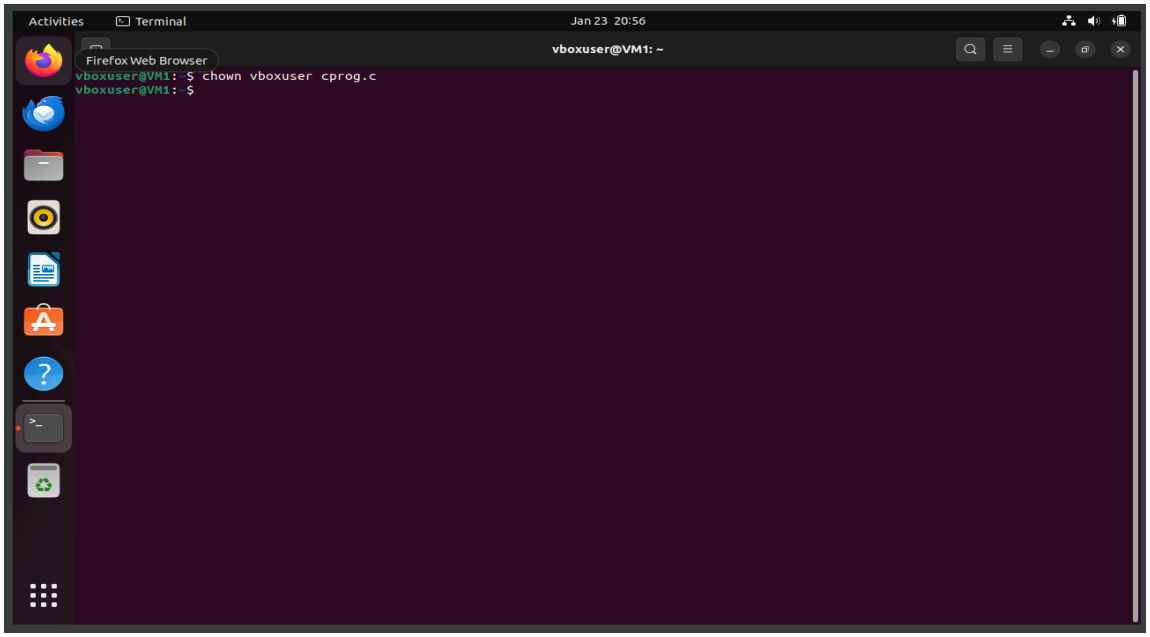




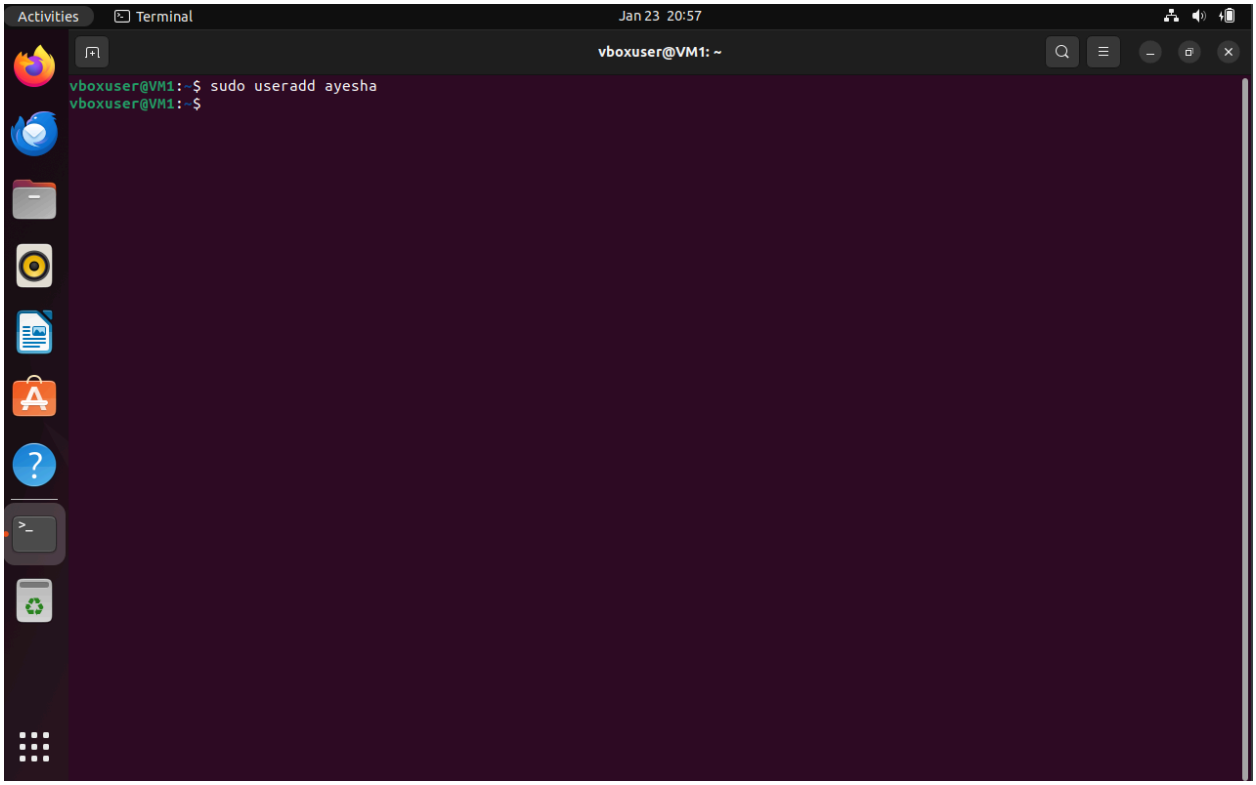
34) chmod(Change Mode):
Description: Changes the permissions of a file or directory.
Example Usage: chmod +x script.sh (grants execute permission to a script)



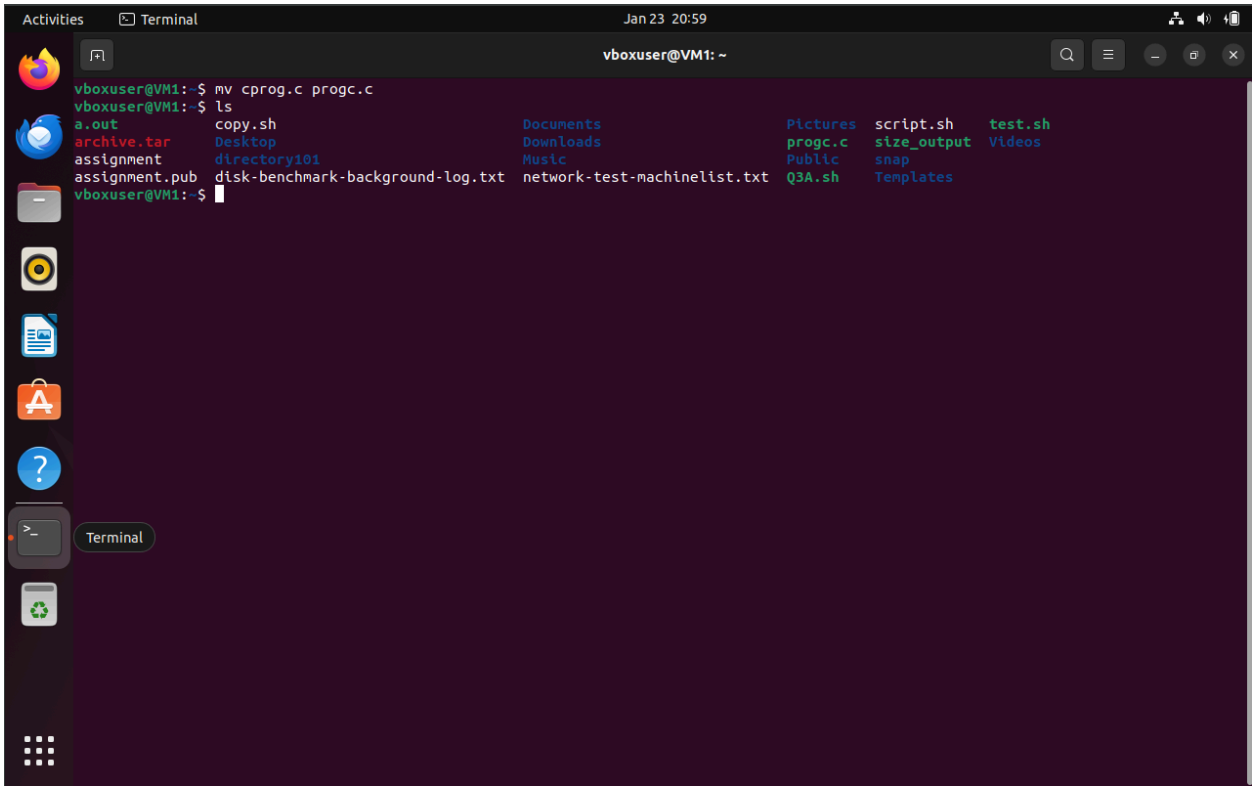
35) chown(Change Owner):
Description: Changes the owner of a file or directory.
Example Usage: chown new_owner:new_group filename.txt



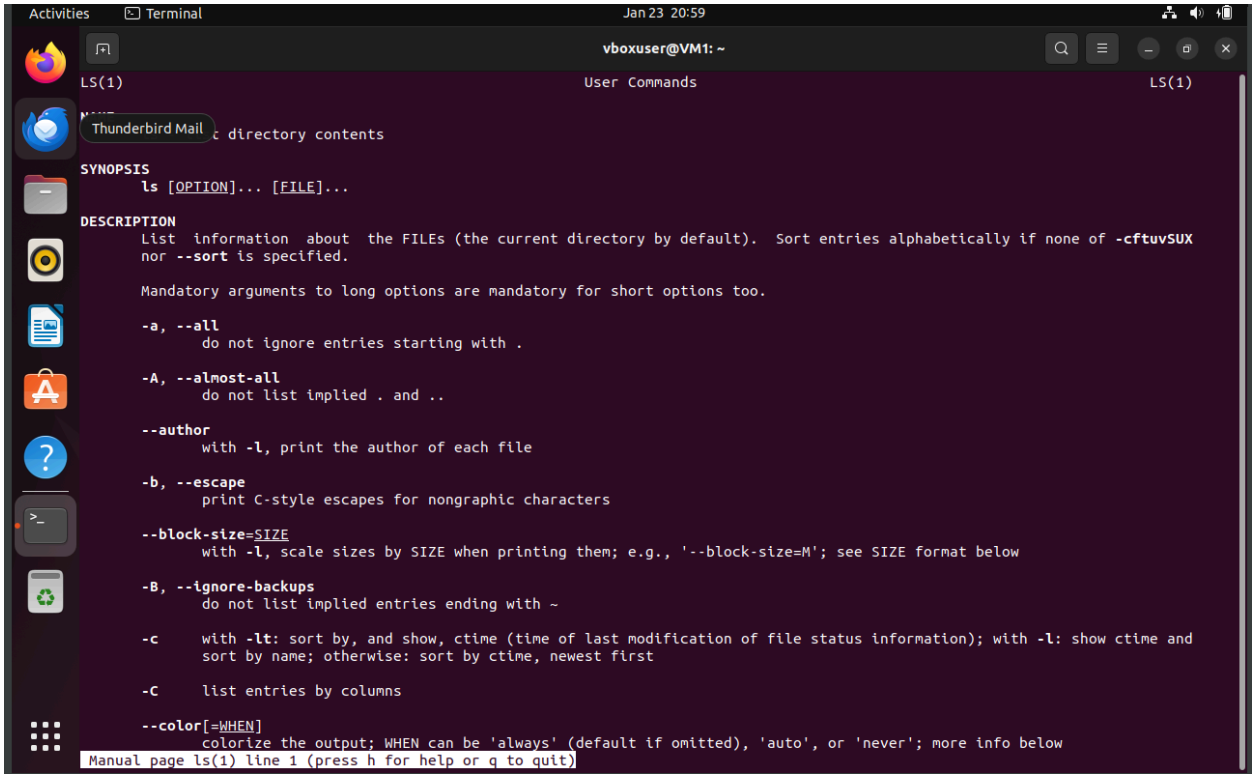
36) useradd:
Description: Adds a new user to the system.
Example Usage: sudo useradd new_username



37) mv (Move):
Description: Moves or renames files and directories.
Example Usage: mv old_filename new_filename



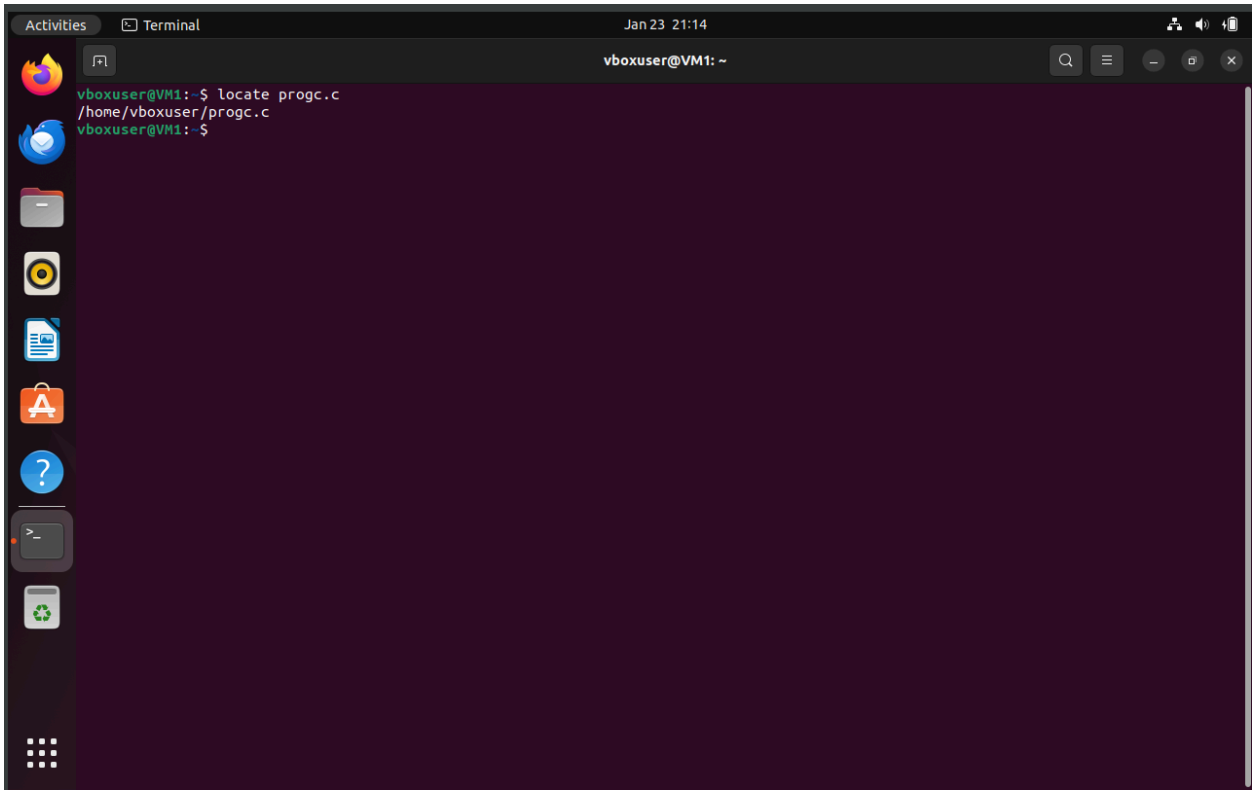
38) man (Manual):
Description: Displays the manual page for a command.
Example Usage: man ls



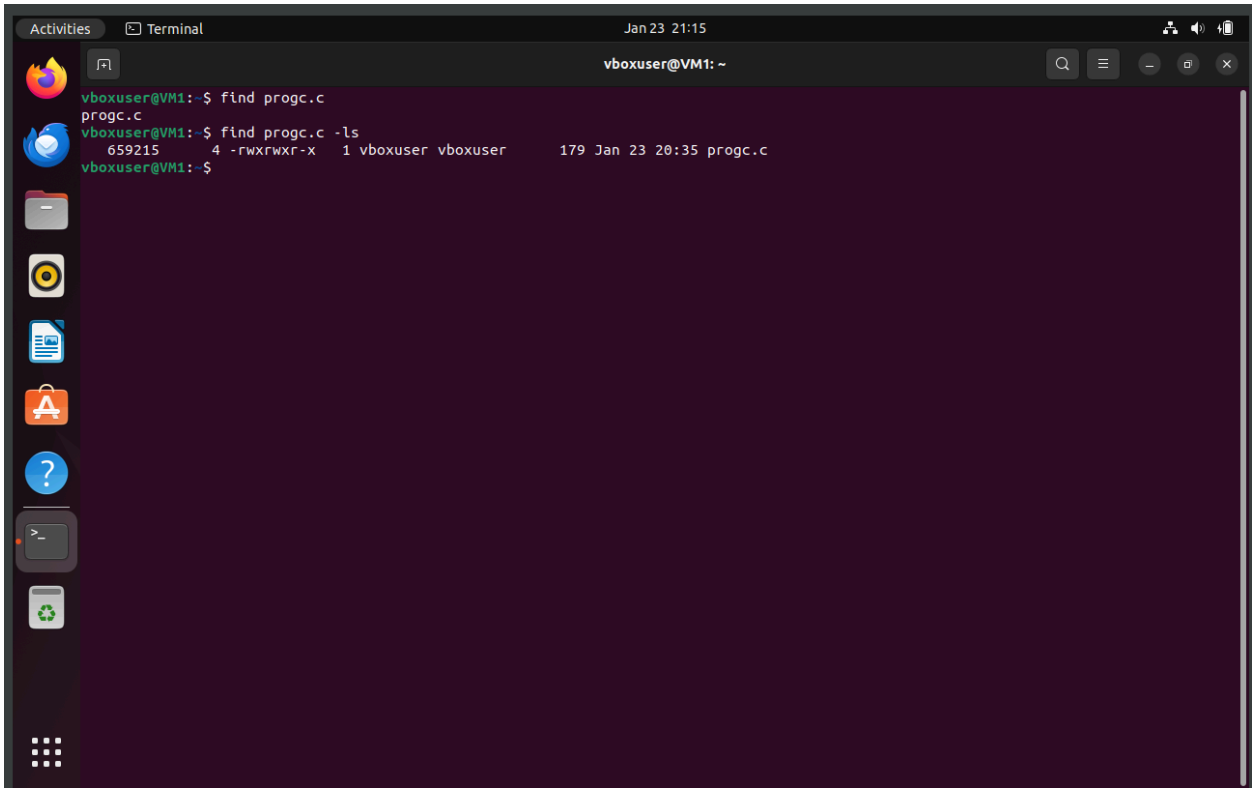
39) locate:

Description: Finds the location of files and directories.E

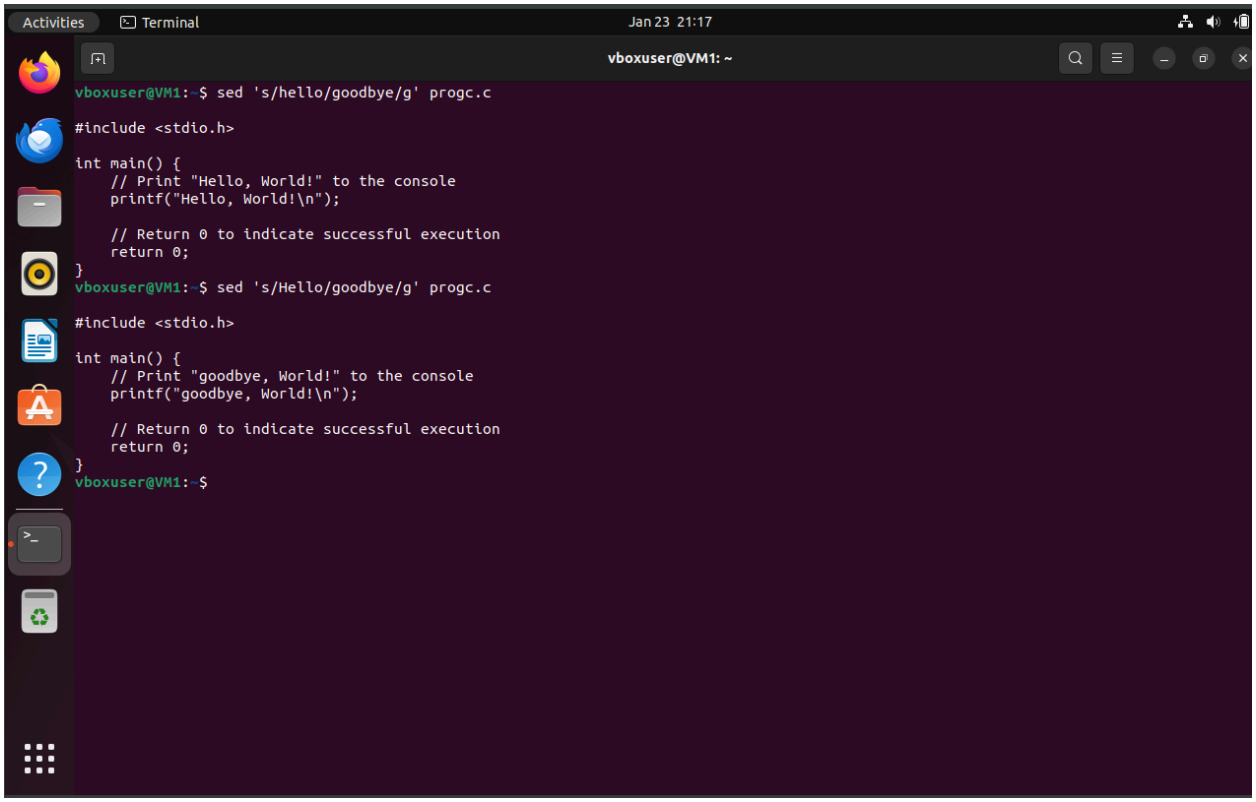
Example Usage: locate filename



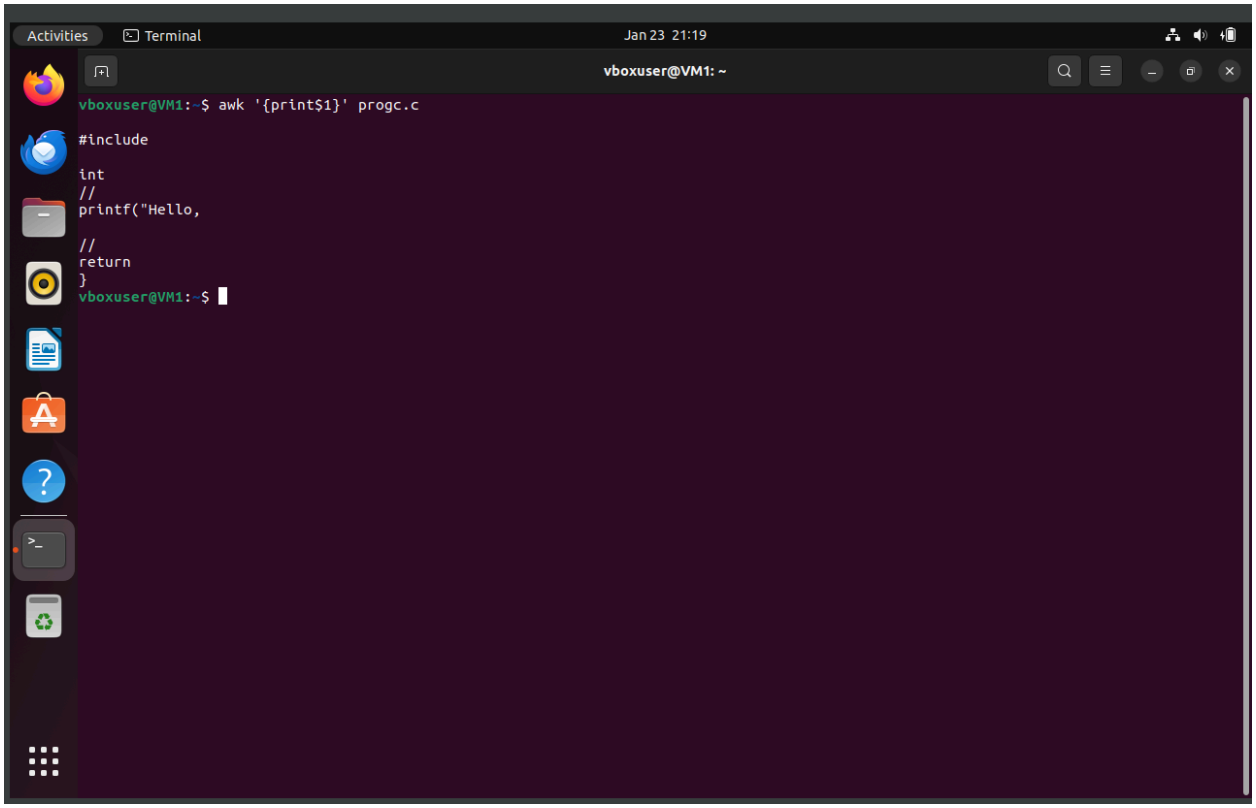
40) find:
Description: Searches for files and directories in a directory hierarchy.
Example Usage: find /path/to/search -name "file_pattern"



41) sed (Stream Editor):
    Description: Edits text by scriptable commands.
    Example Usage: sed 's/old_string/new_string/g' filename.txt



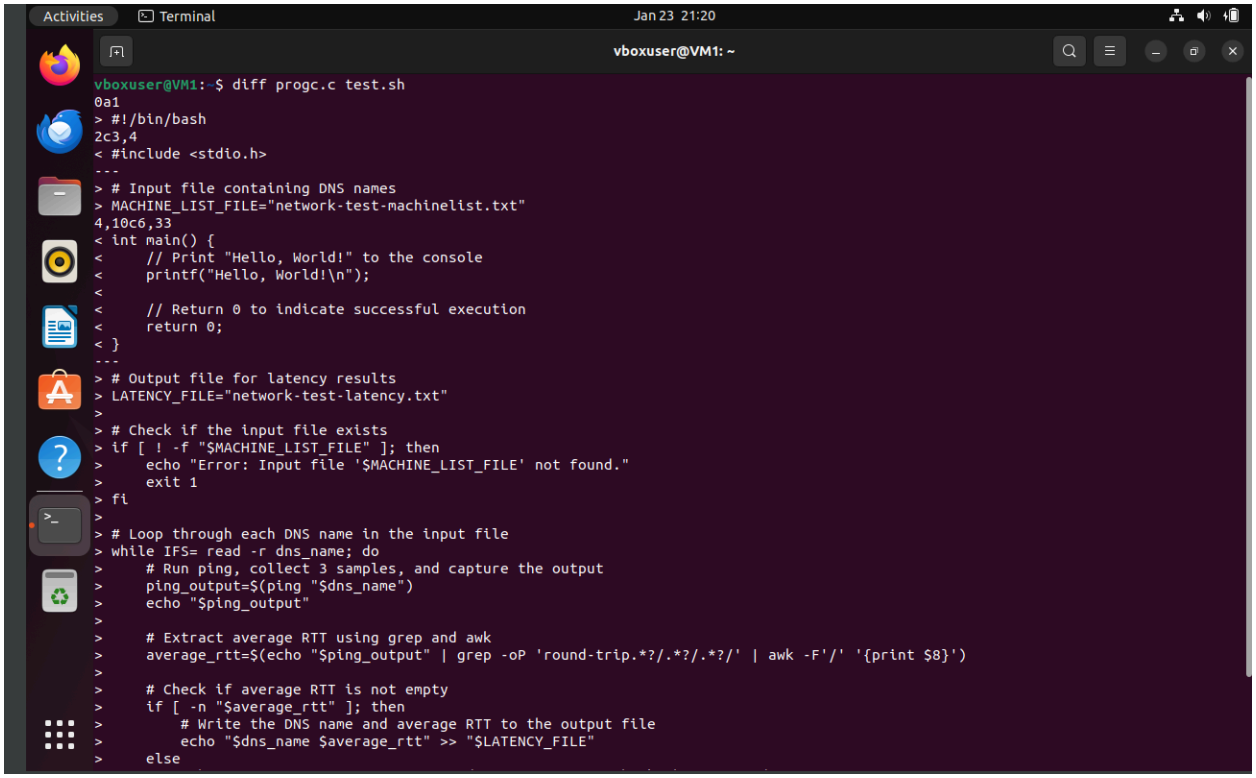
42) awk:
Description: A text processing tool for pattern scanning and processing.
Example Usage: awk '{print $1}' filename.txt (prints the first column of a file)



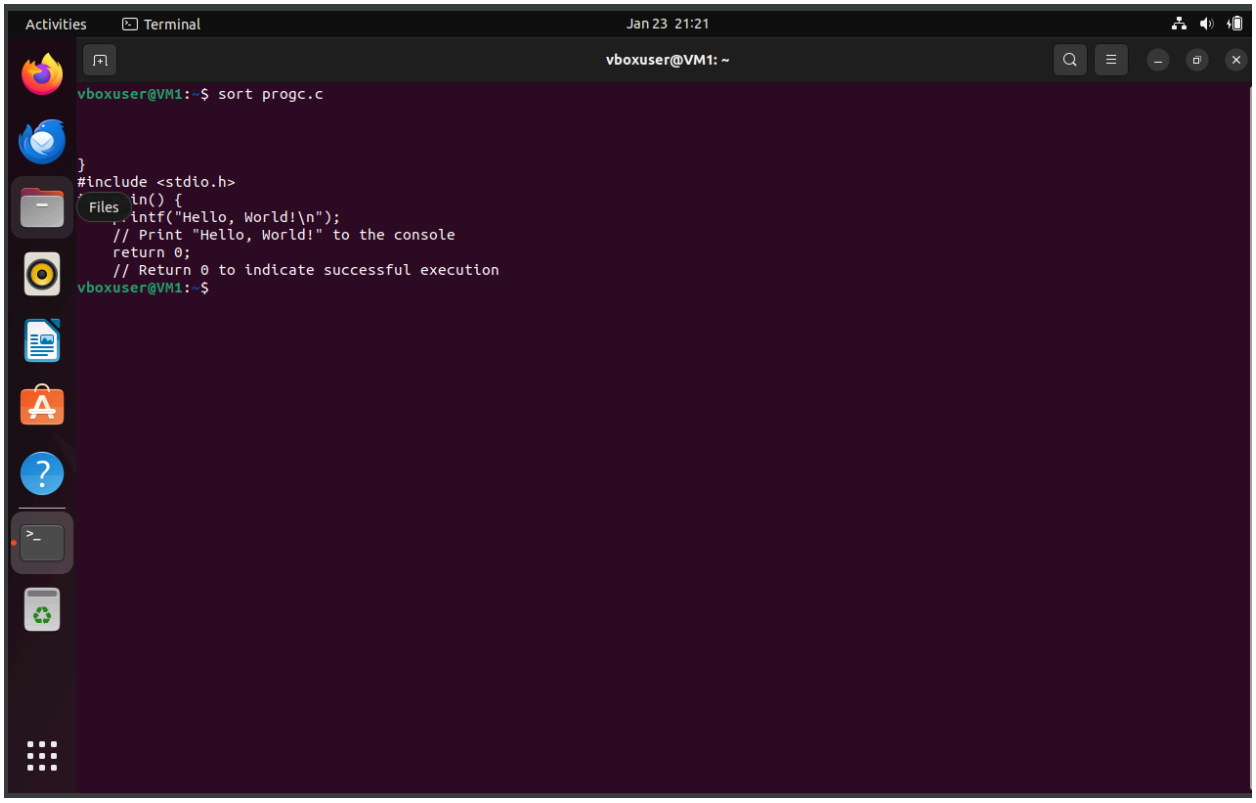
43) diff:

Description: Compares files line by line.

Example Usage: diff file1.txt file2.txt



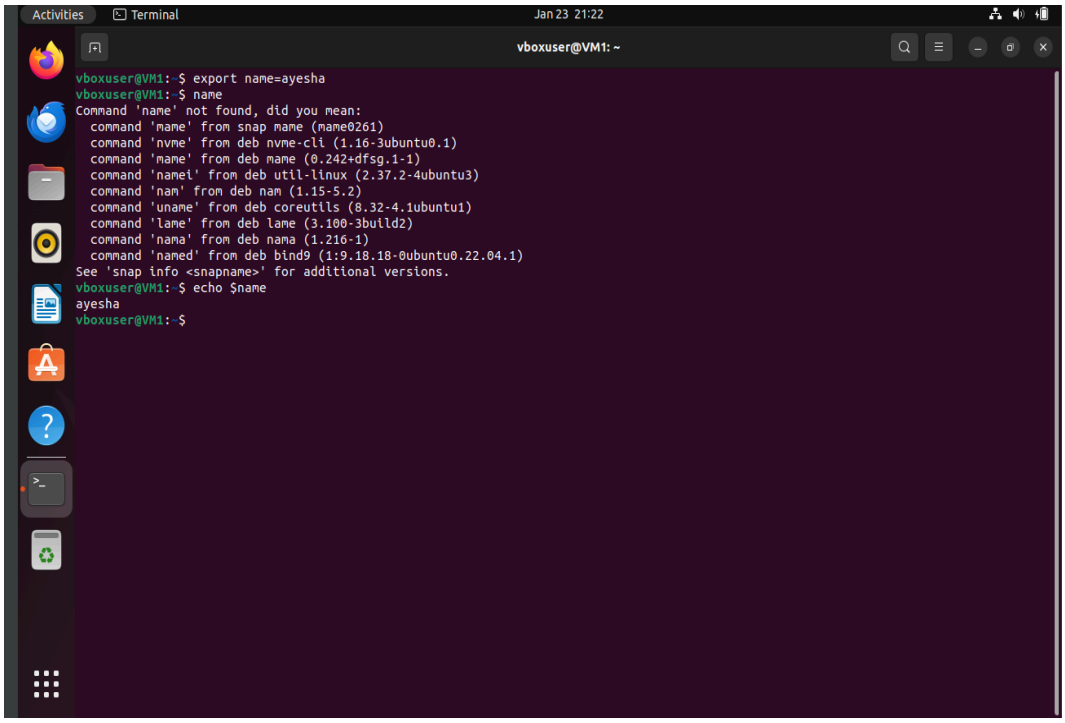
44) sort:
Description: Sorts lines of text files.
Example Usage: sort filename.txt



45) export:

Description: Sets environment variables.

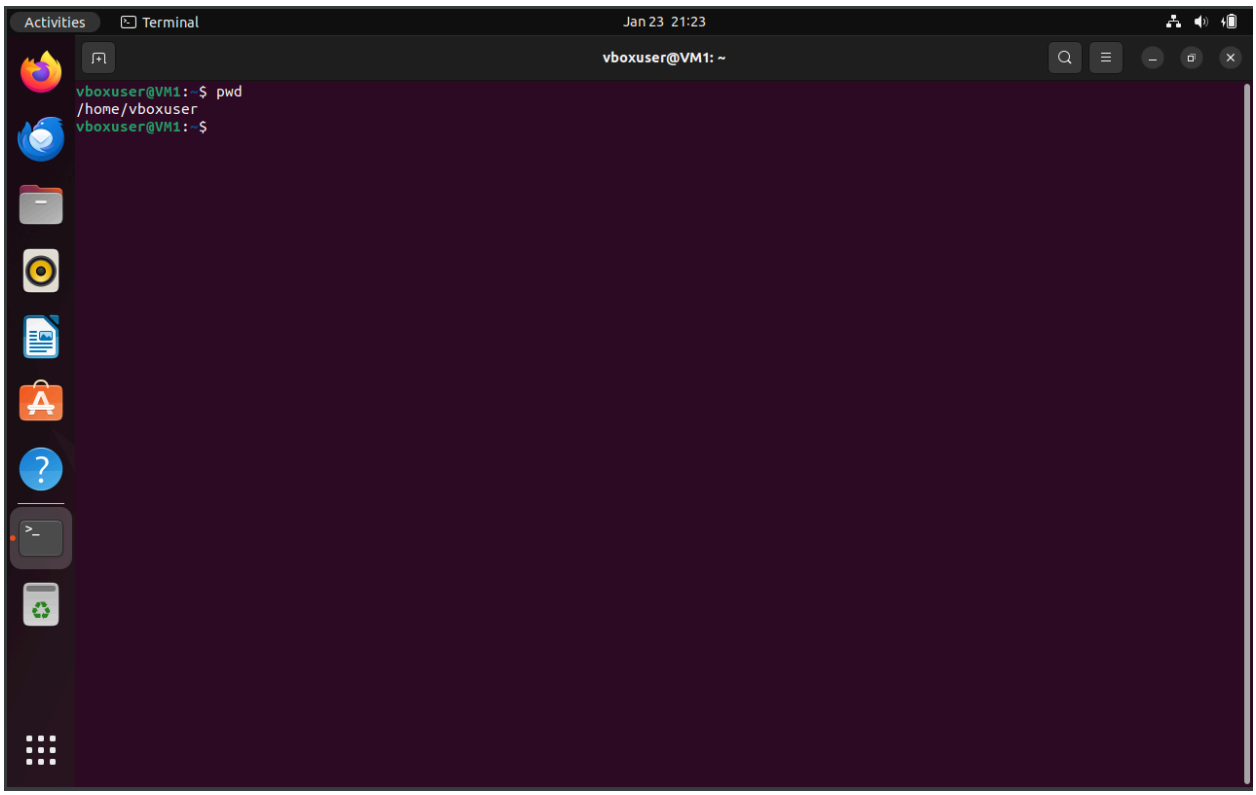
Example Usage: export VARIABLE_NAME=value



46) pwd:

Description: Prints the current working directory.

Example Usage: pwd



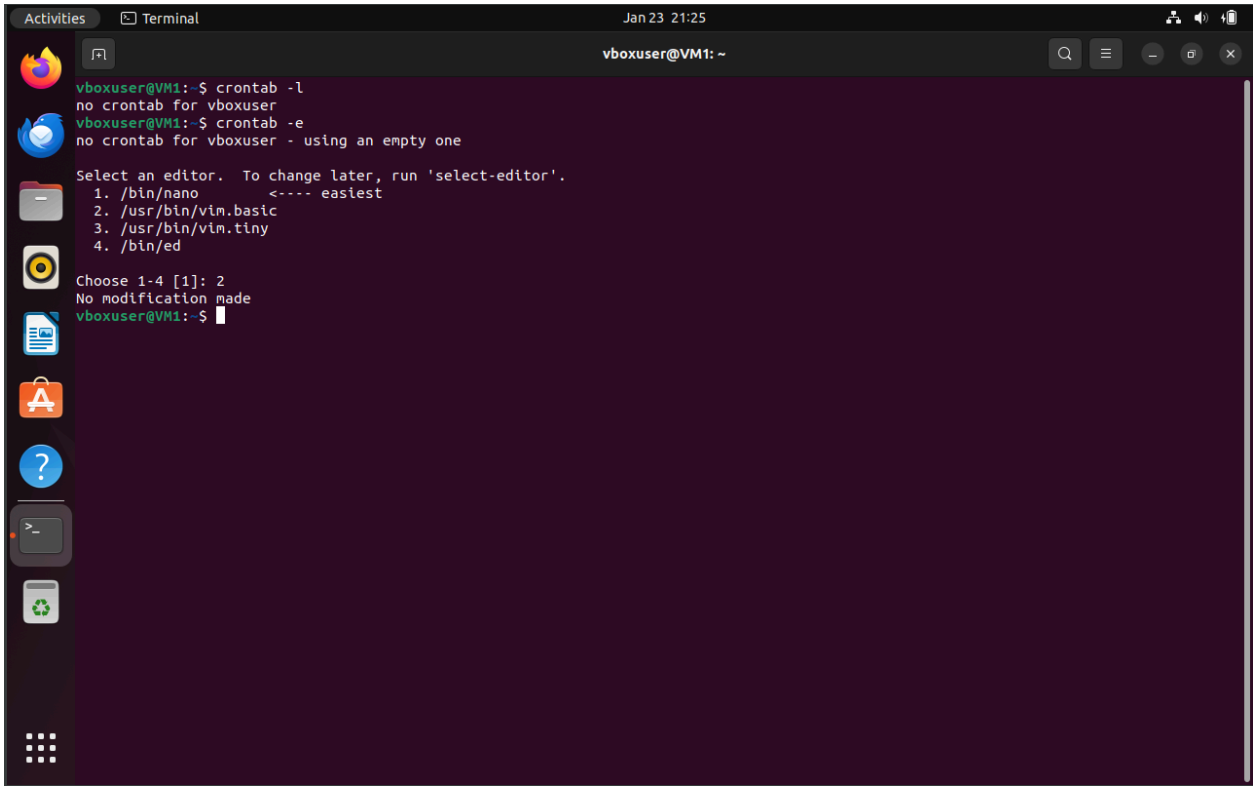
47) crontab

Description: Schedules tasks to run periodically.

Example Usage:

To edit the crontab for the current user: crontab -e

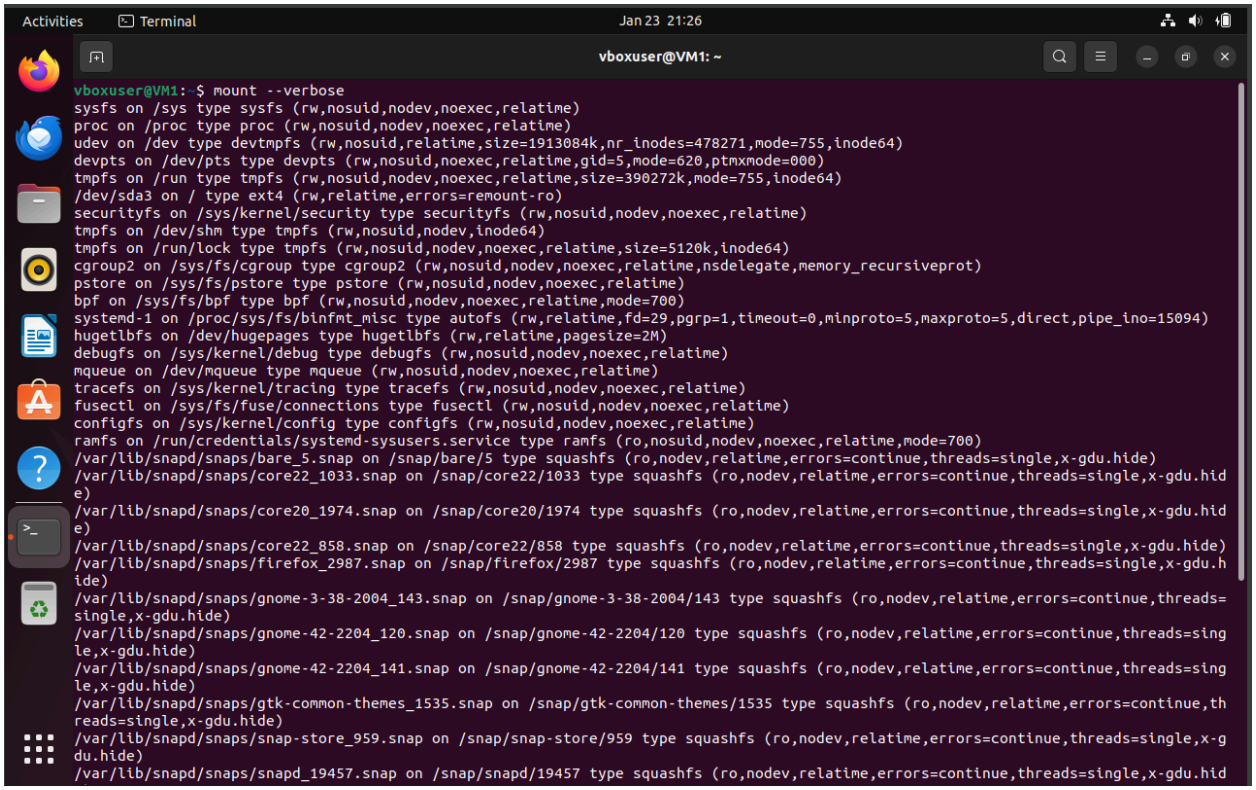
To view the crontab for the current user: crontab -l



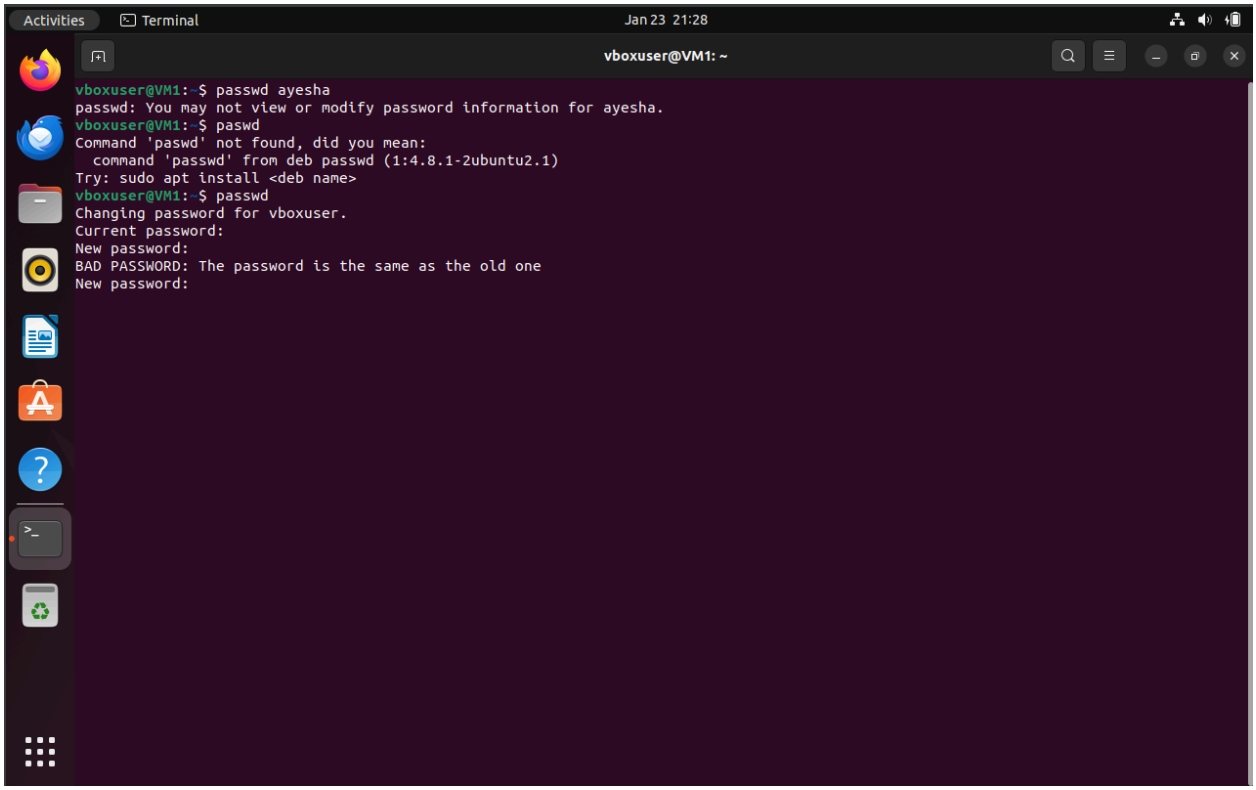
48) mount:

Description: Mounts a file system to the directory tree.

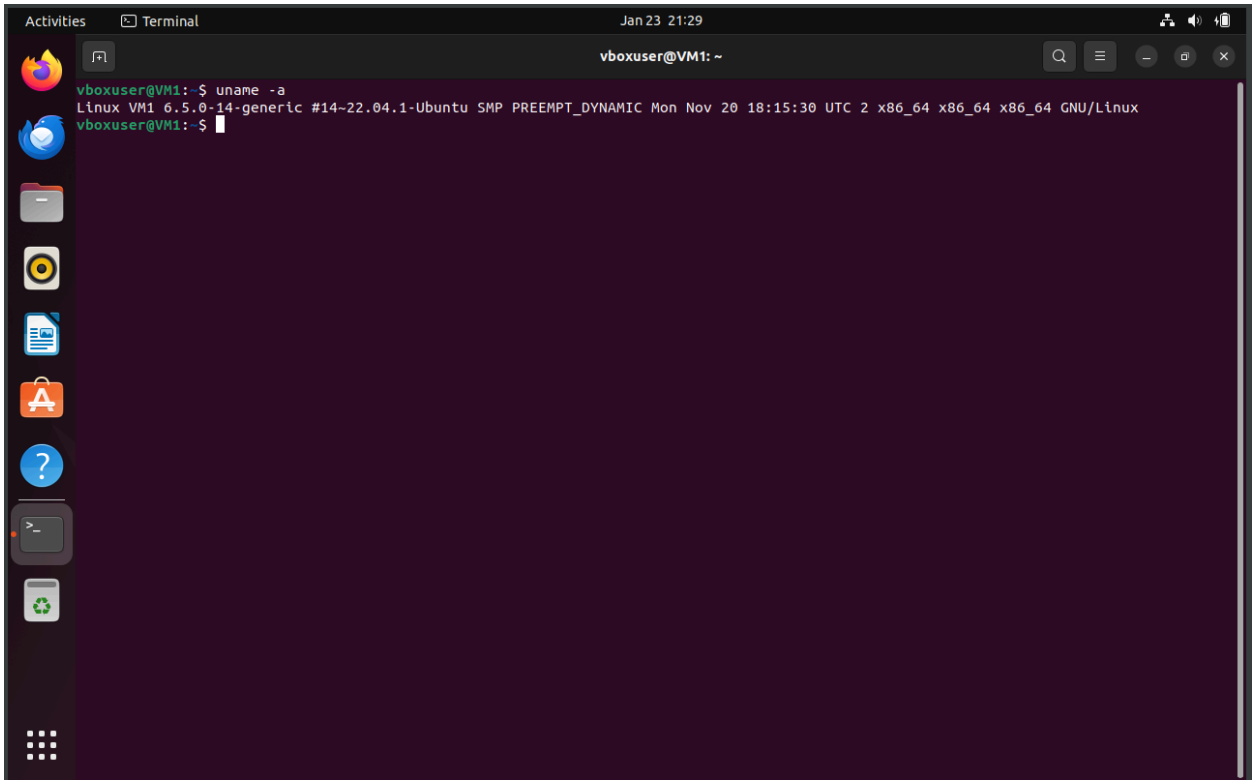
Example Usage: sudo mount /dev/sdX1 /mnt



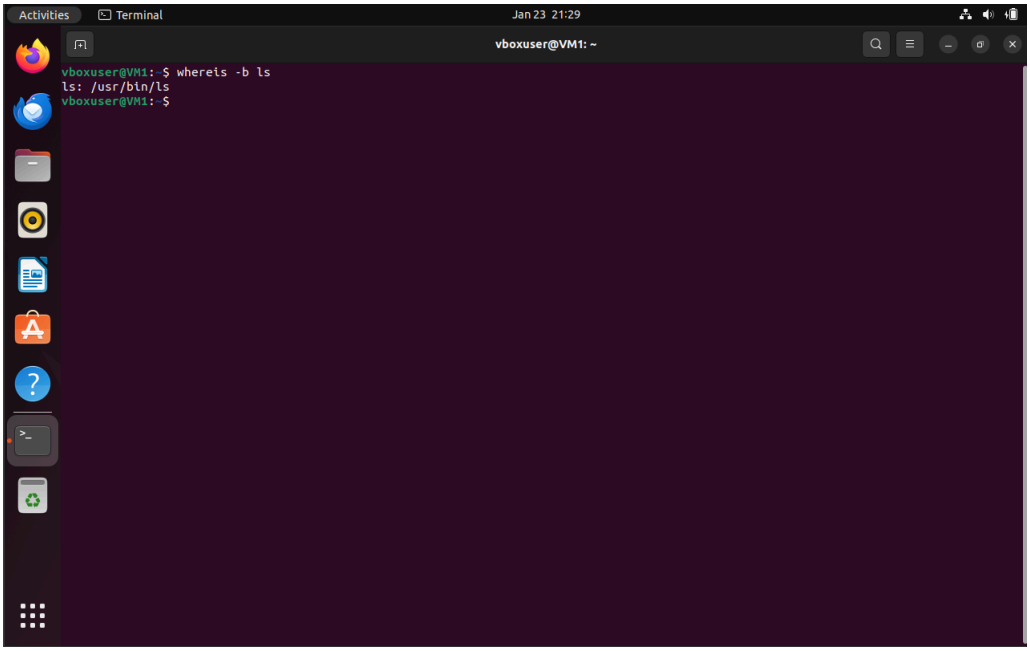
49) passwd:
Description: Changes the password for a user.
Example Usage: passwd



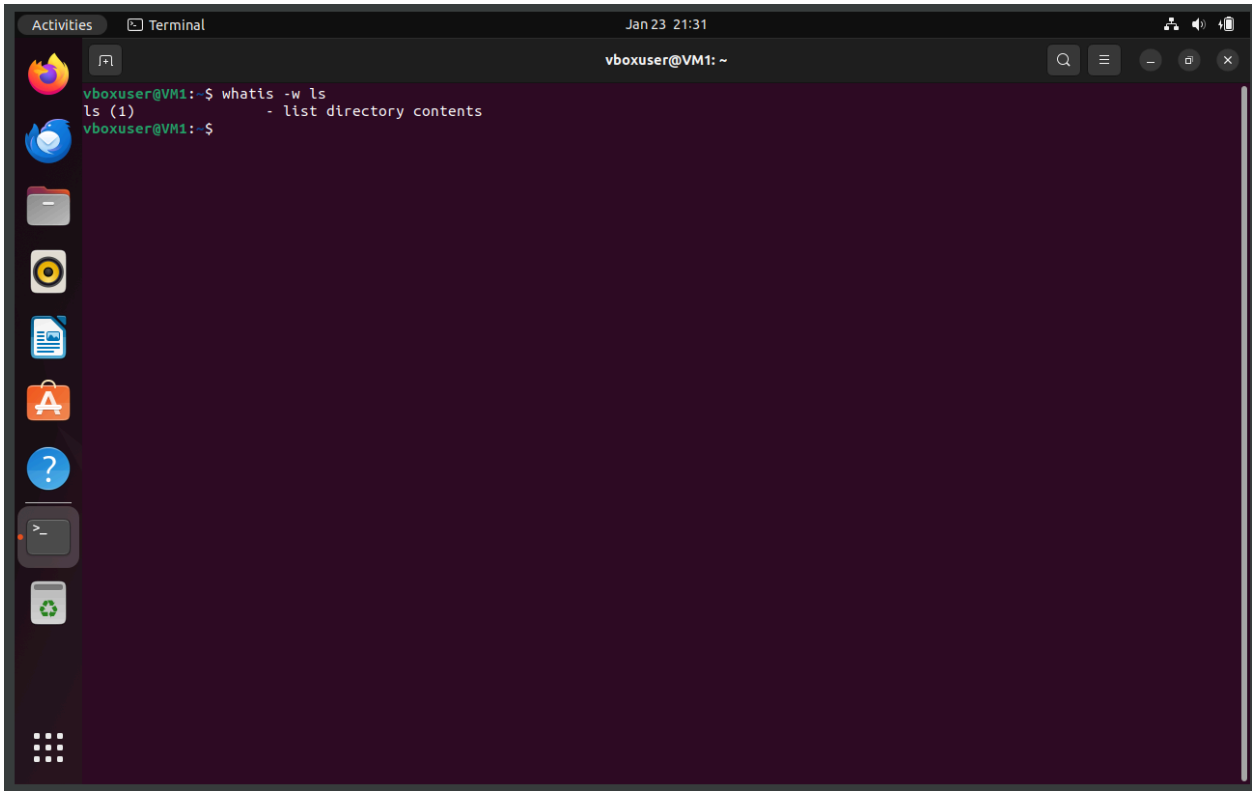
50) uname (Unix Name):
Description: Displays system information.
Example Usage: uname -a



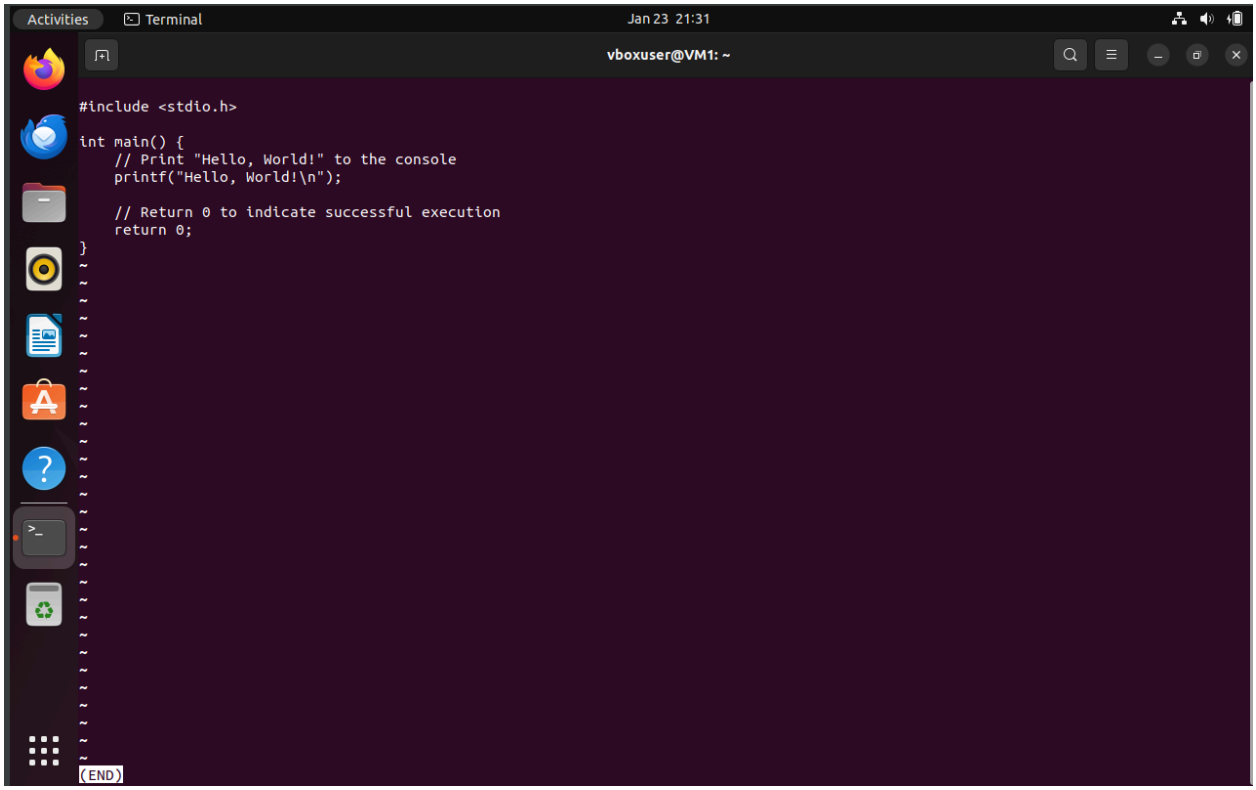
51) whereis:
Description: Locates the binary, source, and manual page files for a command.
Example Usage: whereis command



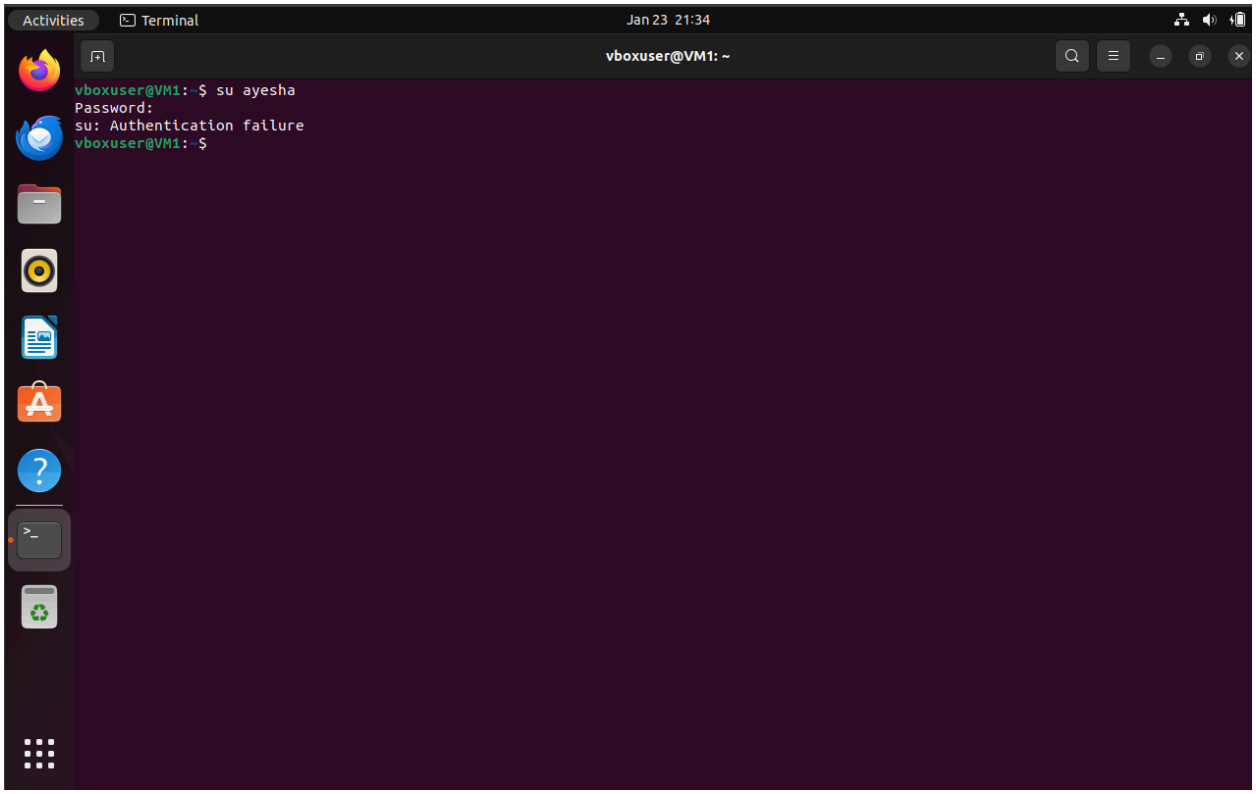
52) whatis:
Description: Displays a one-line description for a command.
Example Usage: whatis command



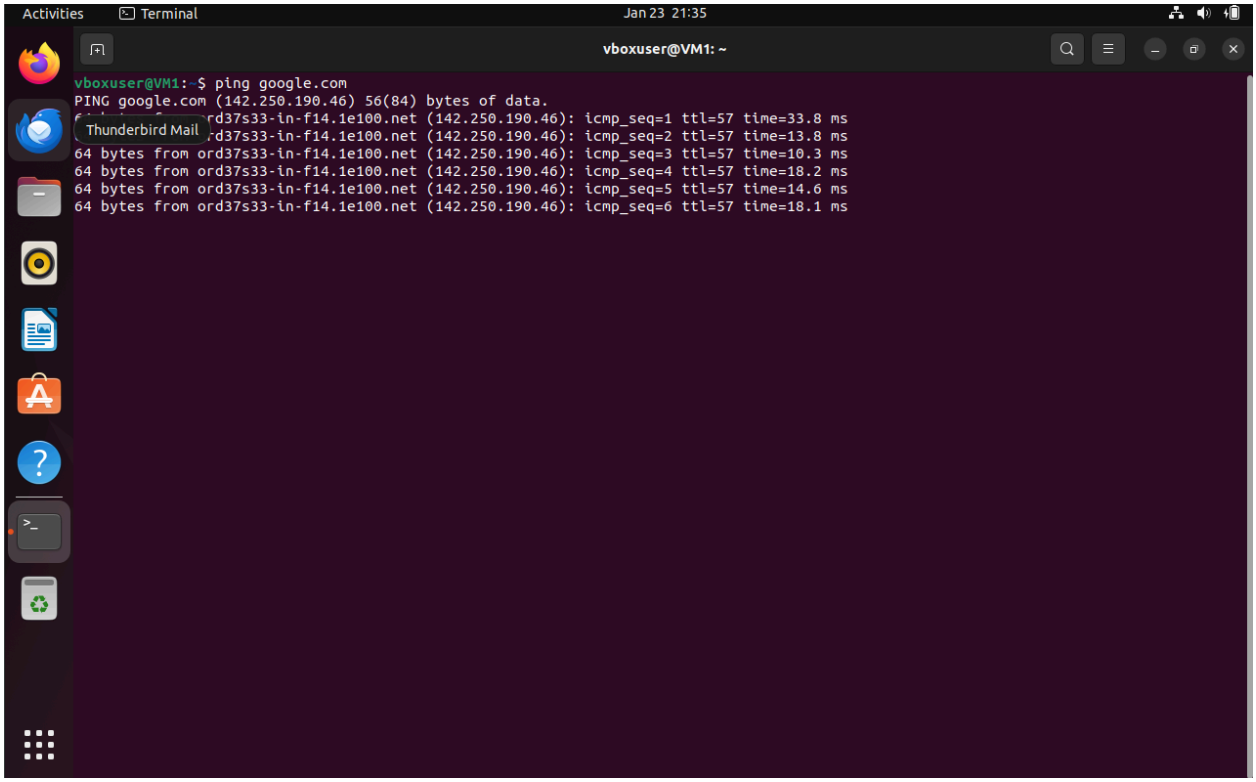
53) Less:
Description: A paginator for viewing text files
Example Usage: less filename.txt



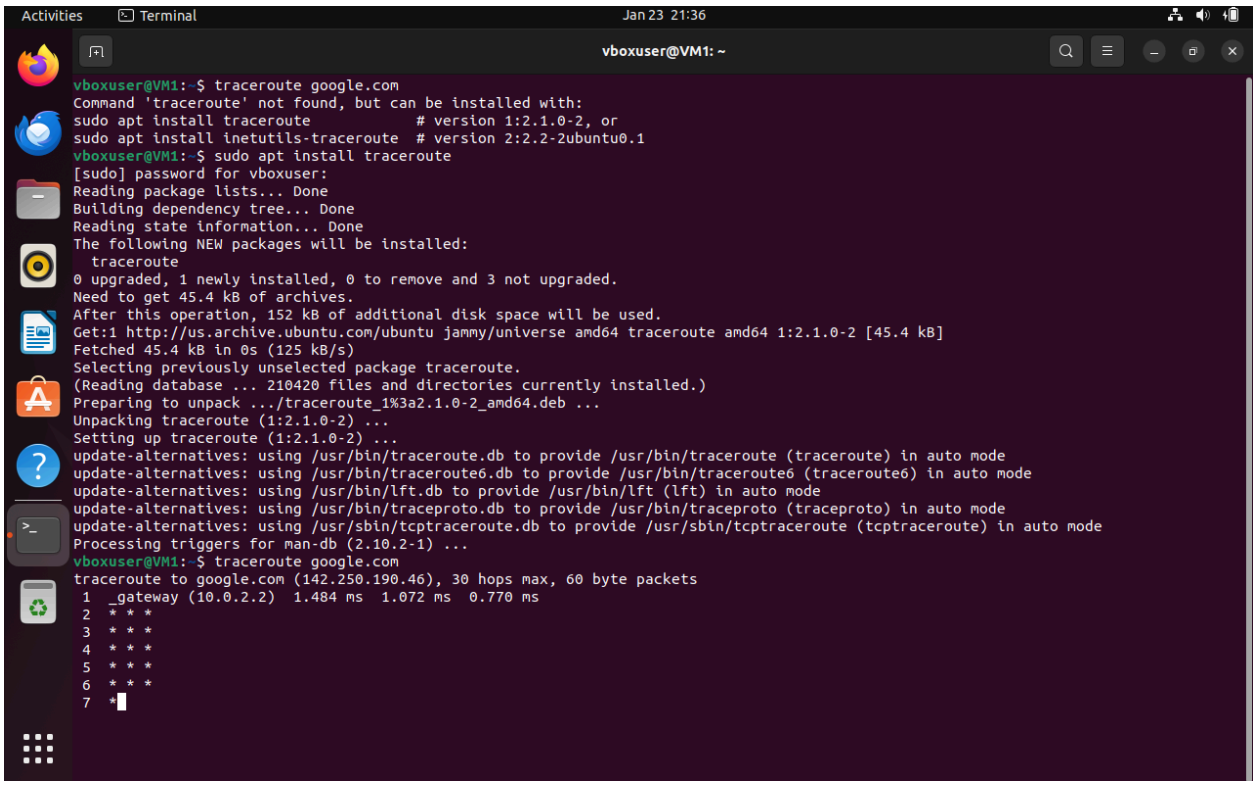
54) bsu (Switch User):
Description: Changes user identity to another user.
Example Usage: su username



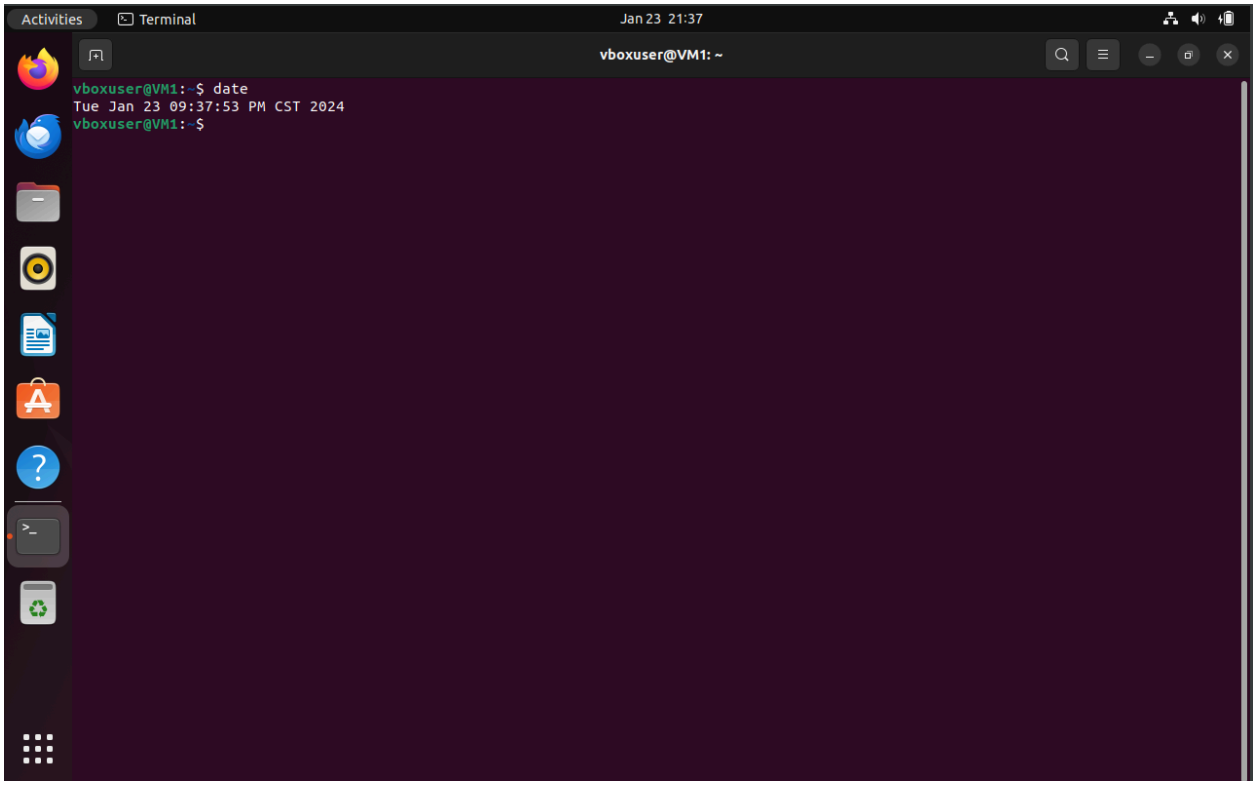
55) ping:
Description: Tests the reachability of a host on a network.
Example Usage: ping example.com



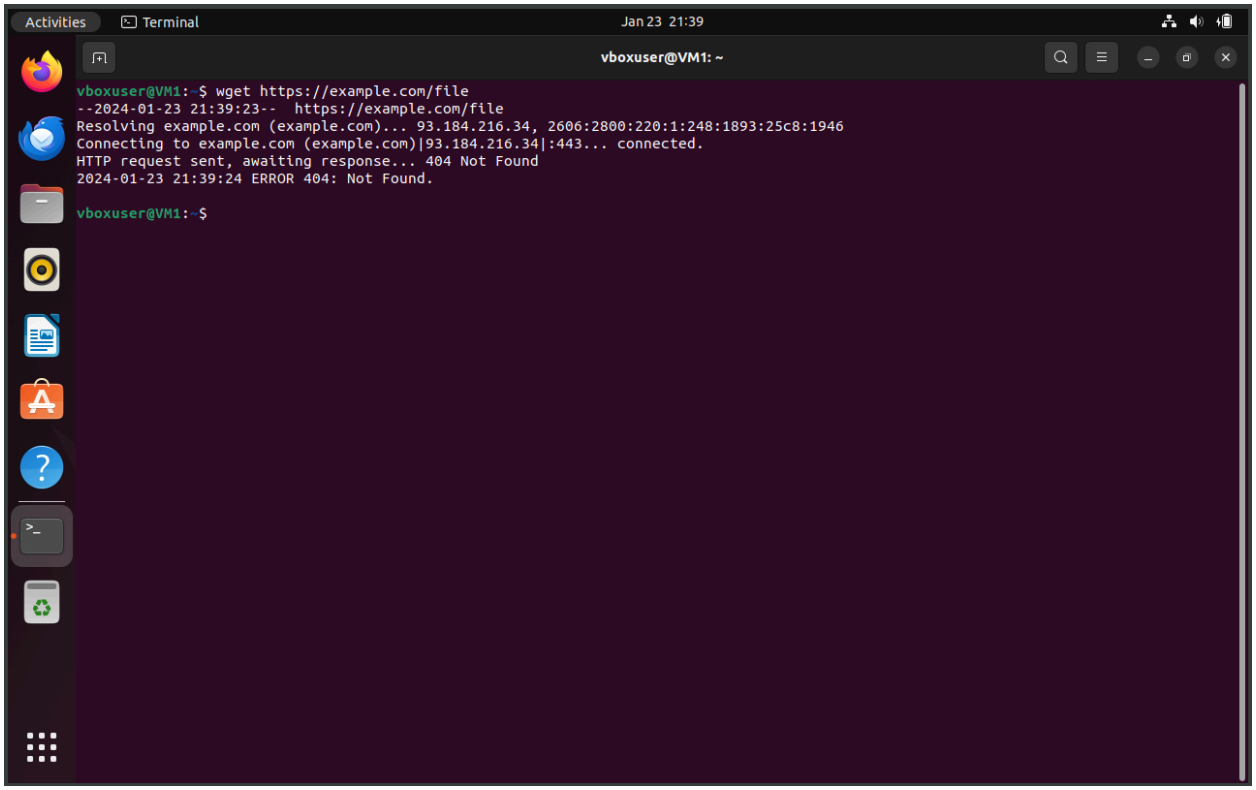
56) traceroute:
Description: Traces the route that packets take to reach a network host.
Example Usage: traceroute example.com



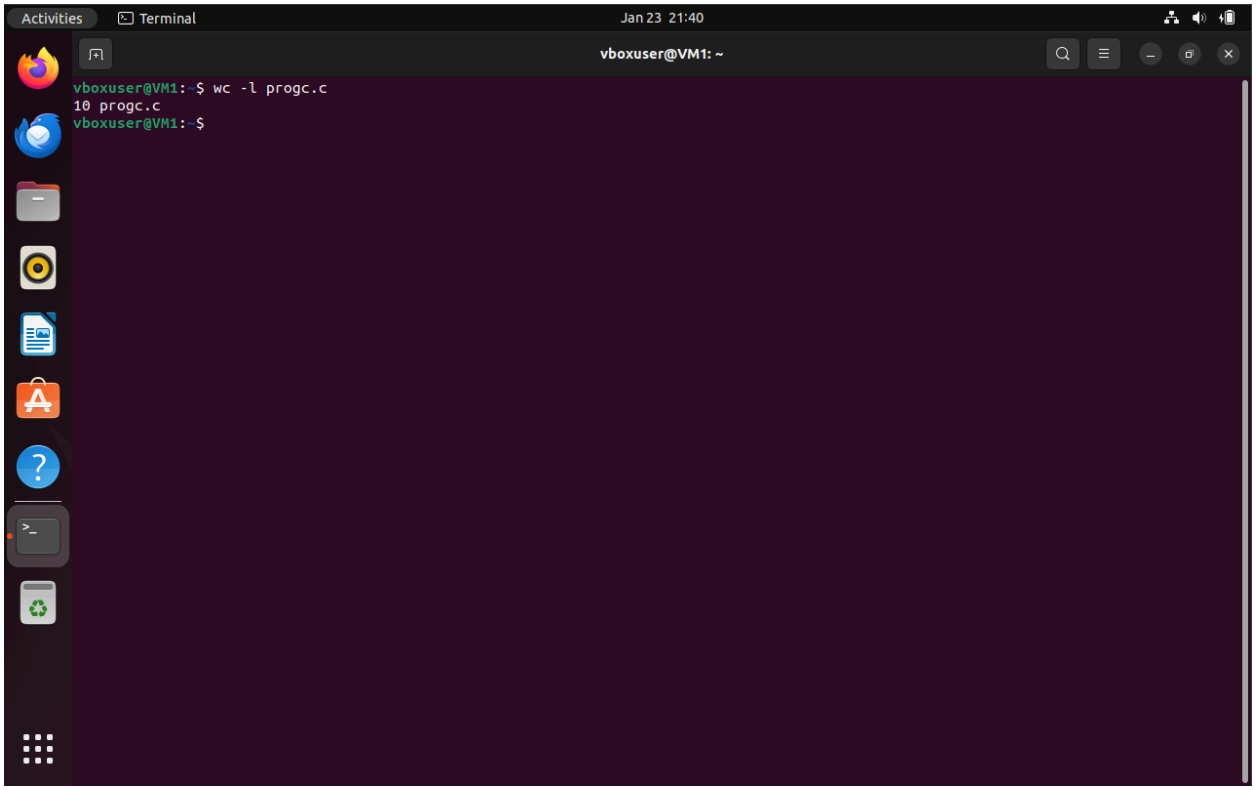
57) date:
Description: Displays the current date and time.
Example Usage: date



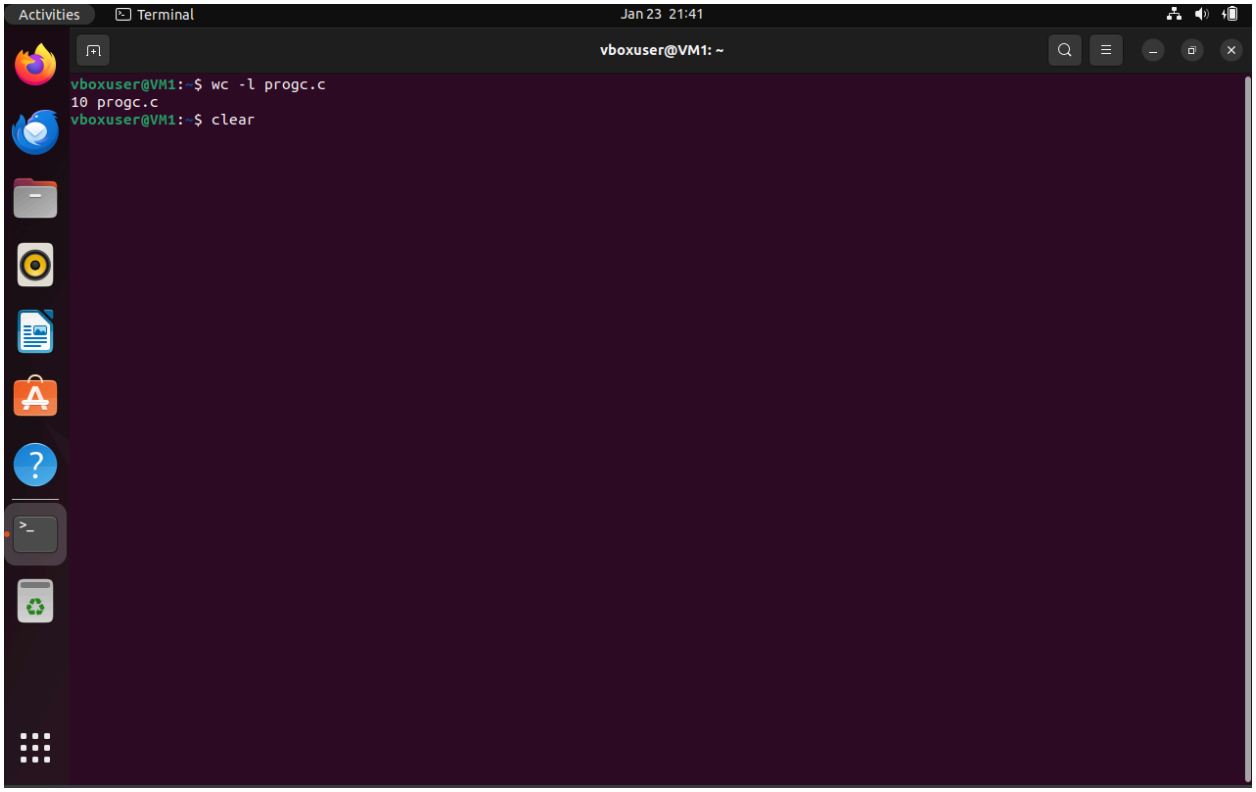
58) wget
Description: Downloads files from the internet.
Example Usage: wget https://example.com/file

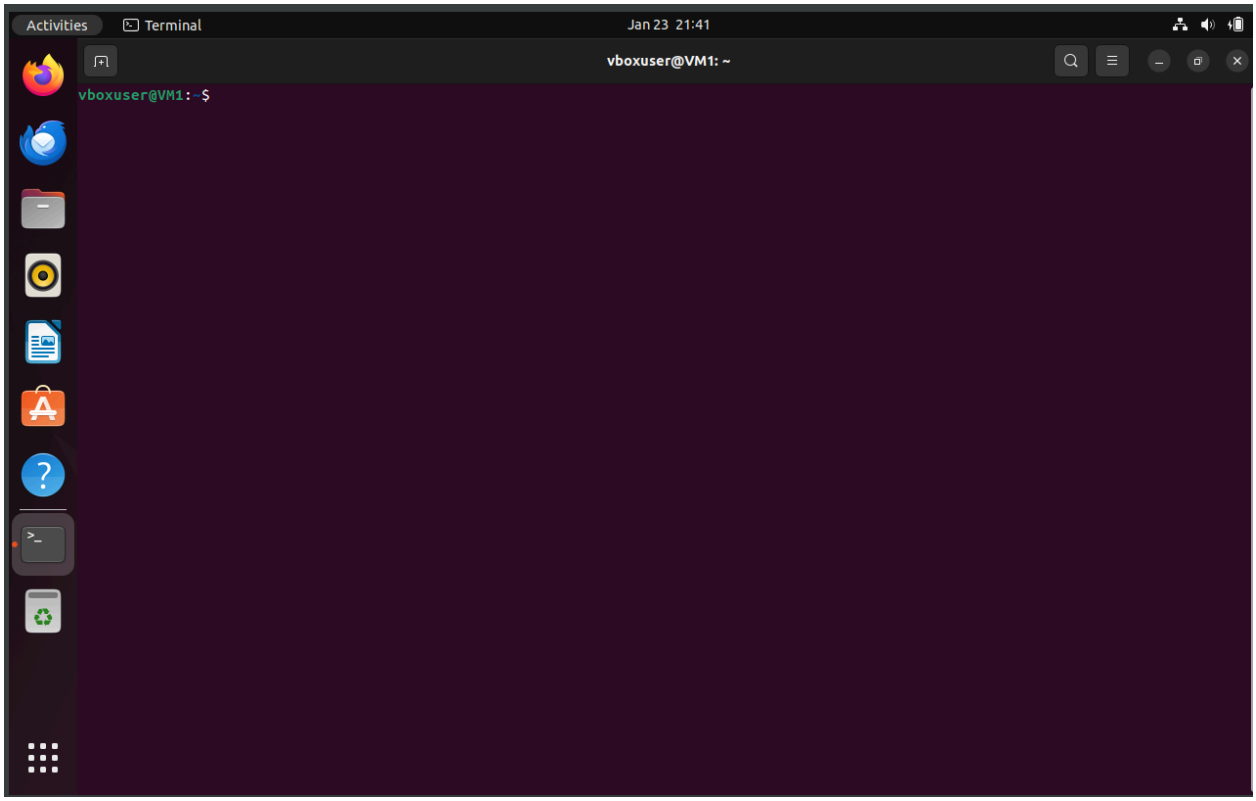


59) wc:
Description: Counts lines, words, and characters in a file.
Example Usage: wc filename.txt



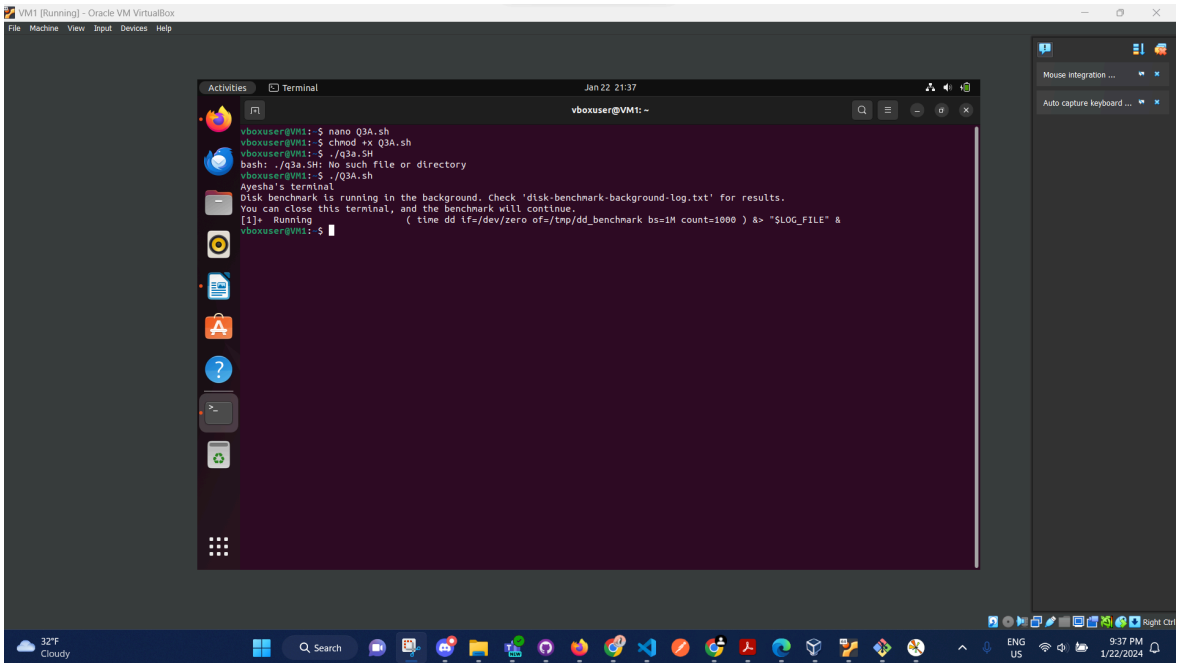
60) clear:
Description: Clears the terminal screen.
Example Usage: clear

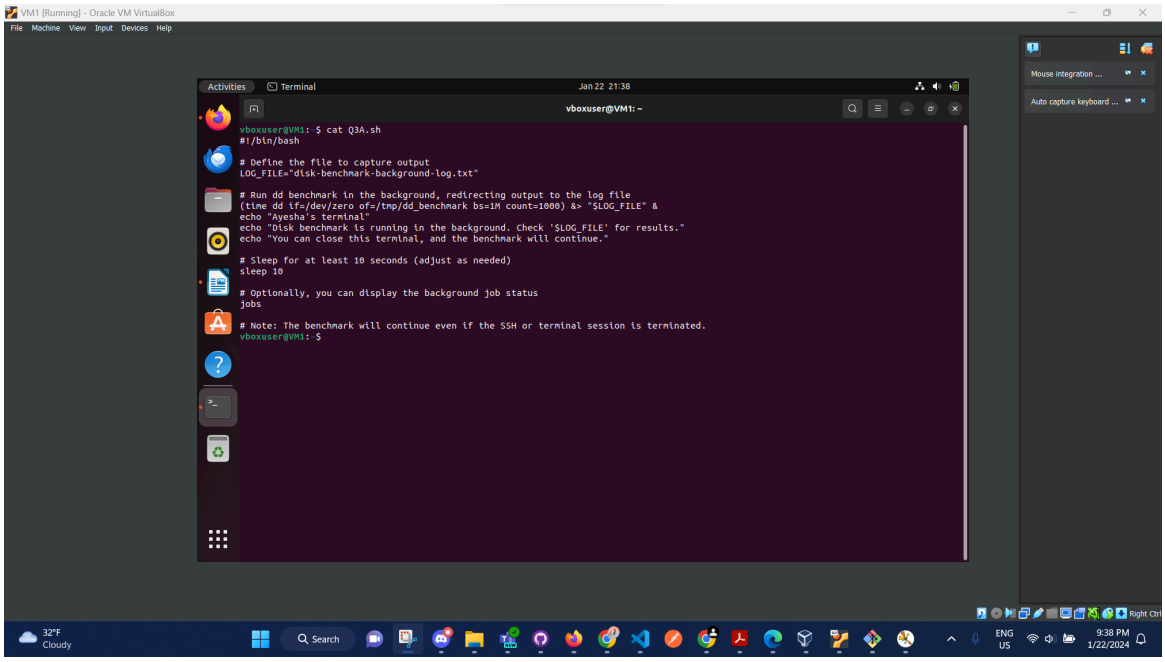

Activities
Terminal
Jan 23 21:41
vboxuser@VM1: ~
vboxuser@VM1:~$ wc -l progc.c
10 progc.c
vboxuser@VM1:~$ clear


Activities
Terminal
Jan 23 21:41
vboxuser@VM1: ~
vboxuser@VM1:~$

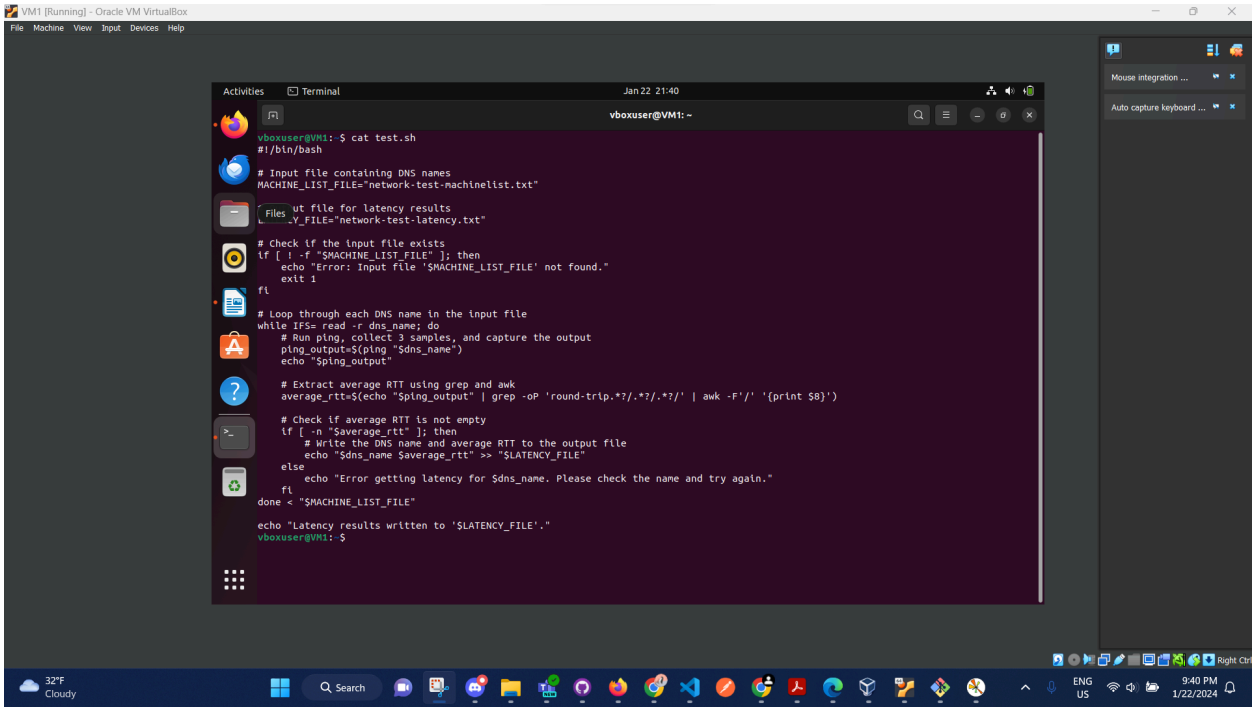
3. Write Bash scripts to do the following:

   a. Write a script called “disk-benchmark-background.sh” that uses the dd command to run a benchmark against the local disk in the background, that captures all the output (both standard out and error output) to a file “disk-benchmark-background-log.txt”. Use the “time” command to show how long the benchmark took to complete. The benchmark should run for at least 10 seconds, and it should be completed even if the ssh (or bash) session is terminated.

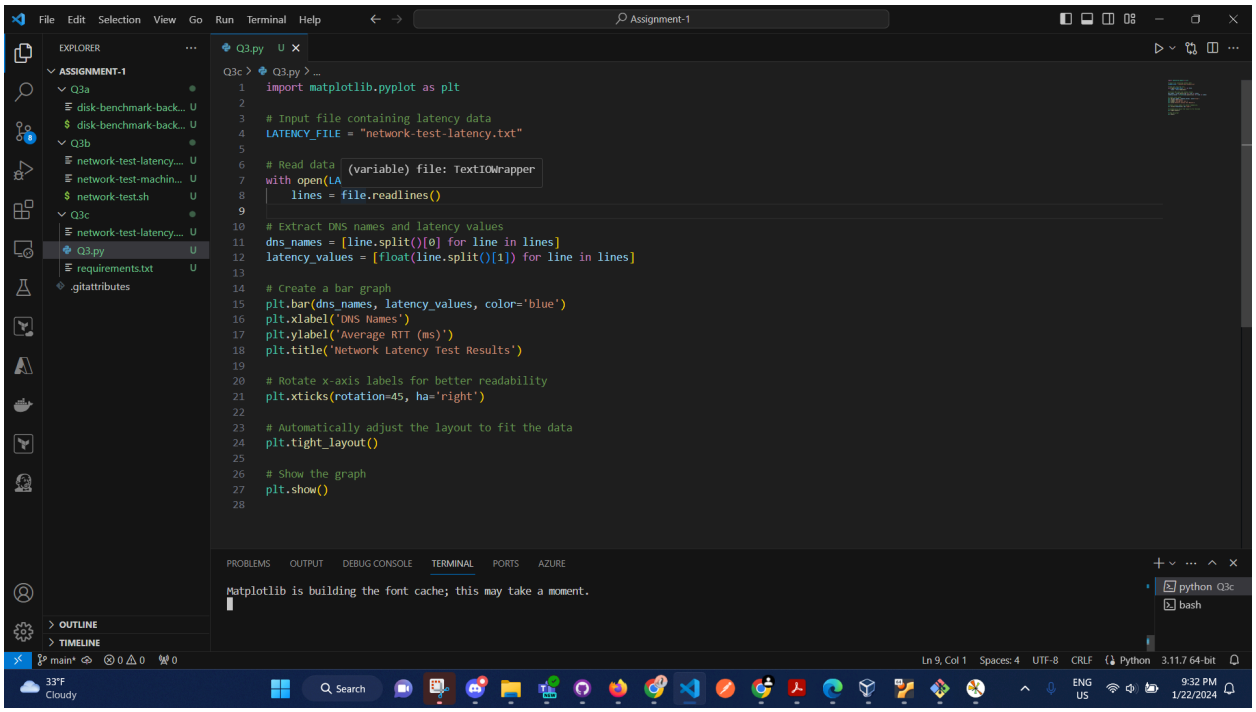




b. Write a script called “network-test.sh” that takes input a file “network-test-machinelist.txt” with a list of DNS names (e.g. google.com, iit.edu, anl.gov), each name on a separate line, and runs the ping utility collecting 3 samples from each DNS name, and writing the RTT (round trip time) average latency into a file “network-test-latency.txt” where each line will have the DNS name and average RTT separated by a space. Make sure it works with at least 10 DNS names, but it should work for an unspecified number of DNS names.



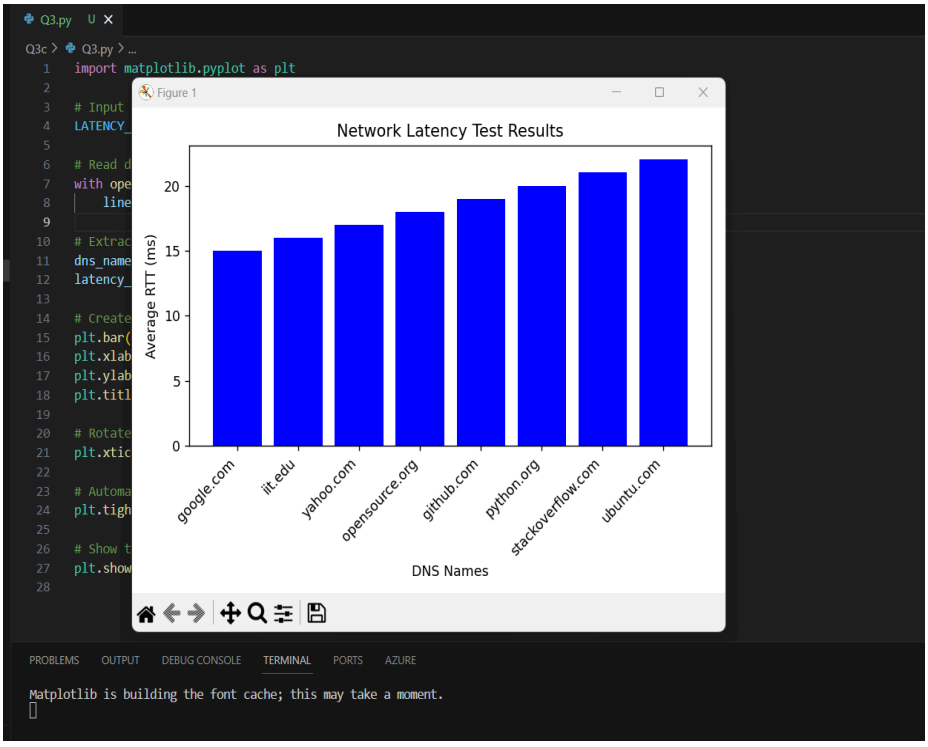
c. Write a Python matplotlib script to generate a graph of the “network-test-latency.txt” data. The graph should automatically adjust to the number of entries, and the scale of the data.



```python
import matplotlib.pyplot as plt

# Input file containing latency data
LATENCY_FILE = "network-test-latency.txt"

# Read data
with open(LA
    lines = file.readlines()

# Extract DNS names and latency values
dns_names = [line.split()[0] for line in lines]
latency_values = [float(line.split()[1]) for line in lines]

# Create a bar graph
plt.bar(dns_names, latency_values, color='blue')
plt.xlabel('DNS Names')
plt.ylabel('Average RTT (ms)')
plt.title('Network Latency Test Results')

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, ha='right')

# Automatically adjust the layout to fit the data
plt.tight_layout()

# Show the graph
plt.show()
```

(variable) file: TextIOWrapper

```
Matplotlib is building the font cache; this may take a moment.
```



4. **Answer the following questions about VMs:**

   a. **<u>In the system configuration of the VM, explain how changing the number of processors changes the behavior of your VM. Explain a scenario where you want to set this to the minimum, and a scenario where you want to set it to the maximum. Why is setting it to the maximum potentially a bad idea?</u>**

Adjusting the number of processors in a virtual machine (VM) has a profound impact on performance, primarily through the allocation of virtual CPUs. Let's delve into scenarios at both ends of the spectrum – minimum and maximum settings – and explore the nuanced effects on VM behavior.

Impact of Processor Configuration on VM Behavior:

1. **Enhanced Performance with Increased Processors:**
   - *Analogy:* Increasing processors in a VM is akin to expanding a workforce, offering heightened parallel processing capacity.
   - *Example:* Consider a VM hosting a web server experiencing a surge in incoming requests. Augmenting processors facilitates efficient handling of concurrent tasks, leading to improved response times.
2. **Resource Contention and Overhead with Fewer Processors:**
   - *Analogy:* Reducing processors can lead to resource contention, similar to downsizing a team, resulting in prolonged task execution due to inadequate computational power.
   - *Example:* In scenarios with a VM executing complex computational tasks, lowering the processor count may lead to slower execution times due to insufficient resources.
3. **Balancing Act for Optimal Resource Utilization:**
   - *Analogy:* Achieving optimal VM behavior involves a delicate balance of processors for efficient resource utilization without causing contention or underutilization.
   - *Example:* In a VM serving as a virtual desktop for multiple users, an adequate number of processors ensures smooth multitasking without one user's activities hindering others.
4. **Considerations for Host Machine Resources:**
   - *Analogy:* Crucially, host machine resources must be considered. Allocating excess processors without sufficient physical cores may lead to resource exhaustion and degraded performance.
   - *Example:* If a host machine has limited CPU cores, assigning excess processors to multiple VMs can result in contention, adversely affecting overall system performance.

This exploration underscores the importance of carefully configuring processors to optimize VM behavior, considering the diverse demands of different scenarios and the underlying host machine resources.

Scenario for Minimizing Processor Allocation:

- In situations where multiple VMs share the same host and have low individual processing requirements, configuring processors to the minimum ensures equitable distribution of resources.
    - Illustration 1: Within a development environment running lightweight VMs concurrently, minimal processors promote efficient resource utilization.
    - Illustration 2: When engaged in batch processing for data analysis tasks with a more sequential nature, setting processors to the minimum is adequate for reasonable execution.

Scenario for Maximizing Processor Allocation:

- When dealing with a single VM hosting resource-intensive applications, maximizing processors contributes to enhanced overall performance.
    - Illustration 1: A high-traffic e-commerce website experiences peak times and benefits from maximum processors to efficiently manage increased workloads.
    - Illustration 2: A database server handling substantial concurrent transactions thrives with maximum processors, improving responsiveness and overall throughput.

Caution Regarding Maximum Processor Allocation:

- While maximizing processors can elevate performance, it is imperative to consider the overall resources of the host machine to prevent potential resource contention.
    - Cautionary Example: Allocating an excess number of processors without sufficient physical cores may result in resource exhaustion, adversely impacting the performance of all VMs.

In essence, adjusting VM processor configurations demands a thoughtful assessment of specific workloads and host resources, ensuring optimal performance without introducing contention or risking resource depletion.

b. <u>In the system configuration of the VM, under the Acceleration Tab, explain the difference between the paravirtualization options: None, Legacy, Minimal, Hyper-V, and KVM. Explain which one would be best to use with Ubuntu Linux, and why.</u>

The elucidation below delves into the paravirtualization options available under the Acceleration Tab within the system configuration of a virtual machine (VM).

Overview of the Acceleration Tab: In the configuration settings of a VM, the Acceleration Tab provides users with a spectrum of paravirtualization options aimed at enhancing performance. These options bring forth specialized interfaces and optimizations, proving especially advantageous for tasks that demand substantial computing resources.

Paravirtualization Options:

1. **None:**
    - *Definition:* The "None" option means there are no paravirtualization features, relying solely on traditional virtualization methods.
    - *Examples/Usage:*
        1. *Compatibility Priority:* Choose "None" for broad compatibility with various guest operating systems, especially those lacking support for paravirtualized drivers.
        2. *Legacy System Emulation:* Opt for "None" when emulating legacy systems that are incompatible with paravirtualization enhancements.
        3. *Generic Testing Environments:* Ideal for testing environments that support various operating systems without specific optimizations.
2. **Legacy:**
    - *Definition:* The "Legacy" option provides basic paravirtualization support, often emulating legacy interfaces for compatibility.
    - *Examples/Usage:*
        1. *Supporting Older Operating Systems:* Opt for "Legacy" when running VMs with older guest operating systems lacking compatibility with modern paravirtualized drivers.
        2. *Legacy Software Testing:* Suitable for environments focused on testing legacy software or systems.
        3. *Hardware Compatibility:* Appropriate when dealing with hardware dependent on legacy interfaces for proper functioning.
3. **Minimal:**
    - *Definition:* "Minimal" paravirtualization offers essential features without unnecessary complexity, maintaining a balanced approach.
    - *Examples/Usage:*
        1. *Lightweight Workloads:* Choose "Minimal" for VMs with lightweight workloads, benefiting from basic paravirtualization without added intricacy.
        2. *General Testing Scenarios:* Suitable for creating VMs in general testing scenarios, achieving a balance between enhanced performance and simplicity.
        3. *Educational Environments:* Effective in educational settings that prefer a simplified paravirtualization setup.
4. **Hyper-V:**
    - *Definition:* The "Hyper-V" option customizes paravirtualization support for Microsoft's virtualization platform.
    - *Examples/Usage:*
        1. *Running on Hyper-V Hosts:* Opt for "Hyper-V" when the VM operates on a Hyper-V host, ensuring optimized performance and compatibility.
        2. *Integration with Hyper-V Features:* Ideal for utilizing features unique to the Hyper-V environment, such as enhanced integration services.
        3. *Cross-Platform Compatibility:* Useful when maintaining compatibility with both Hyper-V and non-Hyper-V environments is essential.

- *Definition:* The "KVM" (Kernel-based Virtual Machine) paravirtualization option is crafted for peak performance on Linux hosts leveraging the KVM hypervisor.
- *Examples/Usage:*
    1. *Linux Server Virtualization:* Opt for "KVM" when your VM operates on a Linux server utilizing KVM, ensuring optimal performance through native virtualization capabilities.
    2. *Ubuntu Linux Hosts:* Choose "KVM" specifically for Ubuntu Linux hosts, aligning with the inherent virtualization capabilities of the Linux environment.
    3. *Efficient Resource Utilization:* Ideal when emphasizing efficient resource utilization on virtualization hosts based on the Linux platform.

Best Option for Ubuntu Linux:
Given the strong support for KVM in Ubuntu Linux, selecting the "KVM" paravirtualization option proves to be the most fitting. This decision guarantees peak performance and harmonizes with the inherent virtualization capabilities of Linux, delivering effective and smooth virtualization on Ubuntu hosts.

c. <u>In storage devices when configuring the VM, there are multiple types of storage controllers: explain the difference between the IDE, SATA, and NVMe controller. Give an example for each type of storage controller of a scenario where you may want to use this type of controller.</u>

When establishing a virtual machine (VM) and configuring storage settings, the term "storage controllers" refers to virtualized components responsible for managing communication between the VM and its storage devices. These controllers replicate the functionalities of physical storage controllers within the virtual environment. Various virtualization platforms provide diverse types of storage controllers, including IDE, SATA, SCSI, and NVMe controllers. The selection of a storage controller depends on factors such as the VM's operating system and the required storage devices, providing flexibility in VM configuration.

Here is a breakdown outlining the distinctions between storage controllers(IDE, SATA, and NVMe),in the context of configuring virtual machines (VMs).

Controller Type:
1. **IDE (Integrated Drive Electronics):**

    - ***Description:*** IDE serves as a traditional storage controller primarily used in older systems.
    - ***Key Differences:***
      a) The transfer rate is comparatively slower.
      b) Only two devices are allowed for one channel.
      c) It is compatible with older hard drives and optical drives.

- **Real-Life Examples:**
  Scenario 1: Legacy System Support
  Explanation: In older systems without compatibility for SATA or NVMe, opting for an IDE controller enables the connection of traditional hard drives or optical drives.

  Scenario 2: Legacy Software Testing
  Explanation: When conducting tests on legacy software dependent on IDE controllers, configuring a VM with an IDE controller accurately replicates the hardware environment.

  Scenario 3: Basic Data Storage
  Explanation: In situations where high-speed data transfer is not a critical requirement, such as basic file storage needs, IDE controllers are suitable for cost-effective solutions.

2. **NVMe (Non-Volatile Memory Express):**
   - ***Description:*** In the realm of SSDs, NVMe stands as a high-performance storage controller.
   - ***Key Differences:***
     a) NVMe exhibits the fastest data transfer rates among its counterparts.
     b) It is optimal for use with SSDs and offers low latency.
     c) NVMe is a preferred choice for high-performance workloads due to its support for multiple queues and parallelism.

   - **Real-Life Examples:**
     Scenario 1: Mainstream Desktop Virtualization
     Explanation: SATA controllers find widespread usage in VMs emulating mainstream desktop environments, utilizing regular hard drives or SSDs for storage.

     Scenario 2: Multimedia Content Editing
     Explanation: For VMs engaged in multimedia content editing, where accelerated data transfer rates are advantageous, SATA controllers offer an optimal balance of performance and cost.

     Scenario 3: Distributed Database Storage
     Explanation: In situations demanding distributed database storage, SATA controllers facilitate the support of multiple drives, ensuring data redundancy and reliability.

3. **SATA (Serial ATA):**
   - *Description:* SATA stands out as a widely used controller that integrates well with modern systems.
   - *Key Differences:*
     a) It boasts faster transfer rates compared to IDE.
     b) SATA supports the hot-swapping of drives.
     c) It is commonly used with both SSDs and hard drives.

- **Real-Life Examples:**
  Scenario 1: High-Performance Virtualization
  Explanation: NVMe controllers stand out in virtual environments that demand high-performance storage, such as those running intricate simulations or resource-intensive applications.

  Scenario 2: Real-Time Data Processing
  Explanation: For VMs engaged in real-time data processing tasks, like financial transactions or scientific simulations, NVMe controllers guarantee minimal latency and swift data access.

  Scenario 3: Big Data Analytics
  Explanation: VMs dedicated to big data analytics find advantage in NVMe controllers due to their exceptional data transfer rates, contributing to the efficiency of data-intensive operations.

Hence, the selection of a storage controller—IDE, SATA, or NVMe—hinges on the distinct performance requirements and use cases of the virtualized environment. Each controller type possesses unique strengths, catering to diverse storage needs within VM configurations.

d. <u>In the network configuration of the VM, there are multiple types of network adapters: explain the difference between NAT, Bridged Adapter, Internal Network, and Host-only Network. Give an example for each type of network of a scenario where you may want to use this type of network.</u>

Network Adapters:

Network adapters within virtual machines facilitate communication between the virtual machines and the external network, playing a pivotal role in determining how VMs connect and engage with the broader network environment.

Types of Network Adapters:

1. **NAT (Network Address Translation):**
   - *Description:* NAT allows a VM to utilize the host's IP address, providing internet access while concealing the VM's individual IP address.
   - *Example Scenario:* When a VM requires internet access without directly exposing its own IP address.
2. **Bridged Adapter:**
   - *Description:* The Bridged Adapter connects a VM directly to the physical network, presenting it as a distinct device on the same network as the host.
   - *Example Scenario:* In situations where a VM needs to function as an independent device with its own IP address on the network.
3. **Internal Network:**

- *Description:* The Internal Network facilitates communication between VMs on the same host but isolates them from external networks.
- *Example Scenario:* When establishing an isolated network for VMs needing interaction with each other but not with the external network.

4. **Host-only Network:**
   - *Description:* The Host-only Network allows communication between VMs and the host while keeping them isolated from external networks.
   - *Example Scenario:* In cases where VMs need exclusive communication with each other and the host without external network access.

Real-Life Examples for Each Network Adapter:

1. **NAT:**
   - *Scenario 1: Software Development Environment*
     - *Explanation:* When developing software requiring internet access for updates or downloads, NAT ensures seamless connectivity without exposing individual VM IP addresses.
   - *Scenario 2: Web Browsing Sandbox*
     - *Explanation:* Establishing a sandbox environment for web browsing, where VMs share the host's IP address, protects individual VMs from potential internet threats.
   - *Scenario 3: System Updates*
     - *Explanation:* In situations where VMs need to fetch updates from external servers, NAT allows them to share a common external IP address for update retrieval.
2. **Bridged Adapter:**
   - *Scenario 1: Web Server Testing*
     - *Explanation:* When testing a web server within a VM that needs to be accessible from external devices, using a Bridged Adapter ensures the VM has its own IP address on the host's network.
   - *Scenario 2: Remote Device Testing*
     - *Explanation:* In scenarios where VMs need to communicate with physical devices on the host's network, using a Bridged Adapter provides direct access to the external network.
   - *Scenario 3: Network Monitoring VM*
     - *Explanation:* Creating a VM for network monitoring that needs to capture external network traffic is facilitated by a Bridged Adapter, allowing direct interaction with the external network.
3. **Internal Network:**
   - *Scenario 1: Database Server and Client VMs*
     - *Explanation:* When setting up VMs for a database server and client applications that need to communicate with each other, using an Internal Network ensures internal communication without external access.
   - *Scenario 2: Development and Testing Environment*
     - *Explanation:* In a development environment where VMs simulate different components of a system, using an Internal Network keeps the communication isolated within the host for testing purposes.
   - *Scenario 3: Training Labs*

        - *Explanation:* Setting up training labs with VMs that need to interact with each other but not with external networks can be achieved using an Internal Network for a controlled learning environment.
4. **Host-only Network:**
    - *Scenario 1: Secure File Sharing*
        - *Explanation:* When VMs need to share files securely with the host but not with external networks, using a Host-only Network ensures isolated and secure communication.
    - *Scenario 2: Development and Debugging Environment*
        - *Explanation:* Creating a VM for development and debugging purposes where the VM needs exclusive communication with the host is facilitated by a Host-only Network.
    - *Scenario 3: Collaborative Project Environment*
        - *Explanation:* In collaborative projects where VMs need to interact with each other and the host but remain isolated from external networks, a Host-only Network provides a secure collaboration environment.

The selection of a network adapter type—NAT, Bridged Adapter, Internal Network, or Host-only Network—depends on specific use cases and requirements, offering flexibility in configuring VMs for various networking scenarios. Each network adapter type serves distinct purposes, from internet access to isolated internal communication, ensuring efficient and secure network configurations within virtualized environments.

e. **For the USB configuration of the VM, explain the difference between USB 1.1, 2.0, and 3.0 controllers.**

Details regarding USB controllers—USB 1.1, USB 2.0, and USB 3.0—are outlined as follows:

USB 1.1:

- *Description:* USB 1.1, also referred to as USB 1.0, marked the initial widespread adoption of the USB standard in 1996.
- *Key Characteristics:*
    - *Maximum Data Transfer Rate:* 12 Mbps.
    - *Connector:* Standard-A and Standard-B connectors.
    - *Introduced features like Plug and Play and hot-swapping.*
    - *Utilized for basic peripherals such as keyboards, mice, and printers.*

USB 2.0:

- *Description:* Released in 2000, USB 2.0 represented an improvement over USB 1.1, providing higher data transfer rates.
- *Key Characteristics:*
    - *Maximum Data Transfer Rate:* 480 Mbps.
    - *Backward compatible with USB 1.1.*
    - *Enhanced power management.*

- *Supported a broader range of devices, including external hard drives and digital cameras.*

USB 3.0:

- *Description:* Introduced in 2008, USB 3.0 brought significant advancements in speed and efficiency.
- *Key Characteristics:*
  - *Maximum Data Transfer Rate:* 5 Gbps.
  - *Introduced a new SuperSpeed transfer mode.*
  - *Larger bandwidth for faster data transfer.*
  - *Improved power efficiency.*
  - *Backward compatible with USB 2.0.*
- USB 1.1 (USB 1.0):
  - **Description:** USB 1.1, also known as USB 1.0, was the initial widely adopted USB standard introduced in 1996.
  - **Data Transfer Rate:** 12 Mbps.
  - **Connector Types:** Standard-A, Standard-B.
  - **Key Features:** Introduced Plug and Play, hot-swapping; used for basic peripherals like keyboards, mice, and printers.
- USB 2.0:
  - **Description:** USB 2.0, released in 2000, improved upon USB 1.1, offering a higher data transfer rate of 480 Mbps.
  - **Backward Compatibility:** Yes, with USB 1.1.
  - **Enhancements:** Introduced enhanced power management.
  - **Extended Device Support:** Included external hard drives and digital cameras.
- USB 3.0:
  - **Description:** USB 3.0, introduced in 2008, brought significant advancements in speed and efficiency.
  - **Data Transfer Rate:** 5 Gbps.
  - **Transfer Mode:** Introduced SuperSpeed transfer mode.
  - **Connector Types:** Standard-A, Standard-B, Micro-B.
  - **Power Efficiency:** Improved power management.
  - **Backward Compatibility:** Yes, with USB 2.0.

USB technology has evolved over these standards, providing users with increasingly faster and more efficient data transfer options across various devices.