**Optimizing Linux Server Performance: Configuring Swap Memory on AWS EC2**

Key takeaways:

1. **Understanding Swap Memory**
2. **Configuring Swap Memory on Linux**
3. **Automating Swap Configuration**:
4. **Stress Testing for Performance Evaluation**
5. **Monitoring System Performance**:

**Introduction:**

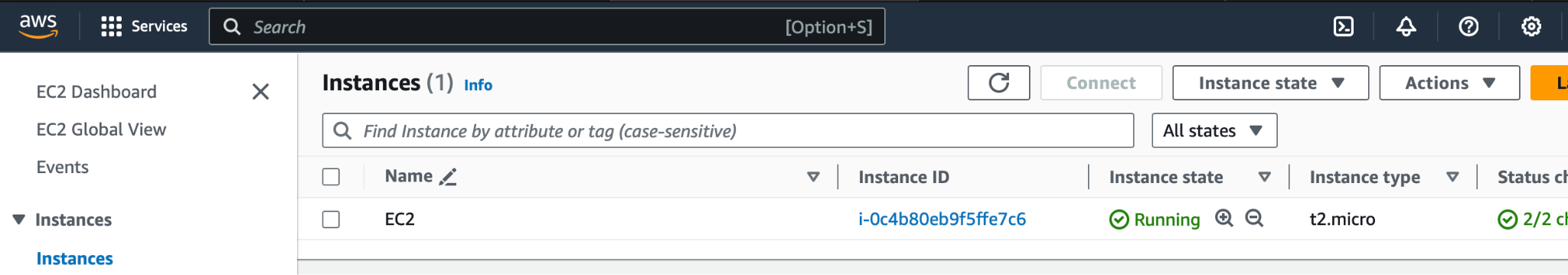
In this guide, we'll walk through the implementation of swap memory on an Ubuntu virtual machine hosted on AWS EC2((Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2024-03-01).Swap memory, also known as swap space, plays a crucial role in managing system memory effectively, especially when the RAM is fully utilized. Additionally, we'll explore stress testing to evaluate system performance under heavy CPU load.

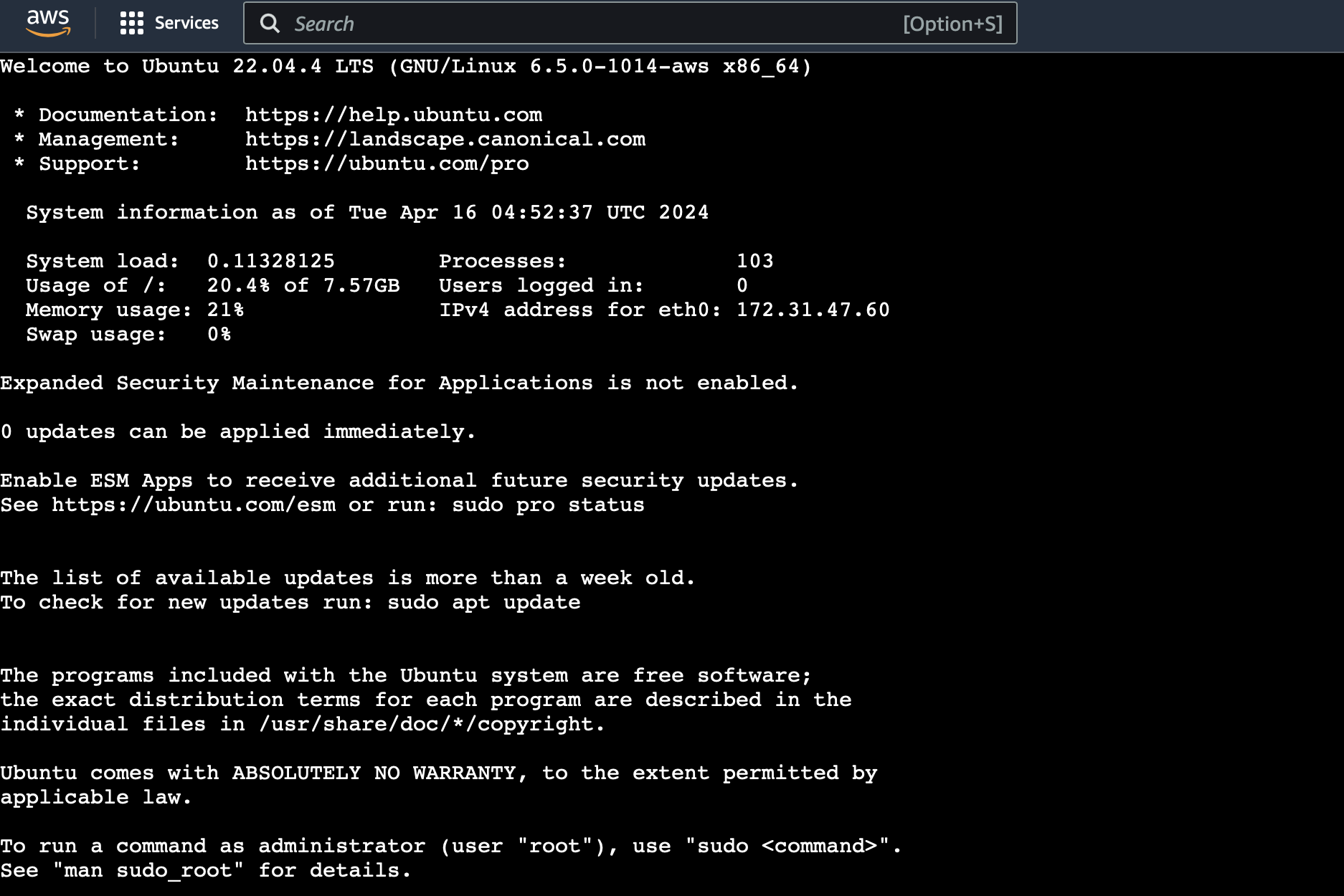
**What is Swap Memory?**

Swap memory, often known as swap space, Swap memory serves as an extension of physical RAM, residing on disk space. When RAM is fully utilized, the operating system transfers inactive data from RAM to swap space to make room for active processes. While accessing data from swap space is slower compared to RAM, it prevents the system from running out of memory.

**Steps Involved:**

1.Creating an AWS EC2 Instance: Launch an Ubuntu EC2 instance to configure swap memory.

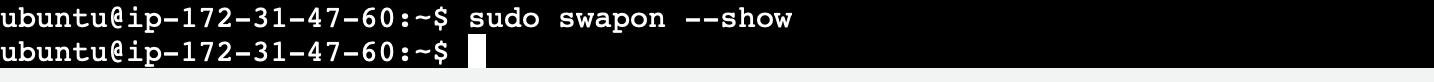




**2.Checking Current Swap Status:**

Before proceeding, let's check the current swap status. Although it's possible to have multiple swap files or partitions, usually one suffices. To verify, execute the following command:

**$Sudo swapon --show**



If you receive no output, it indicates that your system currently lacks swap space. You can verify that there is no active swap using the free utility:



In the output, you can observe that there is no active swap on the system, as indicated in the Swap row.

**3.Creating a Swap File: Determine the appropriate swap size based on RAM and create a swap file using "fallocate" command.**

When deciding how much swap space to use based on your RAM size and whether you want hibernation, keep in mind that hibernation requires more space because it saves your system's state when you turn it off and restores it when you turn it back on

This table lists the swap sizes recommended depending on your RAM

| **RAM Size** | **Swap size (without hibernation)** | **Swap size (with hibernation)** |
| --- | --- | --- |
| 256MB | 256MB | 512MB |
| 512MB | 512MB | 1GB |
| 1GB | 1GB | 2GB |
| 2GB | 1GB | 3GB |
| 3GB | 2GB | 5GB |
| 4GB | 2GB | 6GB |
| 6GB | 2GB | 8GB |
| 8GB | 3GB | 11GB |

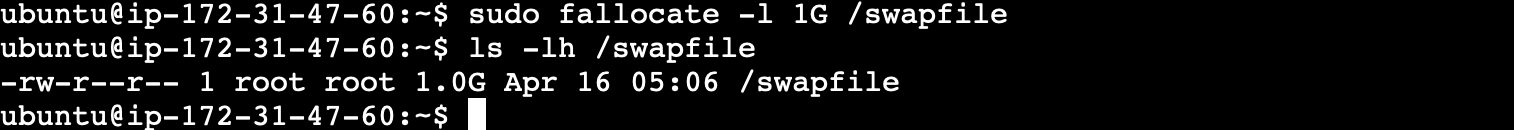
So in my case my system has 1GB of main memory. So we will create a swap file of 1 GB in size.

The best way of creating a swap file is with the fallocate program. This command instantly creates a file of the specified size.In my case my system has 1GB of main memory and am going to create a swap file of 1 GB in size.

sudo fallocate -l 1G /swapfile

We can verify that the correct amount of space was reserved by typing:

**ls -lh /swapfile**

****

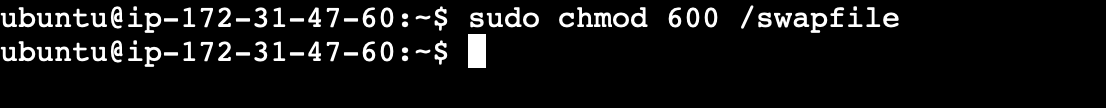
## **4.Enabling Swap File: Set appropriate permissions using "chmod" and mark the file as swap space using "mkswap". Enable swap file usage with "swapon".**

## 

To make the file accessible only to root, type the following command:

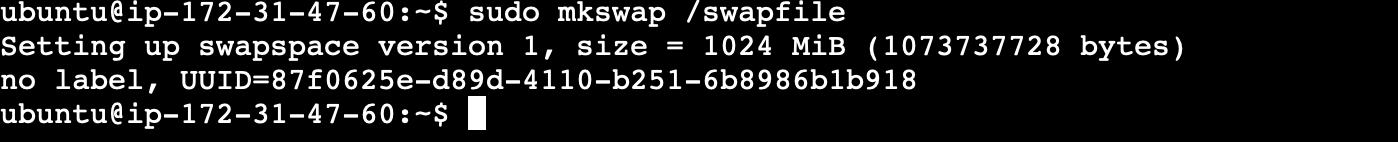
Sudo chmod 600 /path/to/your/swapfile

This command sets the permissions of the swap file to allow read and write access only for the owner (root), and no access for other users.



We can now mark the file as swap space by typing:

sudo mkswap /swapfile



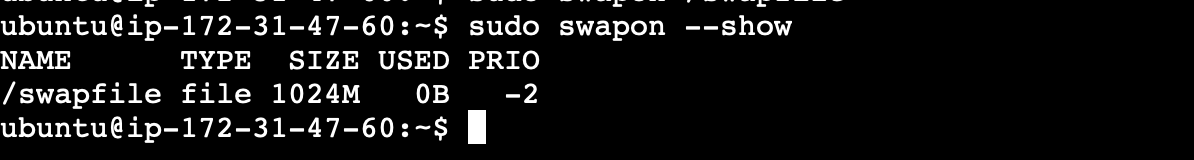
After marking the file, we can enable the swap file, allowing our system to start using it:

**sudo swapon /swapfile**

**5.Verifying Swap Memory:**

Verify that the swap is available by typing:

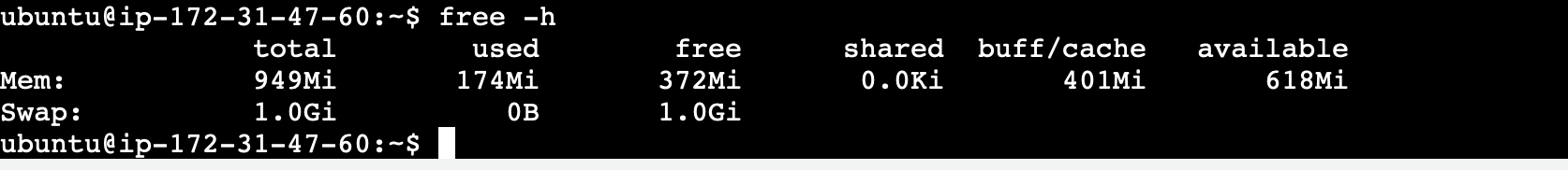
**​​sudo swapon --show**

****

**6.Making the Swap File Permanent**

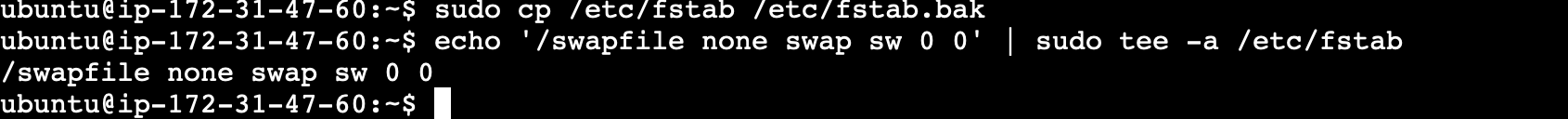
We have successfully added swap memory to our system. Execute one of the below commands to view current active swap memory on our system:

free -h



## 

Our recent changes have enabled the swap file for the current session. However, if we reboot, the server will not retain the swap settings automatically. We can Add the swap file to "/etc/fstab" for automatic activation on system boot.Back up the /etc/fstab file in case anything goes wrong:



**7. Updating Swappiness Parameter**

Adjust the swappiness parameter to control the frequency of swapping between RAM and swap space.

The swappiness parameter determines how frequently your system moves data from RAM to the swap space. It's a value ranging from 0 to 100, representing a percentage.

With values near zero, the system avoids swapping data to disk unless absolutely necessary. Remember, accessing the swap file takes much longer than accessing RAM and can significantly reduce performance. Thus, reducing reliance on swap generally improves system speed.

Values closer to 100 prioritize moving more data into swap to keep more RAM space available. Depending on your application's memory usage or server purpose, this approach might be preferable in certain scenarios.

We can see the current swappiness value by typing:

**cat /proc/sys/vm/swappiness**

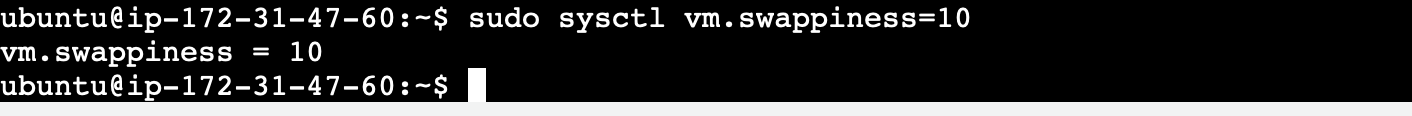
****

Now change the swappiness kernel parameter as per our requirement. It tells the system how often system utilize this swap area.

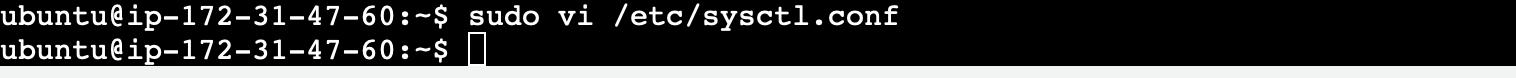
We can set the swappiness to a different value by using the sysctl command.

For instance, to set the swappiness to 10, we could type:

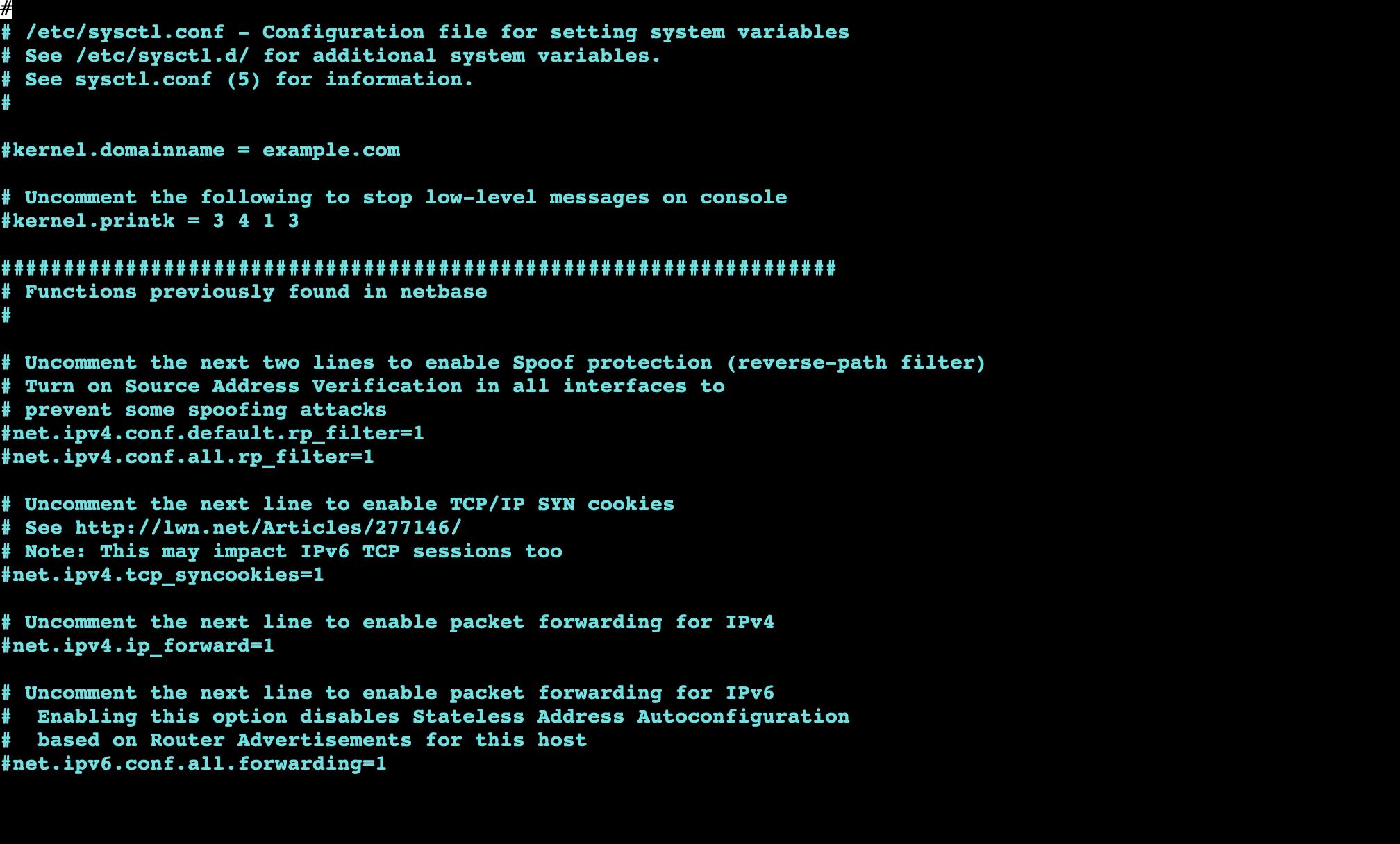
sudo sysctl vm.swappiness=10



This setting will persist until the next reboot. We can set this value automatically at restart by adding the line to our /etc/sysctl.conf file:

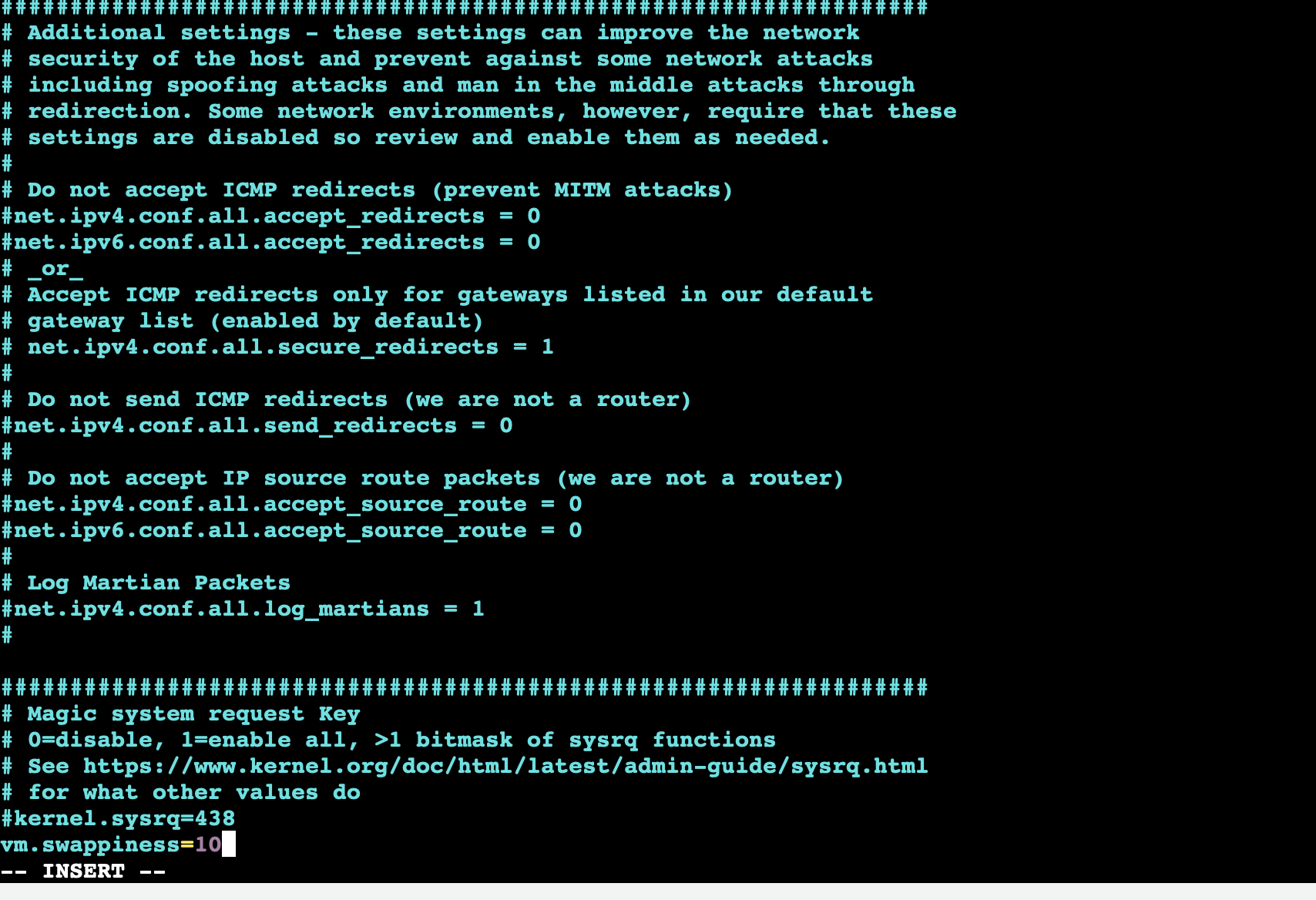


A text editor is open as below



Insert the following command at the end of the file, then save it.

vm.swappiness=10



Save and close the file

**8.Installing the "stress" Tool: Install the stress tool to simulate heavy CPU load for stress testing.**

I am using the stress tool here to increase CPU utilization

(The stress tool is a command-line utility used for testing the robustness of computer systems, particularly their stability under heavy loads. It's designed to simulate high CPU, memory, I/O, and disk usage to stress-test the system and uncover potential weaknesses or issues)

Steps to install stress tool

**sudo apt update**



sudo apt-get install stress



**9.Conducting Stress Testing: Use the stress tool to increase CPU utilization and observe system behavior.**

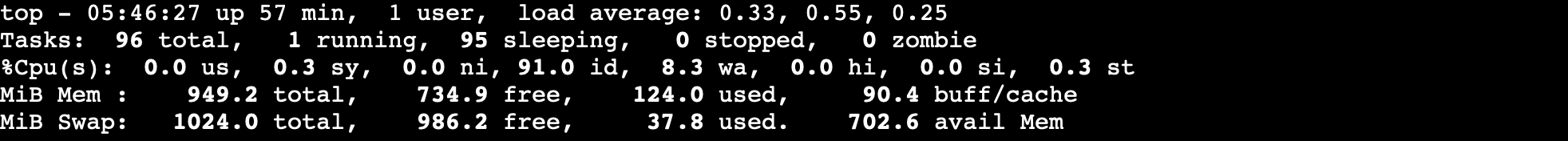
command:

**stress --cpu 8 --io 4 --vm 4 --vm-bytes 1024M --timeout 01m**

This command aims to increase CPU utilization to more than 75%.

Monitor the system to observe the transfer of load from RAM to swap memory using top command.

the resulting readings.

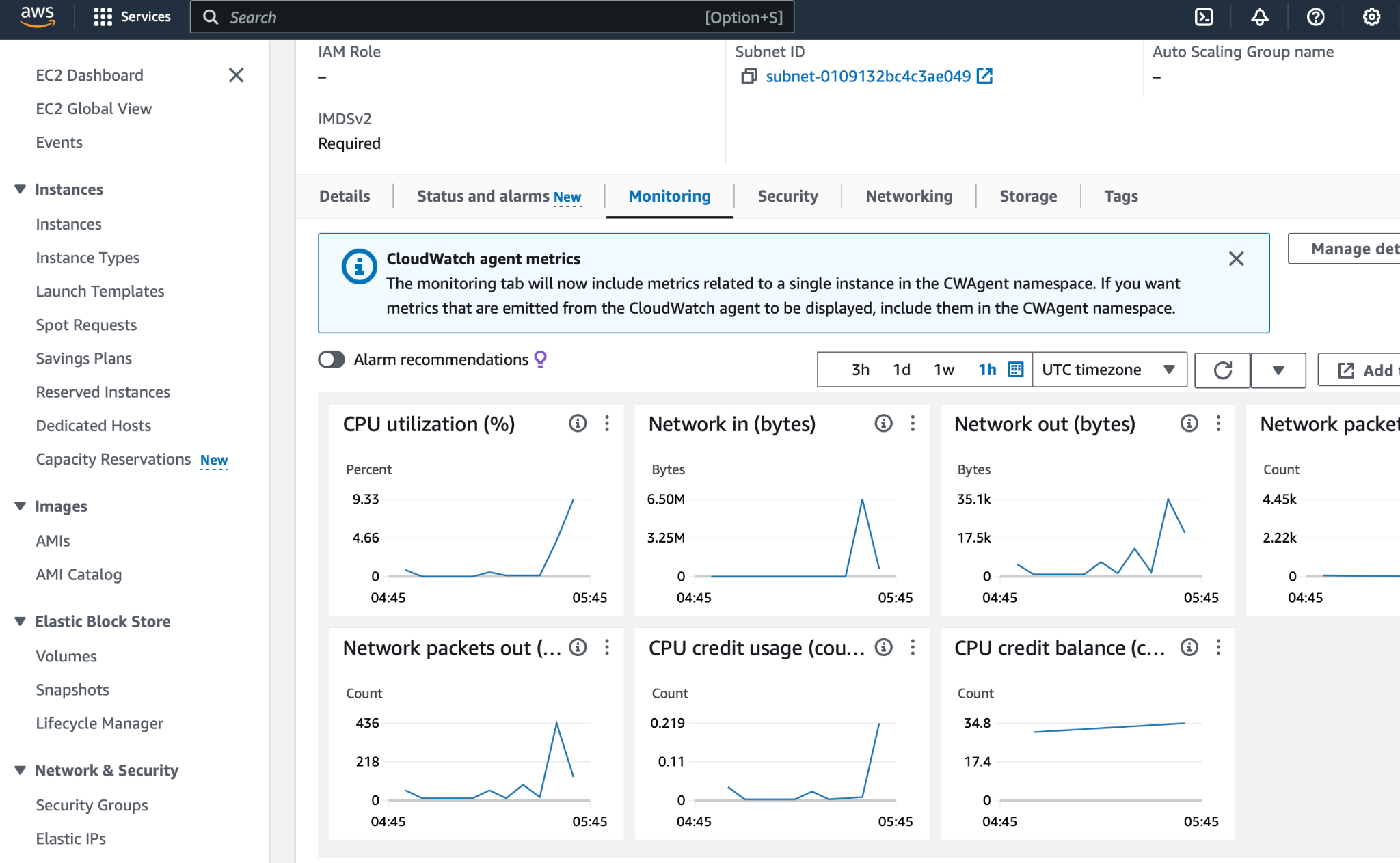


**10.Monitoring System Performance: Utilize AWS EC2 Dashboard under the Monitoring tab to monitor CPU utilization.**

As we can see the swap memory has taken load off ram memory and the readings are stable.



If you look at the AWS EC2 Dashboard under the Monitoring tab, you can see that the CPU utilization has increased



**Conclusion:**

Configuring swap memory on Linux servers, especially in cloud environments like AWS EC2, is essential for optimizing system performance and ensuring stability under varying workloads. By following the steps outlined in this guide, you can effectively manage system memory and enhance overall server performance. Additionally, stress testing provides valuable insights into system behavior under heavy loads, enabling proactive performance optimization.