**Supervisión de Procesos y Optimización de Rendimiento en Linux**

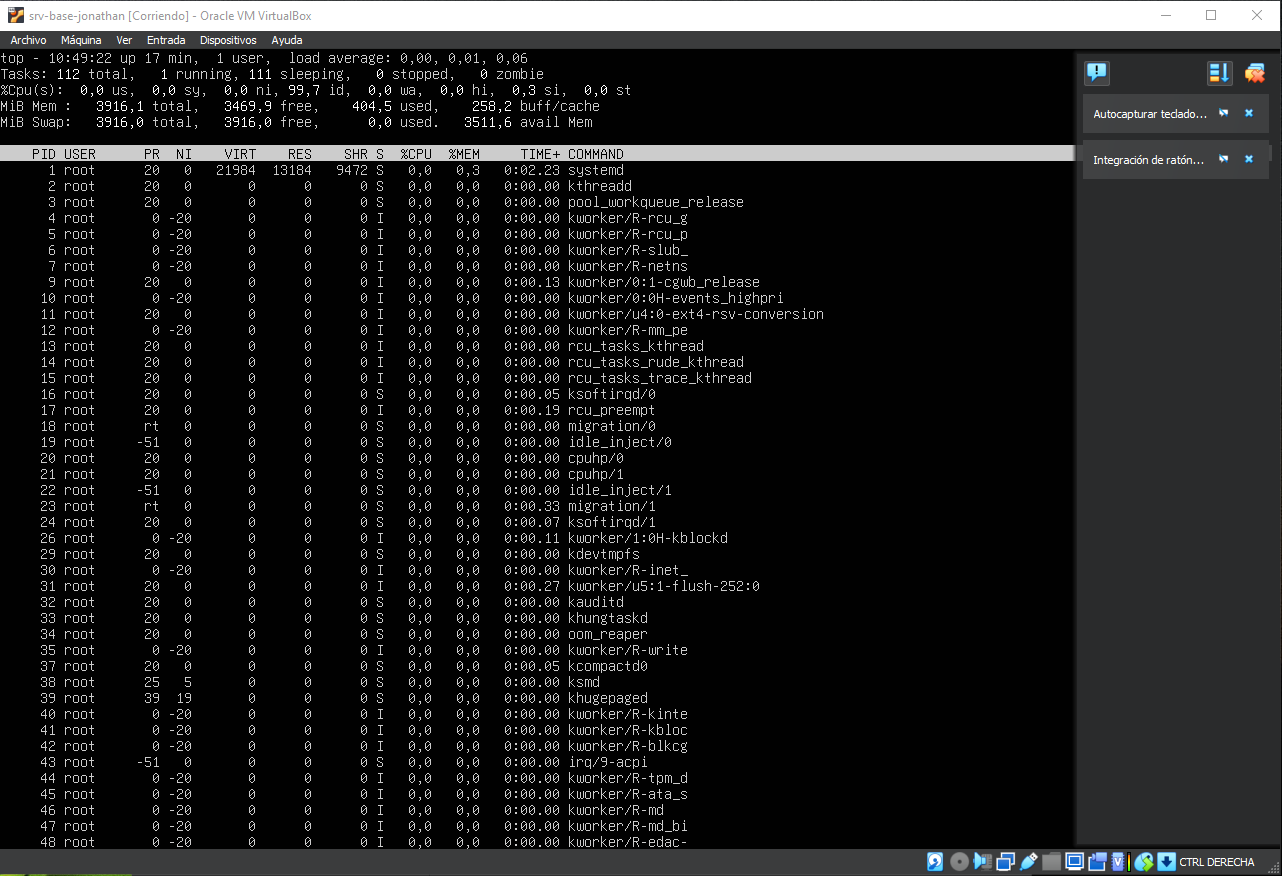
### 

# **Fase 1: Análisis en tiempo real del sistema**

**1.1. Usos de procesos y consumos**

Con el siguiente comando vamos a poder ver todos los recursos activos y el uso que están haciendo de cpu y memoria.

**top**

****

Podemos comprobar todos los procesos con sus respectivos consumos de cpu y memoria, systemd es el que más memoria está consumiendo y que por el momento la cpu no hay recursos usados.

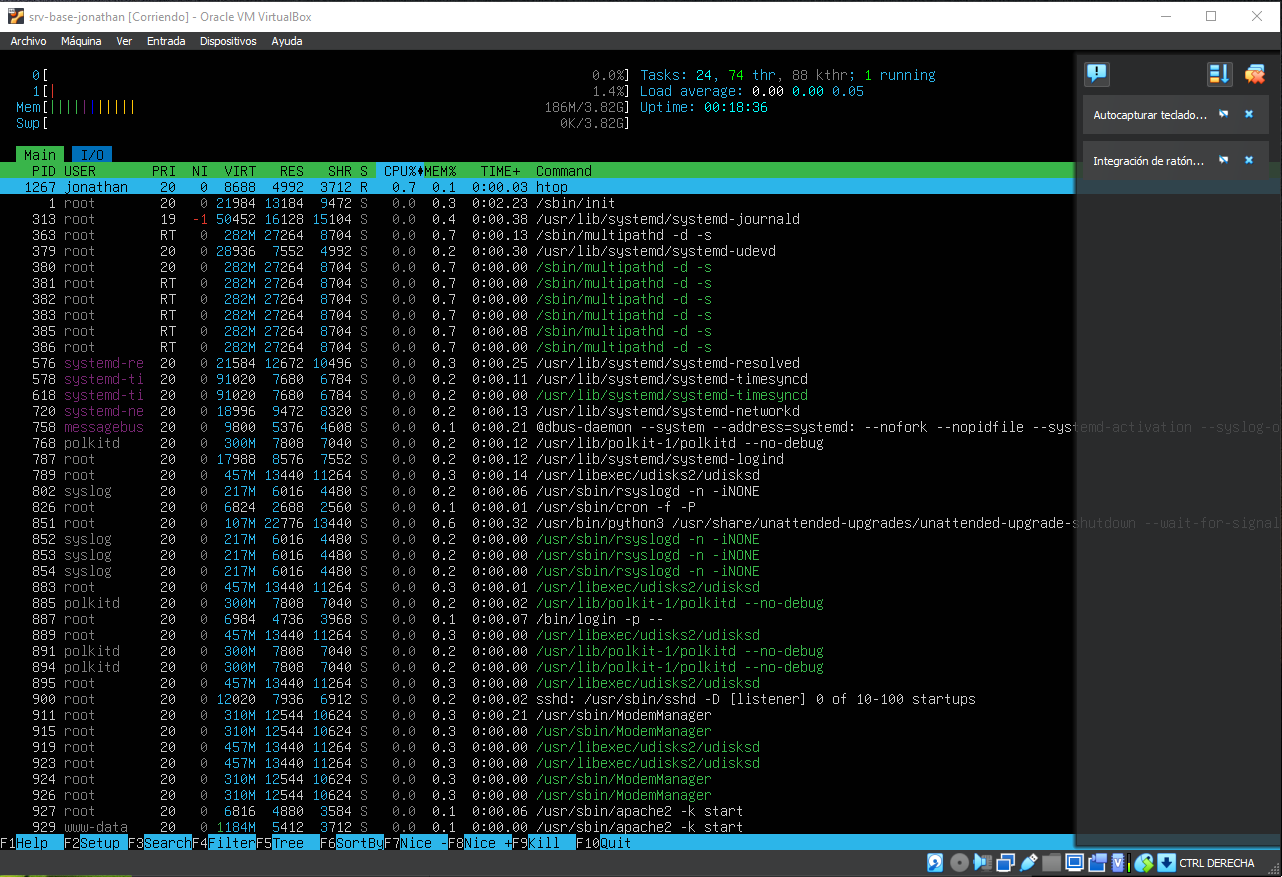
**1.2. Instalación y uso de htop**

Si no tenemos instalado htop los instalaremos con el siguiente comando:

sudo apt install htop

y ahora ejecutaremos el comando:

htop

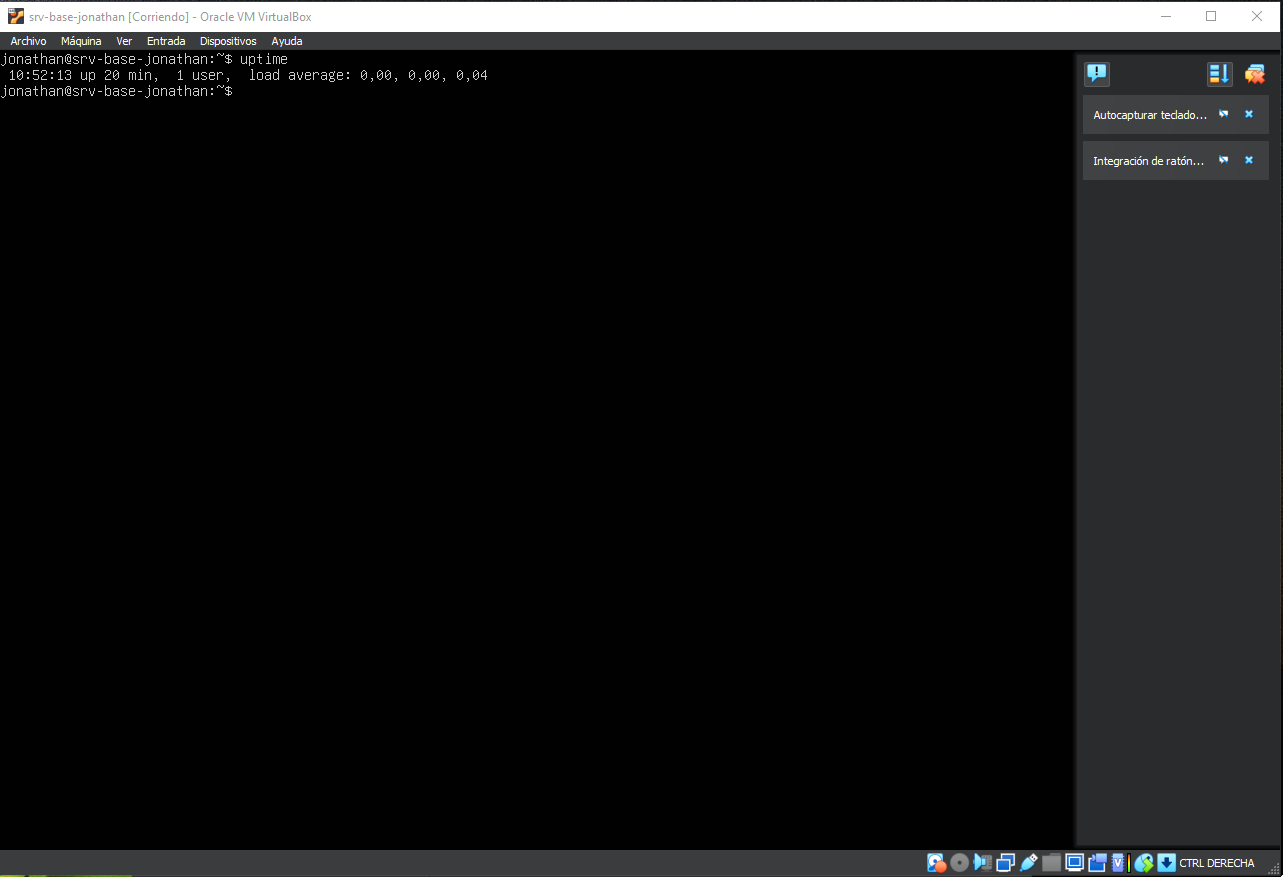


Podemos comprobar que la cpu al ejecutar htop es la que más recursos está teniendo y que multipathd es el que más memoria está consumiendo

**1.3. Tiempo encendido del sistema y carga promedio**

Vamos a comprobar el tiempo que lleva encendido el sistema desde que estamos trabajando con el y la carga de promedio que ha hecho:

**uptime**

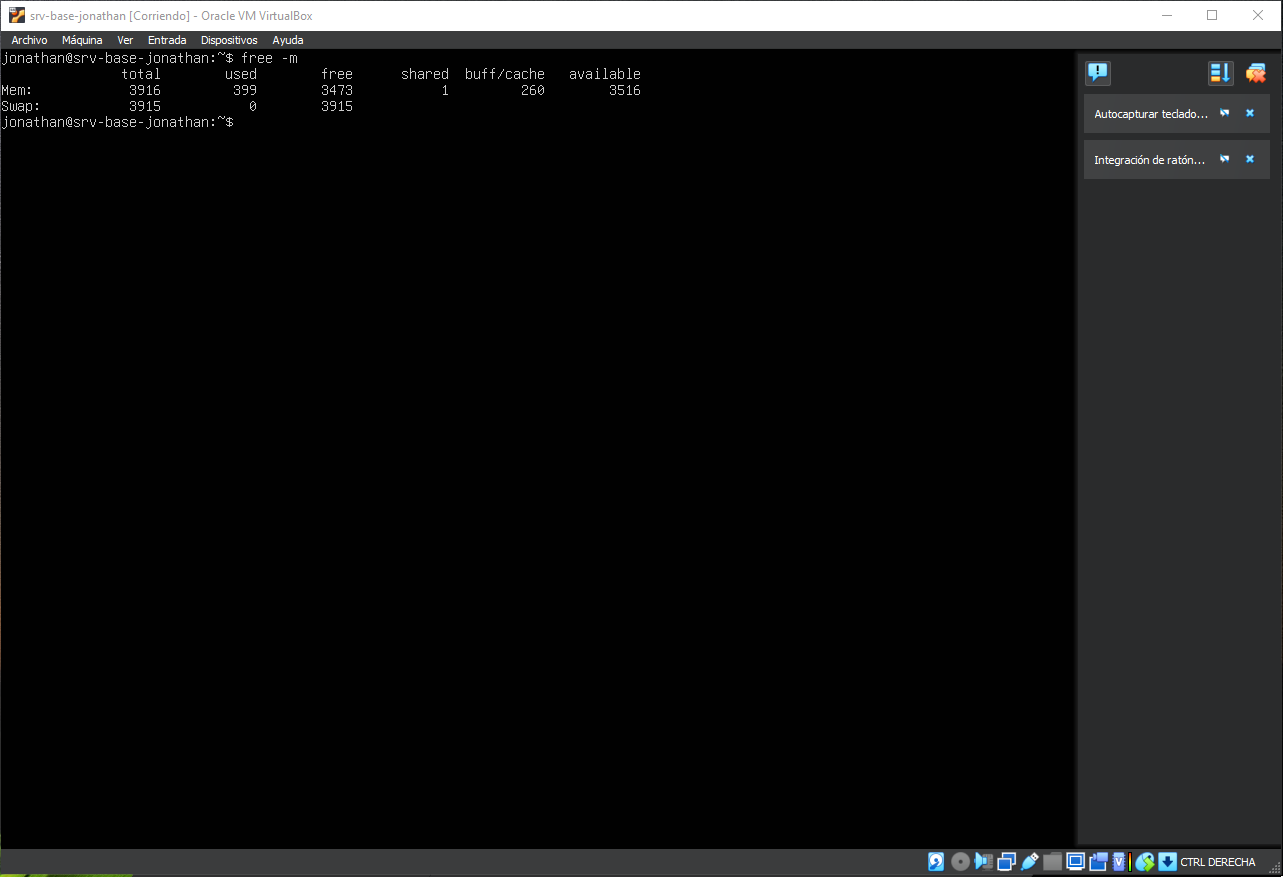


Como podemos comprobar en load average no tiene una carga elevada ya que lleva poco tiempo el sistema encendido.

**1.4. Uso de memoria**

Vamos a comprobar la memoria que estamos usando la total y la que tenemos libre:

free -m

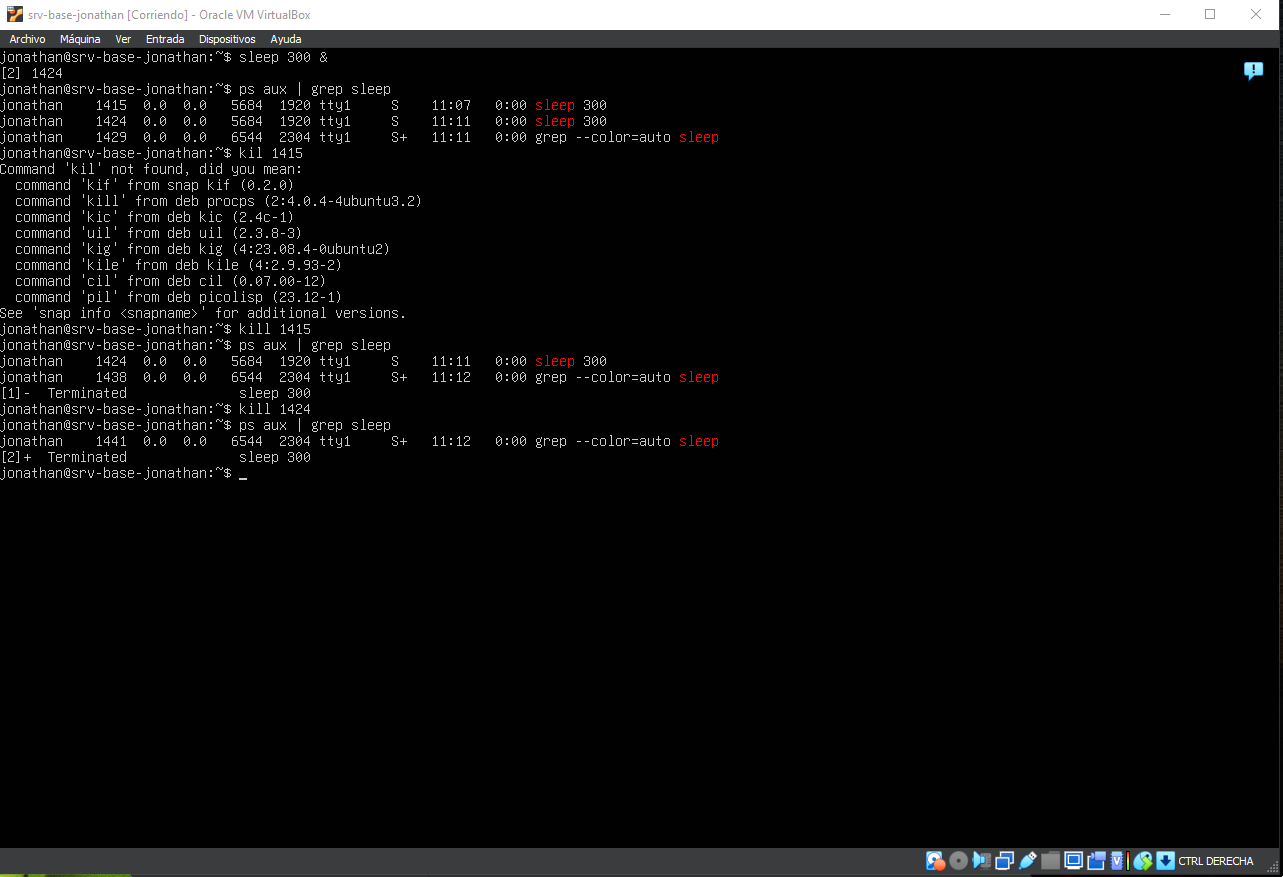


# Fase 2: Gestión activa de procesos

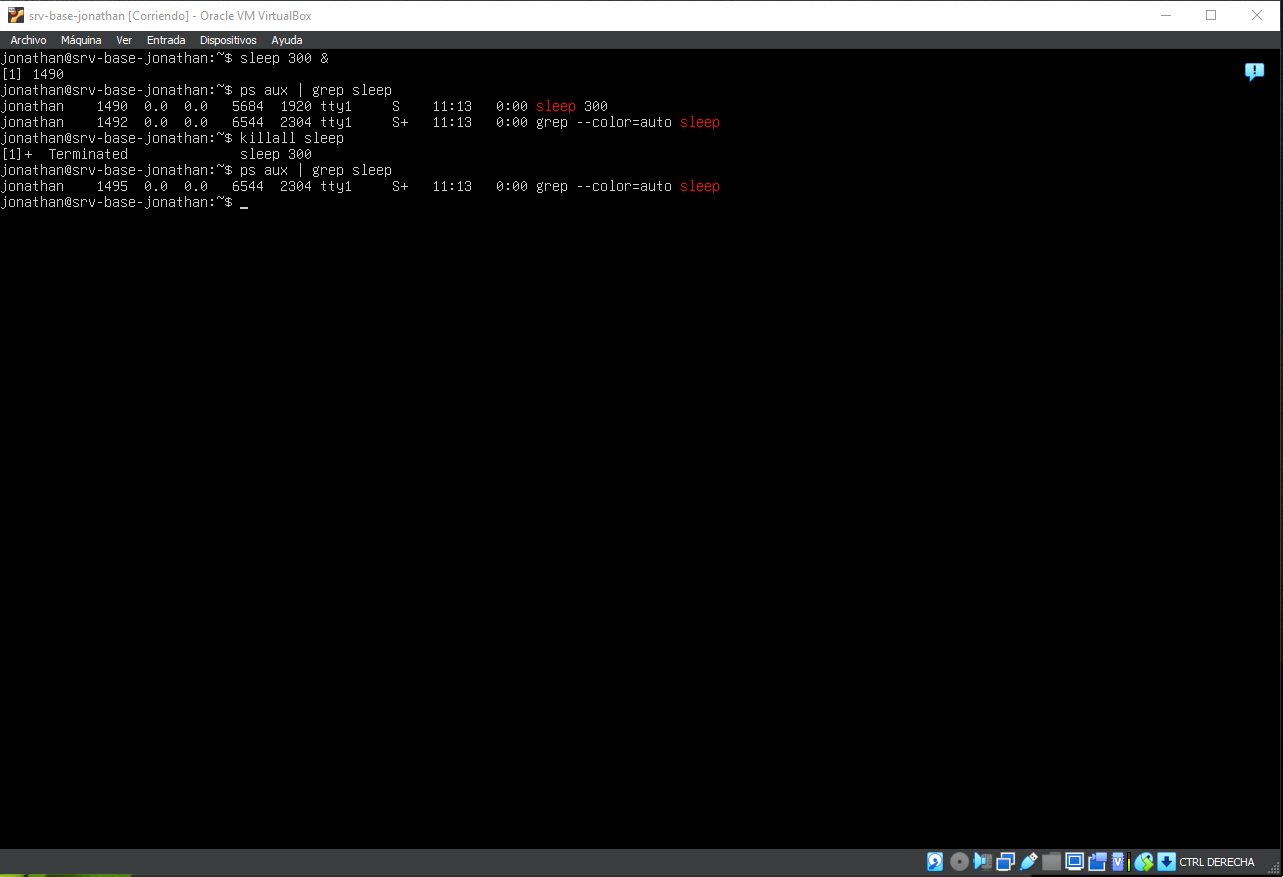
**2.1. Finalizar un proceso no esencial (kill, killalll o pkill)**

Para matar un proceso no esencial primero tendremos que levantar uno como por ejemplo, sleep 300:

* sleep 300 &
* ps aux | grep sleep
* kill <1415>

****

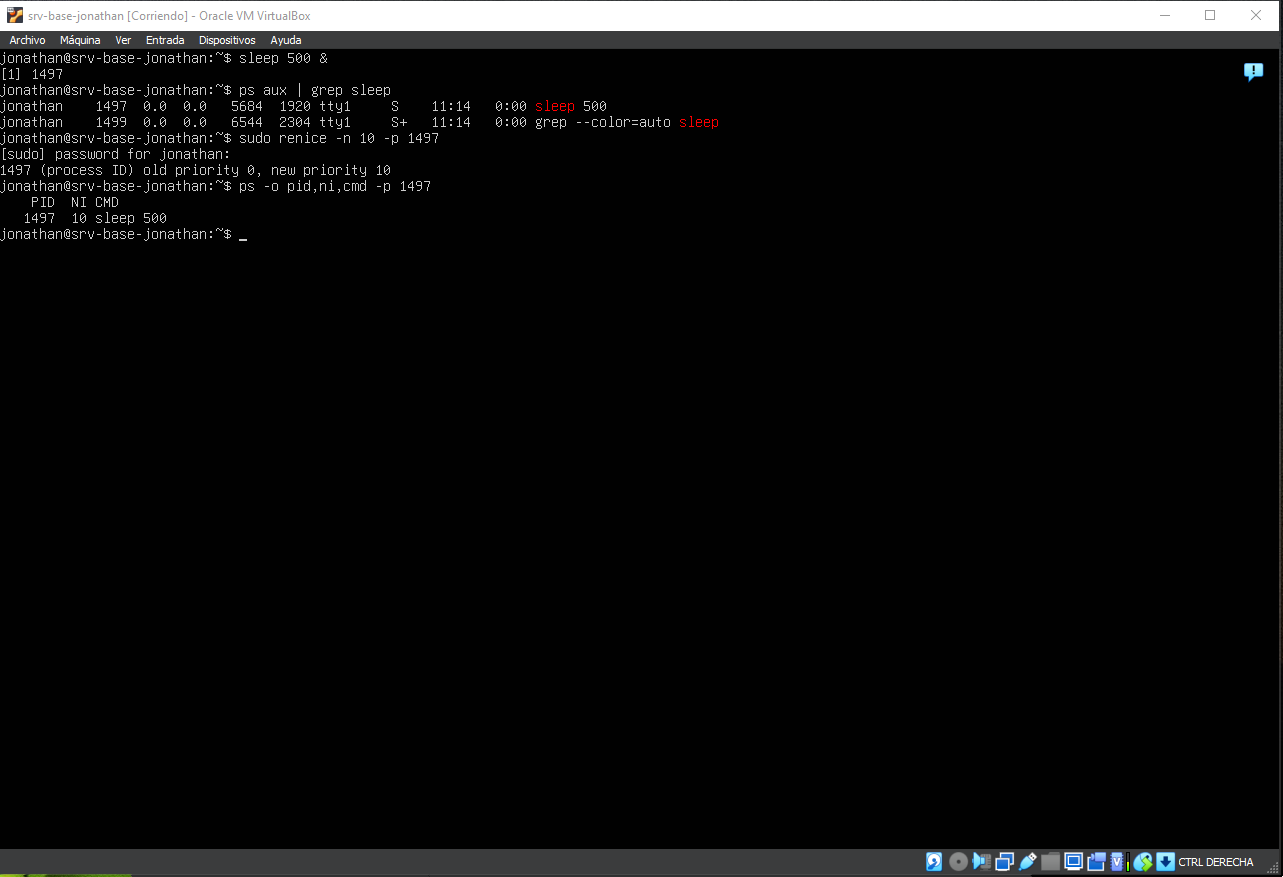
* killall sleep

****

**2.2. Cambiar la prioridad de un proceso en ejecución con renice.**

Vamos a levantar un nuevo proceso y a cambiarle la prioridad con el comando renice.

* sudo renice -n 10 -p 1497

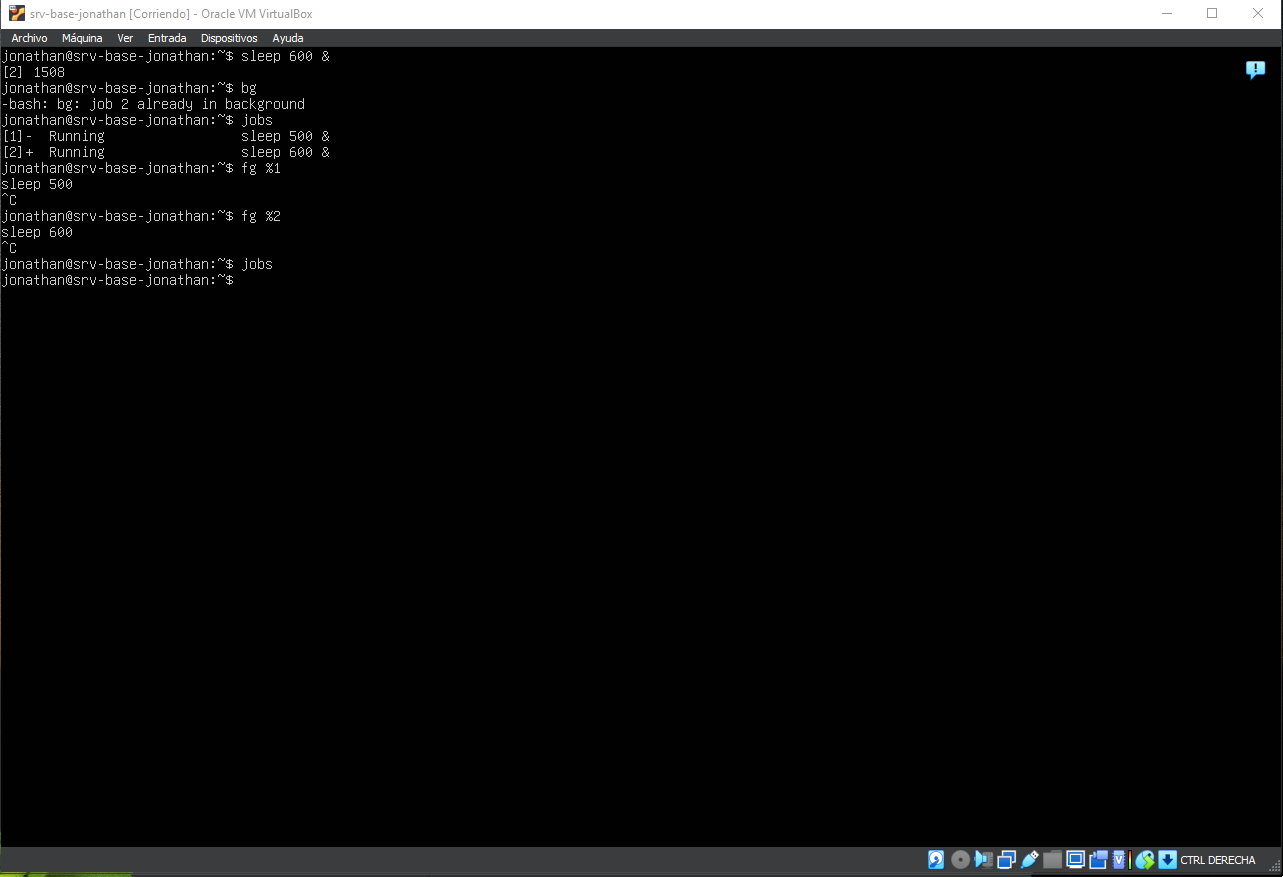
****

Esto baja la prioridad del proceso.

**2.3. Lanzar un proceso en segundo plano (&) y enviarlo al primer plano con fg.**

Vamos a lanzar otro proceso pero esta vez en segundo plano para traerlo de vuelta al primero.

* sleep 600 &
* bg
* jobs
* fg %2

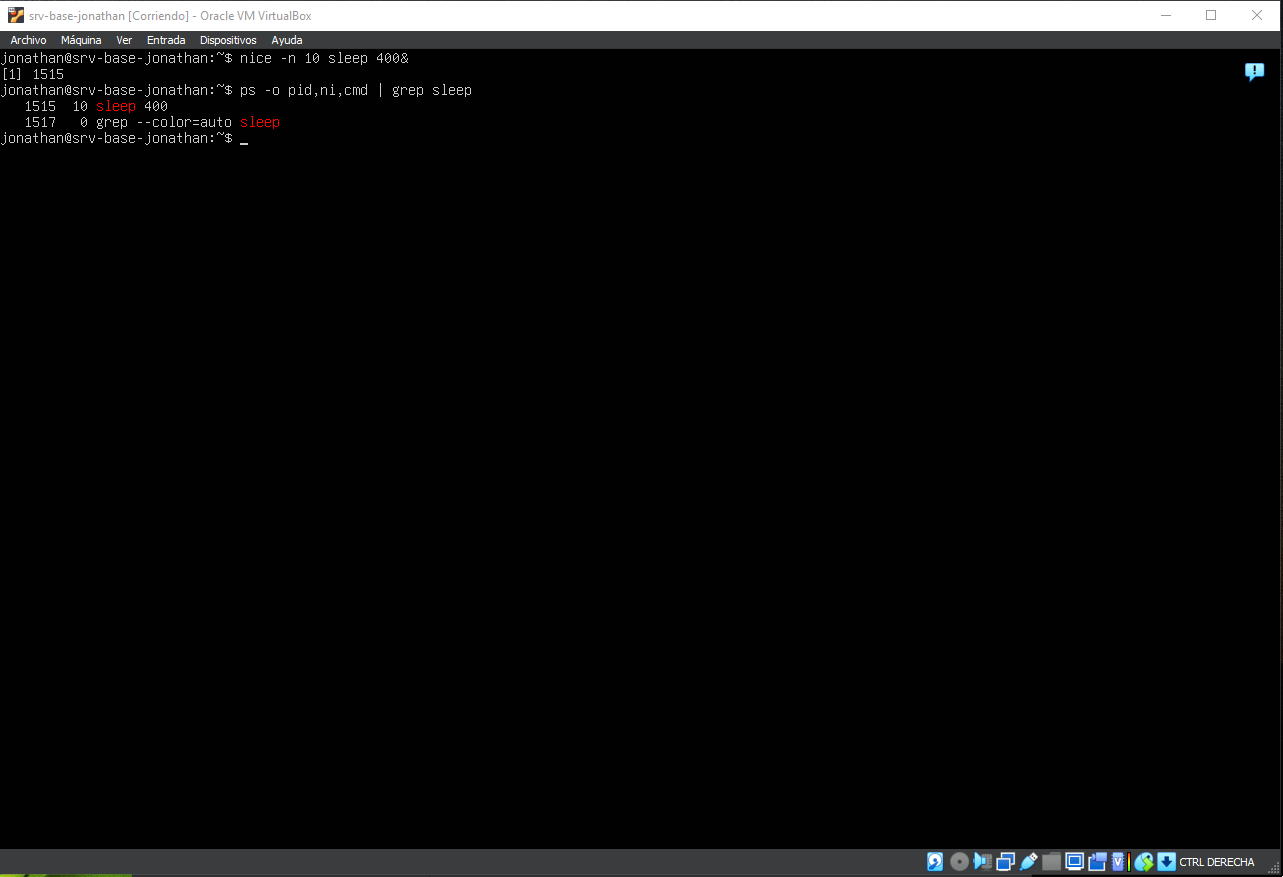


con el comando fg %2 trae el proceso número 2 al frente ya que teníamos dos procesos.

**2.4. Usar nice para iniciar un proceso con prioridad baja**

Usar el comando nice para dar menos prioridad al proceso directamente.

* nice -10 sleep 400&



# Fase 3: Monitorización y registro del uso de recursos

**3.1. Registrar estadísticas con vmstat**

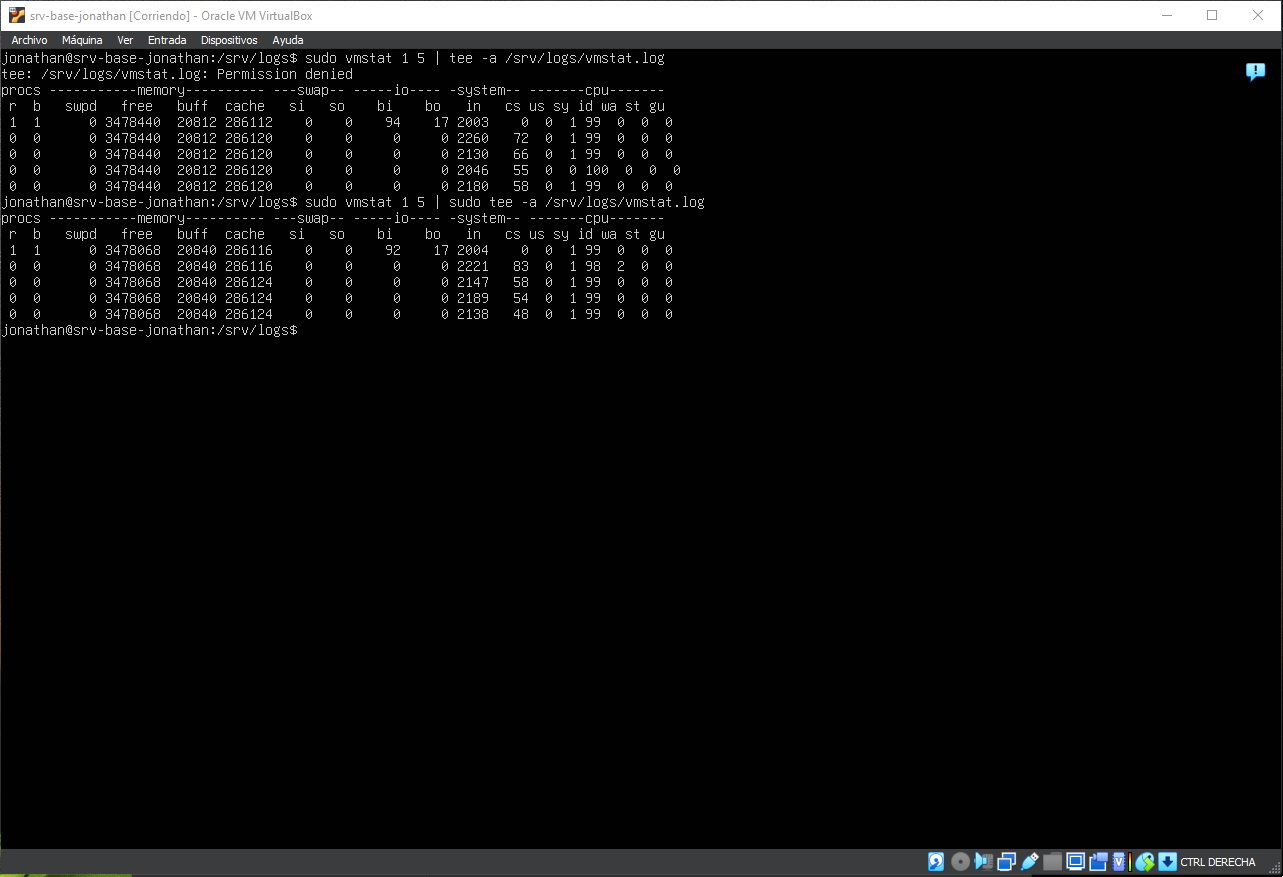
Para registrar estadísticas vamos a usar el comando vmstat y a parte vamos a guardar su salida en un archivo /srv/logs/vmstat.log.

Si no tenemos la carpeta creada la creamos con:

* sudo mkdir -p /srv/logs

Y luego ejecutamos vmstat:

* vmstat 1 5 | sudo tee -a /srv/logs/vmstat.log



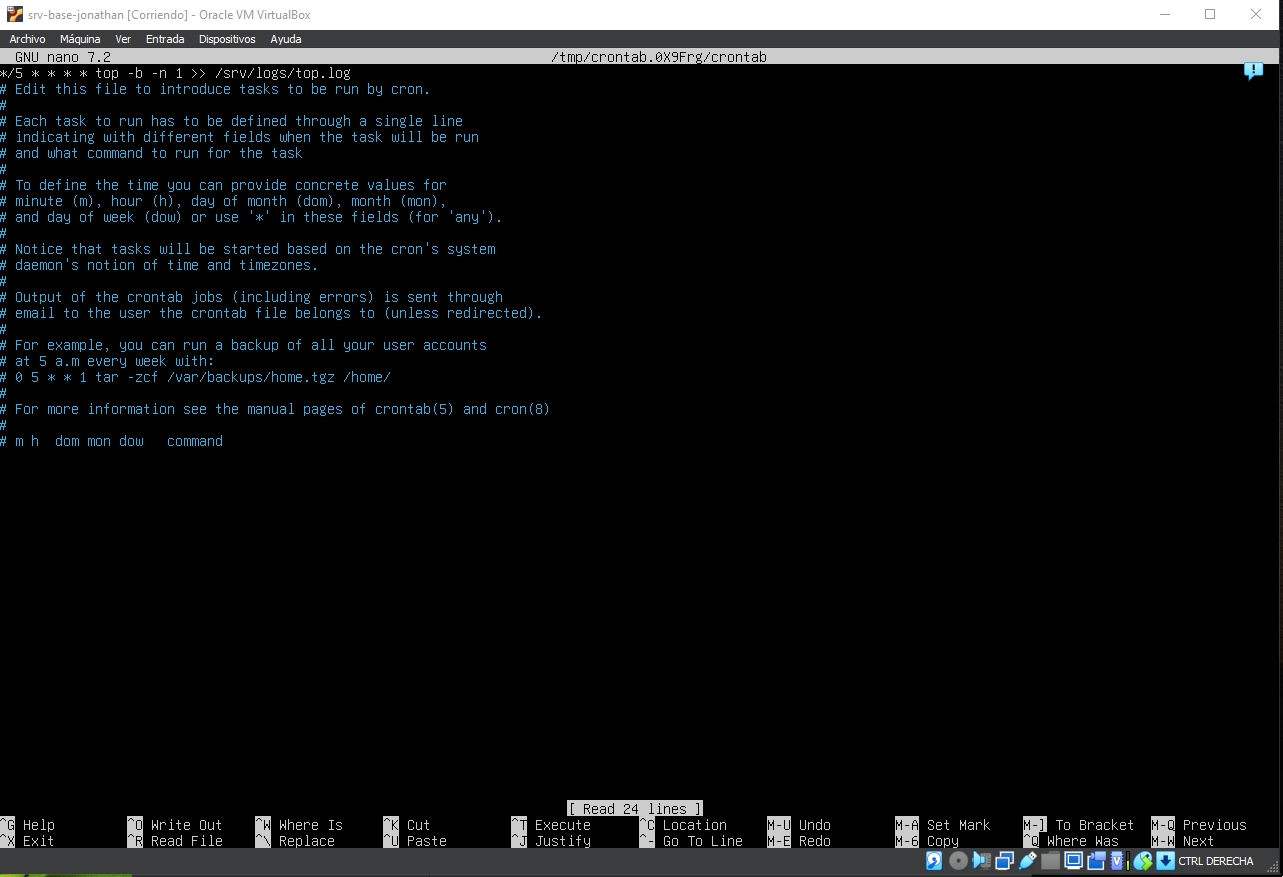
**3.2. Programar tarea en crontab**

Vamos a programar una tarea en crontab para que capture top cada 5 minutos:

* crontab -e

y ahora añadimos la linea:

* \*/5 \* \* \* \* top -b -n 1 >> /srv/logs/top.log



Guardamos y salimos, ya tendremos la tarea programada y esto agrega automáticamente una captura cada 5 minutos.

**3.3. Explorar iotop (si está instalado)**

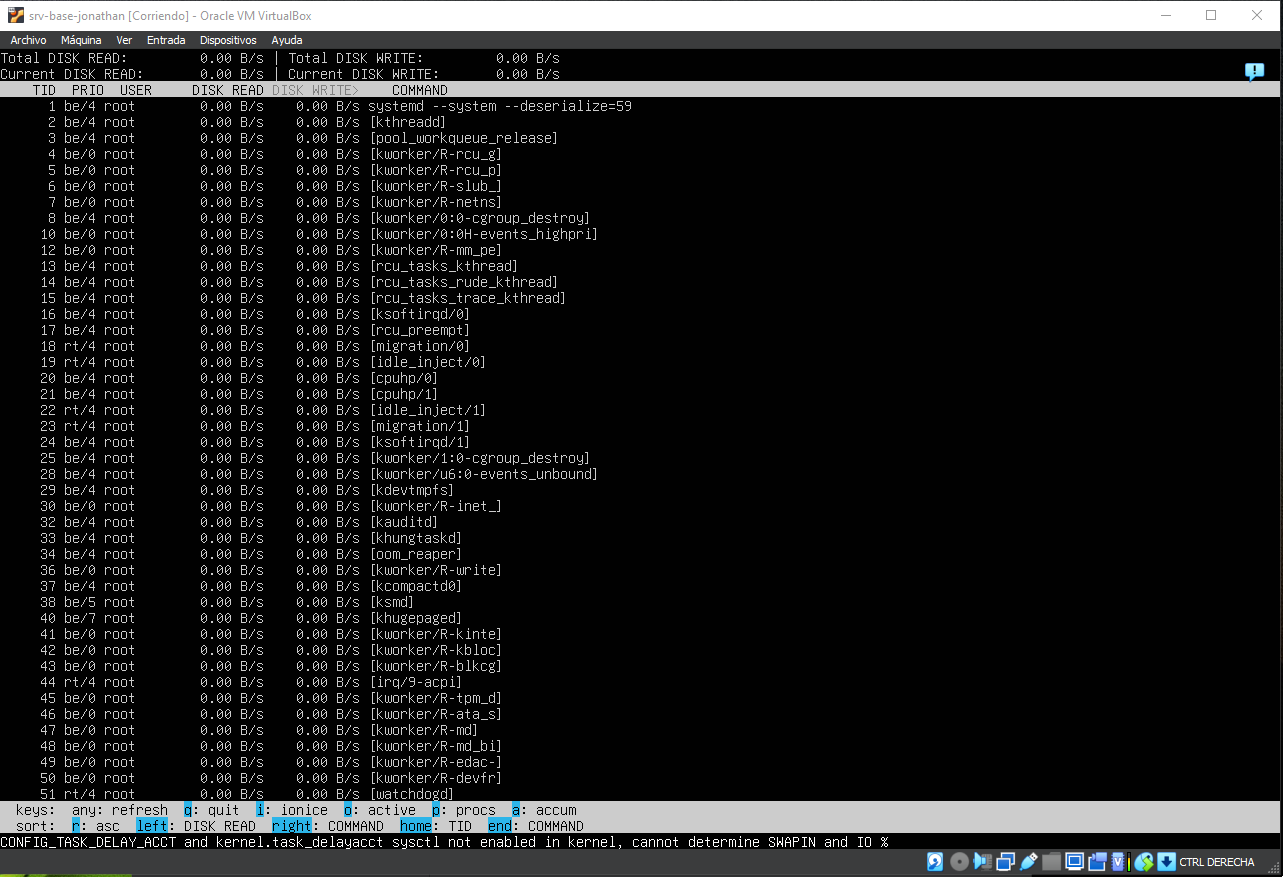
Vamos a ejecutar iotop para ver los procesos que hacen uso del disco:

Si no lo tenemos instalado ejecutamos:

* sudo apt install iotop

y ahora lo ejecutamos

* sudo iotop



Aquí observamos todos los procesos que tiene el disco.

# Fase 4: Simulación de carga controlada

4.1. Vamos a instalar stress o stress-ng para hacer una prueba de estrés, para ello ejecutamos los siguientes comandos:

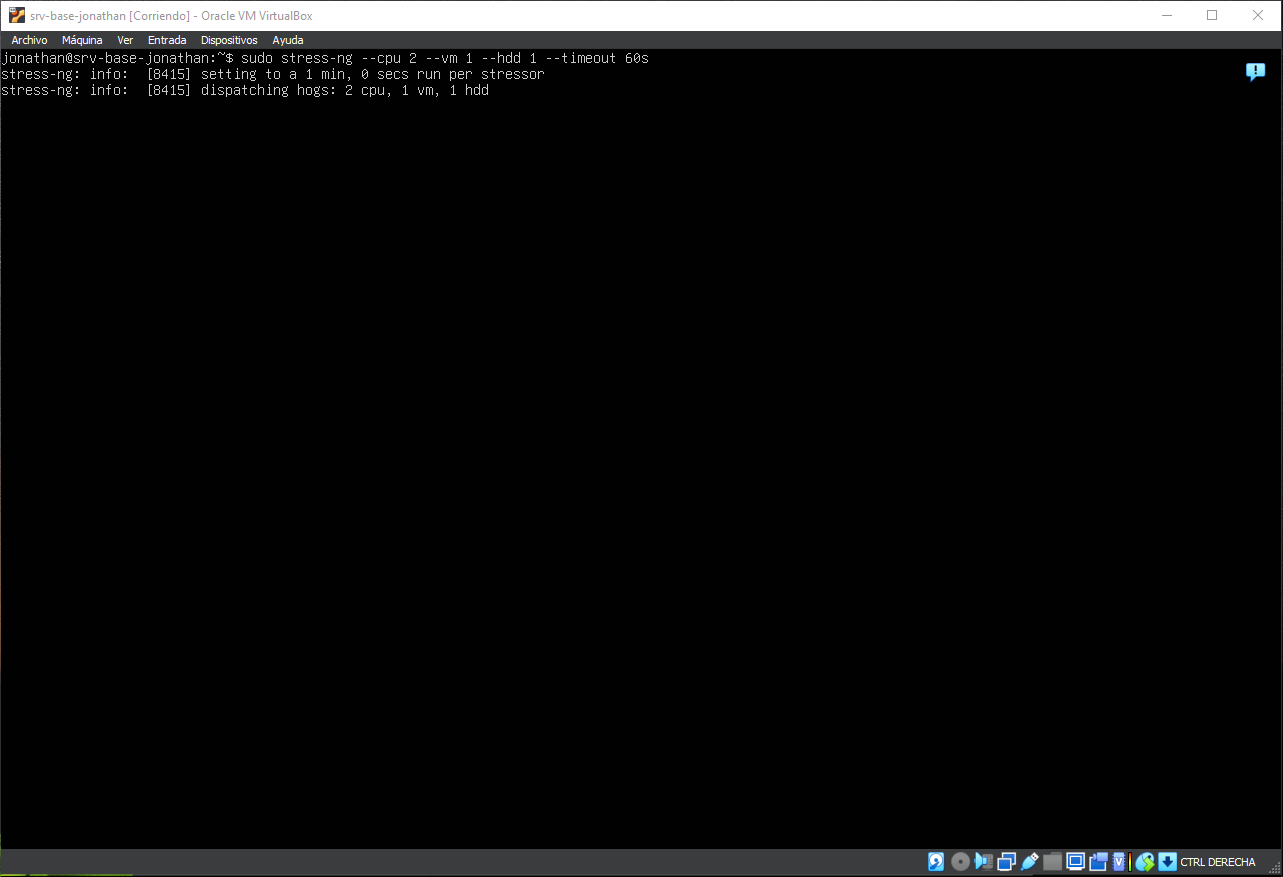
* sudo apt install stress
* sudo apt install stress-ng

este último es más visual como entorno gráfico.

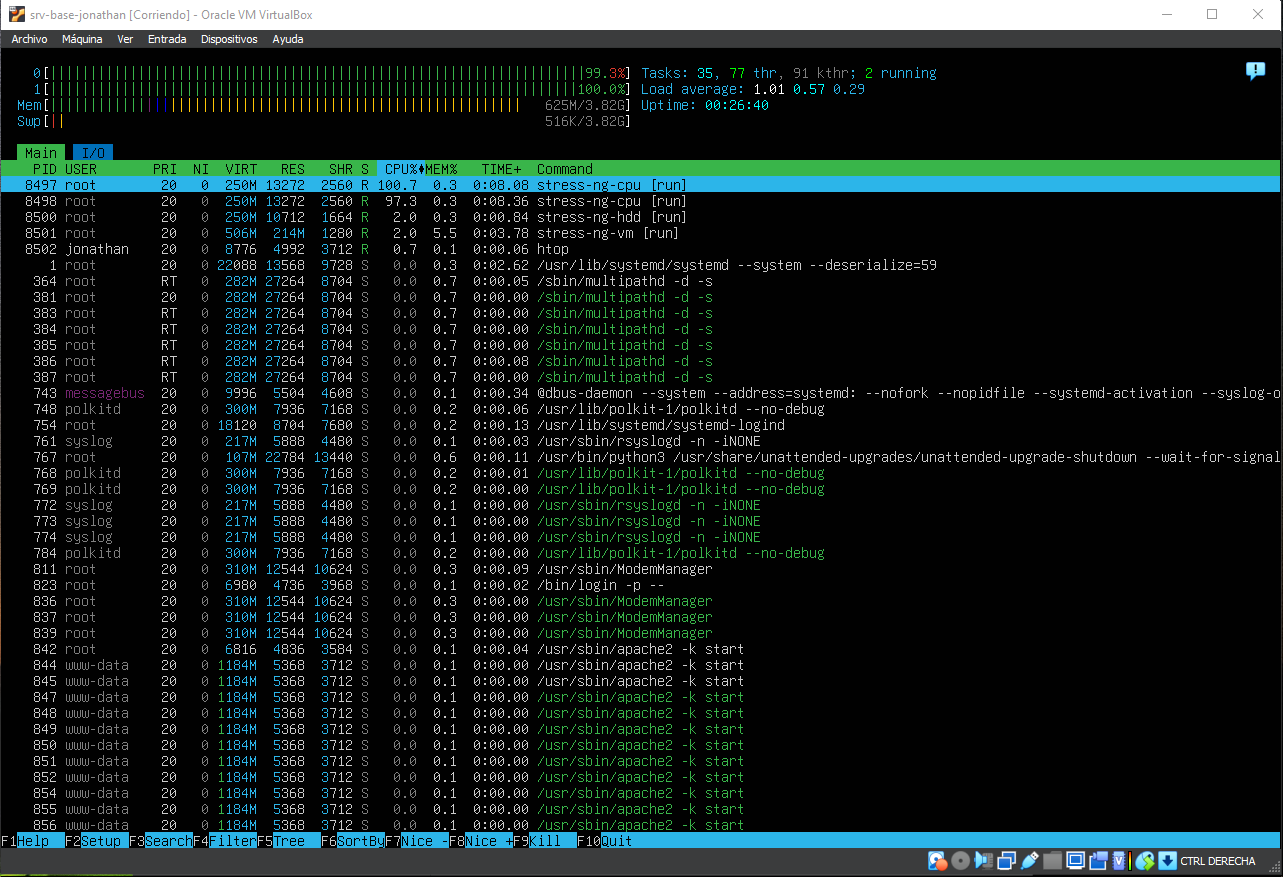
4.2. Ejecutamos una prueba de estrés por 60 segundos:

* sudo stress-ng --cpu 2 --vm 1 --hdd 1 --timeout 60s

Con este comando le decimos que queremos estresar 2 núcleos de la cpu 1 de la memoria virtual 1 disco, durante 60 segundos.



Con otra terminal abierta ejecutamos htop y observaremos que la prueba está en marcha



Podemos comprobar que todo está al 100% de rendimiento.