

Proyecto Sistemas Operativos 2° Corte

Juan Manuel Beltran Mendez, Kevin Andres Leon Tarapues, Angie Tatiana Ruiz Ruiz

Abril 2025

El objetivo de esta práctica es la creación de un entorno híbrido de pruebas en el que se instalarán tres sistemas operativos en máquinas virtuales QEMU de distribuciones generales. Además, se crearán tres contenedores Docker con imágenes de otros tres sistemas operativos generales. Un contenedor adicional se configurará como sistema central encargado de la supervisión y monitoreo de los servidores a través de herramientas de monitoreo como **Grafana** y **Zabbix**.

Creación

iniciamos con la figura 1 donde se ve la creación de la imagen de disco virtual manjaro de 10 GB en formato QCOW2, luego se realiza la instalación de la máquina virtual con virt-install de una VM llamada “manjaro-linux” con 2 GB RAM, 2 vCPUs y red NAT, luego procede el instalador de Manjaro desde una ISO montada en CDROM, este se conecta automáticamente vía VNC para que puedas ver e interactuar con el proceso de instalación y reinicia la máquina una vez completada la instalación para arrancar el sistema recién instalado.

```
juanbeltran@juanbeltran:~$ sudo qemu-img create -f qcow2 /var/lib/libvirt/images/manjaro11.qcow2 10G
[sudo] contraseña para juanbeltran:
Formatting '/var/lib/libvirt/images/manjaro11.qcow2', fmt=qcow2 cluster_size=65536 extended_l2=off compression_type=zlib size=10737418240 lazy_refcounts=off refcount_bits=16
juanbeltran@juanbeltran:~$ virt-install \
--name manjaro-linux \
--ram 2048 \
--vcpus 2 \
--disk path=/var/lib/libvirt/images/manjaro11.qcow2,format=qcow2 \
--os-variant archlinux \
--network network=default \
--graphics vnc \
--cdrom /var/lib/libvirt/images/manjaro-xfce-25.0.0-250414-linux612.iso

Empezando la instalación...
Creando dominio...
Running graphical console command: virt-viewer --connect qemu:///system --wait manjaro-linux

(virt-viewer:17475): virt-viewer-WARNING **: 14:40:45.715: vnc-session: got vnc error Server closed the connection
Creación de dominio completada.
Reiniciando invitado.
Running graphical console command: virt-viewer --connect qemu:///system --wait manjaro-linux
```

Figura 1: Creación de MV

Continuamos con la organización de las carpetas para tener ISO y discos virtuales bien ubicados y se realizó la creación de la máquina virtual "rocky-linux", la cual tiene como características Nombre: “rocky-linux”, RAM: 2 GB, CPUs: 2 núcleos virtuales, Disco: Usa rocky.qcow2, Sistema operativo variante: similar a Rocky 8.5 (por compatibilidad de parámetros), Red: Conectada a la red virtual por defecto (NAT), Gráficos: habilita VNC para conexión gráfica, y CD-ROM: monta la ISO de instalación de Rocky y después se hace la conexión gráfica con virt-viewer.

```

juanbeltran@juanbeltran:~$ sudo mkdir -p /var/lib/libvirt/images/isos
juanbeltran@juanbeltran:~$ sudo mv ~/VMs/rocky.qcow2 /var/lib/libvirt/images/
sudo mv ~/isos/Rocky-9.5-x86_64-minimal.iso /var/lib/libvirt/images/isos/
juanbeltran@juanbeltran:~$ virt-install \
--name rocky-linux \
--ram 2048 \
--vcpus 2 \
--disk path=/var/lib/libvirt/images/rocky.qcow2,format=qcow2 \
--os-variant rocky8.5 \
--network network=default \
--graphics vnc \
--cdrom /var/lib/libvirt/images/isos/Rocky-9.5-x86_64-minimal.iso

Empezando la instalación...
Creando dominio...
Running graphical console command: virt-viewer --connect qemu:///system --wait rocky-linux

(virt-viewer:9603): virt-viewer-WARNING **: 05:29:15.647: vnc-session: got vnc error Server closed the connection
Creación de dominio completada.
Reiniciando invitado.
Running graphical console command: virt-viewer --connect qemu:///system --wait rocky-linux
^Z
[1]+  Detenido                  virt-install --name rocky-linux --ram 2048 --vcpus 2 --disk path=/var/lib/libvirt/images/rocky.qco
w2,format=qcow2 --os-variant rocky8.5 --network network=default --graphics vnc --cdrom /var/lib/libvirt/images/isos/Rocky-9.5-x8
6_64-minimal.iso
juanbeltran@juanbeltran:~$ sudo mv ~/ISOS/manjaro-xfce-25.0.0-250414-linux612.iso /var/lib/libvirt/images/
qemu-img create -f qcow2 /var/lib/libvirt/images/manjaro.qcow2 20G
[sudo] contraseña para juanbeltran:
mv: no se puede efectuar 'stat' sobre '/home/juanbeltran/ISOS/manjaro-xfce-25.0.0-250414-linux612.iso': No existe el archivo o e

```

Figura 2: Creación de MC rocky

Se emite el comando `virsh list --all` para ver las máquinas creadas y activadas

```

juanbeltran@juanbeltran:~$ virsh list --all
 Id    Name               State
-----
 2     rocky-linux        running
 5     manjaro-linux       running
juanbeltran@juanbeltran:~$

```

Figura 3: máquinas creadas

Después de esto como se puede ver en la figura 4 muestra la ejecución de una máquina virtual corriendo el sistema operativo Manjaro Linux 64 bits, versión de kernel 6.12.21-4-MANJARO, utilizando el hipervisor QEMU/KVM. En la captura se visualiza un monitor de sistema abierto dentro de la terminal, proporcionando información detallada sobre el rendimiento del sistema.

Se observa lo siguiente:

- CPU: Frecuencia de 2.00 GHz con un uso actual de aproximadamente 2.8%.
- Memoria RAM: Uso de 38.2% de un total de 2 GB asignados.
- SWAP: Sin uso (0.0% utilizado).
- Load Average: Carga del sistema mínima, indicando un estado de baja actividad.
- Interfaz de red: `enp1s0` con actividad de recepción mínima (520 bytes recibidos).
- Discos: Sin operaciones activas de lectura/escritura (`vda` y `vda1`).

Sistema de archivos:

`/`, `/home`, `/var/cache` y `/var/log` se encuentran montados en una partición de 10 GB, usando aproximadamente 5.1 GB en cada uno.

Procesos activos:

Entre los principales consumidores de CPU y memoria se encuentran procesos como `python`, `xfwm4`, `Xorg`, `msm_notifier`, `xfdesktop` y otros servicios del entorno de escritorio XFCE.

El entorno gráfico de Manjaro muestra accesos rápidos en el escritorio como Carpeta personal, Sistema de archivos y Papelera, lo que confirma que el entorno XFCE está operativo en la máquina virtual.

Esta captura evidencia que la máquina virtual se encuentra funcionando de manera estable, con un uso eficiente de los recursos asignados, y que el sistema operativo ha sido correctamente instalado y configurado.

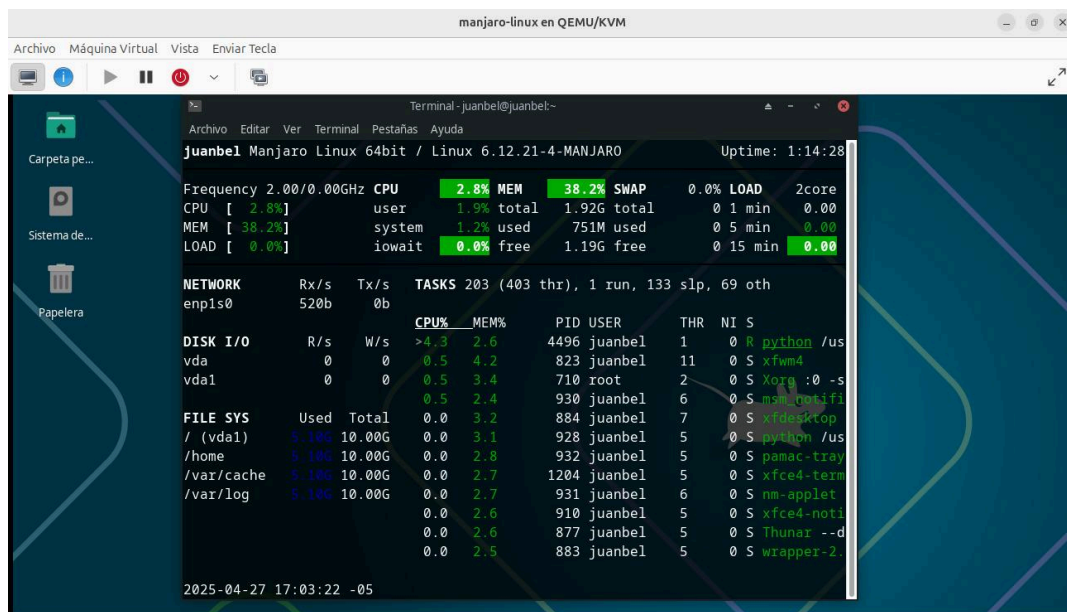


Figura 4:Glances anjaro

Se procede a ver el estado general del sistema:

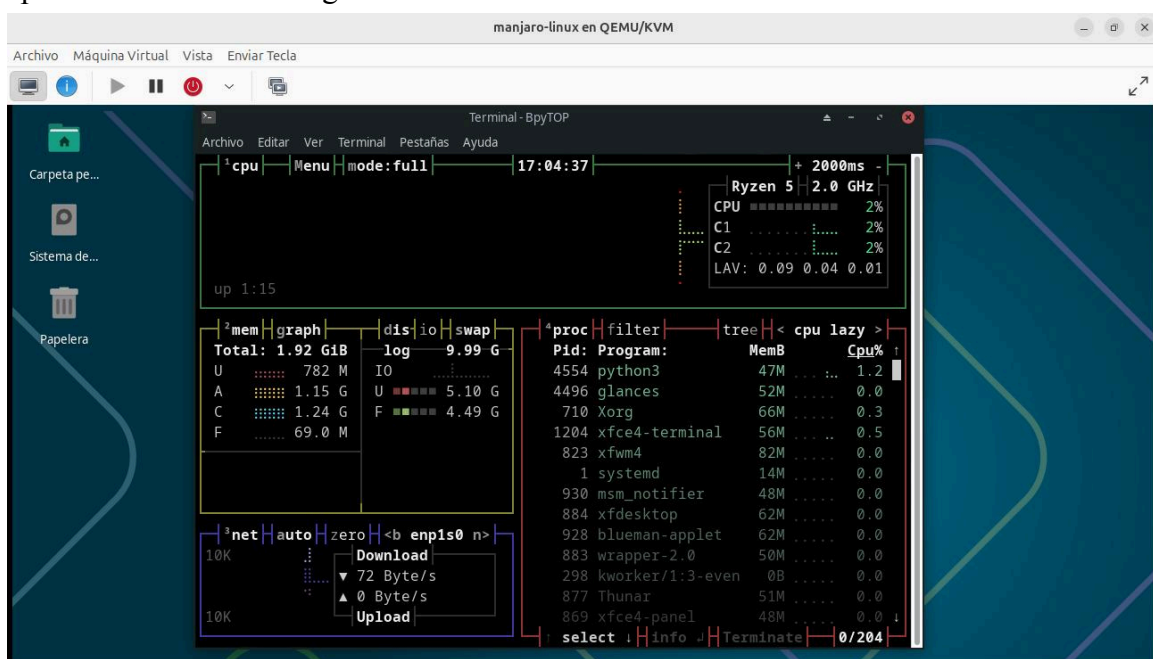


Figura 5:bpytop

Visualizar recursos de manera gráfica:

```
[juanbel@juanbel ~]$ systemctl list-units --type=service
UNIT                                LOAD    ACTIVE SUB    DESCRIPTION
accounts-daemon.service            loaded active running Accounts Servi>
apparmor.service                   loaded active exited Load AppArmor >
avahi-daemon.service               loaded active running Avahi mDNS/DNS>
colord.service                     loaded active running Manage, Instal>
cronie.service                     loaded active running Command Schedu>
cups.service                       loaded active running CUPS Scheduler
dbus-broker.service                loaded active running D-Bus System M>
kmod-static-nodes.service          loaded active exited Create List of>
lightdm.service                    loaded active running Light Display >
lvm2-monitor.service              loaded active exited Monitoring of >
ModemManager.service              loaded active running Modem Manager
NetworkManager.service            loaded active running Network Manager
pamac-daemon.service              loaded active running Pamac Daemon
plymouth-quit-wait.service         loaded active exited Hold until boo>
plymouth-quit.service             loaded active exited Terminate Plym>
plymouth-read-write.service        loaded active exited Tell Plymouth >
plymouth-start.service            loaded active exited Show Plymouth >
polkit.service                    loaded active running Authorization >
qemu-guest-agent.service           loaded active running QEMU Guest Age>
rtkit-daemon.service              loaded active running RealtimeKit Sc>
```

Figura 6:systemctl list-units --type=service

Listar servicios activos:

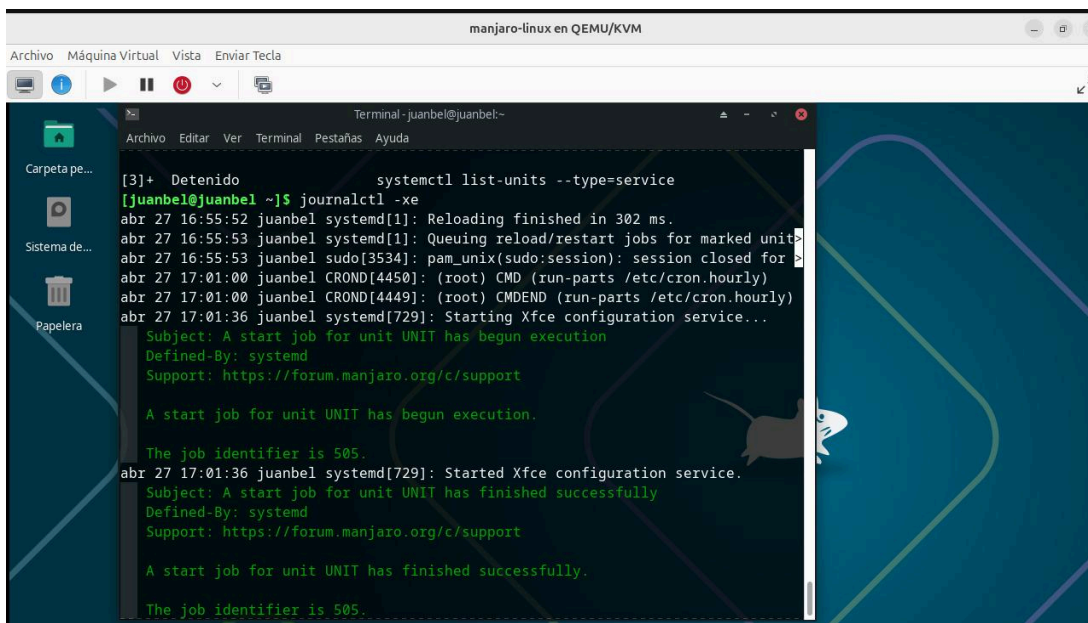


Figura 7: journalctl -xe

Para ver los eventos del sistema: analizar servicios y procesos en manjaro

Continuamos con la figura 8 se muestra la pantalla de selección de interfaces de red en Wireshark, donde se deben seguir los siguientes pasos:

1. **Acceso a Wireshark:** Abrir Wireshark en el sistema operativo anfitrión o en el contenedor/servidor de monitoreo según se haya implementado.
2. **Selección de interfaz de red:**

- Se visualiza una lista de interfaces disponibles.
- Se debe identificar la interfaz activa para capturar tráfico real. En este caso, la interfaz **enp1s0** ha sido seleccionada, que corresponde a una conexión Ethernet.
- Al seleccionarla, se muestra un pequeño gráfico de tráfico en tiempo real, indicando actividad de red.
- Además, se visualizan las direcciones IP asociadas: **IPv4: 192.168.122.163** e **IPv6: fe80::ad0b:ab16:3bb0:782c**.

3. Configuración adicional:

- Se ha activado la opción "**Activar modo promiscuo en todas las interfaces**", lo cual permite capturar todo el tráfico que pase por la interfaz, no solo el dirigido a la máquina local.
- Se puede definir un **filtro de captura** si se desea restringir los paquetes capturados a ciertos protocolos o direcciones específicas.

4. Iniciar captura:

- Una vez seleccionada la interfaz correcta, se procede a dar clic en el botón "**Iniciar**" para comenzar la captura de paquetes de red.

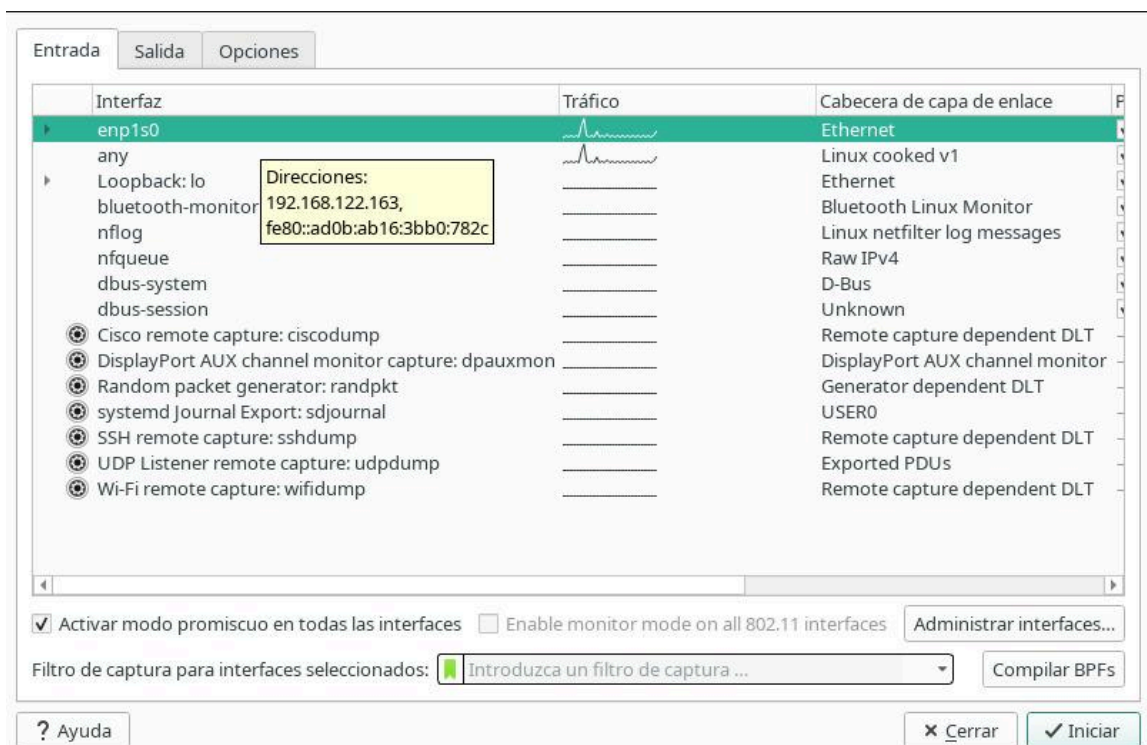


Figura 8: interfaz de red en Wireshark para monitoreo de red

Procedemos a ver el análisis de tráfico de red en Wireshark se observan paquetes del **protocolo STP (Spanning Tree Protocol)**, utilizado generalmente para prevenir bucles en redes de nivel 2 (capa de enlace de datos).

También se capturan paquetes de **SSDP (Simple Service Discovery Protocol)**, que permiten descubrir dispositivos en redes locales (por ejemplo, impresoras, routers, dispositivos IoT).

En la parte inferior de la ventana se muestra el **detalle del paquete seleccionado**, dividido en tres secciones:

- **Resumen de encabezados Ethernet e IEEE 802.3.**
- **Información de control de enlace lógico (LLC).**
- **Datos del protocolo específico capturado (STP).**

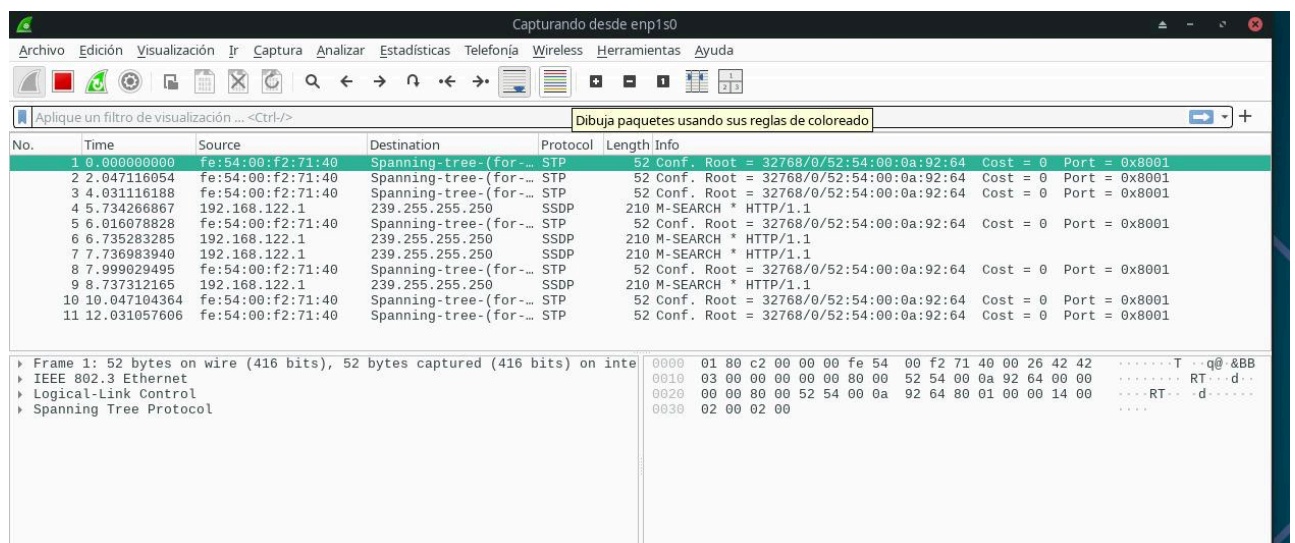


Figura 9: análisis de tráfico de red en Wireshark

Procedemos a ver el monitoreo de tráfico de red en tiempo real desde terminal donde en la figura 10 donde la terminal para monitorear el tráfico de red en tiempo real.

Detalles principales:

- IP detectada: 192.168.122.255 (broadcast de la red).
- Destino: _gateway (puerta de enlace de la red).

Tráfico:

- Transmitido (TX): 218 bytes.
- Recibido (RX): 1,83 KB.
- Total: 2,04 KB.

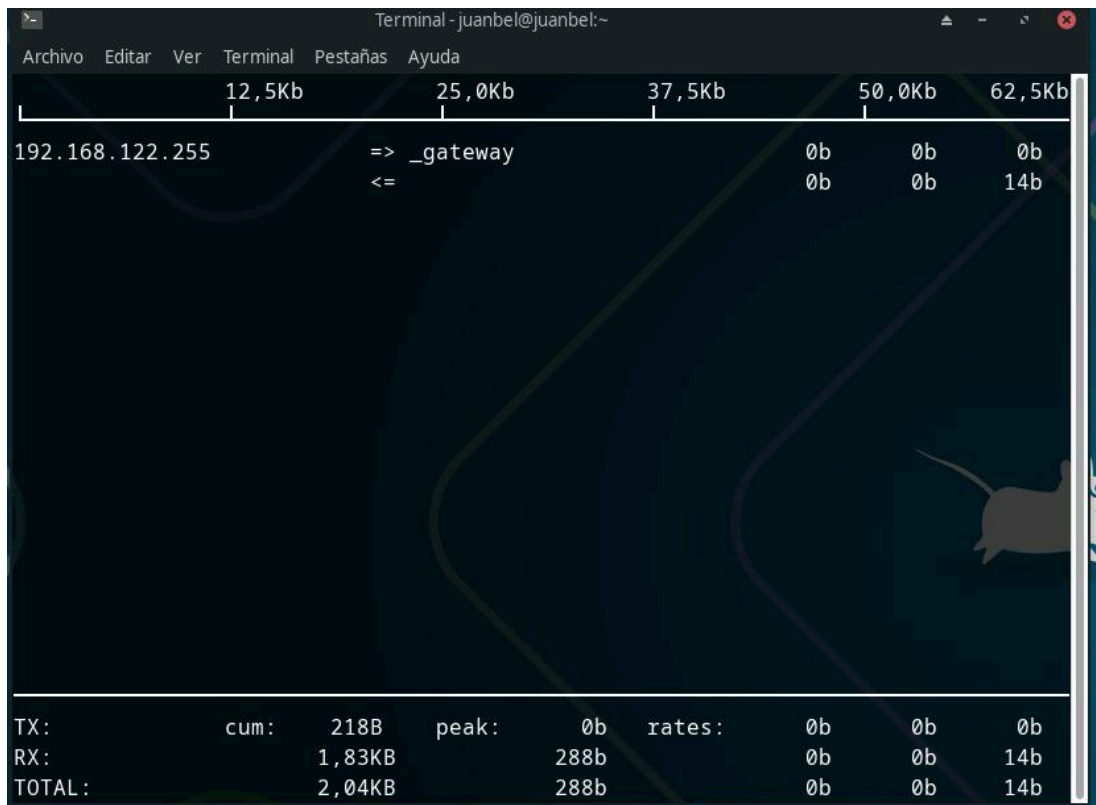


Figura 10: monitorear el tráfico de red en tiempo real.

También se usó **NetHogs** para ver el tráfico de red clasificado por proceso en tiempo real, se puede identificar rápidamente qué aplicaciones usan la red en la máquina virtual o contenedor y detectar si algún proceso genera tráfico sospechoso o inesperado.

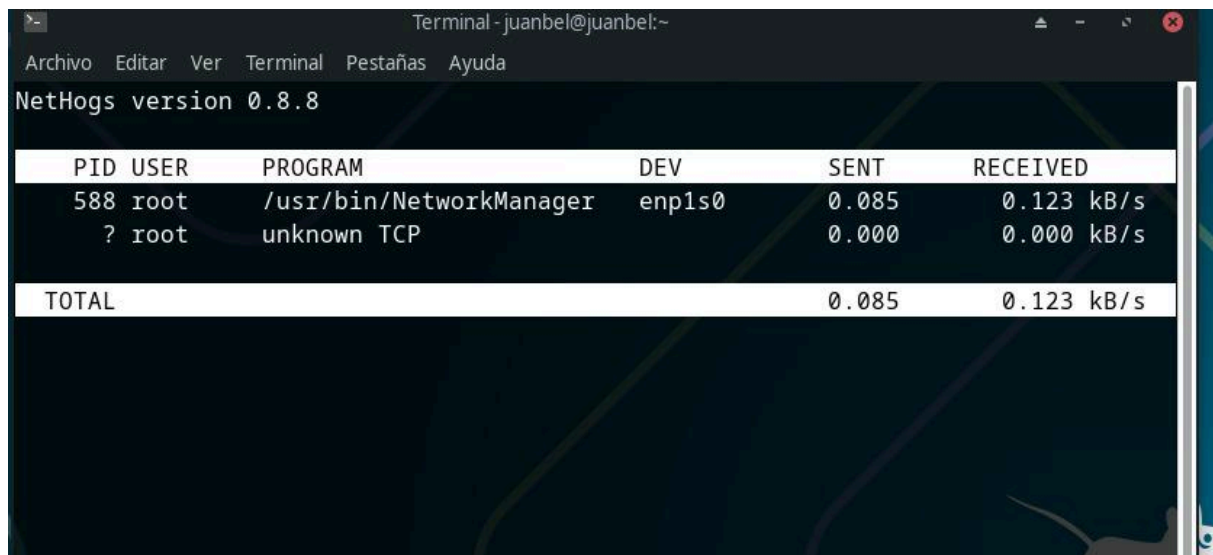


Figura 11: netHog

Continuamos con la figura 12 donde se ejecutó ncd (una herramienta para revisar el uso de espacio en disco en terminal), donde podemos verificar qué carpetas están consumiendo más almacenamiento, y también se facilita encontrar archivos pesados o liberar espacio si es necesario.

```
Terminal - juanbel@juanbel:~
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda
ncdu 2.7 ~ Use the arrow keys to navigate, press ? for help
----- / -----
7.4 GiB [#####] /usr
342.5 MiB [ ] /var
176.6 MiB [ ] /boot
14.1 MiB [ ] /etc
2.2 MiB [ ] /root
2.1 MiB [ ] /home
1.2 MiB [ ] /run
48.0 KiB [ ] /tmp
24.0 KiB [ ] desktopfs-pkgs.txt
8.0 KiB [ ] rootfs-pkgs.txt
4.0 KiB [ ] .manjaro-tools
@ 4.0 KiB [ ] sbin
@ 4.0 KiB [ ] lib64
@ 4.0 KiB [ ] lib
@ 4.0 KiB [ ] bin
0.0 B [ ] /proc
0.0 B [ ] /sys
0.0 B [ ] /dev
0.0 B [ ] /srv
e 0.0 B [ ] /opt
e 0.0 B [ ] /mnt
*Total disk usage: 7.9 GiB Apparent size: 128.0 TiB Items: 525.806
```

Figura 12: Análisis del uso de disco con ncdu

En la figura 13 y 14 se muestran archivos principalmente en el directorio /etc/xdg/, relacionados con configuraciones de escritorio XFCE y otros componentes.

```
[juanbel@juanbel ~]$ sudo rsync -av /etc /tmp/respaldo
[sudo] contraseña para juanbel:
sending incremental file list
created directory /tmp/respaldo
etc/
etc/.pwd.lock
etc/.updated
etc/adjtime
etc/anacrontab
etc/arch-release -> /etc/manjaro-release
etc/bash.bash_logout
etc/bash.bashrc
etc/bindresvport.blacklist
etc/cpufreq-bench.conf
etc/cron.deny
etc/crontab
etc/crypttab
etc/dnsmasq.conf
etc/e2scrub.conf
etc/environment
```

Figura 13: Copia de respaldo


```

Terminal - juanbel@juanbel:~
Archivo Editar Ver Terminal Pestañas Ayuda

etc/xdg/autostart/xiccd.desktop
etc/xdg/menus/
etc/xdg/menus/gnome-applications.menu
etc/xdg/menus/xfce-applications.menu
etc/xdg/menus/xfce-settings-manager.menu
etc/xdg/menus/xfce4-screensavers.menu
etc/xdg/systemd/
etc/xdg/systemd/user -> ../../systemd/user
etc/xdg/tumbler/
etc/xdg/tumbler/tumbler.rc
etc/xdg/xfce4/
etc/xdg/xfce4/Xft.xrdb
etc/xdg/xfce4/helpers.rc
etc/xdg/xfce4/xinitrc
etc/xdg/xfce4/panel/
etc/xdg/xfce4/panel/default.xml
etc/xdg/xfce4/panel/xfce4-clipman-actions.xml
etc/xdg/xfce4/xfconf/
etc/xdg/xfce4/xfconf/xfce-perchannel-xml/
etc/xdg/xfce4/xfconf/xfce-perchannel-xml/xfce4-keyboard-shortcuts.xml
etc/xdg/xfce4/xfconf/xfce-perchannel-xml/xfce4-session.xml
etc/xdg/xfce4/xfconf/xfce-perchannel-xml/xsettings.xml
etc/zsh/
etc/zsh/zprofile

sent 7.484.236 bytes received 27.102 bytes 15.022.676,00 bytes/sec
total size is 7.375.347 speedup is 0,98
[juanbel@juanbel ~]$

```

Figura 14:Copia de seguridad

A continuamos en la figura 15 se muestra un resumen de la herramienta: Glances en Rocky Linux donde se pueden ver sus procesos destacados:

- python3 (Glances y otro proceso Python en ejecución).
- NetworkManager, systemd, sshd, rsyslog, y bash.
- Actividad de usuario juanbel.

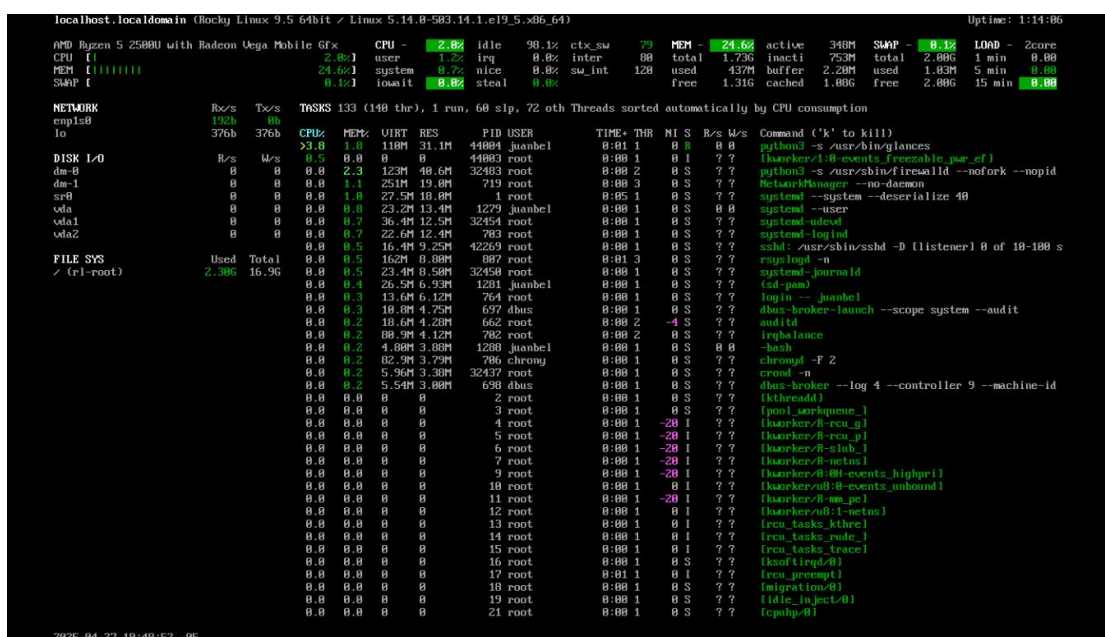


Figura 15:glances con rocky linux

En la figura 16 se ve como el servidor Rocky Linux está en **excelente estado**: bajo consumo, muy pocos procesos activos, ideal para un servidor de pruebas, producción ligera o configuraciones iniciales.

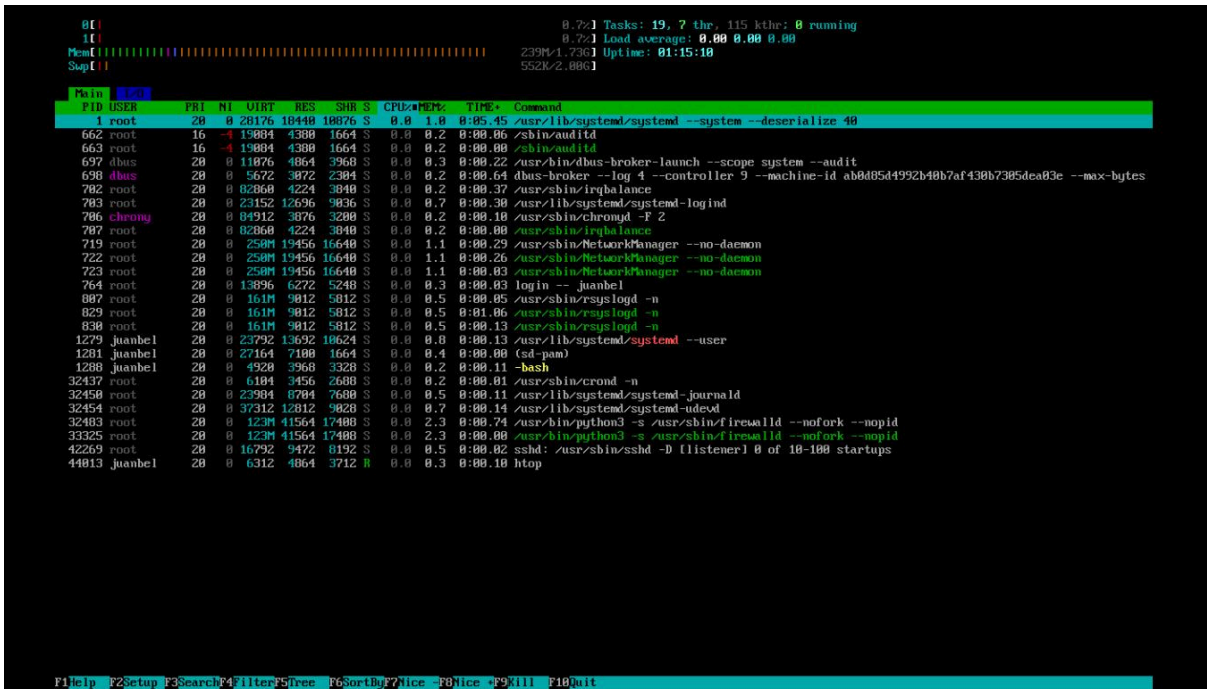


Figura 16:htop procesos

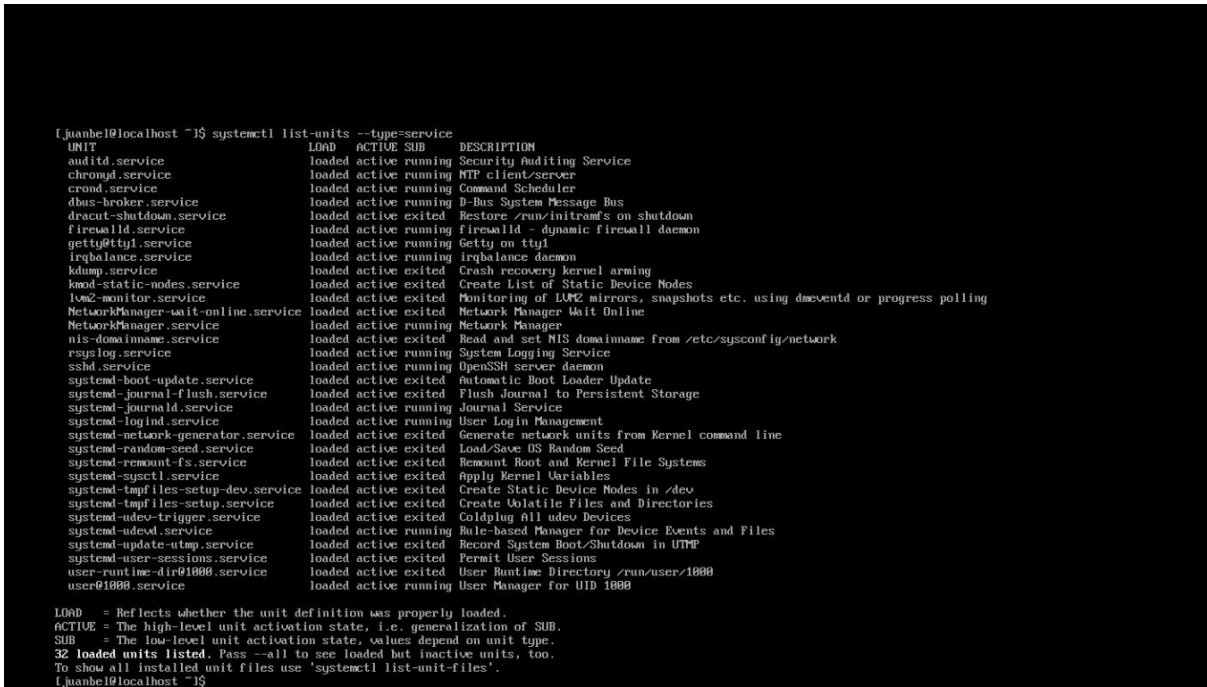


Figura 17: servicios activos

De la misma forma en la figura 18 todo indica que el servicio SSH fue apagado correctamente. No hay fallos graves ni corrupción, donde el sistema (systemd) confirmó que sshd entró en estado 'dead' (muerto / detenido), luego, el objetivo relacionado

sshd-keygen.target también fue detenido, posteriormente, los servicios de generación de llaves (sshd-keygen-ecdsa.service y sshd-keygen-ed25519.service) se saltaron porque no se cumplieron las condiciones de disparo (normal en ambientes donde las llaves ya existen) y todo terminó con "finished successfully", sin errores.

```

    A stop job for unit sshd.service has begun execution.
    The job identifier is 1847.
Apr 27 19:17:47 localhost.localdomain sshd[745]: Received signal 15: terminating.
Apr 27 19:17:47 localhost.localdomain systemd[1]: Stopped OpenSSH server daemon.
    Subject: A stop job for unit sshd.service has finished
    Defined-By: systemd
    Support: https://wiki.rockylinux.org/rocky/support
    The unit sshd.service has successfully entered the 'dead' state.
Apr 27 19:17:47 localhost.localdomain systemd[1]: Stopped OpenSSH server daemon.
    Subject: A stop job for unit sshd.service has finished
    Defined-By: systemd
    Support: https://wiki.rockylinux.org/rocky/support
    A stop job for unit sshd.service has finished.
    The job identifier is 1847 and the job result is done.
Apr 27 19:17:47 localhost.localdomain systemd[1]: Stopped target sshd-keygen.target.
    Subject: A stop job for unit sshd-keygen.target has finished
    Defined-By: systemd
    Support: https://wiki.rockylinux.org/rocky/support
    A stop job for unit sshd-keygen.target has finished.
    The job identifier is 1939 and the job result is done.
Apr 27 19:17:47 localhost.localdomain systemd[1]: Stopping sshd-keygen.target...
    Subject: A stop job for unit sshd-keygen.target has begun execution
    Defined-By: systemd
    Support: https://wiki.rockylinux.org/rocky/support
    A stop job for unit sshd-keygen.target has begun execution.
    The job identifier is 1939.
Apr 27 19:17:47 localhost.localdomain systemd[1]: OpenSSH ecDSA Server Key Generation was skipped because no trigger condition checks were met.
    Subject: A start job for unit sshd-keygen@ecdsa.service has finished successfully
    Defined-By: systemd
    Support: https://wiki.rockylinux.org/rocky/support
    A start job for unit sshd-keygen@ecdsa.service has finished successfully.
    The job identifier is 1935.
Apr 27 19:17:47 localhost.localdomain systemd[1]: OpenSSH ed25519 Server Key Generation was skipped because no trigger condition checks were met.
    Subject: A start job for unit sshd-keygen@ed25519.service has finished successfully
    Defined-By: systemd
    Support: https://wiki.rockylinux.org/rocky/support
    A start job for unit sshd-keygen@ed25519.service has finished successfully.
lines 1491-1539/1723 94%
```

Figura 18:Journalctl (logs del sistema)

Asimismo en la figura 19 se ve el uso de la herramienta iftop, que sirve para monitorear el tráfico de red en tiempo real en Linux, similar a como lo hace top con procesos.

```

      12.5Kb      25.0Kb      37.5Kb      50.0Kb      62.5Kb
Host display:
n - toggle DNS host resolution
s - toggle show source host
d - toggle show destination host
t - cycle line display mode
Port display:
N - toggle service resolution
S - toggle show source port
D - toggle show destination port
p - toggle port display
Sorting:
1/2/3 - sort by 1st/2nd/3rd column
< - sort by source name
> - sort by dest name
o - freeze current order
iftop, version 1.0pre4

General:
P - pause display
h - toggle this help display
b - toggle bar graph display
B - cycle bar graph average
T - toggle cumulative line totals
j/k - scroll display
f - edit filter code
l - set screen filter
L - lin/log scales
t - shell command
q - quit

TX:      cum:  964B  peak:    0b      rates:    0b    0b    0b
RX:      3.83KB      1.53Kb      0b    0b    171b
TOTAL:   4.77KB      1.53Kb      0b    0b    171b
```

Figura 19:IFTOP (ver consumo de red en consola)

Se procede con la figura 20 en la cual se visualiza la cantidad de espacio que ocupa.

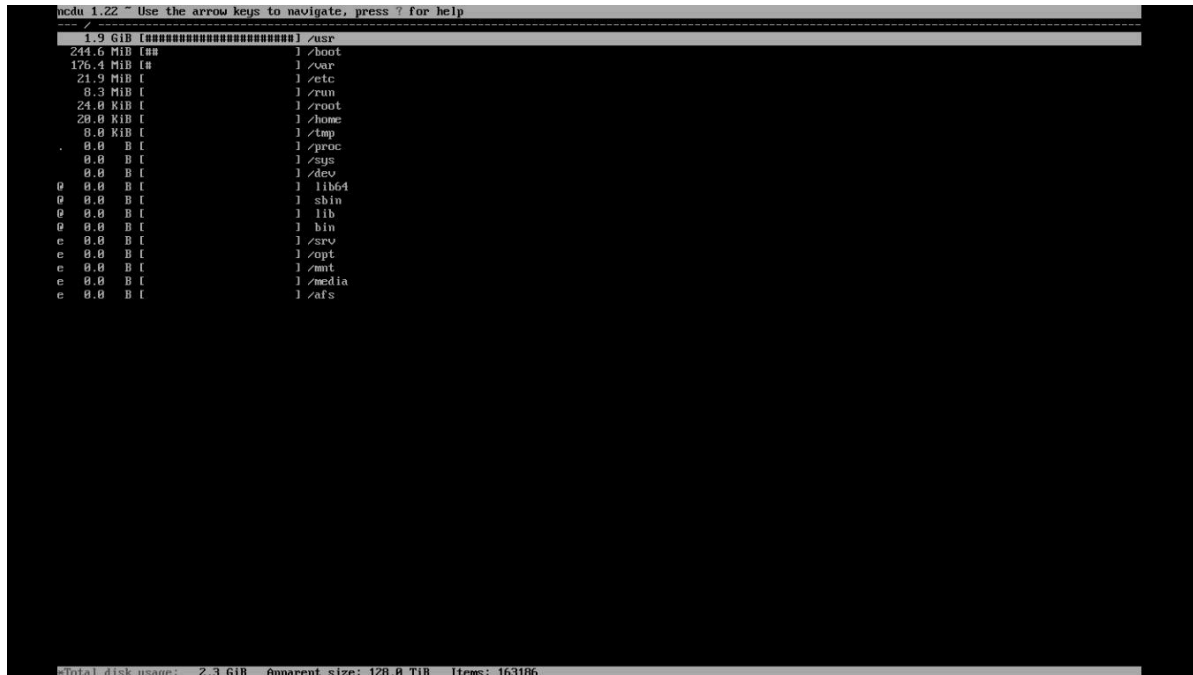


Figura 20:NCDU (ver qué ocupa espacio)

Por otro lado en la figura 21 se muestra como se está usando la herramienta tree, que muestra gráficamente la estructura de directorios desde el punto donde se ejecuta el comando.

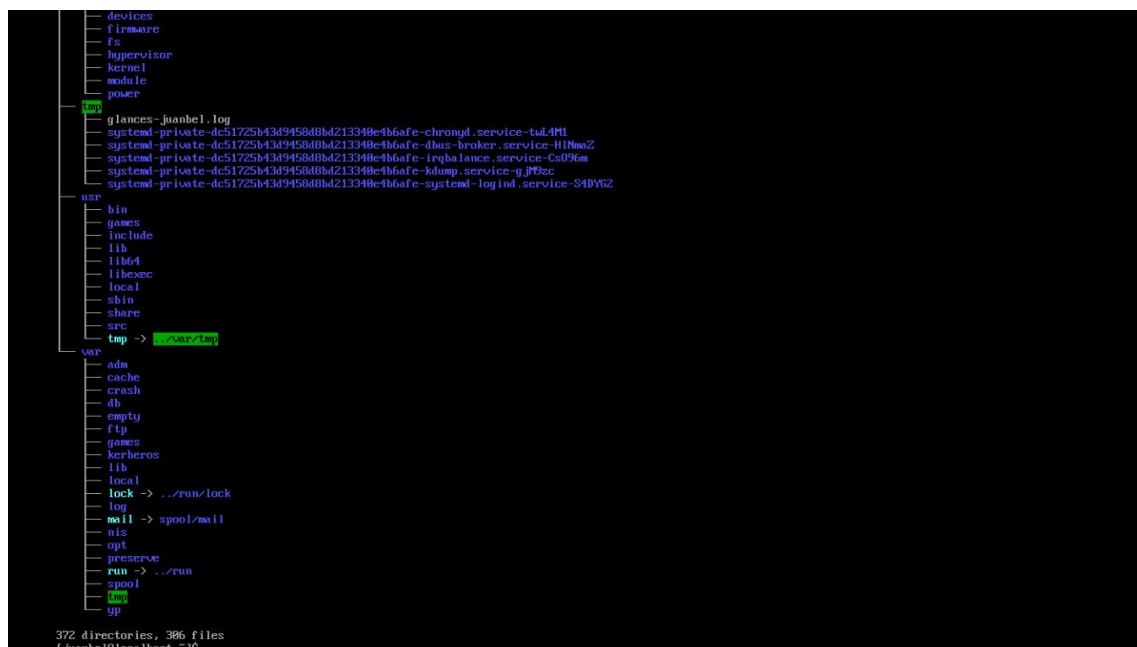


Figura 21: TREE (ver estructura de carpetas)

En la Figura 22 se muestra la salida del comando `lshw`, que da información detallada sobre el

hardware del sistema, donde se ven las Capacidades y extensiones del CPU donde: incluye muchas extensiones avanzadas como:

- SSE/SSE2/SSE4.1/SSE4.2/AVX/AVX2, etc. → útiles para procesamiento intensivo (multimedia, cálculos científicos, etc.).
- Virtualización: svm, v_vmsa, ve_vmlaunch, etc. → el CPU soporta virtualización por hardware (ideal para usar máquinas virtuales).
- Seguridad y eficiencia: soporte para nx, mmxext, tsc, fpu, smep, rdseed, etc.

```
/ # lshw
fe93fc156576
  description: Computer
  width: 64 bits
  capabilities: smp vsyscall32
*-core
  description: Motherboard
  physical id: 0
*-memory
  description: System memory
  physical id: 0
  size: 23GiB
*-cpu
  product: AMD Ryzen 5 2500U with Radeon Vega Mobile Gfx
  vendor: Advanced Micro Devices [AMD]
  physical id: 1
  bus info: cpu@0
  version: 23.17.0
  size: 1312MHz
  capacity: 2GHz
  width: 64 bits
  capabilities: fpu fpu_exception wp vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr
sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp x86-64 constant_tsc rep_good nopl nonstop_tsc cpuid extd_apicid aperfmpe
rf rapl pni pclmulqdq monitor ssse3 fma cx16 sse4_1 sse4_2 movbe popcnt aes xsave avx f16c rdrand lahf_lm cmp_legacy svm extapic
cr8_legacy abm sse4a misalignsse 3dnowprefetch osvw skinit wdt tce topoext perfctr_core perfctr_nb bpext perfctr_llc mwaitx cpb
hw_pstate ssbd ibpb vmcall fsgsbase bmi1 avx2 smep bmi2 rdseed adx smap clflushopt sha_ni xsaveopt xsavec xgetbv1 clzero irper
f xsaveerptr arat npt lbrv svm_lock nrip_save tsc_scale vmcb_clean flushbyasid decodeassists pausefilter pfthreshold avic v_vmsa
ve_vmload vgif overflow_recov succor smca sev sev_es cpufreq
  configuration: microcode=135270411
```

Figura 22: Docker alpine

En conjunto, en la figura 23 se ven diferentes componentes los cuales están bien configurados y compatibles con sistemas Linux. Si estás evaluando compatibilidad o rendimiento de hardware para algún propósito, todo indica que tu equipo está en buen estado técnico.

```
logical name: pcmC1D0p
version: 00
width: 32 bits
clock: 33MHz
capabilities: bus_master cap_list
configuration: driver=snd_hda_intel latency=0
resources: irq:62 memory:e0880000-e0887fff
*-input
  product: HD-Audio Generic Front Headphone
  physical id: 0
  logical name: input18
  logical name: event14
*-pci:4
  description: PCI bridge
  product: Raven/Raven2 Internal PCIe GPP Bridge 0 to Bus B
  vendor: Advanced Micro Devices, Inc. [AMD]
  physical id: 8.2
  bus info: pci@0000:00:08.2
  version: 00
  width: 32 bits
  clock: 33MHz
  capabilities: pci normal_decode bus_master cap_list
  configuration: driver=pcieport
  resources: irq:30 memory:e0400000-e04fffff
*-sata
  description: SATA controller
  product: FCH SATA Controller [AHCI mode]
  vendor: Advanced Micro Devices, Inc. [AMD]
  physical id: 0
```

Figura 23: Inventario hardware:lshw

En la figura 24 se puede ver como es la salida de inxi -F, donde se muestra un resumen y algunas observaciones son: En el Sistema Modelo: Acer Nitro AN515-42, Motherboard: Freed_RRS v: 1.18, BIOS: Insyde v1.18 (fecha: 18/06/2020), Sistema operativo: Alpine Linux 3.21, Kernel: 6.11.0-24-generic, Arquitectura: 64 bits (x86_64), en la batería ID: BAT1, Carga: 100% (55.4 Wh) y Condición: 101.3%, para la CPU tenemos: Modelo: AMD Ryzen 5 2500U con Radeon Vega Mobile Gfx, Núcleos: 4 físicos, Velocidad promedio: 1700 MHz, Caché L2: 2 MiB, y para la Red: Interfaz detectada: eth0, Velocidad: 10 Gbps, con MAC: 82:c7:a6:8c:9b:46

```
capabilities: platform
/ # inxi -F
System:
Host: fe93fc156576 Kernel: 6.11.0-24-generic arch: x86_64 bits: 64
Console: pty pts/1 Distro: Alpine Linux v3.21
Machine:
Type: Laptop System: Acer product: Nitro AN515-42 v: V1.18
serial: NHQ4TAA0018380CAF63400
Mobo: RR model: Freed_RRS v: V1.18 serial: NBQ3R110018385538A3400
BIOS: Insyde v: 1.18 date: 06/18/2020
Battery:
ID-1: BAT1 charge: 55.4 Wh (100.0%) condition: 55.4/54.7 Wh (101.3%)
CPU:
Info: quad core model: AMD Ryzen 5 2500U with Radeon Vega Mobile Gfx
bits: 64 type: MT MCP cache: L2: 2 MiB
Speed (MHz): avg: 1700 min/max: 1600/2000 cores: 1: 2000 2: 1600 3: 1600
4: 1600 5: 2000 6: 1600 7: 1600 8: 1600
Graphics:
Message: Required tool lspci not installed. Check --recommends
Device-1: Chicony HD WebCam driver: uvcvideo type: USB
Display: server: No display server data found. Headless machine?
tty: 80x40
API: N/A Message: No API data available in console. Headless machine?
Audio:
Message: No device data found.
Network:
Message: Required tool lspci not installed. Check --recommends
IF-ID-1: eth0 state: up speed: 10000 Mbps duplex: full
mac: 82:c7:a6:8c:9b:46
```

Figura 24: Resumen del sistema: inxi -F

En la figura 25 nos muestra el disco y sus particiones, también nos muestra los dispositivos loop0 a loop15 son archivos montados como discos virtuales, típicamente usados para:

- Paquetes .apk instalados como imágenes squashfs
- Contenedores tipo snap/flatpak
- Sistemas de archivos temporales

```
inxi: 3.3.35
/ # lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
loop0 7:0 0 4K 1 loop
loop1 7:1 0 73.9M 1 loop
loop2 7:2 0 258M 1 loop
loop3 7:3 0 516M 1 loop
loop4 7:4 0 11.1M 1 loop
loop5 7:5 0 10.8M 1 loop
loop6 7:6 0 44.4M 1 loop
loop7 7:7 0 91.7M 1 loop
loop8 7:8 0 568K 1 loop
loop9 7:9 0 210.4M 1 loop
loop10 7:10 0 63.8M 1 loop
loop11 7:11 0 184.7M 1 loop
loop12 7:12 0 349.7M 1 loop
loop13 7:13 0 44.4M 1 loop
loop14 7:14 0 73.9M 1 loop
loop15 7:15 0 210.2M 1 loop
sda 8:0 0 931.5G 0 disk
├─sda1 8:1 0 100M 0 part
├─sda2 8:2 0 16M 0 part
├─sda3 8:3 0 726.5G 0 part
└─sda4 8:4 0 204.9G 0 part /etc/hosts
                                     /etc/hostname
                                     /etc/resolv.conf
/ #
```

Figura 25: Ver discos: lsblk

A continuación en la figura 26 se muestra un visor de logs en modo texto, logview o una interfaz tipo less/mcview.

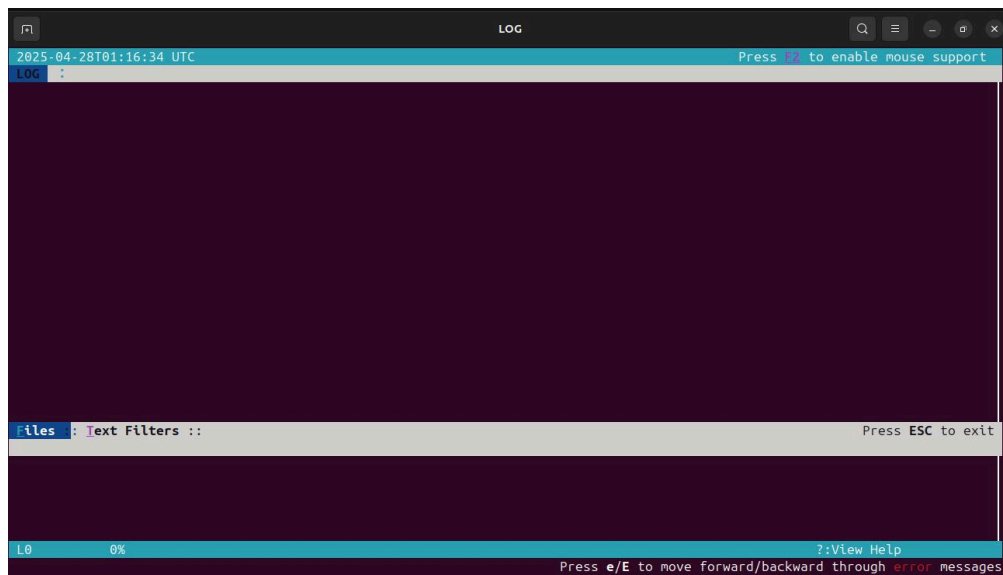


Figura 26: Instalar: `apk add lnav goaccess` Luego: Ver logs: `lnav /var/log/`

Ahora se procede con la figura 27 donde está la creación del contenedor central donde se crea la carpeta `* mkdir contenedor_central`, luego se procede a crear el archivo para el despliegue del contenedor

- `touch central-analytics`
- `nano central-analytics`

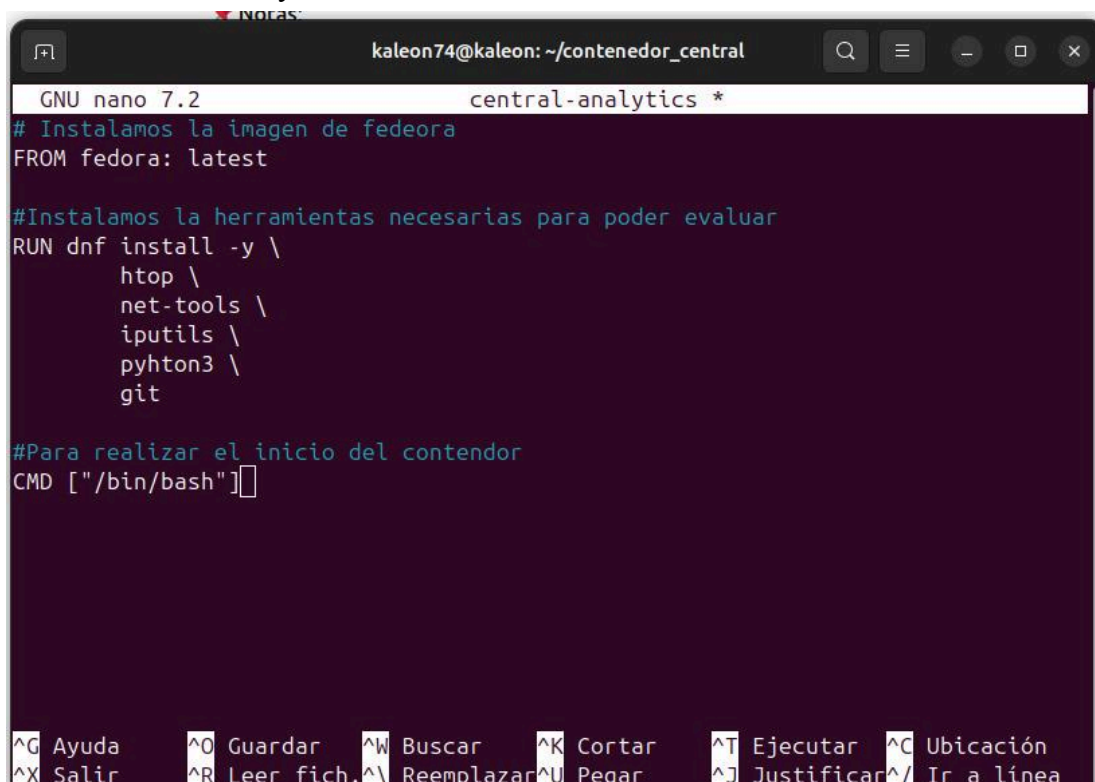
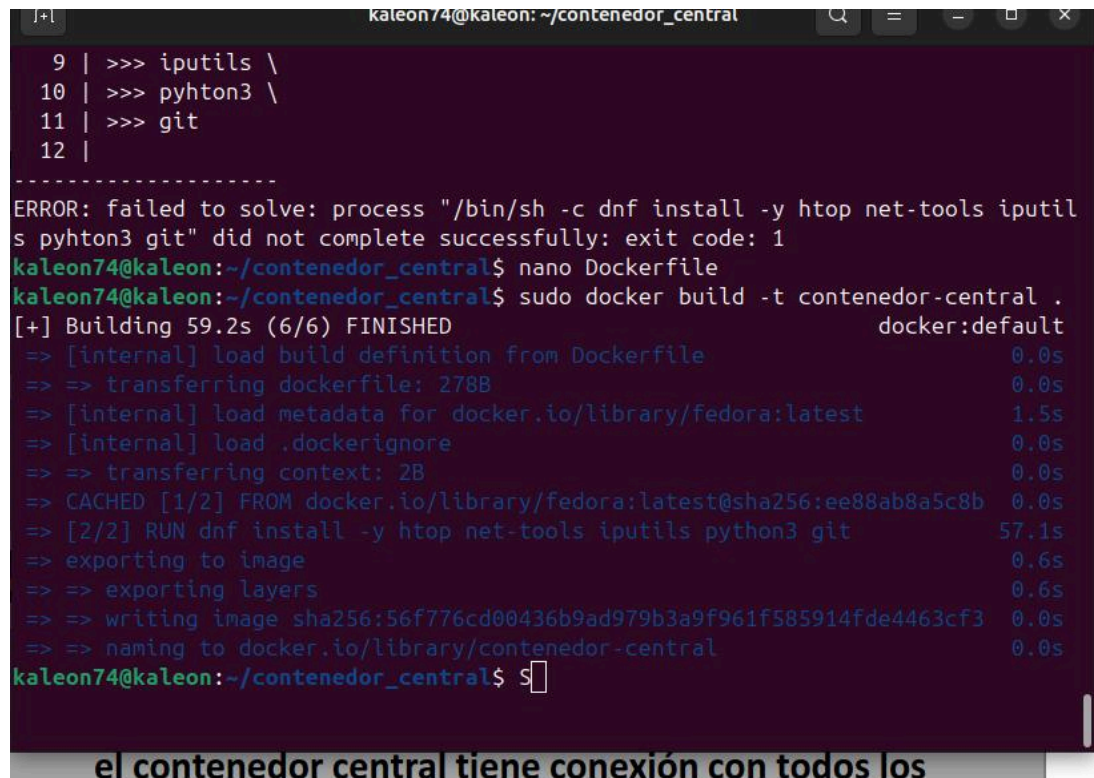


Figura 27: Contenedor central

Como siguiente tenemos la figura 28 donde se ve el despliegue del contenedor central con fedora y los servicios https, net-tools, iputils, python3 y git.



```
[+]
kaleon74@kaleon: ~/contenedor_central

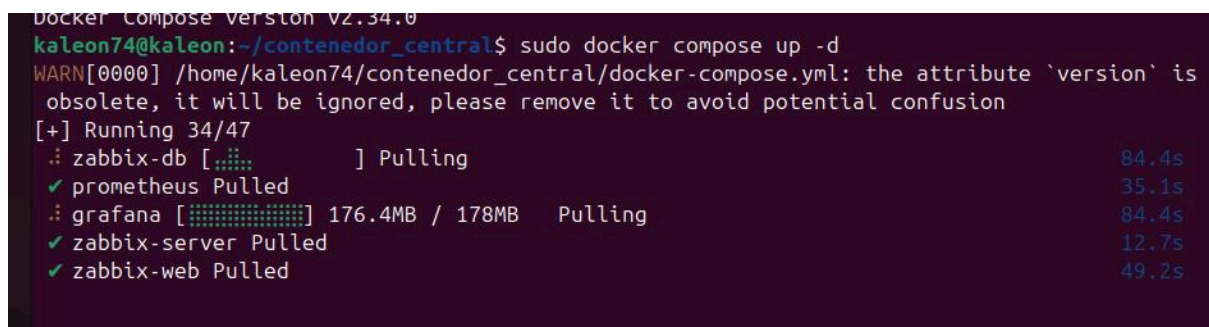
9 | >>> iputils \
10 | >>> pyhton3 \
11 | >>> git
12 |

-----
ERROR: failed to solve: process "/bin/sh -c dnf install -y htop net-tools iputils
python3 git" did not complete successfully: exit code: 1
kaleon74@kaleon:~/contenedor_central$ nano Dockerfile
kaleon74@kaleon:~/contenedor_central$ sudo docker build -t contenedor-central .
[+] Building 59.2s (6/6) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 278B                               0.0s
=> [internal] load metadata for docker.io/library/fedora:latest    1.5s
=> [internal] load .dockerignore                                   0.0s
=> => transferring context: 2B                                     0.0s
=> CACHED [1/2] FROM docker.io/library/fedora:latest@sha256:ee88ab8a5c8b 0.0s
=> [2/2] RUN dnf install -y htop net-tools iputils python3 git     57.1s
=> exporting to image                                              0.6s
=> => exporting layers                                             0.6s
=> => writing image sha256:56f776cd00436b9ad979b3a9f961f585914fde4463cf3 0.0s
=> => naming to docker.io/library/contenedor-central              0.0s
kaleon74@kaleon:~/contenedor_central$ s
```

el contenedor central tiene conexión con todos los

Figura 28:Despliegue del contenedor central.

Continuamos con la figura 29 donde se muestra los despliegue de los servicios grafana zabbix y prometheus



```
Docker Compose version V2.34.0
kaleon74@kaleon:~/contenedor_central$ sudo docker compose up -d
WARN[0000] /home/kaleon74/contenedor_central/docker-compose.yml: the attribute 'version' is
obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 34/47
  :: zabbix-db [.....] Pulling                                84.4s
  ✓ prometheus Pulled                                         35.1s
  :: grafana [.....] 176.4MB / 178MB Pulling                 84.4s
  ✓ zabbix-server Pulled                                       12.7s
  ✓ zabbix-web Pulled                                          49.2s
```

Figura 29: Despliegue de los servicios grafana zabbix y prometheus

Continuamos con la figura 30 y 31, donde se muestra en listado los servicios Arriba del contenedor Integrador Compose Docker y en la figura 32 los listado de los contenedores y servicios desplegados


```
kaleon74@kaleon:~/contenedor_central$ sudo docker compose up -d
WARN[0000] /home/kaleon74/contenedor_central/docker-compose.yml: the attribute `version` is
obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 47/47
  ✓ zabbix-db Pulled 138.7s
  ✓ prometheus Pulled 35.1s
  ✓ grafana Pulled 87.1s
  ✓ zabbix-server Pulled 12.7s
  ✓ zabbix-web Pulled 49.2s

[+] Running 4/5
  ✓ Container contenedor_central-zabbix-db-1 Started 0.7s
  ✓ Container contenedor_central-zabbix-web-1 Started 0.7s
  ⋮ Container contenedor_central-grafana-1 Starting 0.7s
  ✓ Container contenedor_central-prometheus-1 Started 0.7s
```

Figura 30: Servicios en el contenedor de docker

```
[+] Running 4/5
  ✓ Container contenedor_central-zabbix-db-1 Started 0.7s
  ✓ Container contenedor_central-zabbix-web-1 Started 0.7s
  ⋮ Container contenedor_central-grafana-1 Starting 0.7s
  ✓ Container contenedor_central-prometheus-1 Started 0.7s
  ✓ Container contenedor_central-zabbix-server-1 Started 0.7s
Error response from daemon: failed to set up container networking: driver failed programmin
g external connectivity on endpoint contenedor_central-grafana-1 (b8ef0262fe69f9b48f45a063a
ba256085c89d93fae1d75b308adff25c9e93ecb): failed to bind host port for 0.0.0.0:3000:172.18.
0.5:3000/tcp: address already in use
```

Figura 31: Listado de los contenedores y servicios desplegados

```
kaleon74@kaleon:~/contenedor_central$ sudo docker ps
[sudo] contraseña para kaleon74:
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS      PORTS
0d893ddb75bf   prom/prometheus                     "/bin/prometheus --c..." About an hour ago Up About an hour 0.0.0.0:9090->9090/tcp, [::]:9090->9090/tcp
tenedor_central-prometheus-1
db4739328917   zabbix/zabbix-web-nginx-mysql       "docker-entrypoint.sh"   About an hour ago Up About an hour (unhealthy) 0.0.0.0:8080->8080/tcp, [::]:8080->8080/tcp, 8443/tcp
tenedor_central-zabbix-web-1
257ac1741377   mysql:8.0                           "docker-entrypoint.s..." About an hour ago Up About an hour 3306/tcp, 33060/tcp
tenedor_central-zabbix-db-1
kaleon74@kaleon:~/contenedor_central$
```

Figura 32:Listado de los contenedores y servicios desplegados

Además en la figura 33 se muestra el script docker compose donde se realiza esto con el objetivo de levantar un entorno de monitoreo con Prometheus, Grafana y Zabbix, incluyendo la base de datos MySQL para Zabbix.

```
GNU nano 7.2 docker-compose.yml
version: '3'

services:
  prometheus:
    image: prom/prometheus
    ports:
      - "9090:9090"
    volumes:
      - ./prometheus.yml:/etc/prometheus/prometheus.yml
    networks:
      - red-monitoreo

  grafana:
    image: grafana/grafana
    ports:
      - "3000:3000"
    networks:
      - red-monitoreo

  zabbix-server:
    image: zabbix/zabbix-server-mysql
    ports:
      - "10051:10051"
    environment:
      DB_SERVER_HOST: zabbix-db
    networks:
      - red-monitoreo

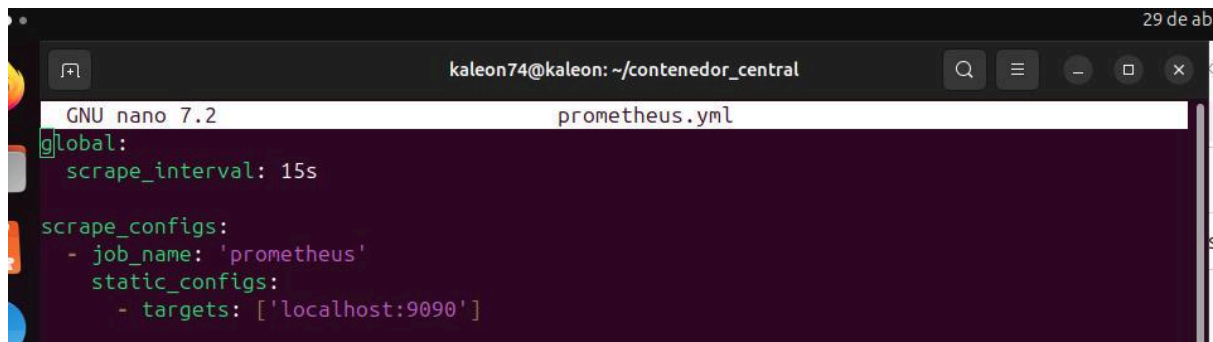
  zabbix-db:
    image: mysql:8.0
    environment:
      MYSQL_ROOT_PASSWORD: zabbix
    networks:
      - red-monitoreo

  zabbix-web:
    image: zabbix/zabbix-web-nginx-mysql
    ports:
      - "8080:8080"
    environment:
      DB_SERVER_HOST: zabbix-db

[ 48 líneas leídas ]
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich. ^_ Reemplazar  ^U Pegar      ^J Justificar ^_ Ir a línea
```

Figura 33: script docker compose

En la figura 34 se muestra el script en el cual se realiza para probar la instalación de Prometheus.



```
GNU nano 7.2 prometheus.yml
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['localhost:9090']
```

Figura 34: Prometheus ejecutable .yml

Procedemos con la figura 35 donde se evidencia el despliegue Dashboard Grafana y Prometheus

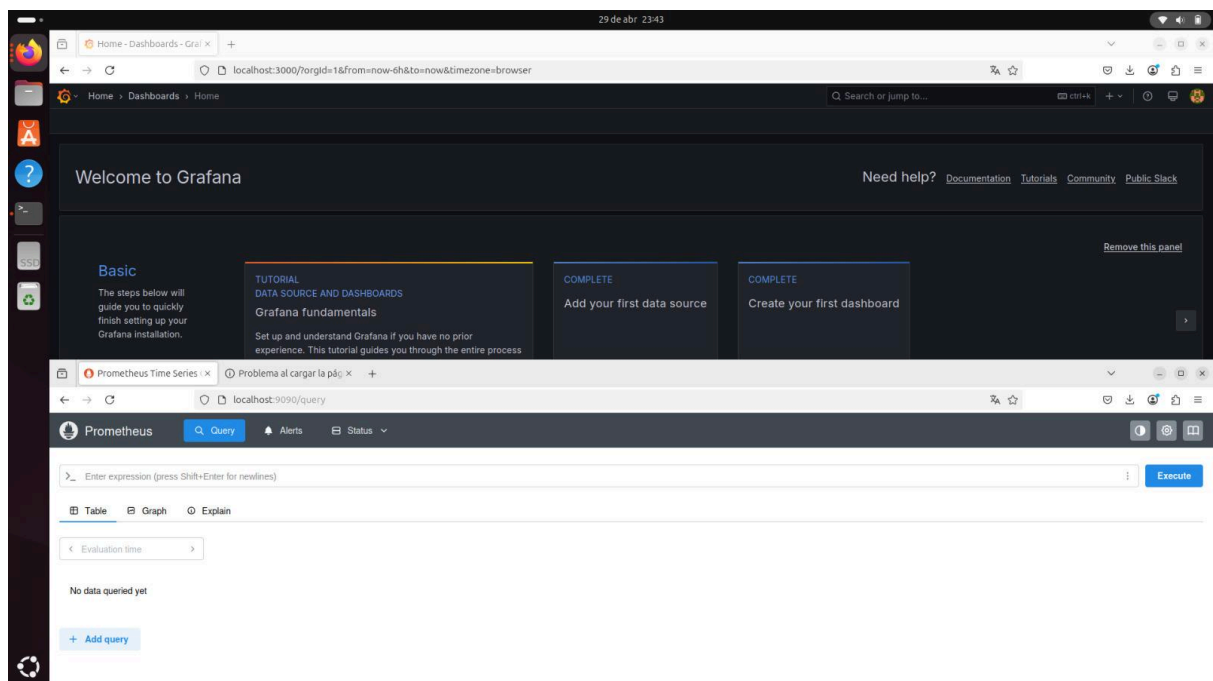


Figura 35: Despliegue Dashboard Grafana y Prometheus

Para finalizar se añade el enlace del repositorio en github para que el docente pueda ver mas a profundidad, https://github.com/kaleon74/Proyecto_Contenedores y por otro lado en tambien la segunda parte está en el repositorio del perfil JuanBeltran54