

S.I.G.P.D.

Adm. de Sistemas Operativos

CEITAROX

Rol	Apellido	Nombre	C.I	Email
Coordinador	Rodríguez	Piero	5.592.003-7	rpierina286@gmail.com
Sub-Coordinador	Gracés	Tatiana	5.713.652-9	tatianatareasutu@gmail.com
Integrante 1	Caballero	Juancruz	5.714.462-1	zuzuonetwothree@gmail.com
Integrante 2	Cardozo	Celeste	5.716.176-6	celestecardozo1803@gmail.com
Integrante 3	Vázquez	Ignacio	5.479.448-7	nachosfera14@gmail.com

Docente: Martínez, Santiago

**Fecha de
culminación
14/07/2025**

PRIMERA ENTREGA



Índice

1. Desarrollo del proyecto
2. Justificación de la elección del sistema operativo
3. Manual de instalación de Fedora
 - 3.1. Instalar el sistema operativo Fedora en PC
 - 3.2. Instalar Fedora Server en VirtualBox
4. Creación del menú de gestor de usuarios y grupos
 - 4.1 Menú principal
 - 4.2 Menú de usuario y grupos
5. Instalación de paquetes
6. Revisión del fireware, rich rules e iptables
7. Configuración del protocolo SSH



1. Desarrollo del proyecto

En esta carpeta se expondrá nuestro proyecto de egreso, con el nombre de CEITAROX presentamos un sistema de seguimiento y gestión para el juego de mesa de tipo "draft" de nombre Draftosaurus de manera online, el cual hemos bautizado como FlowerDraft, donde ganará el jugador con la mejor florería.

En esta carpeta presentaremos el procedimiento de creación de un servidor desde una máquina virtual que permitirá el acceso remoto de los usuarios finales a la base de datos de la aplicación. Se detallará el transcurso de la instalación de su sistema operativo, sus paquetes, la configuración y habilitación del protocolo SSH.



2. Justificación de la elección del sistema operativo

El grupo de proyecto eligió la línea de Linux Fedora, específicamente la versión para servidores número 42 (Fedora Server 42), porque:

- Nuestro profesor de Adm. de Sistemas Operativos nos enseñó cómo utilizar Fedora, por lo tanto estamos más familiarizados con este sistema operativo.
- Nuestro enfoque se centra en la creación, gestión, configuración y comunicación de un servidor remoto, por ende no precisamos de una línea genérica de Fedora.
- Ofrece el uso de diversas tecnologías adaptadas en forma de paquetes, como por Ej. Cockpit, que contribuyen a una gestión y seguridad robusta.
- Pertenece a la familia RedHat, lo que nos permite trabajar con menos complicaciones en nuestras instrucciones con el servidor y los usuarios involucrados.
- Fedora usa un software moderno sin comprometer la estabilidad.
- Fedora tiene una interfaz intuitiva y fácil de usar.
- Tiene buenas herramientas, soporte para muchos lenguajes y contenedores integrados.
- Es un sistema operativo seguro y se actualiza frecuentemente.
- Con el comando DNF install puedes tener lenguajes como Python, Java, etc.



3. Manual de instalación de Fedora

3.1. Instalar el sistema operativo Fedora en PC

Requisitos:

- Una PC o laptop con al menos 2gb de RAM.
- USB de al menos 4GB..
- Acceso a internet (opcional, pero recomendable).

Paso 1: Descargar Fedora

1. Ve al sitio oficial <https://getfedora.org> . Haga click en “download now” bajo la opción “workstation”.

Paso 2: Crear USB booteable.

Usa uno de estos programas:

- Rufus (Windows)
- Balena Etcher (Windows/macOS/Linux)
- Fedora media writer (oficial)

Pasos (con Balena Etcher):

1. Abra Balena Etcher
2. Seleccione la ISO de Fedora que descargaste
3. Seleccione tu USB
4. Realice click en “flash”

Paso 3: Instalar desde el USB

5. Inserte el USB en la computadora donde vas a instalarlo.
6. Reinicie y entre al menú de arranque (presionar las teclas F12, F2, esc, o DEL según tu pc)
7. Seleccione el USB



Paso 4: Cuando aparezca el menú de Fedora, seleccione: “Install to hard drive”

Paso 5: Elegir el idioma e iniciar instalación

8. Seleccione su idioma.
9. Haga click en “continue”

Paso 6: Configurar instalación

Debe revisar:

- *Destino de instalación (disco):* elija el disco donde piensa instalar Fedora.
- *Zona horaria:* seleccione su ubicación en el mapa.

Paso 7: Instalar Fedora

Haga click en “begin installation” y espere unos minutos a que termine.

Paso 8: Reiniciar el dispositivo

Cuando termine la instalación, haga click en “finish installation” y reinicie el equipo. Retira el USB cuando te lo indique.

Paso 9: Configurar usuario

Al iniciar por primera vez cree su usuario y escoja una contraseña.



3.2. Instalar Fedora Server en VMware Workstation Pro

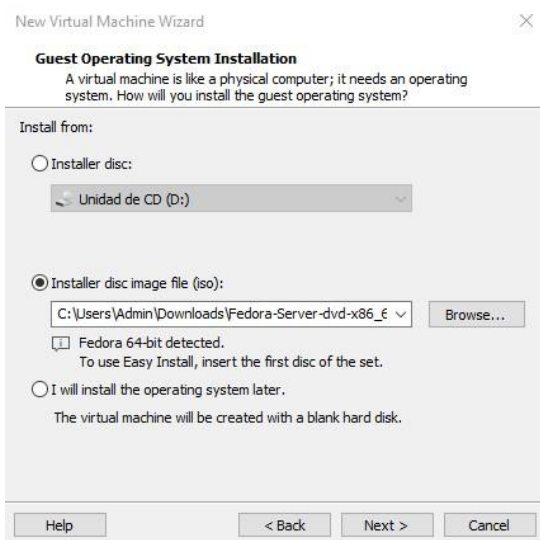
Debido a que algunos integrantes poseen otros sistemas operativos y desean conservarlos en sus dispositivos, hemos hecho este manual alternativo para poder instalar Fedora en la máquina virtual.

Antes de la instalación:

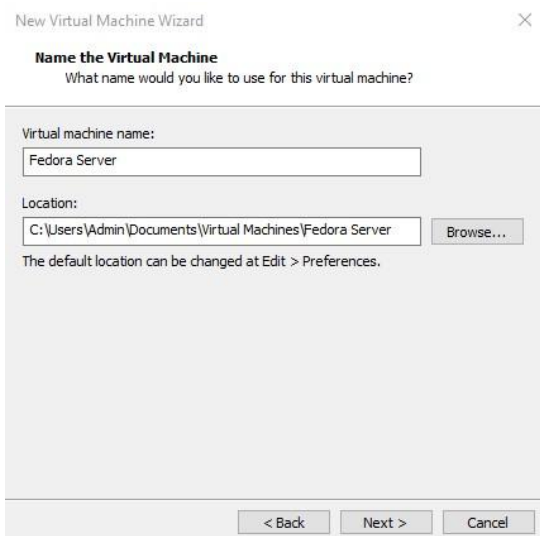
- Descargar el archivo .ISO de Fedora Server 42 desde la página oficial. Instalamos la versión Server y no la versión Workbench ya que no resulta necesario tener la interfaz gráfica, además Fedora Server pesa menos en el disco duro.
- Ya tener instalado VMware Workstation Pro



Luego de haber instalado el VMware y haber descargado el archivo .ISO, presionaremos el botón para crear una nueva máquina virtual, y cambiaremos el modo de creación a “Típico”.



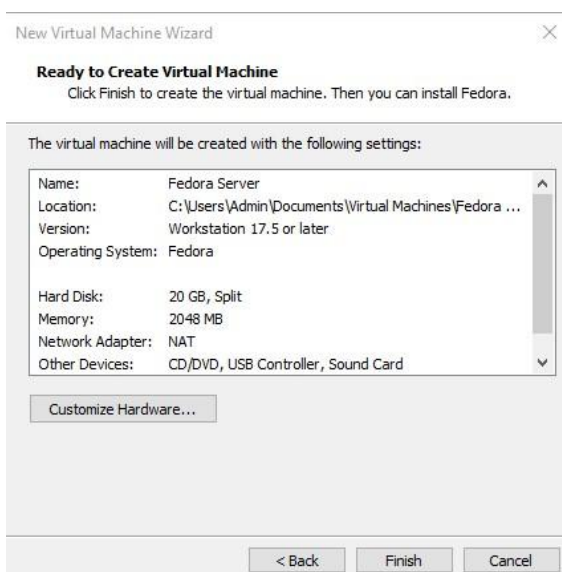
El próximo paso es definir el tipo de disco duro que queremos en la máquina virtual, nosotros utilizaremos una imagen lógica, para ello hay que especificarlo con la ubicación donde se encuentra el .ISO que hemos descargado.



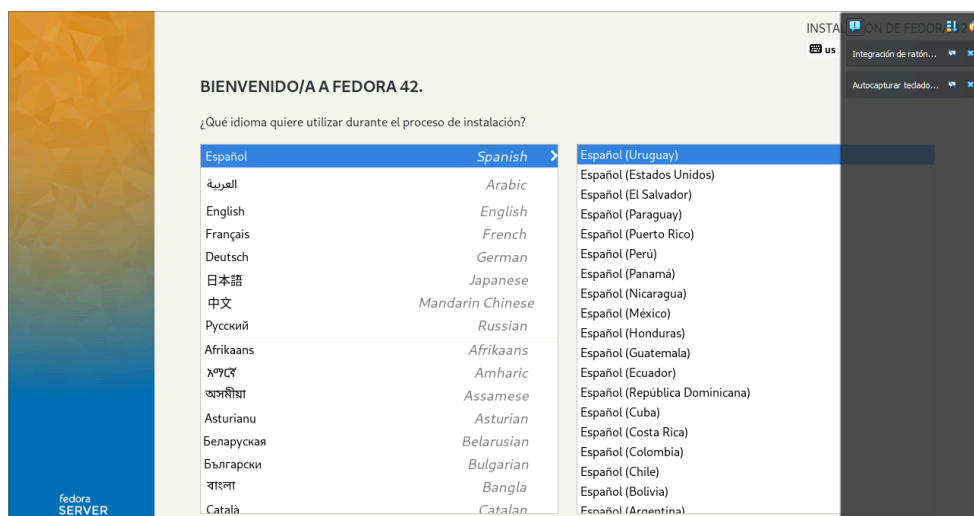
En la siguiente pantalla le indicaremos el nombre de nuestra máquina virtual y la ubicación de dónde se ejecutará.



En la pestaña de disco duro nosotros estableceremos 20GB de almacenamiento..



Por último se muestra en forma resumida las configuraciones de hardware que hemos realizado en los pasos anteriores, si se desea cambiar cierta propiedad debe presionar el botón “Customize Hardware”.



Una vez iniciada, nos recibirá una interfaz gráfica, donde seleccionaremos como idioma el Español, y continuamos a la siguiente pantalla.



Nos enfocaremos en las opciones de ajustes de usuario



La cuenta root se utiliza para administrar el sistema.


El usuario root (también conocido como superusuario) tiene acceso completo a todo el sistema. Por esta razón, es mejor iniciar sesión en este sistema como usuario root solo para realizar el mantenimiento o la administración del sistema.

☐ **Desactivar la cuenta de root**


Desactivar la cuenta de root bloqueará la cuenta y desactivará el acceso remoto con la cuenta de root. Esto evitará el acceso administrativo involuntario al sistema.

☒ **Activar la cuenta de root**

Habilitar la cuenta de root le permitirá establecer una contraseña de root y, opcionalmente, habilitar el acceso remoto a la cuenta de root en este sistema.

Contraseña administrativa: 

Débil


Confirmar: 


☐ Permitir el acceso SSH de root con contraseña

Crearemos una cuenta root para administrar, seleccionaremos la opción de “Activar la cuenta de root”, luego crearemos una contraseña; en la imagen se muestra un ejemplo de contraseña, sin embargo se recomienda crear una contraseña que no sea vista como débil por Fedora.

Una vez terminado presionamos el botón “hecho” para volver a la pantalla anterior y nos dirigiremos a “Creación de usuario”

AJUSTES DE USUARIO

 **Cuenta de root**
Contraseña de root establecida

 **Creación de usuario**
No se creará ningún usuario



Nombre completo: Piero

ID de usuario: piero

☒ Añadir privilegios administrativos a esta cuenta de usuario (membresía al grupo wheel)

☒ Se requiere una contraseña para usar esta cuenta

Contraseña: soypiero

Confirmar contraseña: ••••••••

Avanzado...

Aquí mostramos un ejemplo de cuenta de usuario, donde activaremos la opción para que el usuario esté dentro del grupo wheel, esto lo haremos para evitar administrar y movernos por archivos utilizando el usuario root constantemente. Una vez terminado de crear el usuario con su respectiva contraseña, daremos de nuevo click al botón de “hecho”.

Selección de dispositivo

Seleccione los dispositivos en que le gustaría instalar. Se mantendrán sin tocar hasta que pulse el botón «Comenzar instalación» del menú principal.

Discos estándares locales

20 GiB

ATA VBOX HARDDISK

sda / 20 GiB libre

Discos especializados y de red

Añadir un disco...

Configuración de almacenamiento

☒ Automática ☐ Personalizada ☐ Personalizada avanzada (Blivet-GUI)

☐ Libere espacio eliminando o redimensionando particiones existentes

Cifrado

☐ Cifrar mis datos. Usted fijará una frase de paso después.

Extra: En destino de instalación hemos dejado las opciones por default, por lo que si también te ha salido en rojo la opción, solo deja las opciones predeterminadas.



Luego de configurar instalaremos el sistema operativo. Una vez terminada la instalación iniciarás sesión con el usuario creado anteriormente, y con esto ya estará listo Fedora Server para ser utilizado.

4. Creación de script de suba y baja de usuarios y grupos

Diseñamos un script modulado para gestionar usuarios y grupos.

Su funcionamiento consta de 2 scripts: uno llamado menuPrincipal.sh, donde se corroborará la autenticación del usuario y del acceso a las pantallas de gestor de usuarios y grupos en caso de que el usuario sea aceptado, y otro llamado gestiones.sh, que se encargará de realizar la función de alta y baja de los usuarios y grupos.

4.1 Menú principal

```
#!/bin/bash
if [ $EUID -eq 0 ]; then
    source ./gestiones.sh
    opc=0
    while (( $opc != 3 )); do
        clear
        echo ":::::::::Usuarios y grupos:::::::::"
        echo "*****"
        echo " "
        echo "1. Gestión de usuarios"
        echo "-----"
        echo "2. Gestión de grupos"
        echo "-----"
        echo "3. Salir"
        echo "-----"
        read -p "Eliga una opción:" opc
        case $opc in
            1) menuUsuario;;
            2) menuGrupo;;
            3) echo "Saliendo..."
                read -p "Presione ENTER para salir:"
                echo "Opción no válida"
                read -p "Presione ENTER para continuar:"
            *)
            esac
        done
    else
        echo "(!!) Se requieren privilegios de superusuario (!!)"
    fi
```

Este script es la entrada principal al gestor de usuarios y grupos, su función principal es verificar si el usuario quien está intentando ejecutar el programa forma parte del grupo 'wheel'.

Para lograr el resultado se consideró la variable de entorno 'EUID', que verifica si el usuario activo tiene privilegios de superusuario con valores binarios.



Sabiendo que el valor 0 determina que el usuario en cuestión tiene privilegios del grupo wheel se diseña un if cuya condición sea que la variable 'EUID' tenga asignado el valor 0, ya que cuando un usuario con privilegios ejecuta un comando utilizando 'sudo', su valor cambia por un segundo a 0, el Identificador de usuario de la cuenta root.

Si la condición anterior no se cumple, se imprimirá un mensaje de error: "Se requieren privilegios de de superusuario".

Si la condición anterior se cumple, se extraerá la información del script gestiones.sh mediante el comando 'source', esto con el objetivo de ejecutar las funciones que se encuentran dentro de este script en el script menuPrincipal.sh

Continuando con la condición verdadera, se realizará un menú de opciones, para esto creamos la variable 'opc' para poder realizar la condición de una estructura while para el menú. Este while se crea con la intención de que el menú esté en bucle hasta que la variable 'opc' sea de valor 3 (opción de "Salir" en el menú). Mediante el comando 'read' junto con el delimitador -p para mostrar un mensaje de solicitud al usuario de elegir entre la opción 1 "Gestión de usuarios", la opción 2 "Gestión de grupos" y la opción 3 ya mencionada "Salir", la respuesta dada al comando será almacenada en la variable 'opc'. Con el comando 'case' se utilizará el valor de la variable dado por el comando 'read' para ejecutar las funciones del menú del script gestiones.sh para el caso 1 y 2, el caso 3 hará cerrar el ciclo while y, por ende, el programa. Si se encuentra un valor distinto a 1, 2 o 3 se imprimirá un mensaje default "Opción no válida", y luego de dar ENTER se reiniciará el menú de opciones gracias al 'while'.



4.2 Menú de gestión de usuarios y grupos

```
#!/bin/bash

function menuUsuario(){
    local opc=0
    while (( $opc != 3 )); do

        clear
        echo ":::::Bienvenido al menú de usuarios:::::"
        echo "*****"
        echo "1. Alta usuario"
        echo "-----"
        echo "2. Baja usuario"
        echo "-----"
        echo "3. Volver al menú principal"
        echo "-----"
        read -p "Ingrese una opción:" opc

        case $opc in
            1)altaUsuario;;
            2)bajaUsuario;;
            3)return;;
            *) echo "Opción no válida"
              read -p "Presione ENTER para continuar";;
        esac

    done
}

function altaUsuario(){
    clear
    read -p "Ingrese el nombre del usuario: " usuario
    read -p "Ingrese el directorio de home:" directorio

    if grep -q "^$usuario:" /etc/passwd; then
        echo "El usuario $usuario ya existe"
    else
        useradd -d "/home/$directorio" -m -s /bin/bash "$usuario"
        echo "$usuario:$usuario" | chpasswd
        chage -d 0 "$usuario"
        echo "Se he creado el usuario $usuario en /home/$directorio con contraseña temporal $usuario"
    fi

    read -p "Presione ENTER para continuar"
}

function bajaUsuario(){
```

Función menuUsuario():

Se crea una variable local 'opc' con valor 0, y al igual que el menú de gestión de menuPrincipal.sh, se realiza un 'while' para generar un bucle para este nuevo menú de gestión de usuario.

A comparación del menú principal las opciones pasan a ser: opción 1 "Alta usuario", donde como resultado se ejecutará la función altaUsuario(). Opción 2 "Baja usuario", donde como respuesta se ejecuta la función bajaUsuario(). Por último, opción 3 "Volver al menú principal", donde se ejecuta el comando 'return' para regresar al script anterior. También se aplica una opción default con mensaje de error si los valores no son ni 1, 2 o 3.



Función altaUsuario():

Luego de limpiar la pantalla utilizando 'clear' para más comodidad visual, se le pide al usuario mediante dos 'read' ingrese el nombre de usuario a crear y el nombre del directorio de trabajo home, las respuestas se almacenarán en las variables 'usuario' y 'directorio' respectivamente.

Se verifica con un 'grep' si el usuario ya existe, para esto se revisa su existencia en el directorio '/etc/passwd', donde se guarda la información de los usuarios, incluyendo su nombre. Si la condición es verdadera, se imprime un mensaje indicando que el usuario ya existe. Si la condición es falsa, entonces se ejecutará el comando 'useradd' para crear usuarios, donde le daremos la habilitación del uso del lenguaje Bash y la dirección del directorio home del nuevo usuario utilizando la variable 'directorio' al final de la dirección. Luego de crear la carpeta ('-m') e indicar el shell ('-s'), colocamos la variable 'usuario' para mostrarle al comando el nombre del nuevo usuario.

Creamos una contraseña temporal, para esto utilizamos el 'chpasswd', indicándole que la contraseña del usuario nuevo es su nombre. Con la sintaxis 'chage -d 0' forzamos al usuario a cambiar la contraseña en el momento que inicie sesión. Luego de esto imprimimos un mensaje indicando que se creó el nuevo usuario con respectivo directorio y contraseña temporal exitosamente

.



```
clear
read -p "Ingrese el nombre del usuario a eliminar: " usuario
read -p "¿Desea eliminar también el directorio de home? (y/n): " caso

if [[ $caso == "y" ]]; then
    userdel -r "$usuario"
    echo "Se ha eliminado el usuario $usuario y su directorio home"
elif [[ $caso == "n" ]]; then
    userdel "$usuario"
    echo "Se ha eliminado el usuario $usuario"
else
    echo "Se canceló la baja de usuario por un error en la opción ingresada"
fi

read -p "Presione ENTER para continuar"
}

function menuGrupo() {
    local opc=0
    while (( $opc != 3 )); do
        clear
        echo ":::::Bienvenido al menú de grupos:::::"
        echo "*****"
        echo " "
        echo "1. Alta grupo"
        echo "-----"
        echo "2. Baja grupo"
        echo "-----"
        echo "3. Volver al menú principal"
        echo "-----"
        read -p "Ingrese una opción: " opc

        case $opc in
            1)altaGrupo;;
            2)bajaGrupo;;
            3)return;;
            *) echo "Opción no válida"
              read -p "Presione ENTER para continuar" ;;
        esac
    done
}
```

Función bajaUsuario():

Para realizar la baja de usuario, utilizamos un 'read' para pedirle al usuario el nombre del usuario a eliminar, almacenando el nombre en la variable 'usuario'. También se consulta al usuario si se desea eliminar también el directorio home del usuario a eliminar, con las opciones y(yes) o n(no), la respuesta almacenandose en la variable 'caso'. Se crea una condicional para la variable 'caso', si la opción es 'y' entonces se ejecutará el comando 'userdel' y se indicará el nombre de usuario a eliminar con la variable 'usuario', para que se elimine también su directorio se utiliza '-r'. Si la respuesta es 'n', se ejecuta el mismo

comando sin el '-r'. Si no se ejecuta ninguna de esas dos opciones, se imprime un mensaje de error default.

Función menuGrupo():

Se realiza un menú igual a la función menuUsuario(), donde lo único que cambia son las opciones de alta y baja de usuario, las cuales pasan a ser de alta y baja de grupos. Si se elige la opción 1, se ejecutará la función altaGrupo(), si se elige la opción 2, se ejecuta la función bajaGrupo().

```
2. Baja grupo
-----
3. Volver al menú principal
-----
read -p "Ingrese una opción: " opc

case $opc in
    1)altaGrupo;;
    2)bajaGrupo;;
    3)return;;
    *) "Opción no válida"
      read -p "Presione ENTER para continuar" ;;
esac

done
}

function altaGrupo(){
    read -p "Ingrese el nombre del grupo: " grupo
    if [[ -z "$grupo" ]]; then
    else
        "Nombre de grupo no válido"
    groupadd "$grupo"
        "Se ha creado el grupo $grupo"
    fi
    read -p "Presione ENTER para continuar"
}

function bajaGrupo(){
    read -p "Ingrese el nombre del grupo a eliminar: " grupo
    if [[ -z "$grupo" ]]; then
    else
        "Nombre de grupo no válido"
    sudo groupdel "$grupo"
        "Se eliminó el grupo $grupo"
    fi
    read -p "Presione ENTER para continuar"
}
```

Función altaGrupo():

Se le pide al usuario, con 'read' como las anteriores veces, que ingrese el nombre del grupo nuevo, la información guardándose en la variable 'grupo'. Se realiza una condicional para verificar que la variable 'grupo' no esté vacía, si la variable está vacía se imprime un mensaje de error indicando que el nombre no es 'válido'. Si la variable no está vacía, se llama al comando 'groupadd' y le damos la variable 'grupo', por último se muestra un mensaje indicando que el nuevo grupo fue creado con éxito.



Función bajaGrupo():

Se le pide al usuario, con 'read' como las anteriores veces, que ingrese el nombre del grupo a eliminar, la información guardándose en la variable 'grupo'. Se realiza una condicional para verificar que la variable 'grupo' no esté vacía, si la variable está vacía se imprime un mensaje de error indicando que el nombre no es 'válido'(En un futuro mejoraremos esta condicional para que verifique también si existe un grupo con el mismo nombre que la información dentro de la variable). Si la variable no está vacía, se llama al comando 'groupdel' y le damos la variable 'grupo', por último se muestra un mensaje indicando que el grupo fue eliminado con éxito.



5. Instalación de paquetes

Ya instalado el sistema operativo procedimos a agregar los paquetes Cockpit, OpenSSH-Server y NetworkManager-TUI para gestionar más adelante la configuración de red, el control remoto de parte de los integrantes y nuestra aplicación web al servidor.

```
tatiana@10:~$ dnf search networkmanager-tui
Actualizando y cargando repositorios:
Fedora 42 openh264 (From Cisco) - x86_64          100% | 2.0 KiB/s | 5.8 KiB | 00m03s
Fedora 42 - x86_64                                100% | 1.0 MiB/s | 35.4 MiB | 00m36s
Fedora 42 - x86_64 - Updates                      100% | 2.0 MiB/s | 7.5 MiB | 00m04s
Repositorios cargados.
Matched fields: name
NetworkManager-tui.x86_64: NetworkManager curses-based UI
```

Se inició la búsqueda de los paquetes por medio del comando “dnf” para investigar los repositorios necesarios y concretos para la posterior gestión comenzando por NetworkManager-TUI, el cual escogimos para la configuración IP del servidor ofreciendo una interfaz gráfica que facilita el trabajo.

```
tatiana@10:~$ sudo dnf install NetworkManager-tui
Actualizando y cargando repositorios:
Repositorios cargados.
Package      Arch      Version      Repository      Size
Installing:
NetworkManager-tui      x86_64      1:1.52.0-1.fc42      fedora          711.8 KiB
Installing dependencies:
newt          x86_64      0.52.25-1.fc42      fedora          213.7 KiB
slang         x86_64      2.3.3-7.fc42        fedora           1.4 MiB

Transaction Summary:
Installing:    3 packages

El tamaño total de paquetes entrantes es 775 KiB. Se necesita descargar 775 KiB.
Después de esta operación, 2 MiB extra serán utilizados (instalar 2 MiB, eliminar 0 B).
Is this ok [y/N]:
```

Mediante la modalidad de superusuario se procede a su instalación con el subcomando “install” pidiendo el sistema la autorización del superusuario que indicó el comando antes de su ejecución.



```
El tamaño total de paquetes entrantes es 775 KiB. Se necesita descargar 775 KiB.
Después de esta operación, 2 MiB extra serán utilizados (instalar 2 MiB, eliminar 0 B).
Is this ok [y/N]: y
[1/3] newt-0:0.52.25-1.fc42.x86_64 100% | 96.2 KiB/s | 114.7 KiB | 00m01s
[2/3] NetworkManager-tui-1:1.52.0-1.fc42.x86_64 100% | 155.8 KiB/s | 227.4 KiB | 00m01s
[3/3] slang-0:2.3.3-7.fc42.x86_64 100% | 247.5 KiB/s | 433.3 KiB | 00m02s
-----
[3/3] Total 100% | 120.8 KiB/s | 775.5 KiB | 00m06s
Ejecutando transacción
[1/5] Verificar archivos de paquete 100% | 37.0 B/s | 3.0 B | 00m00s
[2/5] Preparar transacción 100% | 3.0 B/s | 3.0 B | 00m01s
[3/5] Instalando slang-0:2.3.3-7.fc42.x86_64 100% | 4.4 MiB/s | 1.4 MiB | 00m00s
[4/5] Instalando newt-0:0.52.25-1.fc42.x86_64 100% | 189.6 KiB/s | 226.9 KiB | 00m01s
[5/5] Instalando NetworkManager-tui-1:1.52.0-1.fc42.x86_64 100% | 255.3 KiB/s | 713.5 KiB | 00m03s
¡Completado!
```

Una vez autorizado se inicia su instalación.

```
tatiana@10:~$ dnf search cockpit
Actualizando y cargando repositorios:
Repositorios cargados.
Matched fields: name (exact)
cockpit.x86_64: Web Console for Linux servers
Matched fields: name, summary
cockpit-389-ds.noarch: Cockpit UI Plugin for configuring and administering the 389 Directory Server
cockpit-bridge.noarch: Cockpit bridge server-side component
cockpit-composer.noarch: Composer GUI for use with Cockpit
cockpit-doc.noarch: Cockpit deployment and developer guide
cockpit-files.noarch: A filesystem browser for Cockpit
cockpit-ha-cluster.noarch: Cockpit application for managing Pacemaker based clusters
cockpit-image-builder.noarch: Image builder plugin for Cockpit
cockpit-kdump.noarch: Cockpit user interface for kernel crash dumping
cockpit-machines.noarch: Cockpit user interface for virtual machines
cockpit-navigator.noarch: A File System Browser for Cockpit
cockpit-networkmanager.noarch: Cockpit user interface for networking, using NetworkManager
cockpit-ostree.noarch: Cockpit user interface for rpm-ostree
cockpit-packagekit.noarch: Cockpit user interface for packages
cockpit-podman.noarch: Cockpit component for Podman containers
cockpit-selinux.noarch: Cockpit SELinux package
cockpit-session-recording.noarch: Cockpit Session Recording
cockpit-sosreport.noarch: Cockpit user interface for diagnostic reports
cockpit-storaged.noarch: Cockpit user interface for storage, using udisks
cockpit-system.noarch: Cockpit admin interface package for configuring and troubleshooting a system
cockpit-ws.x86_64: Cockpit Web Service
cockpit-ws-selinux.x86_64: SELinux security policy for cockpit-ws
subscription-manager-cockpit.noarch: Subscription Manager Cockpit UI
Matched fields: summary
polarsys-b612-fonts.noarch: Sans-serif fonts designed for reading comfort and safety in aeroplane cockpits
polarsys-b612-mono-fonts.noarch: Monospace fonts designed for reading comfort and safety in aeroplane cockpits
```

En la búsqueda de repositorios para Cockpit se escogió la comentada para consola web ya que los demás repositorios ofrecen funciones no relacionadas a nuestros objetivos planteados para su mantenimiento. A su vez este repositorio ofrece la ventaja de su utilización en formato sitio web para dar soporte al estado y necesidades del servidor.

```
tatiana@10:~$ sudo dnf install cockpit
[sudol contraseña para tatiana:
Actualizando y cargando repositorios:
Repositorios cargados.
Package "cockpit-336.2-1.fc42.x86_64" is already installed.
Nothing to do.
```

En este caso no hubo necesidad de instalarlo debido a que se instaló de forma predeterminada durante la instalación de Fedora.



```
tatiana@10:~$ dnf search openssh
Actualizando y cargando repositorios:
Repositorios cargados.
Matched fields: name (exact)
openssh.x86_64: An open source implementation of SSH protocol version 2
Matched fields: name, summary
lxdm-openssh-askpass.x86_64: Askpass openssh transition dialog for LXQt desktop suite
lxdm-openssh-askpass-l10n.x86_64: Translations for lxdm-openssh-askpass
openssh-askpass.x86_64: A passphrase dialog for OpenSSH and X
openssh-keycat.x86_64: A mls keycat backend for openssh
openssh-ldap-authkeys-selinux.noarch: SELinux module for openssh-ldap-authkeys
openssh-sk-dummy.x86_64: OpenSSH SK driver for test purposes
perl-Net-OpenSSH.noarch: Perl SSH client package implemented on top of OpenSSH
rust-openssh-keys+default-devel.noarch: Read and write OpenSSH public keys
rust-openssh-keys-devel.noarch: Read and write OpenSSH public keys
Matched fields: name
gsi-openssh.x86_64: An implementation of the SSH protocol with GSI authentication
gsi-openssh-clients.x86_64: SSH client applications with GSI authentication
gsi-openssh-keysign.x86_64: A helper program used for host-based authentication
gsi-openssh-server.x86_64: SSH server daemon with GSI authentication
openssh-clients.x86_64: An open source SSH client applications
openssh-keysign.x86_64: A helper program used for host-based authentication
openssh-ldap-authkeys.noarch: Python script to generate SSH authorized_keys files using an LDAP directory
openssh-server.x86_64: An open source SSH server daemon
Matched fields: summary
keychain.noarch: Agent manager for OpenSSH, ssh.com, Sun SSH, and GnuPG
rssh.x86_64: Restricted shell for use with OpenSSH, allowing only scp and/or sftp
x11-ssh-askpass.x86_64: A passphrase dialog for X and not only for OpenSSH

tatiana@10:~$ sudo dnf install openssh
[sudo] contraseña para tatiana:
Actualizando y cargando repositorios:
Repositorios cargados.
Package "openssh-9.9p1-10.fc42.x86_64" is already installed.

Nothing to do.
```

Lo mismo ha trascendido con el montaje de OpenSSH-Server, el cual será utilizado para el control remoto del servidor.

Se piensa próximamente agregar el repositorio Docker, el cual tendrá como misión la conexión de la base de datos con el servidor y almacenar de forma indefinida la información almacenada en esta siendo autorizado su ingreso o eliminación por parte del usuario final.



6. Revisión del firewall, rich tables e iptables

Una vez verificado que el servicio SSH está instalado en el sistema operativo se hacen más procesos de observación al resto del entorno de Fedora. En esta carpeta solo nos centraremos en el firewall, los rich tables y las iptables con el objetivo de determinar si SSH está totalmente habilitado en la máquina virtual para próximamente iniciar las conexiones remotas.

```
tatiana@18:~$ systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/service.d
            └─10-timeout-abort.conf, 50-keep-warm.conf
   Active: active (running) since Sat 2025-07-12 15:46:23 -03; 1h 31min ago
   Invocation: d3f606b1065644579506a691eecc049
   Docs: man:sshd(8)
         man:sshd_config(5)
  Main PID: 833 (sshd)
    Tasks: 1 (limit: 2302)
   Memory: 1.4M (peak: 1.6M)
      CPU: 217ms
   CGroup: /system.slice/sshd.service
           └─833 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

jul 12 15:46:19 localhost.localdomain systemd[1]: Starting sshd.service - OpenSSH server daemon...
jul 12 15:46:23 localhost.localdomain sshd(833): Server listening on 0.0.0.0 port 22.
jul 12 15:46:23 localhost.localdomain sshd(833): Server listening on :: port 22.
jul 12 15:46:23 localhost.localdomain systemd[1]: Started sshd.service - OpenSSH server daemon.
```

Se inició revisando si el sistema lo estaba bloqueando con el uso del comando 'systemctl', el cual se encarga de gestionar los servicios y demonios del mismo sistema operativo, y el subcomando 'status', que se utiliza para listar servicios. Esto al combinarlo con el nombre del servicio SSH ('sshd') dió como resultado la información de su estado actual indicando de que estaba habilitado, para una mejor corroboración se reinició el servicio usando el subcomando 'restart'.

En caso de que no estuviera habilitado se iniciaría el servicio con la sentencia 'systemctl enable sshd'.

```
tatiana@18:~$ sudo firewall-cmd --list-services
[sudo] contraseña para tatiana:
cockpit dhcpv6-client ssh
```

Ya comprobado el sistema se inicia analizando el cortafuegos con privilegios de superusuario con una sentencia parecida pero diferente de la anterior usando el comando 'firewall-cmd', que es lo mismo que systemctl pero destinado para los protocolos de red e IPs, con el delimitador '--list-services', que lista los protocolos habilitados para utilizar en la



máquina, obteniendo la confirmación de que el servicio SSH está también autorizado para su uso en la red.

Aún así para confirmarlo en su totalidad se han revisado las rich tables e iptables, ya que forman parte de la jerarquía del comando 'firewall-cmd' y son las que realmente impiden el correcto funcionamiento de ciertos protocolos de internet, como SSH.

```
tatiana@10:~$ sudo iptables -L | grep ssh  
[sudo] contraseña para tatiana:
```

```
firewall-cmd --list-recognized --arg='--list-rich-rules'  
tatiana@10:~$ sudo firewall-cmd --list-rich-rules  
tatiana@10:~$
```

La primer sentencia busca en su de lista de IPs (delimitador -L) la IP con nombre SSH.

La segunda utilizado nuevamente el comando 'firewall-cmd' busca directamente en las rich rules sis se encuentra un protocolo

En ambas sentencias no se halló ningún protocolo.

7. Configuración del protocolo SSH

Por última para la total definición de este puerto se van a modificar algunas propiedades del mismo, más específicamente la dirección IP de ingreso al servidor (en caso de ser necesario) y el ingreso como root al servidor. Las mencionadas propiedades se encuentran en uno o más archivos '.conf' dentro de este directorio: /etc/ssh. por lo que se tuvieron que analizar los archivos de esa ruta.



Una vez llegada a la ruta se encontraron 2 archivos relacionados a específicamente su configuración siendo sshd_config el principal (el que abarcaba todas la preferencias) y ssh_config.el secundario.

En ssh_config;

```
# Configuration data is parsed as follows:
# 1. command line options
# 2. user-specific file
# 3. system-wide file
# Any configuration value is only changed the first time it is set.
# Thus, host-specific definitions should be at the beginning of the
# configuration file, and defaults at the end.

# Site-wide defaults for some commonly used options. For a comprehensive
# list of available options, their meanings and defaults, please see the
# ssh_config(5) man page.

# Host *
# ForwardAgent no
# ForwardX11 no
# PasswordAuthentication yes
# HostbasedAuthentication no
# GSSAPIAuthentication no
# GSSAPIDelegateCredentials no
# GSSAPIKeyExchange no
# GSSAPITrustDNS no
# BatchMode no
# CheckHostIP no
# AddressFamily any
# ConnectTimeout 0
# StrictHostKeyChecking ask
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# IdentityFile ~/.ssh/id_ecdsa
# IdentityFile ~/.ssh/id_ed25519
# Port 22
# Ciphers aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc
# MACs hmac-md5,hmac-sha1,umac-64@openssh.com
# EscapeChar ~
# Tunnel no
# TunnelDevice any:any
# PermitLocalCommand no
# VisualHostKey no
# ProxyCommand ssh -q -W %h:%p gateway.example.com
# RekeyLimit 1G 1h
# UserKnownHostsFile ~/.ssh/known_hosts.d/%k

# This system is following system-wide crypto policy.
# To modify the crypto properties (Ciphers, MACs, ...), create a *.conf
# file under /etc/ssh/ssh.config.d/ which will be automatically
# included below. For more information, see manual page for
# update-crypto-policies(8) and ssh_config(5).
include /etc/ssh/ssh.config.d/*.conf
"ssh_config" [Sólo lectura] 55L, 1916B
```

55,29

Final

No hubo necesidad de modificación ya que no se encontraron las preferencias mencionadas



En sshd_config:

```
# $OpenBSD: sshd_config,v 1.104 2021/07/02 05:11:21 dtucker Exp $
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.
# This sshd was compiled with PATH=/usr/local/bin:/usr/bin
# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.
# To modify the system-wide sshd configuration, create a *.conf file under
# /etc/ssh/sshd_config.d/ which will be automatically included below
include /etc/ssh/sshd_config.d/*.conf
# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
#
# Ciphers and keying
#RekeyLimit default none
#
# Logging
#SyslogFacility AUTH
#LogLevel INFO
#
# Authentication:
#
#LoginGraceTime 2m
#PermitRootLogin no_
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
#PubkeyAuthentication yes
#
# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile .ssh/authorized_keys
```

Se encontraron las propiedades a buscar ('ListenAddress', y 'PermitRootLogin') y se modificó una de ellas, siendo esta la que corresponde al ingreso como root al servidor. Se seleccionó 'no' debido a que es una cuenta muy poderosa e importante no apta para el uso de un usuario corriente sin conocimientos previos de Linux. Próximamente generaremos un sistema más optimizado de seguridad por ende no se aplica la opción 'prohibit-password' ya que se utiliza con un sistema de seguridad ajeno a la contraseña.