Configuración y personalización de un entorno Linux orientado a pruebas de penetración y seguridad ofensiva

Daniel López Gala

19 de Octubre de 2022

1. Introducción.

Este trabajo consiste en la instalación, configuración y personalización de un sistema operativo ParrotOS en una máquina virtual, siguiendo esta guía. Durante el proceso, se implementaron modificaciones en los diferentes archivos de configuración, así como otros pasos diferentes a los de la referencia, algunos no descritos en este trabajo, con el fin de optimizar y personalizar aún más el sistema.

Además de la implementación de un nuevo gestor de ventanas, entorno de escritorio, emulador de terminal, configuración de la terminal, y capas de personalización como la polybar, se adaptó la configuración a las últimas versiones de Parrot, pues la guía es de 2021. El resultado final puede probarse importando el archivo adjunto de la máquina virtual.

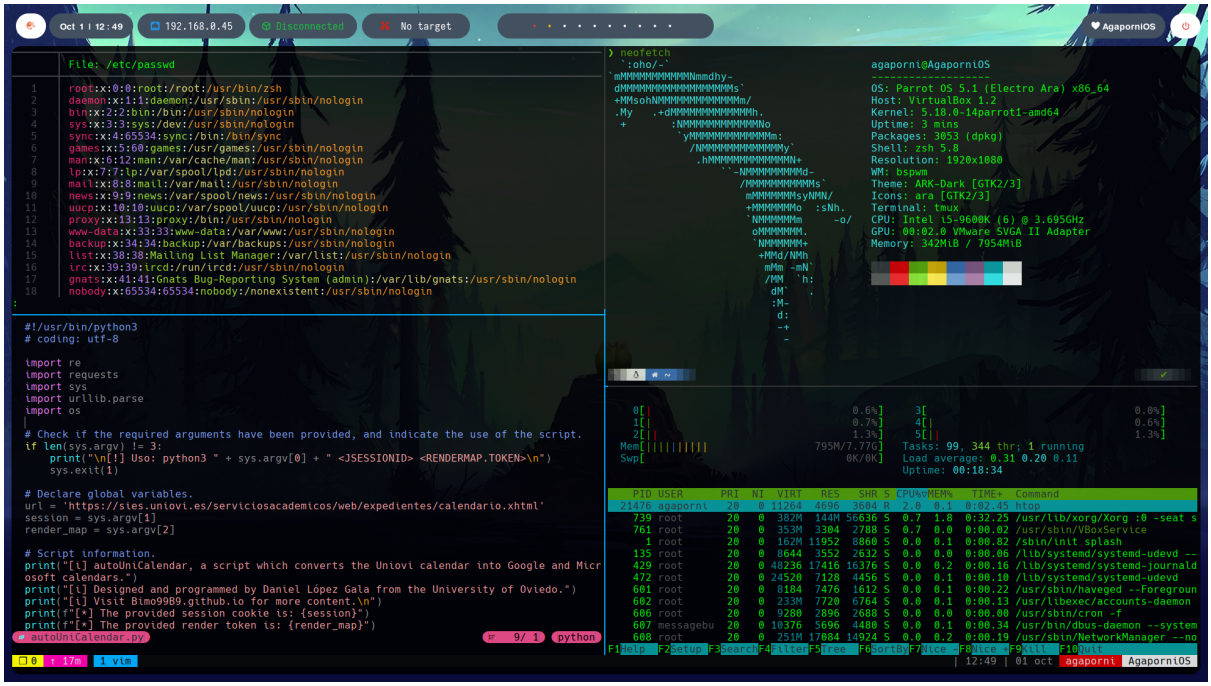


Figura 1: AgaporniOS

2. Instalación en VirtualBox

En primer lugar, se instala la ISO original de Parrot OS (Security Edition) en VirtualBox. La instalación la realicé con 150GB de disco, 8GB de RAM, y 128MB de memoria gráfica.

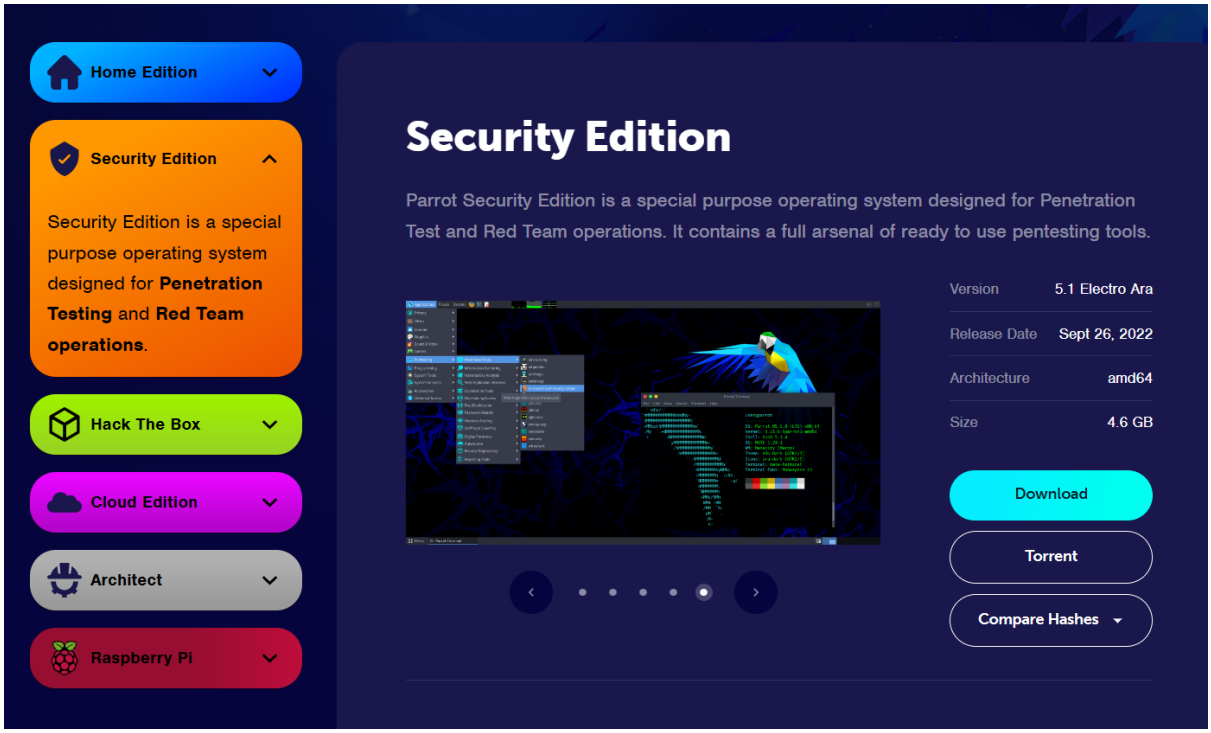


Figura 2: ParrotOS ISO

3. Instalación del Sistema Operativo

Dentro del sistema desde la ISO, se procede a la instalación del sistema. Para ello, los parámetros utilizados fueron *American English* para el idioma, Español para el keyboard, se estableció una contraseña de usuario y de administrador, y **se cifró el disco** con una contraseña de cifrado.
Al reiniciar el sistema, actualizamos el mismo e instalamos los paquetes esenciales de la siguiente manera:

```
sudo apt update
sudo parrot-upgrade
sudo apt install build-essential git vim xcb libxcb-util0-dev libxcb-ewmh-dev
libxcb-randr0-dev libxcb-icccm4-dev libxcb-keysyms1-dev libxcb-xinerama0-dev
libasound2-dev libxcb-xtest0-dev libxcb-shape0-dev
```

También cambié el nombre del host modificándolo en */etc/host* y en */etc/hostname*.

4. Bspwm y sxhkd

Bspwm es un *tiling window manager*, es decir, un gestor de ventanas flotantes, con el que reemplazaremos el MATE Desktop Environment (DE) que viene instalado por defecto. De esta manera, toda la configuración del entorno de trabajo será configurada a mano posteriormente, pues este gestor de ventanas no recibe input del teclado ni del ratón. En bspwm las ventanas se organizan con estructura de árbol binario, y la inserción de nuevas hojas (ventanas), se hace siguiendo diferentes algoritmos de inserción de nodos que se pueden configurar. Por defecto crea una secuencia de Fibonacci en el escritorio activo. En el repositorio de GitHub hay más información disponible.

Por otro lado, Sxhkd se encargará de asignar los *key bingings* correspondientes para crear nodos, escritorios, y poder utilizar Bspwm correctamente.

La instalación se realiza clonando los repositorios correspondientes con *git clone* y ejecutando *make* y *sudo make install* en ambas carpetas de los repositorios.

La configuración inicial consiste en cargar los archivos de configuración por defecto disponibles en el repositorio clonado, en el directorio ∼/**.config** asignando permiso de ejecución a *bspwmrc* con *chmod +x*

Después edité estos archivos con la configuración deseada para mi entorno. Los archivos están disponibles en la máquina virtual, y adjuntos en la entrega. También configuré un script para modificar el tamaño de las ventanas en bswpm.

5. Polybar - Instalación

La Polybar es una barra personalizable al completo en la que añadiremos el menú de encendido, apagado, etc. del sistema, información sobre el mismo, y otros bloques personalizables. Este apartado difiere de lo explicado en la guía pues el repositorio fue actualizado y las dependencias cambiaron, por lo que tuve que hacer varios intentos hasta lograr configurarla adecuadamente.

En el repositorio se indican estas dependencias, y los comandos a utilizar para instalarlas. El proceso para compilar la polybar clonando el repositorio, es el siguiente:

```
cd ~/Downloads
git clone --recursive https://github.com/polybar/polybar
cd polybar/
mkdir build
cd build/
# Faltaba el paquete libuv, instalarlo.
sudo apt install libuv1-dev
cmake ..
make -j$(nproc)
sudo make install
```

6. Picom - Instalación

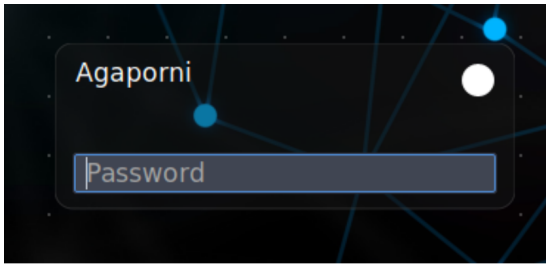
Picom es un *compositor*, es decir, un programa que *compone* las ventanas entre ellas. De esta manera, podemos gestionar la opacidad, animaciones al cerrar o abrir ventanas, cambiar de escritorio, bordes de las ventanas, *screen tearing*... Esto es muy útil debido a que esta instalación se está haciendo con Bspwm, que inicialmente no tiene nada. Pues aunque aún estemos utilizando el entorno que incluye la distribución de Parrot por defecto, al activar Bspwm nos encontraremos con un escritorio en negro sin input de ratón y teclado, y tendremos que transformarlo en un escritorio funcional con todo lo que necesitamos en un sistema operativo. Con Picom configuraremos la parte visual de nuestras ventanas flotantes.

El *fork* utilizado del proyecto de Picom es el siguiente. Su instalación se hace instalando las dependencias y siguiendo los pasos del propio repositorio. En nuestro caso, los pasos serían estos:

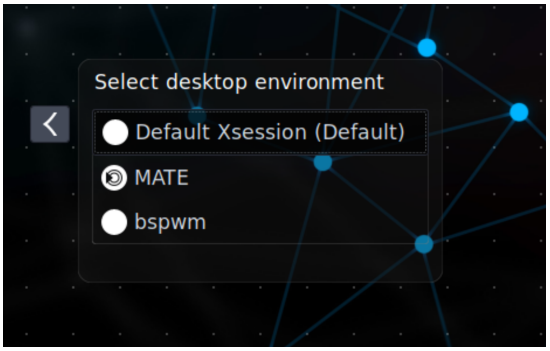
```
cd ~/Downloads
git clone https://github.com/ibhagwan/picom.git
cd picom/
git submodule update --init --recursive
meson --buildtype=release . build
ninja -C build
sudo ninja -C build install
```

7. Activación de Bspwm

Para cambiar al escritorio instalado, debemos cerrar sesión o reiniciar el sistema y escoger "Bspwm"haciendo click en el círculo blanco que aparece al lado del nombre de usuario en la pantalla de bloqueo.



(a) Botón de selección.



(b) Selección de Bspwm

Figura 3: Activación de Bspwm

Una vez en Bswpm, si todo está configurado correctamente, aunque sólo veamos un escritorio en negro, podemos abrir una terminal con *Super + Return* (*Windows + Intro*). A partir de aquí seguiremos configurando el sistema e instalando todo lo que hace falta.

8. Firejail

Firejail restringe el entorno de ejecución de aplicaciones a un entorno limitado para garantizar la seguridad. En este caso, lo configuramos para Firefox. No tiene dependencias, es fácil de configurar, y cualquier aplicación puede ser lanzada con firejail utilizando *firejail (aplicación)*.

Como aún no tenemos Rofi instalado, podemos abrir Firefox ejecutando *sudo firefox* en la terminal. Desde el navegador, descargaremos la última versión de Firefox y la instalaremos de la siguiente manera;

```
sudo su
cd /
chown agaporni:agaporni opt/
cd !$
su agaporni
mv /home/agaporni/Downloads/firefox-* .
tar -xf firefox*
rm firefox-*
cd firefox/
```

Por último, instalamos firejail como root y asignamos un atajo de teclado a *firejail firefox*, para abrir firefox siempre con firejail.

```
sudo su
apt install firejail
su agaporni
nano ~/.config/sxhkd/sxhkdrc

# Agregar en la configuración:
super + shift + f
    firejail /opt/firefox/firefox
```

Después, en Firefox, se configuran los ajustes de privacidad, navegador predeterminado, buscador (Duckduckgo)... Para restringir las carpetas de Firefox: *cd ~/Downloads/ && mkdir Firefox* En ajustes de Firefox - General - Files and Applications, podemos cambiar la carpeta de descargas a esta nueva que hemos creado.

9. HackNerdFonts

Ahora procederemos a instalar un paquete de fuentes que necesitaremos y personalizar la fuente en la terminal. Este paquete de fuentes nos da la posibilidad de utilizar iconos utilizados en todas sus fuentes. Abriendo una terminal con *Windows + Enter*, entramos a *Edit - Profile Preferences*. En este menú de ajustes podemos cambiar la forma del cursor I-Beam, en vez de bloque, y deshabilitar la scrollbar de la terminal. Volviendo al menú General, desactivaremos la *menubar*, la *terminal bell*, y guardamos los ajustes.

Abrimos una ventana de Firefox (*Windows + Shift + F*), y en la web Hack Nerd Fonts accedemos a *Fonts Downloads*, y descargamos la *Hack Nerd Font*, que se guardará en un .zip en *∼/Downloads/Firefox/*.

Para cargar las fuentes:

```
cd /usr/local/share/fonts/
sudo su
mv /home/agaporni/Downloads/Hack.zip .
unzip Hack.zip
rm Hack.zip
```

Para utilizar las fuentes descargadas bastará con abrir una terminal, volver al *menubar* con click derecho, *show menubar*, y deseleccionamos *Use the system fixed width font*. En fuentes, escogí la Regular de las Hack Nerd Fonts con tamaño 12.

10. FoxyProxy

FoxyProxy es un *addon* de Firefox que permite configurar el navegador para realizar conexiones mediante proxy-servers de manera sencilla, creando diferentes perfiles y activando y desactivándolo desde la barra superior cuando sea necesario. Es una herramienta indispensable para trabajar con *Burpsuite* y analizar las solicitudes a un servidor web. Adjunto un ejemplo de uso propio. Lo instalaremos desde la tienda de addons.

En opciones de *FoxyProxy* crearemos un nuevo proxy con los siguientes ajustes.

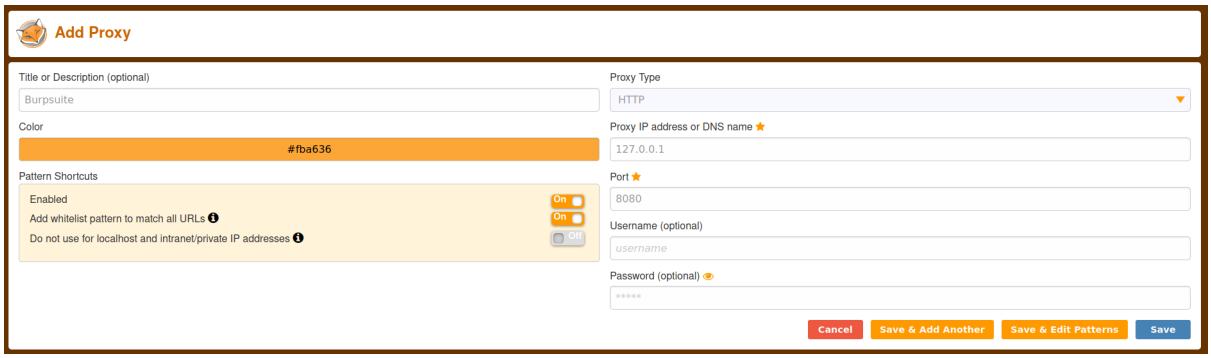


Figura 4: FoxyProxy

11. Feh

Ya que partimos de un sistema prácticamente vacío, no tenemos ni visor de imágenes en este momento. Feh es un programa para visualizar imágenes, pero lo utilizaremos para gestionar el fondo de pantalla. Puede instalarse con *sudo apt install feh*.

Podemos descargar un fondo de pantalla desde internet, aunque yo lo envié desde mi máquina host abriendo un servidor web con *python3 -m http.server 8000* y descargándolo en la máquina con *wget http://(IP):8000/Agapornis.png*. Obviamente esto es posible ya que la red está configurada con adaptador puente en VirtualBox.

Para aplicar el fondo de pantalla cada vez que iniciamos el sistema, debemos añadir la línea que lo establece al final del archivo *bspwmrc*.

```
sudo nano ~/.config/bspwm/bspwmrc
# Añadir la siguiente linea
feh --bg-fill /home/agaporni/Pictures/Agapornis.png
```

Nota importante si se está siguiendo la instalación o si se importa la máquina: *Windows + Alt + R* recargará gráficamente el sistema. Lo usaremos mucho próximamente configurando la Polybar para visualizar los cambios. A veces VirtualBox tiene problemas con la resolución de pantalla. Si no se ve correctamente algo, se debe recargar de esa manera. Con *Windows + Alt + Q* podemos cerrar sesión. Al iniciar de nuevo debería haberse aplicado el fondo de pantalla.

12. Rofi - Instalación

Rofi es un launcher de aplicaciones (y mucho más), que necesitamos para poder lanzar los programas sin necesidad de la terminal.

Su instalación es muy sencilla, ya que puede realizarse con *sudo apt install rofi*. En los archivos de configuración de sxhkd se incluye un atajo de teclado para iniciar rofi con *Windows + D*. Posteriormente configuraremos el tema Nord para cambiar su apariencia a la de la siguiente imagen.

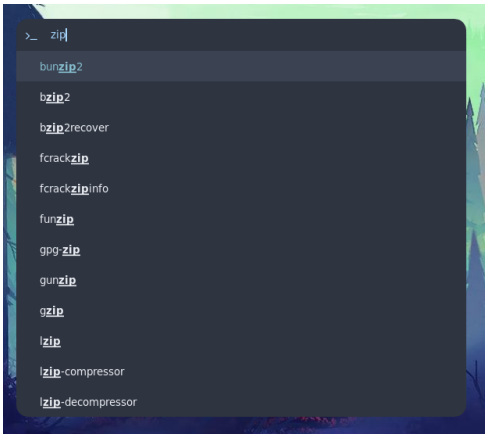


Figura 5: Rofi.

13. Polybar - Configuración

Para establecer un tema base para nuestro sistema, usaremos blue-sky. En el repositorio oficial se incluyen todos los pasos necesarios para instalarlo. Primero activaremos la polybar añadiendo el *launch.sh* al archivo de configuración del gestor de ventanas *bspwmrc*. Posteriormente añadiremos las fuentes.

```
# Clonar el repositorio.
cd ~/Downloads/
git clone https://github.com/VaughnValle/blue-sky.git
# Copiar el tema de la polybar a su carpeta de configuración.
cd ~/Downloads/blue-sky/polybar/
cp * -r ~/.config/polybar
# Activar la polybar.
echo '~/.config/polybar/./launch.sh' >> ~/.config/bspwm/bspwmrc
cd fonts
sudo cp * /usr/share/fonts/truetype/
fc-cache -v
```

Ahora recargamos el escritorio con *Windows + Alt + R*, y veremos aparecer la polybar en la parte superior de la pantalla. Retocar la polybar es un proceso complejo que consiste en abrir el archivo *~/.config/polybar/launch.sh* para identificar la ruta del archivo de configuración del módulo de la polybar a retocar, como se puede ver en la siguiente imagen.

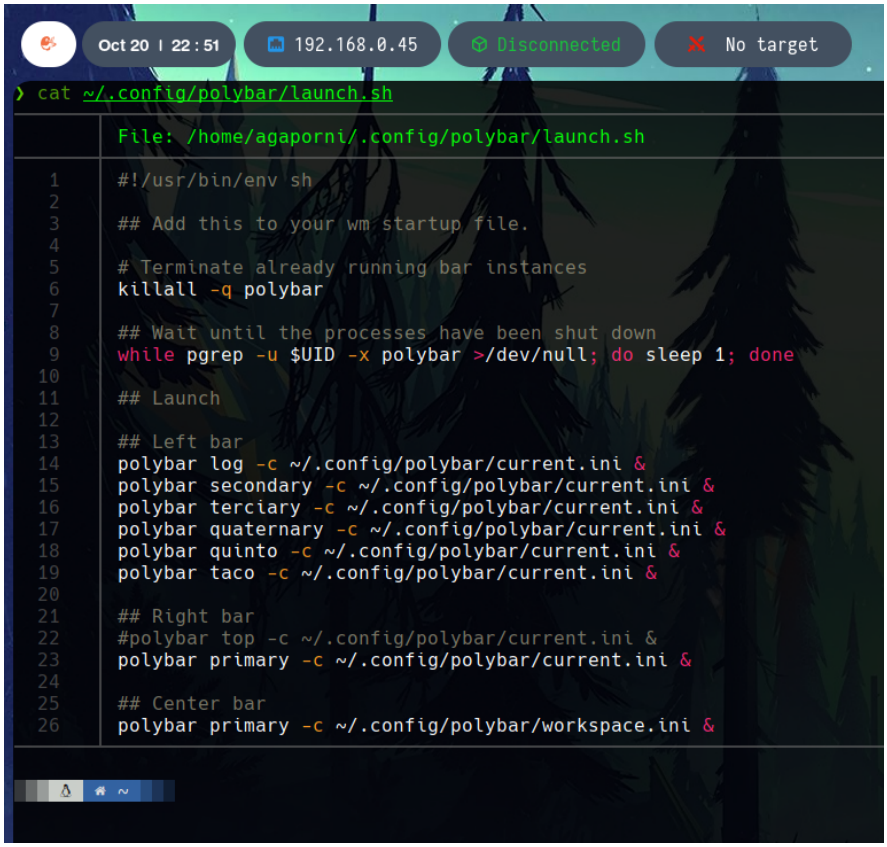
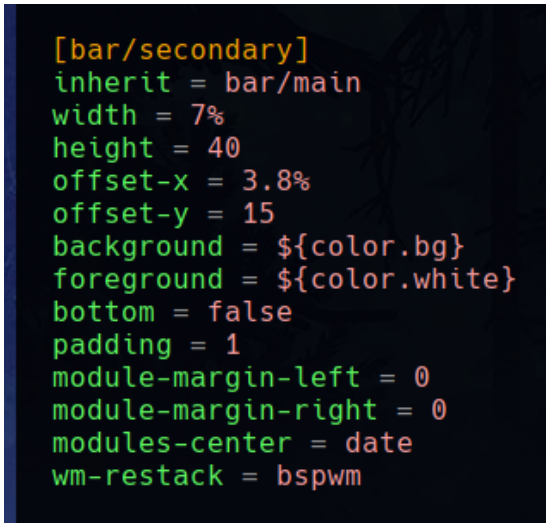
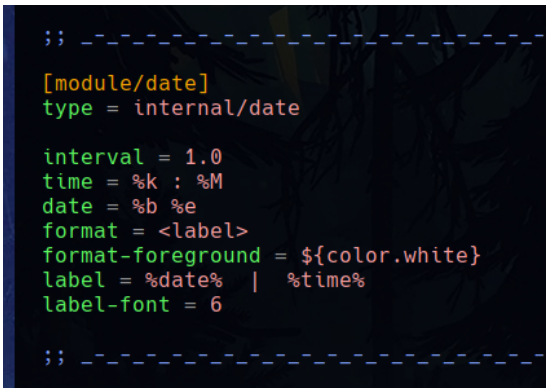


Figura 6: Polybar launch.sh

Una vez localizado el módulo, se accede a su configuración. Por lo general, tienen una configuración visual, y cargan un módulo que puede apuntar a un archivo .sh externo. De esta manera se pueden configurar funcionalidades que veremos más adelante. Cada módulo tiene una configuración general, y una concreta para el "texto"que se muestra. El formato es el siguiente:



(a) Botón de selección.



(b) Selección de Bspwm

Figura 7: Configuración y personalización de cada módulo de la Polybar.

Podemos ver que la *bar* "secondary", llama al módulo "date", y que finalmente la label mostrada tiene el formato *" %date % | %time %"*. De la misma manera, se configuran módulos más complejos que apuntan a un script que permite cambiar el texto mediante comandos personalizados, o el cambio de la interfaz de red. En la .ova adjunta, esto se comprueba con los comandos *vpn*, *settarget (IP)* y *cleartarget*. Estos dos últimos modifican un archivo que lee el script incluido en la configuración de la polybar.

Este es el resultado final tras configurar cada módulo individualmente.

Figura 8: Polybar.

En *./config/bin* se encuentran los scripts para detectar la IP de la interfaz de red y actualizarla en la polybar. **Puede hacer falta modificar este script (*ethernet_ status.sh*) si al importar la máquina la interfaz cambia de nombre.** Simplemente hay que sustituir el nombre de la interfaz en el script.

14. Picom - Configuración

Con Picom ya instalado, queda configurar las transparencias y los bordes de las ventanas en sus archivos de configuración. Creamos un directorio para estos archivos, y tomamos como base el de *blue-sky*.

```
mkdir ~/.config/picom
cd ~/.config/picom
cp ~/Downloads/blue-sky/picom.conf .
```

Debido a que estamos utilizando una máquina virtual, opté por deshabilitar todas las opciones relativas al renderizado gráfico GLX dentro del archivo *picom.conf* para optimizar el rendimiento. Debido a que la carga gráfica es mínima, el procesador debería ser suficiente para gestionar las transparencias y demás.

Para que se cambie automáticamente la selección de la ventana activa a aquella donde tengamos el cursor encima, debemos añadir "*bspc config focus_follows_pointer true*" al final del archivo *~/.config/bspwm/bspwmrc*. Para aplicar los bordeados, hay que añadir también *bspc config border_width 0* y *picom experimental-backends &* en el *bspwmrc*. Ahora sí, recargamos la configuración con *Windows + Alt + R*.

Picom permite definir la opacidad de aplicaciones concretas mediante una *opacity-rule* en el archivo *picom.conf*. Para averiguar el nombre de clase de la aplicación a la que queremos ajustar la transparencia se puede ejecutar *xprop WM_CLASS* y hacer click en la aplicación. Yo configuré burpsuite, ghidra, firefox y rofi para que no tuviesen transparencia.

```
opacity-rule = [
        "100:class_g = 'Rofi'",
        "99:class_g = 'firefox'",
        "100:class_g = 'burp-StartBurp'",
        "100:class_g = 'ghidra'"
];
```

15. Rofi - Personalización

Para aplicar el tema Nord a Rofi, copiaremos el tema desde el directorio de *blue-sky* a un nuevo directorio de temas de rofi en configuración.

```
mkdir -p ~/.config/rofi/themes
cp ~/Downloads/blue-sky/nord.rasi ~/.config/rofi/themes
```

Primero, recargamos con *Windows + Alt + R*. Después, seleccionamos el tema "nord" ejecutando en la consola *rofi-theme-selector*. Los cambios se aplican con *Alt + A*.

16. Powerlevel10k y zsh

Zsh (Z shell) es un intérprete de comandos como bash o sh. Por otro lado, Powerlevel10k es un tema para Zsh. En esta sección configuraremos y personalizaremos la terminal.

En primer lugar, se instala la zsh y se clona el repositorio de powerlevel10k:

```
cd
sudo apt install zsh
git clone --depth=1 https://github.com/romkatv/powerlevel10k.git ~/powerlevel10k
echo 'source ~/powerlevel10k/powerlevel10k.zsh-theme' >> ~/.zshrc
zsh
```

Al ejecutar el último comando, si todo fue correctamente, se inicia un asistente de personalización de la terminal que grabará los ajustes escogidos en un archivo de configuración. Si queremos repetir el proceso de personalización, podemos hacerlo con el comando *p10k-configure*.

Ya que todo esto lo estamos haciendo como el usuario *agaporni*, hay que repetirlo para *root* y después crear un enlace simbólico entre el archivo de configuración *.zshrc* de *agaporni* y de *root*.

```
sudo su
cd
git clone --depth=1 https://github.com/romkatv/powerlevel10k.git ~/powerlevel10k
echo 'source ~/powerlevel10k/powerlevel10k.zsh-theme' >> ~/.zshrc
ln -s -f /home/agaporni/.zshrc .zshrc
```

Ahora definiremos zsh como la shell predeterminada del sistema para los dos usuarios.

```
usermod --shell /usr/bin/zsh agaporni
usermod --shell /usr/bin/zsh root
```

Para resolver unos errores de permisos al cambiar de usuario, se pueden cambiar ciertos permisos de la siguiente manera.

```
# resolver errores de permisos
chown agaporni:agaporni /root
chown agaporni:agaporni /root/.cache -R
chown agaporni:agaporni /root/.local -R
```

Por último, instalé plugins de autocompletado, sugerencias de comandos, y reconocimiento de sintaxis en el directorio */usr/share/*.

17. Bat, lsd, Ranger y fzf

Estos cuatro paquetes permiten personalizar la experiencia en el sistema. Bat es un *cat* que añade reconocimiento de sintaxis, integración con git, paginación automática, y mucho más. Para usarlo por defecto se añade un *alias* en *.zshrc* (*alias cat='bat'*), para que al ejecutar el comando *cat*, se use *bat*.

lsd hace lo mismo con el comando ls. fzf permite realizar una **búsqueda** en el directorio, a modo de explorador de archivos por línea de comandos. Estos tres paquetes se instalan descargando la versión amd64 de los repositorios y extráyendolos con el comando *dpkg -i (paquete)_ amd64.deb*.

Ranger sirve para explorar el directorio de forma visual en la consola. En el repositorio hay más información sobre la herramienta. Se instala con *sudo apt install ranger*.

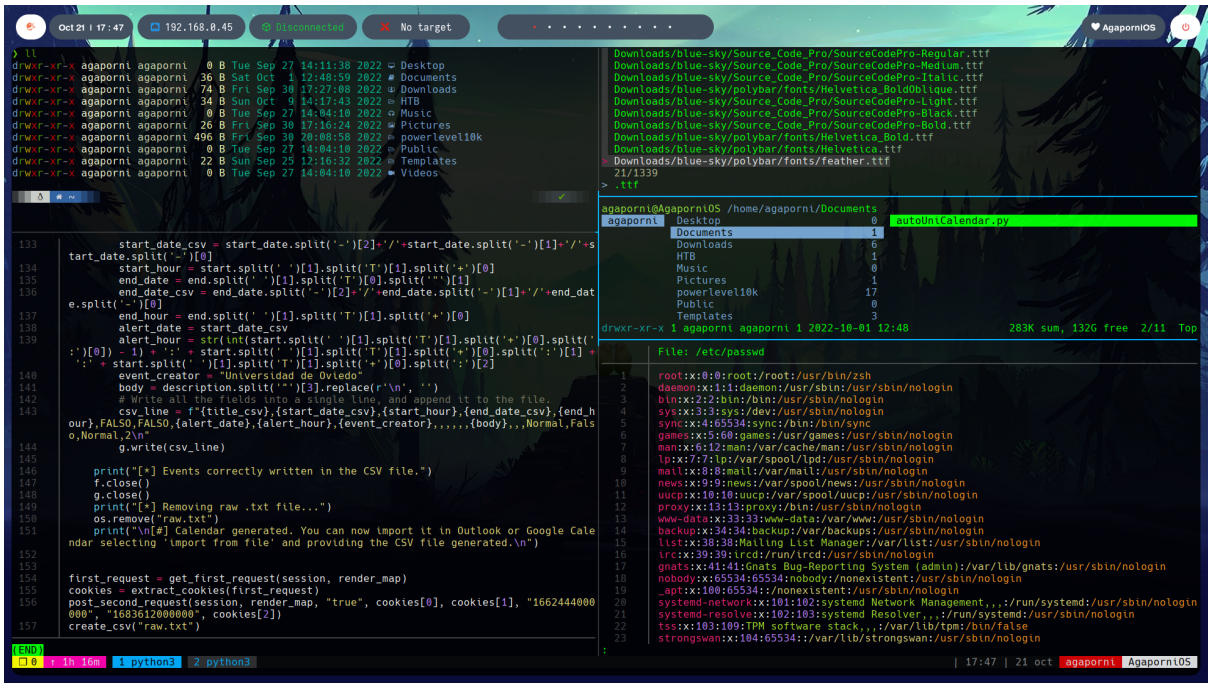


Figura 9: En orden, lsd, fzf, ranger y bat.

18. Vim

Para personalizar vim, instalaremos el tema Nord Vim. En lugar de seguir la instalación completa del repositorio, simplemente copiaremos los archivos de configuración de los colores al directorio *~/.config/nvim/* con *mv nord-vim-main/colors/ ~/.config/nvim/*. Después eliminé el archivo *init.vim* y lo sustituí por el del tema lotus de la siguiente manera. Al final del nuevo *init.vim*, añadí la línea *syntax on* para reconocimiento de sintaxis en vim.

```
rm init.vim
wget https://raw.githubusercontent.com/Necros1s/lotus/master/lotus.vim
wget https://raw.githubusercontent.com/Necros1s/lotus/master/lotusbar.vim
wget https://raw.githubusercontent.com/Necros1s/lotus/master/init.vim
```

Puede ocurrir que tras modificar vim, al editar un archivo con este editor y cerrarlo, al volver a la consola el cursor vuelva a modo bloque en vez de cursor. Esto puede solucionarse editando el archivo de configuración de zsh *.zshrc* y añadiendo las siguientes líneas:

```
# Change cursor shape for different vi modes.
function zle-keymap-select {
  if [[ $KEYMAP == vicmd ]] || [[ $1 = 'block' ]]; then
    echo -ne '\e[1 q'
  elif [[ $KEYMAP == main ]] || [[ $KEYMAP == viins ]] || [[ $KEYMAP = ''
  ↪  ]] || [[ $1 = 'beam' ]]; then
    echo -ne '\e[5 q'
  fi
}
zle -N zle-keymap-select

# Start with beam shape cursor on zsh startup and after every command.
zle-line-init() { zle-keymap-select 'beam'}
```

Oh My Tmux

Para personalizar Tmux (el multiplexador de terminal), usamos los ajustes de este repositorio. Siguiendo la instalación ahí indicada, se haría de la siguiente manera.

```
cd
git clone https://github.com/gpakosz/.tmux
ln -s -f .tmux/.tmux.conf
cp .tmux/.tmux.conf.local .
```

Por último, se aplica la misma configuración para root

```
sudo su
cd
git clone https://github.com/gpakosz/.tmux
ln -s -f .tmux/.tmux.conf
cp .tmux/.tmux.conf.local .
```

19. Spicetify

Con esto, el sistema estaría listo. Ya que se busca que la instalación sea ligera, no se instalan más cosas, pero quise añadir Spicetify, un cliente personalizado para Spotify.

19.1. Instalación de Spotify

Lo primero que tenemos que hacer es instalar Spotify. Como nuestro sistema es Parrot OS, está basado en Debian, por lo que según la Spotify, se instala añadiendo el repositorio y posteriormente descargando el cliente mediante el gestor de paquetes apt.

```
curl -sS https://download.spotify.com/debian/pubkey_5E3C45D7B312C643.gpg
↪  | sudo apt-key add -

echo "deb http://repository.spotify.com stable non-free" | sudo tee
↪  /etc/apt/sources.list.d/spotify.list

sudo apt-get update && sudo apt-get install spotify-client
```

19.2. Instalación de Spicetify

Según la documentación del proyecto, Spicetify se instala de la siguiente manera.

```
curl -fsSL
↪  https://raw.githubusercontent.com/spicetify/spicetify-cli/master/install.sh
↪  | sh
```

Si da error, es porque no hemos iniciado sesión previamente en el Spotify original. Bastará con abrirlo e iniciar sesión para que quedé guardado el perfil correctamente.

Por último, asignaremos los permisos correspondientes

```
sudo chmod a+wr /usr/share/spotify
sudo chmod a+wr /usr/share/spotify/Apps -R
```

19.3. Personalización de Spotify

Ahora seguiremos los pasos de blue-sky para la personalización, con algún cambio porque algunos daban errores. Lo primero es lanzar Spotify usando spicetify.

```
spicetify
spicetify backup apply enable-devtools
spicetify update
```

Por último, descargaremos e importaremos el tema.

```
cd ~/Downloads
git clone https://github.com/morpheusthewhite/spicetify-themes.git
cd spicetify-themes
cp -r * ~/.config/spicetify/Themes/
cd ~/.config/spicetify/Themes/Dribbblish/
cp dribbblish.js ../../Extensions
spicetify config extensions dribbblish.js
spicetify config current_theme Dribbblish color_scheme nord-dark
spicetify config inject_css 1 replace_colors 1 overwrite_assets 1
spicetify apply
```

Tras estos ajustes, ya hemos modificado nuestro Spotify. Podemos lanzarlo usando Rofi con *Windows* + *D*, y buscando Spotify. El resultado es el siguiente.

Figura 10: Spicetify.