How-to Personalizar el entorno de trabajo

#entorno #personalización #kali

Instalando bspwm y sxhkd

nos ponemos como **root :
`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`

(es posible q "xcb" nos de problemas, lo quitamos y listo.)

Hacemos apt update && apt upgrade.

Pasamos al usuario **no-root**, vamos al directorio Downloads `cd Downloads` y ejecutamos:
`git clone https://github.com/baskerville/bspwm.git` y:
`git clone https://github.com/baskerville/sxhkd.git` .
Entramos al directorio `bspwm` y hacemos `make` y luego un `sudo make install` .
Una vez hecho esto haremos lo mismo en el directorio `sxhkd` : `make` y `sudo make install` .

Ahora ya instalamos **bspwm** con `sudo apt install bspwm` .
Creamos dos directorios (bspwm y sxhkd):
`mkdir ~/.config{bspwm,sxhkd}`
Nos movemos a `~/Downloads/bspwm/examples` y nos copiamos el "bspwmrc" al directorio que acabamos de crear: `cp bspwmrc ~/.config/bspwm/` y el "sxhkdrc" al suyo:
`cp sxhkdrc ~/.config/sxhkd/`
El bspwmrc necesita permisos de ejecución (`chmod +x bspwmrc`)

También instalamos ya la kitty:
`sudo apt install kitty -y`

Creamos el directorio y archivo:
`mkdir /home/kali/.config/bspwm/scripts/` y
`touch bspwm_resize` y le damos permisos de ejecución (`chmod +x bspwm_resize`)

El script `bspwm_resize` :

```
#!/usr/bin/env dash

if bspc query -N -n focused.floating > /dev/null; then
        step=20
else
        step=100
fi

case "$1" in
        west) dir=right; falldir=left; x="-$step"; y=0;;
        east) dir=right; falldir=left; x="$step"; y=0;;
        north) dir=top; falldir=bottom; x=0; y="-$step";;
        south) dir=top; falldir=bottom; x=0; y="$step";;
esac

bspc node -z "$dir" "$x" "$y" || bspc node -z "$falldir" "$x" "$y"
```

Instalando la polybar

pasamos a usuario root y usamos el comando:

`apt install cmake cmake-data pkg-config python3-sphinx libcairo2-dev libxcb1-dev libxcb-util0-dev libxcb-randr0-dev libxcb-composite0-dev python3-xcbgen xcb-proto libxcb-image0-dev libxcb-ewmh-dev libxcb-icccm4-dev libxcb-xkb-dev libxcb-xrm-dev libxcb-cursor-dev libasound2-dev libpulse-dev libjsoncpp-dev libmpdclient-dev libuv1-dev libnl-genl-3-dev` (si da problemas sphinx, se quita del comando).

Pasamos a usuario **no-root** , vamos a /Downloads y clonamos:
`git clone --recursive https://github.com/polybar/polybar`
Nos movemos al dir /polybar `cd polybar` y creamos el dir /build `mkdir build`, nos movemos a él `cd build` y ejecutamos:
`cmake ..`

`make -j$(nproc)`
`sudo make install`

Si todo lo anterior da problemas, instalamos la polybar con: `sudo apt install polybar`

Pasamos de nuevo a root y ejecutamos:
`apt install meson libxext-dev libxcb1-dev libxcb-damage0-dev libxcb-xfixes0-dev libxcb-shape0-dev libxcb-render-util0-dev libxcb-render0-dev libxcb-composite0-dev libxcb-image0-dev libxcb-present-dev libxcb-xinerama0-dev libpixman-1-dev libdbus-1-dev libconfig-dev libgl1-mesa-dev libpcre2-dev libevdev-dev uthash-dev libev-dev libx11-xcb-dev libxcb-glx0-dev`

Instalando picom y retocando sxhkdrc

Volvemos al usuario **no-root** y vamos al dir Descargas `cd Downloads` y clonamos:
`git clone https://github.com/ibhagwan/picom.git`
Entramos al nuevo directorio `cd picom` y ejecutamos los siguientes comandos:
`git submodule update --init --recursive`

`meson --buildtype=release . build` *-> si da problemas probar* `sudo apt install libpcre3-dev`

`ninja -C build`
`sudo ninja -C build install`

---

posible solución al "congelamiento" de la maquina virtual. (VMware) apagar la maquina virtual click derecho sobre la maquina virtual y click en 'Open VM directory' abran el archivo que termina en .vmx con el Notepad al final agregarle: keyboard.allowBothIRQs = FALSE keyboard.vusb.enable = TRUE guardar e iniciar la maquina virtual.

Instalamos **rofi** como root :
`apt install rofi` y pasamos a usuario **no root**
editamos el `~/.config/sxhkd/sxhkdrc` para añadir el atajo a rofi (Win+d). En el apartado

"program launcher" añadimos `super + d` y la ejecución es: `rofi -show run`
como root de nuevo instalamos bspwm:
`apt install bspwm`

---

ejecutamos un `kill -9 -1` para ir al panel de inicio de sesión y elegimos bspwm como entorno.
Ahora podemos configurar un poco más el `~/.config/sxhkd/sxhkdrc` para editar los atajos a nuestro gusto y añadir firefox como nuevo atajo `win + shift + f`
-Close and kill : `super + shift + q`
-Quit/Restart bspwm: `super + shift + r`

Configurando fuentes, la kitty e instalación de Feh

Vamos a www.nerdfonts.com para descargarnos la Hack Nerd Font
Cambiamos al dir Downloads, pasamos a root y movemos el .zip descargado con la fuente: `mv Hack.zip /usr/local/share/fonts/` y nos movemos hacia ese directorio: `cd /usr/local/share/fonts/` y descomprimimos `unzip Hack.zip` y borramos `rm Hack.zip`

Para habilitar el copiado/pegado bidireccional (entre host y vm) abrimos el archivo `~/.config/bspwm/bspwmrc` y al final del mismo añadimos: `vmware-user-suid-wrapper &`

Pasamos a root de nuevo e instalamos zsh en caso de no tenerlo `apt install zsh`

Volvemos a usuario **no-root** y vamos al directorio `~/.config.kitty` y creamos el kitty.conf `nano kitty.conf` y lo editamos. Hacemos lo mismo con el colors.ini .

Volvemos a root e instalamos:
`apt install zsh-autosuggestions zsh-syntax-highlighting zsh-autocomplete`

Visitamos github para obtener la última versión de la kitty y descargamos el archivo correspondiente (linux amd64 binary bundle). Nos movemos a /opt `cd /opt/` creamos el dir kitty `mkdir kitty` y ahí dentro `cd kitty` movemos el archivo descargado: `mv /home/kali/Downloads/kitty-version.txz .` y lo descomprimimos con 7z `7z x kitty-version.txz` y podemos borrar el .txz `rm kitty-version.txz` . Nos queda otro comprimido así q lo descomprimimos: `7z -xf kitty-version.tar` y lo borramos también con `rm kitty-version.tar` .

En el directorio `/opt/kitty/bin` tenemos la nueva kitty. Borramos la kitty en binario antigua con `apt remove kitty` y ya podemos decirle al sxhkdrc la ruta absoluta de la nueva kitty q queremos usar `/opt/kitty/bin/kitty`.

Vamos al dir `~/.config/kitty` como root y copiamos todo lo del directorio del usuario **no-root** aquí `cp /home/kali/.config/kitty/* .` Así tenemos la misma configuración para el super usuario y para el usuario no-root.

Podemos instalar `sudo apt install imagemagick` y pasamos también a instalar Feh: `sudo apt install feh` y una vez instalado tenemos que añadir la siguiente línea al final del bspwmrc: `feh --bg-fill RUTA/ABSOLUTA/fondo.png &` dónde "fondo.png" será el nombre del archivo que quieras usar como fondo de pantalla. Recuerda declarar la ruta absoluta o no lo detectará.

Desplegando la polybar

Nos vamos a Downloads como usuario **no-root** `cd Downloads` y clonamos lo siguiente: `git clone https://github.com/VaughnValle/blue-sky.git` Ahora procedemos a movernos al dir /Downloads/blue-sky/polybar y usamos el siguiente comando para copiar todo de manera recursiva al dir ~/.config/polybar: `cp -r * ~/.config/polybar/`

También hay que decirle al bspwmrc q al lanzar el entorno nos despliegue la polybar: `echo "~/.config/polybar/./launch.sh" >> ~/.config/bspwm/bspwmrc` Atención al doble "mayor que", si solo se usa uno vamos a reemplazar todo el bspwmrc por el "launch.sh".

Nos movemos al dir /home/kali/Downloads/blue-sky/polybar/fonts y desde ahí copiamos todo hacia `sudo cp * /usr/share/fonts/truetype` Después usamos: `fc-cache -v`

Ahora al pulsar `win + shift + r` se debería de actualizar el entorno y mostrarnos la polybar en la parte superior

---

Configurando picom

Nos vamos al dir ~/.config y creamos el dir picom `mkdir picom` nos movemos a él y crearemos el archivo de configuración picom.conf `nano picom.conf` .

Una vez creado abrimos el bspwmrc y añadimos al final: `picom &` Y ya deberían aplicarse las transparencias y otras configuraciones del picom. Si quieres quitar los bordes añadimos también `bspc config border_width 0` .

Configurando zsh e instalando la Powerlevel10k

Usamos: `git clone --depth=1 https://github.com/romkatv/powerlevel10k.git ~/powerlevel10k` echo 'source ~/powerlevel10k/powerlevel10k.zsh-theme' >>~/.zshrc `en el mismo comando. Y al usar` zsh` nos debería saltar el wizard de la Powerlevel10k

Abrimos el .p10k.zsh con nano `nano ~/.p10k.zsh` y podemos comentar el right side para quitar iconos del lado derecho de la powerlevel y añadimos en el left-prompt-elements `context` `command_execution_time` `status`

Tenemos ahora que repetir el proceso para el usuario root

De darnos error al ejecutar `zsh` desde root una vez finalizado lo interior usamos `compaudit` y cambiar el propietario del archivo de la ruta `chown root:root RUTA/COMPLETA/DELRESULTADO/DEL-COMPOUND`

Asignamos tanto al user normal como al root la zsh:
`usermod --shell /usr/bin/zsh kali`
`usermod --shell /usr/bin/zsh root`

También creamos un link simbólico del archivo .zshrc del usuario al usuario root, para que éste reciba todos los cambios que hagamos en el .zshrc del usuario. Como root tenemos que usar: `ln -s -f /home/kali/.zshrc .zshrc` Abrimos el archivo .zshrc y agregamos la ruta completa a la línea donde pone `source` del powerlevel para evitar errores.

Volvemos a hacer con el usuario root:
Abrimos el .p10k.zsh con nano `nano ~/.p10k.zsh` y podemos comentar el right side para quitar iconos del lado derecho de la powerlevel y añadimos en el left-prompt-elements `context` `command_execution_time` `status`

Para poner un icono a la p10k cuando estamos como usuario root entramos a configurar el /root/.p10k.zsh y en la línea "CONTEXT_ROOT_TEMPLATE" podemos poner el icono que queramos que se muestre al ser root. En la línea "DIR_ANCHOR" podemos quitar el bold.
El comando para reconfigurar por UI la powerlevel es: `p10k configure`

Definiendo propiedades de consola e instalando batcat y lsd

Vamos a `https://github.com/sharkdp/bat/releases` y descargamos la última versión del archivo amd64.deb
Vamos al dir Downloads y `dpkg -i bat-version-amd64.deb`
Podemos usar `sudo apt install zstd` para que no dé problemas lsd
Hacemos lo mismo con lsd, vamos a `https://github.com/lsd-rs/lsd/releases` lo descargamos y `dpkg -i lsd-version-amd64.deb`

Instalamos locate `sudo apt install locate` y hacemos un `updatedb` Si da error hay que desmontar la ruta que nos da el updatedb con umount `umount /run/user/1000/doc` y volvemos a pasar el `updatedb` .

Ahora tenemos que crear los links simbólicos para nuestro bat y lsd al usar cat y ls:
En el archivo `.zshrc` Creamos nuestros custom aliases:

```
# Custom Aliases
# -------------------------------------------
# bat

alias cat='bat'
alias catn='bat --style=plain'
alias catnp='bat --style=plain --paging=never'

# ls

alias ll='lsd -lh --group-dirs=first'
alias la='lsd -a --group-dirs=first'
alias l='lsd --group-dirs=first'
alias lla='lsd -lha --group-dirs=first'
alias ls='lsd --group-dirs=first'
```

```
Posteriormente, sobre el archivo ‘.deb‘, ejecutad los siguientes comandos:

ar x lsd_1.0.0_amd64.deb (O la versión que corresponda si te has descargado
otra)
unzstd control.tar.zst
unzstd data.tar.zst
xz control.tar
xz data.tar
rm lsd_1.0.0_amd64.deb
ar cr lsd_1.0.0_amd64.deb debian-binary control.tar.xz data.tar.xz
dpkg -i lsd_1.0.0_amd64.deb
```

Configurando la polybar

Nos movemos a `cd ~/.config/polybar`

En Launch especificamos qué es lo que tiene q lanzar la polybar al iniciarse.

En `current.ini` es donde configuramos las barras y los módulos, cogemos de ejemplo uno de los ya creados y construimos nuestras propias barras y módulos al gusto. También

tenemos que añadir la fuente deseada a continuación de las que ya están definidas siguiendo el mismo patrón: `font-7 = "Hack Nerd Font Mono:size=13;3"` .

La mayoría de opciones importantes de la polybar están en el `current.ini` .

En el `launch.sh` es en el que tenemos que añadir los módulos que creemos para que los lance al iniciar, seguimos el patrón de los otros para hacer los nuestros.

Para crear nuestros propios módulos lo haremos en el `current.ini` copiando una barra ya existente y cambiando los parámetros necesarios al gusto o para los scripts añadiremos simplemente:

```
[module/nombre_del_modulo]
type = custom/script
interval = 2
exec = ~/.config/bin/nombre_del_script a usar por el módulo
```

Recordad que hay que crear esos scripts a mano en la ruta especificada.
Como ejemplo el script que comprueba cuál es nuestra ip privada actual sería:

```
#!/bin/sh
echo "%{F#2495e7}el_icono %{F#ffffff} $(/usr/sbin/ifconfig eth0 | grep "inet "
| awk `{print $2}`)%{u-}"
```

Y el de la vpn por ejemplo:

```
#!/bin/sh

IFACE=$(/usr/sbin/ifconfig | grep tun0 | awk '{print $1}' | tr -d ':')

if [ "$IFACE" = "tun0" ]; then

        echo "%{F#1bbf3e}ICONO %{F#ffffff}$(/usr/sbin/ifconfig tun0 | grep
"inet " | awk '{print $2}')%{u-}"
```

```
else

        echo "%{F#1bbf3e}ICONO %{u-} Disconnected"
fi
```

Hay que hacer lo mismo para el apartado módulos y luego añadirlos al `launch.ini`

Más módulos para la polybar

En este caso configuramos la barra derecha "target" para ver en todo momento la máquina víctima objetivo.

Primero creamos la barra `[bar/target]` en el `current.ini` y la ponemos en su posición. Le decimos que el módulo será target_machine y creamos el módulo.

Vamos a `~/.config/bin` creado previamente para almacenar los scripts y creamos el target_machine.sh y le daremos permisos de ejecución. También creamos ahora el archivo `target` para que el script funcione correctamente ya que es necesario.

```
#!/bin/bash

ip_address=$(/bin/cat /home/s4vitar/.config/bin/target | awk '{print $1}')

machine_name=$(/bin/cat /home/s4vitar/.config/bin/target | awk '{print $2}')

if [ $ip_address ] && [ $machine_name ]; then

        echo "%{F#e51d0b}ICONO %{F#ffffff}$ip_address%{u-} - $machine_name"

else

        echo "%{F#e51d0b}ICONO %{u-}%{F#ffffff} No target"

fi
```

En el `~/.zshrc` creamos nuevas funciones:

```
function settarget(){
    machine_name=$1
    ip_address=$2
    echo "$ip_address $machine_name" > /home/s4vitar/.config/bin/target
}
```

```
function cleartarget(){
        echo '' > /home/s4vitar/.config/bin/target
}
```

En el `workspace.ini` podemos configurar los colores de la barra central con el entorno de trabajo activo, los inactivos, en uso y demás. También aquí es donde podemos modificar los iconos de ésta.

Plugins y NvChad + neovim

Buscamos en el navegador `sudo plugin zsh` nos ponemos como root y vamos al directorio `cd /usr/share` y creamos el directorio: `mkdir zsh-sudo` y le cambiamos el propietario y el grupo a los del user no privilegiado: `chown kali:kali zsh-sudo` nos movemos a ese nuevo dir y cambiamos al usuario **no-root** para tirar un:
`wget https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/plugins/sudo/sudo.plugin.zsh`
Y le daremos permiso de ejecución: `chmod +x sudo.plugin.zsh`
Abrimos el `.zshrc` y ponemos lo siguiente:

```
if [ -f /usr/share/zsh-sudo/sudo.plugin.zsh ]; then
        source /usr/share/zsh-sudo/sudo.plugin.zsh
fi
```

Este es el procedimiento a la hora de añadir cualquier plugin.

Ahora nos ponemos como usuario root y: `apt install npm`
Como usuario **no-root** borramos el dir `rm -r ~/.config/nvim`

Pasamos a clonar el NvChad con un:
`git clone https://github.com/NvChad/NvChad ~/.config/nvim --depth 1`

Ahora vamos al github de neovim para descargar la última versión `nvim-linux64.tar.gz`
Volvemos a usuario root y `cd /opt` y movemos lo descargado aquí:
`mv /home/kali/Downloads/nvim-linux64.tar.gz .` y `tar -xf nvim-linux64.tar.gz`
borramos el comprimido `rm nvim-linux64.tar.gz` y entramos al nuevo dir y al dir bin dentro de éste. Si ahora lo ejecutamos como usuario **no-root** veremos que nos pide si queremos instalar una config de ejemplo, decimos que no y ya se instalará automáticamente.

Lanzamos un `sudo apt remove neovim` pq solo queremos el que acabamos de instalar y solo nos quedará añadir la nueva ruta del nvim `/opt/nvim-linux64/bin` al $PATH en el `.zshrc`

Si queremos quitar las sugerencias en el nvim tenemos que ir a editar el `~/.config/nvim/lua/plugins/init.lua` , filtrar por "cmp" y comentar o borrar la función "load luasnips + cmp related in insert mode only" (líneas 153 a la 199 a la hora de hacer esta guía).

Para aplicar la configuración de nvim para el usuario root debemos copiar todo lo que hay en el `.config/nvim` del usuario al directorio para el root :
Como root :
`cd ~/.config/`
`rm -rf nvim`
`cp -r /home/kali/.config/nvim .`
Y ya debería instalarse la config al iniciarlo como root.

Podemos instalar el FZF si queremos, como usuario **no-root** `git clone --depth 1 https://github.com/junegunn/fzf.git ~/.fzf ~/.fzf/install`

Notas: hay algunas partes que no he contemplado porqué no las uso como el i3lock, o el flameshot. Dicho lo cual el proceso de instalación no difiere de cualquier otro package.