

2 de junio de 2023

Desarrollo de Entorno Profesional para la Ciberseguridad

Realizado por:

Álvaro González Sayago en CFGS 2ASIR



SUMMARY SHEET – PROJECT

Title of the project: Development of a Professional Environment for Cybersecurity	
Author: Alvaro Gonzalez Sayago	Date: 06/02/2023
Tutor: Lopez Fernandez, Olga Maria	
Title: CFGS 2 Network Computer Systems Administration	
Keywords: Cybersecurity, professional environment, kitty, bspwm, zsh, parrot, arch, black arch, pentesting, picom, polybar, sxhdk.	
Project summary: Create a professional environment oriented to cybersecurity completely from scratch with a base distribution, being able to fully customize both the graphic design of our environment and the keyboard shortcuts for the most comfortable use possible.	

INDEX

SUMMARY SHEET – PROJECT.....	2
Introduction	6
Introduction to memory.	6
Description.	6
General objectives and benefits.....	6
Personal motivations.....	7
Memory structure.	7
Viability study.....	8
Introduction.....	8
System Requirements.....	8
Requirements.....	8
Prioritization of requirements.....	8
Alternatives and selection of the solution.....	9
Alternative 1. KITTY on Arch Linux.	9
Alternative 2. BSPWM over Parrot.....	9
Conclusions.....	9
Project planning.....	10
Temporary planning.....	10
Risk assessment.....	eleven
Possible risks.....	eleven
Contingency plan.	eleven
Budget.....	eleven
Estimate of material cost	eleven
Summary and cost-benefit analysis.	eleven
Conclusions	eleven
Benefits.....	eleven
Disadvantages	eleven
Software analysis.....	12
Introduction.....	12
Software used.....	12
pacstrap	12
awesomeWM	12
PICOM	13
ZSH	13

LSD	14
BAT	14
powerlevel10k	fifteen
FZF	fifteen
NeoVim	16
mdcat	16
NeoFetch	17
NvChad	17
BurpSuite	18
nmap	18
RustScan	19
place	19
Packet	19
Reply	twenty
whatweb	twenty
wfuzz	twenty-one
Evil-WinRM	twenty-one
MetaSploit	22
ExploitDB	22
Environment customization.....	2. 3
Introduction.....	2. 3
Personalization	2. 3
Shortcuts & Conveniences Manual	26
Introduction.....	26
Shortcuts and conveniences.....	26
General tests	27
Introduction.....	27
Usage tests.....	27
Results obtained.....	29
Conclusions	29
Webgraphy.....	30
Final conclusion.....	31
End of document	31

Introduction

Introduction to memory.

In this memory I will compile all the necessary documentation to explain my project as well and as completely as possible, in addition, it will be divided by the sections that I consider are most important for its subsequent explanation.

The documentation will consist of information such as the objectives of the project, costs, requirements, advantages, installation/configuration manuals, planning, webgraphy, glossary of words, etc.

Description.

This project is based on the creation of a professional cybersecurity environment, Configured completely from scratch, it will use license-free software and packages, plus a clean, base distribution.

Of the packages that the project will be composed of, most of all must be installed and configured, the installation source will always be provided.

In addition to the package configuration, with this project we will achieve an easy-to-use and highly customizable system.

Emphasize that the inspiration of this project is thanks to S4vitar:

<https://www.youtube.com/@s4vitar>

<https://www.twitch.tv/s4vitaar>

<https://s4vitar.github.io>

General objectives and benefits.

The general objectives of this project are:

- Learn about the different packages that are used.
- Prioritize user comfort.
- Get a fully customizable interface.
- Get modifiable keyboard shortcuts.
- Put an end to a system suitable for carrying out pentesting and cybersecurity techniques.
- Use of zero-cost packages through free or open source licenses.

Personal motivations.

It is a project that really catches my attention since it is getting a completely different environment from the one we have at the beginning of the project as well as having many new features.

It caught my attention from the beginning for being based on Linux and having so many different packages with many different utilities, each with its own configuration.

I propose it as a challenge to myself since with so much work on a single distribution, it is very difficult for several errors not to appear along the way, be they versions, configurations, code, etc.

Memory structure.

I will try to make the structure as simple as possible. It will be divided into 12 blocks, which will be the following:

- Introduction. (This same block)
- Viability study. (Block below)
- Analysis.
- Environment customization.
- Shortcuts & Comforts Manual.
- General tests
- webgraphy
- Final conclusion
- End of document

Exception:

- Installation & Configuration Manual. (separate document)

Each block may be divided into sections, and these same sections into subsections.

Each block will be separated from the previous one by a page break (if they are large enough), so that the document is as clean as possible.

Between section and section there will be a minimum space to increase the clarity of the document.

Viability study

Introduction.

In this block I will document the viability of the project in the best possible way, in general aspects, I will talk about requirements, alternatives, resource planning, risks, etc. It will be the largest section and therefore the one that organizes the project the most.

System Requirements

Requirements.

As I said before, the requirements will be practically the same as the base distributions to use. In this case, the base distribution will be Arch Linux:

	MINIMUM	RECOMMENDED
ARCHITECTURE	x86	x64
RAM	512MB	2GB
STORAGE	1GB	20GB

As a personal recommendation, since you will be using quite a few large packages and configuration files, as well as a GUI installation from scratch, I would use the recommended requirements configuration.

Prioritization of requirements.

The main requirements would be storage and RAM, although 2GB is recommended, I would try to get 4GB or even 6. Also, for a good operating system, I would also try to reach 40GB of storage to ensure that we do not end up with a lack of space.

Alternatives and selection of the solution.

This project can be covered in two large ways (they are the ones I have proposed/found, but it can really be done in infinitely different ways and with any distribution).

Alternative 1. KITTY on Arch Linux.

The first alternative is to use KITTY (GPU-accelerated terminal emulator) and (Windows-like Window Manager, but for Linux) on top of Arch Linux. Using the packages:

- pacstrap
- Bootloader
- AUR
- vmware-tools
- awesomeWM
- PICOM
- ZSH
- LSD
- BAT
- powerlevel10k
- FZF
- neovim
- mdc



Alternative 2. BSPWM over Parrot.

The second and last alternative, but no less important, is to use BSPWM over Parrot. Using the packages:

- SXHKD
- Polybar
- ROFI
- PICOM
- slim lock
- powerlevel10k
- BAT
- LSD
- FZF
- ZSH
- Rangers
- Oh My Tmux
- fastTCPscan



conclusions.

The two alternatives are acceptable and very broad, in my case I will use the first since I consider that it can give more work and is more interesting. If the project in the end turned out not to be big enough, I would also do the second alternative (it would appear in this final document later).

Project planning

In this section I will explain what will be the structure to follow when creating the project, I will try to divide the planning into steps as small as possible so that everything important is compiled.

Temporary planning.

The planning of the project will be as follows:

- 1.**VMware Download.
- 2.**Arch ISO download.
- 3.**Creation of the virtual machine with Arch.
- 4.**Define partitions with cfdisk.
- 5.**Define SWAP partition.
- 6.**Format the partitions.
- 7.**Installation of necessary packages with PACSTRAP.
- 8.**Create fstab file.
- 9.**GRUB installation.
- 10.**We enable AUR.
- eleven.**Installing the Black Arch repositories on Arch Linux.
- 12.**Installation of the graphical interface.
- 13.**Installation of KITTY, VMware tools, Firefox, AwesomeWM and PICOM.
- 14.**Installation of plugins for ZSH, LSD and BAT.
- fifteen.**KITTY and PICOM configuration.
- 16.**Installation and configuration of Powerlevel10k, FZF, Neovim.
- 17.**Installation of pentesting tools.
- 18.**Key configuration.
- 19.**Definition of the new custom shortcuts.
- twenty.**Installation of MDCAT to view Markdown files.

Note that each of the packages used will later have its explanation section on its operation, configuration and links/commands of interest.

Risk assessment.

In this section I will name the biggest and most common mistakes that can happen to us, as well as how to avoid them.

Possible risks

As it is a process that takes several hours on the same system, installing and doing various configurations, and several of them have to do directly with the system, it is quite likely that we may have a critical failure at some point in the process.

Contingency plan.

As a recommendation, I would take snapshots in the virtual machine after each installation/configuration to be carried out correctly, so if we make a mistake we can return to the previous step without problem.

Also, I would try not to have more than 3 snapshots active at the same time to avoid unnecessary storage. For this we can delete old snapshots as we progress in the project.

Once we have the final OS, we will take a last snapshot with our system working perfectly.

Budget

In this section I will present the material cost in addition to the cost-benefit analysis of the project.

material cost estimate

This project will cost us a total of €0. All packages, distributions, images and configurations used are open source or freely licensed so anyone can have it.

Summary and cost-benefit analysis.

Being a professional work environment, with a cost of €0 we could get infinite profitability, it depends on our work.

conclusions

Benefits

- Obtaining a professional pentesting environment.
- 100% customizable environment.
- Totally free and mostly open source.
- Configurable shortcuts.

Drawbacks

- Long to do.
- Many different packages.
- Many different configurations.
- Many steps to follow.

software analysis

Introduction

In this block, I will name all the software used for the project. I'll skip the basics like VMware but I'll document the utilities I consider weirder. In addition, I will insert if possible its official page or, failing that, its GitHub page.

Software used.

Section where I will document in the most complete way possible each utility of the project.

pacstrap

Pacstrap is a package installer that is included in the base Arch Linux distribution. Similar to apt-get or apt.

Links of interest:

Arch Handbook: <https://man.archlinux.org/man/pacstrap.8>

GitHub: <https://github.com/archlinux/arch-install-scripts/blob/master/pacstrap.in>

awesomeWM

Also called Awesome Window Manager, based on C and LUA. Designed to improve the graphical environment and designed for developers.



Links of Interest:

Official website: <https://awesomewm.org>

GitHub: <https://github.com/awesomeWM/awesome>

Intro video: https://www.youtube.com/watch?v=qKtit_B7Keo

PICOM

Picom is a composer for Xorg. Very useful when using it with window managers (Like AwesomeWM). It allows us to apply shadows, transparencies, 3D effects, etc.

Links of Interest:

Arch Handbook: <https://wiki.archlinux.org/title/picom>

GitHub: <https://github.com/yshui/picom>

Intro video: <https://www.youtube.com/watch?v=t6Klg7CvUxA>

ZSH

Also called. It is a very powerful terminal for Unix-based operating systems. It is also currently used on Kali Linux. OhMyZsh is completely optional but improves the use and configuration of ZSH.



Links of Interest:

Arch Handbook: <https://wiki.archlinux.org/title/picom>

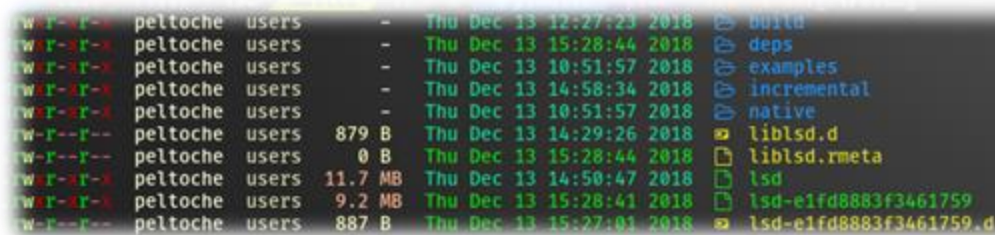
GitHub OhMyZSH: <https://github.com/ohmyzsh/ohmyzsh>

GitHubZSH: <https://github.com/zsh-users/zsh>

Intro video: <https://www.youtube.com/watch?v=ADytC9a2q2Y>

LSD

Also called LSDeluxe. It is an improvement of the LS command that allows us to color the output, add icons and make it much more visual.



```
rw-r--r-- peltoche users - Thu Dec 13 12:27:23 2018 build
rw-r--r-- peltoche users - Thu Dec 13 15:28:44 2018 deps
rw-r--r-- peltoche users - Thu Dec 13 10:51:57 2018 examples
rw-r--r-- peltoche users - Thu Dec 13 14:58:34 2018 incremental
rw-r--r-- peltoche users - Thu Dec 13 10:51:57 2018 native
rw-r--r-- peltoche users 879 B Thu Dec 13 14:29:26 2018 liblsd.d
rw-r--r-- peltoche users 0 B Thu Dec 13 15:28:44 2018 liblsd.rmeta
rw-r--r-- peltoche users 11.7 MB Thu Dec 13 14:50:47 2018 lsd
rw-r--r-- peltoche users 9.2 MB Thu Dec 13 15:28:41 2018 lsd-e1fd8883f3461759
rw-r--r-- peltoche users 887 B Thu Dec 13 15:27:01 2018 lsd-e1fd8883f3461759.d
```

Links of Interest:

GitHub: <https://github.com/lsd-rs/lsd>

Intro video: <https://www.youtube.com/watch?v=YQSGq9oWSTo>

BAT

An enhancement to the CAT command. Like LSD with LS, BAT allows us to see the output of a CAT but with colors and code identification.



Links of Interest:

GitHub: <https://github.com/sharkdp/bat>

Intro video: <https://www.youtube.com/watch?v=myq9zTf5aC8>

Power level 10k

Powerlevel10k is simply a theme for the ZSH shell. It is very easily configurable and also allows us to add icons to the prompt.



Links of Interest:

GitHub: <https://github.com/romkatv/powerlevel10k>

Intro video: <https://www.youtube.com/watch?v=KNLNyX9V1L8>

FZF

Also called Fuzzy Finder. It is a fast and optimized search engine for files and folders using the command line that shows us the files it finds in real time.



Links of Interest:

GitHub: <https://github.com/junegunn/fzf>

Intro video: <https://www.youtube.com/watch?v=J0QuhUThM5E>

NeoVim

As its name indicates, it is VIM but with an improved interface and utilities. It uses 30% less code than the original VIM.



Links of Interest:

Official website: <https://neovim.io>
GitHub: <https://github.com/neovim/neovim>
Intro video: <https://www.youtube.com/watch?v=x1TEI0MxmKI>

mdcat

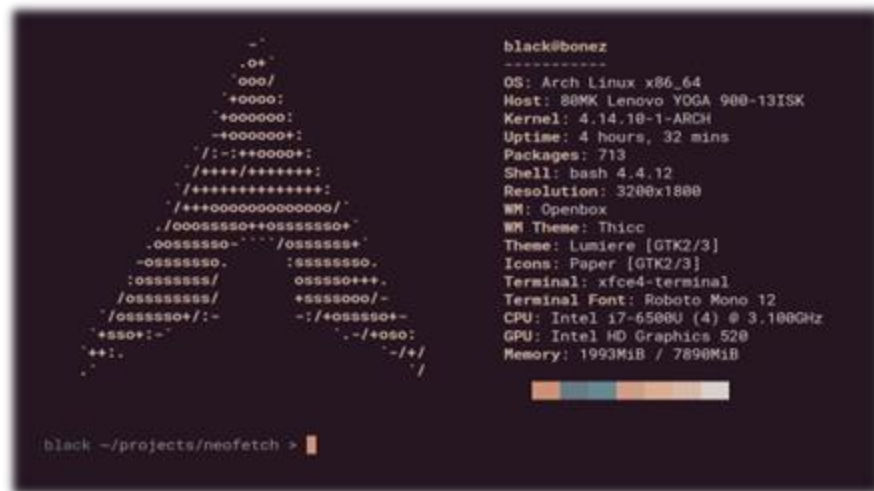
Mdcat allows us to easily view Markdown files through the console.

Links of Interest:

GitHub: <https://github.com/swsnr/mdcat>
Arch Handbook: <https://man.archlinux.org/man/community/mdcat/mdcat.1.en>
Intro video: <https://www.youtube.com/watch?v=HvAhAytRdY>

NeoFetch

Utility to search for system information easily from the command line.



Links of Interest:

GitHub: <https://github.com/dylanaraps/neofetch>
 Arch Handbook: <https://archlinux.org/packages/community/any/neofetch/>
 Intro video: <https://www.youtube.com/watch?v=btyrzunEwII>

NvChad

Utility that allows us to improve NeoVim and even make it very similar to VSCode.



Links of Interest:

GitHub: <https://github.com/NvChad/NvChad>
 Official website: <https://nvchad.com>
 Intro video: <https://www.youtube.com/watch?v=WaSriehjG6o>

burp suit

Graphical platform to test the security of Web applications. It has a Professional and Community version, the latter being free.

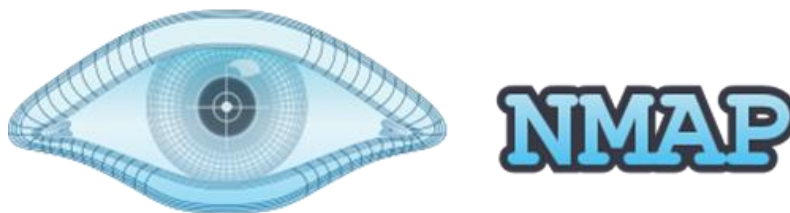


Links of Interest:

GitHub: <https://github.com/thehackingsage/burpsuite>
Official website: <https://portswigger.net/burp>
Intro video: <https://www.youtube.com/watch?v=nXm324qSfRA>

nmap

Default port scanner, the most used.



Links of Interest:

GitHub: <https://github.com/nmap/nmap>
Official website: <https://nmap.org>
Intro video: <https://www.youtube.com/watch?v=blZXOqJwZt4>

RustScan

A modern port scanner, comparable to Nmap.



Links of Interest:

- GitHub: <https://github.com/RustScan/RustScan>
- Archive page: https://archlinux.org/packages/community/x86_64/rustscan/
- Intro video: <https://www.youtube.com/watch?v=a6j8D6LhByM>

placate

Tool to quickly locate files throughout the system.

Links of Interest:

- GitHub: <https://github.com/Aetf/plocate>
- Archive page: <https://man.archlinux.org/man/plocate.1>
- Intro video: <https://www.youtube.com/watch?v=GnRvtqry1-M>

Packet

A series of tools created in Python to work with network protocols.

Links of Interest:

- GitHub: <https://github.com/fortra/impacket>
- Archive page: <https://archlinux.org/packages/community/any/impacket/>
- Intro video: https://www.youtube.com/watch?v=Ule_iBhhawQ

Reply

Tool to quickly obtain credentials and possible remote access.



Links of Interest:

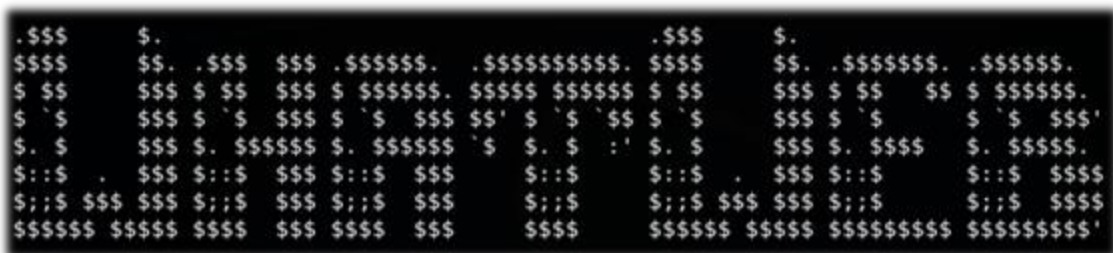
GitHub: <https://github.com/lqandx/Reply>

Archive page: <https://aur.archlinux.org/packages/responder>

Intro video: <https://www.youtube.com/watch?v=8VtzipIbvX2U>

whatweb

Tool to extract data and identify a specific website.



Links of Interest:

GitHub: <https://github.com/urbanadventurer/WhatWeb>

Archive page: <https://aur.archlinux.org/packages/whatweb>

Intro video: <https://www.youtube.com/watch?v=7Uo8bVL1OxI>

wfuzz

Utility to scan directory structures using brute force. It serves the same purpose as GoBuster but in a different way. You can also use ffuf, dirb, dirbuster, and dirsearch. They all serve the same purpose but it depends on the comfort of the user.



Links of Interest:

GitHub: <https://github.com/xmendez/wfuzz>
Archive page: <https://aur.archlinux.org/packages/wfuzz>
Intro video: <https://www.youtube.com/watch?v=AIj3pPKck9Y>

Evi I-Wi nRM

Used on Windows systems with the WinRM service active, normally running on port 5985. Created in ruby.



Links of Interest:

GitHub: <https://github.com/Hackplayers/evil-winrm>
Archive page: <https://aur.archlinux.org/packages/ruby-evil-winrm>
Intro video: <https://www.youtube.com/watch?v=-6j3P0VB8ak>

MetaSploit

Framework that gathers vulnerabilities, as well as CVEs for use in pentesting. It is not recommended to use it without looking directly at the scripts.



Links of Interest:

GitHub:	https://github.com/rapid7/metasploit-framework
Archive page:	https://wiki.archlinux.org/title/Metasploit_Framework
Official website:	https://www.metasploit.com
Intro video:	https://www.youtube.com/watch?v=ZOFsQi1NVsM

ExploitDB

Database with all registered CVEs and how to exploit them.



Links of Interest:

GitLab:	https://gitlab.com/exploit-database/exploitdb
Official website:	https://www.exploit-db.com
Intro video:	https://www.youtube.com/watch?v=ZeBwhK2hQr4

Environment customization.

Introduction

Next, I will show which are the files that we must edit to customize our environment. I will not show how to edit them since that is explained in the document for the installation and configuration of the environment.

Personalization

All the configuration files to customize our environment are in “~/.config”. Especially, important folders are **bspwm**, **kitty**, **picom** and **sxhkd**.

```

A> cd ~/.config ll
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 14:52:17 2023 bspwm
drwx----- alvarogs alvarogs 4.0 KB Sat Apr 22 15:16:16 2023 calcurse
drwx----- alvarogs alvarogs 4.0 KB Mon May 1 15:42:46 2023 dconf
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 14:52:17 2023 dunst
drwx----- alvarogs alvarogs 4.0 KB Sat Apr 22 13:44:35 2023 evolution
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 14:52:17 2023 eww
drwx----- alvarogs alvarogs 4.0 KB Sat Apr 22 14:53:28 2023 gnome-session
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 13:44:35 2023 gpg-1.0
drwx----- alvarogs alvarogs 4.0 KB Sat Apr 22 13:44:36 2023 gtk-3.0
drwx----- alvarogs alvarogs 4.0 KB Sat Apr 22 13:44:35 2023 ibus
drwxr-xr-x alvarogs alvarogs 4.0 KB Mon May 1 14:02:32 2023 kitty
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 13:44:46 2023 nautilus
drwxr-xr-x alvarogs alvarogs 4.0 KB Sun Apr 16 12:12:53 2023 neofetch
drwxr-xr-x alvarogs alvarogs 4.0 KB Mon May 8 15:31:05 2023 nvim
drwxr-xr-x alvarogs alvarogs 4.0 KB Mon May 1 15:15:31 2023 picom
drwx----- alvarogs alvarogs 4.0 KB Sat Apr 22 13:44:44 2023 pulse
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 14:52:17 2023 rofi
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 14:52:17 2023 starship
drwxr-xr-x alvarogs alvarogs 4.0 KB Sat Apr 22 14:52:17 2023 sxhkd
-rw-r--r-- alvarogs alvarogs 1.3 KB Sun Nov 27 23:10:57 2022 dleyna-renderer-service.conf
-rw-r--r-- alvarogs alvarogs 633 B Sat Apr 22 13:35:51 2023 user-dirs.dirs
-rw-r--r-- alvarogs alvarogs 1 B Sat Apr 22 13:35:51 2023 user-dirs.locale
A> cd ~/.config
  
```

File “**bspwmrc**”. It is used to define keyboard shortcuts and window behavior.

```

GNU nano 7.2 bspwm/bspwmrc
#!/bin/sh

#####
#
#
#
#
#
##### By: Rxyhn #####
#
##### ENV VARS #####
#
#####
## Environments
export PATH="${PATH}:${HOME}/.config/bspwm/bin"

# Parse colors from ~/.Xresources"
xrdb -override "${HOME}/.Xresources"

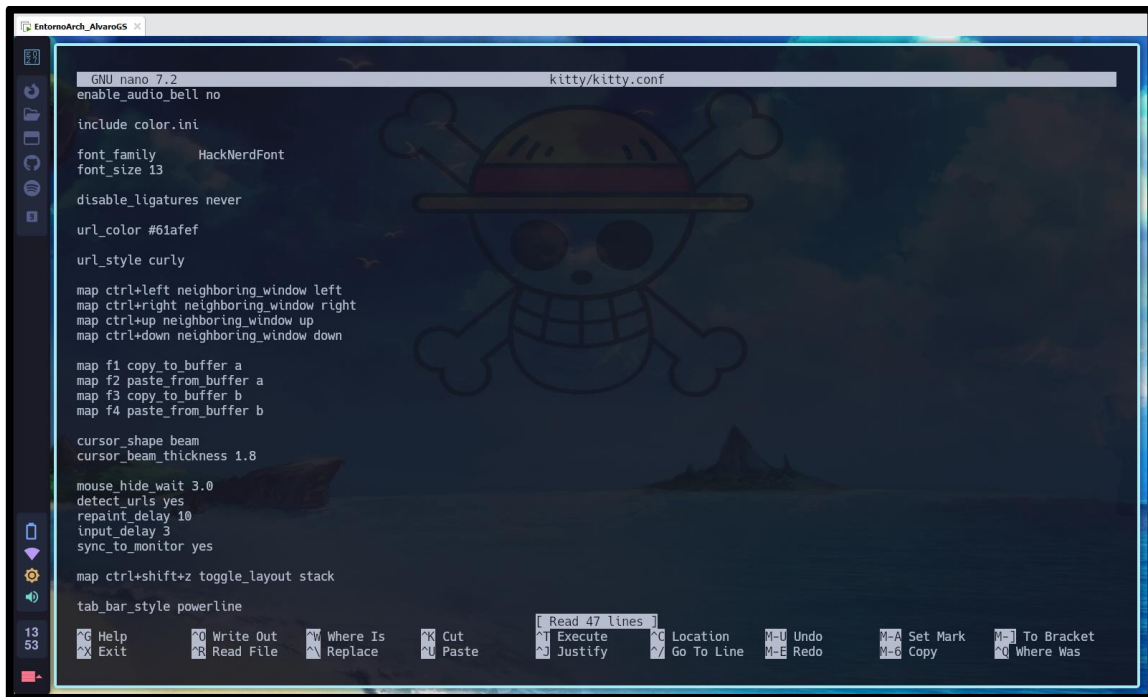
xrdb_query()
{
    [ -n "$XRDB_QUERY" ] || XRDB_QUERY="(xrdb -query)"

    echo "$XRDB_QUERY" | while IFS=: read -r STRING; do
        [ "${1}" = "${STRING%% *}" ] || continue
        echo "${STRING##*\ }"
        break
    done
}

getcolors()
{
    #FOREGROUND="${xrdb_query '*foreground:'}"

    Read 135 lines
    Execute Justify Location M-U Undo M-A Set Mark M-J To Bracket
    Paste M-E Redo M-C Copy M-Q Where Was
  
```

File “kitty.conf”.It is used to modify the appearance of our terminal, assign keyboard shortcuts and functionality to certain keys or combinations.



```

GNU nano 7.2 kitty/kitty.conf
enable_audio_bell no

include color.ini

font_family HackNerdFont
font_size 13

disable_ligatures never

url_color #61afef
url_style curly

map ctrl+left neighboring_window left
map ctrl+right neighboring_window right
map ctrl+up neighboring_window up
map ctrl+down neighboring_window down

map f1 copy_to_buffer a
map f2 paste_from_buffer a
map f3 copy_to_buffer b
map f4 paste_from_buffer b

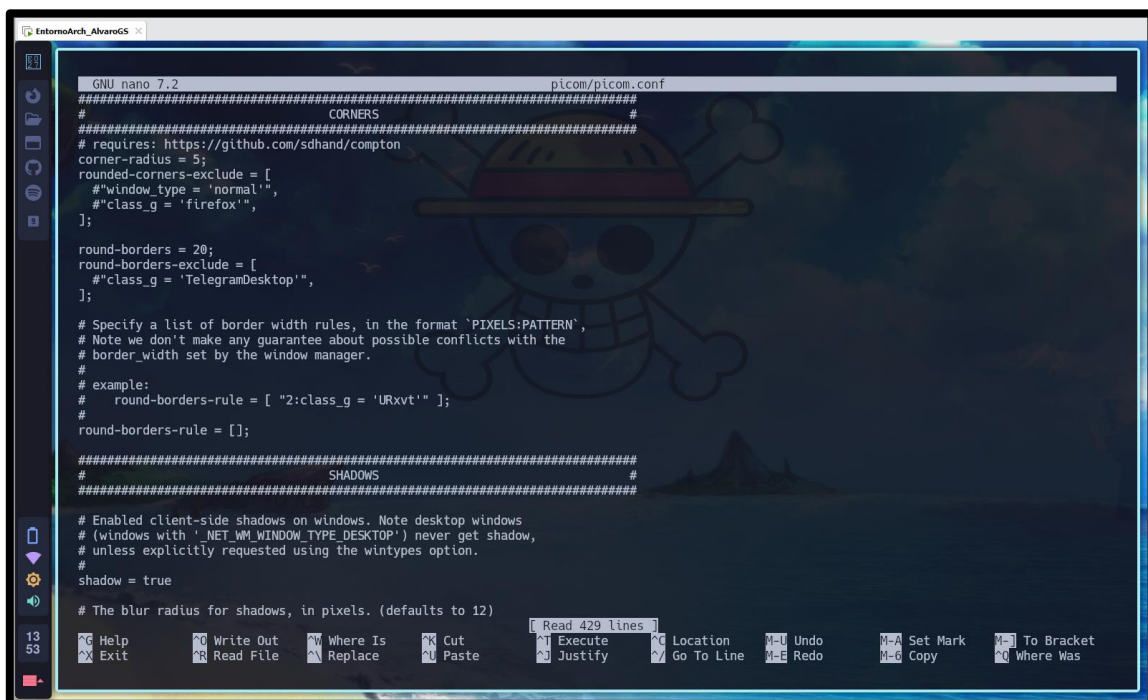
cursor_shape beam
cursor_beam_thickness 1.8

mouse_hide_wait 3.0
detect_urls yes
repaint_delay 10
input_delay 3
sync_to_monitor yes

map ctrl+shift+z toggle_layout stack

tab_bar_style powerline
  
```

File “picom.conf”.Used for transparency and opacity effects, window shadows, animations, and window border effects.



```

GNU nano 7.2 picom/picom.conf
#####
# CORNERS
#####
# requires: https://github.com/sdhand/compton
corner-radius = 5;
rounded-corners-exclude = [
# "window_type = 'normal'",
# "class_g = 'firefox'",
];

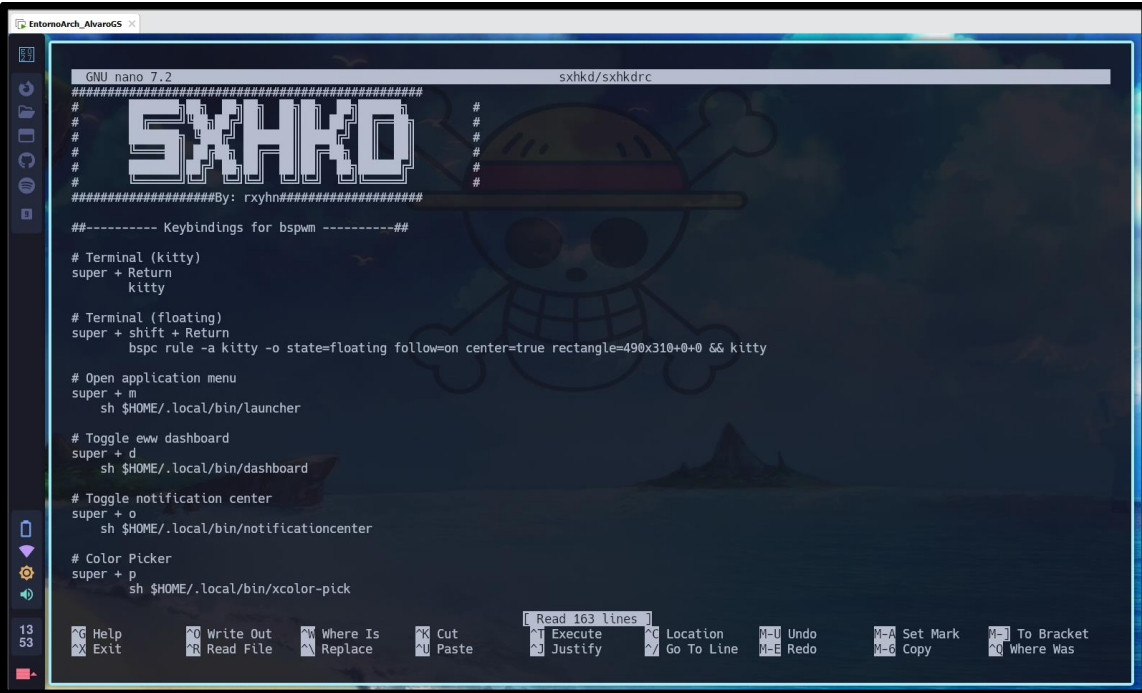
round-borders = 20;
round-borders-exclude = [
# "class_g = 'TelegramDesktop'",
];

# Specify a list of border width rules, in the format 'PIXELS:PATTERN'.
# Note we don't make any guarantee about possible conflicts with the
# border_width set by the window manager.
#
# example:
# round-borders-rule = [ "2:class_g = 'URxvt'" ];
round-borders-rule = [];

#####
# SHADOWS
#####
# Enabled client-side shadows on windows. Note desktop windows
# (windows with 'NET_WM_WINDOW_TYPE_DESKTOP') never get shadow,
# unless explicitly requested using the wintypes option.
#
shadow = true

# The blur radius for shadows, in pixels. (defaults to 12)
  
```


File "sxhkdr".Used for key combinations and keyboard shortcuts as well as custom commands.



```
GNU nano 7.2 sxhk/sxhkdr
#####
# SXHKD #
#####By: rxyhn#####
#----- Keybindings for bspwm -----##
# Terminal (kitty)
super + Return
kitty

# Terminal (floating)
super + shift + Return
bspc rule -a kitty -o state=floating follow=on center=true rectangle=490x310+0+0 && kitty

# Open application menu
super + m
sh $HOME/.local/bin/launcher

# Toggle eww dashboard
super + d
sh $HOME/.local/bin/dashboard

# Toggle notification center
super + o
sh $HOME/.local/bin/notificationcenter

# Color Picker
super + p
sh $HOME/.local/bin/xcolor-pick

[ Read 163 lines ]
13 53  Help Write Out Where Is Cut Execute Location M-U Undo M-A Set Mark M-J To Bracket
Exit Read File Replace Paste Justify Go To Line M-E Redo M-C Copy M-Q Where Was
```

Shortcuts & Conveniences Manual

Introduction

In this section, I will show the shortcuts that are configured by me by default when downloading the virtual machine. All of these shortcuts can be changed in the SXHKD file, I'll just document a few.

Additionally, I will leave inside the home of the MV itself, a cheatsheet file that has all the useful commands.

Shortcuts and conveniences

Utility	Combination
Search Command	Ctrl+R
search interactively	Ctrl+T
open terminal	SUPER+ENTER
Close terminal/tab	SHIFT+CTRL+W
change terminal	SUPER + ARROW + LEFT/RIGHT
open tab	SHIFT+CTRL+T
switch tab	SHIFT + CTRL + LEFT/RIGHT ARROW
rename tab	SHIFT + CTRL + ALT + T
lock system	CTRL+ALT+L
delete line	Ctrl+U
Emoji Table	CTRL+SHIFT+U
Restart GUI	CTRL+SHIFT+R
open firefox	CTRL+SHIFT+F
Open BurpSuite	CTRL+SHIFT+B

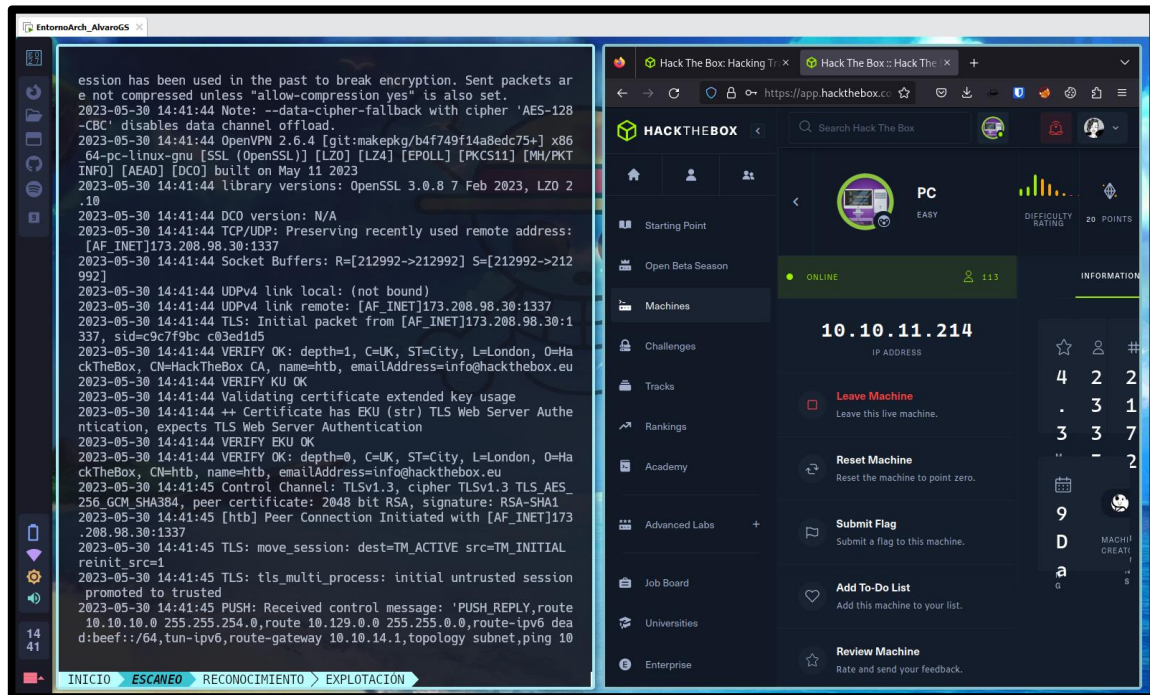
General tests

Introduction

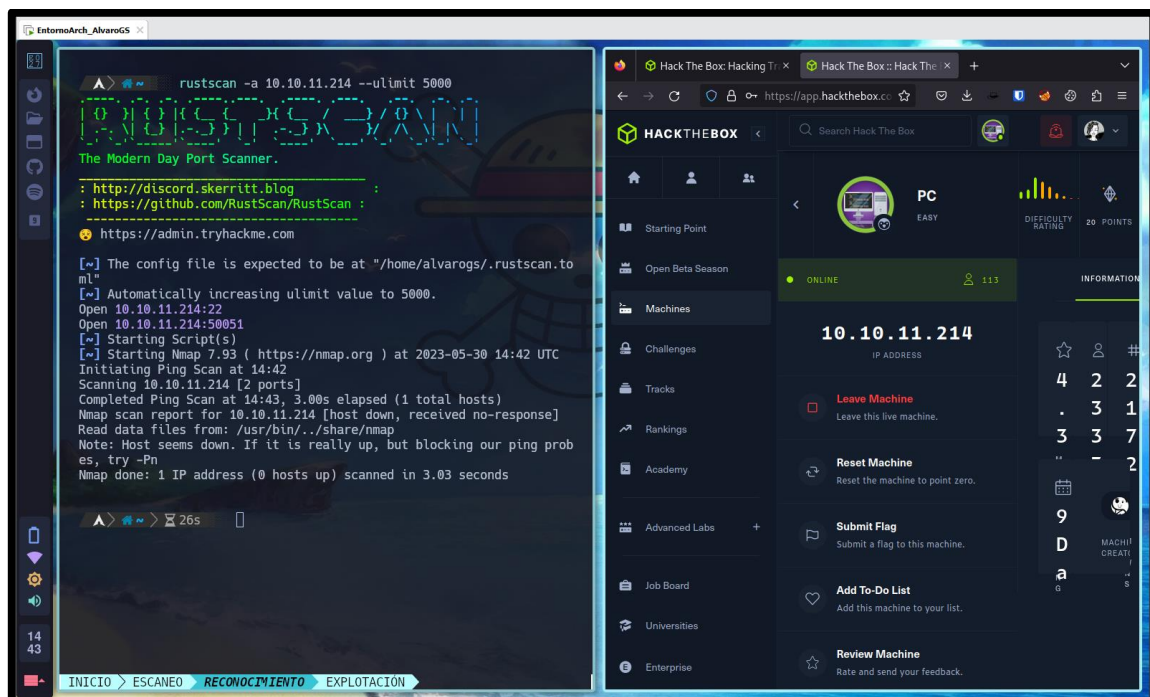
In this block, I will test the different utilities to verify their operation, like this as well as the performance of the system in general.

Usage tests.

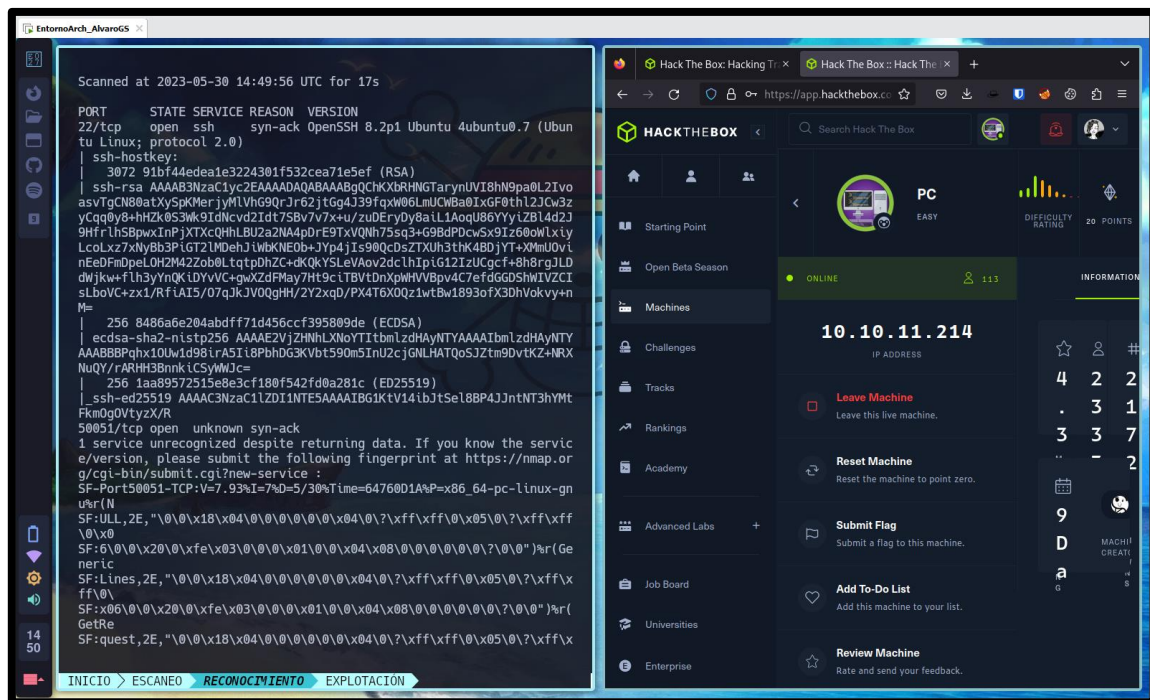
Using a HackTheBox VPN to connect to a virtual machine.



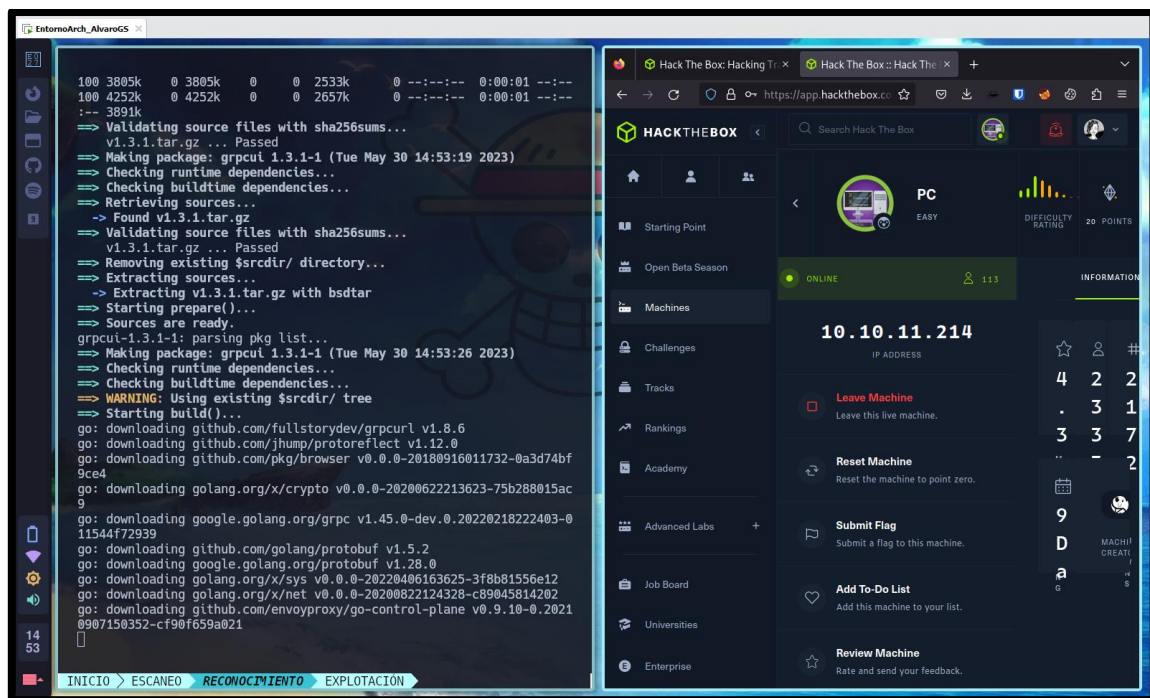
Initial port scan with Rustscan.



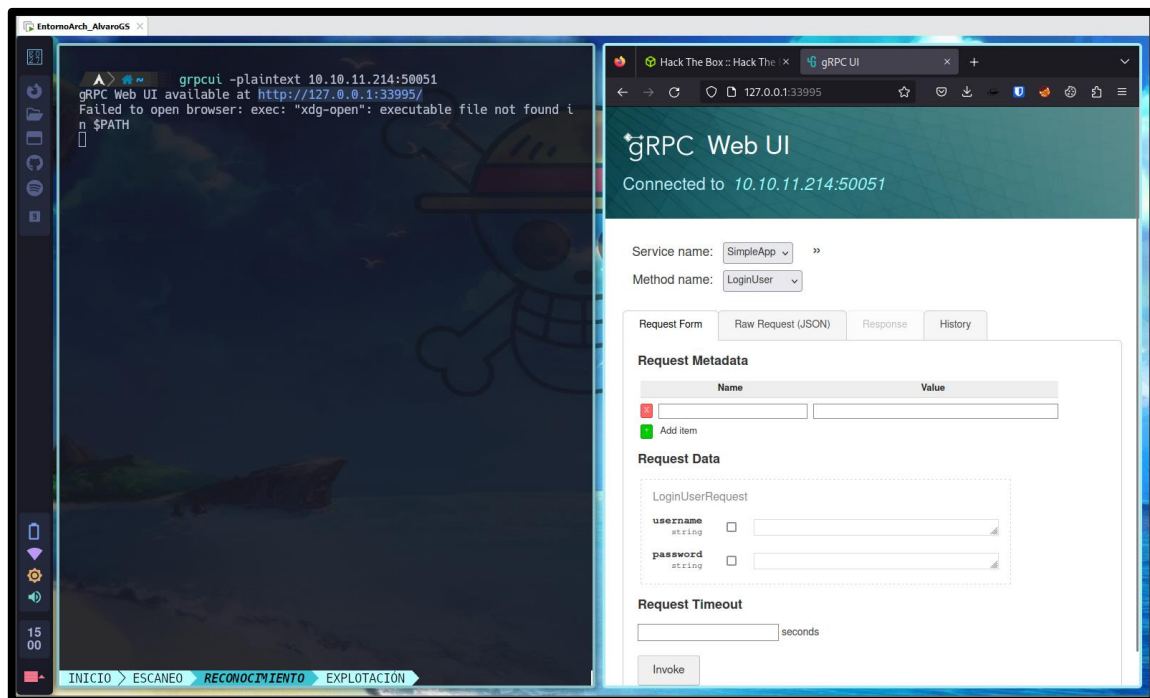
Intensive scan to ports found with RustScan (22 and 50051).



Installation of grpcui utility.



Using the new utility against the virtual machine to get a page exposed.



Results obtained

After using the environment for a while, making use of some utilities and keyboard shortcuts already shown above, I can say that it helps to work comfortably and quickly even without being 100% used to the environment yet.

conclusions

I consider it a useful and simple environment for pentesting, and even more so if I was used to 100%. Regarding the expectations I had, it could have been better but it still meets what I set out to do at the beginning.

webgraphy

<https://keepcoding.io/blog/what-is-metasploit-ciberseguridad/>

<https://www.ochobitshacenunbyte.com/2019/09/18/que-es-zsh-y-como-utilizar-sus-temas-y-complementos/>

<https://www.exploit-db.com/about-exploit-db>

<https://www.hackingarticles.in/a-detailed-guide-on-evil-winrm/>

<https://wfuzz.readthedocs.io/en/latest/>

<https://www.geeksforgeeks.org/rustscan-faster-nmap-scanning-with-rust/>

<https://nmap.org/book/>

<https://www.geeksforgeeks.org/what-is-burp-suite/>

<https://linuxhint.com/what-is-neofetch-for-linux/>

<https://neovim.io/charter/>

<https://www.freecodecamp.org/news/fzf-a-command-line-fuzzy-finder-missing-demoa7de312403ff/>

<https://platzi.com/tutoriales/2383-prework-linux/12567-bat-y-lsd/>

<https://www.instructables.com/Bspwm-Installation-and-Configuration/>

<https://distro.tube/quest-articles/installing-and-using-sxhkd.html>

<https://dev.to/abdfnx/oh-my-zsh-powerlevel10k-cool-terminal-1no0>

https://wiki.archlinux.org/title/Arch_User_Repository

<https://www.linuxfordevices.com/tutorials/linux/picom>

https://www.reddit.com/r/i3wm/comments/vxfy9/what_is_picom_used_for/

<https://itsfoss.com/pacman-command/>

Final conclusion

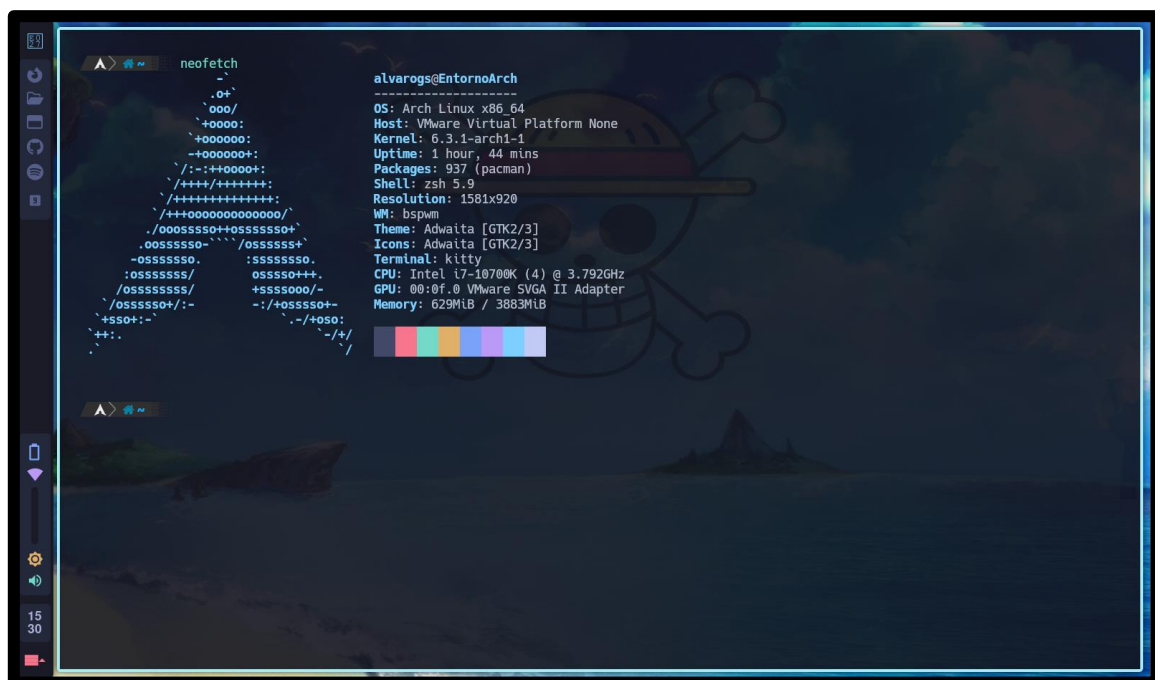
In summary, I really liked doing this project. I have found many errors and unforeseen events along the way but in the end thanks to the documentations and GitHub pages I have managed to find the solution.

Building the environment little by little and from scratch has also helped me see how an environment works on the inside and how they are built. As well as how to implement different graphic environments and how to edit them.

I'm also quite happy with the final performance it has, although it could be improved, especially when opening applications like FireFox, which takes a little longer than expected.

In addition, I have managed to make it useful enough to be used in cybersecurity work in a professional way, or at least comfortable enough to solve HackTheBox machines from the environment in a simple way.

In general, I am quite satisfied with my project.



End of document