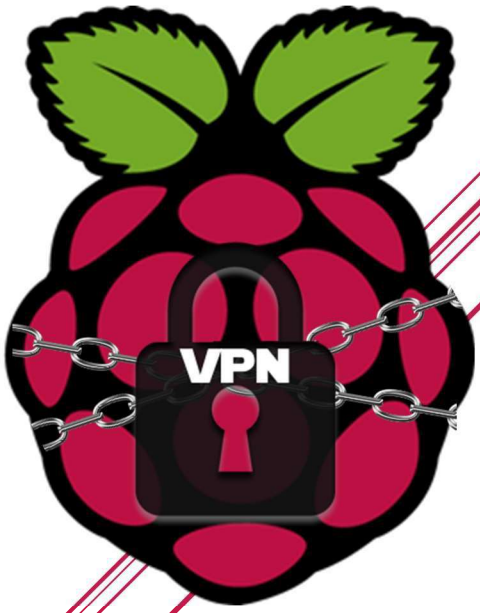
PRIVATE NETWORK SECURITY PROJECT

CARLOS D. YAQUE JEFFS

JUNE 15th, 2020



CEU

CEU ANDALUCIA

CARLOS D. YAQUE JEFFS

Table of Contents

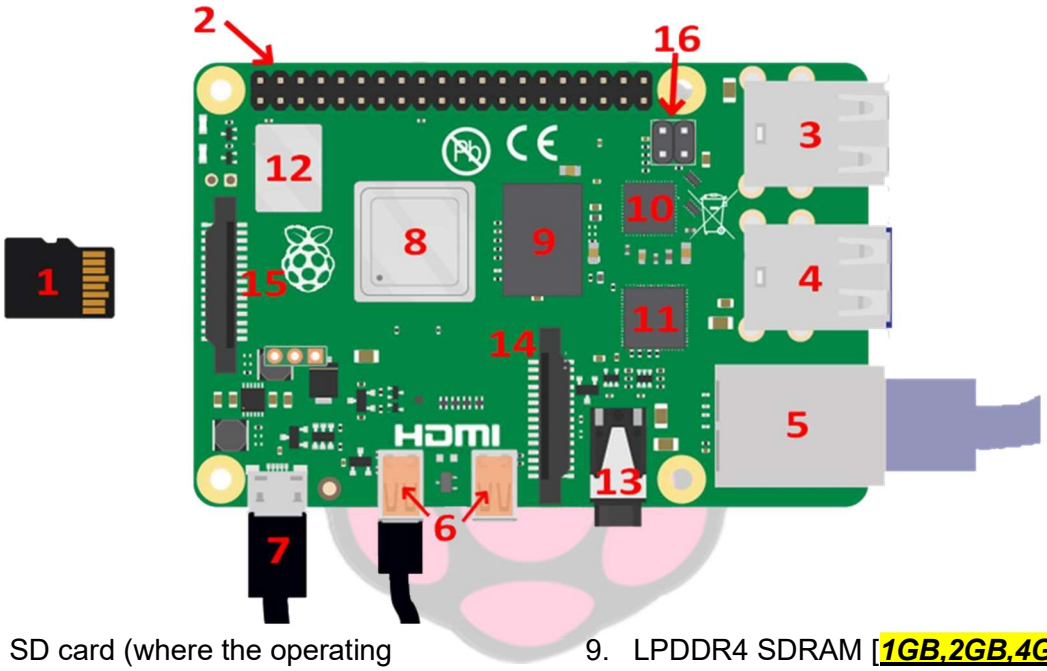
Justification

The main concept of the project is for anyone to be able to configure a small security server on their private network at home or business. This will be done with a device that is widely used in the tech community due to its price, utility, and size. This micro-computer is called “Raspberry Pi” in this case we will use the model 4.

I have chosen this topic because of the importance of computer security in today’s world. From our personal lives to our work lives, we are surrounded by computers, mobile phones and electronic devices that use the internet and are connected to the world wide web. Because of this, there are people who use it to steal, destroy, and ruin the lives and businesses of others. For these reasons I think this project can bring an easy and cheap way to add another security barrier against cybercrime.

Basics Concepts

- **Software Used:**
  - PuTTY – for Remote Connection
  - UltraVNC – For remote desktop control
  - PI-Hole – To create the Proxy/DNS server and filter ads and malicious pages.
  - OpenVPN – To create our VPN server.
- **Diagram of a Raspberry Pi 4**



1. SD card (where the operating system will be installed)
2. 40-Pin Header
3. 2x USB 2.0 Ports
4. 2x USB 3.0 Ports
5. Puerto Gigabit Ethernet
6. 2x Micro-HDMI Ports
   a. Single = 4K60fps
   b. dual = 4k30fps
7. 5V@3A USB-C Power Input
8. CPU 1.5GHz quad-core Cortex A72
9. LPDDR4 SDRAM [***1GB,2GB,4GB***]
10. Ethernet Driver
11. USB Driver
12. WIFI Dual Band (2.5GHz and 5GHz) and Bluetooth 5.0
13. Stereo output and composite video port
14. CSI Camera Port
15. DSI Display Port
16. PoE support with PoE HAT

Theoretical Framework

The basic theoretical framework of my project is security and computer networks. Two things that in today's world everyone should have a basic knowledge of due to the large amount of technology that surrounds us. The main objectives of my project are:

- Protect
  - Protect users with less knowledge of the dangers and risks of the Internet, and espionage. Blocking pages known to be malicious and dangerous, in addition to preventing data tracking of companies and corporations.
- Facilitate
  - Facilitate access to resources such as being able to connect to services and devices on your private network from anywhere thanks to the configured VPN. It also facilitates access to a secure network to be able to browse the internet with sensitive information without risk of data theft.
- Secure/Authenticate
  - Ensuring the security of all users and that all information, including passwords, is protected so as not to have the risk of data leakage.

Project Objectives

Protect local users will be the main objective of the project. This will be done with the following measures:

- A type of proxy/firewall to prevent and block the possibility that users with less computer knowledge are not susceptible to types of attack or malicious scams.
- Have a VPN that protects data transmission between computers and their destination in case there is an intrusion into the private network.
- And ensure that everything is well assembled to always have it available with little maintenance.

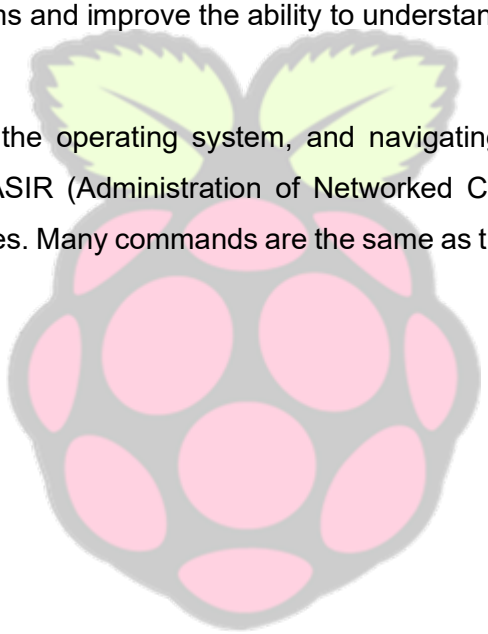
Research methodology

Most of the information used to implement this project comes from various official software forums and instructional pages and computer communities with tutorials and tips to help with these types of programs and operating systems. The main sites I used were:

- Reddit
- Pi-hole Forum
- Linus Tech Tips
- GitHub

Also, several videos when errors or problems arose were found on YouTube that help to visualize what may be causing problems and improve the ability to understand what the person is trying to explain.

When installing, configuring the operating system, and navigating the Linux distribution, that knowledge came from our ASIR (Administration of Networked Computer Systems) operating system and networking classes. Many commands are the same as the ones we used and learned in class.



Results and analysis

➢ Materials Needed:

- Raspberry Pi (Model 4 with 1gb RAM used in this manual)

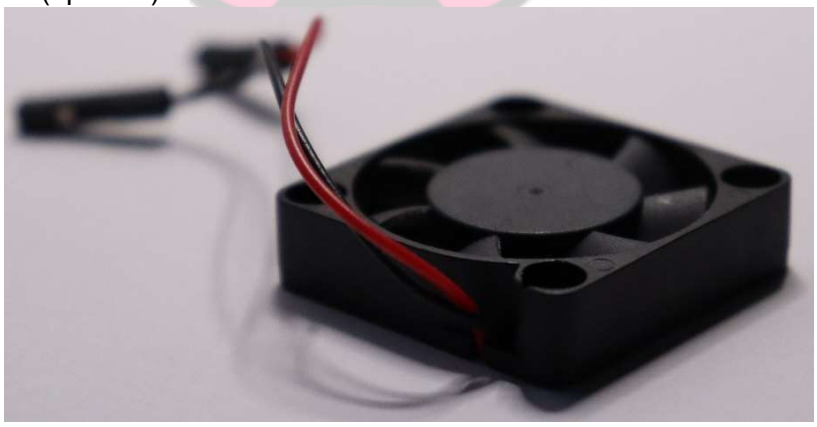
- Ethernet cable (Cat 5 or higher preferred)

- SD card of 8gb or more (for the Raspberry Pi 4 it has to be SD mini)

- HDMI cable or micro-HDMI for model 4

- A monitor
- Case for Raspberry Pi (optional)

- Fan or heatsink (optional)

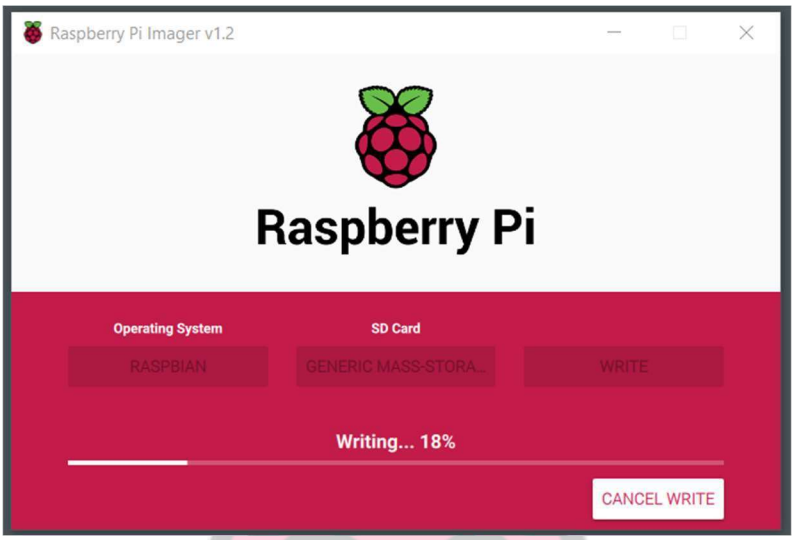


➢ Installing the Operating System:

1. The first thing to do is as with any new system, choose an operating system to use in our project. To make it simple, I chose to use Raspbian. This Linux distribution is lightweight and optimized for use on a Raspberry Pi's, although other lightweight distributions can be used.

This can be installed directly to the SD card or use a bootable Pen drive to install the system directly like any operating system.

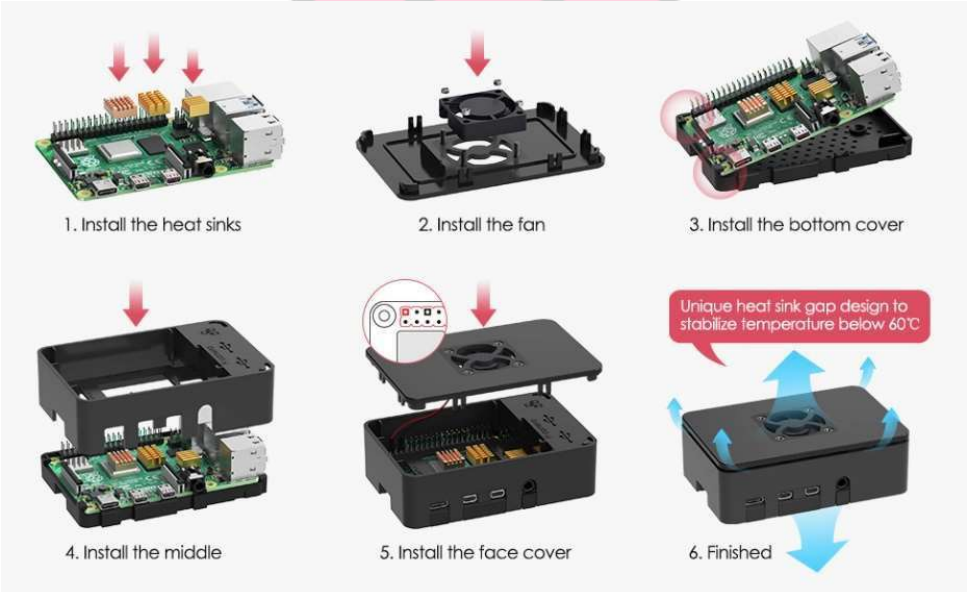
a. Installation can be done in several ways, with the software directly downloadable from: https://www.raspberrypi.org/downloads/



(Installing Raspberry Pi – Image 1)

b. Or with any software to create or install system images.

2. Then it is important that all components such as the heatsink, fan and Raspberry are all assembled correctly and that all necessary cables are connected.



(Raspberry Pi Case Assembly Example – Image 2)

(For initial installation you need to have a mouse and keyboard plugged directly into the Raspberry Pi.)

3. When we finish the Raspbian Installation we will have a graphical interface (or if we choose not to install it we will have a terminal). From here we can start installing the services that will make it possible to use the Pi remotely without having to have a keyboard, mouse, and monitor connected.

➢ Operating System Setup:

1. The first thing to install is the SSH or "Secure Shell Protocol" service. This is a service that allows you to securely connect network services over an unsecured network. In our case we are going to use it to use any computer on our network and connect to the Raspberry Pi. As we can see in image 3 it is very possible that OpenSSH (the software we are going to use for SSH) is already pre-installed. Also, with the systemctl status ssh command we can see if the service is active and running or not.

```
pi@raspberrypi:~ $ sudo apt-get install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-server is already the newest version (1:7.9p1-10+deb10u2).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@raspberrypi:~ $
```

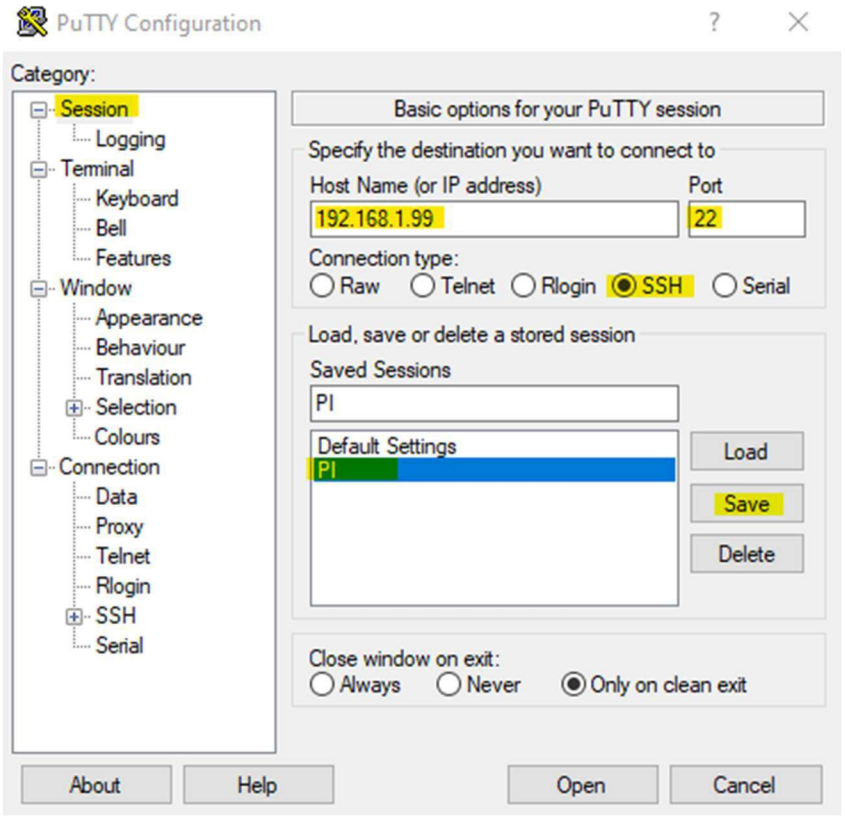
(SSH Server Installation – Image 3)

```
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-05-15 11:25:56 CEST; 1 weeks 2 days ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 559 (sshd)
    Tasks: 1 (limit: 1599)
   Memory: 4.9M
   CGroup: /system.slice/ssh.service
           └─559 /usr/sbin/sshd -D

May 19 19:57:47 carlos-raspberrypi sshd[26075]: pam_unix(sshd:session): session opened for user pi by (uid=0)
May 19 23:23:19 carlos-raspberrypi sshd[6741]: Connection closed by 192.168.1.50 port 53901 [preauth]
May 21 13:20:24 carlos-raspberrypi sshd[4827]: Accepted password for pi from 192.168.1.50 port 58155 ssh2
May 21 13:20:24 carlos-raspberrypi sshd[4827]: pam_unix(sshd:session): session opened for user pi by (uid=0)
May 21 15:40:45 carlos-raspberrypi sshd[15233]: Accepted password for pi from 192.168.1.50 port 59406 ssh2
May 21 15:40:45 carlos-raspberrypi sshd[15233]: pam_unix(sshd:session): session opened for user pi by (uid=0)
May 22 10:50:56 carlos-raspberrypi sshd[16436]: Accepted password for pi from 192.168.1.50 port 64044 ssh2
May 22 10:50:56 carlos-raspberrypi sshd[16436]: pam_unix(sshd:session): session opened for user pi by (uid=0)
May 24 13:07:27 carlos-raspberrypi sshd[22422]: Accepted password for pi from 192.168.1.50 port 51163 ssh2
May 24 13:07:27 carlos-raspberrypi sshd[22422]: pam_unix(sshd:session): session opened for user pi by (uid=0)
```

(SSH Server Status Check – Figure 4)

2. Once we have the service working, we can disconnect the keyboard, mouse and monitor and connect remotely. To do this from a Windows 10 computer, we're going to need software called PuTTY. This software uses the SSH protocol (and several others) to make remote terminal connections.
3. When we open PuTTY we have several options presented for different types of connection. We are only interested in the "SSH" option. As we can see in image 5, we use the IP of the Raspberry PI. The IP at this time will be dynamic until we make it static.



(PuTTY SSH Patch Panel – Figure 5)

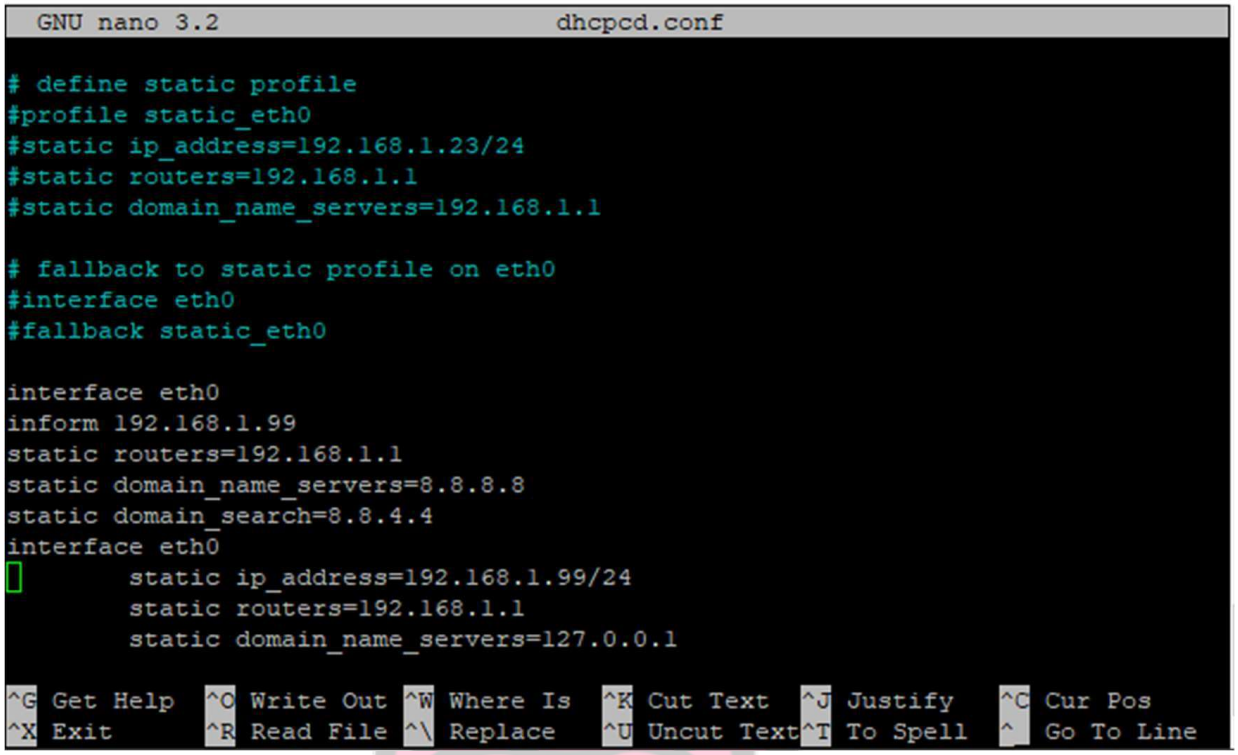
4. To log into the machine we use the pi username and the raspberry password we set during the OS setup.
5. As soon as we have access to the Raspberry Pi we can enter /etc using:

```
cd /etc
```

6. And then we can go inside the dhcpcd.conf file using:

```
nano dhcpcd.conf
```

7. Now in here we can use this configuration file to set a static IP for the "server"



(DHCP Configuration File – Image 6)

8. Here we can replace what you see above in white with what is provided below:

```
interface eth0
inform 192.168.99
static routers=192.168.1.1
static domain_name_servers=8.8.8.8
static domain_search=8.8.4.4
interface eth0
        static ip_address=192.168.1.99/24
        static routers=192.168.1.1
        static domain_name_servers=127.0.0.1
```

9. After doing that we restart the machine to ensure that the settings have been saved correctly.
10. To check that what we have done has worked, we use the command:

```
ifconfig
```

Now we can always connect with PuTTY to the same IP address.

10. Now we can change the default password to our own password. This is done with the command:

```
sudo raspi-config
```

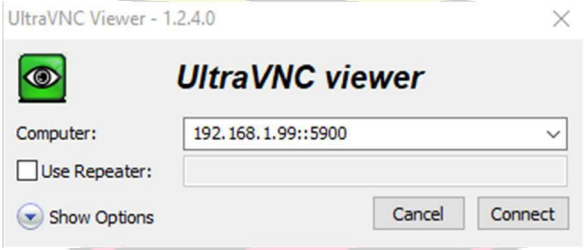
11. Now to access it graphically we have to install a VNC service. In our case we're going to use x11vnc. Using the following command will start the installation:

```
sudo apt-get install x11vnc
```

12. After installing the service in PuTTY we can use the following command to start the program:

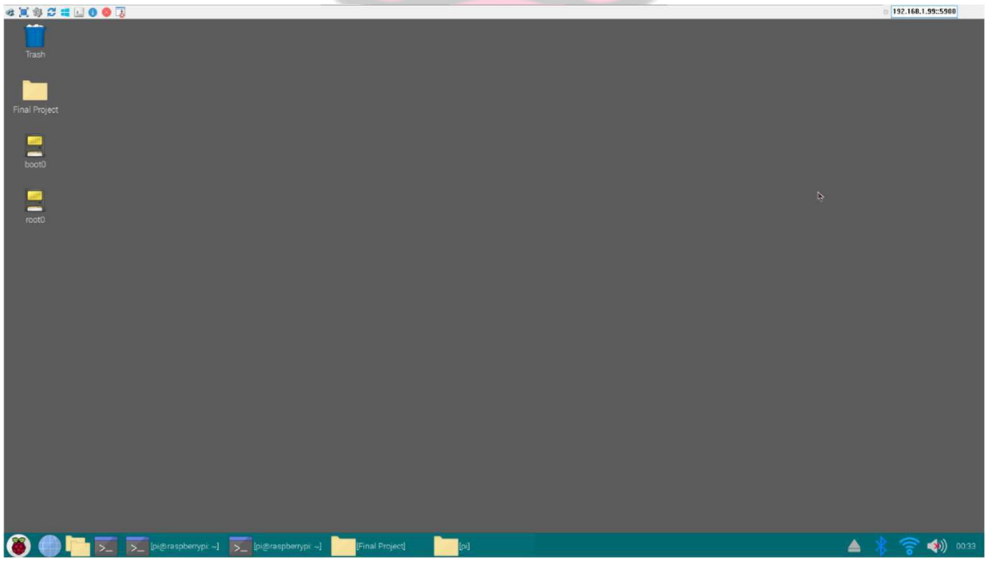
```
x11vnc
```

13. And with the UltraVNC software that we downloaded from the internet (https:// www.uvnc.com/) we can put the fixed IP and port 5900 like this to connect:



(UltraVNC Viewer VNC Patch Panel – Image 7)

14. And as we can see in image 8 we are connected with a GUI:



(Connecting over VNC to the Raspberry Pi server – Image 8)

➢ Installing the First Service (Pi-Hole):

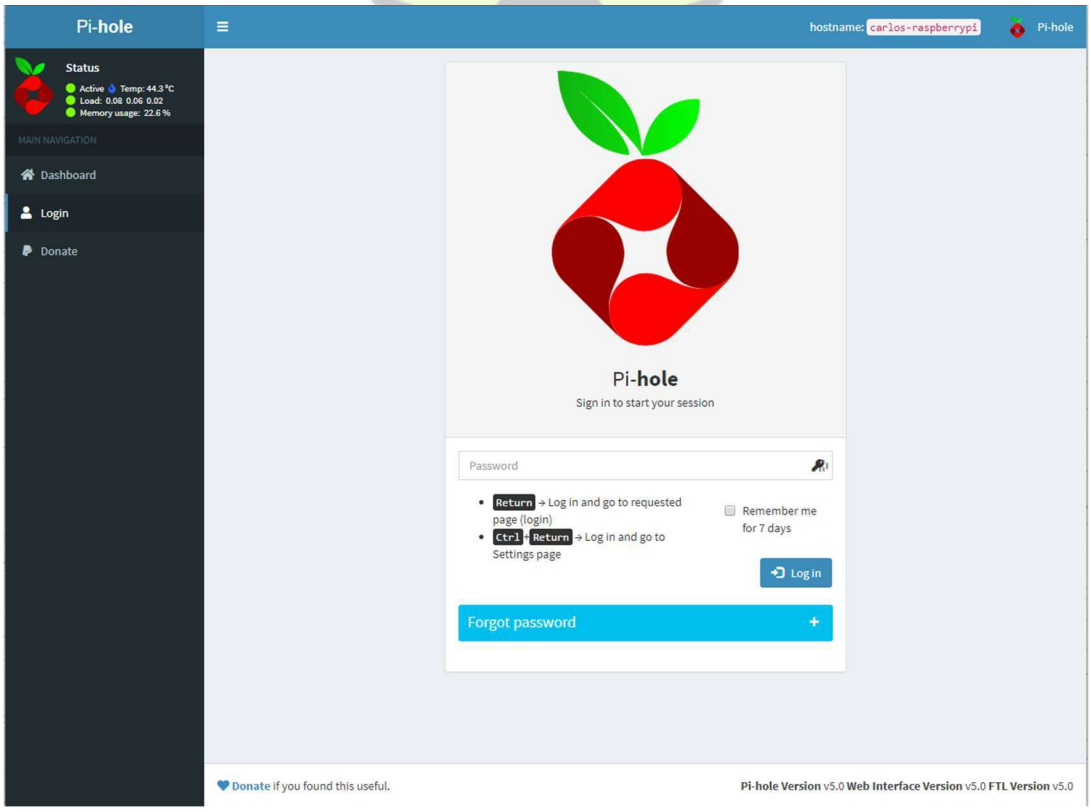
1. In theory, Pi-Hole is a very simple installation process. The first step is to use:

```
curl -sSL https://install.pi-hole.net | bash
```

2. As soon as we use this command the PI-Hole installation will start.
    a. First, we choose a DNS for the server to work.
    b. Next, we assign the IP and the gateway.
3. When the installation is finished, we restart the machine.
4. With the command we change the password for the administrator panel:

```
pihole -a -p
```

5. To access the admin panel we go to any search engine and use the URL http://192.168.x.y/admin but instead of the x, and y we put the IP that we assign it.
6. And after that you should see the following page:

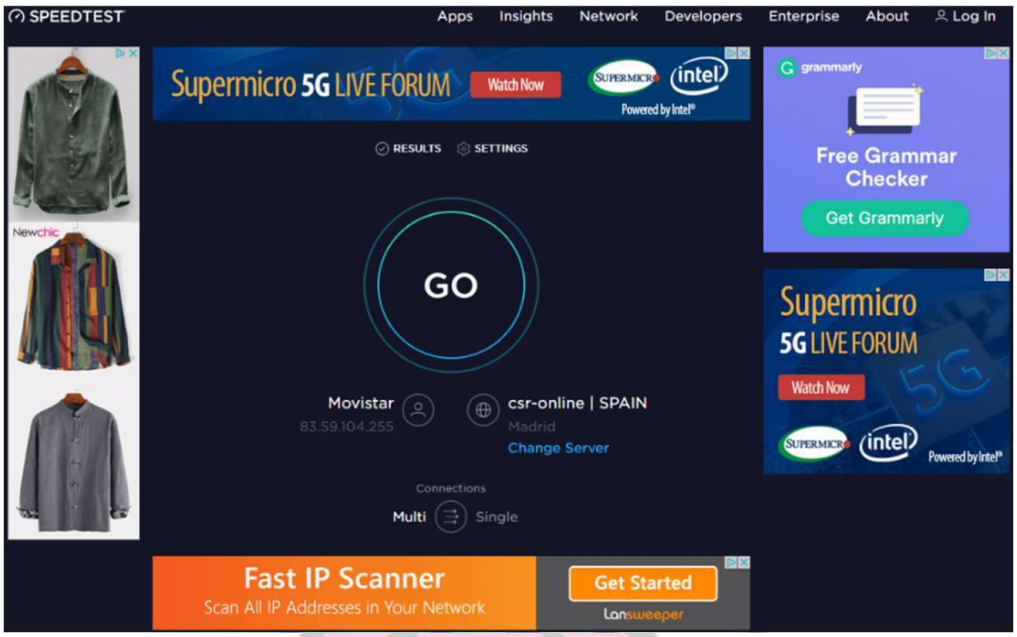


(Pi-Hole Administrative Panel – Image 9)

7. When we enter, we can enter all the pages. History, block and allow lists, tools, and more.

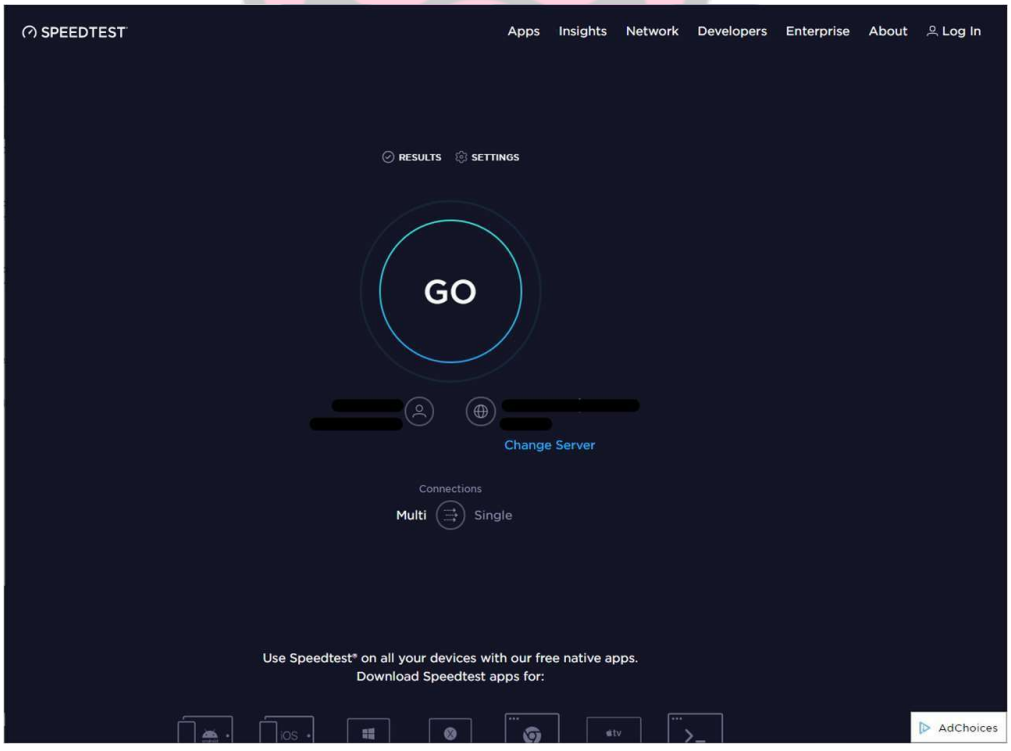
➢ Service Check (Pi-Hole)):

Here we have an example of what the Pi-Hole does. As we can see, before using the service, there are 5 different ads.



(Example page with advertising – Image 10)

As soon as we change our DNS server to our server we can see below how there is no longer any advertisements:



(Ad-free page thanks to Pi-Hole – Image 11)

And to check that it is really our server that has achieved this we can go to the admin panel at 192.168.1.99/admin and see that various ad pages have been blocked:

Time	Type	Domain	Client	Status	Reply	Action
2020-05-18 17:57:57	A	adserver-us.adtech.advertising.com	192.168.1.50	Blocked (gravity)	- (1.2ms)	Whitelist
2020-05-18 17:57:57	A	ib.adnxs.com	192.168.1.50	Blocked (gravity)	- (0.3ms)	Whitelist
2020-05-18 17:57:57	A	ookla-d.openx.net	192.168.1.50	Blocked (gravity)	- (0.6ms)	Whitelist
2020-05-18 17:57:57	A	hbopenbid.pubmatic.com	192.168.1.50	Blocked (gravity)	- (0.6ms)	Whitelist
2020-05-18 17:57:57	A	as-sec.casalemedia.com	192.168.1.50	Blocked (gravity)	- (3.6ms)	Whitelist
2020-05-18 17:57:56	A	secure-us.imrworldwide.com	192.168.1.50	Blocked (gravity)	- (0.6ms)	Whitelist
2020-05-18 17:57:56	A	gurgle.zdbb.net	192.168.1.50	Blocked (gravity)	- (0.3ms)	Whitelist
2020-05-18 17:57:56	A	www.google-analytics.com	192.168.1.50	Blocked (gravity)	- (0.2ms)	Whitelist
2020-05-18 17:57:56	A	www.google.es	192.168.1.50	OK (forwarded)	IP (31.0ms)	Blacklist
2020-05-18 17:57:56	A	analytics.google.com	192.168.1.50	Blocked (gravity)	- (0.2ms)	Whitelist
Time	Type	Domain	Client	Status	Reply	Action

(Blocked domains page – Image 12)

The most important things that we see blocked here are analytics.google.com or google-analytics.com. This is a Google service that collects information about the users it connects to. It collects information such as age, gender, interests, nationality, and much more.

We also see that several ad pages have been blocked, such as hboopenbid.pubmatic.com which may be a link for HBO ads, and adserver- us.adtech.advertising.com which from what it says looks like a server that manages ads to various web pages.

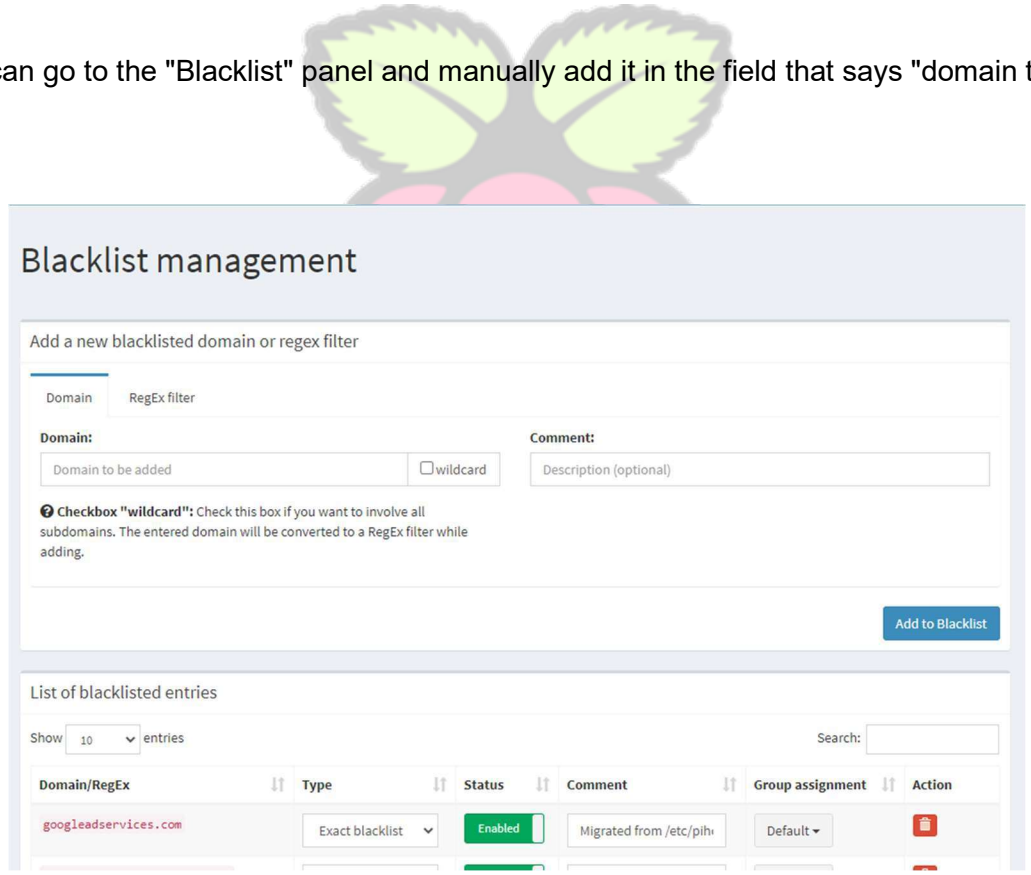
➢ Service Configuration (Pi-Hole):

Now, if for some reason we discover that Pi-hole is allowing a malicious page to pass, or we have an ad there are two ways to add that link to the list of blocked domains, either we go into the list and click "Blacklist" as we see next to www.google.com:

2020-05-18 17:57:56	A	www.google-analytics.com	192.168.1.50	Blocked (gravity)	- (0.2ms)	Whitelist
2020-05-18 17:57:56	A	www.google.es	192.168.1.50	OK (forwarded)	IP (31.0ms)	Blacklist

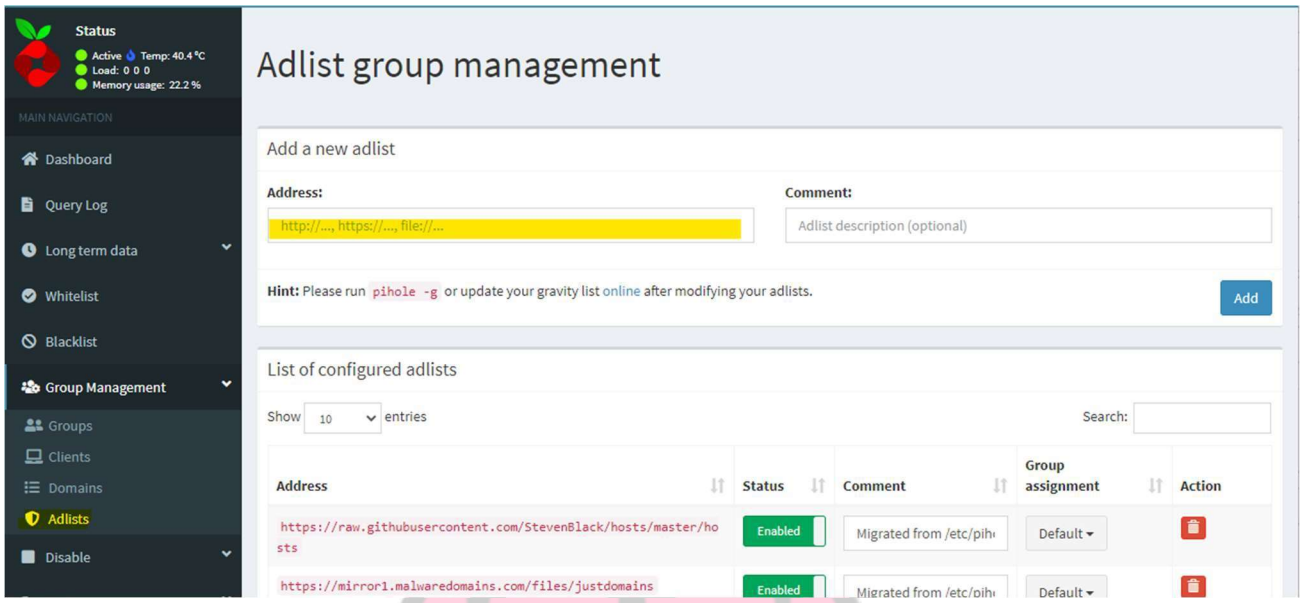
(Common Google Blocked Domains – Image 13)

Or, we can go to the "Blacklist" panel and manually add it in the field that says "domain to be added":



(Blacklist Page – Image 14)

Another useful tool is the "Adlist" section within "Group Management". Here we can enter lists generated by people online. For example, I can go to the Pi-Hole forum and find a list of all the pages related to “Sevilla FC” that a person has created, enter it on this page, and then automatically all those pages will be blocked:



(Adlists Page – Image 15)

➢ Installing the second service (OpenVPN):

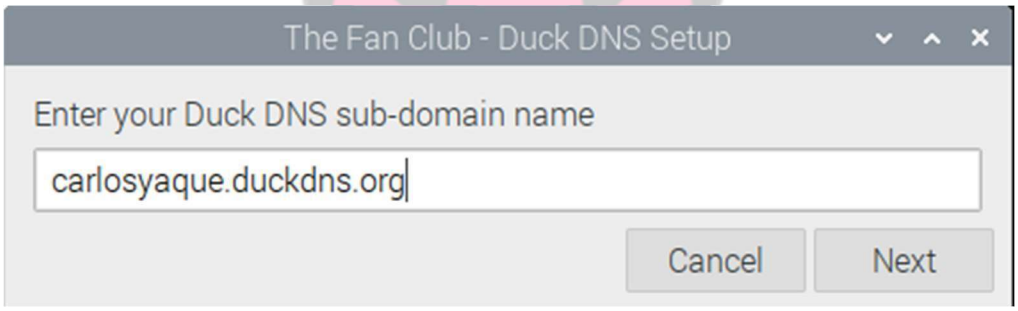
The next service we are going to install is a VPN server so that the client can connect securely from anywhere and have a secure connection.

1. The first thing we must do is set up a DNS server. Since most private home networks don't have a static public IP, we need to set up the DNS so that clients don't have problems connecting. For this we are going to use a free DNS called Duck DNS. If we go to https://www.duckdns.org/ and create an account, we can enter the domains section and create one in this case I have created one with my name and surname:
    a. carlosyaque.duckdns.org

2. After creating our domain, we must associate our public IP with that domain and always have the IP updated. This is done by installing software either on our router, or on the Pi. We do this with the following command in GUI mode:

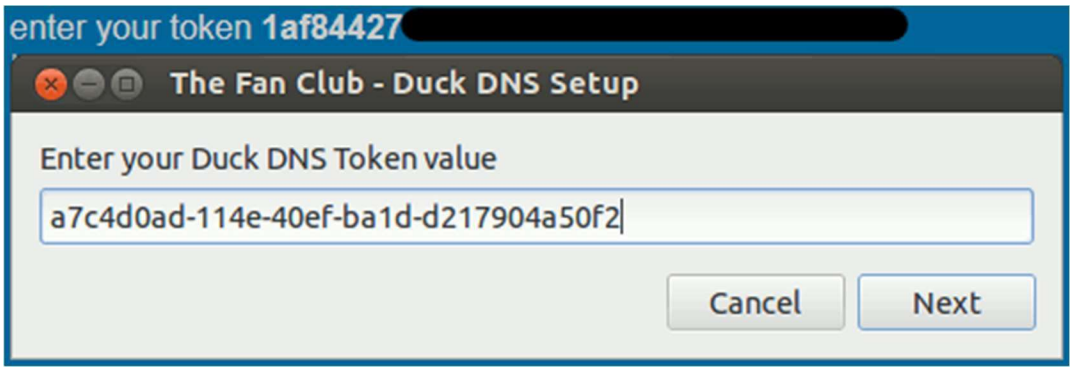
```
sudo apt-get install zenity cron curl
chmod +x duck-setup-gui.sh
./duck-setup-gui.sh
```

3. When we finish these commands a panel should appear with a message, here we have to enter our domain. Mine is carlosyaque.duckdns.org.



(DuckDNS Settings Panel – Image 16)

4. Then it will ask us for the "token" this is achieved by going into the DuckDNS website and entering the "Install" section selecting your domain and entering the linux GUI :

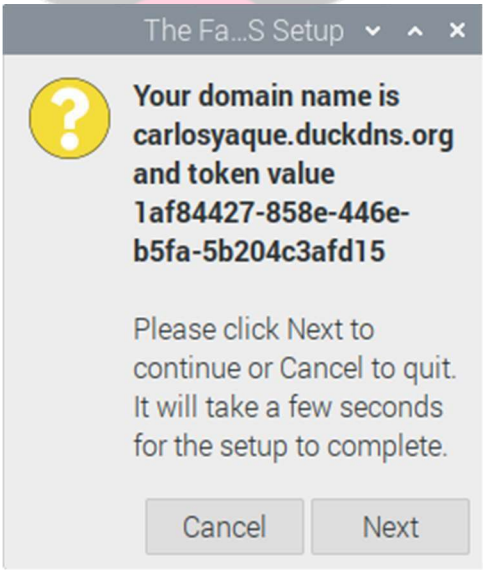


(DuckDNS Settings Panel – Image 16.2)

5. Here, as we see in image 16.3, my token starts with "1af84427". We copy the whole number and put it in the box that comes out in the installer:

(DuckDNS Settings Panel – Image 16.3)

6. Finally, to finish the DNS installation, you just have to press "next" and check that the information is correct:



(DuckDNS Settings Panel – Image 16.4)

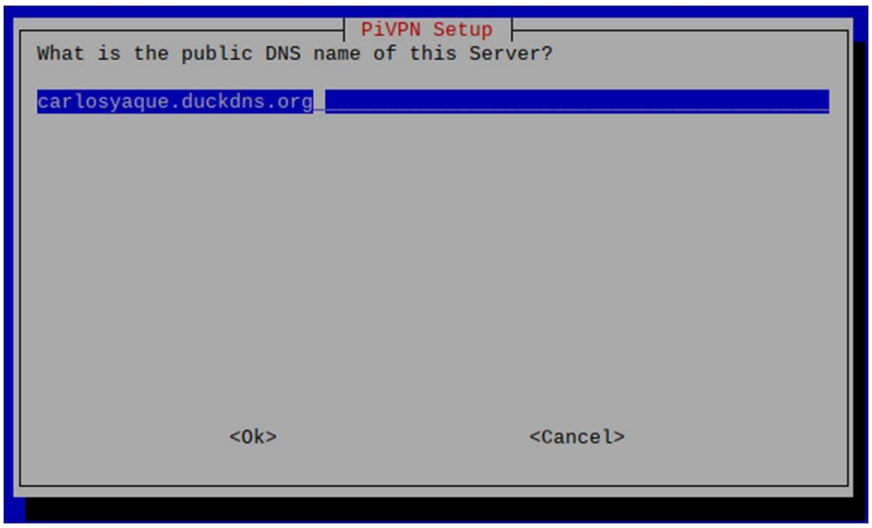
7. Now we can install the VPN. The first thing to do is to use the following command:

```
curl -L https://install.pivpn.io | bash
```

8. As soon as we put this command, a page will appear that will start the installation of the service as we see in image 17.1:

(Pi-VPN Installation – Image 17.1)

9. In this installer we will have to configure the following parameters:
   a) The IP of the server
   b) The VPN distribution we want to use (in this case we are going to use OpenVPN)
      i. Wireguard
      ii. OpenVPN
   c) The Port
      i. Which defaults for OpenVPN is 51820
   d) The next page detects that we have pi-hole installed and asks us if we want to use the Pi-hole proxy for the VPN.
   e) Next, the installation asks if we want clients to connect with our public IP, or a DNS. Let's select DNS since we have already created it in the beginning.



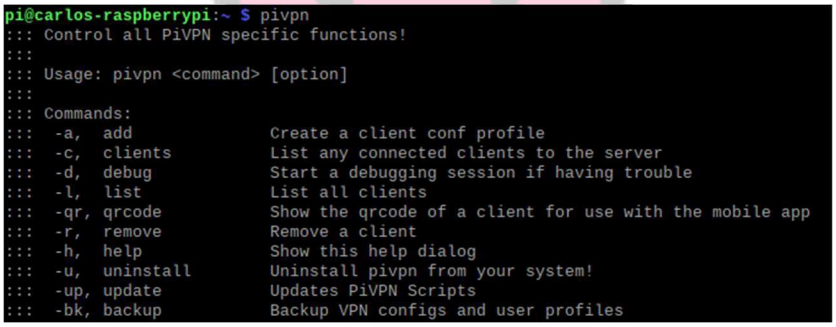
(Pi-VPN installation – Image 17.2)

f) After this, start generating the public and private keys for data encryption. This may take 5-10 minutes.

g) Finally, it will ask to restart the server.

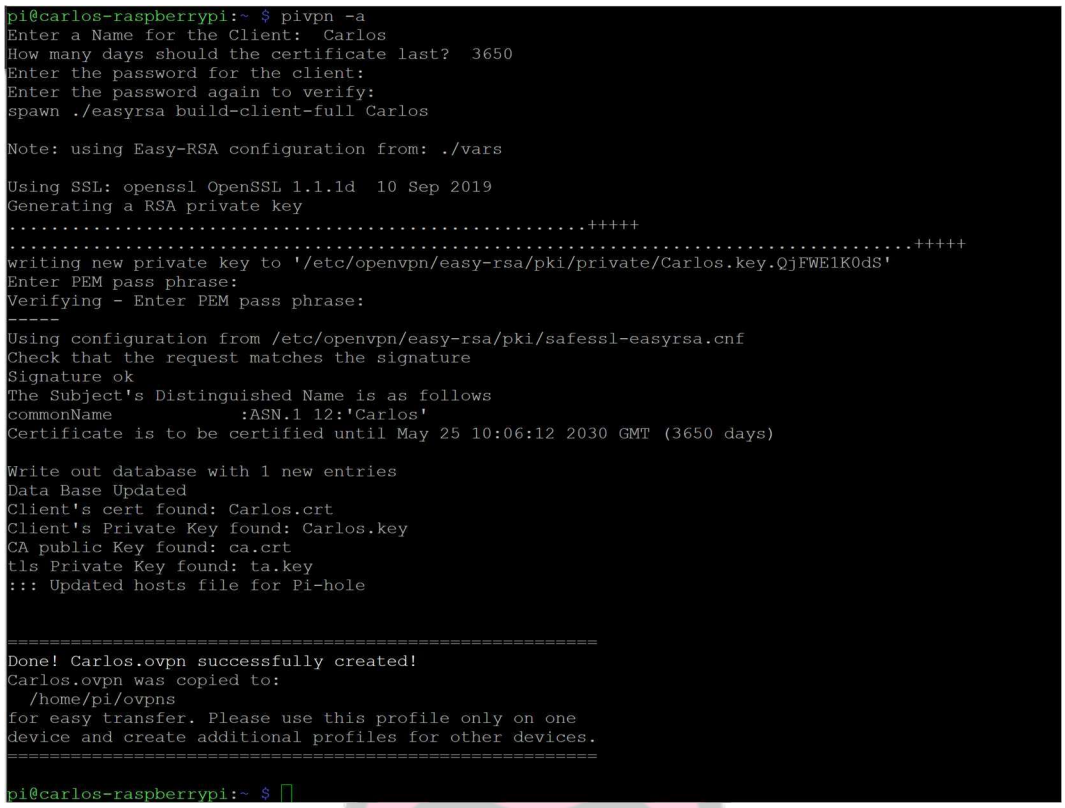
➢ Service Verification and Configuration (OpenVPN):

1. To check that the service has been installed correctly, what we can do is type the pivpn command and see if a list of all the different commands that we can use for this service comes up.



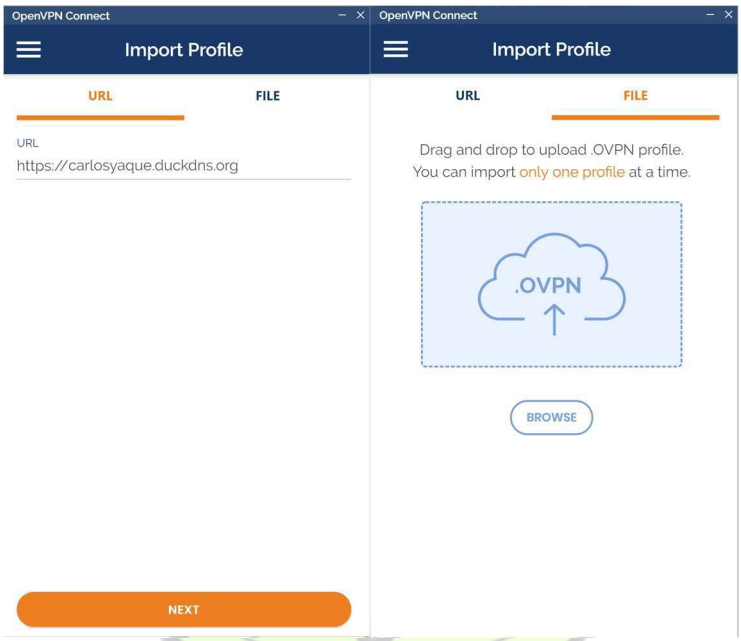
(Pi-VPN Commands – Image 18)

2. After checking that the service is booted, we can start configuring it. The first thing we should do is use the pivpn -a command to add a new client.



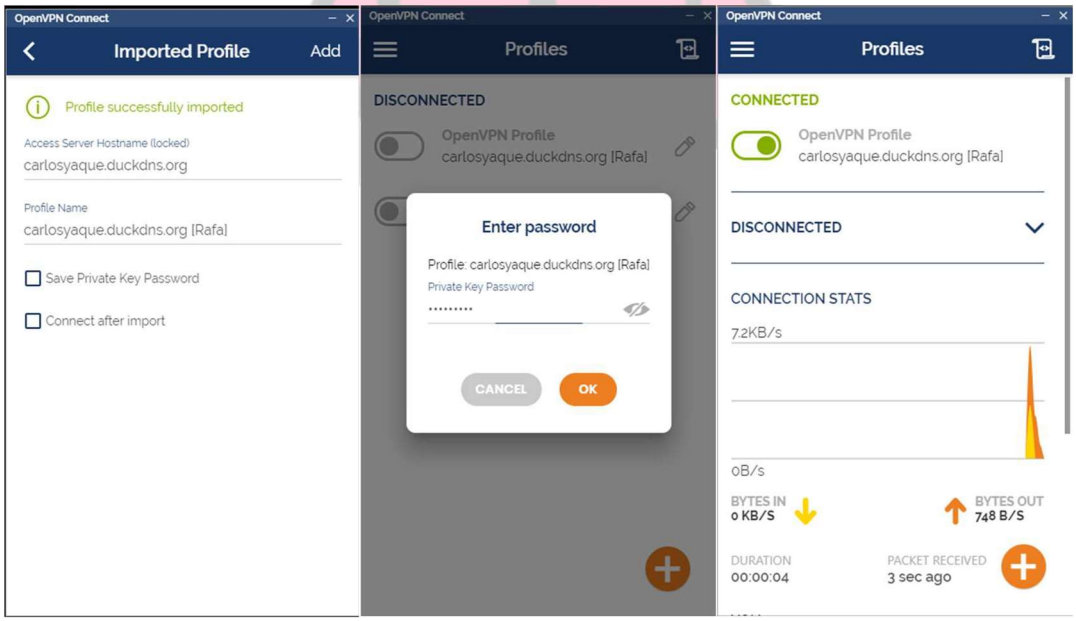
(User Creation for Pi-VPN – Image 19)

3. When we create a new user, it will ask us for a name, and a password for the client. In this case I have created one called Carlos with the password Carlos. When I finish creating the key and so on, it will create an .ovpn file that is what we are going to connect to the VPN with. (If we had chosen to install WireGuard, the other VPN, we would also have the option to add the VPN via QR code).
4. Now with a drive or some way to move the PI file to the computer we want to connect to, we go to the home/pi/ovpns file and copy the Carlos.ovpn file to our disk/pen drive.
5. Now on the other device we are going to: https://openvpn.net/client-connect-vpn-for-windows/ to download the Windows client. When we have it downloaded, we can connect to the DNS that we have mounted, or to the file that was created with the user:



(OpenVPN App – Image 20)

6. After using either method, you should be able to click, import, add, and enter the password to connect. As we see in the images 21 classmate Rafa, he was able to connect to my VPN with a user created for him from home:



(OpenVPN's connection to our VPN server – Image 21)

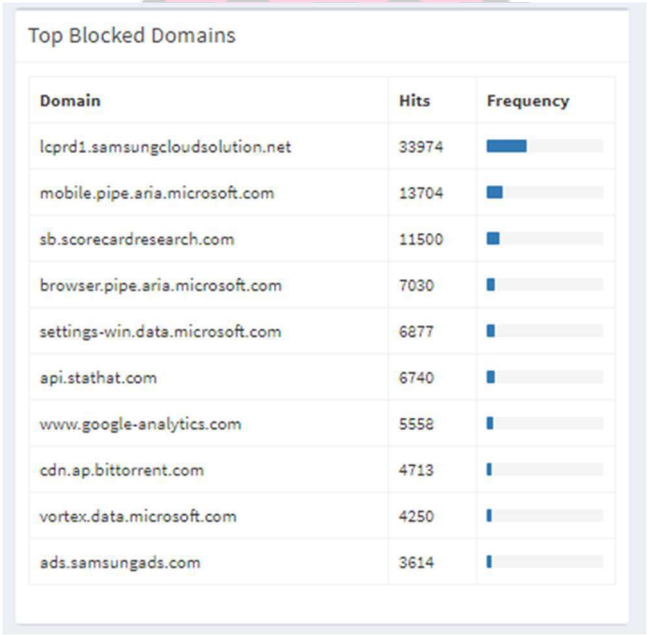
Conclusions

➢ Pi-Hole:

I initially installed and tested the new device on March 17, 2020. And on May 13, 2020 I reviewed the statistics to see how well it has worked:

(Pi-Hole Data – Image 22.1)

As we can see in the image above, in just 1 month and 26 days out of the 20,184 requests 4,496 were blocked by our Pi-hole. And as we can see, almost 25% of all traffic from just 3 devices on our network. That's a lot of traffic that on normal, unprotected networks is constantly coming and going. Things like ads, malware, spyware, phishing, and many more things. This is also thanks to the added lists that our "Blocklist" increased to 135,036 blocked domains.



Top Blocked Domains

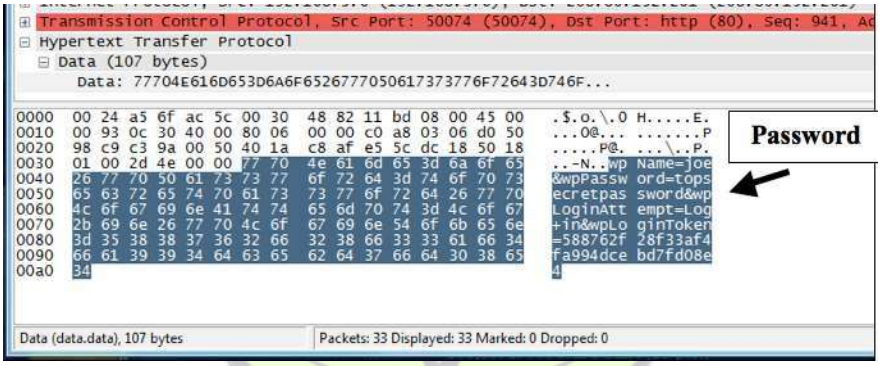
Domain	Hits	Frequency
lcprd1.samsungcloudsolution.net	33974	
mobile.pipe.aria.microsoft.com	13704	
sb.scorecardresearch.com	11500	
browser.pipe.aria.microsoft.com	7030	
settings-win.data.microsoft.com	6877	
api.stathat.com	6740	
www.google-analytics.com	5558	
cdn.ap.bittorrent.com	4713	
vortex.data.microsoft.com	4250	
ads.samsungads.com	3614	

(Pi-Hole Data – Image 22.2)

The most blocked domain is a Samsung one. The reason for this is the Samsung Smart TV connected to the Pi-Hole. In addition, other most blocked are those of Microsoft and Google that help against the sale of personal data and ads that are personalized by searches.

➢ PiVPN (OpenVPN):

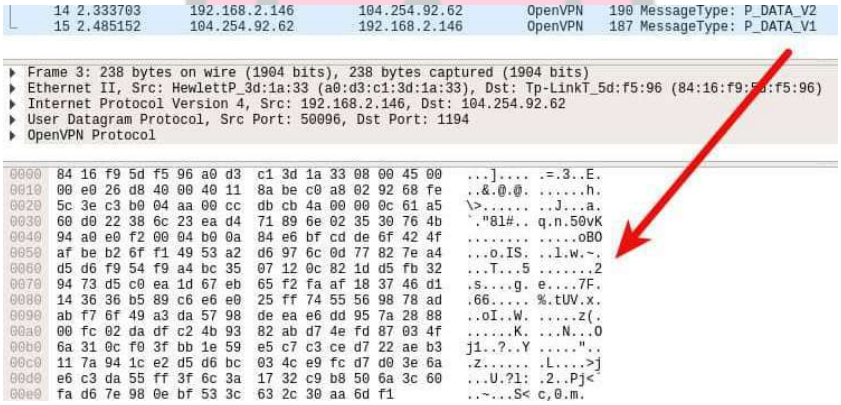
The VPN if properly assembled can help protect the data and privacy of any customer, as we see in image 23, using a software called Wireshark, a free software that anyone can download, we can "sniff" the network and see passwords, usernames, pins, bank details, and any other sensitive information that we do not want anyone to see:



(Unencrypted Package – Image 23)

Image source: (Project 3: Stealing Passwords with a Packet Sniffer, n.d.)

And as we can see in image 24, after activating the VPN the information becomes incomprehensible:



(Encrypted Package – Image 24)

Image source: (Watson, 2018)

OpenVPN uses "256-bit OpenSSL encryption" which means that each key generated will have 115,792,089,237,316,195,423,570,985,008,687,907,853,269,984,665,640,564,039,457,584,007 ,913,129,639,936 (78 digits) possible combinations. Even using the most powerful supercomputer in the world, it would take millions of years to "crack" that key. (Nohe, 2019)

References:

(n.d.). Retrieved from Github: https://github.com/

(n.d.). Retrieved from Reddit: reddit.com

(n.d.). Retrieved from LinusTechTips: linustechtips.com

(n.d.). Retrieved from Pi-Hole Forums: https://discourse.pi-hole.net/

Nohe, P. (2019, Mayo 2). *How strong is 256-bit Encryption?* Retrieved from Hashedout: https://www.thesslstore.com/blog/what-is-256-bit-encryption/

Project 3: Stealing Passwords with a Packet Sniffer. (n.d.). Retrieved from samsclass.info: https://samsclass.info/123/proj10/p3-sniff.htm

Watson, J. (2018, June 20). *What is packet sniffing and how can you avoid it?* Retrieved from comparitech: https://www.comparitech.com/blog/information-security/what-is-packet- sniffing/