Project torpi

Ver 0.01

Pynckels & Pynckels

January 12, 2022

Abstract:

This document describes the elements to take into account and the steps to take to create a reliable, confidential and cheap Internet environment that can be used to prevent thirds to eavesdrop on what you are communicating by means of Internet or on what you are doing on Internet.

There will be more technical parts to explain the exact issues at hand, but at the end of the document, there will be a list of steps to take in order to create the secure Internet environment that is described in this document yourself.

The authors of this document will be happy to help you out in case you have problems with the contents of this document, or with the creation of the described secure Internet environment.

Links to documents:

http://users.telenet.be/pynckels/.../BlueBrain.pdf
http://users.telenet.be/pynckels/.../BlueBrain.zip

Contents

List of Figures

List of Tables

Introduction

Not so long ago, I heard a director-general of the Belgian government reason that his organisation should be an exception on the GDPR. Meaning that his direction should be able to store data without listing up what data they keep, nor how they keep it safe. The only argument that was given as to why this should be the case was: ”workload”. His colleagues didn’t seem to have a problem with the idea nor with the argument.

It seemed strange at the moment, and in all honesty even stranger now that I have had time to think it through, that a director-general at the head of a governmental organisation is not willing to invest into applying the GDPR whereas private organisations don’t have much of a choice to do so. On top of that, it is worrying that a governmental organisation wants to store information without citizens being aware that their data is stored and used, and without citizens being able to count on the fact that there is enough transparency to permit specialized thirds to audit the security of the data storage environment.

Since this is hardly the only case of an organisation that, for all kinds of good or other reasons, wants to capture and store data on citizens, it only seems fair for citizens to have the possibility to prevent certain information to be obtained by their, or for that matter, any government or organisation. So, the authors of this document, have created an Internet environment that makes it harder for governmental or other (such as Internet providers) to gather information about the data that is send, the sites that are visited, etc. Note that nothing is impossible, and that there are no guarantees in life, but that accessing your data can be made very hard. And that is what this document is all about.

1 Section

1.1 Subsection

1.1.1 Subsubsection

2 Installation instructions

This chapter will guide you through the installation process of the different components of the secure Internet environment.

2.1 Raspberry TOR router

2.1.1 Prepare the SD card

The following steps will prepare an SD card for your Raspberry Pi. This can be done by means of a program (can be downloaded from https://www.raspberrypi.com/software/) that copies the necessary software on the SD card.

If you are using a Linux pc to prepare the SD card can, this can also be done on the command line by following the steps below. We presume that your SD card is device /dev/sda

An easy way to find the name of the SD card device is to go to the command line and to execute the command lsblk. When you get the results, connect the SD card and reexecute the command lsblk. Then, it's just a matter of finding the difference...

To prepare the SD card, execute the following commands on the command line:

```
sudo apt-get -y install buffer
sudo apt-get -y install gunzip
sudo apt-get -y install wget

wget -qO- https://downloads.raspberrypi.org/raspios_lite_armhf
/images/raspios_lite_armhf-2021-11-08/2021-10-30-raspios-
bullseye-armhf-lite.zip
| buffer
| gunzip -c -
| sudo dd of=/dev/sda bs=16M status=progress

sudo umount /dev/sda1
sudo umount /dev/sda2
```

To prevent misunderstandings: the name of the file will change with future versions of Raspberry OS. It's advisable to use the latest version that is available to prevent as many security issues as possible.

2.1.2 Prepare the Raspberry Pi

The time has come to put the SD card in the Raspberry Pi and to boot the device. From here on, we work on the device itself except when otherwise stated.

— DO NOT CONNECT THE DEVICE TO A NETWORK YET —

Power on the device. After a number of seconds, a command prompt will be visible, asking for a userid. The default userid is ***pi***. Then a password is asked for. The default password is ***raspberry***. Please note that the default keyboard is querty, meaning that, e.g. on an azerty keyboard, the password must be typed as ***rqspberry***.

Now we must enter the Raspberry configuration program:

```
sudo raspi-config
```

Setup the keyboard:

If you have a non qwerty keyboard you must alter the keyboard. Now choose the options:

```
Localisation options -> Keyboard
```

Choose the right keyboard layout and country setting. The keyboard should now have the correct layout. From now on, the password at logon can be typed exactly as it is.

Setup the device host name:

While in raspi-config choose:

```
System Options -> Hostname
```

and follow the program directives. It is advisable to choose a name that blends in in the environment where the device will be used. A strange name will get more attention, and that is exactly what you do not want when you want to stay under the radar.

Change user password:

While in raspi-config choose:

```
System Options -> Password
```

and follow the program directives. It is advisable to choose a strong password that is at least 20 tokens long and that can not be found in the usual password lists.

To change the password, you can also use the terminal command passwd and follow the instructions. Since we were already in the raspi-config program, it seemed appropriate and more user friendly to do it from within that program.

Enable remote access via SSH:

While in raspi-config, we will enable SSH to provide remote access to the device. In first instance, this access will go through the normal network. Later, we will provide access through and only through the wireless access point that we will install on the device.

Choose:

```
Interface Options -> SSH -> Yes
```

To close raspi-config choose:

```
Finish
```

To check if the SSH service will be running from now on, even after a reboot, enter the command:

```
sudo reboot
```

Wait for the reboot to finish (this can take 30 seconds, and the screen may go dark in between), log on with the default userid and with the newly chosen password and enter the command:

```
sudo systemctl status ssh
```

Check if somewhere in the result of the last statement, a green text "active (running)" is shown.

Change user name:

To make life of intruders more difficult, we will also change the name of the default user to something only you know (like the password).

Please do all of the below actions consecutive without powering the device off, or without logging off.

We suppose that the new name of the user you have chosen is "ducky".

Please chose an other name for safety sake. Preferably, your user name will be something that is not in a dictionary word list, and that is compliant with the regular expression ^[a-z][-a-z0-9]*$ Meaning that a user name must start with a..z, then can contain a..z and 0..9 multiple times.

Substitute "ducky" by the name you have chosen in the following commands.

```
cd ~
sudo mkdir              /home/ducky
sudo cp .bash_logout    /home/ducky
sudo cp .bashrc         /home/ducky
sudo cp .profile        /home/ducky
sudo chown -R pi:pi     /home/ducky
sudo nano /etc/passwd
```

Now find the line:

```
pi:x:1000:1000:,,,:/home/pi:/bin/bash
```

and change the line as follows:

```
pi:x:1000:1000:,,,:/home/ducky:/bin/bash
```

Save the file and quit the editor.

Enable root by changing the the root password:

```
sudo su
passwd
```

Choose a temporary password for root (for instance "ip"). This password can be short and simple since we will disable root immediately after renaming the default user name.

Exit from root and from pi:

```
exit
exit
```

Log in as root with the password "ip", and rename the default user:

```
usermod -l ducky pi
groupmod -n ducky pi
exit
```

Log in as ducky (or the user you choose) with the password correct password, disable root and clean up the pi directory:

```
sudo usermod --pass="*" root
sudo rm -fr /home/pi
sudo cat /etc/shadow
```

The last statement should show a list of users. Check that there is an asterix (*) after the root user. This is the proof that the root user is disabled again. The user ducky (substitute the user name you choose) should contain a lot of clutter after the user name. This is the encrypted password you have chosen for the default user.

Secure root access:

Now is the moment to enforce the fact that the root user can not log on. This is already the case by means of the asterix in the /etc/shadow file, but we can do better than that:

```
sudo nano /etc/passwd
```

Find a line that starts with root,such as:

```
root:x:0:0:root:/root:/bin/bash
```

and change the text after the last colon. In case of the example above, this would become

```
root:x:0:0:root:/root:/usr/bin/nologin
```

Save the file and quit the editor.

This way, even if one can log in as root, one does not fall into a shell which makes it difficult to continue as root.

Now, we need to force asking the password when the default user wants to execute a "super user do" (sudo) command. On the one side, this prevents making unintended errors, on the other side, it makes the lives of intruders so much more difficult since sudo scripting will be partially blocked.

Enter the command:

```
sudo rm /etc/soduers.d/010_pi-nopasswd
sudo nano /etc/sudoers
```

We are aware that in theory a per user sudo setup in /etc/sudoers.d is the way to go, but in this case the goal is not to open a door for a user, but to close the door for all users. Hence editing the sudoers file immediately.

Find the lines comparable to:

```
Defaults    env_reset
Defaults    mail_badpass
Defaults    secure_path="/usr/local/sbin:/usr/local/bin:..."
```

and change insert an extra line. In case of the example above, this would become

```
Defaults   env_reset
Defaults   mail_badpass
Defaults   timestamp_timeout=0
Defaults   secure_path="/usr/local/sbin:/usr/local/bin:..."
```

Save the file and quit the editor.

This makes that each user without NOPASSWD privileges will have to give his password for each sudo command that he wants to execute. Take that, sudo scripters...

One could opt to drop the secure paths, but that would be overkill. However, if you want to go for top security, it is an option. In that case, almost all system commands will need a password and a sudo to execute. Nota that this will endanger a lot of scripts however.

2.1.3 Connect to the device with SSH

We have changed the

2.1.4 Connect to a local network

Now is the moment to connect the device to a network with an ethernet cable.The device is still not really protected against intruders, but since we are connecting on a private network, and due to the fact that root logon is blocked and that the password of the user id pi is long, we can take the risc.

Note however that the device should not be left powered on if you are not continuing the setup below. It is advisable in that case to:

```
sudo shutdown now
```

and to power off the device.

2.1.5 Connect to the device with SSH

The last thing we need to do to facilitate

2.1.6 Update software

We will now update/upgrade the entire system:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove --purge
sudo reboot
```

After rebooting, the default userid/password are still valid. Now we can assume that the latest patches are applied and that the system is up to date.

The necessary packages can now be installed:

```
sudo apt-get install hostapd
sudo apt-get install udhcpd
```

2.1.7 Configure DHCPD

Conclusion

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.