Journal

300M Wireless N Router

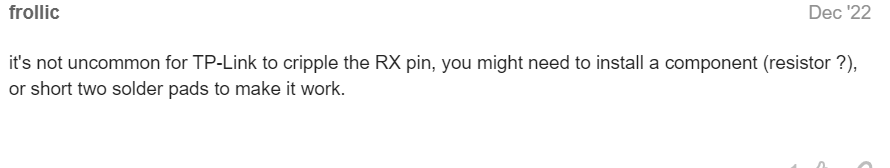
Model No. TL-WR845N

31.10.2023

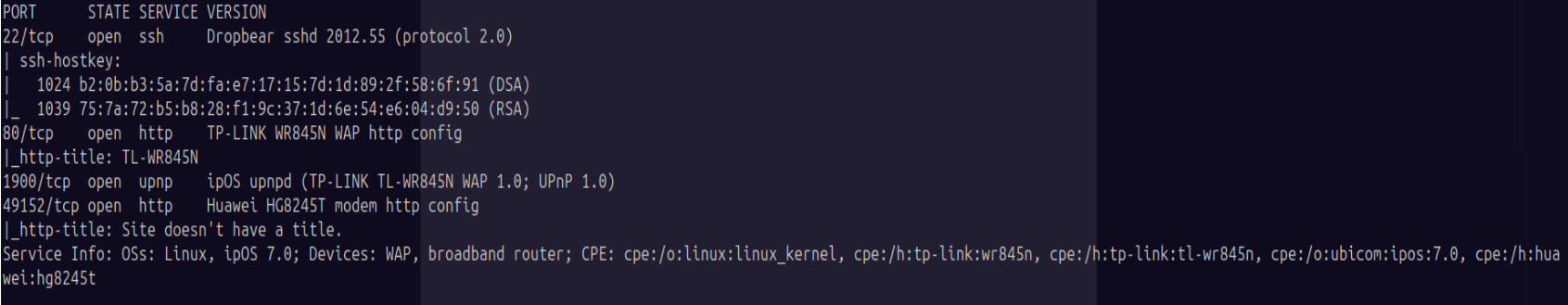
Firstly, I opened up the router, not to lose time with the web interface I wanted to get the filesystem as fast as possible to have a larger surface to explore.

Opening the router revealed a PCB of course. After a bit of looking around I found what I was looking for, the UART port. Usually, this UART port lets you connect to the U-Boot shell or gain a shell straight as a user (admin) on the device.

It seemed easy as first, but after configuring my Bus Pirate with the baud rate of 115200 and default configuration for the rest of the options, data started coming so router’s TX was working but I could not send any data, RX was somehow crippled ☹. After reading some posts online it seemed that especially TP Link likes to cripple serial interfaces on its routers.

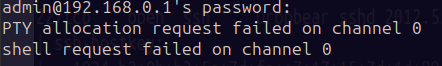


Ok so taking a break on hardware stuff would seem a great idea, a breadth first approach is better in initial information gathering and only after something looks promising a depth first approach is suitable. Let’s then perform a port scan on the router.



Of course first thing that I have tried was to SSH as root:root. Didn’t work.

After searching a bit on google for default user and password found that it is admin:admin. Tried and….

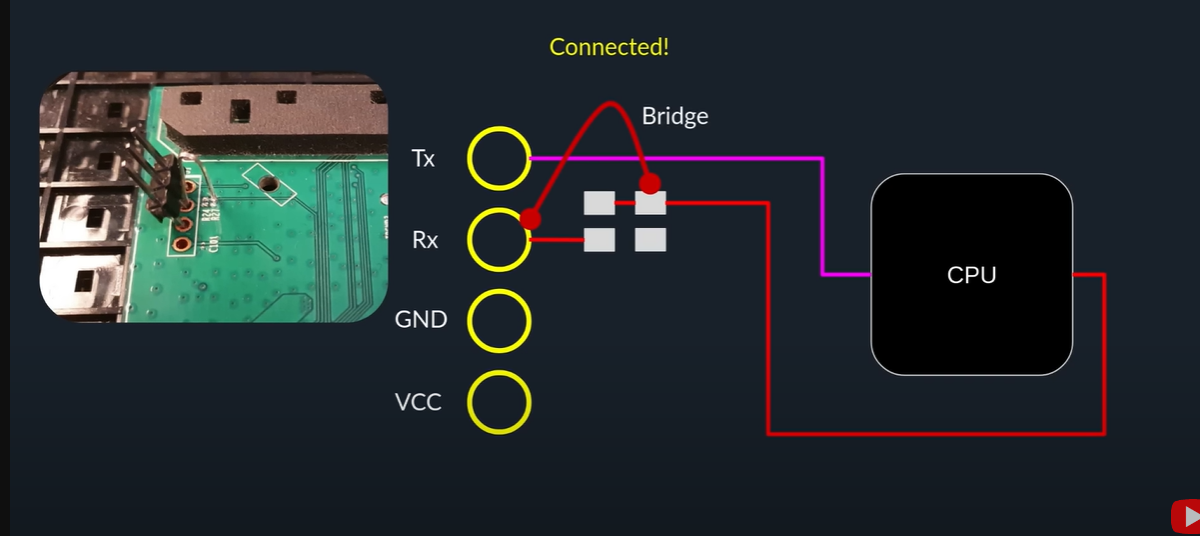
  
Did not succeeded to make it work, putty on windows returns the same so I guess it is something to do with a configuration.

01.11.2023

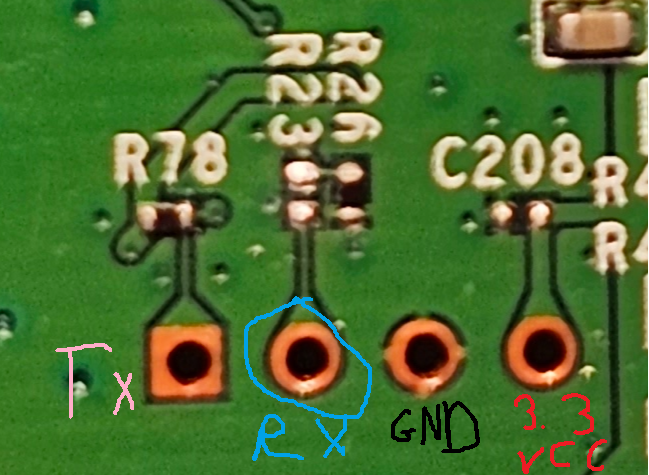
Decided this to be a hardware day, wanted to get the UART working. Found a nice youtube video with someone having the same problem but different router.

<https://www.youtube.com/watch?v=01mw0oTHwxg&ab_channel=FlashbackTeam> from 14:40, he has too a disabled RX pin.

This was his solution

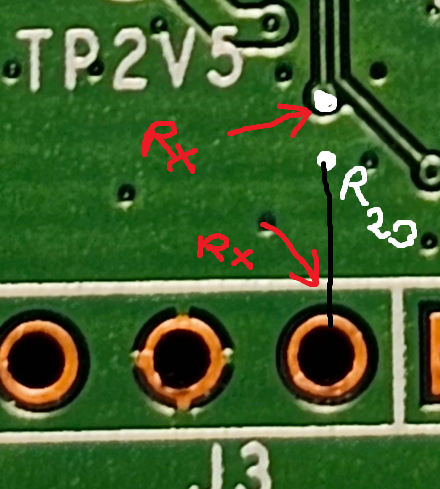


At first glance I have something that looks the same



I also have two Rxx ports and checking with a multimeter reveals that RX is connected to the lower part of R23 but it is not connected to the upper part of R23, which when traced with the eyesight seems to go straight to the micro controller.  
 At this point there are a lot of options, either connect R23 pins, or if I see correctly upper R23 is connected with upper R26 and maybe lower R23 should be connected with lower R26 so in this way the current will pass R26 resistor. But that is a speculation.

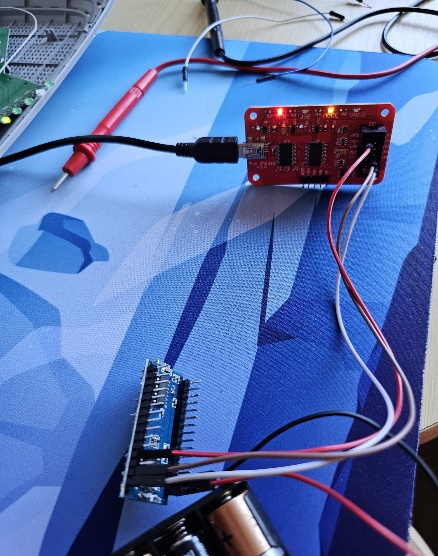
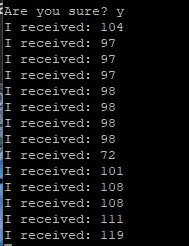
Looking at the other side and measuring with a piece of metal reveals that the connection going to the microcontroller would fit upper part of R23.



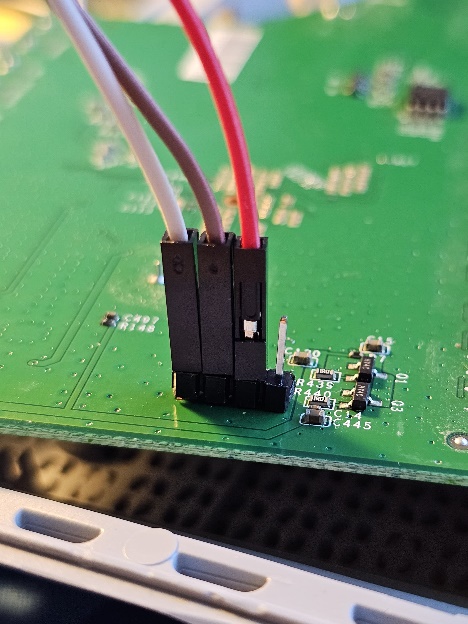
It should be easy right? Just hold my TX wire to that connection on the PCB, well I tried and it did not work I could not see my characters echoed on the screen. Tomorrow is another day!

02.11.2023

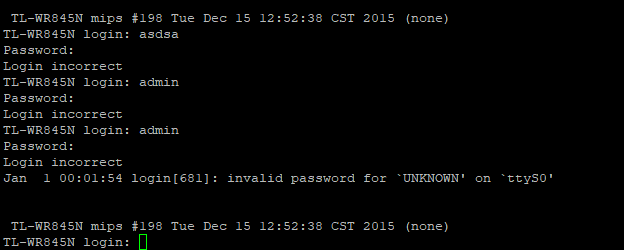
Started to get paranoic that my Bus Pirate does not work so I tested it with Arduino and I have seen that it works fine. A simple echo program.



Firstly, I soldered pins for a better stability during the connection.

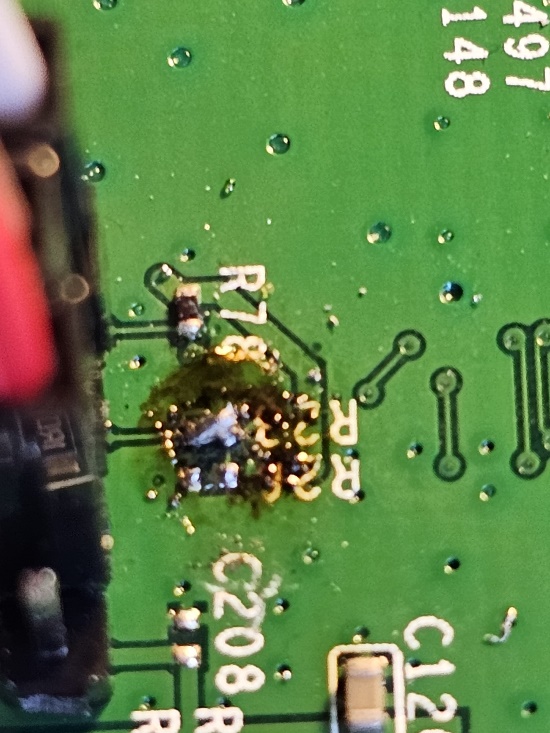


After that I tried holding a wire to the upper part of R23 for input and after a lot of reboots and tries, I could see my output being displayed 😊.



Unfortunately, no combination of trivial usernames or password was correct.

The input was unstable because I needed to fix a wire with my hand so I decided so solder R23 ports together resulting in a stable output. I am sure there was supposed to be a resistor but I did not want to try random values. 0 Ohm resistor it is.

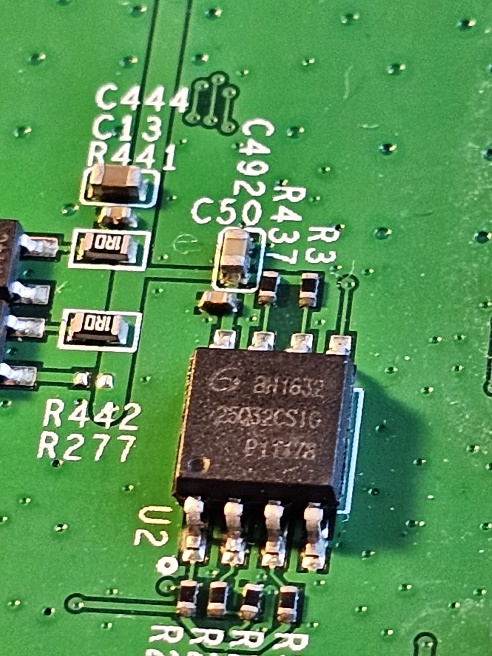


I know it does not look very professional but the contact patches are really small.

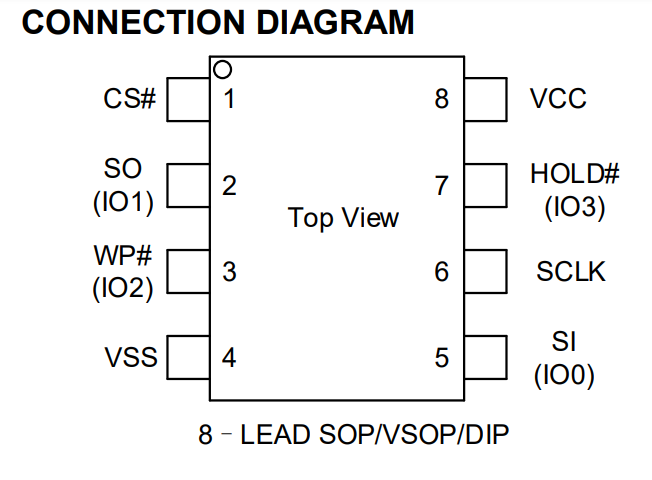
Now I wanted to perform a naive jailbreaking by letting the bootloader load from the flash but do not let the OS run so I can change the boot arguments.

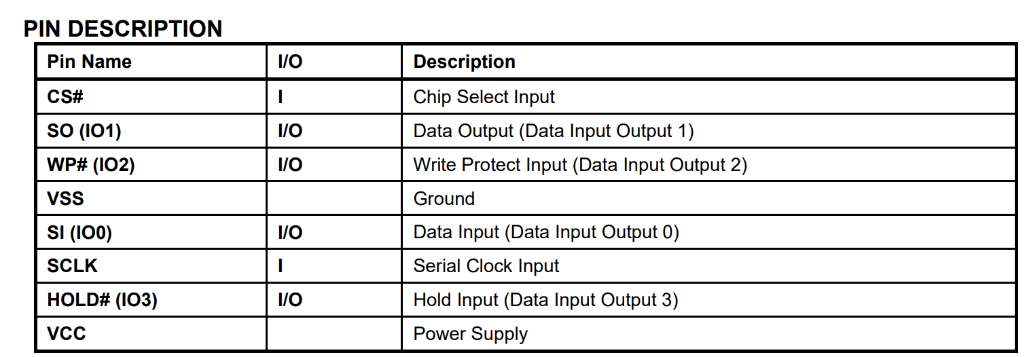
Especially I was interested in the **init** argument.

Would be nice to change it from */sbin/init* to */bin/sh* in order to bypass the login.

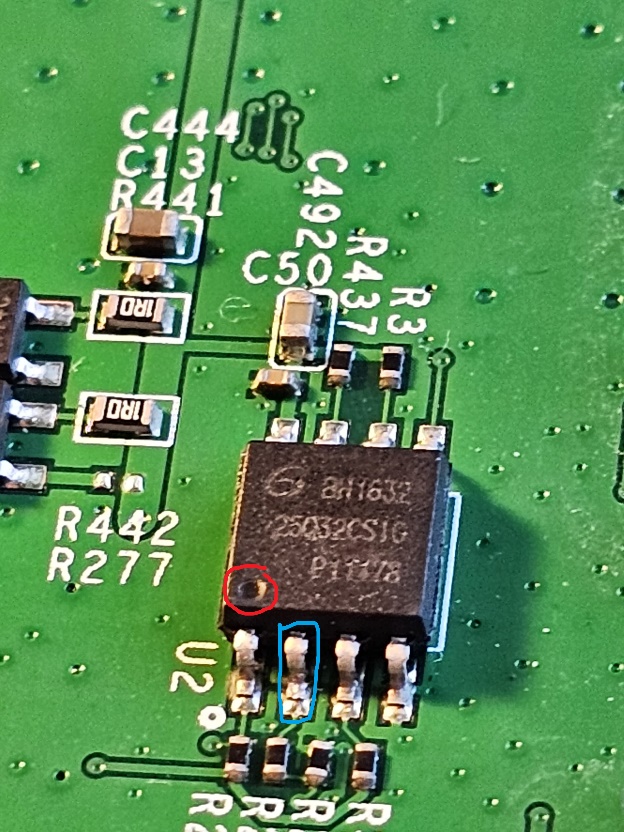
Firstly, I needed to find the flash memory. It is easy to find since usually it is a little squared circuit.

After searching on the internet the code in the second row ***25Q32CSIG*** we could easily found its datasheet: <https://eu.mouser.com/datasheet/2/870/gd25q32c_v4.1_20200319-1668129.pdf>





We are interested in the SO port, to redirect the data output to GND.

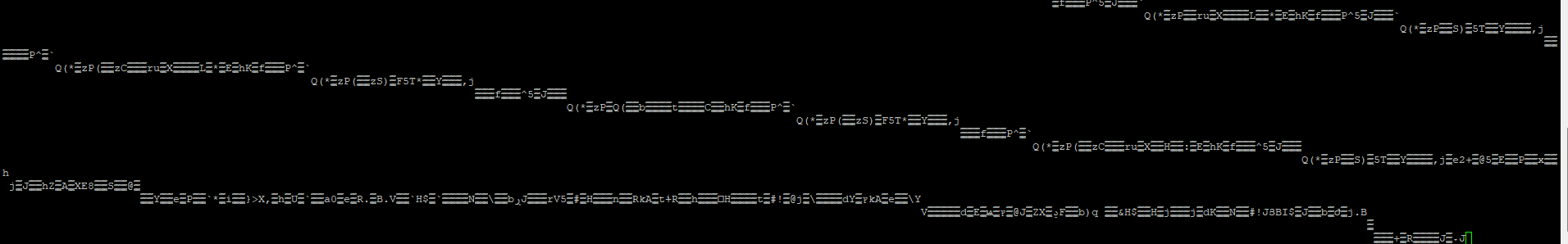


Red circle shows the dot on the chip so we can follow the schematic, and blue circle shows the SO (Data Out) pin.

Now we have to time when we make the SO -> GND connection in order to interrupt the normal boot process and be able to modify those boot arguments.

After a lot of tries so far, I ended up in 2 situations:

* To fast



Ended up in this region, I do not know what it is here, I suppose it is the bootloader.

* To late



I only suppose it is to late, because I cannot input anything from the keyboard.

06.11.2023

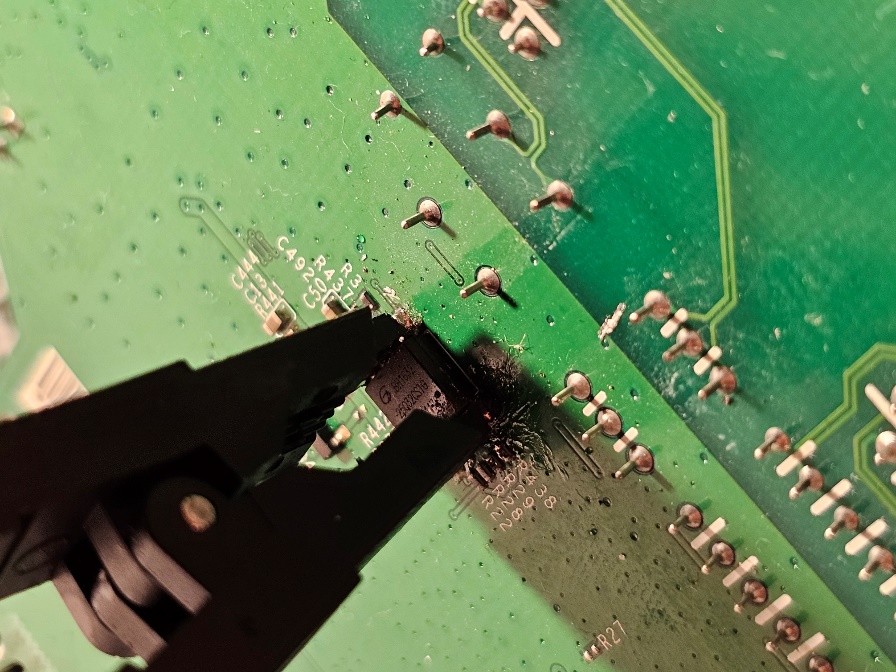
Today I tried to dump the contents of the flash chip. I have tried using the Bus-Pirate with the **Linux** command **flashrom** which supports bus pirate as the intermediate device. Tried a lot of firmware versions of the Bus-Pirate firmware but none work.

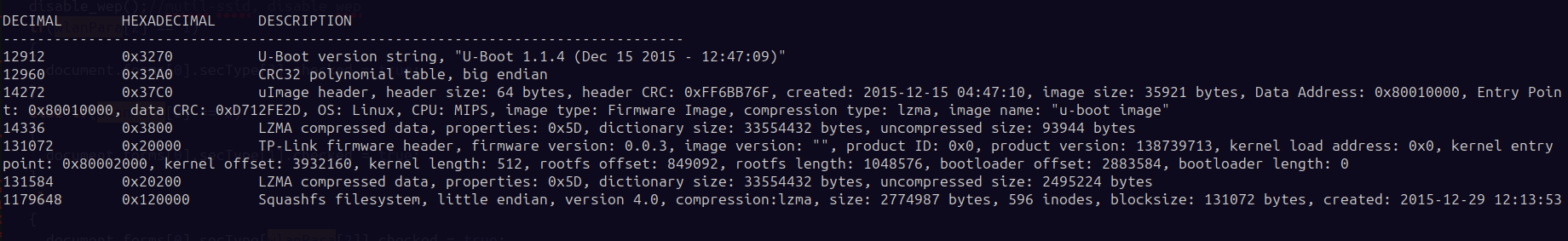
I also tried to read the contents by using the SPI from my Raspberry Pi 3B+ but it did not work again. In both cases I get no chip found so the **flashrom** command does not recognize my chip from its list but it is in there.

Long day 8h+ of trying and soldering with no result. Next time will try to read the flash from another router.

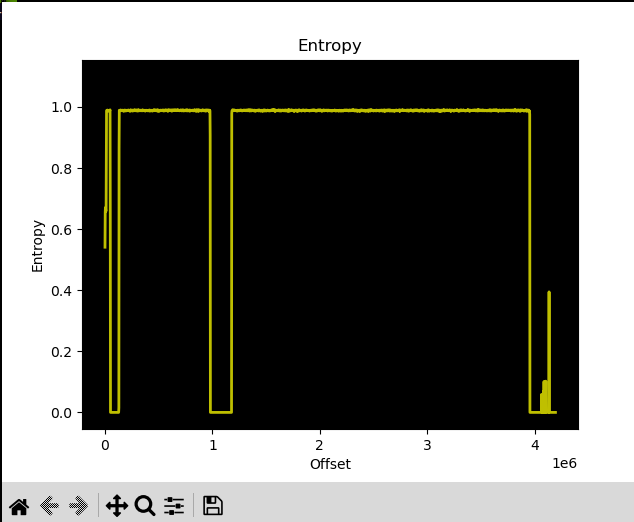
25.11.2023

After a long break, my **CH340** programmer arrived and I was able to read the flash on the router using **AsProgrammer V2.1.0.13**.



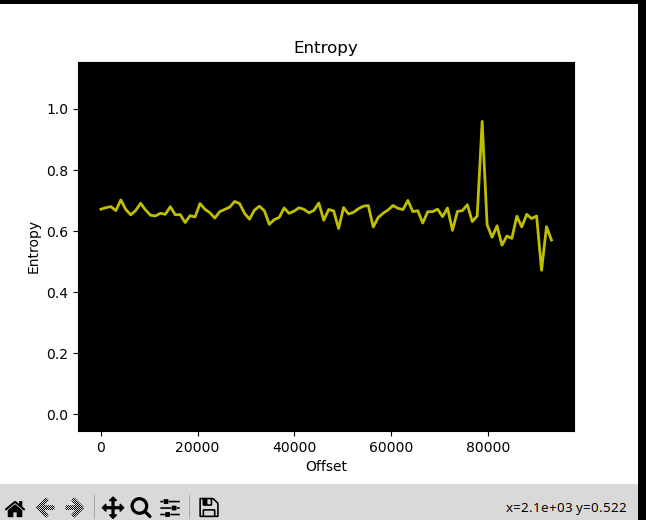
Quick **binwalk** command reveals this:  


We can see that most of it is compressed.

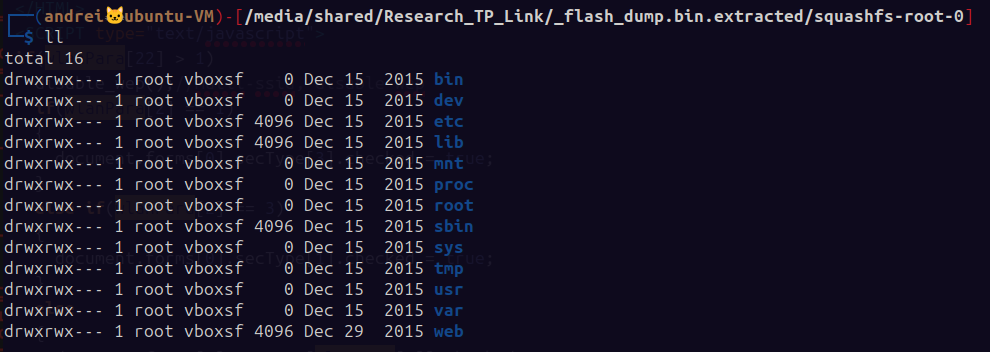


In the rest of the files not much information is useful, either U-Boot, certificates or kernel.

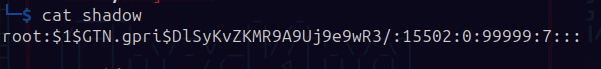
Decompressed lzma files and after running `binwalk -E` again on them it appears their entropy is somewhat bellow 0.7 which is the threshold for compressed/encrypted data.



In the squashfs a valid Linux file system is present.

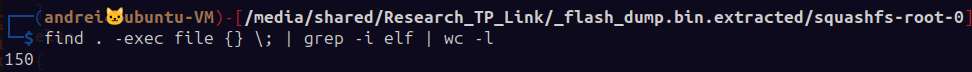


In /etc of the squashfs is the **shadow** file.



While trying to bruteforce the hash with **rockyou.txt** as wordlist. I just googled the hash online and found a lot of results, saying that the root password is **sohoadmin**. Rushed to login as root via ssh with this password but did not work ☹.

After running a find command to look for ELF binaries we see that there are 150, all of them are either default networking binaries or libraries and one other is busybox.



I also have another router that I was able to extract its firmware, if I am stuck here maybe I will give that one a try.