Tu Lam

CS 373 (Defense Against the Dark Arts)

Dr. Bram Lewis

October 27th, 2021

Lab #3 (Software Defined Radio)

This time in lab no.3, we are looking at a software called GQRX to look at couple of the file that the class provided for us. Through this, we are looking at the frequency, hearing, and decipher the file to find out what kind of content each file hold. From this, we can see how to detect the frequency and be able to crack the file using the GQRX program through VMWare.

At the start, I extracted the three files that was part of this lab and label them as number 1, 2, and 3.iq to make easier to distinguish them when doing the lab. Below will be a picture on what I rename them. Then I use the code "***sudo apt update***" to update the Kali VM to get all the latest thing install before install the program ***GQRX*** on it. Then I ran the code "***sudo apt install gqrx-sdr***" to install the ***GQRX*** program (*Figure #2* the code I enter in to install GQRX).

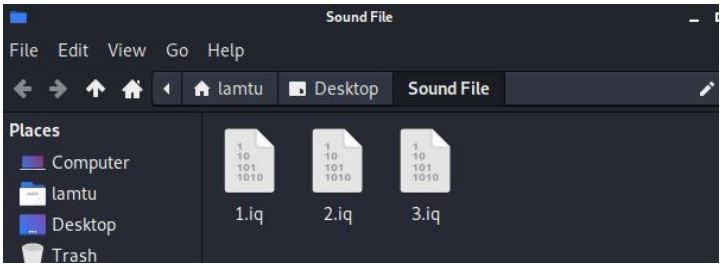

Figure #1: Showing the renaming of each .iq file from the lab

```
lamtu@kali ~/Desktop % sudo apt install gqrx-sdr
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
```

Figure #2: The code that was used to install GQRX

Then I was stuck in the process of installing ***GQRX*** but ran into trouble when it comes to installing it. Finally, after messing it around, I was able to get GQRX install and run it through the command line by using "***gqrx***". After that, the GQRX popup a window, which is shown in the figure below.

Figure #3: The screen that popup after launching GQRX

From there, on the lab canvas page, there is a video provided by the course to show us how to use GQRX. Using Dr. Kevin McGrath, we will use the frequency and setting output from the file that was given to us. From the ***first iq*** file, we find the frequency to be 50.090 MHz and 2Msps for setting up the option under GQRX. Below is the figure showing me putting in the input as for the setting, and in the setting, we must set the path to find iq file and by doing that, I found my path to be ***/home/lamtu/Desktop/Sounds/f1.iq/***. In the video by McGrath, he mentions that the sample rate must be more than twice the maximum frequency in order to identify the signal uniquely. So, for that, I double the rate up from 2ksps to 2Msps, and then keeping the 50.090 MHz the same with the LNB LO at 2.0 MHz just to test it out.

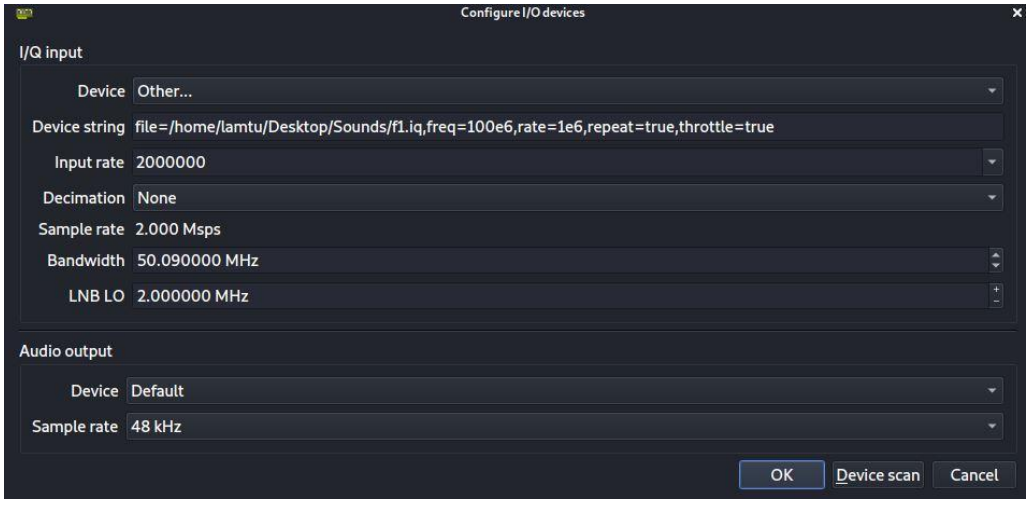

Figure #4: Configuring the setting during the initial setup page

After pressing the “OK” button the file opens, and then what I did is press the play button and the file start to pay a sound. From the sound, I recognized as ***Morse Code*** and trying to listen to it and looking the Morse Code up to decipher it. Below will be a screen showing how the program run using the file iq and how it is display. From here, you can see that the recording show spike in the middle to indicate sound, while there is a histogram showing the activity of the iq file.

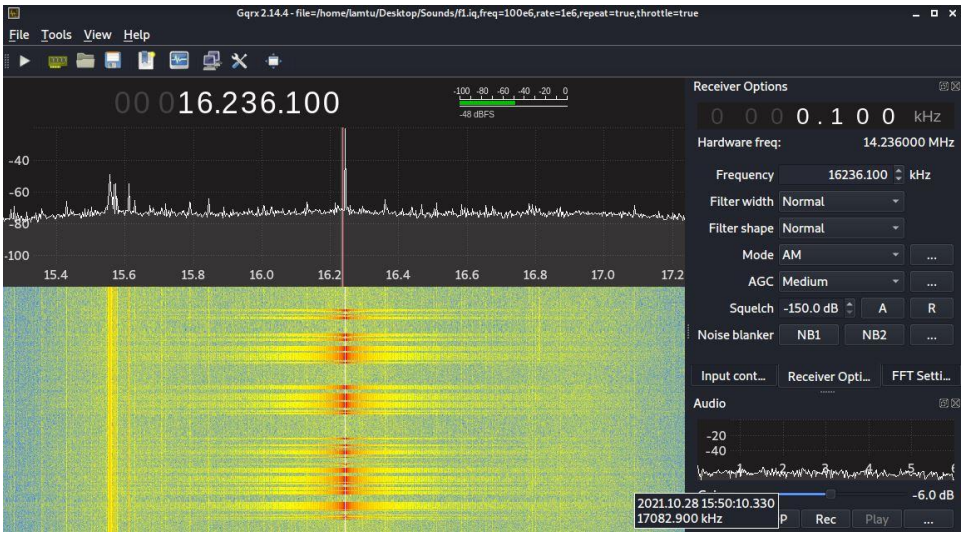

Figure #5: Showing the recording of the first iq file being played

Through carefully listening and deciphering the Morse Code on the first file. I end up with the code to be “***Sample 2 was capture at 146520KHz Sample rate 2M***”. This means that the bandwidth to produce the next iq file is at ***14.6520 Mhz*** and the input rate is at ***2Msps***. With this in consideration, I try that combination to open sample ***2.iq*** file. Below is the screenshot I enter in the new bandwidth and input rate for the file 2 and finding the path to it.

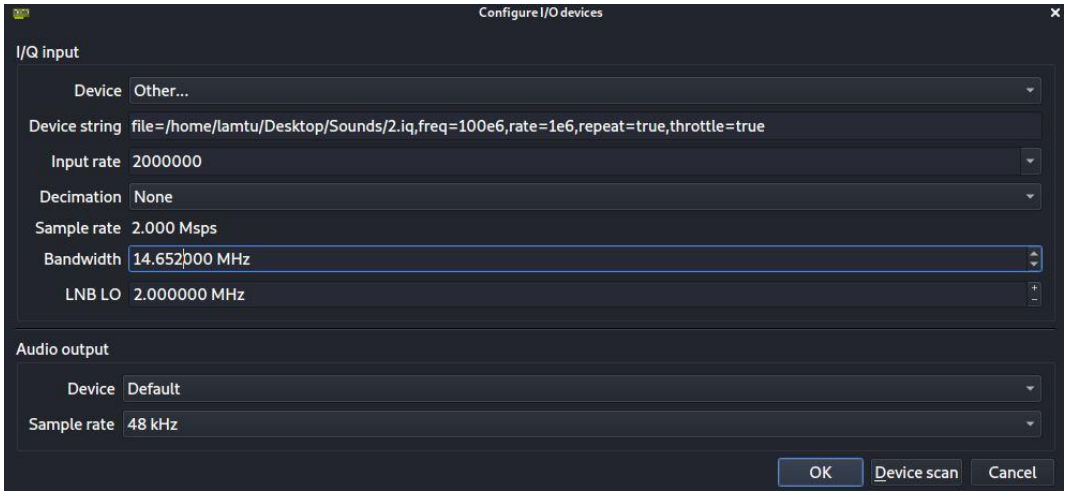

Figure #6: The new input setting when decoded the Morse Code in file 1

After pressing the “OK” button and start playing the file, I can hear a very muffle voice coming through to the file and the figure below will show it. From here, I would have to look around

the GQRX program and mess around the file to see if I can amplify the audio to produce a sound that we could hear and decode this file message.

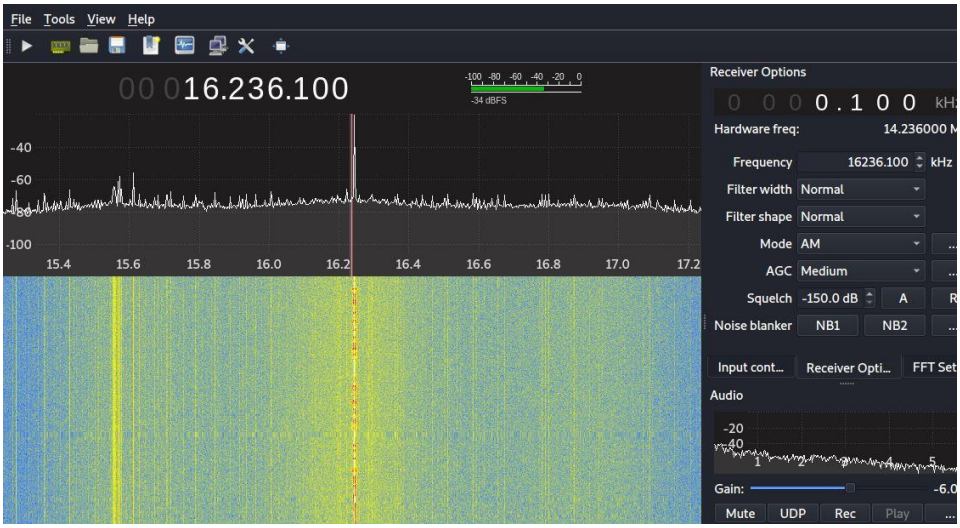

Figure #7: The output audio of the file 2.iq when use the frequency provided in 1.iq

When fiddling around with the GQRX program, I found out that the receiver options was at ***0.100 KHz***, and I play around with it and increase it to ***2.500 KHz*** to see if I get anywhere. Miracles happen and when I start playing the file again, I was able to hear the message clearly. The voice came through and said that "***The third file was capture at 7.171 MHz at 2.5 Msps***". Through this, we have deciper the 2nd file in the GQRX and can use this frequency to move onto the 3rd file and the last one. Using that number, I was able to setup the final file and below will be an image reflecting my input setting before I hit the "OK" button to continue with the program and try to understand the 3rd file.

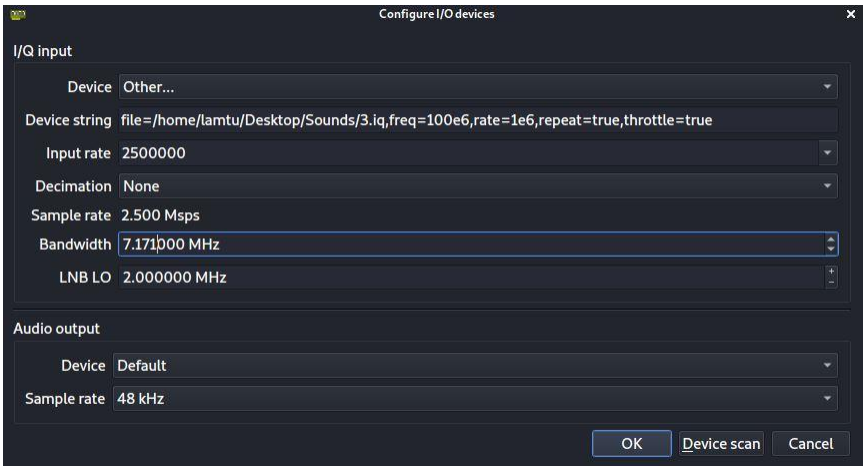

Figure #8: The configuration setting for the 3rd and last file in lab

Opening the third file and pressing the play button, I was only able to hear like a static popping noise coming from the file. This was very confusing to me as I thought maybe it needs to be amplified more to be hearing the sound. Then I try slowly incrementing the receiver options up and

up until I reach ***6.000 KHz*** and it produces a noise sound like a static noise of am machine printing something. Below is the image of the 3rd file being play on the GQRX program.

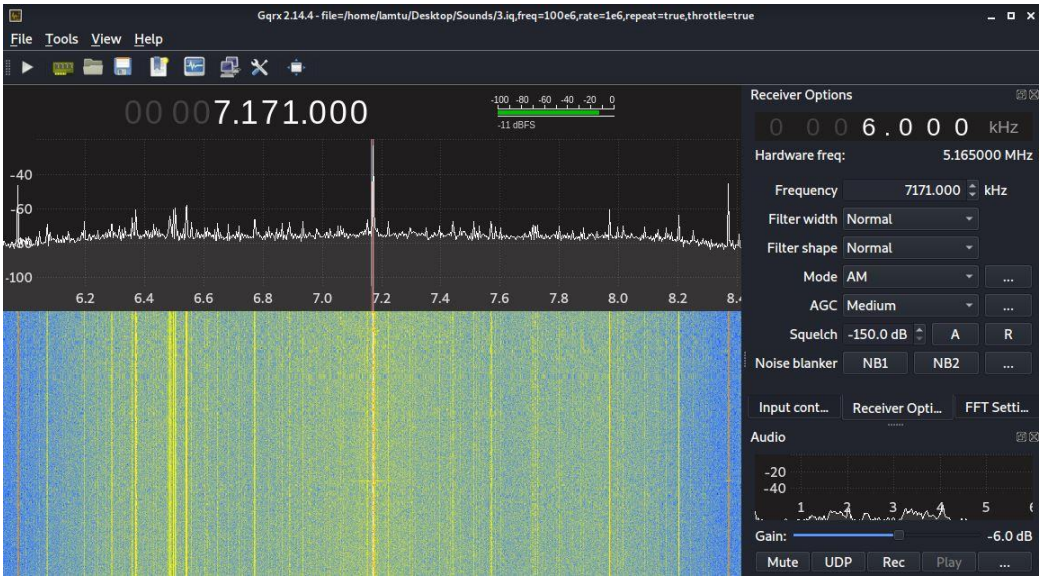

Figure #9: Show the 3rd file playing a sound at 6.000 kHz

Through this, I didn't know what the file is doing with this static noise. Then I did some research about it and found out that the audio is a sound that will produce an image once a program listens to the sound in the file. Through this, I found that ***QSSTV*** will work well with this file on hand. ***QSSTV*** is defined to be using radio signal to transmit an image either in color or black & white. Using this program, we can then debug the sound coming off this file. To do the installation, I would do the command "***sudo apt-get install qsstv***" for it to install on the terminal. Then moving on, I try to mess around with the file to record the static sound that the file is given off. From here, we can use this recording file to be use in ***QSSTV*** and use it to produce the image based on this noise. To record an audio, in the GQRX program, you go to ***Audio > Rec***. From there, you can setup your own path to where the recording will be saved, and the file is going to be in a .wav format. At that moment, I recorded the sound, and I raised the "Gain" section in the audio to pick up the decibel more easily and the static was record and got the audio file.

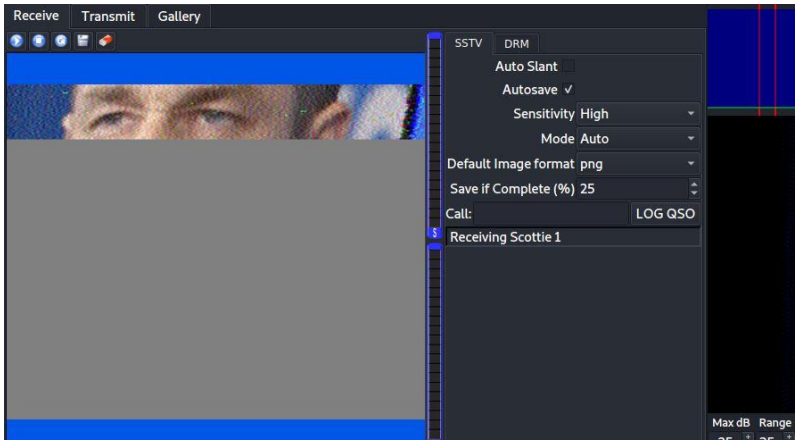

Figure #10: The product of using QSSTV to produce an image base on the sound file

Then once I got the audio file, I then imported to the QSSTV program by going to ***Option > Configuration > Sound tab > Check the import by file***. Through this, when we hit the play icon on the QSSTV, it will prompt you to select an audio file to be play. I recommend on the QSSTV setting to pick "sensitivity" and set it to high to get an image to be produce. In the figure 10 & 11, it shows my attempt to produce the image in the third iq file and I manage to get it to produce halfway correctly.

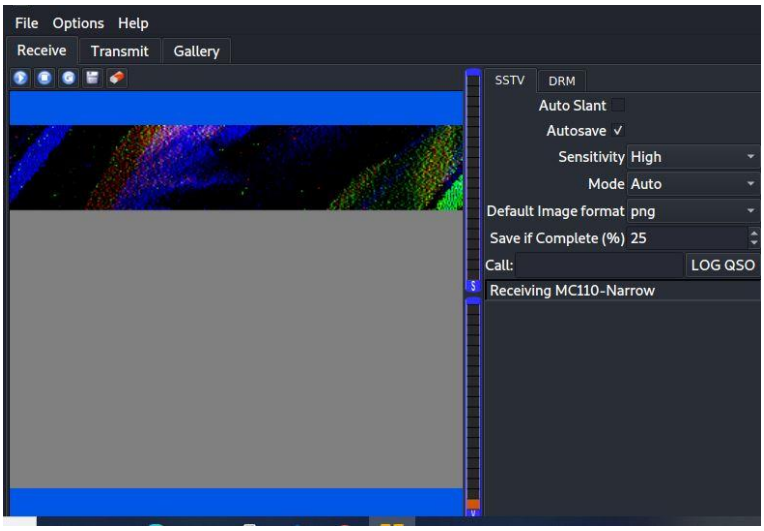

Figure #11: Another picture produce by QSSTV through audio

Through many tries and error, I was managed to produce something decent enough to represent what the image looks like from the 3rd iq file. Below is the figure that show the result of it. And from here, we are done with the lab.

Overall, the lab for this week let us deal with radio code and try our best to decipher the code ourselves with these files as the example. There were some tough one to decipher, but it was a fun effort to try out on the lab.

Figure #12: The final product of what the image represents through the iq file