GRCon 2022 CTF

Dune Challenges

Dune

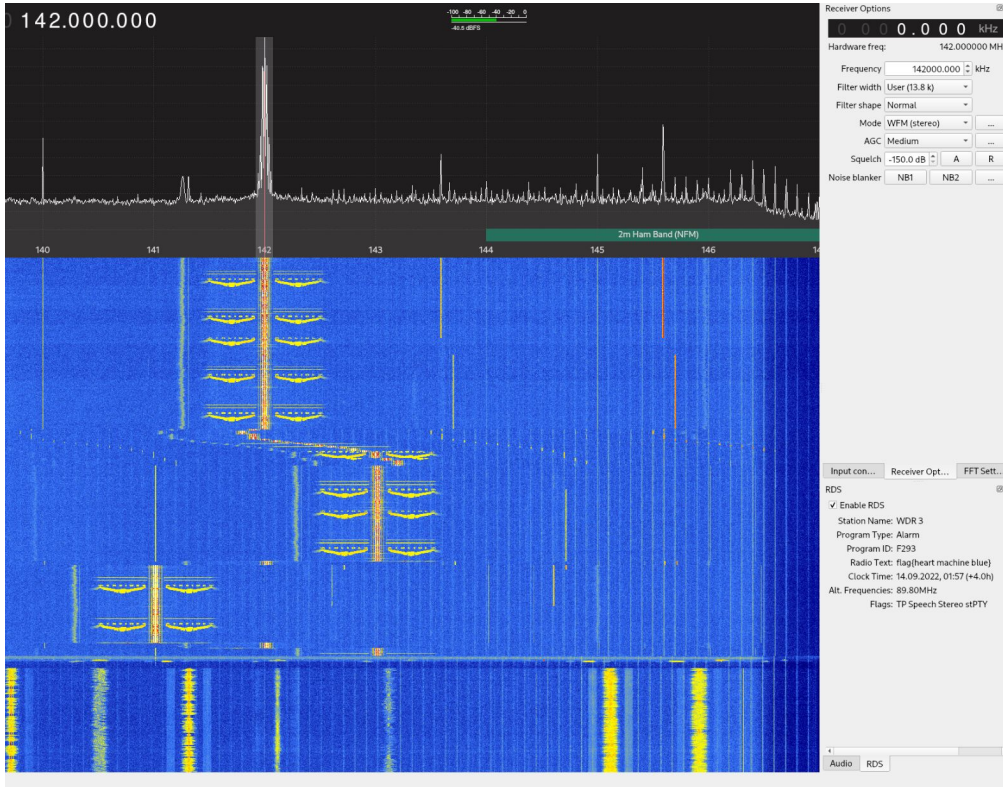
All challenges made by @muaddib on matrix. They had a scheduling conflict, so I am presenting these, including my solutions.

CHALLENGE "I can kill with a word..."

This is an RDS encoded stream with gr-paint banners (House Atreides) on either side. The banners are there to guide competitors to this signal from a hint in the challenge description (reference to the banner of house atreides).

flag{heart machine blue}

Flag was in RDS of FM carrier



CHALLENGE "Take Back Arrakis"

Challenge 1:

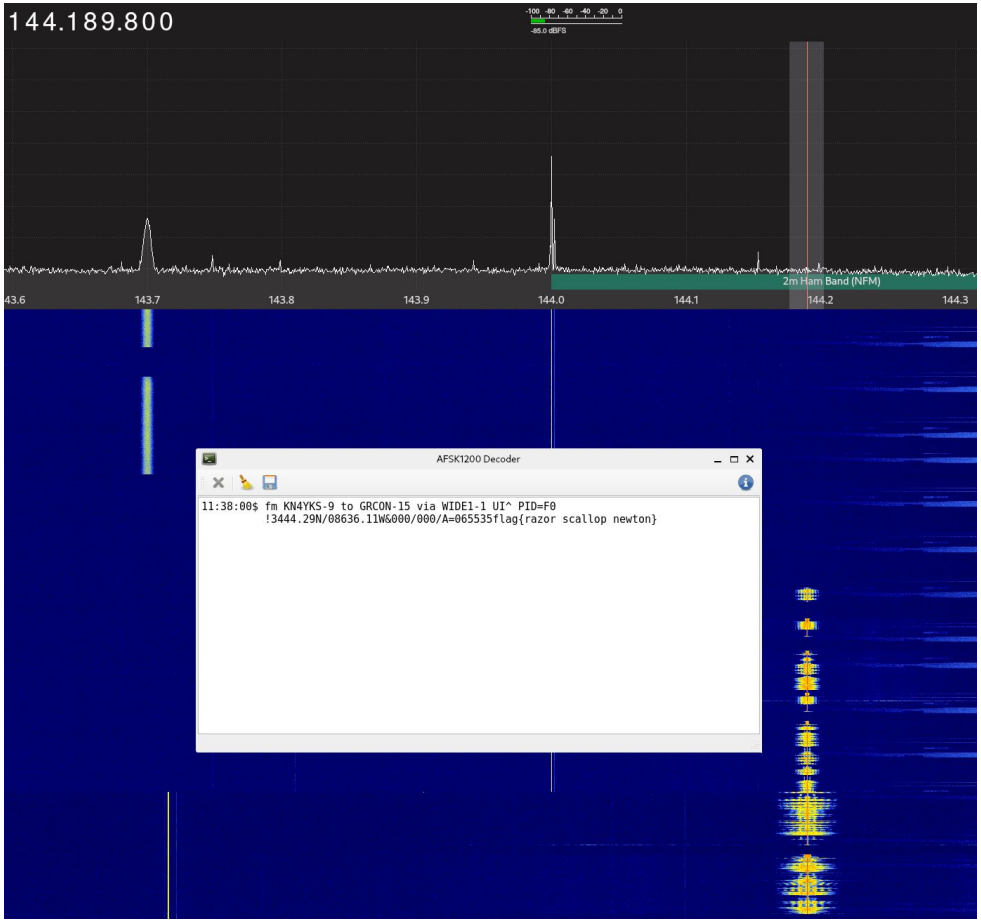
Analog APRS generated by an Anytone DT878UV, successfully decoded with GQRX AFSK1200 Decoder

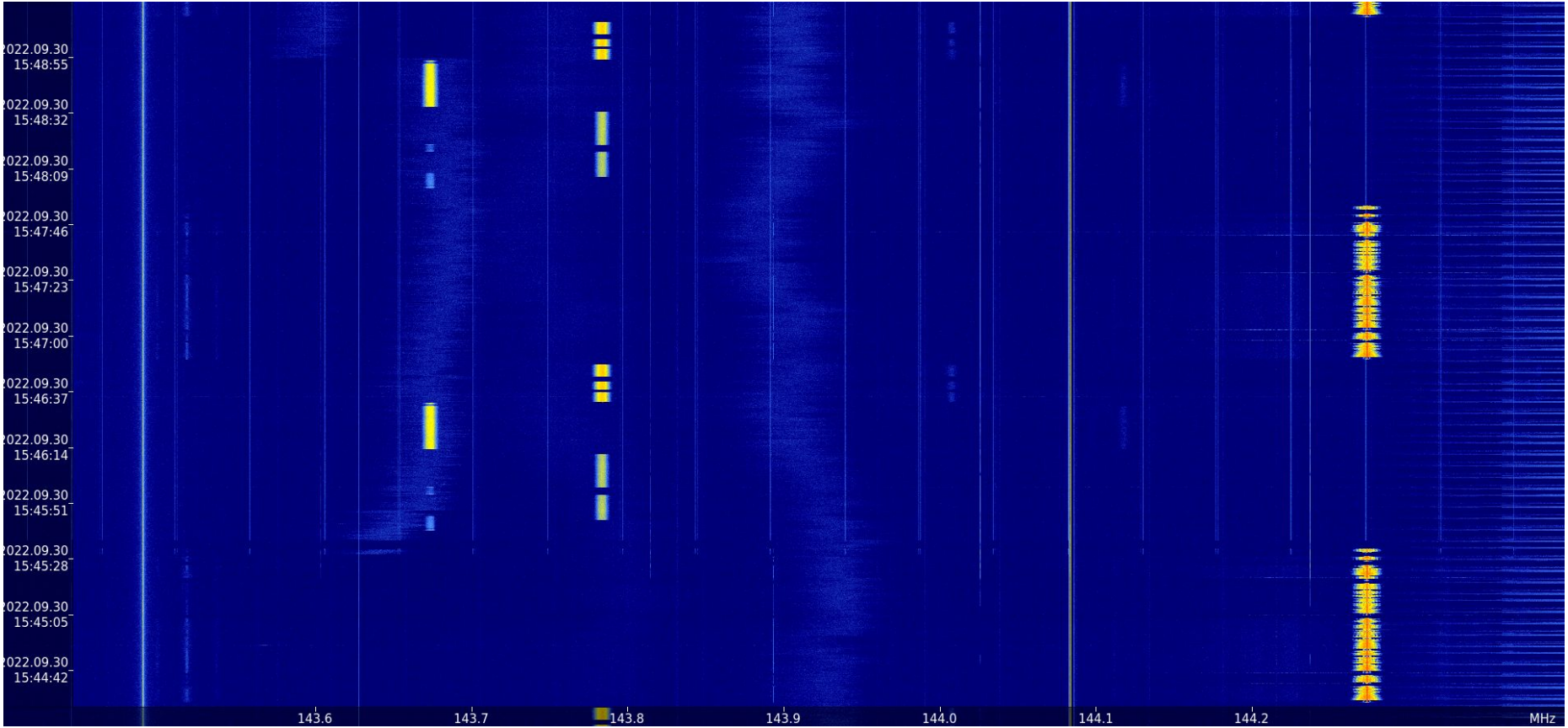
Center Freq:144.39MHz

Analog FM voice is telling the story most of the time with intermittent APRS bursts, each one contains the flag but the final one is the one that's decodable. In the audio dialog it says "switching to primary comms". In sync with that dialog, a pair of alternating carriers become active around 145.7MHz

Challenge 1 Flag:

flag{razor scallop newton}




2022.09.30 15:48:55
2022.09.30 15:48:32
2022.09.30 15:48:09
2022.09.30 15:47:46
2022.09.30 15:47:23
2022.09.30 15:47:00
2022.09.30 15:46:37
2022.09.30 15:46:14
2022.09.30 15:45:51
2022.09.30 15:45:28
2022.09.30 15:45:05
2022.09.30 15:44:42
143.6
143.7
143.8
143.9
144.0
144.1
144.2
MHz

CHALLENGE "Switching to Primary" / "The Key!"

Challenge 2 Part 1: "Switching to Primary"

DMR generated by an Anytone DT878UV

Center Freq: 145.79MHz

Use gr-DSD to decode voice stream where the flag is audible

Alternatively, GQRX UDP to DSD worked as well:

https://gqrx.dk/doc/streaming-audio-over-udp / while true; do nc -l -6 -u localhost 7355 | sox -t raw -esigned-integer -b16 -r 48000 - -esigned-integer -b16 -r 48000 -t raw - gain -6 | pv -r | padsp dsd -v -e -i - -o /dev/dsp -fr -mc; done

Challenge 2 flag:

flag{brain matter archer}

In the DMR Audio Stream there is a hint to watch for a transmission containing a key as well as encrypted information being sent on 'the high band'. The key is a plaintext LoRa transmission generated with a Heltec ESP32 Wifi board, it was recorded and shifted into the VHF frequency band. The key's transmission bursts line up with the last words as they are spoken "the key....you must receive the key!".

Challenge 2 Part 2: "The Key!"

You must listen to both DMR audio streams. The words from the stream without the audible flag are used for the next flag and as a segway to the next challenge.

flag{the key}

CHALLENGE "One Time Password" / "The Mind Killer"

LoRa message generated using gr-lora_sdr

(https://github.com/tapparelj/gr-lora_sdr OR for GNURadio 3.9+ https://github.com/bkerler/gr-lora_sdr port works)

LoRa Parameters (see NOTE below):

-samp_rate=250e3

-bandwidth=125e3

-spreading factor=7

-code rate=1

-center_freq=145.4e6

Challenge 3 Part 1: (key/freq is needed for Part 2):

key{LONGLIVEDUKELETO}

NOTE: this can only be decoded with SDR as the frequency is out of range for most of the LoRa chipsets.

Upper Band Lora Packets:
ae4b6954bb7011d920010c821f8a30ff7431697f2a0447ceda804febd11d0951 39243
ae4b6954bb7011d920010c821f8a30ff7431697f2a0447ceda804febd11d0951 39244
ae4b6954bb7011d920010c821f8a30ff7431697f2a0447ceda804febd11d0951 39245

Lower band packets:
4c4f4e474c49564544554b454c45544f 2032
4c4f4e474c49564544554b454c45544f 2033
4c4f4e474c49564544554b454c45544f 2034

Aka
LONGLIVEDUKELETO 2

CHALLENGE "The Mind Killer"

Following Part 1, there is a transmission that repeats for a few bursts at the end of switching to primary. In 915MHz (the high band) there's a second Lora Transmitter (Heltec ESP32 WIFI) which was running the whole CTF, it is sending an encrypted message that can be decoded with the key.

Encryption: AES128 ECB KEY: LONGLIVEDUKELETO

ENCRYPTED MESSAGE (Base64): rktpVLtwEdkgAQyCH4ow/3QxaX8qBEfO2oBP69EdCVE=

ENCRYPTED MESSAGE (Hex): ae 4b 69 54 bb 70 11 d9 20 01 0c 82 1f 8a 30 ff 74 31 69 7f 2a 04 47 ce da 80 4f eb d1 1d 09 51

DECRYPTED MESSAGE (Flag): flag{spindle negative comet}

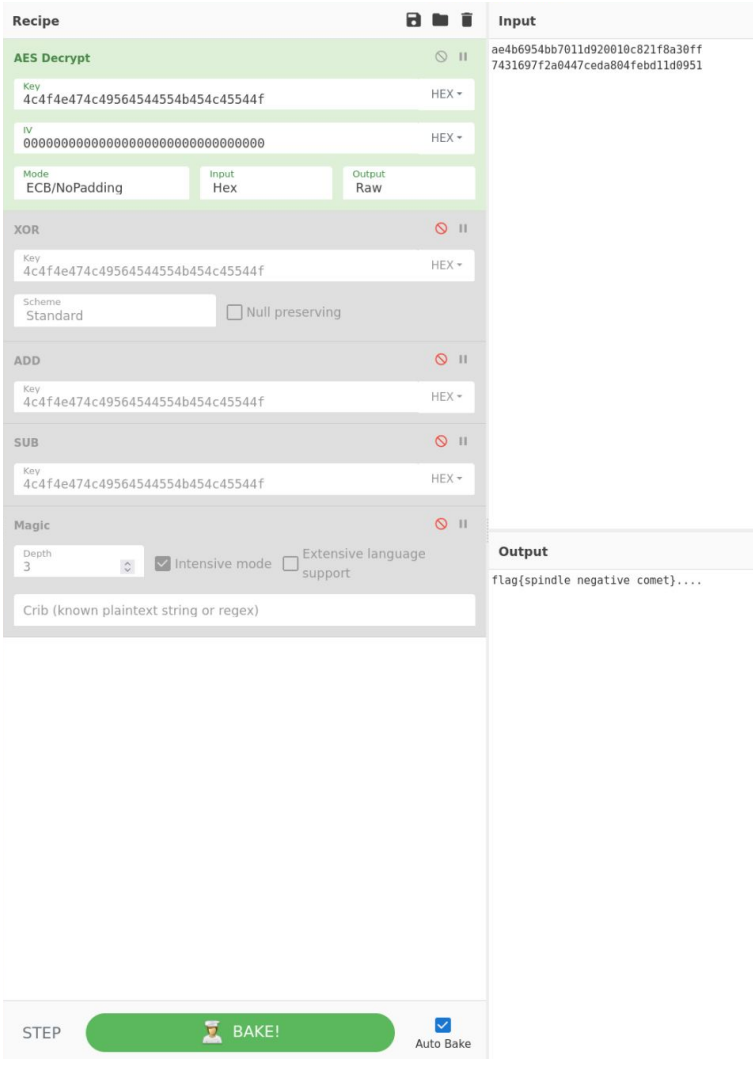
Decryption was tested with the 3 top hits on a google search for "AES128 encryption online":

https://encode-decode.com/aes-128-ecb-encrypt-online/

https://www.devglan.com/online-tools/aes-encryption-decryption

https://www.javainuse.com/aesgenerator

For my solve i used cyberchef:
https://gchq.github.io/CyberChef/#recipe=XOR(%7B'option':'Hex','string':'4c4f4e474c49564544554b454c45544f'%7D,'Standard',false/disabled)ADD(%7B'option':'Hex','string':'4c4f4e474c49564544554b454c45544f'%7D/disabled)SUB(%7B'option':'Hex','string':'4c4f4e474c49564544554b454c45544f'%7D/disabled)Magic(3,true,false,''/disabled)AES_Decrypt(%7B'option':'Hex','string':'4c4f4e474c49564544554b454c45544f'%7D,%7B'option':'Hex','string':'00000000000000000000000000000000'%7D,'ECB/NoPadding','Hex','Raw',%7B'option':'Hex','string':''%7D,%7B'option':'Hex','string':''%7D)&input=YWU0YjY5NTRiYjcwMTFkOTIwMDEwYzgyMWY4YTMwZmYKNzQzMTY5N2YyYTA0NDdjZWRhODA0ZmViZDExZDA5NTEK



CHALLENGE "HEIGHLINER"

Blockstream Satellite. This is a live recording from Galaxy 18 at K-Band, downconverted to 1.2GHz with an 18" dish for tailgating and a cheap LNB.

The message was generated on the blockstream transmission site and paid for with a bitcoin lightning wallet account. https://blockstream.com/satellite-queue/

-use the DVBS2-rx demod/decode from this repo: https://github.com/igorauad/gr-dvbs2rx

-install bitcoind

-use the blocksat api found here: https://github.com/Blockstream/satellite

use this command for the receiver:

dvbs2-rx --gui --in-repeat --mon-server --mon-port 8777 --source file --in-file /home/user/data/HEIGHLINER_1266400000Hz_2000000sps_75.0dB_2022_09_05_03_09_38.cfile --in-real-time --usrp-gain 70 --usrp-args rx2 --out-fd 4 --freq 1266400000 --samp-rate 2000000.0 --sym-rate 1000000 --rolloff 0.2 --modcod qpsk3/5 --frame-size normal --pilots on --ldpc-iterations 25 --mon-server 4>&1 1>&2 | tsp --realtime --buffer-size-mb 1.0 --max-flushed-packets 10 --max-input-packets 10 -P mpe --pid 32 --udp-forward --local-address 127.0.0.1 -O drop

Bitcoin:

run the bitcoind daemon

Blockstream/satellite

run with the following command:

./blocksat-cli.py api listen --no-password --plaintext --save-raw

message will be logged in .blocksat/api/downloads

flag{holder watchmen stilgar}