Practical IoT Hacking: Introduction to Multi-Band Hacking with the CatSniffer

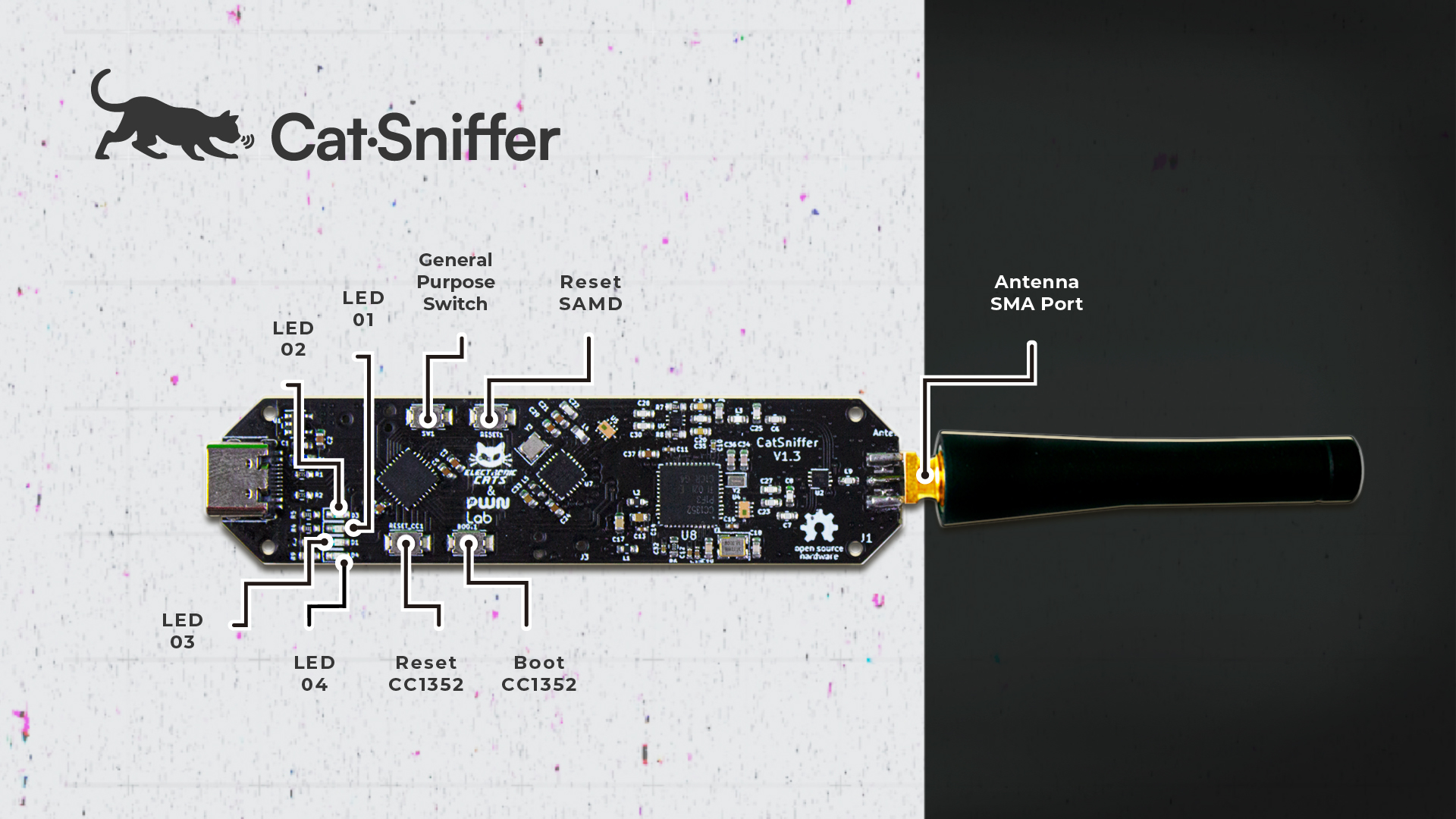
# 

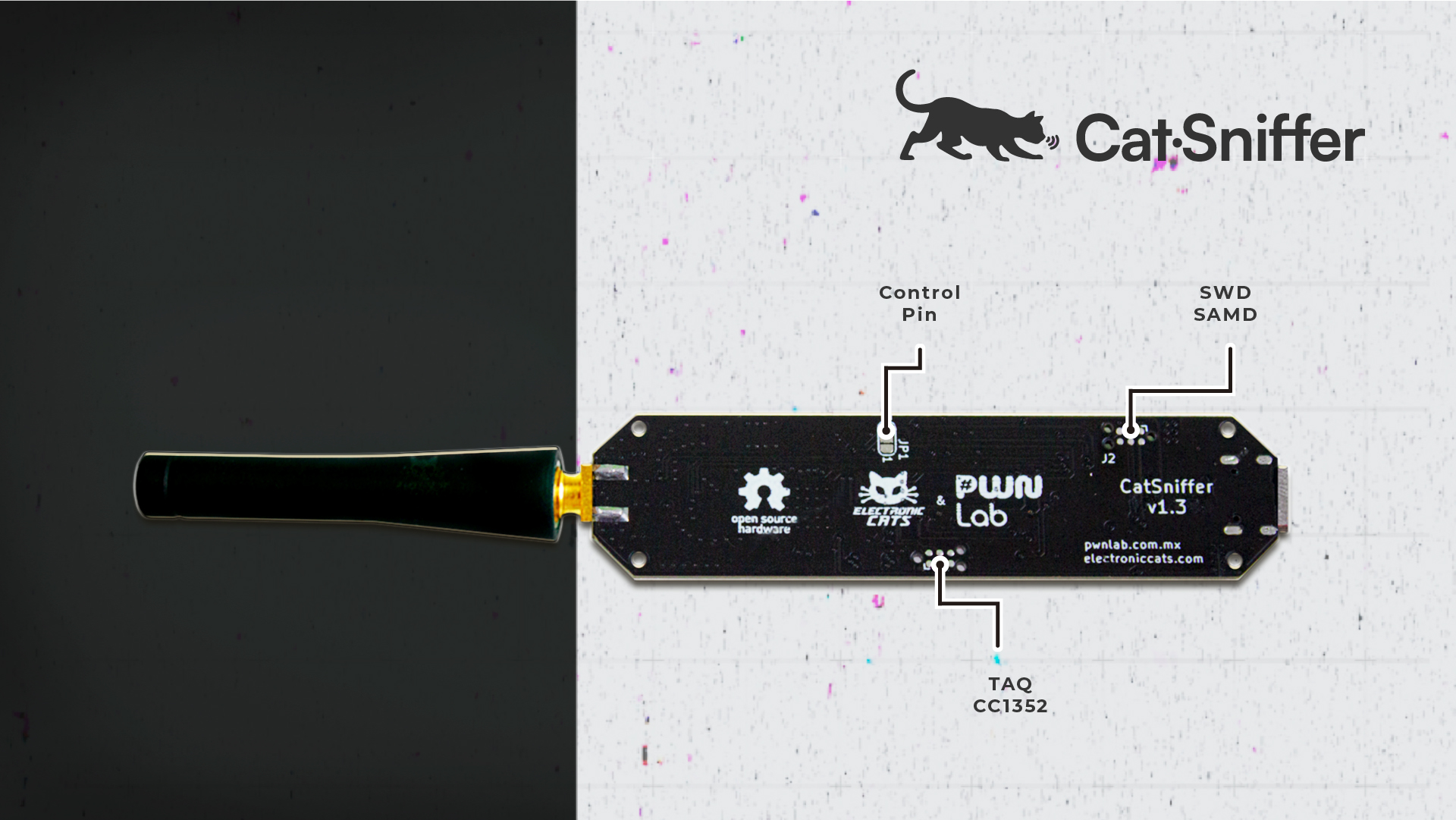
# Introduction

CatSniffer (😼) is an innovative, multi-protocol, multi-band circuit board for sniffing, communicating, and attacking IoT (Internet of Things) devices. It was designed as a highly portable USB stick that integrates TI CC1352, Semtech SX1262, and Microchip SAMD21E17 (V1.x and V2.x)/RP2040 (V3.x). This board is a Swiss army knife for IoT security researchers, developers, and enthusiasts.

It's highly versatile and compatible with a wide array of software, including:

* Packet Sniffers
* Security auditing tools
* Custom firmware developed by ElectronicCats/PWNLab





# Prerequisites for this lab

* Python3 and the dependencies described in the corresponding requirements.txt files.
* Code sources from the experimental branch:

https://github.com/ElectronicCats/CatSniffer-Tools

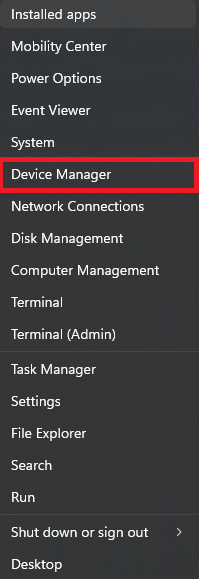
* Intro slides are available here, although everything you need is in this booklet: <https://docs.google.com/presentation/d/1PE6eKdoGi42qmx4hGn2er37uRNrHWFq4EXpf2alXad8>

# 

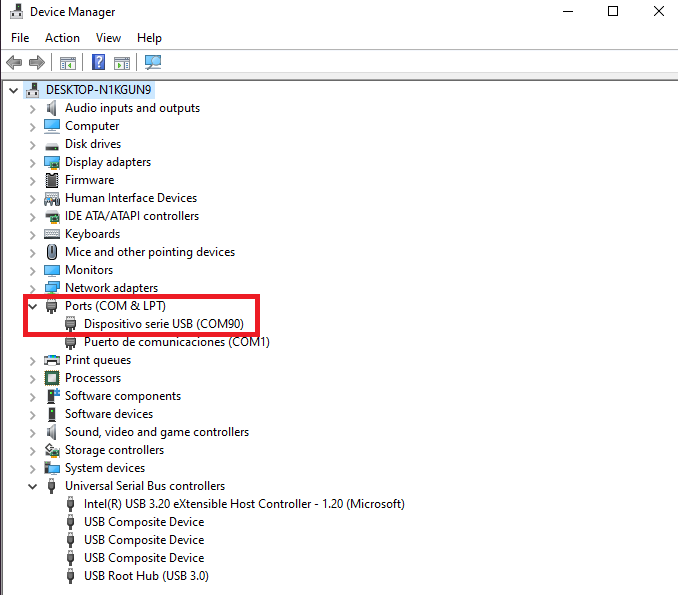
# Determine the serial port used by the Catsniffer

Windows instructions:

1. Make sure the CatSniffer is disconnected from your computer
2. Open the Device Manager (you can right-click on the start Windows button):



1. See the list of available devices under Ports (COM & LPT). Now, connect your CatSniffer via USB and list the serial devices available again. The new device in the list should be your Catsniffer:



1. In this case, your computer's serial port is COM90. Note that this value may change upon reconnections and is affected by the number of serial devices available in your computer.

# Flashing firmware with the Catnip uploader tool

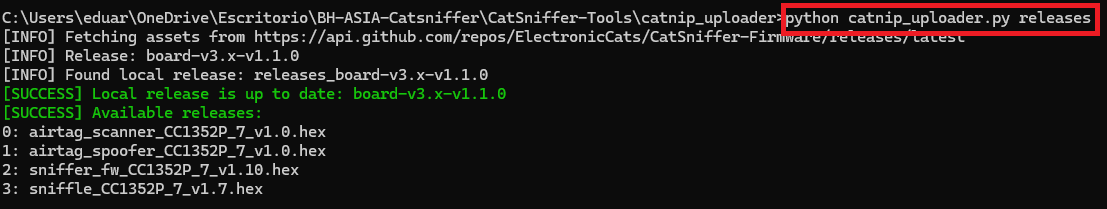
The Catnip tool is an open-source Python tool for configuring the Catsniffer in bootloader mode and uploading pre-built firmware distributed by Electronic Cats and PWNLabs.

You can read more about this tool [here](https://github.com/ElectronicCats/CatSniffer/wiki/05.-Uploading-Firmware#catnip-uploader-tool-catnip_uploaderpy).

In order to use this tool, you will need to install all the dependencies listed on the **“requirements.txt”** file with the command: **pip install -r requirements.txt**. This should already be done on the computer you are working on in this workshop**.**

The first thing you want to do is list the available firmware releases:

**python catnip\_uploader.py releases**



This will list the compatible firmware files for your catsniffer that you can install:

* 0 airtag\_scanner - Scanner for active airtags nearby with BLE activated.
* 1 airtag\_spoofer - Emulate the payload from an airtag in BLE.
* 2 sniffer\_fw - Sniffer firmware to work with Wireshark and our tool “pycatsniffer\_bv3” also compatible with the TI sniffer 2
* 3 sniffle - Sniffer firmware to work with Wireshark and the project Sniffle.

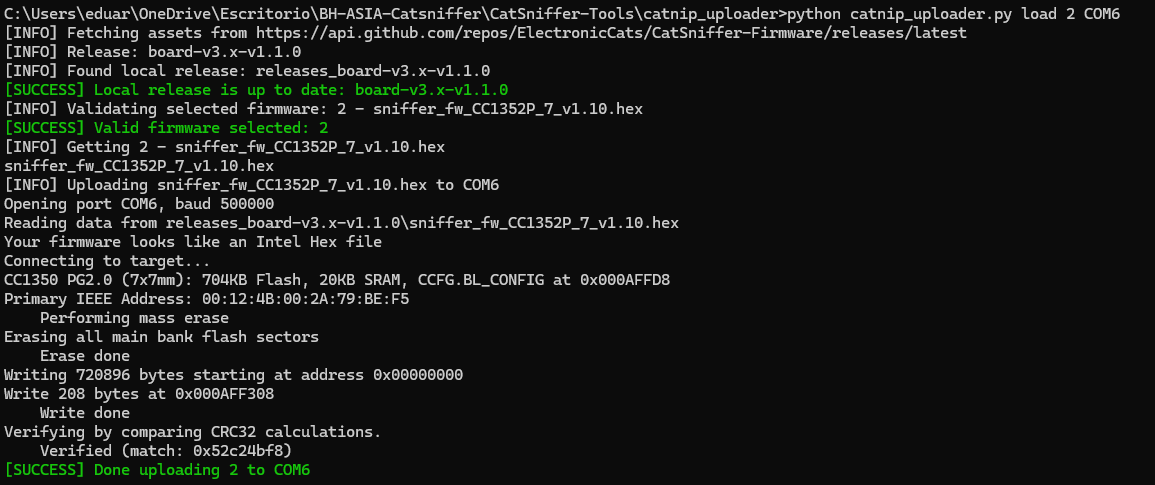
Now you can upload any firmware from the list using the command “load” followed by the firmware id and the serial port:

**python catnip\_uploader.py load <firmware id> <serial port>**

For example, to load the firmware id #0 using the serial port COM90, you would use the following command:

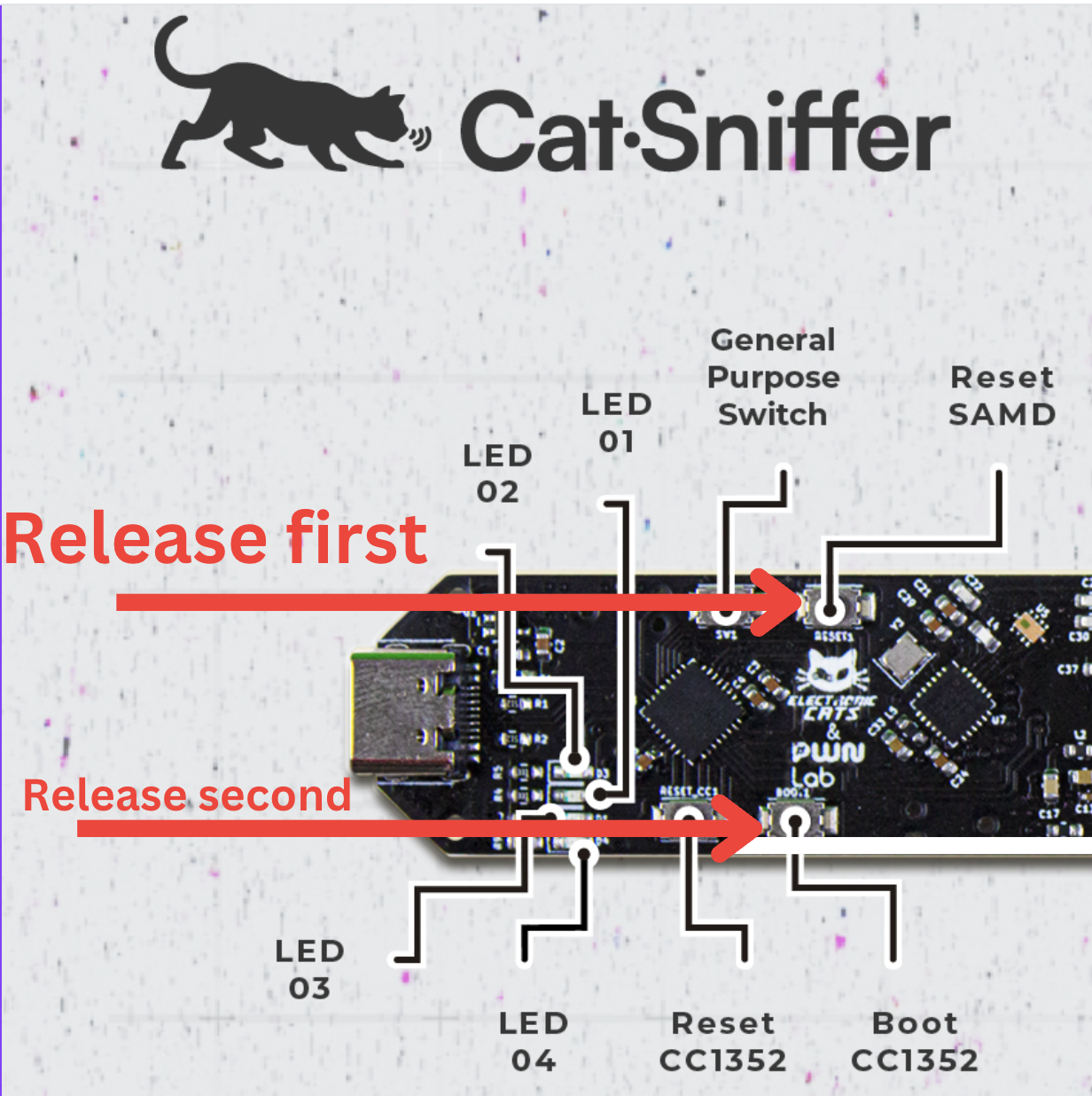
**python catnip\_uploader.py load 0 COM90**

Note: See the section “[Determine the serial port used by the Catsniffer](#5g2urotrxoag)” if you need help determining which serial port to use.

****

# Putting the Catsniffer in upload mode

1. Connect the Catsniffer to your computer
2. Press and hold the buttons Reset SAMD and Boot CC1352
3. Release the button Reset SAMD while holding the button Boot CC1352
4. Release the button Boot CC1352
5. You should see an LED cascading animation to indicate the CatSniffer is in upload mode.



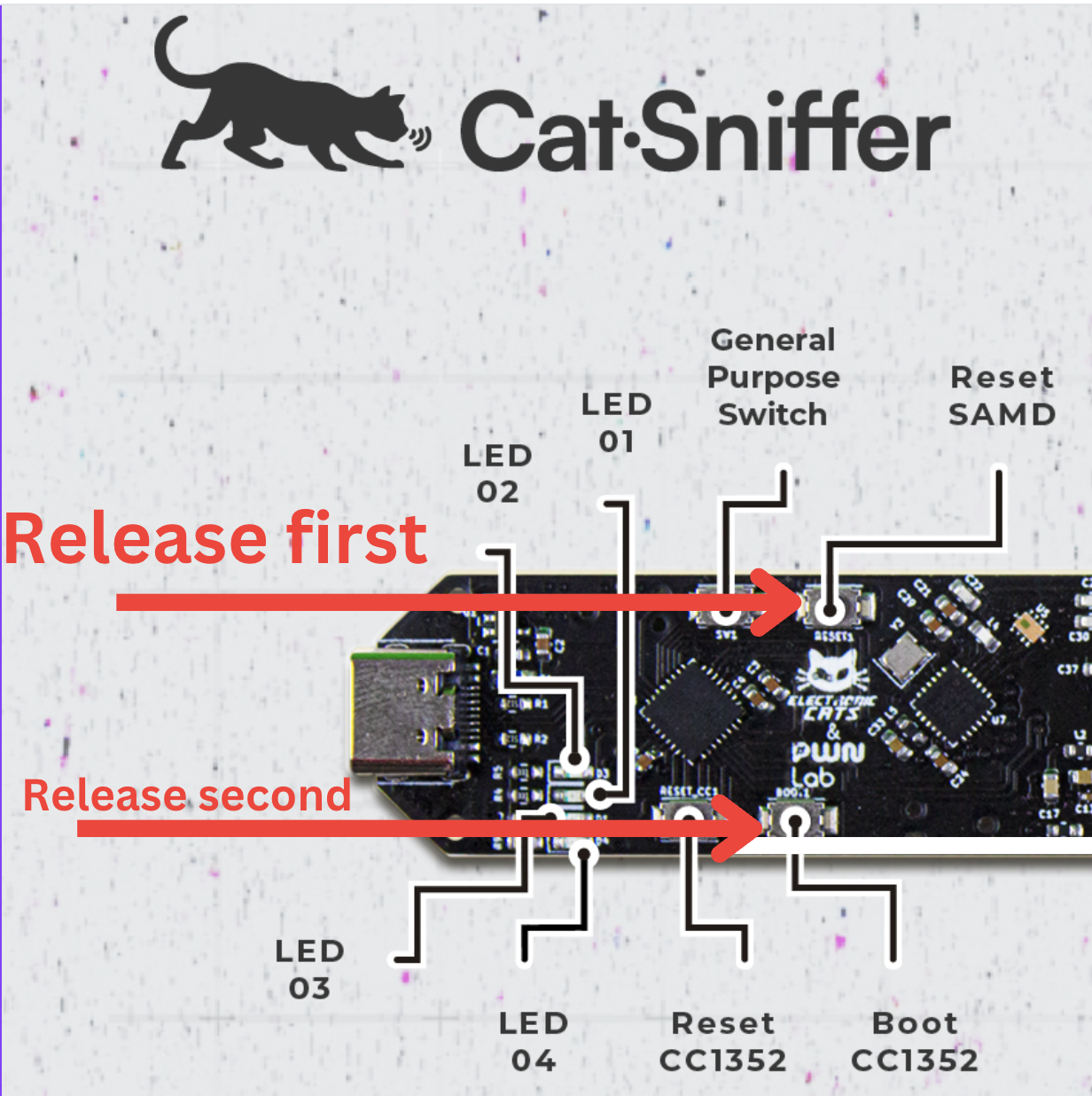
# Flashing firmware with the utility cc2538-bsl

To program your CatSniffer with one of the many firmware files available:

1. Locate the desired firmware. You will usually find the firmware in .hex format.

<https://github.com/ElectronicCats/CatSniffer/tree/master/firmware>

1. Put the CatSniffer in upload mode for the CC1352P7 (Catsniffer v3.x) by holding both the reset button for the RP2040 and the boot for the CC1352 (See image below), then releasing the reset button first and then the boot button. You should see an animation on the LEDs in the CatSniffer.



1. Locate the utility cc2538-bsl.py in the folder /Code/ (Or /tools/cc2538-bsl/ if you are working with the latest version of the Catsniffer’s GitHub repository).
2. Install the requirements for cc2538-bsl.py by issuing the following commands:

$python3 -m pip install pyserial

$python3 -m pip install intelhex

1. Reset the RP2040 (Reset SAMD in v2.x) once you have flashed the CC1352P chip.
2. Run the command

**python3 cc2538-bsl.py -e -w -v -p <SERIAL PORT> <HEX FILE>**

# Hands-on Lab #1: Sniffing BLE Packets

## Objectives

* Flash firmware for sniffing Bluetooth Low Energy connections
* Learn to analyze BLE traffic
* Study the packet structure of different types of BLE packets

## Challenge

Flash firmware for sniffing and capturing BLE traffic to analyze and learn more about the different types of advertisement packets in Bluetooth Low Energy devices. Attempt to identify the beacons emitted by the personal trackers around the lab and spoof your trackers.

## Walk-Through

1. Connect the Catsniffer and identify its serial port number. See [Determine the serial port used by the Catsniffer](#5g2urotrxoag) if you need help.
2. Open a command prompt and move to the working directory of the workshop “Catsniffer-bhasia24”.

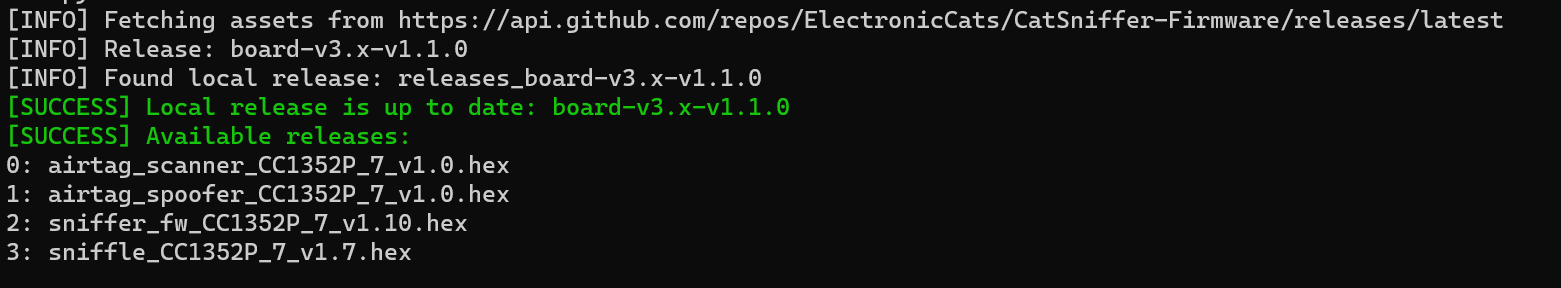
cd C:\Users\<computer username>\Desktop\Catsniffer-BHAsia24

1. Move to the folder containing the catnip\_uploader tool:

cd CatSniffer-Tools-experimental\CatSniffer-Tools-experimental\catnip\_uploader

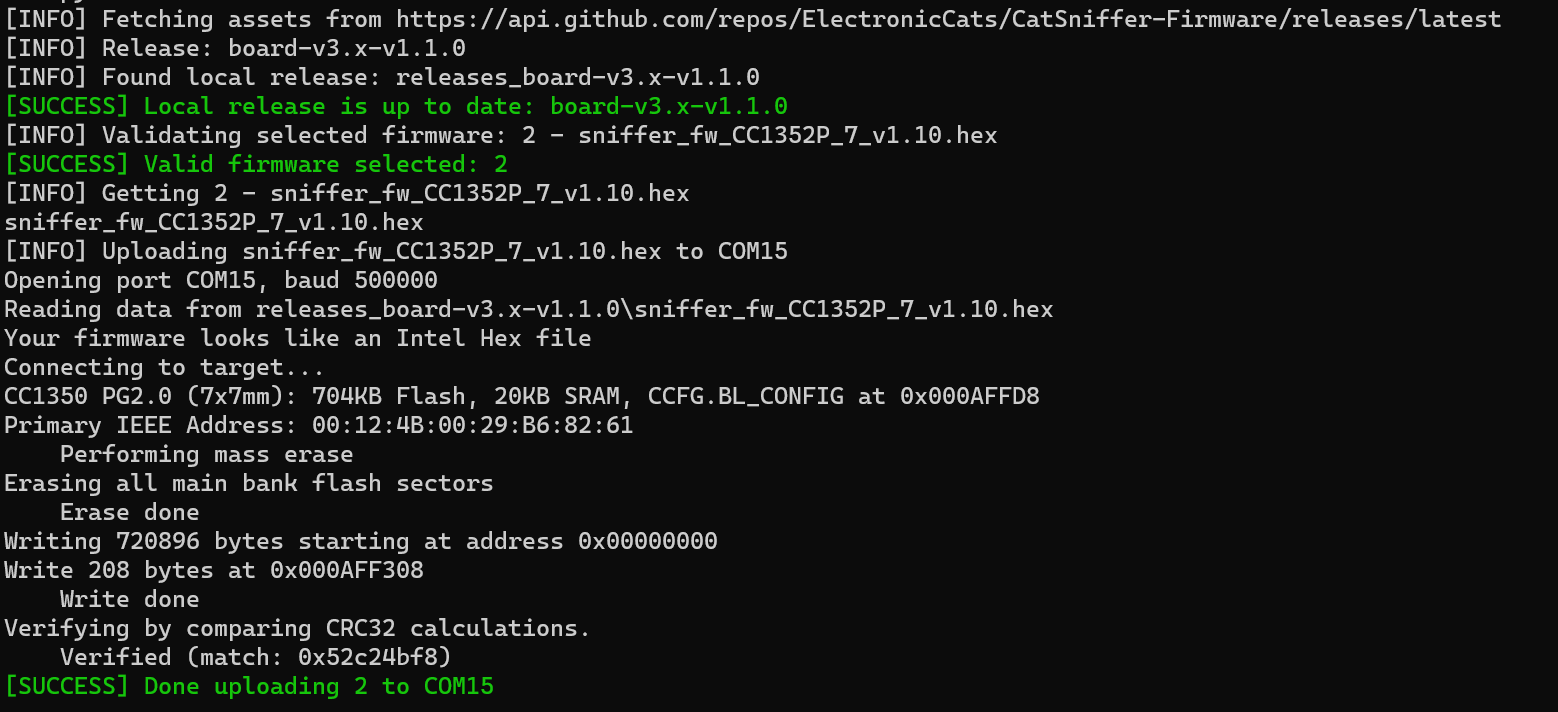
1. Find the firmware id of “sniffer\_fw\_CC1352P\_7\_v1.10.hex” in the list of available firmware releases using the following command:

python catnip\_uploader.py releases



1. Flash the firmware “sniffer\_fw\_CC1352P\_7\_v1.10.hex” using the **load** command as follows. Make sure you replace <SERIAL PORT> with the serial port you obtained in Step 1:

python catnip\_uploader.py load 2 <SERIAL PORT>



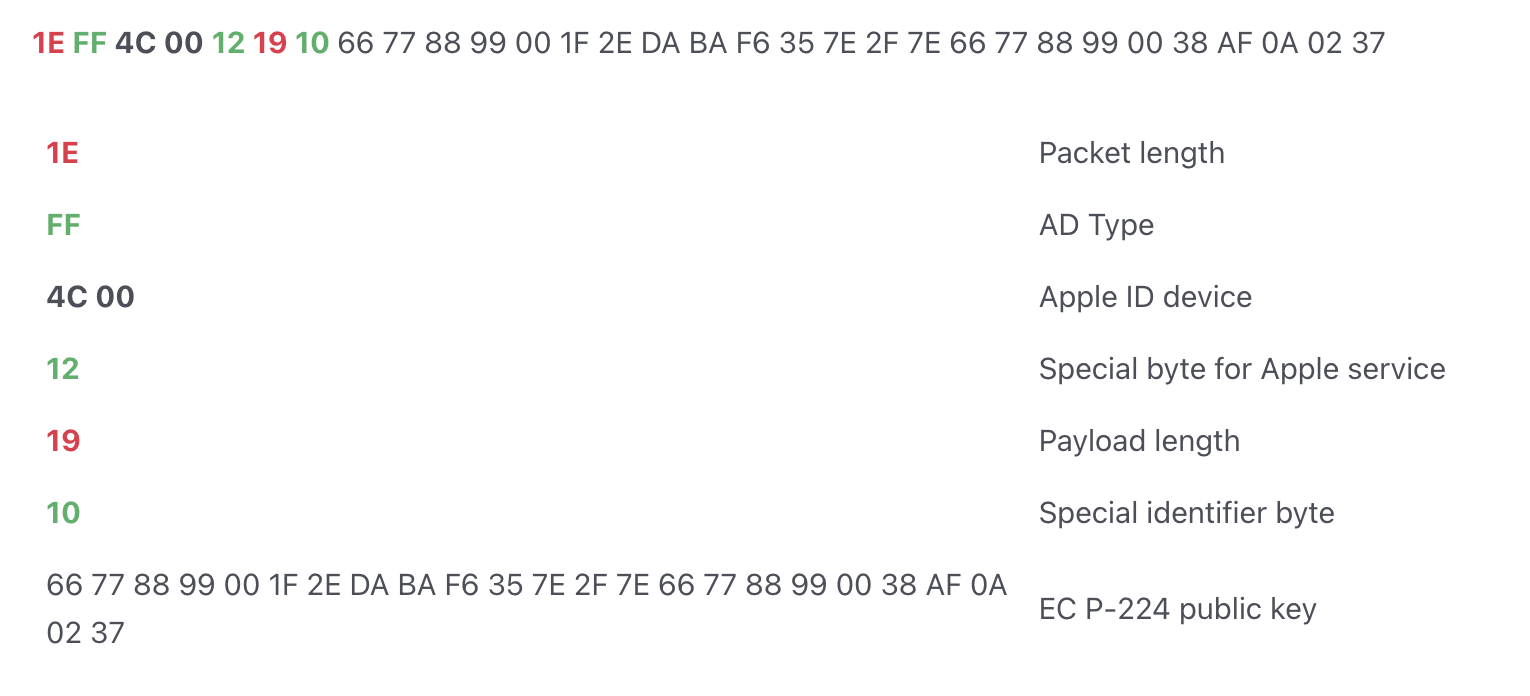
1. Disconnect and re-connect the Catsniffer from the USB port (Or click the RESET button on the Catsniffer)
2. Run the BLE sniffer with Wireshark, and you should see BLE traffic now.

$ python cat\_sniffer.py sniff <SERIAL PORT> -phy 0 -ch 39 -ff -ws

1. Let Wireshark capture some BLE packets and look for the specific BLE beacons from personal trackers. Add a filter to see the BLE packets from a specific manufacturer (Apple).

btcommon.eir\_ad.entry.company\_id == 0x004c

1. Analyze the received beacons and look for packets with a similar packet structure as shown in the following diagram. These are the packets broadcasted by Apple Airtags:



# Hands-on Lab #2: Sending BLE Packets

## Objectives

* Flash firmware for crafting Bluetooth Low Energy packets
* Spoof personal trackers by crafting a custom BLE packet

## Challenge

Flash firmware for crafting BLE packets and transmit beacons to spoof personal trackers.

## Walk-Through

1. Connect the Catsniffer and identify its serial port number. See [Determine the serial port used by the Catsniffer](#5g2urotrxoag) if you need help.
2. Open a command prompt and move to the working directory of the workshop “Catsniffer-bhasia24”.

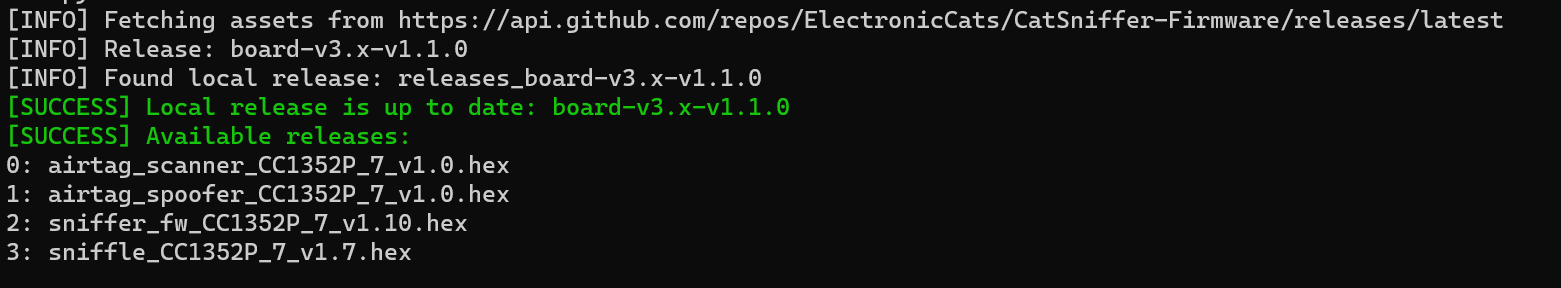
cd C:\Users\<computer username>\Desktop\Catsniffer-BHAsia24

1. Move to the folder containing the catnip\_uploader tool:

cd CatSniffer-Tools-experimental\CatSniffer-Tools-experimental\catnip\_uploader

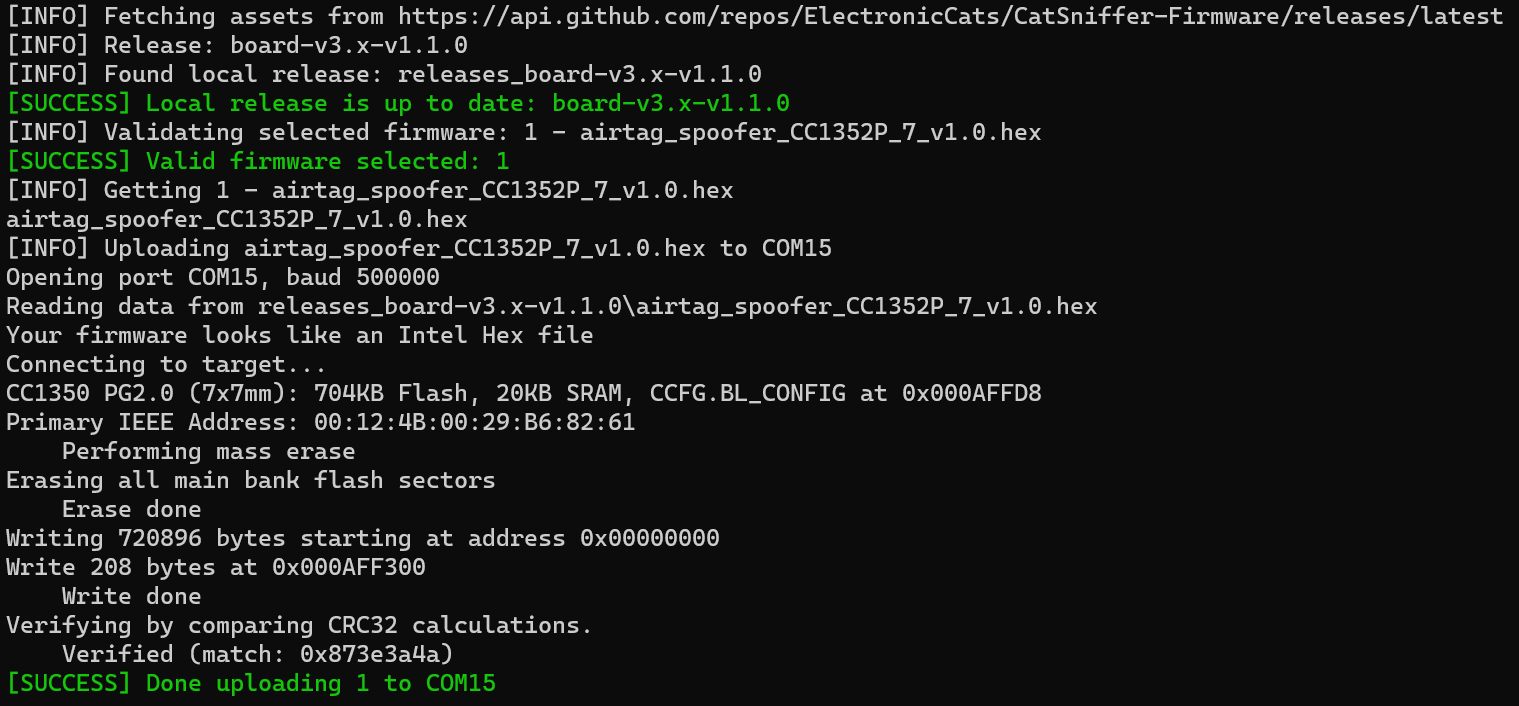
1. Find the firmware id of “airtag\_spoofer\_CC1352P\_7\_v1.0.hex” in the list of available firmware releases using the following command:

python catnip\_uploader.py releases

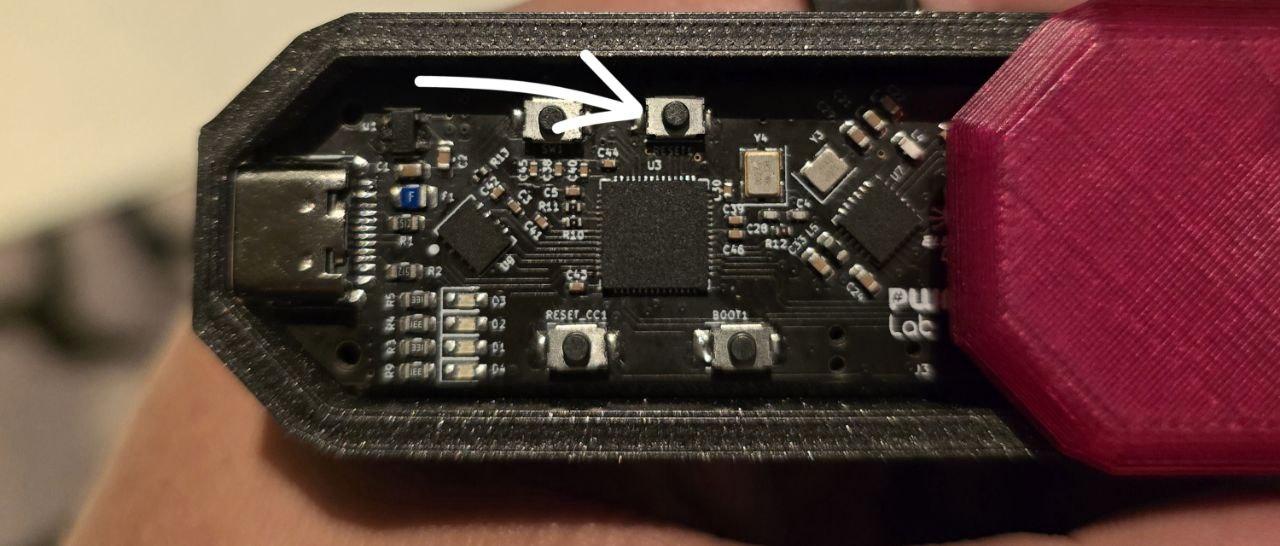


1. Flash the firmware “airtag\_spoofer\_CC1352P\_7\_v1.0.hex” using the **load** command as follows. Make sure you replace <SERIAL PORT> with the serial port you obtained in Step 1:

python catnip\_uploader.py load 1 <SERIAL PORT>



1. Disconnect and re-connect the Catsniffer from the USB port (Or click the RESET button on the Catsniffer)
2. The firmware running on the Catsniffer will emit a BLE beacon with the Apple Airtag format everytime it starts. Use the official tracker application to check our spoofing attack was succesful.
3. Feel free to press RESET many times to emit more beacons.
4. Study the file <https://github.com/ElectronicCats/CatSniffer-Firmware/blob/v3.x/CC1352P7/airtag_spoofer_CC1352P_7/Application/airtag_spoofer.c#L888> to learn how easy it is to craft custom packets with the Catsniffer.



# Hands-on Lab #3: Sniffing unencrypted Zigbee Packets

## Objectives

* Learn to analyze Zigbee traffic
* Study the packet structure of plain text Zigbee packets

## Challenge

Sniff unencrypted Zigbee packets shared over the air at the lab to obtain the super secret message.

## Walk-Through

1. If your Catsniffer is running the firmware “sniffer\_fw\_CC1352P\_7\_v1.10.hex”, feel free to skip to Step 8.
2. Connect the Catsniffer and identify its serial port number. See [Determine the serial port used by the Catsniffer](#5g2urotrxoag) if you need help.
3. Open a command prompt and move to the working directory of the workshop “Catsniffer-bhasia24”.

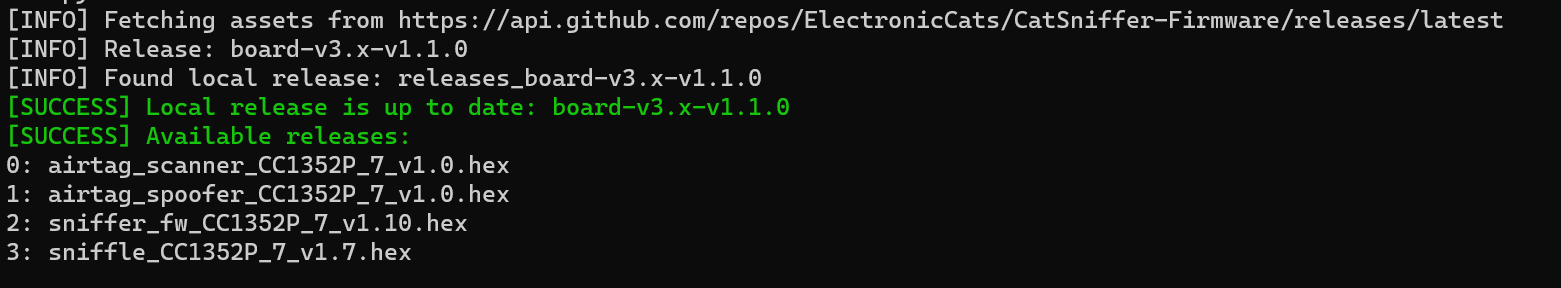
cd C:\Users\<computer username>\Desktop\Catsniffer-BHAsia24

1. Move to the folder containing the catnip\_uploader tool:

cd CatSniffer-Tools-experimental\CatSniffer-Tools-experimental\catnip\_uploader

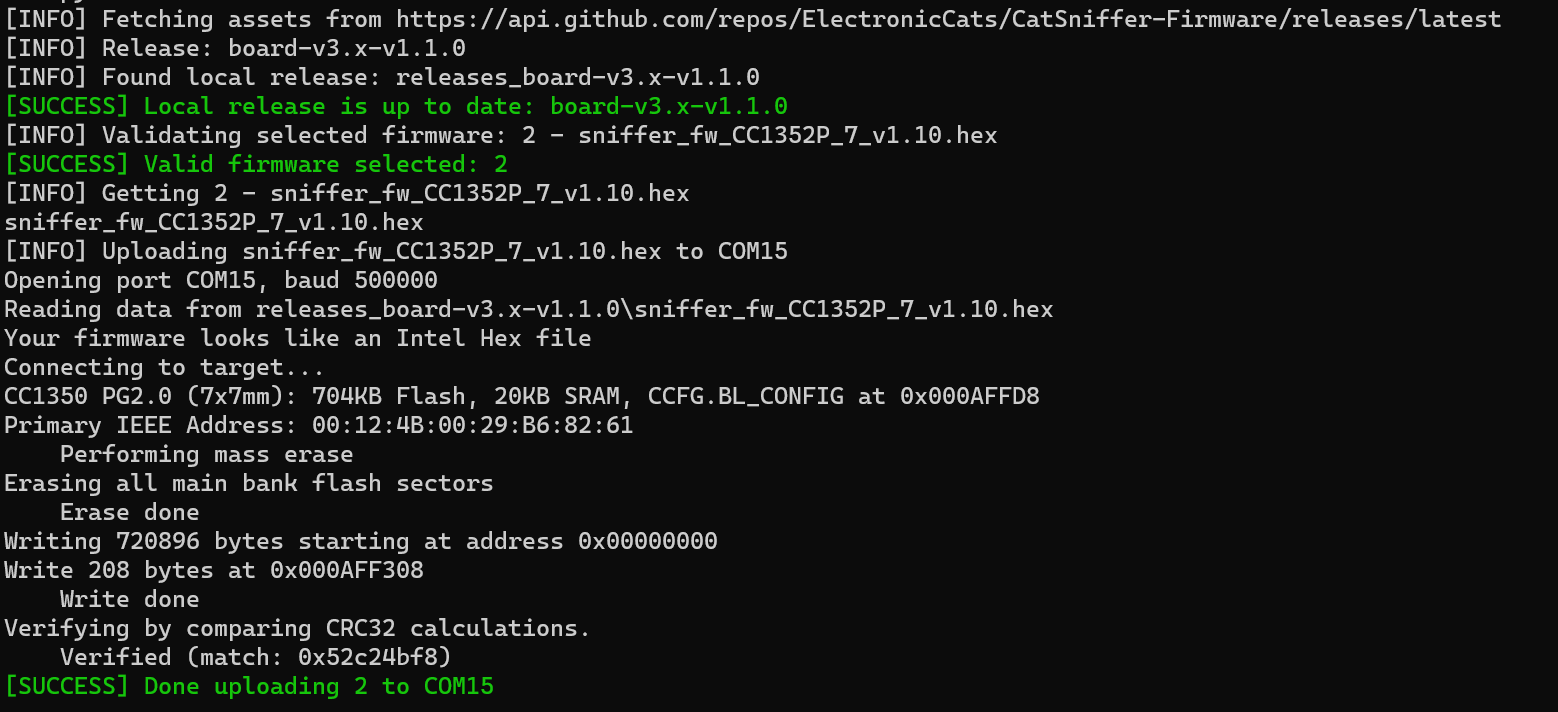
1. Find the firmware id of “sniffer\_fw\_CC1352P\_7\_v1.10.hex” in the list of available firmware releases using the following command:

python catnip\_uploader.py releases



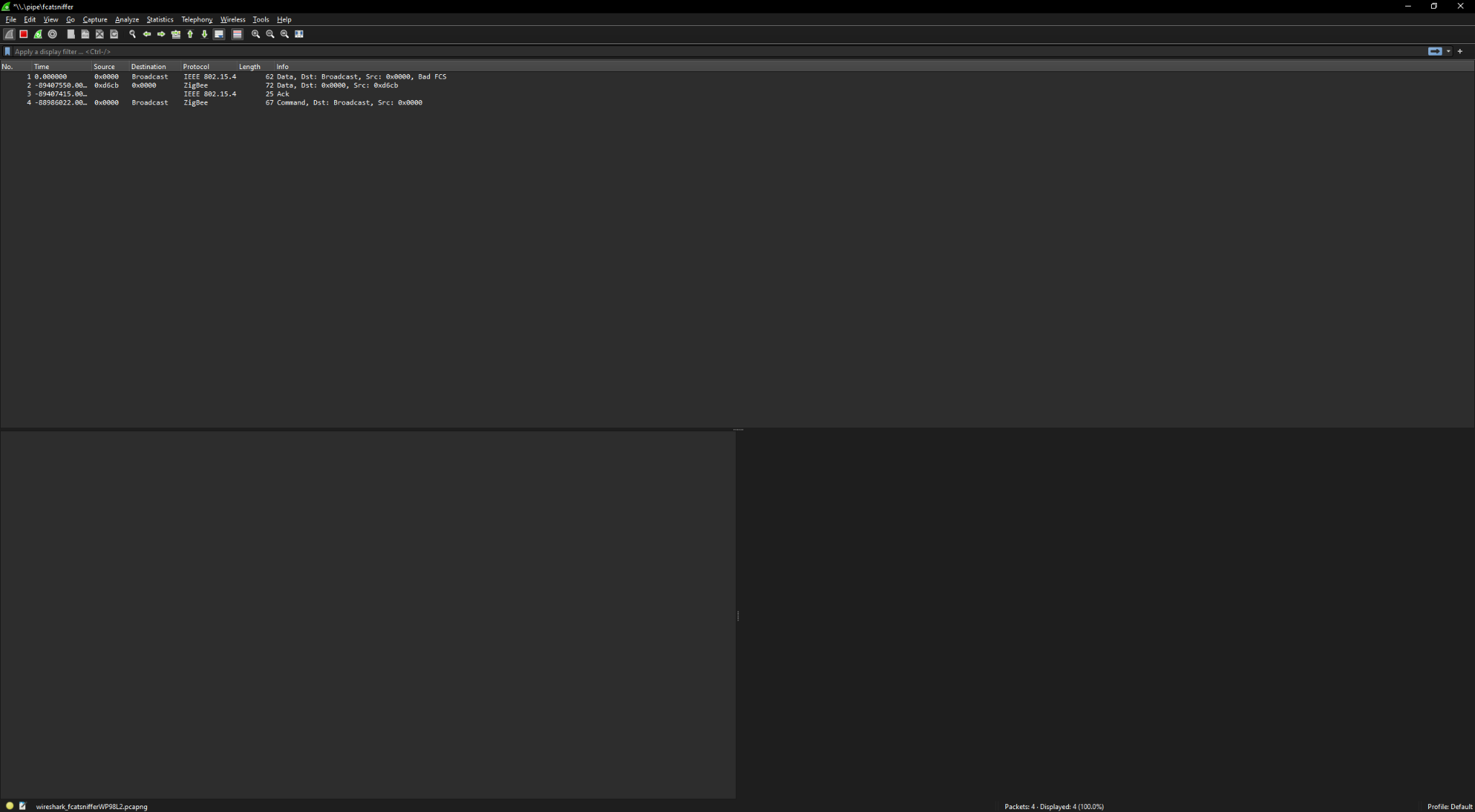
1. Flash the firmware “sniffer\_fw\_CC1352P\_7\_v1.10.hex” using the **load** command as follows. Make sure you replace <SERIAL PORT> with the serial port you obtained in Step 1:

python catnip\_uploader.py load 2 <SERIAL PORT>



1. Disconnect and re-connect the Catsniffer from the USB port (Or click the RESET button on the Catsniffer)
2. Start sniffing with the CatSniffer and the PyCatSniffer tool using the following command:

python cat\_sniffer.py sniff <SERIAL\_PORT> -ff -ws -phy 1 -ch 25



1. Once Wireshark is running, find the unencrypted Zigbee packet that contains the super secret message.

# Hands-on Lab #4: Sniffing Distributed ZigBee Networks

## Objectives

* Learn about how Zigbee distributed networks work
* Learn how to grab Zigbee keys over the air to decrypt data packets
* Learn ZigBee Device Profiles works

## Challenge

Capture Zigbee traffic and intercept the transport key to decrypt the data transmitted by the end devices around the lab.

## Walk-Through

1. If you completed Lab #4: Sniffing unencrypted Zigbee packets, feel free to skip to Step 8.
2. Connect the Catsniffer and identify its serial port number. See [Determine the serial port used by the Catsniffer](#5g2urotrxoag) if you need help.
3. Open a command prompt and move to the working directory of the workshop “Catsniffer-bhasia24”.

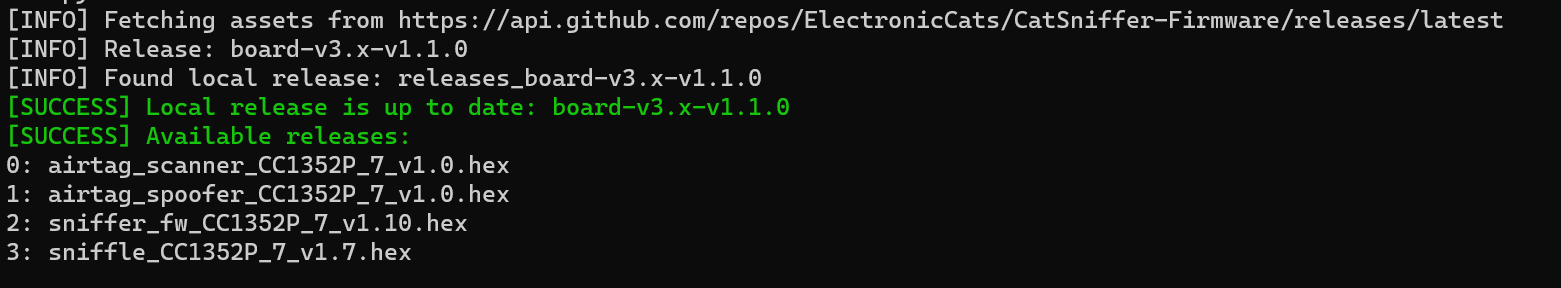
cd C:\Users\<computer username>\Desktop\Catsniffer-BHAsia24

1. Move to the folder containing the catnip\_uploader tool:

cd CatSniffer-Tools-experimental\CatSniffer-Tools-experimental\catnip\_uploader

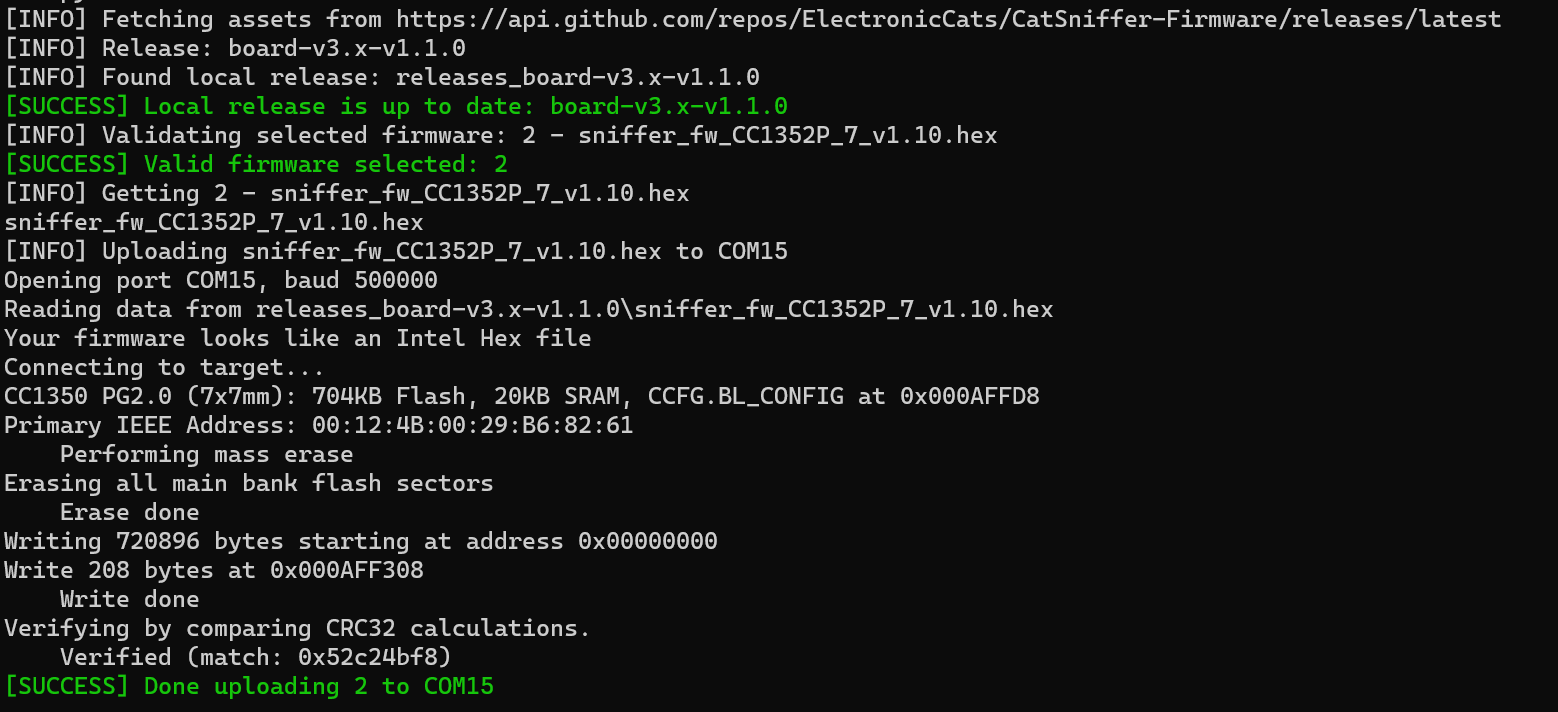
1. Find the firmware id of “sniffer\_fw\_CC1352P\_7\_v1.10.hex” in the list of available firmware releases using the following command:

python catnip\_uploader.py releases



1. Flash the firmware “sniffer\_fw\_CC1352P\_7\_v1.10.hex” using the **load** command as follows. Make sure you replace <SERIAL PORT> with the serial port you obtained in Step 1:

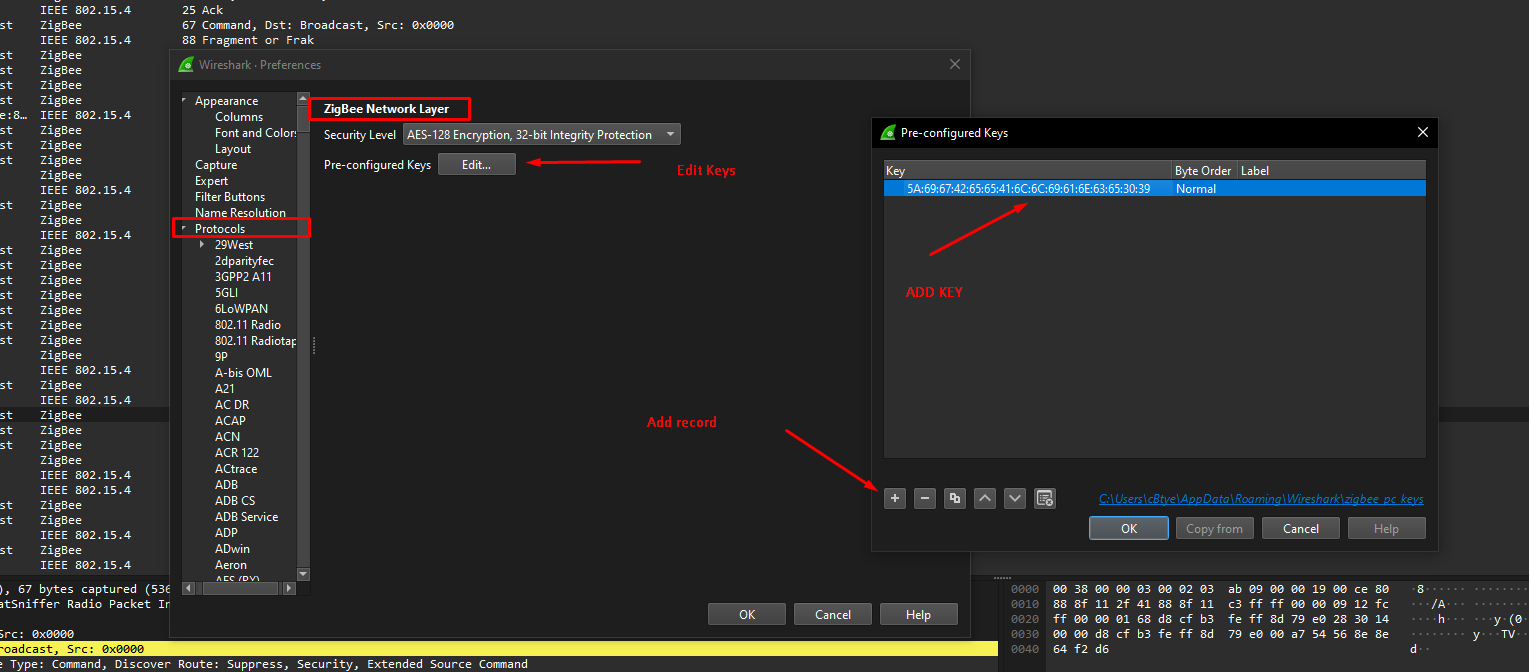
python catnip\_uploader.py load 2 <SERIAL PORT>



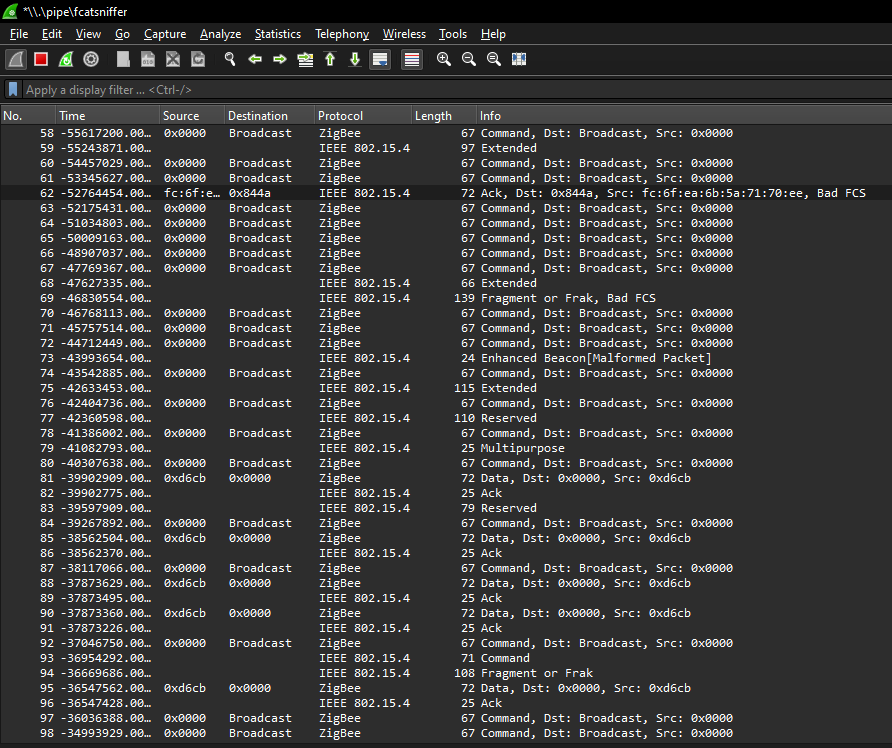
1. Disconnect and re-connect the Catsniffer from the USB port (Or click the RESET button on the Catsniffer)
2. Start sniffing with the CatSniffer and the PyCatSniffer tool using the following command:

python cat\_sniffer.py sniff <SERIAL\_PORT> -ff -ws -phy 1 -ch 25

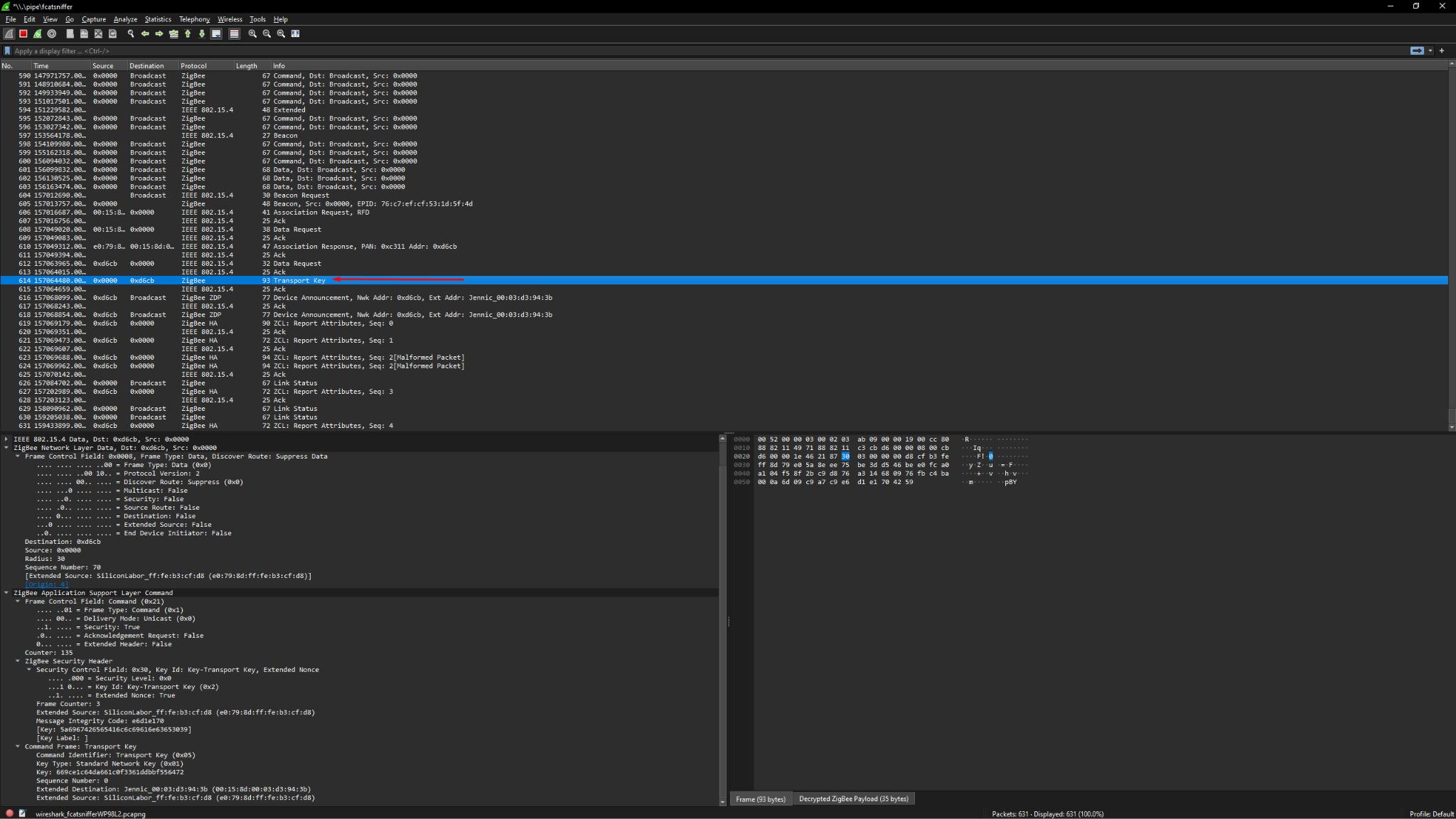
1. Once Wireshark is running, we can view the encrypted communication between the end devices and the Zigbee Coordinator The first step to finding a vulnerable target is to use the Zigbee key from the Zigbee Alliance:
   1. Go to Edit > Preference > Protocols
   2. Zigbee > Pre-configured Keys > Edit…
   3. Add the following new record:
      1. **Key**: 5A:69:67:42:65:65:41:6C:6C:69:61:6E:63:65:30:39
      2. **Byte Order**: Normal
      3. **Label**: Optional
   4. Close the modal dialog with the “OK” Button



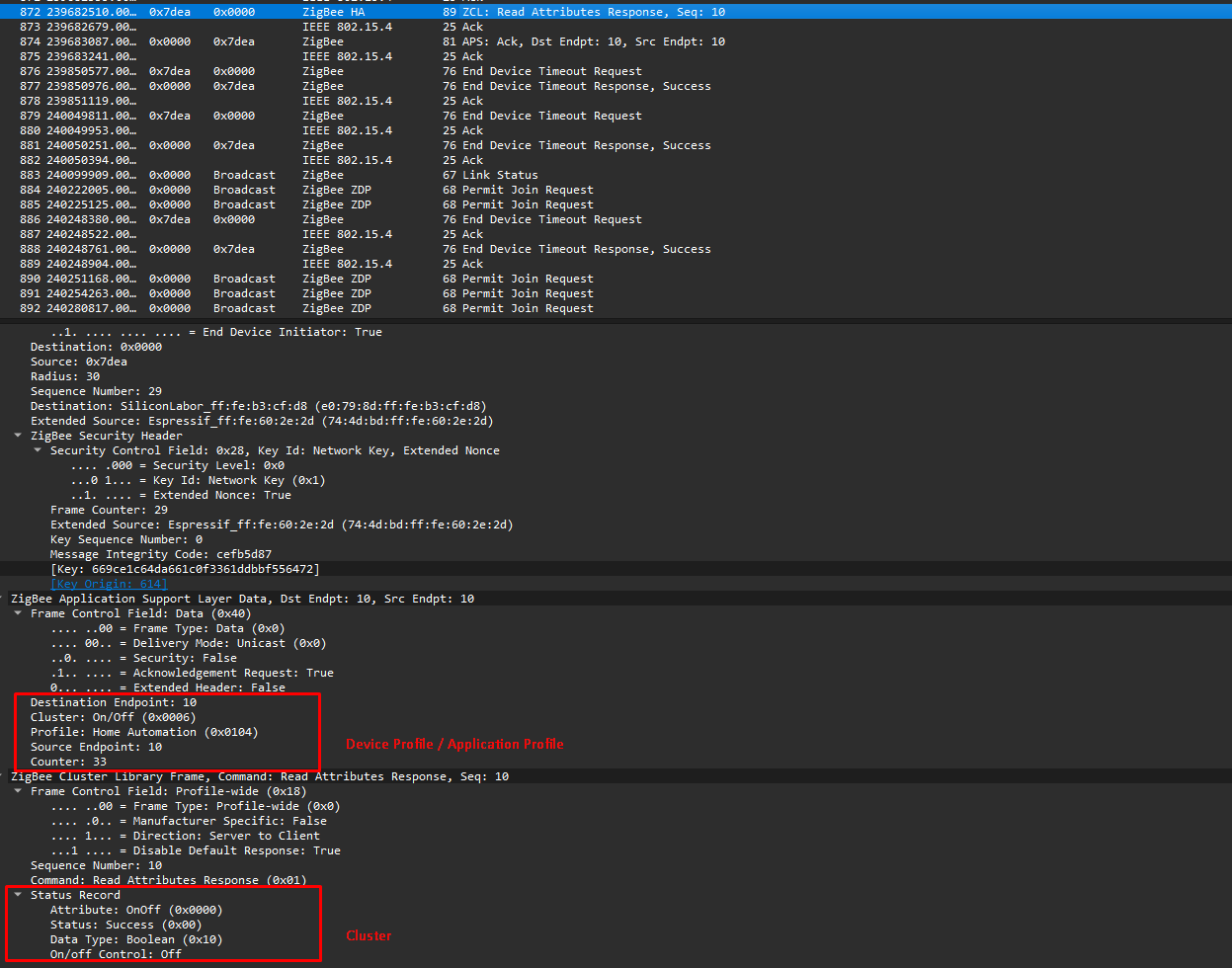
1. When we close the dialog, we can see the first decryption layer of the Zigbee devices. This layer sees the interaction as “Commands” in the **Info** column.



1. When we are sniffing a distributed network, we need to wait until a device tries to reconnect or a new device wants to join the network. When the Zigbee coordinator initiates the pairing process, it sends announcement packets. The end device reads these packets, and the pairing process starts. Wait until you identify a pairing process. You should see “Transport Key” in the information column:



1. Use the key obtained in the “Transport Key” message and add it similarly to Step 9. With this new key, we have complete decryption of the communication between the end device and the coordinator, and we can view the Application/Device Profile and the cluster information



1. Analyze the Zigbee packets again to see the fully decrypted messages and observe the structure of some of the commands.