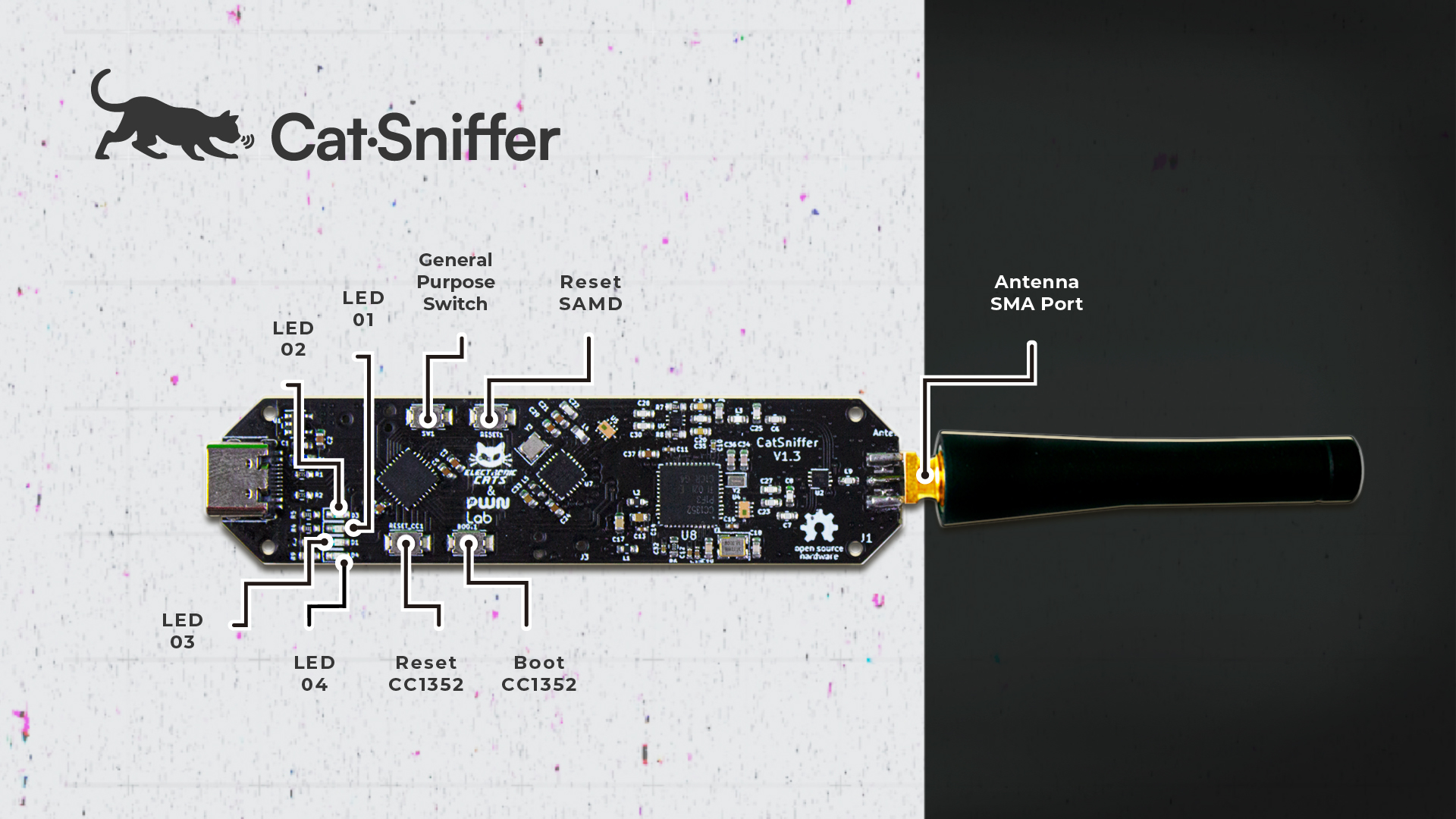
Practical IoT Hacking: Introduction to Multi-Band Hacking with the CatSniffer

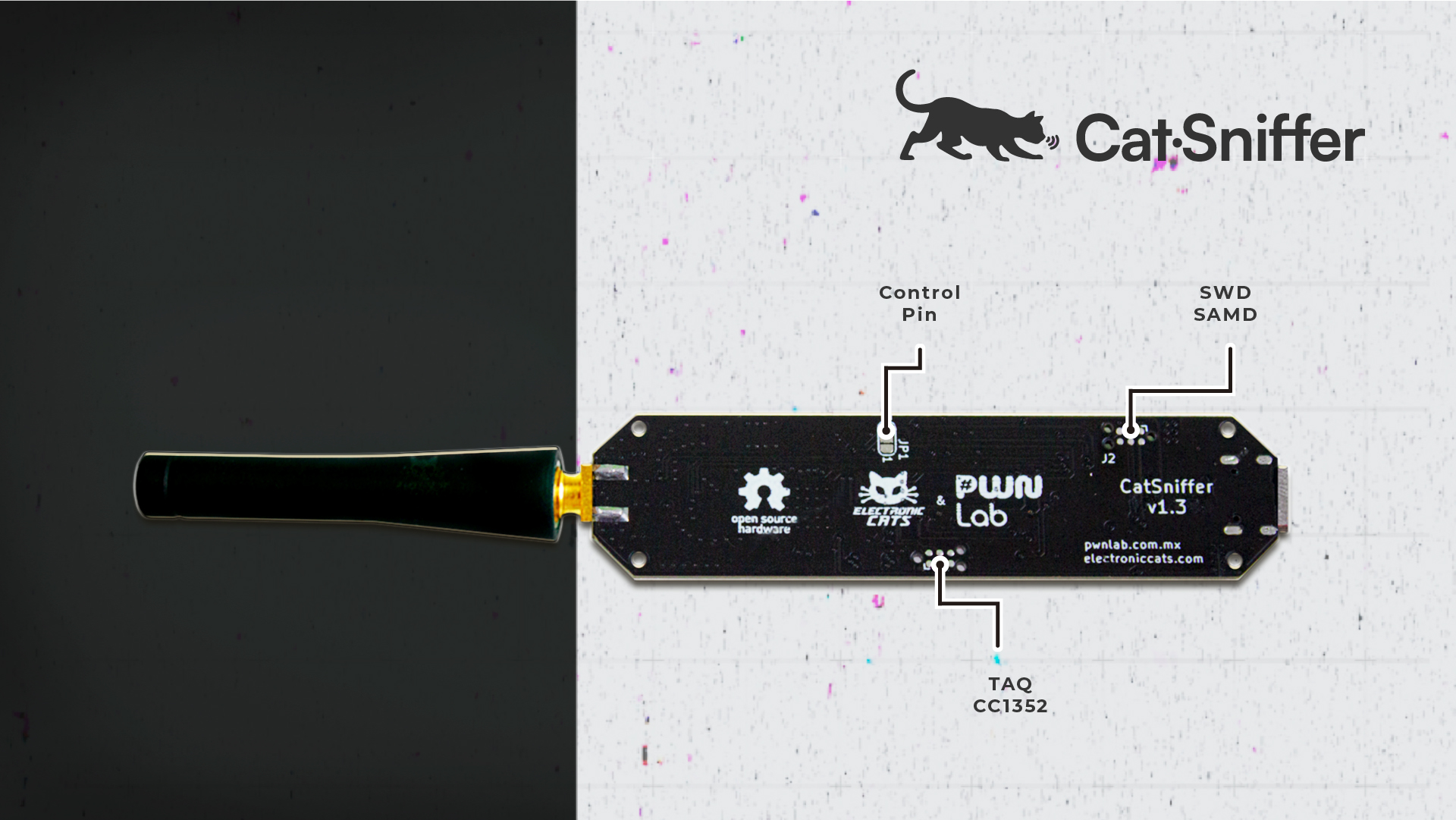
# Introduction

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

* Code Composer Studio from <https://www.ti.com/tool/CCSTUDIO>
* Code sources from <https://github.com/ElectronicCats/CatSniffer/>
* Code sources from <https://github.com/nccgroup/Sniffle>
* Code sources from this lab <https://drive.google.com/drive/u/0/folders/1gAhvl0LL4fa0lKgpV9UGBT3YMNm0QPBm>
* Intro slides are available here, although everything you need is in this booklet: <https://docs.google.com/presentation/d/1xlIV50EbMossOGwUigmQGIvOejEj4H3sAvrZ4WhhDQM/>
* Or just download everything from here:

<ADD LINK HERE>

# 

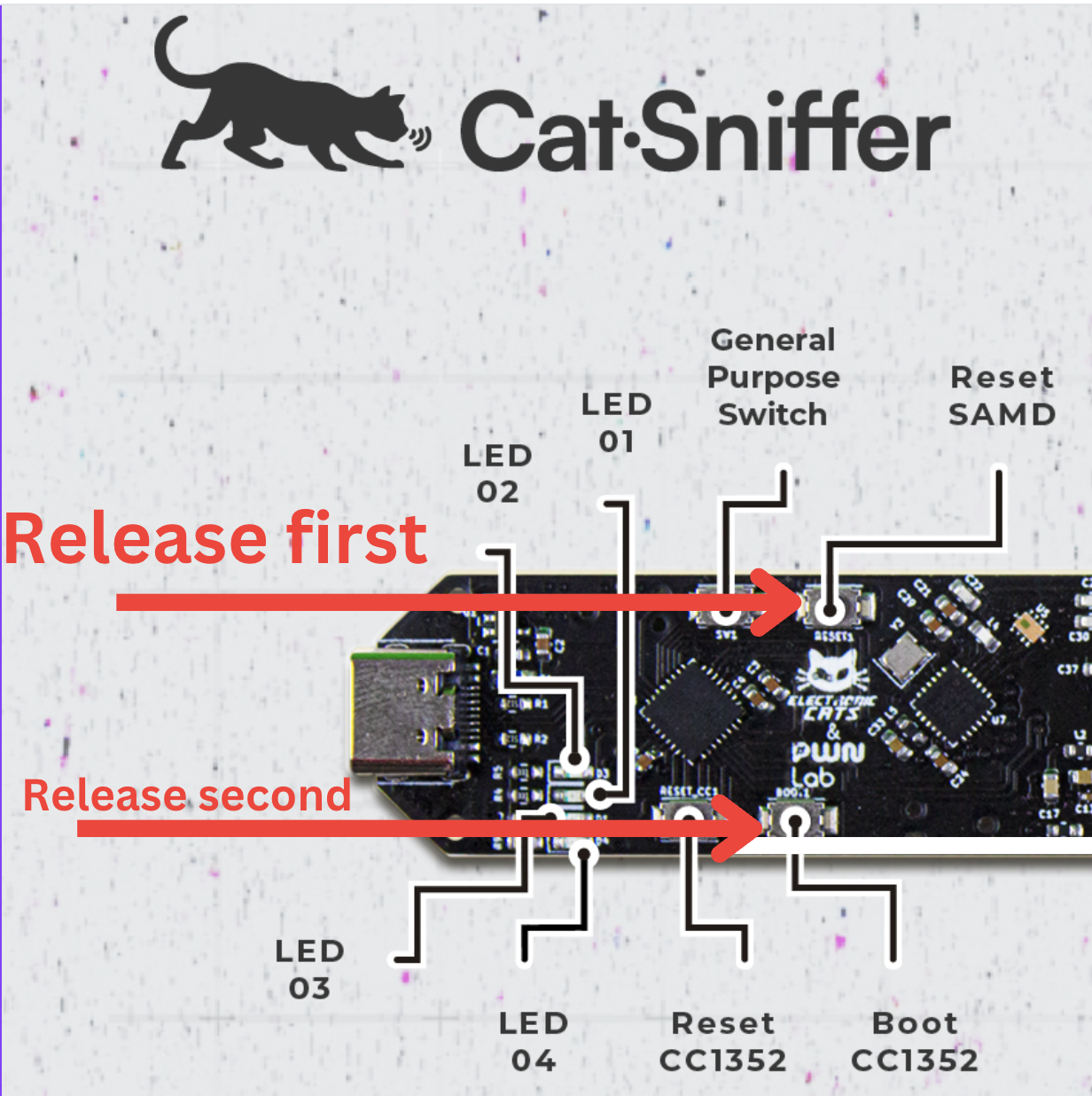
# Flashing firmware from Linux/Mac

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. If you want to read serial information from a terminal, put the CatSniffer back in serial passthrough mode with the correct speed using Arduino IDE. (Optional)
3. Run the command

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

# Putting the CatSniffer in serial passthrough mode

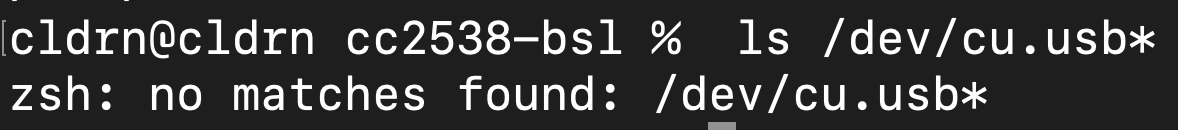
1. Connect the Catsniffer to your computer
2. Press the button: Reset SAMD
3. You should see the LED 03 blink every ~1 sec.

# Determine the serial port used by the Catsniffer

Linux Instructions:

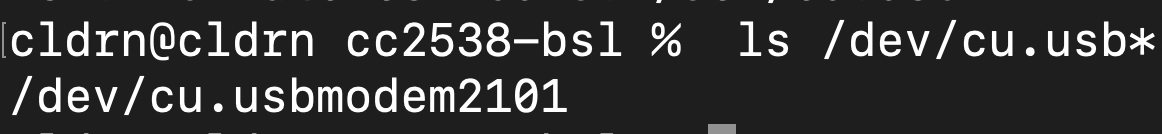
1. Make sure the CatSniffer is disconnected from your computer
2. List the serial devices available before connecting the Catsniffer:

% ls /dev/cu.usb\*



1. Connect your CatSniffer via USB and list the serial devices available again. The new device in the list should be your Catsniffer:

% ls /dev/cu.usb\*

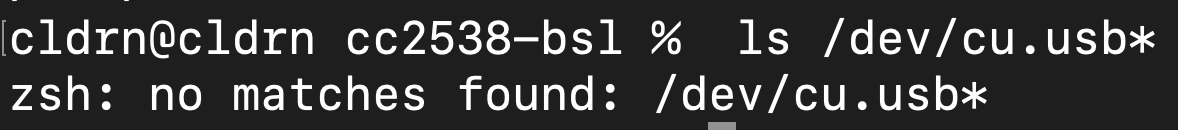


1. In this case, the serial port in your computer is ***/dev/cu.usbmodem2101***. Note that this value may change upon reconnections, and it is affected by the number of serial devices available in your computer.

Mac Instructions:

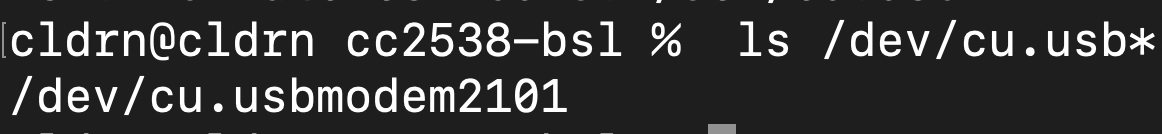
1. Make sure the CatSniffer is disconnected from your computer
2. List the serial devices available before connecting the Catsniffer:

% ls /dev/cu.usb\*



1. Connect your CatSniffer via USB and list the serial devices available again. The new device in the list should be your Catsniffer:

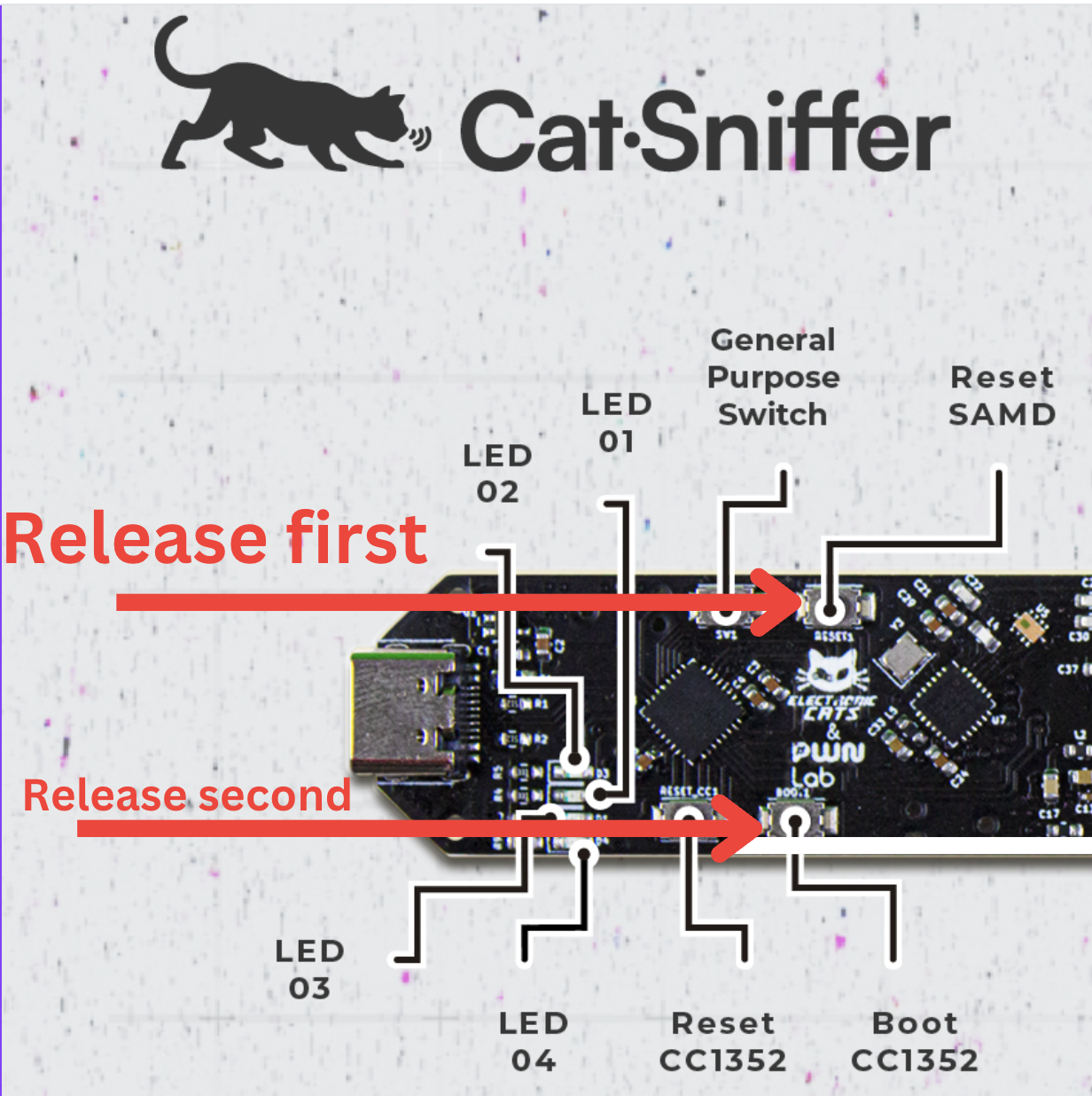
% ls /dev/cu.usb\*



1. In this case, the serial port in your computer is ***/dev/cu.usbmodem2101***. Note that this value may change upon reconnections and is affected by the number of serial devices available on your computer.

# 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.



# Hands-on Lab #1: Sniffing BLE Beacons

## Objectives

* Flash firmware for sniffing Bluetooth Low Energy connections
* Basic usage of tools for analyzing BLE traffic
* Packet structure of different types of BLE packets

## Challenge

Flash firmware for sniffing traffic 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.

## Walk-Through

1. Flash the firmware “Sniffle.hex” into the Cat Sniffer::

python3 cc2538-bsl.py -e -w -v -p <SERIAL PORT> Sniffle.hex

1. Run the Python script called sniff\_receiver.py with the specific serial port number

./sniff\_receiver.py -s <SERIAL PORT>

You should see BLE data on the terminal!

1. To configure Wireshark to read the traffic, we need to copy the Sniffle scripts into your personal EXTCAP folder:
   1. Find the location of your Personal Extcap folder in the Folders tab in the 'About Wireshark' dialog (Unix: *Help* > *About Wireshark* > *Folders* > *Personal Extcap path | Mac: Wireshark > About Wireshark > Folders >* ).
      1. On Unix systems, this folder is typically located at ~/.config/wireshark/extcap.
      2. On Mac systems, this folder is located at

/Users/<user>/.config/wireshark/extcap

1. Copy the following files from the folder python\_cli from the sniffle project directory into your Personal Extcap folder:
2. pcap.py
3. packet\_decoder.py
4. sniffle\_hw.py
5. sniffle\_extcap.py
6. sniffle\_extcap.bat (Windows only)
7. We need to modify the Python script “sniffle\_extcap.py” as an executable. In Unix/Mac, use the following command:

chmod +x ~/.config/wireshark/extcap/sniffle\_extcap.py

1. For Mac users only, you also need to make sure pyserial is installed in the python version used by Xcode:

cd /Applications/Xcode.app/Contents/Developer/Library/Frameworks/Python3.framework/Versions/Current/Resources/Python.app/Contents/MacOS

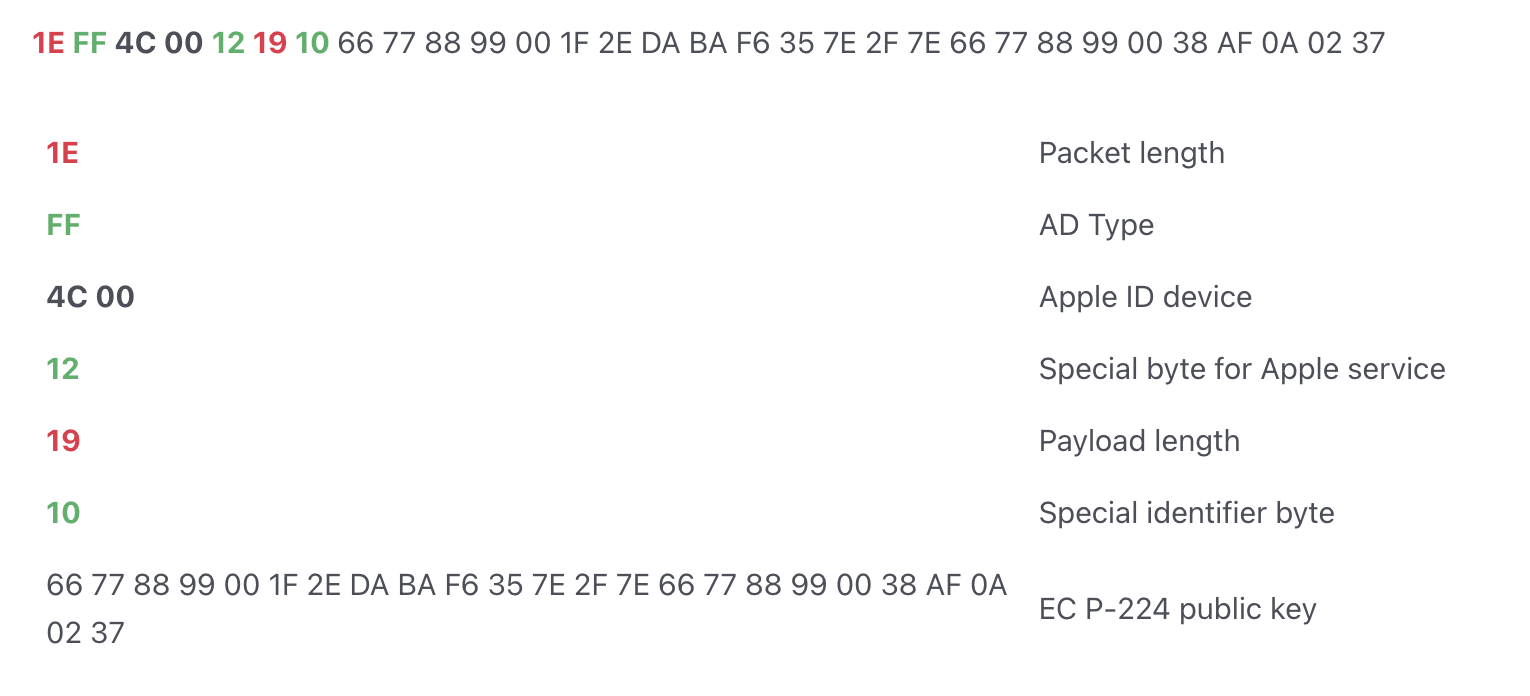
./Python -m ensurepip --default-pip

./Python -m pip install --user pyserial

1. Open Wireshark, and you should see a sniffle BLE interface listed now.
2. Ensure the correct serial port is configured in the interface before starting your capture. Reviewing some of the initial configuration options now.
3. Now, we will add a filter to see the BLE packets from a specific manufacturer.

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

1. Search for the specific packet that can match the next payload



# 

# 

# 

# Hands-on Lab #2: Detecting personal trackers

## Objectives

* Learn about the technologies used in personal trackers
* Learn how to detect personal trackers
* Write custom firmware to detect personal trackers

## Challenge

Finish the implementation of Lab2-Detecting-Trackers and detect all personal trackers around you with the Catsniffer.

## Walk-Through

1. Import the project Lab2-Detecting-Trackers into Code Composer Studio

(File -> Import).

If you see errors related to missing references, go to Project -> Properties -> Resource -> Linked Resources and update their location.

1. Once the project is loaded in Code Composer Studio, get familiar with the folder structure. Browse around the code base. The most important files for our purposes will be in the Application folder.
2. Go to the file Application/Lab2-Detecting-Trackers.c and locate the function AirtagScanner\_processAppMsg. This function will contain the following nested-if statements to filter beacons that are registered and unregistered:

if(pAdvRpt->pData[0]==0x1E && pAdvRpt->pData[2]==0x4c && pAdvRpt->pData[3]==0x00) {

if (pAdvRpt->pData[4] == **<Fill In>** && pAdvRpt->pData[6] == **<Fill In>**){

Display\_printf(dispHandle, 0, 0, "Airtag detected! -> %s Status: Registered and active", Util\_convertBdAddr2Str(pAdvRpt->addr));

}

else if(pAdvRpt->pData[4] == **<Fill In>** && pAdvRpt->pData[6] == **<Fill In>**){

Display\_printf(dispHandle, 0, 0, "Airtag detected! -> %s Status: Unregistered", Util\_convertBdAddr2Str(pAdvRpt->addr));

}

}

1. Fill in the relevant bytes according to the analysis of the packet structure.
2. Build the project by clicking the Build icon on the toolbar
3. Locate the generated .hex file by right-clicking on the project name and then clicking on Show In Local Terminal -> System Explorer. This is the firmware file that you will be flashing to the Catsniffer, copy or move it to your desired location.
4. Flash the firmware into the Catsniffer as described in Flashing firmware from Linux.
5. Press the Reset button RP2040 (Catsniffer 3.x) or Reset SAMD (Catsniffer 2.x - 1.x)
6. We need to read the serial data to read the output generated by the Catsniffer. In Unix/Mac you may use the utility ‘screen’ installed by default in most systems:

$screen <device> <baud rate>

To use the serial port /dev/ttyACM0 and baud rate of 921600 (default baud rate), use the following command:

$screen /dev/ttyACM0 921600

1. You should see a message for every personal tracker detected!

# Hands-on Lab #3: Spoofing personal trackers

## Challenge

Finish the implementation Lab3-Spoofing-Trackers and spoof personal trackers with the Catsniffer

## Objectives

* Learn how to spoof personal trackers
* Learn to craft custom BLE packets
* Write firmware to communicate with devices over BLE

## Walk-Through

1. Import the project Lab2-Detecting-Trackers into Code Composer Studio (File -> Import).

If there are errors related to missing references, go to Project -> Properties -> Resource -> Linked Resources and update their location.

1. Once the project is loaded in Code Composer Studio, get familiar with the folder structure. Browse around the code base. The most important files for our purposes will be in the Application folder.
2. Go to the file Application/Lab3-Spoofing-Trackers.c and locate the function AirtagSpoofer\_processGapMessage. Inside this function, you will find the C array airtag\_adv\_data. This is the structure containing the data used in the advertisement packet.
3. Update the content of this structure to match the content of an advertisement packet.
4. Build the project by clicking on the Build icon on the toolbar
5. Locate the generated .hex file by right-clicking on the project name and then clicking on Show In Local Terminal -> System Explorer. This is the firmware file you will be flashing to the Catsniffer, copy or move it to your desired location.
6. Flash the firmware into the Catsniffer as described in Flashing firmware from Linux.
7. Press the Reset button RP2040 (Catsniffer 3.x) or Reset SAMD (Catsniffer 2.x - 1.x)
8. We need to read the serial data to read the output generated by the Catsniffer. In Unix/Mac you may use the utility ‘screen’ installed by default in most systems:

$screen <device> <baud rate>

To use the serial port /dev/ttyACM0 and baud rate of 921600 (default baud rate), use the following command:

$screen /dev/ttyACM0 921600

1. Use the app Tracker Detect (Android/iOS). Every time you press the reset button, you should see a new personal tracker in the list!