LoRaWAN/Bluetooth Mesh Network Square Kilometre Array

This system will use **LoRaWAN-enabled Raspberry Pi nodes** deployed across a **1 km² area**, with each node featuring a **Bluetooth Low Energy (BLE) scanner**. The goal is to **detect a specific MAC address** and **relay an alert through the mesh network to a central node**.

System Overview

1. **LoRaWAN Nodes (Raspberry Pi-based)**
    - Each node has a **LoRa transceiver (SX1276/SX1262-based)**
    - Integrated **Bluetooth Low Energy (BLE) scanner** constantly scans for a target MAC address
    - On detection, the node **sends an alert** via LoRa to the central node
    - Nodes forward messages from neighboring nodes (mesh networking)
2. **Central Node (Raspberry Pi Gateway)**
    - Acts as a **LoRaWAN gateway**
    - Receives alerts and forwards them to a server (via Wi-Fi/Ethernet or cellular)
    - Could include a **LoRaWAN Network Server (LNS)** such as ChirpStack

---

Network Topology

- **Mesh LoRaWAN nodes** deployed at **strategic points** to ensure coverage
- Nodes communicate with **neighboring nodes** to relay alerts to the **central gateway**
- Central gateway **processes and logs alerts** (e.g., alerts stored in a database or sent as an SMS/email notification)
- **Fallback Mechanism**: If a node cannot reach the gateway, it **forwards the message via other nodes**

System Architecture

1. Hardware Requirements

Component	Description
Raspberry Pi (Zero 2W, 3B+, 4B)	Main processing unit for nodes & gateway
LoRa Transceiver (RAK811, SX1276, SX1262, RFM95W)	LoRaWAN communication module
BLE Module (Internal or External USB Adapter)	BLE scanner to detect target MAC
Power Supply (Battery/Solar)	Battery-powered for remote deployment
Gateway (Raspberry Pi + LoRa Hat or Commercial LoRaWAN Gateway)	Central node for message processing

---

2. Software Stack

Component	Function
Raspberry Pi OS (Lite)	Lightweight Linux OS for Raspberry Pi
Python + BlueZ (BLE Scanning)	Scans BLE for target MAC addresses
LoRaWAN (Python, PyLoRa, or RAK811 Library)	Sends alerts over LoRaWAN
Mesh Protocol (LoRaWAN Forwarding Script)	Relays messages across nodes
ChirpStack / TTN (Optional)	LoRaWAN Network Server for processing alerts
MQTT / HTTP API	Sends alert messages to the cloud

---

3. Message Flow

1. **BLE Scanner:** Continuously scans for the **target MAC address**.
2. **Detection Event:** If detected, generate an **alert packet**.
3. **LoRa Transmission:** Send the alert packet via **LoRaWAN**.
4. **Mesh Forwarding:** If a node is out of range from the gateway, it **forwards the message** to a closer node.
5. **Gateway Reception:** The central node receives the alert and processes it (e.g., sends an **email, SMS, or database update**).

Implementation Details

1. BLE Scanner Script (Python)

This script runs on each **Raspberry Pi node** and **continuously scans for a target MAC address**.

```
python
CopyEdit
from bluepy.btle import Scanner

TARGET_MAC = "00:11:22:33:44:55"  # Replace with target MAC

def scan_ble():
    scanner = Scanner()
    devices = scanner.scan(10.0)  # Scan for 10 seconds

    for dev in devices:
        if dev.addr.upper() == TARGET_MAC:
            print(f"Target MAC {TARGET_MAC} detected!")
            return True  # Alert must be sent

    return False

if __name__ == "__main__":
    if scan_ble():
        print("Triggering LoRa alert...")
        # Call LoRa transmission function
```

---

2. LoRaWAN Alert Script (Python)

When BLE detects the MAC address, this script **sends an alert via LoRaWAN**.

```
python
CopyEdit
import time
import serial

LORA_PORT = "/dev/serial0"  # Adjust for your LoRa module
TARGET_MAC = "00:11:22:33:44:55"

def send_lora_message(message):
    with serial.Serial(LORA_PORT, baudrate=9600, timeout=5) as lora:
        lora.write(message.encode())
        print(f"Sent LoRa Message: {message}")

if __name__ == "__main__":
    message = f"ALERT: MAC {TARGET_MAC} detected!"
    send_lora_message(message)
```

---

3. Mesh Routing Logic

To relay messages **if the gateway is out of range**, each node listens for incoming LoRa packets and forwards them to other nodes.

```
python
CopyEdit
import serial

LORA_PORT = "/dev/serial0"

def listen_and_forward():
    with serial.Serial(LORA_PORT, baudrate=9600, timeout=5) as lora:
        while True:
            incoming = lora.readline().decode().strip()
            if incoming and "ALERT" in incoming:
                print(f"Received alert: {incoming}")
                # Forward the message
                lora.write(incoming.encode())

if __name__ == "__main__":
    listen_and_forward()
```

- Nodes **listen** for messages.
- If a message is received and the gateway is not in range, **it is forwarded**.

---

4. Central Node (Raspberry Pi Gateway)

- **Receives LoRa alerts**
- **Logs alerts**
- **Sends notifications (via SMS, Email, API, etc.)**

```
python
CopyEdit
import serial
import requests

LORA_PORT = "/dev/serial0"
WEBHOOK_URL = "https://example.com/alert"

def process_alerts():
    with serial.Serial(LORA_PORT, baudrate=9600, timeout=5) as lora:
        while True:
            alert = lora.readline().decode().strip()
            if alert:
                print(f"Received Alert: {alert}")
                requests.post(WEBHOOK_URL, json={"alert": alert})

if __name__ == "__main__":
    process_alerts()
```

Deployment Plan

1. **Deploy Nodes in a Grid Layout**
    - Nodes spaced **200-300 meters apart** in urban settings
    - **500-800 meters apart** in open rural areas
    - Each node should be **elevated** for maximum range
2. **LoRa Range Optimization**
    - Use **high-gain antennas** for better transmission
    - Adjust **spreading factor (SF12)** for maximum distance
3. **Gateway Placement**
    - Place the **central gateway at a high location** (e.g., a tower or rooftop)
    - Ensure a **clear line of sight** to at least **one or two relay nodes**

Expected Performance

- **Coverage Area:** 1 km²
- **Detection Latency:** <5 sec (real-time alert propagation)
- **Battery Life:** 1+ year (with power optimizations & solar options)
- **Security:** AES encryption (optional) for LoRa messages

Potential Enhancements

- **GPS on each node** → For node tracking
- **Encryption for secure transmission**
- **Cloud integration (AWS, Azure, or TTN)**

Final Thoughts

This system provides a **low-power, long-range mesh network** using **LoRaWAN and BLE scanning**. It effectively detects **specific MAC addresses** and **relays alerts to a central node**, ensuring coverage of **1 km²**.