BluePath

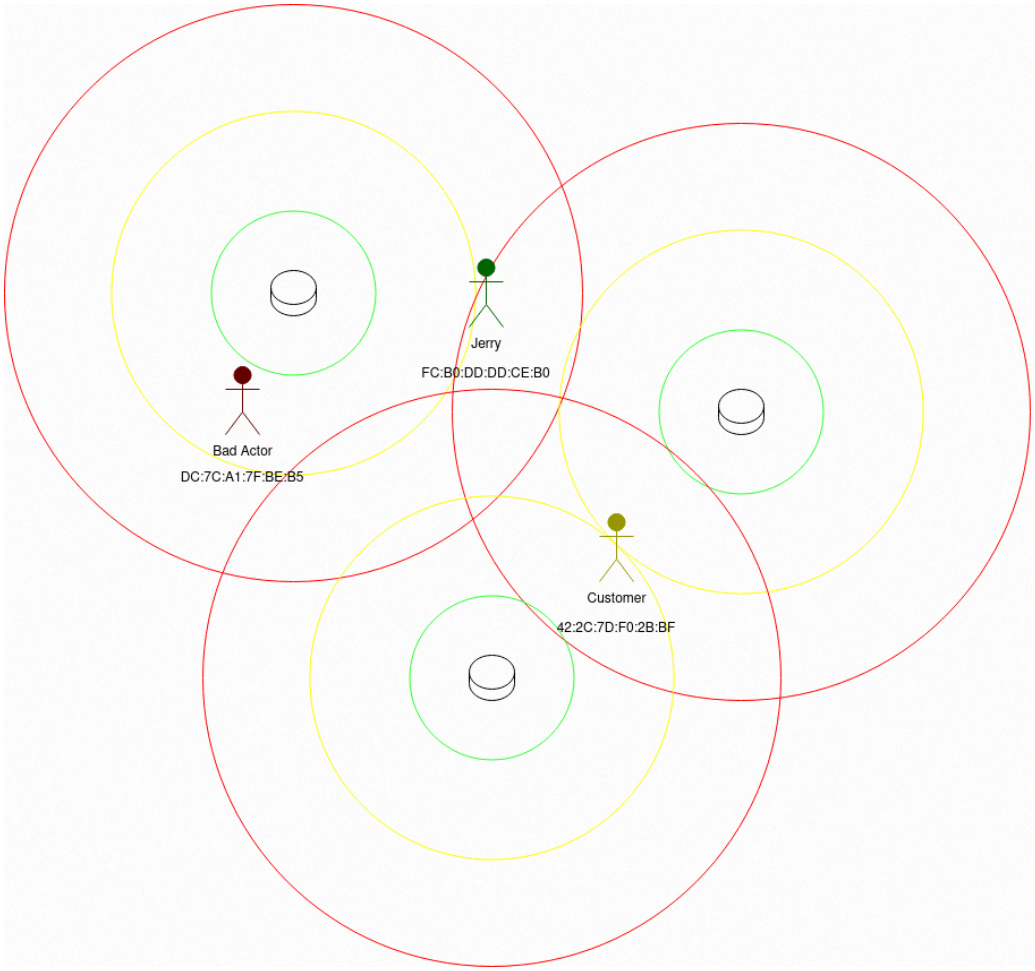
Cameron Woolley - A01347028

Student Background

My name is Cameron Woolley. I'm a student at BCIT pursuing a Bachelor's degree in Networking and Cybersecurity. I have also completed a diploma in Computer Information Systems from KPU. My passion lies in developing projects, ranging from small to large, that involve Internet of Things (IoT) devices, servers, automation tools, and various software programs. I have experience in creating websites with SQL backend and developing local area network programs that can send information to other devices, bots, and scrape data from the internet. Usually developed in C++, or Python. I particularly enjoy working with Wi-Fi and Bluetooth interactions, and my passion lies in how devices connected to the internet can communicate with any other device at near-instant speeds. In the past, I have had the opportunity to help develop a self-hosted photo backup server for customers to use at a cellphone/computer tech repair shop at my work. I enjoy utilizing technology to create projects that can help people and automate tasks.

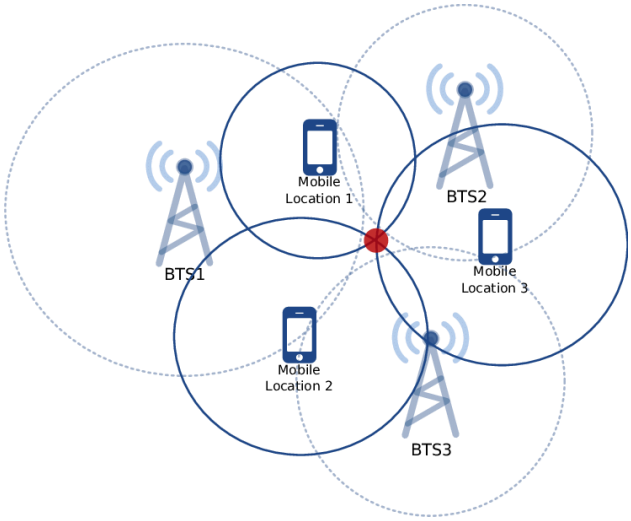
Project Summary

This project aims to be a Bluetooth security and tracker system. Using the bluepath will be able to scan for nearby bluetooth signal regardless if they are in discoverable mode or not with one of the devices you can detect how many device are in an area and tag each devices MAC address to a specific person or device name allowing you to see if unauthorize devices/people are in a location. Utilizing multiple of these devices will allow you to additionally locate the device triangulating the signal strength and if they are in range of a Bluepath device. All of this data will be presented in a simple dashboard with a map of devices with their tag and location through the website interface. Placing and modifying the location of each device with a drag and drop configuration



Problem Statement and Background

Ninety percent of all Americans own a smartphone (Pew Research Center, 2024)[2], and this number is expected to increase as more aspects of daily life become digitized. Most smartphones utilize two familiar technologies: WiFi and Bluetooth. Bluetooth, in particular, is integral to many short-range applications, including wireless audio devices, keyboards, mice, smart home systems, fitness trackers, and smartwatches. Bluepath seeks to capitalize on the ubiquitous nature of Bluetooth to enhance security by device tracking. While Bluetooth technology is primarily recognized for its consumer use, it has significant potential in security and monitoring systems. Research into Bluetooth-based locating and tracking systems, such as the Bluetooth low energy (BLE) Beacon system (Opoku & S. K, 2012)[3], highlights its application in indoor tracking, offering a low-cost and efficient alternative to GPS. Similarly, security papers like the Tracking Anonymized Bluetooth Devices (Becker et al., 2019)[1] demonstrate how Bluetooth signals can be detected and logged even in non-discoverable mode. Even mentioning how you can “identify tokens from the pay-loads of advertising messages for tracking purposes”. Bluepath builds on these concepts by integrating device detection, identification, and location triangulation into a centralized dashboard, offering another layer of security that complements the traditional systems like cameras, motion detectors, and network access control.



Inventiveness

Key Innovations:

1. **Real-time Tracking:** Bluepath enhances device monitoring by offering real-time location tracking, providing instant feedback on unauthorized devices' presence and movement. Existing Bluetooth systems often lack this immediacy, as highlighted in studies like in (Oosterlinck. D et al., 2016)[4], which explore bluetooth based tracking of crowds and not individual people but as a heatmap of an area
2. **MAC Address-based Identification:** Bluepath uses MAC addresses to uniquely identify devices, linking them to user-defined names. While this approach is common in network security, Bluepath applies it to physical security, addressing gaps in existing research on combining MAC tracking with geolocation as in (Opoku, S. K, 2012)[3]'s project as emphasis is made on the tracking of devices that have previously been identified and set up.
3. **Simplified Security Interface:** Bluepath's web-based dashboard integrates real-time data, device tagging, and notifications. It combines the location tracking, device identification and simple to configure and use interface not found in other solutions

Complexity

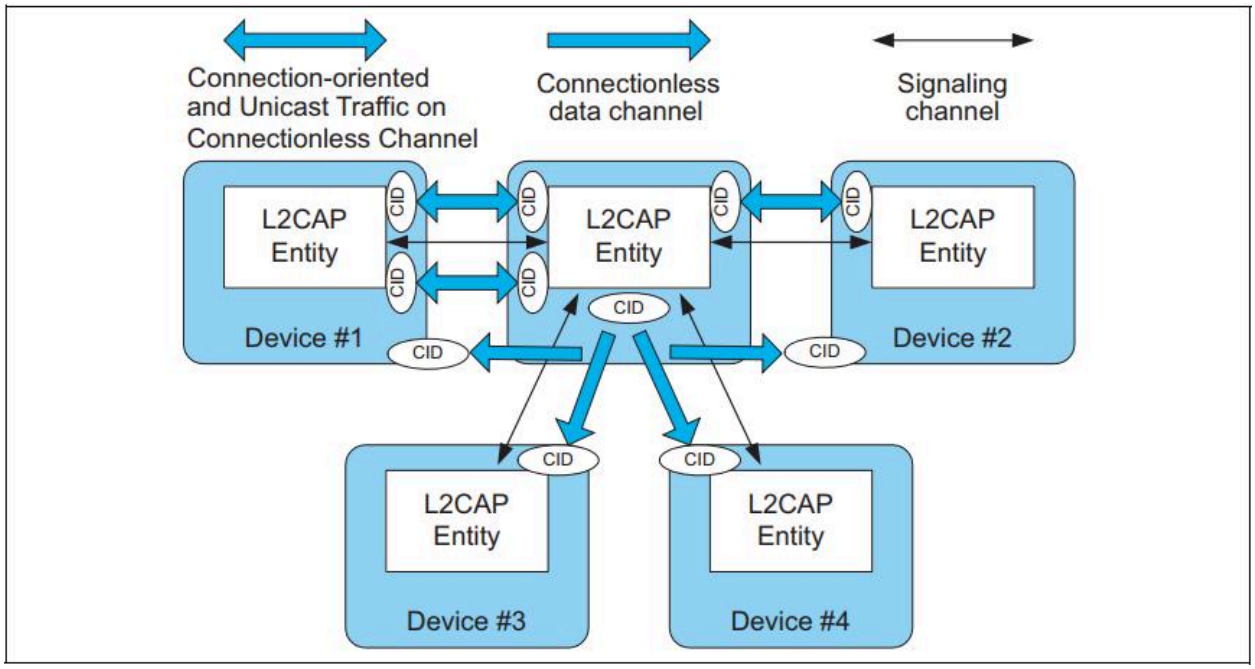
This project involves handling data and implementing a communication standard not fully covered in the BCIT program, with the most complex aspect being device triangulation. Bluepath utilizes BLE protocols, such as GATT for communication and L2CAP for data transport, to track devices by measuring their RSSI values from multiple nodes. Triangulation requires precise modeling and setup to account for factors like walls and signal interference, which can distort RSSI readings. The system must aggregate and synchronize data from all nodes to accurately calculate device locations in real time. This data is then processed and visualized on the server dashboard. Achieving this requires efficient transport and visualization to handle real-time data processing and ensure scalability as the number of devices and nodes increases

Technical Challenges

This project involves dealing with two technical challenges that I have not worked on before. The first is hardware communication, and the second is the Bluetooth protocol. I will need to work with microcontrollers or mini PCs, such as an ESP32 or Raspberry Pi, to build the network. This requires research into how they can be programmed and set up to run the required programs automatically, with failsafes in case of errors. Since Bluepath uses Bluetooth, a comprehensive knowledge of the Bluetooth protocol and how it communicates is required. I will need to develop libraries and algorithms to capture the MAC address and identifiers of each device on this unfamiliar architecture. Reading into (Bluetooth for Programmers, 2005) for information on how programming bluetooth application should be done

Methodology and Design

The design of the Bluepath system involves a multi-component architecture, utilizing networking protocols, server-side technologies, and client communication. The primary components of the system include a User Datagram Protocol (UDP) server for receiving log data, a Flask-based web server for real-time log visualization, and the pydbus library for handling the bluetooth communication between various system components. Blow is a diagram showing directly how a bluetooth device communicates with L2CAP which is directly seen by Bluepath and used in the capturing of data



Software Design Methodology

The system follows a modular, client-server architecture, where the server collects, processes, and stores log data, while client devices send log data via UDP packets. The client devices operate as independent nodes that periodically send data to the server. The server processes incoming data asynchronously, ensuring smooth real-time operation.

UDP Server

The server side is initialized by the udp_server function. This function creates a UDP socket and binds it to a specific IP address and port, where it listens for incoming UDP packets containing log data from client devices. UDP is used for its speed at communicating the information at real time as new information is sent every 3 seconds dropping one packet isn't worth resending. When a packet is received, the server decodes the data, extracts the sender's IP address, and forwards the log data to the Flask web server via an HTTP POST request, allowing the server to handle multiple incoming log data streams concurrently.

Flask Web Server

The Flask web server, running on port 8080, provides a simple RESTful API to handle incoming POST requests from the UDP server. The /receive_log endpoint processes the incoming JSON payload, which includes the IP address of the sender and the log message. The server stores the logs in a dictionary indexed by the IP address of each sender, ensuring that logs are grouped and accessible by their corresponding device.

pydbus Library

pydbus is used to facilitate communication with **D-Bus**-based services for managing inter-process communication (IPC). The library is integrated to monitor system events, track available devices, and ensure synchronization between components of the system. This approach provides a standardized method for interacting with system-level services and allows seamless device integration.

Concurrency and Multi-threading

To ensure the system operates efficiently, the UDP server runs in a separate thread. This allows the UDP server to continue receiving data from clients without blocking the Flask web server's operations. The Flask web server runs concurrently, providing real-time access to logs via a web interface, which can display the logs either as a map or list, depending on user preferences.

Real-time Display

The web interface serves as a dashboard for viewing signals from connected devices. It can be displayed in real time, with data visualizations showing the logs either as a list or a map of devices. The server's Flask framework, with periodic updates being pushed to the client browser.

Technologies Used

- **UDP**: For efficient, lightweight communication between client devices and the server to transmit log data.
- **Flask**: A Python-based micro web framework to provide a RESTful API for receiving log data and displaying it in real-time.
- **pydbus**: For inter-process communication (IPC) with system-level services, ensuring synchronization between components and managing device events.
- **HTML/CSS/JavaScript**: For the frontend, used to build the user interface for displaying logs on a map or in a list format.
- **Pycrptodome**: For RSA encryption for secure node to server communications

Test Plan

Hardware Communication Testing

Power on the nodes and server, then connect each node and verify the connection status on the server. Configure nodes to send periodic test signals, such as dummy data or pings, and log the data received by the server, including timestamps and signal strength. Send acknowledgment signals from the server to the nodes and confirm their receipt. Simulate disconnections by powering off a node, waiting 10 seconds, and powering it back on to check automatic reconnection and proper logging of the interruption. Incrementally add nodes to test load handling, monitoring metrics like latency and throughput. Finally, place nodes at different distances (e.g., 1m, 5m, 10m) to measure signal strength and confirm stable communication across the operational range.

Bluetooth Protocol Testing

Power on the devices and configure them to use Bluetooth Low Energy (BLE). Test Bluetooth profiles like GATT and L2CAP by scanning for nearby devices and verifying detection on the server. Check if the system correctly parses Bluetooth signals by capturing and logging essential device information such as MAC addresses and signal strength. Simulate varying Bluetooth environments (e.g., multiple devices in range) to test the system's ability to handle different profiles and data parsing. Ensure the system can detect and display devices accurately, even in challenging conditions, such as interference or overlapping signals.

Real-time Tracking Testing

Place multiple Bluetooth nodes around the area and configure them to capture real-time the RSSI values from nearby devices. As devices move within the area, monitor how quickly and accurately the system updates their positions on the user interface. Test the triangulation algorithms by moving devices through different areas and verifying the accuracy of the reported location. Simulate various movement speeds and directions to test minimal latency and consistent tracking. Test the system under different scenarios, such as signal interference with many devices or varying distances

MAC Address-based Identification Testing

Add devices with known MAC addresses to the system and configure them with predefined names or tags. Monitor the system's ability to correctly identify each device by its unique MAC address and ensure it registers the device accurately. Test the system by introducing unauthorized devices with unknown MAC addresses and verify that the system flags them as unauthorized or unknown. Simulate various scenarios, such as adding new authorized devices or modifying existing device information, to ensure the system updates and maintains correct identification. Finally, verify that the system generates alerts for unauthorized devices detected in the secured area.

Security and Privacy Testing

Implement RSA encryption using the pycryptodome library to secure data transmission between devices and the server. Verify that data is securely encrypted using RSA during communication. To test this, generate a key pair (public and private) using pycryptodome, encrypt sample data with the public key, and decrypt it with the private key to confirm correct encryption and decryption processes. Test authentication protocols by attempting to access the web interface with both valid and invalid credentials, ensuring that only authorized users can interact with the system. Verify that authorization protocols restrict access to sensitive data based on user roles and permissions. Simulate potential security threats, such as man-in-the-middle attacks, to confirm that no data is leaked.

Scope and Depth

Scope:

The scope of the Bluepath project is to develop a Bluetooth security and tracking system capable of scanning for nearby Bluetooth signals, including non-discoverable devices, and mapping their locations. The system will tag device MAC addresses to specific individuals or device names and identify unauthorized devices within a designated area. It will leverage multiple devices for triangulation based on signal strength to pinpoint device locations. The project will also include the creation of a dashboard with a map to display device tags and locations, along with a web interface for viewing the collected data in a complete and simple interface.

Depth:

This project's depth is in its ability to collect and visualize Bluetooth signals, providing a unique solution for tracking and identifying devices in real-time. By incorporating multiple nodes for triangulation, the system can provide a more accurate location mapping, making it distinct from simpler Bluetooth monitoring tools. The use of a web interface for visualization and drag-and-drop configuration offers flexibility and user interaction, enhancing the usability of the system. Moreover, the security aspects related to unauthorized device detection and the potential for real-time application contribute to the project's depth, highlighting both its technical and practical implications.

Development Schedule and Milestones

Milestone 1: Initial Setup and Design

Milestone 1 focuses on establishing the hardware and software foundations of the Bluepath system. This involves setting up nodes and server PCs as nodes for Bluetooth detection and data transmission. The node must support Bluetooth Low Energy (BLE) protocols for device detection. Simultaneously, the software setup includes configuring a UDP server to receive log data and a Flask web server to display the logs. Initial testing will focus on verifying the reliability of the bluetooth data received

Milestone 2: Bluetooth Integration and MAC Address-based Identification

In Milestone 2, Bluetooth identification and map functionality is integrated into the system, enabling the detection of nearby devices using BLE protocols. The system will scan for devices, extract their MAC addresses, and use them as unique identifiers for tracking. The challenge in this milestone is ensuring accurate device identification. Testing will verify the reliability of Bluetooth detection, secure MAC address handling, and consistent tracking of devices across the area.

Milestone 3: Real-time Tracking and Compatibility Testing

Milestone 3 is centered on real-time tracking and compatibility testing. The system will measure the RSSI of Bluetooth signals from multiple nodes to triangulate the location of devices. The goal is to display device locations more accurately on a web interface in real time. Compatibility testing will ensure the system works across various Bluetooth-enabled devices and operating systems. The challenge lies in improving the accuracy of triangulation and ensuring the system functions across diverse environments and device configurations. Also the start of non-discoverable device identification will be implemented

Milestone 4: Security and Privacy Testing

Milestone 4 focuses on ensuring the security and privacy of the Bluepath system. This involves implementing encryption protocols for secure data transmission and storage, as well as authentication mechanisms to prevent unauthorized access. Privacy testing will ensure that MAC addresses and other sensitive information are securely handled. It also includes the improvement of the web interface allowing more options and simplified but still intuitive UI to use for any user.

Milestone 5: Scalability and Performance Optimization Testing

Milestone 5 will test the system’s scalability and performance. This involves simulating an increasing number of devices to see how well the system handles growing data loads. Performance optimization will focus on reducing latency in data processing and real-time visualization. Bottlenecks will be identified, and an optimization will be implemented to maintain normal performance if more nodes and devices are added.

Week	Phase	Due Dates	Dates
1	Initial Setup **(Old Proposal)**	**Proposal Draft 1 Due**	Sept 17, 2024
1	Initial Setup **(Old Proposal)**		Sept 17 - Sept 23, 2024
2	Initial Setup **(Old Proposal)**		Sept 24 - Sept 30, 2024
3	Design and Prototyping **(Old Proposal)**		Oct 1 - Oct 7, 2024
4	Design and Prototyping **(Old Proposal)**	**Prototype Due**	Oct 8, 2024
5-6	New Proposal work	**Proposal Draft 2 Due**	Oct 15 - Oct 28, 2024
7	ReCreate Prototype		Oct 29 - Nov 4, 2024
8	Bluetooth Integration		Nov 5 - Nov 11, 2024
8		**Milestone 1 Due**	Nov 5, 2024
9-10	Data Processing and MAC IDing		Nov 12 - Nov 25, 2024
10	Proposal Work	**Proposal Draft 3 Due**	Nov 19, 2024
11	Inter-Device Communication		Nov 26 - Dec 2, 2024
12	Data Visualization UI		Dec 3 - Dec 9, 2024
13	Final Proposal Work	**Final Proposal Due**	Dec 3, 2024
14		**Milestone 2 Due**	Dec 10, 2024

References

1. Becker, J. K., Li, D., & Starobinski, D. (2019). Tracking anonymized Bluetooth devices. *Proceedings on Privacy Enhancing Technologies*. https://petsymposium.org/popets/2019/popets-2019-0036.php
2. Pew Research Center. (2024, January 31). Mobile fact sheet. *Pew Research Center*. https://www.pewresearch.org/internet/fact-sheet/mobile/
3. Opoku, S. K. (2012). An indoor tracking system based on Bluetooth. *arXiv*. https://arxiv.org/pdf/1209.3053
4. Oosterlinck, D., Benoit, D. F., Baecke, P., & Van De Weghe, N. (2016). Bluetooth tracking of humans in an indoor environment: An application to shopping mall visits. *Applied Geography*, 77, 89-98. https://doi.org/10.1016/j.apgeog.2016.11.005
5. IEEE Xplore. (n.d.-c). An overview of the Bluetooth wireless technology. *IEEE Journals & Magazine*. https://ieeexplore.ieee.org/document/968817/
6. Gopikrishnan, K. (2020, October 20). The Bluetooth standard - A simple guide to the protocol for beginners. *Technobyte*. https://technobyte.org/bluetooth-standard-beginners-explained-guide/#Inquiry
7. Huang, A., & Rudolph, L. (n.d.). Bluetooth for programmers. *MIT CSAIL*. https://people.csail.mit.edu/rudolph/Teaching/Articles/BTBook.pdf

Change Log

Date	Version	Addition
9/10/2024	1	First draft for Student Background, Project Summary, Problem Statement done
4/11/2024	1.1	Finished draft for all section accept Testing and Schedule
19/11/2024	1.2	Improved Project summary, Problem statement and background, Inventiveness and Complexity
6/10/2024	1.3	Finalized document by improving, Test plan, Scope and Depth and Technical Challenges