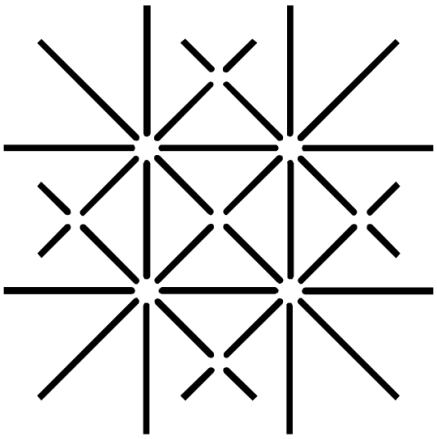
Group 9: Location Verification - Reliable Transmission of Location Data using GPS and RFID technology

Rasmus Willi
Lukas Vogel
Nakarin Srijumrat

July 18, 2021



UNI
BASEL

Introduction to Internet and Security
Spring Semester 2021

Abstract

This report documents our project for the lecture Introduction to Internet and Security, where we would implement a system to verify a users location into a network. We would build a scan station that utilizes RFID and GPS technology to read tags associated with objects and integrate this system into BACnet, a peer-to-peer based network. We would use properties of both, the RFID identification system and the GPS coordinate system to create dictionary-like entries in a feed public to those in the network. The project provided a good balance between hardware and software with the end result being distributed between physical products and a codebase for further extension.

Contents

1 Introduction

For any type of network, having a method to share or access location information of its members is a service of general interest. In society, we have street names and house numbers, in infrastructure databases, we have buildings and room numbers and on the internet, IP-Adresses are used. The means to share information on ones location varies from use-case to use-case, though the most prevalent one is through the Internet, where members can easily connect, share, and retrieve data.

What if the Internet was to shut down one day? How would people keep track of goods or broadcast their own location into their communities? Our project realizes a solution for sharing a verified location securely into a network disconnected from the Internet (in our case BACnet) using RFID and GPS technology. Because IDs are used, those can be assigned to network members as well as products, goods and containers.

2 Background

2.1 BACnet and Secure Scuttlebutt

Project BACnet[1] is an implementation of a network free of political and governmental intervention using the Secure Scuttlebutt protocol. In various lectures and seminars held by the Computer Networks Research Group, applications and back-end infrastructures are developed and implemented in order to extend the network capabilities. The BACnet has been built from scratch in the lecture Internet and Security Spring Semester 2020.

Created by Dominic Tarr, the Secure Scuttlebutt (SSB) protocol uses peer-to-peer communication and immutable append-only logs to synchronize and share feeds among network members. Furthermore, messages are signed using SHA256 and the author's private key to ensure authenticity and prevent forging of messages. Because the network using SSB is decentralized, every device has its own copy of the relevant feeds and the means keeping those feeds up-to-date is free for the user to choose (Tarr, 2019).

2.2 RFID Technology

Radio-frequency identification[2] is a versatile system that uses tags and readers to identify objects associated with the stored information. Utilizing electromagnetic fields, a reader (powered by electricity) can request data transmission from a tag in range. The energy is delivered to the tag alongside the electromagnetic interrogation pulse. In many industries, RFID is the standard technology used to track goods during transport on small (warehouses) and large (cross-country transportation) scale. In addition, RFID cards are used for access systems and identification in buildings and corporate environments.

For our project, we are only interested in the Unique Identifier (UID) of a tag located in the header of a RFID data packet. The UID consists of 4 hex-values and represents the cards' serial number and thus, is immutable. Interpreted as an identifier for a person or goods, the UID serves as a reference to the associated entity.

2.3 GPS Technology

The Global Positioning System[3] is an implementation of satellite navigation operated by the United States government. The system uses communication with satellites to provide information on location and time on a three-dimensional sphere. A GPS receiver can get a so-called fix if it has line of sight to four or more (of the in total 32) satellites. GPS coordinates are split into 3 values: longitude (east to west), latitude (north to south) and altitude. Longitude and latitude are expressed in degrees, where the equator serves as the 0° latitude line and the prime meridian (through Greenwich Park) serves as the 0° longitude line.

Our project uses the Arduino Uno as a microcontroller, managing the different modules.

[1] https://github.com/cn-uofbasel/BACnet
[2] https://en.wikipedia.org/wiki/Radio-frequency_identification
[3] https://en.wikipedia.org/wiki/Global_Positioning_System

3 Hardware Implementation

Our hardware implementation uses the Arduino Uno, a GPS module (NEO-6M) and a RFID reader module (RC522). In addition, we added a LCD screen to provide visual feedback from the system to the user.

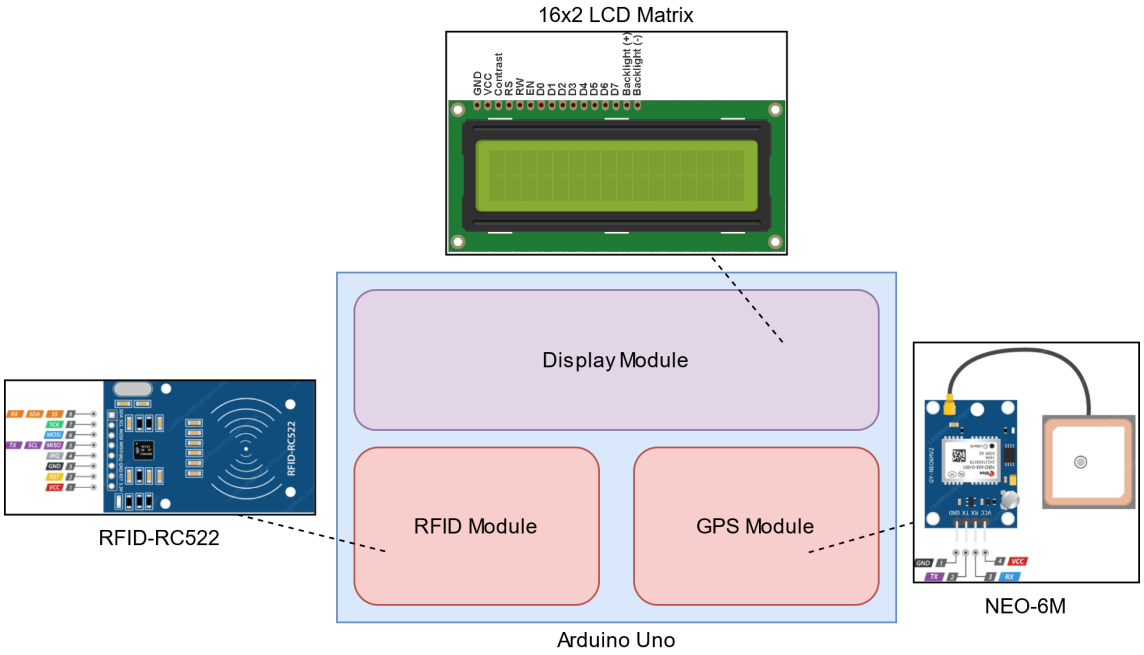


Figure 1: Hardware Concept

The Arduino Uno handles input of both, RFID reader and GPS module, and displays the current location (if available) until a RFID tag is held against the reader. On contact, the system recognizes the UID from the tag and stores/displays it onto the screen for a brief moment. The system regularly updates the GPS data, which is stored and displayed on the LCD screen with four decimal places[4].

Reliability of the location data is strongly correlated to the quality of the antenna connected to the GPS module and number of visible satellites. Concrete, windows and dense materials can block the already weak GPS radio waves. For that reason, the device may not work indoors. Because RFID and GPS use different frequency ranges, no interference issues occur at local scale. To check the reception of the antenna, it is possible to directly access the raw GPS data using the u-center software by u-blox[5] (exclusively for Windows) designed for use with modules of the NEO model.

Because the RFID module and the GPS module both require 3.3V voltage to operate, some soldering or a small breadboard is needed. We decided to go for the latter and mounted all parts onto a piece of MDF wood using nuts and bolts (See appendix A). Also, we drilled holes onto the wood board to feed the cables through for proper management and visual effect. The display module has a SDA/SCL interface in order to save pin space on the microcontroller board.

Using the `TinyGPS++`[6] library, it was possible to extract and convert the raw GPS data from the NEO-6M into simplified, human-readable coordinates.

The main goal for the hardware was to store all the relevant data in a usable manner for the software to access. Thus, we opted to write the data into the serial bus of the Arduino Uno and preceded every message with a prefix to define its content - whether the line written was GPS data or a tag's UID.

[4] https://xkcd.com/2170/
[5] https://www.u-blox.com/en/product/u-center
[6] http://arduiniana.org/libraries/tinygpsplus/ and https://github.com/mikalhart/TinyGPSPlus

4 Software Implementation

4.1 Extracting Data from Arduino

The Arduino Uno, running the code explained in the section before, has to be connected to a device able to run Python code. The user can then execute a Python script to directly read the serial of the Arduino Uno after determining its port. This makes it easy to store the relevant data into global variables for further processing. Reading from the serial bus is a continuous process and the variables are updated as the location changes or if an RFID Tag is scanned. A thread is used for this part of the script such that processing and transmission can happen concurrently.

4.2 Transmitting Data to BACnet

BACnet uses the Secure Scuttlebutt protocol (see 2.1) using feeds and keys. Feeds are represented using .pcap-files. Thus, we need a tool to create and append to feeds (using a signature originating from a private key).

Fortunately, other groups of the lecture have already been working on various tools acting as an interface for BACnet. The following tools created by other students are used in our project:

- **logMerge**[7]: A python package providing an API to the BACnet transport layer groups
- **EventCreationTool**[8]: Tool for creating BACnet feeds and events - by the creators of logMerge
- **logSync**[9]: An API simplifying synchronization of two databases

Our script creates a private key and a feed (if none were found) as a .pcap-file in the current working directory and determines the Arduino Uno port. The serial bus of said port is scanned continuously for data. Upon a new entry in the serial bus, the script appends the data in the feed and transmits the new feed into the BACnet or onto another medium (e.g a USB-Stick). This procedure utilizes the previously created private key to sign the new additions to the feed. After 5 entries, the local feed is printed onto the console for visual feedback and on request, the main feed located on the BACnet.

5 Results

The product is working and the local feed is updated when expected. We have built one device acting as a reader for the system using various hardware components. Connected to a device running our .py-script, we are able to communicate and share data with other members of the BACnet.

Regarding hardware, we had initial struggles with the GPS module because of its lacking visual feedback. For example, there was only one LED mounted on the GPS and its function was unambiguous. Thus, we could not tell which part of the GPS was defunct if it didn't work and had to re-solder or replace the module three times throughout the project. All other components worked as expected from the first try.

Regarding software, we spent a lot of time researching about how SSB and BACnet interact with their applications and were in contact with other groups who provided the code-base for various BACnet functionalities.

We prepared also two videos shwoing the usage of our system. The videos are located in the folder labeled 'Demo' in the repository.

[7] https://github.com/cn-uofbasel/BACnet/tree/master/20-fs-ias-lec/groups/04-logMerge/logMerge
[8] https://github.com/cn-uofbasel/BACnet/tree/master/20-fs-ias-lec/groups/04-logMerge/eventCreationTool
[9] https://github.com/cn-uofbasel/BACnet/tree/master/20-fs-ias-lec/groups/12-logSync

6 Conclusion and Review

The project was an overall success, with the product being a system that could verify the location of an entity in the BACnet. We were able to combine hardware and software, integrating them into an already established network and finally, extend said network with our implementation.

That said, conducting the project throughout the COVID-19 crisis was a challenge, but surmountable using Discord[10] as our main communication tool as well as a shared Google Docs worksheet where we would document our progress and keep general notes. Difficulties in the work process could be addressed this way and finding solutions would always include all team members where possible.

Our main takeaway is the insight gained on how large networks work on a lower level and the subsequent appreciation for things that seemed for granted. Also, the project allowed us to use the acquired skills from the lecture hands-on and internalize the ideas that define the Internet how it was intended to be.

7 Future Work

Recommendations for extending the system would include a proper front-end or identifiers for UIDs using a dictionary. Another idea is to define coordinate ranges and link names to set areas. The front-end application would combine visual representation of stored data and general clarity on the data (concrete names instead of UID, location names instead of coordinates, etc.). On a lower level, it is also possible to decrease the amount of redundancy by optimizing the serial outputs in the Arduino code. Our final implementation can be found in the main BACnet repository.

[10] https://discord.com/

References

Tarr, D. (2019). *Secure Scuttlebutt: An Identity-Centric Protocol for Subjective and Decentralized Applications*. ICN '19: 6th ACM Conference on Information-Centric Networking.

Used Libraries

- **Mfrc522**[11]: Arduino RFID Library for MFRC522 (SPI)
  Read/Write a RFID Card or Tag using the ISO/IEC 14443A/MIFARE interface.
  Author and Maintainer: GitHub Community
- **Liquid Crystal I2C**[12]: A library for I2C LCD displays.
  The library allows to control I2C displays with functions extremely similar to LiquidCrystal library.
  Author: Frank de Brabander, Maintainer: Marco Schwartz
- **TinyGPS++**: A compact Arduino NMEA (GPS) parsing library
  Author and Maintainer: Mikal Hart
- **pySerial**[13]: Python Serial Port Extension for Win32, OSX, Linux, BSD, Jython, IronPython
  Author and Maintainer: Chris Liechti
- **EventCreationTool**: A simple tool for creating BACnet feeds and events
  Authors: Joey Zgraggen, Günes Aydin, Nikodem Kernbach
- **LogMerge**: project that provides an API to BACnet transport layer groups for importing and exporting events from the local BACnet database
  Authors: Joey Zgraggen, Günes Aydin, Nikodem Kernbach

A Photographs

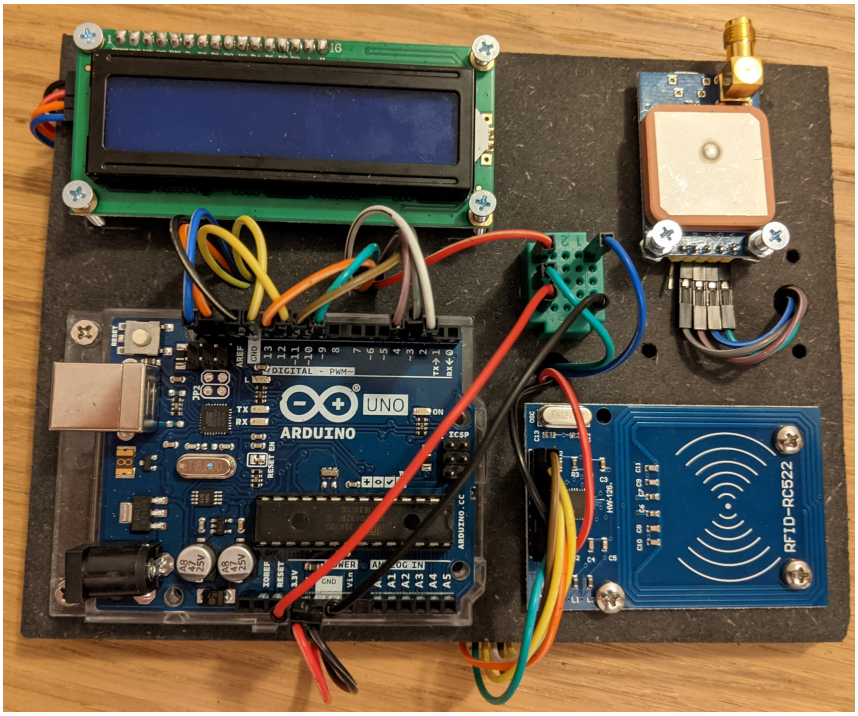


Figure 2: Finished scan station

[11] https://www.arduino.cc/reference/en/libraries/mfrc522/
[12] https://www.arduino.cc/reference/en/libraries/liquidcrystal-i2c/
[13] https://pypi.org/project/pyserial/

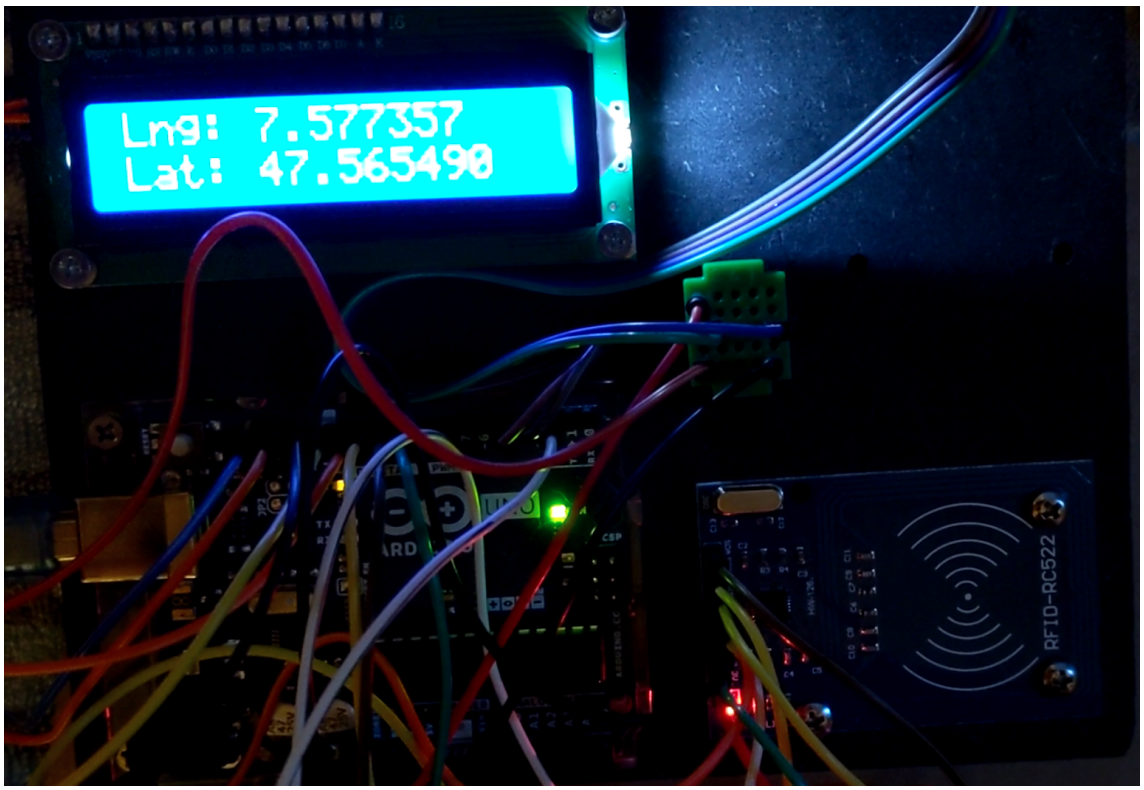


Figure 3: GPS hardware testing

B Division of Labor

Task	**Description**	**Responsible**
Obtaining hardware parts	Research and purchase of required hardware for the project	Nakarin
Assembly of Scanstation	Solder, arrange and attach hardware parts	All
Wiring and communication of/between parts	Implement and display internal exchange/processing of data (.ino script)	Nakarin/Rasmus
Communication with BACnet	Preparation and transmission of data to BACnet and ensuring authenticity of data (.py script)	Lukas/Rasmus
Project report first draft	Draft and structure of project report including graphics and appendix	Nakarin
Project report polish	Refinement and finishing touches to project report	All

C Hardware Cost

	CHF
Arduino Uno	34
GPS Modules	27
RFID Reader	5
Display	14
Desoldering Pump	14
Shipping	36
TOTAL	**130**

D Hardware Wiring

	Module Pin	Arduino Uno Pin
RFID-RC522	3.3V	3.3V
	RST	9
	GND	GND
	IRQ	2
	MISO	12
	MOSI	11
	SCK	13
	SDA	10
16x2 LCD Display	GND	GND
	VCC	5V
	SDA	SDA
	SCL	SCL
NEO-6M	VCC	5V
	RX	3
	TX	4
	GND	GND