Lehigh University

CREG 258: Senior Capstone Project

# Zero Trust Architecture on Internet of Things Networks

| *Student* | *Signature* | *Date* |
|---|---|---|
| John Alexander Carr | _____ | _____ |
| Emma Mirabelli | _____ | _____ |

| *Advisor* | *Signature* | *Date* |
|---|---|---|
| Dr. Mihail Cutitaru | _____ | _____ |

# Contents

## Abstract

This project involved creating a network of Internet of Things (IoT) devices upon which to implement a security framework known as a Zero Trust Architecture (ZTA). This network consisted of two IoT devices and a central hub computer all connected over a WiFi network. The IoT devices acted as clients that communicated with a server using Message Queuing Telemetry Transport (MQTT) guidelines for communication. These devices sent collected pressure and temperature data over the network to server device which then processed the data and stored it in a database. To ensure the security of the network, a ZTA including the following measures was implemented on the network: MQTT communication encrypted with SSL/TLS encryption, database entry encryption, and multi-factor authentication for devices upon each action within the network. A web interface hosted locally on the server displayed the data collected from the IoT devices and the ZTA security statistics.

# Introduction

IoT devices have become increasingly common in many households, taking the form of temperature sensors, smart locks, cameras, light switches, and many other products. Each of these devices requires collecting and relaying sensitive data across a network to be saved and processed in a central hub. Because this information is traveling over a network, it becomes exposed to attacks in which sensitive information can be stolen. To combat these attacks, a thorough security architecture is extremely important in preserving the credibility and dependability of IoT devices.

Because there is a wide range of IoT devices with just as many unique manufacturers, widely accepted security standards have not been established. This lack of standards creates a major security problem, as there are no baseline requirements for data transfers between IoT devices which makes networks vulnerable to many cyber-attacks.

The best way to protect against these cyber-attacks is to implement a security framework across the network. ZTA is a security approach that provides utmost protection by removing the assumption of trust given to devices and, instead, evaluating requests and communications continually in real time. This approach requires devices to verify their identity for all communication and adheres to a zero trust policy, which only gives devices access to precisely what they need and nothing more.

# Problem Statement

Due to the wide range of IoT device types and multitudes of data being relayed over a network, there is no standard level of security for IoT devices. A ZTA is and advanced security tactic that removes the assumption of trust given to devices and continually verifies and authenticates devices in a network. Implementing ZTA on a network of IoT devices increases the security of the network and ensures safe data transfer between devices.

# Design & Development

## Hardware

The hardware used in this project consisted of a Raspberry Pi 4 Model B and Keysight Technologies U3814A board. The Raspberry Pi acted as a central computing hub in the network. It hosted the server, MQTT broker, and web interface. This is where all of the IoT data being collected over the network was relayed and stored. The Raspberry Pi can be seen below.

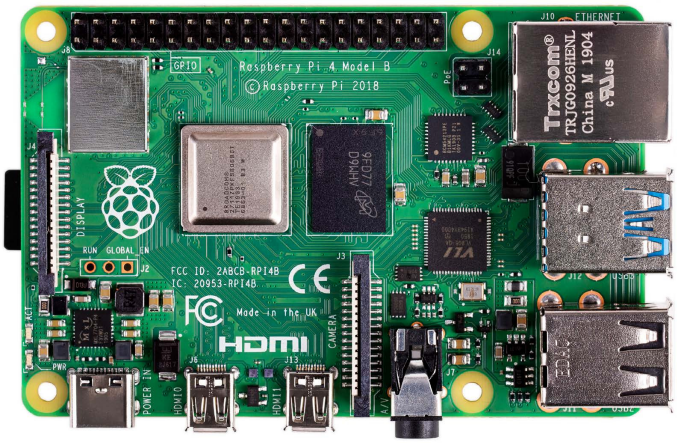


Figure 1: Raspberry Pi 4 Model B

The Keysight board was part of an IoT Educational Kit, which included all of the necessary tools to connect various sensors, like the digital pressure and analog temperature sensors used in this project. Two of these boards acted as IoT devices in the network, sending sensor information to the server hosted on the Raspberry Pi. Below is an image of the IoT Educational Kit and the connected pressure and temperature sensors.

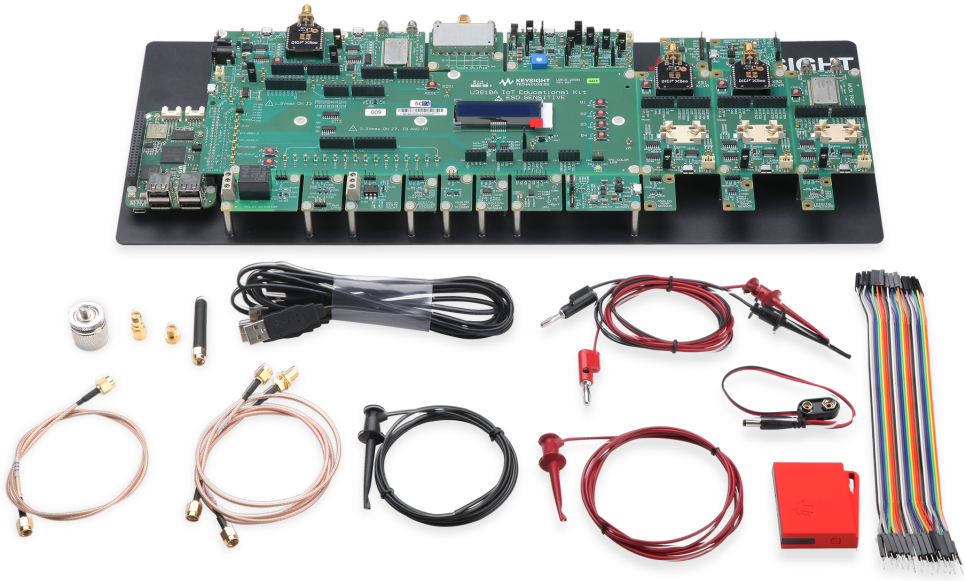

Figure 2: Keysight Technologies IoT Educational Kit

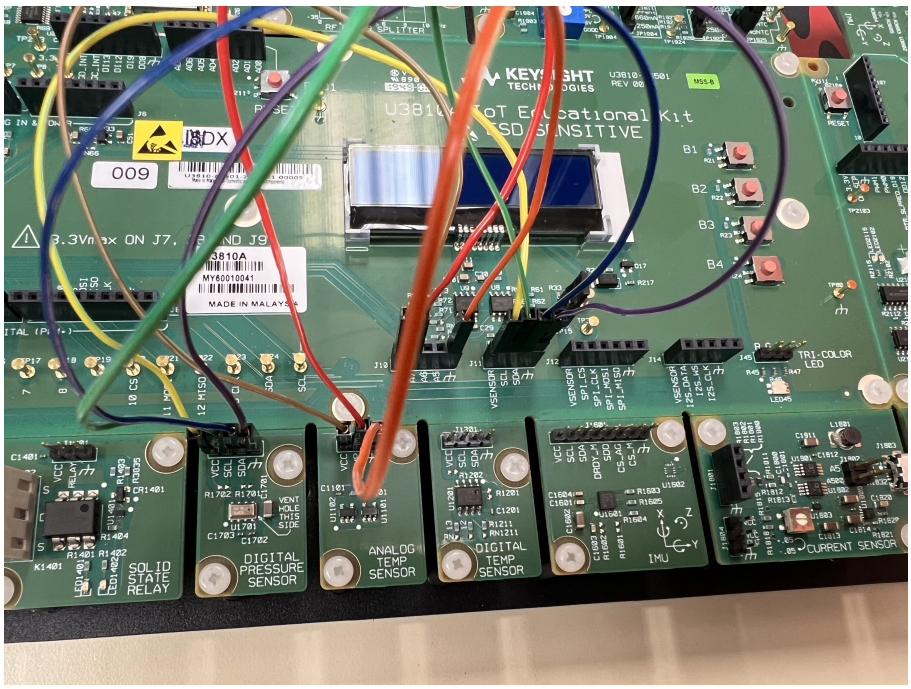


Figure 3: Keysight Board Digital Pressure and Analog Temperature Sensors

## Software

Below is the Class Diagram describing the structure of the network. The network consisted of two IoT device clients and a Raspberrry Pi running the MQTT Broker, server, and web interface.

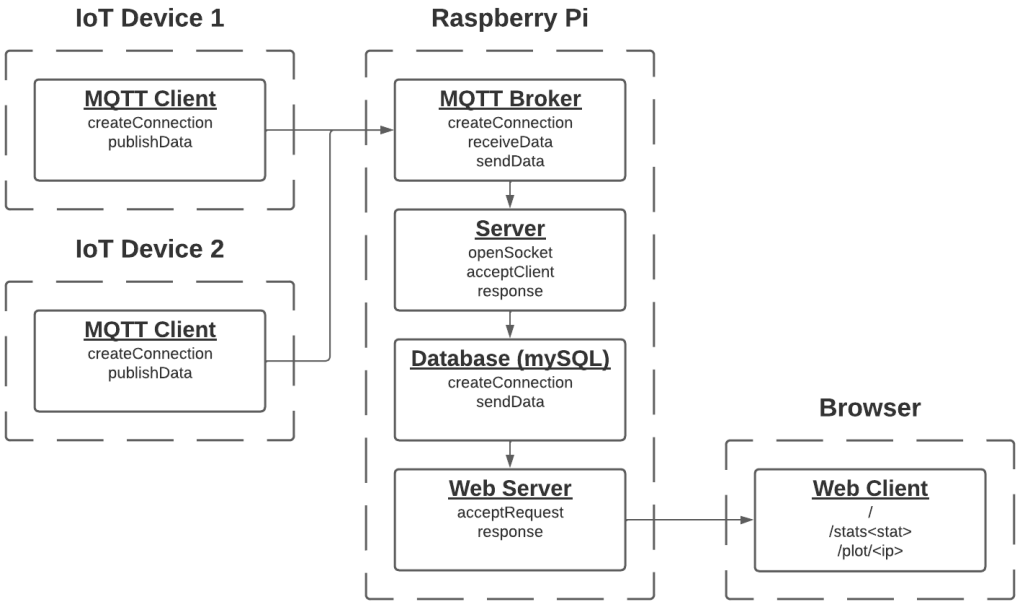


Figure 4: Class Diagram

### MQTT

All communication within the IoT network followed MQTT protocols. MQTT acts a publish and subscribe service that utilizes an MQTT broker as an intermediary between clients and server. All client devices within the network collected temperature and pressure data from their sensors and published data to their respective topics data/temperature and data/pressure located on the MQTT Broker. The server subscribed to these same topics on the MQTT broker and received the data published to these topics at any time. The MQTT broker managed the connected devices and subscribed to system topics itself. All of the sending and receiving of data was encrypted through SSL/TLS encryption to ensure that attackers could not intercept any data information.

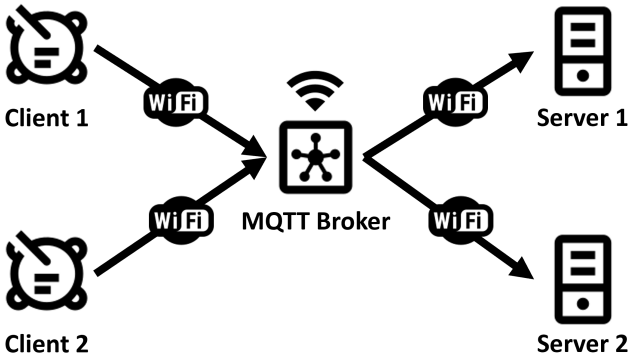


Figure 5: An Example MQTT Network Containing a Broker, Two Servers, and Two Clients

### IoT Devices

Two IoT devices are registered in the network. Once registered, the data collected using the analog temperature sensor and digital pressure sensor is published to the corresponding topics using MQTT communication. As a part of this message, the device's IP address, MAC address, and a timestamp are also added to the message as additional metadata used for authentication and verification by the server.

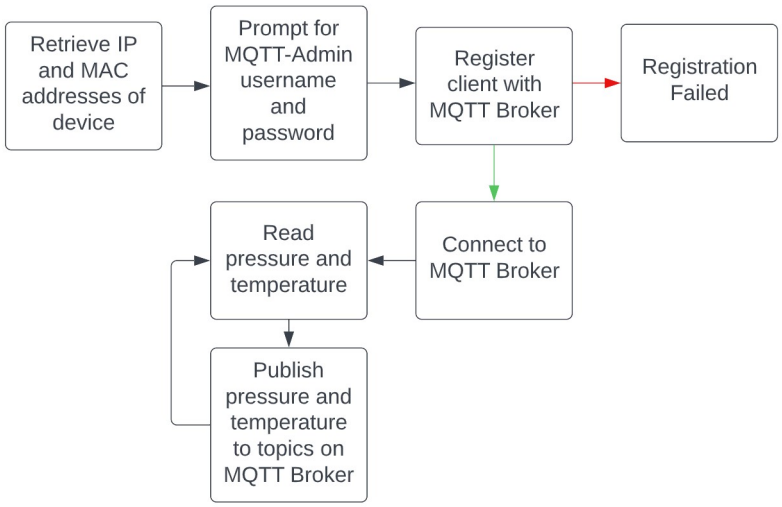


Figure 6: mqtt_client.cc

### MQTT Broker

The MQTT broker was run on the Raspberry Pi in the network. The broker used mosquitto to manage all of the connected devices and subscription topics. It handled accepting the publish messages from the IoT devices and sending the messages to the subscribed server.

Mosquitto is a MQTT broker library that handles the hosting of an MQTT broker. Using mosquitto is important because it provides a locally hosted broker, which allows its configuration to managed within the WiFi network without outside internet access. In addition, mosquitto supports SSL/TLS communication and can be extended with an authentication plugin, allowing all aspects of the ZTA to be implemented on the broker. Therefore, mosquitto is the perfect solution to provide end-to-end security throughout MQTT communication in the ZTA IoT network.

### Server

The server was run on the same Raspberry Pi as the MQTT broker in the network. The server, once registered, subscribed to the data/temperature and data/pressure topics in order to receive information from the clients. Upon receiving each piece of information, the server verified that the data packets were coming from a trusted, registered client before pushing it to the encrypted database. Additionally, the broker subscribed to the following $SYS topics in order to be used for security analysis: number of clients registered, number of clients connected, number of messages sent, number of messages received, and number of subscriptions.

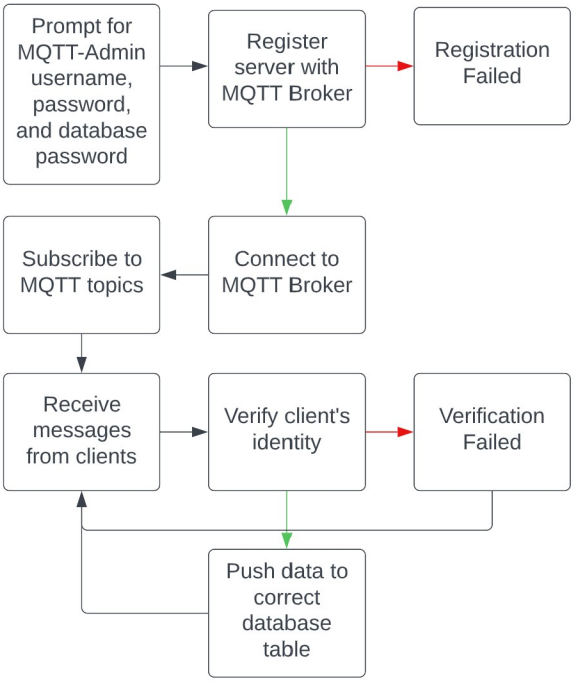


Figure 7: server.py

### Web Interface

The web interface was also hosted on the Raspberry Pi in the network. The web interface acted as a dashboard to display the security information from the $SYS topics on the MQTT broker and also as a way visualize the live data collection being sent from clients to the server. All of the components on the web interface were live updating and worked by grabbing the latest information from the database.

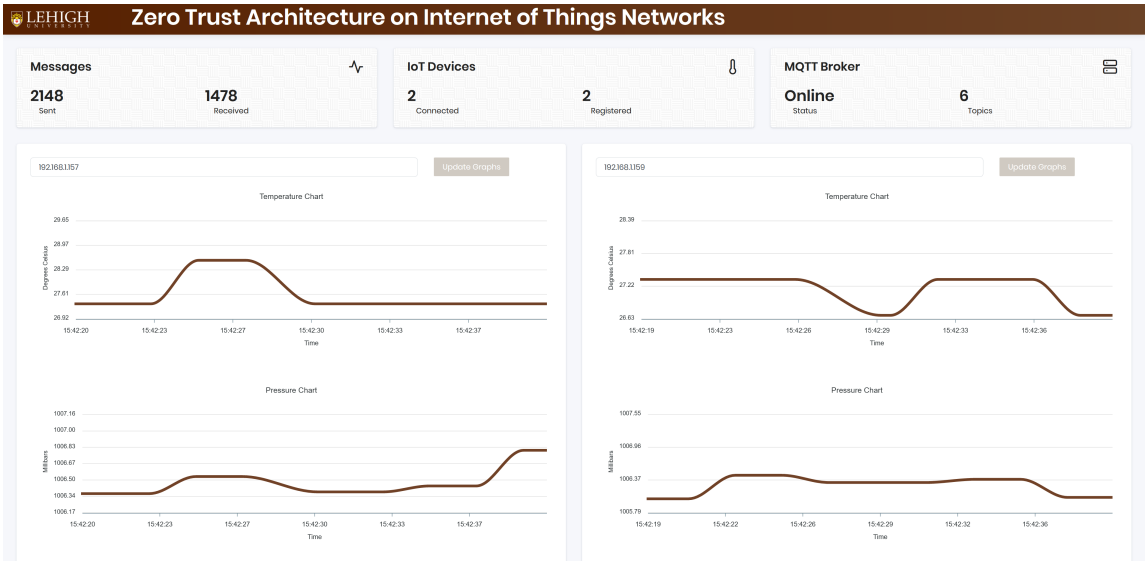


Figure 8: Web Interface

### ZTA: Device Registration

Upon trying to connect to the network, a new client was required to register. The registration process required the device to enter the correct username and password and possess the correct certificates signed by a Certificate Authority created using OpenSSL. The server was also required to register with the same criteria with the addition of the password required to access the encrypted database.

### ZTA: Device Certificates

Certificates for the client devices and the server were signed using a certificate authority generated using OpenSSL. These certificates are used as verification for secure communication between devices. With every transaction, the certificate of the two parties are compared against each other. If they match, then the communication between the two devices can continue. However, if they do not match then the transaction is cancelled. This verification process ensures that all devices are communicating with other recognized and verified devices. In this project, device certificates are used in all forms of communication, including when both the IoT device and server connect to the MQTT broker.

### ZTA: Device Permissions

Mosquitto on the MQTT broker assigns permissions to devices having certain roles. For example, the devices registered as "admin" were able to create other roles and assign permissions whereas the device registered as "server" could only subscribe to specific data and statistic topics. In addition, any permissions that were not assigned to a role were disabled by default. This meant that even though a "server" was not assigned a permission explicitly preventing it from publishing data, the "server" still does not have the ability to publish data by default. This method of zero trust is most important in this scenario, as it reduces the damage bad actors can do in the network.

| Role | Permissions |
|---|---|
| Admin | Create roles, assign permissions, subscribe to all topics |
| Server | Subscribe to data and statistics topics |
| IoT Device | Publish to data topics |

Table 1: Permissions

### ZTA: Encrypted Database

All of the data collected from the MQTT subscription topics was pushed to an SQLite database encrypted using SQLCipher. The database consisted of the tables below however all fields were stored in ciphertext as opposed to the plaintext shown for example. The database key was required to access the database each time.

| IP | MAC | Timestamp | Temperature |
|---|---|---|---|
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-10 01:19:29.452069 | 25.2 |
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-10 01:19:52.648277 | 25.2 |
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-10 01:20:15.880278 | 25.2 |
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-09 20:50:54.866247 | 25.4 |

Table 2: Temperature Database Table

| IP | MAC | Timestamp | Pressure |
|---|---|---|---|
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-10 01:19:29.452069 | 1003.1 |
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-10 01:19:52.648277 | 1003.2 |
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-10 01:20:15.880278 | 1002.9 |
| 192.168.1.211 | f8:01:72:c5:16:e4 | 2021-11-09 20:50:54.866247 | 1003.0 |

Table 3: Pressure Database Table

## Intellectual Property

Although there exist no standard security requirements across IoT devices, there are some frameworks that have been patented by large institutions in the IoT space. Two patents that pertain specifically to this project are one submitted by Bank of America, which is an IoT Protection Retro-System, and one submitted by T-Central, which is a system and method for IoT security and management.

The IoT Protection Retro-System is a method for securing communication in a network of IoT devices. It requires an authentication hub device, which stores client identification data in a local database. When IoT devices want to transmit data over the network, they are required to authenticate first with the hub, which confirms their identity with the information stored in the database. This framework ensures that any device communication that occurs is authorized first, allowing intended IoT devices to communicate on the network.

The system and method for IoT Security and management describes a way to connect multiple IoT devices for secure communication over the network. It uses an exchange of certificates between two devices to establish trust, and then passes encryption keys. These encryption keys are used for all communications between the devices, ensuring that a third device that tries to receive communications would not be able to successful understand the message. This security management framework protects against malicious devices intercepting IoT device communication.

The investigation of current patents develops strong background information for the design of this project. By understanding the current processes that are taking place to implement security on IoT devices, this project can build on top of the ideas they present.

## Alternate Design Considerations

At the beginning of this project, a major design decision was made. Two networks are common among IoT devices, known as ZigBee and WiFi. ZigBee is a specialized network that usually only has IoT devices connected to it and utilizes a hub as the entry point. Due to being built specifically for IoT devices, ZigBee is low bandwidth, which makes it incompatible with some higher bandwidth IoT devices like security cameras. WiFi, on the other hand, is a general network that is most commonly used by devices to connect to the Internet. WiFi has much larger bandwidth and does not always require a hub to connect devices, so its compatibility extends across a wider range of IoT devices. In addition, because WiFi is commonly used by other devices in addition to IoT devices, the network can consist of many devices, known and unknown. Therefore, WiFi's wider compatibility with IoT devices and greater exposure to attacks on a more public network make it a better test of a ZTA framework for this project.

Another important design decision involved the two IoT communication protocols. As the project progressed, the server had support for both MQTT and Constrained Application Protocol (CoAP). However, when it was time to encrypt the communication over the protocols, CoAP had some problems. Unfortunately, because CoAP uses the UDP protocol to send messages, it uses DTLS instead of SSL/TLS for encryption. DTLS does not have many libraries or support, and it was difficult to find any framework with trustworthy implementation of CoAP. This lack of support made attempting to utilize CoAP in the project more of a security risk than benefit, which led to the server no longer supporting CoAP communication. As a result, the final project was built using only the MQTT communication protocol.

## Bill of Material & Cost Analysis

Due to already having access to the Keysight boards in Lehigh's lab, it was only necessary to purchase a Raspberry Pi and MicroSD cards. The original plan was to host the server on one Raspberry Pi and the MQTT broker on a second Raspberry Pi, which is the reason for ordering two. However, in the final demonstration of the project, both the server and MQTT broker were hosted on the same Raspberry Pi. Below is the bill of materials that displays the total cost of this project.

| Product | Cost | Quantity | Total Cost |
|---|---|---|---|
| *Keysight Technologies U3814A IoT Board* | $0.00 | 2 | $0.00 |
| *Raspberry Pi 4 Model B* | $126.00 | 2 | $252.00 |
| *128 GB MicroSD Card* | $18.00 | 2 | $36.00 |
| ***Net Cost*** | | | **$284.00** |

Figure 9: Bill of Materials

## Governmental Regulations

IoT is a relatively recent technological development, so there is not as much relevant government regulation. In fact, there is a significant lack of regulation. In the United States, there currently exists no federal law the regulates the collection and use of personal information by IoT devices. This means that IoT devices, which could consist of sensors, security systems, or even security cameras, can collect personal information to be used by the manufacturer how they like. With no protection from personal data collection, the only laws limiting manufacturers are general internet privacy laws, which do not take into account the specificity of IoT devices.

The regulation that does exist for IoT devices covers only government issued devices. The IoT Cybersecurity Improvement Act of 2020 allows the National Institute of Standards and Technology to manage and set cybersecurity standards and risks for devices acquired by the federal government. This type of standardized security management of IoT devices is an important step to protecting all communication across devices, but currently it only exists on devices used by the government. Therefore, the average consumer does not get this protection.

Overall, the lack of regulation in the IoT space leaves many security and privacy decisions up to the individual device manufacturers. This results in many different frameworks and technologies being used in different IoT devices, creating many compatibility problems and a lack of security. The standardized security of government IoT devices is an important step, and something this project hoped to accomplish for consumers by creating a ZTA on a network of IoT devices.

## Industry Standards

Like government regulations, there is a significant lack of industry standards for IoT devices. The closest thing to industry wide standards are a set of suggested guidelines from the National Institute of Standards and Technology. These guidelines are not required, only serving as a reference for manufacturers of IoT devices. When designing and creating their products, manufacturers can implement some of the security suggestions from the guidelines, but even these guidelines only provide conceptual security advice. This means that there are no standardized technologies suggested across IoT devices, which makes compatibility extremely difficult. A lack of compatibility across the industry makes any sort of widespread security protocol difficult to implement as well. Overall, this lack of industry standards create an industry that has a lot of variety, which is why this project is important in attempting to create a security architecture that is compatible with a variety of devices.

### Project Management

This project took place over two semesters. The first involved investigative research on IoT devices and ZTA, as well as some primitive implementation. This can be seen in the gantt chart below.

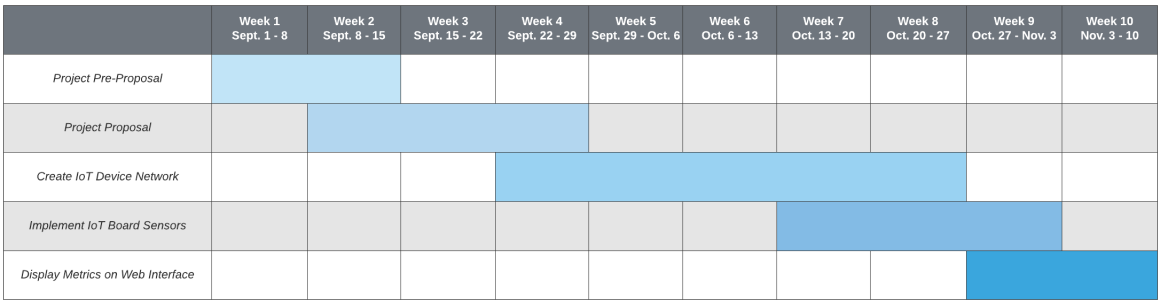


Figure 10: 1st Semester Gantt Chart

The second semester encompassed most of the implementation found in the final demonstration, which included supporting encrypted MQTT communication, multi-factor authentication, and security statistics displayed on a web interface. These can be seen on the gantt chart below.

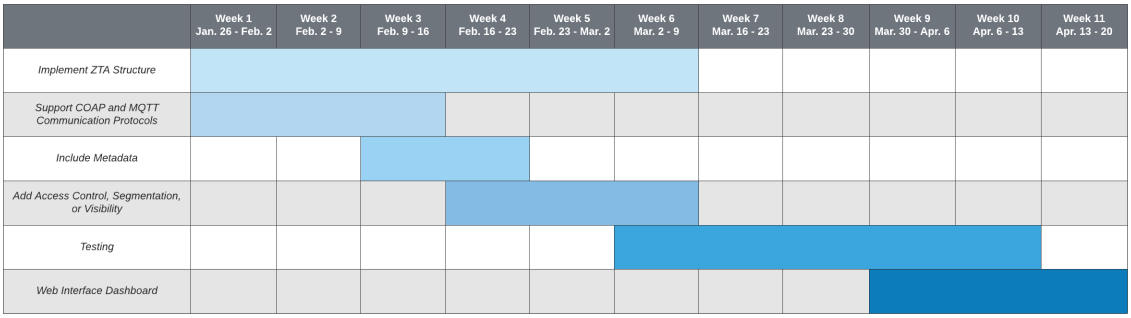


Figure 11: 1st Semester Gantt Chart

Both of these gantt charts were quite accurate with the actual timeline of the project, with the exception of the implementation of CoAP support in the second semester. The creation of gantt chart in the beginning of each semester helped lay out a specific plan for the project, establishing a list of goals that needed to be accomplished. These goals helped keep the project on track and focused on the end result of implementing ZTA on a network of IoT devices.

## Evaluation

In order to evaluate the ZTA designed in this project, tests simulating common cyber-attacks were conducted on the network. The first test was a Man in the Middle attack, in which the packets being transmitted over the network were intercepted by an unwanted third party. The other two tests consisted of bad actors attempting to subscribe to MQTT topics they did not have permissions to access and logging in on an IoT device not registered with the server.

The Man in the Middle attack simulated a malicious third party trying to steal sensitive information. It was conducted using Wireshark on the ZTA network. While an IoT device was sending information to the server, Wireshark intercepted some of the packets on a third party device connected to the network. The intercepted packet had information on the IP address of the IoT device and server, but the data being transmitted was encrypted. This meant that even though the packet was intercepted, the data being transmitted was still

safe, which illustrates how encryption is used to secure the network.

The two bad actor tests are similar, but they each evaluated different aspects of the ZTA. The first bad actor attack simulated a malicious device attempting to subscribe to sensor information from the MQTT broker. Although this device was already registered with the server and had a valid login, it was denied the ability to subscribe to sensor information. This was because the MQTT broker implemented an access control policy which prevented all IoT devices from subscribing to any topics. These permissions played an important part of the zero trust aspect of the project, as even after a device was registered, it was not trusted to perform actions outside of its intended scope. The second bad actor attack simulated a malicious, unregistered device that had the correct login information, but not the correct certificates. When attempting to register with the server, the IoT device failed due to the invalid certificates. This test highlighted the importance of multi-factor authentication. With only one requirement for authentication, this malicious device could have successfully connected to the network as an imposter. However, because there were multiple requirements for authentication, this device was unable to connect and the network remained secure.

Overall, after conducting tests on the ZTA network, it was clear that it could handle the most common cyber-attacks. Although this did not mean the network is perfect, as cybersecurity is constantly changing, it meant that it could provide a general standard level of security across a network of IoT devices.

## Conclusion

The outcome of this project was a fully functioning, secure network of IoT devices connected over WiFi and a web interface displaying data and statistics. All devices in the network were verified upon registration and transaction using multi-factor authentication. The initial plan supported both MQTT and CoAP communication protocols, however, after extensive implementation testing, we concluded that CoAP does not currently support the encryption functionality needed to be secure. Therefore, all data was transmitted through SSL-encrypted MQTT.

## Future Work

In the future, the ZTA can be extended to include additional authentication methods as part of the multi-factor authentication like an RSA token or a Time-Based One-Time Passcode. Increasing the number of factors in the authentication process will further increase the security of the network. Additionally, CoAP communication could be implemented within the network once more secure, encrypted communication using DTLS becomes available. Being able to secure both CoAP and MQTT communications will allow the server connect with nearly all IoT devices in the network, taking one step closer to standardized security framework that is compatible with all IoT devices across a network.

## Acknowledgements

Firstly, thank you to our project advisor, Professor Cutitaru, who has provided direction and support throughout the semester. Secondly, thank you to Professor Zheng for introducing us to and allowing us to utilize the Keysight IoT Educational Kits. Finally, thank you to Professor Norian, Professor Frey, and Xiyuan Zhu for coordinating CREG 258 and offering continual feedback.

## References

"Implementing a Zero Trust Security Model at Microsoft." IT Showcase, https://www.microsoft.com/en-us/insidetrack/implementing-a-zero-trust-security-model-at-microsoft.

Kravitz, David W., et al. System and Method for Internet of Things (IOT) Security and Management. 25 July 2017.

Kurian, Manu. Internet of Things ("IOT") Protection Retro-System. 14 Sept. 2021.

"Minimum Security Standards: Internet of Things (IOT) Devices." Minimum Security Standards: Internet of Things (IoT) Devices | University IT, https://uit.stanford.edu/guide/securitystandards/iot.

Rose, Scott W., et al. \Zero Trust Architecture." NIST, 23 Mar. 2021, https://www.nist.gov/publications/zero-trust-architecture.

"The Importance of Security in IOT." Logz.io, 1 Nov. 2021, https://logz.io/blog/the-importance-of-security-in-iot/.