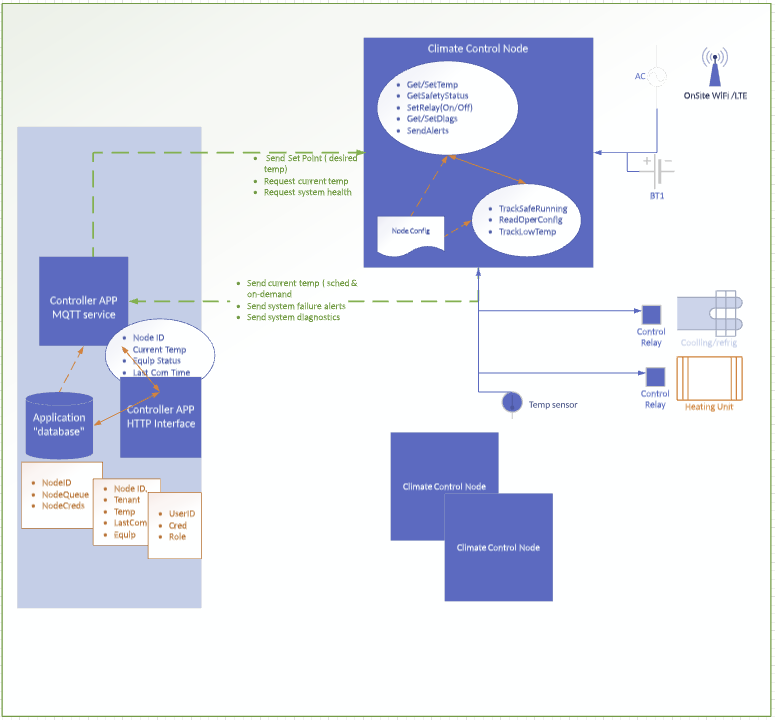
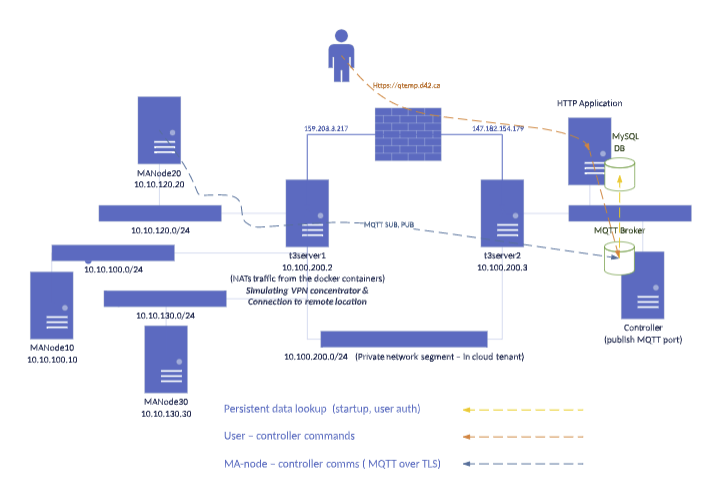
## Solution Description

The Internet of Things (IoT) climate control solution (Fig.1) was designed to implement in various settings, including multi-tenant residential buildings and business complexes. However, the security controls and mitigations were considered in worst-case scenarios to address the risks and vulnerabilities of a business complex with critical temperature requirements. Sound programming principles were applied by moving database, encryption, and control-unit into their own Python module to enable decoupling.



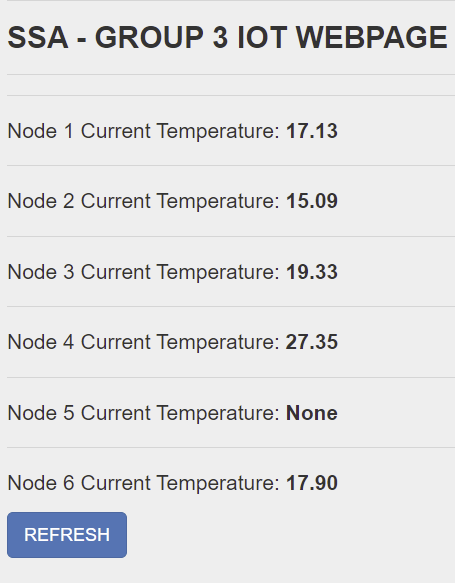
*Figure 1 – IoT Climate Control Solution Conceptual Design*

The solution’s logical design (Fig.2) employed network segmentation enabling individual MA-Nodes to communicate only with the controller, not other MA-Nodes.



*Figure 2 – IoT Climate Control Solution Logical Design*

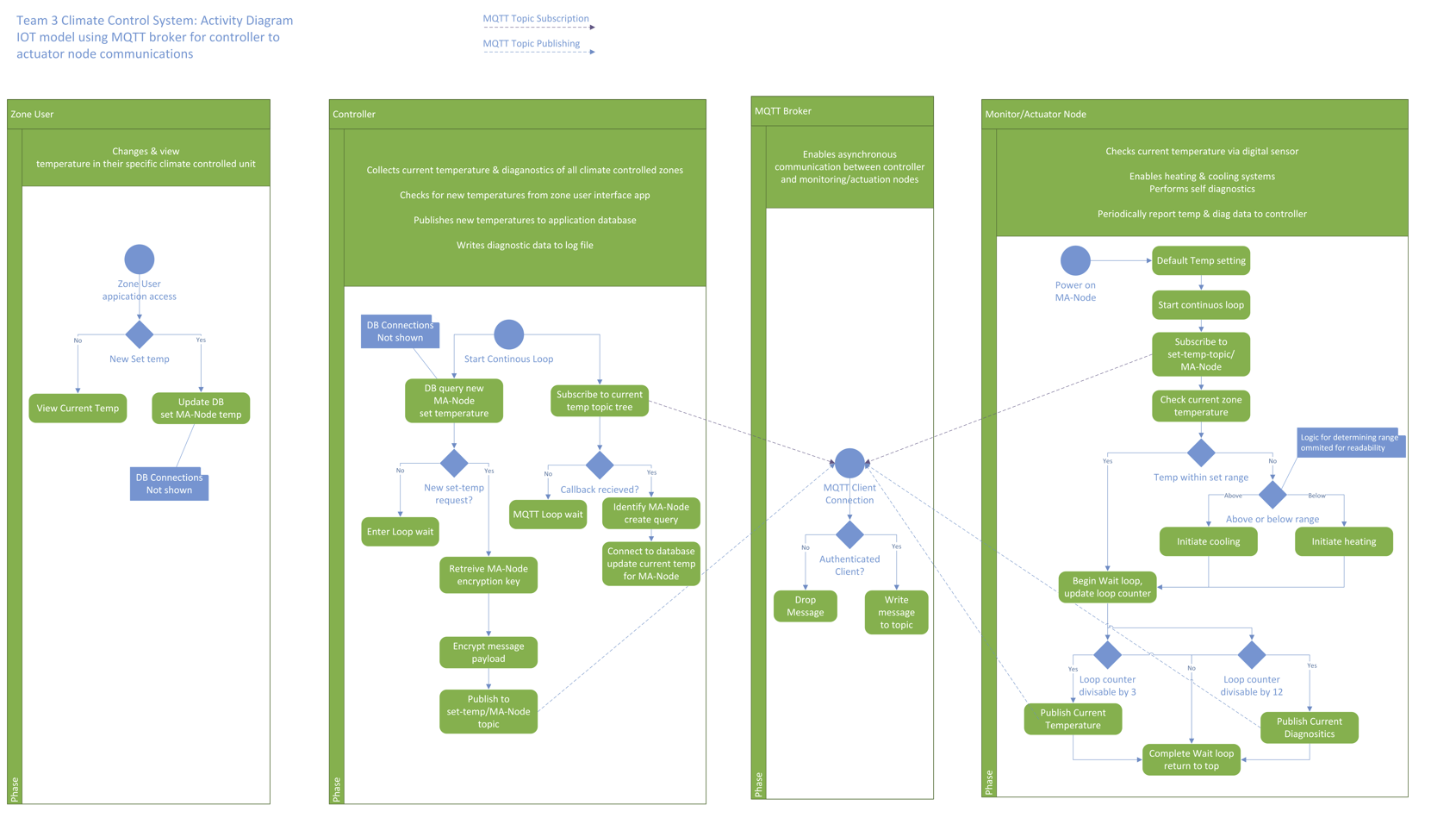
A web interface (Fig.3) was developed to simulate user interaction with the solution can be reviewed at <http://qtemp.d42.ca:5000/>. The interface requires further enhancement for true secure user interaction with the solution and not in project scope.



*Figure 3 – Web Interface*

## Code Execution

Code execution outlined in the sysml activity diagram (See Fig.4) located at [Link](https://github.com/narunanthy74/arun/blob/master/Part%202%20Upload/Activity-SYSML.pdf), was also demonstrated in the following video recording: <https://drive.google.com/file/d/1u1dCEQ9z3CAprL0MiB9PEVuNywQ7HS-a/view?usp=sharing>.



*Figure 4 – Execution Steps*

## Microservice implementation

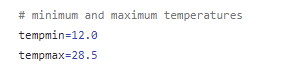
The modelled solution utilizes eight Docker containers interacting at the service level using network communications. Application logic and control have been implemented via customized containers ([ssat3\_controllerClientV2\_2.Dockerfile](https://github.com/narunanthy74/arun/blob/master/Part%202%20Upload/SSAT3_Controller-Build%202.0.docx), [ssat3\_manodeV2\_0.Dockerfile](https://github.com/narunanthy74/arun/blob/master/Part%202%20Upload/SSAT3_MA-Node-Build%202.0.docx)); message communications and application state utilize official Mosquitto and Mysql and docker images.

## Discussion of Vulnerabilities

Risks and vulnerabilities were addressed by managing operational risks and cybersecurity vulnerabilities.

Operational Risks and Mitigations:

Safety limits (Fig.5) were hardcoded into the module and not exposed as adjustable settings to ensure maximum and minimum temperatures will not be exceeded regardless of temperature change input values. This simulates the out of band control often put into systems via physical measures.

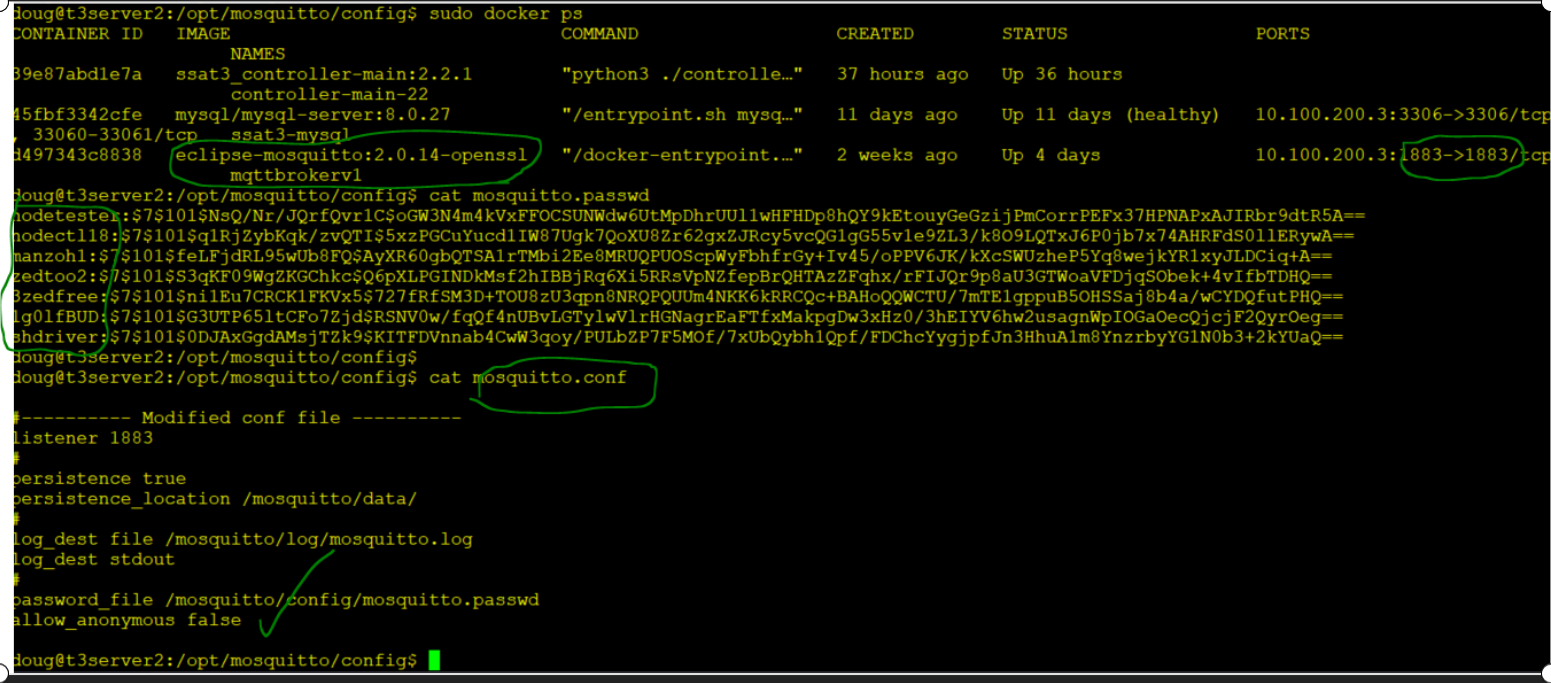


*Figure 5 – Temperature Boundaries Configured*

Security Risks and Mitigations:

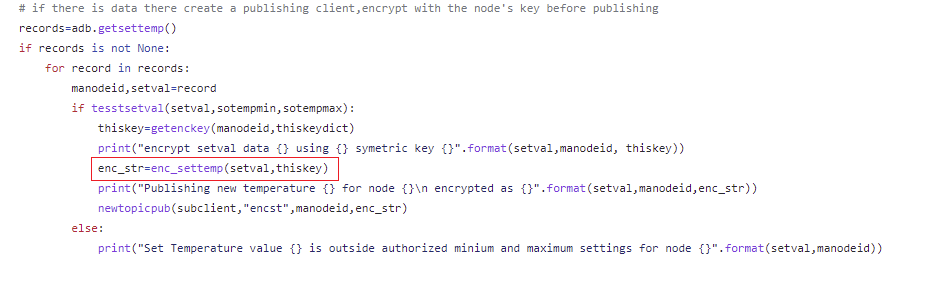
Processing malicious messages by the MQTT nodes or brokers can result in extensive system impact. Therefore, the listed application security measures were implemented to protect service integrity (Firdous, et al., 2017):

* Anonymous subscribing and publishing (See Fig.6) were disabled (mosquitto, n.d.)



*Figure 6 – Configuration to disable anonymous subscribing and publishing*

* Message payloads were encrypted for non-repudiation (see Fig.7)
* Availability of the MQTT broker service was limited to the private network only



*Figure 7 – Encryption Configuration*

In addition, the following controls were implemented to mitigate administrative account or OS compromises:

Customised container (controller and MA-node) application components were implemented as microservices, preventing OS access if exploited; In addition, non-privileged users (Fig.7) were used to execute the Docker containers (Jain, 2020).



*Figure 7 – Non-privileged users were used to executing the commands*

### Current Solution Limitations and Future Remediations

* Currently, the traffic between the device to the controller is not encrypted. However, this traffic can be encrypted using TLS or similar in the future.
* According to OWASP, all encryption keys must be stored in a cryptographic vault or isolated cryptographic service (OWASP, n.d.). A vault service can be utilised to store symmetric keys for additional security during a production implementation (Vaultproject, n.d.).
* In addition, temperature values reported by the MA-nodes were not encrypted, which may result in report manipulation. This incident can occur due to a physical compromise or an insider attack.

## Testing and Verification:

Figure 8 summarizes the secure architecture requirements, test methodology and results for system verification and acceptance.

Graphical user interface, text

Description automatically generated

*Figure 8 Table of system requirements and verification methods.*

## References:

OWASP. (n.d.) Key Management - OWASP Cheat Sheet Series. Available from: https://cheatsheetseries.owasp.org/cheatsheets/Key\_Management\_Cheat\_Sheet.html [Accessed 16 Dec. 2021].

Firdous, S., Baig, C., ValliC & Ibrahim, A. (2017) ‘Modelling and Evaluation of Malicious Attacks against the IoT MQTT Protocol’, *EEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. Exeter, United Kingdom, 21-23 June 2017. IEEE. 748-755.

Jain, R. (2020) Running Docker Container as a Non-Root User. Available from: https://www.tutorialspoint.com/running-docker-container-as-a-non-root-user [Accessed 16 Dec. 2021].

Vaultproject. (n.d.) *Introduction*. Available from: https://www.vaultproject.io/docs/what-is-vault [Accessed 16 Dec. 2021].

Paho. (n.d.) Paho - Eclipsepedia. Available from: https://wiki.eclipse.org/Paho [Accessed 19 December 2021].‌

Mosquitto. (2020) Mosquitto Conf Man Page. Available from: https://mosquitto.org/man/mosquitto-conf-5.html [Accessed 19 December 2021].‌

‌

‌