DAT 235

IoT Security

Prosjekt Dat-235

Author:
Bendik Egenes Dyrli
Nikolai Kjærem Ellingsen
Jens Martin Håsæther
Mathias Solheim Jansen

Supervisor:
Geir Myrdahl Køien

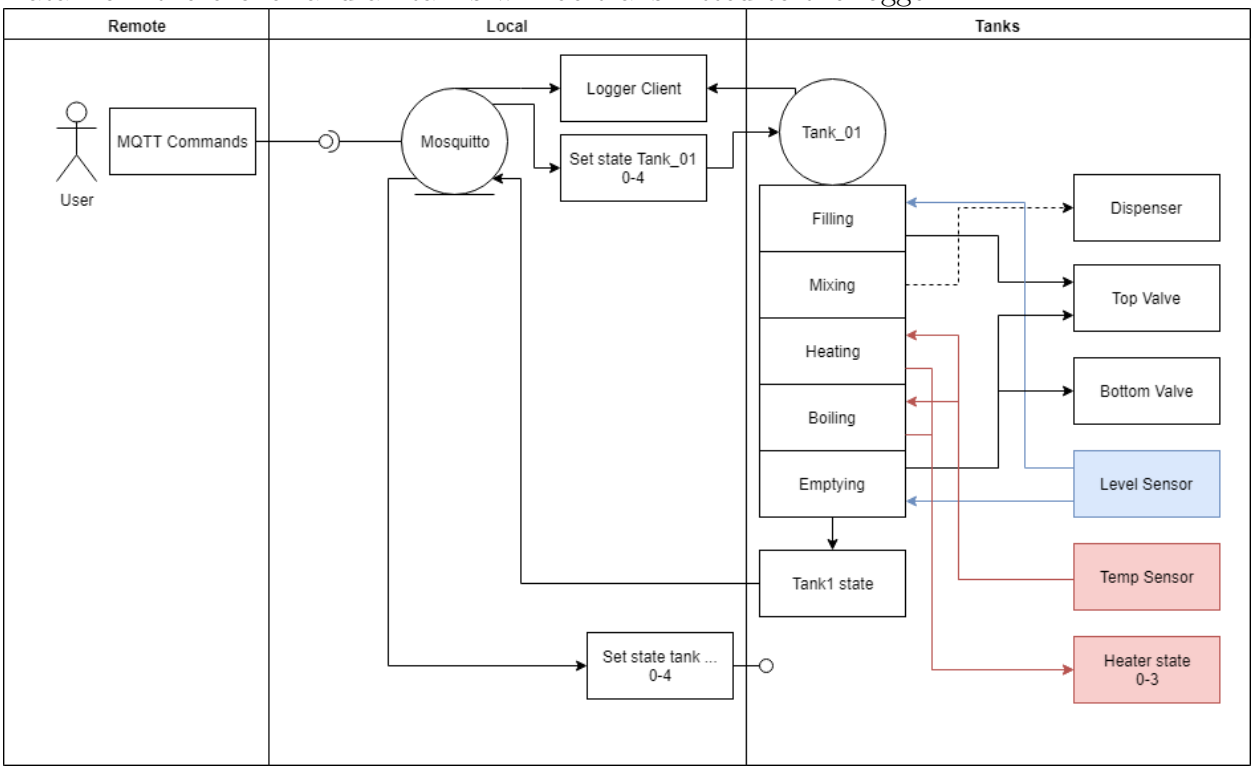
2017

Contents

Chapter 1

Introduction

In this task, we are given several specifications to adhere to with the end goal of designing and later building a simple control system for a chemical process plant. The system will consist of four parts; a remote user, a local broker, a logger and up to 5 tanks each with their of raspberry pi. Only tank 1 is shown due to space constraints and general simplification. Data from the broker and all tanks will be transmitted to the logger.



Chapter 2

Conops

The system itself is a container that is used to first mix a substance with a cold liquid, and then warm the liquid, close to boiling temperature temperature and keep it at this state for a certain amount of time. For this purpose we are given the following equipment and restrictions:

The system itself will be compromised of 2 sub-systems: The "Boiling Unit" or BU, and a "Control Unit" or CU that can manage, log and control the given equipment. The communication protocol that will be used in this case is MQTT.

System 1 - Boiling Unit:

- Ideal Mixing volume - between 1000-1100 liters
- Ideal Boiling Temperature - between 80-88 degrees Celsius
- Boiling Time - 100 Seconds
- Max Capacity - 1200 Liters
- Needs to withstand 100 degrees Celsius
- Output capacity of 100 liters/sec
- 4 Actuators (switched with MQTT)
  - Heater unit with 4 settings
  - Input Valve
  - Output Valve
  - Dispenser
- 2 Sensors (info travels on MQTT)

  - Level Indicator
  - Temperature Sensor

System 2 - Control Unit

- Manage
  - Read logs
  - Switch Controls according to logs
- Log
  - Temperature
  - Heating Time
  - Input data
- Control
  - Switch Actuators
  - Signal Control
  - Encryption
  - Clients+

Abstract process flow:

1. A cold liquid (water) is introduced into the boiling tank through the "Input Valve" until a certain capacity.
2. The "Dispenser" introduces the chemical/solvent to the water.
3. The "Heater" unit will be switched on maximum to raise the temperature to around 80-88 which will be measured by the "Temperature Sensor".
4. When the temperature reaches 80-88 degrees, the "Heater" will be have to be managed so that the temperature stays in this range for a total of 100 seconds.
5. When 100 seconds has passed the mixed liquid will be removed from the Boiling tank with the Output Valve.

The values in this example will compromised of the assumed liquid, the boiling tank, and whatever data that is deemed valuable on the control unit.

Assuming that this process doesn't require a physical person to oversee anything happening with the BU, there is only need for a one-time setup with CU, and maintenance for both sub-systems.

When it comes to security the most sensitive data is the control commands for the actuators, these should have some fault-tolerance as well as tamper-resistance as it would not be lucrative to have the liquid compromised in any way. The logging in and of itself is of little to no significance in this case and has no inherent need to be encrypted.

Chapter 3

Base line security

3.1 Platforms and hardware

The system will consist of several raspberry pi zero wireless' running raspian streach lite. Which is a Linux operating system based on Debian. These devices are chosen due to their modularity and reasonalbe cost/availability. They also contain numerous features which will allow less complexion in the setup. Such as wifi, bluetooth and serial interface. The devices will be setup to only allow specific connections on specific ports and will as a precaution have all passwords changed from defaults. Likewise the setup will only contain the bare necessities to function; minimizing potential attack vectors.

3.2 Software and communication

Mosquitto will be used to allow MQTT communication between the devices which will be discussed later. The raspberry pi's will interface with each other with TCP/IP over bluetooth 4.1. This uses the Secure Simple Paring protocol which provides authentication with SAFER+ although vulnerable to brute force attacks due to it's 128-bit key size, it is deemed sufficient in this application as a result of it's low range. Encryption on the other hand uses E0 which is completely useless due to numerous vulnerabilities.[1] Due to this, basic encryption and checksums will be implemented on the software level with payload encryption. This gives end to end encryption, but it also means that potential passwords sent during connection are not secure. [2, 3]

Chapter 4

MQTT

In this project we are obliged to use MQTT, but this part will be more about how we'll secure the communications with the Raspberry Pi Zero. All communication will use MQTT; the server will run Mosquitto MQTT Broker, the tank pi's Paho Python MQTT and the users will be allowed the freedom to choose.

4.1 Security

First priority is getting the MQTT working properly as well as acquiring certificates from letsencrypt which will ensure proper https/tls encryption. However, if we'll have the time we'll try to setup MQTT passwords, meaning that you'll need to have an account to be able to communicate with the server.

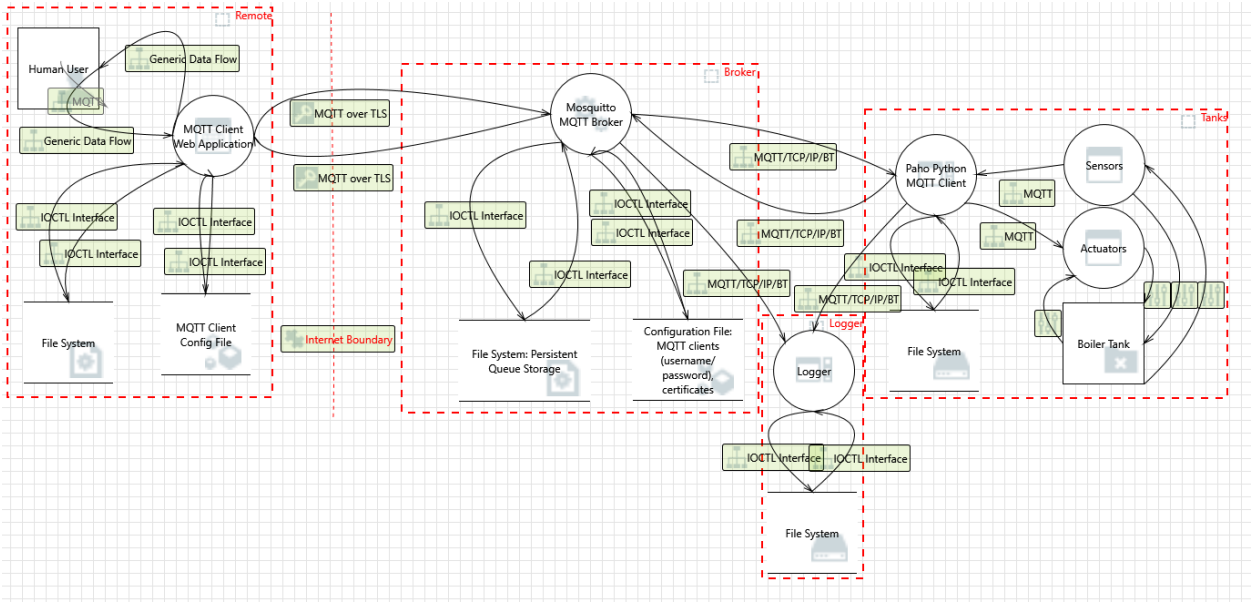
4.2 Broker

The broker will be hosted locally by a raspberry pi, with a backup server should the main server become unavailable. This backup will only contain the initial setup needed to interface with the tank pi's and broker and will be kept offline. When it comes to the communicating with the Raspberry Pi Zero W, there will be taken the precaution of giving the remote users certificates and only allowing connections to the broker from specific IP addresses. The certificates will ensure that the IP's cannot be spoofed to gain access or perform denial of service attacks.

Chapter 5

Threat Modeling

Threat modeling is an important part of the starting phase. It is important that every kind of possible threat has been looked at and evaluated. The full threat modelling report will be found as an appendix however a short summary will be explained here. Many of the problems listed in the threat model will be solved with a simple authentication mechanism, this will make sure that only certain users are allowed to do certain things. Every system will also be secured physically so that only those who are allowed has access to the hardware, this will partially mitigate the possibility of tampering. All inputs and outputs must be sanitized to make sure that nothing out of the usual is allowed. As a continuation of the security measures, training will be provided to employees. This will reduce potential user error as well as provide faster error handling. Denial of service will be partially mitigated due to a list of trusted sources being allowed to interface with the broker. See appendix 1

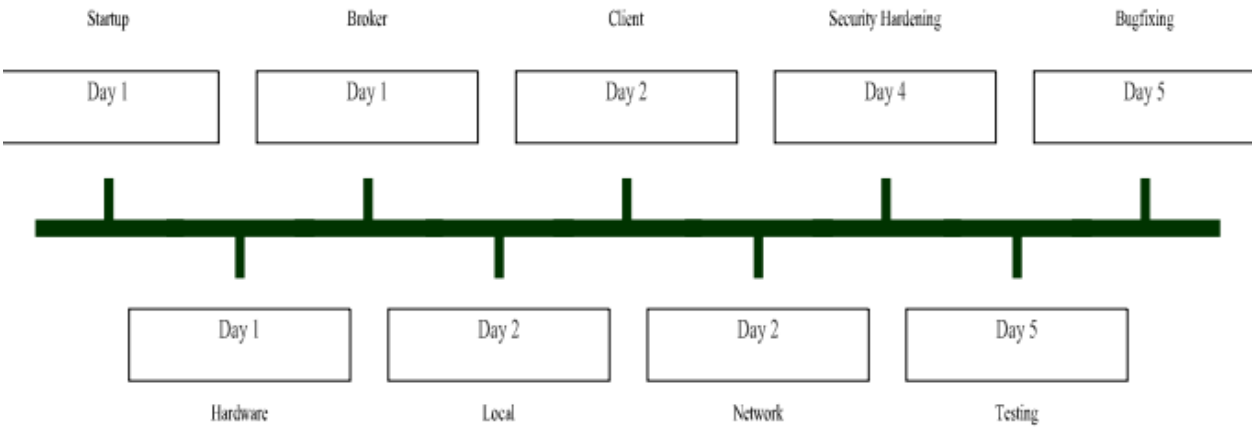


This page is intentionally left blank

Chapter 6

Setup

This setup is estimated to take 30 hours divided among 4 people. Which will be completed within 7 days with the schedule below. As seen below, the setup should only take 5 days. The reasoning behind this is to allocate time to write the report. It will also functioning as a buffer to potential mishaps and/or complications. Systems may be dropped or simplified if deemed too time consuming within the projects time frame. This will be included in Phase 1. The physical security of the hardware will be included if time permits it, but this is deemed out of scope for now.



Bibliography

[1] Difference between Secure Simple Pairing and Secure Connections in Bluetooth? [Online]
https://security.stackexchange.com/questions/116027/difference-between-secure-simple-pai
[Accessed:9.November 2017]

[2] MQTT Security Fundamentals: MQTT Payload Encryption [Online]
https://www.hivemq.com/blog/mqtt-security-fundamentals-payload-encryption
[Accessed:9.November 2017]

[3] MQTT Security Fundamentals – MQTT Message Data Integrity [Online]
https://www.hivemq.com/blog/mqtt-security-fundamentals-mqtt-message-data-integrity
[Accessed:10.November 2017]