Decentralised Blockchain Security Architecture for IoT and MQTT Brokers

Project Plan

Konrad Dryja

k.dryja.15@abdn.ac.uk

Department of Computing Science,
University of Aberdeen, Aberdeen AB24 3UE, UK

Introduction

Internet of Things (further referred as IoT) is a growing field with its presence getting larger from day to day. Equipment such as bus timetables, doorbells, thermostats or even pacemakers capable of sending alerts directly to emergency services are omnipresent. It's a system of computers, chips, phones, sensors deployed in our lives capable of communicating with each other without requiring any human intervention.

Sensitive data is very often handled and leak or misrepresentation can have catastrophic consequences. Additionally, those devices have limited computation power, which is sacrificed in favour of extended lifespan to avoid frequent maintenance and/or battery replacement. This makes it a very lucrative target for malicious actors. Researchers from University of Michigan performed a demo attack on a pacemaker, extracting personal information and changing the configuration.[1] This example and more clearly presents a need for solution capable of securing and closing off access to unwanted agents.

What is more, those devices frequently do not directly communicate on a peer-to-peer basis, but instead pass through intermediary resource handling the distribution of data. MQTT Brokers[1] are one of them, providing a Publisher/Subscriber architecture, allowing for information exchange without the constant need of interconnectivity between clients, instead of relying on created topics used as a relay. It's very important to point out that MQTT is not a piece of software, but rather a standard describing the operation of such backend and leaves the implementation up to end-users - the implementation is very often referred as MQTT brokers.

The documentation also clearly states that the security is not a part of the standard, but rather leaves this decision up to the implementors, offering several alternatives[2]. And while the transport of data can be assumed to remain secure (as MQTT operates over TCP/IP, thus encrypting the data using TLS), the access is often not. For example, an attack could be conducted, where malicious device communicates with the broker, impersonating another IoT hardware and finally compromise confidentiality.

Common implementation of MQTT brokers includes basic authentication based on a

[1] https://mqtt.org/

[2] https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html#_Toc3901261

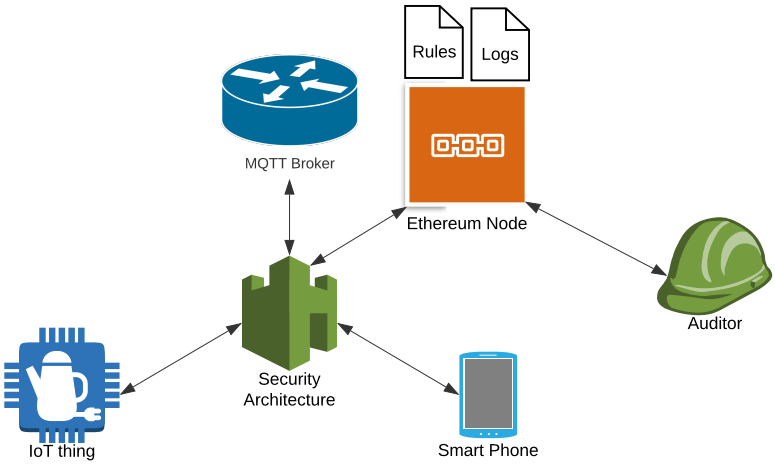


Figure 1: Proposed architecture

username and a secret password, offering limited to no logging capabilities. For example, one of the most popular brokers being Mosquitto[3], as of version 1.6.8 offers password based authentication with ACL capabilities restricting some credentials only to given topics. Related effort was presented in TrustLens project from University of Aberdeen presenting solution of attaching semantic annotations to ongoing operation of a broker and in process allowing higher transparency of broker's state[2]. And while it does not provide neither Authentication nor Authorization, it tackles Accountability.

I would like to propose a universal, pluggable module following the description of MQTT standard (and thus potentially making it independent from the used MQTT Broker) handling Authentication, Authorization and Accountability of IoT devices and consumers attempting connection to the brokers. In order to make the platform more resilient and independent from single points of failure (often being the central ACL repository), I will deploy it on a blockchain - which by convention supports recording of transactions (by placing them on the blockchain) and authentication via ID of the presented wallet. Figure 1 represents sketch of the proposed architecture, where the Security element would handle all incoming connections and proxy them accordingly. Due to the public nature of blockchain architecture, transactions would also be available to view by third parties for the purposes of auditing and transparency.

As mentioned, logging (that is, Accountability) will be the main focus of the project. Normally, it would not be feasible to record every event happening on the broker, as the number operation would greatly exceed the storage capacity. Relevant rules that would determine whether the event is worth recording or not would be created, those could include multiple failed attempts, high-profile operations (handling PII) or administrative tasks (such as permission changes).

Preliminary research revealed similar efforts by scientist in Khon Kaen University [3], although focusing mostly on Hyperledger Fabric and optimization of the configuration

[3] `https://mosquitto.org/man/mosquitto-conf-5.html`

through Genetic Algorithms. The paper doesn't make any mention of logging and provenance functionality, which I would like to put more attention towards - along with focusing on Ethereum platform instead.

Goals

The project can be divided onto main goals and extras, leaving some field for maneuvering in case of road blocks or difficulties resulting from the challenges faced in the dissertation. By having flexible targets, I will be able to stop sooner in case of overestimating the schedule, or carrying on with extra work, should I find myself meeting the targets quicker than expected. Moreover, I would like to also put some attention to listing the things that the project is **not** trying to be, to avoid sidetracking.

Main Goals:

- Design Blockchain network, relevant data models that would be placed on the blockchain and deploy on Ethereum platform, capable of recording transactions and allowing for modification of ACLs, i.e. which wallet ID is permitted to access given topics. Part of this goal would also be determining which information is important enough to be recorded and which should be discarded.
- Design rules and data models that would drive the premise behind the rules used for data capture.
- Design containerised software acting as a secure proxy between brokers and connecting clients. This will handle both authentication and logging performed action as immutable transactions. Logging would only be performed if the requested operation triggers some pre-defined rules.
- Deploy and test on at least one MQTT Broker (e.g. Moquette, Mosquitto), simulated with software.

Extra Goals:

- Create public API for the auditors to freely view the contents of blockchain and thus transactions containing information about suspicious operations.
- Explore and implement implementation of the architecture to any, arbitrary broker.
- Connect physical LoRaWAN gateways to the software.

No-goals:

- Design a new blockchain platform from scratch.
- Write / modify operating system of IoT devices.

- Design a new MQTT Broker – although modification of existing solutions might be required.
- Create a solution for hardware identification.

Methodology

For the project, I will be following iterative approach of fulfilling my targets, reflecting back on the sprints, which would be independent from each other (although one might rely on completion of another). Doing so would allow me to modify the requirements in case of facing blocks or difficulties that would not be feasible to solve in the given time span.

One could distinguish the following main tasks, that would be crucial when it comes to determining the success of the project:

- Preliminary research revealed some related work which I will need to review (along with others) to determine main focus points on the project and to apply to findings determined in other academic papers.
- At the start of this project, my knowledge of blockchain networks and their implementation is very limited, a further reading and study would be required.
- I have also never interacted with any MQTT brokers, so familiarisation by setting up lab environment would be necessary. Either Moquette or Mosquitto would be used, depending on the findings from the literature review. The respective documentation would also be reviewed.
- Having reviewed background information, I can proceed to designing the architecture of my system. This can be divided onto two smaller sub-tasks:
  - Design mock-ups of the system and seek peer-review on the implementation.
  - Majority of the project involves creation of relevant rules for transactions capture.
  - Implement the mock-ups in Golang, which would be deployed in containerised environment.
- With finalised software, testing and debugging can be commenced, again can be split onto two phases.
  - Testing in virtualised, hermetic environment
  - Field tests on physical hardware. This is optional and depends on the velocity of the project.

Resources Required

For the purposes of the project, laptop and a workstation should be sufficient, as the MQTT brokers can be simulated via software, without the need of physical IoT devices sending the messages. For the second stage of the project, I will be considering putting the software on a physical LoRaWAN gateway, enhancing the current broker implementation. TrustLens team from University of Aberdeen recently acquired such gateway, which I will be able to use given the need. Moreover, the team also owns several PuckJS sensors and Arduino boards, which would allow to deploy the whole system and test its behaviour in the real world.

To summarize the requirements:

- Linux PC with the following software:
  - Golang 1.12+ – runtime for the server application
  - Docker 19.03.5+ – deployment environment
  - Bazel 2.0.0+ – for building the source code
  - MQTT Broker – this could be either Moquette or Mosquitto
  - Suitable text editor (Vim)
- Additionally, for tests in the field:
  - LoRaWAN Gateway
  - IoT sensors (PuckJS, Arduino)
  - Mobile Device
  - Should the above not be sufficient, a Raspberry Pi running standard Linux environment

Extra requirements might arise throughout the project, although I do not expect anything that might require big time or financial investments.

Evaluation

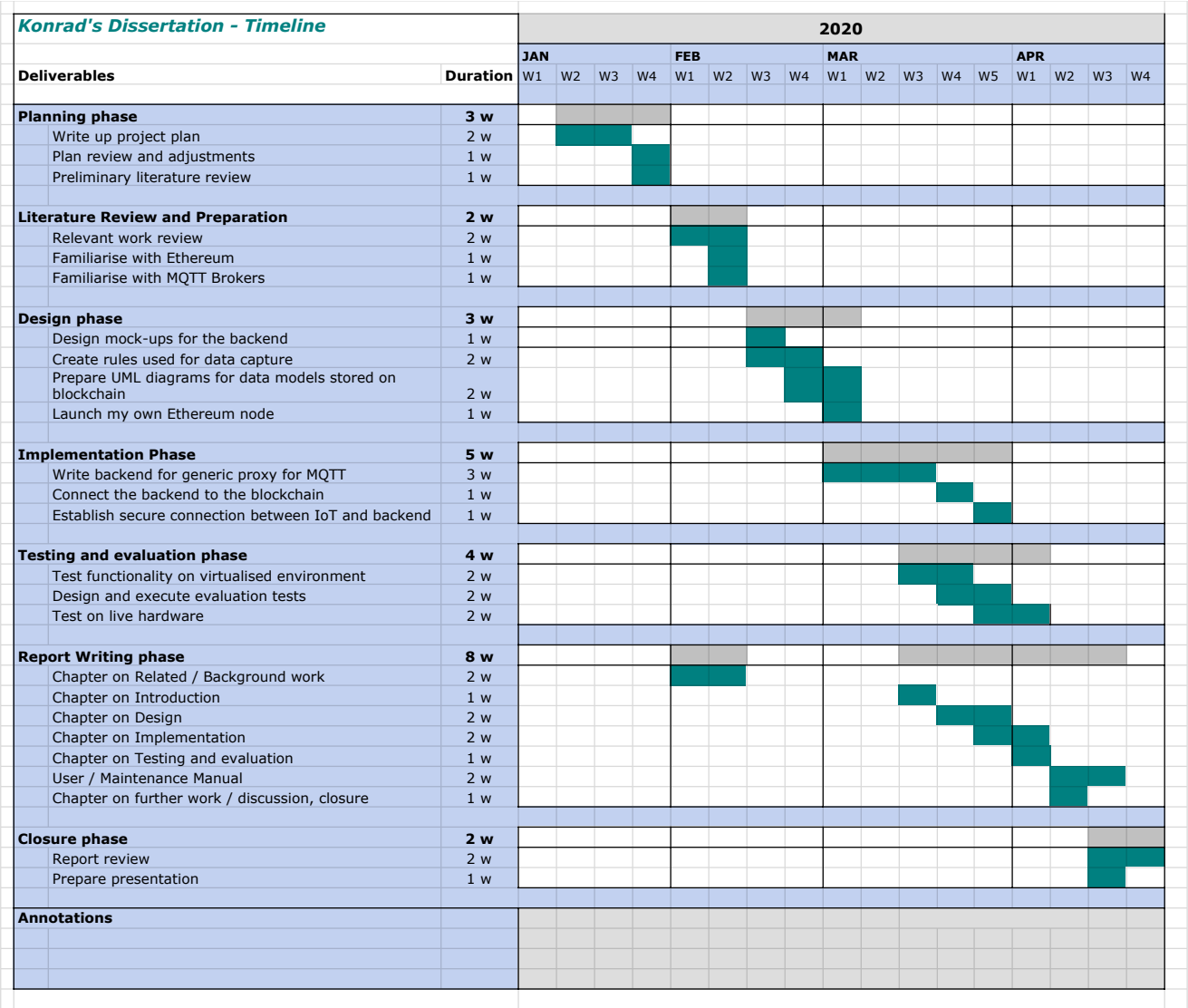
To determine the success of the project, a substantial amount of the time will need to be spent on evaluating the performance and functionality. As my project is likely to introduce operational and performance overhead, comparison between vanilla or similar solution will have to be in order to determine the feasibility of the proposed system. Several test scenarios will be considered:

- Time to process messages of new system with increasing number of messages, 100, 10k, 100k and so on.
- Determine whether my system is capable of stopping and/or capturing attacks which would've gone unnoticed in a vanilla MQTT Broker. This could included DoS type of attacks, brute forcing or unusual patterns of operation.

Risk Assessment

Risk	**Mitigation**	**Level**
• LoRaWAN gateway owned by the university utilises a completely different tech stack, than the one that I will be working on. • IoT devices run proprietary software and replacing it would require substantial amount of time.	• Virtualise the environment, without having to physically put software on any hardware. • Simple container with minimal Linux Image should be representative enough. • Alternatively, purchase a new Raspberry Pi.	Medium
• Ethereum doesn't meet my needs or understanding it takes substantial amount of time. • I face a block relating to knowledge of Ehtereum effectively stopping me from reaching my goals.	• Switch to an implementation that I am more familiar with, although with slightly different tradebacks, that is, Hyperledger Fabric.	Medium
• Unexpected hardware failure or theft. • Any hazards resulting from the natural wear and tear of utilized hardware	• All code, documentation and report should be frequently backed up and versioned using git (and thus, uploaded to GitHub), allowing for data recovery in case of a disaster. • If LoRaWAN gateway or other IoT equipment was to become faulty or technologically infeasible, focus on virtualised solutions, as per first row.	Low

Timetable[4]



References

[1] D. Halperin, T. S. Heydt-Benjamin, B. Ransford, S. S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. H. Maisel. Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 129–142, May 2008.

[2] Milan Markovic and Peter Edwards. Enhancing transparency of mqtt brokers for iot applications through provenance streams. In *Proceedings of the 6th International*

[4]Live version can be seen at `https://bit.ly/2RTiRxV`

Workshop on Middleware and Applications for the Internet of Things, M4IoT '19, page 17–20, New York, NY, USA, 2019. Association for Computing Machinery.

[3] N. Klaokliang, P. Teawtim, P. Aimtongkham, C. So-In, and A. Niruntasukrat. A novel iot authorization architecture on hyperledger fabric with optimal consensus using genetic algorithm. In *2018 Seventh ICT International Student Project Conference (ICT-ISPC)*, pages 1–5, July 2018.