

BUCHAREST UNIVERSITY OF ECONOMIC STUDIES

Cybernetics, Statistics and Economics Informatics Faculty

Department of Economics Informatics

**DISSERTATION**

Scientific Coordinator

Assoc. Prof. Cristian-Valeriu TOMA, Ph.D.

Graduate

Andrei-Robert CAZACU

Bucharest

2022

Logo

Description automatically generated

BUCHAREST UNIVERSITY OF ECONOMIC STUDIES

Cybernetics, Statistics and Economics Informatics Faculty

Department of Economics Informatics

Secure IoT solution for office building monitoring

DISSERTATION

Scientific Coordinator

Assoc. Prof. Cristian-Valeriu TOMA, Ph.D.

Graduate

Andrei-Robert CAZACU

Bucharest

2022

**TABLE OF CONTENTS**

[**1.** **INTRODUCTION** 4](#_Toc101966171)

[**2.** **TECHNICAL AND MATHEMATICAL CONCEPTS** 6](#_Toc101966172)

[**2. 1. What is Internet of Things?** 6](#_Toc101966173)

[**2. 2. Cryptography** 6](#_Toc101966174)

[**2. 3. Blockchain** 6](#_Toc101966175)

[**2. 4. MQTT** 6](#_Toc101966176)

[**2. 5. CoAP** 6](#_Toc101966177)

[**2. 6. Cloud computing** 6](#_Toc101966178)

[**2. 7. Machine Learning and TinyML** 6](#_Toc101966179)

[**3.** **PROPOSED SOLUTION** 7](#_Toc101966180)

[**3. 1. High Level Solution Description** 7](#_Toc101966181)

[**3. 2. Bill of Materials** 7](#_Toc101966182)

[**3. 3. In-depth explanation of the solution** 7](#_Toc101966183)

[**4.** **CONCLUSION** 8](#_Toc101966184)

[**REFERENCES** 9](#_Toc101966185)

# **INTRODUCTION**

Despite IoT being in the spotlight for several years now, Gartner still predicts a five time increase in number of devices from 2018 to 2028 [1], up to 1.9 billion units, which further intensifies the need for security among IoT nodes. Through the years, a significant amount of effort has been spent standardizing and improving inter-operability among devices, spawning several lightweight application layer protocols such as CoAP [2], standard formats for data during transit such as IPSO objects [3], or means to encode data that further reduces the size of the payload such as CBOR [4]. While Internet of Things wasn’t officially a concept until 1999, one of the first examples of such a device surfaced in the early 1980s, an internet connected Coca Cola vending machine placed at the Carnegie Mellon University which local programmers modified to report whether a drink was available, and if it was cold or not before making the trip [5]. The phrase “Internet of Things” was coined by Kevin Ashton, then Executive Director of Auto-ID Labs, during a presentation he made for Procter & Gamble.

Recently, great efforts have been poured into securing IoT devices, empowered by the rapid advancements in technology which allows complex cryptographic operations to be executed on device in a reasonable amount of time. While solutions based on existing cryptographic algorithms have recently surfaced, such as Connectivity Standards Alliance’s Matter [6], which relies on the Public Key Infrastructure model, new lightweight cryptographic algorithms are being developed in an attempt to make security a thing for even the smallest of devices. This pursuit is fuelled by NIST which initiated a process to solicit, evaluate, and standardize lightweight cryptographic algorithms in August 2018 [7].

The purpose of this paper is to combine existing and emerging technologies into a polished IoT product that can be used for office building monitoring, but still be expandable to suit other needs such as smart cities, or smart homes. This modularity is baked into the architecture of the product, making no assumption about the type of the data sent or received, nor about the type of sensors attached to the node. Another imposed constraint was developing a security model that would be able to ensure the confidentiality of the payload without requiring a gateway, leading to greater computational effort that has to be placed upon the IoT node, with the benefit of being able to push the data directly into the cloud and having almost zero configuration needed. While this architecture may not confine to previously agreed standards of having several nodes connected via Intranet to a gateway which then pushed data into a public or private cloud, big names in the cloud computing industry such as Amazon Web Services, Microsoft Azure and Oracle have started shipping solutions that cater to such kind of cloud connected nodes. Solutions have also appeared in the hobbyist market, such as Arduino Cloud, which shows the industry wide trend of IoT edge computing and edge AI.

For clarity, this dissertation has been structured into four chapters:

1. Introduction
2. Technical and Mathematical Concepts, where concepts fundamental to this dissertation are discussed
3. Proposed solution, where the architecture and the implementation are discussed
4. Conclusions, a chapter which discusses the pros and cons of the designed solution

# **TECHNICAL AND MATHEMATICAL CONCEPTS**

## **2. 1. What is Internet of Things?**

## **2. 2. Cryptography**

## **2. 3. Blockchain**

## **2. 4. MQTT**

## **2. 5. CoAP**

## **2. 6. Cloud computing**

## **2. 7. Machine Learning**

# **PROPOSED SOLUTION**

## **3.** **1. High Level Solution Description**

A single IoT node was built as a proof of concept to demonstrate the feasibility and power of the solution.

The node features an ESP32 which has several sensors attached to it, such as DTH11 temperature and humidity sensor, SW-420 vibration sensor, and MQ-2 gas sensor. While this may not provide an exhaustive list of sensors that can be attached to the node, it serves the purpose of demonstrating the flexibility with which the architecture handles different types of data.

Before pushing any data to the cloud, the device needs to have the Wi-Fi connection configured, but also to have its identity attested by the attestation server. This is done by using public key cryptography. At manufacture time, three files are burnt into the flash storage of the device: the root certificate and device certificate which has both a public and a private key. Impersonation of the device is avoided by encrypting the flash storage before shipping the device to the end-user, making the key uncompromisable. This is done by leveraging a built-in feature of the ESP32 which transparently handles encryption and decryption of data on the fly, saving the used AES key in eFUSE [8].

Following the attestation of the node, a symmetric key has been established which will then be burnt into the encrypted flash storage for use the next time the node will be restarted. The integrity of the key is assured by using Elliptic Curve Digital Signature Algorithm (ECDSA), signed with the device’s own private key. This key will expire each 24 hours, which ensures forward secrecy if the key is compromised. Following the expiration of the key, it will be removed from non-volatile storage alongside its signature, and the node restarted to perform the attestation process again.

The symmetric key is used in establishing MQTT connection in TLS-PSK mode, which is less computationally expensive than TLS with public key cryptography. This means that the cost of establishing a key is incurred only once each 24 hours when attesting the origin of the device.

The payload will be CBOR encoded and formatted according to the IPSO guidelines and then sent to the MQTT broker in the cloud. From here, the possibilities for data manipulation are endless, enabling functional style programming, subscribing to each topic that is of interest and performing actions when data is received, or persisting the data in a relational or non-relational database for later use.

For this proof of concept, besides the previously stated components, a cloud hosted clustered MySQL setup will be used, enabling fast responses while creating no bottleneck for simultaneous data insertions and retrievals. Two microservices have been developed using the Spring Boot framework, one that subscribes to all the available topics of devices that have undergone the attestation process and persists the information into the database, and another one that performs queries which are exposed via a HTTP REST interface.

## **3.** **2. Bill of Materials**

Table 1. Bill of materials

|  |  |  |  |
| --- | --- | --- | --- |
| COMPONENT | UNIT PRICE(LEI) | QUANTITY | TOTAL PRICE |
| ESP32 DEV BOARD | 42.99 | 1 | 42.99 |
| DTH11 TEMP AND HUMIDITY SENSOR | 11.99 | 1 | 11.99 |
| SW-420 VIBRATION SENSOR | 4.89 | 1 | 4.89 |
| MQ-2 GAS SENSOR | 12.49 | 1 | 12.49 |
| BATTERY PACK | 109.99 | 1 | 109.99 |
| TOTAL | 182.35 | | |

1. ESP32 Development Board

ESP32 is a series of low-powered system on a chip microcontrollers featuring Wi-Fi and Bluetooth, developed by Espressif Systems, a Shanghai-based Chinese company, and manufactured by TSMC using their 40nm node.

The particular flavour used here is ESP32S onto a ESP-WROOM-32 derived development board. This features a dual core Tensilica LX6 clocked at either 160 or 240MHz with a 32bit architecture, has an Ultra-Low Power coprocessor, 520KB of RAM and 4MB of flash storage that can be partitioned to the user’s preference. It also features Wi-Fi 802.11 b/g/n and Bluetooth 4.2 with BLE support.

This board is used as an IoT node and performs cryptographic operations, sensor reading and sends data over MQTT to the broker hosted in a cloud instance.

1. DTH11 Temperature and Humidity Sensor

DHT11 is a temperature and humidity sensor that output a digital signal on the data pin. Temperature is measured using a negative coefficient thermistor (NTC) and the relative humidity is measured using a capacitive sensor. The humidity sensing range can be between 20 and 90 % RH and its measurement accuracy is of +/- 5% RH. The temperature measurement range is from 0 to 60◦C and its accuracy is of +/- 2◦C. The supply voltage accepts ranges between 3.3V and 5V.

1. SW-420 Vibration Sensor

The vibration sensor used has a LM393 comparator and uses the SW-420 vibration sensor module that is normally closed, outputting a digital signal on the data pin. Under normal circumstances, the vibration switch is under closed conduction state, outputting digital low. Under strong vibrations, the sensor will output high. It has operating voltage ranging from 3.3V to 5V.

1. MQ-2 Gas Sensor

MQ-2 gas sensor is highly response and very sensitive, capable of detecting gases such as: butane, liquefied petroleum gas, propane, methane, alcohol, smoke, hydrogen, and other harmful gases.

It is based on a Metal Oxide Semiconductor type sensor, also known as chemiresistors, as the detection is based upon change of resistance of the sensing material when the gas meets the material. This sensor has an operating voltage of 5V.

1. Battery pack

The battery pack provides power to the IoT node, so it can be placed where there are no wall plugs.

## **3. 3. In-depth explanation of the solution**

For clarity, this solution can be split into two parts, the IoT node and the cloud applications.

Diagram

Description automatically generated

Figure 3‑1. High Level Architecture Overview

The IoT node is composed of an ESP32 with several sensors connected to it using a breadboard. When designing the node, careful planning had to be done regarding the processing power, memory capacity and dimensions while still keeping the costs as low as possible in order to provide a marketable solution. While using a Raspberry Pi for example would have rendered faster processing times and easier development, the footprint of the node would be bigger, but the price would be the real deterrent, ballooning the cost at least 5 times. Since this is a price elastic market [9], such a device would need an attractive price in order to gain market traction.

After sourcing the node, another important decision had to be made regarding the development stack used. While the OS of the device is FreeRTOS, the manufacturer offers two possible abstractions on top: be it Arduino framework, or their own ESP-IDF (Espressif Integrated Development Framework). The difference is significant, Arduino offering an OOP paradigm along with a huge amount of community created libraries, while ESP-IDF is task oriented and has manufacturer developed libraries, along with a tight integration with the hardware and access to low level functions. The flip side is that both use a modified version of the GCC (GNU Compiler Collection), offering some level of portability. This proved to be very important, since back and forth movement was performed during the development process as issues arose.

The solution started as an Arduino project, but the security-sensitive aspect of the solution proved to be impossible to implement without the tight knit integration with the hardware of the ESP-IDF. This is due to the fact that the Arduino abstraction is shipped as a binary distribution, precompiled with the settings the Espressif deemed most relevant while not taking up unnecessary space in the non-volatile memory, while the ESP-IDF allows to build from source using a highly configurable architecture. The configurability is achieved by using a custom build system based partly on CMake, with compile time settings being altered by using the *menuconfig* [10]. This switch allowed free access to all hardware features, such as the transparent runtime encryption of flash memory needed in order to fill in the security puzzle, being able to ensure the confidentiality of the files stored in flash non-volatile memory. Another reason for the switch was the absence of the PSK option from the Arduino binaries, needed to smoothen the connection process to the broker. While this solved most of the issues, drawbacks were present such as the loss of important Arduino libraries that were used. While they were written in C++, and with enough effort any code can be built with any build system, dependencies of the Arduino framework were baked into every third-party library. Even though the dependencies could be removed, be it rewriting the third-party library or writing new code from scratch, time is of the essence, being one of the most valuable resources. This is where the flexibility of the ESP-IDF build system came into play, tying in the whole solution together, allowing the compiling of the Arduino sources and linkage with the other components. Each Arduino library still had to be integrated with the CMake based build system, but it required significantly less effort than writing the required functionality from scratch.

With the underlying components detailed, attention can now be shifted to the functional part of the product.

The node features an ESP32 which has several sensors attached to it, such as DTH11 temperature and humidity sensor, SW-420 vibration sensor, and MQ-2 gas sensor. While this may not provide an exhaustive list of sensors that can be attached to the node, it serves the purpose of demonstrating the flexibility with which the architecture handles different types of data.

# **CONCLUSION**

# **REFERENCES**

|  |  |
| --- | --- |
| [1] | MMH, “Gartner’s Hype Cycle: IoT in “trough” but transformational stage on the way,” 9 September 2020. [Online]. Available: https://www.mmh.com/article/gartners\_hype\_cycle\_iot\_in\_trough\_but\_transformational\_stage\_on\_the\_way. |
| [2] | IETF, “RFC 7252 - The Constrained Application Protocol (CoAP),” [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7252. |
| [3] | Hewlett Packard Enterprise, “IPSO Object Reference Guide,” [Online]. Available: https://techlibrary.hpe.com/docs/otlink-wo/IPSO-Object-Reference-Guide.html. |
| [4] | IETF, “RFC 7049 - Concise Binary Object Representation (CBOR),” [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7049. |
| [5] | K. D. Foote, “A Brief History of the Internet of Things,” [Online]. Available: https://www.dataversity.net/brief-history-internet-things/#. |
| [6] | Connectivity Standards Alliance, “Build With Matter,” [Online]. Available: https://csa-iot.org/all-solutions/matter/. |
| [7] | NIST, “Lightweight Cryptography,” [Online]. Available: https://csrc.nist.gov/Projects/lightweight-cryptography. |
| [8] | K. Sovani, “Understanding ESP32’s Security Features,” [Online]. Available: https://blog.espressif.com/understanding-esp32s-security-features-14483e465724. |