# CRELSA: property control access based on the internet of things

TIAGO DA SILVA ALMEIDA, LEONARDO REZENDE COSTA
WARLEY GRAMACHO DA SILVA, RAFAEL LIMA DE CARVALHO
Universidade Federal do Tocantins
Department of Computer Science
Avenida NS-15, ALCNO-14, Palmas / TO
BRAZIL
tiagoalmeida@uft.edu.br, leonardorec1@gmail.com
wgramacho@uft.edu.br, rafael.lima@uft.edu.br

*Abstract:* This paper presents the development of an electornic system for access control of smart buildings. As a testbed, this system is proposed to work in the Federal University of Tocantins, campus of Palmas, in order to automatically grant access to authorized professors only, under reservation criteria. Currently, the access is done by requiring the presence of security staff which overwhelms their activities. In our proposal, all reservation and access information is decentralized to a working server. However, our tests were done in a decoupled way from the server to allow generalization of the proposal for the automation of any building. Another important point is the simplicity of the proposal to minimize problems of incompatibility between solutions. As result, we show that even with very low-cost components and parts it is possible create smart solutions to daily problems.

*Key–Words:* Arduino, RFID, IoT, Access Control, Door Lock

## 1 Introduction

Currently we are experiencing a phase of popularization of projects involving some kind of electronic automation. This is possible due to the low integration costs of a coupled circuits (or within the same encapsulation, so-called System-on-Chip or SoC only), bringing simplicity to the construction of automatic systems. Futhermore, a research area appeared, called Internet of Things (IoT), along with the also simplified area of Web development.

The IoT is an emerging market that will generate US$14.4 trillion in value [10]. From an industry perspective, four industries make up more than half of the value. These four industries include: manufacturing at 27%; retail trade at 11%; information services at 9%; and finance and insurance, also at 9%. Other industries such as wholesale, healthcare, and education lag behind in terms of value generation, with a range between 1% and 7% [10]. This data show a big grow and popularization in IoT field.

This popularization at great speed also brings important problems that need to be taken into account, such as *network overhead* - due to the large number of connected devices; *data security* - it is necessary to evaluate whether the web services used allow the data to be integrated, such as the impossibility of data theft; and *update* - the speed with which these systems evolve is very large, and effective and more automated upgrade forms are necessary [9].

Another problem arising from the popularization is the lack of standardization of IoT-based systems. Evidenced fact in the papers of [9] and [1]. The lack of standardization is so important that it goes from the logical architecture of the system as well as the physical architecture. Thus, different vendor solutions can not interact with each other.

In this way, this paper aims to development a simplified and low-cost electronic system for access control and reserves of laboratories of the Federal University of Tocantins.

As related work, we cited [13]. This paper proposed a framework to incorporate different components from IoT proposed in the literature, in order to efficiently integrate smart home objects in a cloud-centric IoT based solution. [8] proposed a system of smart-home wherein the day to day household activities are controlled by a wireless RF sensor. The control activity is based on the detection of existence of human being inside any specific area of the home. The confirmation of the human existence is based on the detection of human respiration rate and heartbeats.

[6] proposed mechanism supports rights delegation and a more sophisticated access control customization and described a capability based access control system that enterprises, or even individuals, can use to manage their own access control processes to services and information. [11] proposed certificate-

less signcryption scheme and then design an access control scheme for the IWSNs (Industrial Wireless Sensor Networks) in the context of the industrial IoT using the certificateless signcryption.

### 1.1 Description of Problem

Among the huge range of applications, access control to sites is an important and common branch in IoT. Hardly any place has unrestricted access from anyone. The Federal University of Tocantins, has this problem and unfortunately there is not an adequate management of access to the places frequented by professores, students and administrative technicians.

This project, entitled CRELSA (*Controle de Reservas Eletrônica Laboratoriais, Segurança e Acesso*), use of low-cost electronic components and implementation, to facilitate the development, maintenance and expansion of the project in a later manner by other professors and students, when necessary.

The Arduino microcontroller platform was used, because it is a platform also with ample documentation. The Arduino was developed in Italy, and it is an environment composed of microcontroller on a printed circuit board, integrated with the recorder and cross compiler for programming.

The cross-compiler allows software development to be done on a conventional computer using the C/C++ language. Subsequently, this compiler generates the Assembly required for recording and operating the microcontroller.

Currently, the university has a system, developed in the own campus, to control the reserves of rooms and laboratories for teaching. This system is hosted on its own server and already has the mapping of the rooms and laboratories available and the schedules of the courses. So, each module installed in the rooms will make a connection with this system to check the user's availability to access a certain location. The access of the user to the site is authorized only upon compliance with the university room allocation system.

The CRELSA access interface was made using RFID (Radio-Frequency IDentification) tags. Each RFID tag has a unique identifier and is manufactured and shaped like a keychain, stickers, cards etc. This feature allows it to be used instead of a password and individual login, used in online system, for example.

There are already several access control systems that use RFID, for this reason, the maintenance and documentation for its use becomes easier. There is also the need to use electronic locks on each door. This lock is also part of CRELSA and there are several low-cost commercial models to implement the proposed solution.

Another important point in relation to the proposed control is decentralization. Since each port represents only a high-security access interface, only to validate the user's information on the server hosted at the university itself.

The rest of the text is organized as follows: Section 2, presents all the components involved in the development of the proposed prototype, as well as its configurations and how they were interconnected; Section 3 is explained as the tests were performed and finally Section 4 presents our conclusions and final remarks.

## 2 Proposed solution

The proposed solution was constructed according to Figure 1. The circuit is composed of the Arduino UNO R3 [3] platform as a processing module, Ethernet Shield as a connection module to the server (based on the Message Queuing Telemetry Transport protocol - MQTT), a Liquid Crystal Display (LCD) with an Inter-Integrated Circuit ($I^2C$) module for visual presentation with the user, RFID reader for user validation and a solenoid-type electric lock door of 12 V to open the lock.

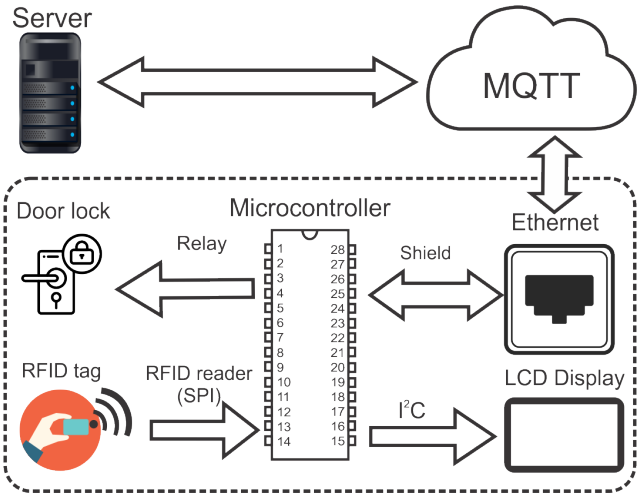


Figure 1: General schematic diagram of the CRELSA project, the dotted rectangle represents the proposed circuit and the arrows are the means of communication with the inegrant parts.

### 2.1 Microcontroller

It is important to understand that the Arduino board includes a microcontroller, and this microcontroller is the one that executes the instructions of the program. The ATmega328 microcontroller is the microcontroller used in the Arduino UNO R3. ATmega328

is an AVR family microcontroller and has 8-bits word, which means that its data bus architecture and internal registers are designed to handle 8 parallel data signals [4].

The ATmega328 has three types of memory:

- Flash Memory: 32 KB of non-volatile memory used to store applications.
- SRAM memory: 2 KB volatile memory used by the application while it is running.
- EEPROM memory: 1KB of non-volatile memory that can be used to store data that must be available even after the board is turned off and then turned on again.

This microcontroller has three data ports: PORTB, PORTC and PORTD. Some pins that can be configured as PWM (Pulse-Width Modulation) output, these pins are marked with " " on the Arduino board.

Another example, pin0 for pin5 of PORTC can be ADC (Analog to Digital Converter) inputs with 10-bit resolution instead of digital I / O and there are others pins for ADC converter operation, which are:

- AVCC: The power pin for the ADC unit.
- AREF: The input pin optionally used as external voltage reference ($V_{ref}$) for ADC instead of internal $V_{ref}$.

The ATmega328 has only one UART module (Universal Asynchronous Receiver / Transmitter), which is a serial interface. The UART (RX, TX) pins are connected to a USB-to-UART converter circuit and also connected to pin0 and pin1 in the digital connector. Use of the UART should be avoided when USB data sending / receiving is already in operation [4].

The ATmega328 has only one SPI (Serial Peripheral Interface) module, another serial interface. In addition to using it as a serial interface, it can also be used to program the microcontroller using an independent programmer.

## 2.2 $I^2C$ module

The $I^2C$ LCD module is ideal for use in projects involving LCDs, and may be present in projects with any microcontrollers that support the $I^2C$ protocol where there are several devices (not necessarily LCDs) that will communicate between themselves with only two rows of data.

The $I^2$ C protocol was developed to connect several devices using only the two data pins mentioned SDA (Serial Data) and SCL (Serial Clock). The goal is to set a hexadecimal address for each device and at the moment of communication only the requested device will respond [14]. The module used has the following components:

- DIP SWITCH: Sets the address of your device (it can go from 0x20 to 0x27 in hexadecimal).
- Trimpot (potentiometer): To quantify the brightness of the LCD Display Backlight
- Latch (or IDC) connectors: Using M / F or F / F Jumpers you can connect the module to both your Master and its Slaves.

The position and configuration of the DIP SWITCH for addressing of the module $I^2C$ is shown in Table 1.

Table 1: Map of positioning and configuration of the addresses available in the $I^2C$ communication module.

| Address | DIP Switch | | |
|---|---|---|---|
| | 3 | 2 | 1 |
| 0x20 | ↑ | ↑ | ↑ |
| 0x21 | ↑ | ↑ | ↓ |
| 0x22 | ↑ | ↓ | ↑ |
| 0x23 | ↑ | ↓ | ↓ |
| 0x24 | ↓ | ↑ | ↑ |
| 0x25 | ↓ | ↑ | ↓ |
| 0x26 | ↓ | ↓ | ↑ |
| 0x27 | ↓ | ↓ | ↓ |

The Figure 2 shows the schematic connection diagram of the Arduino UNO R3 and the communication module $I^2$ C with the LCD display 16x2. This display was used for its low cost.

## 2.3 RFID reader

The RFID reader has a serial communication with the SPI protocol (more details on the operation of the SPI protocol can be found in [2]) with the following specifications:

- Working current: 13-26mA / DC 3.3V
- Idle current: 10-13mA / 3.3V
- Sleep Mode Current: $< 80\mu$A - Pico de corrente: $< 30$mA
- Operating Frequency: 13,56MHz

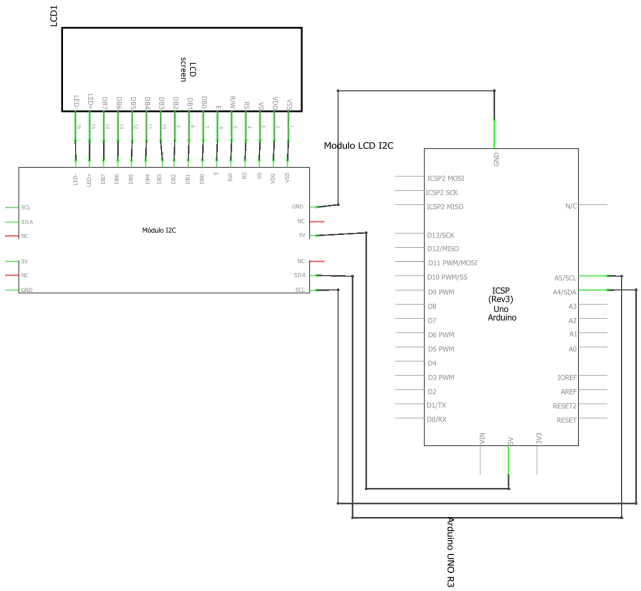


Figure 2: Schematic diagram of the connection between the Arduino UNO R3 and $I^2C$ module with LCD display.

- Card types supported: Mifare1 S50, S70 Mifare1, Mifare UltraLight, Mifare Pro, Mifare Desfire
- Transfer rate: 10 Mbit/s

Figure 3 illustrates the schematic diagram of how the RFID card reader module with SPI protocol and Arduino UNO R3 were connected.

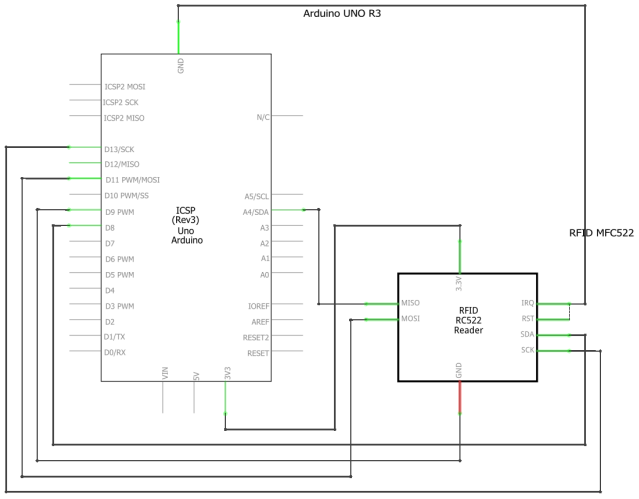


Figure 3: Schematic diagram of the connection between the Arduino UNO R3 and the RFID reader module with SPI interface.

## 2.4 Ethernet connection

Figure 3 illustrates the schematic diagram of how the Ethernet Shield W5100 for Arduino and Arduino UNO R3 were connected. The Ethernet Shield W5100 is based on the Ethernet chip of the WIZnet Ethernet W5100 that provides Internet Protocol (IP) or Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) protocols and also communicates with the Arduino UNO R3 using protocol SPI [15].

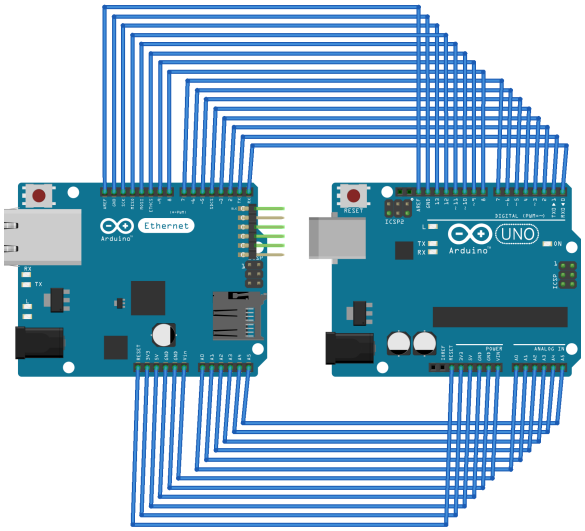

Figure 4: Arduino UNO R3 Connection Block Diagram with Ethernet Shield W5100 for Arduino.

## 2.5 Door lock

Since the chosen lock is a 12 V solenoid-type electrical lock, a simple relay type solves the problem of door actuation, as shown in Figure 5.

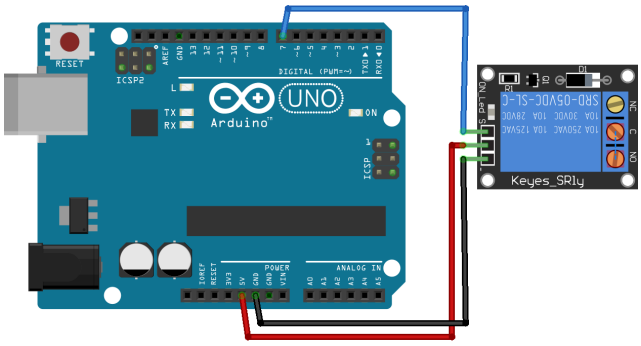


Figure 5: Connection block diagram of the Arduino UNO R3 with the relay for activating the solenoid type electric lock.

## 2.6 Programming

The programming follows the flowchart of Figure 6. As the Arduino UNO R3 programming is based on C/C++, the code is simplistically divided into routines. The program waits for the reading of an RFID tag and displays a welcome message from the display. When the contact is opened an internet connection is sent and a message is sent to the broker with the identifier of the tag. At this moment he is waiting for a return. The broker communicates with the server and verifies the user's access permission. At that moment, the server and how the data are organized are no longer relevant thanks to the broker's performance (MQTT server).

Full explanation of the MQTT protocol is outside the scope of this paper and more details are available in [7].

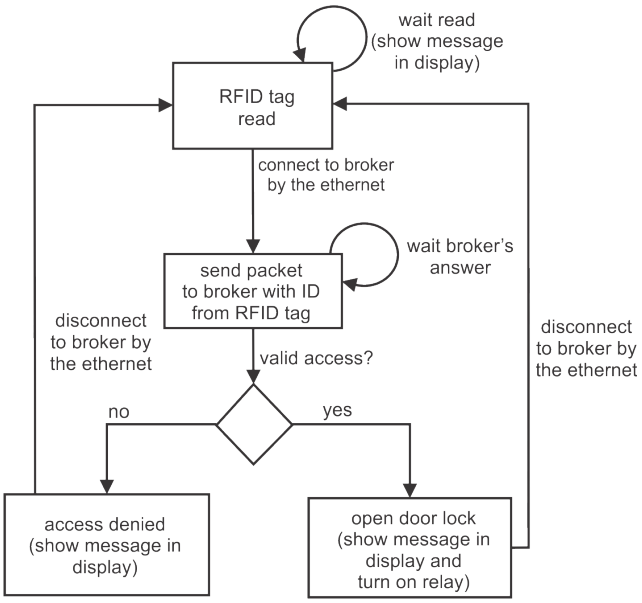


Figure 6: Flowchart of the program executed in Arduino UNO R3.

The broker's response is sent back to the system, simply following a released or unreleased data field. In case of release the relay is triggered and it is unlocked from the door. In both responses, whether access or not, messages are displayed to users through the display.

# 3 Tests architectures

One point worth mentioning is that in our university we did not have access to the application server with the room allocation system. This did not prevent the development of a prototype. For simulation of the application server, the Python language and an server were made executing the code, as the list below:

```
1  import paho.mqtt.client as paho
2
3  # Define event callbacks
4  def on_connect(mqttc, obj, flags, rc):
5  print("rc: " + str(rc))
6
7  def on_message(mosq, obj, msg):
8  print(msg.topic + " " + str(msg.qos) + " "
       + str(msg.payload))
9  if msg.topic == 'CRELSA/ID':
10 if str(msg.payload) == "b' 53 40 95 35'":
11 mqttc.publish("test", "free")
12 else:
13 mqttc.publish("test", "denied")
14
15 def on_publish(mosq, obj, mid):
16 print("mid: " + str(mid))
17
18 def on_subscribe(mosq, obj, mid,
       granted_qos):
19 print("Subscribed: " + str(mid) + " " +
       str(granted_qos))
20
21 def on_log(mosq, obj, level, string):
22 print(string)
23
24 mqttc = paho.Client()
25 # Assign event callbacks
26 mqttc.on_message = on_message
27 mqttc.on_connect = on_connect
28 mqttc.on_publish = on_publish
29 mqttc.on_subscribe = on_subscribe
30
31 # Connect
32 mqttc.connect('10.10.31.183', 1883)
33
34 # Start subscribe, with QoS level 0
35 mqttc.subscribe("CRELSA/#", 0)
36
37 # Continue the network loop, exit when an
       error occurs
38 rc = 0
39 while rc == 0:
40 rc = mqttc.loop()
41 print("rc: " + str(rc))
```

The code itself is simple, importing the package to create an MQTT client (line 1), opening the connection (line 4), checking the message received by the Arduino UNO R3 (line 7), publishing the private message to the Arduino UNO module R3 (line 15), subscription of messages to broker (line 18), generation of log files (line 21), signing of methods (lines 24 to 29), connection establishment (line 32), start of server execution (lines 39 and 40).

The code base for creating the test server is available in [5] and the broker with the MQTT protocol was installed using the Mosquitto, it is available in [12].

Since this work is very technical, all prototype tests worked properly, even for the simplicity of the proposal.

# 4 Conclusion

The good functioning of the prototype demonstrates that the solution and the use of the most common protocol in IoT allows the generalization of the proposal for the property control in any environment. Even the tests were discharged from the use of a real university server.

Another point to emphasize is simplicity of the implementation, is made with few lines of code and little use of the hardware of the microcontroller. It is reasonable to conclude that IoT protocols and methodologies considerably reduce effort in building automation projects like this. Even with very low-cost components and parts, such as the Arduino platform and its connection and expansion Shields. This shows that even budget constrained sites can create smart solutions to everyday problems.

The next and most important step in relation to this project is the implantation of the prototype in some environments for use tests in real operating situations and later, if possible, implementation throughout the university.

**Acknowledgements:** The research was supported by the Federal University of Tocantins, Brazil.

*References:*

[1] S. A. Al-Qaseemi, H. A. Almulhim, M. F. Almulhim, and S. R. Chaudhry. Iot architecture challenges and issues: Lack of standardization. In *2016 Future Technologies Conference (FTC)*, pages 731–738, Dec 2016.

[2] Arduino Foundation. A brief introduction to the serial peripheral interface (spi). Website, 2017. https://www.arduino.cc/en/Reference/SPI.

[3] Arduino Foundation. Documentation. Website, 2017. https://store.arduino.cc/usa/arduino-uno-rev3.

[4] Atmel Company. 8-bit avr microcontrollers atmega328/p datasheet complete. Website, 2017. http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf.

[5] CloudMQTT. Python. Website, 2017. https://www.cloudmqtt.com/docs-python.html.

[6] Sergio Gusmeroli, Salvatore Piccione, and Domenico Rotondi. A capability-based security approach to manage access control in the internet of things. *Mathematical and Computer Modelling*, 58(5):1189 – 1205, 2013. The Measurement of Undesirable Outputs: Models Development and Empirical Analyses and Advances in mobile, ubiquitous and cognitive computing.

[7] HiveMQ. Everything you need to know about mqtt. Website, 2017. https://www.hivemq.com/mqtt-essentials/.

[8] Brijesh Iyer, N. P. Pathak, and D. Ghosh. Rf sensor for smart home application. *International Journal of System Assurance Engineering and Management*, 9(1):52–57, Feb 2018.

[9] P. Kansakar and A. Munir. Selecting microarchitecture configuration of processors for internet of things (iot). *IEEE Transactions on Emerging Topics in Computing*, pages 1–1, 2018.

[10] In Lee and Kyoochun Lee. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, 58(4):431 – 440, 2015.

[11] Fagen Li, Jiaojiao Hong, and Anyembe Andrew Omala. Efficient certificateless access control for industrial internet of things. *Future Generation Computer Systems*, 76:285 – 292, 2017.

[12] G. Mathew. Setting up mqtt mosquitto broker in ubuntu linux. Website, 2017. https://www.wingsquare.com/blog/setting-up-mqtt-mosquitto-broker-in-ubuntu-linux/.

[13] Biljana L.Risteska Stojkoska and Kire V.Trivodaliev. A review of internet of things for smart home: Challenges and solutions. *Journal of Cleaner Production*, 140:1454 – 1464, 2017.

[14] J. Valdez and J. Becker. Understanding the i2c bus. Website, 2017. http://www.ti.com/lit/an/slva704/slva704.pdf.

[15] WIZnet Co. W5100 datasheet version 1.1.6. Website, 2017. https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf.