5-9-2016

Wireless Authentication of Smart Doors Using RFID

Austin A. Hentrup
Loyola Marymount University, ahentrup@gmail.com

Deyi Lu
Loyola Marymount University, sixgetone@gmail.com

Peter R. Roldan
Loyola Marymount University, proldan@lion.lmu.edu

LMU | LA
University Honors Program

Wireless Authentication of Smart Doors Using Radio Frequency Identification (RFID)

A thesis submitted in partial satisfaction

of the requirements of the University Honors Program

of Loyola Marymount University

by

Austin Hentrup, Deyi Lu, and Peter Roldan

May 9, 2016

Table of Contents

Abstract

This final report outlines the implementation of an Internet-enabled door authentication system using radio frequency identification (RFID). The project was undertaken as a senior Electrical Engineering CAPSTONE project. RFID is an exciting technology that could have the potential to revolutionize convenience and security for consumers worldwide. This project served primarily as a means of investigating this potential, with a particular emphasis placed on controlling overall system cost while offering security. The importance of this consideration is in direct response to the perceived over inflation in price of professional RFID solutions for offices, businesses and universities. It is the position of the team undertaking this project that a system with many of the convenient features of a professional solution could be built for a small fraction of the cost.

The inexpensive system attempts to maximize the available features without sacrificing the low price point of approximately $78 per door. Each lock in the system is independently Internet-enabled to eliminate the need for a dedicated connection to a central network controller device. Instead, each lock is permitted to communicate directly over the Internet to a central cloud-based database server, provided by Amazon Web Services. This server is maintained by a door administrator who is able to use the web interface to manage user access profiles, remotely disable lost RFID cards, view door access logs and suspicious activity, and easily send messages to other administrators within the system. The web interface for the door administrator allows for advanced system control without the need for expensive proprietary software, advanced knowledge of the system's architecture, or the high installation and maintenance costs associated with dedicated communication wiring.

Security is also an important consideration that was addressed in this project. AES-128 CBC encryption has been implemented to enhance the resilience of communication between each RFID module and the central server against man-in-the-middle attacks. Of particular concern is the interception of RFID card ID numbers being transmitted in the clear when the RFID module attempts to poll the online database, and when the database sends a positive authentication message to the RFID module in order to open the door. It is on these two communication branches that encryption would be applied. Additionally, the system has been insulated against denial of service attacks through the use of free CloudFlare services, which dynamically blacklist IP addresses that attempt to hog server resources through excessive http requests.

I. Introduction

In an increasingly interconnected digital world, the traditional metal lock-and-key method of security is becoming more outdated and inconvenient compared to the flexibility, enhanced security, and reduced long-term cost that modern access control solutions have to offer. The trend is to be smarter, faster, lighter, more secure and more integrated than before. With this in mind, the concept of carrying a heavy and bulky keyring around everywhere to potentially access dozens of doors is a century-old solution that is being applied to the current pace and various needs of our modern society today. In addition, metal keys can be easily lost or stolen which means that the conventional response to this costly mistake is to change the locks on all the doors to which that key or those keys can access and to reissue new, updated keys to all of the people whose access rights were affected due to the installation of a new lock.

Several solutions for door authentication that take advantage of modern technologies include magnetic stripe cards and integrated cell phone applications. However, none of them are capable of providing a comprehensive solution that is convenient, secure and economic. While magnetic stripe cards are inexpensive and used widely throughout the world, the private information encoded in the magnetic stripe card is exposed to physical damage and is more likely to cause unsuccessful reading of the information contained (after having taken the time to properly orient the card in the mag-stripe reader, of course).

Cell phone applications can also be used to control door locks and are often very difficult to be hacked into. However, one drawback is that maintaining applications on several different cell phone platforms can create additional development costs that may be passed down to the final customer. Additionally, Android fragmentation has made this problem even worse: currently, seven different Android versions are running on Android devices over the world, which means to develop an open lock software that is compatible for all Android devices is almost impossible.

II. Project Objective

a. Problem Statement

Doolan Hall is one of the major buildings of LMU's Electrical Engineering Department and encompasses two classrooms, two study rooms and approximately fifteen offices. It has been a long time since Doolan Hall was built, and a sizeable quantity of laptops, desktops, and other valuable property are still protected by locks with metal keys. These have been outdated since the university-wide implementation of the magnetic stripe card (OneCard) system more than a decade ago.

Metal keys have always been used for the locks in the Electrical Engineering department at LMU. Recently, certain faculty members have expressed the idea that the students could design a solution to overcome the disadvantages of metal keys which were discussed earlier. Dr. Barbara Marino had mentioned that her key ring for Doolan Hall has far too many keys on it, and it usually costs her a significant amount of time to find the proper key each time she wishes to open a door. She had also mentioned her concern about losing her

master key which has access to almost all electrical engineering department rooms, including the circuits lab where numerous and valuable equipment is stored and used by students almost everyday.

In order to eliminate the inconvenience of opening doors and alleviate faculty concerns about losing the master key, this project strived at creating an affordable and secure system that could be scaled up to the infrastructure level. LMU's OneCard system is one of the possible solutions, but with the cost per door ranging from $150-$3000, the OneCard system has proven itself prohibitively expensive. Additionally, opting to outsource the solution to the OneCard system would be a wasted opportunity for students in the electrical engineering department to use their skills, creativity, and sincere efforts to such a problem. This CAPSTONE project has allowed those opportunities this semester, and it has brought attention to solving a problem that can save the electrical engineering department money and assure faculty, staff, and students that their research and education can continue the next day, because their equipment and workspace is protected and ready for easy access.

b. Customer Requirement

With a price range of $150-$3000 per door for LMU's OneCard system, this project was able to beat that price and offer a more affordable solution to the electrical engineering department at the lower cost of $78 per door.

III. Proposed Solution

a. Trades Leading to Proposed Solution

Several of the discussed solutions for door authentication were compared and arranged into a decision matrix (See Table 1). This table organizes design considerations and places a weighting on each of them. It, then, shows the relative score for each criterion that each option has been assigned. Each of these scores were compiled into a final overall score for each option. As seen in the last row of Table 1, RFID scores the highest overall among the four considered options for door authentication. It was particularly attractive for its high convenience, which was the design criterion that received the most weight.

Although the cell phone app excelled greatly in range, this was not factored heavily into the design considerations, as the maximum desired range for reading an RFID tag had to be within 10 cm.

Table 1 Decision Matrix for Access Control Systems

		Metal Key	Swipe Card	RFID Tag	Cell Phone App
Installation & Maintenance	0.26	0.33	0.19	0.19	0.13
Security	0.26	0.12	0.38	0.23	0.26
Range	0.20	0.11	0.11	0.22	0.56
Convenience	0.27	0.15	0.18	0.45	0.22
Cost per user	0.14	0.08	0.31	0.23	0.38
Score		0.20	0.26	0.38	0.33

b. Technical Requirements

Current technical requirements demand the following of the proposed system:

- The system should not take longer than 2 seconds to open the door after an RFID card is read
- RFID cards should be capable of being read from a distance of at least 2 cm but not greater than 10 cm
- In the event of system failure, the system should still permit normal (manual) operation of the door
- The electronics of the door should be powered with the electrical supply available in the wall
- Each door's RFID module should be capable of connecting to the Internet, either through Wi-Fi or Ethernet
- The dimensions of the system's circuitry enclosure should permit the module to be installed within the wall. If the customer objects to installing the system within the wall, the enclosure should allow for the system to be mounted on the wall beside the door on the inside of the room

c. System Description

An RFID card receives its power wirelessly from the RFID reader located at the door. This allows the RFID card to wirelessly transmit its identification information to the reader. The reader then passes this information to the Atmega328P microcontroller of the Arduino Uno

R3.

Next, the RFID unique identifier (UID) is encrypted in Javascript by Node.js which is run by the Linux operating system, OpenWrt, on the Yún shield. After the encrypted UID is sent to the server, the server decrypts it in order to compare it to the registered UIDs that are stored in the database. If the transmitted UID corresponds to a user who has access rights to the door that polled the server, the database sends an encrypted positive authorization message back to the RFID module which is first received by the Yún shield through the same process that made the request. After the authorization message is decrypted, the Uno can interpret the information and decide whether or not to unlock the electric strike plate. For example, a positive authorization message that was set as “202” would be interpreted by the microcontroller to send a logic 1 to the input of the NPN Darlington switch circuit which amplifies the current and unlocks the electric strike plate for an arbitrary amount of seconds, thereby allowing the door to be pushed open. After the preset number of seconds has passed, the electric strike plate will lock.

Figure 2 shows a higher-level diagram of the system, envisioned with multiple doors and the door system administrator’s workstation. Each door has an electric strike plate with the connected RFID module. The system administrator can manage the entire access control system from a single computer and even specify a single RFID module for analysis or upgrading over a secure shell (SSH) connection. The management GUI permits the ability to program new RFID cards, change user access profiles, remotely disable key cards, view door access logs and suspicious activities, and set temporary access schedules. The RFID module for programming new cards is nearly identical to the other RFID modules in the system except that it does not require a strike plate or the switch circuit - only the RFID reader, Arduino Uno, and Iduino Yún shield.

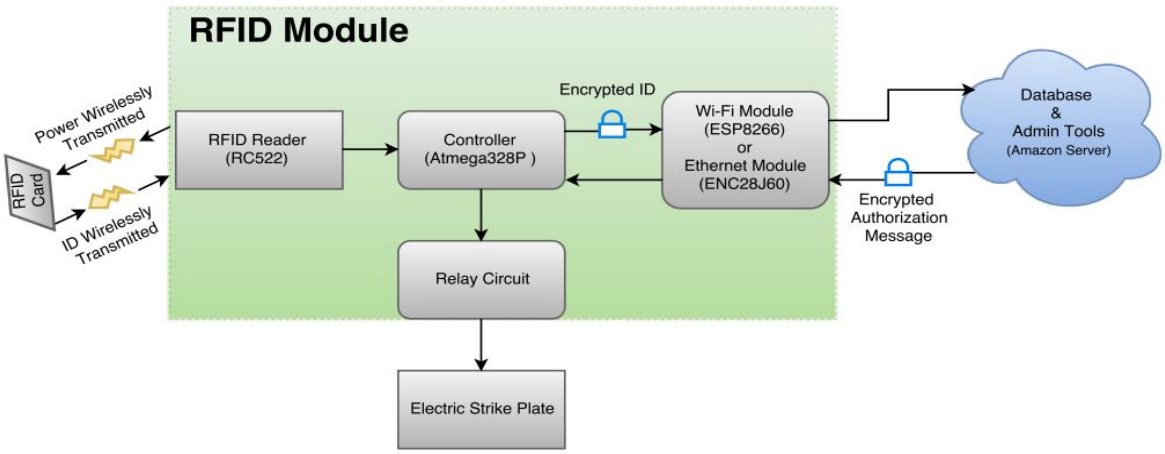

Figure 1 System Diagram

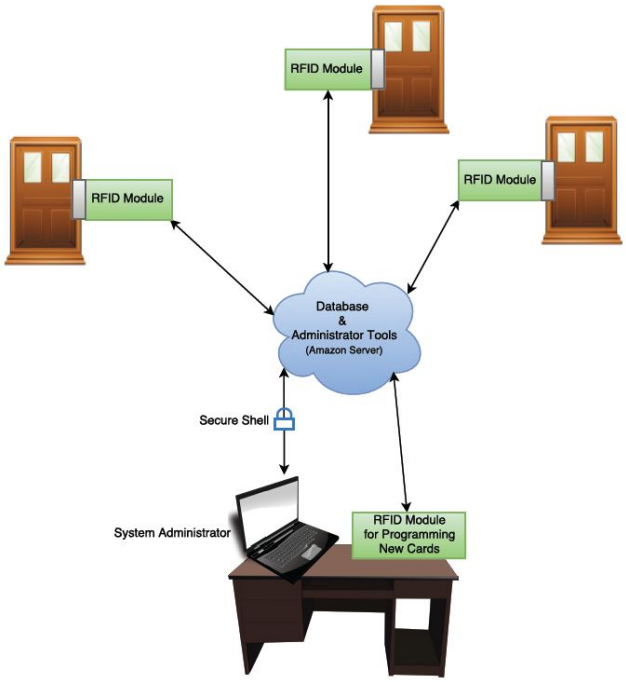

Figure 2 RFID System Diagram

d. Standards

Some of the IEEE standards associated with our project are Wi-Fi (802.11) and Ethernet (802.3). There are two types of Wi-Fi standards that this project adhered to: 802.11b and 802.11g. Their specifications are as follows:

I. IEEE 802.11b

A. Direct Sequence Spread Spectrum
B. Operates in the ISM band at 2.4 GHz in 5 MHz steps
C. Low power < 100mW; range < 100m
D. Designed for network operations
E. Bandwidth: 22 MHz; data rates up to 11 Mb/s

II. IEEE 802.11g

A. 2.4 GHz; up to 54 Mb/s
B. Applies to wireless LANs
C. Provides 20+ Mbps in the 2.4 GHz band.
D. Most recently approved standard
E. Wireless transmission over relatively short distances at up to 54 megabits per second (Mbps) compared with the 11 megabits per second of the 802.11b standard.
F. Operates in the 2.4 GHz range and is compatible with 802.11b, 802.11g

III. IEEE 802.3

A. The ethernet wiring standard used is 10BASE-T which applies to 10 Mbps at a maximum distance of 100 meters over a pair of twisted, unshielded wires that have RJ-45 connectors on both ends.

e. Constraints

i. **Economic** - The price of one device must not exceed more than $40, the final system including two devices and administrator control part must not cost more than $110

ii. **Environmental** - The unit should be able durable and be able to operate under everyday work conditions. The power consumption should also be kept minimal.

iii. **Manufacturability** - The final system must be modular and easy for BTC Solutions to reproduce.

iv. **Health and Safety** - The device should not be able to injure users or anyone around it.

v. **Lawful** - FCC regulations on RF transmission require less than 4 Watts of power be emitted from any unlicensed source.

IV. Electrical Design

a. RFID Module

The hardware setup of the online RFID authentication system includes the following components:

1. MFRC-522 reader for scanning RFID tags/cards
2. Arduino Uno R3 microcontroller
3. Iduino (generic brand) Yún shield with an embedded Linux server
4. Electric strike plate used to lock or unlock a door
5. NPN Darlington Switch Circuit to drive electric strike plate
6. LEDs to provide visual feedback of scan result to user

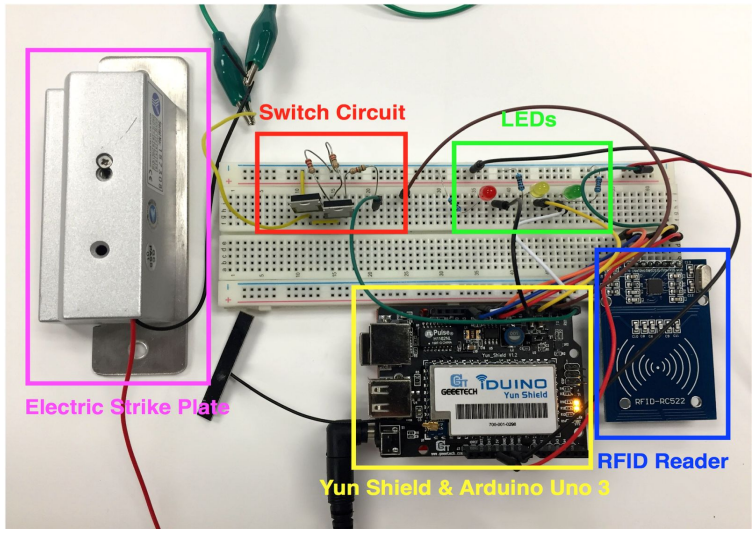

Figure 3 RFID Module

The actual picture of the RFID module is shown in Figure 3, the six different components listed above can be seen from the it. The Arduino Uno R3 microcontroller is inserted into the Arduino Yún shield so that Yún shield could provide a internet connectivity for Uno R3. In Figure 3, the Uno R3 sits behind the Yún shield. The ports of Yún shield is connected to RFID Reader, RED circuits, and switch circuits according to the following table:

Table 2 Yún shield Pins Layout

Yún shield	MFRC522 Reader
9	RST
10	SDA
11	MOSI
12	MISO
13	SDK
Yún shield	LED Circuits
4	Green
6	Yellow
7	Red
Yún shield	Switch Circuit
2	Input

As shown in the Table 2 above, pins 9 to 13 are connected to the MFRC-522 RFID reader at corresponding ports to scan RFID cards and send UIDs to the microcontroller. Pins 4, 6, and 7 are connected to each of three LED circuits (LED in series with a pull-down resistor) with different colors to to represent different situations. Pin 2 is connected to the NPN Darlington switch circuit to “open the door” when a right RFID card is present.

We purchased cheap Arduino Uno R3 board in order to reduce cost, yet it wasn’t very reliable and may break easily, which once took us a while to find it out.

b. Circuits Sections

i. LED Circuits

The three LED circuits each consist of a light emitting diode (LED) and a 300Ω resistor. The LEDs are red, yellow and green in the three circuits and are connected to Pin 7, 6 and 4 of Yún shield respectively.

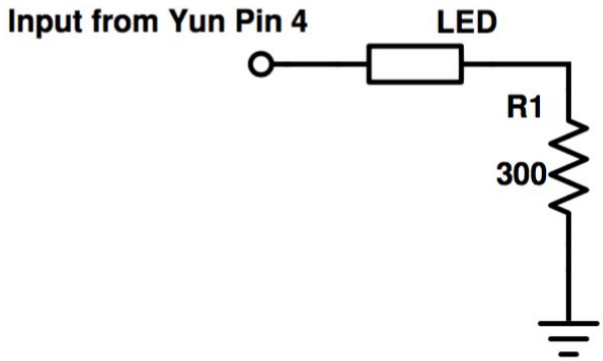

Figure 4 LED Circuit Diagram

ii. Switch Circuit

Since the Arduino Yún shield is unable to provide sufficient voltage and current (only 5 V and 40 mA) to drive the electric strike plate, the NPN Darlington switch circuit is used to qualify the voltage and current requirement of 12 V and 330 mA of the electric strike plate. The diagram of the switch circuit is shown below in Figure 5 The Darlington pair of NPN transistors are used to amplify the input current.

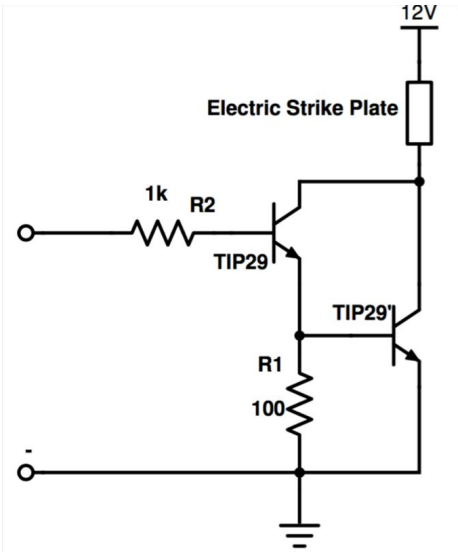

Figure 5 Switch Circuit Diagram

c. System Design - Flowcharts & Algorithms

Once the main program is running, it scans for the presence of RFID tags indefinitely. This is an issue that needs to be addressed such that the program can enter into standby mode if nobody is trying to gain access after a certain period of time. Otherwise, it will continue scanning which consumes a lot of unnecessary power. Not only is this a waste of electricity, but the wasted power, also, exposes the entire RFID module - RFID reader, Arduino Uno, Iduino Yún shield, and the USB drive used to expand the disk space of the Linux file system - to a noticeable amount of heat which could be damaging after extended periods of time. This issue can be addressed by setting an interrupt on the Uno.

After an RFID tag has been detected, the “send_request()” function will be executed which concatenates the unique identifier (UID) of the RFID tag and the office number or “doorid” to the command that will be run in Linux. An example of the command is “node /root/rfid/encryption/encrypt.js bc5fef75 0205.” The command “node” executes the Node.js program which needs to be installed prior to using that command. The next string is the file directory where the encrypt.js file is located. This is the Node.js file in which the following two arguments - the scanned RFID tag and the office number - will be encrypted and sent to the server whose internet protocol (IP) address and port number are contained in the encrypt.js file.

After the request has been made, the Uno will, then, enter the “wait_response()” function for as long as the initialized Linux process that was used to make the request is available. The running process in Linux is similar to any other process or event running in an operating system such as those viewed from the Task Manager in Windows. Therefore, if the process is unavailable, the Uno will not be able to hear back from the server. If the process is available, the time spent in the “wait_response()” function will be brief beyond being noticed, and the “read_response()” function will be entered immediately.

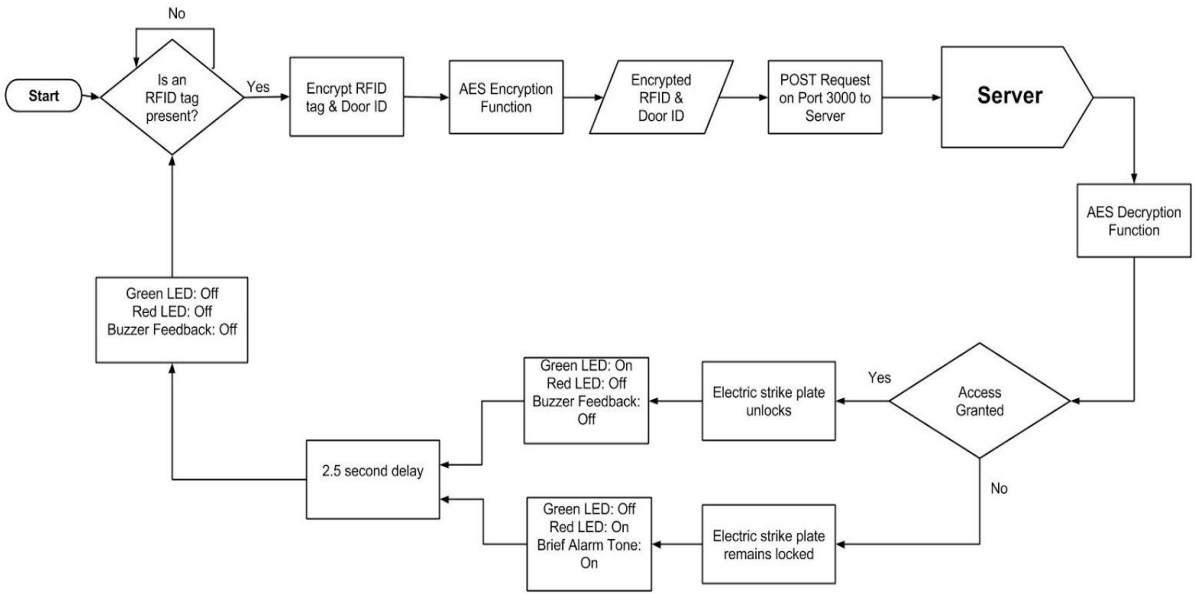

Figure 6 RFID Module Flowchart

In the “read_response()” function, the the process called “Request” will be looking for particular predefined numbers by using the parseInt() function. The number that is found is the server’s response to the request that the Uno had sent to the server earlier. These are arbitrary numbers that represent different situations - the user should be granted access, the user should not be granted access, and error. It is necessary to encrypt this response from the server, because if an unauthorized person with some hacking skills knows what numbers will unlock the electric strike plate, that person can potentially tap into that same running process and send the Uno the proper code in order to gain access. One of the preventative measures for this is that the Request process should not always be open. After the process is used for what it is meant to do, it should be immediately closed. This, then, implies that the Request process should be opened when it is needed.

If the server tells the Uno that a person presented an authorized RFID tag, the Uno will send a logic 1 to the input of the NPN Darlington switch circuit to which the electric strike plate is connected. The solenoid in the electric strike plate will, then, relieve pressure from the latch, and the door can be opened.

d. Bill of Materials

Table 3 Bill of Materials for Entire Project

Components	**Quantity**	**Price Per Unit**
Yún Shield	3	$30
Electric Strike Plate	2	$24
Wall Adapter Power Supply	3	$11
Arduino Uno R3	3	$9
RFID Reader	3	$4
Enclosures	4	$5
Grand Total	18	$230

e. Product Cost Estimate (per door)

Table 4 Cost Per Door

Components	**Cost**
Yún Shield	$30
Electric Strike Plate	$24
Wall Adapter Power Supply	$11
Arduino Uno R3	$9
RFID Reader	$4
Total	$78

V. Software Design

a. Software Requirements

This system server utilizes the following software technologies, which must be installed on any freshly-launched virtual machine:

- Apache - HTTP server
- MySQL & Database - needed to store and query data in database
    - (optional) Phpmyadmin - allows for viewing and editing of the system database in Internet browser
- PHP - used for server-side scripting by browser GUI
- Javascript - used by server-side scripting
- Node.js - server-side Javascript environment used for encryption and communication with Yun.

Essentially, the server utilizes the “LAMP” software bundle with node.js used for its encryption (crypto) and POST request modules. The relevant server files are included separately with the submission of this report.

Additionally, as an optional layer of protection against denial-of-service attacks (which this project faced during development), free CloudFlare services can be employed.

The Arduino Yun’s embedded Linux server must also include node.js, as it runs a script that is complementary to the main web server’s server.js script. This will likely require expansion of the Yun’s disk space with a flash drive/SD card, as node’s installation size exceeds the 7 MB storage space provided by Yun out of the box.

b. System Design - Figures and Flowchart

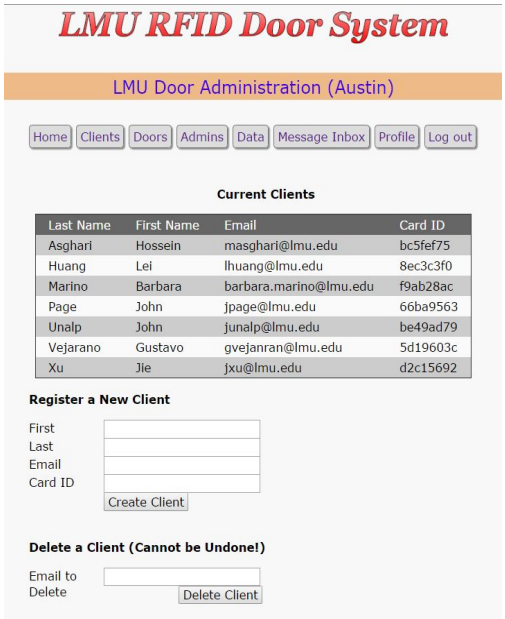

Figure 7 Client web page as viewed by administrator

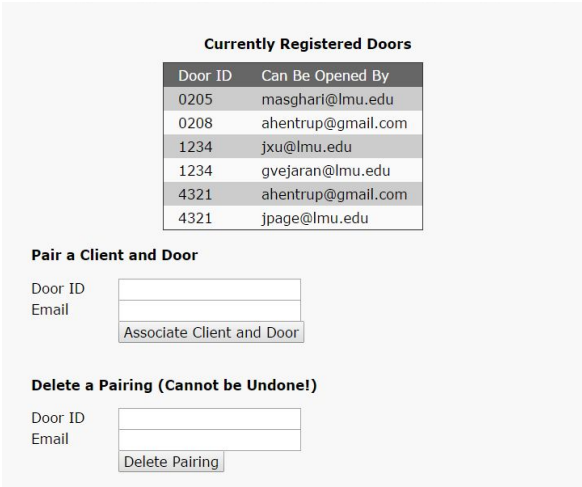

Figure 8 Management of user access rights

Access Control Event Log

EventID	Date and Time (GMT)	Door Serial	Scanned Card ID
1	2016-05-03 11:06:16	0205	bc5fef75
2	2016-05-03 11:39:29	0201	bc5fef75
9	2016-05-03 19:42:50	test	bc5fef75
10	2016-05-04 02:47:27	0201	b3c9a023

Figure 9 Administrator Access Control Scan Log on Web GUI

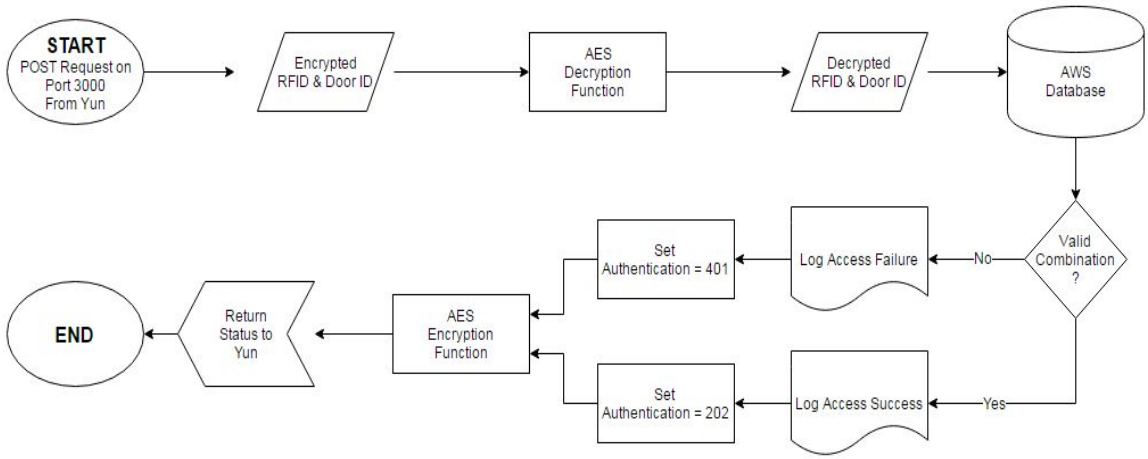

Figure 10 Server Flowchart

c. Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) was used to encrypt the data transmission between server and Arduino Yun shield. It is a symmetric block cipher used by the U.S. government to protect classified information and is implemented in software and hardware throughout the world to encrypt sensitive data.The AES that's implemented in this system is Cipher Block Chaining(CBC) AES. For every encryption process, there are three input; a key and an initialization vector will be used to encrypt the plaintext to generate the encrypted data. The key is prestored in Arduino Yun and online server. Every time a message is about to be sent to server, a new initialization vector is randomly generated sent with the message, so that the server will be decrypt the message with prestored key, received initialization vector and the encrypted data.

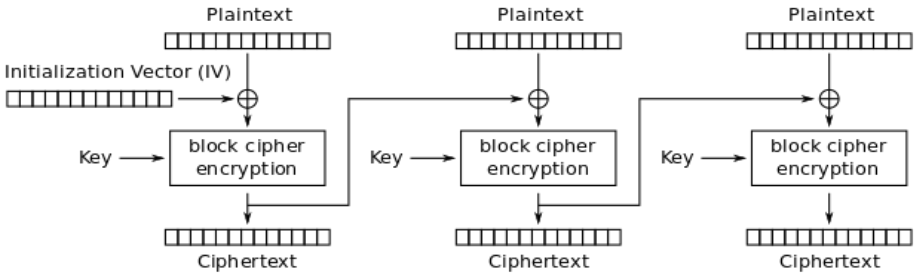

Figure 11 Encryption

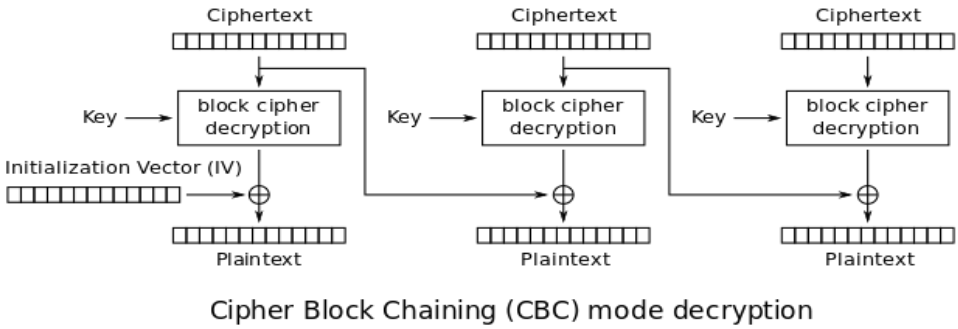

Figure 12 Decryption

VI. Experimental Test and Demonstration

a. Working Prototype

The working prototype of this project includes a small door model with a electric strike plate. Two enclosures are installed on both sides of the “wall” next to the metal key lock, the small one in the front contains a RFID reader and LED circuits and the big one in the back includes a Arduino Yún Shield, Arduino Uno R3 and a switch circuit. Working prototype’s picture is shown in Figure 13 below.

Figure 13 RFID module end product installed on door model

b. Test used to benchmark the system performance

In order to measure the power consumption of this system, a voltmeter was connected in parallel with the power adapter to measure the voltage, and an ammeter was connected in series with the power adapter; instrument readings were taken for 20 times under different circumstance: while the system was idle, and while the system was reading cards and processing. The power consumption of the whole system can be calculated by multiplying average voltage and current values.

A timer was used to roughly measure the time period this system need to finish a one-time scanning-authentication process. It starts when a RFID card is present at the reader, and stops when the Yún Shield receives match information from server and turn LED on.

c. Test Results

The test results are shown in the table below. The voltages of the system while idle and processing are both 12 V. The currents of the system while idle and processing are both 220 mA, yet it is constantly at 220 mA while idle, and fluctuating while processing. Therefore, the power consumption is averages to about 2.64 W.

The response time of this system is about 12 s. The parameters that were chosen for the specific AES algorithm mode that was implemented in the project are what contribute to the long delay. During testing, it was observed that the majority of the delay was due to the

Yún Shield trying to send the encrypted UID of the RFID tag. There does not appear to be any delay when the Yún Shield receives and decrypts the authorization message from the server. It is suspected that the parameters for the encryption algorithm are not the most compatible with the architecture (400 Mhz, 24 MIPS) AR9331 microprocessor in terms of speed.

Table 5 Testing Results

RFID Module Performance	
Power dissipation while idle	
Avg. Current	0.22A
Voltage	12V
Power	2.64W
Power dissipation while reading card	
Avg. Current	0.22A
Voltage	12V
Power	2.64W
Response Time	
Avg.Time	11.98s

VII. Ethical Considerations

One aspect of the project that motivates the team is the investigation of RFID technology for the purposes of determining a reasonable price point for the finished product. Early in the conception of the project, the team discovered that professional door authentication systems for schools and businesses could easily cost up to $3000 per door. This number seems unreasonably high given the price of the electronic components themselves. While installation costs and “proprietary” costs could account for a portion of the inflated price, there are few other reasons aside from profit gouging that the team could come up with to explain the discrepancy. As such, this project is undertaken in the spirit of making more information available to potential customers, so that they can make better informed consumer decisions.

Another dimension that motivates the design choices made in this project is customer safety. The team pledges that every aspect of the system design is created to fail safely and gracefully. This is to ensure that system failures will not cause a person to become trapped

inside/outside of a room, which could prove catastrophic in the event of an emergency. Additionally, the system should not fail in such a way that it neglects to properly secure a person's belongings. Upon failure, the planned system will remain locked, however it will allow for the door to open from the inside by turning the handle. From the outside, the door can be opened using the original metal key. Under normal circumstances, the system will provide an added layer of convenience to users. Under failure conditions, the door will operate as it normally would if the system were not there at all.

Finally, the team endeavors to protect the privacy of all of the system's users. Care was taken to prevent any personally identifying information from being shared with anyone other than the door administrator. This was achieved by protecting the sensitive data with AES encryption.

VIII. Contribution to ABET program, LMU Values, Diversity, Social Communities, Multidisciplinary, IEEE Values

Although the proposed RFID door authentication project was undertaken as a senior design project for the Electrical Engineering department, unique aspects of the project require considerations that go beyond the purely technical. One of the most important skills that this project emphasized was communication within a group setting. While technical knowledge forms the bedrock of any competent engineer, the ability to work well with others is fundamental for success in a professional or research environment. The emphasis placed on both technical and communication skills was consistent with LMU's mission to foster the "Education of the Whole Person" [reference]. The ability to communicate is also an explicit student outcome of any ABET-accredited engineering program.

A critical goal of this project was the investigation of a trending technology and its associated benefits, security vulnerabilities, and financial costs. This was to determine the feasibility of the technology coming to define the modern standard for access control, and make the market cost (independent of any profit-seeking motive) of the technology more transparent to the general public. Through our consideration of the technology and its potential impact on society, this project was consistent with one of the core values of IEEE, which is to be "a trusted and unbiased source of technical information...for technical dialog and collaboration." Additionally, these goals fulfill several ABET Student outcomes, including:

(a) an ability to apply knowledge of mathematics, science, and engineering

(c) an ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability

(e) an ability to identify, formulate, and solve engineering problems

(f) an understanding of professional and ethical responsibility

(g) an ability to communicate effectively

(h) the broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context

(j) a knowledge of contemporary issues

(k) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.

IX. Conclusion

Given the request of the electrical engineering faculty for an RFID system for their offices, there was much motivation and enthusiasm for creating a simple, inexpensive, and reliable access control system that used RFID technology. The baseline cost of $3,000 per online door for LMU's current OneCard system necessitated this senior project. In addition, today's RFID technology offers improved methods of access control, especially when compared to magnetic stripe technology which is starting to become antiquated just as much as metal keys are. All the more surprising, metal keys are still being used in the electrical engineering department. One key per lock is unnecessary, inefficient, and unsafe considering that thousands of dollars accrue in both metal key and mag-stripe systems due to proprietary maintenance and upgrades which could lead to a serious flaw in the system if avoided. The risk of damage or theft of valuable assets is highly disproportionate to a single key that is worth cents. This team's design addressed these issues by designing the system such that a single, lightweight RFID card can hold multiple access rights, maintenance of the system is extremely low and can be easily handled by a new administrator, and the quick termination of lost or stolen RFID cards by the system administrator ensures the safety of people and their assets. As engineers are known to produce efficient, cost-effective, and practical solutions, this senior project progressed such skills with the added benefit of the opportunity to help and give back to LMU.

- An Internet-enabled door system with RFID can be implemented for a fraction of the cost of current commercially-available solutions
- A custom online administration system has been created to easily manage the system
- This cost-effective system is easy to maintain, secure, and convenient

X. Suggestions

1. Optimize Arduino code for power efficiency

    a. The C code that is running on the Uno is set such that it scans for RFID tags indefinitely. There was a strong desire to implement interrupts in order to address this issue so as to save power, but time restraints prevented that.

Otherwise, the intention was to set an interrupt to put the Uno in standby mode after a certain amount of time has passed without a presented RFID tag. There are certain pins that need to remain active, however, in order to listen for a pin change. The pin to which the MFRC-522 connects is the pin that would change. First, the pin that changes when an RFID tag is presented to the MFRC-522 needs to be determined. Then, the interrupt can be configured such that the Uno will wake up or leave standby mode when that particular pin changes states.

   b. Since the ethernet port is being used for internet communication, there is no need for Wi-Fi interface. Therefore, the Wi-Fi interface should be powered off which can be done through the web GUI that comes with the Yún shield.

2. Enhance web GUI
3. Increase speed of Arduino Yún's Internet query
   a. It was only until the AES encryption was implemented that there was an unreasonable delay of 12 seconds. The solution is to simply implement another AES algorithm mode such that this response time is less than a second.

XI. References

[1] Ford, Ralph M, and Chris S Coulston. *Design For Electrical And Computer Engineers*. Boston: McGraw-Hill, 2008. Print.

[2] Symbol Technologies, "RFID – A Revolution in Asset Management". Available at <http://www.symbol.com/products/rfid/rfid.html>.

[3] C. Wassel, and A. Onda "RFID Technology Briefing," Electronic Presentation, Center for eBusiness and Advanced IT, copyright 2006.

[4] Siar, Omer. "Arduino RC522 RFID Door Unlock". *Instructables.com*. N.p., 2012. Web. 14 Dec. 2015.

[5] Ieee.org,. "IEEE IEEE Strategic Plan 2015-2020". N.p., Web. 21 June 2015. Available at <http://www.ieee.org/about/ieee_strategic_plan.html>

[6] *CRITERIA FOR ACCREDITING ENGINEERING PROGRAMS Effective For Reviews During The 2015-2016 Accreditation Cycle*. 1st ed. Baltimore: ABET, 2014. Print.

[7] Mission.lmu.edu,. "Mission". N.p., 2015. Web. Available at <http://mission.lmu.edu/missionstatement/>

XII. Appendices

Teammate Roles & Responsibilities

A. Austin Hentrup

Austin, our software and hardware engineer, was responsible for designing and implementing the functionality of the online server. He set up the AWS server, built the graphical user interface of the website, and implemented server-side encryption. He was also responsible for implementing protection against denial of service (DoS) attacks.

B. Deyi Lu

Deyi, our hardware engineer, was responsible for identifying and purchasing components for the project. He was also responsible for designing the RFID module for the project, investigating functions of encryption, and building the prototype circuits.

C. Peter Roldan

Peter, our software and hardware engineer, was responsible for writing the source code for the hardware and establishing communication from the RFID module to the server. He, also, configured, integrated, and maintained the Uno and the Linux operating system on the Yún shield to handle the files and data that were necessary for hardware functionality.

D. Team

We designed the overall circuits, built a functioning prototype for the proposed access control system that uses RFID technology, and implemented functions of multiple user access and encryption. We collaborated to identify system problems and discussed them during the project in order to integrate the software and hardware between the source code on the Arduino Uno with the software that was running on the server.

Selected Microcontroller Code

```
//SPI Library
#include <SPI.h>

//RFID Library
#include <MFRC522.h>

//Bridge Library
#include <Bridge.h>
#include <Process.h>
#include <Console.h>
```

Figure 14 Core Libraries to run the C code on Uno

```
//Global Variables
#define RST_PIN 9
#define SS_PIN 10
MFRC522 mfrc522(SS_PIN, RST_PIN);
#define powerPin 6                //Yellow LED
#define failPin 7                 //Red LED
#define passPin 4                 //Green LED
#define doorPin 2                 //Electric Strike Plate
#define alarmPin 5                //Alarm
```

Figure 15 Global Variables

```
//Dr. Asghari's Office in Doolan
//int doorid[4];
String ScannedCard = "";
String doorid = "0205";
```

Figure 16 Variables for the data to be transmitted

```
//Create a process.
Process Request;
```

Figure 17 Initialize a process in Linux to handle internet communication

```
void setup()
{
  Bridge.begin();                   //Initialize communication between Uno and Yun Shield
  SPI.begin();                      //Initialize SPI bus
  Console.begin();                  //Initialize Console monitor (equivalent of Serial monitor)
  mfrc522.PCD_Init();               //Initialize MFRC522

  //Set outputs
  pinMode(powerPin, OUTPUT);
  pinMode(failPin, OUTPUT);
  pinMode(passPin, OUTPUT);
  pinMode(doorPin, OUTPUT);
  pinMode(alarmPin, OUTPUT);
  digitalWrite(doorPin, LOW);
  digitalWrite(alarmPin, LOW);

  //TESTING ONLY: The program waits here until the Console monitor is opened.
  //Note: The Console monitor is the equivalent of the Serial monitor.
  while(!Console);

  //Flash LEDs to indicate that a new program has been uploaded.
  for (int i=0;i<3;i++)
  {
    digitalWrite(powerPin, HIGH);
    digitalWrite(failPin, HIGH);
    digitalWrite(passPin, HIGH);
    delay(250);
    digitalWrite(powerPin, LOW);
    digitalWrite(failPin, LOW);
    digitalWrite(passPin, LOW);
    delay(250);
  }

/*Final Implementation: Each RFID module would have its corresponding DoorID
 * stored in the EEPROM. This allows the associated office number to be used
 * by the corresponding RFID module even if the Uno were reprogrammed remotely.
 * This requires writing DoorID to EEPROM beforehand.
  for(int i=0;i<4;i++)
  {
    doorid[i]=EEPROM.read(i);
  }

  digitalWrite (doorPin, HIGH);
*/

}
```

Figure 18 Initializations before main program is run

```
//If access is granted, open door and illuminate green LED.
void openDoor()
{
    digitalWrite(doorPin,HIGH);
    digitalWrite(passPin,HIGH);
    digitalWrite(powerPin,LOW);
    digitalWrite(failPin,LOW);
    delay(2500);
    digitalWrite(doorPin,LOW);
    digitalWrite(passPin,LOW);
}                                                    \
```

Figure 19 Function to Unlock Electric Strike Plate

```
//If access is denied, sound alarm and illuminate red LED.
void failed()
{
    //Security redundancy: deactivate electric stike plate.
    digitalWrite(doorPin, LOW);
    digitalWrite(passPin,LOW);
    digitalWrite(powerPin,LOW);
    digitalWrite(failPin,HIGH);
    digitalWrite(alarmPin,HIGH);
    delay(500);
    digitalWrite(alarmPin,LOW);
    delay(2000);
    digitalWrite(failPin,LOW);
    noTone(5);
}
```

Figure 20 Function to keep Electric Strike Plate Locked

```
void send_request()
{
  //Online Server: http://uesc.space:3000/rfidrequest
  //Command Line Interface (CLI) Instruction: "node <location of encryption file> <rfid> <doorid>";
  //Example: node /root/rfid/encryption/encrypt.js bc5fef75 0205
  String Testcmd = "node /root/rfid/encryption/encrypt.js";
  Testcmd += " ";
  Testcmd += ScannedCard;
  Testcmd += " ";
  Testcmd += doorid;

  Request.runShellCommand(Testcmd);

  //TESTING ONLY
  Console.println("");
  Console.print("Sent Request: ");
  Console.println(Testcmd);

  //Data should not be saved locally.
  ScannedCard = "";
  return;
}
```

Figure 21 Function to send UID of presented card and office number to server

```
void wait_response()
{
  if(!Request.available())
  {
    //Implement Timer Interrupt (non-blocking) to avoid
    //hanging if no response is received from the server.

    return;
  }
}
```

Figure 22 Function to handle no response from server

```
void read_response()
{
  Console.println("Reading response from AWS...");
    while(Request.available())
    {
      //Look for integer
      int Response = Request.parseInt();
      if(Response == 202)
      {
        openDoor();
        return;
      }
      else if(Response == 401)
      {
        failed();
        return;
      }
      //Implement Timer interrupt here to avoid hanging
      else
      {
        //Error: return to main program.
        digitalWrite(failPin, HIGH);
        digitalWrite(powerPin, HIGH);
        delay(1000);
        digitalWrite(failPin, LOW);
        digitalWrite(powerPin, LOW);
        return;
      }
    }
}
```

Figure 23 Function to search for decrypted authorization message

```
void loop()
{
  //Look for new cards
  if (!mfrc522.PICC_IsNewCardPresent())
  {
    //Hangs here until a card is presented
    //To save power, incorporate interrupt.
    return;
  }

  //Select one of the cards
  if (!mfrc522.PICC_ReadCardSerial())
  {
    //Hangs here until a card is presented
    //To save power, incorporate interrupt.
    return;
  }

  //Capture RFID tag's UID as a string
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
    //Eventually, the UID bytes should be encrypted as they are
    //read instead of saving them to a variable.
    ScannedCard += String(mfrc522.uid.uidByte[i] < 0x10 ? "0" : "");
    ScannedCard += String(mfrc522.uid.uidByte[i], HEX);
  }

  send_request();
  wait_response();
  read_response();
}
```

Figure 24 Main program which scans indefinitely for RFID tags