Introduction

Joshua N. Satterfield

Advisor: Dr. Yu-Ju Lin

Charleston Southern University

Table of Contents

[Abstract 3](#_Toc442130943)

[Introduction 4](#_Toc442130944)

[Background 5](#_Toc442130945)

[Goals 7](#_Toc442130946)

[Design and Methodology 7](#_Toc442130947)

[Wi-Fi 8](#_Toc442130948)

[Protocol 9](#_Toc442130949)

[Encryption 9](#_Toc442130950)

[Challenges 9](#_Toc442130951)

[Implementation 10](#_Toc442130952)

[Initialization Vector 11](#_Toc442130953)

[Implementation 11](#_Toc442130954)

[Conclusion 12](#_Toc442130955)

[Future work 12](#_Toc442130956)

Abstract

In this project we investigate how to wirelessly lock and unlock a door with off the shelf parts. The goal of this project is to create a system that is both affordable and secure for future homes. We used an Arduino, Arduino Wi-Fi Shield with Android as our platform. We designed a protocol that utilizes the transport layer protocol UDP for communication between an Android smartphone and the platform. We are able to successfully lock and unlock a door wirelessly. There are issues remaining to be resolved. The form factor needs to be reduced for better integration with the user’s door. Integrating with the Touch-ID API for added security in case of a lost device must be accomplished as well. This will be a step to avoiding storing the key on the mobile device, increasing safety.

Introduction

Securing homes is an important issue. Unsecured access is a vulnerability for your family. The traditional physical lock system uses a literal lock and key. The lock has pins inside it. The indentation on the key must be the exact counterpart to the length of the pins inside the lock in order for the lock to be turned. When the pins are pushed away the right distance they no longer block the inner mechanism, the tumbler, from turning. This system works, but there are several drawbacks. First, while there are theoretically many combinations of pin lengths, the exact procedure can be circumvented. By hitting the pins with a small tool you can force them all away from the tumbler. Then if you twist it at the correct time you can keep them from falling back in to their protruding states. This can be done in a few seconds by a professional. Second, the number of possible combinations of the pins is limited. The number can vary depending on implementation. A more secure mechanism can be purchased. The high price, however, keeps most home owners away. A more secure modern day solution is required to remedy this situation.

The Internet of Things is a system of devices, primarily embedded, which communicate together for things like information gathering and automation. Kevin Ashton, who coined the term describes a system of devices which operates without human intervention. In his usage the term was originally intended to describe a kind of system, his implementation of which used RFID. The term has now been greatly expanded to refer to other implementations, including the Wi-Fi domain, which allows us to cover a wider range. We will use the same concept to solve the previously mentioned locking systems.

The goal of the project is to research how to implement a modular, customizable system that will be able to secure a future home. We choose a two-part system, integrating with the user’s phone as the key and an Arduino as the lock mechanism. A method of symmetric encryption was chosen for securing communication between the smartphone and the receiver.

In this paper we investigate how to wirelessly lock and unlock a door with off the shelf parts. We first survey the existing systems on the market. We will then discuss the difficulties we encountered in the research and methodology, and how to overcome each obstacle. We then present our experiments and findings in the Implementation section. Finally, we present our conclusion.

# Background

In this section we survey the existing related technology. We first investigate US patent 5,942,985. It is a solution which uses a stored key to insure authorization before opening the door. Then we discuss present commercial solutions and their disadvantages. We then present our solution which is able to overcome the shortcomings.

The lock in US patent 5,942,985, from 1999 describes a method of unlocking automatically when the key is near. The lock controller sends a pilot signal, and when the key device detects a valid signal, it sends its key. If the key stored on the lock controller matches the one it receives, it unlocks and notifies the key device. It describes various wireless signals that can be used including radio and microwaves. This system does not allow the door to be controlled without being present. It can also be vulnerable to replay attacks, if it accepts the same unaltered key each time. If so, once someone captures the key, they can open the door with it at any time.

The August Smart Lock replaces the inward facing part of a deadbolt and can be turned manually on the inside. It integrates with the user’s smartphone and then uses a mobile application to authenticate the users. It unlocks the door when the phone is near using proximity detection and can lock the door automatically. It allows you to issue keys to people in your phones contacts. Since these virtual keys are distinct it is more secure. It also allows you to give people access at certain times. It uses Bluetooth and it logs accesses. It has four AA batteries for backup power. However, it has a major flaw.

The password reset system sends your phone a number from one to a million, which is a large problem space for a human but to a computer it can brute force it in a short amount of time. Often websites limit the frequency of attempts but the August software does not contain such precautions [Jmaxxz pt. 2]. As a result, someone can create a program which try all the combinations until one is successful, resetting your password and unlocking your door without really being authorized.

The Kevo lock’s uses a Smartphone app for authorization and is unlocked by touching it. It uses Bluetooth but not its security features. Its authentication uses public key encryption and thus is secure [Peter Ha]. It also includes a key fob that is recognized by the lock when the lock is touched. Use uses four AA batteries and does not use home electricity. However, Bluetooth has a limited range, and does not allow it to be controlled remotely.

These systems all have their drawbacks. To overcome these issues, we propose to build a system that is both affordable and secure for future homes. We do not want insecure devices as the gatekeepers of our homes lending themselves to be attack vectors. Instead, a strong 128-bit asymmetric encryption along with proper protocol to further enhance the security. By using a mechanism that unlock your door from the inside only when authorized. We want the “Internet of Things” to integrate into or life without sacrificing security. We also want the convenience of locking and unlocking our doors while we are away without compromising safety.

# Goals

We need the communication to be secured for the safety of your own home and family so we will encrypt the communication between devices. We designed a simple protocol for handling the information exchange. We created it without any unnecessary complexity, focusing on the most practical use cases. We need a platform that is small, power efficient and inexpensive. It needs to be small so it can be installed easily and integrate with your homes design.

We have built a compact system which attaches on the face of the deadbolt which allows for installation without removing your door knob. The platform needs to be easy to customize. We have developed two well commented open-source programs which can be installed by simply using the correct USB cable. No extra flashing tools are needed to customize the program and its functionality.

# Design and Methodology

We choose the UDP protocol for its simplicity. It is faster and less complex than TCP. Since in the case of dropped data the commands can quickly be sent again by the user we concluded that it is an appropriate tradeoff for speed and simplicity.

To communicate we send short packets with our commands over Wi-Fi. The Arduino then responds with the appropriate action, whether it is to turn the lock to a certain position or respond to authentication attempts.

In order to communicate over Wi-Fi, we first tried using a small wireless component called the XBee ZigBee. It, however, was not designed to communicate using Wi-Fi. Instead it uses its own protocol optimized for mesh networks, with several devices communicating to each other. Therefore, it cannot communicate with a standard smartphone using the Wi-Fi protocol. Its protocol operates differently than Wi-Fi. Although it is wireless, it also runs own a different incompatible frequency. Likewise, it cannot communicate with the user’s home network, or their PC. This prevents the lock from being controlled over the internet without additional hardware.

Thus we attempted utilizing the XBee Wi-Fi module which does use the appropriate frequency and therefore should be able to communicate with the user’s home network and smartphone. However, there was no official library for use with the Arduino. We attempted to take advantage of a third party library but we could not initialize the device using that system.

Then we started using the Arduino Wi-Fi shield. The shield comes with a useful network library. Detailed documentation for it is available as well. Using the library documentation, we expanded a simple communication example until it fulfilled our purposes. We added logic after the networking code to react in different ways when certain inputs were received.

In this study, we investigate a scenario of a home environment which needs protection from unwanted entrance. We used Arduino, its Wi-Fi shield, and an Android smartphone to accomplish this research.

## Wi-Fi

By using Wi-Fi, we can communicate with the user’s cell phone easily. Wi-Fi is more widely available for home networks than Bluetooth. Although phones are typically equipped with Bluetooth, it could not be controlled over the Internet, unlike the Wi-Fi solution. Cellular modules have the advantage that the do not need to be set up with the user’s home network information but they have the extra cost of a cellular bill and cannot integrate with their home network directly. So by using W-Fi we achieve lower cost than cellular/GSM and more interoperability than Bluetooth and ZigBee, leave it open to the possibility of control from the user’s home network. This allows a system to be constructed that sends commands to the user’s house from the internet using the existing receiver on the Arduino. In contrast to Bluetooth or GSM where an additional receiver would be needed to access Wi-Fi. The disadvantage is that the Arduino needs authorization to use the home network although it can be set up to scan SSIDs. The advantage is that you can communicate as desired from other systems on your network directly or externally using port forwarding.

## Protocol

The protocol uses ASCII encoding for its commands, which are a single character. In version one of the protocol the Arduino would enact commands received immediately.

|  |  |  |
| --- | --- | --- |
| l | u | a |
| Lock | Unlock | Authorize |

In version one of the protocol the Arduino would enact commands received immediately. In version two first the phone must send the authorization command, then wait for a response. The Arduino responds with the AES initialization vector. The phone then encrypts its command using the IV and sends the packet to the Arduino.

## Encryption

### Challenges

For encryption I began researching the DES protocol. It is now vulnerable to feasible attacks and 3DES is questionable. Despite iterating over the algorithm more times it can still be vulnerable to attacks. It can be slow for software implementations since it was designed for hardware [Pornin]. We then investigated AES – The Advanced Encryption Standard. It has acceptable security for the foreseeable future. An attack has been discovered which allows decryption with less than brute force work but it still has high computational complexity [Bogdanov]. This means it will still take a long time to find the key. I was able to find an Arduino that implemented the basic features of AES and CBC mode. We discuss the functionality of AES and CBC below.

### Implementation

We implement a secure method of communication using AES. AES uses an algorithm to encrypt our data. In order to decrypt it you must provide the same key as the one provided to encrypt it, or else the resulting output will be garbage. The key length is typically 128-bits, which is secure enough. This is because because we will not be alive when the computer finishes cracking it by brute force (c.f. Arora). The encrypted packet will naturally be the length of the AES key, which is 128 bits. In the case of a message being longer than a block (the key length) an algorithm must be used to encrypt each block without becoming insecure. We choose the CBC block chaining algorithm because it uses a random IV to prevent replay attacks and it is more secure than EBC. EBC can leave noticeable patterns in the data outputted. The initialization vector is distinct from the key; it is not secret. Thus the IV can be sent over the wire.

The key is implemented as a byte array. When using hex literals to represent bytes compactly it looks like

byte[] key = { 0xde, 0xed, 0xbe 0xef, … };

In Java bytes are signed. That means you can store positive or negative numbers, unlike unsigned numbers. Since Java excepts unsigned bytes it will not store high numbers in bytes for fear that the upper part of the number will cause them to be interpreted as negative numbers. This we resolved by casting each number to the int type. This still fits in a byte, it would only need more space to keep such a number from being interpreted as negative. This is because while unsigned bytes take value from 0 to 225, signed bytes take values from -128 to 127. The leftmost number indicates negation in signed numbers, and takes up on bit. We have enough bits, they just won’t be interpreted positive if printed. Here is the way to cast to integers:

byte[] key = { (int) 0xde, (int) 0xed, (int) 0xbe (int) 0xef, … };

### Initialization Vector

We use an initialization vector to increase security. As the initial state for the AES algorithm the IV effects the end result end a manner different than the key which is used. If you encrypt the same message with the same key twice you will get the same result. Thus if someone records you sending an encrypted command signifying “unlock” they could replay this command at any time to manipulate your device. This is a vulnerability that can exist in encrypted systems. To prevent this a random IV is generated each time the module sends an authorization command.

The 16 byte IV comes from the Arduino to the Smartphone and is dealt with using the javax.crypto.Cipher library. The known key (which may be stored on the phone) is combined with the random IV using the AES algorithm and the message is sent back. The next authorization command from a phone will send back a different IV, so the same message will no longer make sense to the Arduino when decrypted using an old IV. This is because it is using an old IV.

# Implementation

The implementation uses Android API 23 with java.net.DatagramSocket for the UDP socket on the smartphone. We used an asynchronous function for networking code to keep the UI from blocking. if we just called a long running function the graphics would freeze until the function was running. Since networking can take time this is strongly recommended. We also used the Arduino Wi-Fi library to receive information from the phone. When it receiving a command if it regards the lock it will turn the servo. The servo is wired to this Arduino this way:

|  |  |  |
| --- | --- | --- |
| Red | Black | White |
| 5V | GND | Pin 9 |

The lock command turns it to 100 degrees and the unlock command turn it to zero degrees. Choosing 100 degrees instead of 180 keeps to motor from trying to go to far and stuttering.

Using AES CBC with a random IV we prevent the replay attack exploit vector. Using the Arduino microcontroller prototyping platform and Android Studio we built a system which can run on inexpensive and easily obtainable hardware.

One use case our platform does not currently handle is automatically open when you approach the door. Safeties would have to be put in place you solely want to look outside the door. This could be controlled from the mobile app. Handling the approach could be possible by sending out a pilot signal informing the phone that there is a lock nearby, then the phone can reply with a command. The phone would need a program running in the background to react to this situation.

# Conclusion

This has been an insightful project. We have learned a lot about practical application of computing concepts in networking, automation and security. We found that using Wi-Fi communication provides an adequate layer for our networking to be built upon and that the speed of transmission was swift. By using the Arduino modular prototyping platform for the receiver we were able to create a quick prototype for that functionality and interface it with the Android application. The platforms we used enabled quick prototyping and the addition of strong encryption for the benefit of the consumer.

# Future work

To decrease the size of the device we will investigate the possibility of designing a single chip with both a microcontroller and Wi-Fi. The Arduino uses an AVR microcontroller which can be obtained at a less expensive price as a standalone part, especially in bulk. We can then create an inexpensive design and pass on the benefits to the consumer.

References

Apple Support. “Apple Computers and Displays: Powering peripherals through USB” <https://support.apple.com/en-us/HT204377> Retrieved 2015-12-04

Thomas Pornin. Answer to “Why is AES more secure than DES?”. <http://stackoverflow.com/questions/3929325/why-is-aes-more-secure-than-des> Retrieved 2015-12-04.

Morris Dworkin. “Recommendation for Block Cipher Modes of Operation”. NIST. <http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf> Retrieved 2015-12.

Matthew Green. “How (not) to use symmetric encryption”. [http://blog.cryptographyengineering.com/2011/11/how-not-to-use-symmetric-encryption.html Retrieved 2015-12](http://blog.cryptographyengineering.com/2011/11/how-not-to-use-symmetric-encryption.html%20Retrieved%202015-12).

Chin, Seo-Young. Automatic Locking/unlocking Device and Method Using Wireless Communication. Samsung Electronics Co., Ltd.2, assignee. Patent 5,942,985. 24 Aug. 1999. Print.

"The August Smart Lock." August Smart Lock. Web. Retrieved 25 Feb. 2015. http://august.com/

"UniKey The Evolution of the Key Is Here - As Seen on Shark Tank." UniKey. UniKey. Web. Retrieved 25 Feb. 2015. <http://www.unikey.com/>.

Drf1090 regarding "August Smart Lock." Engadget. 10 Dec. 2014. Web. 26 Feb. 2015. <http://www.engadget.com/products/august/smart-lock/reviews/14f6/>.

Jmaxx. “The August Smart Lock’s not so 2-Factor Authentication (Part 1)”. <https://jmaxxz.com/blog/?p=476> retrieved 2015-02-01

Jmaxx. “The August Smart Lock’s not so 2-Factor Authentication (Part 2)”. < <https://jmaxxz.com/blog/?p=498>> retrieved 2015-02-01

Ha, Peter. “Are Smart Locks Secure, or Just Dumb?” Gizmodo. <<http://gizmodo.com/are-smart-locks-secure-or-just-dumb-511093690>> retrieved 2015-02-01

Bogdanov, Andrew et. al. “Biclique Cryptanalysis of the Full AES”. Microsoft Research. 2011. <<http://research.microsoft.com/en-us/projects/cryptanalysis/aesbc.pdf>> retrieved 2015-02-01

Arora, Mohit. “How secure is AES against brute force attacks?” EE Times. <<http://www.eetimes.com/document.asp?doc_id=1279619>> retrieved 2015-02-01