캡스톤 디자인- 박상오 교수님

캡스톤 프로젝트

최종리포트- 슥~

Team name: Team Back

Team Member: 20144453_김용준

20153710_정건모

20150524_천준현

2018년 12월 13일

목차:

1. 개요

현재 국내 IT 기업 및 도어락 업체 등에서 Smart Door Lock 을 도전하고 있다. 이미 시중에는 다양한 제품들이 판매되는데 기존의 도어락을 교체를 해야 하거나 높은 가격으로 인해 접근성이 떨어진다. 이런 시장 상황에 착안하여 기존의 도어락에 추가 부착만으로 동작하는 Smart Door Lock 을 제시한다.

물론 국내가 아닌 해외, 구체적으로 일본에서는 이미 이와 같은 부착형 Smart Door Lock 이 존재한다. 대부분의 제품들은 단순히 암호화된 Bluetooth 신호만을 가지고 문의 개폐 여부를 결정한다. 하지만 이는 사용자가 원치 않는 상황에서 문이 열릴 수 있는 가능성을 가지고 있고 실제 사용 사례에서도 이와 같은 현상이 발생한다는 후기가 존재한다. 좀 더 개선한 제품에서는 GPS 를 사용하지만 실내에서의 GPS 오작동으로 인해 여전히 같은 문제점이 남아있다.

서버를 사용하지 않고, 오직 스마트폰 어플리케이션 내에서 동작하도록 개발할 것이다. GPS 센서 같은 경우는 사용자에 따라 민감하게 받아들일 수 있는 부분이 존재하므로 데이터를 외부로 유출하지 않기 위해 스마트폰과 도어락 간의 네트워크에서만 동작할 예정이다.

또한 기존에 GPS 신호가 급격하게 변하는 것에 대해 대비하기 위해 스마트폰 내장 센서들을 활용한다. GPS 신호의 불안정성을 다른 센서 값들을 이용해 실내 및 실외를 정확하게 구분하기 위해서이다.

도어락 마다 다양한 형태로 이루어져 있지만 개폐 방식에 대해서는 레버를 돌리는 일관된 방식을 사용한다. 이에 착안하여 부착형 도어락을 제작해 범용성을 높이고 기존의 도어락을 유지한다는 면에서 사용자에게 매력적으로 다가설 수 있다.

2. 프로젝트 세부사항

2.1 목표

- 시제품 목업 제작

- Smart Door Lock 임베디드 개발

- 스마트폰 어플리케이션 개발

2.2 역할분담 및 내용

Members	To-Do
김용준	목업 제작 및 스마트폰 어플리케이션 보조
정건모	도어락 임베디드 개발
천준현	스마트폰 어플리케이션 개발
공통	도어락 기술 개발 및 개선

시제품 목업 제작은 학교에서 지원하는 3D 프린터를 이용해서 진행되었다. 도어락과 통신은 블루투스를 이용해서 진행되고, 모터 동작 및 통신 프로그래밍은 아두이노를 기반으로 진행되었다. 스마트폰 어플리케이션의 알고리즘은 GPS 로깅을 활용해서 보안 로직이 들어가 있으며 기타 센서들을 복합적으로 사용하여 보안성이 강화되었다.

2.3 개발 환경

- Android Studio 3.14

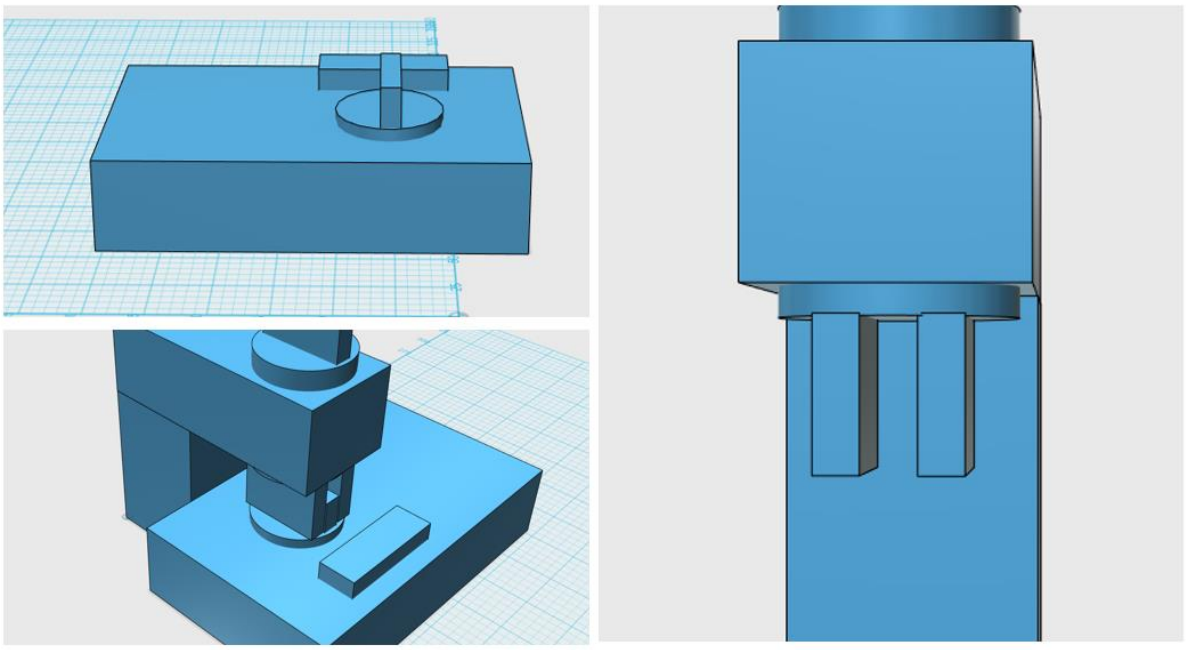
- JAVA SDK 1.8

- Android API 28

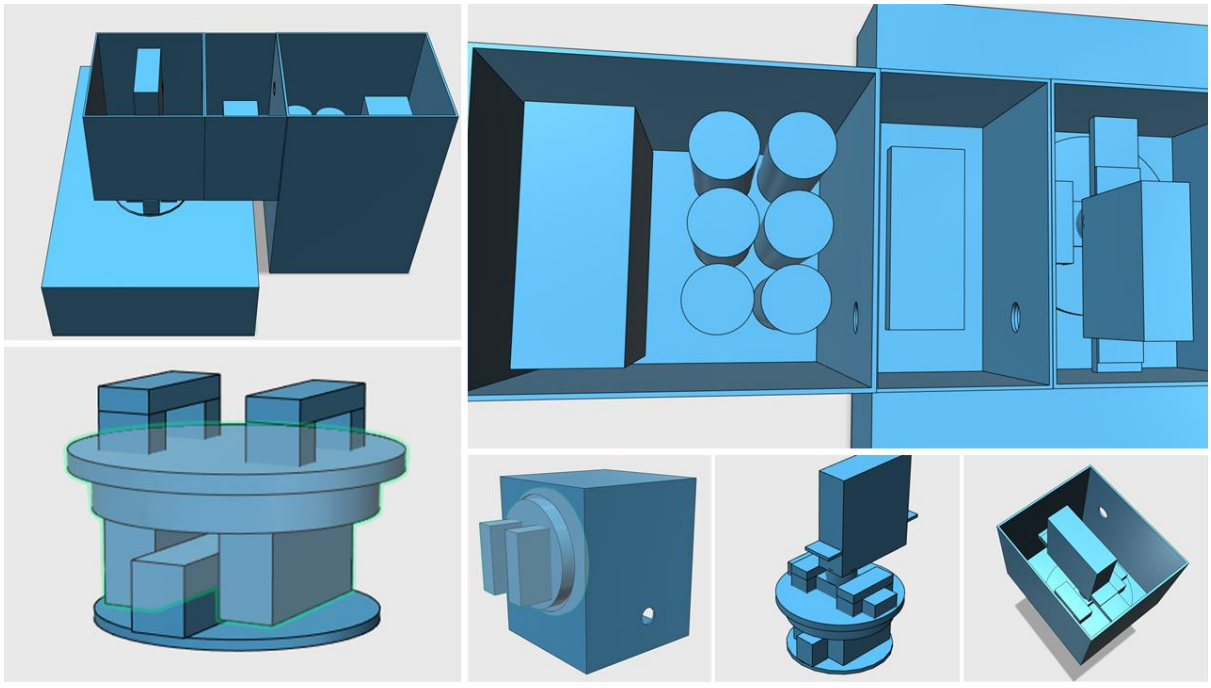
- Arduino IDE

3. 프로젝트 구현

3.1 목업 제작

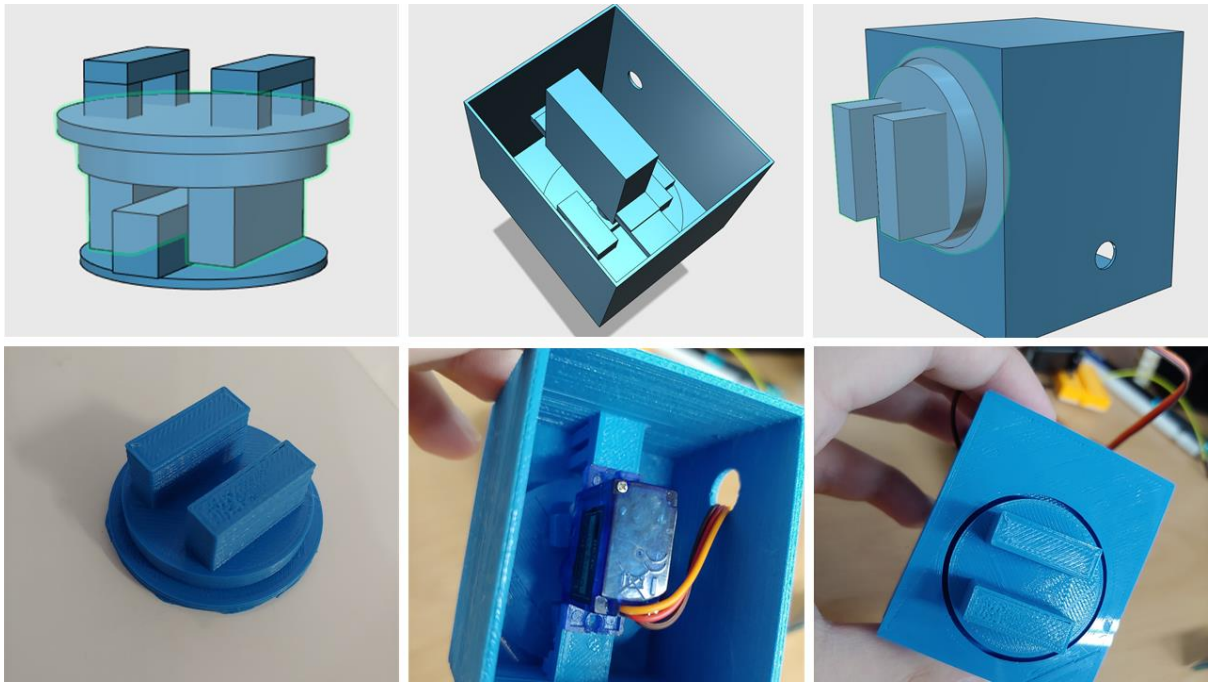


초기 도어락 형태 모델링



도어락 내부 모델링

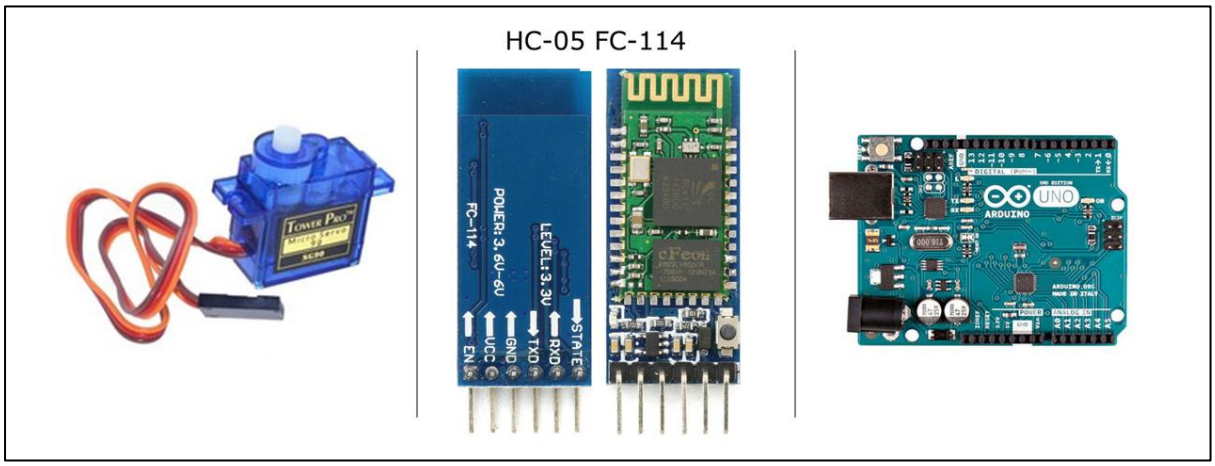
도어락 출력 문제



모델링 프린팅

Ultimaker 3d printer 를 통해 출력을 진행했다. 초기 도면출력 파일의 세부설정이 이루어지지 않아서 출력이 제대로 진행되지 않은 모습이다. 내부 벽면의 필라멘트 양과 보조출력 세팅을 통해서 위와 같이 성공적으로 3d 프린팅을 할 수 있었다.

3.2 도어락 임베디드 개발



모터, 블루투스 모듈, 아두이노 우노

Hardware:

Arduino Uno

Bluetooth HC-05 module

Arduino SG-90 servo motor

Arduino Uno:

Microcontroller: ATmega328

Operating Voltage: 5V

Input Voltage (recommended): 7 - 12V

Input Voltage (limits): 6 - 20V

Digital I/O Pins: 14 (of which 6 provide PWM output)

Analog Input Pins: 6

DC Current per I/O Pin: 40 mA

DC Current for 3.3V Pin: 50 mA

Flash Memory: 32 KB of which 0.5 KB used by bootloader

SRAM: 2 KB

EEPROM: 1 KB

Clock Speed: 16MHz

아두이노 우노는 위와 같은 스펙을 가지고 있다.

이 중 보안 코드, AES KEY, 비밀번호 등과 같은 중요한 데이터를 flash memory 에 저장하게 되면, 아두이노가 꺼진다면 이 데이터들이 모두 다 날라가게 된다. 이런 중요한 데이터가 날라간다면 문제가 많이 발생하게 되어 이 데이터들은 EEPROM 에 저장한다. EEPROM 에 저장하고 읽어온다면 아두이노가 꺼졌다가 켜져도 해당 데이터를 읽어 올 수 있어 안정적인 작동을 할 수 있다.

Data	Memory Addr	notes	Data	Memory Addr	notes	Data	Memory Addr	notes
AES key	0	1~255	password	20	0~9	SecureKey	40	2~255
AES key	1	1~255	password	21	0~9	SecureKey	41	2~255
AES key	2	1~255	password	22	0~9	SecureKey	42	2~255
AES key	3	1~255	hasSecureKey	23	0/1	SecureKey	43	2~255
AES key	4	1~255	SecureKey	24	2~255	SecureKey	44	2~255
AES key	5	1~255	SecureKey	25	2~255	SecureKey	45	2~255
AES key	6	1~255	SecureKey	26	2~255	SecureKey	46	2~255
AES key	7	1~255	SecureKey	27	2~255	SecureKey	47	2~255
AES key	8	1~255	SecureKey	28	2~255	SecureKey	48	2~255
AES key	9	1~255	SecureKey	29	2~255	SecureKey	49	2~255
AES key	10	1~255	SecureKey	30	2~255	SecureKey	50	2~255
AES key	11	1~255	SecureKey	31	2~255	SecureKey	51	2~255
AES key	12	1~255	SecureKey	32	2~255	SecureKey	52	2~255
AES key	13	1~255	SecureKey	33	2~255	SecureKey	53	2~255
AES key	14	1~255	SecureKey	34	2~255	SecureKey	54	2~255
AES key	15	1~255	SecureKey	35	2~255	SecureKey	55	2~255
hasPassword	16	0/1	SecureKey	36	2~255			
password	17	0~9	SecureKey	37	2~255			
password	18	0~9	SecureKey	38	2~255			
password	19	0~9	SecureKey	39	2~255			

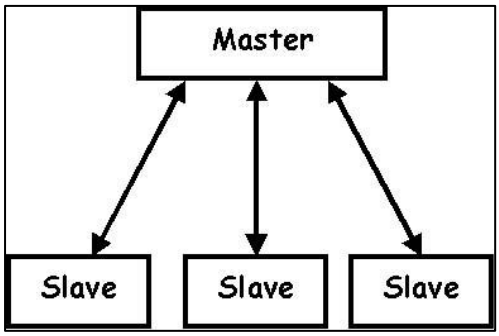
위 그림은 EEPROM 의 메모리 구조를 보여준다.

Servo Motor:

Servo motor 는 각도를 지정하여 해당 각도에 맞게 회전 하는 모터이다. SG-90 을 사용한 이유는 가격이 저렴하고, 문을 열기에 적당한 파워를 가지고 있다. 하지만 전력을 너무 많이 소모하여 블루투스에 충분한 전력이 들어가지 않아 오작동을 한다. 이를 방지하기 위하여 사용하지 않을 때는 servo motor 를 detach 하여 불필요한 전력 낭비를 방지한다.

Bluetooth:

휴대폰과 아두이노간 서로 블루투스 통신을 하기 위해 페어링을 한다. 페어링 시 휴대폰이 master, 아두이노가 slave 가 된다. 휴대폰에서 아두이노에게 페어링 연결을 시도한다. 연결이 완료된 후, 서로 통신한다.



위 그림과 같이 하나의 master 는 여러 개 (최대 7 개)까지의 slave 를 가질 수 있다. 하지만 아두이노 Bluetooth 모듈인 hc-05 는 1 명의 master 에 1 개의 slave 만 연결을 할 수 있다. 또한, 휴대폰에서 아두이노에게 연결을 하기 위해 아두이노를 slave 로 지정을 해 뒀다.

Security:

이 통신 방법 중 위험한 것은 주변에서 패킷을 듣고, 똑같은 패킷을 보내게 된다면 문이 열릴 수 밖에 없을 것이다. 이러한 경우는 보안이 매우 취약한 경우다. 이와 같은 경우는 자동차 키에서도 똑같은 문제점을 가지고 있다. 자동차 키가 보내는 패킷을 그대로 보내면 자동차 문을 열 수 있다. 이 방법을 해결하기 위해서 처음에는 패킷에 타임스탬프를 넣고, 그 패킷을 암호화하여 슥~에 보내면 매번 다른 패킷을 보낼 수 있어서 안전하다고 생각을 했지만, 아두이노에는 Real Time Clock(RTC)가 따로 내장되어 있지 않고, 다른 모듈을 설치해야 해서 다른 방법을 택했다. 이는 보안 코드를 사용하는 것이다.


* 이 표는 인터넷뱅킹, 텔레뱅킹 서비스에 동일하게 사용합니다.

01 NO. 01008322

1	10 97	2	24 70	3	10 97	4	39 18	5	06 68
6	82 76	7	97 10	8	82 76	9	87 33	02	75 03
11	03 16	12	70 00	13	90 16	14	93 05	15	04 75
16	27 74	17	64 72	18	27 74	19	87 73	20	28 79
21	00 01	22	85 35	23	00 01	24	84 65	25	27 21
26	29 70	27	33 82	28	29 70	29	24 14	30	83 17

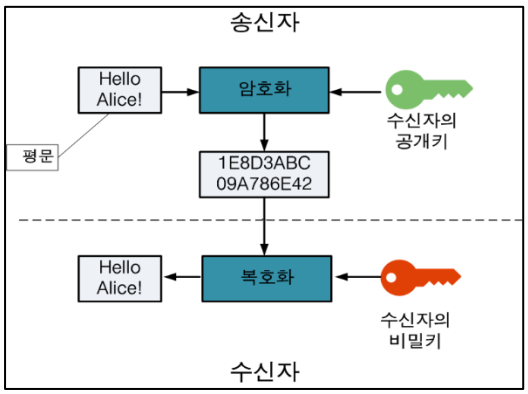
* 이 카드가 타인에게 노출되지 않도록 주의하시고, 분실하시면 즉시 신고 바랍니다.
* 고객상담전화 : 지역번호 없이 1588-1599

보안 코드는 우리가 흔히 알고 있는 은행의 보안 카드와 비슷한 역할을 한다. 위 그림과 같이 해당 인덱스에 임의의 숫자들을 해당한다. 이 해당 된 번호를 처음에 공유를 한 뒤, 인덱스를 질문하면 인덱스에 해당하는 숫자를 답하고, 확인하는 방법을 이용하여 보안을 강화했다. 이 보안 코드는 휴대폰에서 만들어 암호화하여 아두이노에게 전송한다. 이 보안 코드는 비밀번호를 이용하여 재설정 할 수 있다.

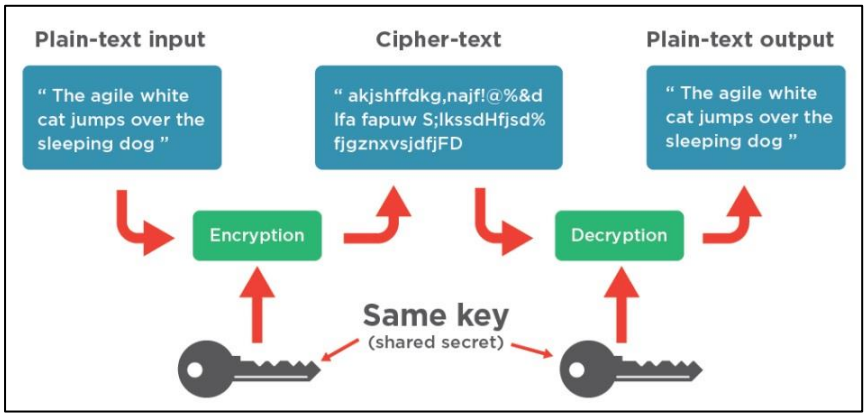
휴대폰에서 아두이노에게 보안 코드를 받아오기 위해서는 아두이노에 저장되어 있는 비밀번호를 전달해야 한다. 이 비밀번호는 최초의 등록자가 비밀번호를 정하여 아두이노에게 전송해주면, 그

비밀번호를 아두이노에서 저장을 해 둔다. 이 비밀번호는 저장된 보안 코드를 통하여 재설정이 가능하다.

이 방법 이외에도 데이터 암호화를 이용해 서로간의 패킷을 아예 읽지를 못하게 만들었다. 데이터 암호화 방식으로는 AES 와 RSA 를 동시에 사용했다.



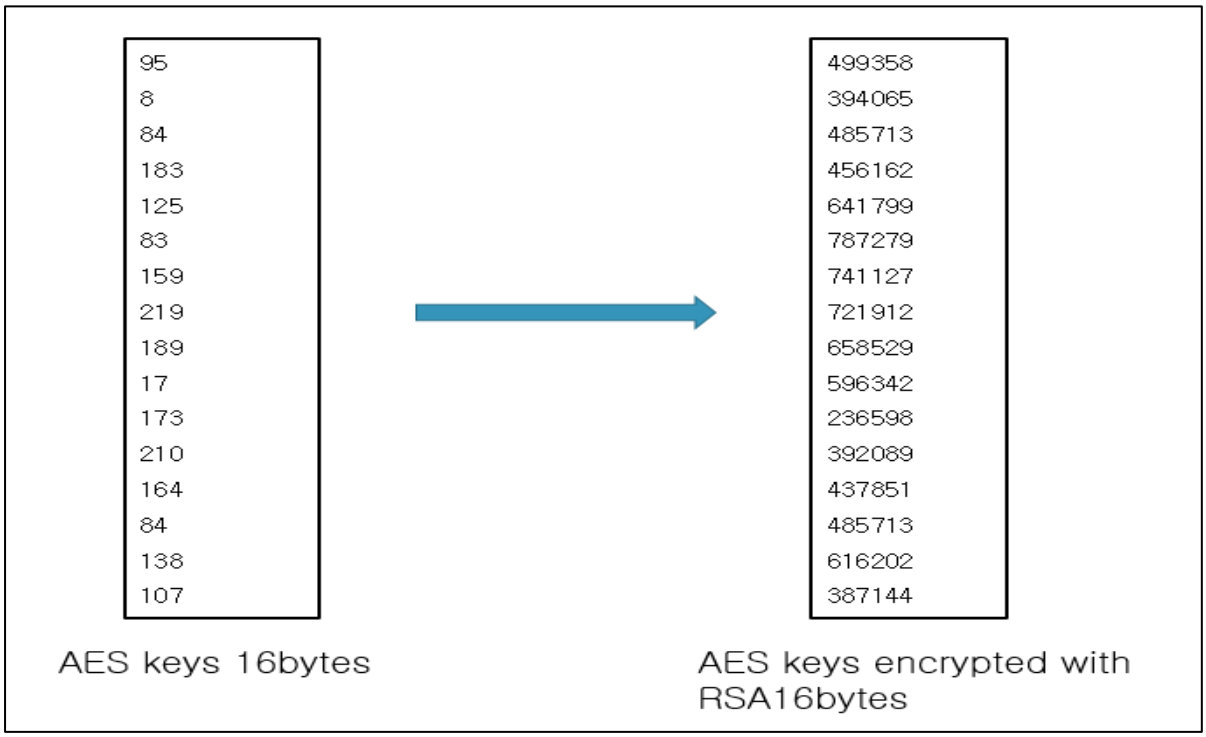
- **RSA:** RSA 는 현재 ssh 연결, 은행 업무 등에서 사용되는 유명한 암호화 방식이다. RSA 는 비대칭 키를 이용하여 암호화 한다. 비대칭 키란 송신자와 수신자가 다른 암호화 키를 가지고 있으며, 송신자가 가지고 있는 key 를 가지고 암호화 하면 수신자가 가지고 있는 키만으로 복호화 할 수 있는 방식이다. 이때 송신자가 가지고 있는 키를 public key, 수신자가 가지고 있는 키를 private key 라고 한다. Public key 와 private key 를 만들어내는데 있어 서로 다른 두 소수를 이용해서 만들어낸다. 이 암호화 방식은 서로 다른 키를 사용해 매우 안전하고 암호화를 풀기 위해 아주 많은 시간이 든다. 하지만 계산이 복잡하여 아두이노에서 사용하기에는 부적합하다.



- **AES:** AES 암호화는 RSA 와 다르게 하나의 key 를 가지고 암호화 및 복호화를 이루어낸다. AES 는 매우 빠르고 안전하지만, 처음에 AES key 를 보내 줄 때 매우 위험하다. AES KEY 는 16Byte 로 고정했다.

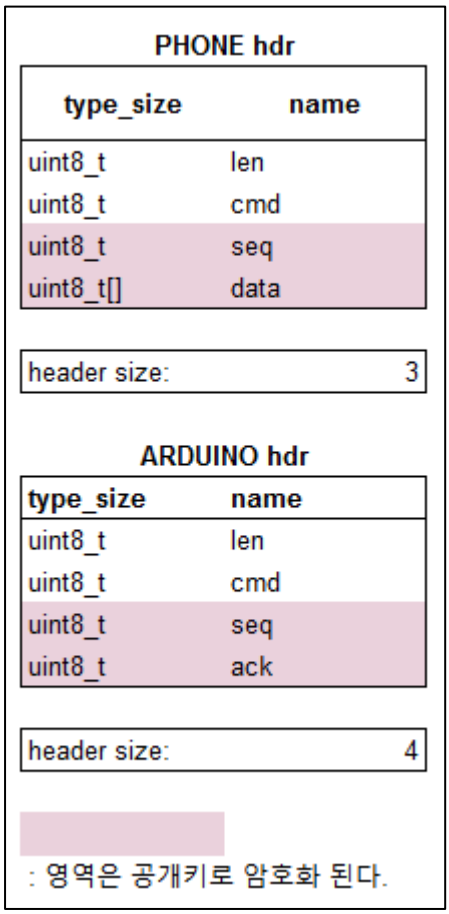
이 두 암호화 방식을 적절히 이용하여 휴대폰과 아두이노 사이의 통신이 이루어 진다. 처음에 아두이노에서 만들어진 AES KEY 를 휴대폰에 옮겨줘야 한다. 그냥 아무런 암호화 없이 AES

KEY 를 휴대폰에 보낸다면 주변 사람들이 듣고 다음 패킷들은 모두 해독할 수 있다는 문제점이 있다. 이 점에서 우리는 비대칭 키를 가지는 RSA 를 도입시켰다. RSA 의 public key 를 AES KEY 를 요청하는 명령과 함께 아두이노에 보내면 아두이노는 AES KEY 를 휴대폰의 RSA public key 를 이용하여 암호화 한다. 암호화 된 AES KEY 를 휴대폰에게 보내준다면 휴대폰에서 RSA private key 를 이용하여 AES KEY 를 복호화 할 수 있다. 그 후, 모든 통신은 AES 암호화를 이용한다.



위 사진은 AES key 16bytes 를 RSA public key 로 암호화 한 데이터이다.

Packet Header:



Command List:

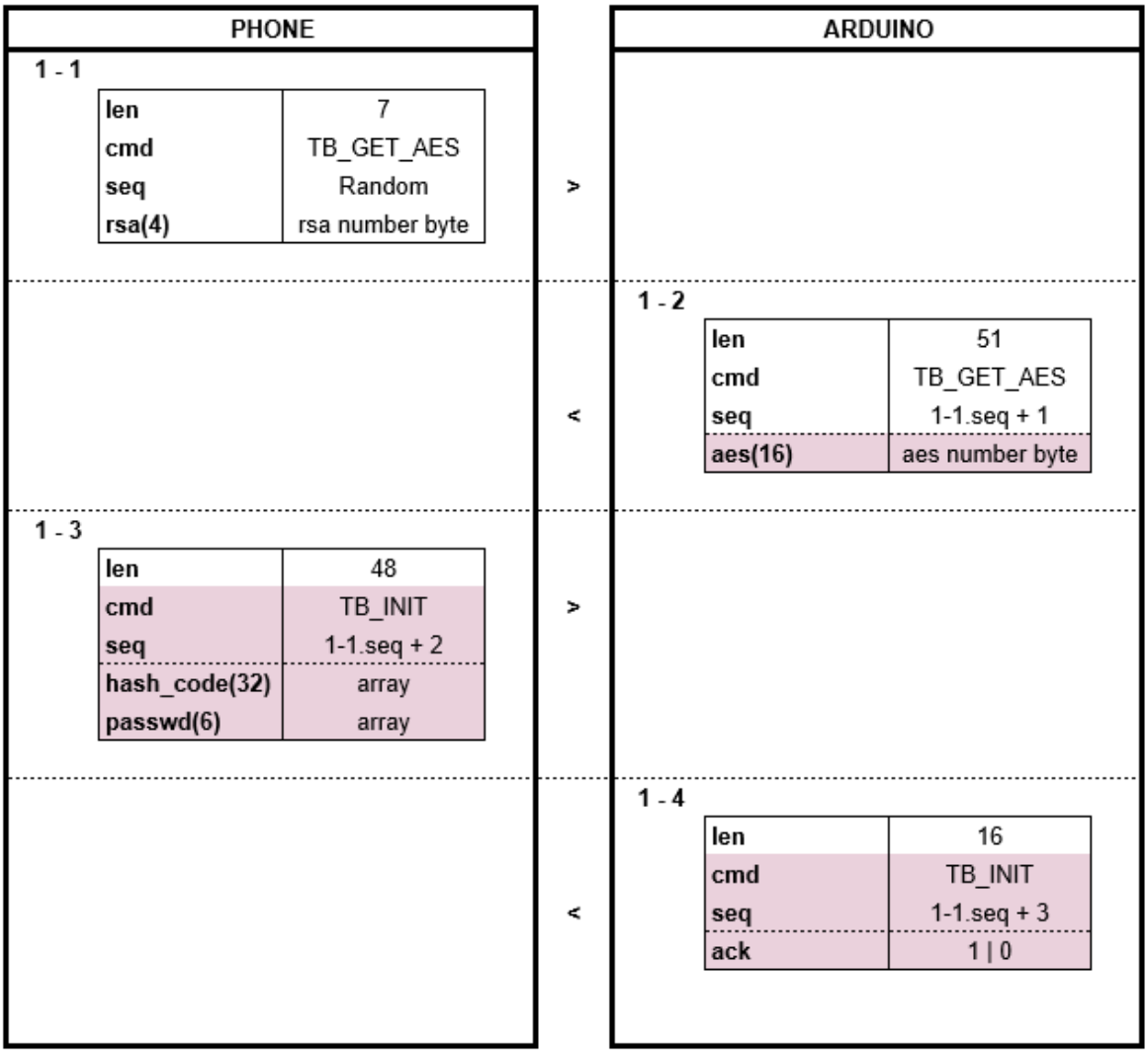
Name	Value	Flow
TB_GET_AES	0	1. 휴대폰의 RSA_PUB_KEY 를 안드로이드로 보낸다. 2. 안드로이드의 AES_KEY 를 휴대폰의 RSA_PUB_KEY 로 암호화하여 안드로이드에게 보낸다. 3. 휴대폰에서 안드로이드의 AES_KEY 를 저장한다

TB_INIT	1	1. 휴대폰에서 비밀번호와 보안 코드를 보낸다. 2-1 휴대폰에서 비밀번호가 저장되지 않은 경우에는 비밀번호와 보안 코드를 모두 저장한다. 2-2 휴대폰의 비밀번호가 안드로이드의 비밀번호와 일치하는 경우 보안 코드를 저장한다. 3. 비밀번호가 일치하면 1, 일치하지 않으면 0 의 ack 을 휴대폰에게 전송한다.
TB_OPEN TB_OPEN_ANS	2 3	1. 휴대폰에서 아두이노에게 OPEN 을 보낸다. 2. 아두이노에서 미리 저장된 보안 코드 중 임의의 값 3 개를 요구한다. (인덱스 값을 전달, 인덱스 순서에 맞게 값을 넣어줘야 함) 3. 핸드폰은 3 개의 보안 코드 값을 순서에 맞게 보낸다. 4. 순서에 맞는 보안 코드 값일 경우 문을 열어준다. 맞으면 응답 1 아니면 응답 0 을 보낸다.
TB_RESET_PASSWORD TB_RESET_PASSWORD_ANS	4 5	1. 휴대폰에서 아두이노에게 RESET_PASSWORD 을 보낸다. 2. 아두이노에서 미리 저장된 보안 코드 중 임의의 값 3 개를 요구한다. (인덱스 값을 전달, 인덱스 순서에 맞게 값을 넣어줘야 함) 3. 핸드폰은 3 개의 보안 코드 값을 순서에 맞게 보낸다.

		4. 순서에 맞는 보안 코드 값일 비밀번호를 바꾼다. 맞으면 응답 1 아니면 응답 0 을 보낸다.
TB_GET_CODE	6	1. 휴대폰이 아두이노에게 비밀번호를 포함한 CODE REQUEST 를 보낸다. 2-1. 비밀번호가 맞다면 아두이노는 휴대폰에게 보안 코드와 ack1 을 보낸다. 2-2. 비밀번호가 틀렸다면 아두이노는 휴대폰에게 ack 0 을 보낸다.

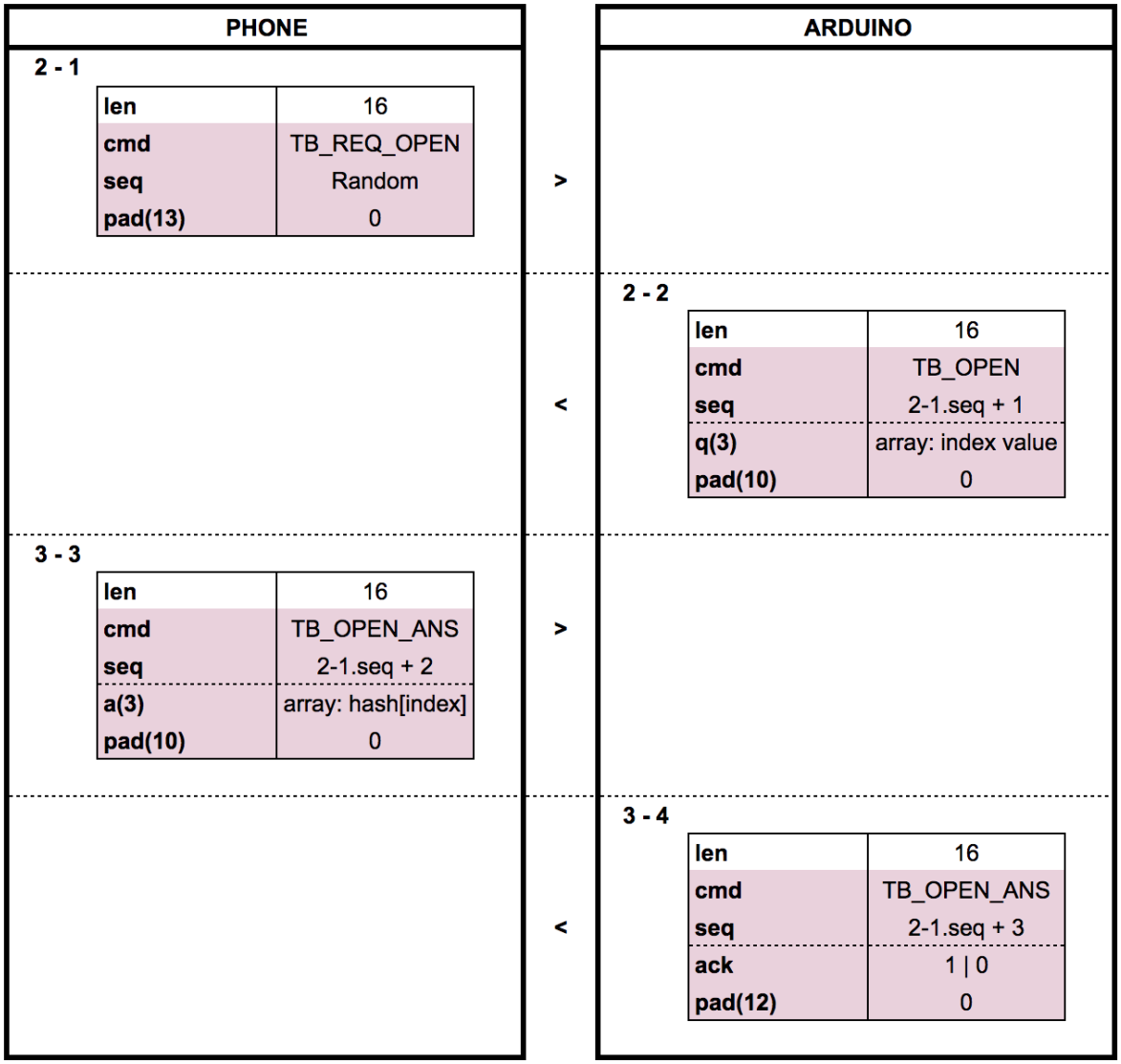
Scenario:

Scenario - 1: AES key 요청 및 도어락 등록



먼저 휴대폰의 비대칭 키의 공개 키를 아두이노에게 전송한다. 전송받은 아두이노에서 생성된 AES 키를 공개키로 암호화하여 응답한다. 휴대폰은 비공개 키를 이용해서 전달받은 AES 키를 저장한다. 2~255 의 수를 가지는 32 개의 보안 코드를 새로 생성하고 저장된 비밀번호 6 자리를 AES 키로 암호화하여 전송한다. 아두이노는 보안 코드와 비밀번호를 제대로 저장하면 ack 를 1 로 응답한다.

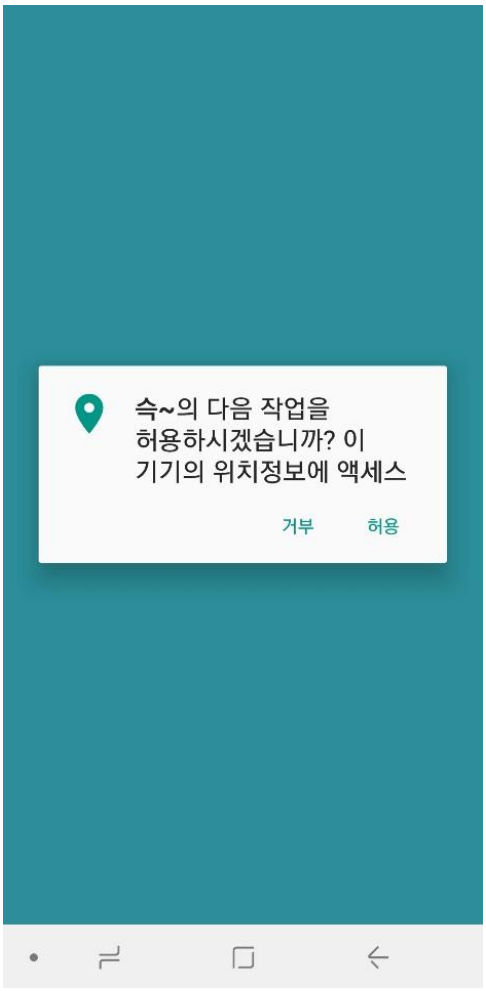
Scenario - 2: Open 요청



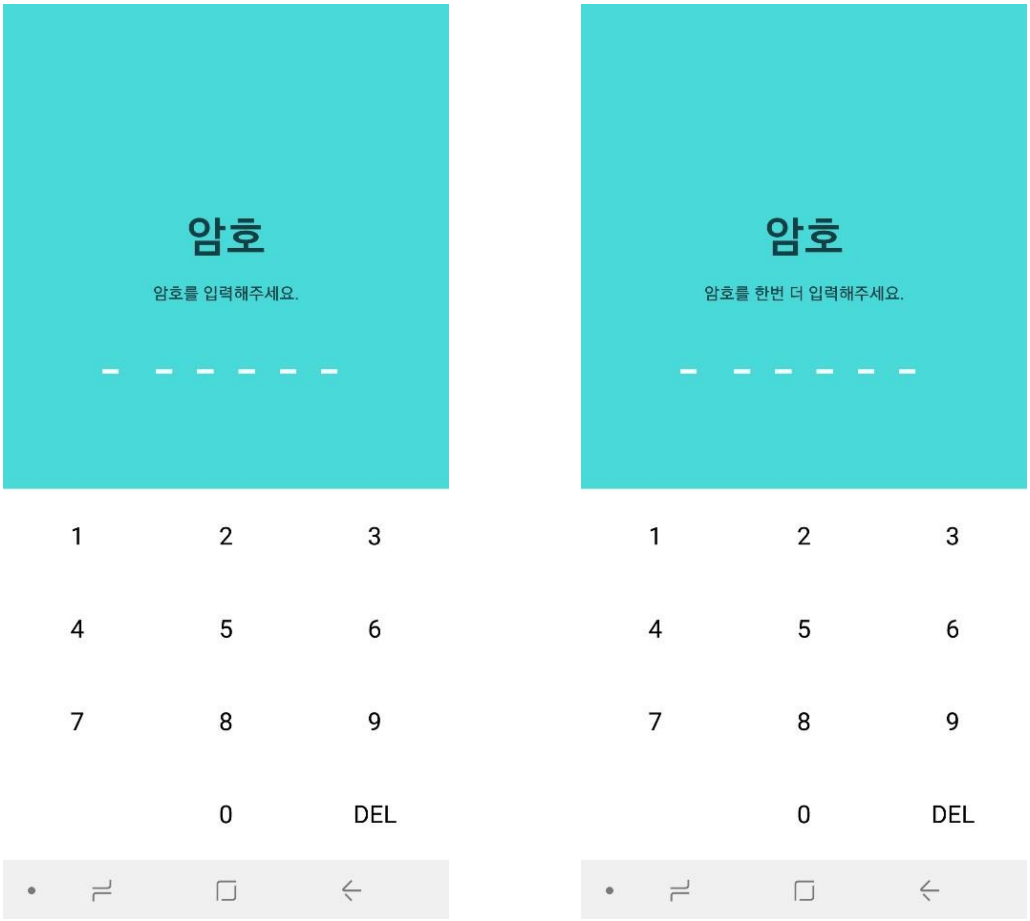
문을 여는 요청은 길이를 제외하고 모두 AES 로 암호화가 이루어진다. 핸드폰에서 문을 여는 요청을 하면 아두이노는 보안코드에서 32 개 중 3 개를 랜덤하게 선택하여 휴대폰에게 묻는다. 핸드폰은 저장된 아두이노가 묻는 인덱스에 해당하는 값을 패킷에 넣어서 응답한다. 아두이노에 저장된 보안 코드와 휴대폰이 전달한 값과 일치하면 문을 열고 ack 를 1 로하여 응답한다.

3.3 도어락 모바일 어플리케이션

어플리케이션 개발을 위해 2.3 절에서 언급했듯이 Android Studio 3.14, JAVA SDK 1.8, Android API 28 환경에서 진행되었고 사용한 스마트폰은 Galaxy S8 이다.



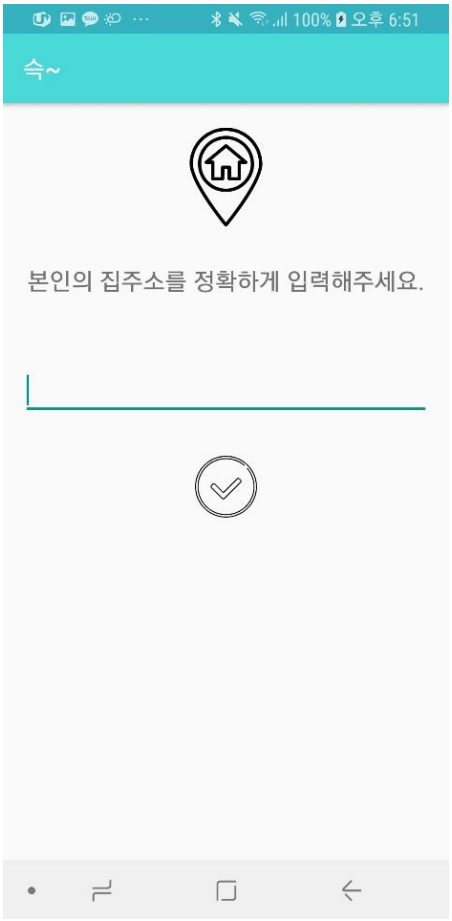
위 이미지는 어플리케이션을 처음 실행했을 때 사용자에게 Runtime Permission 을 요청하는 팝업을 띄우는 Activity 이다. 어플리케이션 동작을 위해서 BLUETOOTH, BLUETOOTH_ADMIN, ACCESS_COARSE_LOCATION, ACCESS_FINE_LOCATION, 총 4 개의 Permission 이 요구되고 사용자가 거부할 시 Permission 이 필요하다는 Toast 와 함께 어플리케이션이 종료되고, 허용할 시 다음 화면인 비밀번호 설정 Activity 로 넘어간다.

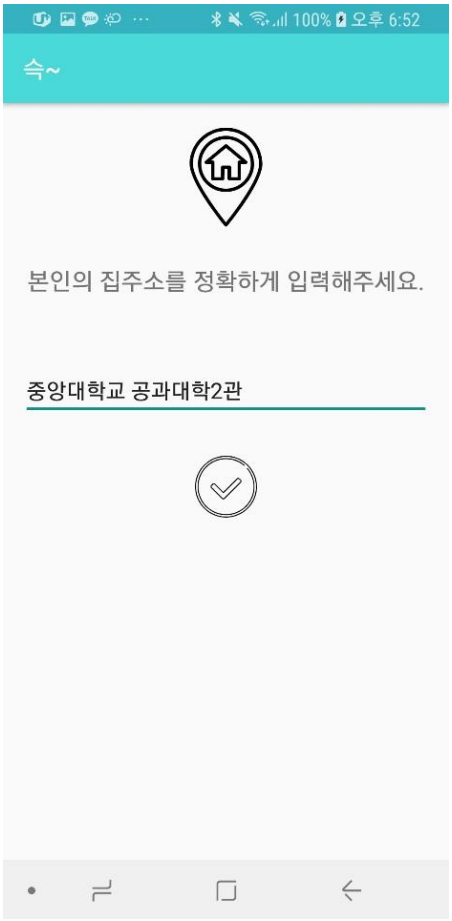


위 이미지들은 비밀번호 설정 Activity 로 왼쪽 Activity 에서 비밀번호 6 자리를 모두 입력하면 오른쪽 Activity 와 같이 "암호를 입력해주세요." 라는 텍스트가 "암호를 한번 더 입력해주세요." 로 변경되면서 다시 한 번 더 비밀번호를 입력하게 한다. 다시 입력한 비밀번호와 이전에 입력한 비밀번호가 일치할 시 사용자의 집 주소(위도 및 경도)를 설정하는 Activity 로 넘어가게 되고 일치하지 않을 시 다시 한번 더 비밀번호 입력을 요구한다.

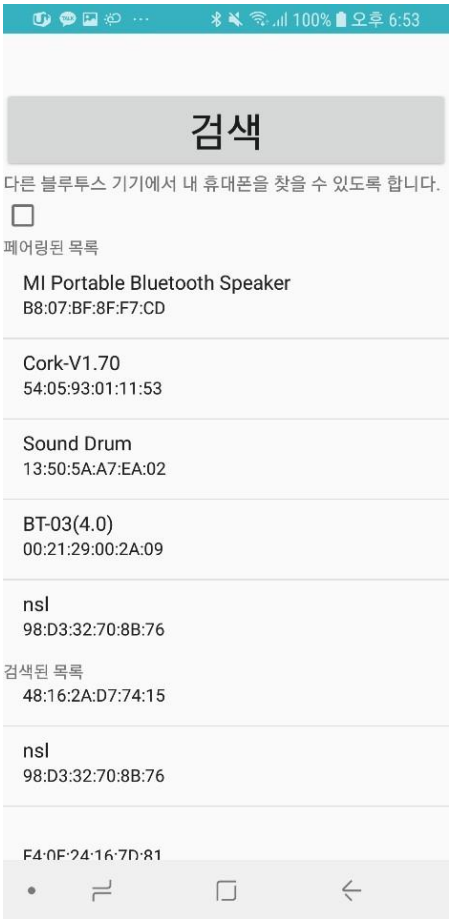
추후 어플리케이션을 재실행할 시 비밀번호 설정을 이미 마쳤다면 비밀번호 설정 Activity 가 아닌 비밀번호 확인 Activity 를 불러와 비밀번호를 입력 후 설정한 비밀번호와 일치하면 다음 Activity 로 넘어가고 일치하지 않으면 비밀번호 재입력을 요구한다.

비밀번호를 설정할 때 어플리케이션은 SharedPreferences 를 이용해 비밀번호를 저장한다. SharedPreferences 는 데이터를 간단하게 Key-Value 형태로 xml 파일에 저장한다.

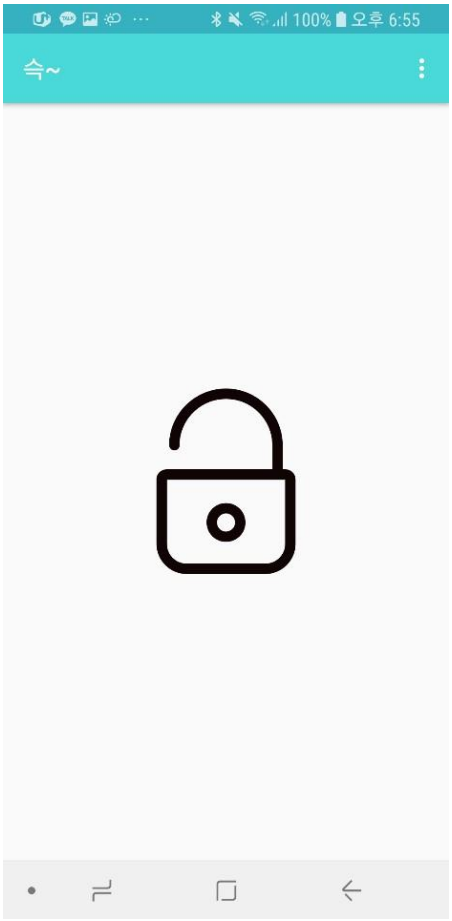


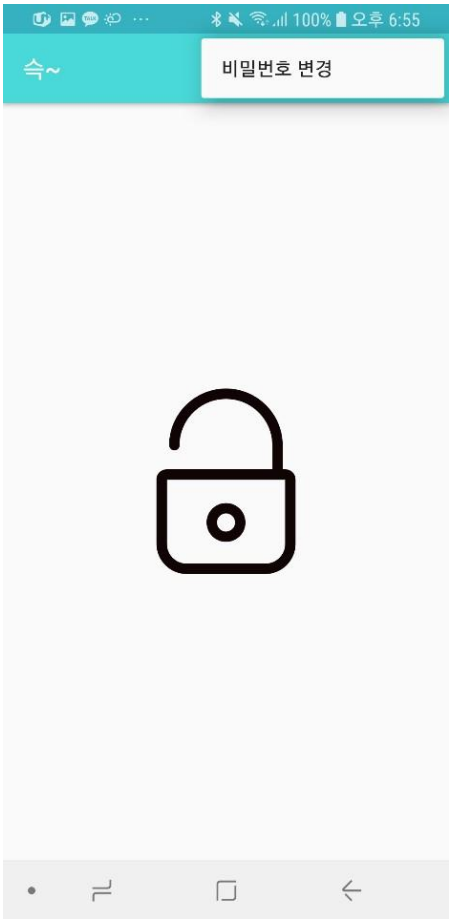


위 이미지들은 GPS 를 활용해 현재 사용자의 위치와 집 사이의 거리를 계산하기 위해 사용자의 집 주소를 입력하는 Activity 이다. Geocoding Service 를 이용해 사용자의 주소를 위도, 경도 좌표 값으로 변경해준다. Geocoding Service 는 구글에서 제공하는 서비스로 안드로이드 API 내에서 Geocoder 객체와 Address 객체를 통해 사용이 가능하다. 위와 같이 정상적인 주소를 입력 시 해당 주소에 상응하는 위도, 경도 좌표 값을 SharedPreferences 를 이용해 저장하고 비정상적인 주소 및 오타를 입력 시 해당 주소를 찾을 수 없다는 Toast 와 함께 재입력을 요구한다.



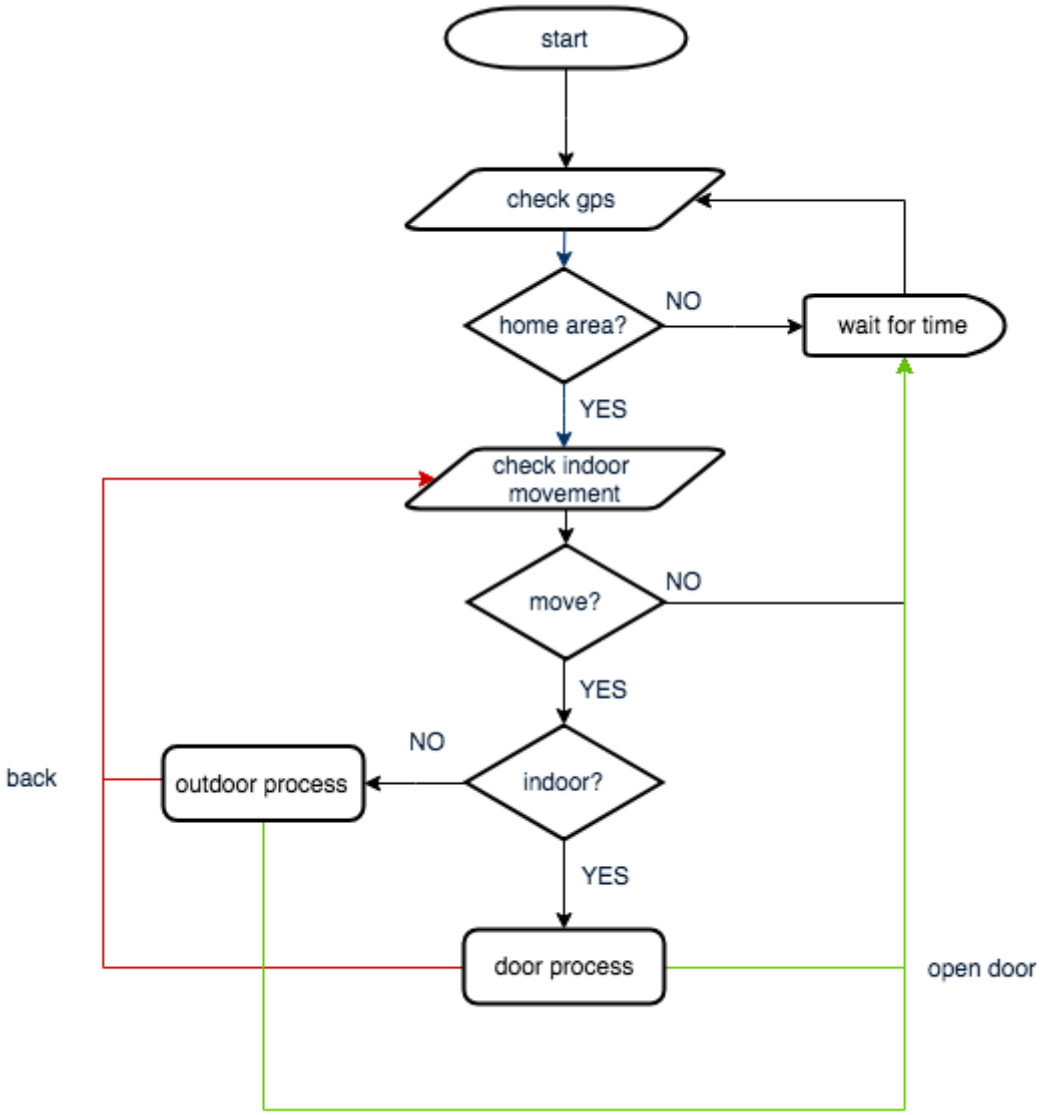
위 이미지들은 주소 설정 후 나타나는 Activity 들이다. 왼쪽 Activity 가 우선적으로 나타나고 페어링된 블루투스 기기 목록들을 보여준다. 검색 버튼을 클릭 시 하단의 "검색된 목록" 밑으로 현재 검색되는 블루투스 기기 목록들을 보여준다. "검색된 목록"에서 도어락 관련 블루투스 기기를 선택 시 페어링이 진행되고 페어링된 목록에 해당 기기를 추가한다. 이후 페어링된 목록에서 도어락 관련 블루투스 기기를 선택하면 "도어락 설정 중~"이라는 텍스트가 띄워진 Activity 로 넘어가 아두이노가 자신의 AES 키를 휴대폰의 RSA Public Key 로 암호화하여 스마트폰에 전달을 마칠 때까지 대기한다. 전달이 끝나면 도어락 서비스 Activity 로 넘어간다.





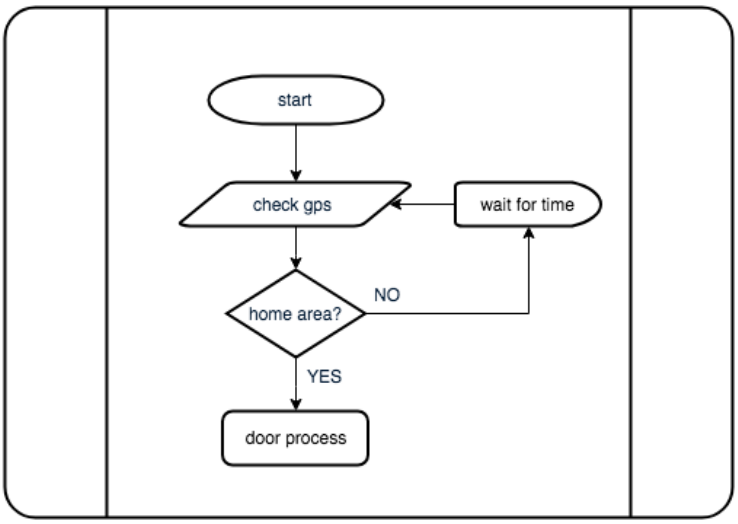
위 이미지들은 시나리오에 따른 도어락 서비스가 실행 중일 때 나타나는 Activity 들이다. 왼쪽 Activity 는 "도어락 설정 중~" Activity 가 작업을 끝낸 후 나타나는 Activity 이다. 해당 Activity 가 나타났다는 것은 스마트폰 내에서 백그라운드로 도어락 서비스가 동작하고 있다는 것을 의미한다고 볼 수 있고 시나리오에 따라 도어락 문 개폐 여부를 결정한다. 오른쪽 Activity 는 오른쪽 위 메뉴 버튼을 클릭 시 비밀번호 변경 항목이 나타나는 것을 보여주고 있고 해당 항목을 클릭 시 앞서 봤던 비밀번호 설정 Activity 로 넘어가 새로운 비밀번호를 설정하고 다시 도어락 서비스 Activity 로 돌아온다.

3.4 도어락 기술 개선 및 개발

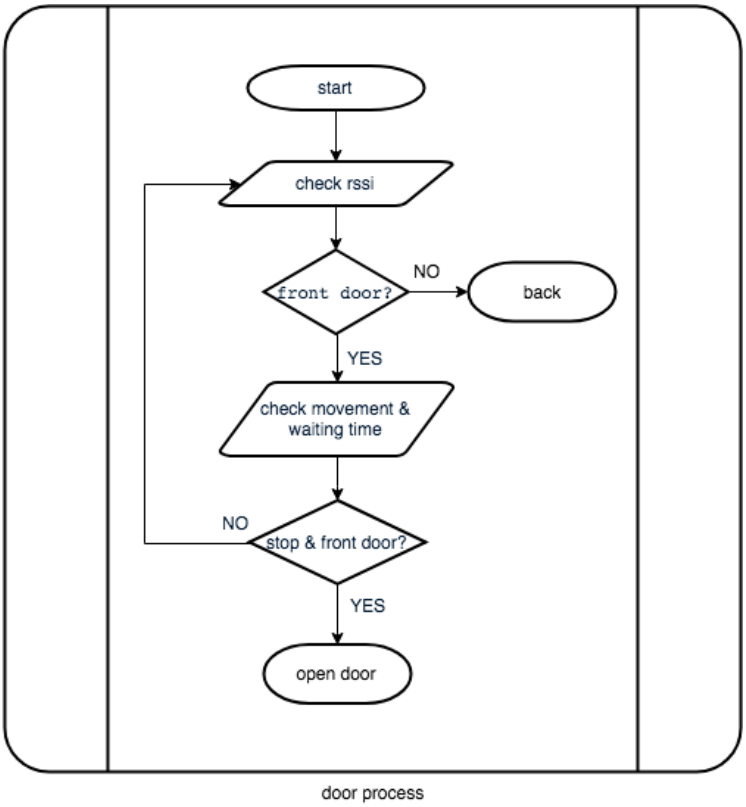


Main Process

개선된 도어락 자동 개폐 서비스는 3 개의 process 로 구성되어 있다. 집 구역을 벗어난 경우 동작을 담당하는 outdoor process 와 집 구역에서 동작을 담당하는 door process 가 있다. 먼저 GPS 신호를 확인해서 집 구역 내에 있는지 일정시간을 간격으로 확인한다. 만약 집 밖에 있다면 일정 딜레이 이후 집 구역에 도착했는지 확인한다. 반대로 집 구역 안이면 움직임을 먼저 파악한다. 움직임을 파악하기 위해서 가속도 센서를 확인한다. 100 번 샘플링을 두차례 시도하여 변화량을 측정해서 일정 임계값을 두 번 넘기게 될 경우 움직이는 것으로 판단한다. 실내인지 판단하는 방법은 자기장 센서와 GPS 센서에서 확인되는 인공위성 개수와 자기장센서의 변화량을 통해서 판단한다. 실내의 경우 인공위성의 개수가 일정 수 이하로 확인되며, 자기장센서는 실내에서 변화량을 가지게 된다.



Outdoor process 는 GPS 센서를 통해서 집 구역에 도착하면 door process 로 바로 진행하게 된다. Main process 와의 차이는 실내인지와 움직임을 파악하는 과정이 생략되었다. 왜냐하면 outdoor process 는 집 밖을 벗어나서 움직이고 있다고 판단이 된 상태이기 때문이다.



Door process는 간단하게 rssi 신호의 세기와 움직임을 기준으로 진행된다. 일정한 임계 신호 세기 값을 넘는지 확인한 이후 움직임이 멈추면 문 앞에 있다고 간주하여 문을 여는 신호를 도어락에게 전달한다. 만약 문 앞이 아니다면 실내에서 움직이는지 파악을 계속해서 진행한다.

4. 결론

결과적으로 프로젝트 제안서에 기술했던 부분들을 모두 완성하였다. 휴대폰과 "슥~"을 서로 연결하고, 외부의 악의적 행동들로부터 안전한 방식으로 패킷을 자동으로 주고받으며 여러 센서들을 통해 문의 개폐 여부 및 실내외 구분을 잘 이루어낸 프로젝트이다. 문이 열리고 닫히는 것은 당연한 것이고, 이 프로젝트에서 가장 중요하게 여겼던 부분은 실내외 판단과 보안성이었다. 보안성에 대해 먼저 말하자면, 기존 가정집에 존재하는 도어락에 부착되는 장비이고 문의 개폐 여부에 직접적인 영향을 끼치므로 보안성이 낮다면 제 3 자가 문을 열 수도 있는 최악의 사건이 발생할 수 있다. 따라서 이러한 사건을 예방하기 위해 보안 코드, 비밀번호, AES 암호화, RSA 암호화 같은 방식을 이용하여 보안성을 높였다. 실내외 판단의 경우, 기존의 GPS 만을 이용하여 실내외를 판단하는 방식은 사용자가 실내에 위치할 시 신호를 제대로 받지 못하는 경우가 발생한다. 이로 인해 값이 튀어 사용자가 실외에 있다고 판단되어 집안에 있음에도 불구하고 문이 열리는 경우가 발생한다. 우리는 실내외 구분과 관련된 여러 논문들을 읽어 보고 활용할 수 있는 스마트폰 내장 센서들을 이용하여 이런 점들을 보완하였다.

한 학기 동안 팀원들 간의 협업을 통해 열심히 프로젝트를 진행하였지만 최종 데모에서 잘 동작하지 않아 아쉬움이 있다. 많은 trouble-shooting 을 통해 여러 문제점들을 해결하였지만 근본적인 문제인 블루투스 모듈 HC-05 의 불안정성을 극복해내지 못하였다. 여러 AT 명령어들을 제대로 실행할 수 없었으며, 연결 진행 후 SPP Protocol Socket 이 정상적으로 살아있음에도 불구하고 휴대폰에서 전송하는 패킷을 아두이노에서 받지 못하는 경우들이 발생하였다. 반대의 경우도 마찬가지였고 아두이노와 HC-05 모듈 사이의 시리얼 통신에서도 스마트폰과 HC-05 모듈 사이에서 아무런 통신 과정을 진행하지 않는데도 쓰레기 값이 계속해서 수신되었다. 쓰레기 값에 대해서 예외 처리를 하고, 통신이 제대로 되지 않는 상황에서는 계속해서 재연결 후 통신을 시도하고 여러 방법을 통해 표면적으로는 문제들을 해결하였지만 이러한 처리들로 인해 문의 개폐에 있어 많은 지연이 발생하였다.

추후에 "슥~"에 대해 보완할 점이 있다면 블루투스 클래식 모듈인 HC-05 자체를 BLE 모듈로 변경하는 것이다. 더 많은 AT 명령들을 내릴 수 있고 통신의 안정성에 대해서도 여러 Reference 들을 통해 확인할 수 있었기 때문이다. 비용적인 면에서 부담이 더 생기겠지만 프로젝트의 안정성과 성능을 높일 수 있다고 확신하기 때문에 필수 보완 사항이라고 생각한다.