Server

Overview

+ Contains Private Keys and Encryption Algorithm
+ Responsible for Encrypting the Data
+ Contains Unique Id for All locks
+ Contains the user information.
+ Responsible For assigning locks to the user which the user has selected.

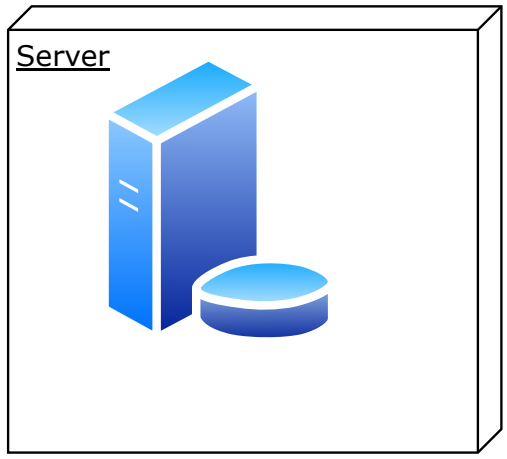
App

Application

Overview

+ User can signin on app and app will show all the locks available and this information will be get from Server.
+ User can register the lock on the app and app sends the info of user and lock on the server, so server can make a pairing for this lock
+ When user open the app, the app starts to search for locks and it sends all the data of the found devices to server, server checks that what unique ID of the lock is assigned to the user and if in the devices that unique ID is there the server sends command to application to connect to the following lock.

smartLock



1)Server will have Unique IDs for all the lock devices during production

2)At each location the lock is placed, server will also have that location coordinates. Reason for this is that when user registers the lock on the app, obviously the lock will shown in on a location to user so user can register it, that information is sent to server with location and server will have unique IDs before hand

3)when user opens an app for the first time to connect to lock, all of the locks data in the vicinity of the app is sent to server and then server compares the locks data to the one stored with this particular user information to see even if this user has registered any lock and if the information matches the app connects to that particular lock.

4)Another important aspect of the server will be to perform encryption and decryption on all the messages, none of the messages will run locally on the app.

5) When user unlocks his designated SmartLock, the app notifies the server and timing for how long the user can use the bike, starts on the server and when the time is exceeded the server removes the user from particular SmartLock from its database.

6)Server is basically handling all of the main decision making.

7)This also minimizes the security concerns.

Connection to Lock

Server

Server Receives the ble devices data from the App, and starts to match the unique ID, all the devices unique ID to the one that is registered to this user credentials.

Server

If the credentials matched with the one stored in the server for the particular user, server sends command to the app to connect to following ble device with following unique ID.

9

smartLock

6

server Start to receive the discovered ble devices from user

7

Connection Proccess started and Connected

3

User comes near the lock and opens the app

4

App Starts to search for all the available devices in the vicinity of the lock.

5

As shown in Fig 1, the app starts to show discovered BLE devices, and when the app discovers ble device with Smart_Lock name it extracts the advertisement packet from it as shown in Fig 2 and sends it to the server for all the ble devices with the name Smart_Lock.

8

App receives the command to connect to the following unique ID. As the app has already disected the advertisement packet it has all the unique ID for all the Smart_Lock in vicinity and it connects to the one which server sent.

App

Application

User

2

1

Following images are for reference to give an IDEA

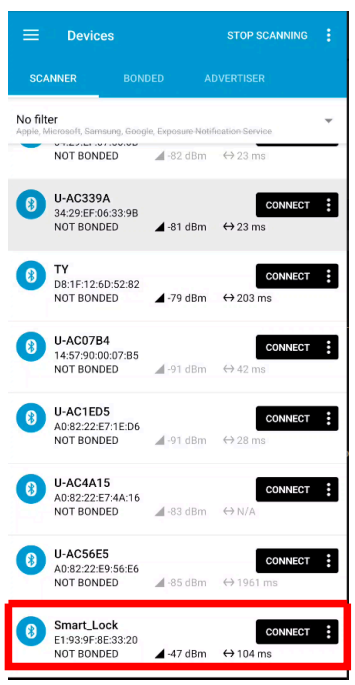


Fig 1

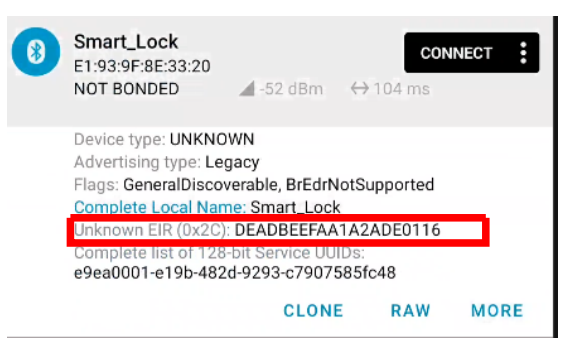
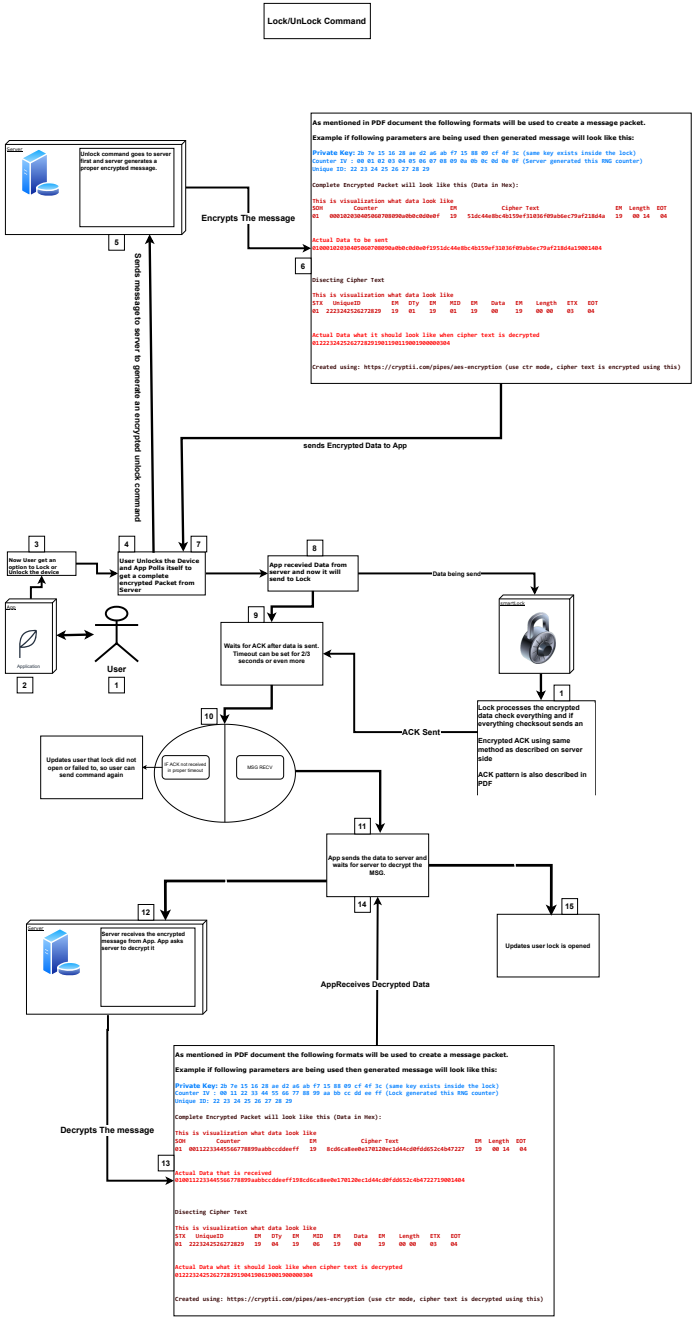


Fig 2


Lock/UnLock Command
Unlock command goes to server first and server generates a proper encrypted message.
Encrypts The message
5
6
As mentioned in PDF document the following formats will be used to create a message packet.
Example if following parameters are being used then generated message will look like this:
Private Key: 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c (same key exists inside the lock)
Counter IV : 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f (Server generated this RNG counter)
Unique ID: 22 23 24 25 26 27 28 29
Complete Encrypted Packet will look like this (Data in Hex):
This is visualization what data look like
SOH Counter EM Cipher Text EM Length EOT
01 00010203040506070809a0b0cd0de0f 19 51dc44e8bc4b159ef31836f09ab6ec79af218d4a 19 00 14 04
Actual Data to be sent
0100102030405060708090a0b0cd0de0f1951dc44e8bc4b159ef31836f09ab6ec79af218d4a19001404
Dissecting Cipher Text
This is visualization what data look like
STX UniqueID EM DTy EM MID EM Data EM Length ETX EOT
01 22232425262728 29 19 01 19 01 19 00 19 00 00 03 04
Actual Data what it should look like when cipher text is decrypted
01222324252627282919011901190019000304
Created using: https://cryptii.com/pipes/aes-encryption (use ctr mode, cipher text is encrypted using this)
sends Encrypted Data to App
sends message to server to generate an encrypted unlock command
3
4
7
Now User get an option to Lock or Unlock the device
User Unlocks the Device and App Polls itself to get a complete encrypted Packet from Server
8
App received Data from server and now it will send to Lock
Data being send
Application
User
2
1
9
Waits for ACK after data is sent. Timeout can be set for 2/3 seconds or even more
1
Lock processes the encrypted data check everything and if everything checksout sends an
Encrypted ACK using same method as described on server side
ACK pattern is also described in PDF
ACK Sent
10
Updates user that lock did not open or failed to, so user can send command again
IF ACK not received in proper timeout
MSG RECV
11
App sends the data to server and waits for server to decrypt the MSG.
14
15
12
Server receives the encrypted message from App. App asks server to decrypt it
Updates user lock is opened
AppReceives Decrypted Data
Decrypts The message
13
As mentioned in PDF document the following formats will be used to create a message packet.
Example if following parameters are being used then generated message will look like this:
Private Key: 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c (same key exists inside the lock)
Counter IV : 00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff (Lock generated this RNG counter)
Unique ID: 22 23 24 25 26 27 28 29
Complete Encrypted Packet will look like this (Data in Hex):
This is visualization what data look like
SOH Counter EM Cipher Text EM Length EOT
01 00112233445566778899aabbccddeeff 19 8cd6ca8ee0a17012bec1d44cd0fdd652c4b47227 19 00 14 04
Actual Data that is received
0100112233445566778899aabbccddeeff198cd6ca8ee0a17012bec1d44cd0fdd652c4b4722719001404
Dissecting Cipher Text
This is visualization what data look like
STX UniqueID EM DTy EM MID EM Data EM Length ETX EOT
01 22232425262728 19 04 19 06 19 00 19 00 00 03 04
Actual Data what it should look like when cipher text is decrypted
01222324252627281919061900190019000304
Created using: https://cryptii.com/pipes/aes-encryption (use ctr mode, cipher text is decrypted using this)