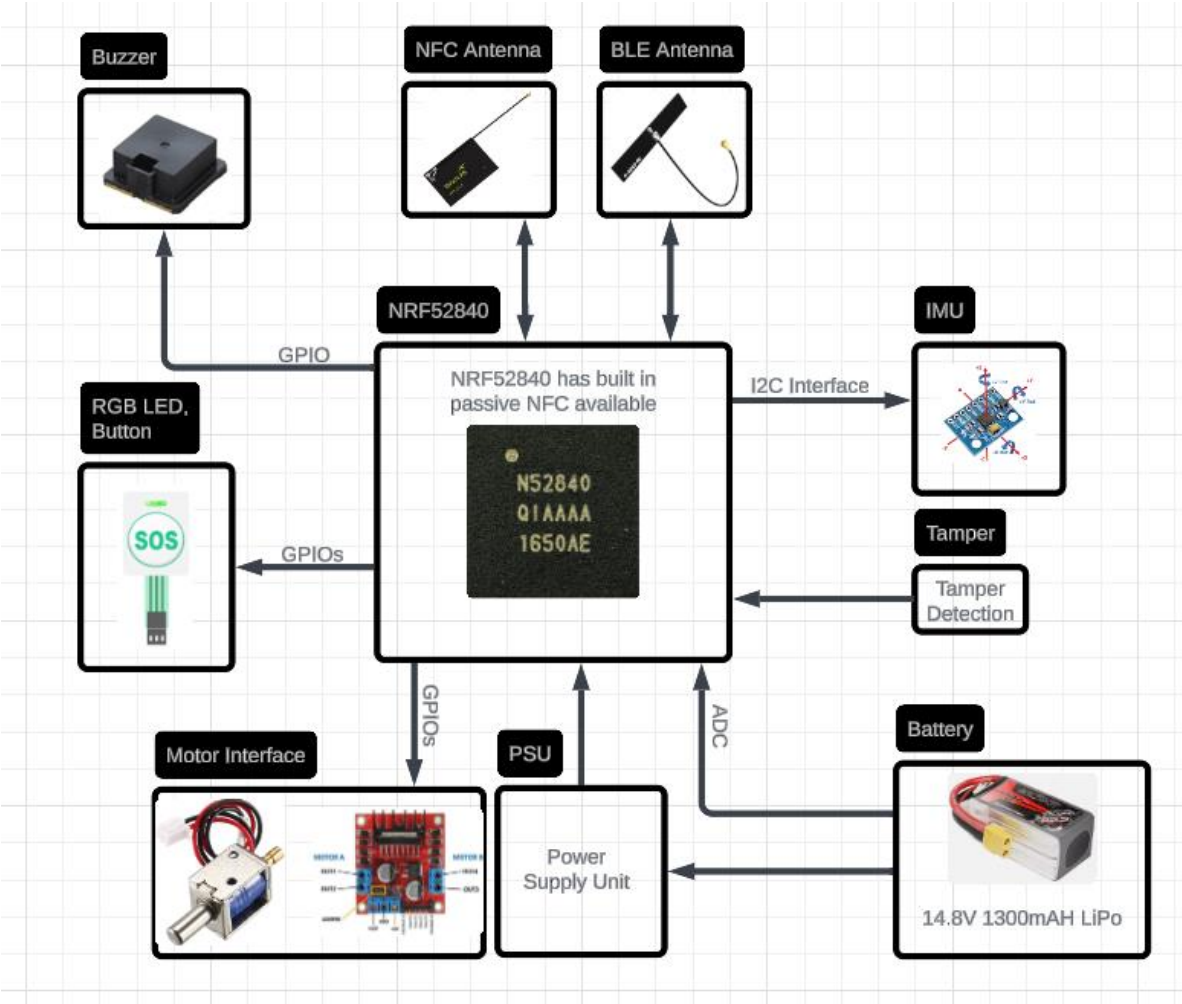
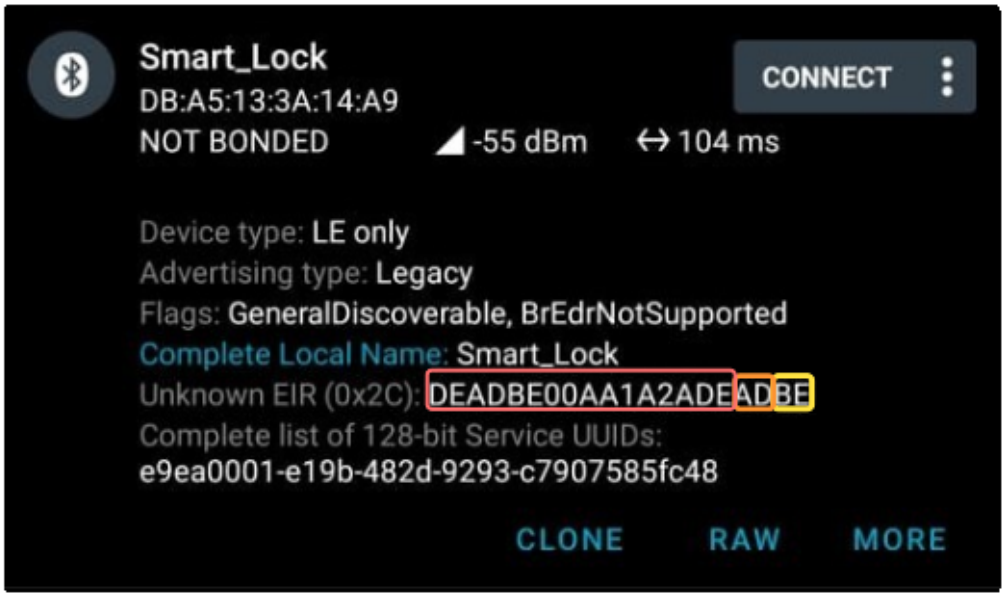
Smart Lock Complete System

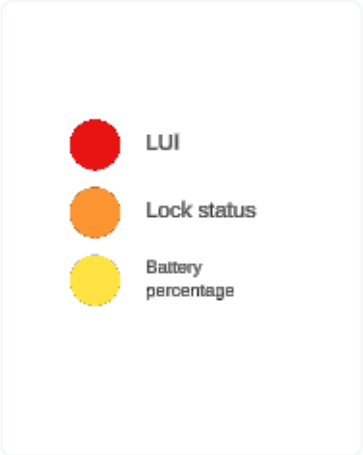
Block Diagram:



Advertise Packet:

1)Smart Lock will advertise itself with following name "Smart_Lock"
2)The Advertisement Packet will contain following information:
* Lock Unique Id (LUI) (Data Length: 8bytes)
* Lock Status (Whether Lock is Open or Close) (Data Length: 1byte)
* Battery Percentage (Data Length: 1byte)
All the Data is in HEX form which will be parsed by the App





Encryption Explained:

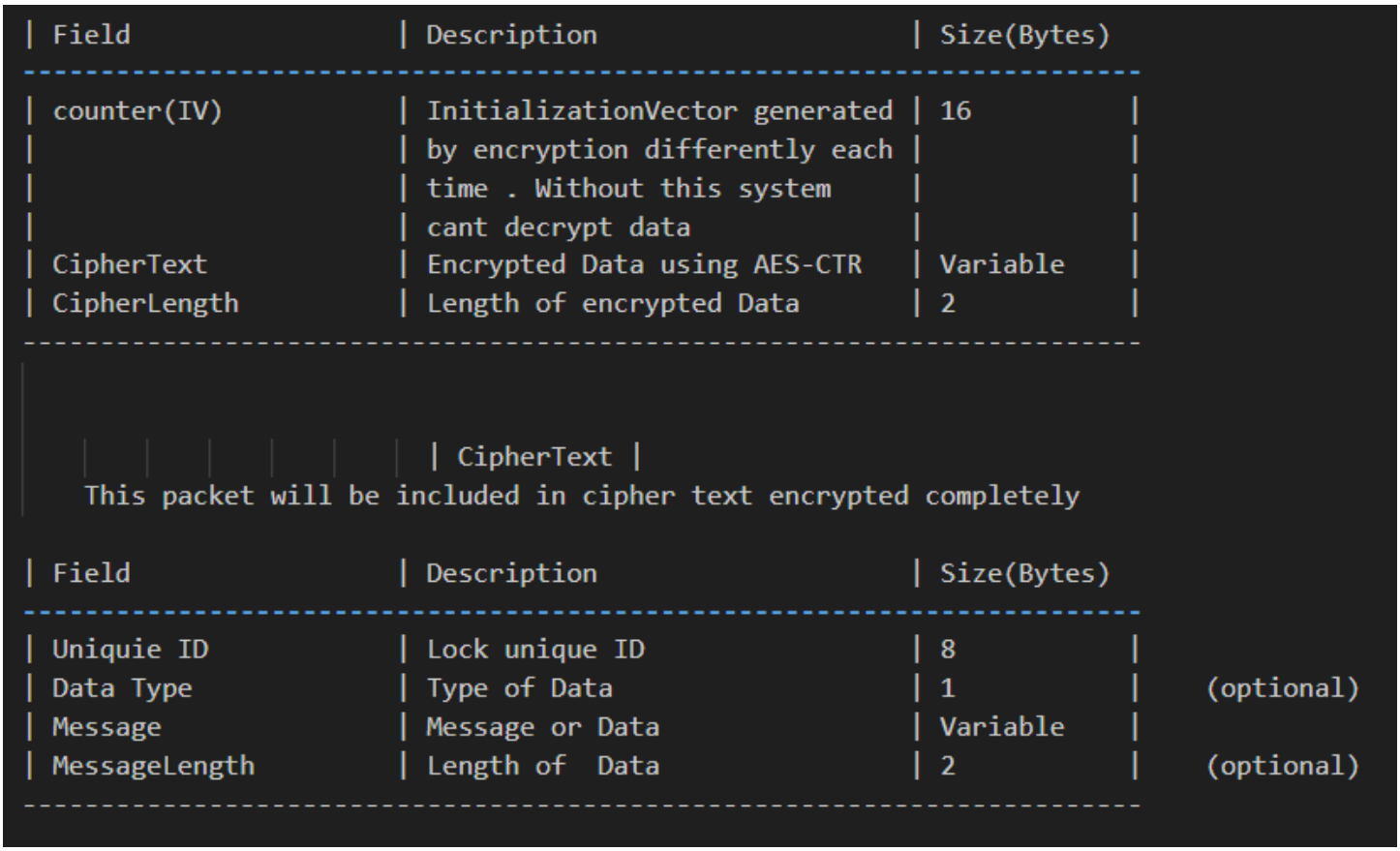
1. What is AES-CTR?
AES (Advanced Encryption Standard) in CTR (Counter) mode is a secure method of encrypting data, often used in real-time communications.

2. How it Works
AES-CTR encrypts the data by combining (or "XORing") it with a series of encrypted "blocks" that are generated using a **counter value**. The counter changes for each message, making the encryption unique every time.

3. Counter Value:
Imagine a **counter** like a simple number, starting from 0.
For **each message** sent, the counter value **increases by 1**.
This counter is combined with a **secret key** and **encrypted** to generate a unique "block" (which looks like random data).
The data (like commands for the smart lock) is then **XORed** with this block to get the encrypted message.

4. Why the Counter is important:
The counter ensures that even if two messages have the same data (like "unlock the door"), the encrypted version will look completely different. This way, an attacker cannot guess any patterns from the data, because each message uses a **unique counter** for its encryption.

Communication Packet:



Field	Description	Size(Bytes)	
counter(IV)	InitializationVector generated by encryption differently each time . Without this system cant decrypt data	16	
CipherText	Encrypted Data using AES-CTR	Variable	
CipherLength	Length of encrypted Data	2	

| CipherText |

This packet will be included in cipher text encrypted completely

Field	Description	Size(Bytes)	
Uniquie ID	Lock unique ID	8	
Data Type	Type of Data	1	(optional)
Message	Message or Data	Variable	
MessageLength	Length of Data	2	(optional)

Message Packet Explanation:

Message Structure for encrypted packet

Field	Description	Size(Bytes)	Example
<SOH>	Start of heading	1	Indicates start of encrypted message
Counter(IV)	Generated by encryption	16	Used to decrypte message
<EM>	Separator	1	
CipherText	Encrypted Message	variable	Containing otherinfo and some data
<EM>	separator	1	
CipherLength	Length of encrypted data	2	
<EOT>	End of Transmission	1	Signals that transmission is complete

Message Structure for CipherText (when message is decrypted)

Field	Description	Size(Bytes)	Example
<STX>	Start of Transmission	1	Indicates start of message
Unique ID	Unique ID of the lock	8	
<EM>	Separator	1	
Data type	Type of message being sent	1	0x01 for command, 0x02 for status and so on
<EM>	separator	1	
Message ID	Specific Message within Data type	1	0x01 for lock, 0x02 for unlock
<EM>	separator	1	
Data	The data associated with the message (if applicable)	variable	Time/date or encryption key etc
<EM>	separator	1	
Message Length	Length of data field	2	
<ETX>	End of Transmission	1	
<EOT>	End of Transmission (final indicator)	1	

Data types and message ID and Data explanation

Data type	Description	Message ID	Data	Lock Response
(CMD)0x01	Instruction to lock/unlock sent from the app to lock	0x01 : Lock 0x02 : Unlock	NULL	ACK / NACK
(Status)0x02	Request to Check the Lock state or battery status in percentage	0x01 : Lock status 0x02 : Battery level	NULL	0x01 : Locked 0x02: Unlocked Battery %
(Config)0x03	Settings Change, like time or encryption keys	0x11 : Set time/date 0x21 : Set Encryption Key 0x12 : Get Time/date 0x22 : get encryption key	00:12:23|22/12/24 (time/date) PrivateKey	For setting ACK/NACK For case of 0x12: returns time in format 00:12:23|22/12/24 0x22: return saved private key
(ACK/NACK) 0x04	Acknowledgement	ACK:0x06 NACK:0x15	NULL	N/A

Complete Message Packet Examples

Eample: Unlock Command

1. App to Lock: Unlock Command
   <SOH><Counter(IV)><EM><CipherText><EM><CipherLength><EOT>

   CipherText breakdown:
   <STX><UniqueID><EM><0x01><EM><0x02><EM><NULL><EM><0000><ETX><EOT>

- **Data Type 0x01**: Command
- **Message ID 0x02**: Unlock
- **Data**: None, as unlocking doesn't require additional data
- **Message Length**: `0000`, since no data is included.

2. Lock Response: Ack for Unlock Command
   <SOH><Counter(IV)><EM><CipherText><EM><CipherLength><EOT>

   CipherText breakdown:
   <STX><UniqueID><EM><0x04><EM><0x06><EM><NULL><EM><0000><ETX>
   <EOT>

- **Data Type 0x04**: Acknowledgment
- **Message ID 0x06**: ACK (Command succeeded)
- **Data**: None, as this is just an acknowledgment.

Another Example: **App to Lock - Check Battery Level**

1. **App to Lock**: Request to check battery level
   <SOH><Counter(IV)><EM><CipherText><EM><CipherLength><EOT>

   CipherText breakdown:
   <STX><UniqueID><EM><0x02><EM><0x02><EM><NULL><EM><0000><ETX>
   <EOT>

- **Data Type 0x02**: Status request
- **Message ID 0x02**: Battery level
- **Data**: None required for the request
- **Message Length**: `0000`

2. **Lock Response**: Battery Level
   <SOH><Counter(IV)><EM><CipherText><EM><CipherLength><EOT>

   CipherText breakdown:
   <STX><UniqueID><EM><0x02><EM><0x02><EM><75%><EM><0003><ETX><EOT>

- **Data Type 0x02**: Status request
- **Message ID 0x02**: Battery level
- **Data**: `75%` (battery level in percentage)
- **Message Length**: `0003`