Università degli Studi di Bergamo
Scuola di Ingegneria
Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione
Corso di laurea in Ingegneria Informatica

Gestione abbonamenti autobus tramite Arduino e NFC

Relatore: Prof. Paraboschi Stefano

Prova finale di:
Daniele Ravasio
Matricola 1045934

Anno Accademico 2018-2019

Abstract - Italiano

Il progetto MyNBS consiste nella realizzazione di un abbonamento per i mezzi tramite l'uso di chip NFC, l'obiettivo è quello di avere con se un abbonamento facilmente trasportabile e poco ingombrante, inoltre l'utilizzo di quest'ultimo rende anche più semplice alle autorità il controllo e il rinnovo. In questo documento saranno evidenziate le principali tecnologie utilizzate per la creazione di questo progetto ponendo l'attenzione sui protocolli e sui meccanismi di sicurezza implementati. Sarà poi presentato un esempio del funzionamento e sviluppi futuri.

Abstract - Inglese

MyNBS consists in the creation of a bus pass using NFC technology, the aim is to have an easy to carry and a less bulky pass, and even the authorites are facilitated when they have to check the bus pass or when they want to renew a pass. In this document we'll discuss about the technology used for the creation of this project, focussing on the protocols and the security mechanism implemented. After that there will be a practical example of the project and a section of future developments

Indice

1 Introduzione

1.1 Idea

L'idea di base è nata, durante un viaggio nei paesi nordici quando vidi che salendo sui mezzi di trasporto pubblici, le persone utilizzavano delle tessere magnetiche, analizzando e chiedendo scoprii poi essere tessere con all'interno dei chip NFC, andando avanti negli anni scoprii che in altri paesi, oltre all'utilizzo delle tessere in parallelo venivano usati anche gli smartphone con in chip direttamente incluso nel telefono.
Da li ho preso spunto chiedendomi "Perché non portare anche nel nostro paese una tecnologia del genere?" .l'idea di fondo è quindi quella di avere un abbonamento sempre a portata di mano, facilmente rinnovabile, meno ingombrante, e anche più sicuro, inoltre è anche un'idea **eco-friendly**, infatti si può pensare che al posto di comprare un biglietto od un carnet di biglietti, per poi buttarli via dopo l'utilizzo, si ha a disposizione una tessera magnetica nella quale viene caricato il biglietto/carnet, e dopo l'utilizzo basterà semplicemente rinnovarlo o cambiarne la tipologie, evitando così uno spreco di carta.

1.2 MyNBS

MyNBS (My NFC Bus Subscription) è un'applicazione sviluppata in Java con un'interfaccia grafica che permette la sottoscrizione di un abbonamento per i pullman o il controllo di un abbonamento già esistente. Abbiamo quindi due funzionalità che vanno a dividersi in molteplici step:
Sottoscrizione abbonamento:

1. L'operatore inserisce i dati dell'utente e le zone volute per l'abbonamento in un'interfaccia grafica

2. L'operatore posizione sull'antenna NFC il tag, nel quale verranno scritti in maniera codificata i dati dell'utente.

Controllo abbonamento:

1. Il controllore seleziona la/e zona/e dove è in questo momento

2. Posizione sopra il lettore il tag NFC, sia che l'abbonamento è valido per quella zona che non è valido verrà segnalato, nel secondo caso verranno evidenziate le zone di validità dell'abbonamento.

1.3 Obiettivi

L'obiettivo principale è quello di avere con se un abbonamento facilmente trasportabile, poco ingombrante e sicuro, infatti tramite la tecnologia NFC e l'implementazione della crittografia è possibile avere un'autenticazione sicura ed evitare anche che qualcuno di esterno riesca ad interpretare i dati del chip anche se dovesse riuscire a copiarlo. Inoltre per le forze dell'ordine è molto più semplice controllare quel chip e i dati associati piuttosto che dover guardare una carta che andando avanti nel tempo subirebbe l'usura e risulterebbe quindi di difficile comprensione.

2 Strumenti utilizzati

Per la realizzazione della tesi sono state usati i seguenti strumenti:

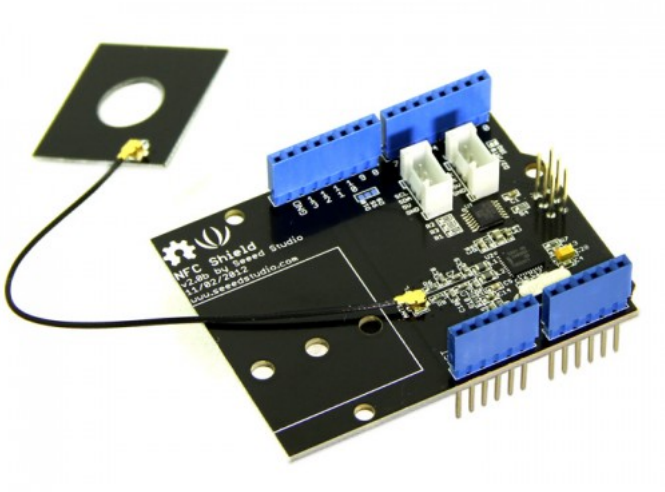
- Arduino
- NFC Shield v2.0
- NFC Tag
- Java
- NoSQL Database

2.1 Arduino Uno

Arduino è una piattaforma hardware composta da una serie di schede elettroniche dotate di un microcontrollore. È stata ideata e sviluppata nel 2003 da alcuni membri dell'Interaction Design Institute di Ivrea come strumento per la prototipazione rapida e l'utilizzo in vari ambiti, per esempio la robotica e la domotica.

2.2 NFC Shield v2.0

Le shield sono schede che possono vengono inserite sopra l'Arduino, permettono l'estensione delle capacità della scheda stessa.La shield usata un questo progetto è quella NFC composta da un'antenna che collegandosi ad Arduino abilita la capacità di leggere/scrivere sui chip NFC



2.3 NFC Tag

La tecnologia NFC è una combinazione d'identificazione senza contatto (**RFID**) e altre tecnologie di connettività.NFC permette una comunicazione bidirezionale: quando due apparecchi NFC (initiator e target) vengono accostati entro un raggio di 4 cm, viene creata una rete peer-to-peer tra i due ed entrambi possono inviare e ricevere informazioni.

La tecnologia NFC opera alla frequenza di 13,56 MHz e può raggiungere una velocità di trasmissione massima di 424 kbit/s.

Il formato dei chip NFC usato nel progetto è **NDEF**

2.3.1 Tipologie di tag

Prima di iniziare questa sezione è bene sapere che esistono solo cinque tipi di tag standardizzati e appunto per questo il loro funzionamento è garantito con gli accessori che possediamo in questo momento. I nuovi tipi di tag richiederanno una standaridzzazione e potrebbero anche richiedere un aggiornamento dell'architettura NFC che possediamo attualmente.

- **Tipo 1**: il primo tipo è quello più semplice a causa di questo viene anche considerato come il più lento, ed essendo così semplice anche a livello di prezzo risulta essere molto economico. Il problema riguarda il fatto che certe funzionalità per delle applicazioni specifiche potrebbero mancare. Solitamente questi tag vengono usati per:

  - Applicazioni in sola lettura

  - Accoppiamento di dispositivi bluetooth

  - Lettura di un tag specifico quando c''e ne sono tanti

- **Tipo 2**: Prima di iniziare questa sezione è bene sapere che esistono solo cinque tipi di tag standardizzati e appunto per questo il loro funzionamento è garantito con gli accessori che possediamo in questo momento. I nuovi tipi di tag richiederanno una standaridzzazione e potrebbero anche richiedere un aggiornamento dell'architettura NFC che possediamo attualmente.

- **Tipo 1**: il primo tipo è quello più semplice a causa di questo viene anche considerato come il più lento, ed essendo così semplice anche a livello di prezzo risulta essere molto economico. Il problema riguarda il fatto che certe funzionalità per delle applicazioni specifiche potrebbero mancare. Solitamente questi tag vengono usati per:

  - Applicazioni in sola lettura

- Accoppiamento di dispositivi bluetooth

- Lettura di un tag specifico quando c'è ne sono tanti

• **Tipo 2**: il secondo tipo è quello più popolare, perché offre ottime funzionalità per un giusto prezzo, ed inoltre riesce a raggiungere un'ampia varietà di scopi. È più veloce del tipo 1, quindi viene usato per applicazioni dove l'utente si aspetta una risposta immediata, quindi:

- Transazioni di basso valore

- Ticket per eventi

- Redirect tramite URL

• **Tipo 3**: il terzo tipo si basa su standard differenti rispetto agli altri. Vengono anche conosciuti come i tag della Sony FeliCa, sono un'innovazione giapponese e molto usata in Asia. Offre molte funzionalità però il prezzo è molto alto, vengono appunto usati molto in Giappone e per questi tipi di applicazioni:

- Biglietti

- Moneta elettronica

- Dispositivi per l'assistenza sanitaria

• **Tipo 4**: il quarto tipo di tag offre la maggior memoria e flessibilità tra tutti. Il prezzo è variabile però può aggirarsi dal medio all'alto, in funzione alla quantità di memoria che vuoi. La funzione più importante di questo tag è la *sicurezza*, infatti è equipaggiato con funzionalità che permettono la "true authentication". Inoltre questo tag è l'unico che supporta lo standard ISO 7816 relativo alla sicurezza, inoltre permette l'auto-modifica del contenuto NDEF. Solitamente questo tag viene usato per applicazioni di biglietteria.

- **Tipo 5**: il quinto tipo offre supporto per la specifica ISO 15693, viene supportata la Modalità di Comunicazione Attiva, che permette di ottenere performance in ambito di trasferimento dei dati simili alle tecnologie RF. La distanza di lettura è uguale a quella degli altri tag NFC, solitamente questi tag vengono usati per:

  - Impacchettamento di prodotti
  - Biglietti
  - Dispositivi per l'assistenza sanitaria

2.3.2 Standard ISO

Ci sono diversi standard ISO usati per la tecnologia NFC tra cui: ISO 15693, 18092 e 21481 poi ECMA 340, 352 e 356 ed ETSI TS 102 190. NFC è inoltre compatibile con la diffusa architettura delle smart card contactless, basate su ISO 14443 MIFARE e Sony FeliCa (tipo 4).

- ISO 15693: Lo standard ISO 15693 utilizza la frequenza 13.56 MHz, viene offerta una distanza di lettura che può variare tra 1 metro ed 1.5 metri, poiché le carte devono operare a distanza, è richiesta la presenza di campi magnetici inferiori rispetto a quelli usati per altre carte. Inoltre a differenza di altri standard c'è una funzione di anticollisione implementata, questo serve per consentire una lettura simultanea di più card senza incorrere in errori o fail di ricezione.

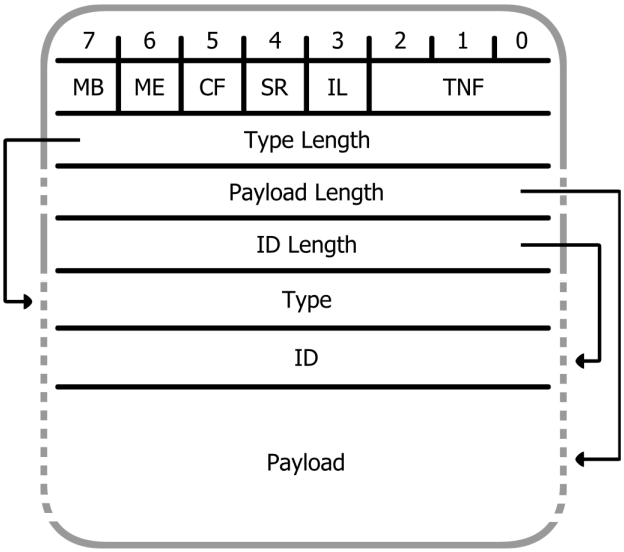
- ISO 18092, definisce lo standard di una smartcard contactless, viene prevalentemente usato nei sistemi RFID (pre-NFC). Questa tecnologia viene usata con i tag di tipo 4 quindi in Giappone. Questa specifica è formata dalle ISO 15693 (vista prima) e dalla ISO 9798 che regolamenta il protocollo di comunicazione a mutuo riconoscimento, questo implica che il tag può essere letto solo da un lettore già programmato.

3 Tecnologia Implementata

In questa sezione verranno illustrate le tecnologie implementate per la realizzazione del progetto

3.1 NDEF e Sicurezza

NDEF è un preciso standard per un formato di dati sui chip NFC, viene utilizzato in applicazioni quali le carde di credito, o gli smart poster, la struttura di un messaggio è quella vista nella seguente figura:



In ordine abbiamo:

Header

- Message Begin (MB)
- Message End (ME)
- Chunk Flag (CF)
- Short record (SR)
- ID Length present (IL)

- Type Name Format (TNF)

- Length fields

- Type

- ID

I primi 5 parametri sono dei Flag, per finire invece abbiamo il **Payload** che è il messaggio.
Detto questo andiamo a concentrarci meglio su determinati campi quindi:

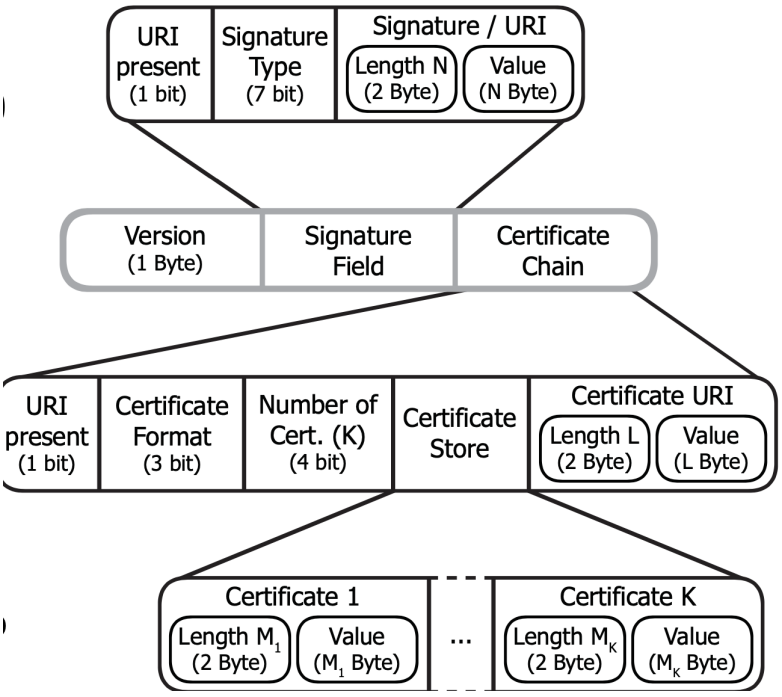
- CF: indica il record che fa parte di una catena di record, quando il suo valore è pari a 1 vuol dire che c'è *almeno* 1 altro record nella catena, invece quando non è settato vuol dire che ci troviamo o nel caso di record singolo o nel caso di ultimo elemento della catena. Una cosa da notare è che ME e CF non possono essere entrambi settati a 1

- SR: quando viene messo ad 1 vuol dire che il record corrente è di tipologia short, questo comporta che il campo Payload Length, che si trova dentro Length fields, viene espresso tramite 1 byte, in alternativa viene espresso da 4 byte

- IL: indica se nel record è presente un identificatore, se non dovesse essere abilitato questo comporta che non vi saranno il campo ID Length e il campo ID che è quello di identificazione del record.

- TNF: serve per indicare la struttura del campo Type, è composto da 3 bit e può assumere solo i valori da 0 a 6 perché il numero 7 è riservato.

NDEF ha anche una firma



L'ultimo documento di specifica è stato rilasciato nel Novembre 2010, sostanzialmente ha una struttura fatta da un **campo firma**, che può essere una firma o una referenza URI ad una firma e da una **catena di certificati**, che è una catena di certificati PKI su un percorso sicuro.
Il record con la firma viene aggiunto ad una sequenza di record, e questo firma ogni record tra il record di firma appena precedente e se stesso, infatti un messaggio NDEF può contenere più di una firma.

3.1.1 Cosa viene firmato?

I campi che non vengono firmati sono MB/ME, questo perché se venissero firmati la firma non potrebbe essere agganciata al messaggio NDEF già firmato. Type, ID e Payload invece vanno firmati per assicurare l'integrità dei dati, mentre quando TNF viene cambiato, l'intero significato del record cambia, può essere quindi usato per nascondere dei record (identificati da type "Unknown").

3.1.2 Sicurezza

La tecnologia NFC è un'evoluzione dell'RFID, da un lato risulta meno predisposta ad attacchi esterni ma dall'altro lato è soggetta alle problematiche di sicurezza del

suo predecessore. Le possibili minacce di sicurezza a cui sono sottoposti sono quelle riguardanti l'acquisizione, o l'alterazione dei dati contenuti nel tag, queste minacce possono avvenire mediante interrogazioni fraudolente o mediante intercettazione delle informazioni grazie a ricevitori radio durante la lettura da parte di un lettore autorizzato.

3.1.3 Intercettazioni

Nell'ambito dell'NFC ma soprattutto in maniera generale, nel campo delle comunicazioni wireless l'intercettazione dei dati è uno degli attacchi più comuni. Per effettuare quest'attacco serve attrezzatura progettata ad hoc, quindi antenne e lettori fatti su misura. Per quanto riguarda il caso specifico NFC è un attacco molto difficile da realizzare a causa sei seguenti fattori:

i potenza emessa dallo strumento sotto intercettazione

ii fattori ambientali

iii presenza della crittografia

Quindi da questo capiamo che anche in base al tipo di tag [1] un attacco può essere più facile o più difficile, per esempio un attacco su un tag di tipo 1 sarà molto più semplice che su un tag di tipo 4.
Ci sono anche contromisure come per esempio quella di diminuire il campo magnetico, magari aumentando il fattore di direzionalità delle antenne, oppure usare algoritmi di cifratura per il messaggio, algoritmi per esempio AES.

3.1.4 Modifica dei dati

La modifica dei dati è un problema molto pericoloso, questo perché risulta trasparente all'utente, ha come scopo quello di modificare i dati trasmessi e renderli "validi". Fortunatamente risulta un attacco molto difficile da eseguire perché bisognerebbe riuscire ad intercettare completamente ogni bit del messaggio e rimandarlo sulle

[1] cap 2 par 2.3.1

frequenze precise, quindi ogni volta modulando il campo delle frequenze in modo specifico e diverso.

3.1.5 Man in the middle

È uno degli attacchi più pericolosi, infatti può arrecare molti danni ai sistemi coinvolti. Mentre sysA e sysB stanno comunicando, in mezzo a loro arriva l'hacker usando un apparato esterno sysH. Durante la comunicazione sysH altera il dialogo che hanno sysA e sysB, mettendosi in mezzo e fingendosi sysA per sysB e sysB per sysA. La soluzione a questo attacco è quella di instaurare un canale sicuro, quindi usando una chiave per criptare i dati. Potrebbe succede che sysH cerchi di negoziare una chiave ai due sistemi però è molto complesso perché richiederebbe la visibilità di sysH.

3.2 NoSQL Database

NoSQL è una tecnologia che promuove sistemi software dove la persistenza dei dati è in generale caratterizzata dal fatto di non utilizzare il modello relazionale. L'espressione "NoSQL" fa riferimento al linguaggio SQL, che è il più comune linguaggio di interrogazione dei dati nei database relazionali.
Questi archivi di dati il più delle volte non richiedono uno schema fisso (schemaless), evitano spesso le operazioni di giunzione (join) e puntano a scalare in modo orizzontale. Gli accademici e gli articoli si riferiscono a queste basi di dati come memorizzazione strutturata (structured storage). Per il progetto è stata utilizzata questa tecnologia per tenere in memoria fisica (tramite un file con estensione .txt) gli abbonamenti dei vari utenti

3.3 WindowBuilder

Per la realizzazione dell'interfaccia grafica in Java è stato usato il plug-in WIndowBuilder di Eclipse. Questo plug-in è composto a partire dalle liberire SWT Designer e Swing Designer e rende comoda e veloce la creazione di interfacce grafiche (GUI) per le applicazioni Java. Usando il WYSIWYG visual designer e gli strumenti di

layout è possibile creare finestre complesse, e per ogni elemento messo verrà generato il codice contente la posizione dell'elemento e la sua dichiarazione.
Inoltre, il codice generato da questa libreria non richiede l'uso di altre librerie personalizzate per compilarlo ed eseguirlo, inoltre è possibile dall'interfaccia grafica creare eventi che poi andranno compilati mediante codice scritto in dei blocchi di *ActionListener*, è possibile generare diversi tipi di eventi, dal click del mouse, alla pressione di un tasto sulla tastiera, o persino al movimento nella finestra del mouse.

3.4 Cassandra

Apache Cassandra è un DBMS distribuito e open source. Si tratta di un progetto Top-Level, sviluppato da Apache Software Foundation per gestire grandi quantità di dati dislocati in diversi server, fornendo un servizio orientato alla disponibilità.
È una soluzione NoSQL che inizialmente fu sviluppata da Facebook, un modello di dati simile a BigTable in esecuzione su un'infrastruttura tipi Amazon-Dynamo. Cassandra fornisce una struttura di memorizzazione chiave-valore, con Eventual Consistency.

Alle chiavi corrispondono dei valori, raggruppati in famiglie di colonne: una famiglia di colonne è definita quando il database viene creato. Tuttavia le colonne possono essere aggiunte a una famiglia in qualsiasi momento.
Le colonne sono aggiunte solo specificando le chiavi, così differenti chiavi possono avere differenti numeri di colonne in una data famiglia. I valori di una famiglia di colonne sono memorizzati insieme, questo perché Cassandra adotta un approccio ibrido tra DBMS orientato alle colonne e la memorizzazione orientata alle righe.
Come caratteristiche principali abbiamo:

- Decentralizzazione: i nodi nel cluster sono identici, non c'è alcun single point of failure

- Fault-tolerance: i dati vengono replicati in maniera automatica su più nodi, la replica è supportata tramite vari data center e la sostituzione dei nodi può avvenire senza downtime.

- Tunable consistency: il livello di coerenza può essere modificato (da writes never fail a block for all replicas to be readable).

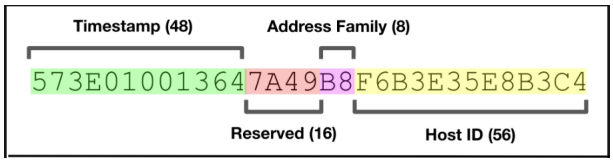
- Elasticità: il throughput di lettura o scrittura scala linearmente con l'aggiunta di nuove macchine, senza downtime e senza interruzione di alcun applicativo.

3.5 UUID

Un UUID[2] conosciuto anche come Universally Unique Identifier, è un numero di 128 bit utilizzato per identificare informazioni nei sistemi informatici, lo si può trovare nell'rfc 4122.

3.5.1 Come è formato?

Il codice è composto da 16 byte, solitamente viene identificato da 32 caratteri esadecimali, a livello di sicurezza assume $3 \cdot 10^{38}$ possibili combinazioni



A differenza di altri sistemi l'unicità del codice generato non dipende da un'autorità centrale di registrazione o dal coordinamento delle parti che lo generano, però la sua probabilità di essere duplicato è talmente vicina a zero che viene considerata trascurabile.

[2]https://tools.ietf.org/html/rfc4122

3.5.2 Analisi a livello matematico

Considerando i 128 bit dell'UUID versione 4, 6 bit vengono riservati, quattro per la versione e due per altri parametri, quindi un UUID generato randomicamente ha 122 bit casuali. La probabilità che due UUID hanno lo stesso valore può essere calcolata usando la teoria delle probabilità (Paradosso del compleanno [3]). Usando quindi quest'approssimazione possiamo calcolare:

$$p(n) \approx 1 - e^{-\frac{n^2}{2 \cdot 2^x}} \tag{1}$$

Notiamo quindi che quando il termine $\frac{n^2}{2 \cdot 2^x}$ è vicino allo zero, la probabilità può essere direttamente approssimata in questo modo:

$$p(n) \approx \frac{n^2}{2 \cdot 2^x} \tag{2}$$

Analizzandolo quindi in modo numerico potremo pensare che anche generando 1 miliardo di UUID ogni secondo per i prossimi 100 anni la probabilità di creare almeno un duplicato sarebbe circa del 50%

3.6 Crittografia

[3] https://betterexplained.com/articles/understanding-the-birthday-paradox/

4 Implementazione

In questa sezione verranno sottolineate i frammenti più importanti del codice.

4.1 Apertura della Connessione con Java

```
private static final String PORT_NAMES[] = {
                        "/dev/cu.usbmodem143301", // Mac OS X
                        "COM3", // Windows
        };
private static final int DATA_RATE = 9600;

serialPort = (SerialPort) portId.open(this.getClass().getName(),
                                      TIME_OUT);
```

Nel frammento di codice appena visto c'è la dichiarazione delle porte che si vanno ad utilizzare, il **DATA_RATE**, ovvero la quantità di dati digitali che possono essere trasferiti su un canale in un determinato intervallo temporale e l'apertura della connessione tramite la funzione **portId.open**

4.2 Rilevazione presenza tag NFC Arduino

```
NfcTag tag = nfc.read();
if(tag.hasNdefMessage())
{[...]
    NdefRecord record = message.getRecord(i);
    int payloadLength = record.getPayloadLength();
    byte payload[payloadLength];
    record.getPayload(payload);
    Serial.write(payload,payloadLength);
  [...]}
```

In questo frammento di codice viene evidenziato come il chip NFC se presente viene scannerizzato e l'unica cosa che verrà presa sarà il payload e non l'intestazione!

4.3 Scrittura su tag NFC

4.4 Connessione al DB NoSQL

```
public void connect(final String node, final int port) {
                CodecRegistry codecRegistry = new CodecRegistry();
                codecRegistry.register(new DateCodec(TypeCodec.date(),
                 Date.class));
                cluster = Cluster.builder().addContactPoint(node).
                withPort(port).withCodecRegistry(codecRegistry).build();

                final Metadata metadata = cluster.getMetadata();
        [...]
                session = cluster.connect();
        }
```

In questo frammento di codice viene mostrato il metodo con il quale viene fatta la connessione al database NoSQL Cassandra!

4.5 Avvio di Cassandra

```
\sim % source \sim/.bash_profile
~ % cassandra -f
```

Automaticamente il controllore mettendo i suoi dati (validi) apre la connessione al database e va direttamente alla pagina di controllo degli abbonamenti

4.6 Scrittura nel DB dei dati

4.7 Algoritmo di Cifratura

5 Sviluppi futuri

In questa sezione verranno illustrati alcuni sviluppi futuri possibili per il progetto

5.1 Applicazione per smartphone

5.2 Apple pay, Google pay

5.3 Uso di altre tecnologie