Escuela Técnica Superior de Ingenieros
Industriales y de Telecomunicación

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

EVALUACION DE LA LIBRERÍA TAPLINX PARA EL DESARROLLO DE APLICACIONES MOVILES NFC
(EVALUATION OF TAPLINX FRAMEWORK FOR MOBILE NFC APPLICATIONS DEVELOPMENT)

Para acceder al Título de

Graduado en

Ingeniería de Tecnologías de Telecomunicación

Autor: Joseba Gabella Castillo

Octubre
- 2020

Resumen

Acercar una tarjeta a un lector para identificarnos o realizar el pago es una tienda es algo que se ha convertido en una acción habitual en nuestras vidas. Cada vez más, se está extendiendo el uso de terminales móviles como tarjeta o como elemento para acceder a información de almacenada en ellas, véase en paradas de autobuses, en anuncios publicitarios, etc.

La plétora de modelos de tarjetas, con sus particularidades en términos de seguridad, de mecanismos de acceso, etc. hace inviable para un desarrollador de aplicaciones móviles profundizar en todos los estándares y plantear todas variaciones que puedan surgir.

Si bien los entornos de desarrollo habilitados en sistemas operativos móviles cada vez abstraen más la operativa de lectura y escritura de tarjetas inteligentes, todavía queda mucho trabajo por hacer para hacerlo lo más sencillo y transparente posible. En este sentido, los fabricantes de tarjetas, como NXP, uno de los líderes en el sector con su familia de tarjetas sin contacto MIFARE, proporcionan un API que habilita un acceso abierto a las funcionalidades de su tarjeta, empleando interfaces sencillos y ocultando toda la dificultad subyacente.

El presente trabajo busca evaluar el framework TapLinx para el desarrollo de aplicaciones móviles NFC sobre el sistema operativo Android para la operativa con tarjetas de la familia MIFARE.

Abstract

Bringing a card to a reader to identify ourselves or make payment is a store is something that has become a common action in our lives. Increasingly, the use of mobile terminals as a card or as an element to access information stored on them, see at bus stops, in advertisements, etc.

The variety of card models, with their particularities in terms of security, access mechanisms, etc. makes it unfeasible for a mobile application developer to delve into all the standards and raise all variations that may arise.

While development environments enabled on mobile operating systems are increasingly abstracting smart card reading and writing operations, there is still much work to be done to make it as simple and transparent as possible. In this sense, card manufacturers, such as NXP, one of the leaders in the sector with its family of MIFARE contactless cards, provides an API that enables open access to the functionalities of its card, using simple interfaces and hiding all the difficulties. underlying.

The present work seeks to evaluate the TapLinx framework for the development of NFC mobile applications on the Android operating system for the operation with MIFARE family cards.

Índice

Resumen 2
Abstract 3
Índice 4
Lista de figuras 6
Lista de tablas 7
Acrónimos 8
1 Introducción 9
1.1 Motivación y objetivos 10
1.2 Estructura del documento 10
2 Estado del arte de las tarjetas inteligentes 12
2.1 Tipos de tarjetas chip 12
2.1.1 Tarjetas de memoria 13
2.1.2 Tarjetas inteligentes 13
2.2 Tarjetas con contactos 13
2.2.1 Arquitectura 13
2.2.2 Características físicas 14
2.2.3 Protocolos de comunicación 15
2.2.4 Sistema operativo 17
2.3 Tarjetas de proximidad 17
2.3.1 RFID 18
2.3.2 ISO 14443 20
2.3.3 NFC 20
2.3.4 Vulnerabilidades 22
3 Familia de tarjetas MIFARE 24
3.1 Funcionamiento 24
3.2 Identificación del tipo de chip 25
3.3 MIFARE Classic 27
3.4 MIFARE Plus 28

3.5 MIFARE Ultralight 29
3.6 MIFARE Ultralight C 30
3.7 MIFARE DESFire 30
3.7.1 Mecanismos de seguridad 31
4 librería TapLinx 34
4.1 Obtención de TapLinx 34
4.1.1 Primeras pruebas 35
4.2 Lista de comandos usados 38
4.2.1 Generales 38
4.2.2 Mifare Classic 38
4.2.3 Mifare DESFire 38
4.2.4 Mifare Ultralight 39
4.3 Ventajas de la librería TapLinx 39
5 aplicación de control de acceso a un concierto 41
5.1 Requisitos 41
5.2 Formateo de las tarjetas 42
5.2.1 MIFARE Ultralight 42
5.2.2 MIFARE Classic 43
5.2.3 MIFARE DESFire 43
5.3 Menú principal 44
5.4 Entrada estándar 45
5.5 Entrada VIP 46
5.6 Uso de la tarjeta MIFARE Ultralight 47
5.7 Error en la validación 48
6 Conclusiones y líneas futuras 49
Bibliografía 50

Lista de figuras

Figura 2.1 Clasificación de tarjetas chip. 13
Figura 2.2 Arquitectura del circuito integrado de una tarjeta inteligente 14
Figura 2.3 Contactos del chip de una tarjeta con contactos 15
Figura 2.4 Modelo de comunicación 16
Figura 2.5 Estructura y formato APDU 16
Figura 2.6 Alimentación de una etiqueta RFID o transpondedor (derecha) mediante la corriente inducida por el campo magnético generado por el lector (izquierda). 18
Figura 2.7 Pila de protocolos NFC 21
Figura 2.8 Ataque por escucha o eavesdropping 22
Figura 3.1 Comienzo de activación de la tarjeta. 25
Figura 3.2 Ejemplo de todas las tarjetas MIFARE y como identificarlas. 26
Figura 3.3 Estructura de una MIFARE Classic. 27
Figura 3.4 Organización de memoria de la MIFARE Plus 29
Figura 3.5 Organización de memoria de MIFARE Ultralight 29
Figura 3.6 UID/Serial number de una MIFARE Ultralight C. 30
Figura 3.7 Estructuración de los datos en una DESFire EV1. 31
Figura 3.8 Autenticación mediante algoritmo AES 32
Figura 4.1 Código para añadir TapLinx vía Maven Repository 35
Figura 4.2 Código usado para el registro de la aplicación 35
Figura 4.3 Código usado para habilitar los eventos 36
Figura 4.4 Código empleado para habilitar los permisos 36
Figura 5.1 Esquema de funcionamiento del programa. 42
Figura 5.2 Menú principal de la aplicación 44
Figura 5.3 Validación entrada estándar. 45
Figura 5.4 Validación entrada VIP. 46
Figura 5.5 Error en la validación de la entrada. 48

Lista de tablas

Tabla 5.1 Formato de los datos en MIFARE Classic 43
Tabla 5.2 Formato de los datos en MIFARE DESFire..... 43

Acrónimos

ISO	*International Organization for Standardization*
IEC	*International Electrotechnical Comission*
CPU	*Central Processing Unit*
ROM	*Read Only Memory*
EEPROM	*Electrical Erasable Programmable Read-Only Memory*
RAM	*Random Access Memory*
GSM	*Global System for Mobile communications*
SIM	*Subscriber Identity Module*
ATR	*Answer To Reset*
APDU	*Application Protocol Data Unit*
MF	*Master File*
DF	*Dedicated File*
EF	*Elementary file*
ASK	*Amplitude-Shift Keying*
FSK	*frequency-Shift Keying*
PSK	*Phase-Shift Keying*
RFID	*Radio Frequency Identification*
NFC	*Near Field Communication*
PCD	*Proximity Coupling Device*
PICC	*Proximity Integrated Circuit Card*
REQA	*Request Command, Type A*
ATQA	*Answer To Request according to ISO 14443-4*
SAK	*Select Acknowledge, type A*
UID	*Unique ID*
BCC	*Check Character Bytes*
DES	*Data Encryption Standard*
AES	*Advanced Encryption Standard*
RATS	*Request Answer To Select*
OTP	*One Time Programmable*
EULA	*End-User License Agreement*
VIP	*Very Important Person*

1 Introducción

Las necesidades de seguridad crecen a medida que evoluciona la tecnología. Para cubrir estas necesidades en el entorno del usuario y facilitar la operativa de los múltiples servicios de almacén seguro de información e identificación, se idearon las tarjetas inteligentes con o sin contactos. Estas tarjetas presentan una solución muy cómoda para el usuario, en especial las tarjetas sin contactos, ya que con el simple el hecho de aproximar la tarjeta al lector para operar las hace de ellas un elemento muy atractivo.

A día de hoy, no es de extrañar que la mayoría de las personas posean una tarjeta de transporte público o una tarjeta de crédito, con información sensible relacionada con el usuario como datos personales o referencias financieras. Es por eso que estas tarjetas deben poseer unos estándares de seguridad muy elevados.

Aunque las primeras aproximaciones de lo que hoy en día es una tarjeta inteligente se plantearon durante los años 70 [19], las tarjetas inteligentes sin contactos se utilizaron por primera vez para la emisión de billetes electrónicos en 1995 en Seúl, Corea del Sur [18]. Desde entonces, las tarjetas sin contactos se han vuelto uno de los medios más populares para aplicaciones de pago y emisión de billetes. Los diversos estándares emergentes de aquella época tenían un enfoque particular, por lo que no eran compatibles unos con otros. Algunos ejemplos incluyen las tarjetas MIFARE Classic de Philips (hoy en día NXP) que tuvieron y mantienen una gran cuota de mercado en Estados Unidos y Europa.

El éxito de la tarjeta MIFARE Classic fue tal, que los fabricantes empezaron a lanzar al mercado chips compatibles con una operativa y gestión de memoria similar a MIFARE Classic. Estos chips, como ya se ha nombrado, poseen una especificación propietaria, lo que complica el desarrollo de aplicaciones si se desconocen los comandos propios de estas tarjetas, al no seguir parcialmente los estándares vinculados a las tarjetas inteligentes sin contacto, como por ejemplo ISO/IEC 14443 [1].

A partir de esa época, las necesidades de seguridad siguieron en aumento, lo que llevo al desarrollo de dos familias de productos de alta seguridad: MIFARE Plus y MIFARE DESFire. Estas tarjetas, son las primeras de la marca MIFARE en adaptarse por completo al estándar ISO/IEC 14443.

Para facilitar la interacción con una tarjeta de la familia MIFARE de NXP y eliminar la necesidad de conocer las especificaciones en profundidad y tener que desarrollarlas para cada proyecto específico, se creó la librería MIFARE SDK [20], que permitía una comunicación transparente con la tarjeta, sin necesidad de conocer sus comandos particulares a nivel byte. Con el paso del tiempo, esta librería pasaría a conocerse como TapLinx, actualizándose según salían nuevas tarjetas al mercado.

Esta librería permite a los usuarios crear aplicaciones relacionadas con las tarjetas MIFARE, sin necesidad de unos conocimientos avanzados en materia de tarjetas inteligentes ni de ciberseguridad, haciéndola muy atractiva para el público en general.

1.1 Motivación y objetivos

Las tarjetas inteligentes surgen como un elemento capaz de almacenar de forma segura de almacenar nuestra información sensible y operar con ella para identificarnos, etc. y se han convertido en un elemento de uso cotidiano en nuestras vidas.

Debido a esto, conocer estas tecnologías es de suma importancia, ya que hay multitud de ciberataques posibles para sustraernos dicha información. La tecnología de las tarjetas inteligentes se encuentra hoy día muy diversificada, teniendo cada opción sus ventajas y sus inconvenientes, por lo que es inviable estudiarlas todas.

Para este proyecto nos hemos decantado por la tecnología denominada MIFARE de NXP, debido a que fue una de las marcas pioneras en este tipo de tecnologías, y a día de hoy es una de las marcas más importantes suponiendo alrededor del 80% de las tarjetas inteligentes contactless en uso [21]. Para operar con ellas se dispone de las librerías TapLinx, válidas tanto para entonos móviles Android como para desarrollo de soluciones de escritorio. Esta librería proporciona un gran nivel de abstracción a la hora de programar las tarjetas, permitiendo un uso sencillo y fácil de entender.

Una vez decidida la tecnología a emplear, se han de estudiar todas las opciones dentro de la familia MIFARE, según los requisitos de seguridad, siendo la MIFARE Classic la más sencilla, y la MIFARE DESFire la más compleja y segura. Para ello se realizará un análisis de cada una de las tarjetas, su nivel de seguridad y sus capacidades de memoria.

Por último, se realizará una aplicación móvil de ejemplo para entornos móviles Android que emulará la gestión y validación de entradas de concierto, diferenciando entre unas entradas estándar usando la tecnología MIFARE Classic, y unas entradas VIP usando la tecnología MIFARE DESFire. También se analizarán y comparará el uso de estas librerías frente a las librerías nativas de Android.

1.2 Estructura del documento

Este documento está organizado en 6 capítulos, tal como se muestra a continuación:

- En el primer capítulo, se sitúa y expone el marco del trabajo, describiendo las motivaciones que dan lugar al mismo y los objetivos perseguidos.
- En el segundo capítulo se estudiará la estructura y funcionamiento de una tarjeta inteligente, que será clave en la realización del proyecto.
- El tercer capítulo está centrado en las tarjetas inteligentes de la familia MIFARE, explicando el funcionamiento de cada una de ellas.
- En el cuarto capítulo se centrará en el uso de la librería TapLinx, que nos permitirá la comunicación con los diferentes modelos de tarjetas de la familia MIFARE.
- En el quinto capítulo, usando la teoría establecida se creará una aplicación sencilla para evaluar el uso de estas tarjetas.

- En el sexto y último capítulo de la memoria, se expondrán las conclusiones a las que se han llegado, y se explicara las posibles ampliaciones del proyecto para su mejora.

2 Estado del arte de las tarjetas inteligentes

Antes de explicar en detalle el funcionamiento de las tarjetas de la familia MIFARE, se ha creído conveniente hablar de las características y funcionalidades de las tarjetas inteligentes y su evolución.

Desde que a mediados del siglo XX surgieran las primeras tarjetas plásticas hasta nuestros días, muchas y muy variadas han sido las mejoras introducidas en el sector. Aunque de apariencia muy similar a una tarjeta de crédito, la tarjeta inteligente se presenta como un dispositivo muy diferente. Dotada de un circuito integrado en su cuerpo de plástico, las tarjetas inteligentes disponen de la capacidad para proteger procesar datos almacenados en ellas. Esta inteligencia que les proporciona el chip, las convierte en un ordenador portable y las diferencia de sus predecesoras, las tarjetas de banda magnética.

El origen de la tarjeta inteligente data de principios de los años 70, y se trataba en realidad de un dispositivo de memoria. Cuatro años más tarde, el francés Ronald Moreno ideo lo que podemos considerar la primera tarjeta inteligente, ya que estaba dotada de ciertos mecanismos de seguridad, como el uso de un número de identificación personal o PIN, tecnología aun usada en la actualidad. Este número cubría las necesidades de seguridad de aquella época de almacenamiento y seguridad, aunque no sería hasta la década de los 80 cuando se inicializaría la comercialización de estas tarjetas inteligentes, también llamadas smartcards, empleadas en como tarjetas bancarias o de telefonía.

A medida que se realizaban avances tecnológicos en circuitos integrados y criptografía, se ha permitido la evolución de las tarjetas inteligentes como dispositivos más seguros, potentes y baratos. Gracias a esta evolución y a sus características, su función es perfecta para almacenar, proteger y procesar información confidencial, por lo que son especialmente útiles para generación y custodia de claves secretas y ejecución de algoritmos criptográficos involucrados.

2.1 Tipos de tarjetas chip

Las numerosas ventajas de las tarjetas chip frente a las de banda magnética son:

- Mayor capacidad de almacenamiento
- Protección de accesos no autorizados gracias a la seguridad
- Memoria resistente al deterioro de la información almacenada

Es común denominar a todas las tarjetas que poseen contactos dorados o plateados tarjetas inteligentes. Sin embargo, es conveniente hacer una clasificación rigurosa agrupándolas

según las características de su componente más importante, su chip. La Figura 2.1 muestra dicha clasificación.

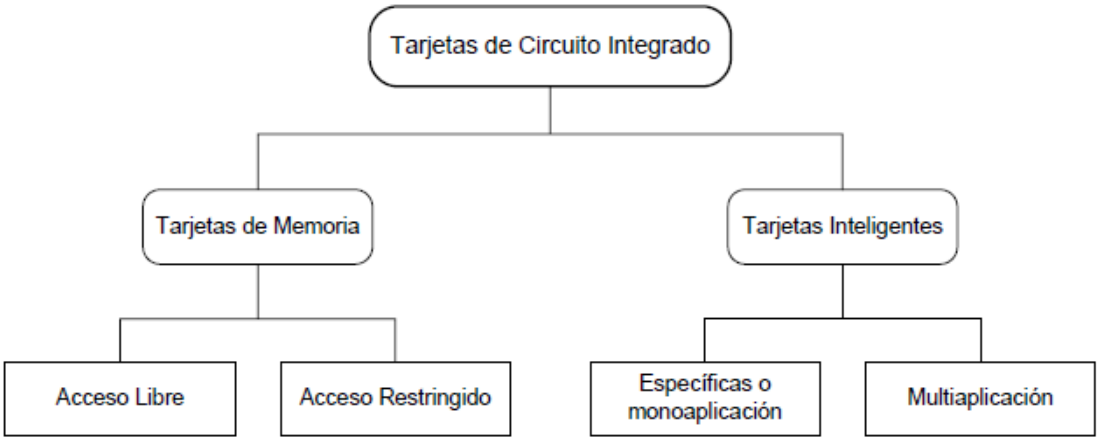


Figura 2.1 Clasificación de tarjetas chip.

2.1.1 Tarjetas de memoria

Se trata de tarjetas cuyo chip dispone de capacidad únicamente de almacenamiento de datos. En función de la información, puede estar protegida por circuitos de seguridad a partir de una clave.

Como ejemplo de éstas se consideran las tarjetas telefónicas, que únicamente almacenan el saldo restante dejando el control de los datos a aplicaciones externas.

2.1.2 Tarjetas inteligentes

Estas tarjetas poseen un microprocesador en su circuito integrado, elemento que las hace inteligentes, por lo que pueden ejecutar comandos y trabajar con los datos que contiene, y los suministrados por aplicaciones externas.

Las tarjetas inteligentes pueden transferir datos a partir de los contactos de su superficie o mediante campos magnéticos, en el caso de las tarjetas sin contactos.

El deterioro de los contactos es la principal fuente de fallo, limitación superada por las tarjetas sin contactos. Sin embargo, estas últimas presentan otros inconvenientes, como la potencia que necesitan para su activación. Además, al combinar tecnología analógica y digital, las tarjetas sin contactos poseen una integración más complicada, encareciendo los costes muy por encima de las con contactos. Las tarjetas inteligentes con contactos son actualmente las más extendidas a día de hoy.

2.2 Tarjetas con contactos

2.2.1 Arquitectura

La mayoría de las tarjetas inteligentes se componen de una arquitectura de maquina Von Neumann. El microcontrolador de una tarjeta inteligente está compuesto por un microprocesador y tres tipos de memoria: ROM, EEPROM y RAM (Figura 2.2)

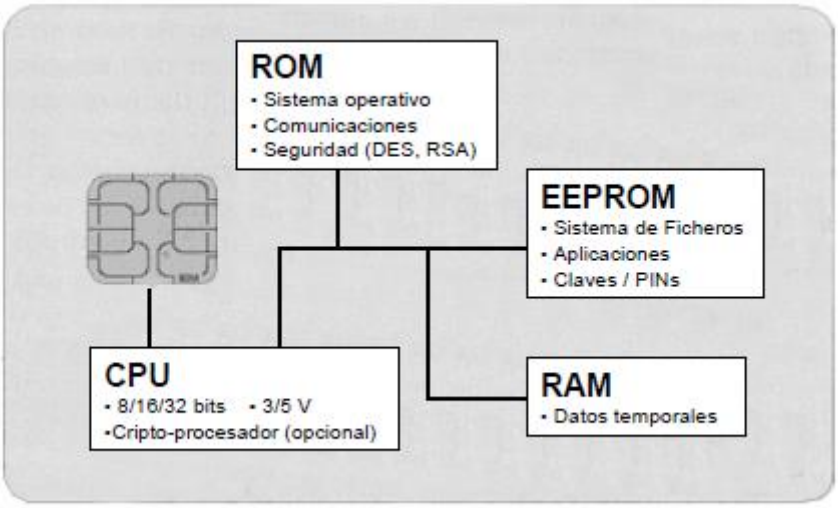


Figura 2.2 Arquitectura del circuito integrado de una tarjeta inteligente

- La Unidad Central de Proceso o CPU es la encargada de realizar los cálculos y procesar la información. Los más extendidos son de 8 bits, aunque existen de 16 y 32 bits. Algunas tarjetas poseen un coprocesador criptográfico, preparado para la realización de operación aritméticas, incrementando el rendimiento de las operaciones de seguridad y criptográficas.
- La memoria ROM se utiliza para almacenar los datos permanentes en la tarjeta, que se escriben en la fase de fabricación de esta: SO, aplicaciones y datos de usuario fijos. No es posible escribir en ella después de la fase de fabricación, y no es necesaria alimentación para este tipo de memoria
- La memoria EEPROM es una memoria no volátil en la que se encuentran los datos de usuario y aplicación. La memoria EEPROM es similar a la memoria ROM, con la diferencia de que se puede escribir en ella con algo de tiempo. Es una memoria lenta pero persistente, y funcionalmente equivale al disco duro de un ordenador.
- La memoria RAM es la memoria de trabajo del microprocesador. Es una memoria volátil, por lo que los datos se pierden al desconectar la alimentación. Se usa para mantener los datos temporales durante la sesión. La velocidad de lectura frente a la EEPROM es la misma, pero es mil veces más rápida en lectura.

De las tres memorias, la memoria ROM es la más barata, puesto que ocupa menos espacio por celda. Una celda EEPROM ocupa hasta cuatro veces más que una ROM, y una RAM cuatro veces aproximadamente que una EEPROM

2.2.2 Características físicas

Por razones de compatibilidad, el tamaño y forma están estandarizados. [3] Los formatos más utilizados son el ID-1 e ID-000.

- El formato ID-1 es el más usado, por su semejanza con las tarjetas de banda magnética.

- El tipo ID-000 surge como una alternativa de tamaño más pequeño, requerido en tecnologías en miniatura, como la telefonía móvil. Actualmente, solo es empleado en tecnologías GSM (Global System for Mobile communications), con la tarjeta denominada SIM (Subscriber Identity Module).

Los contactos se encuentran en forma de cuadrado dorado o plateado, ubicado en la superficie de la tarjeta. Esta área es usada de enlace entre el dispositivo lector y la tarjeta, y también es la vía por la que el microprocesador toma la alimentación para sus circuitos o lleva a cabo la transmisión de los datos, los contactos están representados en la Figura 2.3.

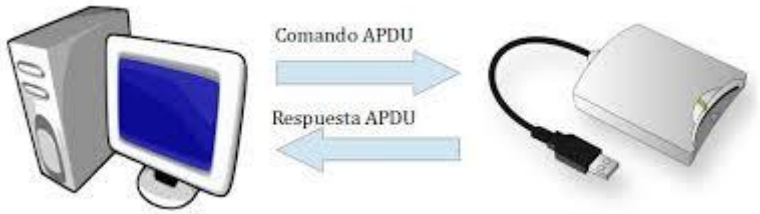
Figura 2.3 Contactos del chip de una tarjeta con contactos

- Vcc: Se utiliza para administrar alimentación al chip. El voltaje que se aplica es de 3V o 5V.
- RST: Es el contacto por el que se envía la señal de reset al microprocesador.
- CLK: Señal externa de reloj, que se usara como referencia para el reloj interno.
- GND: Conexión a tierra.
- Vpp: Conector actualmente en desuso, ya que se usaba para proporcionar poder para grabar y borrar las memorias EEPROM, pero actualmente ese voltaje se genera internamente,
- I/O: Se usa para transmitir datos entre dispositivo lector y la tarjeta en modo half-duplex.

2.2.3 Protocolos de comunicación

Toda comunicación con la tarjeta inteligente es iniciada por un dispositivo externo. Una tarjeta no transmite información sin una petición externa previa, lo que crea una relación maestro-esclavo, siendo el terminal el maestro y la tarjeta el esclavo.

La transmisión de datos se realiza empleando el sistema asíncrono. Debido a la existencia de un único canal entre los dispositivos externos y la tarjeta, es necesario que ambos se turnen a la hora de realizar la transmisión de datos, estableciéndose lo que se denomina canal half-duplex (Figura 2.4). El procedimiento full-duplex aún no se encuentra disponible, aunque la mayoría de microprocesadores incorporan un canal de entrada y otro de salida.



.

Figura 2.4 Modelo de comunicación

A la hora de insertar la tarjeta en el lector, sus contactos se conectan a los del terminal y se procede a activarlos. La tarjeta entonces se enciende y envía una respuesta llama ATR (Answer To Reset) hacia el terminal. El ATR contiene información referente a cómo tiene que realizarse la comunicación tarjeta-lector, estructura de los datos, protocolo de transmisión, … Una vez que el lector interpreta el ATR, envía la primera instrucción. La tarjeta procesa la instrucción y envía la respuesta asociada.

La información de niveles superiores se intercambia mediante unidades APDU (Application Protocol Data Unit). Su formato está especificado en ISO 7816-4 [5], pero el contenido y significado es especifico de cada aplicación. Existen dos tipos de unidades APDU, representados en la Figura 2.5:

- comando APDU: Encargado de transferir datos de la aplicación externa a la tarjeta.
- Respuesta APDU: Enviada de la tarjeta al lector.

Comando APDU

CLA	INS	P1	P2	Lc	Datos Opcionales	Le

Respuesta APDU

Datos Opcionales	SW1	SW2

Figura 2.5 Estructura y formato APDU

Cada comando está formado por:

- Un byte CLA, que indica la clase de instrucción.
- Un byte de instrucción INS, que determina el comando enviado.
- Dos bytes, P1 y P2, que se utilizan como parámetros específicos.
- Un byte Lc, especificando la longitud de los datos, si los hubiera.
- Una serie de bytes de datos de longitud variable con datos opcionales.
- Un byte Le con la longitud prevista de los datos de la APDU de respuesta.

La respuesta contiene:

- Datos opcionales como respuesta.
- Dos bytes de estado SW1 y SW2, conocidos como Status Word, que almacenan el resultado de la operación.

Las APDU son transmitidas por el protocolo de nivel inferior a través de TPDU (Transmission Protocol Data Unit). En ISO 7816-3 [4] se especifican varios protocolos de transmisión asíncrono y half-duplex. El primero T=0, es un protocolo orientado a byte (cada byte es transmitido de forma independiente). El método de corrección de errores empleado es un bit de paridad, y todo byte erróneo es retransmitido. El segundo T=1, es un protocolo orientado al bloque (paquetes de bytes). Este modo permite control de flujo en ambas direcciones, por lo que la tarjeta puede tomar control de la aplicación. T=0 es un protocolo más sencillo y es el elegido en la mayoría de aplicaciones de monedero electrónico y en GSM.

2.2.4 Sistema operativo

El sistema operativo de una tarjeta inteligente soporta un pequeño conjunto de instrucciones a través del cual interactuar con ´esta para ejecutar las operaciones definidas. ISO 7816-4 [5] estandariza un amplio rango de instrucciones en forma de APDU, que pueden, o no, estar implementadas en su totalidad por el sistema operativo.

La mayoría de los sistemas operativos de las tarjetas inteligentes están basados en ISO 7816-4, que siguen una estructura jerárquica distinguiéndose tres tipos de ficheros:

- Fichero Maestro o MF (Master File): es el directorio raíz del sistema, seleccionado automáticamente al iniciar la tarjeta. En él están todos los directorios y ficheros, y cuenta con sus propios atributos de seguridad.
- Fichero Dedicado o DF (Dedicated File): es un directorio que puede contener ficheros, o bien otro DF. Usualmente cada fichero dedicado contiene información de una aplicación distinta.
- Fichero Elemental o EF (Elementary File) es un archivo que contiene datos. Su tamaño se ha de especificar al momento de su creación, y se pueden distinguir tres tipos:
    - Transparentes: no poseen ningún tipo de estructura y los datos se encuentran en secuencias de bytes.
    - Lineales: basados en celdas o registros de longitud fija o variable.
    - Cíclicos: formados por registros de longitud fija accedidos cíclicamente.

Estos sistemas operativos, centrados en la gestión de ficheros, son los más extendidos en las tarjetas inteligentes hoy día, aunque con el paso del tiempo, están surgiendo nuevos sistemas que soportan un modelo multinivel o multicapa, en las que se incluyen las tarjetas inteligentes Java Card.

Java Card se define como una plataforma segura, portable y multiplicación, que permite a las tarjetas ejecutar pequeñas aplicaciones, denominadas applet, que utilizan la tecnología Java.

2.3 Tarjetas de proximidad

Las tarjetas inteligentes sin contactos o contactless utilizan tecnologías de radiofrecuencia para la transmisión de información de manera bidireccional con lectores de proximidad. Estas tarjetas están diseñadas de modo que sea posible adherir opcionalmente un modulo de chip inteligente de contacto. Esto hace posible la fabricación de tarjetas inteligentes

duales o aquellas que permiten el acceso al procesador mediante interfaces con contacto e inalámbricos.

Para iniciar la comunicación, el lector provee energía a la tarjeta mediante un campo magnético, este campo se crea haciendo uso de una bobina. Cuando una tarjeta se acerca al campo magnético, una parte de energía es absorbida y almacenada por la bobina de la tarjeta, dotándola de la energía necesaria para la comunicación. Si la tarjeta quiere enviar datos al lector, se usará una técnica de modulación digital, destacando las más utilizadas: ASK (Amplitude-Shift Keying), FSK (Frequency-Shift Keying) y PSK (Phase-Shift Keying).

Los estándares más importantes son:

- ISO 14443: Define toda la información relacionada con las tarjetas de proximidad, y es el más extendido en la actualidad. Trabaja a una frecuencia de 13.56 MHz y permite una distancia máxima de lectura de 10 cm, aunque en usos cotidianos suele ser más limitada.
- ISO 15693 [9]: Es un estándar que defina las tarjetas de vecindad. Estas tarjetas aumentan la distancia de lectura, alcanzando hasta un metro de distancia. Trabajan en la banda de 13.56 MHz, al igual que en ISO 14443.

La mayoría de tarjetas sin contactos utilizadas actualmente se conocen como tarjetas MIFARE, cuya tecnología esta descrita en el estándar ISO/IEC 14443 Tipo A.

2.3.1 RFID

RFID (Radio Frequency Identification) es un sistema de almacenamiento de información cuyo objetivo principal es el poder identificar un sujeto por radiofrecuencia, y gracias a esto, no es necesario una visión directa entre emisor y receptor. En la Figura 2.6 se puede observar la arquitectura del sistema.

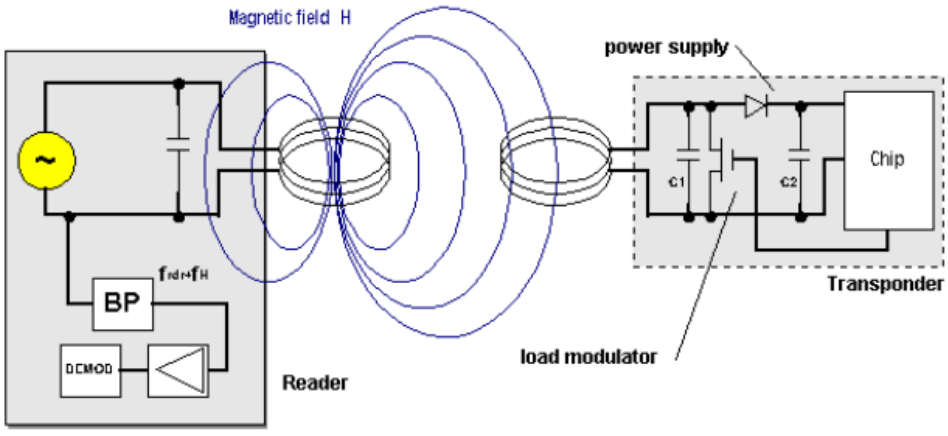


Figura 2.6 Alimentación de una etiqueta RFID o transpondedor (derecha) mediante la corriente inducida por el campo magnético generado por el lector (izquierda).

Un sistema RFID está formado por los siguientes componentes:

- **Etiqueta RFID**: compuesta por una antena, un transductor radio que transforma la energía electromagnética en energía eléctrica para alimentar el

circuito contenido en la etiqueta y un chip que contiene la información de identificación. El objetivo de la antena es permitir al chip intercambiar con el lector la información que almacena. Según el tipo de memoria del chip, existen dos grupos:
    - Solo lectura: la información es grabada en el proceso de fabricación de la etiqueta y no puede ser modificada
    - De lectura y escritura: la información de la etiqueta puede ser consultada y modificada por el lector.
- **Lector RFID**: Compuesto por una antena cuya función es la misma que la de la etiqueta RFID, un transceptor y un decodificador. El lector envía periódicamente señales para comprobar si existe alguna etiqueta presente en el campo electromagnético que genera y, cuando capta la presencia de una etiqueta, empieza el proceso de anticolisión para identificar a una tarjeta en el caso que haya más de una y extrae la información de identificación para pasársela al subsistema de procesamiento de datos.
- **Middleware RFID**: es el subsistema de procesamiento de datos que proporciona el procesado y almacenamiento de los mismos. Se trata de un sistema intermedio entre los lectores y los sistemas que hay por detrás de estos cuya función es tratar de manera apropiada la información de identificación captada por el hardware y transmitirla a los sistemas que, posteriormente, tendrán que hacer uso de ella.

Existen diferentes tipos de etiquetas RFID. Según la manera de establecer la alimentación del chip y la comunicación física podemos clasificarlas en tres grupos:

- **Pasivas**: esta clase de etiquetas no poseen alimentación eléctrica propia. Cuando se aproximan al lector y son introducidas en el campo magnético que este genera, se crea una corriente inducida de baja intensidad, pero que es suficiente para alimentar el chip de la etiqueta y que esta sea capaz de generar una respuesta y devolverla al lector. La distancia a la que pueden operar este tipo de etiquetas es de unos pocos centímetros. Son las más utilizadas actualmente debido a su bajo coste y son las que se han utilizado en este proyecto.
- **Semi-pasivas**: el comportamiento de este tipo de etiquetas es el mismo que en las pasivas, pero se diferencian en que poseen una pequeña fuente de alimentación propia que utilizan, principalmente, para alimentar el microchip y no tener que hacerlo a partir de la señal recibida por parte del lector como en el caso de las pasivas. Su rango de operación sigue siendo de unos pocos centímetros.
- **Activas**: de igual forma que las semi-pasivas, las etiquetas activas poseen su propia fuente de energía que utilizan tanto para alimentar sus circuitos integrados como para generar y propagar su señal al lector. Estas son mucho más fiables en relación a la generación de errores y, gracias a su fuente de energía, son capaces de transmitir señales más potentes que las de las pasivas, lo cual implica que la distancia de operación con el lector pueda ser mayor hasta alcanzar incluso algunos metros.

El sistema de identificación RFID puede trabajar también a diferentes rangos de frecuencia, que se pueden dividir en cuatro grupos:

- Baja frecuencia (“Low Frequency”) 125 o 134.2 KHz.
- Alta frecuencia (“High Frequency”) 13.56 MHz.
- Frecuencia ultra elevada (“Ultra High Frequency”) de 860 a 960 MHz.
- Microondas: 2.4 GHz

En cuanto a los estándares de las etiquetas RFID, el que más interesa para este proyecto es el estándar ISO 14443. Esta trata de las tarjetas de corto alcance o de proximidad. En esta categoría se incluyen desde las tarjetas bancarias hasta las tarjetas MIFARE que se han utilizado en este proyecto. Esto engloba a las tarjetas del rango 13.56 MHz, que trabajan a un rango de unos 10 cm del lector.

Acorde a las clasificaciones descritas, este proyecto se basa en el uso de elementos pasivos, como lo son las tarjetas de la familia MIFARE, que se alimentan gracias a un elemento activo, el teléfono móvil.

2.3.2 ISO 14443

ISO/IEC 14443 es un estándar internacional relacionado con las tarjetas y dispositivos de seguridad de identificación personal electrónicas, en especial las tarjetas de proximidad, que operan a una distancia de 10 cm y a una frecuencia de 13.56 MHz.

El estándar ISO/IEC 14443 consta de cuatro partes y se describen dos tipos de tarjetas: tipo A y tipo B. Las principales diferencias entre estos tipos se encuentran en los métodos de modulación, codificación de los planes (parte 2) y el protocolo de inicialización de los procedimientos (parte 3). Las tarjetas de ambos tipos (A y B) utilizan el mismo protocolo de alto nivel (llamado T=CL) descrito en la parte 4.

- Parte 1 se encarga de definir las características físicas, y hacen referencia a otros estándares como son el ISO 7810:2918 [7], ISO/IEC 7816 [2] o el ISO/IEC 15457-1 [8].
- Parte 2 especifica la potencia de radiofrecuencia y la señal de la interfaz para cada tipo de tarjeta (A y B).
- Parte 3 explica los mensajes enviados entre el lector y la tarjeta, y se define para cada tipo de tarjetas un método anticolisión. Para el tipo A se utiliza un método de anticolisión denominado arbitraje sabio y para el tipo B es el método de intervalo de tiempo.
- Parte 4 se define un protocolo de transmisión de bloque semidúplex. Esta sección define el formato de los datos y los elementos que permiten la comunicación durante una transmisión de información.

2.3.3 NFC

Las siglas NFC hacen referencia al nombre Near Field Communication. Es una tecnología de comunicación de corto alcance (10 cm) y alta frecuencia (13.56 MHz), que está basado en el estándar ISO 14443, ya que hereda del RFID.

Esta tecnología está siendo ampliamente utilizada sobre todo en tarjetas bancarias para realizar pagos más rápidamente. También se ha introducido en los móviles de última generación y facilita el intercambio de datos entre diferentes terminales a la vez que se pueden emular tarjetas bancarias en el propio teléfono.

Como se especifica en la norma ISO 14443, NFC se comunica usando el mismo concepto de inducción magnética que usa RFID y trabaja en la banda de los 13,56MHz. La velocidad de transferencia de datos puede variar desde los 106 kbps a los 848 kbps. Esto hace que las tarjetas con NFC deben estar muy próximas al lector, casi en contacto con este, para poder funcionar.

En la Figura 2.7 se ilustra la pila de protocolos NFC.

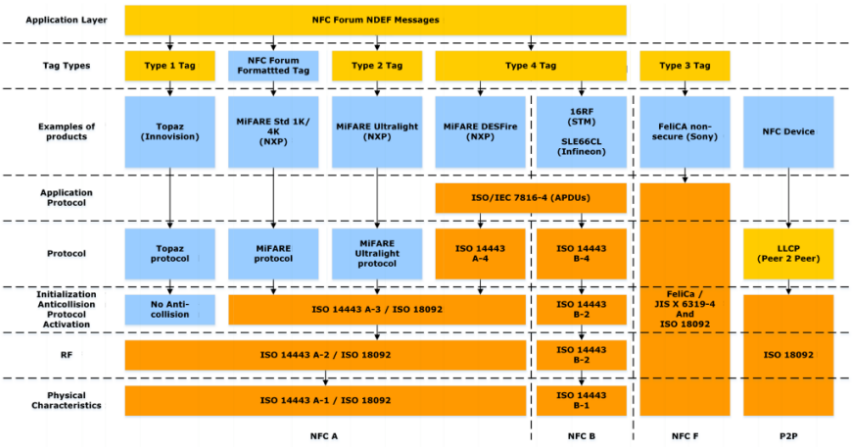


Figura 2.7 Pila de protocolos NFC

En la forma de funcionar de NFC, podemos encontrar dos partes claramente diferenciadas a las que se les conoce como dispositivo iniciador y el objetivo. El primero es quien inicia la comunicación y se encarga de mantener toda la sesión y gestionar la transferencia de los datos, y el segundo simplemente se limita a responder a las peticiones del iniciador. La comunicación que se produce entre los dos componentes puede realizarse de dos modos distintos, al igual que RFID, activo o pasivo.

Es posible también establecer una clasificación según los elementos que participan en ésta y los modos de operación o roles que se toman en la comunicación. Estos elementos son siempre utilizados en uno de los siguientes modos de operación:

- -**Modo Lector/Escritor**: en este modo de operación el dispositivo activo (normalmente un lector NFC o un teléfono móvil) es el que inicia la comunicación y puede tanto leer como modificar datos almacenados en etiquetas NFC. Normalmente el otro extremo es un elemento pasivo que recibe energía del elemento activo por medio de la corriente generada por el campo magnético y contesta las peticiones del lector.
- -**Modo Peer-to-peer**: en este modo entran en juego dos dispositivos activos donde ambos nodos interactúan sin que exista una relación maestro-esclavo. Cualquiera de los dos extremos puede establecer una conexión bidireccional para intercambiar dicha información.
- -**Modo de emulación de tarjetas**: en este modo de operación los dispositivos NFC (típicamente teléfonos móviles o relojes) usan protocolos similares a los definidos en los estándares para tarjetas contactless para emular el comportamiento de estas vías software y usando también el hardware de estos dispositivos. Para poder igualar la seguridad que daría un elemento NFC real, se

almacenan las aplicaciones y la información sensible en el elemento seguro (Secure Element), que en el caso de los teléfonos móviles podría ser la tarjeta SIM (Subscriber Identity Module) .

En el caso del sistema presente en este proyecto, el modo de operación utilizado es el Lector/Escritor, donde el teléfono móvil ejerce el papel de elemento escritor y lector de las tarjetas MIFARE.

2.3.4 **Vulnerabilidades**

La posibilidad de integrar muchas funcionalidades en un único teléfono móvil hace muy atractivo el uso de la tecnología NFC. Sin embargo, también conlleva algunas vulnerabilidades e inseguridades para el usuario. Gracias a las distancias cortas entre terminales necesarias para su funcionamiento, se impide en gran medida las escuchas o data "sniffing".

Aun así, existen varios tipos de ataques conocidos por el público. Estos métodos principales son:

- Eavesdropping o espionaje: en este escenario, el atacante hace uso de una antena para recoger la información enviada entre terminales, como se muestra en la Figura 2.8. Este ataque puede realizarse incluso pese a las limitaciones de distancias de la tecnología NFC, ya que no siempre este ataque es usado para robar la información, en ocasiones es usado para corromper los datos enviados, haciéndolos inútiles.

Figura 2.8 Ataque por escucha o eavesdropping

  Para prevenir estos ataques, el método más usado es establecer un canal seguro, mediante el uso de algoritmos de encriptación.

- Modificación de datos o man-in-the-middle: en este escenario, los datos transmitidos son capturados y modificados por el dispositivo de radiofrecuencia del atacante. Normalmente, este tipo de ataque es difícil de realizar, pero hay ocasiones en las que es posible su implementación.

  Para realizar este ataque, se usa un perturbador de frecuencia o "jammer", modificando la frecuencia para engañar al emisor, haciéndole creer que el dispositivo atacante es el más cercano y probablemente más valido.

- Relay attack o retransmisión: se trata de un tipo de ataque man-in-the-middle, ya que el atacante interviene directamente en la comunicación, haciéndose pasar por uno de los participantes. El atacante tiene que retransmitir las respuestas a ambos lados en tiempo real, para hacerse pasar por un participante real.

  Este ataque puede ser utilizado para aumentar el rango de comunicación haciendo uso del propio canal de comunicación entre los dispositivos. Esto posibilita la comunicación a una distancia mayor de lo teóricamente posible. Para proteger al usuario de este ataque, se pueden implementar restricciones temporales en la comunicación para evitar información ilegitima.

3 Familia de tarjetas MIFARE

MIFARE es la marca comercial propiedad de NXP Semiconductors que fue división de Philips Semiconductors hasta que la empresa se separó de Philips.

MIFARE surge como una de las primeras tarjetas en ser usadas de manera comercial, por lo que al ser tan prematuras, no satisfacen estándares como tal. Aunque cumplían con el ISO 14443 en sus tres primeras partes, para comunicarse con ellas era necesario usar unos comandos únicamente válidos para esta familia de tarjetas.

Según avanzó la tecnología de MIFARE, se crearon nuevos modelos de tarjetas, entre las que se engloban MIFARE Plus y MIFARE DESFire, que presentaban una mayor seguridad, y cumplían con el estándar ISO 14443.

La marca MIFARE ofrece diferentes productos y tecnologías basada en el estándar ISO/IEC 14443 Type A 13.56 MHz. Las aplicaciones más comunes son:

- Sistema de billetaje de transporte público
- Identificación de tarjetas para controles de acceso
- Tarjetas para sistemas de fidelización, micro pagos, sistemas de pago en peajes, billetaje en móviles (mediante uso de pegatinas contactless), tarjetas de identificación ciudadana, identificación para acceso en parkings, y un largo etc.

Cabe destacar que otra cosa que define el estándar ISO/IEC 14443, es la existencia de dos tipos de tarjetas a nivel de comunicación, conocidas como NFC A y NFC B. En concreto, nos interesa el tipo NFC A, ya que es la categoría en la que se encuentra la familia MIFARE. Este tipo A, utiliza el llamado “Miller encoding”, que es usado con una amplitud de modulación del 100%, transmitiéndose los datos a 106 kbps.

3.1 Funcionamiento

Antes de realizar cualquier operación sobre la memoria de la tarjeta es de obligado cumplimiento la ejecución de una autenticación mutua y securización del canal de comunicación entre el lector y la tarjeta. Dicho procedimiento se lleva a cabo mediante un triple intercambio según ISO 9798-2 [6]. El proceso consiste en la generación de un numero aleatorio que, haciendo uso de las claves alojadas en la tarjeta, se envía al lector. El lector responde entonces con un mensaje de confirmación a la tarjeta. Esta operación se realiza tres veces para verificar que la tarjeta que se ha presentado ante el lector es válida. En caso de resultar satisfactorio el procedimiento comentado se garantiza el acceso a la información dentro del bloque o sector autenticado.

3.2 Identificación del tipo de chip

Normalmente, el PCD (Proximity Coupling Device) sondea en busca de PICCs (Proximity Integrated Circuit Card) en el entorno cercano, mediante el uso de REQA (Request Command, Type A). Cuando un PICC recibe el REQA, cualquier PICC de la familia MIFARE retorna un ATQA (Answer To Request).

La completa activación de la tarjeta se realiza acorde con la Figura 3.1.

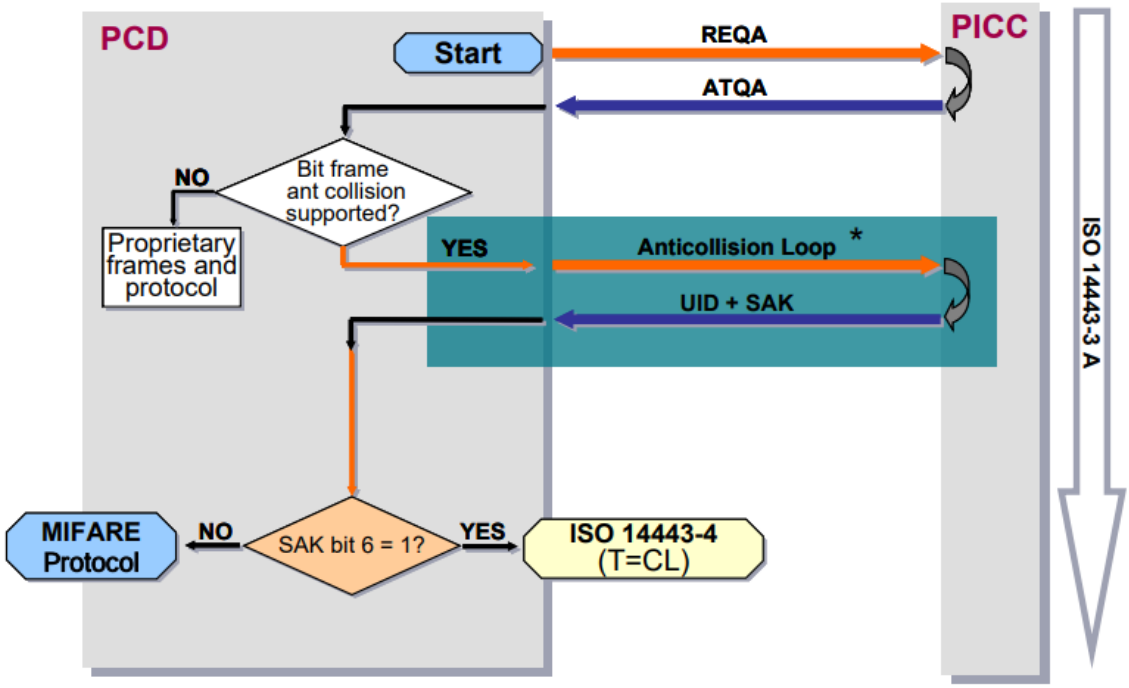


Figura 3.1 Comienzo de activación de la tarjeta.

Gracias al contenido del SAK (Select Acknowledge), el lector puede diferenciar el tipo de tarjeta de la familia MIFARE que se está comunicando, como se muestra en la Figura 3.2. En esta figura, también se pueden observar las tarjetas que cumplen con la ISO 14443-4 y las que únicamente soportan hasta el nivel ISO 14443-3 [26].

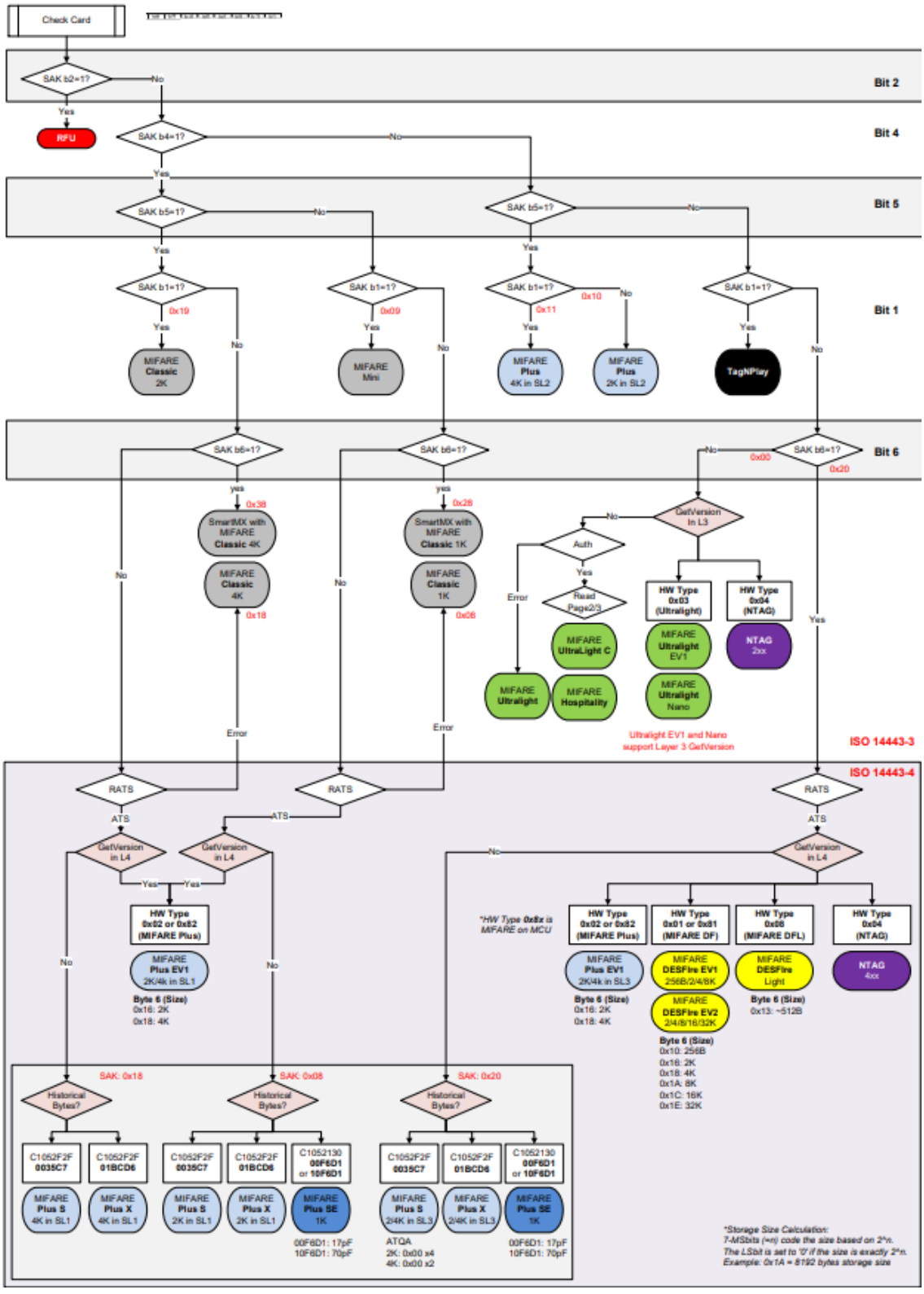


Figura 3.2 Ejemplo de todas las tarjetas MIFARE y como identificarlas.

3.3 MIFARE Classic

El chip MIFARE Classic fue lanzado en 1994 y básicamente se trata de una memoria que se encuentra dividida en sectores y bloques con un sistema de seguridad bajo [15].

Bajo la familia de MIFARE Classic se ofrecen varios productos:

- MIFARE Classic de 1K con una capacidad de 1024 bytes, dividida en 16 sectores y protegida por 2 claves de seguridad diferentes (A y B). Cada sector presenta 4 bloques de 16 bytes.
- MIFARE Classic de 4K con una capacidad de 4096 bytes, dividida en 40 sectores.
- MIFARE mini con 320 bytes divididos en 5 sectores.

Para que el lector pueda leer y escribir el contenido de un sector, es decir, cualquiera de los 4 bloques de datos que tiene, necesita en primera instancia autenticarse contra él. Este proceso se realiza si y solo si la clave utilizada por el lector coincide con la que el sector tráiler alberga. El esquema de memoria viene representado por la Figura 3.3

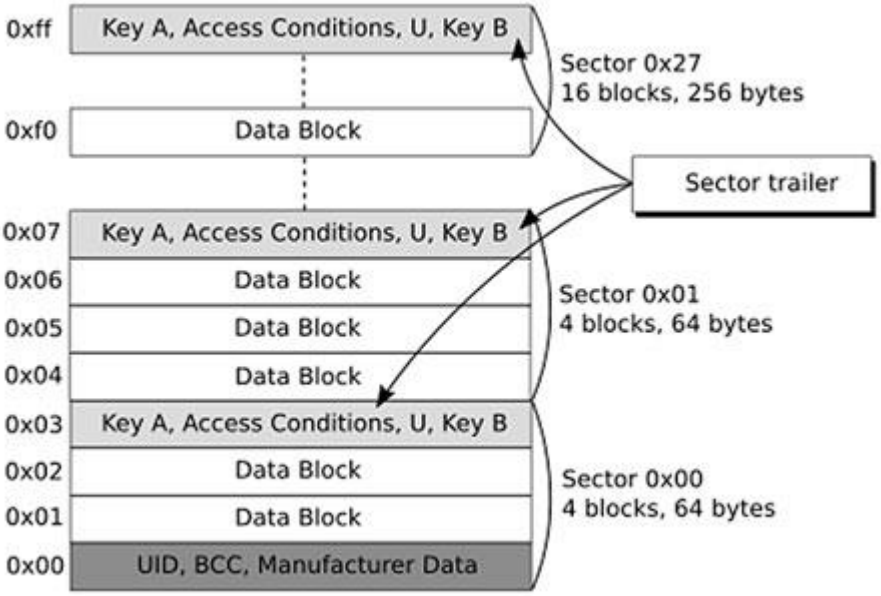


Figura 3.3 Estructura de una MIFARE Classic.

Si nos ponemos a analizar la estructura de memoria de una tarjeta MIFARE Classic más detenidamente, lo primero que nos encontramos es el UID (Unique ID) de la tarjeta, ubicado en el sector 0. Este UID es un identificador único a nivel mundial puede ser de 4, 7 o 10 bytes.

El resto de sectores, se organizan mediante el siguiente esquema:

- Bloques 0, 1 y 2: estos bloques son los que contienen la información. Los datos se pueden guardar en formato del tipo Valor (contador), para después aplicarles operaciones de incremento/decremento, o como bytes puros de información.
- Bloque 3 o sector tráiler: en este bloque se almacenan las claves criptográficas (A y B), para poder acceder a los datos que guarda este sector, y los bits de acceso, que indican las operaciones que podemos realizar sobre el sector.

En 2008, un grupo de investigadores de la universidad de Radboud consiguieron publicar el algoritmo Crypto-1, que fue inventado por Philips, y hasta ese momento era desconocido, por lo que el algoritmo quedo completamente roto [10]. Gracias a esto, es posible extraer las claves de una forma más sencilla evitando ataques por fuerza bruta, que podrían durar una media de 9 horas, acorde a la información del reporte TNO 34643 [11]

Hoy en día, además es posible adquirir tarjetas MIFARE Classic a las que es posible cambiar el UID, que debería ser único, por lo que es posible la clonación de tarjetas [22].

Para solventar todos estos inconvenientes, se creó la MIFARE PLUS, que es la transición lógica de la MIFARE Classic

3.4 MIFARE Plus

La MIFARE Plus, nació para solventar los problemas de seguridad de la MIFARE Classic, y dispone de algoritmos AES que impiden su copia o falsificación. Estas tarjetas están disponibles en dos versiones, MIFARE Plus 2K y MIFARE Plus 4K de memoria, y poseen un número de serie único de 7 bytes [12].

Estas tarjetas suponen un alto nivel de seguridad, para usuarios cuyo presupuesto no les permite optar por las MIFARE DESFire. Incluso permite su uso en una infraestructura existente de MIFARE Classic, aunque de ser necesarias las características de seguridad mencionadas deberá realizarse la actualización del entorno (software de gestión y lectores). Estas tarjetas disponen también de compatibilidad con el estándar ISO 14443-4, al contrario que la MIFARE Classic.

Su memoria está organizada en 32 sectores de 4 bloques cada uno, como se puede observar en la Figura 3.4. Como se puede observar, el primer sector, posee el denominado “manufacturer block”, en la que se encuentra la información como es el UID, y no puede ser modificada, ya que posee atributo de solo lectura. En el resto de bloques, como en la MIFARE Classic, es posible guardar datos en bytes, o en formato del tipo Valor, donde es posible realizar operaciones de incremento, decremento de su contenido numérico.

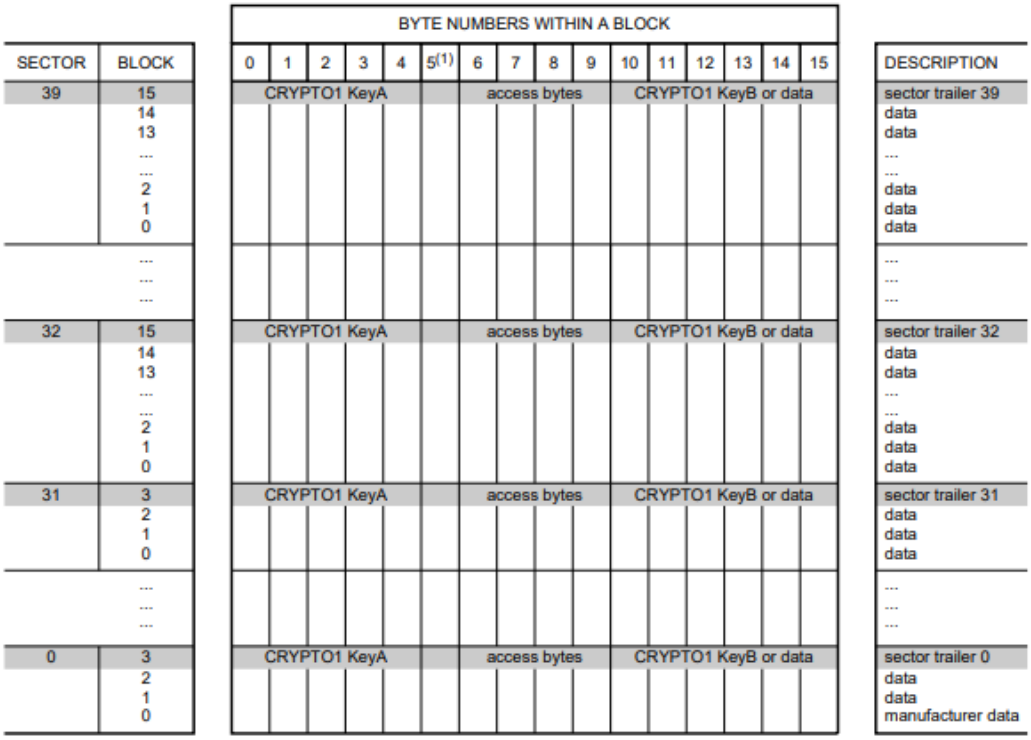


Figura 3.4 Organización de memoria de la MIFARE Plus

3.5 MIFARE Ultralight

La principal característica de las tarjetas MIFARE Ultralight es que no ofrece seguridad criptográfica y dispone tan solo de 64 bytes de memoria, dividida en 16 páginas de 4 bytes [13].

Esta tarjeta está enfocada a las aplicaciones de un solo uso, por ello al escribir en memoria una sola vez, se bloquea la función de escritura.

Sin embargo, su evolución, la MIFARE Ultralight EV1, incluye funciones de seguridad como son: bloqueo (por página) de los bits de reescritura con contadores configurables, password de 32 bits y contadores para evitar recargas.

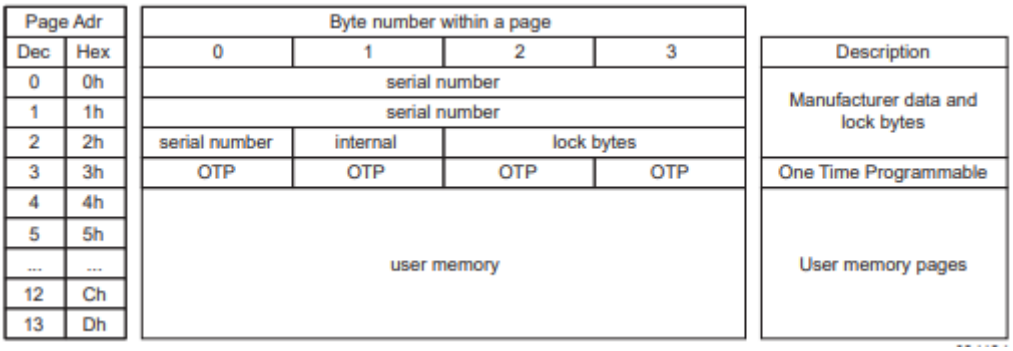


Figura 3.5 Organización de memoria de MIFARE Ultralight

Como se puede observar en la Figura 3.5, los cuatro primeros bloques no son zonas diseñadas para la escritura. En ellas se encuentra el UID de 7 bytes, así como datos de la propia tarjeta. En el tercer bloque, está reservado como zona OTP (One-Time

Programmable) de solo escritura, y a partir del cuarto bloque, es memoria dedicada al usuario, hasta 320 bits.

Estas tarjetas, al carecer de seguridad, están pensadas para un uso único, por lo que es común verlas en el formato pegatina. Debido a la ausencia de seguridad, y para solventar la ausencia de autenticación, se creó la evolución lógica, conocidas como MIFARE Ultralight C.

3.6 MIFARE Ultralight C

Se trata de una versión de bajo coste para aplicaciones de un sólo uso, pero con seguridad 3DES que permite ser integrado en muchas infraestructuras existentes sin tener que realizar modificaciones. La autentificación 3DES previene la clonación de las tarjetas [14].

Este chip cuenta con una capacidad de 192 bytes con la misma escritura que la MIFARE Ultralight (páginas de 4 bytes). Está dividida de manera similar a la MIFARE Ultralight original, incluyendo también zonas OTP, de 32 bits. Además, cumple con las normas ISO/IEC 14443 partes 1-3.

El número de serie único de 7 bytes (UID), junto a sus dos Block Check Character Bytes (BCC), están escritos en los primeros 9 bytes de la memoria, al igual que la MIFARE Ultralight, como se puede observar el la Figura 3.6.

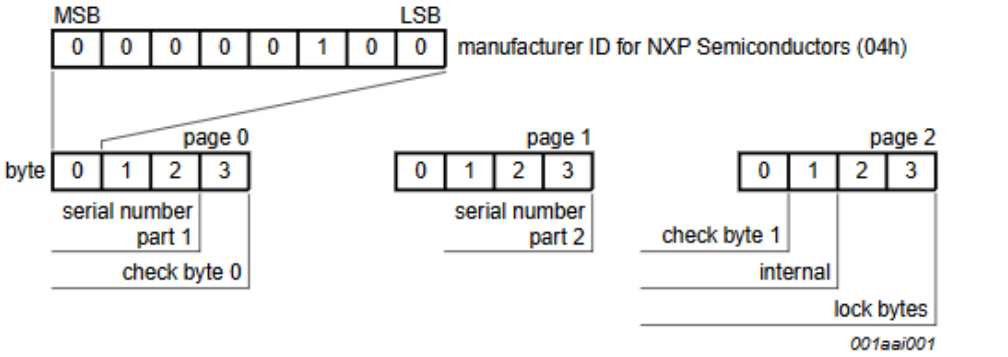


Figura 3.6 UID/Serial number de una MIFARE Ultralight C.

3.7 MIFARE DESFire

MIFARE DESFire [16] ofrece características de seguridad en hardware y software avanzadas en comparación a las características de MIFARE Classic.

Una de las mayores diferencias que añaden es que tienen su propio sistema operativo que implementa un sistema de ficheros flexible y habilita transacciones más rápidas y seguras.

Una gran mejora que añadieron estas tarjetas fue el cambio de algoritmo de autenticación y cifrado, pasando a usar el algoritmo DES (Dara Encryption Standard), con la posibilidad de poder usar DES simple o triple DES (3DES) con dos o tres claves. El algoritmo 3DES se basa en el algoritmo DES, que aplica una serie de operaciones básicas para convertir un texto en otro cifrado, empleando una clave criptográfica. 3DES es el algoritmo que hace triple cifrado del DES, es decir, aplicando el algoritmo DES tres veces seguidas, obteniendo un resultado más seguro [35].

Las tarjetas MIFARE DESFire EV1, al igual que la generación anterior, poseen un sistema de ficheros flexible.

La Figura 3.7 muestra la estructuración del sistema de ficheros.

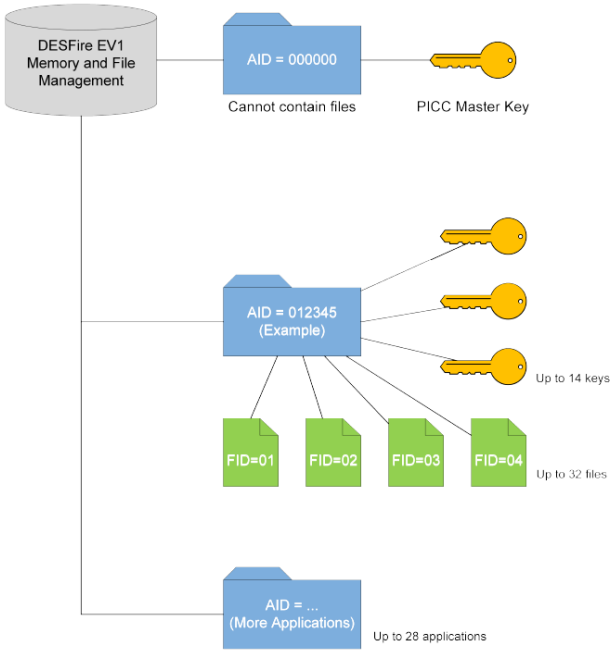


Figura 3.7 Estructuración de los datos en una DESFire EV1.

Existen 4 variantes según su capacidad de memoria y características de encriptación:

- Memoria con 4 KBytes , encriptación 3DES, para MIFARE DESFire
- Memoria con 2 KBytes, encriptación AES, para MIFARE DESFire EV1
- Memoria con 4 KBytes, encriptación AES, para MIFARE DESFire EV1
- Memoria con 8 KBytes, encriptación AES, para MIFARE DESFire EV1

3.7.1 **Mecanismos de seguridad**

En el caso de MIFARE DESFire, existe el denominado “Diversified keys”, que ofrece una capa de protección extra. Este protocolo calcula las claves usando una clave maestra, y el denominado “diversification input”, el cual contiene el UID de la tarjeta, y datos introducidos por el propio usuario, por lo que aseguramos un resultado distinto para cada tarjeta [23]. Este UID es grabado en cada tarjeta en el proceso de producción, y no puede ser alterado, asegurando la autenticidad del producto.

La MIFARE DESFire posee comandos compatibles con el ISO 14443-4, como es el uso del RATS (Request Answer To Select), que permite la identificación del tipo de MIFARE DESFire al PCD, lo que supone una mayor compatibilidad con los sistemas actuales. También posee comandos únicos, relacionados bien con la seguridad o bien con procesos de lectura/escritura de la tarjeta.

Uno de los añadidos de la versión EV1 frente a la tarjeta Mifare DESFire original, es que permite utilizar el algoritmo AES (Advanced Encryption Standard) para los procesos de

autenticación y cifrado con claves de 128 bits. Esto garantiza al desarrollador alcanzar máxima garantía de seguridad, al usar claves más largas y crear un mejor sistema global.

Antes de poder ejecutar cualquier operación sobre los ficheros de una aplicación, se debe establecer un canal seguro de comunicación previa autenticación, proceso durante el cual el PCD y la PICC demuestran conocer un secreto común (en este caso una clave criptográfica). La Figura 3.8 muestra todo el proceso que se describe a continuación:

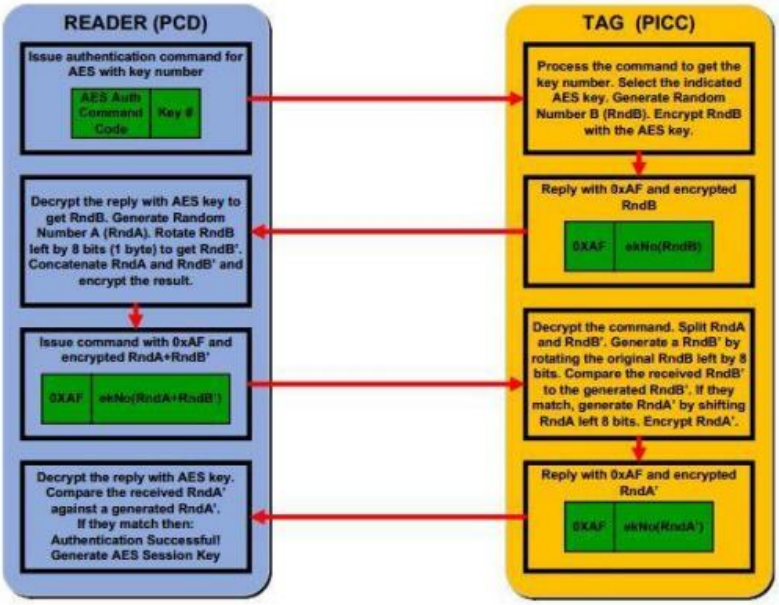


Figura 3.8 Autenticación mediante algoritmo AES.

1. El lector inicia la comunicación solicitando la ejecución del comando Authenticate e indicando el número de clave (key #) con la que se desea autenticar. En caso de que el número de clave sea 0x00, será la clave maestra (de la tarjeta o de la aplicación, dependiendo de la que se haya seleccionada previamente) la que se utilice para la autenticación
2. Si dicha clave se encuentra almacenada en la tarjeta, esta responde a la petición mediante el cifrado de un número aleatorio generado por la tarjeta (RndB7) con la clave especificada por el lector. En caso contrario, se devuelve un código de error.
3. El lector descifra el mensaje recibido usando la misma clave para recuperar el número aleatorio RndB y lo rota hacia la izquierda 8 bits (RndB'). A continuación, el lector genera otro número aleatorio (RndA) y lo concatena con RndB'. Este aplica una operación de descifrado sobre el resultado de la concatenación con la clave seleccionada y lo envía de vuelta a la tarjeta.
4. La tarjeta cifra el mensaje recibido y, de este modo, es capaz de verificar el número RndB' obtenido y compararlo rotando internamente el original hacia la izquierda en 8 bits. Si la verificación es correcta, se realiza la misma rotación sobre el número aleatorio RndA, obteniendo RndA', y se envía el resultado cifrado de vuelta al lector. En este punto, si todo ha ido bien, la tarjeta ya confía en el lector.

5. Finalmente, el lector descifra el mensaje recibido por la tarjeta y compara el valor RndA' con el RndA original rotado. Si ambos coinciden, el lector podrá confiar en la tarjeta y el proceso de autenticación habrá sido satisfactorio.Una vez las dos partes están autenticadas, se genera una clave de sesión, que puede variar entre 8 y 24 bytes en función del algoritmo elegido.

La MIFARE DESFire EV2 [17] surgió como una evolución de la MIFARE DESFire EV1, posee el mismo sistema flexible de ficheros que la DESFire EV1, pero implementa mejoras en el mismo, ya que no posee límite de aplicaciones simultaneas. Posee un mayor nivel de seguridad al implementar control de proximidad, lo que permite a la tarjeta confirmar al lector que realmente se encuentra cerca del lector, evitando ataques de man-in-the-middle.

Mifare DESFire EV2 es totalmente compatible con la infraestructura relacionada con la familia DESFire EV1. Esto quiere decir que los lectores compatibles con la DESFire EV1 son compatibles con el modelo EV2, permitiendo una actualización del sistema de seguridad de EV1 a EV2 sin invertir grandes cantidades de dinero [24].

4 librería TapLinx

MIFARE SDK constituyó la aplicación de desarrollo original para personas que deseaban crear sus propias aplicaciones NFC para teléfonos inteligentes Android. Sin embargo, NXP ahora ha hecho una actualización de MIFARE SDK a TapLinx, que permite diseñar y crear aplicaciones de manera más fluida y rápida.

Esto supone la compatibilidad con una mayor variedad de productos, más funciones y una API abierta con nuevo diseño. Por tanto, el diseño de aplicaciones NFC nuevas, se ha simplificado para facilitar el trabajo de los programadores.

El producto se distribuirá gratuitamente sin ningún derecho de licencia, a diferencia del MIFARE SDK avanzado.

Las ventajas principales que ofrece TapLinx a sus desarrolladores, radica en la sencillez de su uso, haciendo este fácil y rápido. A lo anterior hay que unir también a su gran comunidad de desarrolladores, junto a los foros de ayuda creados. En caso necesario, existe también el soporte técnico de TapLinx para dudas más específicas.

4.1 Obtención de TapLinx

La primera fase de este proyecto, consiste en familiarizarse con la librería TapLinx, realizando ejemplos sencillos de lectura/escritura, sobre los diferentes tipos de tarjetas. Todos estos ejemplos serán desarrollados sobre la plataforma AndroidStudio. Para instalar TapLinx, es necesario seguir una serie de pasos. Lo primero es registrarse en su página web y aceptar el EULA (End-User License Agreement). Una vez hecho esto, es necesario elegir un nombre del paquete, el cual debe ser el mismo que el nombre de nuestro proyecto de Android. Tras esto, se nos dará una clave única para nuestro proyecto, que deberá ser usada más adelante, denominada *Package key*.

Después, se procederá a descargar e instalar la API, en este caso mediante el “Maven Repository”, aunque TapLinx también nos ofrece la posibilidad de instalarlo manualmente mediante un archivo AAR. Para ello, es necesario modificar el fichero *gradle* de nuestro proyecto, añadiendo el código de la Figura 4.1.

```
 repositories{
     flatDir{
         dirs 'libs'
     }
     maven{
         credentials{
             username "user"
             password "pass"
         }
url "http://maven.taplinx.nxp.com/nexus/content/repositories/taplinx/"
     }
 }
```

Figura 4.1 Código para añadir TapLinx vía Maven Repository

Es importante introducir un username y un password válidos, los mismos que la cuenta registrada en TapLinx.

Una vez hecho esto, se nos instalare de forma automática la API de TapLinx. Lo siguiente será registrar nuestra aplicación, lo que nos permitirá el uso sin límites de la librería, para ello es necesario añadir la siguiente línea en el fichero *main.java*, como se muestra en la Figura 4.2

```
m_libInstance = NxpNfcLib.getInstance();
try {
    m_libInstance.registerActivity(this, "Package key","Offline key");
} catch (Exception e) {
    e.printStackTrace();
}
```

Figura 4.2 Código usado para el registro de la aplicación

Este extracto es para un registro de la aplicación offline, por lo que no será necesaria una conexión a internet para registrar nuestra aplicación.

Estas líneas es recomendable introducirlas en un método, denominada en nuestro caso *initializeLibrary*, que se ejecutara dentro del método *onCreate*.

4.1.1 **Primeras pruebas**

Como primer paso, se trabajó con la Mifare Classic, ya que su nivel de complejidad es menor y permite un mejor acercamiento tanto a estas tecnologías como a la librería TapLinx.

El primer programa consistió en la lectura y escritura de datos en un bloque. Previamente a el desarrollo de esta actividad, fue necesario realizar una serie de cambios en el programa para permitir a la tecnología NFC del teléfono móvil reconocer la tarjeta.

Lo primero es activar el NFC del teléfono móvil, pues en caso contrario la aplicación dará error y se cerrara automáticamente, ya que todas las operaciones con la tarjeta precisan de esta tecnología. Una vez el NFC este activado, el teléfono comenzara a emitir un campo magnético a la espera de que se aproxime una tarjeta.

Cuando esto ocurra, el móvil puede reaccionar de diferentes maneras según el contenido de la tarjeta. Los eventos a los que el móvil puede responder son tres:

1. ACTION_NDEF_DISCOVERED: las tarjetas NFC pueden contener información en un formato estandarizado, llamado NDEF, que fue creado por el NFC Forum con el objetivo de poder almacenar identificadores de recursos (URIs) en la tarjeta y, al acercar la aplicación, se abriera una aplicación. Esta tecnología esta presenta en la librería Android por defecto, por lo que no seria necesario tener la librería TapLinx.
2. ACTION_TECH_DISCOVERED En el caso de que el lector detecte una tarjeta, pero esta no posea datos en formato NDEF el siguiente paso es intentar reconocer algunas tecnologías definidas en el paquete `android.nfc.tech`.
3. ACTION_TAG_DISCOVERED: Si no se detecta ninguna tarjeta que cumpla las condiciones anteriores, este es el evento por defecto.

Una vez conocido esto, es importante habilitarlas en nuestra aplicación, esto se realiza mediante configuraciones en el fichero AndroidManifest.xml, como muestra la Figura 4.3.

```
<intent-filter>
   <action android:name="android.intent.action.MAIN" />
   <action android:name="android.nfc.action.NDEF_DISCOVERED" />
   <action android:name="android.nfc.action.TAG_DISCOVERE" />
   <action android:name="android.nfc.action.TECH_DISCOVERED" />
   (…)
</intent-filter>
```

Figura 4.3 Código usado para habilitar los eventos

También es necesario dar permisos a la aplicación para poder leer/escribir de la tarjeta, nuevamente en el AndroidManifest.xml, como lo mostrado en la Figura 4.4:

```
<uses-permission android:name="android.permission.NFC" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission  android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
<uses-permission   android:name="android.permission.READ_EXTERNAL_STORAGE"
/>
```

Figura 4.4 Código empleado para habilitar los permisos

Posteriormente, se descarga e integra la librería TapLinx con el método ya explicado anteriormente, y con ella instalada, se procedió a la creación del primer programa.

Una vez escritos los datos en los bloques correspondientes, se procedió a su lectura y, finalmente, a la validación de los mismos.

Es importante destacar que tanto la lectura, como la escritura en una tarjeta MIFARE Classic ha de realizarse del bloque completo, es decir, 16 bytes. Si se quisiera escribir un tamaño menor, sería necesario añadir relleno de bloque, y si se requiere de un espacio mayor, habría que saltar a escribir en el siguiente bloque (sin contar los *sector trailers*).

Una vez se entendió el funcionamiento de la MIFARE Classic, se pasa a trabajar con la DESFire, que posee un nivel de complejidad mayor. Se creó al principio una aplicación con los ficheros anteriormente descritos, pero se usó la clave y protocolo triple DES (3DES), y su clave por defecto.

A continuación, se realizaron las medidas de verificación anteriormente descritas, para asegurarse protección frente a ataques.

Por último, destacar que se trabajó y estudio también una MIFARE de la clase Ultralight. Esta tarjeta debe escribirse en bloques de 4 bytes, y no permite escribir tamaños mayores, por lo que habría que dividir el mensaje en bloques de 4 bytes y grabar cada uno de estos en un bloque distinto. Al tratarse de una tarjeta MIFARE Ultralight, no posee clave de cifrado, por lo que el uso en esta aplicación está limitado por la falta de seguridad que presenta. Aun así, se ha creído conveniente escribir datos en ella, para entender mejor su funcionamiento.

Otra de las ventajas por la que se ha creído conveniente su estudio, es por existir un formato del tipo pegatina, que se puede adherir a superficies, como un teléfono móvil o una pulsera, lo que hace muy cómodo su uso.

4.2 Lista de comandos usados

En este apartado se enumerarán y explicarán brevemente la lista de comandos más importantes en la realización de este proyecto, pertenecientes a la librería TapLinx, tanto comandos generales, como propios de cada tarjeta.

4.2.1 Generales

```
CardType cardType = m_libInstance.getCardType(intent);
```

Extrae, del intent creado al acercar la tarjeta y leerla con la tecnología NFC, el tipo de tarjeta, valiéndose del SAK.

```
m_libInstance.registerActivity(this,"Package Key","Offline Key");
```

Registra nuestra aplicación en el servidor TapLinx, lo cual nos habilita el uso de la librería. En nuestro caso se ha usado un registro offline, por lo que el dispositivo no necesita de estar conectado a una red móvil o wifi para realizar la verificación, por ello es necesaria la offline key, proporcionada por TapLinx.

4.2.2 Mifare Classic

```
IMFClassic object =
    ClassicFactory.getInstance().getClassic(MifareClassic.get(tag));
```

Crea una instancia del tipo Mifare Classic, a partir del tag leído, para permitir ejecutar los comandos posteriores.

```
objClassic.authenticateSectorWithKeyA(int SectorIndex,byte[] key);
```

Autentica el sector seleccionado con la clave seleccionada. Si la autenticación resulta satisfactoria, permite operación I/O en ese sector.

```
byte[] data = objClassic.readBlock(int SectorIndex);
```

Permite leer los datos del sector seleccionado, si se ha realizado la autenticación permitente.

```
objClassic.writeBlock(int SectorIndex, Byte [] data);
```

Nos permite escribir en el sector seleccionado los bytes seleccionados.

4.2.3 Mifare DESFire

```
IDESFireEV1 object =
    DESFireFactory.getInstance().
        getDESFire(m_libInstance.getCustomModules());
```

Crea una instancia del tipo Mifare DESFire, para permitir ejecutar los comandos posteriores.

```
objDESFireEV1.selectApplication(int);
```

Sirve para seleccionar el ID de la aplicación sobre la que se desea trabajar.

```
objDESFireEV1.authenticate(int cardKeyNumber,
                           IDESFireEV1.AuthType authenticationType,
                           KeyType keyType,
                           IKeyData key);
```

Nos permite autenticarnos contra la tarjeta, usando la clave con la que se ha programado cada aplicación, siendo la clave por defecto la del triple DES.

```
byte[] data = objDESFireEV1.readData(int file, int offset, int length);
```

Lee los datos pertenecientes al ID del fichero que queramos perteneciente a la aplicación en la que nos encontremos.

```
objDESFireEV1.writeData(int file, int offset, byte[] data);
```

Escribe los datos contenidos en data, en el fichero seleccionado.

4.2.4 Mifare Ultralight

```
IUltralight ultralightBase =
    UltralightFactory.getInstance().
        getUltralight(m_libInstance.getCustomModules());
```

Crea una instancia del tipo Mifare Ultralight, sobre la que ejecutar los posteriores comandos.

```
byte[] data = ultralightBase.read(Int block);
```

Lee los datos del bloque indicado.

4.3 Ventajas de la librería TapLinx

Como ya se ha comentado, el uso de la librería TapLinx facilita enormemente la comunicación entre el terminal y la tarjeta, ya que simplifica los comandos usados para ello.

Todos los comandos de bajo nivel, son simplificados de tal manera que cualquier persona sin conocimientos previos en tarjetas inteligentes sea capaz de trabajar con ellas. Android posee una librería con los comandos necesarios para la Mifare Classic, pero para el resto de tarjetas, es necesario es uso de la librería TapLinx.

Para el caso de la DESFire, todo se complica más aun, ya que en el caso de poseer los datos en un mensaje del tipo NDEF, Android lo detectara automáticamente, pero si no es el caso, es necesario usar comandos más complejos mediante la clase IsoDep [36], enviando los comandos convertidos en cadenas de bytes, y recibiendo la respuesta también en bytes. Esto conlleva un conocimiento del funcionamiento de cada tarjeta, para poder enviar correctamente los comandos necesarios.

Gracias a la librería TapLinx, comunicarse con la tarjeta se reduce a una serie de comandos fáciles de entender, como es el siguiente ejemplo:

```
int[] app_Ids = objDESFireEV1.getApplicationIDs();
```

Este comando es usado para obtener los ID de las aplicaciones en una DESFire EV1.

Pese a sus numerosas ventajas, TapLinx posee también algunas desventajas frente a la librería nativa de Android, como es el tener que registrarse tanto en su página web, como registrar cada aplicación que se cree, así como no poder acceder a ciertos documentos si no se firma un contrato de confidencialidad.

5 aplicación de control de acceso a un concierto

Para asentar todos los conocimientos sobre tarjetas de la familia MIFARE, se ha diseñado una aplicación de ejemplo que trabaje sobre ellas, cuyo objetivo es familiarizarse con el uso en un entorno real de las mismas.

5.1 Requisitos

Para poder utilizar esta aplicación, cada usuario debe llevar consigo una tarjeta de la familia MIFARE, acorde al tipo de entrada que se haya adquirido. La aplicación será usada por el personal de seguridad del concierto, necesitando simplemente un teléfono Android con tecnología NFC, necesitando este teléfono una versión superior a Android 4.4 *KitKat*.

En cuanto al funcionamiento, permite, tras acercar la entrada al terminal, detectar automáticamente si la entrada es válida y su tipo. Esto es especialmente útil en nuestro caso, ya que, si es un proceso sencillo y rápido para el personal, las colas para entrar se reducirán, facilitando al cliente la espera.

En la Figura 5.1 se muestra el esquema de funcionamiento del programa.

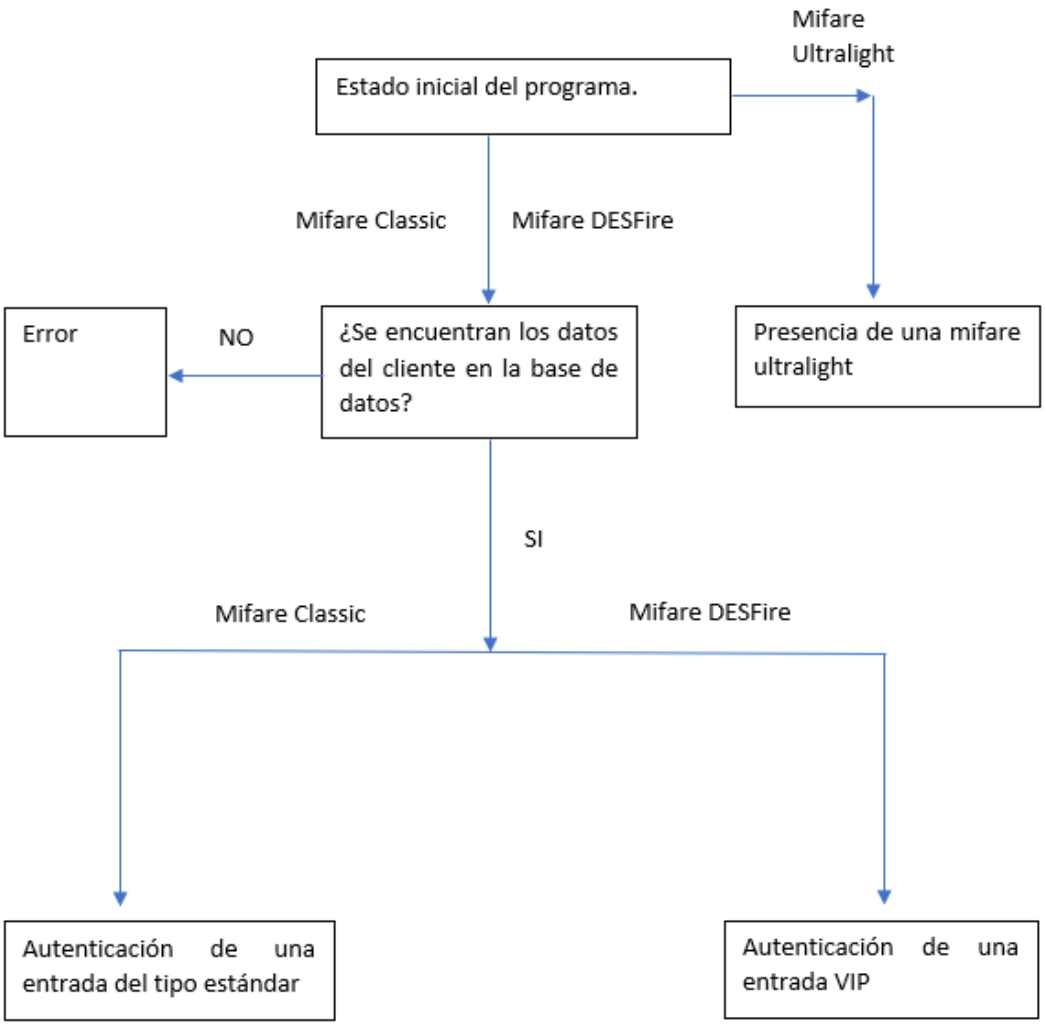


Figura 5.1 Esquema de funcionamiento del programa.

5.2 Formateo de las tarjetas

Para llevar a cabo el control de acceso, cada tarjeta debe disponer de unos datos para validarlos, siendo estos datos tanto el nombre como ambos apellidos, y el DNI. Estos datos se han escrito en cada tarjeta de una manera distinta, adecuándose a la estructura de memoria de las mismas.

5.2.1 MIFARE Ultralight

Como ya se ha comentado, esta tarjeta no posee ningún tipo de seguridad, por lo que simplemente se ha guardado el DNI del dueño, para aplicarlo en caso de que se quiera verificar la identidad del usuario.

Este DNI se ha escrito comenzando en el bloque 0x04, que es el primero dedicado al usuario, y se ha terminado en el bloque 0x06.

5.2.2 MIFARE Classic

En el caso de la Mifare Classic, se han asignado los siguientes bloques predeterminados, descritos en la Tabla 5.1:

Bloque	Contenido
0x04	Nombre
0x05	Primer apellido
0x06	Segundo apellido
0x07	Sector trailer (no es posible la escritura)
0x08	DNI

Tabla 5.1 Formato de los datos en MIFARE Classic

Una vez escritos los bloques, es necesario leerlos en el dispositivo móvil para la validación de estas tarjetas. Para poder leer es necesario, como ya se ha explicado, autenticarse usando las claves que posee la tarjeta. Una vez hecho esto, y con los datos almacenados, se ha creído conveniente verificar que esa entrada en cuestión es auténtica, y posee los datos de un comprador real, por lo que el DNI leído del bloque 0x08 es comparado con el DNI oficial en la base de datos del programa, realizando así la verificación y evitando falsas entradas.

5.2.3 MIFARE DESFire

En el caso de las tarjetas MIFARE DESFire, es necesario un nivel extra de verificación al tratarse de una entrada del tipo VIP, por lo que se ha introducido un link a una imagen del usuario, verificando así la identidad del dueño.

Como todas las tarjetas MIFARE DESFire, estas contienen una aplicación principal, llamada aplicación master, que se representa con el AID (“Application ID”) 0x00. Para todos los usos relacionados con la tarjeta, es necesario primero seleccionar esta aplicación, y autenticarse contra ella, mediante la clave triple DES configurada en el programa. El objetivo de la aplicación es permitir al lector ejecutar comandos a nivel de tarjeta (e.g. crear nuevas aplicaciones, leer los datos de estas, etc.).

Una vez configurada la aplicación maestra, se ha creado una nueva aplicación cuyo ID es 0x130000. Una vez creada la aplicación, es necesario seleccionarla y autenticarse nuevamente, pero la clave en este caso es una clave AES128 generada en el programa. En esta aplicación se han creado dos ficheros, guardando en cada uno los datos necesarios, como muestra la Tabla 5.2:

FILE ID	Contenido
0x01	Nombre, apellidos y DNI.
0x02	Enlace a la foto del propietario.

Tabla 5.2 Formato de los datos en MIFARE DESFire

Se ha enviado el texto en plano, ya que está protegido por la clave AES, y no se requerirá seguridad adicional, ya que se disponen de varios mecanismos de seguridad en el propio programa.

También se podría haber seleccionado comunicación cifrada, para evitar escuchas al canal, aunque para su estudio es necesario acceder a documentación protegida por TapLinx, y sería necesario poseer una dirección comercial y firmar un contrato de confidencialidad, cosa que no ha sido posible en este proyecto.

Para la validación de la entrada en este caso, se trata de comparar los valores leídos del fichero 0x01 con los almacenados el programa.

5.3 Menú principal

Lo primero que nos encontramos al iniciar la aplicación, es un texto pidiéndonos por favor, acercar nuestra entrada al teléfono móvil, como se observa en la Figura 5.2. Esto permite a la aplicación diferenciar las entradas usadas, de una manera simple. Una vez se acerque la tarjeta al móvil, este ejecutará una activity dependiendo del tipo de tarjeta que sea.

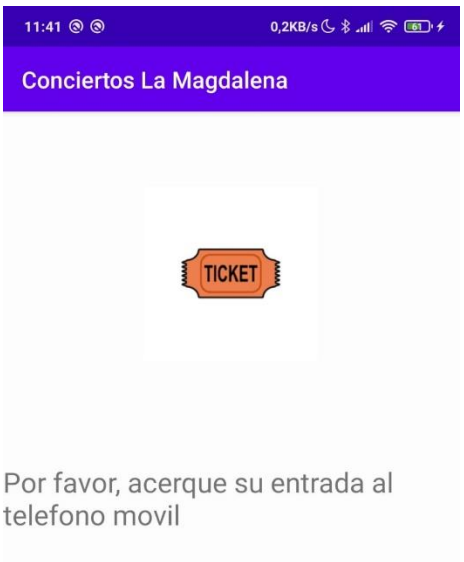


Figura 5.2 Menú principal de la aplicación

Las activities del sistema se explican a continuación:

- MainActivity: esta actividad, se encarga de recoger los datos de la tarjeta, procedentes de un intent del sistema, y distinguir el tipo de tarjeta de la que se trata, para actuar en consecuencia. De cualquiera de las 3 tarjetas posibles, esta actividad se encarga también de leer los datos y de validarlos. En caso de ser correctas se enviarán a una de las siguientes activities.
- Classic: se encarga de representar los datos incluidos dentro de la tarjeta MIFARE Classic correspondiente.
- Desfire: se encarga de representar los datos de la tarjeta MIFARE DESFire, así como de proporcionar el hipervínculo a la imagen de validación de la tarjeta.

5.4 Entrada estándar

Como ya se ha comentado, al acercar una tarjeta del tipo MIFARE Classic al lector, este reacciona transmitiendo los datos leídos de la tarjeta, a la activity Classic.java, que se encarga de proporcionar la interfaz necesaria para la visualización de los datos, quedando como lo que se observa en la Figura 5.3

Figura 5.3 Validación entrada estándar.

Para autenticar cada uno de los sectores que contienen los datos, se ha usado el siguiente comando:

```
objClassic.authenticateSectorWithKeyA(objClassic.blockToSector(i),
MifareClassic.KEY_DEFAULT);
```

Siendo i el número de sector a autenticar, y KEY_DEFAULT la clave por defecto de la tarjeta MIFARE Classic, siendo la siguiente, 0xFFFFFFFFFFFF.

Posteriormente se han leído los datos, y se han almacenado.

5.5 Entrada VIP

Del mismo modo que con la tarjeta MIFARE Classic, con la tarjeta MIFARE DESFire los datos leídos de la tarjeta, son enviados a Desfire.java, para gestionar la visualización y representación de los mismos:

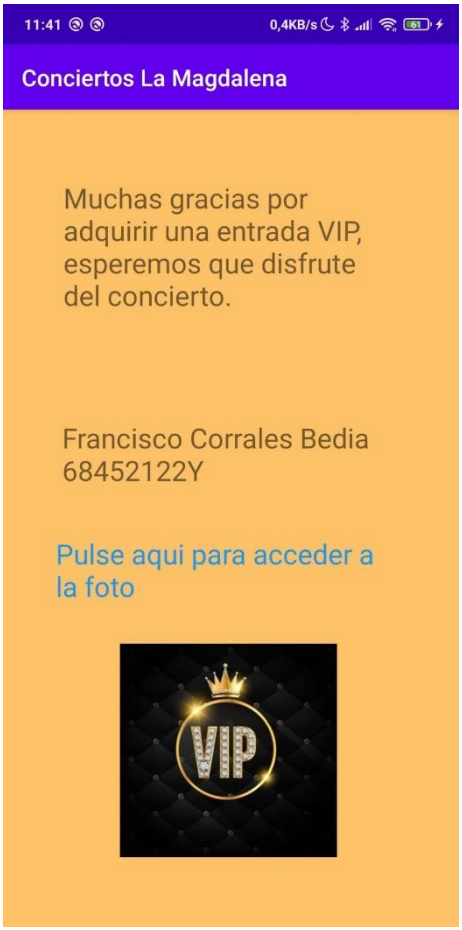


Figura 5.4 Validación entrada VIP.

Esta tarjeta presenta, como se puede observar en la Figura 5.4, un link a una imagen del dueño de la tarjeta, para una seguridad adicional. También se ha usado para la tarjeta DESFire, una clave AES128, que se ha creado en el programa, por lo que no es la clave por defecto. La clave usada es la siguiente:

0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF.

Para poder cambiarla, se ha empleado el comando:

```
objDESFireEV1.changeKey(0, KeyType.AES128, DEFAULT_KEY_2kDES, NEW_KEY_AES, (byte) 0).
```

Una vez hecho esto, la clave de la aplicación queda cambiada. Después para autenticarse se usa el siguiente comando:

```
objDESFireEV1.authenticate(0, IDESFireEV1.AuthType.Native, KeyType.AES128, defKeyData2);
```

Como se puede observar, la clave usada es del tipo AES128, para mayor seguridad.

5.6 Uso de la tarjeta MIFARE Ultralight

Como ya se ha comentado, esta tarjeta al no poseer ningún tipo de clave de protección de los datos, únicamente se ha empleado para guardar un DNI y mostrarlo por pantalla.

El comando usado es el siguiente:

```
byte[] leido1 = ultralightBase.read(i);
```

Al no poseer clave, no es necesaria una autenticación.

Se podría valorar la inclusión de esta tarjeta en forma de pulsera, para la identificación de los asistentes.

5.7 Error en la validación

Si se produce un error en la validación de la tarjeta, se emitirá un mensaje de “ERROR”, y se le denegará el permiso a acceder al recinto. Este error puede deberse a que la tarjeta presentada no tiene los datos necesarios para la validación, a que la autenticación ha fallado y los datos no coinciden, o a que la tarjeta no es de la familia MIFARE con la que se trabaja en este proyecto.

Figura 5.5 Error en la validación de la entrada.

6 Conclusiones y líneas futuras

El uso de estas tarjetas esta hoy día extendido por todo el mundo, en multitud de ámbitos, por lo que su estudio es de gran interés, ya que nos permite entender multitud de aplicaciones que van desde el control de acceso (enfoque de este proyecto), hasta métodos de pago usado con tarjetas contactless. Este proyecto se ha centrado tanto en temas de estudio del funcionamiento de las mismas, así como de elaborar una aplicación sencilla que trabaje con varios tipos de tarjetas de la familia MIFARE.

La librería TapLinx es una librería amigable con el usuario, ofreciendo una serie de comandos fáciles de entender para trabajar con las tarjetas. No se ha estudiado en toda su extensión, ya que no se dispone de la documentación necesaria, al estar protegida esta por un documento de confidencialidad. Esto limita el trabajo para tareas que requieran de un mayor nivel de seguridad para desarrolladores independientes, para solventarlo sería necesario poseer una dirección comercial válida y firmar el contrato.

Tanto a nivel de aplicación como de desarrollo, se ha conseguido una aplicación funcional y que cumple los requisitos de seguridad pensados, pero también es destacable que es mejorable en varios aspectos, por lo que me gustaría plantearlos para su estudio en futuros proyectos.

El primero es, aunque no sea una parte fundamental del proyecto, es altamente mejorable la presentación de la aplicación, ya que ayudaría a los usuarios a entender mejor la aplicación, y hace que se sientan más cómodos, priorizando nuestra aplicación frente a otras opciones.

Otra mejora que me gustaría proponer, sería el uso de servidores remotos, para interactuar de manera dinámica con la tarjeta, lo que permitirá el uso de estas tarjetas para varios conciertos, o para otras funciones más variadas.

Por último, en tema de seguridad, al usar los servidores remotos ya comentados, es posible usar el teléfono móvil como una simple pasarela entre los servidores y el lector, ya que un teléfono móvil no es el mejor elemento donde almacenar información sensible, como lo son las claves criptográficas. La idea principal seria que, una vez el servidor se autentique contra la tarjeta, existiera un flujo de información cifrada entre ambas partes, por lo que ningún elemento fuera del sistema podría intervenir en el mismo.

Bibliografía

[1] Organization for Standardization and International Electrotechnical Commission, ISO/IEC 14443-3:2011 Identification cards Contactless integrated circuit cards Proximity cards

[2] International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 7816-1 – Identification Cards – Integrated circuit(s) cards with contacts – Part 1: Physical characteristics, 2011.

[3] International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 7816-2 – Identification Cards – Integrated circuit(s) cards with contacts – Part 2: Dimensions and location of the contacts, 2007.

[4] [International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 7816-3 – Identification Cards – Integrated circuit(s) cards with contacts – Part 3: Electrical interface and transmission protocols, 2006.

[5] International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 7816-4 – Identification Cards – Integrated circuit(s) cards with contacts – Part 4: Organization, security and commands for interchange, 2013.

[6] International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 9798-2:2019 IT Security techniques — Entity authentication — Part 2: Mechanisms using authenticated encryption.

[7] International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 7810:2003 Identification cards – Physical characteristics.

[8] International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 15457-1:2008 Identification cards – Thin flexible cards – Part 1: Physical characteristics.

[9] International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 15693:2006 Identification cards – Contactless integrated circuit cards – vicinity cards,

[10] Digital Security Group, Radboud University Nijmegen (2008): [en línea] http://www2.ru.nl/media/english.html [última visita 14 Octubre de 2020]

[11] TNO report 34643, Security Analysis of the Dutch OV-Chipkaart® (2008) [en línea] https://files.gendo.ch/TNO_ICT_-_Security_Analysis_OV-Chipkaart_-_public_report.pdf [ultima visita 14 Octubre de 2020]

[12] MF1SEP(H)10x1 MIFARE PLUS SE - Secure contactless smart card IC for seamless migration Rev. 3.1 – 16 March 2017. Product short data sheet.

[13] MF0UN(H)00 MIFARE Ultralight Rev. 3.1 — 7 September 2016. Product data sheet.

[14] MF0ICU2 MIFARE Ultralight C - Contactless ticket IC Rev. 3.3 — 30 July 2019 Product data sheet.

[15] MF1S50YYX_V1 MIFARE Classic EV1 1K - Mainstream contactless smart card IC for fast and easy solution development. Rev. 3.2 — 23 May 2018. Product data sheet.
[16] MF3ICDx21_41_81 MIFARE DESFire EV1 contactless multi-application IC Rev. 3.2 — 9 December 2015. Product short data sheet.
[17] MF3D(H)x2 MIFARE DESFire EV2 contactless multi-application IC Rev. 3.2 — 12 June 2019. Product short data sheet.
[18] Wikipedia, La enciclopedia Libre, tarjetas sin contacto – contactless Smart cards. [en línea] https://es.qaz.wiki/wiki/Contactless_smart_card#History. [última visita 13 de Octubre de 2020].
[19] El país.es -- Roland Moreno, creador de la tarjeta inteligente (2009), [en línea] https://elpais.com/tecnologia/2012/05/01/actualidad/1335905381_416813.html. [última visita 17 Octubre de 2020].
[20] Mifare – TapLinx oficial site, [en linea] https://mifare.net/es/productos/tools/taplinx/ [última visita 10 Octubre de 2020].
[21] Smart card Handbook 4th edition, Wolfgang Rankl, Junio 2010.
[22] Hacking Mifare Classic cards, Márcio Almeida (marcioalma@gmail.com) [en línea] https://www.blackhat.com/docs/sp-14/materials/arsenal/sp-14-Almeida-Hacking-MIFARE-Classic-Cards-Slides.pdf [ultima visita 16 Octubre de 2020].
[23] CardExchange ® producer help, specifying key values, [en linea], https://help.cardexchangesolutions.com/en/step_2_desfire_specifying_key_values.htm [ultima visita 17 Octubre de 2020].
[24] Colorid.com, David Stallsmith, Should we wait for MIFARE DESFire EV2 cards? [en línea] https://www.colorid.com/ev1-vs-ev2.html.
[25] AN11876 Starting development with TapLinx SDK, Rev. 2.0 – 1 September 2020
[26] AN10834 MIFARE ISO/IEC 14443 PICC Selection, Rev. 4.1 – 20 March 2020.
[27] AN10833 MIFARE Type Identification Procedure, Rev. 3.6 – 11 July 2016.
[28] A3m.eu MIFARE: la tecnología, los productos, su uso y su precio [en linea] https://a3m.eu/es/mifare-la-tecnologia-los-productos-su-uso-y-precio [ultima visita 19 Octubre de 2020].
[29] Texas Instruments, Ralph Jacobi and Josh Wyatt, MIFARE DESFire EV1 AES Authentication With TRF7970A, [en línea] https://www.ti.com/lit/an/sloa213/sloa213.pdf?ts=1602002295618 [ultima visita 17 Octubre de 2020].
[30] INFOSEC, Pierluigi Paganini (2013), Near Field Communication (NFC) Technology. Vulnerabilities and Principal Attack Schema [en línea] https://resources.infosecinstitute.com/near-field-communication-nfc-technology-vulnerabilities-and-principal-attack-schema/ [ultima visita 13 Octubre de 2020].
[31] Wikipedia La enciclopedia libre, RFID [en línea] https://es.wikipedia.org/wiki/RFID#Arquitectura [ultima visita 17 de Octubre de 2020].
[32] FQIngenieria, tipos de chips MIFARE, 5 de Marzo de 2015 [en línea] https://www.fqingenieria.com/es/conocimiento/tipos-de-chips-mifare-52 [ultima visita 18 de Octubre de 2020].
[33] SecurityArtwork Hacking RFID, rompiendo la seguridad de MIFARE (I), Roberto Amado (29 de Enero de 2010) [en línea]

https://www.securityartwork.es/2010/01/29/hacking-rfid-rompiendo-la-seguridad-de-mifare-i/ [ultima visita 19 de Octubre de 2020].
[34] TapLinx main site, Java documentation [en línea] https://www.mifare.net/developer/javadoc/android/ [ultima visita 19 de Octubre de 2020].
[35] NUOPlanet, Algoritmos de encriptación (27 de enero de 2017) [en linea] https://nuoplanet.com/blog/algoritmos-encriptacion-aes-3des/ [ultima visita 19 de Octubre de 2020].
[36] Android developers, IsoDep class documentation, [en línea] https://developer.android.com/reference/android/nfc/tech/IsoDep [ultima visita 19 de Octubre de 2020]