UNIVERSIDAD DE VALLADOLID

ESCUELA TÉCNICA SUPERIOR

DE INGENIEROS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

EVALUACIÓN DE LA SEGURIDAD DE TARJETAS CRIPTOGRÁFICAS

AUTOR: Marta Barrios Barrigón

TUTOR: Juan Carlos García Escartín

22 de octubre de 2018

TITULO:	Evaluación de la Seguridad de Tarjetas Criptográficas
AUTOR:	Marta Barrios Barrigón
TUTOR:	Juan Carlos García Escartín
DEPARTAMENTO:	Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática

Miembros del Tribunal

PRESIDENTE:	Pedro Chamorro Posada
SECRETARIO:	Julio Sánchez Curto
VOCAL:	María Jesús González Morales
SUPLENTE 1:	Juan Carlos García Escartín
SUPLENTE 2:	Ramón de la Rosa Steinz

FECHA DE LECTURA: 22 de octubre de 2018

CALIFICACIÓN:

RESUMEN DEL TRABAJO

El DNI (Documento Nacional de Identidad) es un documento necesario e imprescindible para acreditar la identidad y la nacionalidad española del titular que figura en él. Con el avance tecnológico, la forma de utilizar el DNI se ha visto modificada a lo largo de los años. Actualmente se utiliza el DNI electrónico 3.0, el objeto de estudio de esta memoria.

El presente proyecto estudia la seguridad del DNIe3.0 analizando su resistencia frente a ataques de fuerza bruta utilizando la tecnología NFC y estudiando las características de los números aleatorios generados en la comunicación.

ABSTRACT

The DNI (National Identity Document) is a necessary and essential document to prove the identity and the Spanish nationality of the holder. As technology improves, the way of using the DNI has been modified over the years. Nowadays the electronic DNI 3.0 is used, the object of study of this report.

This project studies the security of the DNIe3.0 analising its resistance to brute force attacks using the NFC technology, and studying the characteristics of the random numbers generated in the communication.

PALABRAS CLAVE

DNIe3.0, NFC (*Near Field Communication*), Ataque de fuerza bruta, Números aleatorios

KEYWORDS

DNIe3.0, NFC, Brute Force Attack, Random numbers

Agradecimientos

En primer lugar, me gustaría agradecer a mi tutor, Juan Carlos, su paciencia y ayuda cuando lo he necesitado y por mostrarse siempre cercano e interesado por mi trabajo.

En segundo y último lugar, gracias a todas aquellas personas que de una manera u otra me han acompañado durante todos estos años, en especial a mi familia, a Alberto y a Noé.

ÍNDICE

Introducción

El Documento Nacional de Identidad (o DNI) aparece en 1951 como documento para la identificación de los ciudadanos españoles. Estaba formado por una foto facial y los datos personales del propietario del carné. Con el paso de los años, las prestaciones del DNI han aumentado, guiadas por el avance tecnológico y permitiendo a las personas utilizarlo para muchos otros fines.

La mejora más significativa que se introdujo fue la capacidad de usar el DNI de forma electrónica para poder utilizarlo como documento de acreditación en operaciones por Internet. A esto hay que añadir la tecnología NFC que implementa el chip dual de la última versión del DNI, mediante la cual se puede utilizar el DNI acercándolo a otro dispositivo con dicha tecnología.

Esta evolución del DNI ha supuesto una mejora en la calidad de vida de las personas, ya que ahora mismo muchos trámites se pueden realizar desde el domicilio particular sin necesidad de presentarse físicamente en la oficina correspondiente, lo que supone un ahorro de tiempo, personal y recursos, tanto para el interesado como para las diferentes entidades implicadas.

Sin embargo, como cualquier avance tecnológico tiene también una parte negativa que habrá que analizar y solventar, o al menos mitigar. Las formas de cometer delitos han evolucionado en cuanto a recursos, conocimiento y sofisticación. Por ello, es fundamental trabajar en la seguridad y fiabilidad del DNI para reaccionar de manera óptima ante ataques como el robo de identidad y así dar a la ciudadanía mayor seguridad y confianza en las transacciones electrónicas, muchas veces infrautilizadas por miedo o desconocimiento. Extender el uso de herramientas como el DNI electrónico o la tarjeta de crédito para compras online es clave a la hora de que la

sociedad participe de la nueva "Economía Digital" y así poder beneficiarse de todas las ventajas que la revolución tecnológica trae consigo.

Propósito y objetivos

El propósito principal es estudiar la seguridad del Documento Nacional de Identidad analizando la última versión del mismo.

Como objetivo, se pretende comprobar si la Fábrica Nacional de Moneda y Timbre (FNMT) ha corregido un error, que afectaba a la privacidad de los datos de los ciudadanos españoles, observado por un alumno de la Universidad de Zaragoza en su Trabajo Fin de Grado [Bal]. También se pretende analizar la calidad de los números aleatorios que generan tanto el móvil como la tarjeta durante la comunicación.

Estructura del proyecto

Teniendo presentes los objetivos enumerados previamente, el trabajo se ha estructurado en cuatro partes fundamentales.

La primera parte, correspondiente a los capítulos primero y segundo, pretende ser una puesta en situación. En primer lugar se trata de dar una visión general sobre la generación de números aleatorios y el análisis de los mismos. Por su lado, el segundo capítulo se basa en la explicación de unos conceptos básicos fundamentales para entender el resto del trabajo.

En la segunda parte, correspondiente al capítulo tercero, se estudia y analiza en profundidad el Documento Nacional de Identidad electrónico 3.0. Esto incluye sus características físicas, los elementos de seguridad que implementa y los protocolos que se usan para comunicarse con él.

La tercera parte, capítulos cuarto y quinto, se centra en el análisis de toda la experimentación realizada en el laboratorio. En el capítulo cuarto se estudian varias aplicaciones modificadas e instaladas en un dispositivo móvil mediante el uso de `Android Studio`; mientras que en el capítulo quinto se estudian los números generados por el móvil y el DNIe3.0, con y sin interferencias.

Por último, el trabajo finaliza con las conclusiones que se han alcanzado, con el planteamiento de posibles líneas de estudio de futuro y con la bibliografía utilizada. También se adjunta una carpeta con las diferentes pruebas realizadas, tanto los números generados como los resultados de los tests obtenidos.

Capítulo Primero. Introducción a la generación de números aleatorios.

En este capítulo se pretende realizar una aproximación básica a la generación de números aleatorios. En concreto, se analizan dos generadores de números aleatorios y varios tests que sirven para comprobar la verdadera aleatoriedad de los mismos, así como la definición de números aleatorios y números pseudoaleatorios. Esta pequeña introducción a la generación de números aleatorios se realiza debido a la importancia de los mismos en las comunicaciones entre distintos dispositivos, principalmente en el campo de la seguridad y la criptografía. Además, los números aleatorios son fundamentales para generar claves seguras y evitar diferentes ataques en comunicaciones privadas.

1.1. ¿Qué son los números aleatorios?

Los números aleatorios nacen como resultado de una variable al azar y son generados a partir de fuentes de aleatoriedad. Una secuencia aleatoria se puede explicar como el resultado de lanzar una moneda al aire. En cada tirada, el resultado es impredecible (cara o cruz, 1 ó 0 si hablamos de números binarios) y los resultados de tiradas consecutivas son independientes, es decir, el resultado de una tirada no afecta a lanzamientos posteriores [RSNSB].

Por otro lado, las secuencias formadas por números pseudoaleatorios se caracterizan por no mostrar, desde un punto de vista estadístico, un patrón aparente, pero su generación se basa en un algoritmo determinista, es decir, mismas condiciones iniciales arrojan resultados iguales.

En este trabajo, vamos a generar y a analizar tanto números aleatorios como pseudoaleatorios. Para comprobar las diferencias existentes entre estos dos tipos de números se han realizado dos ejemplos prácticos.

1.2. Generadores

Se han utilizado dos generadores (RNG, *Random Number Generator*):

- QUANTIS: se trata de un generador de números aleatorios que utiliza la física cuántica para la generación de los mismos. Proporciona entropía desde el principio siendo idóneo para la creación de claves seguras, a diferencia de otros generadores que necesitan generar entropía desde fuentes externas antes de ser capaces de producir una primera clave segura. Además, genera números aleatorios a tasas de hasta 16 Mbps [IDQ].

- URANDOM: en los sistemas operativos tipo Unix podemos encontrar dos archivos que generan números aleatorios o pseudoaletorios, siendo éstos `/dev/random` y `/dev/urandom`. Ambos se ejecutan directamente desde la consola de comandos. El archivo `/dev/random` produce números aleatorios de alta calidad, mientras que `/dev/urandom` genera números pseudoaleatorios. No se recomienda el uso de /dev/random ya que es lento. En cuanto a `/dev/urandom`, si se modifica la semilla que utiliza, se consigue unos buenos resultados.

1.3. Tests

El propósito principal del uso de diversos tests para el estudio de la fiabilidad y eficacia de dichos generadores reside en poder encontrar indicios de aleatoriedad en una secuencia bajo estudio, ya que no se puede demostrar que algo es, o no, aleatorio. Además, permiten comprobar que no hay fallos evidentes que permitan predecir la secuencia analizada. A continuación se analizan los generadores descritos previamente mediante dos conjuntos de test: `Ent` y `Dieharder`.

- **Ent:** se trata de un test que se puede utilizar directamente desde la consola de comandos en un sistema operativo de tipo UNIX [Wal]. Estudia diversos parámetros:

  1. Entropía.- Es una medida de desorden. Un buen generador maneja una entropía en sus datos generados muy próxima a 8, que es el valor máximo posible (este valor guarda relación con el hecho de que los datos se tratan byte a byte).

  2. Compresión óptima.- Indica el nivel de compresión que permite el archivo estudiado. Para una entropía máxima, su valor ideal es del 0%.

  3. Distribución "^{x} cuadrado".- También conocida como distribución de Pearson. Viene dada mediante un valor numérico y un porcentaje, indicando la frecuencia en la que una secuencia verdaderamente aleatoria debería exceder al calculado. En función del porcentaje podemos decir:

     a. 0% - 1% y 99% - 100% → Secuencia casi con certeza que no es aleatoria.

     b. 1% - 5% y 95% - 99% → Secuencia probablemente no aleatoria.

     c. Resto → Secuencia puede ser no aleatoria.

  4. Media aritmética.- Se obtiene como la suma de todos los valores dividido entre el número de valores. Mejor cuanto más cerca esté de 127.5.

  5. Valor de Pi (Montecarlo).- Determina el valor de pi calculado mediante el método Montecarlo. A grandes rasgos, este método lo que hace es, suponiendo que tiramos dardos en un cuadrado que contiene un círculo, contabilizar el número de dardos que caen en el círculo.

  6. Coeficiente de correlación serial.- Mide la relación lineal existente entre dos variables aleatorias.

- **Dieharder:** conjunto de tests de números pseudoaleatorios que se utiliza interactivamente en la consola de comandos y de forma similar tanto en sistemas operativos Windows como en UNIX, pudiendo elegir el número de tests que se desea aplicar al fichero de números aleatorios en cuestión. En concreto, se ha trabajado con Linux. Además, alguna versión del mismo permite al usuario añadir nuevos tests [BEB].

Tras el análisis de estos tests, se ha llegado a varias conclusiones:

1. Test con 1G de datos generados con QUANTIS y analizado con `Ent`: se observa un buen funcionamiento del generador ya que en todas ellas los distintos tests arrojan buenos resultados. Un ejemplo de esto es el valor de la entropía, siempre en su máximo posible, 8 (trabaja con bytes), o el valor de la distribución "X cuadrado", cuyo porcentaje se encuentra siempre en el rango en el que la secuencia evaluada puede ser no aleatoria (en vez de en los rangos de "seguro" o "casi seguro" de que la secuencia es no aleatoria).

2. Test con 1.1G de datos generados con URANDOM analizado con `Dieharder`: este test informa del resultado de cada prueba realizada mediante diferentes evaluaciones: PASSED, WEAK, GOOD, SUSPECT, DO NOT USE… En este análisis, todas varían entre PASSED y WEAK, siendo un resultado bueno ya que PASSED significa que se ha superado el test correspondiente. El hecho de que algunas de las pruebas arrojen el resultado WEAK puede deberse a una cantidad insuficiente de números generados. Cabe destacar que si se analizan diferentes secuencias obtenidas por un mismo generador y el resultado WEAK aparece en diferentes pruebas, eso es sinónimo de buen funcionamiento del generador, mientras que un mal resultado sería que apareciera siempre en la misma prueba.

3. Test con 1M de datos generados con QUANTIS analizado con `Ent`: al tener un menor volumen de datos, los resultados de ambos tests son peores.

Capítulo Segundo.

Conceptos básicos.

En este capítulo se pretende dar una visión general sobre la historia del DNI, la interfaz NFC y el ataque de fuerza bruta. La razón de este capítulo reside en que estos conceptos serán claves a lo largo del presente trabajo. Comenzaremos explicando uno a uno cada concepto y después analizaremos la relación existente entre los mismos.

2.1. Evolución del DNI

Desde que fue emitido por la Dirección General de la Policía (Ministerio del Interior), el Documento Nacional de Identidad (DNI) ha ido evolucionando junto con las innovaciones tecnológicas de cada momento. Inicialmente, se utilizaba únicamente de manera física y acreditaba la identidad y la nacionalidad española del titular que figuraba en él. Con los avances tecnológicos se ha incorporado la capacidad de verificar los datos personales de cada ciudadano de forma electrónica así como la capacidad de realizar firmas digitales con la misma validez jurídica que la firma manuscrita [Nac].

La Ley 59/2003, de 19 de diciembre, de firma electrónica, atribuye al Documento Nacional de Identidad estas nuevas funcionalidades electrónicas. En añadido, el Real Decreto 1553/2005, de 23 de diciembre, por el que se regula la expedición del Documento Nacional de Identidad y sus certificados de firma electrónica, establece diversas pautas en cuanto a la expedición del DNI electrónico, sus características y su utilización. Entre ellas cabe destacar que los certificados electrónicos, con

independencia de la validez del Documento Nacional de Identidad, tienen un período de vigencia máximo de cinco años, pudiendo solicitarse nuevos certificados sin necesidad de cambiar la tarjeta del DNI siempre y cuando ésta siga vigente [BOE].

La primera versión del Documento Nacional de Identidad electrónico (DNIe) aparece en enero de 2006. La principal diferencia con su predecesor se encontraba en la incorporación de un chip, es decir, un circuito integrado que permitía guardar y procesar de forma segura la información del titular [Nac].

A continuación, la Figura 1.1 muestra la primera versión del DNI electrónico, de las mismas dimensiones que el DNI tradicional y con la incorporación del chip mencionado previamente.

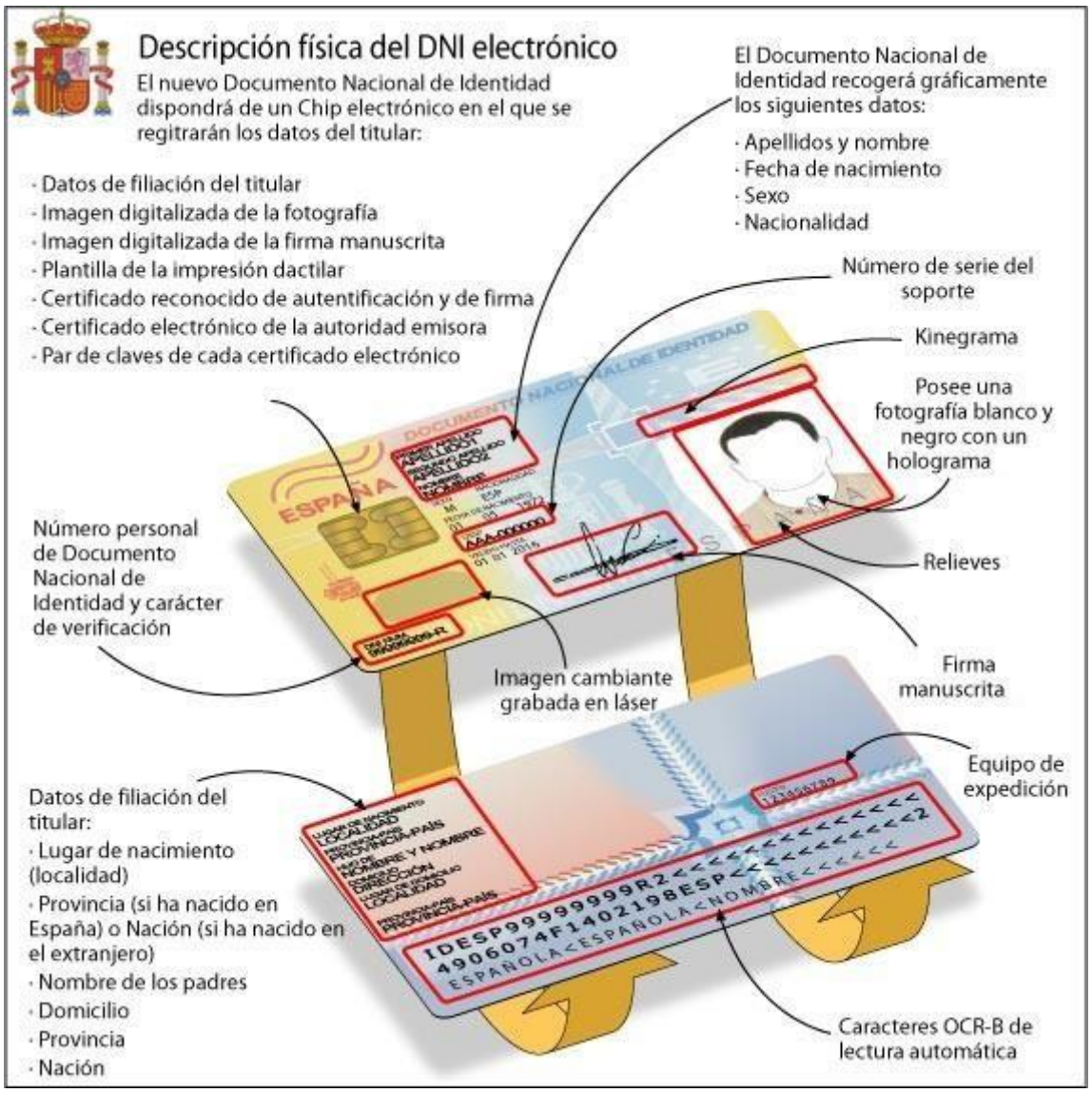

Figura 1.1 DNI electrónico [Nac]

Este primer DNIe obligaba a todo aquel que quisiera utilizar su funcionalidad electrónica a obtener un dispositivo hardware e instalar los drivers necesarios para que fuera capaz de leer los certificados existentes en el chip y conectarse a los diferentes servicios digitales.

Con el objetivo de mejorar las prestaciones del DNIe, en 2015 la Dirección General de la Policía comenzó a implementar el DNIe3.0, una versión actualizada del DNI electrónico cuya principal novedad era la incorporación de un chip con una interfaz dual, permitiendo tanto la conexión mediante un dispositivo hardware como de forma inalámbrica haciendo uso de la tecnología NFC.

En la Figura 1.2 se muestra la descripción del DNIe3.0. Las características físicas del DNIe3.0, documento que utilizaremos en nuestro estudio, así como otros aspectos relevantes del mismo, se explican en el siguiente capítulo.

La principal diferencia entre la primera versión del DNI electrónico y el DNIe3.0 se encuentra en que con este último no se necesita poseer un lector de tarjetas, únicamente es necesario disponer de un teléfono *Smartphone* o *tablet* con tecnología NFC y la aplicación del servicio que se quiere utilizar. Para hacer uso de la funcionalidad inalámbrica, basta con acercar el DNIe3.0 a la antena NFC del dispositivo a una distancia igual o inferior a 1 cm.

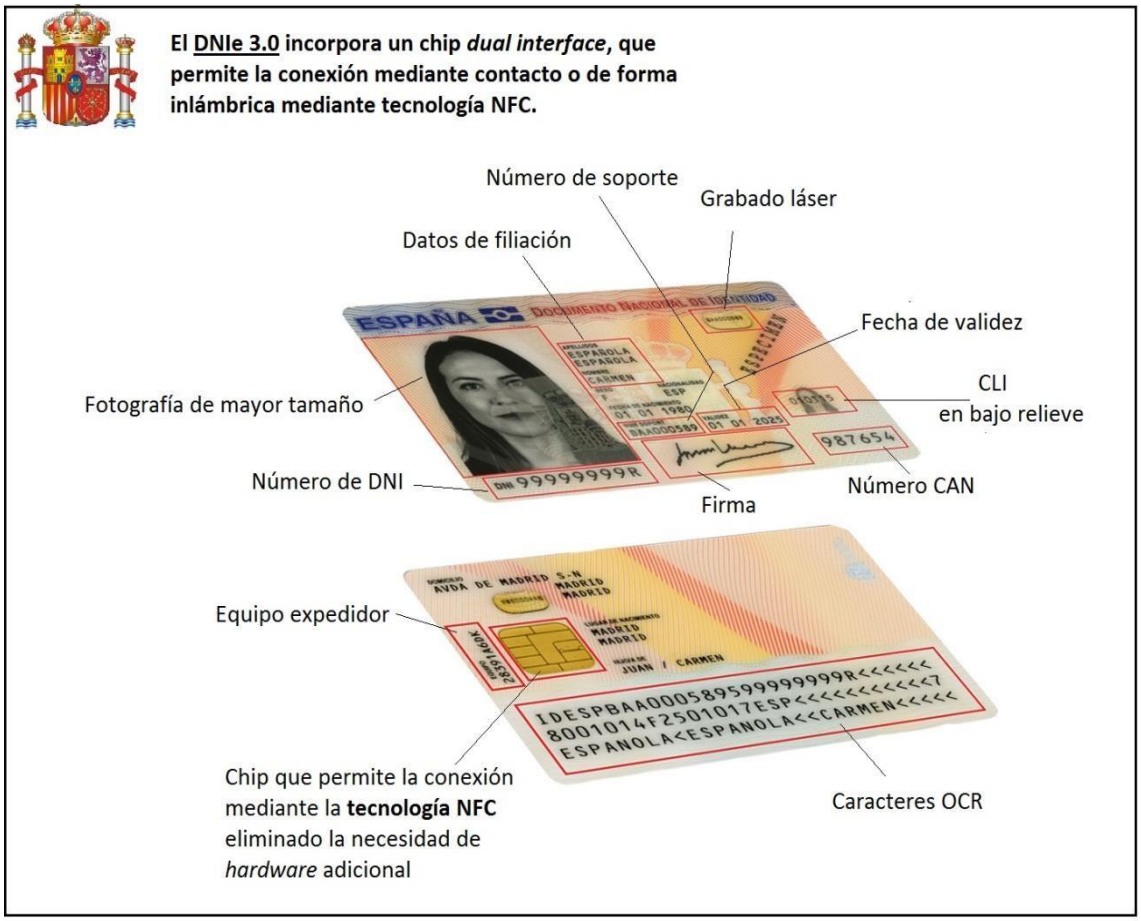

Figura 1.2 DNIe3.0 [Nac]

Actualmente coexisten ambas versiones, aunque desde diciembre de 2015 únicamente se puede obtener el DNIe3.0 en todas las Oficinas de Expedición del territorio nacional.

La implementación del DNIe3.0 ha supuesto una gran mejora en la relación entre los ciudadanos y las Administraciones Públicas (Comunidades Autónomas, Entidades Locales y Administración General del Estado) [Nac]:

1. Tal y como se recoge en el artículo 16.2 de la Ley 59/2003 de Firma Electrónica: "La Administración General del Estado empleará, en la medida de lo posible, sistemas que garanticen la compatibilidad de los instrumentos de firma electrónica incluidos en el documento nacional de identidad electrónico con los distintos dispositivos y productos de firma electrónica generalmente aceptados".

2. El acceso a los datos de un ciudadano se podrá realizar directamente a partir de los datos contenidos en el chip utilizando algoritmos que no requieran PIN, lo que implicará un menor número de errores de transcripción.

3. Los trámites con la Administración Pública se tornarán más eficientes ya que no hará falta la presencia física para garantizar la identidad del ciudadano.

2.2. Tecnología NFC

La tecnología NFC (*Near Field Communication*) es una tecnología de corto alcance basada en identificación por radio frecuencia (RFID, *Radio Frequency Identification*). La organización encargada de la regulación de esta tecnología se crea en 2004 bajo el nombre de “NFC-Forum”. A pesar de que inicialmente esta organización estaba formada únicamente por las multinacionales Nokia, Philips y Sony, en la actualidad otras empresas se han unido a esta asociación. “NFC-Forum” nace con el objetivo de promover el uso y la implementación de la tecnología NFC en la electrónica de consumo, en los dispositivos móviles y en los PCs [NFC].

La familia ISO/IEC 14443 es el estándar RFID utilizado en Europa y EEUU. Dicho estándar distingue dos tipos de tecnología, A y B, siendo la segunda la utilizada por el chip NFC del DNIe3.0, y está formado por cuatro partes [IOS]:

1. ISO/IEC 14443-1: Características físicas.

2. ISO/IEC 14443-2: Potencia de radiofrecuencia e interfaz de señal.

3. ISO/IEC 14443-3: Inicialización y anticolisión.

4. ISO/IEC 14443-4: Protocolo de transmisión.

Una de las características principales de la tecnología NFC es que no se requiere ningún tipo de licencia para su uso ya que trabaja en la banda libre de 13,56 MHz. Además, se puede distinguir entre dos modos de operación en función de una propiedad de los dispositivos que desean establecer una conexión: modo activo y modo pasivo. En el modo pasivo, sólo uno de los dispositivos es activo, es decir, tiene una fuente de alimentación propia y es el encargado de generar una señal electromagnética. El otro dispositivo es pasivo, por lo que utiliza la energía del campo magnético incidente para alimentar su propio circuito (esto es lo que se conoce como acoplamiento inductivo). Por otro lado, en el modo activo, los dos dispositivos poseen una fuente de alimentación propia, generando ambos su propio campo magnético [DMG].

Este trabajo se basa en el establecimiento de una conexión en modo pasivo, siendo un dispositivo móvil el elemento activo y el DNIe3.0 el elemento pasivo.

2.3. Ataque de fuerza bruta

Los ataques de fuerza bruta se utilizan para conseguir entrar en un sistema protegido con una contraseña. Este ataque prueba todas las posibles combinaciones posibles de caracteres para obtener la contraseña y avisa en cada intento de si se ha tenido éxito o no. A veces se usa junto con ataques de diccionario, esto es, se prueba con palabras de diccionario en vez de con caracteres aleatorios, lo que disminuye el número de combinaciones posibles y en consecuencia aumenta la probabilidad de éxito.

En este trabajo vamos a realizar un ataque de fuerza bruta mediante la utilización del protocolo PACE (*Password Authentication Connection Establishment*), aunque el DNIe3.0 también utiliza el protocolo BAC (*Basic Access Control*). Ambos protocolos serán estudiados más adelante.

Capítulo Tercero.

Análisis del DNIe3.0.

En este capítulo se pretende analizar detalladamente tanto las características físicas como los elementos de seguridad implementados en el DNIe3.0. Tras ello se estudiarán las ideas más relevantes de la interfaz NFC, en particular hablaremos de los protocolos PACE y BAC.

3.1. Características físicas

El DNIe3.0 tiene unas dimensiones de 85.60 mm de ancho y 53.98 mm de alto, las mismas que una tarjeta de crédito. Está hecho de policarbonato y contiene diversa información del usuario que lo posee [Nac].

- **ANVERSO**

  Tal y como se puede observar en la Figura 3.1, el anverso del DNIe3.0 está formado por:

  - **Datos de filiación del ciudadano.-** En orden comenzando por arriba: apellidos, nombre, sexo (M, hombres y F, mujeres), nacionalidad (ESP, española) y fecha de nacimiento (en formato dd mm aaaa).
  - **Fotografía.-** Foto en blanco y negro del ciudadano.

- **Imagen fantasma.-** Foto del ciudadano en pequeño. Los números que aparecen indican la fecha de expedición del documento (en formato dd mm aa).
- **Número DNI letra identificación fiscal.-** Número de identificación fiscal (NIF) del ciudadano. Formado por ocho dígitos y una letra obtenida a partir de los mismos.
- **Número de soporte.-** Formado por tres letras y seis dígitos.
- **Firma.-** Firma manuscrita del ciudadano.
- **Fecha de validez.-** Indica la fecha hasta la cual el documento tiene validez.
- **Número CAN.-** Número de acceso a la tarjeta (*Card Access Number*).

➢ **REVERSO**

En esta parte del documento se pueden observar los siguientes datos (Figura 3.2):

- **Caracteres OCR.-** Zona para ser leída por máquinas y que está formada por un conjunto de letras, números y caracteres.

  1. Tipo de documento [ID]
  2. País emisor del documento [ESP]
  3. Número de serie del soporte físico de la tarjeta [BAA000589]
  4. Primer dígito de control, relativo al número de serie [5]
  5. Número del DNI con letra incluida [99999999R]
  6. Relleno [<<<<<<]
  7. Fecha de nacimiento en formato aa mm dd [800101]
  8. Segundo dígito de control, relacionado con la fecha de nacimiento [4]
  9. Sexo del titular [F]
  10. Fecha de caducidad del documento en formato aa mm dd [250101]
  11. Tercer dígito de control, relacionado con la fecha de caducidad del documento [7]
  12. Nacionalidad del titular [ESP]
  13. Relleno [<<<<<<<<<<<]
  14. Cuarto dígito de control, relacionado con el número de serie y su dígito de control, el número de DNI, la fecha de nacimiento y su dígito de control, y la fecha de caducidad y su dígito de control [7]
  15. Nombre y apellidos del titular en el siguiente orden: PRIMER_APELLIDO<SEGUNDO_APELLIDO<<NOMBRE [ESPAÑOLA<ESPAÑOLA<<CARMEN]
  16. Relleno [<<<<<]
- **Datos del ciudadano.-** Domicilio, lugar de nacimiento (ciudad y provincia) y nombres de los padres.
- **Equipo.-** Número del equipo de expedición del documento (código de la comisaría donde fue expedido).

Figura 3.1. Anverso del DNIe3.0 [Nac]

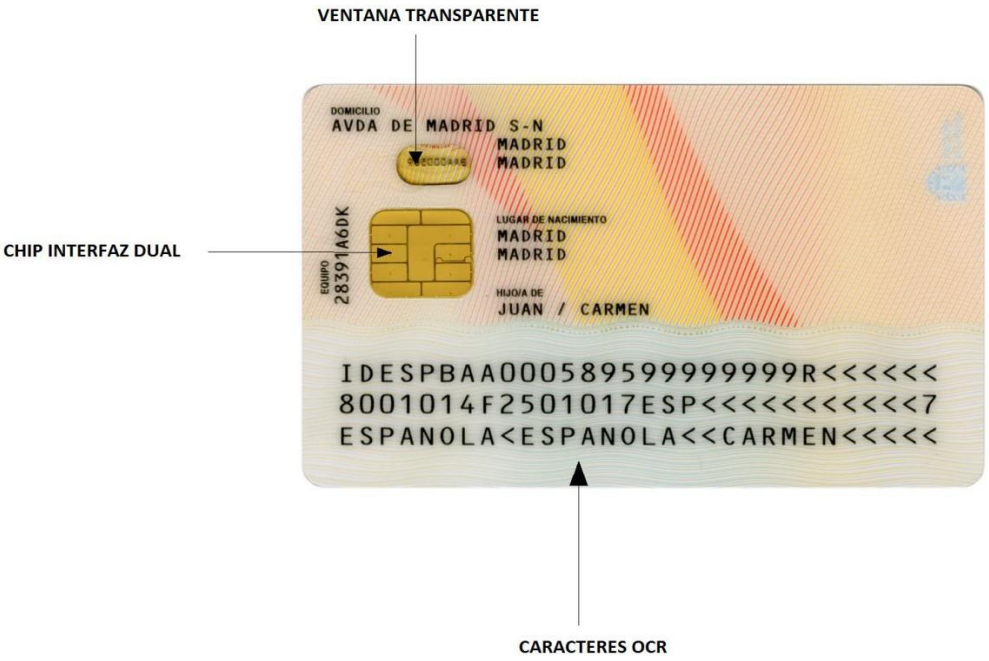

Figura 3.2. Reverso del DNIe3.0 [Nac]

3.2. Elementos de seguridad

El DNIe3.0 implementa nuevos elementos de seguridad además de los ya existentes en el documento anterior [Nac]. Todos ellos se pueden observar en la Figura 3.3:

- **Kinegrama.**- Se utiliza como protección anti-fotocopia con imágenes difractivas.
- **Tinta óptica variable.-** Cambio de color según el ángulo de visión.
- **Ventana transparente con grabado láser**
- **CAN.-** *Card Access Number*, clave numérica compuesta por seis dígitos.
- **CLI bajo relieve.-** CLI, *Changeable Laser Image*. Formada por una imagen más pequeña del usuario con la fecha de expedición del documento por encima.
- **Embosados efecto mate.-** Franjas oblicuas que modifican la impresión del documento.
- **Embosados en alto relieve.-** Otorga relieve a diversas zonas del documento.
- **3 tintas UV en iris.-** Visibles únicamente con luz ultravioleta.
- **Tinta oasis.-** Tinta opaca que oculta las capas inferiores.
- **Chip de interfaz dual.-** Se utiliza para el almacenamiento y procesamiento de datos. La información que contiene se divide en dos grupos en función del nivel y las condiciones de acceso: zona pública, lectura sin restricciones; y zona de seguridad, lectura accesible en los Puntos de Actualización del DNIe.

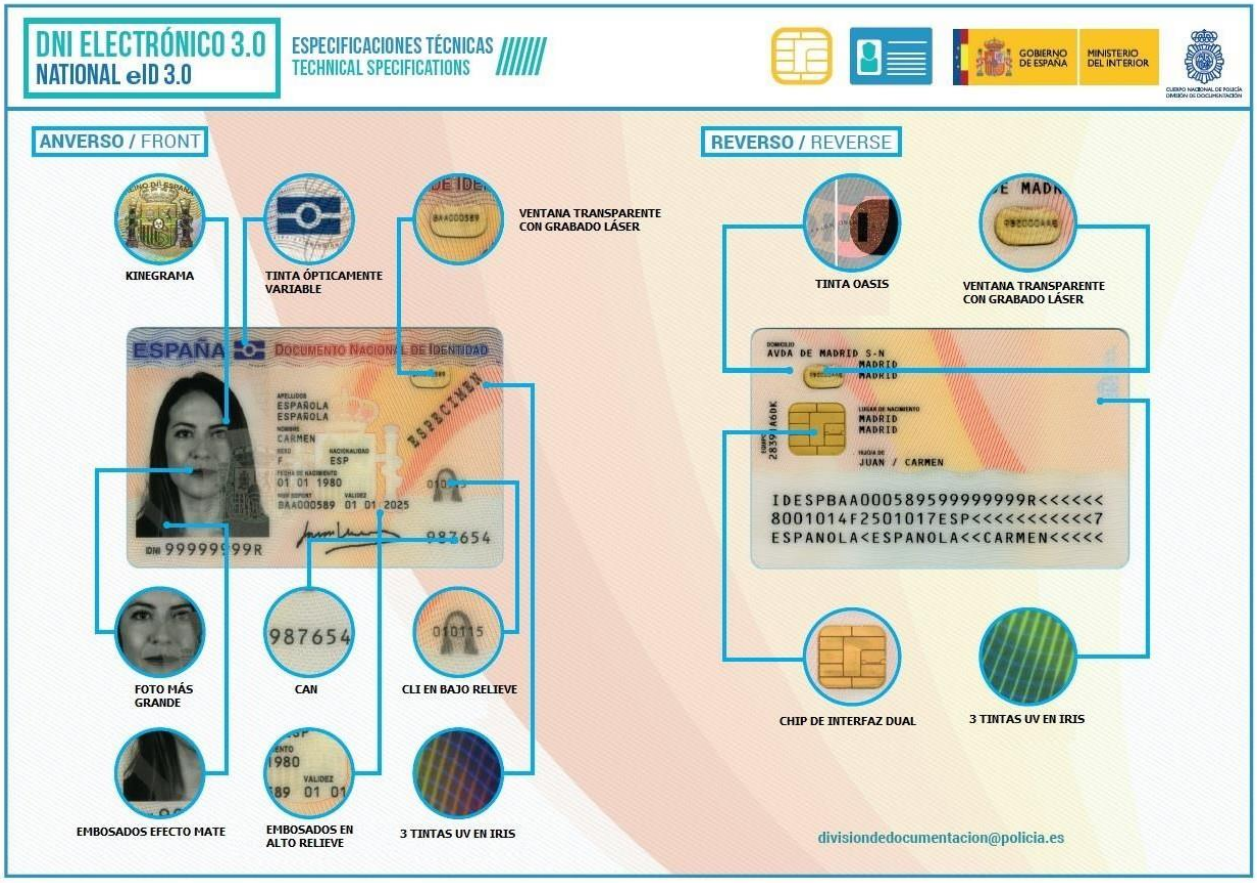

Figura 3.3. Elementos de seguridad del DNIe3.0 [Nac]

3.3. Interfaz NFC

El chip de interfaz dual que aparece en el reverso del DNIe3.0 permite acceder al mismo mediante un dispositivo con tecnología NFC (móvil o *tablet*, por ejemplo). Anteriormente, para poder llevar acabo esta conexión era necesario el uso de un lector de tarjetas inteligentes y un ordenador. Gracias a esto se pueden realizar diversas tareas con el DNIe3.0 de forma más rápida y sencilla.

Para conectarse al DNIe3.0 vía NFC se hace uso de dos protocolos mencionados en el capítulo anterior, BAC y PACE. A continuación, vamos a explicar cada uno detalladamente.

3.3.1. BAC (*Basic Access Control*)

Protocolo diseñado para la seguridad de datos menos sensibles y basado en la utilización de una clave simétrica para decidir la autenticación (técnica disponible en aquellos chips compatibles con ISO 1443). Utiliza la fecha

de nacimiento del usuario, la fecha de validez del documento y el código MRZ situado en la zona inferior del reverso del DNI para generar las claves de sesión.

El uso de criptografía simétrica implica que las claves vienen determinadas por la entropía del código MRZ, estimada en 73 bits como máximo. Una alternativa que mejoraría la defensa frente a posibles ataques al chip sería poder ralentizar la ejecución de este protocolo para que el atacante necesitara estar en contacto con el documento durante más tiempo. Para mejorar esta pequeña debilidad, y alguna otra más, aparece el protocolo PACE, cuya principal característica reside en el uso de criptografía asimétrica. En este protocolo la entropía inicial se usa para establecer una clave de sesión temporal de mayor entropía [BK].

3.3.2. PACE (*Password Authentication Connection Establishment*)

Protocolo publicado en el año 2007 por la Oficina Federal Alemana para la Seguridad de la Información (BSI) que incluye cuatro pasos [BK]:

1. El chip elige un número de forma aleatoria, encriptándolo con una clave derivada de la contraseña y enviándolo al terminal para que sea recuperado.

2. Se utiliza una función de mapeo asignando el número aleatorio a los diversos parámetros de la criptografía asimétrica.

3. Se emplea un protocolo Diffie-Hellman basado en los parámetros generados en el paso anterior.

4. El chip y el terminal intercambian y confirman los *tokens* de autenticación, confirmando de esta manera las claves de sesión derivadas.

- Protocolo Diffie-Hellman

Protocolo de establecimiento de claves entre pares, sin contacto previo y de forma anónima, utilizando un canal público. Para llevar a cabo dicho protocolo, ambas partes de la comunicación deben elegir una clave privada y una clave pública (común a ambas pero distinta en cada conexión). Tras ello y mediante la utilización de dichas claves en diversas fórmulas matemáticas, ambos extremos se intercambian los resultados obtenidos de forma pública para establecer una clave de sesión [DH]. Este procedimiento aparece explicado en la Figura 3.4 donde,

A → clave privada elegida por Alice

B → clave privada elegida por Bob

p,g → forman la clave pública común a ambos

a,b → resultado que se intercambia de forma pública, obtenido a partir de la clave pública y las claves privadas

Diffie-Hellman Key Exchange

Step	Alice	Bob
1	Parameters: p, g	
2	$A = \text{random}()$ $a = g^A \pmod p$	$\text{random}() = B$ $g^B \pmod p = b$
3	$a \longrightarrow$ $\longleftarrow b$	
4	$K = g^{BA} \pmod p = b^A \pmod p$	$a^B \pmod p = g^{AB} \pmod p = K$
5	$\longleftarrow E_K(data) \longrightarrow$	

Figura 3.4. Explicación matemática protocolo Diffie-Hellman

Para explicar el funcionamiento de este protocolo, vamos a poner un ejemplo numérico.

Elegimos los valores iniciales:

p = 3 q = 4 A = 10 B = 17

Obtenemos ahora a y b, números intercambiados de manera pública:

a = (mod 3) = 1

b = (mod 3) = 2

Comprobamos cómo se obtiene el mismo número secreto K:

K = (mod 3) = 1

$$K = \quad (\mathrm{mod}\ 3) = 1$$

Hay que destacar que la característica más importante de este protocolo es que si una tercera parte está escuchando la conversación le resultará complicado obtener el número secreto de cada extremo (A y B) a partir del número intercambiado (a y b), por lo que no podrá obtener el número secreto (K) que se utilizará para la encriptación de la comunicación. Este ejemplo maneja números pequeños, pero si se utilizan grandes números un ordenador no podría obtener la clave secreta de alguna de las partes en un tiempo razonable.

Capítulo Cuarto.

Experimentación. Aplicaciones móviles.

En este capítulo se pretende analizar tres pruebas realizadas con el DNIe3.0. Una de ellas se centra en la realización de un ataque de fuerza bruta al número CAN del DNIe3.0. Las otras dos se basan en el estudio de dos aplicaciones obtenidas a partir de la página web de la Dirección General de la Policía Nacional. También se comentarán todos los dispositivos y programas que se han utilizado para la realización de las diferentes pruebas.

4.1. Entorno de trabajo

Se ha contado con los medios disponibles en el laboratorio 2L021, correspondiente al Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática de la Universidad de Valladolid.

Se ha utilizado un dispositivo móvil Xiaomi MI5 con capacidad NFC, versión de Android 7.0 NRD90M, una memoria RAM de 3 GB y un procesador Quad-core Max 2.15GH; un ordenador Lenovo con sistema operativo Windows 8.1 de 64 bits y un DNIe3.0 [Xia][Len].

El principal motivo que nos ha llevado al uso de Android ha sido la existencia de dos proyectos de código abierto basados en el establecimiento del protocolo PACE mediante NFC. Además, actualmente Android lidera la cuota de

mercado, por lo que si alguien intenta realizar algún ataque al DNIe3.0 hay una mayor probabilidad de que utilice un dispositivo Android en vez de cualquier otro.

El dispositivo móvil se ha utilizado para la instalación y ejecución de las tres aplicaciones móviles que se han analizado: DNIe3.0 brute force v2, Android_DGPApp_LECTURA y Android_DNIeApp_FIRMA, que explicaremos más adelante.

Por otra parte, el ordenador ha permitido modificar las aplicaciones mencionadas previamente, así como analizar los resultados a partir de los datos recabados. Para realizar estas tareas se ha trabajado, principalmente, con dos programas:

- Android Studio: entorno de desarrollo integrado (IDE) oficial para Android. Está basado en el software IntelliJ IDEA de JetBrains y ha reemplazado a Eclipse como el IDE oficial para el desarrollo de aplicaciones Android [Stu].

- MATLAB: software matemático utilizado para el tratamiento de los resultados obtenidos en las diferentes pruebas, principalmente para la realización de gráficas.

4.2. DNIe3.0 brute force v2

Para la implementación de un ataque de fuerza bruta al código CAN que aparece en el anverso del DNIe3.0 se ha utilizado una aplicación Android llamada DNIe3.0 brute force v2 [Rod], basada en Androsmex, un proyecto accesible desde GitHub [And].

El funcionamiento de la aplicación es el siguiente: a partir de un número introducido (0 por defecto), se prueba si éste coincide con el número CAN del DNIe3.0 que se está utilizando. Si no coincide, el número se incrementa en uno y se vuelve a hacer la comprobación. Así, hasta que se encuentra el número CAN deseado y se puede obtener la información del titular del DNIe3.0 que se está atacando. El número CAN está formado por seis dígitos, lo que supone un total de posibles combinaciones. Por cada intento fallido, la aplicación informa del tiempo transcurrido en dicho intento.

En un Trabajo de Fin de Grado previo, realizado por Víctor Sánchez Ballabiga y tutorizado por Ricardo J. Rodríguez, de la Escuela de Ingeniería y Arquitectura de la Universidad de Zaragoza y con fecha septiembre de 2016, ya se analizó el

comportamiento del DNIe3.0 frente a ataques al número CAN. Observaron que el tiempo entre cada intento era siempre el mismo, aproximadamente unos 1.5 segundos. Esto permitía que un atacante pudiera "robar" la identidad de otra persona en un tiempo máximo de 17 días [Bal] [Rod].

Con vistas a mejorar la seguridad del DNIe3.0, ambos autores comunicaron a la Fábrica Nacional de Moneda y Timbre (FNMT) ese fallo de seguridad, aconsejando la introducción de un retardo mayor en cada nuevo intento. Debido a esto, el objetivo de esta parte del trabajo se basa en determinar si se ha llevado a cabo la modificación sugerida y, si la respuesta es afirmativa, analizar el nuevo comportamiento del DNIe3.0 frente a un ataque de fuerza bruta al número CAN.

Durante la ejecución del ataque se ha comprobado que el código de la aplicación no tenía en cuenta aquellos números CAN que empezaban por 0. Para solucionar este problema, se han añadido diversas líneas al código. Básicamente, mediante sentencias `if`, `else if` se comprueba el tamaño del número CAN. Si éste no es igual a seis, se añaden ceros por la izquierda hasta obtener un número formado por seis dígitos. El código que se muestra en la Figura 4.1 se ha añadido en dos partes del proyecto: una para establecer el número por el que se empezará el ataque y otra para el autoincremento en uno del número CAN a cada intento.

```
if (canNumber.length()==1)
    canNumber = "00000" + canNumber;

else if (canNumber.length()==2)
    canNumber = "0000" + canNumber;

else if (canNumber.length()==3)
    canNumber = "000" + canNumber;

else if (canNumber.length()==4)
    canNumber = "00" + canNumber;

else if (canNumber.length()==5)
    canNumber = "0" + canNumber;
```

Figura 4.1. Código añadido a `MainActivity.java`

Nuestro objetivo inicial era comprobar qué ocurría con el tiempo entre intentos. Para ello, se ha lanzado el ataque diversas veces sobre el DNIe3.0 y se han guardado los tiempos en un fichero de texto externo. Se ha observado que el comportamiento del documento frente a este tipo de ataque es siempre el mismo. A continuación, en la Figura 4.2 se muestra este comportamiento, obtenido mediante el promedio de siete pruebas. Esta gráfica se ha realizado con `MATLAB` a partir de un promediado de siete pruebas y representa la evolución del tiempo con respecto al número de intento, eje de ordenadas y eje de abscisas, respectivamente.

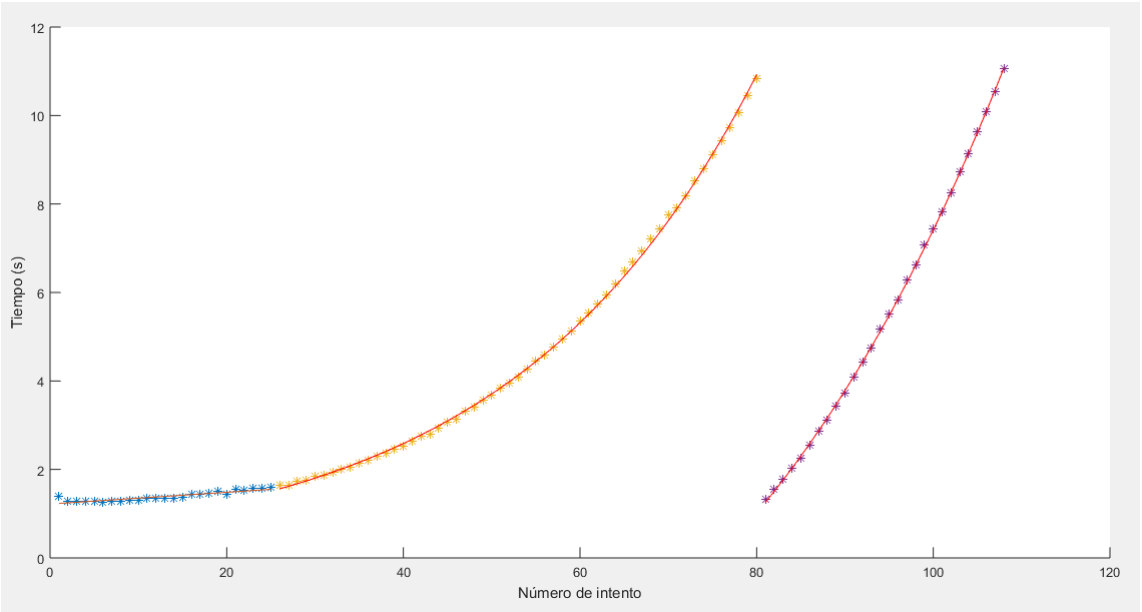

Figura 4.2 Gráfica tiempo/intento promedio de 7 pruebas de ataque de fuerza bruta

Para la realización de la gráfica se han utilizado algunas funciones interesantes de `MATLAB` [MAT]:

- `polyfit(x,y,n)`: permite el ajuste de los datos de las variables `x` e `y` con una curva polinómica de grado `n`. Devuelve los coeficientes del polinomio. En este caso en particular, se ha conseguido un buen ajuste con una curva polinómica de grado 1. Los coeficientes y del polinomio ( ) obtenidos han sido:

= 0.0133

= 1.2185

- `fit(x,y,tipo)`: ajusta los datos de las variables `x` e `y` con el modelo especificado en `tipo`. Para conseguir el mejor ajuste se han utilizado dos modelos: 'exp1' y 'exp2', tramo amarillo y morado de la gráfica, respectivamente. Coeficientes obtenidos:

```
General model Exp1:
f1(x) = a*exp(b*x)
Coefficients (with 95% confidence bounds):
a =      0.6128  (0.6028, 0.6227)
b =     0.03601  (0.03577, 0.03624)
```

```
General model Exp2:
f2(x) = a*exp(b*x) + c*exp(d*x)
Coefficients (with 95% confidence bounds):
a =  -1.624e+05  (-7.493e+12, 7.493e+12)
b =     0.01486  (-9.714, 9.743)
c =   1.624e+05  (-7.493e+12, 7.493e+12)
d =     0.01486  (-9.713, 9.742)
```

- `polyval(f,x)`: da como resultado el valor de un polinomio de grado `n` evaluado en `x`, donde `f` es un vector de longitud `n+1` cuyos componentes son los coeficientes, en orden descendiente, del polinomio a evaluar (`f` es el polinomio obtenido con la función `polyfit`).

Tal y como se puede observar en la Figura 4.2, se ha introducido un retardo por cada nuevo intento. El comportamiento que presenta el DNIe3.0 frente a este tipo de ataques es el siguiente: inicialmente se puede observar un tramo lineal (color azul), hasta los veinticuatro intentos aproximadamente; tras ello, los tiempos obtenidos aumentan de forma exponencial hasta llegar a los 10-11 segundos (color amarillo); inmediatamente después el tiempo vuelve a empezar desde 1 segundo aproximadamente, pero esta vez aumenta más rápidamente, y una vez alcanzados los 10-11 segundos, se pierde la conexión NFC entre el dispositivo móvil y el DNIe 3.0 (color morado).

Como se ha comentado previamente, antes de introducir el retardo el tiempo para poder comprobar todos los números CAN posibles se estimaba en 17 días. Ante esta nueva situación, se ha calculado el tiempo como promedio de las 7 pruebas realizadas. El resultado ha sido de 457.9367 segundos por cada prueba. Si tenemos en cuenta que cada prueba analiza 108 números distintos, el tiempo estimado para probar todos los números posibles ( ) es de 49 días aproximadamente, bastante superior al tiempo obtenido antes de introducir el retardo. Además, hay que tener en cuenta que en el caso de los 17 días, el dispositivo móvil no perdía la conexión NFC y podía comprobar todos los números de seguido, mientras que con el retardo, en el intento número 108 se pierde la conexión con el DNIe3.0, siendo necesario apartar el dispositivo móvil y volver a acercarlo en cada prueba indicando el nuevo número de inicio del ataque. Por ello, si lo vemos de una forma un poco más realista y suponemos que la acción de quitar-acercar el móvil y escribir el nuevo número de inicio cada vez que se pierda la conexión NFC supone unos 8 segundos, el tiempo estimado sería de unos 50 días. Todo esto supone una mejora considerable con respecto a la situación anterior.

Hay que destacar que el DNIe3.0 siempre tiene el mismo mecanismo de defensa, independientemente de si los números probados son más cercanos o no al número CAN del DNI utilizado en las pruebas. En concreto, se ha comprobado haciendo pruebas con los tres primeros dígitos del CAN fijos, variando los tres últimos. Esto supone un buen mecanismo de defensa frente a posibles ataques de fuerza bruta al código de acceso solicitado por el protocolo PACE.

4.3. Android_DGPApp_LECTURA

Esta aplicación pertenece a la Policía Nacional y se puede descargar el código de la misma de forma gratuita desde su página web [Nac]. También se encuentra disponible en la plataforma `Google Play Store` bajo el nombre "Ejemplo DNIe Lectura Datos". En la Figura 4.3 se pueden observar algunas capturas de la interfaz gráfica, las cuales, de izquierda a derecha y de arriba abajo muestran:

1. Pantalla de inicio.

2. "Opciones de Configuración", que permiten al usuario elegir qué información desea obtener del documento.

3. "Selección del CAN". En esta pantalla se muestran los números CAN utilizados previamente y se da la opción tanto de elegir un número CAN nuevo (círculo rosa) como la de modificar la información que se desea extraer (círculo amarillo).

4. "Datos de Identificación". Se muestran todos los datos obtenidos de un usuario.

Tras realizar diversas pruebas con distintos documentos se ha observado que la aplicación no funcionaba correctamente con aquellos DNI cuyo número CAN empezaba con un cero, ya que tras utilizarlo una vez, almacenaba el número de forma errónea y no se podía reutilizar el número CAN, teniendo que introducirlo manualmente cada vez.

Para solucionar este problema se ha modificado, haciendo uso de `Android Studio`, el código fuente de la aplicación, disponible en el portal web de la Dirección General de la Policía Nacional [Nac]. En la Figura 4.4 se puede observar este cambio realizado en el código en la clase `DNIeCanSelection.java`, similar al implementado en la clase `NFCOperationsEncKitKat.java`, y que tiene en cuenta no solo a aquellos números CAN que empiecen por un 0 si no a todos aquellos que empiecen por uno, dos, tres, cuatro o cinco ceros.

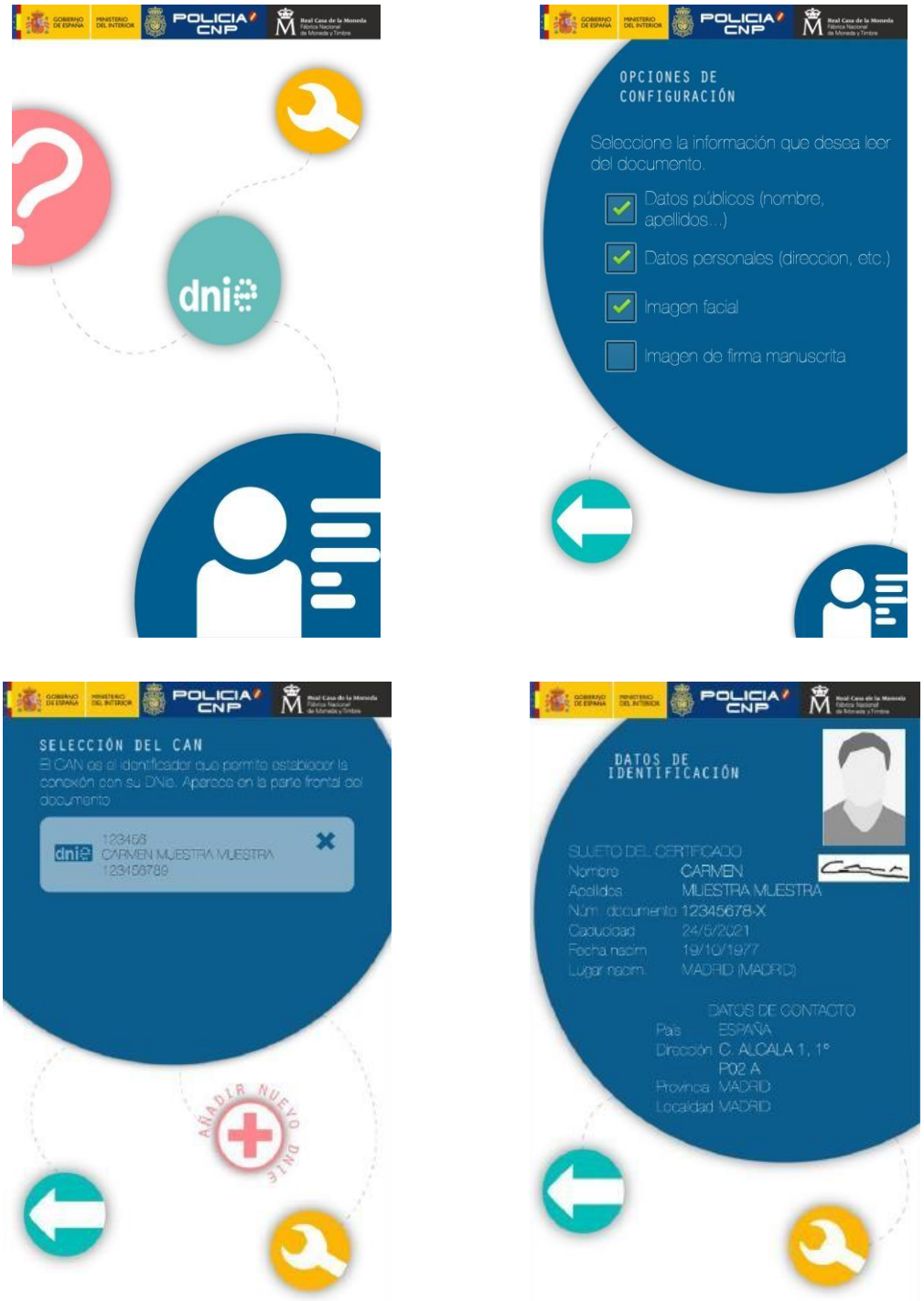

Figura 4.3. Interfaz gráfica Android_DGPApp_LECTURA

DNIeCanSelection.java

```
108         int idx=0;
109         while(idx < listA.getCount())
110         {
111             CANSpecDO canItem = listA.getItem(idx);
112             String can6digitos = canItem.getCanNumber();
113             mrtdItems.add(new MrtdItem(can6digitos, canItem.getUserName(), canItem.getUserNif()));
114             idx++;
115         }
```

DNIeCanSelection.java

```
108         int idx=0;
109         while(idx < listA.getCount())
110         {
111             CANSpecDO canItem = listA.getItem(idx);
112             String can6digitos = canItem.getCanNumber();
113
114             if (can6digitos.length()==1)
115                 can6digitos = "00000" + can6digitos;
116             else if (can6digitos.length()==2)
117                 can6digitos = "0000" + can6digitos;
118             else if (can6digitos.length()==3)
119                 can6digitos = "000" + can6digitos;
120             else if (can6digitos.length()==4)
121                 can6digitos = "00" + can6digitos;
122             else if (can6digitos.length()==5)
123                 can6digitos = "0" + can6digitos;
124
125             mrtdItems.add(new MrtdItem(can6digitos, canItem.getUserName(), canItem.getUserNif()));
126             idx++;
127         }
```

Figura 4.4 Modificación realizada en la clase DNIeCanSelection.java

4.4. Android_DNIeApp_FIRMA

Al igual que sucedía con la aplicación previa, el código se puede obtener de forma gratuita a través del portal web de la Policía Nacional [Nac]. Esta aplicación permite la firma de documentos con extensión .pdf con el DNIe3.0 a través de un dispositivo con tecnología NFC.

El funcionamiento de la aplicación es el siguiente:

1. Inicialmente, hay que vincular la aplicación con una cuenta de email.

2. Cada vez que se acceda a la aplicación, ésta comprobará automáticamente si existen documentos pendientes de firma (aquellos .pdf que provengan de una dirección de correo específica) en el email establecido anteriormente.

3. Tras seleccionar un documento para firmarlo electrónicamente, se pide el número CAN.

4. Se verifica que el número CAN introducido coincide con el número CAN del DNIe3.0 que se va a utilizar para realizar la firma. Con el fin de no tener que introducir este número cada vez que se opere con el mismo DNIe, la aplicación almacena los números CAN ya utilizados. En este punto se ha encontrado un fallo que se comentará más adelante.

5. Si el número CAN es correcto, se pide el número PIN.

6. Si el número PIN es correcto, se procede a firmar el documento.

7. Se envía el documento firmado a la dirección de correo establecida en el código.

A continuación se pueden observar unas capturas de la interfaz gráfica de esta aplicación. De izquierda a derecha y de arriba a abajo, en la Figura 4.5 nos encontramos con:

- Pantalla de inicio de la aplicación.

- Menú principal: "Buzón de entrada" (documentos pendientes de firma), "Enviados" (documentos firmados correctamente y enviados) y "Configuración" (para cambiar la cuenta de correo que se utiliza).

- Lista de los documentos que se pueden firmar (esta captura aparece cuando entramos en "Buzón de entrada").

- Tras darle al botón de "Autorizar" de la captura previa, se pide el número CAN. En este apartado hay dos opciones: introducir un número CAN nuevo o seleccionar uno utilizado previamente.

Como hemos comentado anteriormente, el almacenamiento de los números CAN utilizados tenía un fallo. Éste afectaba a aquellos números cuyo primer dígito era el 0, ya que guardaba los números sin tener en cuenta el 0 inicial y después, al intentar establecer conexión con el DNIe3.0 daba un fallo al montar el canal PACE determinando que el número CAN seleccionado y el del documento no coincidían. Para solucionar este problema se ha modificado el código en las clases `PrkOperations_kitkat.java` y `DNIeCanSelection.java`. El cambio realizado en estas clases del proyecto ha sido muy similar al mostrado anteriormente para la aplicación "Android_DGPApp_LECTURA".

Se ha enviado un correo a la dirección de atención al usuario de la Policía Nacional notificando la existencia de los errores explicados anteriormente en las dos aplicaciones (Firma y Lectura) para que puedan ser corregidos.

Figura 4.5 Interfaz gráfica Android_DNIeApp_FIRMA

Capítulo Quinto.

Experimentación. Números aleatorios.

Como se ha comentado en el primer capítulo, obtener secuencias de números aleatorios está directamente relacionado con la seguridad existente en las comunicaciones. Por ello, en esta parte del proyecto se van a analizar los números generados por el dispositivo móvil utilizado y el DNIe3.0. Hay que recordar que en el protocolo PACE, los dos extremos de la comunicación generan secuencias aleatorias durante el protocolo.

5.1. Entorno de trabajo

En esta parte de la experimentación, además del dispositivo móvil, del DNIe3.0 y del portátil mencionados en el capítulo anterior, se ha utilizado un lector externo `KLT0085 Lector multitarjetas y dni electrónico`, un dispositivo `HackRF One`, que es un periférico de radio definido por software (SDR, *Software Definition Radio*) y un programa denominado `GNU Radio` [Klo][Hac].

Se ha utilizado la consola de comandos para ejecutar los tests sobre los ficheros de datos obtenidos, así como para realizar la conversión a formato binario de los números generados por el móvil.

5.2. Dispositivo móvil

El estudio de los números aleatorios generados por el dispositivo móvil utilizado se ha llevado a cabo modificando el código de la aplicación `DNIe3.0 Brute Force v2` [Rod]. En esta nueva versión, al pulsar el botón "Start PACE", el programa entra en un bucle infinito dentro del cual se generan números aleatorios y se almacenan en un fichero de texto.

Se han recopilado diez ficheros de 1.27 MB de tamaño aproximadamente. Cada número generado por el móvil es de 32 bits, es decir, 4 bytes; lo que implica que cada fichero tenía de media unos 317500 números generados aleatoriamente.

Se han convertido estos números en números binarios y se han aplicado sobre ellos los tests `Ent` y `Dieharder`. Los resultados recabados han sido muy satisfactorios, ya que todas las pruebas han arrojado soluciones esperadas. Cabe destacar que con `Dieharder`, las pruebas con resultado WEAK han sido distintas en cada fichero analizado, lo que es sinónimo de indicios de aleatoriedad.

Con los resultados obtenidos se puede afirmar que el dispositivo móvil utilizado, `Xiaomi MI5`, es un buen generador de números aleatorios, y por tanto, las claves generadas por el mismo implican una buena seguridad en condiciones normales.

5.3. DNIe3.0

La recopilación de los números aleatorios que se generan en el chip del DNIe3.0 se ha llevado a cabo mediante el uso de un lector de tarjetas. Para poder acceder a las funcionalidades de la tarjeta, se ha utilizado la herramienta `pkcs11-tool`, accesible a partir de `OpenSC`, que es un conjunto de herramientas y bibliotecas que permite trabajar con tarjetas inteligentes mediante la línea de comandos. La utilidad `pkcs11-tool` permite administrar los datos de tarjetas inteligentes con seguridad PKCS #11. Entre las opciones que tiene destaca el poder leer los PIN, las claves y los certificados de la tarjeta. Además, si la operación así lo requiere, será necesaria la autenticación del usuario mediante el número PIN [PKC]. Desde la consola de comandos se ha ejecutado la siguiente instrucción:

```
pkcs11-tool --generate-random N > ficherosalida
```

Esta línea de comandos pide al documento `N` bytes de números aleatorios y guarda el resultado en `ficherosalida`. Los números aleatorios obtenidos están en formato binario, luego se pueden pasar directamente por los tests `Ent` y `Dieharder` sin necesidad de realizar una conversión de formato.

En el Apéndice se puede observar el resultado de estos tests para un fichero de 100 MB de datos aleatorios. Como cabía esperar, los resultados son muy buenos, es decir, el DNIe3.0 genera números con muchos indicios de aleatoriedad. Esto implica una mayor barrera y protección frente a posibles ataques.

5.4. Interferencias en la alimentación

En esta sección se va a analizar la aleatoriedad de los números generados por el DNIe3.0 introduciendo a la vez interferencias en la alimentación. Para ello se ha utilizado `GNU Radio` y `HackRF One`.

- `GNU Radio`: software de desarrollo libre que posee bloques de procesamiento de señal que permiten implementar sistemas de radio [GNU].

- `HackRF One`: periférico de software definido por radio (SDR) fabricado por Great Scott Gadgets capaz de transmitir o recibir señales de radio en la banda de 1 MHz - 6 GHz [Hac].

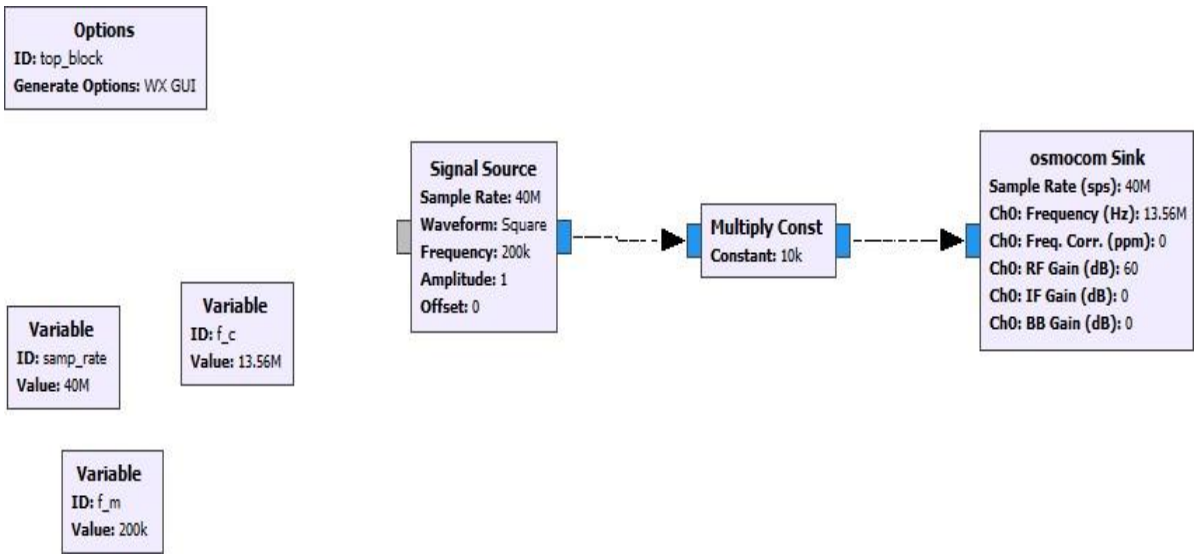

Figura 5.1. Esquema básico GNU Radio

Inicialmente, vamos a explicar el esquema básico que se ha utilizado. Como se puede observar en la Figura 5.1 nuestro esquema está formado por diferentes bloques:

- **Options:** en este bloque se señalan, entre otras características, el identificador o nombre del esquema (`ID`) y el tipo de código que se genera (`Generate Options`, WX GUI en nuestro caso).

- **Variable:** este bloque permite establecer y modificar directamente el valor de diversas características de las señales que se generan y transmiten.

- **Signal source:** se ha decidido generar una señal con diversas características:
  - **Sample rate:** tasa de muestreo en muestras por segundo, 40M.
  - **Waveform:** forma de la señal, cuadrada.
  - **Frequency:** frecuencia en hercios, 200K.
  - **Amplitude:** amplitud en voltios, 1.
  - **Offset:** 0 V.

- **Multiply const:** multiplica la amplitud de la señal generada en el bloque anterior por el valor del parámetro `Constant`, 10000 en nuestro caso.

- **osmocom Sink:** permite al dispositivo HackRF transmitir las señales. Características escogidas:
  - **Sample Rate (sps):** tasa de muestreo en muestras por segundo, 40M.
  - **Frequency (Hz):** frecuencia en hercios, 13.56M.
  - **Freq. Corr. (ppm):** factor de corrección de frecuencia en partes por millón, 0.
  - **RF Gain (dB):** ganancia de Radio Frecuencia en decibelios, 60.
  - **IF Gain (dB):** ganancia de Frecuencia Intermedia en decibelios, 0.
  - **BB Gain (dB):** ganancia de Banda Base en decibelios, 0.

La alimentación de la tarjeta mediante una fuente externa se ha realizado según se muestra en la Figura 5.2. El dispositivo `HackRF One` se ha utilizado como transmisor con dos antenas diferentes, una de banda ancha y otra NFC (ésta última es la que se muestra en la Figura 5.2). El dispositivo `HackRF One` emite la señal descrita anteriormente a través de una antena NFC, por ello ésta se encuentra muy próxima al DNI, el cual a su vez está introducido en un lector de tarjetas externo.

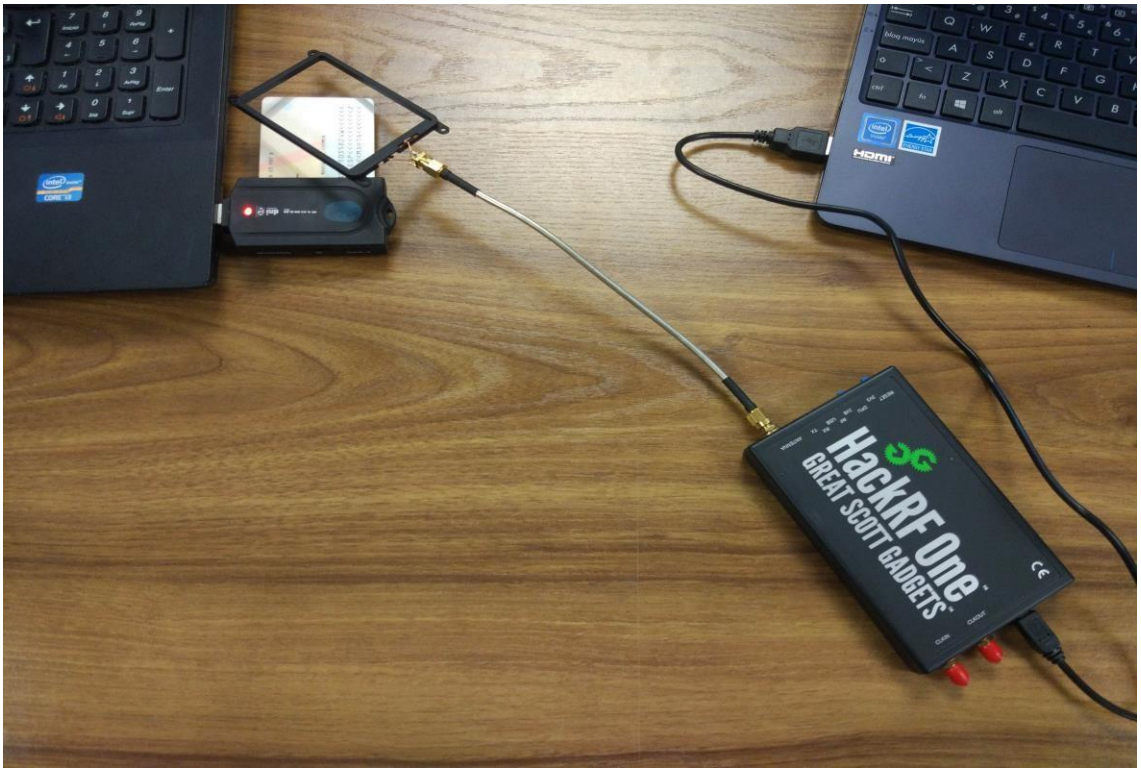

Figura 5.2. Montaje interferencias en la alimentación

La experimentación llevada a cabo se puede dividir en dos pruebas:

1. La primera prueba realizada ha sido generar números aleatorios con la tarjeta con los dos tipos de antena descritos previamente. En ambos casos se ha generado un fichero de unos 20 MB aproximadamente y se ha pasado por los tests `Ent` y `Dieharder`, obteniendo unos resultados muy buenos.

2. La segunda prueba ha tenido como objetivo observar si la tarjeta mostraba algún comportamiento diferente si se generaban continuamente secuencias de números con y sin el dispositivo `HackRF One` conectado. Para ello, se han generado diez secuencias de 100KB con el dispositivo encendido y diez secuencias con el dispositivo apagado. Se ha repetido esta situación alternando un par de veces. En ambos casos, los resultados arrojados por los tests `Ent` y `Dieharder` han sido buenos y el tiempo de ejecución ha sido el mismo que cuando no había interferencias.

Tras la realización de las tres pruebas descritas previamente, se puede llegar a la conclusión de que la introducción de una interferencia en la alimentación no modifica la calidad de los números aleatorios generados por el DNIe3.0 ni el tiempo utilizado para la generación de los números.

Conclusiones

Tras el análisis y el desarrollo de los objetivos iniciales se han podido extraer diversas conclusiones sobre el objeto de estudio de este trabajo, el DNIe3.0.

La mejora en la seguridad introducida por la FNMT en el DNIe3.0 implica que, para que un usuario pueda obtener los datos personales de otro vía NFC sin conocer su número CAN, necesitaría poder estar en contacto con el DNI durante un tiempo aproximado de 50 días, lo que hace casi inviable que pueda conseguir el objetivo del ataque.

En este caso el uso del programa `Android Studio` ha resultado fundamental a la hora de modificar el código de las aplicaciones como se ha ido considerando oportuno. Asimismo, la posibilidad de realizar los tests `Ent` y `Dieharder` directamente desde la consola de comandos de UNIX, ha permitido un entorno de trabajo cómodo y sencillo en lo relativo al análisis de las secuencias de números generadas.

A partir del análisis de las dos aplicaciones proporcionadas por la Policía Nacional se ha descubierto la existencia de un fallo que afectaba a aquellos documentos cuyo número CAN empezaba por un 0. Este error se ha notificado a la Policía Nacional.

Como punto y final, hay que destacar la importancia en la seguridad de las tarjetas criptográficas, principalmente del DNI y de las tarjetas bancarias, ya que un ataque a una de ellas puede ocasionar consecuencias muy graves.

Líneas futuras

A partir del trabajo realizado y de los resultados obtenidos, se pueden determinar algunas líneas de investigación que amplíen el campo de estudio utilizando como base este proyecto:

- Este trabajo ha sido orientado hacia el estudio del funcionamiento del protocolo PACE y la seguridad que el mismo aporta al DNIe3.0. Otra posible línea de investigación sería centrarse en el estudio del protocolo BAC, trabajando de esta manera sobre el número de soporte del DNI, que está formado por número y letras.

- Sería interesante profundizar en el estudio de la seguridad frente a otros ataques que se pueden llevar a cabo haciendo uso de la tecnología NFC, como puede ser el robo de datos privados en tarjetas bancarias.

- Se ha realizado un análisis de calidad de la generación de secuencias de números aleatorios del móvil utilizado, en este caso un `Xiaomi MI5`. Podría ampliarse este análisis a otros dispositivos móviles realizando una comparativa de la seguridad que presenta cada uno de ellos.

- Otra posible línea de investigación sería centrarse en la tecnología NFC, en las características, el funcionamiento y las vulnerabilidades que ésta presenta con el objetivo de mejorar la seguridad en las comunicaciones que utilicen dicha tecnología, ya que el desarrollo digital está guiando a la ciudadanía a un uso cada vez más frecuente de sus tarjetas y dispositivos móviles mediante NFC.

Bibliografía

[And] Proyecto Androsmex. Último acceso: 26 de septiembre de 2018. https://github.com/tsenger/androsmex

[Bal] Sánchez Ballabiga, V. Análisis de la resistencia del DNIe3.0 ante el robo de identidad mediante tecnología NFC. Universidad de Zaragoza, Escuela de Ingeniería y Arquitectura, 2016.

[BEB] Robert G. Brown, Dirk Eddelbuettel and David Bauer. Dieharder: A Random Number Test Suite, Version 3.31.1

[BK] Jens Bender and Dennis Kügler. Introducing the PACE solution. *Keesing Journal of Documents & Identity*, 30:26-29, 2009.

[BOE] Boletín Oficial del Estado. Número 307, de 24/12/2005, páginas 42090-42093. https://boe.es/buscar/act.php?id=BOE-A-2005-21163&b=13&tn=1&p=20150530#a12

*DH+ Whitfield Diffie, Martin Hellman. New directions in cryptography. IEEE Transactions on information theory, vol. It.-22, no.6, noviembre 1976. Disponible en: https://ee.stanford.edu/~hellman/publications/24.pdf

*DMG+ Bueno Delgado, M. V.; Pavón Mariño, P.; Gea García, A. La tecnología NFC y sus aplicaciones en un entorno universitario. En: Espacio-Teleco: revista de la ETSIT-UPCT, vol. 2. Cartagena: Universidad Politécnica de Cartagena, Escuela Técnica Superior de Ingeniería de Telecomunicación, 2011. p.14-23.

[GNU] GNU Radio, The Free & Open Software Radio Ecosystem. Último acceso: 26 de septiembre de 2018. https://www.gnuradio.org/

[Hac] Especificaciones técnicas del periférico HackRF One. Último acceso: 26 de septiembre de 2018. https://greatscottgadgets.com/

*Her+ Mancilla Herrera, A. M. Números aleatorios. Historia, teoría y aplicaciones. En: Revista Científica Ingeniería y Desarrollo, No 8 (2000). Universidad del Norte. p.51-52.

[IDQ] ID Quantique SA. Quantis Random Number Generator. Último acceso: 26 de septiembre de 2018. https://www.idquantique.com/random-number-generation/overview/

[IOS] International Organization for Standardization. ISO/IEC 1443:2016: Identification cards - Contactless integrated circuit cards - Proximity cards, March 2016.

*KLT+ KLT0085 Lector multitarjetas y dni electrónico. Último acceso: 26 de septiembre de 2018. http://kloner.es/lectores-de-tarjetas-y-dni-e/149-lector-externo-de-dni-e-y-multitarjetas.html

*Len+ Lenovo B590. Especificaciones. Último acceso: 26 de septiembre de 2018. https://www.magitech.pe/laptop-lenovo-b590-59392028-intel-core-i3-3110m-2-4ghz.html

*MAT+ Documentación de MATLAB para las funciones: polyfit, polyval y fit. Último acceso: 26 de septiembre de 2018. https://es.mathworks.com/help/matlab/

[Nac] Dirección General de la Policía Nacional. Último acceso: 26 de septiembre de 2018. https://www.dnielectronico.es/PortalDNIe/

*NFC+ NFC Forum. Especificación Técnica. Último acceso: 26 de septiembre de 2018. http://www.nfc-forum.org

[PKC] PKCS#11 man page. Último acceso: 26 de septiembre de 2018. https://manpages.debian.org/stretch/opensc/pkcs11-tool.1.en.html

[RSNSB] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker. A statistical test suite for random and pseurandom number generators for cryptographic applications, 2001. Último acceso: 26 de septiembre de 2018. http://www.dtic.mil/docs/citations/ADA393366

[Rod] Rodríguez, R. Proyecto DNIe3.0 brute force v2. Último acceso: 26 de septiembre de 2018. https://github.com/ricardojrdez/DNIe3.0_brute_force_v2

[Stu] Android Studio. Último acceso: 26 de septiembre de 2018. https://developer.android.com/studio/intro/

[Wal] John Walker. ENT, A Pseudorandom Number Sequence Test Program, January 28th, 2008. Último acceso: 26 de septiembre de 2018. http://www.fourmilab.ch/random/

[Xia] Xiaomi Mi5. Especificaciones técnicas. Último acceso: 26 de septiembre de 2018. https://www.xiaomi-shop.es/comprar-xiaomi-mi5-espana.html

Apéndice
Resultados obtenidos con los tests Ent y Dieharder

A continuación se muestra un ejemplo de los resultados obtenidos por los tests `Ent` y `Dieharder` tras haberlos aplicado sobre un fichero de 100MB de números aleatorios generados por el DNIe3.0.

➢ Ent

```
Entropy = 7.999999 bits per byte.

Optimum compression would reduce the size
of this 104040000 byte file by 0 percent.

Chi square distribution for 104040000 samples is 206.92, and
randomly would exceed this value 97.50 percent of the times.

Arithmetic mean value of data bytes is 127.4919 (127.5 = random).
Monte Carlo value for Pi is 3.141657439 (error 0.00 percent).
Serial correlation coefficient is 0.000027 (totally uncorrelated =
0.0).
```

➢ Dieharder

```
#=============================================================##
dieharder version 3.31.1 Copyright 2003 Robert G. Brown      #
#=============================================================#
   rng_name    |           filename             |rands/second|
        mt19937|                     DNI100.txt|  1.19e+08  |
#=============================================================#
      test_name   |ntup| tsamples |psamples|  p-value |Assessment
   #==============================================================#
       diehard_birthdays|   0|    100|     100|0.51582013|   PASSED
         diehard_operm5|  0|   1000000|      100|0.35990133| PASSED
   diehard_rank_32x32|   0|  40000|      100|0.51047965|    PASSED
      diehard_rank_6x8|   0|  100000|     100|0.14702443|   PASSED
      diehard_bitstream|   0| 2097152|     100|0.23791501|   PASSED
        diehard_opso|   0|  2097152|     100|0.37570783|   PASSED
        diehard_oqso|   0|  2097152|     100|0.34623975|   PASSED
         diehard_dna|   0|   2097152|     100|0.29103464|  PASSED
        diehard_count_1s_str| 0| 256000|  100|0.85267902|  PASSED
        diehard_count_1s_byt| 0| 256000|  100|0.61107634|  PASSED
           diehard_parking_lot| 0| 12000| 100|0.61013501|  PASSED
    diehard_2dsphere|   2|      8000|     100|0.99923771|   WEAK
    diehard_3dsphere|   3|      4000|     100|0.43076814|  PASSED
     diehard_squeeze|   0|    100000|     100|0.74107340|  PASSED
        diehard_sums|   0|       100|     100|0.00131427|   WEAK
        diehard_runs|   0|    100000|     100|0.85706794|  PASSED
        diehard_runs|   0|    100000|     100|0.98189167|  PASSED
       diehard_craps|   0|    200000|     100|0.85005925|  PASSED
       diehard_craps|   0|    200000|     100|0.83736127|  PASSED
```

```
 marsaglia_tsang_gcd|   0|  10000000|      100|0.85077666|  PASSED
 marsaglia_tsang_gcd|   0|  10000000|      100|0.97309442|  PASSED
         sts_monobit|   1|    100000|      100|0.45562560|  PASSED
            sts_runs|   2|    100000|      100|0.94303590|  PASSED
          sts_serial|   1|    100000|      100|0.79563292|  PASSED
          sts_serial|   2|    100000|      100|0.74589954|  PASSED
          sts_serial|   3|    100000|      100|0.99794568|   WEAK
          sts_serial|   3|    100000|      100|0.71438269|  PASSED
          sts_serial|   4|    100000|      100|0.22824870|  PASSED
          sts_serial|   4|    100000|      100|0.70153036|  PASSED
          sts_serial|   5|    100000|      100|0.04810141|  PASSED
          sts_serial|   5|    100000|      100|0.20932288|  PASSED
          sts_serial|   6|    100000|      100|0.64779316|  PASSED
          sts_serial|   6|    100000|      100|0.82997735|  PASSED
          sts_serial|   7|    100000|      100|0.91421169|  PASSED
          sts_serial|   7|    100000|      100|0.76100637|  PASSED
          sts_serial|   8|    100000|      100|0.68509389|  PASSED
          sts_serial|   8|    100000|      100|0.09322100|  PASSED
          sts_serial|   9|    100000|      100|0.03934848|  PASSED
          sts_serial|   9|    100000|      100|0.21152289|  PASSED
          sts_serial|  10|    100000|      100|0.22610519|  PASSED
          sts_serial|  10|    100000|      100|0.87958520|  PASSED
          sts_serial|  11|    100000|      100|0.32472761|  PASSED
          sts_serial|  11|    100000|      100|0.50131874|  PASSED
          sts_serial|  12|    100000|      100|0.04899673|  PASSED
          sts_serial|  12|    100000|      100|0.21310621|  PASSED
          sts_serial|  13|    100000|      100|0.20661699|  PASSED
          sts_serial|  13|    100000|      100|0.54329517|  PASSED
          sts_serial|  14|    100000|      100|0.50576770|  PASSED
          sts_serial|  14|    100000|      100|0.48540694|  PASSED
          sts_serial|  15|    100000|      100|0.95559929|  PASSED
          sts_serial|  15|    100000|      100|0.94844289|  PASSED
          sts_serial|  16|    100000|      100|0.62525270|  PASSED
          sts_serial|  16|    100000|      100|0.78623981|  PASSED
         rgb_bitdist|   1|    100000|      100|0.19508169|  PASSED
         rgb_bitdist|   2|    100000|      100|0.12033190|  PASSED
         rgb_bitdist|   3|    100000|      100|0.82439561|  PASSED
         rgb_bitdist|   4|    100000|      100|0.58543536|  PASSED
         rgb_bitdist|   5|    100000|      100|0.06464673|  PASSED
         rgb_bitdist|   6|    100000|      100|0.57646180|  PASSED
         rgb_bitdist|   7|    100000|      100|0.21302270|  PASSED
         rgb_bitdist|   8|    100000|      100|0.25632308|  PASSED
         rgb_bitdist|   9|    100000|      100|0.93485845|  PASSED
         rgb_bitdist|  10|    100000|      100|0.40151405|  PASSED
         rgb_bitdist|  11|    100000|      100|0.92090250|  PASSED
         rgb_bitdist|  12|    100000|      100|0.78947732|  PASSED
rgb_minimum_distance|   2|     10000|     1000|0.93025044|  PASSED
rgb_minimum_distance|   3|     10000|     1000|0.16135701|  PASSED
rgb_minimum_distance|   4|     10000|     1000|0.39664774|  PASSED
rgb_minimum_distance|   5|     10000|     1000|0.34818022|  PASSED
    rgb_permutations|   2|    100000|      100|0.55323173|  PASSED
    rgb_permutations|   3|    100000|      100|0.86846366|  PASSED
    rgb_permutations|   4|    100000|      100|0.12630018|  PASSED
    rgb_permutations|   5|    100000|      100|0.67504180|  PASSED
      rgb_lagged_sum|   0|   1000000|      100|0.05155769|  PASSED
      rgb_lagged_sum|   1|   1000000|      100|0.80712641|  PASSED
      rgb_lagged_sum|   2|   1000000|      100|0.83151947|  PASSED
      rgb_lagged_sum|   3|   1000000|      100|0.98043457|  PASSED
      rgb_lagged_sum|   4|   1000000|      100|0.16722928|  PASSED
      rgb_lagged_sum|   5|   1000000|      100|0.69394645|  PASSED
```

```
      rgb_lagged_sum|   6|   1000000|      100|0.03652381|  PASSED
      rgb_lagged_sum|   7|   1000000|      100|0.65792840|  PASSED
      rgb_lagged_sum|   8|   1000000|      100|0.62208782|  PASSED
      rgb_lagged_sum|   9|   1000000|      100|0.67690891|  PASSED
      rgb_lagged_sum|  10|   1000000|      100|0.29101240|  PASSED
      rgb_lagged_sum|  11|   1000000|      100|0.00421134|   WEAK
      rgb_lagged_sum|  12|   1000000|      100|0.96940508|  PASSED
      rgb_lagged_sum|  13|   1000000|      100|0.93768333|  PASSED
      rgb_lagged_sum|  14|   1000000|      100|0.29380970|  PASSED
      rgb_lagged_sum|  15|   1000000|      100|0.98959946|  PASSED
      rgb_lagged_sum|  16|   1000000|      100|0.95451001|  PASSED
      rgb_lagged_sum|  17|   1000000|      100|0.54218805|  PASSED
      rgb_lagged_sum|  18|   1000000|      100|0.22988810|  PASSED
      rgb_lagged_sum|  19|   1000000|      100|0.77124591|  PASSED
      rgb_lagged_sum|  20|   1000000|      100|0.48693762|  PASSED
      rgb_lagged_sum|  21|   1000000|      100|0.71452359|  PASSED
      rgb_lagged_sum|  22|   1000000|      100|0.83825666|  PASSED
      rgb_lagged_sum|  23|   1000000|      100|0.82230115|  PASSED
      rgb_lagged_sum|  24|   1000000|      100|0.48955479|  PASSED
      rgb_lagged_sum|  25|   1000000|      100|0.95393644|  PASSED
      rgb_lagged_sum|  26|   1000000|      100|0.80076189|  PASSED
      rgb_lagged_sum|  27|   1000000|      100|0.58251110|  PASSED
      rgb_lagged_sum|  28|   1000000|      100|0.16650751|  PASSED
      rgb_lagged_sum|  29|   1000000|      100|0.37145770|  PASSED
      rgb_lagged_sum|  30|   1000000|      100|0.38241018|  PASSED
      rgb_lagged_sum|  31|   1000000|      100|0.75488822|  PASSED
      rgb_lagged_sum|  32|   1000000|      100|0.33294293|  PASSED
    rgb_kstest_test|   0|     10000|     1000|0.23656467|  PASSED
    dab_bytedistrib|   0|  51200000|        1|0.94937865|  PASSED
            dab_dct| 256|     50000|        1|0.82712701|  PASSED
Preparing to run test 207.  ntuple = 0
       dab_filltree|  32|  15000000|        1|0.94806881|  PASSED
       dab_filltree|  32|  15000000|        1|0.89997610|  PASSED
Preparing to run test 208.  ntuple = 0
      dab_filltree2|   0|   5000000|        1|0.20532524|  PASSED
      dab_filltree2|   1|   5000000|        1|0.27459037|  PASSED
Preparing to run test 209.  ntuple = 0
       dab_monobit2|  12|  65000000|        1|0.16382664|  PASSED
```