Trabajo Fin de Grado

Análisis de la resistencia del DNIe3.0 ante el robo de identidad mediante tecnología NFC

Security analysis of DNIe3.0 against NFC-based identity theft

Departamento de Informática e Ingeniería de Sistemas

Grado en Ingeniería Informática

Autor: Víctor Sánchez Ballabiga

Director: Ricardo J. Rodríguez

UNIZAR

Septiembre de 2016

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Víctor Sánchez Ballabriga,

con nº de DNI 73412089E en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Análisis de la resistencia del DNIe3.0 ante el robo de identidad mediante tecnología NFC,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 29 de Agosto de 2016

Fdo: Víctor Sánchez Ballabriga

Análisis de la resistencia del DNIe3.0 ante el robo de identidad mediante tecnología NFC

RESUMEN

Near Field Communication (NFC) (o comunicación de campo cercano, en español), es una tecnología de identificación por radiofrecuencia que permite la comunicación sin contacto entre dispositivos, limitando la distancia de comunicación a unos pocos centímetros (hasta 10 cm). Dichos estándares cubren distintos protocolos de comunicación e intercambio de datos basados en protocolos de nivel superior, como ISO/IEC 14443 o FeliCa.

La incursión de esta tecnología en el mercado ha hecho que muchos empresas e instituciones añadan NFC como forma de identificación o pago a sus servicios para facilitar la experiencia de uso a sus usuarios. En los últimos años esta tecnología también se ha introducido en los pasaportes y documentos de identidad de diferentes países. El uso de este tipo de documentos personales ofrece tanto a las autoridades como a los ciudadanos una nueva forma de identificar e identificarse de manera más ágil. En España, esta tecnología ya está funcionando en los documentos de identidad desde diciembre de 2015.

Sin embargo, esta nueva interfaz sin contacto añade nuevos riesgos que hacen que si no se dota a los documentos de identidad de una buena seguridad puede dar lugar al robo de identidad.

Este proyecto pretende realizar una investigación sobre las medidas de seguridad implementadas en esta nueva interfaz NFC de la última versión del Documento Nacional de Identidad (DNIe3.0) para detectar así posibles vulnerabilidades. En concreto, se van a estudiar los protocolos de conexión con esta interfaz NFC, desarrollando pruebas de concepto. De este modo, se han detectado las fortalezas de los mecanismos de seguridad implantados, como el generador de números pseudoaleatorios perfectamente uniforme, o la seguridad en las comunicaciones, bien cifradas en todo momento. Existe, sin embargo, una vulnerabilidad explotable: la ausencia de defensas contra ataques de fuerza bruta.

Índice general

Índice de figuras

Capítulo 1

Introducción

1.1. Motivación

Desde la expansión de Internet y el uso de ordenadores personales de forma masiva se ha pasado de los primeros documentos nacionales de identidad (DNI) en papel en 1944 [ABC13] hacia los DNI electrónicos que facilitan al ciudadano su identificación en páginas web de la Administración Pública. Este avance se produjo en 2006 con el primer DNI electrónico español, llamado DNIe2.0, que incorporaba un chip para ser leído por un lector de tarjetas inteligentes. Más adelante, en el año 2015, se introdujo la versión 3.0 (llamado DNIe3.0), que incorpora conexión *Near Field Communication* (NFC) [SER15].

El crecimiento de la tecnología de comunicación sin contacto NFC (o comunicación de campo cercano, en español) estos últimos años hace interesante su estudio en cualquier ámbito, como la vulnerabilidad descubierta en las tarjetas Mifare Classic [GKGM$^+$08] o el estudio sobre los ataques de retransmisión en tarjetas de crédito [VR15]. Pero en el caso concreto del DNIe3.0 toma mayor relevancia, ya que se trata de la información más personal del ciudadano, que ha de estar bien protegida cuando se dota al documento de conectividad inalámbrica.

Además, la poca documentación disponible sobre la implementación de esta nueva interfaz NFC por parte del Ministerio del Interior hace necesaria esta investigación para conocer dicha implementación, sus posibles vulnerabilidades, y las defensas implementadas en el mismo.

1.2. Objetivo

El objetivo de este TFG es realizar un estudio sobre la seguridad de la nueva interfaz NFC añadida a la última versión 3.0 del DNI español ante el robo de identidad, un delito que en el siglo XXI ha ganado mayor popularidad y rentabilidad para los delincuentes [Leg14]. En concreto, se ha realizado un estudio teórico de los distintos vectores de ataque

posibles y una prueba de concepto del ataque más factible estudiado (concretamente, el ataque de fuerza bruta).

1.3. Organización

El documento está dividido en siete capítulos y pretende seguir un orden que permita la lectura continua.

El capítulo 2 define algunos conceptos previos que servirán al lector para comprender el resto del documento. Incluye una pequeña introducción a la familia de protocolos NFC, una descripción de qué son y cómo funcionan los ataques de fuerza bruta. También se da una noción de qué es la trazabilidad de números aleatorios, así como una introducción al lenguaje de modelado UML y a los diagramas de secuencia.

En el capítulo 3 se hace un análisis más detallado de los elementos que conforman la versión 3.0 del DNI, así como de los protocolos de acceso por NFC. Además, se realiza un análisis de la cardinalidad del espacio de claves de los códigos de seguridad utilizados para acceder a la información del DNIe3.0.

Después, en el capítulo 4, se comienza con una explicación de las herramientas utilizadas para realizar la experimentación relativa al proyecto. A continuación, se explica el ataque de fuerza bruta implementado contra uno de los protocolos de acceso al documento, así como los resultados obtenidos. Finalmente, se pasa a explicar el estudio realizado sobre la distribución de los números pseudoaleatorios usados por dicho protocolo.

A continuación, en el capítulo 5 se proponen una serie de ideas para aumentar la resistencia del DNIe3.0 ante los ataques de fuerza bruta. Seguidamente, en el capítulo 6 se hace un repaso de las investigaciones previas realizadas en este ámbito y qué aporta este proyecto. Por último, el capítulo 7 incluye las conclusiones del trabajo y posibles líneas futuras de investigación.

Además, el proyecto cuenta con dos apéndices. En el Apéndice A se incluye un diagrama de Gantt con las horas dedicadas a las diferentes partes de este proyecto, así como una breve explicación. En el Apéndice B se incluye el código fuente de la clase principal de la aplicación desarrollada para la prueba de concepto.

Capítulo 2

Conocimientos previos

En este capítulo se definen conceptos básicos para la comprensión de la memoria. Se pretende dar una introducción a las distintas temáticas que se van a abordar en capítulos posteriores, como los estándares que utiliza NFC, los ataques de fuerza bruta, una breve explicación de la trazabilidad de números pseudoaleatorios y un breve resumen sobre UML.

2.1. Tecnología NFC

La tecnología *Near Field Communication* (NFC) [Int13] engloba a la familia de estándares para establecer una comunicación inalámbrica entre dos dispositivos en proximidad. Dichos estándares cubren distintos protocolos de comunicación e intercambio de datos y están basados en otros estándares de identificación por radio frecuencia (RFID) como ISO/IEC 14443 [fS16] o FeliCa [Jap10]. FeliCa es utilizado principalmente en Japón, siendo la familia ISO/IEC 14443 el estándar usado en Europa y EE.UU. De hecho, el DNI electrónico español se basa en dicho estándar ISO/IEC.

La característica más relevante de NFC es la capacidad de disponer de elementos pasivos, los cuales se apoyan en el campo electromagnético generado por otro dispositivo para comunicarse. Una comunicación consta de dos componentes: el *Proximity Coupling Device* (PDC), que tiene corriente (elemento activo) y actúa como lector/escritor, y el *Proximity Integrated Circuit Card* (PICC), que es la tarjeta (elemento pasivo) y cuyo chip se alimenta por inducción gracias al campo electromagnético generado por el lector. El DNIe3.0 funciona como PICC.

Dentro de la ISO/IEC 14443 [fS16] se pueden distinguir dos tipos de tecnologías: A y B, que a pesar de trabajar ambas en la misma frecuencia (13,56 MHz) se diferencian en la modulación utilizada. El chip NFC del DNIe3.0 es de tipo B.

El estándar consta de cuatro partes:

ISO/IEC 14443-1: Describe las características físicas.

ISO/IEC 14443-2: Describe la potencia y señal de la radiofrecuencia.

ISO/IEC 14443-3: Detalla los algoritmos de inicialización y anti-colisión.

ISO/IEC 14443-4: Protocolo de transmisión.

2.2. Ataques de fuerza bruta

Los ataques de fuerza bruta son uno de los métodos de ataque más antiguos conocidos. Normalmente se utilizan para romper sistemas con identificación mediante contraseña, introduciendo en el sistema todas sus posibles combinaciones de caracteres hasta encontrar la contraseña correcta, haciendo que el sistema haga su comprobación e informe de si se ha tenido éxito en ese intento. En caso contrario, se pasa a probar la siguiente combinación. En muchos casos este tipo de ataques suelen combinarse con ataques de diccionario (combinando palabras de un diccionario en vez de caracteres aleatorios), reduciendo así las posibles combinaciones y por tanto aumentando la probabilidad de acierto.

Este tipo de ataque tiene dos problemas principales. El primero es la cardinalidad del espacio de claves: un espacio de claves con una cardinalidad grande hace crecer exponencialmente el tiempo necesario para averiguarla (en el peor de los casos). El segundo problema son los mecanismos de defensa que incorporan algunos protocolos: detectado un ataque, los intentos sucesivos de autenticación son cada vez más lentos o incluso se bloquea el proceso de autenticación.

2.3. Trazabilidad de números pseudoaleatorios

Debido a que los ordenadores son máquinas deterministas no es posible implementar un algoritmo que genere números perfectamente aleatorios, por lo que existen generadores de números pseudoaleatorios. Estos algoritmos intentan simular una distribución aleatoria partiendo de un valor inicial (llamado semilla).

Una sucesión de números pseudoaleatorios es *trazable* cuando su distribución no es uniforme, y por lo tanto se puede prever con un cierto grado de exactitud cuál será el siguiente número generado o en qué rango de valores se encontrará.

Así pues, un algoritmo que utilice un generador predecible presenta una vulnerabilidad que un atacante puede aprovechar.

2.4. UML y diagramas de secuencia

Unified Modeling Language (UML) es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación o esquemas de bases de datos [OMG11].

En este proyecto se van a usar un tipo concreto de diagramas UML, los diagramas de secuencia. Estos diagramas muestran una interacción, que representa la secuencia de

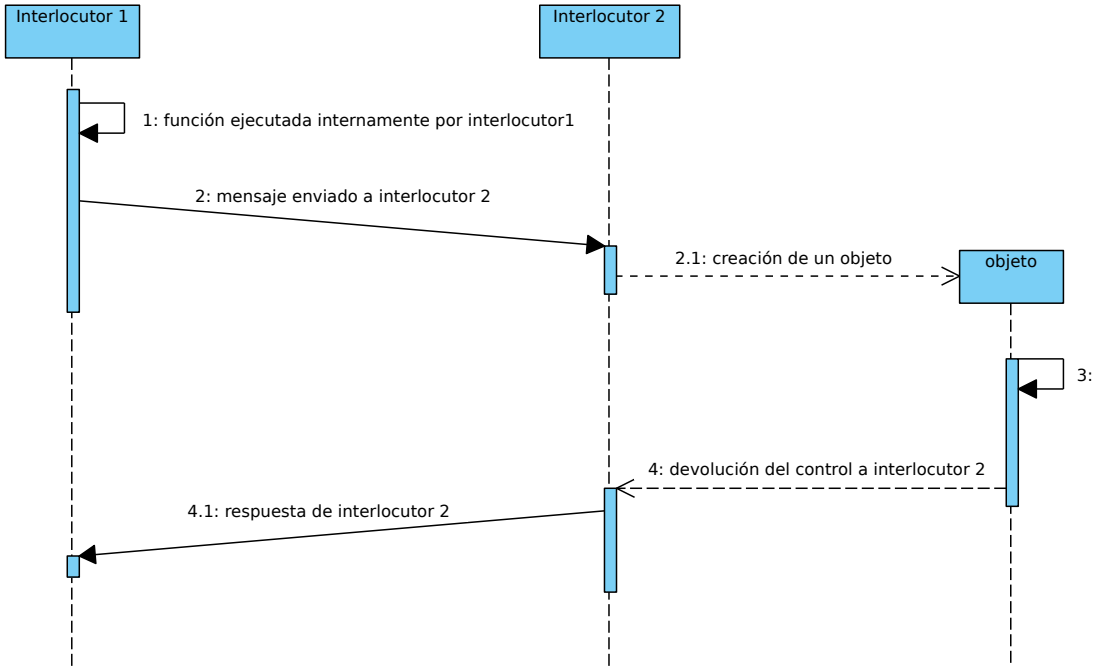


Figura 2.1: Ejemplo de diagrama de secuencia.

mensajes entre instancias de clases, componentes, subsistemas o actores. El tiempo fluye por el diagrama y muestra el flujo de control de un participante a otro (véase como ejemplo el diagrama de la Figura 2.1).

Capítulo 3

Análisis del DNIe3.0

En este capítulo se va a explicar todo lo relativo al DNIe3.0, tanto sus características físicas como las formas de comunicación con su chip NFC.

3.1. Características físicas del DNIe3.0

El DNIe3.0 está hecho de policarbonato y tiene las mismas dimensiones que una tarjeta de crédito (85,60 mm de ancho y 53,98 mm de alto) [fS03]. Aparece diversa información impresa relativa al ciudadano, tanto en el anverso como en el reverso [Nacc]. En concreto:

Anverso: En esta cara del documento, como se puede ver en la Figura 3.1 se encuentran: A la izquierda la foto en blanco y negro del rostro del ciudadano (1), con el número de identificación fiscal (NIF) debajo (2), el cual consta de un número de 8 dígitos y una letra de control obtenida a partir de estos. A la derecha de la foto se encuentran, por fila: Los apellidos del ciudadano (3), su nombre (4), el sexo ("M" para hombres y "F" para mujeres) (5) y la nacionalidad ("ESP") (6) (ambos en la misma fila), en la siguiente fila se encuentra la fecha de nacimiento (en formato "dd mm aaaa") (7), después aparece el número de soporte (3 letras y 6 dígitos aleatorios) (8), y la fecha de expiración (en formato "dd mm aaaa") (9) en la última fila. Debajo de esta última fila se encuentra la firma manuscrita (10).

Por último, en la esquina inferior derecha se encuentra el *Card Access Number* (CAN) (11) (o número de acceso a la tarjeta, en español). Es un número de seis dígitos que se utiliza como clave para establecer una conexión segura a través del protocolo *Password Authenticated Connection Establishment* (PACE), explicado en la Sección 3.2.2.

Reverso: En esta cara del documento, como se puede ver en la Figura 3.2 se encuentran: Dirección del ciudadano (1). En la parte izquierda, de forma vertical está el número de equipo de expedición del documento (código de la comisaría donde

Figura 3.1: Anverso del DNIe3.0.

se ha expedido el documento) (2). A la derecha del chip se encuentra la ciudad y provincia de nacimiento (3). Debajo aparecen el nombre de los progenitores (padre / madre) (4). Y en la parte inferior de esta cara está el *Machine Readable Zone* (MRZ) (o zona legible por una máquina, en español), son tres líneas con la información del documento en el formato adecuado para el reconocimiento óptico de caracteres (se rige por el estándar ISO/IEC 1073-2 [fS76]).

3.1.1. Elementos de seguridad físicos

Este nuevo modelo de DNI cuenta con los mismos elementos físicos de seguridad que la versión anterior, e incorpora algunos nuevos. Todos estos elementos se definen a continuación y se pueden ver en la Figura 3.3:

- **Kinegrama:** Estructura de difracción microscópica, en la que al mover la imagen muestra animaciones gráficas.
- **Tinta ópticamente variable:** Tinta que contiene en su estructura finas partículas que otorgan el efecto óptico de variabilidad del color cuando se cambia el ángulo de incidencia de la luz y de observación.
- **Ventana transparente con grabado láser del número de soporte del documento**.
- ***Changeable Laser Image* (CLI) en bajo relieve:** Consta de una imagen "fantasma", igual que la foto del ciudadano de pequeñas dimensiones con la fecha de expedición en formato "ddmmaa" encima.
- **Embosados efecto mate:** Cambio en la impresión del documento en dos franjas oblicuas.

Figura 3.2: Reverso del DNIe3.0.

- **Embosados en alto relieve:** Da relieve a pequeñas zonas del documento.
- **Tintas UV en iris:** Tintas solo visibles mediante luz ultravioleta.
- **Tinta Oasis:** Tinta opaca que oculta las capas que hay por debajo. Cuando se ve a través de un filtro de polarización revela las capas y los agujeros por debajo, con el fin de mostrar la tinta fluorescente en las zonas de los agujeros que dibuja el patrón de seguridad.

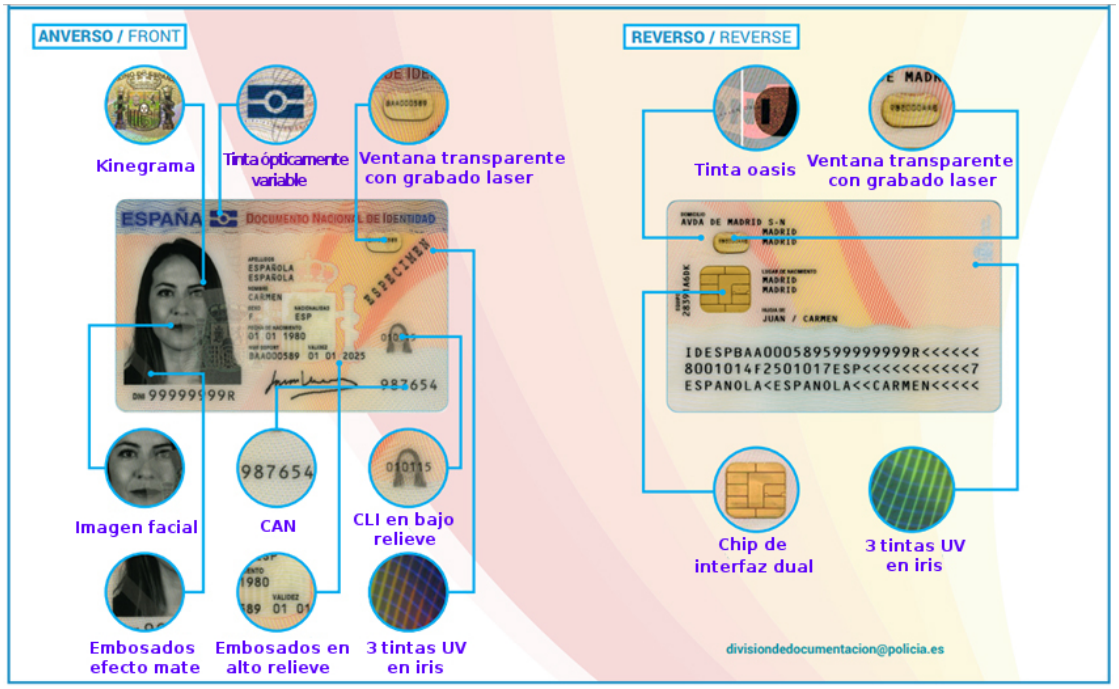


Figura 3.3: Elementos físicos de seguridad del DNIe3.0.

3.2. Interfaz NFC del DNIe3.0

La interfaz NFC del DNIe3.0 permite añadir una capa más de conectividad al documento, pretendiendo adaptarlo a las nuevas tecnologías sin contacto. De este modo, se puede utilizar un dispositivo con NFC como puede ser un teléfono móvil para realizar las tareas que hasta ahora se debían hacer con un lector de tarjetas inteligentes y un ordenador. Un ejemplo de esta nueva forma de uso es la aplicación móvil *VIDSigner* desarrollada por *ValidatedID*[1], que permite firmar electrónicamente documentos usando el DNIe3.0.

Existen dos protocolos para conectarse vía NFC con el DNIe3.0, explicados a continuación.

3.2.1. Protocolo BAC

Basic Access Control (BAC) [ICA15] es un protocolo criptográfico diseñado para asegurar que sólo las personas autorizadas pueden leer de forma inalámbrica la información personal de los documentos de identidad o pasaportes con chip NFC. Utiliza datos tales como el número del documento, la fecha de nacimiento y fecha de vencimiento del documento (ambas fechas en formato "aammdd") para negociar una clave de sesión. Esta

[1] http://www.validatedid.com/

clave se utiliza para cifrar la comunicación entre el chip y el dispositivo de lectura. Con este protocolo se puede acceder a toda la información almacenada en el DNI, incluyendo certificados digitales.

Como se describe en [LKLRP07], el protocolo comienza derivando unas claves iniciales K_{ENC} (clave de cifrado) y K_{MAC} (*Message Authentication Code*) aplicando una clave de cifrado de tipo SHA-1 a los 24 bytes del MRZ que forman los tres datos necesarios. Una vez obtenidas estas dos claves iniciales, se establece la clave de sesión mediante el protocolo *three-pass authentication*, explicado a continuación.

Como se puede ver en el diagrama de secuencia de la Figura 3.4, el protocolo comienza con el DNIe3.0 generando un *nonce* ($nonceDNI$) de 8 bytes, que es enviado al dispositivo. El dispositivo ahora genera dos números aleatorios: $nonceDevice$ (de 8 bytes) y K_{Device} (de 16 bytes), y forma una variable X concatenando $nonceDevice$ con $nonceDNI$ y con K_{Device}, y lo cifra con la clave K_{ENC} para crear E_{Device} (cifrado 3-DES). También crea M_{Device} aplicando la clave K_{MAC} a la variable E_{Device} (para autenticar el mensaje). Estas dos variables se concatenan formando EM_{Device} y se envía al documento. Cuando lo recibe, el DNIe3.0 descifra y verifica EM_{Device}, y si es correcto se genera K_{DNI}, un *nonce* de 16 bytes. A continuación se crea la variable Y concatenando $nonceDNI$, $nonceDevice$ y K_{DNI}. Esta variable se cifra con 3-DES utilizando K_{ENC} y se crea M_{DNI} con la clave K_{MAC}, igual que se ha hecho antes en el dispositivo. El documento ahora genera la clave de sesión que se usará una vez establecida la conexión, KS_{Seed}, mediante la fórmula $KS_{Seed} = K_{Devive} \oplus K_{DNI}$. Por último, genera EM_{DNI} concatenando E_{DNI} y M_{DNI}, y lo envía al dispositivo. El dispositivo descifrará y verificará este valor, y si es correcto generará su clave de sesión KS_{Seed} de la misma manera que el DNIe3.0.

De esta manera, tanto dispositivo como DNIe3.0 tendrán la misma clave simétrica de sesión para comunicarse de forma segura a partir de ahora.

3.2.2. Protocolo PACE

Password Authenticated Connection Establishment (PACE) [fSidI15] (o establecimiento de conexión autenticada mediante contraseña, en español), es un mecanismo criptográfico de intercambio seguro de claves de sesión usando como semilla un código de seguridad con una cardinalidad reducida. En el caso del DNIe3.0, este código consiste en el CAN (elemento 11 en Figura 3.1), que es un número de seis dígitos. Este protocolo usa el protocolo Diffie-Hellman [DH76, Mer78] explicado a continuación para generar las claves de sesión.

Protocolo Diffie-Hellman

Es un protocolo de establecimiento de claves entre partes que no han tenido contacto previo, utilizando un canal inseguro, y de manera anónima (no autentificada). Para ello cada interlocutor elige un número público, que será compartido con el otro, y un número secreto. Usando una fórmula matemática, que incluye la exponenciación, cada interlocutor hace una serie de operaciones con el número público y los números secretos respectivos. A continuación, los interlocutores se intercambian los resultados de forma

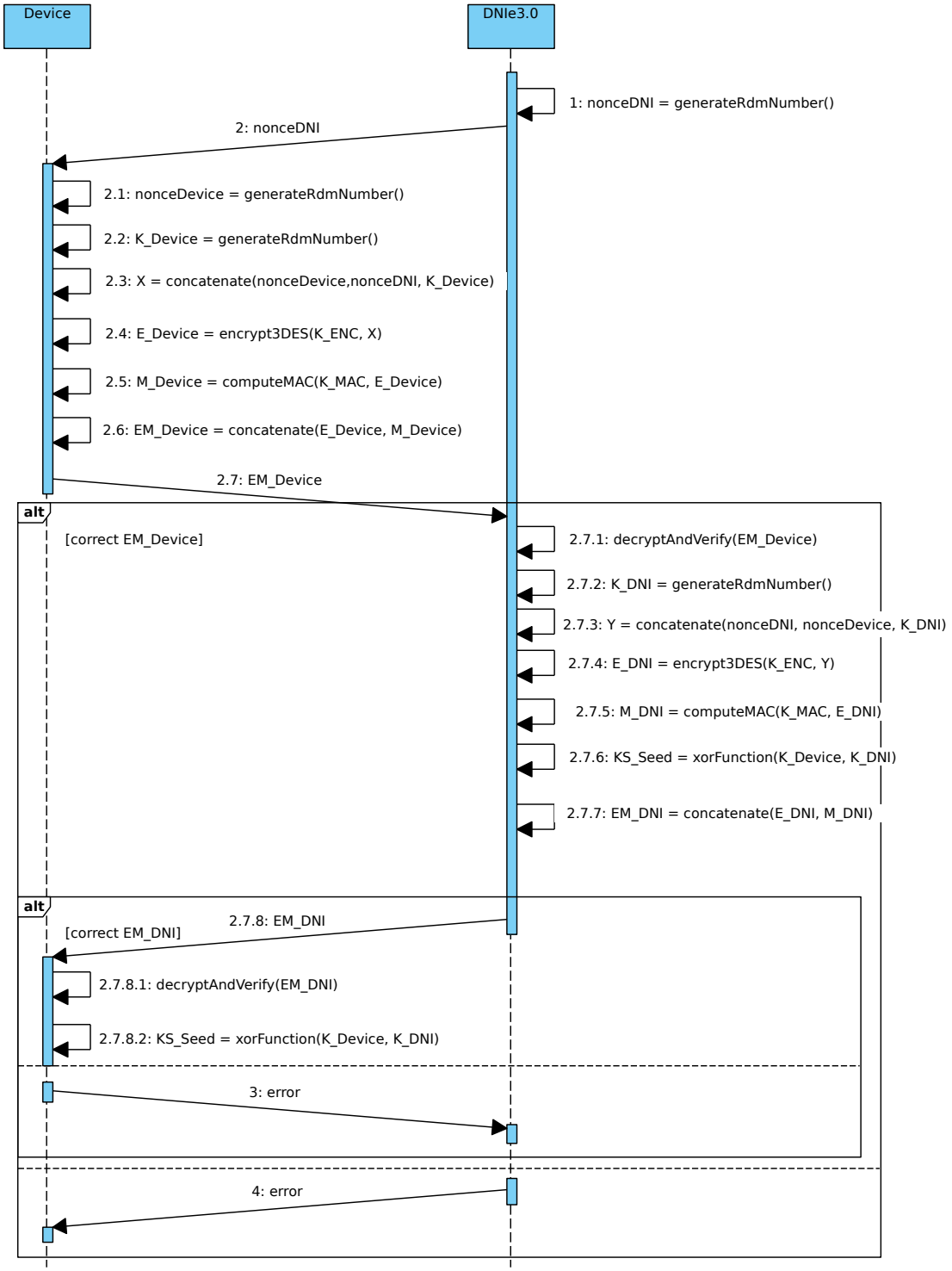


Figura 3.4: Diagrama de secuencia del protocolo BAC.

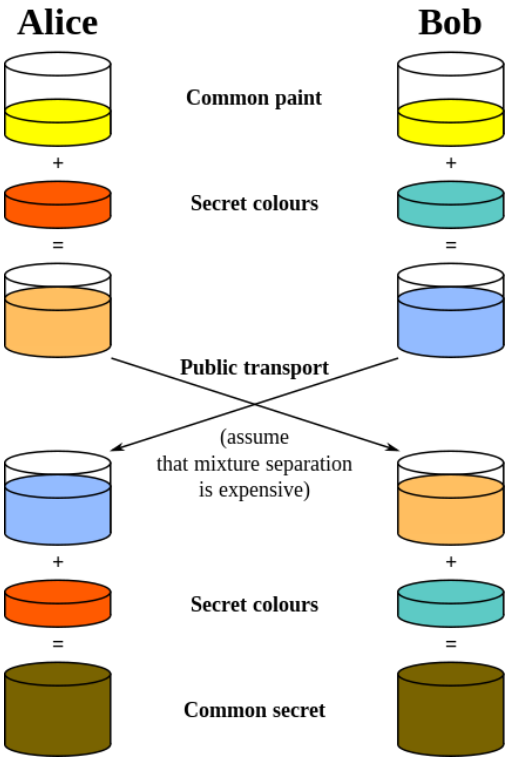


Figura 3.5: Explicación del protocolo Diffie-Hellman con colores (extraído de [Wik]).

pública para establecer una clave de sesión. En el caso del DNIe3.0, la parte pública del algoritmo se genera a partir del número aleatorio (*nonce*) generado por el documento.

Como ejemplo para explicar este protocolo abstrayéndose de la parte matemática, se puede ver este algoritmo de la siguiente manera (véase Figura 3.5): Alice y Bob inician el protocolo eligiendo un color arbitrario, que será conocido por ambos (pero deberá ser distinto cada vez que se utilice el protocolo), en este caso amarillo. Ahora cada uno elije un color secreto con el que mezclarán el amarillo. A continuación estos, se intercambian los colores mezclados, y mezclan este color con su color secreto. El resultado es que las dos partes acaban teniendo el mismo color final (marrón en este ejemplo), pero si una tercera parte ha estado escuchando la conversación, le resultará difícil a partir del color mezclado intercambiado saber cuáles son los colores secretos de cada interlocutor, por lo que no podrá obtener el color marrón final (de hecho, usando grandes números en vez de colores, no es posible para un ordenador descubrir la parte secreta de cada interlocutor en un tiempo razonable).

En la Figura 3.6 se muestra el diagrama de secuencia del protocolo PACE. En él se puede ver como a partir del código CAN se genera una clave derivada mediante SHA-1 que servirá para descifrar los mensajes que le llegarán desde el DNIe3.0. El documento, tras recibir una petición para empezar el protocolo, genera la clave derivada del CAN al

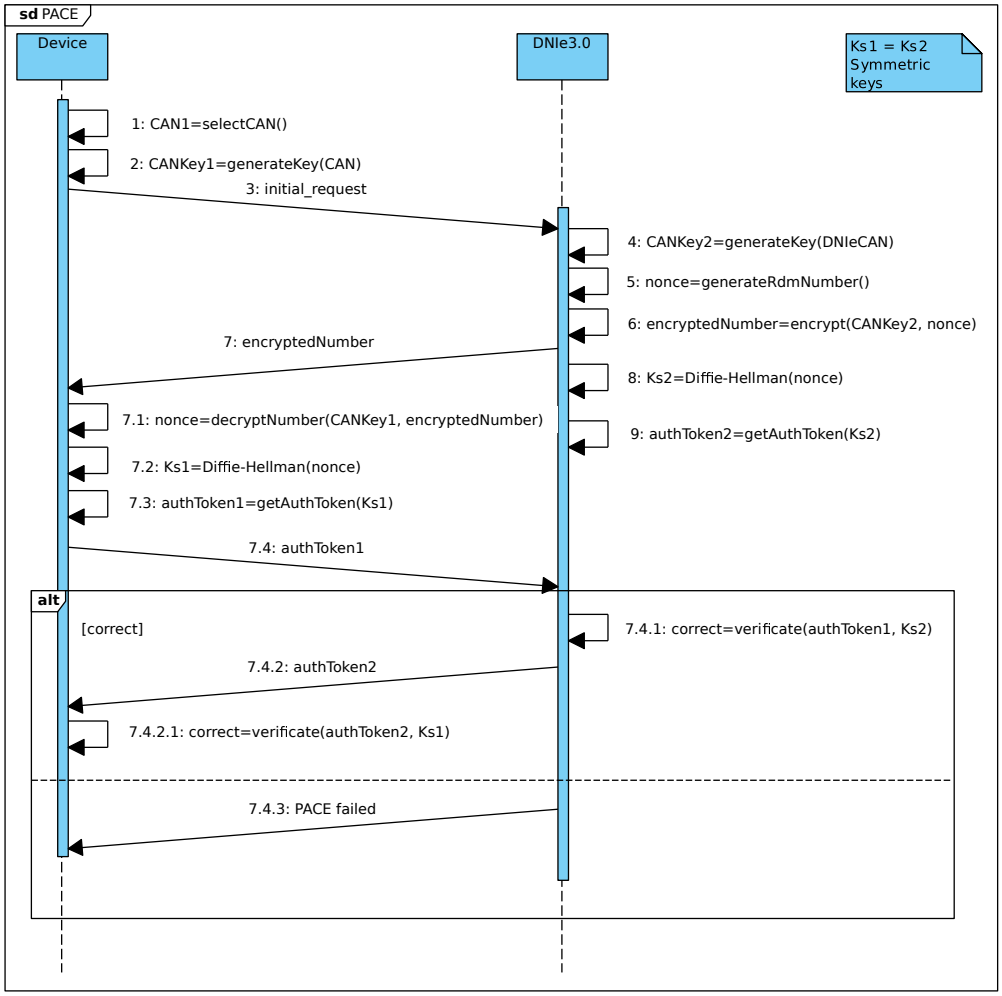


Figura 3.6: Diagrama de secuencia del protocolo PACE.

igual que el dispositivo, y genera un *nonce*. A continuación, el *nonce* es cifrado con la clave derivada del CAN y se envía al dispositivo, donde se descifra y se inicia el protocolo Diffie-Hellman, mientras se realiza la misma operación en el DNIe3.0. A partir de esta clave generada por ambos interlocutores, cada uno crea un *token* de autenticación y se lo envía al otro. En el caso de que el dispositivo haya introducido al inicio el CAN correcto, todo este proceso habrá generado un *token* de sesión que será validado por el DNIe3.0, y será aceptada la conexión. A partir de este momento se empieza a usar la clave simétrica de sesión generada para continuar con la comunicación.

El protocolo PACE es muy efectivo en el sentido de que a partir de una semilla de un rango con una cardinalidad muy baja (6 dígitos en este caso), es capaz de generar una clave de sesión de un rango con una cardinalidad muy alta (16 bytes en este caso).

3.3. Análisis de la cardinalidad del espacio de claves

A continuación se va a realizar un análisis para comprobar la cardinalidad de los dos tipos de código de seguridad que se usan en el acceso al DNIe3.0 descritos anteriormente.

Cardinalidad del código usado por BAC

Este análisis se va a realizar optimizando el espacio de búsqueda de las claves necesarias para establecer comunicación con el DNIe3.0 mediante BAC. Como punto de partida se va tomar el rango de edad desde los 14 hasta los 70 años, ya que antes de los 14 no es obligatorio poseer documento de identidad y a partir de los 70 ya no es necesario renovarlo, por lo que las probabilidades de encontrar el modelo 3.0 entre estos conjuntos de población es reducida (pocos menores de 14 años tienen DNI y a fecha de 2016 muchos mayores de 70 años se han quedado con versiones anteriores).

Los tres códigos necesarios para acceder a la información del DNIe3.0 a través del BAC, junto con la combinatoria de estos, son:

- *Fecha de nacimiento:* Según lo dicho anteriormente, la fecha de nacimiento a partir de la cual se deberá empezar a probar combinaciones es el 1 de enero de 1946, y hasta el 31 de diciembre del 2002 (no se tienen en cuenta años bisiestos). Esto deja las siguientes posibles combinaciones:

$$\begin{gathered} 1 \text{ año} = 365 \text{ días} \\ 2002 - 1946 = 57 \text{ años (incluyendo el año 1946)} \\ 365 \cdot 57 = 20805 \end{gathered}$$

- *Fecha de expiración del documento:* Se debe hacer un barrido desde el año 2016 (no se tienen en cuenta años bisiestos), ya que los ciudadanos que hayan renovado el DNI este año ya tendrán este nuevo modelo. El año 2015 no se añade al barrido debido a que el DNIe3.0 comenzó a expedirse oficialmente en toda España en diciembre de 2015, por lo que es probable que tan solo un pequeño porcentaje de la población renovara su documento de identidad en ese mes. La validez del DNI varía entre 5 y 10 años dependiendo de la edad del propietario: cinco años en el caso de ser menor de 30 años y diez en caso de ser mayor. Así, si el barrido comienza a partir de los 14 años hasta los 70 se debe aplicar la validez mayor. Por tanto, habría que comprobar fechas hasta el año 2026, éste no incluido. Esto da como posibles combinaciones:

$$365 \cdot 10 = 3650$$

- *Número de soporte:* Es una combinación de 3 caracteres alfabéticos y 6 numéricos (se ignora la letra "ñ"). Dado que el alfabeto (sin "ñ") tiene 26 posibles letras, y los dígitos están comprendidos entre 0 y 9, esto deja las siguientes posibles combinaciones:

$$26^3 \cdot 10^6$$

La combinación de estas tres secuencias genera las siguientes posibles combinaciones de claves BAC:

$$20805 + 26^3 \cdot 10^6 + 3650 \leq 2^{34}$$

Para hacerse una idea de lo grande que es este número, se estima que la edad del universo (según los últimos cálculos) se encuentra entre 2^{33} y 2^{34} años [Ló15].

Cardinalidad del código usado por PACE

El código CAN es una secuencia de 6 dígitos, por lo que la cardinalidad de su espacio de claves es mucho menor que el de BAC. La combinatoria es sencilla de obtener en este caso:

$$10^6$$

Debido a esta gran diferencia en las cardinalidades de los espacios de claves entre los dos protocolos, se decidió orientar la investigación sobre la posibilidad de un ataque de fuerza bruta hacia el protocolo PACE.

3.4. Vulnerabilidades estudiadas

En esta sección se van a exponer los distintos vectores de ataques estudiados, así como una explicación de por qué se descartó o se aceptó este vector como posible vulnerabilidad en el DNIe3.0.

3.4.1. Sniffing

Un ataque de *sniffing* se basa en la captura de la información durante una comunicación. El *sniffing* a través de la interfaz NFC ha quedado descartada debido a que todos los mensajes enviados entre un dispositivo y el DNIe3.0 están cifrados, por lo que aunque esta información fuese obtenida no podría ser utilizada. Si se interceptan los mensajes durante el establecimiento de la clave de sesión tanto el protocolo PACE como el protocolo BAC intercambian los mensajes de forma cifrada. Del mismo modo, si se interceptan después de haber establecido la clave de sesión los mensajes irán cifrados con dicha clave.

3.4.2. Resistencia a ataque de fuerza bruta

El ataque de fuerza bruta, aplicado al acceso mediante el protocolo PACE ha sido el más factible de los estudiados debido a la reducida cardinalidad del espacio de claves del código CAN (véase Sección 3.3). Esta idea se ha usado para implementar la prueba de concepto explicada en la Sección 4.2.

3.4.3. Distribución de los números aleatorios usados por el protocolo PACE

Como se ha comentado anteriormente, en una máquina determinista no es posible generar números perfectamente aleatorios, por lo que el chip del DNIe3.0 implementa un generador de números pseudoaleatorios. Por tanto, se ha realizado un análisis de la distribución de los números aleatorios usados para establecer comunicación mediante el protocolo PACE. Los resultados serán expuestos en la Sección 4.3.

Capítulo 4

Experimentación realizada

En este capítulo se van a explicar tanto las decisiones previas a la experimentación realizada en este TFG, así como una descripción de la prueba de concepto implementada. También se explica el análisis realizado sobre la distribución de los números aleatorios utilizados. Al final de cada sección se incluyen las conclusiones obtenidas.

4.1. Entorno de experimentación

4.1.1. Elección del sistema operativo y hardware

Para la realización de los experimentos y de la prueba de concepto se ha decidido usar `Android`, ya que el chip NFC de `iOS` solo permite ser usado para *Apple Pay* y `Windows Phone` tiene una cuota de mercado ínfima. Además, `Android` es más accesible, y se disponía tanto de tablet `Android` como del entorno de programación de licencia gratuita `Android Studio`. De cara a la prueba de concepto, se puede presuponer que en caso de que alguien quiera desarrollar una aplicación maliciosa optará por `Android` debido a que posee la mayor cuota de mercado [Cor15].

Además de estas razones, un último motivo que empujó al uso de `Android` en este proyecto fue la disponibilidad en forma de código abierto de un proyecto elaborado en Alemania que ofrece la base para el establecimiento del protocolo PACE. Además, la Policía Nacional ofrece otra aplicación para leer la información impresa en el DNIe3.0 a través de NFC con el protocolo PACE. Ambas aplicaciones estaban desarrolladas para `Android`, lo cual ahorraba muchas horas de programación.

Respecto al hardware utilizado, se ha trabajado con una tablet `SONY Xperia Z3 Tablet Compact`, que incorpora un procesador quad-core 2.5GHz `Qualcomm Snapdragon 801 MSM8974AC v3` y 3 GB de memoria RAM, e incorpora un chipset `Broadcomm` para la comunicación NFC.

4.1.2. Lenguajes de programación utilizados

Para la realización de los experimentos y pruebas de concepto se han usado los siguientes lenguajes de programación:

- *Android:* Para la implementación de las aplicaciones *DNIe3.0 Brute Force* y *DNIe3.0 Nonce Generator*, explicadas más adelante.
- *Java:* Para el análisis de la distribución de los números aleatorios utilizados por el protocolo PACE (explicado en la Sección 4.3). Se eligió Java porque con su librería *BigInteger* permite trabajar con enteros de 16 bytes sin desbordamientos.
- *Shell script:* Para el formateo de los ficheros generados por las aplicaciones creadas. Se eligió por ser un lenguaje no compilado que para una tarea tan simple era más rápido que otros.
- *Matlab:* Para crear las gráficas de este proyecto, ya que ofrece la posibilidad de tratar con enteros de gran tamaño y por la facilidad y calidad de las gráficas creadas.

4.2. Ataque de fuerza bruta contra PACE

Para implementar esta prueba de concepto se partió de una aplicación `Android` llamada AndroSmex[1], que proporciona una implementación básica de la conexión con el documento de identidad alemán mediante el protocolo PACE. Dado que el DNIe3.0 español sigue la misma implementación que el alemán, se ha podido adaptar esta aplicación sin modificarla demasiado.

Se analizaron tanto el protocolo PACE como el protocolo Diffie-Hellman que sucede durante el establecimiento de canal seguro. Como se muestra en el diagrama de secuencia de la Figura 3.6, todo el protocolo se puede resolver correctamente para resolver PACE si se conoce el CAN impreso en el anverso del documento (véase número 11 de la Figura 3.1).

A pesar de que todos los mensajes se cifran mediante una clave derivada con SHA-1 del código CAN, y aunque el protocolo Diffie-Hellman contenido dentro del protocolo de nivel superior PACE, encargado de generar las claves simétricas de sesión, aumenta en gran medida la cardinalidad del espacio de claves, este proceso se puede realizar correctamente si se sabe el código CAN, ya que el número aleatorio podrá ser descifrado correctamente, y resolver así el protocolo Diffie-Hellman, obteniendo la clave de sesión válida.

Implementación de la aplicación *DNIe3.0 Brute Force*

En la Figura 4.1 se muestra el diagrama de secuencia del ataque de fuerza bruta contra el protocolo PACE. Como se puede ver, se trata del mismo procedimiento para

[1] https://github.com/tsenger/androsmex

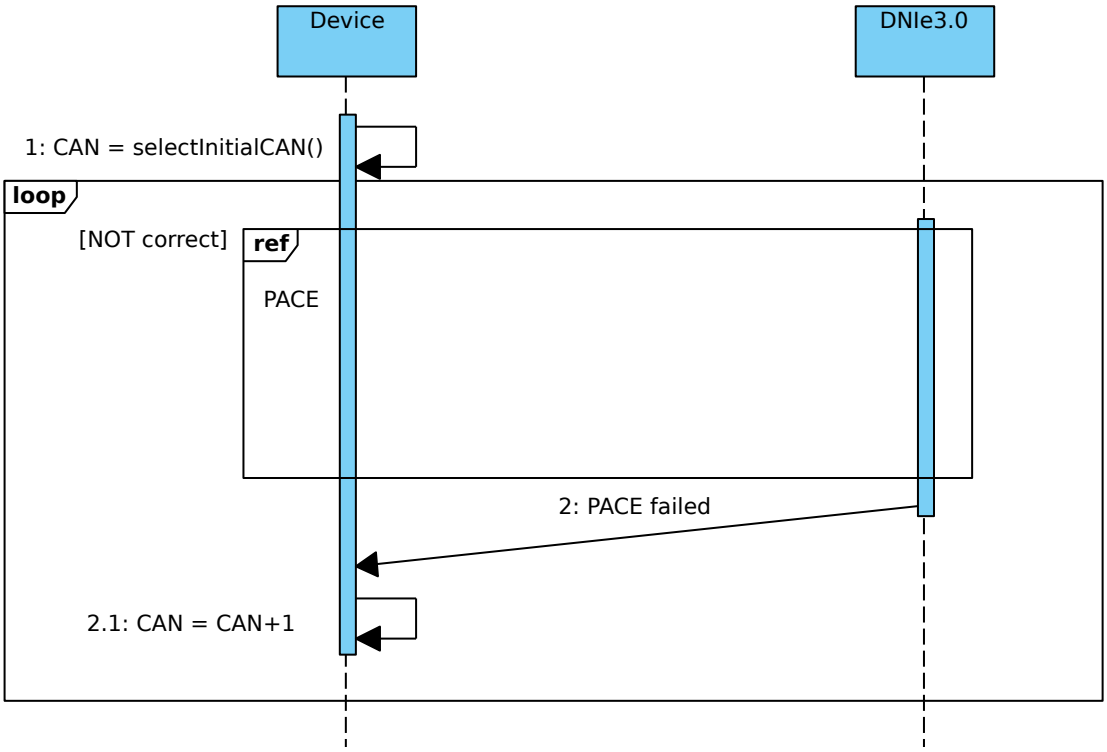


Figura 4.1: Diagrama de secuencia del ataque de fuerza bruta.

establecer conexión mediante el protocolo PACE explicado en la Sección 3.2.2, con la variante de que la selección del código CAN sólo se realiza de forma manual en la primera iteración del algoritmo. En las sucesivas iteraciones, si al llegar al final del protocolo el DNIe3.0 rechaza la conexión, la aplicación aumentará el código CAN y volverá a ejecutar el mismo proceso.

Conclusiones de experimentación

Tras la realización de las pruebas descritas anteriormente se llega a las siguientes conclusiones:

- Cada intento de establecimiento de conexión mediante el protocolo PACE cuesta 1,5 segundos en media (véase la Figura 4.2). Este tiempo se divide de la siguiente forma: 200 ms son usados para obtener y tratar el *nonce* generado por el documento, otros 1200 ms se utilizan para el protocolo Diffie-Hellman y 100 ms para generar y comprobar los *token* de sesión generados por ambas partes.

- El DNIe3.0 no cuenta con ningún tipo de defensa ante este tipo de ataques en el protocolo PACE, ya que como se observa en la gráfica de la Figura 4.2 el tiempo

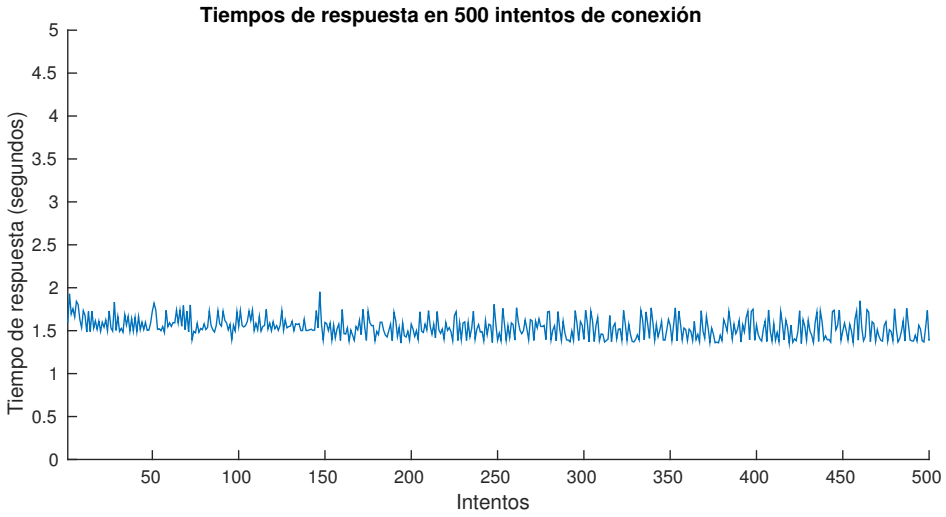


Figura 4.2: Gráfica tiempo/intento de 500 intentos consecutivos en un ataque de fuerza bruta.

de conexión tras intentos fallidos sucesivos no aumenta, ni en ningún momento el documento bloquea la comunicación.

- La cardinalidad del espacio de claves del código CAN es muy pequeño, tan sólo 10^6 posibles combinaciones.

Combinando estas tres características se puede calcular que en el peor caso, y suponiendo acceso ininterrumpido a la tarjeta, el tiempo total que requeriría este tipo de ataque sería de: $1{,}5 \cdot 10^6 \approx 17$ días (aproximadamente), lo cual no es un tiempo excesivamente alto.

Estos 17 días en el peor caso impide un ataque de forma directa al documento (es inviable estar con un dispositivo de lectura NFC durante 17 días cerca de una víctima), pero daría pie a, por ejemplo, aplicaciones maliciosas que finjan hacer algo atractivo para los usuarios, pero que en segundo plano cada vez que detecten un DNIe3.0 en alcance NFC, lo identifiquen y procedan con el ataque de fuerza bruta mientras lo tengan en rango, guardándose el último número probado al perderlo. Una situación donde este tipo de aplicación maliciosa podría actuar se muestra en la Figura 4.3, que muestra una cartera dejada encima de la mesa con el móvil encima.

Figura 4.3: Situación frecuente en la que una app maliciosa podría actuar.

4.3. Comprobación de la distribución de los números aleatorios utilizados por el protocolo PACE

Tal y como se ha descrito en la Sección 3.2.2, el protocolo PACE usa un número aleatorio durante la fase de establecimiento de clave de sesión. Ahora bien, si este número aleatorio no es generado de forma uniforme puede dar lugar a adivinar el CAN del DNIe3.0 de la víctima. Así, las posibilidades de éxito del ataque de fuerza bruta pasan por una cuestión estadística, cruzando la posibilidad de derivar la clave del CAN correcto junto con el número de *nonces* con una mayor probabilidad de aparecer.

Implementación de la aplicación *DNIe3.0 Nonce Generator*

Para comprobar la distribución de estos números aleatorios se ha realizado una variante de aplicación *DNIe3.0 Brute Force* implementada anteriormente. Se realizan intentos de establecimiento del protocolo PACE, pero cuando el DNIe3.0 envía el número aleatorio cifrado, éste se descifra con el CAN correcto (introducido previamente para obtener siempre los números correctamente descifrados) y a continuación se manda al documento un mensaje mal formado, provocando así un mensaje de error y la cancelación del establecimiento de conexión. En el dispositivo se captura este error y de forma recursiva se inicia un nuevo hilo de ejecución en segundo plano que repite este proceso (véase la Figura 4.4).

Una vez obtenida una colección suficientemente grande de *nonces*, 10^5 en este experimento, se ha optado por utilizar la técnica de *Delayed Coordinates* [Möl12] para comprobar su distribución. Con este proceso, cada *nonce* es comparado con el generado anteriormente, con el generado dos intentos atrás y con el generado tres intentos atrás, pudiendo así observar la distancia entre los distintos nonces generados.

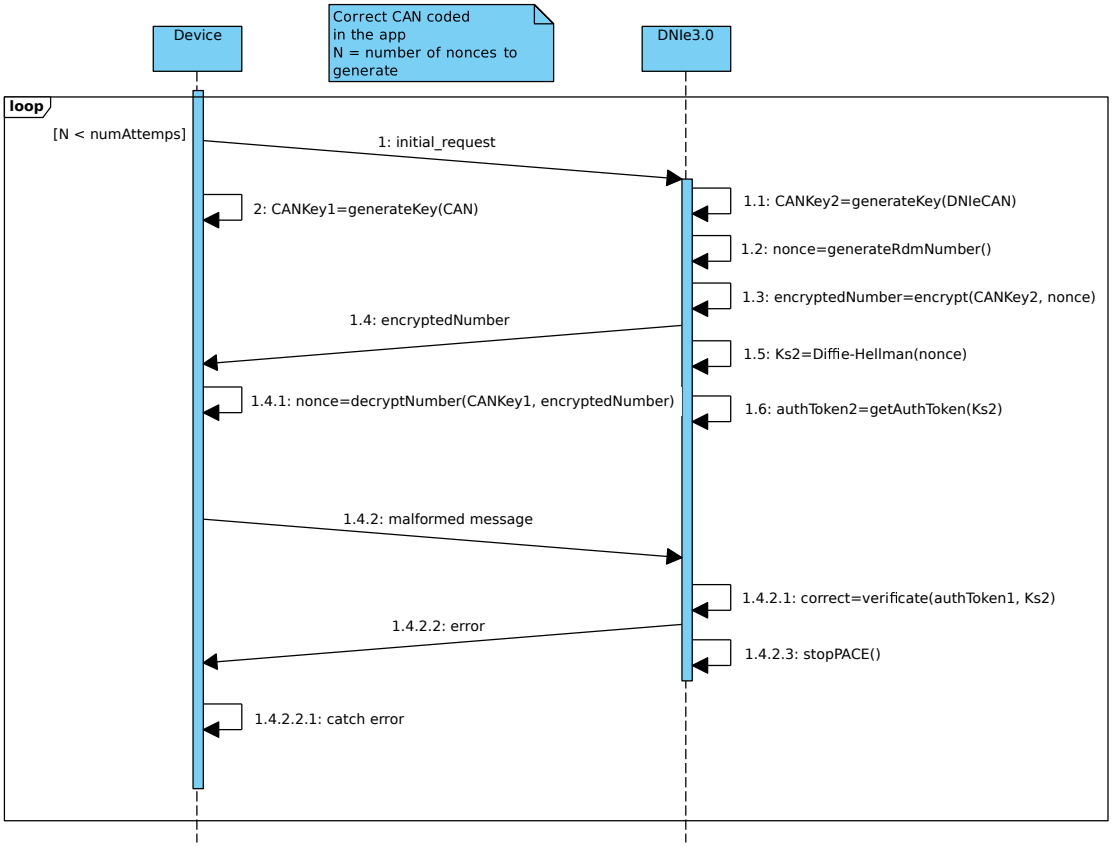
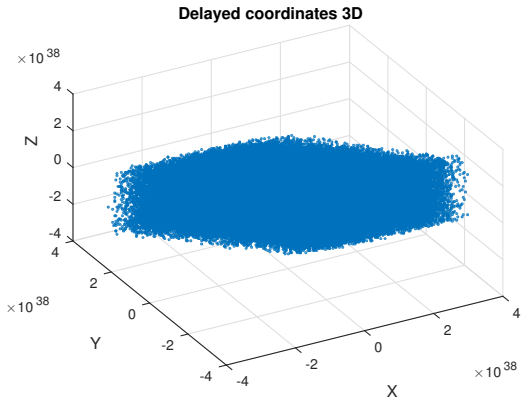
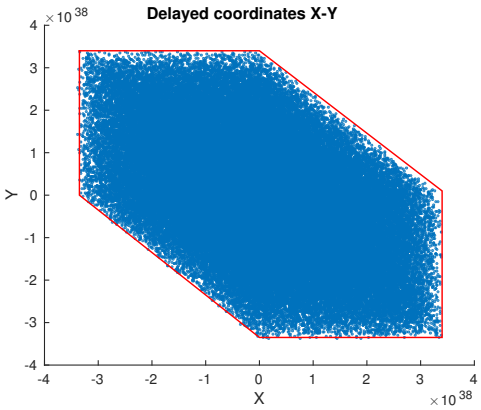


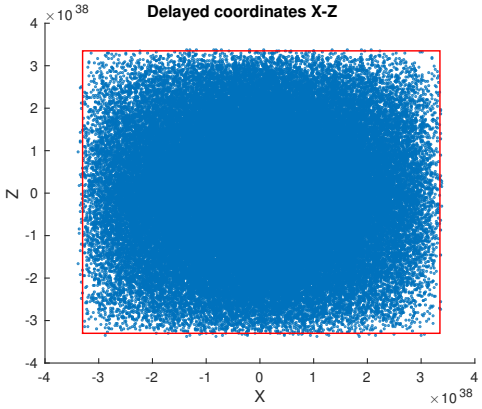
Figura 4.4: Diagrama de secuencia de la aplicación *DNIe3.0 Nonce Generator*.



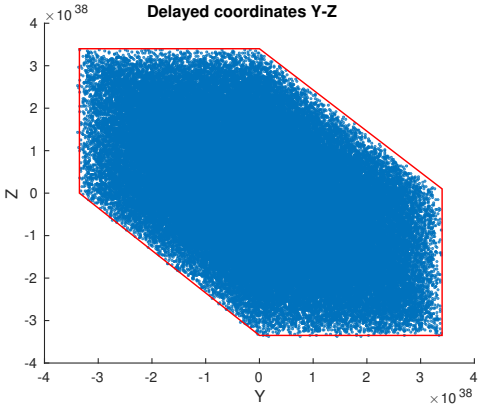
(a) *Delayed coordinates* 3D



(b) *Delayed coordinates* ejes X-Y



(c) *Delayed coordinates* ejes X-Z



(d) *Delayed coordinates* ejes Y-Z

Figura 4.5: Gráfica *Delayed Coordinates* en 3D y sus correspondientes vistas.

Partiendo de esta colección unidimensional de *nonces* $A = \{a(1); a(2); ...\}$ se pasa a una colección de *Delayed Coordinates* tridimensionales mediante la fórmula:

$$
\begin{aligned}
a_x(i) &= a(i) - a(i-1) \\
a_y(i) &= a(i-1) - a(i-2) \\
a_z(i) &= a(i-2) - a(i-3) \\
\forall i &\in [4; |A|] \cap \mathbb{N}
\end{aligned}
$$

La Figura 4.5a muestra la distribución de los *nonce* mediante *Delayed Coordinates* en 3D, mientras que en las Figuras 4.5b, 4.5c y 4.5d se muestran sus correspondientes vistas. Como se puede observar, los *nonces* generados por el DNIe3.0 están distribuidos uniformemente formando un paralelepípedo perfecto, lo que imposibilita la predicción del siguiente número generado.

Capítulo 5

Mejoras de seguridad

Este capítulo se basa en mejorar la robustez del protocolo PACE frente a los ataques de fuerza bruta. En concreto, se proponen tres posibles mejoras: el aumento de la cardinalidad del espacio de claves del código CAN, el aumento del tiempo de respuesta tras fallos sucesivos en el protocolo PACE y la incorporación de un bloqueo hardware para la interfaz NFC.

5.1. Aumento de la cardinalidad del espacio de claves del código CAN

Una posible solución al problema de los ataques de fuerza bruta consiste en el aumento de la cardinalidad del espacio de claves del código de acceso usado por el protocolo PACE. Esta técnica consiste en el aumento de la longitud del código CAN, ya que con simplemente un dígito más el tiempo para romperlo por fuerza bruta en el peor de los casos aumenta de 17 días a 47 años. Dado que la validez máxima de un DNI es de 10 años, imposibilitaría un ataque de fuerza bruta.

5.2. Aumento del tiempo de respuesta tras intentos sucesivos fallidos

Otra solución que no requeriría cambiar el diseño físico del documento sería la detección de intentos sucesivos fallidos de conexión con el DNIe3.0. Al cabo de un número predefinido de intentos el documento empieza a responder a los siguientes con un tiempo de retardo cada vez mayor. El pasaporte electrónico francés, como se muestra en [ABHC+16], dispone de este tipo de protección.

Supóngase que esta defensa se implementa aplicando la siguiente fórmula para obtener el tiempo de retardo en la respuesta en cada intento:

$$f(i) = \begin{cases} t(i) & \text{si } i \leq 5, \\ 1{,}1^i & \text{si } i \leq 15, \\ 15 & \text{si } i > 15 \end{cases}$$

donde $f(i)$ es el tiempo de respuesta (en segundos) en el intento de conexión i, y $t(i)$ es el tiempo de respuesta sin retardo en el intento de conexión i.

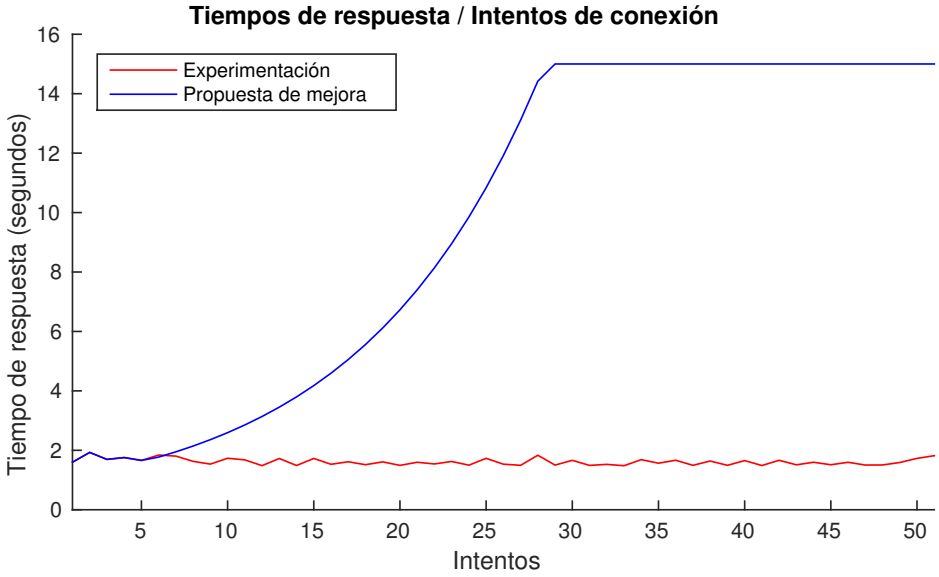


Figura 5.1: Gráfica tiempo de respuesta/intento en ataque de fuerza bruta.

Con esta solución se frustrarían los ataques de fuerza bruta, mientras que para un usuario verídico sería inapreciable aunque se equivocase al introducir el CAN. Los tiempos de respuesta frente a los intentos de conexión suponiendo la implementación de dicha función f se muestran en la Figura 5.1.

5.3. Mecanismo hardware de bloqueo NFC

Esta solución está inspirada en el botón de bloqueo de escritura que llevan algunas tarjetas de memoria SD (véase Figura 5.2), que impide su escritura de forma manual. Implantando el mismo mecanismo en una nueva versión del DNIe3.0 podría bloquearse manualmente las conexiones NFC con el documento. Cuando el ciudadano necesitase de su uso, simplemente tendría que habilitar esta funcionalidad.

Figura 5.2: Bloqueo manual en tarjeta de memoria SD.

Esta solución, al igual que la extensión del código CAN en la parte impresa, necesitaría de un rediseño del documento físico y de una concienciación ciudadana para su uso correcto.

Capítulo 6

Estado del arte

Existen artículos en Europa sobre esta nueva interfaz NFC dentro de los documentos de identidad y pasaportes electrónicos de países que los implantaron antes que España, como por ejemplo Alemania, Bélgica o Francia [ABHC+16, fSidI15, LKLRP07]. La mayoría de estos trabajos analizaban el método de acceso mediante el protocolo BAC, pocos se centraban en el protocolo de acceso PACE, aunque el problema de la reducida cardinalidad en el espacio de claves del código CAN ya había sido destacado también en [BK09].

Respecto a las investigaciones desarrolladas sobre el DNIe3.0 español, no se ha encontrado ninguna. La única documentación institucional recopilada ha sido una ficha técnica con las características del DNIe3.0 [Nacb] (pero sin detalles de implementación), una guía de referencia [Nac15] con una explicación algo más extensa de sus características y una descripción de las aplicaciones demo en `Android` desarrolladas por la Policía Nacional [Naca] (en las cuales desgraciadamente no se explica su implementación). En cuanto a artículos de prensa la mayor parte de lo que hay publicado habla sobre la polémica que se formó cuando el ministro del Interior anunció el nuevo DNIe3.0 y erró en sus palabras diciendo que posibilitaría la identificación de ciudadanos a distancia [Con15]. En [Wor15] se habla sobre este asunto, aunque sin aportar documentación que avalase estas palabras.

Como se ha explicado, la Policía Nacional ofrece tres aplicaciones `Android` [Naca] para desarrolladores, en las cuales se implementan una aplicación de lectura del DNI, otra de autenticación y otra de firma digital. No obstante, la implementación de estas aplicaciones no es fácil de entender por la falta de comentarios en el código.

En el campo de la seguridad relativa a la nueva interfaz NFC del DNIe3.0 español no se ha encontrado tampoco ningún tipo de documento. Sin embargo, en otros países europeos como Alemania, Francia o Bélgica sí que se han encontrado investigaciones similares sobre sus documentos de identificación y/o pasaportes electrónicos [LKLRP07], [fSidI15].

Este proyecto ha abarcado la investigación de los elementos de seguridad añadidos a la nueva interfaz NFC del DNIe3.0, inexistente hasta el momento.

Capítulo 7

Conclusiones y trabajo futuro

En este capítulo se exponen las conclusiones sobre el proyecto realizado y se plantean las líneas de futuro que podrían tomarse para continuarlo.

7.1. Conclusiones

Este proyecto ha comprobado que, en general, la implementación de la interfaz NFC del DNIe3.0 español es segura. Sin embargo, se ha observado que el DNIe3.0 no implementa ningún mecanismo de defensa frente a posibles ataques de fuerza bruta al código de acceso solicitado por el protocolo PACE.

También es importante resaltar que debido a la envergadura de un TFG, esta investigación no ha llegado a profundizar tanto como podría ser posible, ya que el protocolo BAC ha sido estudiado sólo teóricamente, y no se ha profundizado en la seguridad de los protocolos criptográficos.

A título personal, me ha encantado realizar este proyecto ya que he podido ver y aprender sobre cómo se desarrolla un proyecto de investigación, y más concretamente sobre una investigación en seguridad informática, el área en la que me gustaría especializarme en un futuro.

7.2. Líneas futuras de investigación

Debido a la envergadura de un TFG y a todas las vías de investigación posibles se ha abarcado solo una pequeña parte del tema.

En el futuro, sería interesante estudiar otras posibles vulnerabilidades del DNIe3.0 frente a ataques más elaborados. Por ejemplo, realizar un ataque de denegación de servicio saturando la red para impedir la comunicación entre las partes legítimas. Otro ataque a estudiar sería el ataque de retransmisión para firmar digitalmente documentos con el DNIe3.0 de una víctima, una vez conseguido su CAN.

Bibliografía

[ABC13] ABC. Vida y origen del DNI, 2013. http://www.abc.es/espana/20130619/abci-vida-historia-estos-fueron-201306181259.html.

[ABHC+16] Gildas Avoine, Antonin Beaujeant, Julio Hernandez-Castro, Louis Demay, and Philippe Teuwen. A Survey of Security and Privacy Issues in the ePassport Protocols. *ACM Computing Surveys*, 2016.

[BK09] Jens Bender and Dennis Kügler. Introducing the PACE solution. *Keesing Journal of Documents & Identity*, 30:26–29, 2009.

[Con15] El Confidencial. ¿Identificar a distancia con el nuevo DNI 3.0?, 2015. http://www.elconfidencial.com/tecnologia/2015-01-16/identificar-a-distancia-con-el-nuevo-dni-3-0-la-n-de-nfc-significa-near_623075/.

[Cor15] International Data Corporation. Smartphone OS Market Share, Q2 2015, 2015. http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[DH76] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, Nov 1976.

[fS76] International Organization for Standardization. *1073-2:1976: Alphanumeric character sets for optical recognition – Part 2: Character set OCR-B – Shapes and dimensions of the printed image*, December 1976. http://www.iso.org/iso/catalogue_detail?csnumber=5568.

[fS03] International Organization for Standardization. *ISO/IEC 7810:2003: Identification cards – Physical characteristics*, November 2003. http://www.iso.org/iso/catalogue_detail?csnumber=31432.

[fS16] International Organization for Standardization. Iso/iec 1443:2016: Identification cards – contactless integrated circuit cards – proximity cards, March 2016. http://www.iso.org/iso/home/search.htm?qt=14443&sort=rel&type=simple&published=on.

[fSidI15] Bundesamt fü Sicherheit in der Informationstechnik. *Advanced Security Mechanisms for Machine Readable Travel Document and eIDAS Token - Part 2*, 2015. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TR03110/BSI_TR-03110_Part-2-V2_2.pdf?__blob=publicationFile.

[GKGM+08] Flavio D. Garcia, Gerhard Koning Gans, Ruben Muijrers, Peter Rossum, Roel Verdult, Ronny Wichers Schreur, and Bart Jacobs. Dismantling MIFARE Classic. In *Proceedings of the 13th European Symposium on Research in Computer Security: Computer Security*, ESORICS '08, pages 97–114, Berlin, Heidelberg, 2008. Springer-Verlag.

[ICA15] ICAO. *Machine Readable Travel Documents - Part 11: Security Mechanisms for MRTDs*, 2015. http://www.icao.int/publications/Documents/9303_p11_cons_en.pdf.

[Int13] International Organization for Standardization. *ISO/IEC 18092:2013: Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol (NFCIP-1)*. Geneva, Switzerland, March 2013.

[Jap10] Japanese Industrial Standard. *JIS X 6319-4:2010: Specification of implementation for integrated circuit(s) cards – Part 4: High speed proximity cards*, October 2010. http://www.webstore.jsa.or.jp/webstore/PrevPdfServlet?dc=JIS&fn=pre_jis_x_06319_004_000_2010_e_ed10_i4.pdf.

[Leg14] Legalitas.com. El observatorio legálitas alerta del aumento de los casos de robo de identidad en España, 2014. https://www.legalitas.com/actualidad/El-Observatorio-Legalitas-alerta-del-aumento-de-los-casos-de-.

[LKLRP07] Yifei Liu, Timo Kasper, Kerstin Lemke-Rust, and Christof Paar. *E-Passport: Cracking Basic Access Control Keys*, pages 1531–1547. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[Ló15] César Tomé López. De la edad y tamaño del universo. 2015. http://culturacientifica.com/2015/01/13/de-la-edad-y-tamano-del-universo/.

[Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, April 1978.

[Möl12] Frederik Möllers. An analysis of traceability of electronic identification documents. Master's thesis, Universität Paderborn, 2012. https://www.uni-saarland.de/fileadmin/user_upload/Professoren/fr11_ProfSorge/Paper-Downloads/master-fred.pdf.

[Naca] Policía Nacional. Aplicaciones demo y documentación. http://www.dnielectronico.es/PortalDNIe/PRF1_Cons02.action?pag=REF_1120.

[Nacb] Policía Nacional. *Características Técnicas DNIe 3.0*. http://www.dnielectronico.es/PDFs/CARACTERISTICAS%20TECNICAS%20DNIe%203.0.pdf.

[Nacc] Policía Nacional. *Descripción DNIe3.0*. http://www.dnielectronico.es/PortalDNIe/PRF1_Cons02.action?pag=REF_103.

[Nac15] Policía Nacional. *Guía de Referencia del DNIe con NFC*, 2015. http://www.dnielectronico.es/PDFs/Guia_de_Referencia_DNIe_con_NFC.pdf.

[OMG11] OMG. *Unified Modelling Language: Superstructure*. Object Management Group, August 2011. Version 2.4, formal/11-08-05.

[SER15] Cadena SER. El DNI 3.0, disponible ya en toda España, 2015. http://cadenaser.com/ser/2015/12/18/ciencia/1450455277_492010.html.

[VR15] José Vila and Ricardo J. Rodríguez. Practical experiences on NFC Relay Attacks with Android: Virtual Pickpocketing Revisited. In *Proceedings of the 11th International Workshop on RFID Security (RFIDsec)*, volume 9440 of *Lecture Notes in Computer Science*, pages 87–103. Springer, 2015.

[Wik] Wikipedia. Diffie-hellman key exchange. https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange.

[Wor15] Security Art Work. Identificación a distancia... DNI 3.0. 2015. http://www.securityartwork.es/2015/09/03/identificacion-a-distancia-dni-3-0.

Apéndice A

Extensión temporal del proyecto

La distribución temporal de este proyecto ha sido más dispersa de lo normal debido a que durante el segundo cuatrimestre estuve trabajando con un contrato de prácticas a media jornada que se prolongó hasta finales de mayo, que junto con las asignaturas de dicho cuatrimestre restaban muchas horas al día, por lo que mi dedicación diaria al TFG durante este periodo no superó las 4 horas diarias.

Hablando ya de la organización del proyecto, el estudio de documentación se ha prolongado desde que se empezaron a estudiar los primeros artículos hasta que ha concluido el trabajo práctico. Esto se debe a que según se iban produciendo las reuniones con el director de proyecto e iban descartándose las diferentes propuestas para intentar encontrar alguna posible vulnerabilidad en el DNIe3.0 se hacía necesario replantear lo que se estaba haciendo y lo que se podría hacer en el futuro.

Una vez se tenía suficiente información teórica como para llevar a cabo el siguiente paso en la experimentación, se procedía a analizar ese aspecto en la aplicación AndroSmex para poder llevar a cabo las pruebas de concepto. En julio se terminó con la experimentación y se empezó con la propuesta de mejoras para la defensa del DNIe3.0, todas ellas analizadas de forma teórica, y se empezó a desarrollar la memoria, que se alargó hasta agosto pasando por varias correcciones del director de proyecto.

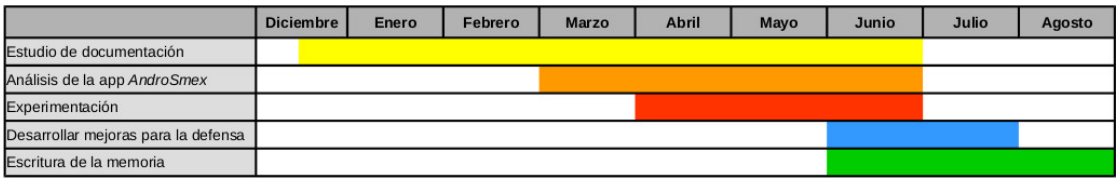


Figura A.1: Diagrama de Gantt.

Apéndice B

Código fuente de la clase *Main* de *DNIe3.0 Brute Force*

```
1  package app.unizar.dnie3_brute_force;
2
3  import ...
4
5  public class MainActivity extends Activity implements
   ↪  NfcAdapter.ReaderCallback, AsyncResponse {
6
7      private static final Logger asLogger =
       ↪  Logger.getLogger("DNIe3.0_brute_force");
8      public static Tag discoveredTag = null;
9      // NFC Adapter
10     static private NfcAdapter myNfcAdapter = null;
11     private static FileHandler fh;    //Guardar logs en fichero
12     private static String canNumber = "0";
13     private static TextView textView = null;
14     private static Tag tagFromIntent = null;
15     private Activity myActivity;
16     private boolean readerModeON = false;
17     private int attemps = 1;
18     private long beginTime, endTime, totalTime;
19
20     @Override
21     protected void onCreate(Bundle savedInstanceState) {
22         super.onCreate(savedInstanceState);
23         setContentView(R.layout.activity_main);
24
25         Context myContext = MainActivity.this;
```

```
26          myActivity = ((Activity) myContext);
27
28          // Obtenemos el adaptador NFC
29          myNfcAdapter = NfcAdapter.getDefaultAdapter(this);
30          myNfcAdapter.setNdefPushMessage(null, this);
31          myNfcAdapter.setNdefPushMessageCallback(null, this);
32
33          //Mostrar logs en TextView
34          textView = (TextView) findViewById(R.id.textView);
35          textView.setMovementMethod(new ScrollingMovementMethod());
36          textView.setTypeface(Typeface.MONOSPACE);
37          TextViewHandler handler = new TextViewHandler(this, textView);
38          asLogger.addHandler(handler);
39          asLogger.setLevel(Level.parse("ALL"));
40
41
42      }
43
44      /**
45       * Al pulsar en el botón se llamara a esta función, que empezará el
46       * ataque de fuerza bruta a partir del número introducido en el
47       * campo 'editText' o por 0 si no se ha escrito nada.
48       */
49      public void onClickStartPACE(View v) {
50          asLogger.log(Level.ALL, "START\n");
51          TextView pinText = (TextView) findViewById(R.id.editText);
52          canNumber = pinText.getText().toString();
53          if (canNumber.equals("")) {
54              canNumber = "0";
55          }
56
57          beginTime = System.currentTimeMillis();
58          DGLoader dgl = new DGLoader(asLogger, tagFromIntent, canNumber,
            ↪  getApplicationContext());
59          dgl.delegate = this;
60          dgl.execute((Void[]) null);
61      }
62
63      @Override
64      public void onResume() {
65          super.onResume();
66          if (!readerModeON)
67              readerModeON = EnableReaderMode(1000);
```

```
68      }
69
70      private boolean EnableReaderMode(int msDelay) {
71          // Ponemos en msDelay milisegundos el tiempo de espera para
            ↪  comprobar presencia de lectores NFC
72          Bundle options = new Bundle();
73          options.putInt(NfcAdapter.EXTRA_READER_PRESENCE_CHECK_DELAY,
            ↪  msDelay);
74          myNfcAdapter.enableReaderMode(myActivity,
75                  this,
76                  NfcAdapter.FLAG_READER_SKIP_NDEF_CHECK |
77                          NfcAdapter.FLAG_READER_NFC_B,
78                  options);
79          return true;
80      }
81
82      private boolean DisableReaderMode() {
83          // Desactivamos el modo reader de NFC
84          myNfcAdapter.disableReaderMode(this);
85          readerModeON = false;
86          return true;
87      }
88
89      /**
90       * Se llamará a la función si se encuentra un tag NFC.
91       */
92      @Override
93      public void onTagDiscovered(Tag tag) {
94          try {
95              tagFromIntent = tag;
96
97              asLogger.log(Level.ALL, "TAG encontrado");
98
99          } catch (Exception e) {
100             asLogger.log(Level.ALL, "Ocurrió un error durante la lectura
                ↪  de ficheros.\n" + e.getMessage());
101         }
102     }
103
104     @Override
105     public boolean onCreateOptionsMenu(Menu menu) {
106         // Inflate the menu; this adds items to the action bar if it is
            ↪  present.
```

```
107         getMenuInflater().inflate(R.menu.menu_main, menu);
108         return true;
109     }
110 
111     @Override
112     public boolean onOptionsItemSelected(MenuItem item) {
113         // Handle action bar item clicks here. The action bar will
114         // automatically handle clicks on the Home/Up button, so long
115         // as you specify a parent activity in AndroidManifest.xml.
116         int id = item.getItemId();
117 
118         //noinspection SimplifiableIfStatement
119         if (id == R.id.action_settings) {
120             return true;
121         }
122 
123         return super.onOptionsItemSelected(item);
124     }
125 
126     protected void onDestroy() {
127         super.onDestroy();
128     }
129 
130     /**
131      * Esta función trae la ejecución en segundo plano a este hilo.
132      * 'output' contiene el resultado de la ejecucion
133      */
134     @Override
135     public void processFinish(String output) {
136         if (!output.contains("correcto")) {
137             asLogger.log(Level.ALL, "CAN " + canNumber + "
                ↪  incorrecto.");
138             asLogger.log(Level.ALL, "Tiempo utilizado: " + output);
139             int nextCAN = Integer.parseInt(canNumber) + 1;
140             attemps++;
141             canNumber = String.valueOf(nextCAN);
142             asLogger.log(Level.ALL, "\nSiguiente intento con CAN: " +
                ↪  canNumber);
143             DGLoader dgl = new DGLoader(asLogger, tagFromIntent,
                ↪  canNumber, getApplicationContext());
144             dgl.delegate = this;
145             dgl.execute((Void[]) null);
146         } else {
```

```
147             endTime = System.currentTimeMillis();
148             totalTime = (endTime - beginTime) / 1000;
149             asLogger.log(Level.ALL,
                ↪  "\n=================================================================\n"
                ↪  +
150                     "ATAQUE DE FUERZA BRUTA COMPLETADO CON ÉXITO\n" +
151                     " -Intentos necesitados: " + attemps + ".\n" +
152                     " -Tiempo total utilizado: " + totalTime + "
                        ↪  segundos." +
153
                        ↪  "\n=================================================================
154         }
155     }
156
157 }
```