0.1. Introducción

Anteriormente se han explicado los conocimientos y términos generales relacionados con la seguridad en la autenticidad NFC basada en criptología con curvas elípticas. A continuación se muestra la aplicación de dichos conocimientos en un proyecto software experimental. El objetivo es demostrar la capacidad de los dispositivos NFC para, de la mano de la criptografía con curvas elípticas, llegar a desarrollar un sistema de autenticación óptimo y fiable.

Tanto los nombres, librerías, herramientas, así como el resto del material utilizado se describirán a continuación. A su vez, siguiendo un estándar de desarrollo software basado en metodologías ágiles, se mostrará la utilizada para éste proyecto. Para ello, el autor y director de este TFG (Fidel Abascal y Domingo Gómez) hemos actuado y ejercido tanto de cliente como de contratado para el desarrollo de la aplicación.

Dentro de un ámbito ficticio, una empresa llamada **Alpha - Consultora S.A.**[*], nueva potencia local dentro del campo de la seguridad bancaria, que ha cosechado unos excelentes resultados a lo largo de sus 2 años de existencia. Cuenta con más de 40 trabajadores y su crecimiento y expansión es notoria. Tanto es el éxito de esta empresa que, para dar cabida a su plantilla, ha decidido trasladarse a una nueva sede más moderna, amplia y mejor ubicada. La empresa, antes de instalarse en la nueva sede, decide contratar a unos expertos en seguridad para gestionar el control de accesos mediante un sistema de tarjetas y lectores en las entradas; teniendo en cuenta una inversión mínima pero garantizando un alto grado de seguridad.

Tras buscar incesantemente recurren a la empresa **F-NFC**. Una vez realizado el estudio por parte de F-NFC, se pone en consonancia un acuerdo para elaborar una aplicación que genere información que autentifique a un usuario de la empresa *Alpha* y sea reconocido de forma unívoca para permitir su acceso a la sede. La infraestructura de la empresa propietaria de la nueva sede corresponde a la figura 0.1.

Figura 1: Infraestructura del sistema ficticio.

Los detalles de implementación de la aplicación seguirán un objetivo didáctico y experimental que cumplirán los requerimientos de la situación propuesta; adecuándose a la carencia real de la infraestructura anteriormente mencionada. Se explicará en las secciones siguientes en detalle todos los puntos implicados en la consecución de este objetivo. Más concretamente en el apartado de escenario de la aplicación 0.4.

0.2. Material utilizado

Para el desarrollo del proyecto se ha utilizado, ante todo, software de carácter libre junto a imágenes, textos, estructuras y contenido sin restricciones de uso. Ya sea debido al tipo de licencia de cada elemento o por ser de autoría propia.

[*] Cualquier similitud con la realidad es mera coincidencia

El desarrollo se ha realizado en el lenguaje de programación *Java* con el kit de desarrollo *Java* (JDK) 1.8.65 (*Java 8 update 65*) de *Oracle Corporation* [**?**].

El entorno de desarrollo integrado o *IDE* de la aplicación es el oficial de Google para el desarrollo para dispositivos *Android*: *Android Studio*, versión 1.5.1 [**?**]. Este programa provee las herramientas básicas de desarrollo; entre ellas, un administrador de los kit de desarrollo o *SDKs* para las diferentes versiones y varios elementos descritos a continuación:

- Plataformas SDK
  - API 22: Para la versión Android 5.1.1.
  - API 21: Para la versión Android 5.0.1.
- Herramientas SDK
  - Android Build Tools : Para la construcción de la aplicación.
  - Android SDK Platform Tools v23.1: Soporte para el desarrollo.
  - Repositorio de ayuda Android, rev 30.
  - Libreria de ayuda Android, rev 23.2.1 : Ayudas en la retrocompatibilidad de elementos de interfaz de usuario.
  - Google USB Driver, rev 11 : Conexión entre el servicio de ejecución de la aplicación y los dispositivos USB.
  - Intel x86 HAXM, rev 6.0.1 : Aceleración *hardware* para la emulación.

Estos componentes de *Android Studio* hacen posible el desarrollo de la aplicación objetivo. Este programa, con carencias notorias en ciertos apartados, ha hecho complicado, de cierta forma, la selección de componentes a instalar, versiones y etcétera debido a diversos motivos de compatibilidades de librerías y actualizaciones desfasadas entre los componentes y el propio IDE.

La esencia de la aplicación reside en la criptografía mediante curvas elípticas. Para la implementación de ello ha sido necesario utilizar una librería externa llamada *Bouncy Castle* [**?**] versión 1.54. Dicha librería suple las deficiencias de la implementación base de la propia API (*Application Programming Interface*) de *Java*. La cual tiene limitaciones claras a la hora de la generar curvas elípticas. Gracias a ésta librería se ha podido realizar la parte crítica de la aplicación con una mejora de rendimiento notoria si hubiera que haber utilizado la API de *Java*.

Respecto al almacenaje de datos se ha optado por utilizar una pequeña base de datos en *SQLite* [**?**] descrita más adelante. Para el almacenamiento de unos pequeños registros que utilizará la aplicación. El uso de una base de datos de mayor capacidad y funcionalidades no se ha contemplado factible.

En cuanto a los elementos gráficos de la aplicación, un alto porcentaje son de elaboración propia mediante programas de edición de imágenes. El resto son de libre uso comercial.

La iconografía de la aplicación es autoría de *Google Inc.* [**?**]. Dichos iconos han de ser vectoriales debido a la optimización del tratamiento de imágenes y su renderizado, por lo que en este aspecto, *Google* provee estos iconos en formato *SVG* y *PNG*; también *Android Studio* de forma nativa pero no actualizada. A la hora de incluir estos elementos externos se ha transformado las descripciones vectoriales *SVG* (*Scalable Vector Graphics*) en formato *XML* (*eXtensible Markup Language*) interpretable de forma sencilla por *Android Studio*. En el apartado de diseño y aspecto de la aplicación 0.5 se comenta en detalle el resto de la disposición, motivación y elaboración gráfica de la aplicación.

La API de objetivo del proyecto *Android* ha sido la número 22. Desarrollada para la versión 5.1. (*LOLLIPOP MR1*). Se ha decidido utilizar esta esta API debido a las mejoras sustanciales en cuanto al trabajo del adaptador NFC implementadas desde la versión 5.0 [**?**] y mejoradas en ésta [**?**].

Para el testado de la aplicación *Android Studio* dispone de la tecnología *AVD* para la virtualización de dispositivos *Android*. Sin embargo, el rendimiento es pobre en comparación con el testeo y *debug* en un dispositivo físico. Por lo que se ha utilizado un dispositivo *Android smartphone One Plus One* de la compañía americana *Never Settle*, el cuál dispone de la versión *Android* 5.1.1 y *Cyanogen OS* 12.1.1 en el momento de la elaboración de la aplicación.

Figura 2: Smartphone One Plus One - Never Settle

Por último, el proyecto ha necesitado de dos elementos físicos principales: el dispositivo *Android* mencionado anteriormente y de tarjetas NFC. Debido a la carencia de un presupuesto, se ha optado por utilizar etiquetas adesibles (desde ahora NFC-T) de bajo coste por etiqueta. Se trata del chip *NTAG213* que siguen el estándar ISO 14443-3 [**?**]. Las características de estas etiquetas son las siguientes:

- Tipo de etiqueta : ISO 14443-3A
- Descripción : NXP MIFARE Ultralight (Ultralight C) - NTAG213
- Tecnología : NfcA, Ndef, MifareUltralight
- Formato de datos: NFC Forum Type 2
- Diámetro: 25 mm
- Identificadores del chip
  - Valor ATQA: 0x0044
  - Valor SAK: 0x00
  - Firma: NXP Public Key
- Tamaños y capacidad
  - Memoria: 45 páginas de 4bytes por página (180 bytes)

- Tamaño: 137 bytes
- *UID* (Identificador del contenido): 7 bytes [**?**]
  - *Byte 7*: Valor *UTF-8*, codificación
  - *Byte 6*: Valor *0*, reservado para uso futuro
  - *Byte 5-0*: Tamaño del código del lenguaje *IANA*
- Tamaño utilizable: 130 bytes (Tamaño menos *UID*)

Estas etiquetas se pueden adherir en una superficie que le haga de soporte. Por ejemplo, se podría plastificar junto a dos tapas que cubran el chip y tener la apariencia de una tarjeta común. Su apariencia es la siguiente:

Figura 3: NFC Tag - NTAG213

Hubiera sido preferible utilizar chips *MIFARE* [**?**] de alta capacidad y mayores funcionalidades como los *MIFARE classic* 1K y 4K del productor *NXP*. Estos chips están implementados en una enorme cantidad de aplicaciones en todo el mundo, un ejemplo claro de su uso son el de las tarjetas de crédito y la mayoría de chips NFC actuales que impliquen la necesidad de encriptación (RSA comúnmente) y seguridad. En estas tarjetas es realmente complicado acceder a su contenido ya que se precisan ciertas claves para la lectura y decodificación. Sin embargo, en las utilizadas en este proyecto se escribe en texto plano (siguiendo el objetivo didáctico); en el apartado de seguridad y criptología 0.6 se explica cómo afectaría esta carencia al sistema final.

Todos los elementos descritos en este apartado conforman lo necesario para simular la estructura descrita en la figura 0.1 y la implementada definida en el apartado 0.4.

0.3. Metodología

Metodología ágil basada en iteracciones continuas debido a la comunicación continua con el cliente...
Diagrama gantz

0.4. Escenario

La idea principal del escenario de la aplicación se ha comentado en el aparatado 0.1, en donde se comenta que el objetivo de la aplicación es meramente didáctico y experimental. Alejándose de los modelos pragmáticos del desarrollo profesional.

La complejidad añadida de implementar un sistema cercano a la realidad aplicaría un coste elevado de recursos sin tener relevancia notoria en la finalidad comentada. Elementos tales como: servidores, *WebServices*, *frameworks* de desarrollo, certificación, optimización, etcétera. Dado que la esencia de este TFG se centra en el elemento teórico y didáctico

de la criptología en curvas elípticas, la creación de una aplicación de carácter altamente profesional le añadiría beneficios nimios. Los cuales no compensan la complejidad agregada, por lo que se han realizado labores de simulación y descarte de las partes que se han considerado prescindibles.

0.5. Diseño y aspecto

iconografía vectorial [**?**] Material design intento de material desing (iconos, sombras, disposición de elementos, etcétera Colores que dan seguridad

0.6. Seguridad y criptología

hablar de qué ocurre al tener texto plano en las nfc-t en vez de las mifare classic