UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INFORMÁTICA

PROYECTO FIN DE GRADO

ANÁLISIS DE SEGURIDAD DE APLICACIONES PARA ANDROID

Leganés, 05/09/2012

AUTOR: DAVID RUBIO MATELLANES

TUTOR: JORGE BLASO ALIS

El proyecto tiene como objetivo definir una serie de características a analizar para aplicaciones Android y realizar el análisis sobre un conjunto finito de las mismas.

Tabla de contenido

Tabla de tablas

Tabla de ilustraciones

1 Introducción

Los mercados de aplicaciones móviles cuentan con decenas de categorías que ayudan a los usuarios a seleccionar mejor las aplicaciones en función de sus gustos. Un ejemplo puede ser Google Play (1), que cuenta con 26 categorías como compras, medicina, finanzas, transporte, etc.

Desgraciadamente, las aplicaciones, aunque de diversos tipos, no siempre cumplen con los requisitos mínimos de seguridad que deberían según los datos que tratan. Es decir, las categorías que ofrece Google no se adaptan a las necesidades de un análisis de seguridad porque las aplicaciones de una misma categoría tratos diferentes tipos de datos. Por ejemplo, en el apartado compras, existen aplicaciones que permiten conocer el precio de un artículo a partir de una foto del código de barras y aplicaciones que son el cliente de compras de una pagina web, como puede ser Amazon. La importancia de los datos para los usuarios no es la misma en ambas aplicaciones.

1.1 Motivación

En primer lugar, la falta de una clasificación para agrupar las pruebas de auditoria de seguridad sobre aplicaciones Android y de la ausencia de estudios sobre este asunto, genera la necesidad de crear una clasificación específica.

Posteriormente, a la clasificación se le asociaría un marco común de análisis y pruebas definido para cada grupo de aplicaciones. El objetivo del marco de común es recoger los elementos vulnerables típicos y exponerlos ante las posibles amenazas para observar su comportamiento, de tal manera que las aplicaciones con datos similares se puedan estudiar en un entorno dónde pueda ver el comportamiento de los mecanismos de seguridad que estén implementados.

Este marco común serviría para acelerar y mejorar las pruebas de seguridad de auditoria de una aplicación Android. Así, los costes de desarrollo se verían reducidos y los desarrolladores podrían incluir las medidas propuestas en sus sistemas de producción de software.

Finalmente, se tendrá un baremo con el que poder establecer hasta que punto es segura una aplicación. El sistema de puntación tendrá en cuenta el número de pruebas superadas y la importancia del activo de la prueba no superada.

1.2 Objetivos

El objetivo es obtener un conjunto de pruebas que sirvan de marco para una serie de categorías de aplicaciones. La filosofía de desarrollo para obtener óptimos resultados en la elaboración de este proyecto no puede ser igual que si se tratará de un producto software, por lo tanto, no se trabajará sobre el paradigma de desarrollo software mediante casos de uso, requisitos de usuario, requisitos software, diseño, implementación, pruebas, etc.

El marco de auditoría de seguridad para aplicaciones Android seguirá una serie de pasos su desarrollo diferente al establecido para software.

En primer lugar, se establecerán las categorías de aplicaciones con una serie de datos afines entre sí. De tal manera, que una misma prueba se pueda emplear para

elementos similares de diferentes aplicaciones grupo. El número de optimo de categorías dependerá de la cantidad de conjuntos de datos se pueden asociar.

Posteriormente, se seleccionarán los posibles grupos que pueden formar parte de la categorización y se asignarán a cada una las categorías de Google relacionadas. Como ya se ha comentado, las categorías contienen aplicaciones que operan con datos muy diferentes y, por lo tanto, en la asignación habrá que destacar qué tipo de datos debe maneja las aplicaciones de dichas categorías.

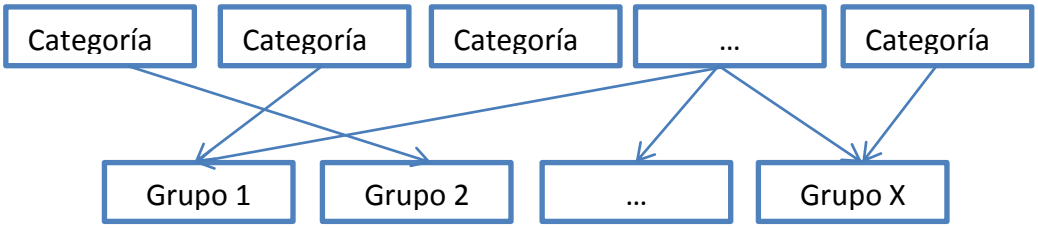


Ilustración 1 - Agrupación de categorías

Para ver el alcance real de este sistema de análisis de seguridad, se seleccionarán una serie de aplicación de cada grupo. El objetivo es cubrir el máximo número de posibilidades que abarca cada grupo para poder observar cómo se puede adaptar el marco a los diferentes tipos de datos de cada aplicación.

A continuación, se detallarán los puntos que se quieren estudiar de cada uno de los grupos y con qué herramientas se podrán realizar el estudio de las aplicaciones. El alcance de las herramientas permitirá conocer hasta que punto se puede determinar si una aplicación se protege contra las amenazas a las que se expone.

Una vez conocidas las herramientas, el objetivo será determinar qué aspectos debe tener en cuenta una aplicación para considerarse suficientemente segura. En ningún caso se podrá asegurar que una aplicación es 100% segura pero conocidas las amenazas y la importancia de cada activo para el usuario, permitirá determinar un umbral mínimo de seguridad que hay que alcanzar.

Finalmente, habrá que especificar las pruebas de cada grupo. Conociendo las herramientas y los aspectos generales que se deben proteger, únicamente es necesario especificar para ese grupo de aplicaciones qué datos son más importantes y con qué herramientas se puede observar si se protegen lo suficiente ante sus amenazas.

Cómo anteriormente se han seleccionado aplicaciones concretas que representarán cada grupo, se probará el conjunto de pruebas sobre cada una de ellas. El objetivo es conocer si las herramientas seleccionadas, los datos marcados como sensibles y los métodos a realizar son los correctos.

Además, el empleo de aplicaciones en un entorno real servirá para conocer el nivel de implicación de los desarrolladores en cuestiones de seguridad en los dispositivos Android.

Este proceso, más ajustado a las necesidades del proyecto, determinará si el marco de análisis de seguridad de aplicaciones Android es correcto y se puede emplear en el desarrollo de aplicaciones.

1.3 Estructura del documento

El documento recogerá todo el proceso de creación y comprobación del método propuesto para realizar auditorias de seguridad de aplicaciones Android.

En primer lugar, se describirá el análisis del entorno realizado para establecer qué grupos se han seleccionado, qué categorías de Google agrupa, las aplicaciones que representan cada grupo, los elementos que hay que tener en cuenta en el análisis y las herramientas empleadas para obtener resultados en las pruebas.

A continuación, se detallará el diseño de las pruebas para grupo de aplicaciones, es decir, qué pruebas se van a llevar a cabo, con qué herramientas, qué datos se van a estudiar y qué información se espera obtener. Además, se especificarán ciertas pruebas adaptadas concretamente a las aplicaciones representativas de manera que se pueda observa mejor el funcionamiento de la aplicación.

Posteriormente, se mostrarán los resultados obtenidos en las pruebas sobre las aplicaciones y se determinará si cada aplicación es lo suficientemente segura.

Para concluir, se realizará un comentario crítico analizando y resumiendo los diferentes resultados obtenidos, tanto a nivel específico de las aplicaciones de muestra como a nivel global de los diferentes grupos.

2 Análisis

Durante esta sección se seleccionarán los grupos de aplicaciones que servirán de referencia, se estudiarán los diferentes aspectos que hay que tener en cuenta para que una aplicación se considere suficientemente segura, se establecerá qué herramientas serán empleadas durante el estudio y se analizarán los grupos para determinar exactamente qué datos tienen en común las aplicaciones y pueden estar bajo la amenaza de un atacante.

2.1 Selección de grupos de aplicaciones

Google ofrece en su catálogo de aplicaciones una clasificación en función de la temática (2). Esta distribución puede resultar ambigua a nivel de seguridad, dentro de un mismo grupo puede haber aplicaciones con acceso a información sensible, requerir un login, realizar un pago, etc. Por lo tanto, es necesario establecer una nueva clasificación de aplicaciones.

Se ha decidido crear diferentes grupos de aplicaciones utilizando como criterio el nivel de seguridad esperado en las mismas. Es decir, el grupo más exigente (grupo 1) albergará aplicaciones que traten información sensible del ámbito económico/bancario y el grupo menos exigente (grupo 4) aplicaciones informativas que no requieren registro.

En los niveles intermedios se establece un abanico con 2 opciones. El grupo seguridad intermedia-alta (grupo 2) será el de redes sociales, que contienen información sensible de tipo personal. El grupo de seguridad intermedia-baja (grupo 3) acogerá aplicaciones que solicitan login de acceso pero que no albergan otra información personal de carácter importante.

GRUPOS	TIPOS DE APLICACIÓN
GRUPO 1	Aplicaciones de bancos y cajas de ahorros.
GRUPO 2	Aplicaciones de redes sociales.
GRUPO 3	Aplicaciones con login inicial (juegos, entrenamiento...).
GRUPO 4	Aplicaciones de consulta (noticias, información bursátil, tiempo atmosférico).

Tabla 1 - Grupos de aplicaciones.

Como ya se ha comentado, las categorías de Google Play no se adaptan perfectamente a los grupos en función de los niveles de seguridad. Sin embargo, la mayoría de las aplicaciones de un grupo pertenecen a una sola categoría en Google Play.

El grupo 1 (o grupo bancos) albergará aplicaciones que aparecen en finanzas. Pero no todas las aplicaciones de finanzas pertenecen a este grupo, existen aplicaciones de consultas bursátiles, control de economía personal o cambios de divisas.

El grupo 2 (o grupo redes sociales) contendrá aplicaciones donde los usuarios comparten con sus conocidos (u otras personas) una serie de datos, comentarios, fotografías o localizaciones. En Google Play se alojan en Sociedad todas ellas, prácticamente todas las aplicaciones de tipo sociedad son cliente oficiales o extraoficiales de redes sociales muy reconocidas.

El grupo 3 (o grupo login) albergará aplicaciones que protegen el acceso a sus datos con un login. Pueden ser de cualquiera de las categorías de Google Play. Se espera que estos datos puedan ser ligeramente sensibles o el usuario quiera controlar el acceso a los mismos.

El grupo 4 (o grupo consultas) acogerá aplicaciones que no guardan ningún tipo de información sobre el usuario. El contenido de estas aplicaciones es meramente informativo y no requiere de ningún tipo de dato del usuario.

2.2 Selección aplicaciones de cada grupo.

Ilustración 2 muestran un pequeño resumen de los grupos de aplicaciones que se van a estudiar.

Como se puede observar, el grupo bancos y el grupo redes sociales se ajustan perfectamente a un nivel de seguridad requerido. Hay que tener en cuenta, que en Finanzas se han excluido las aplicaciones de consulta financiera y solamente se estudiarán las aplicaciones relacionadas con bancos y pagos directos.

Sin embargo, los grupos pagos, login y consultas son muy amplios y se va tomar una muestra de ciertos tipos de aplicaciones que cumplan las condiciones del grupo de estudio.

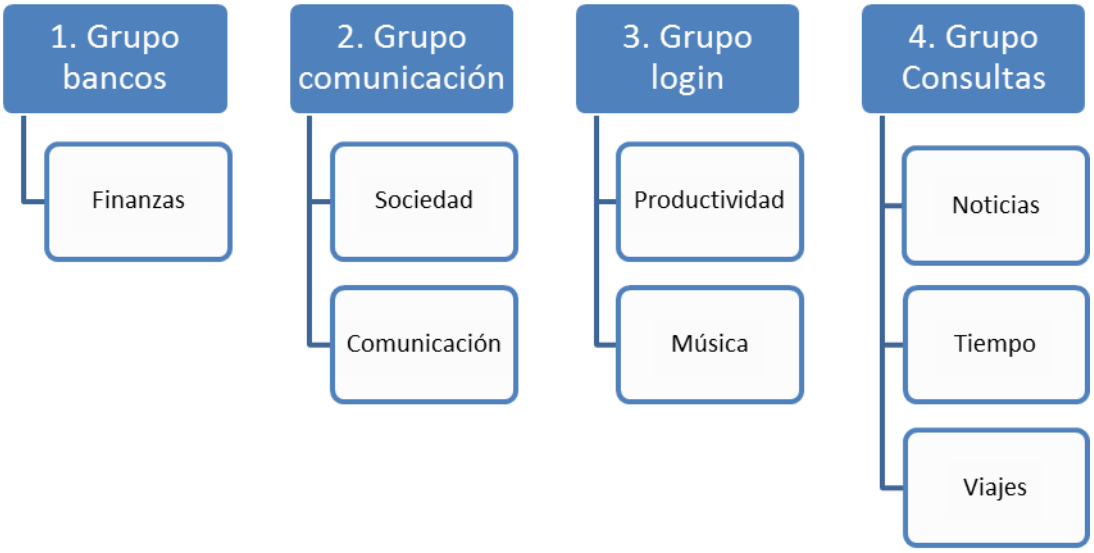


Ilustración 2 – Relación entre los grupos creados y las categorías en Google Play

Cada grupo estará representado por 4 aplicaciones, se seleccionarán las más populares entre las aplicaciones gratuitas de las categorías indicadas que se ajustan al grupo de estudio. El estudio debería hacerse tomando muestras de ambos grupos pero, el coste del estudio subiría considerablemente. Además, la mayoría de las aplicaciones son las gratuitas según Distimo, el portal especializado en analizar mercados de aplicaciones (3).

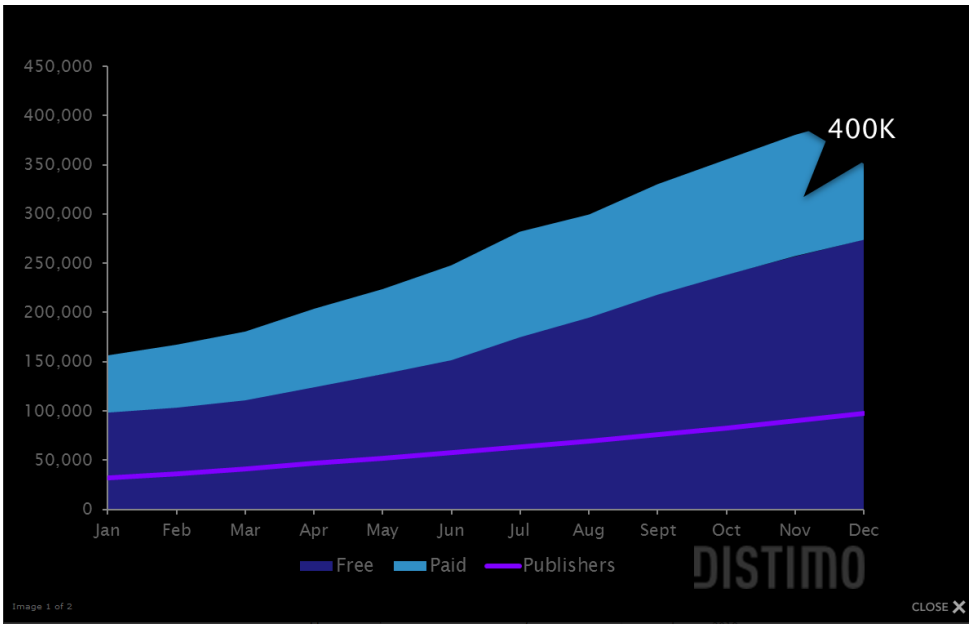


Ilustración 3 - Número de aplicaciones en Google Play

2.2.1 Aplicaciones de bancos

Las cinco aplicaciones que forman la muestra del grupo bancos son:

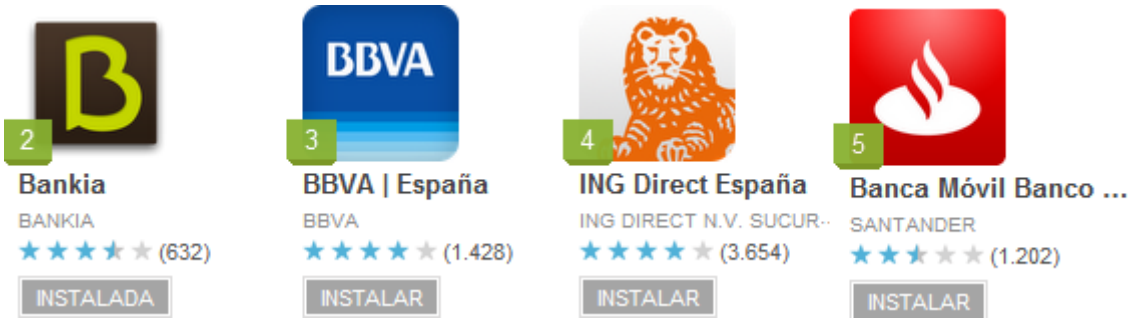


Ilustración 4 - Aplicaciones bancarias

2.2.2 Aplicaciones de comunicación.

Las cinco aplicaciones que forman la muestra del grupo redes sociales son:

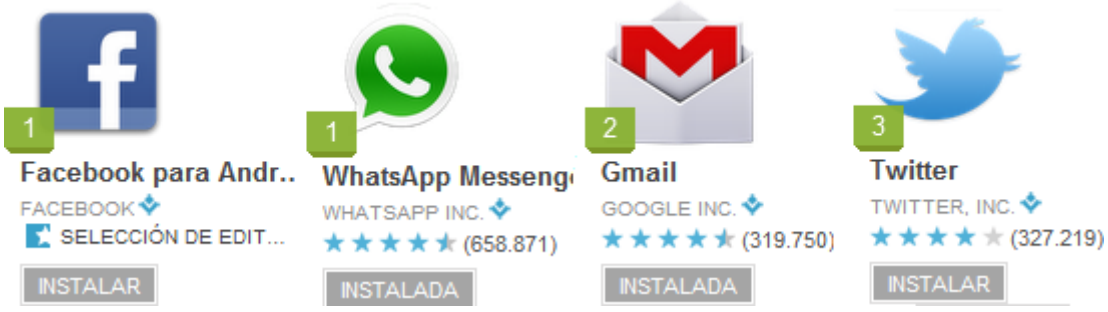


Ilustración 5 - Aplicaciones de comunicación

Son la red social más popular (Facebook), el servicio de mensajería más popular (Whatsapp), el servicio de correo más popular (Gmail) y la red social Twitter enlaza a otras redes sociales.

2.2.3 Aplicaciones con login

Las cinco aplicaciones que forman la muestra del grupo login son:

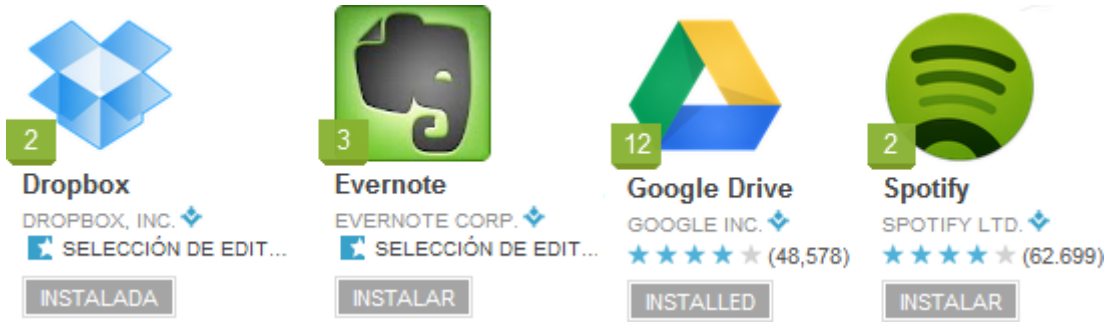


Ilustración 6 - Aplicaciones login

Las aplicaciones de productividad seleccionadas son Evernote y Dropbox, la primera es un gestor de notas y la última es gestor de un directorio en la nube con sincronización automática de archivos. Google Drive es la apuesta reciente de Google para competir contra Dropbox.

La aplicación Spotify de música es un popular reproductor de música alojada en la nube. La aplicación es gratuita pero el acceso al contenido no lo es (ofrecen un mes de prueba gratuito (4) durante el cual se realizarán las pruebas).

2.2.4 Aplicaciones de consulta

Las cinco aplicaciones que forman la muestra del grupo consultas son:

Ilustración 7 - Aplicaciones consulta

Las aplicaciones de noticias seleccionadas son y RTVE noticias y El País. La aplicación más característica de tiempo es tiempo AEMET. La aplicación sobre viajes seleccionada en Google Maps.

2.3 Análisis herramientas y consejos de Google para desarrolladores

Para poder comprender que nivel de seguridad puede alcanzar en una aplicación Android, por un lado, hay que conocer la potencia de las herramientas de análisis y, por otro, lado hay que saber como quiere Google que se desarrollen las aplicaciones sobre su plataforma para que sea suficientemente segura para el usuario.

Google ofrece en Android *developers*, su web de referencia para desarrolladores, manuales de diseño, implementación y pruebas (5). También, proponen una serie de indicaciones que consideran buenas prácticas de desarrollo de productos Android (6).

Las herramientas de depuración, propuestas para realizar pruebas, resultan útiles para obtener información sobre el tratamiento de los datos en memoria y bases de datos por parte de las aplicaciones.

Entre las buenas practicas de desarrollo, se encentran una serie de consejos sobre diseño de seguridad. Las medidas de seguridad recomendadas serán empleadas para estimar los requisitos mínimos de cada nivel de seguridad

El objetivo final del marco desarrollado y las pruebas realizadas a las diferentes aplicaciones es detectar si se han llevado a cabo las buenas prácticas de diseño de seguridad mediante las herramientas de depuración.

2.3.1 Herramientas de depuración

La SDK de Android ofrece suficientes herramientas para realizar una depuración completa de una aplicación. EL único requisito que precisan los métodos de depuración es contar con el depurador-compilador JDWP para poder ejecutar paso a paso, ver el valor de las variables y pausar la ejecución de una aplicación.

Los principales componentes del entorno de depuración son:

- **ADB**: es una herramienta que se sitúa entre el dispositivo y el sistema de desarrollo. Provee al desarrollador de funcionalidades de gestión como sincronización de archivos, consola UNIX y comunicación entre dispositivos conectados y emuladores.
- **Dalvik Debug Monitor Server (DDMS):** es el programa que se comunica con el dispositivo a través de ADB. DDMS puede realizar capturas de pantalla, capturar hilos y la información de la pila y simular llamadas y SMS entrantes.
- **Dispositivo o simulador:** para poder depurar una aplicación hay que ejecutarla sobre un dispositivo o sobre un simulador de dispositivos (AVD en sus siglas en inglés). El ADB ejecuta un cliente en el dispositivo o emulador y provee al host de ADB comunicación con el dispositivo o AVD.
- **Depurador JDWP:** es un protocolo que permite unir el DDMS con la máquina virtual a través de un puerto de comunicación. De tal manera que se puede depurar múltiples aplicaciones si se asigna un puerto a cada una.

La Ilustración 8 - Entorno de depuración muestra como se relacionan los distintos elementos de un entorno de depuración:

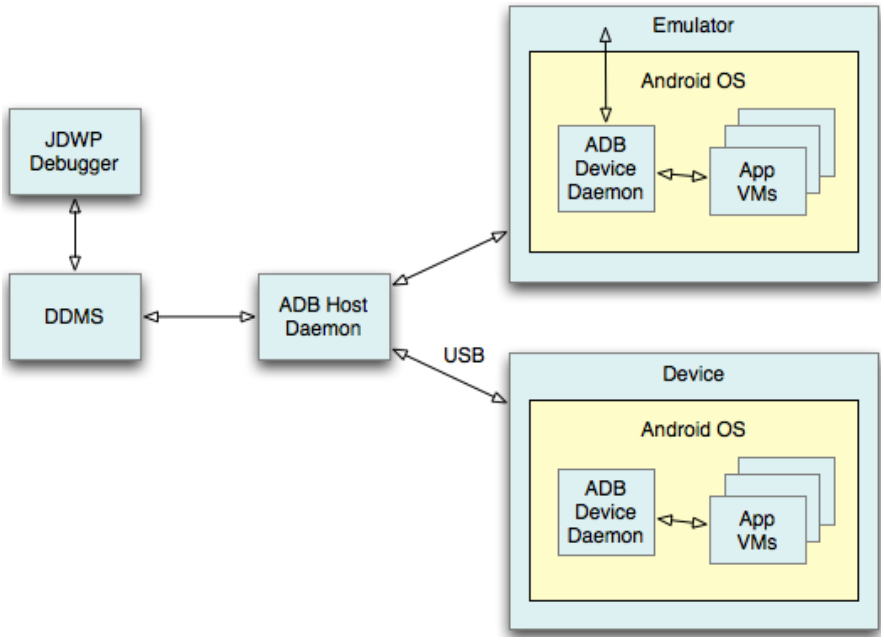


Ilustración 8 - Entorno de depuración (5)

2.3.1.1.1 *Android Debug Brigde*

El Android Debug Bridge, a partir de ahora ADB, se emplea como mecanismo de comunicación entre el desarrollador y la dispositivo Android.

2.3.1.1.2 *Componentes del ADB*

La comunicación del ADB se basa en un modelo cliente-servidor, dónde sus componentes son:

- Cliente: se ejecuta en el entorno de desarrollo desde la consola de comandos. El DDMS también ejecuta clientes ADB y es más sencillo de utilizar.
- Servidor: se ejecuta como un proceso en background en el entono de desarrollo. El servidor controla la comunicación entre el cliente y el demonio que se ejecuta en el dispositivo.
- Demonio: se ejecuta en background sobre cada instancia de un dispositivo.

2.3.1.1.3 *Ciclo de ejecución de ADB*

El ciclo de ejecución del ADB es el siguiente:

1. Se ejecuta el cliente ADB
2. Se comprueba si hay procesos de servidor ADB en marcha.
   2.1. Si no, se crea un proceso de servidor ADB.
   2.2. El servidor hace un *bind* al puerto local 5037 TCP.
   2.3. El servidor escucha los comandos enviados por los clientes (todos se comunican por ese puerto)
   2.4. El servidor configura todos los dispositivos en ejecución (se les aginan puerto impares desde el 5555 al 5585).
   2.5. Cuando el servidor encuentra un demonio ADB, configura la conexión del puerto.

3. Una vez conectado el servidor, se configuran las conexiones de todas las instancias para poder usar comandos de control y acceso a las instancias. Se puede controlar cualquier dispositivo desde cualquier cliente.

2.3.1.1.4 Línea de comandos

Todos los comandos es capaz de interpretar usan esta estructura:

```
adb [-d|-e|-s <serialNumber>] <command>
```

Aunque la interfaz habitual no será la línea de comandos si no un plugin que incluyen en el entorno Eclipse con ADT instalado.

2.3.1.1.5 Funcionalidades que ofrece el ADB

Las funcionalidades que ofrece el ADB están destinadas principalmente a configurar y manejar distintos dispositivos simulados o reales.

- Encolado de instancias de dispositivos emulados y reales: cada dispositivo esta identificado por un número de serie. El número de serie se compone del tipo de dispositivo y el puerto de la consola.

```
[serialNumber] = [type]-[consolePort]
```

- Listados de dispositivos conectados al ADB: lista todo los dispositivos por número de serie e indica su estado. El estado puede ser desconectado (`offline`) o conectado (`device`).

```
[serialNumber] [state]
```

- Comandos directos para instalar aplicaciones: Para instalar la aplicación que se desea depurar o alguna complementaria se emplea este comando.

```
adb install <path_to_apk>
```

- Especificar puertos de renvío: Se puede renviar el flujo de un puerto a otro diferente o a un socket para poder ser analizado.

```
adb forward tcp:src_port tcp:dst_port

adb forward tcp:src_port local:logd
```

- Gestión de ficheros: Se puede enviar ficheros desde el dispositivo o al dispositivo. Si los ficheros son directorios realiza una operación recursiva.
    - Desde el dispositivo:

```
adb pull <remote> <local>
```

- Hasta el dispositivo:

```
adb pull <remote> <local>
```

- Además, se pueden realizar funciones de visualización de información:
  - Imprimir los datos de log en pantalla.
  - Imprimir dumpsys, dumpstate y logcat por pantalla.
  - Imprimir lista de procesos JDWP en un dispositivo.
  - Imprimir número de serie y estado.
- Bloquear ejecución hasta que el dispositivo esté disponible.
- Arrancar/parar el servidor ADB.
- Arrancar consola de comandos: la consola de comandos tienen su propio conjunto de funciones que puede realizar:
  - Examinar las bases de datos sqlite3 desde consola remota. Permite ver el contenido de las tablas y las estancias SQL CREATE realizadas. Se puede introducir cualquier comando SQL.
  - Ejecutar la aplicación Monkey que genera flujos de eventos pseudo-aleatorios. Es interesante para probar cualquier aplicación simulado usos de usuarios.

```
$ adb shell monkey -v -p your.package.name numbreEvents
```

  - Permite parar y arrancar un dispositivo.
  - Imprime datos sobre dumps system, dumps state, radio logging y mensaje de depuración de kernel.
- Habilitar el logcat:
  - Comandos en logcat: se emplea para visualizar y seguir el contenido de los diferentes buffers de sistema. El método de empleo es el siguiente:

```
[adb] logcat [<option>] ... [<filter-spec>] ...
```

  - Filtros en logcat: es una etiqueta que identifica el tipo de mensaje originado.
    - V — Verbose (Menor prioridad)
    - D — Debug
    - I — Info (prioridad por defecto)
    - W — Warning
    - E — Error
    - F — Fatal
    - S — Silent (Prioridad alta)
  - Control de formato de salida: los mensajes contienen metadatos y una etiqueta y prioridad. Pero se puede modificar mostrando la salida de algunos campos específicos. El formato del comando es:

```
[adb] logcat [-v <format>]
```

Los diferentes formatos aceptados son:

- brief – Muestra prioridad y el PID del proceso (por defecto).
- process – Solamente muestra el PID.
- etiqueta — Solamente muestra la prioridad.
- raw — Muestra el mensaje de log integro sin metadatos.
- time — Muestra la fecha, el momento de invocación, la etiqueta y el PID del proceso.
- threadtime — Muestra la fecha, el momento de invocación, el PID del proceso y el TID del *thread*.
- long — Muestra todo los campos de metadatos.

- Ver log de buffer alternativos: Android guarda mensajes de log de múltiples buffers. Para seleccionar el tipo de mensajes adicional que se quiere ver se emplea:

```
[adb] logcat [-b <buffer>]
```

Donde los diferentes buffer seleccionable son:

- `radio` — Mensajes relacionados con la radio y la telefonía.
- `events` — Mensajes relacionados con eventos del sistema.
- `main` — Por defecto, buffer principal.

- Ver salida estándar y salida estándar para error: son herencia del Linux, por defecto el sistema envía a stdout (*System.out*) y stderr (*System.err*) un output. Cuando un proceso esta ejecutando en la maquina virtual de Dalvik, el sistema escribe los mensajes para el log empleado las etiquetas *stdout* y *stderr*, con prioridad I.

2.3.1.2 DDMS (Dalvik Debug Monitor Server)

El DDMS es una herramienta ofrecida con el SDK de Android que provee de servicios de renvío de puertos, capturas de pantalla en dispositivos, información sobre procesos, *logcat*, estado de la radio, *spoofing* de llamadas y SMS, datos de localización,… Básicamente, son las mismas funcionalidades que las de ADB pero con interfaz gráfica, mucho más agradable para el desarrollador.

El ADT incluye en Eclipse una perspectiva llamada DDMS donde se pueden ver las principales funcionalidades del ADB/DDMS entre los accesos directos de la ventana.

2.3.1.2.1 Interacción DDMS con ADB

Al arrancar el DDMS lo primero que hace es conectarse con el ADB. Se conectan ambos módulos mediante un servicio de monitorización de maquina virtual. Sirve para avisar al DDMS si la VM está encendida o no.

Cuando arranca el DDMS recibe el PID del proceso y abre una conexión con el debugger. Lanza el demonio del ADB en el dispositivo y ya puede comunicarse usando el protocolo de conexión personalizada.

El DDMS asigna a cada maquina virtual un puerto, todo el tráfico se desvía a la depuración de la máquina virtual asociada. Solamente se puede asignar un debugger por puerto, pero con el DDMS es sencillo manejar varios debugger.

2.3.1.2.2 Funcionamiento del DDMS

En la perspectiva DDMS podemos ver diferentes componentes que nos ofrecen mucha información de depuración.

Por un lado, ofrece información sobre los dispositivos conectados y los procesos que están corriendo en él. Sobre cada proceso muestra los puertos asociados y el PID de cada proceso. También ofrece un controlador del emulador que permite control sobre voz, datos, velocidad y latencia. Además ofrece gestión sobre funcionalidades telefónicas como llamadas y SMS y servicios de posicionamiento forzado. Después, *LogCat* muestra todos los mensajes de los distintos buffers mostrando hora, tipo, PID, etiqueta y mensaje.

Por otro lado, se puede ver el uso del heap en cada proceso, seguimiento de la asignación de memoria, trabajar con el sistema de ficheros del emulador o dispositivo.

2.3.1.2.3 Visualización del uso del heap *en cada proceso*

El DDMS permite visualiza cuanta memoria emplea cada proceso. Es muy útil para realizar el seguimiento de las aplicaciones durante su ejecución.

El funcionamiento es sencillo, primero se selecciona el proceso que se quiere analizar, segundo se actualiza la información del *heap* y, finalmente, se invoca al recolector de basura para asegurar que la información es actual.

Primero aparece un pequeño resumen con el identificador, el tamaño del *heap*, el espacio reservado, el espacio libre, el porcentaje de uso y el número de objetos.

Despues desglosa la informacion en diferentes grupos de datos: libre, objetos, clases, arrays de 1, 2, 4 y 8 bytes y objetos que no son de Java. De cada uno ofrece el numero de elementos que hay, el tamaño total, el tamaño más pequeño, el tamaño más grande, el tamaño medio y la media de tamaños.

Además, como se puede ver en la siguiente ilustración, de cada linea añade información gráfica con el recuento de los diferentes tamaños.

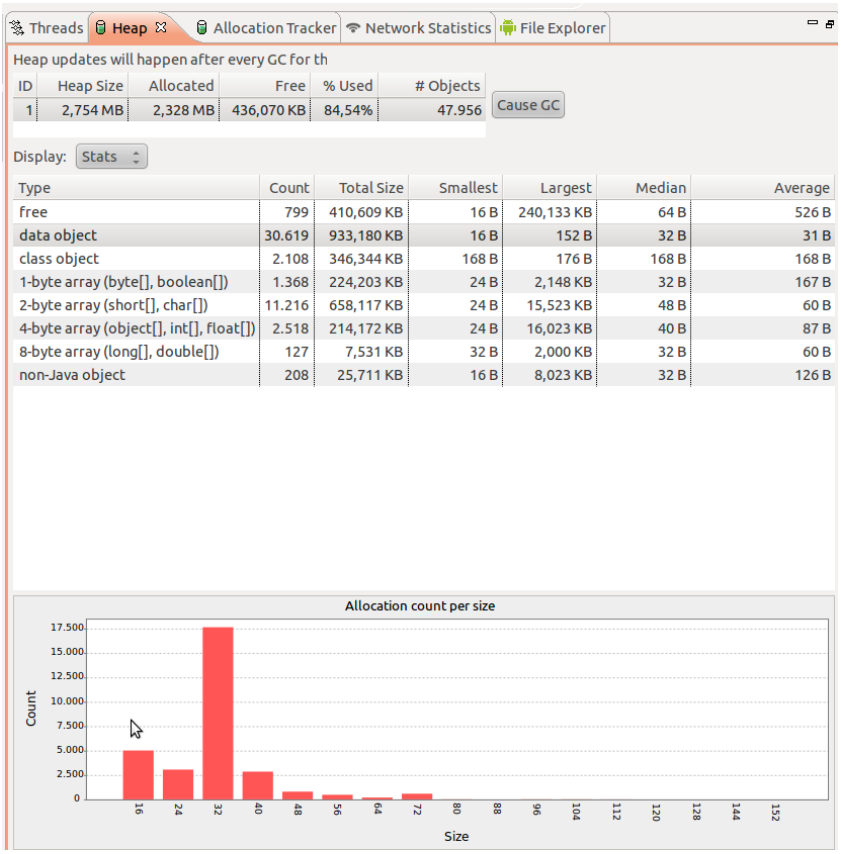


Ilustración 9 - Perspectiva DDMS

2.3.1.2.4 *Seguimiento del espacio reservado*

Se puede realizar un seguimiento de todos los objetos que ha reservado espacio en memoria y ver que clases e hilos los han reservado. Esto permite conocer al desarrollador como afectan los distintos objetos a la memoria principal.

De cada objeto muestra el número de orden en la memoria, el tamaño reservado, la clase reservada, el idetificador de *thread* y donde está alojado el objeto. Luego, se indica la posicion exacta de la invacion expecificando linea, fichero, clase y metodo.

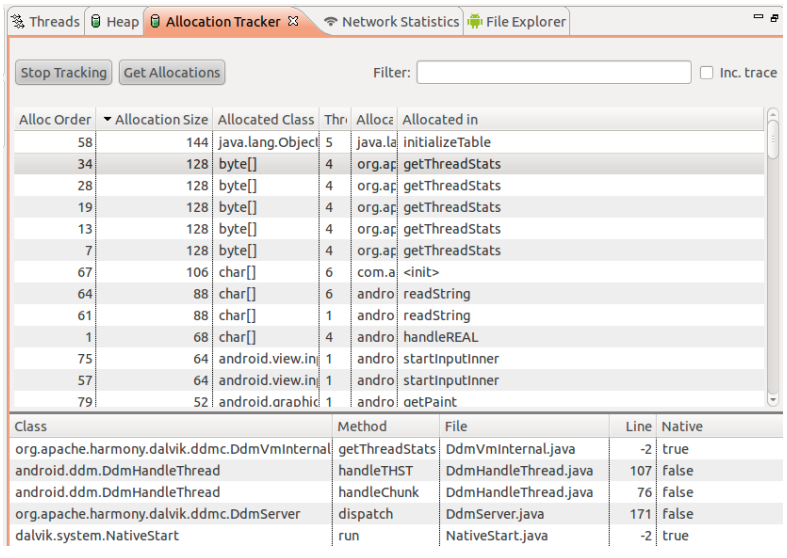


Ilustración 10 - Espacio reservado

2.3.1.2.5 *Sistema de ficheros del dispositivo o emulador*

El DDMS abstrae las funciones *pull* y *push* del ADB a una interfaz gráfica fácil de usar. Muestra el sistema de ficheros y se puede navegar por él como si del de un sistema operativo, como Linux, se tratara. También ofrece información sobre tamaño, fecha de creación y permisos para los usuarios.

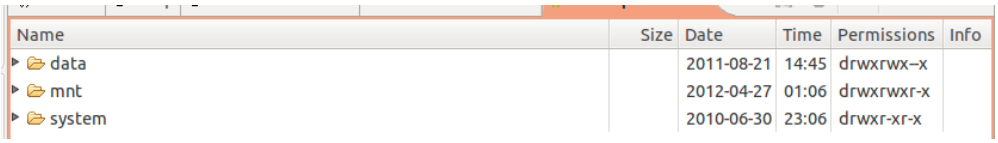


Ilustración 11 - Sistema de ficheros

2.3.1.2.6 *Información de threads*

La información que ofrece de los *threads* es muy básica: idetificador, TID, estado, tiempo y nombre.

La informacion que muestra es similar a seguimiento de espacio reservado, de cada hilo muestra el número de orden en la memoria, el tamaño reservado, la clase reservada, el idetificador de *thread* y donde está alojado el objeto. Luego, se indica la posicion exacta de la invacion expecificando línea, fichero, clase y metodo.

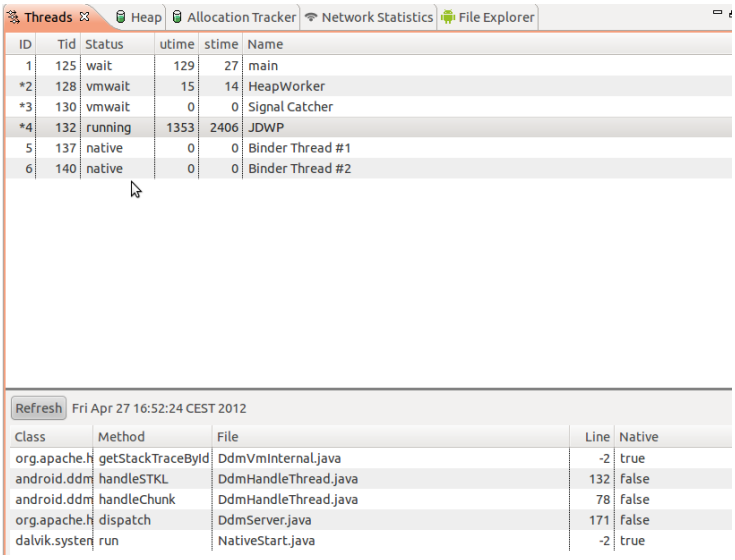


Ilustración 12 - Información de los *Threads*

2.3.1.2.7 *Herramienta de visualización de tráfico de red*

Desde Android 4.0, el DDMS incluye una pestaña de uso detallada de la red que monitoriza el flujo de datos que se hacen con las peticiones en red. Resulta muy útil para saber cuando y como las aplicaciones se conectan y transfieren datos. Permite realizar etiquetado para poder diferenciar diferentes tipos de tráfico.

El sistema de etiquetado esta implementado en un API llamada `TrafficStats` con tres simples métodos se puede establecer una etiqueta a un hilo, etiquetar un socket y desetiquetar un socket.

Apache ofrece otra alternativa a partir de las clases HttpClient y URLConnection, aunque el funcionamiento es muy parecido. Permite establecer y obtener etiquetas de estadísticas de los hilos.

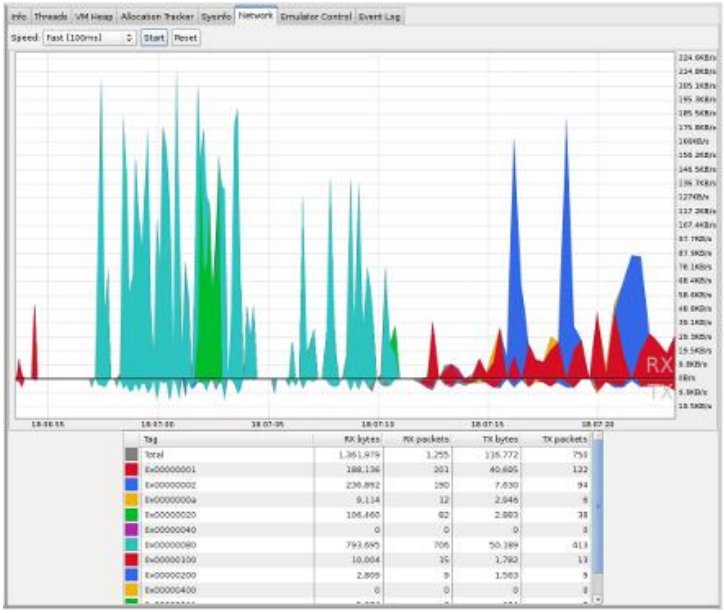


Ilustración 13 - Herramienta de visualización de tráfico de red

2.3.1.2.8 *LogCat*

LogCat también esta integrado en el DDM, muestra los mensajes de las salidas impresos en pantalla. Aparte de los elementos del sistema, una de las clases de salida es la clase Log, que permite mostrar la información etiquetada.

Level	Time	PID	Application	Tag	Text
W	03-04 02:57:54.474	1244	system_process	ThrottleServi	unable to find stats for iface rmnet0
D	03-04 03:03:00.264	1244	system_process	dalvikvm	GC_CONCURRENT freed 518K, 19% free 11501K/1408
V	03-04 03:03:29.784	1244	system_process	BackupManager	Running a backup pass
V	03-04 03:03:29.784	1244	system_process	BackupManager	Backup requested but nothing pending
W	03-04 03:07:54.493	1244	system_process	ThrottleServi	unable to find stats for iface rmnet0
I	03-04 03:09:29.154	1511	com.android.email	Email	ReconcilePopImapAccountsSync: start
I	03-04 03:09:29.214	1511	com.android.email	Email	ReconcilePopImapAccountsSync: done
A	03-04 03:09:51.204	1244	system_process	NetworkStats	problem reading network stats
A	03-04 03:09:51.204	1244	system_process	NetworkStats	java.lang.IllegalStateException: problem parsi
A	03-04 03:09:51.204	1244	system_process	NetworkStats	at com.android.internal.net.NetworkSta

Ilustración 14 - LogCat

2.3.1.3 *Aplicación de las herramientas al análisis de seguridad*

El objetivo del análisis de las aplicaciones de cada grupo es encontrar información sensible y comprobar en que estado se encuentra.

Mediante la herramienta de seguimiento de espacio reservado y la de información de los *threads*, se puede acotar la búsqueda de estos datos a simplemente los ficheros que entran en funcionamiento cuando se emplean los datos.

En el sistema de ficheros se pueden encontrar las bases de datos, los backups y archivos temporales. Es interesante ver como se guardan los datos en la bases de

datos, si los backups están cifrados o que tipo de información mantienen en los temporales.

2.3.2 Buenas prácticas de seguridad

Android ofrece una seguridad mínima por defecto relativamente baja, para que sean los desarrolladores los que decidan las medidas más convenientes para su aplicación. Por supuesto, Android cuenta con una serie de medidas a nivel de sistema operativo:

- Sandbox: el código de ejecución y los datos de las aplicaciones están en memoria principal aisladas entre sí, es decir, cada aplicación puede acceder únicamente a sus datos y código de ejecución.
- Marco de aplicación de Android: tiene implementación robusta de funcionalidades de seguridad como criptografía, permisos e IPC seguro. Cuando una aplicación está en ejecución, Android cuenta con una serie de mecanismos que permiten la comunicación con otros servicios y actividades, la solicitud de recursos al sistema y el almacenamiento de datos.
- Tecnologías como ASLR, NX, ProPolice, safe_iop, dlmalloc OpenBSD, calloc OpenBSD y Linux mmap_min_addr que mejoran la gestión de memoria y se mitigan algunos riesgos asociados. Estos mecanismos se encargan de proteger zonas clave de la memoria, evitan *overflow*, controlan el formato de los datos, reservan memoria dinámicamente y controlan el espacio de memoria virtual disponible.
- Sistema de cifrado del sistema de ficheros completo para proteger los datos en caso de perdida o robo del dispositivo. Este mecanismo esta disponible actualmente de forma manual, es decir, el usuario debe activarlo en ajustes del sistema.

Los desarrolladores deberían implementar una serie de medidas para estar seguro que se han reducido los riesgos que afectan a su aplicación. El despliegue de estas medidas requiere de un compendio de APIs y técnicas de desarrollo enfocadas a prevenir riesgos de seguridad en la aplicación.

Las buenas prácticas recogidas en esta sección será el objeto de estudio en los diferentes grupos de aplicaciones.

2.3.2.1 Código Dalvik

Aunque, la máquina virtual Dalvik está muy revisada, tanto por Google como por la comunidad de desarrolladores, pero todo código contiene errores y, por lo tanto, la posibilidad de explotar alguna vulnerabilidad.

La máquina se encarga de ejecutar Java y comparte ciertos errores y vulnerabilidades de las aplicaciones que ejecutan Java. A diferencia de otras máquinas virtual, la Dalvik puede interactuar con el código nativo en la misma aplicación sin ningún tipo de restricciones de seguridad.

En Android, es común la carga dinámica de clases para aligerar las aplicaciones. Consecuentemente, hay que evaluar donde se recibe la lógica de la aplicación y donde se almacena localmente. El problema de las clases cargadas estáticamente es la falta de verificación de las fuentes, pudiendo incluirse clases maliciosas o con vulnerabilidades.

La mayoría del desarrollo no se realiza directamente sobre Dalvik si no que se emplean las librerías dinámicas del SDK.

2.3.2.2 Código nativo

Desde Google, desaconsejan emplear código nativo ya que hace muy compleja la portabilidad de las aplicaciones y es más sencillo cometer errores sobre memoria, como por ejemplo: desbordamientos de buffer.

El kernel de Android es el de Linux y se desarrollaría la aplicación de manera muy similar a como se haría para este sistema, teniendo en cuenta los mismos patrones de diseño de seguridad. La diferencia más importante es la implementación del sandbox en Android, las aplicaciones se ejecutan en un *framework* ya sea nativas o por medio de SDK. La similitud más destacable es que en ambos sistemas se le asigna un UID único a cada aplicación con límites de permisos.

2.3.2.3 Almacenamiento de datos

El almacenamiento de datos permite mantener información en el dispositivo de manera permanente, ya sea por la propia funcionalidad del dispositivo (un reproductor de música) o por reducir el flujo de datos que se transmiten por las redes de comunicaciones (caché de elementos alojados en servidor).

2.3.2.3.1 Ficheros internos

Por defecto, los ficheros que son creados por una aplicación sólo pueden ser accedidos por la misma. Por norma general, es suficiente.

Existe la posibilidad de emplear ficheros globales (de lectura y/o escritura) pero no son aconsejables, ya que no proporcionan límite de acceso a las aplicaciones ni control sobre los datos. Existe una alternativa, *ContentProvider* (sección 2.3.2.3.3), que permite el control de permisos de lectura y escritura y se puedes gestionar en cada caso permisos dinámicos.

2.3.2.3.2 Alojamiento externo

Ficheros creados en sistemas externos, principalmente tarjetas SD, son de acceso global para escritura y lectura. Son ficheros que simplemente con el medio externo se puede obtener, borrar o modificar, por lo tanto, las aplicaciones no deberían almacenar información sensible en la tarjeta externa.

Las aplicaciones deberían realizar validación de entrada de todos los datos que estén en un medio externo. También es imperante no almacenar ejecutables ni librerías dinámicas en alojamiento externo.

2.3.2.3.3 Proveedores de contenido

Los proveedores de contenido son una estructura de almacenamiento que puede limitarse a la propia aplicación o se envía a otra aplicación que tenga acceso. Por defecto, esta habilitado que los datos se puedan exportar a otras aplicaciones, pero se puede desactivar si se indica en el manifiesto.

Cuando un proveedor de contenidos se crea para exportar datos se puede especificar: un solo permiso para leer y escribir o un permiso para leer y otro permiso para escribir.

Lo mejor es minimizar los permisos creados aquí y asignarlos más adelante en el manifiesto.

Cuando un proveedor de contenidos se crea para intercambiar datos entre dos aplicaciones del mismo desarrollador, es preferible emplear permisos de firma, ya que no requieren confirmación del usuario y están autenticados su origen. Es la opción más cómoda para el usuario final.

Los proveedores de contenidos ofrecen un control de acceso más granular si fuera necesario. Se puede controlar elemento por elemento en las URI y se gestiona mediante dos banderas: `FLAG_GRANT_READ_URI_PERMISSION` y `FLAG_GRANT_WRITE_URI_PERMISSION`, para lectura y escritura respectivamente.

El acceso a los proveedores de contenido está regulado por tres funciones: `query()`, `update()` y `delete()`. Con estas funciones se mitiga la amenaza de un posible ataque por *SQL injection* ya que están preparadas para recibir un cierto tipo de datos, en cambio, si se realizara directamente una consulta sobre SQL, podría introducir código en medio de la consulta y alterar la petición original.

2.3.2.4 Comunicación entre procesos

Algunas aplicaciones implementan sistemas IPC (Interprocess Communication o comunicación entre procesos) tradicionales de Linux, como son los ficheros compartidos o los socket. Pero el SDK de Android ofrece otras alternativas más optimizadas para aplicaciones móviles como: *Intents*, *Binders*, *Servicies* y *Receivers*. Estas herramientas permiten verificar la identidad de las aplicaciones con las que se conectan y establecer una política de seguridad adecuada.

Muchos elementos de seguridad se comparten mediante mecanismos IPC. *Broadcast Receivers*, *Activities* y *Services* se tienen que declarar en el manifiesto y si no es para comunicarse con otra aplicación, hay que desactivar la posibilidad de exportar datos. De esta manera se evita recepción de broadcast externo malicioso, comunicación entre aplicaciones o servicios que no debería estar conectadas.

En caso de querer usar los mecanismos IPC para comunicación con otras aplicaciones, hay que definir una política de seguridad mediante etiquetas de permisos. Al igual que los proveedores de contenido, si el desarrollador es el mismo la mejor solución es declarar permisos a nivel de firma (mejor experiencia para el usuario y mejor control de acceso a los mecanismos IPC).

Los IntentFilter se encargan de validar el formato de entrada de los datos de entrada en los Intents. No deben ser considerados mecanismos de seguridad porque se pueden invocar directamente y tienen que validar la entrada de los datos. La validación puede el formato correcto para el receptor equivocado.

2.3.2.4.1 Intents

Las *Intents* son un mecanismo IPC asíncrono que permite realizar envíos broadcast ordenados o desordenados o directamente una *Intent* a un componente de una aplicación.

Por un lado, los mensajes de broadcast ordenados se clasifican en la aplicación receptora y no todas las aplicaciones están capacitadas para ello. Por otro lado, si se envía una Intent a un receptor específico se debe enviar directamente. Estas restricciones mejoran sensiblemente la seguridad.

Los remitentes pueden verificar que el destinatario tiene permiso en el momento del envío y sólo las aplicaciones que tienen permiso lo pueden recibir. Si los datos de la Intent son información sensible, es necesario que se solicite un permiso para evitar que las aplicaciones maliciosas puedan registrarse como receptor.

2.3.2.4.2 Binder y AIDL interfaces

Los binders permiten formar interfaces RPC bien definidas que permite autenticación mutua (si fuera necesario).

La recomendación es que el diseño de la interfaces se realice para que no sean necesarios permisos para el control de la interfaz. Los binders no se declaran en el manifiesto y por lo tanto no se pueden aplicar permisos por declaración directamente. Por lo general, heredan los permisos declarados para la aplicación o la actividad dentro de la cual están implementadas. Si por cuestión de diseño se tiene que aplicar control de acceso o autenticación, se debe agregar explícitamente el código.

Para realizar las comprobaciones de control de acceso se invoca a `checkCallingPermission()` que verifica los permisos requeridos para el binder. Es importante que antes de acceder al servicio se envíe la identidad a otras interfaces. La invocación de una interfaz de servicios puede fallar si no cumple los permisos requeridos. Si la interfaz es para un uso local de una aplicación, lo mejor es realizar un *reset* de la identidad entrante del IPC en el hilo actual para satisfacer los controles internos de seguridad.

2.3.2.4.3 Broadcast Receivers

Los *broadcast receivers* suelen ser mecanismos asíncronos de petición activados por Intents. Por defecto, se pueden exportar e invocar desde cualquier aplicación y se puede limitar utilizando la etiqueta `<receiver>` en el manifiesto. Esta medida protege a la aplicación de mensajes sin los permisos necesarios.

2.3.2.4.4 Servicios

Los servicios se emplean para ofrecer funcionalidades a otras aplicaciones, cada servicio debe estar declarado en el manifiesto de la aplicación.

Por defecto, los servicios se pueden exportar e invocar por cualquier aplicación, están protegidos a través del manifiesto donde se añade la etiqueta `<service>`. Al hacerlo, otras aplicaciones tendrán dar permisos de uso en su manifiesto para poder interactuar con el servicio.

Un servicio puede proteger llamadas IPC mediante permisos y comprobarlos antes de ejecutar la aplicación llamada. Por lo general, se deberían declarar en el manifiesto.

2.3.2.4.5 Activities

Las activities se emplean frecuentemente como fachada de las funcionalidades de la aplicación. Por defecto, las activities solamente pueden exportar e invocarse desde

aplicación que tienen un IntentFilter o desde el binder que la declaró. En general, lo mejor es declarar específicamente el receptor o el servicio que va a manejar el IPC. Esta modularidad ofrece mucha seguridad ya que no se pueden invocar activities maliciosas desde aplicaciones confiables.

Para poder llamar a una actividad hay que declararla mediante la etiqueta `<activity>` para poder restringir el acceso a las solicitudes con los permisos establecidos.

2.3.2.5 Permisos

Los permisos se encargan de dar a conocer al usuario que tipo de recursos va a consumir la aplicación del dispositivo móvil. Con esta información, el usuario debe decidir si permite la instalación de la aplicación o no, es decir, es un sistema de permisos obligatorio y se aceptan o rechazan los permisos en bloque.

2.3.2.5.1 Solicitar permisos

La recomendación es solicitar el mínimo número de permisos posible, tan solo los que sean estrictamente necesarios para el correcto funcionamiento de la aplicación. Este principio suele ser llamado mínimo privilegio.

Por lo tanto, si es posible diseñar una aplicación sin permisos es preferible. Si el permiso no es necesario no se debería requerir. Además, los permisos se pueden emplear para proteger información sensible ante algún IPC, aunque siempre hay que tratar de evitarlo porque resulta confuso para el usuario. Una buena alternativa suele ser protección por firma digital para aplicaciones del mismo desarrollador.

2.3.2.5.2 Creación de permisos

La creación de permisos es realmente poco común, los ya extenientes ofrecen permisos para las funcionalidades típicas de las aplicaciones en Android. Antes de crear un nuevo permiso, hay que estar seguros de que no existe un método similar y compatible con lo que se pretende implementar.

En primer lugar, si se va a crear un permiso hay que considerar la opción de los permisos de firma. Los permisos de firma son transparentes para el usuario y sólo permite el acceso a los elementos firmados por el desarrollador. Cuando se crea un permiso, se corre el riesgo de que el usuario no entienda o no considere necesario ese permiso y decida no instalar la aplicación.

Si se crea un permiso hay que tener en cuenta unos puntos:

- El permiso se crea con un texto breve explicando al usuario qué permiso está otorgando.
- El texto del permiso debe estar en multitud de idiomas.
- El permiso puede resultar extraño al usuario y puede decidir no instalar la aplicación.
- Las aplicaciones puede solicitar el permiso aunque no se haya instalado aún el creador de los permisos.

2.3.2.6 Redes

Las redes inalámbricas sirven al dispositivo para obtener un punto de acceso a Internet y así poder contactarse a los proveedores de contenidos y servidores que nutren de información las aplicaciones.

2.3.2.6.1 Redes IP

El uso de redes en Android es muy diferente al de Linux. Para estar seguros de realizar transmisiones seguras con información sensible se debe emplear HTTPS, que es HTTP más SSL/TLS para añadir cifrado al canal. El protocolo HTTPS se dirige siempre al puerto 443.

El autenticado y cifrado se realiza a nivel de socket con la clase SSLSocket. Es muy recomendado emplear SSLSocket para mitigar riegos al emplear redes públicas (muy común en los usuarios).

No es aconsejable emplear socket como método IPC en *localhost* puesto que esos datos son accesibles desde otras aplicaciones. En su lugar, se recomienda emplear unos mecanismos de autenticación IPC como un servicio o un binder.

Además, por defecto hay que desconfiar de los datos obtenidos por HTTP, incluyendo datos de entrada que se recomiendan validar siempre.

2.3.2.7 Carga dinámica de código

Es una práctica completamente desaconsejable cargar código externo a una aplicación. Si se hace, aumenta la probabilidad de que la aplicación quede comprometida por la inyección de código malicioso. El código cargado debe cumplir los mismos permisos que el resto de la aplicación. Esto garantiza que se cumplan siempre los permisos que el usuario ha aceptado.

El riesgo de seguridad puede ser mitigado si el código puede ser verificable, al igual que su fuente. La ventaja que ofrecen es que se mantienen siempre actualizadas las clases en todas las aplicaciones.

2.3.2.7.1 WebView

WebView permite introducir elementos HTML y JavaScript directamente desde la web, por lo tanto, también se incluyen los riesgos asociados a estas tecnologías como es el *JavaScript injection*.

Para poder usar JavaScript hay que habilitar la opción en WebView, en caso de no incluir JavaScript lo mejor es no activar dicha opción. Así, se puede evitar la exposición de la aplicación ante un posible *cross-site-scripting* de manera innecesaria.

No hay que fiarse de la información obtenida por HTTP, si es posible es mejor usar HTTPS y si no hay que verificar los datos. El método `addJavaScriptInterface()` no hay que exponerlo sobre HTTP ya que se incrementa notablemente el riesgo de ataque.

2.3.2.8 Validar las entradas

La falta de validación de las entradas es uno de los errores de seguridad más comunes. Los campos de entrada de las aplicaciones nativas para Android ya tienen incluidas

medidas para mitigar los riesgos. Lo mejor es ajustarse a los métodos ofrecidos por el SDK para construir las aplicaciones.

Si se está empleando código nativo hay que asegurarse que los datos recibidos por IPC tengan el formato esperado y el flujo no indique que vaya a haber un desbordamiento del buffer. Android ofrece una serie de mecanismos que reducen la explotación de estos errores pero no resuelven el problema, es más efectivo un cuidadoso control de los punteros y buffers.

Los fragmentos de SQL y JavaScript son los más expuestos ya que son lenguajes basados en cadenas y presentan problemas en los caracteres de fuga. Para evitar problemas con SQL, lo mejor es emplear consultas por parámetros ya que reduce las posibilidades de la consulta a únicamente los parámetros prestablecidos. Una buena selección de permisos de lectura y escritura también reducen notablemente los riesgos.

Empleando WebView los problemas en los campos de entrada están en elementos con funcionalidades en JavaScript, por lo tanto, el principal problema es el *cross-site-scripting* cuando se habilita la opción de incluir código JavaScript, por lo tanto, hay que habilitarlo cuando sea estrictamente necesario.

Otra recomendación es emplear formatos de datos bien estructurados y verificar que se adapta a la estructura. Las listas negras o el carácter de remplazo pueden ser buenas técnicas.

2.3.2.9 Manipulación de datos de usuario

En general, es mejor minimizar el uso de las APIs para acceder a datos confidenciales del usuario y si es posible, no almacenar o transmitir este tipo de información. Es muy recomendable valorar si con un hash o alguna función no reversible sería suficiente para nuestra aplicación.

Las aplicaciones con contraseñas y nombres de usuarios suelen requerir una política de privacidad que explica el uso que se les va a dar a esos datos. Por lo tanto, cuanto menos se acceden a los datos menores riesgos hay de incumplir la política de seguridad.

Si se introducen elementos de terceros, hay que realizar un pequeño estudio de que datos quedan expuestos y como va a afectar al usuario. Es evidente en servicios de terceros, pero no tanto en elementos publicitarios.

También se debe revisar que los datos sensibles del usuario no queden expuestos ante flujos IPC (especialmente permisivos), archivos compartidos o sockets de red.

Si hace falta un número de identificación del usuario, lo mejor es crear uno y guardarlo pero no utilizar el número de teléfono o IMEI.

El desarrollador debe tener en cuenta que los registros son públicos y, aunque temporales, se pueden leer en cualquier momento. A veces, se emplean para observar el funcionamiento de una aplicación y filtran información importante.

2.3.2.9.1 Manejo de credenciales

Por defecto, lo mejor es pedir las menos veces posibles las credenciales de usuario. De esta manera es más evidente un ataque de phishing y menos exitoso.

Siempre que sea posible, el nombre de usuario y la contraseña no deben ser almacenados. En su lugar, se recomienda hacer una autenticación inicial con nombre y contraseña y luego emplear un *short-lived* (un servicio de autenticación por token).

Android cuenta con controlador de cuentas que permite invocar un servicio basado en la nube y no almacena contraseñas en el dispositivo. Por un lado, permite verificar si los credenciales que se pasan son correctos para la aplicación. Por otro lado, se encarga de verificar la firma de las credenciales y da la opción de almacenarlo mediante KeyStore.

2.3.2.10 Criptografía

Android ofrece soporte para cifrado de todo el sistema de ficheros y canales de comunicación seguros.

En general, para recuperar información de manera segura es suficiente con URI y una comunicación HTTPS. Si se necesita una conexión segura punto a punto, lo mejor es emplear HttpsURLConnection o SSLSocket.

Si por necesidades de diseño de la aplicación se necesita implementar un protocolo propio. Si necesita números aleatorios seguros se pueden generar con la clase SecureRandom o claves criptográficas con Keygenerator.

Si es imprescindible almacenar la clave, lo mejor es emplear KeyStore de Android que ofrece un mecanismo de almacenamiento a largo plazo y recuperación de claves criptográficas asimétricas.

2.4 Aplicaciones de terceros

Google ofrece herramientas de depuración a través del SDK de Android y el ADT para Eclipse, pero para completar un buen análisis de seguridad se necesitan herramientas que permitan analizar los flujos de datos hacia la red, los datos almacenados en el dispositivo y el código fuente.

2.4.1 Herramientas para el análisis de la red

Las herramientas de análisis de la red sirven para visualizar los paquetes de los distintos protocolos y así poder obtener información sobre los flujos que genera una aplicación o un dispositivo concreto.

La aplicación más popular y completa es Wireshark, de software libre y multiplataforma. Se suele emplear para redes Ethernet y WiFi. También existe la opción de realizar el análisis desde el propio terminal en redes WiFi y 3G con tPacketCapture.

2.4.1.1 Wireshark

Wireshark está basado en la API *pcap* diseñada para la captura de paquetes de red y añade interfaz de usuario. La aplicación es capaz de filtrar más de 1100 protocolos y

mostrar la información de manera estructurada, con todos los campos de las cabeceras y capas de los paquetes capturados.

2.4.1.1.1 *Interfaces disponibles*

En primer lugar, hay que seleccionar la interfaz sobre la cual se va a realizar el estudio. Para ello, tiene un menú de captura donde permite iniciar, parar, volver a empezar y mostrar la distintas interfaces y sus opciones.

Sobre la interfaces disponibles, ofrece una pequeña descripción que ofrece el sistema operativo, la IP por la que Wireshark captura el flujo de datos, el número de paquetes y el ratio paquetes por segundo.

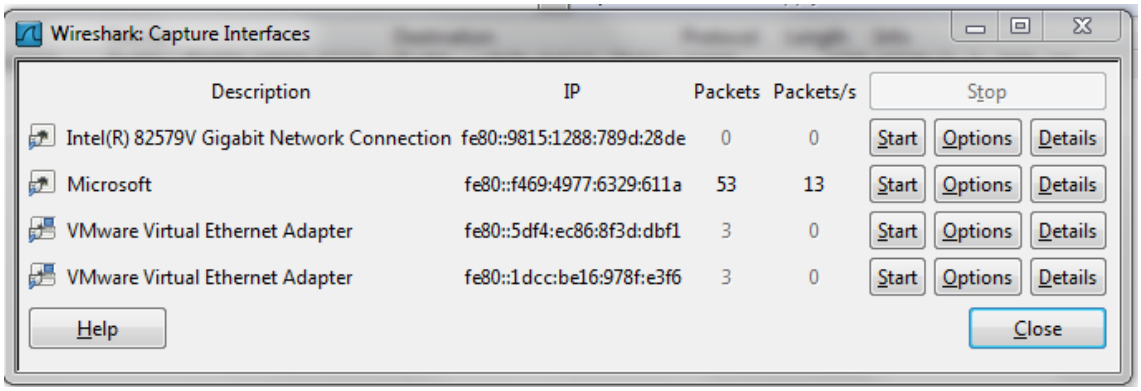


Ilustración 15 - Interfaces de captura

Para seleccionar que interfaz se va a analizar, se seleccionar *Start*. *Options*, ofrece multitud de opciones para la captura de datos. *Details*, informa sobre las propiedades de la interfaz y estadísticas de uso.

2.4.1.1.2 *Opciones de las interfaces*

Las opciones que ofrece son las siguientes (ver Ilustración 16):

- Sobre captura:
  - Tipo de cabecera de capa de enlace: Permite elegir entre Ethernet y *DOCSIS* (*Data Over Cable Service Interface Specification* o Especificación de Interfaz para Servicios de Datos sobre Cable). Son diferentes estándares de interfaz de conexión que se emplean en redes locales, se selecciona el tipo de interfaz que se esté empleando.
  - Captura de paquetes en modo promiscuo: permite la captura de todos los paquetes que fluyen por la red seleccionada.
  - Captura de paquetes en formato *pcap-ng*: el formato *pcap* ofrece más información sobre la captura, como el momento, estadísticas, nombres resueltos, comentarios…
  - Limite de tamaño por paquete: limite, en bytes, del tamaño máximo capturado por paquete.
  - Tamaño de buffer: limite, en megabytes, del tamaño del buffer de almacenamiento intermedio de paquetes capturados.
  - Filtros de captura: permite filtrar por cada campo de los distintos protocolos. Permite operaciones lógicas.

- Sobre ficheros de captura: (se puede capturar con otras herramientas o en otros lugares los paquetes y analizarlos en Wireshark).
    - Ruta del fichero: directorio donde se encuentra el fichero.
    - Múltiples ficheros: se marca si se van a emplear diferentes ficheros.
    - Siguiente fichero: indicar cada cuanto acaba un fichero (en tiempo o tamaño).
    - Llamada al buffer con: número de ficheros que se mantienen en el buffer.
    - Parar de capturar después: numero de ficheros que capturará antes de parar.
- Sobre parar de capturar:
    - Después de cierto número de paquetes: número de paquetes máximo que puede capturar.
    - Después de cierto tamaño: número de megabytes máximo que puede capturar.
    - Después de cierto tiempo: número de minutos máximo que puede estar capturando.
- Sobre opciones de visualización:
    - Actualizar lista de paquetes en tiempo real: actualiza automáticamente la lista de paquetes capturados.
    - Arrastre automático de la lista de paquetes: muestra la lista donde el último paquete capturado.
    - Ocultar dialogo de información de captura: no muestra un dialogo sobre la información de la captura realizada.
- Sobre opciones de nombre:
    - Activar nombre de MAC: muestra el nombre de MAC.
    - Activar nombre de red: muestra el nombre de la red.
    - Activar nombre de transporte: muestra el nombre del transporte.

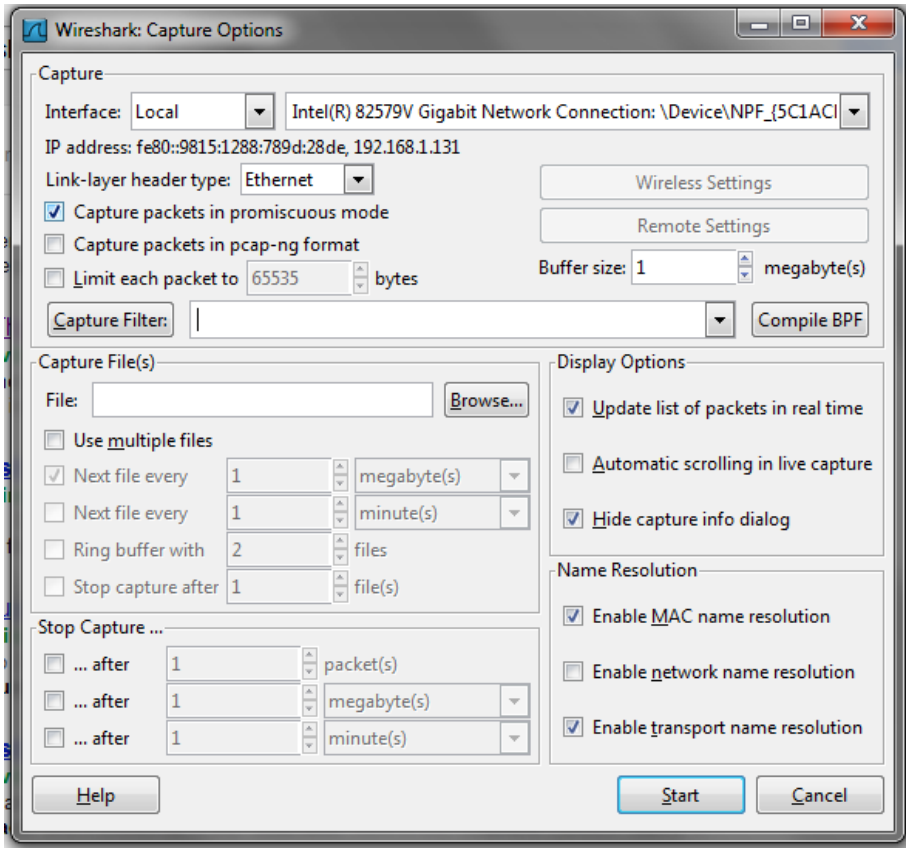
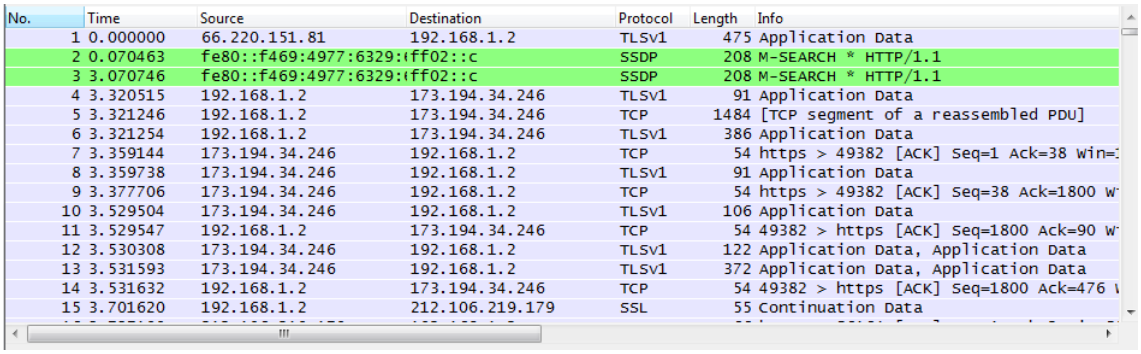


Ilustración 16 - Opciones de captura

2.4.1.1.3 *Información sobre paquetes capturados*

Una vez que se ha iniciado la captura de paquetes, Wireshark los lista en la pantalla principal de la aplicación como se puede ver en Ilustración 17 - Captura paquetes.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	66.220.151.81	192.168.1.2	TLSv1	475	Application Data
2	0.070463	fe80::f469:4977:6329:(ff02::c		SSDP	208	M-SEARCH * HTTP/1.1
3	3.070746	fe80::f469:4977:6329:(ff02::c		SSDP	208	M-SEARCH * HTTP/1.1
4	3.320515	192.168.1.2	173.194.34.246	TLSv1	91	Application Data
5	3.321246	192.168.1.2	173.194.34.246	TCP	1484	[TCP segment of a reassembled PDU]
6	3.321254	192.168.1.2	173.194.34.246	TLSv1	386	Application Data
7	3.359144	173.194.34.246	192.168.1.2	TCP	54	https > 49382 [ACK] Seq=1 Ack=38 Win=
8	3.359738	173.194.34.246	192.168.1.2	TLSv1	91	Application Data
9	3.377706	173.194.34.246	192.168.1.2	TCP	54	https > 49382 [ACK] Seq=38 Ack=1800 W
10	3.529504	173.194.34.246	192.168.1.2	TLSv1	106	Application Data
11	3.529547	192.168.1.2	173.194.34.246	TCP	54	49382 > https [ACK] Seq=1800 Ack=90 W
12	3.530308	173.194.34.246	192.168.1.2	TLSv1	122	Application Data, Application Data
13	3.531593	173.194.34.246	192.168.1.2	TLSv1	372	Application Data, Application Data
14	3.531632	192.168.1.2	173.194.34.246	TCP	54	49382 > https [ACK] Seq=1800 Ack=476
15	3.701620	192.168.1.2	212.106.219.179	SSL	55	Continuation Data

Ilustración 17 - Captura paquetes

Sobre cada paquete, describe una serie de campos básicos que ayudan a buscar alguna información genérica o un paquete en particular. Los campos descriptivos son los siguientes:

- No. : número de identificación del paquete dentro de la captura actual de Wireshark.
- Time: tiempo de captura respeto al memento en que empezó a capturar paquetes.
- Source: Dirección de origen del paquete.

- Destination: Dirección de destino del paquete.
- Protocol: Protocolo del mensaje capturado.
- Length: longitud del paquete en bytes.
- Info: Resumen del cuerpo o del significado del mensaje.

2.4.1.1.4 Contenido del paquete

La información detallada de cada paquete depende del protocolo que implemente, pero todos los paquetes tienen en común información sobre el frame: la información de la cabecera del paquete como el tiempo de llegada, tiempos relacionados con la captura, número de frame, tamaño del frame, protocolos que aparecen y regla que colorea el paquete.

Las siguientes líneas son información de cada nivel de la capa OSI, empezando por la capa de enlace, seguido de la capa de red, la capa de transmisión y la cada de aplicación. En función de cada protocolo, se añaden algunas capas de representación de datos o de información de sesión.

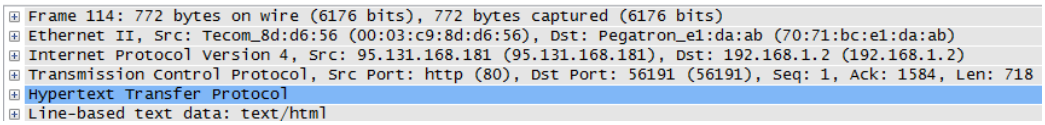


Ilustración 18 - Paquete HTTP capturado

Hay información que por espacio o por *confidencialidad* es trucada de la cabecera. En cualquier caso, la información también se muestra íntegramente tal y como se captura en hexadecimal y ASCII.

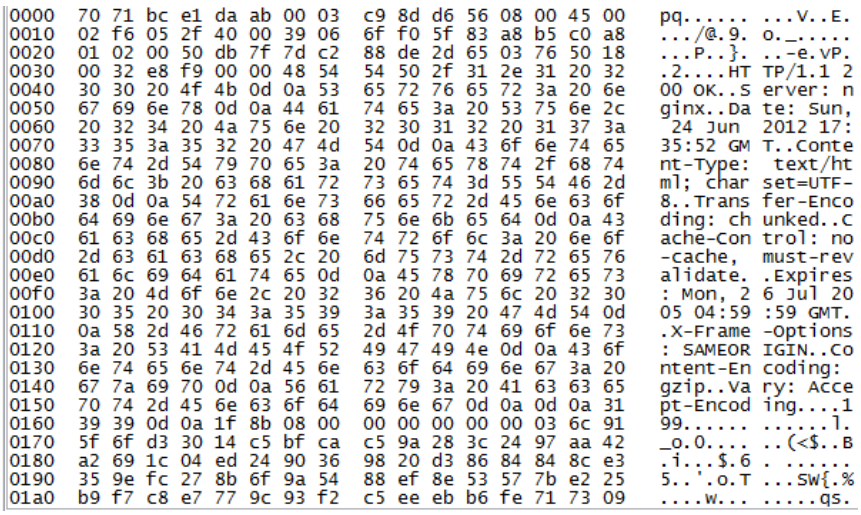


Ilustración 19 - Paquete en HEX y ASCII capturado

2.4.1.1.5 Filtros

En un dispositivo conectado, se puede capturar miles de frames en poco espacio de tiempo. Además, muchos de los frames son para mantener la conexión de algún servicio, peticiones del sistema y datos que nutren aplicaciones.

Los filtros se pueden introducir como opción de captura antes de empezar o una vez empezada la captura que solamente muestra los paquetes que se adaptan a las reglas del filtro.

En primer lugar, se puede filtrar por protocolos. Resulta muy útil para discriminar protocolos que no contienen información relevante o buscar patrones de ciertos protocolos.

Después, se puede filtrar de cada protocolo información de todos los campos. Por ejemplo, TCP contiene información sobre puertos que suele ser muy útil. Como podemos ver en la Ilustración 20 - Ejemplo cabecera TCP, un paquete que implementa HTTPS contiene un campo puerto origen, o destino, con valor 443 (el valor estandarizado para dicho protocolo de seguridad).

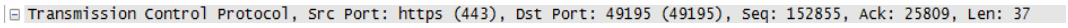


Ilustración 20 - Ejemplo cabecera TCP

Por lo tanto, si se quiere filtrar paquetes que implementen HTTPS hay tres maneras. Uno, conociendo la nomenclatura y estableciendo el filtro directamente en la barra *Filter*. Dos, mediante el asistente que se muestra en la Ilustración 21 - Filtro HTTPS. Tres seleccionar de un paquete HTTPS el campo del puerto 443 y, mediante el menú contextual, aplicar un filtro sobre dicho campo.

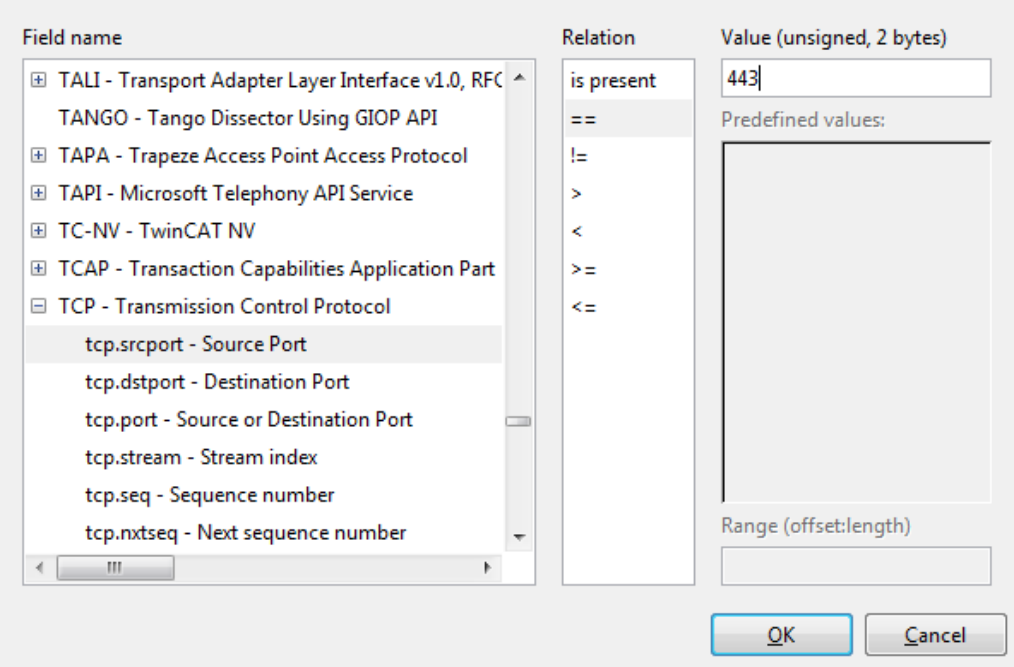


Ilustración 21 - Filtro HTTPS

Este procedimiento se puede realizar sobre cualquier campo de un protocolo y obtener información relevante. Por ejemplo, si la información o el canal por el que se envía no están cifrados, se podrán ver datos como contraseñas, números de tarjeta o identificación, etc….

Durante una sesión de laboratorio de la asignatura Ingeniería de la Seguridad se realizó un ejemplo de un sitio web que permite ver la contraseña y el usuario es el correo de Orange (http://correo.orange.es/). Ya se puede ver en el link adjunto, pero en el paquete TCP capturado, del envío de datos de un usuario ficticio, se puede observar que el puerto de destino es el 80 (HTTP) y no el 443 (HTTPS).

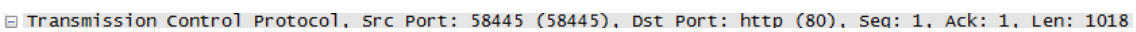


Ilustración 22 - Capa transmisión del paquete enviado a correo Orange

El usuario enviado es David, con dominio orange.es y contraseña “eslacontraseña”. En la línea de datos aparece la información truncada pero se puede ver en la información en bruto capturada. En la Ilustración 23 - Datos del paquete enviado a correo Orange se puede observar que aparece toda la información, incluyendo usuario y contraseña.

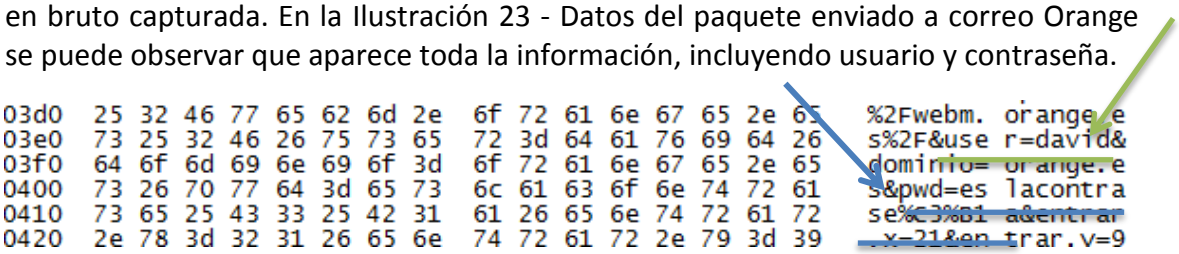


Ilustración 23 - Datos del paquete enviado a correo Orange

2.4.1.2 tPacketCapture

tPacketCapture es una herramienta que sirve para capturar los paquetes directamente desde el dispositivo Android. La principal ventaja respecto a Wireshark es que también captura los datos transmitidos por 3G. El único requisito es darle privilegios de root para poder interceptar todo el tráfico.

tPacketCapture genera un fichero *pcap* estándar, para leerlo se empelara el Wireshark con todas las funcionalidades vistas en el apartado 2.4.1.1.

En la Ilustración 24 - Actividad principal tPacketCapture se pueden ver las funciones principales:

- La ruta del fichero *pcap*: importante para recuperar los datos capturados y analizarlos desde Wireshark. También ofrece una opción de compartir el archivo por varios medios.
- Interruptor de capturar/ejecutando: Sirve para iniciar la captura de paquetes y poder pararla cuando sea necesario. Una captura bien acotada simplifica mucho la labor de búsqueda de datos concretos.
- Información sobre el fichero: únicamente muestra el tamaño que ocupa.
- Información sobre almacenamiento: espacio ocupado y espacio libre del dispositivo.
- Lista de ficheros *pcap*: listado de los archivos *pcap* que se mantienen en el dispositivo.

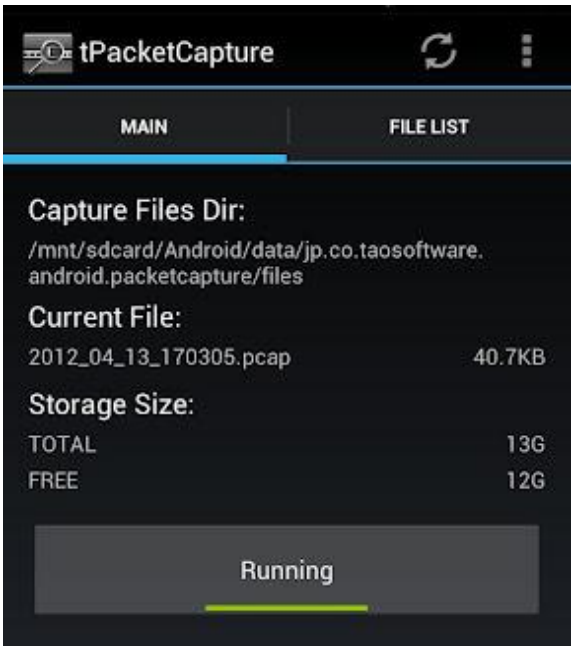


Ilustración 24 - Actividad principal tPacketCapture

2.4.2 Herramientas para el análisis de los datos almacenados

En el aparatado 2.3.2.3, se vio que no era recomendable guardar información sensible en el dispositivo. Por lo tanto, el análisis de una aplicación debe contemplar la posibilidad que sí estén almacenando esos datos.

Las aplicaciones suelen emplear SQLite para almacenar los datos en el dispositivo. También se pueden almacenar datos en archivos temporales o ficheros permanentes.

Los ficheros de bases de datos se pueden abrir desde el dispositivo con aSQLiteManager y los demás ficheros se pueden recuperar con el ADT y estudiar con un editor de texto.

2.4.2.1 aSQLiteManager

aSQLiteManager es una aplicación para Android que permite la visualización y modificación de bases de datos. Las bases de datos suelen guardarse en el directorio *data* de la carpeta de la aplicación y la extensión depende del desarrollador.

Las funcionalidades básicas son abrir una base de datos existente, realizar consultas SQL y ver las tablas, las vistas y los índices o crear una nueva base de datos.

2.4.2.1.1 Abrir base de datos

La funcionalidad para abrir una base de datos es basicamente un explorador de ficheros. Muestra todos los directorios y ficheros, sean o no bases de datos, aunque sólo permite abrir los ficheros compatibles con la aplicación.

Los ficheros válidos tienen multitud de extensiones, mientras tengan formato correcto, la aplicación aceptará el fichero. Normalmente, se emplea extensiones como .db .sql .sqlite.

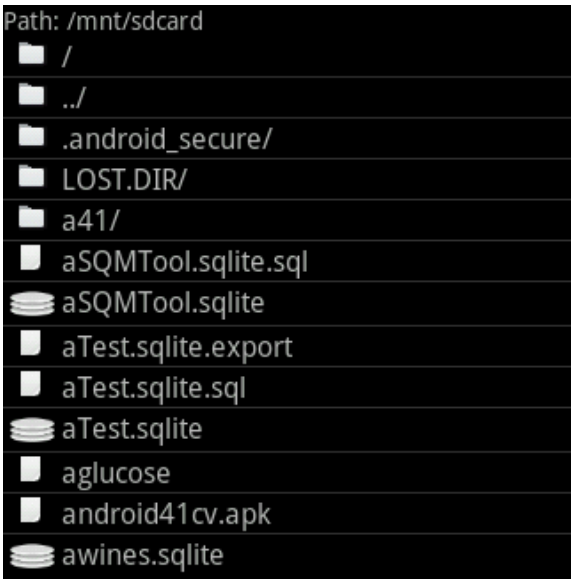


Ilustración 25 - Selección de base de datos

2.4.2.1.2 Ver información de base de datos

Las bases de datos se componen básicamente de tablas. Para mejorar el rendimiento, se crean índices y para mostrar cómodamente cierta información se crean vistas. Aparte, permite realizar consultas SQL y obtener cualquier información de la base de datos.

Las vistas se suele aplicar para facilitar la visión de ciertos datos y puede que no muestren información relevante. Los índices son herramientas de optimización que aceleran el acceso por clave no identificativa. Las consultas son para obtener información concreta de manera manual. Por lo tanto, lo más útil para realizar un análisis será la visualización de las tablas.

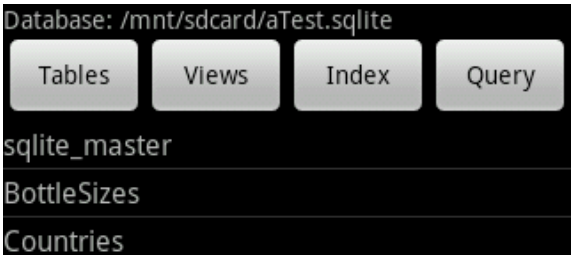


Ilustración 26 - Tablas de la BD

Seleccionando *Tablas* se muestra el listado de todas las tablas disponibles. Seleccionando una, se pasa a la actividad encargada de mostrar los campos de la tabla. Como se puede ver en Ilustración 27 - Campos de tablas, cada campo se identifica por:

- Identificador (Id): número de identificación único en esta tabla.
- Nombre (name): nombre del campo:
- Tipo (type): tipo de dato que contiene el campo.
- Obligatorio (notnull): indica si es obligatorio, es decir, si no puede ser null el valor.

- Valor por defecto (dflt_vlaue): valor que se estable por defecto en ciertos casos.
- Clave primaria (Pk): Indica si es clave primaria para identificación.

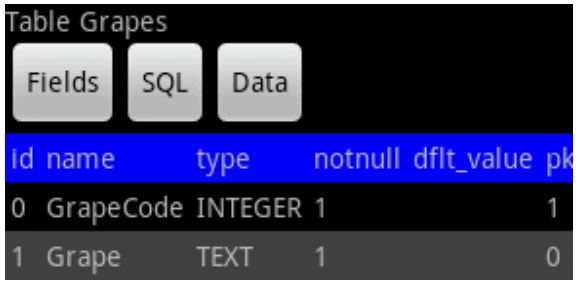


Ilustración 27 - Campos de tablas

El apartado SQL sirve para introducir un nuevo registro adaptándose a los campos existentes.

El apartado datos es la tabla que contiene los valores de cada campo. Con esta actividad de la aplicación, se puede buscar información sensible en claro en los registros. También permite la edición de los datos en la misma línea.

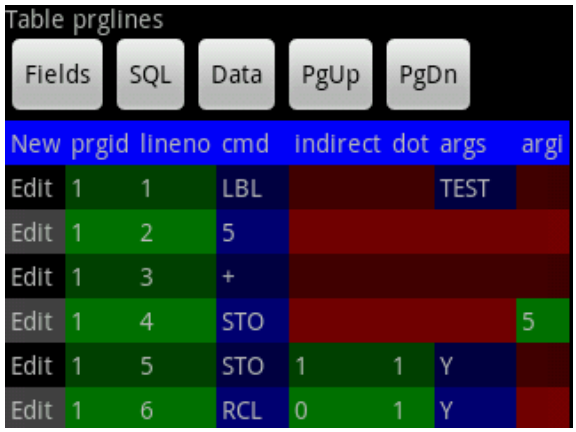


Ilustración 28 - Tabla de valores

2.4.3 Herramientas para el análisis del código fuente

En el código fuente se pueden encontrar vulnerabilidades que en un estudio superficial de los demás medios no aparecerían. Mediante un análisis de las clases se pueden identificar datos importantes, manejo de información sensible, envío de datos, etc…

Existen herramientas que permiten desempaquetar las aplicaciones .apk de Android en los .class que ejecuta máquina virtual de Java. Mediante otra herramienta, se puede obtener los ficheros fuente de las clases compiladas.

La herramienta para desempaquetar los .apk es *Dex2Jar*, es muy sencilla de utilizar y encontrar los resultados. La aplicación es para ejecutarla mediante la línea de comandos:

```
dex2jar app.apk
```

El resultado de esta aplicación es un fichero JAR que empaqueta las clases y mantiene el mismo nombre que la apk.

Un paquete JAR simplemente es un archivo que agrupa ficheros .class. La aplicación *JD-GUI* permite recuperar el fichero fuente de los .class. Como se pude observar en la Ilustración 29 - Código fuente de ficheros .class, es una aplicación gráfica que permite navegar por un sistema de ficheros y muestra el código original en la parte derecha de la aplicación.

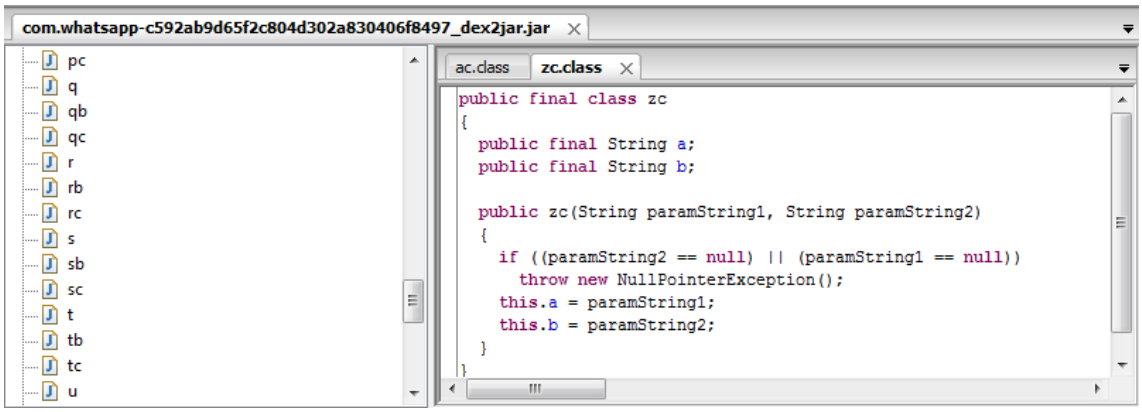


Ilustración 29 - Código fuente de ficheros .class

También permite la exportación de todos los ficheros fuente. Como la mayoría de los desarrolladores codifican sus aplicaciones para complicar la labor de búsqueda de información, es más sencillo buscar directamente sobre los ficheros clases o nombre de datos que puedan resultar interesantes.

2.5 Análisis de riesgos

El análisis de riesgos sirve para detectar los riesgos cualitativamente y poder cuantificarlos.

Lo primero es identificar los activos que van a ser protegidos, es necesario comparar el nivel de seguridad detectado inicialmente con el que se prestableció. En este caso, un criterio diferente se puede entender como que se está sobreprotegiendo o descuidando un activo.

Posteriormente, hay que evaluar el nivel razonable de seguridad que cumpla el objetivo de asegurar el activo.

La tabla que se empleará para el análisis de riesgos es la **¡Error! No se encuentra el origen de la referencia.**:

<table>
<tr><td rowspan="7">Activo</td><td colspan="5">Amenaza</td></tr>
<tr><td colspan="5"></td></tr>
<tr><td colspan="5">Vulnerabilidad explotada</td></tr>
<tr><td colspan="5"></td></tr>
<tr><td colspan="5">Prevención</td></tr>
<tr><td colspan="5"></td></tr>
<tr><td colspan="5">Respuesta de continuidad</td></tr>
<tr><td colspan="6"></td></tr>
<tr><td>Probabilidad</td><td>*</td><td>Impacto</td><td>*</td><td>Riesgo</td><td>*</td></tr>
</table>

Tabla 2 – Plantilla análisis de riesgos

*Los valores de probabilidad, impacto y riesgo se adaptarán a la siguiente valoración:

ATRIBUTO	BAJO	MEDIO	ALTO
PROBABILIDAD	No es un ataque muy frecuente. Suele ser por bajo interés y coste muy alto de recursos.	Es más o menos frecuente. Puede ser fácil de realizar aunque no sea muy alto el beneficio o por ser alto el beneficio pese a ser costoso de realizar.	Es muy frecuente. Suele ser por un alto interés para el atacante.
IMPACTO	El daño al usuario es bajo, no es imprescindible para el usuario.	El daño al usuario es medio, puede no ser muy importante para el usuario pero un ataque causa demasiadas molestias.	El daño es muy alto, causa grandes problemas al usuario.

ATRIBUTO	BAJO	MEDIO	ALTO
RIESGO	El peligro al que se expone el activo es asumible.	El peligro al que se expone el activo es moderado.	El peligro al que se expone el activo es inasumible.

Tabla 3 - Varemos de probabilidad, impacto y riesgo.

Los activos se seleccionaran por los grupos de aplicaciones indicados en la sección 2.1 ya que comparten tipos de datos muy similares. Los grupos pueden compartir activos muy genéricos como usuario y contraseña.

2.5.1 Aplicaciones de bancos

Las aplicaciones de bancos eran las que aparecen en la Ilustración 4. Estas aplicaciones manejan mucha información sensible como número de cuenta, saldo actual, movimientos bancarios, números de tarjetas, usuarios de acceso o contraseña de acceso.

2.5.1.1 Activo número de cuenta

El número de cuenta no es un dato especialmente importante, el único peligro potencial es que domicilien alguna factura a dicho número. En cualquier caso, el número de cuenta no debería ser un dato accesible sin conocimiento del usuario y ni se debería guardar en el dispositivo, únicamente debería ser transmitido desde el móvil al servidor y, a ser posible, codificado.

Activo	Amenaza				
Número de cuenta	Robo de número de cuenta. Junto con otros datos pueden domiciliar facturas.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Sólo consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
	Informar al usuario.				
Probabilidad	Media	Impacto	Medio	Riesgo	Bajo

Tabla 4 – Activo número de cuenta

2.5.1.2 Activo saldo actual

El saldo actual permite conocer el nivel económico de un usuario, por lo tanto, el dato no debería guardarse en el dispositivo. La transmisión del dato debería ser cifrada con un algoritmo fuerte para mantener la privacidad del usuario.

Activo	Amenaza
Saldo actual	Conocimiento del saldo. Pueden estar al tanto de la economía del usuario.
	Vulnerabilidad explotada

	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Sólo consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
	Realizar registro de consultas de saldo. El usuario es el que puede detectar alguna anomalía.				
Probabilidad	Media	Impacto	Medio	Riesgo	Bajo

Tabla 5 – Activo saldo actual

2.5.1.3 Activo movimientos bancarios

Los movimientos bancarios son datos privados que no ofrecen información que pongan en peligro el dinero, pero sirven para trazar hábitos de compras y modo de vida. La transmisión no debería ser en claro ni almacenada en el dispositivo.

Activo	Amenaza				
Movimientos bancarios	Conocimiento de compras y hábitos.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Sólo consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
	Realizar registro de consultas de movimientos. El usuario es el que puede detectar alguna anomalía.				
Probabilidad	Media	Impacto	Bajo	Riesgo	Bajo

Tabla 6 – Activo movimientos bancarios

2.5.1.4 Activo número de tarjeta

El número de tarjeta de crédito sirve para realizar pagos por Internet y como control de seguridad emplea 3 dígitos (tan solo 1000 posibilidades) y que aparece físicamente en la propia tarjeta. Por lo tanto, el número de tarjeta de crédito es sumamente importante y, por lo tanto, debe ser tratado como información sumamente sensible.

Activo	Amenaza
Número de tarjeta	Robo número de tarjeta. Con el número es relativamente sencillo realizar pagos por internet.
	Vulnerabilidad explotada
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.
	Prevención
	No almacenar, por lo menos en claro, en el dispositivo.

	Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Solamente consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
	Notificación de anomalías en los gastos.				
Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 7 – Activo número de tarjeta

2.5.1.5 Activo nombre de usuario

El nombre de usuario permite acceder a la aplicación y a todos los datos del usuario. El acceso debería estar controlado con una contraseña que debe insertar el usuario. El conocimiento de ambos datos permite identificarse al sistema como el usuario.

Activo	Amenaza				
Nombre de usuario	Robo de nombre de usuario. Permite, junto con contraseña, suplantación de identidad.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Sólo consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
	Si la se accede desde diferentes lugares, notificarlo al usuario. Si se trata de cambiar el nombre de usuario, pedir confirmación por otro medio (email, sms,…).				
Probabilidad	Alta	Impacto	Alto	Riesgo	Alto

Tabla 8 – Activo nombre de usuario

2.5.1.6 Activo contraseña de usuario

La contraseña de usuario es el elemento que da al usuario acceso a los datos privados de la aplicación, junto con el nombre de usuario (2.5.1.5). Este activo es el más importante ya que puede dar acceso a todos los demás sin explotar ninguna vulnerabilidad.

Activo	Amenaza
Contraseña de usuario	Robo de contraseña de usuario. Permite, junto con el nombre de usuario, suplantación de identidad.
	Vulnerabilidad explotada
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.
	Prevención
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Sólo consultas desde dispositivos autorizados y autenticados.
	Respuesta de continuidad
	Si la se accede desde diferentes lugares, notificarlo al usuario.

	Si se trata de cambiar la contraseña, pedir confirmación por otro medio (email, SMS,…).				
Probabilidad	Alta	Impacto	Alto	Riesgo	Alto

Tabla 9 – Activo contraseña de usuario

2.5.1.7 *Activo posición geográfica*

La posición geográfica permite trazar un patrón de conducta y conocimiento de la posición exacta a un atacante. Este tipo de información es privada, salvo que el usuario consienta por escrito que permite que sea pública.

Activo	Amenaza				
Posición geográfica	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación.				
	Respuesta de continuidad				
Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 10 – Activo posición geográfica

2.5.2 Aplicaciones de comunicación

Las aplicaciones de comunicación eran las que aparecen en la Ilustración 5. Estas aplicaciones manejan mucha información sensible como información personal (fechas de nacimiento, orientación sexual,…), información de contactos (su información personal, números de teléfono…), imágenes privadas, mensajes privados, posición geográfica o contraseñas de acceso.

2.5.2.1 *Activo información personal*

Información personal en este contexto abarca desde el nombre, apellidos, fecha de nacimiento hasta dirección de residencia, orientación sexual, etc… Es el usuario el que debe decidir si esta información es privada y para quién. Hay que hacer especial hincapié en la orientación sexual ya que la Ley orgánica de protección de datos protege de manera especial, como datos de vida sexual (LO 15/1999 Titulo II, Art 7, Apartado 3 sobre datos especialmente protegidos (7)).

Activo	Amenaza
Información personal	Robo de información personal.
	Vulnerabilidad explotada
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo. Mala gestión de permisos de lectura (solamente a los que autorice el usuario).

	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Correcta implementación del mecanismo discrecional que controla los privilegios de visibilidad de los datos personales.				
	Respuesta de continuidad				
	Suprimir la visualización de esta información, al menos la más importante, hasta solucionar problema. Si los datos se han visto comprometidos, restaurarlos desde una copia de seguridad.				
Probabilidad	Alta	Impacto	Alto	Riesgo	Alto

Tabla 11 – Activo información personal

2.5.2.2 *Activo información de contactos*

Sucede lo mismo que con el activo de información personal (2.5.2.1) pero con los datos que otro usuario permite acceder.

Activo	Amenaza				
Información personal de contactos	Robo de información personal del otro usuario.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo. Mala gestión de permisos de lectura (solamente a los que autorice el usuario).				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Correcta implementación del mecanismo discrecional que controla los privilegios de visibilidad de los datos personales.				
	Respuesta de continuidad				
	Suprimir la visualización de esta información, al menos la más importante, hasta solucionar problema. Si los datos se han visto comprometidos, restaurarlos desde una copia de seguridad.				
Probabilidad	Alta	Impacto	Alto	Riesgo	Alto

Tabla 12 – Activo información de contactos

2.5.2.3 *Activo imágenes privadas*

Las imágenes son un elemento que los usuarios consideran privado o parcialmente privado (deciden a quien se le puede mostrar). Las fotografías pueden llegar a ser comprometidas en cierto contexto y es muy importante protegerlas para que sea el usuario el que decida quien puede verlas.

Activo	Amenaza
Imágenes privadas	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.
	Vulnerabilidad explotada
	Almacenamiento en claro en el teléfono.

	Transmisión no cifrada entre servidor y dispositivo. Mala gestión de permisos de lectura (sólo a los que autorice el usuario).				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Correcta implementación del mecanismo discrecional que controla los privilegios de visibilidad de los datos personales.				
	Respuesta de continuidad				
	Suprimir el servicio de imágenes privadas hasta solucionar el error.				
Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 13 - Activo imágenes privadas

2.5.2.4 Activo mensajes privados

Los mensajes privados son conversaciones personales que mantienen entre dos usuarios y, por lo tanto, nadie más debería poder tener acceso a ellas.

Activo	Amenaza				
Mensajes privados	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo. Almacenamiento en claro en el servidor.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Solamente consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 14 – Activo mensajes privados

2.5.2.5 Activo posición geográfica

Es el mismo tipo de activo que el del punto 2.5.1.7.

Activo	Amenaza
Posición geográfica	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.
	Vulnerabilidad explotada
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.
	Prevención
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación.
	Respuesta de continuidad
	Si la se accede desde diferentes lugares, notificarlo al usuario.

Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 15 – Activo posición geográfica

2.5.2.6 *Activo contraseña de usuario*

Es el mismo tipo de activo que el del punto 2.5.1.6.

Activo	Amenaza				
Contraseña de usuario	Robo de contraseña de usuario. Permite, junto con el nombre de usuario, suplantación de identidad.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Sólo consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
	Si la se accede desde diferentes dispositivos, notificarlo al usuario. Si se trata de cambiar la contraseña, pedir confirmación por otro medio (email, SMS,…).				
Probabilidad	Alta	Impacto	Alto	Riesgo	Alto

Tabla 16 – Activo contraseña usuario

2.5.3 Aplicaciones con login

Las aplicaciones con login eran las que aparecen en la Ilustración 6. Estas aplicaciones manejan información privada, aunque no sea especialmente sensible, el usuario ha considerado que no sean del dominio público. Por ejemplo, documentos, notas o contraseñas de acceso.

2.5.3.1 *Activo documentos*

Los documentos suelen contener información confidencial, puede ser documentación de actividad académica, profesional, contabilidad, personal, etc… Este tipo de información se debe mantener privada para el usuario, o usuarios con los que se ha compartido.

Activo	Amenaza
Documentos	Obtención de información y documentación privada.
	Vulnerabilidad explotada
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.
	Prevención
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación.
	Respuesta de continuidad
	Llevar registro de consultas al documento.

Probabilidad	Alta	Impacto	Medio	Riesgo	Medio

Tabla 17 – Activo documentos

2.5.3.2 Activo notas

En realidad, este activo es muy parecido a un documento. Las aplicaciones con notas emplean pequeños documentos de texto que se suelen emplear a modo de recordatorio. Las notas, normalmente, se pueden compartir y el usuario debe ser quien elija la visibilidad de la nota.

Activo	Amenaza				
Notas	Obtención de información privada.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación.				
	Respuesta de continuidad				
	Llevar registro de consultas al documento.				
Probabilidad	Alta	Impacto	Medio	Riesgo	Medio

Tabla 18 – Activo notas

2.5.3.3 Activo imágenes privadas

Es el mismo tipo de activo que el del punto 2.5.2.3.

Activo	Amenaza				
Imágenes privadas	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo. Mala gestión de permisos de lectura (sólo a los que autorice el usuario).				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Correcta implementación del mecanismo discrecional que controla los privilegios de visibilidad de los datos personales.				
	Respuesta de continuidad				
	Suprimir el servicio de imágenes privadas hasta solucionar el error.				
Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 19 - Activo imágenes privadas

2.5.3.4 Activo contenido de pago

Algunas aplicaciones de login son para acceder a contenido previo pago. Es lógico que los usuarios quieran mantener ese contenido para ellos y, además, las aplicaciones

suelen protegerse de reproducciones simultáneas para evitar que varios usuarios se beneficien de sus servicios con un solo pago.

Activo	Amenaza				
Contenido de pago	Acceso contenido de pago y posible negación de servicio a usuario legítimo.				
	Vulnerabilidad explotada				
	Aunque es poco probable, almacenamiento en el teléfono. Transmisión no cifrada entre servidor y dispositivo. Acceso sin autenticación fuerte.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Autenticar al usuario del servicio.				
	Respuesta de continuidad				
	Notificar de utilización de contenido en varios dispositivos simultáneamente. Solicitar un cambio de contraseña al detectar simultaneidad en la utilización del contenido.				
Probabilidad	Alta	Impacto	Medio	Riesgo	Medio

Tabla 20 – Activo contenido de pago

2.5.3.5 *Activo contraseña de usuario*

Es el mismo tipo de activo que el del punto 2.5.1.6.

Activo	Amenaza				
Contraseña de usuario	Robo de contraseña de usuario. Permite suplantación de identidad junto con el nombre de usuario o el email y, posteriormente, realizar compras.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación. Sólo consultas desde dispositivos autorizados y autenticados.				
	Respuesta de continuidad				
	Si la se accede desde diferentes dispositivos, notificarlo al usuario. Si se trata de cambiar la contraseña, pedir confirmación por otro medio (email, SMS,…).				
Probabilidad	Alta	Impacto	Alto	Riesgo	Alto

Tabla 21 – Activo contraseña de usuario

2.5.3.6 *Activo posición geográfica*

Es el mismo tipo de activo que el del punto 2.5.1.7.

Activo	Amenaza
Posición geográfica	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.
	Vulnerabilidad explotada

	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación.				
	Respuesta de continuidad				
	Si la se accede desde diferentes lugares, notificarlo al usuario.				
Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 22 – Activo posición geográfica

2.5.4 Aplicaciones de consulta

Las aplicaciones de consulta eran las que aparecen en la Ilustración 7. Estas aplicaciones deberían manejar poca información personal, son aplicaciones de consulta de información y, por lo tanto, el usuario introduce poca información sensible.

2.5.4.1 Activo posición geográfica

Es el mismo tipo de activo que el del punto 2.5.1.7.

Activo	Amenaza				
Posición geográfica	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.				
	Vulnerabilidad explotada				
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.				
	Prevención				
	No almacenar, por lo menos en claro, en el dispositivo. Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación.				
	Respuesta de continuidad				
	Si la se accede desde diferentes lugares, notificarlo al usuario.				
Probabilidad	Alta	Impacto	Alto	Riesgo	Medio

Tabla 23 – Activo posición geográfica

2.5.4.2 Activo información en caché

La información en caché puede servir para trazar preferencias personales, inclinaciones políticas, religiosas, etc.

Activo	Amenaza
Información en caché	Ruptura de la supuesta confidencialidad ofrecida por la aplicación.
	Vulnerabilidad explotada
	Almacenamiento en claro en el teléfono. Transmisión no cifrada entre servidor y dispositivo.
	Prevención
	No almacenar, por lo menos en claro, en el dispositivo.

	Transmitir cifrado, ya sea el canal (SSL) o el dato a nivel de aplicación.				
	Respuesta de continuidad				
	Si la se accede desde diferentes lugares, notificarlo al usuario.				
Probabilidad	Medio	Impacto	Medio	Riesgo	Bajo

Tabla 24 – Activo información en caché

2.6 Relación de herramientas con riesgos

La mayoría de los activos analizados se ven expuestos a una amenaza mediante almacenamiento en claro en el teléfono o transmisión no cifrada entre servidor y dispositivo. Las herramientas analizadas en el apartado 2.4 servirán para estudiar los diferentes puntos de la aplicación.

HERRAMIENTAS COMUNICACIÓN	HERRAMIENTAS ALMACENAMIENTO	HERRAMIENTAS CÓDIGO FUENTE
Servirán para comprobar si los datos transmitidos están cifrados o no.	Servirán para conocer si almacenan datos soporte secundario. Y, en caso de almacenarse, estudiar si están cifrados.	Servirá para conocer si la aplicación trata de proteger los activos.

Tabla 25 - Utilización de las herramientas

3 Diseño

El objetivo es diseñar una serie de pruebas Las pruebas se dividirán básicamente en tres grupos:

- Pruebas de comunicación: cada aplicación transmitirá la información de una manera concreta, en función de como sea transmitida así deben ser las pruebas.
- Pruebas de almacenamiento: cada aplicación guarda en su carpeta, si es que guarda información, los archivos o bases de datos que emplea. El objetivo será crear un método de búsqueda que permite asegurar la obtención de todos los archivos.
- Pruebas de código: las pruebas de código variarán en función de si los nombres de las clases son claros y descriptivos o no. En caso de ser claros, se buscarán las clases que manejen los datos comentados en los activos. En caso de no serlo, habrá que buscar nombres de clases genéricas de la API que permitan conocer la funcionalidad que es está implementando en cada caso.

El objetivo de las pruebas es encontrar vulnerabilidades sobre los activos descritos en el apartado 2.5.

3.1 Diseño general de pruebas comunicación

Las pruebas de comunicación tienen como objetivo detectar como se transmite la información mediante la captura de paquetes transmitidos. Es primordial realizar un seguimiento de los diferentes protocolos y tipos de paquetes para detectar el inicio o fin de ciertas acciones.

En primer lugar, los datos en claro se transmiten, normalmente, con el protocolo HTTP. Por lo tanto, hay que filtrar los paquetes HTTP y analizar la sección de datos en busca de elementos marcados como activos de la aplicación.

A continuación, hay que comprobar que los datos están siendo transmitidos por HTTPs. Existen varias posibilidades, pero la más sencilla es buscar los paquetes que tiene como puerto destino el 443.

Las aplicaciones se prueban sin información almacenada en la base de datos del dispositivo o en caché. Es decir, las pruebas se realizan como si la aplicación estuviera recién instalada.

3.2 Diseño general de pruebas almacenamiento

Las pruebas de almacenamiento se realizarán si en el análisis del código se ha detectado una conexión con una base de datos local (SQLite) o algún tipo de creación de fichero para apoyar la persistencia de los datos de una aplicación.

En primer lugar, hay que comprobar si existe la carpeta de esta aplicación y que tipo de contenidos tiene. En este caso, no ha creado una carpeta para almacenar ficheros o archivos de bases de datos pero aun hay que comprobar, a través del código, si no emplea otro lugar para almacenar.

A continuación, hay que comprobar si se validan las entradas. El procedimiento más sencillo es comprobar el código pero es muy probable que no se tenga acceso a la

totalidad del código fuente página. También es posible realizar pruebas por fuerza bruta en los campos de entrada para comprobar que se mantiene la integridad y confidencialidad de los ficheros y/o bases de datos.

Finalmente, hay que demostrar si los datos que se almacén están cifrados o en claro. Sucede lo mismo que con la validación de entradas, lo más sencillo y potente es observar el código pero no es posible en su totalidad. Por lo tanto, si se acceden a los archivos o bases de datos hay que comprobar si los contendidos son legibles o codificados.

3.3 Diseño general de pruebas de código

Las pruebas de código es demasiado complejo realizar un mecanismo automático, además de poco eficaz. El método más efectivo es buscar ciertos elementos y clases importadas. Para cada aplicación, se puede destacar una serie de clases que parezca importantes y estudiar los siguientes puntos:

- Clases de java.security: elementos cifrados/firmados. Si está presente y se emplean para activos señalados servirá para fortalecer la seguridad de la aplicación.
- Clases de java.database: conexiones con bases de datos. La presencia de estas clases implica que la aplicación guarda información en una base de datos que tendrá que ser examinada. El diseño de la base de datos es desconocido, por lo tanto, no se puede automatizar el proceso y habrá que recorrerlo manualmente.
- Clase java.io.File: o clases heredadas para ficheros sobre el dispositivo. Si esta empleando un fichero como caché o almacenamiento permanente de información puede que emplee alguna de estas clases.
- Todos los métodos de comunicación entre procesos: intents, binder, broadcast, servicios y actividades. Son elementos propios de Android empleados para el envío de datos en distintos procesos. Como se ha estudiado durante el apartado 2.3.2, es muy recomendable emplear intents ya que verifican destinatario y ofrecen mucha seguridad. Los binder deben realizar las comprobaciones de control de accesos requeridas para cada uno. Los recibidores de broadcast, los servicios y las actividades se marcan como tal en el manifiesto y ofrecen bastante seguridad a nivel de sistema operativo.
- Como ya se ha comentado, las funciones principales se realizan mediante un WebView de la página para móviles de gestión de banca. Según lo visto durante el apartado 2.3.2, la ejecución de JavaScript debe estar habilitada sólo si se va a ejecutar código de este tipo y el método `addJavaScriptInterface()` debe estar bajo HTTPs. Por lo tanto, esta actividad debe tener activado SSL.

3.4 Diseño de pruebas para grupo de aplicaciones bancarias

En este grupo de aplicaciones, el aspecto más crítico de una aplicación bancaria es perder el control del dinero.

3.4.1 Santander

La aplicación parece ser que trata de ser internacional pero ofrece máximo rendimiento en España, no habrá problemas para realizar las pruebas pero es importante tenerlo en cuenta. Google Play considera el estado de desarrollo de la aplicación poco maduro. Las operaciones que ofrece son similares a las aplicaciones anteriores. Lleva aproximadamente un año sin actualizarse y podría existir alguna vulnerabilidad no cubierta (Agosto del 2011).

3.4.1.1 Diseño especifico de pruebas de comunicación

Para esta aplicación se va a especificar el camino que debe realizar el usuario de la aplicación para, posteriormente, asociar los paquetes con las acciones realizadas.

1. Pulsar sobre SuperNet Móvil
2. Seleccionar NIF
3. Introducir NIF: 50228467D.
4. Introducir contraseña: 7**0**1*. (No se muestran todos los dígitos por seguridad).
5. Seleccionar cuentas
6. Pulsar sobre←, para volver al menú de la sección.
7. Pulsar sobre tarjetas.
8. Pulsar sobre←, para volver al menú de la sección.
9. Pulsar sobre transferencias.
10. Entre mis cuentas.
11. (La cuenta de cargo y de destino debería ser diferente, pero se probará entre misma cuenta por si hace la comprobación en el servidor)(Nota, por la configuración de la cuenta de prueba no es posible probar entre distintas cuentas).
12. Introducir observación: Prueba
13. Introducir importe EUR: 1,00
14. Pulsar sobre continuar.
15. Pulsar sobre posición global.
16. Pulsar sobre oficinas y cajeros.
17. Pulsar sobre usar mi ubicación.
18. Pulsar botón atrás.
19. Introducir código postal: 28044.
20. Pulsar sobre 2844 Madrid, España.
21. Pulsar atrás.
22. Pulsar sobre desconectar. (Dinero directo y activar tarjetas son funciones que no es posible realizarlas)

3.4.1.2 Diseño especifico de pruebas de código

En primer lugar, tras un vistazo rápido de la jerarquía de clases de la aplicación, se puede ver que los nombres las clases son descriptivos y se puede saber cual es la función de cada una. La mayoría de las clases son para gestionar la localización geográfica. Las funciones principales relacionadas con gestiones bancarias se llevan a través de una clase que parsea la web móvil.

Los métodos específicos de banca no están implementados en código nativo y los obtiene de su aplicación web. Por lo tanto, las pruebas sobre la seguridad deben ajustarse más a criterios de páginas web que a aplicaciones Java o Android, exceptuando lo relacionado con posición geográfica.

Las clases que, a priori, resultan más interesantes para el estudio de la seguridad de la aplicación son:

- Clase supernet: buscar los datos que obtiene de la página web y los que proporcionan los usuarios.
- Clase tarjetas: posiblemente trate la información de número de tarjetas, contraseñas, etc…
- Clase myLocation: será la encargada de conseguir la localización actual del dispositivo y, por lo tanto, hay que estudiar como se emplea el dato obtenido.
- Clase comoLlegar/mapaComoLlegar: para indicar como llegar a una oficina necesita la direcciona actual.

3.4.2 Bankia

La aplicación de Bankia para dispositivo Android lleva relativamente poco en el mercado, salió en el tercer trimestre del 2011 (8) y Google Play considera la aplicación poco madura, es decir, no aprovecha al máximo las posibilidades que ofrece Google a nivel de diseño y calidad de la aplicación. En cualquier caso, la aplicación es totalmente funcional, permite ver los datos de nuestras cuentas bancarias y realizar operaciones como transferencias, pagos, etc…

3.4.2.1 Diseño especifico de pruebas de código

Los nombres de los paquetes y de las clases son muy descriptivos y se puede intuir la estructura de la aplicación y que elementos resultan interesantes de estudiar. Esta aplicación parece completamente nativa.

Observando la jerarquía de las clases que componen la aplicación, se puede observar que la aplicación se compone de dos capas. Una capa es el núcleo, el conjunto de clases que se encarga de las funciones más básicas (conexión, precarga, localizador, sesión). La otra capa es más orientada a la aplicación en si misma (menús, búsquedas, barras).

Las clases que, a priori, resultan más interesantes para el estudio de la seguridad de la aplicación son:

- Bankia.core:
    - App (aplicación):
        - AppSession (sesión de aplicación): se encargará de mantener la sesión activa entre el cliente y el servidor. Si esta clase no se en encarga de cifrar los datos de sesión y de autenticar al usuario puede provocar que el usuario sufra suplantación de identidad y, por lo tanto, el atacante tendría total acceso a sus gestiones financieras.
    - HTTP: será el encargado de crear un mantener una conexión HTTP con el servidor. En esta clase se pueden encontrar datos de sesión e información sobre sistemas de seguridad empleados.
    - Locator (localizador): es la clase encargada de localizar geográficamente al usuario. Se estudiará como se tratan los datos obtenidos y como nutre a la aplicación con estos datos.

- Log (registro): esta clase se encargará de llevar un control sobre todo lo que realiza la aplicación. El objetivo es observar como almacena la información y que información es.
- TrustAllManager (manejador de “confianza en todo”): la clase contiene únicamente las cabeceras de funciones de comprobación de confianza de servidor y cliente.
- TrustAllSSLSocketFactory: esta clase será la encargada de crear sockets son SSL activado. Se puede estudiar que mecanismo emplea y si existen vulnerabilidades conocidas.

- Cm.android:
  - Activity (actividad):
    - BankiaActivity (Actividad Bankia): parece ser la principal de acceso puede obtener o solicitar algún dato del usuario.
    - FinderActivity (Actividad Buscador): es la clase que se encarga de obtener de la localización geográfica del usuario.
    - HomeActivity (Actividad inicial): seguramente se trate de la actividad en la que se encuentra el menú principal, puede tener acceso a todos los elementos de la aplicación desde aquí.
    - OfmovilActivity (Actividad Oficina móvil): es la sección donde permite al usuario realizar las gestiones sobre sus cuentas, puede tener acceso a todos los elementos de la aplicación desde aquí.
  - Delegates (delegaciones):
    - AccountDelegate (delegación de cuenta): clase delegada para gestionar los datos de la cuenta. Interesa tanto si se refiere a cuenta bancaria como a cuenta de usuario de la aplicación.
    - CreditCardDelegate (delegación de tarjeta de crédito): clase delegada para gestionar las tarjetas de crédito, tendrá acceso al número de cuenta, a los movimientos y al saldo actual.
    - CreditCardTransactionDelegate (delegación de transacción con tarjeta de crédito): clase delegada para gestionar las transacciones realizadas con tarjetas de crédito. Tendrá acceso a los mismos datos que la anterior.
    - DepositDelegate (delegación de depósitos): clase delegada para gestionar depósitos bancarios.
    - DisposalBinderDelegate (delegación de binder de préstamos): clase delegada para gestionar el binder de préstamos. Tendrá acceso a información económica del usuario.
    - IncomeBinderDelegate (delegación de binder de ingresos): clase delegada para gestionar el binder de ingresos. Tendrá acceso a información económica del usuario.
    - IrpfBinderDelegate (delegación de binder de IRPF): clase delegada para gestionar el binder de IRPF. Tendrá acceso a información detallada y extensa sobre la económica del usuario.
    - KeysTransactionDelegate (delegación de transacción de claves): clase delegada para gestionar de claves.

        - LoanDelegate (delegación de préstamo): clase delegada para gestionar los préstamos.
        - LossBinderDelegate (delegación de Binder de pérdidas): clase delegada para gestionar el binder que reúne las pérdidas.
        - ReceiptsTaxDelegate (delegación de recibo de impuestos): clase delegada para gestionar los recibos de los impuestos. Normalmente, cuentan con número de cuenta e información fiscal.
        - RechargeBinderDelegate (delegación de Binder de recargo): clase delegada para gestionar el binder que reúne funciones sobre los recargos en una cuenta o tarjeta. Puede aparecer información sobre la cuenta y sus movimientos.
        - RefundBinderDelegate (delegación de Binder de rembolso): clase delegada para gestionar el binder que reúne funciones sobre rembolso. Puede aparecer información de la cuenta de origen y de destino.
        - TransactionBinderDelegate (delegación de Binder de transacción): clase delegada para gestionar el binder que reúne funciones sobre transacciones.
        - TransactionDelegate (delegación de transacción): clase delegada encarga de la gestión de transacciones. Puede manejar información sobre las cuentas del usuario y los momentos de las mismas.
        - TransferBinderDelegate (delegación de Binder de transferencia): clase delegada para gestionar el binder que reúne funciones sobre transferencia.
        - TransferDelegate (delegación de transferencia): clase delegada encarga de la gestión de transferencias. Puede manejar información sobre las cuentas del usuario y los momentos de las mismas.
    - Filter (Filtro):
        - StackUpFilterEvaluator (Filtro de evaluación de pila): probablemente, es la clase delegada para verificar ciertos parámetros
    - Finder (buscador):
        - AddressWrapper (Envoltorio de dirección): se encargará de formar un dato manejable con la dirección obtenida y formatea su contenido.
        - AssetPopup (activos emergentes): Esta clase manejará los datos emergentes en relación con la posición geográfica.
        - RouteActivity (Actividad de ruta): esta actividad es la encargada de generar una ruta entre la posición actual y una sucursal.

3.4.3 BBVA

La aplicación del BBVA es similar al resto de permite visualización de saldo en diferentes cuentas o tarjeta, visualización de valores, transacción, transferencias, etc… Según Google Play, el nivel de madurez de la aplicación es mayor que en las anteriores.

3.4.3.1 Diseño especifico de pruebas de código

Al igual que la aplicación de Bankia, la del BBVA se divide en dos capas. Una es el núcleo de configuración básica y otra es la puramente de la aplicación.

En el núcleo vemos que las clases no están bien descritas y no se puede formar una idea general de sus funciones. En la parte de las funciones, la jerarquía es mucho más clara y se puede entender fácilmente para que esta destinado cada paquete, actividad, servicio…

Entre las funciones, destacan cuatro paquetes: cryto, io, location y model. Cryto contendrá las clases encargadas de la criptografía de la aplicación. IO será el paquete con clases para gestionar la entrada y salida de datos. Location gestionará la posición geográfica del usuario. Finalmente, model contiene todas las clases delegadas para obtener información.

Las clases que, a priori, resultan más interesantes para el estudio de la seguridad de la aplicación son:

- Mcb:
  - MCBManager: probablemente lleve la gestión de todos los recursos de los que dispone la aplicación. Lo más interesante será ver si emplea SSL y la gestión la conexión con la base de datos.
- Nxt:
  - Cryto:
    - Cryto (criptografía):
  - IO:
    - HttpInvoker (invocador de HTTP):
    - NotificationCenter (centro de notificaciones):
    - PermissiveSocketFactory (socket permitidos):
    - Updater (actualizador):
  - Location:
    - LocationMonitor (monitor de localización): monitor de localización del usuario. El estudio se centrará en el tratamiento de los datos obtenidos del sensor.
  - Model:
    - AccountList (lista de cuentas), AccountListResponse (respuesta lista de cuentas),AccountTransactionsFilter(filtro de transacciones de cuentas):
    - AppConfigurationResponse (Respuesta de configuración de aplicación): esta clase envía datos de la configuración al servidor. Puede tener información de cualquier tipo sobre la aplicación y sus cuentas.
    - Authorization (autorización), AuthorizationList (lista de autorizaciones), AuthorizationsResponse(respuesta de autorizaciones): Son las clases encargadas de la gestión de usuarios y servidores autorizados. Se encargaran de la autenticación.

- BankAccount (cuenta bancaria): Clase que engloba todos los aspectos de la cuenta bancaria. Probablemente, tenga acceso a todos los elementos de la cuenta bancaria (saldo, numero de cuenta, movimientos,..)
- Card (tarjeta),CardList (lista de tarjetas),CardTransactionsFilter (filtro de transacciones con tarjeta): información sobre las tarjetas relacionadas con un usuario y un filtro de las transacciones que ha realizado con ellas.
- Carrier (portador), CarrierList (Lista de portadores): probablemente se trate de los datos del portador de una tarjeta de crédito.
- Contact (Contacto): datos personales del contacto de referencia una tarjeta o cuenta bancaria.
- Deposit (depósitos),DepositList (lista de depósitos): lista de depósitos que contienen información bancaria del usuario.
- ExtendedOperationSummaryInfo (información resumida de operaciones extendidas): probablemente ofrezca un resumen detallado de todos los movimientos y saldos de las distintas cuentas.
- GeographicCoordinates (posición geográfica): se encargaran de obtener y tratar las coordenadas geográficas. Será interesante ver como trata los parámetros de longitud y latitud.
- Loan (préstamo),LoanList (lista de préstamos): lista de prestamos e información sobre ellos.
- LoginResponse (Respuesta de login): probablemente, sea la respuesta de credenciales de autenticación por parte del servidor. Puede tener información del servidor y del cliente.
- MapRequest (Petición de mapa): probablemente sea la clase encargada de enviar al servidor de mapas la posición actual.
- MobileRecharge (recarga móvil), MobileRechargeInputResponse (repuesta de entrada de recarga móvil), MobileRechargeList (lista de recarga móvil), MobileRechargeResultResponse (Respuesta del resultado de la recarga), PrepaidCardDischargeSetupResponse (respuesta de configuración de descarga de tarjeta prepago), PrepaidCardRechargeSetupResponse (respuesta de configuración de recarga de tarjeta prepago), PreviousMobileRechargesResponse (Respuesta de recarga de móvil previa): clases que dan el servicio de recarga el saldo de los móviles prepago. Puede obtener información sobre el proceso, el número de cuenta o el saldo.
- ServerCard (tarjeta de servidor), ServerCardList (lista de tarjetas de servidor), ServerContact (Contacto de servidor), ServerContactList(Lista de contactos de servidor): parecen ser aplicaciones auxiliares para probar el servidor. Al ser clases de prueba, tienen el mismo acceso que el resto de la aplicación a información sensible.

- SharePaymentSummaryInfo (Resumen de información de pago compartida): esta clase tendrá acceso a pagos, saldo y movimientos para poder realizar el resumen.
- StatusEnabledResponse (Repuesta de estado activo): mandará mensaje al servidor de que esta activo. Un fallo de seguridad aquí podría significar la falta de disponibilidad de la aplicación para el usuario.
- Transaction (transacción), TransactionList (Lista de transacciones): lista de transacciones que mostrar los movimientos de cierta cuenta y datos sobre los extremos (emisor y receptor de la transacción).
- TransferSetupResponse (Respuesta de configuración de transferencia): clase que se encargará de configurar las transferencias y enviar la información al servidor.

3.4.4 ING Direct

La aplicación de ING Directa también cuenta con un grado de desarrollo no muy maduro, según Google Play. Las operaciones que ofrece son las mismas que las anteriores.

3.4.4.1 Diseño especifico de pruebas de código

La aplicación tiene una estructura clara y descriptiva que permite entender la arquitectura de la aplicación. La aplicación cuenta con cuatro paquetes principales: com, junit, net y org. Dentro de com esta la aplicación en sí misma, en el paquete phonegap. Junit contiene las pruebas unitarias que le realizaron. Y net y org contienen paquetes auxiliares empleados en criptografía.

Las clases que, a priori, resultan más interesantes para el estudio de la seguridad de la aplicación son:

- net.sf.andpdf.crypto: directamente vemos una clase llamada, RC4Cipher. Por lo tanto, como no se va a profundizar más allá de conocer el mecanismo de cifrado, no es necesario analizar el código de las clases.
- org.bouncycastle.crypto: sucede parecido que en el paquete anterior. En el paquete engines, se encuentra la clase RC4Engine. Es decir, el mecanismo de cifrado empleado también es RC4.
- com.phonegap
    - ContactAccessor (acceso a contactos), ContactAccessorSdk3_4, ContactAccessorSdk5, ContactManager (controlador de contactos): clases dedicadas a la gestión de los contactos, posiblemente, del teléfono móvil.
    - Device (dispositivo): puede ser una clase que se dedique gestionar y obtener información sobre el dispositivo móvil.
    - DirectoryManager (controlador de directorio):
    - FileTransfer (Transferencia de ficheros), FileUploadResult (resultado de la subida de ficheros), FileUtils(Utilidades para ficheros):
    - GeoBroker (corredor de posición), GeoListener (Escuchador de posición),GpsListene(escuchador de GPS): los escuchadores serán los

encargados de obtener la información de los diferentes sensores y el corredor tratará la información y le dará la utilidad a la aplicación.
- NetworkListener (escuchador de red),NetworkManager(controlador de red):
- SimpleCrypto (criptografía simple): probablemente, se trate de una clase de más alto nivel que los paquetes vistos anteriormente.
- Storage (Almacenamiento): será la clase encargada del almacenamiento. Puede ser en base de datos en el dispositivo o externa, también se puede tratar simplemente de la gestión de ficheros.

Tras estudiar brevemente la aplicación, se puede concluir que no es puramente nativa como las de Bankia y BBVA, si no que obtienen los datos de algún servidor web. Normalmente, aprovechan que ya tienen implementada una aplicación web y simplemente la adaptan a las necesidades de una aplicación móvil.

3.5 Diseño de pruebas para grupo de aplicaciones de comunicación

En este grupo de aplicaciones, el aspecto más critico, según la Ley orgánica de protección de datos, es la información relacionada con la sexualidad de los usuarios.

3.5.1 Facebook

La aplicación de Facebook está continuamente actualizándose y cubriendo vulnerabilidades. Para simplificar el análisis, se va a analizar los datos principales del usuario sin profundizar en el esquema de seguridad y privacidad, ya que normalmente es cada vez más privativo y más seguro. Los contenidos de la aplicación se han considerado de madurez media.

3.5.1.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Introducir usuario/email: davidpaquipalla@hotmail.com.
2. Introducir contraseña: 1*******0.
3. Pulsar sobre Aceptar.
4. Pulsar sobre estado.
5. Escribir “estado de prueba” y adjuntar posicionamiento y una fotografía.
6. Pulsar sobre opciones.
7. Seleccionar David Rubio (Usuario del perfil).
8. Pulsar sobre About.
9. Pulsar sobre opciones.
10. Buscar “Carlos Parreño”
11. Pulsar sobre About.
12. Pulsar sobre el botón menú.
13. Salir de la aplicación.

3.5.1.2 Diseño especifico de pruebas de código

La aplicación de Facebook tiene nombres descriptivos pero con una estructura muy compleja. En principio, parece tener dos núcleos básicos llamados katana y orca. Además, apache y contacts son especialmente interesantes para el estudio de la seguridad de la aplicación.

Katana parece ser el paquete de las funciones de la aplicación; orca, el paquete librería de funciones para complementar katana y apache, el paquete encargado de la conexión con el servidor.

El paquete apache es muy extendido en la comunidad Open Source y ofrece todo tipo de conexiones seguras. Básicamente, se encarga de gestionar las respuestas y las peticiones, formar los mensajes y evaluar la información de control.

Las clases que, a priori, resultan más interesantes para el estudio de la seguridad de la aplicación son:

- Katana:
    - DropdownFriendsAdapter (Adaptador de desplegable de amigos): esta clase se encargará de adaptar la información que ya se tiene sobre los amigos para que sea útil para un desplegable.
    - FacebookAccountReceiver (Recibidor de cuenta Facebook): la clase recibiría datos de la cuenta Facebook, puede tratarse desde configuración de la aplicación a datos de implementación con la API de Facebook.
    - FacebookApplication (Aplicación Facebook): parece tratarse de la aplicación principal y reunirá todo los datos y elementos de la aplicación.
    - FriendsActivity (Actividad amigos), FriendsAdapter (Adaptador de amigos): Estas clases se encargarán de tratar la información de amigos. Estas clases deberían tener acceso a información muy sensible de los amigos.
    - IntentUriHandler (Manejador de intents por URI): Esta clase gestionará la información que recibe o envía la aplicación vía URI.
    - LoginActivity (Actividad de login): Esta clase gestiona el acceso del usuario a la aplicación y todos sus elementos.
    - ProfileInfoActivity (Actividad información de perfil), ProfileInfoAdapter (Adaptador de información de perfil), ProfileSearchResultsActivity(Actividad resultados de búsqueda de perfil): estas clases gestionarán la información de un perfil y la que es posible buscar, deberían tener acceso a información muy sensible del usuario.
    - RequestsActivity (Actividad peticiones), RequestsAdapter(Adaptador de peticiones): estas clases se encargan de las peticiones que realiza o recibe la aplicación. Pueden ser datos importantes de los usuarios o información de configuración.
    - SyncContactsChangeActivity (Actividad sincronización de cambios de contactos), SyncContactsSetupActivity(Actividad sincronización de configuración de contactos): las clases se encargarán de la sincronización de los contactos, los cambios que hayan realizado o la configuración que tengan.
    - UserInfoActivity (Actividad información del usuario), UserInfoAdapter (Adaptador de información de usuario): estas clases se encargan de gestionar la información del propio usuario y adaptarla para las clases

que la emplean. Estas clases deberían tener acceso a información muy sensible del usuario.

- Feed:
  - NewsFeedAdapter (Adaptador del tablón de novedades), NewsFeedFragment (Fragmento de tablón de novedades): estas clases gestionan todos lo elementos que los amigos comparten con el usuario.
- Provider:
  - ConnectionsProvider (Proveedor de conexiones): esta clase puede gestionar el contenido de las conexiones de la base de datos o el servidor.
  - FacebookDatabaseHelper (Ayudante de base de datos de Facebook): esta clase es una abstracción de los diferentes métodos que requiere para trabajar con la base de datos.
  - LoggingProvider (Proveedor de registro): este proveedor ofrecerá contenido para registrar la actividad de la aplicación. Normalmente, el registro es completo, de todos los elementos y acciones.
  - PhotosProvider (Proveedor de fotos): el proveedor obtendrá las fotografías de los distintos elementos que lo soliciten.
  - UserStatusesProvider (Proveedor de estados de los usuarios): este proveedor ofrecerá el estado de los usuarios, es decir, información de los usuarios.
- Service:
  - BackgroundRequestService (Servicio de petición en background): este servicio se encarga de las solicitudes que llegan por detrás de la ejecución de la aplicación. Es posible que traten información importante de la aplicación.
  - FacebookService (Servicio Facebook): Servicio central de la aplicación que se encargará de responder y ofrecer a todos los elementos de la aplicación y del servidor.
  - MediaUploadService (Servicio de subida de multimedia): este servicio se encargará de la subida de fotografías y video. Por lo tanto, hay que observar el tratamiento de estos elementos y su transmisión.
- Util:
  - FBLocationManager (Controlador de localización de Facebook): esta clase se encargará de obtener la posición geográfica del usuario que esté empleando la aplicación de Facebook.
  - IntentUtils (Herramientas de intents): herramientas que gestionan las intents. Será útil estudiar como gestionan los datos con los que trabaja.
  - LocationUtils (Herramientas de localización): estas clases serán las encargadas de obtener los parámetros de posición desde los sensores.

        - Log (registro): esta clase guardará información sobre la ejecución de la aplicación para poder obtener información sobre los errores que se producen.
        - NetworkIdleMonitor (monitor de red sin utilizar): Esta clase comprobará el ancho de banda que no se esta empleando.
- Orca: En un estudio superficial del código, se ha podido determinar que son tipos de datos y funciones de eventos. En ningún momento son clases que puedan exponer a los activos.

3.5.2 Whatsapp

Whatsapp es una aplicación de mensajería, apenas contiene información personal salvo la que intercambie el usuario. Esta basado en localizar contactos a través de número de teléfono del dispositivo y, por lo tanto, es ahí dónde se encuentran todos activos. Los contenidos de la aplicación se han considerado de madurez media.

3.5.2.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. (Después de aceptar los Términos y condiciones) Seleccionar País: España.
2. Introducir código del país y numero de móvil: 63 y 6****41*1 (dígitos ocultados por privacidad).
3. Pulsar sí. (Aceptar restaurar historial de mensajes de la copia de seguridad).
4. Introducir nombre: David Rubio
5. Cambiar la foto por otra cualquiera del dispositivo.
6. Pulsar siguiente.
7. Escribir “esto es una prueba para mi proyecto” en una conversación con una persona. (En este caso el contacto es “Mir”).
8. Pulsar sobre el clip. Seleccionar posición geográfica.
9. Pulsar sobre el clip. Seleccionar imágenes y escoger una foto cualquiera.
10. Escribir “esto es una prueba para mi proyecto” en una conversación en grupo. (En este caso el grupo se llama “Prueba” y está “Mir”, “Corvo”, “Oscar” y “José”).
11. Pulsar sobre el clip. Seleccionar posición geográfica.
12. Pulsar sobre el clip. Seleccionar imágenes y escoger una foto cualquiera.
13. Esperar la respuesta y buscarla en los paquetes.

3.5.2.2 Diseño especifico de pruebas de código

En primer lugar, se puede observar que no todas las clases obtenidas del código de Whatsapp tienen un nombre claro y descriptivo, por lo tanto, es muy complicado localizar mediante las clases activos concretos. En cualquier caso, las clases con nombre descriptivo tampoco son especialmente claras.

En general las principales actividades de la aplicación parecen tener un nombre de clase claro. Las clases que tienen nombre descriptivo muestran un esquema donde cada caso de uso de la aplicación esta realizado por una actividad (en ocasiones se

apoya en clases auxiliares más especificas). Por lo tanto, los activos que se pretender estudiar puede ser fácilmente localizables si se emplean en cierto caso de uso.

El paquete que contiene las clases principales es com.whatsapp y se pueden observar clases como:

- contactPicker (selector de contactos), conversation (conversación), conversationTextEntry (Texto de entrada en conversaciones), conversations (conversaciones): probablemente contenga información de los usuarios además de las propias conversaciones.
- deleteAccount (eliminar cuenta): probablemente tenga acceso directo a datos del usuario en la base de datos. La falta de protección de esta clase puede provocar que la aplicación deje de estar disponible para este usuario.
- groupChatMap (mapa de conversaciones en grupo): esta clase puede contener información sobre los contactos del grupo.
- locationPicker (selector de localización): esta clase permite seleccionar la posición geográfica para poder enviarla a otros contactos. Hay que estudiar si la aplicación obtiene la posición sin petición explicita del usuario.
- mediaGallery (Galería multimedia), MediaGalleryImageView (Galería multimedia, visor de imágenes), TouchImageView (vista imagen táctil): esta clase tiene acceso a contenido multimedia y hay que estudiar si las imágenes son almacenadas, donde, cómo se transmiten y cómo restringe su visualización a los usuarios autorizados.
- registerName (registro de nombre), registerPhone (registro de teléfono): Registro de nombre de usuario y registro del teléfono ofrecen la información única de registro de usuario. Se investigará el tratamiento de estos datos y como se emplean para generar una contraseña.
- VerifyMessageStoreActivity (verificar almacenamiento de mensaje), VerifyMessageStoreListActivity (verificar almacenamiento de lista), VerifyNumber (verificar número), VerifySms (verificar por SMS): son actividades encargadas de verificar mensajes, número de teléfono y envió de SMS, por lo tanto, tienen acceso a información sensible y útil para autenticar al usuario. El objetivo es ver como verifica el mensaje y la lista, que datos accede y si los mantiene en algún medio. El sistema de verificación por SMS debe tener protegido los datos que emplea de verificación para que sea el usuario real el que finalmente se autentique.

3.5.3 Twitter

Twitter es prácticamente una red pública, la información que solicita es para dominio público. Se basa en mensajes publicados que pueden ver todos, es posible añadir fotografía y posicionamiento geográfico. La red permite elegir entre transmisión HTTP y HTTPs y también permite que solamente las personas que el usuario elige pueden leer sus *tweet* con imágenes e información geográfica. Los contenidos de la aplicación se han considerado de madurez media.

3.5.3.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

Antes de nada, desde un navegador web, asegurarse que esta activada la conexión por HTTPs y que la red es privada. Si no, se entiende que el usuario permite que cualquiera pueda tener acceso a sus datos y no seria necesaria las medidas de seguridad para los activos.

1. Pulsar sobre Sign In.
2. Introducir nombre de usuario: pa****ki (empleo la cuenta personal)
3. Introducir contraseña: 1*******0.
4. Pulsar Ok (para aceptar que pueda acceder a la posición geográfica actual).
5. Escribir un nuevo mensaje público (tweet): “Tweet de prueba”.
6. Escribir un nuevo mensaje directo (DM): “Mensaje directo prueba”
7. Cerrar sesión con este usuario.

3.5.3.2 Diseño especifico de pruebas de código

La mayor parte de los nombres de las clases de Twitter no tienen un nombre descriptivo, pero las que si lo tienen parecen conformar el núcleo de la aplicación y permiten entender la jerarquía de las clases.

Las clases se dividen, principalmente, en tres grupos: actividades, red y proveedores:

- Actividades:
    - AccountsActivity (Actividad de cuentas), AccountFragment (Fragmento de cuentas), AccountsDialogActivity(Actividad de diálogo de cuentas), AccountSettingsActivity (Actividad de ajuste de cuenta): estas actividades se encargan de gestionar y utilizar la información sobre la cuenta. Pueden tener acceso a usuario, contraseña, mensajes privados,…
    - AuthenticatorActivity (Actividad autenticadora), AuthorizeAppActivity (Actividad de autorización de aplicación), AuthorizeAppFragment (Fragmento de autorización de aplicación): estas clases se encargan de autenticar al usuario frente a aplicaciones que requieren conexión con Twitter (clientes o API).
    - DraftsActivity (Actividad borradores), DraftsFragment(fragmento borradores): estas clases gestionan los tweet que se han redactado y no se han llegado a enviar y son almacenados.
    - EditProfileActivity (Actividad de edición de perfil): esta actividad podrá acceder a los datos del perfil de usuario y modificarlos.
    - FollowActivity (Actividad de seguir a otro usuario): esta actividad realizará la función de seguir a otro usuario, es decir, que sus mensajes lleguen a la línea de tiempo del usuario.
    - HomeActivity (Actividad principal): actividad inicial, puede tener acceso a algún dato importante. En principio, sólo debería llamar a las clases delegadas de las funciones que realiza la aplicación.
    - ImageActivity (Actividad Imagen): esta actividad será la encarga de controlar los aspectos relacionados con las imágenes que se adjuntan con los mensajes (las imágenes se adjuntan mediante una dirección web).

- LoginActivity (Actividad login): esta clase es la encargada de controlar el acceso a la aplicación por parte de los usuarios. Tendrá acceso a usuario y contraseña.
- MediaPlayerActivity (Actividad reproductor multimedia): esta actividad se encarga de reproducir los elementos compartidos por usuarios que sean videos. También se comparte a través de un enlace.
- MessagesActivity (Actividad mensajes), MessagesFragment (Fragmento mensajes), MessagesThreadActivity (Actividad hilo de mensajes): estas clases gestionan los mensajes directos que se envían los usuarios. Son privados entre ambos y nadie debería tener acceso a información de los mismos.
- PostActivity (Actividad de envío): esta actividad enviará datos a través de las redes.
- ProfileActivity (Actividad perfil), ProfileFragment (Fragmento perfil): esta actividad mostrará toda la información sobre el perfil del usuario. Algunos de estos datos pueden estar marcados como activos.
- SignUpActivity (Actividad de registro): esta aplicación da de alta a un nuevo usuario en la base de datos. Solicitará nombre de usuario, email y contraseña, elementos imprescindibles para la autenticación del usuario frente al servidor.
- TimelineActivity (Actividad línea temporal), TimelineFragment(Fragmento de linea temporal): estas clases gestionaran la linea temporal o TL (Time Line). El TL es el lugar donde se publican todos los *tweets* de los usuarios que se siguen.
- TweetActivity (Actividad tweet), TweetFragment (Fragmento tweet), TweetListFragment (Fragmento de lista de tweet): estas clases se encargan de los tweets (los propios mensajes publicados a todos los seguidores). Esta información es pública salvo que el usuario marque lo contrario en sus ajustes.
- TweetSettingsActivity (Actividad ajustes de tweet): esta clase, entre otras cosas, marcará los tweets como privados o públicos en funciones de los ajustes de la cuenta.
- UserQueryActivity (Actividad de petición de usuarios): esta clase gestiona las peticiones del usuario, probablemente, a la base de datos.
- UsersActivity (Actividad usuarios), UsersFragment (Fragmento de usuarios): estas clases manejan información propia de los usuarios.

- Proveedores:
  - ActivityDataUser (datos de usuario de la actividad): esta actividad se encarga de los datos de usuario como el nombre, la imagen o el nombre de usuario.
  - GlobalDatabaseProvider (Proveedor global de la base de datos): es el proveedor de datos de la base de datos, es decir inicializa la conexión con las bases de datos de los servidores de Twitter y obtiene los datos necesarios. Es interesante analizar como realiza las peticiones (si son parametrizadas) y si la conexión es segura.
  - TwitterProvider (Proveedor de Twitter): este proveedor nutrirá de información sobre la propia aplicación Twitter desde la base de datos.

Por ejemplo, lo temas de tendencia actual (llamados Trending topics o TT) o listas de susarios.

- Network:
    - Ssl: los nombres de este paquete están codificados en a,b,c,d,e y TwitterPins$1. Rápidamente, en las clases se puede observar que se importan paquetes relaciones que seguridad y cifrado/firma digital.

3.5.4 Gmail

Es el cliente de correo electrónico de Google, es el usuario que permite acceso a multitud de aplicaciones. Se estudiará sólo el impacto sobre los mensajes intercambiados por correo y es donde se concentra la información privada. Las cuenta de Google también cuentan con un perfil y la información es muy similar a la que ofrece el perfil de Facebook. Los contenidos de la aplicación se han considerado de madurez baja.

3.5.4.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Pulsar sobre sincronizar ahora.
2. Abrir el primer correo recibido.
3. Marcar como no leído.
4. Pulsar sobre archivar correo.
5. Escribir en destinatario: 100080668@alumnos.uc3m.es .
6. Escribir en asunto: Correo de prueba.
7. Escribir en cuerpo de mensaje: Texto de prueba.
8. Pulsar botón menú > adjuntar archivo y seleccionar una imagen.
9. Enviar desde navegador un correo respuesta con texto: “Repuesta de correo”

3.5.4.2 Diseño especifico de pruebas de código

La aplicación de Gmail sigue un esquema completamente claro y los nombres de las clases son descriptivos. Toda la información útil de las clases está en el paquete com.google.android.gm. A parte de las clases, contiene paquete que pueden resultar importantes como: persistence, provider y contentprovider.

El paquete persistence se encarga de mantener los datos de la base de datos y la aplicación sincronizados. El paquete provider proveerá a la aplicación de algún servicio. El paquete contentprovider se encargará de proveer contenido a la aplicación.

- Gm
    - AccountsChangedReceiver (Recibidor de cambio de cuentas): para ser que se encarga de cambiar de cuentas del sistema multicuentas con el que cuenta Gmail. Probablemente, tenga acceso a usuario y contraseña de las diferentes cuentas.
    - AutoSendActivity (Actividad de auto envío): Esta actividad envía automáticamente los mensajes cuando están es posible realziarlo. Tendrá acceso a información del servidor, emisor y receptor, además del propio mensaje.

- ConversationListContext (Contexto lista de conversaciones), ConversationListFragment (Fragmento de lista de conversación), ConversationView (Vista conversación), ConversationViewState (estado de la vista de conversación), ConversationSubjectDisplayer (mostrador de asunto de la conversación), ConversationTransientState (estado transitorio), ConversationView (vista de la conversación): estas clases gestionan diferentes aspectos de las conversaciones. Las conversaciones son sucesiones de email intercambiados entre dos o más usuarios. Por lo tanto, estas clases tiene acceso a los emails, la información de los usuarios y a los archivos adjuntos.
- ConversationPositionTracker (seguimiento de posición de conversación): Esta clase geolocaliza la conversación del usuario.
- EmailAddress (dirección de email): Dirección de email, es el nombre de usuario de la cuenta de Gmail. El tratamiento que de esta clase a los datos será importante para la seguridad del acceso a la aplicación.
- FilterPopup (Filtro de elementos emergentes): Esta clase se encarga de filtrar los datos de los elementos emergentes para que sean del formato correcto y, eventualmente, no realice una función dañina.
- GmailActivity (Actividad de Gmail), GmailReceiver (Recibidor Gmail): parecen ser actividades centrales de la aplicación, probablemente por aquí pasen la mayoría de los datos que gestiona.
- MailIntentReceiver (Intent de recibidor de correo), MailIntentService (Intent de servicio de correo): son las intents de recibir o enviar los correos. Previsiblemente, aquí reforzará la seguridad de las conversaciones para que no sean accesibles desde el exterior.
- MessageHeaderAttachment (Encabezado de mensaje adjunto), MessageHeaderView (Vista de cabecera del mensaje): maneja las cabeceras y los adjuntos en ellas, tiene acceso a mucha información del mensaje y del usuario.
- NonRestorableTextView (Vista de mensaje no almacenable): trata los mensajes que no se almacenan y habría que estudiar como son tratados.
- QuotedTextView (Vista de mensaje citado): es la clase encargada de mostrar los correos citados de otros usuarios. Puede mostrar información públicamente que no deba.
- RecipientAdapter (Adaptador de destino): Se encarga de transformar la información del destino que se tiene en información coherente para el servidor.
- SenderInfoLoader (Cargador de información del transmisor): Esta clase subirá información sobre el usuario que envía el correo. En principio, no debería ser información sensible.

- Persistence:
  - GmailBackupAgent (agente de backup de Gmail),GmailBackupData(Datos de backup de Gmail): estas clases encargacaran de guardar ciertos datos de la aplicación en la base de datos.

    - Persistence (Persistencia): las clases de persistencia se encargan de mantener los datos sincronizados con la base de datos.
- Provider:
    - ConversationUtil (útiles de conversación): es una clase que reúne funciones que sirven de herramientas para las conversaciones. Puede acceder a muchos elementos de las conversaciones.
    - DatabaseInitializer (inicialización de la base de datos), InternalSettingsDbInitializer (inicializador de ajustes internos de la base de datos): Estas clases inicializan los datos con la base de datos, pueden ofrecer información sobre la conexión.
    - GmailQuery (Petición Gmail): solicitud de datos a Gmail, probablemente sea a una base de datos.
    - LogUtils (útiles de historial): esta clase reúne funciones sobre el Log de la aplicación.
    - MailEngine (Motor de correo): esta clase se encarga de gestionar los protocolos de correo.
    - MailStore (Almacenamiento de correo), MailStoreInitializer (Inicializador de almacenamiento de correo): almacenamiento de email, gestionará la conexión, directa o indirectamente, con la base de datos.
    - MailSync (Sincronizador de correo), MailSyncAdapterService (Servicio adaptador de sincronización de correo), MailSyncObserver (Observador de sincronización de correo): son clases de sincronización con el servidor, tendrá acceso a emails, backups y usuarios.
    - PublicContentProvider (proveedor de contenido público): esta clase provee de datos públicos a la aplicación. El estudio se centrará en que datos públicos se consideran.
- Contentprovider:
    - GmailAccess (Acceso a Gmail), PrivateGmailAccess (Acceso privado a Gmail): estas dos clases proveen el servicio de autenticaciones del usuario frente al servidor para acceder a los datos de la aplicación.

3.6 Diseño de pruebas para grupo de aplicaciones de login

En este grupo de aplicaciones, el aspecto más critico serán los datos que los usuarios han considerado muy relevantes y que ellos pueden consideran privados. En principio, no habrá información muy sensible por si misma, lo único los contenidos de pago que si están en peligro, la amenaza es que el usuario no pueda disfrutar de ellos. En cualquier caso, los documentos pueden contener información sensible y hay que proteger la decisión de privacidad del usuario.

3.6.1 Dropbox

La aplicación Dropbox es una aplicación de almacenamiento en la nube y permite la compartición de ficheros y fotografías fácilmente con otros usuarios. La aplicación de Android cuenta con carga automática de en la nube. Los puntos críticos serán los ficheros privados o no compartidos con otros usuarios.

3.6.1.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Introducir usuario: davidrubiomatellanes@gmail.com
2. Introducir contraseña: d***m*** (ocultada por seguridad).
3. Pulsar sobre Next (Informa sobre capacidad extra).
4. Pulsar sobre Skip (opciones de subida automática de fotografía).
5. En el menú de opciones, seleccionar Nueva carpeta y llamarla prueba.
6. En el menú de opciones, seleccionar nuevo archivo de texto y escribir "texto prueba".
7. En el menú de opciones, seleccionar guardar el archivo y llamarlo myfile.txt.
8. Editar el texto desde un ordenador. Añadir "Línea añadida". (El cliente en Windows sincroniza automáticamente).
9. Abrir en el cliente Android el archivo mi texto con el editor de texto de Dropbox.
10. Pulsar sobre el archivo y marcarlo como favorito.
11. Pulsar subir para ir al directorio principal.
12. Compartir la carpeta con dav1druma@gmail.com y refrescar la aplicación.
13. Pulsar sobre el botón de cargar.
14. Seleccionar una fotografía cualquiera.
15. Compartir en la carpeta anterior.

3.6.1.2 Diseño especifico de pruebas de código

Tras un vistazo rápido de los nombres de las clases, se puede ver que la mayoría de las clases no tienen un nombre descriptivo pero las que lo tienen parecen ser la que realizan las actividades principales de la aplicación. Estas clases serán las que se analizaran en busca del tratamiento que hacen de los distintos activos.

Las clases se dividen, principalmente, en tres grupos: actividades, proveedores y servicios:

- Actividades:
  - AccountsAndSyncSetupActivity (configuración y sincronización de cuentas): parece ser que se encargará de validar el usuario y sincronizar datos de sesión. Será importante estudiar como trata al nombre de usuario y la contraseña.
  - Browser (navegador), BrowserWithHistoryStack (navegador con historial), DropboxBrowser (navegador de Dropbox), SimpleDropboxBrowser (navegador simple), LocalFileBrowseFragment (Navegador de ficheros local), LocalFileBrowserActivity (Navegador de ficheros local): probablemente sea el navegador de archivos y tenga acceso a todos los ficheros y contenido multimedia que están en la nube o en el dispositivo.
  - CameraUploadDetailsActivity (detalles de subida de elementos de la cámara), CameraUploadDetailsFragment (fragmentación de detalles anteriores), CameraUploadGridActivity (rejilla de elementos de la cámara), CameraUploadGridFragment (fragmentación de la rejilla anterior): son clases que se encargaran de subir automáticamente fotografías o videos al espacio en la nube. Solamente es posible si previamente se ha dado permiso explicito por parte del usuario.

    - DropboxSendTo (Enviar a): la clase se encargaría de enviar un elemento de la nube a otro usuario o le puede proporcionar un acceso directo en forma de URL.
    - ForgotPasswordFragment (Fragmento de contraseña olvidada): puede ser la clase encargada de solicitar la nueva contraseña, por lo tanto, debe proteger la introducción la nueva contraseña y la posterior transmisión al servidor.
    - GalleryActivity (galería), GalleryPickerActivity (selector de la galería), GalleryPickerFragment(Fragmento de selector de la galería): Las clases encargadas de mostrar la imágenes en una galería
    - LoginBrandFragment (marcar acceso), LoginFragment (fragmento de acceso), LoginOrNewAcctActivity (acceso o nueva cuenta): dará acceso a la aplicación mediante un usuario y una contraseña. También se encarga de nuevos usuarios.
    - TextEditActivity (Edición de texto): puede ser la clase encargada de la edición de los documentos de texto. Tendría acceso a todos los documentos y a la modificación de los mismos.
    - UploadsActivity (Subidas, a la nube), UploadsActivityGroup (subidas en grupo, a la nube): podría tratarse de las actividades encargadas de mandar los ficheros al espacio en la nube. Será muy importante estudiar elementos de transmisión de datos y/o base de datos.
    - VideoPlayerActivity (reproductor de video): será la clase encargada de reproducir videos. Tendrá acceso a los videos que estén almacenados en la nube y en el dispositivo.
- Proveedores:
    - CameraUploadsProvider (proveedor subidas de cámara): será el proveedor de contenido que se encargue de gestionar las conexión con los servidores o bases de datos que alojen el contenido y poder nutrir a las aplicación. Por lo tanto, tiene acceso a los contenidos multimedia del teléfono.
    - CurrentUploadsProvider (proveedor subidas actuales): probablemente proveerán algún dato de conexión segura y estable a una subida actual de un archivo. Por lo tanto, tiene acceso a todos los contenidos que se han subido a la nube.
    - FileSystemProvider (proveedor sistema de ficheros): será el proveedor de los contenidos del sistema de fichero como por ejemplo, ruta, tamaño, nombre de fichero,.. Es decir, que tiene acceso a todos los archivos que se están compartiendo.
    - SDKProvider (proveedor de SDK): Dropbox cuenta con una SDK para desarrolladores y probablemente se trate de la clase que gestiona las conexiones con otras aplicaciones. Por lo tanto, debería tener acceso a todos los elementos de Dropbox.
    - UploadLogProvider (proveedor de log de subidas): este proveedor se encargará de registrar el historial de subidas de archivos a la nube. Puede contener información relevante sobre los ficheros.

    - ZipperedMediaProvider (proveedor de elementos multimedia), ZipperedUploadProvider (proveedor de subida): son clases que tendrán acceso a todas las subidas y a todos los elementos multimedia.
- Servicios
    - AuthenticationService (servicio de autenticación): se encargará de autenticar al usuario frente al servidor para poder darle acceso a los recursos en la nube.
    - CameraUploadService (servicio de subida de capturas de cámara): será el servicio encargado de la subida automática de los contenido generados por la cámara, sólo en caso de que el usuario lo haya permitido explícitamente.
    - DropboxNetworkReceiver (recibidor de red de Dropbox): probablemente se trate del servicio encargado de recibir información desde Dropbox de cualquier tipo. Interesa comprobar si hay algún tipo de conexión con los activos que se están estudiando o no.

3.6.2 Evernote

La aplicación Evernote permite la creación y la sincronización de notas, audio, posición geográfica, fotografía o ficheros adjuntos. Todos estos elementos son privados y no pueden pertenecer a varios usuarios y por lo tanto, salvo que el usuario lo comparta por otro medio, son los puntos críticos que deben estar protegidos.

3.6.2.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Pulsar sobre Sign in.
2. Introducir en Username: davidpaquipalla
3. Introducir password: d***m***.
4. Pulsar crear nueva nota, en la esquina inferior izquierda.
5. Titular como “Nota prueba”.
6. Escribir como contenido “Contenido de prueba”.
7. Añadir posición geográfica y una fotografía.
8. Modificar en la aplicación web añadiendo, línea extra.
9. Sincronizar de nuevo la aplicación.

3.6.2.2 Diseño especifico de pruebas de código

Los nombres de los paquetes y clases de esta aplicación no son descriptivos, pero cuanta con clases suficientes con el nombre descriptivo como para poder realizar un análisis.

Destacan las actividades del paquete principal de Evernote, el paquete cliente, el paquete note, el paquete provider y el paquete util. El paquete principal tiene actividades principales de la ejecución de la aplicación. El paquete cliente se encarga de gestionar las conexiones con el servidor. El paquete note controla todo lo relacionado con la notas (elemento principal de la aplicación). El paquete provider proveerá de contenido la aplicación, probablemente de una base de datos. El paquete util contendrá clases con herramientas para la ayuda de la implementación de funciones de la aplicación.

- com.evernote:
    - Evernote: esta clase es la principal de la aplicación, puede tener acceso a datos o elementos importantes. En principio, debería estar delegado a otras clases y actividades.
    - Client:
        - EvernoteService (Servicio Evernote): esta clase dará servicio sobre los principales elementos de la aplicación como datos y conexiones. Habrá que estudiar con que datos opera.
        - SyncService (Servicio sincronización): este servicio se encarga de sincronizar el contenido con el servidor de Evernote. Recordemos que también tienen aplicación web y que se puede acceder desde un ordenador a sus servicios.
    - note.componser:
        - FilePickerActivity (Actividad selector de archivo): esta clase será la encargada de mostrar los directorios de archivos que se pueden enviar y permitirá seleccionarlos.
        - NewNoteActivity (Actividad nueva nota): esta actividad se encargará de crear nueva notas que se sincronizaran con el servidor. Suelen ir acompañadas de imagines, video, audio…
        - NewNoteAloneActivity (Actividad solo nueva nota): probablemente sea similar a la anterior pero sin elemento adjuntos.
        - QuickNewNoteActivity (Actividad nueva nota rápida): esta actividad creará nueva notas directamente.
    - provider:
        - EvernoteProvider (proveedor Evernote): esta clase nutrirá a la aplicación de elementos propios de la aplicación. Habrá que estudiar que tipos de datos trabaja.

3.6.3 Google Drive

Es el cliente de Google que ofrece un servicio idéntico que Dropbox. La única diferencia es que los tipos de documentos por defecto son los de Google Docs y se abren en un navegador, o bien, en la aplicación nativa.

3.6.3.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

16. Pulsar en My Drive
17. En el menú de opciones, seleccionar nuevo documento y llamarlo Doc prueba.
18. Escribir “Texto para prueba”.
19. Editar el texto desde un ordenador. Añadir “Línea añadida”. (El cliente en Windows sincroniza automáticamente).
20. Abrir en el cliente Android el archivo mi texto con el editor de texto de Drive.
21. Compartir el archivo con dav1druma@gmail.com.
22. Pulsar subir para ir al directorio My Drive.

23. En el menú de opciones, seleccionar nueva subida.
24. Seleccionar una fotografía cualquiera.
25. Compartir en la fotografía anterior.

3.6.3.2 Diseño especifico de pruebas de código

Esta aplicación es una adaptación de la antigua Google docs por eso el paquete principal de la aplicación es docs. Los nombres de las clases y paquetes son bastante descriptivos y permiten entender la estructura de la aplicación:

Esta aplicación esta muy modulada en paquetes:

- App:
  - AccountListeningActivity (Actividad escucha de cuentas), AccountsActivity (actividad cuentas): estas clases se encargan de gestionar las cuentas y de recibir los datos de la misma.
  - CreateNewDocActivity (Actividad crear nuevo documento): esta activad crear un nuevo documento, puede contener el titulo o información sobre los usuarios con los que esta compartido.
  - DocumentOpenerActivity (Actividad abridor documento): es la clase que se encarga de abrir los documentos. Es posible que sincronice el archivo directamente del servidor y requiera autenticar el usuario para saber si es compartido con él.
  - LocalFileOpenerActivity (Actividad abridor de ficheros locales): esta clase se encargará de abrir los documentos locales. Por lo tanto, tendrá acceso a todos los ficheros locales y podrá modificarlos y/o eliminarlos.
- Docsuploader:
  - UploadQueueActivity (Actividad cola de subida): esta actividad se encarga de gestionar la cola de subida a la nube de los archivos creados o modificados.
  - UploadQueueService (Servicio de cola de subida): este servicio se encarga de gestionar la conexión de con el servidor en background.
- Editors.text:
  - EditText (Editar texto), TextView (Visita texto): son clases que tienen acceso al texto y, por lo tanto, deben contar con alguna medida de protección para que sean lo usuarios autorizados los que puedan realizar estas acciones.
- Providers:
  - DocListProvider (Proveedor de lista de documentos): esta clase obtiene la lista de los documentos del usuario que se encuentran en el servidor.
- Recievers:
  - AccountChangeReceiver (recibidor cambio de cuenta): si hay un cambio en la cuenta actual del usuario, esta clase se encargará de recibir dicho cambio.
  - NetworkChangeReceiver (recibidor cambio de red): Lo mismo sucede con la red para esta clase.
- Shareitem:

        - UploadSharedItemActivity (Actividad subir elemento compartido): esta clase se encarga de subir lo elementos recién compartidos, para que los usuarios con los que se haya compartido pueden acceder al elemento.
- Sharingactivity:
    - ModifySharingActivity (Actividad modificar compartir): esta clase modifica con que usuarios se comparte un elemento.
- Sync:
    - Filemanager:
        - FileProvider (Proveedor de ficheros): esta clase sincroniza los ficheros con los que el usuario tiene en la nube.
    - Syncadapter:
        - ContentSyncReceiver (recibidor de sincronización de contenido): esta clase se encarga de sincronizar el contenido de los documentos.
        - ContentSyncService (Servicio de sincronización de contenido): realiza la sincronización del contenido en background.
        - DocsSyncAdapterService (Servicio adaptar sincronización documentos): esta clase adapta los datos de un documento para su sincronización. Tiene acceso a los documentos completos.
- DocsApplication (Aplicación documentos): es la aplicación principal. En principio, no debería tener acceso directo a los elementos de la aplicación.

3.6.4 Spotify

Spotify es una aplicación de música en la nube. Desde un equipo sobremesa, existen opciones gratuitas sin embargo, desde un dispositivo móvil, todo el contenido es de pago. La aplicación debe asegurar a los usuarios que han pagado por el servicio.

3.6.4.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Seleccionar acceso con Facebook.
2. Pulsar sobre la primera lista de música.
3. Descargar lista de reproducción.
4. Pulsar sobre + (Crear nueva lista)
5. Escribir nueva lista prueba y crearla.
6. Añadir nueva canción (en este caso, “Encantado de conocerte”).
7. En Ajustes > Cerrar sesión.

3.6.4.2 Diseño especifico de pruebas de código

Los nombres de las clases están codificados en su mayoría pero entre las que no lo están y los nombres de los paquetes se pueden visualizar la arquitectura de la aplicación.

De la aplicación destacan cuatro paquetes: core, provider, service. Core es el núcleo de la aplicación, llevará las funciones básicas de conexión y funciones con el sistema. Provider es el paquete encargado de nutrir a la aplicación con el contenido propio de Spotify. Service se encarga de operar con el servidor para ofrecer los servicios de la aplicación.

- Core:
    - ConnectivityListener (Escuchador de conectividad): este escuchador estará pendiente de la conexión de la aplicación ya que de lelos depende la calidad y el servicio de reproducción de música.
    - HttpConnection (Conexión HTTP): se encarga de la conexión HTTP con el servidor, será interesante ver si trata con mecanismos de seguridad.
    - LocalFilePlayer (Reproductor de archivo local): esta actividad reproducirá los archivos locales del dispositivo, por lo tanto, tiene capacidad para acceder a todos los archivos.
- Provider:
    - SpotifyProvider (Proveedor Spotify): proveerá de contenidos, probablemente música y listas, la aplicación.
- Service:
    - LoginActivity (Actividad login): permite acceder a un usuario registrado. Es necesario un usuario y contraseña.
    - SpotifyService (Servicio Spotify): provee de los servicios más basicos a la aplicación.

3.7 Diseño de pruebas para grupo de aplicaciones de consulta

Este tipo de aplicaciones apenas deben obtener información del usuario, como un patrón de utilización de la aplicación y la posición geográfica para mejorar los detalles de la consulta. Por lo tanto, el análisis de las clases no va a ser muy profundo.

3.7.1 RTVE

Ultimas noticias de RTVE, muestra noticias y videos de las noticias separadas por canales o temáticas. Ni siquiera solicita ubicación para afinar las noticias.

3.7.1.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Pulsar sobre internacional.
2. Pulsar sobre cultura.
3. Pulsar sobre España.
4. Pulsar sobre economía.
5. Pulsar sobre Ciencia y Tecnología.
6. Abrir primera noticia con video.
7. Abirir primera noticia sin video.
8. Pulsar telediario.
9. Pulsar sobre A la Carta.
10. Pulsar sobre Info

3.7.1.2 Diseño especifico de pruebas de código

Tras un estudio preliminar de la estructura de la aplicación, destacan tres paquetes: activity (actividades), cifrado y service (servicios). Los nombres de todas las clases son descriptivitos y se puede tener una idea completa de como está hecha la aplicación.

Las actividades se encargan de obtener la información desde una página web, por lo tanto, únicamente hay que estudiar como se tratan los datos de entrada y salida. En

cualquier caso, el único activo marcado ha sido la posición geográfica y se obtendrá nativamente, en caso de obtenerla.

El paquete de cifrado se encarga principalmente de filtrar y parametrizar los datos de entrada del usuario.

Los servicios son para la función de radio y estadísticas de los oyentes. Es probable que traten de realizar un estudio estadístico de posicionamiento de sus oyentes, por lo tanto, sería importante buscar este tipo de información.

Los nombres de las clases resultan muy descriptivos y se han explicado sus funciones en líneas generales, consecuentemente, sólo se clasificaran las clases del estudio:

- Actividades:
  - ActualidadActivity
  - DirectosActivity
  - InfoActivity
  - NoticiaActivity
  - NoticiaGenericoActivity
  - PrincipalActivity
  - SplashActivity
  - VideoActivity
  - VideoDirectoActivity
  - WebViewActivity
- Cifrado:
  - BlowFishEncrypt (cifrado Blow Fish)
  - ConsumerFilterTester (Pruebas de filtros del consumidor)
- Servicios:
  - EstadisticasService
  - RadioEnDirectoService
  - StatsManager (Controlador de estadísticas)

3.7.2 El País

Aplicación muy similar funcionalmente a RTVE, muestra noticias e imágenes sobre noticias organizada por secciones.

3.7.2.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Pulsar sobre la primera noticia.
2. Pulsar atrás: <.
3. Pulsar sobre secciones.
4. Seleccionar Internacional.
5. Arrastrar a la derecha hasta acabar las secciones.
6. Pulsar sobre última hora.
7. Pulsar sobre la estrella (favoritos).

3.7.2.2 Diseño especifico de pruebas de código

Los nombres de los paquetes y las clases son claros y descriptivos. En principio, según las clases que contiene esta aplicación no hay indicios para afirmar que la aplicación trace la posición geográfica.

Sin embargo, en el paquete statdistics encontramos la clase OmnitureSender.java que traza todos los movimientos en la aplicación para estadísticas. Esos datos son legítimos para los propietarios de la aplicación, pero debería estar protegido del exterior, ya que puede dar una idea de ideología política o sindical (aspectos especialmente protegidos).

Entre las actividades principales destacan dos:

- WiFiScanReceiver(recibidor de escaneo de WiFi): parece tratase de una clase encargada de recibir información únicamente por WiFi y, por lo tanto, escanea en busca de una señal WiFi.
- LoadingActivity (Actividad carga): esta clase se encarga de recibir y cargar todos los elementos de la aplicación. Puede tener acceso a todo el contenido, filtrar o priorizar por posición geográfica y conseguir que

3.7.3 Tiempo AEMET

La aplicación es una aplicación muy completa, permite conocer el parte meteorológico oficial de localidades, playas, mares, por estaciones meteorológicas, montañas, por mapa, de radiaciones, rayos, masas de aire e infrarrojo. En cualquier caso, se puede filtrar directamente por la posición geográfica, no muestra la posición exacta en la aplicación, sólo la localidad. La aplicación puede obtener uno de ellos del GPS o de la localización basada en red.

3.7.3.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Pulsar sobre predicción.
2. Pulsar sobre localidades.
3. Pulsar sobre el icono de la esquina superior derecha (icono GPS).
4. Marcar como favorito.
5. Refrescar la actividad.
6. Volver al menú localidades.
7. Selecciona Almería.
8. Selección Abla.

El resto de menús son idénticos a los pasos 6, 7 y 8.

3.7.3.2 Diseño especifico de pruebas de código

Los nombres de paquetes y clases parecen descriptivos pero llevan a confusión. Por lo tanto, ha habido que estudiar los diferentes paquetes y el contenido de sus clases.

En el paquete com.projectsexception.weather.geocoder están las clases encargadas de gestionar la posición geográfica del dispositivo. Por lo tanto, este es el paquete principal de estudio de la aplicación.

En android.support.v4.app, podemos ver que hay una clase llamada _ActionBarSherlockTrojanHorse. Es posible que se trate de un troyano pero habrá que analizar el código de la clase y como se emplean sus métodos en otras clases.

3.7.4 Google Maps

Es el servicio de mapas de Google que permite buscar posiciones concretas, calles, localidades, etc. Además ofrece rutas entre distintos puntos del mapa pudiendo obtener uno de ellos del GPS o de la localización basada en red.

3.7.4.1 Diseño especifico de pruebas de comunicación

Para asegurar que la aplicación expone los diferentes activos ante las amenazas, se va a seguir una serie de pasos:

1. Activar WiFi.
2. Pulsar sobre el botón de la parte superior derecha.
3. Activar GPS
4. Repetir el paso 2.

3.7.4.2 Diseño especifico de pruebas de código

Esta aplicación tiene codificado el nombre de las clases y algunos paquetes. Realmente sólo interesa conocer las funciones que emplean geolocalización y como gestionan esos datos.

El paquete com.google.android.location.localizer se encarga de gestionar la API y obtener la posición geográfica del dispositivo. El estudio se centrará en como gestiona los datos de latitud y altitud obtenidos y como son almacenados, también se estudiará el uso de este paquete.

4 Resultado de las pruebas

En esta sección se realizará el análisis de los datos obtenidos de los procedimientos descritos en el apartado anterior (sección 3). EL objetivo es llegar a resumir en una tabla si los activos de las distintas aplicaciones están protegidos o no frente a amenazas.

4.1 Resultado de las pruebas de grupo de aplicaciones bancarias

4.1.1 Banco Santander

4.1.1.1 Resultado de las pruebas de comunicación en Banco Santander

En primer lugar, se filtran los paquetes HTTP puesto que son los más expuestos a contener información en claro. Durante la captura, sólo se ha capturado un paquete HTTP relacionado con Banco Santander.

Dicho paquete es una petición GET a través de una dirección URI. Los parámetros transmitidos son el sistema operativo, la versión, idioma, modelo del móvil (número de construcción del firmware), creador del firmware, navegador sobre el que se ejecuta y su versión. También trata sobre que tipo de codificación, lenguaje y conjunto de caracteres son aceptados. Como se puede ver, en ningún momento se envían datos de la aplicación a través de los parámetros de la petición por URI

Los siguientes paquetes son de dos tipos TLSv1 y TCP. Los TCP son de control de flujo, básicamente envía ACK, SYN o FIN. TLSv1 son los mensajes que se intercambian el cliente y el servidor cifrados. Un ejemplo de datos cifrados que se envía:

Encrypted Application Data: e8159b659f50b8795320d39470fe8709cf38e51f98bf6794...

En la sección de paquete relacionados con "Oficinas y cajeros", son capturados paquetes HTTP en claro. En estos paquetes se puede observar que aparecen una dirección URI al host maps.google.es con dos tipos de parámetros: las coordenadas geográficas, en el caso de usar mi ubicación o el propio código postal.

Además, para confirmar que no se ha pasado nada por alto, se realiza una búsqueda entre todos los paquetes capturados intentando localizar alguno de los datos transmitidos por el canal. El comando de Wireshark para ello es frame contains "[palabra clave]". Dónde [palabra clave] será cada uno de los datos establecidos en diseño para introducir en la aplicación.

Salvo los datos de posición, ninguno más es detectado por el filtro de Wireshark. El resto de activos se consideran seguros. Lo cierto es que emplea una API externa (concretamente la API de Google Maps) y ofrece conexión HTTPS sólo en su versión de pago. Dado el peso de la entidad Banco Santander y la importancia de todos los datos en este tipo de aplicaciones, Banco Santander debería adquirir el producto de Google que les permite una conexión segura con la API en la aplicación. También tienen la alternativa de buscar un proyecto libre de posicionamiento geográfico, por ejemplo, OpenMaps (9).

Por lo tanto, a nivel de transmisión, **falla la seguridad del activo posición geográfica** aunque sea por la utilización de un servicio externo.

4.1.1.2 Resultado de las pruebas de almacenamiento en Banco Santander

No es posible encontrar datos que guarde la aplicación. No cuenta ni con carpeta en la tarjeta interna (junto con el resto de aplicaciones) o en la externa ni base de datos en el dispositivo. Probablemente, la aplicación consulte continuamente la base de datos desde el servidor.

Por lo tanto, a nivel de almacenamiento, se puede asegurar que la aplicación no se exponen los activos.

4.1.1.3 Resultado de las pruebas de código en Banco Santander

Las clases ya se indiciaron en el apartado de diseño.

- Supernet, tarjeta comoLlegar y mapaComoLlegar reciben los datos por intents y es el Android el encargado de la seguridad de dichos datos.
- Clase myLocation: los parámetros de latitud y altitud se obtienen vía URI por una conexión HTTP. Lo mismo sucede con código postal.

Al implementar concretamente HTTP en la aplicación podrían quedar datos expuestos en las transmisiones de datos. **Se ha demostrado en las pruebas de transmisión que**

la posición queda expuesta ante las amenazas. Realmente, se han categorizado como de bajo riesgo y no resulta crítico para la aplicación.

4.1.2 Bankia

4.1.2.1 Resultado de las pruebas de comunicación en Bankia
No se pueden realizar pruebas por no disponer de cuenta electrónica con este banco.

4.1.2.2 Resultado de las pruebas de almacenamiento en Bankia
No se pueden realizar pruebas por no disponer de cuenta electrónica con este banco.

4.1.2.3 Resultado de las pruebas de código en Bankia
La aplicación Bankia está completamente en claro y, por lo tanto, es muy sencilla de entender. Además, no oculta las medidas de seguridad que se toman.

El intercambio de información entre clases siempre se hace mediante Intents y boradcast. Mecanismos gestionados por el sistema operativo y más seguros que implementaciones propias.

Las clases que gestionan las conexiones emplean HTTPs para realizar las comunicaciones, por lo tanto, se prevé que serán seguras en todos los aspectos.

Las consultas a las bases de datos son parametrizadas y se considera un mecanismo seguro, algunas consultas las realiza a la aplicación web directamente y obtiene una respuesta. Las consultas directamente a la web no son recomendables ya que están expuestas a ataques de *SQL injection*, el atacante puede introducir código en la consulta y alterar la original.

En definitiva, **se considera una aplicación segura desde el punto de vista de la implementación.**

4.1.3 BBVA

4.1.3.1 Resultado de las pruebas de comunicación en BBVA
No se pueden realizar pruebas por no disponer de cuenta electrónica con este banco.

4.1.3.2 Resultado de las pruebas de almacenamiento en BBVA
No se pueden realizar pruebas por no disponer de cuenta electrónica con este banco.

4.1.3.3 Resultado de las pruebas de código en BBVA
La aplicación de BBVA está ligeramente codificada y no resulta trivial su comprensión. Este tipo de prácticas no están aconsejadas a nivel de seguridad, ya que no se debería proteger la aplicación por oscuridad. Probablemente, los desarrolladores quieran proteger su código por cuestiones de diseño más que de seguridad. En cualquier caso es un aspecto a tener en cuenta.

La aplicación emplea generalmente intents para comunicarse con otras clases pero también emplean broadcast para hacerlo con varias actividades, servicios, etc. simultáneamente. Estos mecanismos se consideran seguros y protegidos por el sistema operativo.

Esta aplicación emplea ficheros temporales durante su ejecución pero, parece ser, que los mantiene en la tarjeta una vez cerrada la aplicación. Esta práctica no es muy aconsejable, es mejor mantener los datos temporales en memoria. Todo apunta a que estos temporales son archivos de configuración, ficheros de texto legal, imágenes almacenadas en directorios externos, etc.

La aplicación cuenta con un sistema propio de cifrado *tripleDESEncrypt* (Cifrado con triple DES) más seguro que el DES simple pero no se considerar suficiente (10).

Como se puede ver, se han preocupado enormemente en proteger los datos de la aplicación y, aunque es mejorable, puede ser suficiente para no exponerse ante las amenazas del grupo de aplicaciones bancarias.

La aplicación del BBVA se considera segura a nivel de código.

4.1.4 ING DIRECT

4.1.4.1 Resultado de las pruebas de comunicación en ING Direct

No se pueden realizar pruebas por no disponer de cuenta electrónica con este banco.

4.1.4.2 Resultado de las pruebas de almacenamiento en ING Direct

No se pueden realizar pruebas por no disponer de cuenta electrónica con este banco.

4.1.4.3 Resultado de las pruebas de código en ING Direct

La aplicación adapta métodos de la aplicación web a Android y, por lo tanto, muchas acciones no están implementadas nativamente si no que envía una solicitud JavaScript y JSON.

Nativamente de Android utiliza intents y broadcast para la comunicación, estos mecanismos son controlados por el sistema operativo y se consideran seguros. También implementa métodos de JSON encargados de ejecutar directamente JavaScript sobre Android.

Los métodos JavaScript se emplean para ejecutar métodos de seguridad, conexiones con bases de datos, obtener posición geográfica, etc. Está considerado seguro por Google y aparece en la documentación oficial de Android (11). Las consultas a bases de datos se envían parametrizadas y son seguras.

La conexión de la aplicación la crea sobre el protocolo HTTPs y, por lo tanto, todos los datos transmitidos están cifrados.

La aplicación del banco ING Direct se considera segura a nivel de implementación.

4.2 Resultado de las pruebas de grupo de aplicaciones de comunicación

4.2.1 Facebook

4.2.1.1 Resultado de las pruebas de comunicación en Facebook

Facebook realiza todas las conexiones mediante HTTPs con el protocolo TLSv1. Este protocolo ofrece comunicación de datos cifrada. El resto de paquetes con de control

de flujo (TCP) que garantizan la llegada de todos los paquetes en la comunicación cliente-servidor.

La aplicación no emplea protocolo HTTP para ningún tipo de comunicación y, por lo tanto, no es posible capturar datos en claro transmitidos por Facebook.

El cliente Android de Facebook se considera seguro a nivel de comunicación.

4.2.1.2 Resultado de las pruebas de almacenamiento en Facebook

Facebook no almacena datos en el propio dispositivo, todos los datos que maneja la aplicación los obtiene directamente del servidor. El único que es probable que guarde es el usuario y contraseña pero estos datos son gestionados por el sistema operativo a través de sincronización de cuentas.

A nivel de almacenamiento en soporte secundario, se considera Facebook seguro.

4.2.1.3 Resultado de las pruebas de código en Facebook

La aplicación de Facebook nombra a las clases de manera descriptiva pero no a los atributos. En cualquier caso, la aplicación se puede comprender fácilmente aunque el tamaño es mucho mayor que cualquier otra aplicación.

En líneas generales, emplea principalmente intents para el paso de datos entre clases. Es un mecanismo regulado por el sistema operativo y es confiable.

Las consultas de bases de datos se realizan mediante métodos parametrizados y, por lo tanto, es más seguro que con consultas directas.

Facebook está implementado con buenos mecanismos de seguridad y se considera una aplicación segura a nivel de código.

4.2.2 Whatsapp

4.2.2.1 Resultado de las pruebas de comunicación en Whatsapp

En primer lugar, se filtran paquetes HTTP pero esta aplicación no establece conexiones mediante HTTP, son todos con HTTPs.

Todos los datos se transmiten mediante protocolo TLSv1 y el resto de paquetes son TCP de control de comunicación.

Para asegurar que todos los datos transmitidos están en los paquetes firmados se emplea frame contains “[palabras clave]”. Se han introducido todas las seleccionadas en el apartado de diseño y **el número de teléfono aparece en algunos paquetes SSL**. Uno pertenece a Twitter (posiblemente del servicio encargado de recibir los mensajes del Time Line), los otros dos pertenecen a Whatsapp.

El número de teléfono es el identificador de usuario de esta aplicación. Por lo tanto, fue marcado como activo y debe estar protegió ante sus amenazas. Como es el caso de esta aplicación, se considera parcialmente desprotegida.

4.2.2.2 Resultado de las pruebas de almacenamiento en Whatsapp

Whatsapp emplea una capeta para almacenar todos los archivos que necesita. Esta aplicación la emplea para guardar backups, bases de datos locales, archivos multimedia, imágenes de perfil y miniaturas.

En backups guarda información de restauración, el único archivo almacenado allí se llama wallpapers. Se puede deducir que la información que guarda para restaurar en los fondos de pantalla elegidos para cada conversación.

En la carpeta bases de datos, se encuentra archivos cifrados con el nombre msgstore-[fecha].db.crypt. Se ha probado a abrir con la herramienta para bases de datos en SQL y, efectivamente, esta cifrado y no se puede abrir. Por lo tanto, los mensajes se consideran protegidos en su almacenamiento en el dispositivo.

Archivos multimedia contiene todas las imágenes, videos, audios,… que se han transmitido por las conversaciones sin ningún tipo de protección. En Android, el acceso a la tarjeta SD está permitido por defecto a cualquier aplicación sin solicitar permisos. Por lo tanto, las carpetas deberían estar más protegidas o, incluso, mantenerlas solamente en servidor.

En imágenes de perfil, la aplicación sólo tiene almacenada una foto y no es la actual del contacto. Por lo tanto, no emplea esta carpeta para todas las imágenes de perfil y no queda claro para que es utilizada. La imagen en cualquier caso tampoco está protegida.

Miniaturas es una carpeta que se ha quedado anticuada, en ella guardaba imágenes más pequeñas para mostrar una vista previa de las imágenes transmitidas pero, actualmente, se descarga directamente y muestra una imagen menor de la descargada.

La aplicación Whatsapp no se considera completamente segura a nivel de almacenamiento en soporte secundario, pero los activos más importantes sí están protegidos aunque **los elementos multimedia son accesibles desde cualquier aplicación**.

4.2.2.3 Resultado de las pruebas de código en Whatsapp

La aplicación está relativamente codificada es difícil su comprensión pero se puede observar que los intercambios de datos los realiza normalmente con intents y algún binder, los datos no están codificados pero la conexión sí (tal y como se vio en el aparatado 4.2.2.1) y los archivos multimedia no cuenta con ningún tipo de protección desde la propia aplicación.

Por lo tanto, la a**plicación no se considera protegida ya que no cifra los datos entre cliente y servidor,** es decir, la información será legible en el servidor. Además no es recomendable proteger la aplicación por ofuscación (codificación nombres de clases).

4.2.3 Twitter

4.2.3.1 Resultado de las pruebas de comunicación en Twitter

La aplicación de Twitter emplea conexión HTTP sin cifrar únicamente para conectar con la API de Twitter. A partir de ese momento, el cliente negocia con el servidor un par de claves con el que consiguen una conexión cifrada.

Ninguno de los activos marcados para este grupo queda expuesto antes sus amenazas pero si que es posible encontrar el número de teléfono entre los datos del protocolo SSL y en un mensaje XML.

La aplicación de Twitter para Android se considera segura aunque no debería mostrar el número de teléfono en claro durante las transmisiones.

4.2.3.2 Resultado de las pruebas de almacenamiento en Twitter

Twitter no almacena datos en el propio dispositivo, todos los datos que maneja la aplicación los obtiene directamente del servidor. El único que es probable que guarde es el usuario y contraseña pero estos datos son gestionados por el sistema operativo a través de sincronización de cuentas.

A nivel de almacenamiento en soporte secundario, se considera Twitter seguro.

4.2.3.3 Resultado de las pruebas de código en Twitter

El código de Twitter no es completamente descriptivo pero permite una correcta compresión de la aplicación.

Destaca que las clases que tienen nombre en clave, normalmente, se emplean para gestionar conexiones. Probablemente, Twitter trata de proteger mecanismos eficientes de conexión pero también oscurece las medidas de seguridad y no se considera una buena práctica de implementación.

La aplicación intercambia datos entre clases mediante intents y solicita información de la base de datos exterior de Twitter mediante consultas parametrizadas de SQL. También emplea solicitudes URI para comunicarse. Estas técnicas están protegidas por el sistema operativo y son bastante seguras.

El cliente Android de Twitter se considera seguro a nivel de codificación aunque debe mejorar la claridad del código.

4.2.4 Gmail

4.2.4.1 Resultado de las pruebas de comunicación en Gmail

En primer lugar, Gmail negocia en claro con el servidor un par de claves que servirán para cifrar la conexión entre ambos. A partir de este momento, los datos se transmiten a través del protocolo TLSv1 dejando cifrados todos los activos que se podrían interceptar en la conexión. Aunque sucede igual que con la aplicación de Twitter (4.2.3.1) y es posible interceptar el número de teléfono en un paquete SSL.

El resto de paquetes son TCP para controlar el flujo y garantizar la recepción de la información en el desino.

La aplicación de Gmail se considera segura a nivel de comunicación aunque habría que evitar exponer el número de teléfono por confidencialidad del usuario.

4.2.4.2 Resultado de las pruebas de almacenamiento en Gmail

Los archivos que se almacenan en el dispositivo lo realizan en el directorio de backup de Android. La carpeta contiene algunas bases de datos y ficheros de configuración. Por ejemplo, almacenan datos de configuración de la aplicación como idioma o parámetros de las preferencias.

En las bases de datos se pueden encontrar datos sobre usuarios, direcciones de correo, asuntos, cuerpos de mensajes (un resumen, Ilustración 30), nombres de elementos adjuntos, etc… en claro. Los cuerpos de correo están comprimidos en un formato desconocido para no ocupar tanto espacio pero muestra un resumen en claro. El formato del cuerpo completo comprimido permite recuperar toda la información.

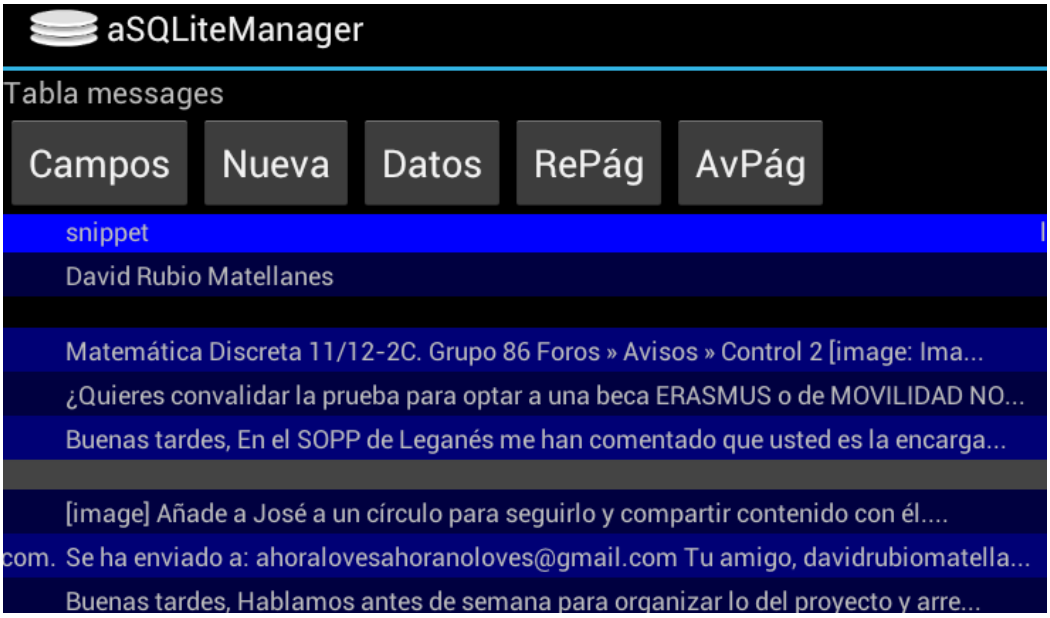


Ilustración 30 - Resumen de cuerpo de correos Gmail

Los ficheros de configuración no contienen información relevante de la aplicación, únicamente datos sobre preferencias y ajustes realizados por el usuario.

El almacenamiento de los datos de la aplicación en el dispositivo no se considera seguro ya que es posible recuperar todos los datos.

4.2.4.3 Resultado de las pruebas de código en Gmail

La aplicación de Gmail tiene nombres muy descriptivitos y permiten comprender fácilmente el código.

La aplicación se comunica internamente mediante intents y mensajes broadcast, estos los controla el sistema operativo y se consideran mecanismos seguros. La aplicación obtiene información externa mediante URI y por sincronización a través del sistema operativo.

Las consultas SQL las realiza siempre mediante métodos parametrizados y mitiga la realización de SQL Injection.

No hay indicios de que la aplicación cifre los **mensajes** antes de enviarlos, por lo tanto, están **en claro en el servidor**. **Tampoco ofrece firma digital en el cliente Android**, algo que debería ser habitual en los clientes de correo.

En conclusión, la aplicación de Gmail se considera segura aunque, por seguridad para los usuarios, **sería recomendable mejorar los aspectos de firma y cifrado digital**.

4.3 Resultado de las pruebas de grupo de aplicaciones de login

4.3.1 Dropbox

4.3.1.1 Resultado de las pruebas de comunicación en Dropbox

En primer lugar, los datos introducidos de usuario y contraseña se emplean para sincronizar una cuenta a través del propio Android y no de manera independiente. Estos datos los tramita el sistema operativo y se envían siempre a través de una conexión HTTPs.

El resto de información es toda transmitida a través de paquetes TLSv1 y controlada por los paquetes TCP. Parece ser que el cliente-servidor crea un nuevo par de claves por cada sincronización cliente-servidor.

Se considera que la aplicación protege la comunicación entre el cliente y el servidor frente a las amenazas a las que se enfrenta.

4.3.1.2 Resultado de las pruebas de almacenamiento en Dropbox

Dropbox no emplea una carpeta propia directamente en el almacenamiento interno, externo si se instala ahí, si no que emplea un directorio anidado dentro del propio Android. En dicha carpeta guarda sólo los elementos de la nube que se han solicitado descargar al dispositivo pero en claro.

La aplicación no almacena datos en una base de datos local, sólo emplea comunicación con el servidor del servicio en la nube.

El mayor problema es que cualquier aplicación puede leer sin restricciones de la SD y, por lo tanto, tendrían acceso a documentos pertenecientes a la nube privada o compartidos con otros usuarios. Los documentos, al menos los privados, deberían almacenarse cifrados para que el resto de aplicaciones no tengan acceso a su contenido.

La **aplicación expone ante las amenazas los documentos y material multimedia sincronizados** desde el dispositivo. Por lo tanto, no se considera del todo segura la aplicación.

4.3.1.3 Resultado de las pruebas de código en Dropbox

En este apartado se puede corroborar que los archivos no son cifrados y, por lo tanto, a nivel de aplicación están en claro. Es decir, son accesibles desde el servidor y desde el cliente. Para ser completamente seguro, debería verificar que el usuario que abre el documento tiene permiso (en la nube) para acceder a él.

No se puede considerar completamente seguro pero estas condiciones son habituales en todos los programas y aplicaciones y hay que tenerlo en cuenta para el método de empleo.

4.3.2 Evernote

4.3.2.1 Resultado de las pruebas de comunicación en Evernote

Evernote no transmite ningún dato sobre HTTP siempre lo realiza sobre conexiones seguras HTTPs y TLSv1, es decir, todos los datos se transmiten cifrados por la red.

El resto de paquetes son de control de flujo para garantizar la llegada de los paquetes al destino.

En conclusión, se puede afirmar que Evernote es seguro a nivel de transmisión de datos.

4.3.2.2 Resultado de las pruebas de almacenamiento en Evernote

La aplicación Evernote almacena datos directamente sobre la tarjeta SD en una carpeta del mismo nombre que la aplicación. La carpeta contiene dos directorios donde almacena las notas: notes (notas) y unsaved_notes (notas sin guardar).

En ambas carpetas se pueden ver carpetas que contienen un archivo de contenido con formato .enml, parece que se trata de un fichero propio basado en XML. Al verlos como documentos de texto, se puede ver que es un lenguaje de marcado que contiene la información de las notas en claro (Ilustración 31). En el archivo, los elementos adjuntos aparecen indicados con un hash y el elemento adjunto aparece como un archivo de datos en el mismo directorio que el archivo de contenido.

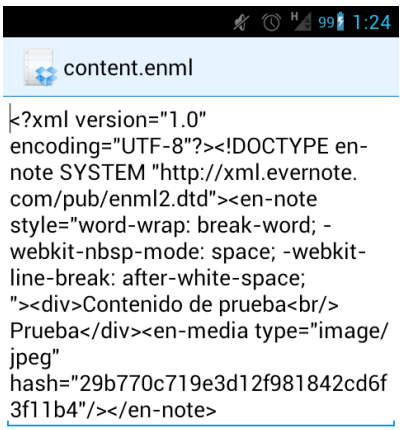


Ilustración 31 - Captura nota Evernote

La aplicación Evernote no se puede considerar protegida a nivel de almacenamiento en el dispositivo.

4.3.2.3 Resultado de las pruebas de código en Evernote

La aplicación Evernote tiene nombres de clases muy codificados y no permite descubrir mucha información de la aplicación. Es una práctica desaconsejable a nivel de seguridad ya que la seguridad no debe obtenerse por seguridad.

Las clases seleccionadas certifican que no se codifican las notas ni los elementos multimedia. Un punto a favor, los datos siempre se comparten entre las clases de la aplicación por intents y es Android el encargado de la seguridad.

El proveedor de contenido de Evernote realiza consultas a la base de datos parametrizadas y, por lo tanto, muy seguras. Algunas solicitudes las realiza mediante URI pero son para obtener rutas de segmentos.

La aplicación Evernote se considera segura a nivel de código en aspectos de seguridad aunque no debería proteger el código por ofuscación.

4.3.3 Google Drive

4.3.3.1 Resultado de las pruebas de comunicación en Google Drive

Google Drive negocia en claro con el servidor un par de claves que servirán para cifrar la conexión entre ambos. A partir de este momento, los datos se transmiten a través del protocolo TLSv1 dejando cifrados todos los activos que se podrían interceptar en la conexión.

El resto de paquetes son TCP para controlar el flujo y garantizar la recepción de la información en el desino.

El cliente Android de Google Drive se considera seguro a nivel de comunicación.

4.3.3.2 Resultado de las pruebas de almacenamiento en Google Drive

Los archivos que se almacena la aplicación Google Drive en el dispositivo lo realizan en el directorio de backup de Android. La carpeta contiene algunas bases de datos y ficheros de configuración.

Las bases de datos no contienen información acerca de los documentos, usuarios, imágenes, etc. sólo datos que emplea para el correcto funcionamiento de la aplicación.

Los ficheros de configuración no contienen información relevante de la aplicación, únicamente datos sobre preferencias y ajustes realizados por el usuario.

El almacenamiento de los datos de la aplicación en el dispositivo se considera seguro ya que apenas almacena datos en el dispositivo. A diferencia de Dropbox, Google Drive trata los ficheros directamente en la nube.

4.3.3.3 Resultado de las pruebas de código en Google Drive

El código fuente de Google Drive está fuertemente codificado y apenas permite su correcta comprensión.

En general se pueden encontrar intents y mensajes de broadcast para el intercambio de información de clases, las consultas SQL se realizan mediante parámetros y no hay indicios de empleo de cifrado de los documentos, dejándolos visibles para el servidor.

La aplicación no tiene un código claro pero cubre las necesidades básicas de seguridad. Por lo tanto, se considera del todo segura.

4.3.4 Spotify

4.3.4.1 Resultado de las pruebas de comunicación en Spotify

Las pruebas de Spotify están divididas en dos. En primer lugar se accede a la aplicación mediante una cuenta de Facebook, una oportunidad para probar la seguridad de la API de Facebook, y posteriormente se probará la aplicación en si misma.

El acceso emplea conexión cifrada HTTPs y transmite los datos por TLSv1 completamente cifrados. Los paquetes SSL ofrecen ningún tipo de información acerca de los datos de negociación de contraseña,

La conexión puramente con Spotify se hace mediante HTTP pero no completamente en claro, es decir, la conexión no está cifrada pero el paquete no se adapta al protocolo HTTP y la información no es accesible y clara.

La aplicación se considera protegida ante las amenazas relacionadas con su grupo.

4.3.4.2 Resultado de las pruebas de almacenamiento en Spotify

La aplicación Spotify guarda, en su carpeta de datos de Android, archivos de caché y ficheros. En caché, se puede observar una carpeta de almacenamiento (storage) y en ficheros, se puede observar una carpeta de usuarios y archivos de configuración y caché.

La carpeta de almacenamiento contiene archivos cifrados y distribuidos en carpetas. Por el tamaño total (unos 180MB) se puede deducir que son las pistas de reproducción almacenadas en el dispositivo. Por lo tanto, los contenidos de pago están almacenados de manera segura.

El la carpeta de ficheros se almacenan tres ficheros: orbit, settings, user-chache. Orbit guarda si está habilitado el modo offline, settings guarda información de proxy y de reloj y user-cache guarda marcas sobre los usuarios, algunos con el nick visible, pero la información está codificada para el entendimiento de la aplicación.

En definitiva, **Spotify emplea de manera segura el almacenamiento en el dispositivo móvil**.

4.3.4.3 Resultado de las pruebas de código en Spotify

Los nombres de las clases de la aplicación y sus atributos son muy poco descriptivos y provoca que la aplicación sea poco legible. Es poco aconsejable mejorar la seguridad de la aplicación por oscuridad, aunque es probable que sea por proteger el propio código.

Los datos internos de la aplicación se comporten mediante intents y broadcast. De los datos exteriores parece ser que emplea URI y servicios con binder. Estas medidas son muy robustas, ya que es el sistema operativo el que gestiona la seguridad de estos elementos.

La aplicación se considera segura pero **el código debería ser más descriptivo**.

4.4 Resultado de las pruebas de grupo de aplicaciones de consulta

4.4.1 RTVE

4.4.1.1 Resultado de las pruebas de comunicación en RTVE

La información se transmite completamente en claro pero en ningún momento se expone la posición actual. De hecho, no solicita el permiso para permitir obtener la posición al instalar la aplicación.

La aplicación se considera segura a nivel de comunicación de datos. Aunque todos los datos son transmitidos en claro.

4.4.1.2 Resultado de las pruebas de almacenamiento en RTVE

Como se ha comentado en el apartado anterior, **la aplicación no trata en ningún momento datos de posicionamiento** y, por lo tanto, se considera seguro.

4.4.1.3 Resultado de las pruebas de código en RTVE

La búsqueda de indicios de tratamiento de posición geográfica corrobora que la aplicación no trata de obtener en ningún momento datos de geolocalización. **Se puede decir que la aplicación no expone los activos ante sus amenazas**.

4.4.2 El País

4.4.2.1 Resultado de las pruebas de comunicación en El País

Los datos obtenidos son muy similares a la aplicación de RTVE (4.4.1.1), peticiones en claro de noticias e imágenes pero no trata información geográfica.

La aplicación se considera segura a nivel de comunicación de datos. Aunque todos los datos sean transmitidos en claro.

4.4.2.2 Resultado de las pruebas de almacenamiento en El País

Como se ha comentado en el apartado anterior, **la aplicación no trata en ningún momento datos de posicionamiento** y, por lo tanto, se considera seguro.

4.4.2.3 Resultado de las pruebas de código en El País

La búsqueda de indicios de tratamiento de posición geográfica corrobora que la aplicación no trata de obtener en ningún momento datos de geolocalización. **Se puede decir que la aplicación no expone los activos ante sus amenazas**.

4.4.3 Tiempo AEMET

4.4.3.1 Resultado de las pruebas de comunicación en Tiempo AEMET

La aplicación transmite todo en HTTP, pero la posición exacta no, únicamente solicita la información meteorológica de un municipio concreto. Por lo tanto, se puede concluir que la aplicación no transmite en ningún momento la posición geográfica exacta.

Aun así, al tratarse de comunicación en claro se pueden trazar hábitos de usuario y se puede considerar como un ataque a la intimidad del usuario.

En conclusión**, la posición geográfica no queda expuesta pero sí todos los movimientos que hay entre el servidor y el cliente**. La aplicación no se puede considerar del todo segura.

4.4.3.2 Resultado de las pruebas de almacenamiento en Tiempo AEMET

Tiempo AEMET no almacena datos en el propio dispositivo, todos los datos que maneja la aplicación los obtiene directamente del servidor. El único que es probable que guarde es el usuario y contraseña pero estos datos son gestionados por el sistema operativo a través de sincronización de cuentas.

Tiempo AEMET se considera seguro a nivel de almacenamiento en soporte secundario.

4.4.3.3 Resultado de las pruebas de código en Tiempo AEMET

El código de la aplicación de AEMET relacionado con posición geográfica se trasmite directamente por HTTP, pero no directamente los datos de latitud y altitud. Confirmando lo observado en el apartado 4.4.3.1.

La clase _ActionBarSherlockTrojanHorse y las relacionadas no parecen realizar ninguna actividad que ponga en peligro la seguridad de la aplicación.

Tiempo AEMET no se considera segura a nivel de código aunque no expone la posición geográfica sí da información acerca de los hábitos del usuario.

4.4.4 Google Maps

4.4.4.1 Resultado de las pruebas de comunicación en Google Maps

La aplicación Google Maps conecta con el dominio www.googleapis.com, que es el encargado de gestionar todas las librerías disponibles por parte de Google. La conexión con la Api de Google se realiza mediante HTTPs y, a partir de ese momento, todos los datos son cifrados.

En ningún momento es posible ver la posición exacta entre los datos enviados al servidor de Google Maps. Ni tampoco ningún otro dato, por lo tanto, no es posible trazar hábitos del usuario sin su permiso.

La aplicación Google Maps protege todos los datos transmitidos y se considera segura.

4.4.4.2 Resultado de las pruebas de almacenamiento en Google Maps

Los archivos que se almacenan en el dispositivo lo realizan en el directorio de backup de Android. La carpeta contiene algunas bases de datos y ficheros de configuración. En ambos datos de configuración de la aplicación como idioma o parámetros de las preferencias.

Entre las bases de datos, está da_destination_history que es la encargada de almacenar los destinos. La tabla sólo viene definida pero no contiene datos. El resto de bases de datos son información de la aplicación pero no información sobre la posición geográfica.

Los ficheros de configuración no contienen información relevante de la aplicación, únicamente datos sobre preferencias y ajustes realizados por el usuario.

El almacenamiento de los datos de la aplicación en el dispositivo se considera seguro ya que los activos no quedan expuestos ante sus amenazas.

4.4.4.3 Resultado de las pruebas de código en Google Maps

La aplicación de Google Maps, al igual que otras analizadas, no tiene el código completamente en claro, es decir, emplean nombres de clases y atributos codificados. Este hecho no permite conocer bien la aplicación y, aunque se puede tratar de un mecanismo anti-copia, no es recomendable ofrecer más seguridad por oscuridad.

En cualquier caso, se puede comprobar que los datos que se transmiten entre las distintas clases de la aplicación se realizan mediante intents y es el sistema operativo el que se encarga de gestionar la seguridad.

También se puede apreciar que se emplea, en relación con la localización, para manejar ficheros .jar y, por lo tanto, puede emplearse para obtener algún tipo de código directamente de las clases. Esta práctica no está recomendada y debería ajustarse a los métodos de las API.

Todas las consultas las realiza mediante peticiones parametrizadas, por lo tanto, más seguras.

En definitiva, el código de la aplicación parece ser seguro pero **no estar claro** y **desempaquetar información directamente de archivos .jar** no son buenas prácticas de programación y podrían exponer la aplicación a multitud de amenazas. Por lo tanto, **no se considera completamente segura la implementación de esta aplicación.**

5 Gestión de proyecto

En esta sección se muestra la evolución entra la planificación realizada para este proyecto y el coste real de su ejecución. A continuación, se detallará el hardware y software empleado para la realización del proyecto. Por último, se realiza un cálculo del coste total del proyecto.

5.1 Planificación del proyecto

En este apartado se describirán las diferentes tareas que se planificaron inicialmente y cuanto tiempo llevo realizarlas. Para facilitar la compresión, se emplearán diagramas de Gantt.

5.1.1 Planificación inicial

En un principio, se planificaron unas fases muy genéricas y el tiempo estimado era de 220 horas. En el siguiente apartado, se podrá observar el tiempo real que se ha necesitado para llevar el proyecto a cabo.

Las tareas inicialmente fueron:

- Análisis:
    - Análisis de grupos de aplicaciones: categorías de aplicaciones en Google Play.
    - Análisis aplicaciones de grupos: el objetivo era encontrar las agrupaciones de aplicaciones con datos similares.
    - Análisis herramientas Google
        - Análisis herramientas Google – memoria: búsqueda de información de herramientas de Google para gestión de memoria.
        - Análisis herramientas Google – almacenamiento: búsqueda de información de herramientas de Google para gestión del almacenamiento.
        - Análisis herramientas Google – comunicación: búsqueda de información de herramientas de Google para gestión de la comunicación.
    - Análisis herramientas terceros
        - Análisis herramientas terceros – memoria: búsqueda de información de herramientas de otras empresas para gestión de la memoria.
        - Análisis herramientas terceros – almacenamiento: búsqueda de información de herramientas de otras empresas para gestión del almacenamiento.
        - Análisis herramientas terceros – comunicación: búsqueda de información de herramientas de otras empresas para gestión de la comunicación.
    - Análisis pruebas propias
        - Análisis pruebas propias – memoria: estudio de la necesidad de crear una nueva herramienta para realizar pruebas propias sobre memoria.

        - Análisis pruebas propias - almacenamiento: estudio de la necesidad de crear una nueva herramienta para realizar pruebas propias sobre almacenamiento.
        - Análisis pruebas propias – comunicación: estudio de la necesidad de crear una nueva herramienta para realizar pruebas propias sobre comunicación.
    - Identificación de comprobaciones
        - Comprobaciones de seguridad mínimas: requisitos mínimos que cada grupo debe cumplir para ser considerado ligeramente seguro.
        - Comprobaciones de seguridad intermedias: requisitos mínimos que cada grupo debe cumplir para ser considerado moderadamente seguro.
        - Comprobaciones de seguridad completas: requisitos mínimos que cada grupo debe cumplir para ser considerado seguro.
- Diseño
    - Diseño pruebas herramientas: relación de pruebas con herramientas.
    - Diseño pruebas memoria: detalle de pruebas para encontrar vulnerabilidades en memoria.
    - Diseño pruebas almacenamiento detalle de pruebas para encontrar vulnerabilidades en almacenamiento.
    - Diseño pruebas comunicación detalle de pruebas para encontrar vulnerabilidades en comunicación.
    - Asignación comprobaciones de seguridad: relación de las comprobaciones concretas con los diferentes grupos.
- Implementación
    - Implementación pruebas herramientas
        - Implementación pruebas memoria: realización de las pruebas de memoria.
        - Implementación pruebas almacenamiento: realización de las pruebas de almacenamiento.
        - Implementación pruebas comunicación: realización de las pruebas de comunicación.
    - Resultados comprobaciones de seguridad: contrastación de resultados.

Destacar que la planificación de cada tarea incluye la ejecución y documentación total de la misma. Por lo tanto, cada tarea es más costosa pero al final la carga de trabajo para obtener la documentación será menor.

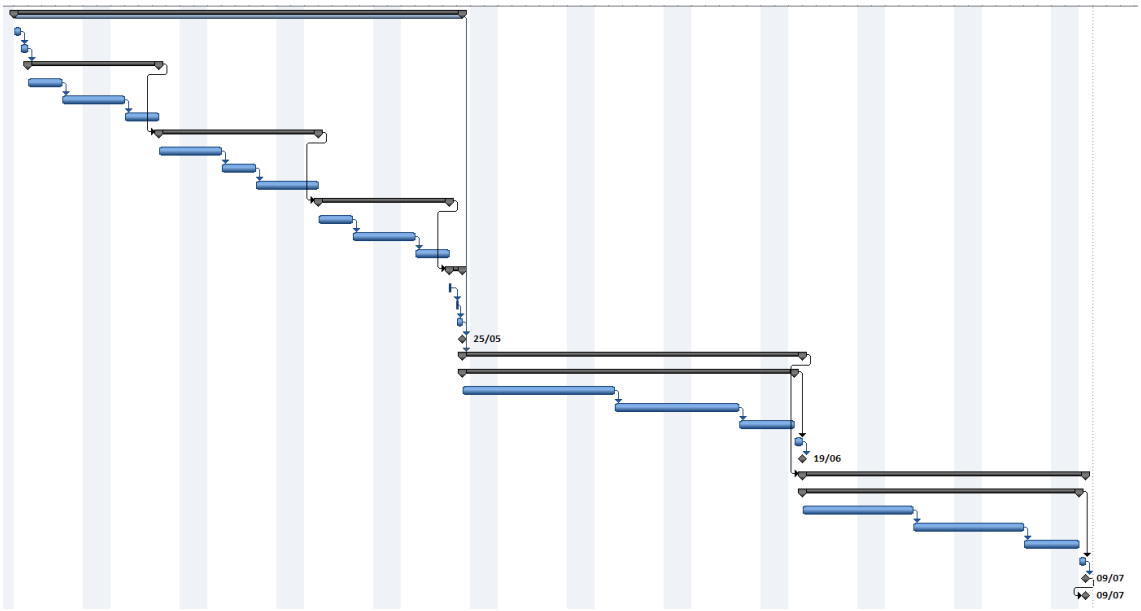


Ilustración 32 - Planificación inicial

5.1.2 Planificación real

La planificación inicial se cumplió durante las primeras tareas hasta en las últimas etapas de análisis y todo el diseño sufrió una fuerte remodelación que retrasó el inicio de las nuevas tareas.

Además, durante los meses Julio y Agosto las conversaciones con el tutor se redujeron, lo que provocó un mayor coste en la realización de cada tarea.

Durante la ejecución de la planificación real se siguió con la idea de realizar las tareas y documentar todos los aspectos. Para que al final, sea más llevadera la tarea de documentación completa de proyecto.

Las modificaciones realizadas fueron:

- Sustitución de pruebas de memoria por pruebas de código: se consideró que en el código se podría encontrar información más interesante, ya que las pruebas de memoria sólo serían útiles sobre dispositivos *rooteados.*
- El diseño de las pruebas se divide en dos secciones: una sección de diseño genérico de pruebas, útiles para todos los grupos de aplicaciones. Y otra sección donde se diseña pruebas concretas para las aplicaciones de muestra de cada grupo.
- La implementación de las pruebas pasa a ser obtención de resultados: en realidad este proyecto no se pueda adaptar a un ciclo de vida del desarrollo de una aplicación software. Carece de sentido implementar las pruebas, en realidad, se realizan y se obtienen resultados.

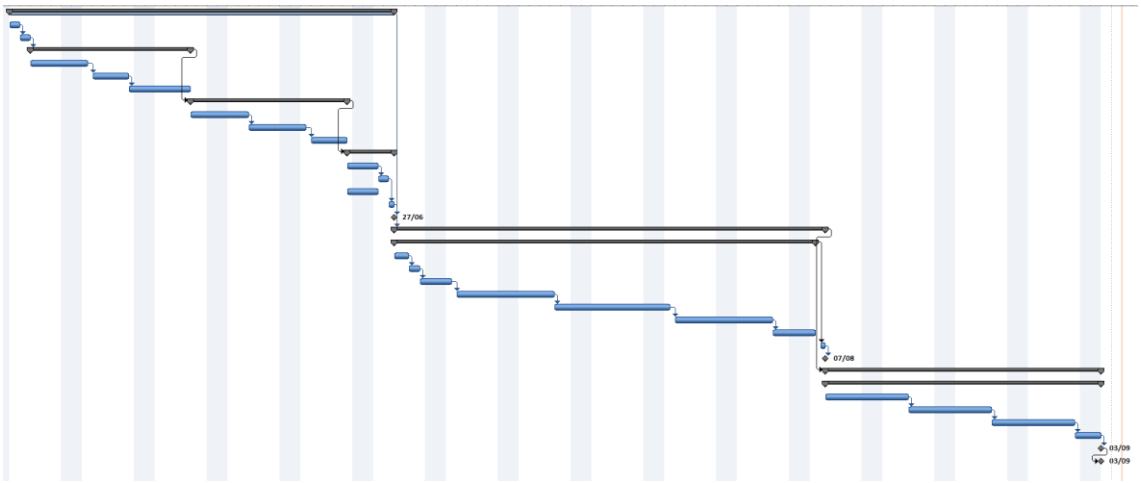


Ilustración 33 - Planificación real

5.2 Medios técnicos empleados

En esta sección se mostrarán los diferentes instrumentos empleados para el desarrollo del proyecto. Tanto de medios hardware como software, se han empleado los mínimos e imprescindibles para llegar a obtener los resultados previstos.

5.2.1 Hardware

En la Tabla 26 están todos los elementos hardware empleados para el desarrollo del proyecto.

ORDENADOR PC	DISPOSITIVO ANDROID
Mountain Xtreme i7-SB, procesador Intel i7 2700k 3.4GHz,8 GB RAM 1333MHz	Samsung Galaxy i9000, procesador 1GHz, 380MB RAM

Tabla 26 - Hardware empleado

5.2.2 Software

En la Tabla 27 están todos los elementos software empleados para el desarrollo del proyecto.

Tipo	Nombre	Página web
Sistema operativo	Windows 7	http://windows.microsoft.com/es-es/windows7/products/home?os=win7&SignedIn=1
	Linux Ubuntu	http://www.ubuntu.com/
Procesador de texto	Word 2010	http://office.microsoft.com/es-es/word/
Bloc de notas	Notepad++	http://notepad-plus-plus.org/
IDE programación	Eclipse	http://www.eclipse.org/
Android SDK	Android SDK	http://developer.android.com/intl/es/sdk/index.html
Gestor de proyectos	Project 2010	http://www.microsoft.com/project/es/es/default.aspx
Gestor bases de datos Android	aSQLitemanager	http://aaa.andsen.dk/aSQLiteManager.html

Capturador paquete Android	tPacketCapture	http://www.taosoftware.co.jp/en/android/packetcapture/
Analizador paquetes	Wireshark	http://www.wireshark.org/
Aplicaciones bancarias	Banco Santander	https://play.google.com/store/apps/details?...
	BBVA	https://play.google.com/store/apps/details?...
	Bankia	https://play.google.com/store/apps/details?...
	ING Direct	https://play.google.com/store/apps/details?...
Aplicaciones comunicación	Facebook	https://play.google.com/store/apps/details?...
	Whatsapp	https://play.google.com/store/apps/details?...
	Twitter	https://play.google.com/store/apps/details?...
	Gmail	https://play.google.com/store/apps/details?...
Aplicaciones login	Dropbox	https://play.google.com/store/apps/details?...
	Evernote	https://play.google.com/store/apps/details?...
	Google drive	https://play.google.com/store/apps/details?...
	Spotify	https://play.google.com/store/apps/details?...
Aplicaciones de consulta	RTVE	https://play.google.com/store/apps/details?...
	El País	https://play.google.com/store/apps/details?...
	Tiempo AEMET	https://play.google.com/store/apps/details?...
	GMaps	https://play.google.com/store/apps/details?...

Tabla 27 – Tabla software empleado

5.3 Análisis económico

En este apartado se va a especificar el coste de cada uno de los elementos del proyecto: recursos humanos, hardware y software.

En primer lugar, se hará un análisis económico sobre el planteamiento inicial y el resultado final. Después se calculará la variación y se verá la diferencia entre lo planificado y lo real.

5.3.1 Metodología de estimación de costes

Los costes del proyecto provienen de cuatro tipos deferentes de recursos:

- Personal: este coste es el salario del ingeniero de seguridad encargado del proyecto.
- Hardware: sólo se aplica la parte proporcional de la vida útil que se emplea en el proyecto.
- Software: sólo se aplica la parte proporcional de la vida útil o el tiempo de la licencia que se emplea en el proyecto.
- Costes indirectos: gastos en luz, teléfono y banda ancha.

5.3.2 Análisis de costes planificados

Es el coste si el proyecto hubiera transcurrido tal y como se planificó la primera vez.

5.3.2.1 Estimación coste de personal

Los costes en recursos humanos se destinan íntegramente al salario del ingeniero de seguridad encargado del proyecto. El salario hora se estima en 50€/h y el proyecto se estimó en 220h.

CONCEPTO	HORAS	HONORARIOS	COSTE
Ingeniero de seguridad	220h	50 €/h	11000€

Tabla 28 - Coste personal estimado

5.3.2.2 Estimación coste del hardware

Seguidamente, se especifican los costes del hardware durante el tiempo que dura el proyecto.

CONCEPTO	UNIDADES	PRECIO UD.	VIDA UTIL ESITMADA	TIEMPO DE USO	COSTE EN PROYECTO
Mountain Xtreme	1	1175€	48 meses	2.7 meses	66.09€
Galaxy S	1	300€	24 meses	2 meses	25€
TOTAL					91.09€

Tabla 29 - Coste hardware estimado

Los costes son todos con el IVA incluido al 18%.

5.3.2.3 Estimación coste del software

A continuación, se especifican los costes del software durante el tiempo que dura el proyecto.

CONCEPTO	UDS	PRECIO UD.	VIDA UTIL ESITMADA	TIEMPO DE USO	COSTE EN PROYECTO
Microsoft Office 2010	1	499€	48 meses	2.7 meses	28.07€
Microsoft Windows 7 Pro	1	309€	48 meses	2.7 meses	17.38€
Microsoft Project 2010	1	150€	48 meses	2.7 meses	8.44€
TOTAL					53.89€

Tabla 30 - Coste hardware estimado

5.3.2.4 Estimación costes indirectos

Estos costes son los más complicados de calcular ya que no se puede medir con exactitud cual ha sido el coste en el proyecto y cual en otras actividades paralelas.

Por lo tanto, se va a estimar la cuota de mensualidad completa para cada mes:

CONCEPTO	PRECIO MENSUAL	TIEMPO DE USO	COSTE PARA EL PROYECTO
CONEXIÓN A INTERNET	45€	2.7 meses	121.5€
LUZ	55€	2.7 meses	148.5€

CONCEPTO	PRECIO MENSUAL	TIEMPO DE USO	COSTE PARA EL PROYECTO
TOTAL			270€

Tabla 31 - Costes indirectos estimados

5.3.2.5 Estimación costes totales

En la Tabla 32 se calcula el total de los costes que acumulan recursos humanos, hardware, software y los costes indirectos.

CONCEPTO	COSTE
COSTE PERSONAL	11000€
COSTE HARDWARE	91.09€
COSTE SOFTWARE	53.89€
COSTES INDIRECTOS	270€
TOTAL	11414.98€

Tabla 32 - Costes totales estimados

El coste total que se estima obtener el proyecto es de 11414.98€. Ahora se procede a calcular el coste real del proyecto.

5.3.3 Análisis de costes reales

Es el coste del proyecto según transcurrió realmente.

5.3.3.1 Coste real de personal

Los costes en recursos humanos se destinan íntegramente al salario del ingeniero de seguridad encargado del proyecto. El salario hora se estima en 50€/h y el proyecto se estimó en 300h.

CONCEPTO	HORAS	HONORARIOS	COSTE
Ingeniero de seguridad	300h	50 €/h	15000€

Tabla 33 - Coste personal real

5.3.3.2 Coste real del hardware

Seguidamente, se especifican los costes del hardware durante el tiempo que dura el proyecto.

CONCEPTO	UNIDADES	PRECIO UD.	VIDA UTIL ESITMADA	TIEMPO DE USO	COSTE EN PROYECTO
Mountain Xtreme	1	1175€	48 meses	4.5meses	110.16€
Galaxy S	1	300€	24 meses	2 meses	25€
TOTAL					135.16€

Tabla 34 - Coste hardware real

Los costes son todos con el IVA incluido al 18%.

5.3.3.3 Estimación coste del software

A continuación, se especifican los costes del software durante el tiempo que dura el proyecto.

CONCEPTO	UDS	PRECIO UD.	VIDA UTIL ESITMADA	TIEMPO DE USO	COSTE EN PROYECTO
Microsoft Office 2010	1	499€	48 meses	4.5 meses	46.78€
Microsoft Windows 7 Pro	1	309€	48 meses	4.5 meses	28.97€
Microsoft Project 2010	1	150€	48 meses	4.5 meses	14.06€
TOTAL					89.81€

Tabla 35 - Coste hardware real

5.3.3.4 Estimación costes indirectos

Estos costes son los más complicados de calcular ya que no se puede medir con exactitud cuál ha sido el coste en el proyecto y cuál en otras actividades paralelas.

Por lo tanto, se va a estimar la cuota de mensualidad completa para cada mes:

CONCEPTO	PRECIO MENSUAL	TIEMPO DE USO	COSTE PARA EL PROYECTO
CONEXIÓN A INTERNET	45€	4.5 meses	202.5€
LUZ	55€	4.5 meses	247.5€
TOTAL			450€

Tabla 36 - Costes indirectos estimados

5.3.3.5 Estimación costes totales

En la Tabla 37 se calcula el total de los costes que acumulan recursos humanos, hardware, software y los costes indirectos.

CONCEPTO	COSTE
COSTE PERSONAL	15000€
COSTE HARDWARE	135.16€
COSTE SOFTWARE	89.81€
COSTES INDIRECTOS	450€
TOTAL	15674.97€

Tabla 37 - Costes totales estimados

El coste total real de la realización del proyecto es de 15674.97€. **La diferencia con el estimado +4259,99, un 37.3% más**. La desviación es muy alta pero el tiempo que se ha empleado de más también ha incrementado mucho. En cualquier caso, sigue siendo un proyecto viable ya que se suele establecer un porcentaje entre el 25% y el 35% en los presupuestos para situaciones de emergencia como retrasos demasiado prolongados.

6 Conclusiones y líneas futuras

En esta sección, se exponen las conclusiones en relación al proyecto y las distintas líneas de futuro para continuar el mismo.

6.1 Conclusiones

El objetivo era crear un marco de análisis eficiente para realizar una auditoria de seguridad sobre aplicaciones Android. Además, también se pretendía estudiar la situación actual de la seguridad en Android.

La clasificación creada está basada en cuatro grupos que comparten funcionalidad y datos críticos.

- El grupo de aplicaciones bancarias son aplicaciones de oficinas virtuales que permiten gestionar cuentas bancarias como si de una oficina digital se tratará.
- El grupo de comunicación permite envío de mensajes en diferentes contextos pero con la intención de relacionarse con otros usuarios. También se podría separar en puramente comunicación, mensajería, correo electrónico de los clientes de redes sociales aunque cada vez sea más complicado. Por ejemplo, Gmail es un proveedor de servicios de correo pero comparte perfil de usuario con la red social Google+ o Whatsapp que poco a poco va añadiendo más información de usuario creando un perfil similar al de las redes sociales.
- El grupo de aplicaciones con login se redujo a aplicaciones con funcionalidades en la nube, puede que no abarque todo el espectro de aplicación que requieren autenticarse pero si son las más empleadas y con contenido más critico.
- El grupo de aplicaciones de consulta compartían la funcionalidad de simplemente informar sobre algún aspecto, ya sean noticias, tiempo o posicionamiento/navegación.

Estos grupos comparten datos muy similares y, por lo tanto, se obtuvieron de cada grupo una serie de activos que podían ser atacados. Estos elementos se convertirían en el objetivo de las pruebas de análisis en función de las vulnerabilidades que se indicaran en cada caso.

Una vez marcados los objetivos del análisis, se diseñó un plan de pruebas para cada grupo. En primer lugar, se separaron pruebas de comunicación, almacenamiento y código y se detallaron algunos aspectos generales, por ejemplo, las comunicaciones seguras implementarían el protocolo HTTPS, el mejor caso era no almacenar nada en soporte secundario y de hacerlo que fuera cifrado y el código debía implementar mecanismos de comunicación interna del sistema operativo (intents, broadcast, etc.)

Posteriormente, se diseñaron pruebas específicas para las aplicaciones de muestra que se seleccionaron de cada grupo. Se indicaba que clases del código debían estudiarse con más cuidado y que acciones había que realizas mientras se capturaba la comunicación de la aplicación, para poder realizar búsquedas entre los paquetes capturados.

Las pruebas diseñadas para cada grupo han resultado ser suficientes para detectar posibles vulnerabilidades en las aplicaciones y cubrían el máximo posible de los datos que podían manejar. En algunos casos, principalmente en el grupo de aplicaciones

comunicación, las pruebas eran desmedidas y no contaban con tantos activos como se tenía previsto en el diseño general del grupo. Pero siempre se cubrían todos elementos posiblemente expuestos ante atacantes y, por lo tanto, las pruebas fueron un éxito.

La Ilustración 34 muestra los resultados obtenidos por cada aplicación en cada prueba, los resultados provienen del apartado anterior (apartado 4). En todos los grupos se han podido encontrar aspectos que indican un nivel de seguridad bajo o muy bajo y también se ha podido determinar que cumple con el objetivo de la prueba y se considera seguro.

TIPO DE PRUEBA	BANCO SANTANDER	BANKIA	BBVA	ING DIRECT	FACEBOOK	WHATSAPP	TWITTER	GMAIL	DROPBOX	EVERNOTE	GOOGLE DRIVE	SPOTIFY	RTVE	EL PAIS	TIEMPO AEMET	GOOGLE MAPS
COMUNICACIÓN	!	-	-	-	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	!	✓
ALMACENAMIENTO	✓	-	-	-	✓	!	✓	✗	!	✗	✓	✓	✓	✓	✓	✓
CÓDIGO	✓	✓	✓	✓	✓	!	!	✓	!	✓	✓	✓	✓	✓	!	!

✗	NIVEL DE SEGURIDAD MUY BAJO
!	NIVEL DE SEGURIDAD BAJO
✓	NIVEL DE SEGURIDAD ACEPTADO
-	IMPOSIBLE REALIZAR PRUEBA

Ilustración 34 - Resultados pruebas sobre todos los grupos

Un nivel de seguridad muy bajo se considera cuando una aplicación expone una gran cantidad de activos o son de gran impacto y/o riesgo. Si la aplicación no protege adecuadamente alguno de los activos y no son de gran impacto o riesgo se considera un nivel de seguridad bajo. Si todos los activos están protegidos, el nivel de seguridad es aceptado.

En muchos casos, las pruebas de código no permitían confirmar algún fallo de seguridad o la falta de algún mecanismo que reforzaría la seguridad. Por lo tanto, no se consideró seguro aunque el resto de pruebas fueron positivas.

El resultado más sorprendente es Gmail, puesto que almacena correos electrónicos en el dispositivo y, aunque el cuerpo entero sea un objeto binario, el resumen en claro es suficiente para poder acceder a información importante. Se espera más protección por parte de una aplicación del mismo desarrollador que el sistema operativo Android.

La aplicación Whatsapp era una aplicación muy vulnerable y durante el 2011 se encontraron multitud de fallos de seguridad. Por ejemplo, el registro de un móvil en el servicio se hacia en claro (12) y las conversaciones se enviaban en claro aunque se emitiera hacia el puerto 443 (13).

Por último destacar que el marco de pruebas ha resultado útil para la detección de ciertas vulnerabilidades en algunas de las aplicaciones. Ante algunas de ellas, se han propuesto alternativas para, al menos, mitigar el impacto producido. Incluso, se podría informar a los desarrolladores de las aplicaciones con vulnerabilidades y transmitirles las medidas propuestas en este documento.

6.2 Líneas futuras

Por un lado, se puede continuar y mejorar el método de análisis. En este documento se ha descrito un método manual y personalizado para ciertas aplicaciones. El objetivo de continuar por esta línea sería automatizar el proceso y flexibilizarlo para que se adaptara a las diferentes variaciones de las aplicaciones.

Por otro lado, ha quedado en el tintero el caso de los terminales con el *root* desbloqueado, es decir, con acceso al usuario administrador. Un dispositivo *rooteado* tiene acceso a la totalidad del dispositivo y puede alterar cualquier elemento de él. El punto más crítico es la memoria principal puesto que puede saltarse el *sandbox* y acceder a los datos de cualquier aplicación.

6.3 Conclusiones a nivel personal

El trabajo en este proyecto ha sido como un resumen de cualquier práctica realizada durante los cuatro años que he cursado la titulación: primero mucha ilusión, después una gran desorientación ante tanta información nueva, a continuación momentos de lucidez mezclados con desesperación y, finalmente, la alegría de haber realizado un gran esfuerzo.

Como con cada práctica realizada durante estos años, no pocas gracias al Plan Bolonia, tendrá que pasar un tiempo para conocer realmente hasta que punto he aprendido sobre un análisis de seguridad de aplicaciones Android. Pero, en este momento, destacaría la diversidad de elementos, dentro del mundo Android, que he podido estudiar y analizar. Además de los diferentes tipos de aplicaciones de mercado que he podido *diseccionar* para ver su funcionamiento real.

Por otra parte, he tenido ocasión de afianzar conocimiento en la rama con la que soñé desde el primer día en la universidad, seguridad en tecnologías informáticas. El proyecto me ha dado la oportunidad de sellar en mi mente los conceptos estudiados en asignaturas pasadas durante la carrera. Espero que esta experiencia me anime a seguir investigando sobre seguridad y dispositivos móviles y, si el mercado laboral me lo permite, vivir cómodamente trabajando en ello.

Por último, quisiera destacar que, pese a los momentos duros que he atravesado realizando el proyecto, he podido contar con mis compañeros de siempre para poder llevar con ánimo el proyecto a buen puerto, sin ellos habría sido mucho más duro.

7 Bibliografía

1. **Google Inc.** [En línea] https://play.google.com/store/apps.

2. —. Googla play. [En línea] 2012. https://play.google.com/store.

3. **Tiuri van Agten.** Distimo.com. [En línea] 2012. http://www.distimo.com/blog/2012_01_google-android-market-tops-400000-applications/.

4. **Spotify.** www.spotify.com. [En línea] 2012. http://www.spotify.com/es/legal/30-days-free-trial-terms-and-conditions/.

5. **Google Inc.** Android developers - Debugging. [En línea] http://developer.android.com/guide/developing/debugging/index.html.

6. —. Android developers - Designing for Security. [En línea] http://developer.android.com/guide/practices/security.html.

7. **BOE.** http://www.boe.es. [En línea] http://www.boe.es/boe/dias/1999/12/14/pdfs/A43088-43099.pdf.

8. **www.smartblog.es.** [En línea] 10 de 2011. http://www.smartblog.es/2011/11/tus-cuentas-y-mucho-mas-en-el-movil-con-bankia/.

9. **openmaps.eu.** [En línea] http://openmaps.eu/.

10. **www.cryptopp.com.** [En línea] www.cryptopp.com/wiki/TripleDES.

11. **Google Inc.** [En línea] http://developer.android.com/reference/org/json/package-summary.html.

12. **Lara, Carlos Fernández de.** [En línea] http://www.bsecure.com.mx/featured/usuarios-de-whatsapp-vulnerables-no-a-uno-sino-a-dos-ataques/.

13. **Berardi, Lisandro.** [En línea] http://www.incubaweb.com/vulnerabilidad-deja-al-descubierto-nuestras-conversaciones-de-whatsapp/.

14. **Pavan, Bárbara.** Bitelia.com . [En línea] 2012. http://bitelia.com/2012/04/geolocalizacion-por-que-foursquare-esta-ganando-la-carrera.

15. **RAJIVVISHWA.** [En línea] 20 de 05 de 2011. http://a4apphack.com/security/sec-code/extract-android-apk-from-market-and-decompile-it-to-java-source.