Universitat Oberta de Catalunya

www.uoc.edu

Proyecto de Fin de Máster

Título del proyecto:	Encrypting your mobile phone photos
Titulación:	Máster interuniversitario en seguridad de las tecnologías de la información y de las comunicaciones
Autor:	Francisco Javier Fernández Conde
Tutor:	Jordi Herrera Joancomartí
Fecha:	14 de junio del 2013

Índice

Reconocimiento-CompartirIgual CC BY-SA

Esta licencia permite a otros remezclar, retocar, y crear a partir de tu obra, incluso con fines comerciales, siempre y cuando te den crédito y licencien sus nuevas creaciones bajo condiciones idénticas. Esta licencia suele ser comparada con las licencias "copyleft" de software libre y de código abierto. Todas las nuevas obras basadas en la tuya portarán la misma licencia, así que cualesquiera obras derivadas permitirán también uso comercial. Esa es la licencia que usa Wikipedia, y se recomienda para materiales que se beneficiarían de incorporar contenido de Wikipedia y proyectos con licencias similares.

Introducción

El valor de la privacidad, vivimos en un mundo en el que constantemente cedemos nuestra información o datos privados a terceros. Nos vemos envueltos en una vorágine de acontecimientos y tendencias en el que apenas tenemos en cuenta uno de los derechos humanos más básicos, la intimidad.

Según el Articulo 12 de la Declaración Universal de los Derechos Humanos; *Nadie será objeto de injerencias arbitrarias en su vida privada, su familia, su domicilio o su correspondencia, ni de ataques a su honra o a su reputación. Toda persona tiene derecho a la protección de la ley contra tales injerencias o ataques*.

Por lo qué si el derecho a la intimidad es tan importante, ¿por qué le damos tan poco valor? Mediante el desarrollo de la aplicación propuesta se pretende colaborar en la difícil tarea de hacer nuestra vida privada un poco más intima. Para ello a lo largo de este documento se define el cómo y el por qué de la aplicación desarrollada a la que se le ha puesto el nombre de *CameraSecure*.

CameraSecure es una aplicación desarrollada en Java para el sistema operativo Android, y permite de una forma sencilla tomar fotografías directamente cifradas con la contraseña establecida. Las fotografías almacenadas mediante la aplicación son almacenadas en formato *PNG* y tienen la particularidad de no almacenar ningún tipo de *metadato*. Además el cifrado es aplicado directamente sobre la información útil de las imágenes por lo que pueden visualizarse con cualquier explorador de archivos.

El aspecto visual de las fotografías capturadas con la aplicación se muestra en la Figura 1.

Figura 1

Objetivos

El objetivo del Trabajo de Fin de Máster es realizar una aplicación que aporte los medios necesarios para proteger las fotografías tomadas mediante el móvil. De esta forma se añade un elemento de seguridad más para que los usuarios puedan proteger su privacidad con el mínimo esfuerzo.

Concretamente los objetivos específicos que debe cumplir la aplicación son los siguientes:

- La aplicación debe ser capaz de capturar imágenes a través de la cámara integrada en los dispositivos móviles.
- Debe permitir almacenarlas cifradas con una contraseña especificada por el usuario.
- Debe respetar el formato de las imágenes para que puedan ser vistas en cualquier navegador de archivos.
- Dicha aplicación debe poder ejecutarse en un terminal genérico de gama media con sistema operativo Android.
- Las contraseñas utilizadas para cifrar las imágenes deben ser seguras y en caso de almacenarse en memoria deben hacerlo de forma segura.

Visión general del proyecto

Desde un punto de vista funcional se puede dividir el proyecto en las siguientes partes bien diferenciadas.

1. Menú de opciones
2. Configuración de preferencias.
3. Captura de imágenes.
4. Almacenamiento de imágenes.
5. Uso de contraseñas seguras.
6. Almacenamiento de contraseñas.
7. Cifrado y descifrado de imágenes.
8. Visualización de imágenes.
9. Créditos.

Menú de opciones.

Se contemplan tres acciones posibles dentro de la aplicación. Desde el botón de menú del dispositivo se muestran las mismas. Este menú presenta el aspecto representado en la Figura2.

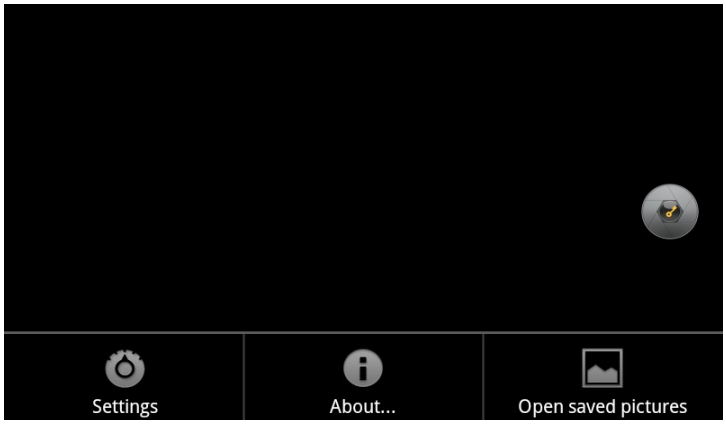

Figura 2

- Settings.- Abre una ventana con la pantalla de configuración.
- About.- Muestra un dialogo con los créditos de la aplicación.
- Open saved pictures.- Abre la galería de imágenes para descifrar las imágenes capturadas.

Configuración de opciones.

Para el correcto funcionamiento de la aplicación se han definido cuatro opciones principales que definirán el comportamiento de la aplicación.

- Password.- Indica la contraseña con la que serán cifradas las imágenes.
- Security Level. Indica el nivel de seguridad con el que se almacenará o no la contraseña elegida. Actualmente existen dos niveles de seguridad:
  - Bajo.- La contraseña se almacena cifrada en la configuración de las preferencias de la aplicación.
  - Alto.- La contraseña nunca se almacena en el teléfono, sino que se almacena su hash.
- Resolution. Define la resolución que tendrán las imágenes capturadas. Actualmente existen tres niveles:
  - 800x600
  - 1024x768
  - 1280x720
- Preview. Establece si después hacer una captura se muestra la imagen capturada o por el contrario la aplicación se queda en disposición de realizar mas capturas.

A continuación en la Figura 3, se muestra el aspecto de la pantalla de preferencias.

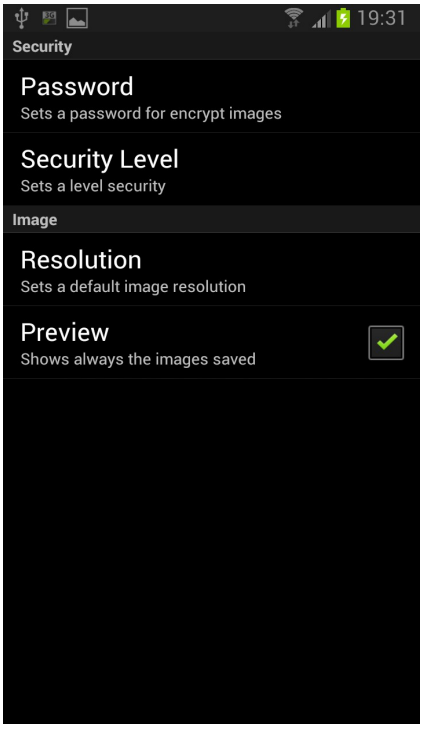

Figura 3

Captura de imágenes

El proceso de captura de imágenes se basa en utilizar el hardware aportado por los dispositivos móviles, cámara, para realizar las instantáneas. Para la comunicación con la cámara se usa la API provista por el sistema operativo, Android. El correcto funcionamiento de esta funcionalidad implica conceder los permisos específicos para usar el hardware. En este caso se debe establecer el siguiente permiso:

```
<uses-permission android:name="android.permission.CAMERA" />
```

Además se permite que la cámara realice el foco automáticamente mediante la siguiente opción de configuración.

```
<uses-feature android:name="android.hardware.camera.autofocus" />
```

Estos permisos implican que cuando se instala la aplicación se realiza una advertencia al usuario sobre qué controles o funciones va a utilizar la misma. Pueden verse en la Figura 4.

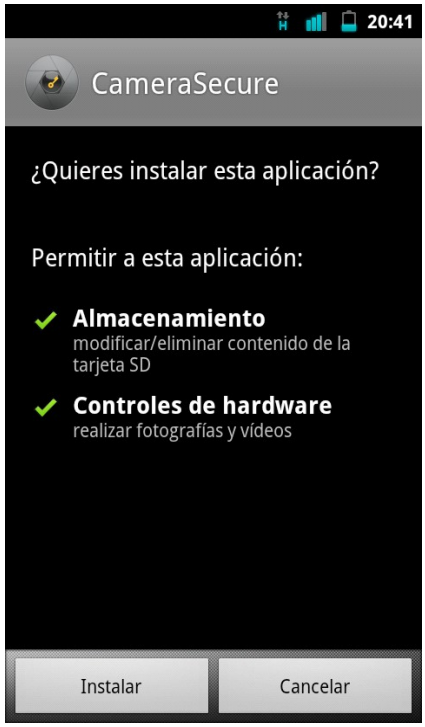

Figura 4

La pantalla designada para la captura de fotografías solamente presenta un botón para capturar las imágenes y muestra el aspecto siguiente,

representado por la Figura 5.

Figura 5

Almacenamiento de imágenes.

Las imágenes se almacenan por defecto en la memoria externa del teléfono. En concreto en la ruta siguiente:

DCIM\Secure images

De la raíz de la tarjeta SD.

De igual forma que en el caso anterior es necesario establecer los permisos de escritura necesarios para que la aplicación pueda escribir en la memoria. En este caso:

```
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

Todos los permisos están definidos en el archivo *AndroidManifest.xml*.

Uso de contraseñas seguras.

La elección de una contraseña segura es fundamental, de nada sirve implementar un sistema de cifrado infranqueable si a la hora de cifrar las imágenes se utiliza una contraseña débil. A modo de ejemplo, una simple búsqueda en Google o cualquier otro buscador nos ofrece listado de contraseñas habituales por los usuarios:

- password
- 123456
- 12345678

- qwerty
- abc123
- monkey

Si se le permite utilizar al usuario este tipo de contraseñas, toda la fortaleza del cifrado quedará en nada. Es por ello que la aplicación no permite el uso de contraseñas inseguras.

Se ha definido que las contraseñas deben tener una longitud mínima de ocho caracteres entre los cuales deben figurar letras y números.

Almacenamiento de contraseñas.

Tan importante es el almacenamiento de las contraseñas como la elección de una contraseña segura. Al igual que en el caso anterior, no sirve de nada establecer una contraseña de cifrado super compleja si luego ésta es almacenada en texto claro. Es por ello que se han definido dos niveles de seguridad dentro de la aplicación; bajo y alto.
Estos niveles de seguridad no están relacionados con el cifrado que se realiza a las imágenes, sino con el modo de almacenar la contraseña establecida.

Nivel de seguridad bajo

Establece que la contraseña se almacena en la memoria del teléfono de forma cifrada utilizando como clave para cifrarla algún elemento específico del teléfono. Actualmente se está utilizando el modelo del dispositivo.
Se puede pensar que utilizar como clave el modelo del dispositivo no aportada nada de seguridad ya que es un dato muy básico y utilizando ingeniería inversa se podría recuperar fácilmente la contraseña elegida, pero no es tal problema, como se describe a continuación.

Este nivel de seguridad implica que el usuario introduce una sola vez la contraseña en la aplicación, y como esta es guardada de forma cifrada en el teléfono, la propia aplicación puede descifrarla y utilizarla para en el cifrado de imágenes. Es decir, se está relegando toda la seguridad al teléfono.

Por ello la utilidad de este nivel seguridad es limitada, aunque una vez que las imágenes cifradas hallan sido sacadas del teléfono, la fortaleza del cifrado de las imágenes es exactamente idéntica a la del nivel medio,.

Nivel de seguridad alto

Este nivel de seguridad implica que las contraseñas nunca se almacenan en la memoria interna o externa del teléfono, en lugar de ello se almacena el *Hash* de las mismas. Se está utilizando el algoritmo SHA-256 para calcular dicho *Hash*.

Como en este nivel no hay forma de recuperar la contraseña original a partir del *Hash*, el programa pide la contraseña al usuario cada vez que se inicia o vuelve al primer plano. De esta manera se hace muy difícil, sino imposible, que un usuario mal intencionado pueda recuperar la contraseña de cifrado en este nivel.
El nivel de seguridad alto es el nivel por defecto en la aplicación.

Cifrado y descifrado de imágenes

Para el cifrado de imágenes actualmente se está usando el algoritmo *RC4*, que es un algoritmo de cifrado de flujo usado en algunos protocolos como *TLS/SSL*.

Visualización de imágenes

Para la visualización de imágenes se está usando una pantalla sacada de los ejemplos de *Android*. En concreto el ejemplo se llama *FullScreen Activity*, y muestra una lista con las imágenes cifradas. Se puede navegar por la lista de imágenes y al pulsar sobre cualquiera de ellas aparece un botón para descifrarlas. En la Figura 6, se muestra un ejemplo de la pantalla de visualización de imágenes.

Figura 6

Créditos

Por último la pantalla de créditos muestra información sobre el autor de la aplicación. Su aspecto visual se muestra en la Figura 7.

Figura 7

Diseño

La aplicación va a dividirse en tres partes principales, cada una de ellas representada por una Vista Android. Las Vistas en Android están representadas por archivos XML y normalmente se guardan bajo la carpeta *layout* del proyecto. De esta forma la aplicación presenta las siguientes Vistas.

- Vista de cámara.
- Vista de galería de imágenes.
- Vista de previsualización.
- Vista de configuración.
- Vista de créditos.

Además para la navegación entre las Vistas se ha definido un *menú* que permite navegar por la aplicación.

Vista de cámara

Esta vista viene definida por el archivo XML *activity_main.xml*:

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#0099cc"
    tools:context=".MainActivity" >

    <!--
         The primary full-screen view. This can be replaced with whatever view
         is needed to present your content, e.g. VideoView, SurfaceView,
         TextureView, etc.
    -->

     <SurfaceView
    android:id="@+id/surfaceView1"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    />

     <ImageButton
            android:id="@+id/imageButton"
             android:layout_width="80dp"
              android:layout_height="80dp"
              android:layout_gravity="right|center_vertical"
              android:background="@null"
              android:scaleType="center"
              android:src="@drawable/ic_launcher"
              android:textColor="#ffffff"
        />
```

```
</FrameLayout>
```

En esta vista se definen dos elementos principales, el primero es una *SurfaceView* que es el que nos permite visualizar las imágenes que se captan por la cámara en tiempo real.

El segundo elemento es un *ImageButton* que representa el botón para realizar la captura de imágenes.

Vista de previsualización.

Esta vista se usa para mostrar al usuario la imagen tomada por la cámara y viene definida en el archivo *preview_layout.xml*.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout  xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/LinearLayout"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <ImageView
        android:id="@+id/imagePreview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" />

</RelativeLayout >
```

Como se puede observar su estructura es muy simple, está formada únicamente por un *ImageView* que permite la visualización de imágenes.

En esta entrega, se están cifrando todas las imágenes tomadas por la cámara, en las versiones posteriores se modificará esta vista para añadir un botón de cifrar imagen, ya que durante el desarrollo de aplicación se ha detectado que es un esfuerzo inútil el cifrar todas las imágenes capturadas. De esta forma en versiones posteriores solamente se cifraran las imágenes indicadas por el usuario, mediante el botón de cifrado.

Vista de galería de imágenes.

Esta vista ha sido generada automáticamente por el generador de código que provee el entorno de desarrollo de Android, para ello se ha creado la actividad como *FullScreen Activity*, y provee un interfaz amigable para mostrar las imágenes. La vista está definida por el archivo

activity_galery.xml.

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#0099cc"
    tools:context=".MainActivity" >

    <!--
         The primary full-screen view. This can be replaced with whatever view
         is needed to present your content, e.g. VideoView, SurfaceView,
         TextureView, etc.
    -->

     <ImageView
         android:id="@+id/imageViewGallery"
         android:layout_width="fill_parent"
         android:layout_height="fill_parent" />

    <!--  <TextView
        android:id="@+id/fullscreen_content2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:keepScreenOn="true"
        android:text="@string/dummy_content"
        android:textColor="#33b5e5"
        android:textSize="50sp"
        android:textStyle="bold" />-->
    <Gallery
        android:id="@+id/gallery"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="top" />

    <!--
         This FrameLayout insets its children based on system windows using
         android:fitsSystemWindows.
    -->

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:contentDescription="@string/app_name"
        android:fitsSystemWindows="true" >

        <LinearLayout
            android:id="@+id/fullscreen_content_controls2"
            style="?buttonBarStyle"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom|center_horizontal"
            android:background="@color/black_overlay"
            android:orientation="horizontal"
            tools:ignore="UselessParent" >

            <Button
                android:id="@+id/dummy_button2"
```

```
                style="?buttonBarButtonStyle"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="@string/decrypt_button" />
        </LinearLayout>
    </FrameLayout>
</FrameLayout>
```

Los elementos principales en la vista son el *ImageView* usado para mostrar las imágenes, el *Button*, que permite descifrar las imágenes mostradas y el elemento *Gallery*, que muestra una previsualización de las imágenes tomadas y permite navegar por ellas.

Vista de configuración

Permite configurar los distintos aspectos de la aplicación, está definida en el archivo *preferences.xml.*

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Copyright (C) 2008 The Android Open Source Project

     Licensed under the Apache License, Version 2.0 (the "License");
     you may not use this file except in compliance with the License.
     You may obtain a copy of the License at

          http://www.apache.org/licenses/LICENSE-2.0

     Unless required by applicable law or agreed to in writing, software
     distributed under the License is distributed on an "AS IS" BASIS,
     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
     See the License for the specific language governing permissions and
     limitations under the License.
-->

<!-- This is a primitive example showing the different types of preferences available.
-->

<PreferenceScreen
        xmlns:android="http://schemas.android.com/apk/res/android">

    <PreferenceCategory
             android:key="@string/Security"
            android:title="@string/securityPreferenceCategory">

         <EditTextPreference
                android:key="@string/preferenceKeyPass"
                android:title="@string/preferenceTitlePass"
                android:summary="@string/preferenceSummaryPass"
                android:dialogTitle="@string/preferenceDialogPass"
                android:password="true" />
        <ListPreference
             android:entries="@array/preference_list_security_level"
             android:entryValues="@array/preference_list_security_level"
             android:summary="@string/preferenceSummaryLevelSecurity"
        android:dialogTitle="@string/preferenceSecurityLevel"
```

```
android:title="@string/preferenceSecurityLevel"
android:key="@string/preferenceSecurityLevel"/>

    </PreferenceCategory>
    <PreferenceCategory
        android:key="@string/Image"
        android:title="@string/imagesPreferenceCategory">
        <ListPreference
             android:entries="@array/preference_list_resolution"
             android:entryValues="@array/preference_list_resolution"
             android:summary="@string/preferenceSummaryListResolution"
             android:dialogTitle="@string/preferenceResolution"
android:title="@string/preferenceResolution"
android:key="@string/preferenceResolution"/>
        <CheckBoxPreference
                android:key="@string/preferenceKeyPreview"
                android:title="@string/preferenceTitlePreview"
                android:summary="@string/preferenceSummaryPreview"
                android:defaultValue="true" />
       <!--<ListPreference
             android:entries="@array/preference_list_storage_method"
             android:entryValues="@array/preference_list_storage_method"
        android:dialogTitle="@string/preferenceStorageMethod"
android:title="@string/preferenceStorageMethod"
android:key="@string/preferenceStorageMethod"/>-->
    </PreferenceCategory>


</PreferenceScreen>
```

Esta pantalla muestra las opciones de configuración de la aplicación descritas anteriormente.

Vista de créditos.

Esta Vista es la que nos permite visualizar información acerca de la aplicación. Solamente muestra una imagen y un texto y se encuentra definida en el archivo *about_layout.xml*.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/LinearLayout"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="horizontal" >

    <ImageView
        android:id="@+id/image"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView" />
</LinearLayout>
```

Implementación de CameraSecure.

Arquitectura básica de las aplicaciones Android.

Los proyectos Android se dividen en diferentes carpetas cada una de las cuales se encarga de la definición de diferentes partes de la aplicación. Como puede verse en la Figura 8 el proyecto *CameraSecure* está formado por las siguientes carpetas.

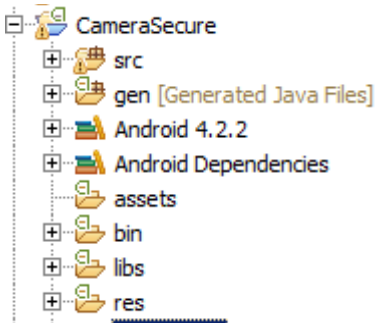

Figura 8

- *Src*.- Esta carpeta contiene todo el código fuente de la aplicación. Es aquí donde se definen las clases que determinan el comportamiento de la aplicación.
- *Gen*.- En esta carpeta se guardan los archivos generados por el *IDE*. Estos archivos se generan automáticamente después de cada compilación por lo que no deben ser modificados. Es aquí donde se ubica la clase *R* que tiene especial importancia porque en ella se encuentran las constantes que representan a los recursos.
- *Bin*.- Contiene los archivos compilados de la aplicación.
- *Lib*.- Contiene librerías externas usadas por la aplicación.
- *Res*.- Contiene los recursos usados por la aplicación. Se entienden recursos a los ficheros de imagen, video y archivos *Xml* que contienen las definiciones de la apariencia de la aplicación así como las constantes usadas en la misma.
- *Resto de carpetas*. Contiene librerías de terceros normalmente son archivos con la extensión *.jar*.

Además en la raíz del proyecto mismo se encuentra el archivo *AndoridManifest.xml* que describe como se empaquetará la aplicación, así como el número de versión específico y los permisos necesarios para su ejecución.

Componentes de una aplicación Android.

Los componentes son los elementos básicos con las que se construyen las aplicaciones Android. Hay cuatro tipos principales de componentes cada uno de los cuales es utilizado con un propósito diferente.

- Actividades. Representan pantallas de las aplicaciones. Las actividades muestran una interfaz de usuario con la cual los usuarios pueden interaccionar.
- Servicios.- son componentes indicados para la realización de tareas en segundo plano, a diferencia de las actividades no presentan interfaz de usuario.
- Proveedores de contenido. Son componentes encargados de gestionar información que las aplicaciones comparte con terceros.
- Intenciones. Son componentes encargados de realizar acciones o tareas, las intenciones permiten por ejemplo pasar de una actividad a otra.

Actividades

La aplicación se encuentra divida en cuatro Actividades, recordemos que cada actividad representa a una vista diferente de la aplicación. Las actividades son las siguientes:

- *MainActivity*
- *PreferencesActivity*
- *PreviewActivity*
- *GalleryActivity*

Desde la actividad principal, *MainActivity* se puede navegar hacia cada una de las restantes, y viceversa. El comportamiento se puede ver resumido en la vista de actividades representada por Figura 9.

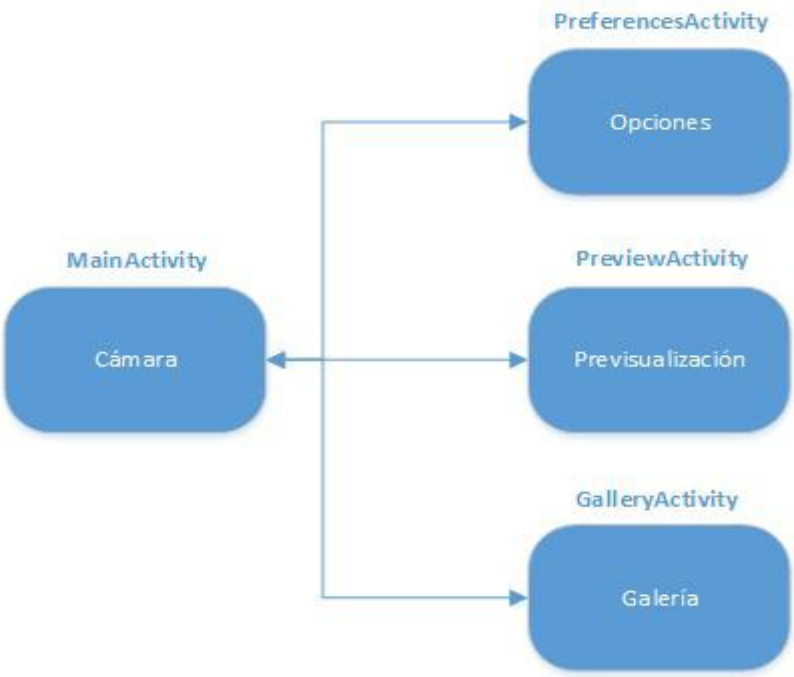

Figura 9

MainActivity.

Es la Actividad de inicio de la aplicación y realiza las siguientes funciones.

- Inicializar la cámara para poder obtener imágenes a través de ella.
- Configurar la cámara con los parámetros adecuados, tamaño de imagen, modo apaisado, flash, etc.
- Leer la configuración de la aplicación, para ello se ayuda de la clase *PreferencesActivity*.
- Provee métodos de navegación hacia las demás actividades.
- Muestra en todo momento las imágenes que la cámara está visualizando y provee un botón para realizar capturas de imagen.
- Captura imágenes y las cifra.

PreferencesActivity

Esta Actividad se encarga de almacenar las opciones de configuración del usuario. Además provee los métodos necesarios para comprobar si ya se ha establecido la contraseña de cifrado en la aplicación. En total se pueden configurar las siguientes opciones:

- Contraseña.
- Nivel de seguridad.
- Resolución.

- Previsualizaación.

PreviewActivity

Es la actividad encargada de mostrar por pantalla las imágenes sin cifrar una vez que han sido capturadas.

GalleryActivity

Permite la sencilla visualización de las capturas realizadas. Para ello expone las imágenes en una lista y ofrece un botón para descifrarlas.

Diagrama de actividades

En la Figura 10 se expone el diagrama de Actividades que representa el flujo natural resultante de la ejecución de la aplicación.

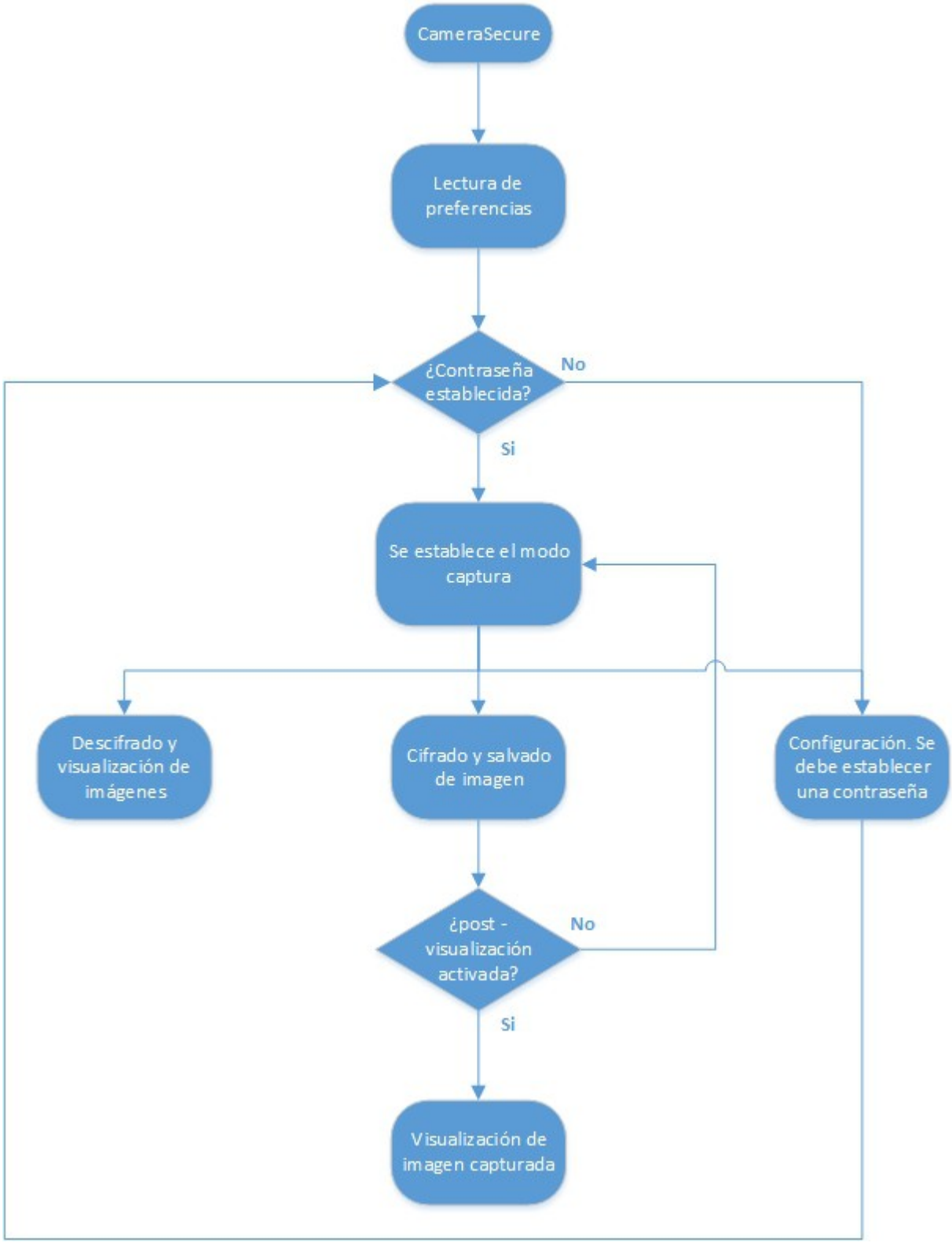

Figura 10

Análisis de seguridad

El proceso de cifrado de imágenes es llevado a cabo mediante los siguientes pasos:

- Captura de imagen.
- Generación del nombre de la imagen a guardar. El nombre utilizado sigue un formato temporal relacionado con la fecha.
- Selección del directorio donde se guardará la imagen. Las imágenes son guardadas dentro de la ruta *DCIM/Secure images* situada en la memoria externa del teléfono.
- Creación de *Bitmap* a partir del *array* de bytes que se obtiene tras capturar una imagen.
- Redimensionado del *Bitmap* a las dimensiones configuradas.
- Obtención de los píxeles que conforman la imagen.
- Cifrado de los píxeles obtenidos mediante el algoritmo *RC4* y la contraseña configurada en las preferencias.
- Establecimiento de los píxeles cifrados a la nueva imagen.
- Almacenado de la imagen cifrada.

Teniendo en cuenta estos aspectos,vamos a destacar las siguientes características de la aplicación desde el punto de vista de la seguridad.

- Las imágenes capturadas nunca se guardan a disco sin cifrar. Esta característica hace imposible la recuperación de las imágenes en claro a partir del disco. No obstante sí que al capturar una imagen ésta permanece sin cifrar durante un breve periodo de tiempo en la memoria *RAM*, pero esto no representa ninguna brecha en la seguridad ya que la memoria *RAM* es sobrescrita continuamente.
- Cifrado. Se está utilizando el cifrado *RC4*, que se basa en el **cifrado de flujo** con clave **simétrica**. La elección del cifrado ha venido condicionada por la imposibilidad de aplicar *AES*, el algoritmo inicialmente propuesto. La diferencia principal entre ambos algoritmos de cifrado es que el primero, *RC4* se basa en el cifrado flujo, mientras que *AES* se basa en el cifrado por bloques.
- Contraseña. Se almacena un *Hash* de la contraseña calculado con el algoritmo *SHA-256*. No se conocen graves vulnerabilidades para este algoritmo.
- Los problemas encontrados al aplicar algoritmos criptográficos basados en el cifrado por bloques vienen dados por las dificultades al realizar el volcado de las imágenes desde sus contenedores, objetos *Bitmap*, hacía fichero. Es decir con la *API* que ofrece Android para guardar un objeto de tipo *Bitmap*, se está utilizado la función *compress*.
  Esta función permite guardar la imagen a disco, pero durante el

volcado a fichero pueden producirse pérdidas de calidad en la imagen lo cuál deriva en que los bytes que se escribieron al objeto *Bitmap*, no son exactamente los que se escriben a disco.
Con un cifrado como *AES* basado en bloques, una pequeña diferencia en cualquiera de los bloques a descifrar provoca un error que hace inviable el descifrado, sin embargo al utilizar cifrados basados en flujo, este problema desaparece es por ello la elección del sistema de cifrado *RC4*.

Vulnerabilidades del protocolo RC4.

Existen dos vulnerabilidades conocidas en este protocolo.
Weak keys.- Las claves débiles son aquellas que para un determinado protocolo de cifrado hace que ciertas funciones del mismo funcionen de manera no deseada reduciendo la fortaleza del mismo. Por ejemplo en algunos algoritmos se ha detectado que con claves débiles se obtenía el mismo resultado al cifrar que al descifrar un mismo secreto. El algoritmo *RC4* adolece de este problema y se ha detectado que presenta claves débiles.
The known-plaintext attack (KPA).- Es un modo de ataque por el cual el atacante tiene muestras tanto del mensaje en texto plano como del mensaje cifrado. A partir de esa base el ataque permite deducir la contraseña de cifrado. Aunque la aplicación, *CameraSecure*, utiliza el cifrado *RC4*, los problemas derivados de este tipo de ataque no deben suponer ninguna criticidad ya que para llevarlo acabo se requerirían muestras tanto de las imágenes cifradas como de las sin cifrar.

Implementación de los protocolos criptográficos.

Con el fin de separar la funcionalidad relacionada con los métodos criptográficos usados tanto en el cifrado de imágenes como en el cifrado de la contraseña se definen las siguientes clases que permiten trabajar con los algoritmos criptográficos de forma sencilla.

RC4Crypt

Esta clase implemente el algoritmo de cifrado *RC4*, y presenta dos funciones principales para el cifrado y descifrado de las imágenes:

```
public byte[] encrypt(byte[] key, byte[] data) throws Exception {
          SecretKeySpec secretKey = new SecretKeySpec(key, "ARC4");
          Cipher cipher = Cipher.getInstance("ARC4", "BC");
          byte[] encryptedData = new byte[data.length];
          cipher.init(Cipher.ENCRYPT_MODE, secretKey);
          int ctLength = cipher.update(data, 0, data.length, encryptedData, 0);
          ctLength += cipher.doFinal(encryptedData, ctLength);
            return encryptedData;
      }
```

```
public byte[] decrypt(byte[] key, byte[] data) throws Exception {

          byte[] decryptedData = new byte[data.length];
          Cipher cipher = Cipher.getInstance("ARC4", "BC");
          cipher.init(Cipher.DECRYPT_MODE, new SecretKeySpec(key, "ARC4"));
          int ptLength = cipher.update(data, 0, data.length, decryptedData, 0);
          ptLength += cipher.doFinal(decryptedData, ptLength);
            return decryptedData;
      }
```

SHACrypt

Realiza el cálculo del hash de una cadena de caracteres. Para ello ofrece la función *computeHash* que a partir de un *String* devuelve el hash del mismo utilizando para su cálculo el algoritmo *SHA-2*. En concreto se está utilizando la versión *SHA-256* que se caracteriza por utilizar una salida de 256 bits.

```
public static String computeHash(String input) throws NoSuchAlgorithmException,
UnsupportedEncodingException {
          MessageDigest digest = MessageDigest.getInstance("SHA-256");
          digest.reset();
          byte[] byteData = digest.digest(input.getBytes("UTF-8"));
          StringBuffer sb = new StringBuffer();
          for (int i = 0; i < byteData.length; i++){
            sb.append(Integer.toString((byteData[i] & 0xff) + 0x100, 16).substring(1));
          }
          return sb.toString();
      }
```

Conclusiones

La aplicación desarrollada representa un avance en la protección de las privacidad de las personas. Ha sido colgada en Google Play y puede encontrase en la siguiente dirección:

https://play.google.com/store/apps/details?id=com.cs.camerasecure

En cuanto a los posibles usos parece claro que el principal es el de almacenar de forma segura las fotografías de carácter personal o privado de las personas. No obstante según voy recibiendo comentarios, me doy cuenta de las inmensas posibilidades que puede llegar a tener. A modo de ejemplo otro posible uso sería el de usar *CameraSecure* para guardar tarjetas de coordenadas o usarla como almacén de contraseñas. Se muestra el ejemplo en la Figura 11.

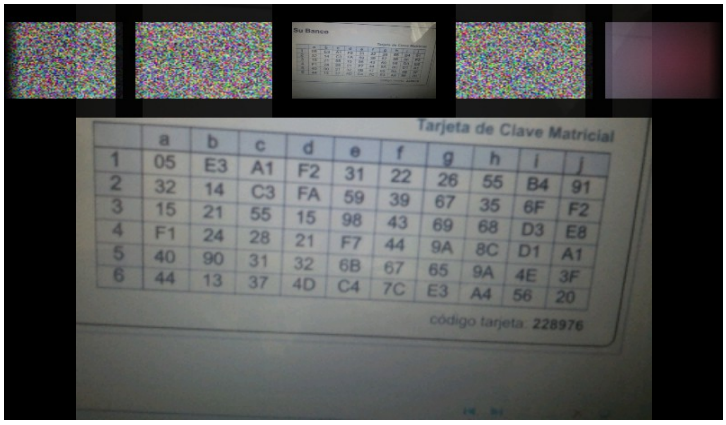

Figura 11

La aplicación está traducida a cuatro idiomas diferentes; inglés, castellano, francés y árabe, no obstante es mi intención la de continuar incorporando más. Puede verse una captura el idioma árabe en la Figura 12.

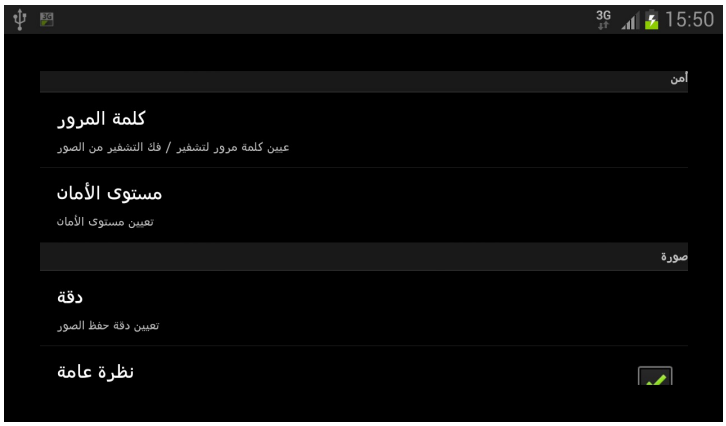

Figura 12

Como posibles mejoras quedaría la opción de hacer más social la aplicación para ello se deberían permitir compartir las imágenes entre grupo de usuarios. Esto añade un nivel de complejidad más a la aplicación ya que entran en juego temas de claves compartidas. No descarto añadir esta funcionalidad en el futuro.

En general creo que *CameraSecure* es una aplicación que cumple su cometido, y que se le pueden añadir fácilmente nuevas funcionalidades abriendo un mundo de nuevas posibilidades en cuanto a su uso.

Referencias

[1] Página principal de desarrolladores Android [online]. URL: http://developer.android.com/

[2] Página de referencia de la API Android [online]. URL: http://developer.android.com/reference/packages.html

[3] Documentación de la clase *Bitmap* [online]. URL: http://developer.android.com/reference/android/graphics/Bitmap.html

[4] Documentación de la clase *Camera* [online]. URL: http://developer.android.com/reference/android/graphics/Camera.html

[5] Encryption error on Android 4.2 [online]. URL: http://stackoverflow.com/questions/13383006/encryption-error-on-android-4-2

[6] Compute SHA256 Hash in Android/Java and C# [online]. URL: http://stackoverflow.com/questions/9661008/compute-sha256-hash-in-android-java-and-c-sharp

[7] Stream Cipher en Wikipedia [online]. URL: http://en.wikipedia.org/wiki/Stream_cipher

[8] RC4 en Wikipedia [online]. URL: http://en.wikipedia.org/wiki/RC4

[9] AES en Wikipedia [online]. URL: http://es.wikipedia.org/wiki/Advanced_Encryption_Standard

[10] Claves débiles en Wikipedia [online]. URL: http://en.wikipedia.org/wiki/Weak_key

Ataque de texto plano en Wikipedia [online]. URL:
[11] http://en.wikipedia.org/wiki/Known-plaintext_attack

[12] SHA-2 en Wikipedia [online]. URL: http://en.wikipedia.org/wiki/SHA-2