



Entrega N° 2 Proyecto

Seguridad Informática

AUDITORIA DE LA APLICACIÓN Y
DECOMPILACIÓN

Camilo Zepeda Hoffmann

Profesor Maximiliano Vega

12 de Octubre 2017

1. Resumen

Ya en conocimiento de la relevancia y puntos importantes de lo que implica realizar una auditoría en todo tipo de sistema, se sostiene la misma ideología basada en la decompilación y comprobación respectiva a la seguridad de la aplicación escogida. En este caso al tratarse de Instagram; sistema que actualmente es delegado por el equipo de Facebook, se considera generar pruebas asociadas a las debilidades, exploits, fallas o no de la aplicación. Es por ésta razón que se propone dar a conocer los pasos efectuados para la visualización de los componentes asociados en conjunto con herramientas que faciliten la depuración de la aplicación, dejando así, la propuesta correspondiente a los futuros incrementales.

2. Data Histórica de Vulnerabilidades

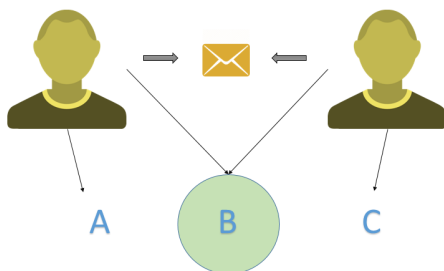
A continuación se da a conocer un compilado de problemas relacionados al software y plataforma de Instagram, luego de la compra de la misma. Ahora bien, se considera éste periodo como relevante al ser las exposiciones más importante y connotadas de filtración de datos.

2.1. Vulnerabilidad en OAuth 2013

Tras la compra de la Plataforma el año 2012 por Facebook, se logró encontrar fallas relacionadas a la privacidad, confidencialidad y filtración de datos, comprometiendo a los usuarios del software al existir errores del software ligados al protocolo OAuth, el cuál, realiza el control de autorización a terceros para acceder a la información, dando permisos bajo el servidor y un determinado cliente. Dicho problema abrió paso a problemas como el acceso a la lista de amigos en Facebook o filtración de imágenes ^[1].



2.2. Cuentas Compartidas 2016



Problema relacionado al software en Sistema Android encontrado el 2016, el cual tiene como repercusión la vista de información no necesariamente del propietario, lo que implica un error en cuánto a la privacidad y confidencialidad de los datos. Entonces ésta vulnerabilidad consiste en la compartición de cuentas por parte de un Sujeto 1 v's un Sujeto 2, donde ambos ya sea por motivos de asociación u otros poseen una cuenta en común B, dando paso a la notificación de cuentas fuera de su alcance, ya

que si por su parte el Sujeto 1 es propietario de la cuenta A y el Sujeto 2 es dueño de la cuenta B, se logra ver información desde A en C, aún sin ser propietario o usuario principal de ella [2]. Por tanto, se implica una filtración de información fuera del consentimiento de cada sujeto.

2.3. Hackeo de Cuentas 2017

Actualmente la plataforma cuenta con un gran número de métodos y propiedades que mitigan las vulnerabilidades, aún bajo ésta tesis existen filtraciones de datos basados en ingeniería social o posibles errores de la plataforma, lo que el pasado mes de agosto del presente año se hizo de gran popularidad, logrando un hackeo masivo de cuentas, comprometiendo la seguridad de los usuarios en redes sociales. Llegando a circular datos de acceso a los perfiles de hasta \$10 dolares cada cuenta. La información fue almacenada en la base de datos **Doxagram** [3] en donde se realiza la venta y comercialización de la información.



3. Análisis de la Aplicación

Previo a la obtención y análisis de la aplicación dado los datos a obtener se propone definir los requerimientos base que conforman la actividad realizada. El listado de Herramientas utilizadas es:

- **Página de Descarga:** se utiliza una página que disponga de la app.
- **Lionsec:** dado que es necesario decompilar la aplicación, se utiliza el nombrado sistema operativo, debido a sus multiples funciones y herramientas que faciliten el trabajo.
- **Apktool:** herramienta que realiza la lectura y depuración de la aplicación.
- **Dex2jar:** herramienta utilizada para convertir los diferentes .dex propios de android a .jar.
- **Java Decompiler:** utilizado para lograr depurar los archivos .jar.

3.1. Obtención de la Aplicación

En primera instancia es necesario conocer que la aplicación se encuentra disponible tanto para sistemas operativos iOS como Android, donde si bien es posible descargar ambos formatos no es posible decompilar los dos por igual. Cabe aclarar que se refiere a decompilar, como la obtención de toda la data proveniente de la aplicación. Entonces si bien, iOS dispone de los layouts o vistas, las cuales pueden ser editadas no se logra editar y ver los códigos base de la app.



Entonces en concordancia con lo anterior, se realiza un análisis de la aplicación **.apk** (Sistema Operativo Android) al ser más amigable al momento de depurar, entregando una serie de archivos y datos correspondientes a la plataforma. Para lograr la obtención de la aplicación se hace uso de **ApkPure** página que dispone de la última versión de Instagram descargable para el computador. Ya en obtención de la aplicación y bajo el uso de

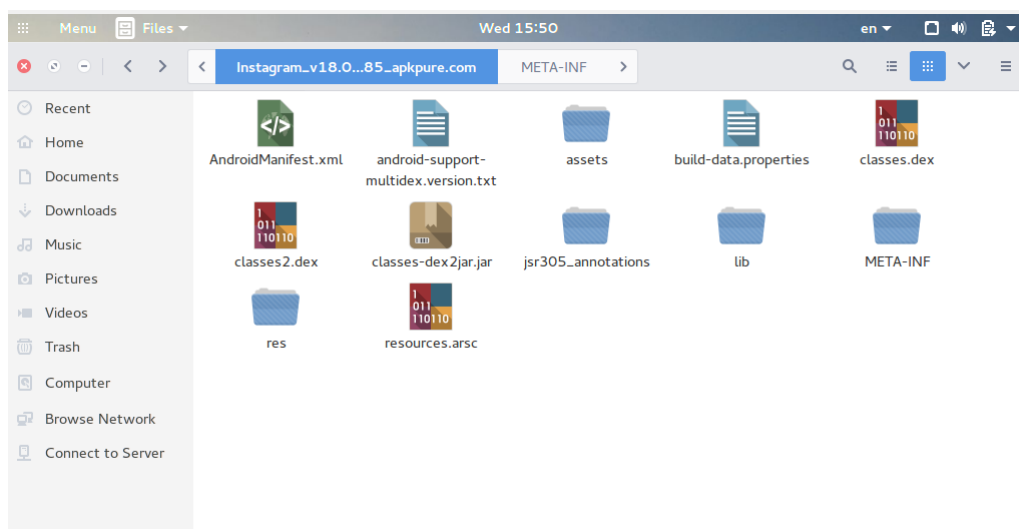
las herramientas mencionadas anteriormente se procede a la depuración de la app.

3.2. Decompilación de la Aplicación

Con tal de lograr un mejor entendimiento de lo realizado se describe una serie de pasos de la mano con el material obtenido, en formato de imagen o similar.

- **Paso 1:** ya en posesión de la .apk correspondiente a la aplicación la primera actividad fue el transformar el archivo de:

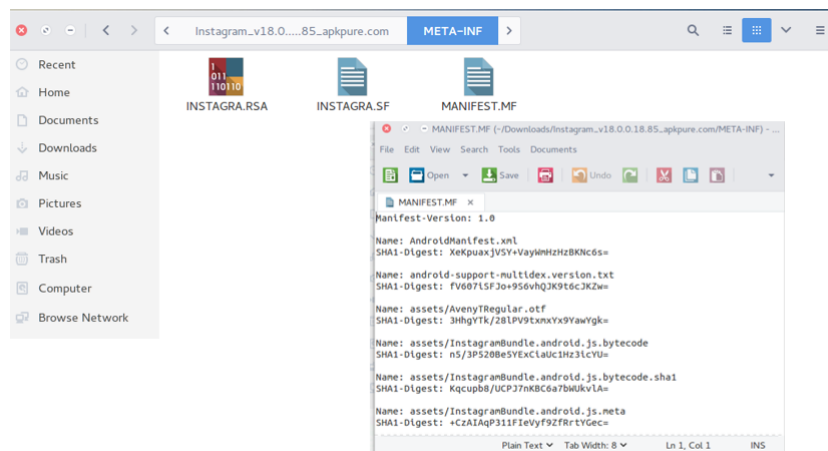
Instagram.apk => Instagram.zip



De la conversión anterior se obtiene acceso a los archivos en bruto de la aplicación, así no, no es posible observar los xml, java o react de la misma, ya que aún no se ha depurado a cabalidad. Igualmente, se obtienen los **.dex** equivalente a las clases utilizadas por la app.

- **Paso 2:** ya extraídos los archivos anteriormente mencionados, se puede observar la carpeta **META-INF** equivalente al certificado de los desarrolladores de la aplicación, la cual

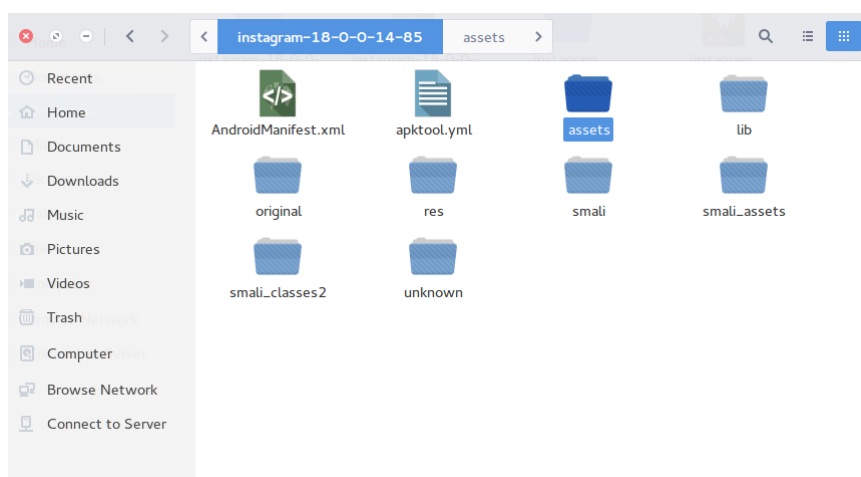
al revisar los datos internos se determina que se usa Hashing Sha1 para el certificado de desarrollador y cifrado RSA, para el certificado entregado por Android. Lo dicho puede verse en la siguiente figura:



- **Paso 3:** Una vez realizado los pasos anteriores fue posible determinar que no se pueden ver tanto los xml, como los java provenientes de la aplicación. Es por esta razón que se hace uso de software **ApkTool** ya interno en Lionsec. Ya familiarizado con la herramienta se realiza el siguiente comando:

```
apktool d Instagram.apk
```

Donde se define **d** como la lectura del archivo, obteniendo entonces los siguientes archivos:



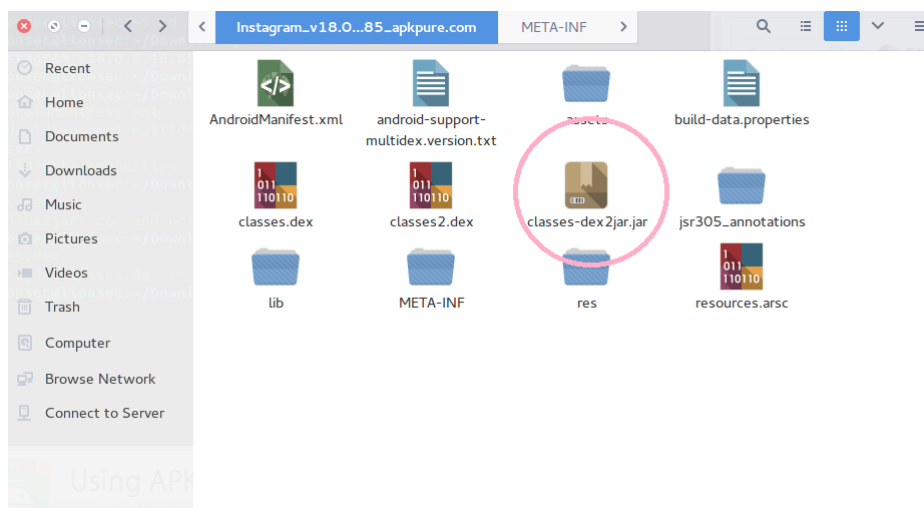
- **Paso 4:** Finalmente tras disponer de la información y datos pertinentes a la depuración de la aplicación. Es necesario revisar los datos obtenidos e información relevante. Los primeros datos revisados fueron los XmlAndroid, los cuales definen parte de los requerimientos y facultades que asocia la aplicación para su desempeño. La siguiente figura da una perspectiva de lo visto.

```
anifest android:installLocation="auto" package="com.instagram.android">
uses-permission android:name="android.permission.INTERNET"/>
uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
uses-permission android:name="android.permission.WAKE_LOCK"/>
uses-permission android:name="android.permission.GET_ACCOUNTS"/>
uses-permission android:name="android.permission.USE_CREDENTIALS"/>
uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
uses-permission android:name="android.permission.VIBRATE"/>
uses-permission android:name="android.permission.CAMERA"/>
uses-permission android:name="android.permission.READ_CONTACTS"/>
uses-permission android:name="android.permission.READ_PROFILE"/>
uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
uses-permission android:name="android.permission.RECORD_AUDIO"/>
uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
uses-permission android:name="android.permission.READ_PHONE_STATE"/>
uses-permission android:name="android.permission.RECEIVE_SMS"/>
uses-permission android:name="com.android.launcher.permission.INSTALL_SHORTCUT"/>
uses-permission android:name="com.android.launcher.permission.UNINSTALL_SHORTCUT"/>
permission android:name="permission.C2D_MESSAGE" android:protectionLevel="signature"/>
permission android:name="permission.RECEIVE_ADM_MESSAGE" android:protectionLevel="signature"/>
uses-permission android:name="permission.C2D_MESSAGE"/>
```

Si bien se puede encontrar una serie de disposiciones y permisos relacionados a la aplicación, no se considera un punto relevante para la depuración, ya que son puntos relativamente comunes para la aplicación, dígase, no se encuentra anomalías que puedan ser explotables en futuros incrementales.

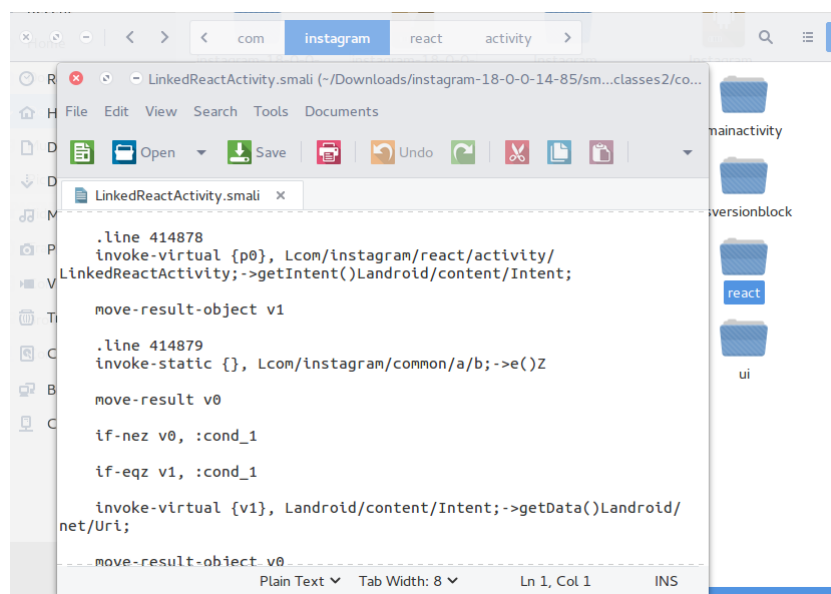
Por otra parte, mediante el uso de la herramienta dex2jar, se realiza una conversión de las clases intrínsecas en la aplicación, mediante el comando:

```
d2j-dex2jar classes.dex
```



Lo que convierte los archivos a un formato .jar, el que por medio de una Java Decompiler y jd-gui interno del mismo, puede lograrse un análisis de los archivos java desarrollados en la aplicación.

Por último se realiza una revisión de los archivos React, los cuales se encuentran en formato **.smali** propio de Java. Obteniendo entonces lo siguiente:



```
.line 414878
invoke-virtual {p0}, Lcom/instagram/react/activity/
LinkedReactActivity;->getIntent()Landroid/content/Intent;

move-result-object v1

.line 414879
invoke-static {}, Lcom/instagram/common/a/b;->e()Z

move-result v0

if-nez v0, :cond_1

if-eqz v1, :cond_1

invoke-virtual {v1}, Landroid/content/Intent;->getData()Landroid/
net/Uri;

move-result-object v0
```

Cabe destacar que los archivos vistos en la figura anterior tienen una real validez en cuanto a vulnerabilidades explotables para la auditoría a realizar, ya que dichas opciones son las actividades, acciones y funciones propias de la aplicación, dando apertura a protocolos, urls, traspaso de información y otras actividades que comprometen la aplicación.

3.3. Propuestas para Tercer Incremento

Con el fin de lograr una mayor comprensión de lo propuesto se define el siguiente listado.

- Analizar en profundidad las funciones Internas React, depurando la información.
- Buscar vulnerabilidades correspondientes a la página de Instagram (inyecciones SQL).
- Buscar posibles Scripts utilizados para robar información.
- Analizar Servicios utilizados por la aplicación.
- Depurar en profundidad los archivos pertinentes de la app.
- Instalar la aplicación, con tal de analizar la BD interna generada por la misma.

Referencias

- [1] A. Plaza, "Un fallo de seguridad en Instagram muestra la vulnerabilidad de las cuentas | Hipertextual", Hipertextual, 2017. [Online]. Available: <https://hipertextual.com/2013/05/fallo-de-seguridad-en-instagram>. [Accessed: 11- Oct- 2017].
- [2] "La posibilidad de usar varias cuentas en Instagram presenta un fallo de seguridad - TreceBits", TreceBits, 2017. [Online]. Available: <http://www.trecebits.com/2016/02/16/la-posibilidad-de-usar-varias-cuentas-en-instagram-presenta-un-fallo-de-seguridad/>. [Accessed: 11- Oct- 2017].
- [3] "El hackeo a Instagram, más grave de lo esperado: 6 millones de cuentas", ADSLZone, 2017. [Online]. Available: <https://www.adslzone.net/2017/09/02/el-hackeo-instagram-mas-grave-de-lo-esperado-6-millones-de-cuentas/>. [Accessed: 11- Oct- 2017].
- [4] <https://www.buzztouch.com/forum/thread.php?tid=B60C6EE41CD7D289048B0F4>
- [5] <https://github.com/arashpayan/appirater>
- [6] <https://iphoneros.com/44273/asi-es-el-modo-reachability-para-poder-utilizar-el-iphone-6-plus-con-una-sola-mano-video>
- [7] <https://curl.haxx.se/docs/history.html>