



UNIVERSIDAD
Finis Terrae

VINCE IN BONO MALUM

Proyecto Seguridad informática: Auditoria a Twitter

Jorge Gonzalez
Profesor: Maximiliano Vega
jgonzalezl@uft.edu
Santiago, Chile.

3 de noviembre de 2017

Índice

1. ¿Que es Twitter?	3
1.1. Historia de Twitter	3
1.2. Tecnología usada	3
2. tipos de ataques	4
2.1. Medidas de protección	4
2.2. Revisiones a realizar	5
3. Segundo incremetal	6
3.1. Investigacion previa	6
3.2. Desencriptado de la aplicación	7
3.3. Escaneo de puertos	9
4. que son las distribuciones basadas en Linux para auditorias	10
4.1. Decompilación de la aplicación con apktool	11
5. Cuarto incremental	13
5.1. Que es SQLmap	13

1. ¿Qué es Twitter?

1.1. Historia de Twitter

Twitter es un servicio gratuito de microblogging, que hace las veces de red social y que permite a sus usuarios enviar micro-entradas basadas en texto, denominadas "tweets", de una longitud máxima de 140 caracteres. El 21 de marzo del 2006, fecha del primer tweet lanzado por su fundador Jack Dorsey. En sus comienzos, la idea de Twitter surgió como proyecto de investigación dentro de Obvious, una pequeña compañía situada en San Francisco.



Trending Topics

la red social amplió su ecosistema con los ya famosos "Trending Topics" o Temas del Momento. El 30 de abril del 2009, Twitter hacía oficial un cambio en su barra de búsquedas. Se trataba de fomentar aquello que originaba más "ruido", los temas que más se repetían entre el flujo de tweets veían como accedían a una categoría mayor, de manera que todos los usuarios podían reconocer o seguir los temas más candentes. Evidentemente y con el paso del tiempo, los Trending Topics han evolucionado, desde el año pasado se añaden también Temas del Momento promocionados.

Hashtag

Otra de las claves de Twitter fue la inclusión de Hashtags, una etiqueta que bajo el símbolo de hash (#) seguida de una palabra o varias concatenadas, permitía realizar un seguimiento de temas a los usuarios.

1.2. Tecnología usada

Twitter está escrita en Ruby on Rails, es un framework de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby, la interfaz de Twitter es tremadamente sencilla. Seguimos y nos siguen. Los usuarios que seguimos aparecen en el timeline y viceversa, lo que comúnmente se conoce en lenguaje "twitero" como followers y followings.

Todos los mensajes de Twitter van a parar al servidor de un software programado en Scala, además, la API está totalmente abierta a desarrolladores, motivo por el cual Twitter puede integrarse en todo tipo de webs, blogs o dispositivos móviles. Twitter confirmó en 2010 una de las actualizaciones más demandadas, incluyendo por fin la posibilidad de poder ver fotos, vídeos y contenido que llegaba de otros alojamientos. Además, desde la semana pasada la compañía añadió la opción que te permite utilizar siempre HTTPS cuando accedes a Twitter.com con el fin de mejorar la seguridad.

2. tipos de ataques

de los que se vieron online existen ataques que dicen funcionar como por ejemplo con exploits como los que se ven a continuacion:



Figura 1: ingreso a efinis con host modificado.

como se puede ver en su mayor parte se realizan acciones con exploits para saber que es un exploit se debe revisar la siguiente definicion: Las definiciones habituales hablan de un programa o código que se aprovecha de un agujero de seguridad (vulnerabilidad) en una aplicación o sistema, de forma que un atacante podría usarla en su beneficio.

2.1. Medidas de protección

- Mantener todas nuestras aplicaciones y sistemas actualizados: sabiendo que los exploits se aprovechan de los agujeros de seguridad, resulta vital cerrarlos cuanto antes. Por eso es necesario mantener una política de actualizaciones eficaz para evitar dejar una ventana de tiempo que pueda ser aprovechada por los atacantes.
- Mitigar los efectos de posibles exploits usados en nuestra contra. Puede suceder que el fabricante del sistema o aplicación vulnerable no haya lanzado todavía una actualización que solucione el problema. En este caso, se pueden utilizar herramientas como el Kit de herramientas de Experiencia de Mitigación mejorada (EMET) para Windows. Esto ayudará a evitar que tu sistema se infecte hasta que aparezca una solución definitiva.
- Contar con una solución de seguridad avanzada como ESET Smart Security, capaz de detectar y bloquear exploits pensados para aprovechar vulnerabilidades en navegadores web y lectores PDF, entre otros.

2.2. Revisiones a realizar

- Análisis dinámico.
- Comunicación con el servidor
- Estudio de la estructura de protección del almacenamiento de datos.
- Decomplilación de la aplicación.
- Revisión del código fuente.

3. Segundo incremental

3.1. Investigacion previa

Lo primero que se hizo fue probar con los supuestos exploits que se tenía en internet aquí un ejemplo del resultado:

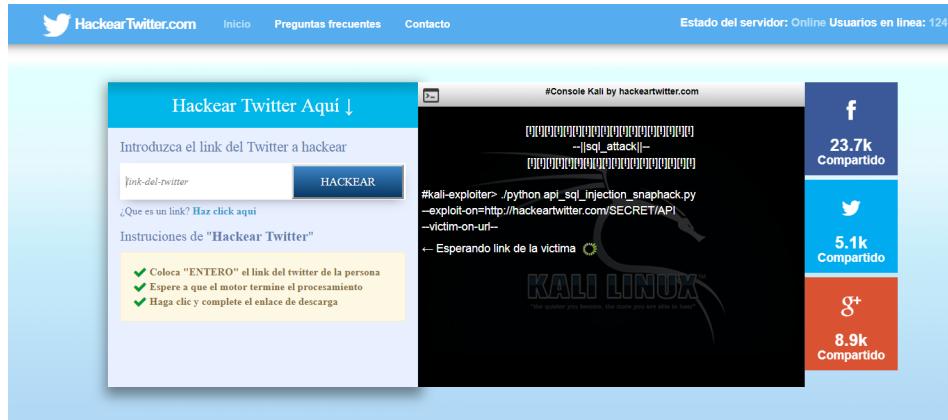


Figura 2:

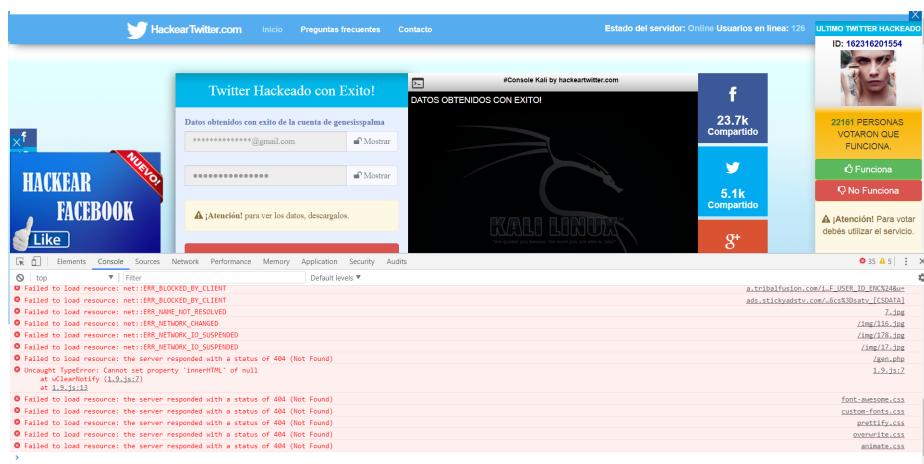


Figura 3:

Como es posible notar al inspeccionar la página se ve que la prueba solo era un “cazas bobos” ya que en el intento de conectarse con un socket solo lograba la falta de respuesta. Luego de verificar esto se intentó ver a qué resultado se había llegado, pero para eso pedía un pago, lo que dejaba más en claro que solo se deseaba hacer perder el tiempo y dinero a quienes lo intentaban, luego de esto ya no se intentó más tratar de encontrar alguna falla con los similares a estos ya que se presumió que todos o la gran mayoría de ellos eran similares y serían perdidas de tiempo.

3.2. Desencriptado de la aplicación

Para poder desencriptar la aplicación se utilizó el sistema operativo lionsec y las herramientas APKtool en concreto lo que se muestra en la siguiente imagen:

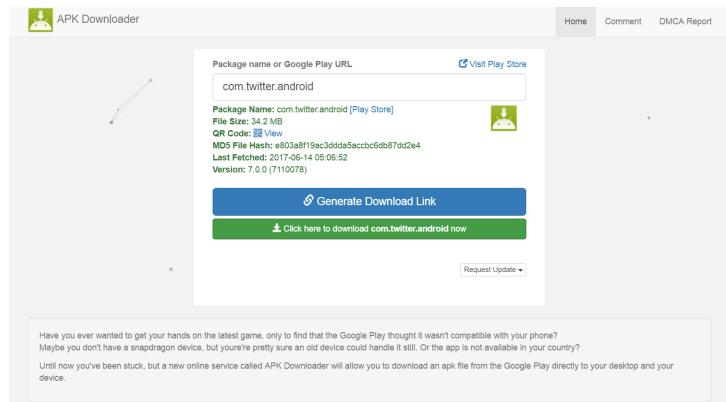


Figura 4:

Ya con la aplicación descargada se pasó a lionsec para poder descompilarla con las herramientas de este como ve se en la imagen:

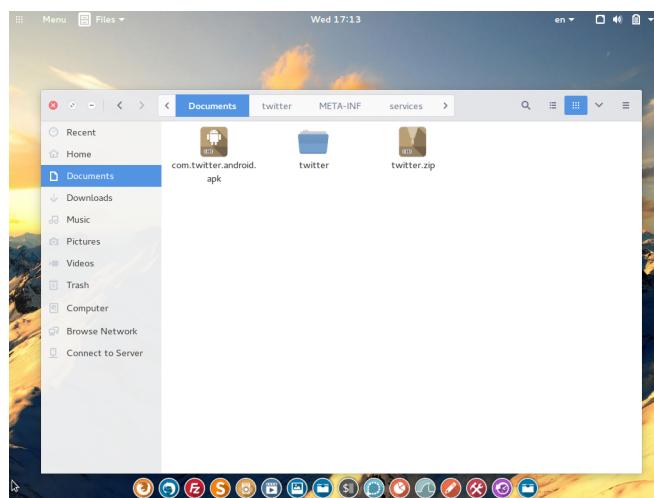


Figura 5:

Como se ve en la imagen después de descompilarla aplicación se creó una carpeta con algunos de los archivos que componen la aplicación para así lograr ver si se tiene alguna vulnerabilidad que se logre explotar. Como se muestra en la siguiente imagen lo único que se logró ver en la aplicación descompilada es el archivo META-INF el cual tiene el cifrado que se utiliza en la aplicación el cual es SHA-1 como se vio en el documento MANIFEST-MF que se ve a continuación:

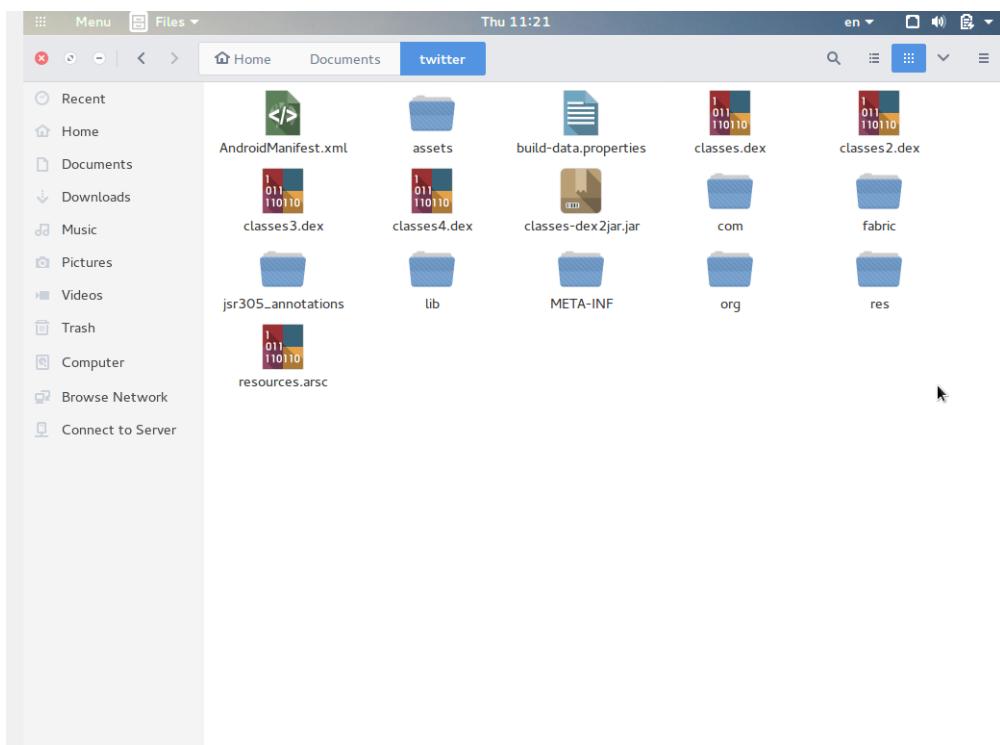


Figura 6:

Thu 22:14

MANIFEST.MF (~/Documents/twitter/META-INF) - gedit

```

File Edit View Search Tools Documents
  Open Save Undo Undo | Cut Copy Paste | Find Replace | Help

MANIFEST.MF x
Manifest-Version: 1.0
Built-By: Generated-by-ADT
Created-By: Android Gradle 2.3.0

Name: res/layout/totp_generator_ui.xml
SHA1-Digest: ETLCI65bgM0yG1NU8yqLnHNCnC

Name: res/drawable-anydpi-v21/vector_delete_icon.xml
SHA1-Digest: E+lKR1ADT2aN6eGU96gg1tVz0/I=

Name: res/layout/dm_rounded_conversation_suggestion_row_view.xml
SHA1-Digest: EL/MAVOXw2XhmlVh+hpSe5nide4=]

Name: res/layout/moments_guide_cell_item_thumbnail_icon.xml
SHA1-Digest: xT6se2B4A0zIMfk8oDOrtEG0iVC=

Name: res/layout/nativecards_consumerpoll_choices_row_moments.xml
SHA1-Digest: 4Imrr/b0D7xUbiz+0jJcfWRHfHs=

Name: res/drawable-hdpi-v4/ic_vector_medium_photo_stroke_tint.png
SHA1-Digest: 2wA85URwKaAvP2Ca3syHlMM/Hy8=

Name: res/drawable-xhdpi-v4/ic_tweet_monetize.png
SHA1-Digest: cmec9JUbeJ9rgp0Y7RJ3AOc4iIM=

Name: res/drawable-nodpi-v4/ic_compose_dm_102h.png
SHA1-Digest: kRFn8SCvaAkjhbaMxqUBJvvn9bM=

Name: res/drawable-nodpi-v4/ic_lightning_small_80h.png
SHA1-Digest: zGE+AcSLQFxrnA8sGRTRE15AoRg=

```

Plain Text ▾ Tab Width: 8 ▾ Ln 12, Col 42 INS

Figura 7:

3.3. Escaneo de puertos

Después de esto se escanearon los puertos con nmap para ver si existía algún puerto descuidado o alguna otra falla lo cual solo se encontró los siguiente:

```

lionsec@lionsec:~$ nmap -F twitter.com\
>

Starting Nmap 7.01 ( https://nmap.org ) at 2017-10-11 17:41 CLT
Stats: 0:00:03 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Stats: 0:00:04 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Stats: 0:00:05 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done

lionsec@lionsec:~$ nmap -F twitter.com

Starting Nmap 7.01 ( https://nmap.org ) at 2017-10-11 17:41 CLT
Nmap scan report for twitter.com (104.244.42.193)
Host is up (0.27s latency).
Other addresses for twitter.com (not scanned): 104.244.42.129
Not shown: 98 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 29.25 seconds
lionsec@lionsec:~$ m

```

Figura 8:

de esto solo se puede inferir que Los puertos abiertos son solo 2 lo que se ven son :

- Puerto 80
- Puerto 443

Por lo que se investigo el Puerto 80 es el puerto por default, por el medio del cual un servidor HTTP “escucha” la petición hecha por un cliente, es decir por una PC en específico Y el puerto 443 seria el que da el indicio de que las comunicaciones se realizan con certificados TLS lo que seria otro punto mas a investigar.

4. que son las distribuciones basadas en Linux para auditorias

las distribuciones de Linux enfocados a las auditorias son, como lo dice su nombre variaciones del sistema operativo Linux que cuantas con una gran cantidad de herramientas y utilidades que son de gran ayuda a la hora de probar funcionalidades y seguridad de aplicaciones, sistemas, páginas web, sistemas web, etc...

estas distribuciones y recopilación de herramientas para la seguridad informática se pueden distribuir en tres categorías: distribuciones para penesting, para auditorias de seguridad y distribuciones seguras.

Las distribuciones más conocidas son:

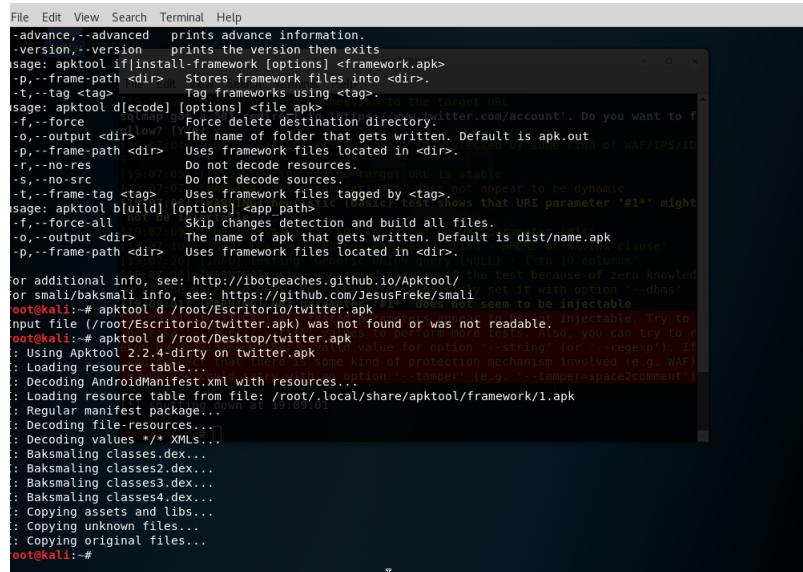
- Kali Linux
- Parrot Security OS
- Xiaopan OS
- WifiSlax
- BackBox
- Samurai Web Testing Framework
- DEFT Linux
- Caine
- Nst
- Santoku Linux
- BlackArch
- Bugtraq

Entre otros, es importante tenerlos en cuenta ya que en muchos casos estas distribuciones son esécializadas en su ámbito lo que las hace muy útiles en el momento que se necesite una distribución para los distintos casos.

En esta ocasión se utilizaron algunas de las herramientas de la distribución de Linux kali, esta distribución está destinada principalmente para la auditoria de aplicaciones en distintos ámbitos. Para probar algunos de los puntos importantes de la aplicación se utilizaron dos herramientas de este sistema: APKtool y SQLmap.

4.1. Descompilación de la aplicación con apktool

con la aplicación ya en el sistema se procedió a descompilarla como se ve en la imagen:



```

File Edit View Search Terminal Help
advanced...advanced prints advance information.
--version,--version prints the version then exits
usage: apktool if|install-framework [options] <framework.apk>
-p,--frame-path <dir> Stores framework files into <dir>
-t,--tag <tag> Tag frameworks using <tag>
usage: apktool d[ecode] [options] <file.apk> ... -o target_dir
-f,--force Force delete destination directory
-o,--output <dir> The name of folder that gets written. Default is apk.out
-p,--frame-path <dir> Uses framework files located in <dir>
-r,--no-res Do not decode resources.
-s,--no-src Do not decode sources.
-t,--frame-tag <tag> Uses framework files tagged by <tag>
usage: apktool b[uild] [options] <app_path> ... -o target_dir
-f,--force-all Skip changes detection and build all files.
-o,--output <dir> The name of apk that gets written. Default is dist/name.apk
-p,--frame-path <dir> Uses framework files located in <dir>
usage: apktool d[ecode] [options] <apk> ... -o target_dir
-f,--force Force delete destination directory
-o,--output <dir> Set it with option --force. Set it with option --force
-p,--frame-path <dir> Set it with option --force
-r,--no-res Do not decode resources.
-s,--no-src Do not decode sources.
-t,--frame-tag <tag> Set it with option --force
usage: apktool b[uild] [options] <app_path> ... -o target_dir
-f,--force-all Skip changes detection and build all files.
-o,--output <dir> Set it with option --force
-p,--frame-path <dir> Set it with option --force
or additional info, see: http://ibotpeaches.github.io/ApkTool/
for small/baksmali info, see: https://github.com/JesusFreke/small
root@kal:~# apktool d /root/Escritorio/twitter.apk
Input file (/root/Escritorio/twitter.apk) was not found or was not readable. injectable. Try to r
root@kal:~# apktool d /root/Desktop/twitter.apk
: Using Apktool 2.2.4-dirty on twitter.apk
: Loading resource table...
: Decoding AndroidManifest.xml with resources...
: Loading resource table from file: /root/.local/share/apktool/framework/1.apk
: Regular manifest package...
: Decoding file-resources...
: Decoding values */* XMLs...
: Baksmaling classes.dex...
: Baksmaling classes2.dex...
: Baksmaling classes3.dex...
: Baksmaling classes4.dex...
: Copying assets and libs...
: Copying unknown files...
: Copying original files...
root@kal:~#

```

Figura 9:

con la aplicación y descompilada se procedió a ver que archivos tenía y si había algunos datos de importancia dentro de esta misma y el archivo más relevante que se encontró fue el archivo Android manifest que es el archivo de configuración de la aplicación está situado en la raíz de esta misma y controla todos los permisos necesarios para que la aplicación funcione.

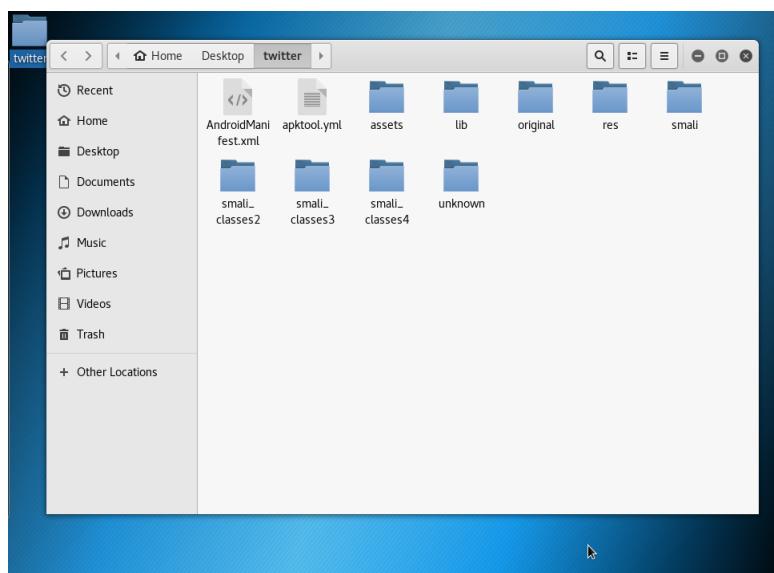
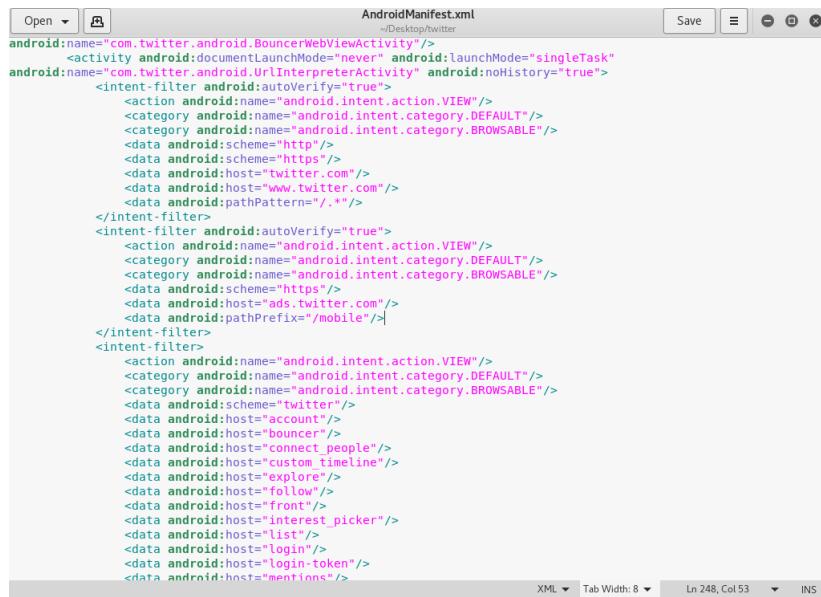


Figura 10:

En el Android manifest se encontraron todos los permisos de la aplicación, pero el dato más importante que se encontró fue el host de la aplicación y distintos tipos de tags con los cuales hacer consultas como se ve en la siguiente imagen:

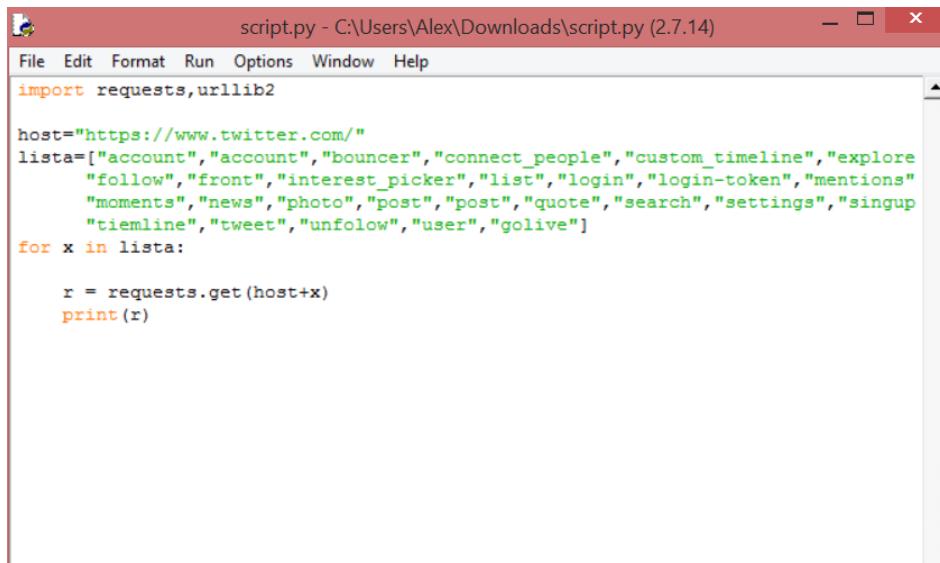


```

    Open [ ] AndroidManifest.xml ~Desktop\Twitter
    android:name=".com.twitter.android.BouncerWebViewActivity"/>
        <activity android:documentLaunchMode="never" android:launchMode="singleTask"
        android:name=".com.twitter.android.UrlInterpreterActivity" android:noHistory="true">
            <intent-filter android:autoVerify="true">
                <action android:name="android.intent.action.VIEW"/>
                <category android:name="android.intent.category.DEFAULT"/>
                <category android:name="android.intent.category.BROWSABLE"/>
                <data android:scheme="http"/>
                <data android:scheme="https"/>
                <data android:host="twitter.com"/>
                <data android:host="www.twitter.com"/>
                <data android:pathPattern="/.*/>
            </intent-filter>
            <intent-filter android:autoVerify="true">
                <action android:name="android.intent.action.VIEW"/>
                <category android:name="android.intent.category.DEFAULT"/>
                <category android:name="android.intent.category.BROWSABLE"/>
                <data android:scheme="https"/>
                <data android:host="ads.twitter.com"/>
                <data android:pathPrefix="/mobile"/>
            </intent-filter>
            <intent-filter>
                <action android:name="android.intent.action.VIEW"/>
                <category android:name="android.intent.category.DEFAULT"/>
                <category android:name="android.intent.category.BROWSABLE"/>
                <data android:scheme="twitter"/>
                <data android:host="account"/>
                <data android:host="bouncer"/>
                <data android:host="connect_people"/>
                <data android:host="custom_timeline"/>
                <data android:host="explore"/>
                <data android:host="follow"/>
                <data android:host="front"/>
                <data android:host="interest_picker"/>
                <data android:host="list"/>
                <data android:host="login"/>
                <data android:host="login-token"/>
                <data android:host="mentions"/>
            </intent-filter>
        
```

Figura 11:

Con estos datos se creo un script en Python junto con la librería request para generar una consulta sobre un URL tomando en consideración todos los urls que se obtuvieron dentro del manifest como se ve en la siguiente imagen:



```

script.py - C:\Users\Alex\Downloads\script.py (2.7.14)
File Edit Format Run Options Window Help
import requests,urllib2

host="https://www.twitter.com/"
lista=["account","account","bouncer","connect_people","custom_timeline","explore"
      "follow","front","interest_picker","list","login","login-token","mentions"
      "moments","news","photo","post","post","quote","search","settings","signup"
      "timeline","tweet","unfollow","user","golive"]
for x in lista:
    r = requests.get(host+x)
    print(r)

```

Figura 12:

Los resultados que se obtuvieron son respuestas 200 y 404 las cuales son OK para recibir la consulta y no se encuentra la pagina como se aprecia en la imagen:

```
===== RESTART: C:/Users/Alex/Downloads/script.py =====
<Response [404]>
<Response [404]>
<Response [200]>
<Response [200]>
<Response [404]>
<Response [200]>
<Response [404]>
<Response [200]>
<Response [200]>
<Response [404]>
<Response [200]>
<Response [200]>
<Response [404]>
<Response [200]>
<Response [200]>
<Response [404]>
<Response [404]>
<Response [200]>
<Response [200]>
<Response [200]>
<Response [404]>
<Response [404]>
<Response [200]>
<Response [200]>
<Response [200]>
<Response [404]>
<Response [404]>
<Response [200]>
>>>
```

Figura 13:

Luego de saber cuales eran las URLs con las que se podía trabajar se probaron los estas en SQLmap para lograr ver si se podía obtener algo, lo cual como se suponía tuvo resultados negativos.

5. Cuarto incremental

dentro de lo que se hizo en este incremental fue encontrar variadas herramientas para lograr realizar otras pruebas necesarias para verificar la seguridad de la aplicación en cuestión.

5.1. Que es SQLmap

SQLmap es una herramienta desarrollada en el lenguaje de programación Python la cual es muy útil para realizar pruebas automatizadas a bases de datos sobre la seguridad de las mismas. El objetivo de esta herramienta es detectar y utilizar vulnerabilidades de inyección sql en aplicaciones web, una vez detectado el host se pueden realizar pruebas de inyecciones sql y otras opciones de las cuales se pueden utilizar serian:

- Enumerar bases de datos
- Enumerar las tablas y columnas de una base de datos.
- Descifrar los hashes de contraseñas .

- Leer archivos específicos del sistema de archivo de un host específico

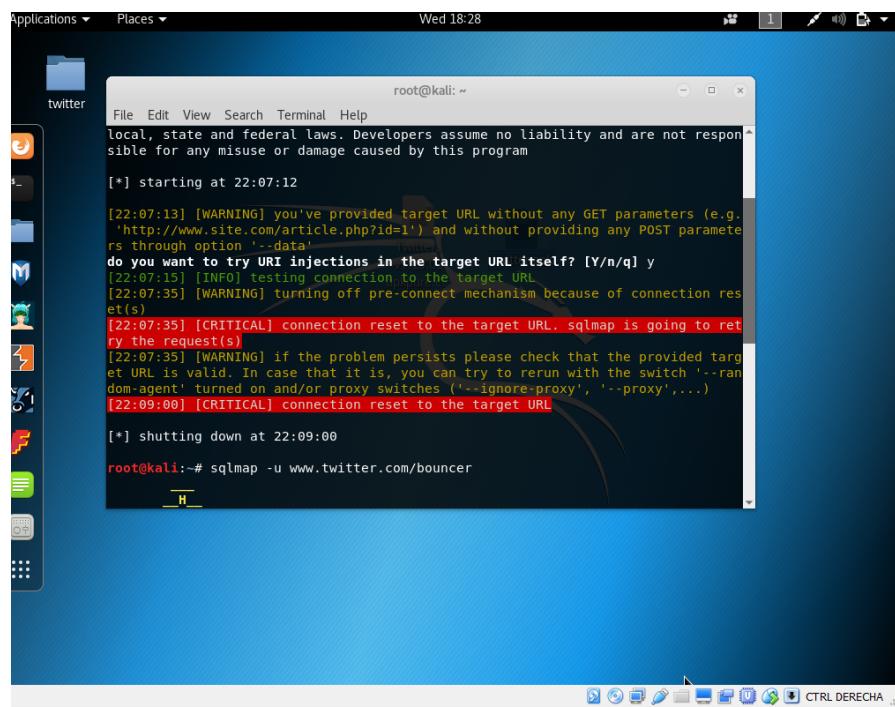


Figura 14:

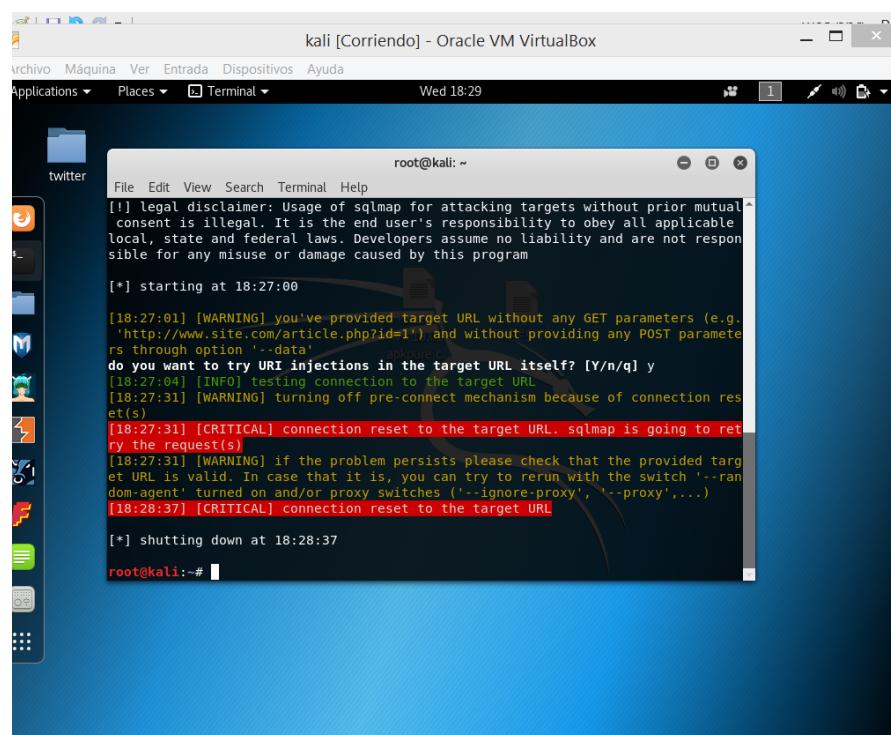


Figura 15:

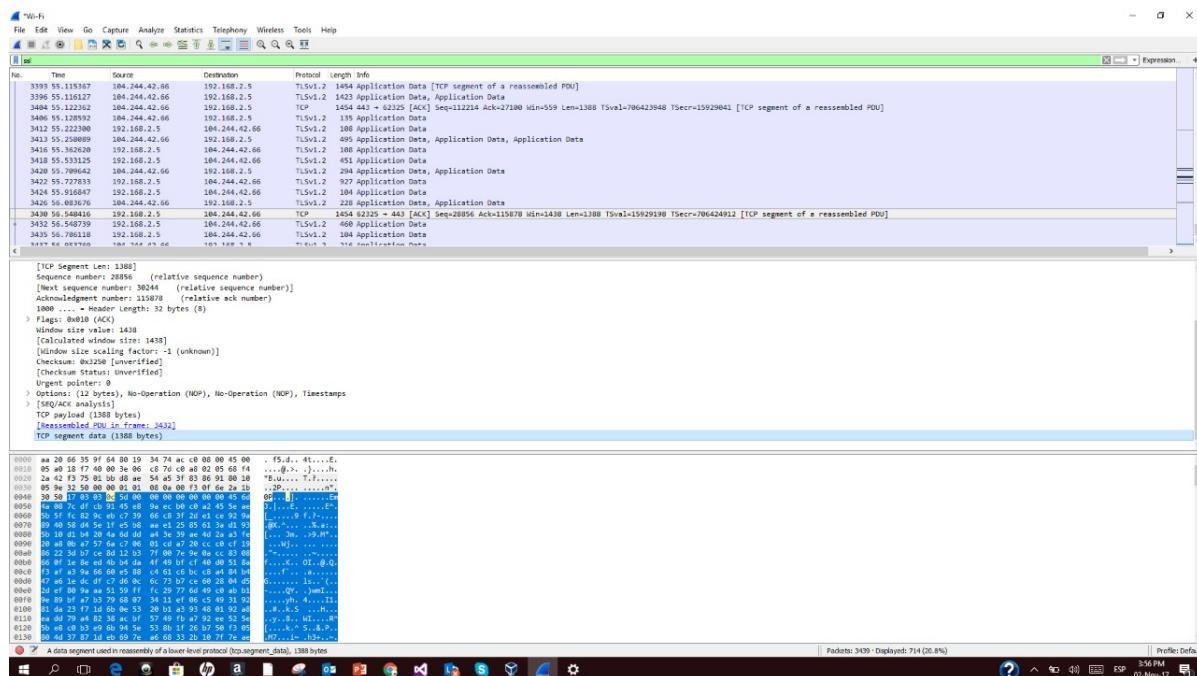


Figura 16:

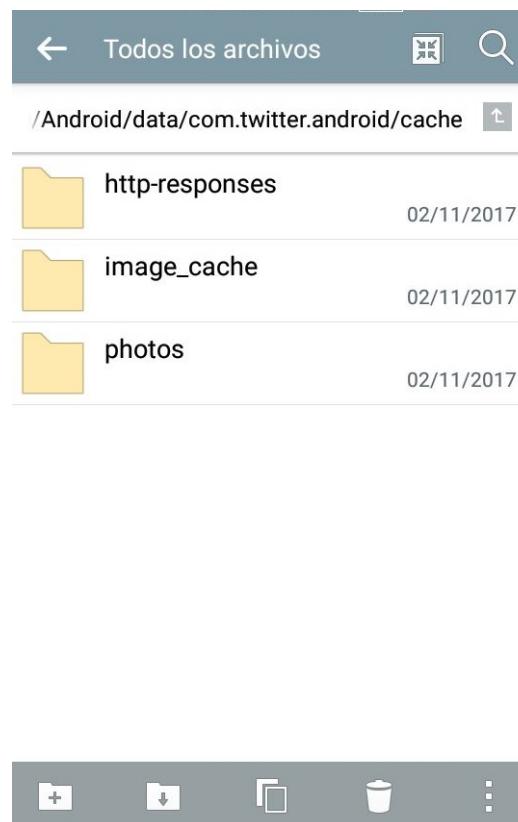


Figura 17:

Referencias

- [1] Hipertextual. (2017). Historia de Twitter. [online] Available at: <https://hipertextual.com/archivo/2011/03/historia-twitter/> [Accessed 28 Sep. 2017].
- [2] Cad.com.mx. (2017). Historia de Twitter. [online] Available at: http://www.cad.com.mx/historia_de_twitter.htm [Accessed 28 Sep. 2017].
- [3] Rubyonrails.org.es. (2017). Ruby on Rails - El desarrollo web que no molesta. [online] Available at: <http://rubyonrails.org.es/> [Accessed 28 Sep. 2017].
- [4] Como hackear una cuenta de Twitter gratis — Hackear Cuentas. [online] Hackearcuentas.com. Available at: <http://hackearcuentas.com/twitter/> [Accessed 28 Sep. 2017].
- [5] Albors, J. and Albors, J. (2017). ¿Sabes qué es un exploit y cómo funciona?. [online] WeLiveSecurity. Available at: <https://www.welivesecurity.com/la-es/2014/10/09/exploits-que-son-como-funcionan/> [Accessed 28 Sep. 2017].
- [6] <https://stackoverflow.com/questions/41216047/apktool-decompile-with-and-apktool/41216097>
- [7] <http://www.openexpo.es/mejores-distribuciones-linux-seguridad/>
- [8] <http://computerhoy.com/paso-a-paso/software/que-es-kali-linux-que-puedes-hacer-41671>
- [9] <http://www.tuprogramacion.com/glosario/que-es-el-android-manifest/>