Smart TV: FLocker Malware Threat Report

Date: 15-07-2021

K. Narahari

Overview:

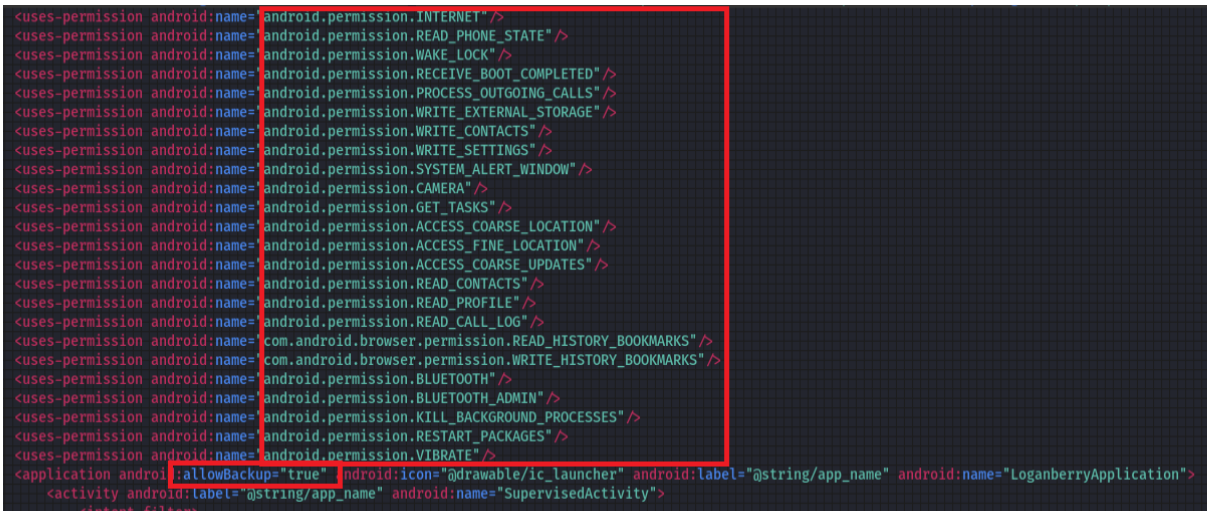
Smart TVs have become a favourite entertainment device as the prices have decreased and the quality has increased. They provide features for running the android applications like Amazon, Netflix, etc. on the television. Smart TVs connect to the internet to run the applications, they are prone of getting hacked.

In our Honeypot, we found a malicious FLocker ransomware which was able to lock the Smart TV's screen by enabling the screen lock. The malware unlocks the screen by paying the demanded ransom. The latest variant of FLocker is a trojan that pretends to be US Cyber Police or other law enforcement agency.

File Hash: **2a66064c4eb25c2234d707ddcaa14bbb**

Technical Analysis:

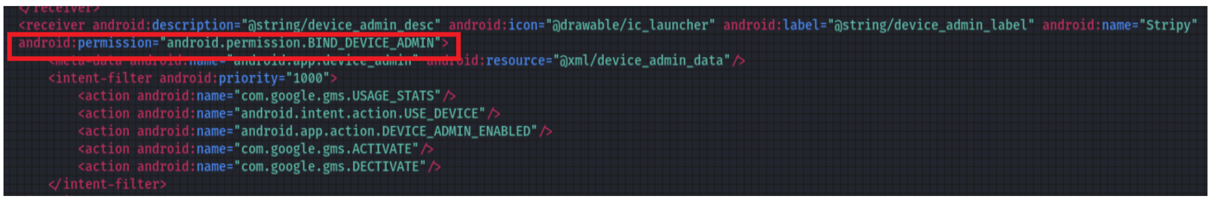
We downloaded the malware sample and reverse-engineered the apk, then performed static analysis and went through the manifest.xml file which is the primary file to begin the analysis.



```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
<uses-permission android:name="android.permission.PROCESS_OUTGOING_CALLS"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_CONTACTS"/>
<uses-permission android:name="android.permission.WRITE_SETTINGS"/>
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.GET_TASKS"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_UPDATES"/>
<uses-permission android:name="android.permission.READ_CONTACTS"/>
<uses-permission android:name="android.permission.READ_PROFILE"/>
<uses-permission android:name="android.permission.READ_CALL_LOG"/>
<uses-permission android:name="com.android.browser.permission.READ_HISTORY_BOOKMARKS"/>
<uses-permission android:name="com.android.browser.permission.WRITE_HISTORY_BOOKMARKS"/>
<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
<uses-permission android:name="android.permission.KILL_BACKGROUND_PROCESSES"/>
<uses-permission android:name="android.permission.RESTART_PACKAGES"/>
<uses-permission android:name="android.permission.VIBRATE"/>
<application androi :allowBackup="true" ndroid:icon="@drawable/ic_launcher" android:label="@string/app_name" android:name="LoganberryApplication">
    <activity android:label="@string/app_name" android:name="SupervisedActivity">
```

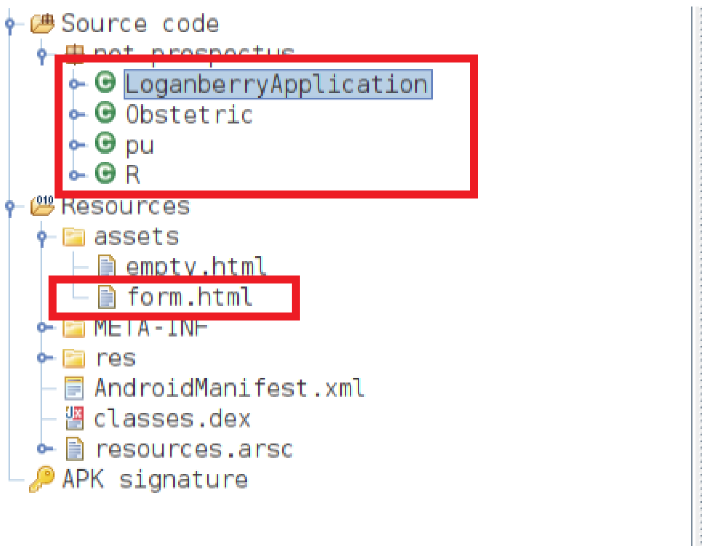
From the permission list used by this .apk, we observed that there are plenty of permissions given which are dangerous and can be used by the malware to perform illegal and malicious tasks. Like camera, contacts, settings, location, restart packages, bluetooth, and a few more which we can see in the screenshot.

These are the different services which is used by this application to perform malicious activities. Along with that "allowbackup" has been set to "true" which means that the complete data which is recorded by this malware can be stored in the device. As this is a ransomware, it can be more dangerous with the enabled permissions.



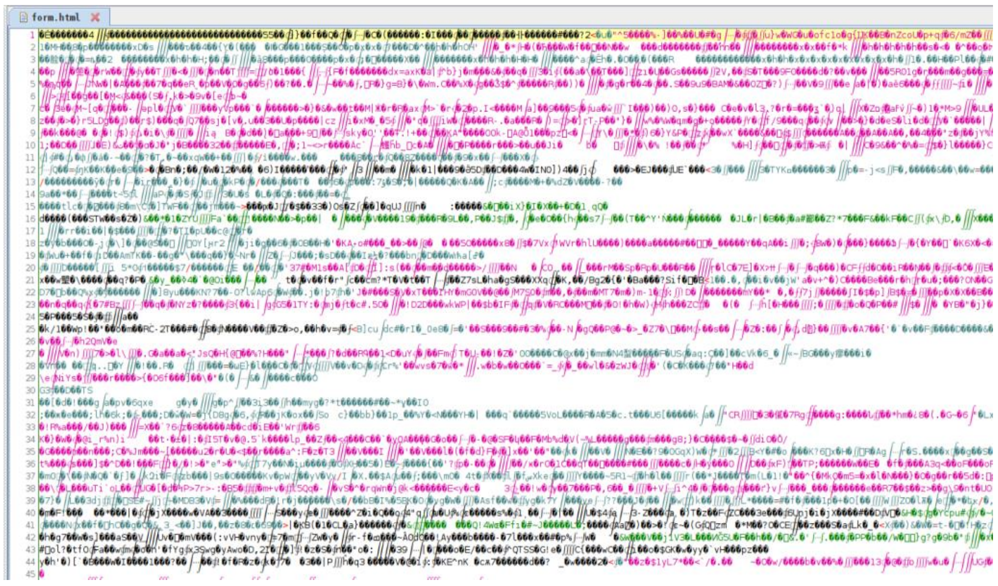
```
<receiver android:description="@string/device_admin_desc" android:icon="@drawable/ic_launcher" android:label="@string/device_admin_label" android:name="Stripy"
android:permission="android.permission.BIND_DEVICE_ADMIN">
    <meta-data android:name="android.app.device_admin" android:resource="@xml/device_admin_data"/>
    <intent-filter android:priority="1000">
        <action android:name="com.google.gms.USAGE_STATS"/>
        <action android:name="android.intent.action.USE_DEVICE"/>
        <action android:name="android.app.action.DEVICE_ADMIN_ENABLED"/>
        <action android:name="com.google.gms.ACTIVATE"/>
        <action android:name="com.google.gms.DECTIVATE"/>
    </intent-filter>
```

In the manifest file, the malware also enables the device administration using the broadcast receivers. The security policies have been setup in the metadata by the attacker. Periodically, the device will connect to command-and-control server (C2). Then, the full access of the device will be maintained and even disabled by the attacker.

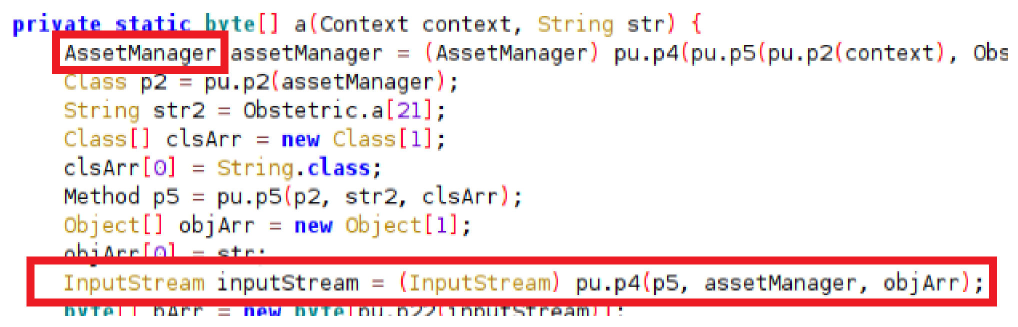


```
Source code
    net.prospectus
        LoganberryApplication
        Obstetric
        pu
        R
Resources
    assets
        empty.html
        form.html
    META-INF
    res
    AndroidManifest.xml
    classes.dex
    resources.arsc
APK signature
```

After reverse engineering the application, we can see only 4 highlighted java classes can be observed. In fact, this malicious app maintains its persistence by hiding the actual code inside the resources > assets > folder.

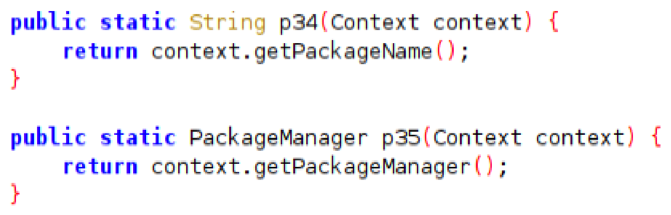


When the application is launched, it decrypts and loads the remaining code from the assets folder. This mechanism is chosen by this ransomware to escape the static analysis of the application and pretending to be normal source code.



```
private static byte[] a(Context context, String str) {
    AssetManager assetManager = (AssetManager) pu.p4(pu.p5(pu.p2(context), Obs
    Class p2 = pu.p2(assetManager);
    String str2 = Obstetric.a[21];
    Class[] clsArr = new Class[1];
    clsArr[0] = String.class;
    Method p5 = pu.p5(p2, str2, clsArr);
    Object[] objArr = new Object[1];
    objArr[0] = str;
    InputStream inputStream = (InputStream) pu.p4(p5, assetManager, objArr);
```

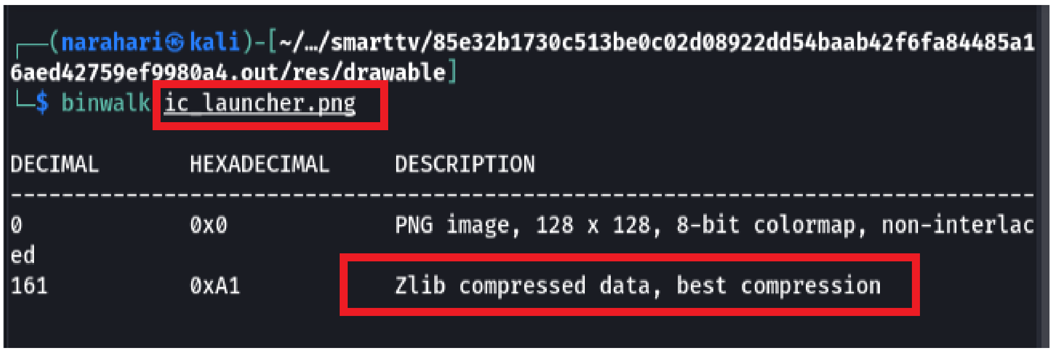
Asset Manager provides access to the application's raw asset files. It is used by most applications that needs to retrieve their resource data. This class presents a lower-level API that allows you to open and read raw files that have been bundled with the application as a simple stream of bytes. In this case, it is using this to load the ransomware actual code which was embedded in the resources folder.



```
public static String p34(Context context) {
    return context.getPackageName();
}

public static PackageManager p35(Context context) {
    return context.getPackageManager();
}
```

Package Manager is used for retrieving various kinds of information related to the application packages that are currently installed on the device. Here, the application is gathering all the applications installed on this device so that it can be send to C2 server and used in device administration for ransomware activities.



```
┌──(narahari㉿kali)-[~/…/smarttv/85e32b1730c513be0c02d08922dd54baab42f6fa84485a1
6aed42759ef9980a4.out/res/drawable]
└─$ binwalk ic_launcher.png

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             PNG image, 128 x 128, 8-bit colormap, non-interlac
ed
161           0xA1            Zlib compressed data, best compression
```

From the above image, we can see that malicious payload has been embedded inside an image. ic_launcher is the general icon image for android applications. When the icon is pressed the embedded compressed payloads is also executed. Thus, maintaining the persistence of the malware by running in the background.

IOCS:

http://42.250.102.101
http://185.215.113.31/api/getbotinjects

MITRE Techniques:

Broadcast receivers (T1402)
Access Contact List (T1432)
Masquerade as Legitimate Application (T1444)
Access Sensitive Data in Device Logs (T1413)
Execution (TA0041)

Subex Secure Protection

Subex Secure detects the android sample as “SS_Gen_Android_Flocker_A”.

Our Honeypot Network

This report has been prepared from the threat intelligence gathered by our honeypot network. This honeypot network is today operational in 62 cities across the world. These cities have at least one of the following attributes:

- Are landing centers for submarine cables
- Are internet traffic hotspots
- House multiple IoT projects with a high number of connected endpoints
- House multiple connected critical infrastructure projects
- Have academic and research centers focusing on IoT
- Have the potential to host multiple IoT projects across domains in the future.

Over 3.5 million attacks a day is being registered across this network of individual honeypots. These attacks are studied, analysed, categorized, and marked according to a threat rank index, a priority assessment framework that we have developed within Subex. The honeypot network includes over 4000 physical and virtual devices covering over 400 device architectures and varied connectivity mediums globally. These devices are grouped based on the sectors they belong to for purposes of understanding sectoral attacks. Thus, a layered flow of threat intelligence is made possible.