SUBEX

Android – Embedded Trojan Dropper Threat Report

Date: 23-04-2021

K. Narahari

With the advancement of the digital world, most of the payments, transactions for shopping, entertainment etc. are becoming cashless. Now, for the ease of users, the developers are building applications that have the feature to make payments through the applications. Thus, the attackers are also focusing on these applications to exploit so that they can make more profit instead of attacking the generic applications.

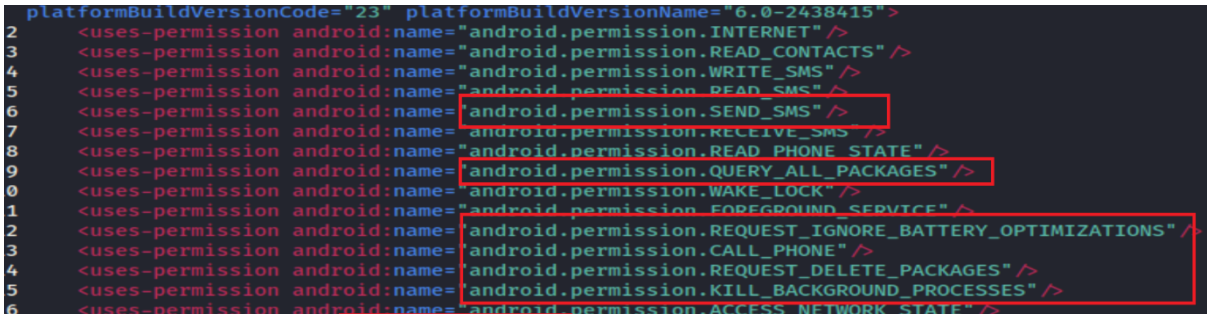
Overview:

The analysed application contains an injected backdoor application package. When this application is installed and launched, the backdoor package is also installed, and the malicious activities are carried out by the attacker.

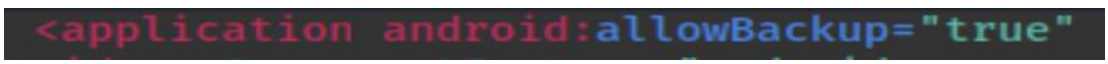
File Hash: 90301aab0ecbc7b43af72f82930e1284

Technical Analysis:

At first, after reverse-engineering the sample, we did the static analysis of the application. We started by examining the permissions from the androidmanifest.xml file.



From the above permissions, there are many malicious permissions out of which **"QUERY_ALL_PACKAGES"** is banned by the Google play store because this is used to know all the applications and packages installed on the device and this can also be used to whitelist any malicious packages. The attackers use **"REQUEST_IGNORE_BATTERY_OPTIMIZATIONS"** as this can run backdoor and backend services which in turn can disable the security and utilize the resources like CPU, memory, etc. They also obtain permission to delete the applications and replace the applications and packages with malicious ones. They also use cellular-related permissions for calling, SMS, and gathering network information, with this we can confirm that they are using malicious permissions for illegal activities.



The attacker takes the backup and stores all the data not only related to this application but for all the applications for which the backup is enabled in other applications. While enabling the backup the user already declares permission to access all the applications on the device. If any network issue or cellular

problem exists then it will make a backup on local storage so that the recorded data can be sent later to the remote attacker.

```
l extends Application implements 003 {
 COMPRESSED_LIBS_ARCHIVE_NAME = "libs.spk.zst";
```

```
public void onCreate() {
  super.onCreate();
  if (created) {
    Log.w("Application onCreate called after application already started");
    00E.A00 = Boolean.FALSE;
    return;
  }
  created = true;
  01f.A01("AppShell/onCreate");
  try {
    this.genderUtils = 01d.A00();
    01T.A0D = true;
```

This is the malicious backdoor package "**libs.spk.zst**" then the backdoor will be installed on the device whenever the application is run and using hooks to run along main activity using Superclass.

```
der stringBuilder = 008.A0R("appshell/debug_info: pkg=");
der.append(getPackageName());
der.append("; v=");
der.append("2.21.3.19-play-release");
der.append("; vc=");
der.append(210319004);
der.append("; p=");
der.append("consumer");
der.append("; e=");
der.append(180L);
der.append("; g=");
der.append("smb-v2.21.3.19-dirty");
der.append("; t=");
der.append(1613594692000L);
der.append("; d=");
der.append(Build.MANUFACTURER);
der.append(" ");
der.append(Build.MODEL);
der.append("; os=Android ");
der.append(Build.VERSION.RELEASE);
```

They used **"QUERY_ALL_PACKAGES"** permission, using this they can acquire all the package details which can be seen in the above image and also, they are gaining shell of the victim's device and gathering installed applications and also, they can install/uninstall or replace the existing or new applications.

```
if ("android.intent.action.CREATE_SHORTCUT".equals(getIntent().getAction()))
  setTitle(((22K)super).A01.A00(2131888894));
setContentView(2131558560);
if (Build.VERSION.SDK_INT >= 21)
  getWindow().addFlags(-2147483648);
OKI OKI = A04();
ContentPickerFragment contactPickerFragment = (ContentPickerFragment)OKI.A0Q.A01("ContactPickerFragment");
```

Creating manual shortcuts to directly access the intents and in this screenshot, they are using this to access the contacts directly.

```
if (intent.hasExtra("android.intent.extra.shortcut.ID"))
  bundle1.putString("jid", intent.getStringExtra("android.intent.extra.shortcut.ID"));
Bundle bundle2 = new Bundle();
bundle2.putString("action", intent.getAction());
bundle2.putString("type", intent.getType());
bundle2.putBundle("extras", bundle1);
paymentContactPickerFragment.A0N(bundle2);
```

They are passing the data between activities of one application to another application for performing payments using the ID and sub-values.

Attacker can even compose a message to victim contacts, read messages as we saw in permissions "**READ_SMS WRITE_SMS SEND_SMS**".

```
    throw new IllegalStateExceptior
  case 27:
    return "restoring-from-local";
  case 26:
    return "msgstore-restored";
  case 25:
    return "return-from-auth";
  case 24:
    return "restoring-from-gdrive";
  case 23:
    return "restore-from-local";
  case 22:
    return "restore-from-gdrive";
  case 21:
    break;
 }
 return "new";
```

From the screenshot we can observe that the attacker enabled the backup=true so using that these actions can be performed regarding not only about messages they can also perform many malicious behaviours.

```
if (OHw2.AOO() == 0) {
  KeyguardManager keyguardManager = (KeyguardManager)this.A04.getSystemService("keyguard");
  if (keyguardManager != null && keyguardManager.isDeviceSecure())
    return true;
  009 0091 = this.A05;
```

For device locking, fingerprints, password, pin keyguard manager is required so attacker uses this to lock/unlock device.

```
quence("title", str);
n("allow_device_credential", true);
n("require_confirmation", false);
```

We can see attacker stealing lock related data and we can also see in the above screenshot that "**require confirmation= false**" and we can observe that without victim awareness the attacker is stealing lock and password data.

IOCS:

http://wa.me
libs.spk.zst (backdoor package)

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_Andro_Trojan_Dropper_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.