RCA Report (EN)

Root Cause Analysis (RCA): Keypad Unresponsive & System ANR

1. Executive Summary

When the user attempted to wake up the device (DVT3 Thorpe, OS-02.01.01.260119), a **System Server ANR** (Application Not Responding) occurred, causing the keypad to become unresponsive and preventing screenshots from being taken. The system recovered automatically later.

Log analysis confirmed the **Root Cause** to be the **GNSS HAL getting stuck**, which blocked the critical `android.fg` thread within `system_server`, thereby stalling the screen wake-up process.

2. Root Cause

The GNSS HAL (GPS) timed out during QMI communication, causing the `android.fg` thread in `system_server` to be blocked for an extended period.

1. **GNSS HAL Timeout**: Around 15:31:26, the GNSS Service attempted to stop the location session (`LOC_STOP`) and set the operation mode (`LOC_SET_OPERATION_MODE`) via the QMI interface.
2. **QMI Usage No Response**: The underlying Modem or QMI interface did not respond. After 25 seconds (15:31:51), a `Function timed out` and `qcError 110` were reported.
3. **System Server Blocked**: The `android.fg` thread (TID 1755) in `system_server` was calling `GnssHal::stop()` and waiting for a response. Due to the HAL being stuck, this thread remained in an `UNINTERRUPTIBLE_SLEEP` or blocked state for over 15 seconds.
4. **ANR Triggered**: When the user tried to wake the screen, the system broadcasted `android.intent.action.SCREEN_ON`. This broadcast is

dispatched by the `android.fg` thread. Since the thread was stuck in the GNSS HAL, the broadcast could not be processed, eventually triggering the Watchdog ANR.

3. Evidence

ANR Information

- **Time**: `01-27 15:32:09.418`
- **Process**: `system_server` (PID 1702)
- **Reason**: `Broadcast of Intent { act=android.intent.action.SCREEN_ON ... }`
- **Blocked Thread**: `android.fg` (TID 1755)

Stack Trace (android.fg)

The thread was blocked at `android::hardware::gnss::V1_0::BpHwGnss::_hidl_stop` waiting for a return:
> **Source**: `FS/data/anr/anr_2026-01-27-15-32-09-441` (Lines 443–463)

```
"android.fg" sysTid=1755
  #00 pc 00000000000c12ec
/apex/com.android.runtime/lib64/bionic/libc.so (__ioctl+12)
  ...
  #06 pc 00000000000a5ec8  /system/lib64/android.hardware.gnss@1.0.so
(android::hardware::gnss::V1_0::BpHwGnss::_hidl_stop...+260)
  #07 pc 0000000000033d50  /system/lib64/libservices.core-gnss.so
(android::gnss::GnssHal::stop()+160)
  ...
  #16 pc 0000000000256af0  /system/framework/services.jar
(com.android.server.location.gnss.GnssLocationProvider.stopNavigating+4
```

GNSS HAL Error Logs (Logcat)

The system attempted to stop GPS but timed out: > **Source**: `bugreport-T70-AQ3A.250408.001-2026-01-27-15-33-02.txt`

```
(Line 22444) 01-27 15:31:26.891 ... I SWIGNSS : Stop:
(Line 22447) 01-27 15:31:26.891 ... D SWIGNSS : -> ... Send 11 bytes
...
... (25 seconds with no response) ...
(Line 22668) 01-27 15:31:51.818 ... W SWIGNSS : Function timed out
(Line 22670) 01-27 15:31:51.818 ... E SWIGNSS : swigps_setPosMode:
Stop failed, qcError 110
```

4. Impact Analysis

- **Why did the Keypad fail?**: While Key events are received by the InputDispatcher, they require the upper-layer WindowManager Service (WMS) and PowerManager Service (PMS) to handle the wake-up logic. These critical services state transitions rely on the `android.fg` thread. When this thread is blocked, the system cannot complete the "wake from sleep" state transition, resulting in no response to user input.
- **Why did screenshots fail?**: Screenshot functionality also depends on `system_server` to process key combinations and capture notifications, which couldn't execute because the core service was blocked.

5. Recommendations

1. **Check Modem/GNSS Status**: The Modem team needs to analyze QMI logs (if available) or a Modem dump to confirm why QMI did not respond to the `LOC_STOP` command around 15:31:26. The Modem might have been in an error state or failed to wake from deep sleep.
2. **HAL Layer Guard**: It is recommended that even if the underlying layer times out, the HAL layer should return an error to the System Server as soon as possible, preventing the main thread (`android.fg`) from blocking for long periods (over 10-20 seconds) to avoid a full system ANR.