[背景]
AP休眠状态下，偶发出现接收不到volte来电，但主叫方却正常振铃

[分析]
从LOG看，RIL已将来电通知给框架，并且ImsServiceClassTracker已经发送了incoming call的广播，但是ImsPhoneCallTracker没有接收到这个广播。
11352 07-28 18:26:25.917 3363 4503 I QImsService: ImsSenderRxr : [UNSL]< UNSOL_RESPONSE_CALL_STATE_CHANGED [id=1,INCOMING,toa=129,norm,mt,0,voc,noevp,, cli=1,,3Call Details = 3 2 callSubState 0 videoPauseState2 mediaId2 Local Ability Peer Ability isValid = true type = 0 status = 2 accTechStatus mode = 14 Status = 2 restrictCause = 0 registered = 0isValid = true type = 3 status = 2 accTechStatus mode = 14 Status = 2 restrictCause = 0 registered = 0 Cause code 0,CallFailCause Code= 0,CallFailCause Extra code = 0,CallFailCause String= null,ECT mask: 0,isEncrypted=false] [SUB0]
11376 07-28 18:26:25.926 3363 4503 I QImsService: ImsSenderRxr : [UNSL]< UNSOL_RESPONSE_CALL_STATE_CHANGED [id=1,INCOMING,toa=129,norm,mt,0,voc,noevp,, cli=1,,3Call Details = 3 2 callSubState 0 videoPauseState2 mediaId2 Local Ability Peer Ability isValid = true type = 0 status = 2 accTechStatus mode = 14 Status = 2 restrictCause = 0 registered = 0isValid = true type = 3 status = 2 accTechStatus mode = 14 Status = 2 restrictCause = 0 registered = 0 Cause code 0,CallFailCause Code= 0,CallFailCause Extra code = 0,CallFailCause String= null,ECT mask: 0,isEncrypted=false] [SUB0]
11429 07-28 18:26:26.010 3363 3363 D QImsService: ImsServiceClassTracker : sending Incoming call intent:Intent { act=com.android.ims.IMS_INCOMING_CALL flg=0x10000000 }

分析发现，ImsServiceClassTracker发送完广播之后，因为AP没有持有wakelock，
所以很快进入休眠。AP休眠之后，ImsPhoneCallTracker不能接收广播。
[250.532674,0] PM: suspend entry 2017-07-28 10:26:26.014363539 UTC
[250.575938,0] Resume caused by IRQ 315, qcom,smd-rpm
[250.575938,0] Resume caused by IRQ 56, ipa
[250.576103,0] Enabling non-boot CPUs ...
[250.651262,5] Suspended for 25.764 seconds
[250.651521,6] PM: suspend exit 2017-07-28 10:26:51.779624842 UTC

[结论]
Qcril将来电消息发给ImsSenderRxr.java后，会释放持有的wakelock。
但框架接收到来电之后没有立即申请wakelock，导致框架处理过程中进入休眠。

[解决方案]
我们有CR2058135，在ImsSenderRxr.java申请一个wakelock，确保Modem主动上报的消息在AP

休眠以前发给应用层。
具体代码修改：
Change-Id: I9fa903822651cb6dbc1209e48ac412582e1bfaff
CRs-Fixed: 2058135
---
ims/src/org/codeaurora/ims/ImsSenderRxr.java | 43 ++++++++++++++++++++++++++++
1 file changed, 43 insertions(+)

diff --git a/ims/src/org/codeaurora/ims/ImsSenderRxr.java b/ims/src/org/codeaurora/ims/ImsSenderRxr.java
index 3aabf0b..fce2106 100644
--- a/ims/src/org/codeaurora/ims/ImsSenderRxr.java
+++ b/ims/src/org/codeaurora/ims/ImsSenderRxr.java
@@ -216,6 +216,7 @@ public final class ImsSenderRxr extends ImsPhoneBaseCommands implements ImsPhone
IFMsg_Rxr mReceiver;
WakeLock mWakeLock;
int mWakeLockTimeout;
+ WakeLock mUnsolWakeLock;
// The number of requests pending to be sent out, it increases before
// calling
// EVENT_SEND and decreases while handling EVENT_SEND. It gets cleared while
@@ -250,6 +251,7 @@ public final class ImsSenderRxr extends ImsPhoneBaseCommands implements ImsPhone

static final int EVENT_SEND = 1;
static final int EVENT_WAKE_LOCK_TIMEOUT = 2;
+ static final int EVENT_UNSOL_WAKE_LOCK_TIMEOUT = 3;

// ***** Constants

@@ -260,6 +262,7 @@ public final class ImsSenderRxr extends ImsPhoneBaseCommands implements ImsPhone
static final String[] SOCKET_NAME_IF = {"qmux_radio/rild_ims0", "qmux_radio/rild_ims1", "qmux_radio/rild_ims2"};
static final String TEST_MODE_SOCKET_NAME = "imstestrunnersocket";
static final int SOCKET_OPEN_RETRY_MILLIS = 4 * 1000;
+ static final int UNSOL_WAKELOCK_TIMEOUT_MS = 200;

private RegistrantList mHandoverStatusRegistrants = new RegistrantList();
private RegistrantList mRefreshConfInfoRegistrations = new RegistrantList();
@@ -272,6 +275,18 @@ public final class ImsSenderRxr extends ImsPhoneBaseCommands

```
implements ImsPhone
private RegistrantList mVopsRegistrants = new RegistrantList();
private RegistrantList mParticipantStatusRegistrants = new RegistrantList();

+ private Handler mUnsolWakeLockTimeoutHandler = new Handler() {
+ @Override
+ public void handleMessage(Message msg) {
+ switch (msg.what) {
+ case EVENT_UNSOL_WAKE_LOCK_TIMEOUT:
+ releaseUnsolWakeLock();
+ break;
+ }
+ }
+
+ };
+
public void registerForPhoneId(int phoneId) {
if (mInstanceId == phoneId) {
Log.i(this, "registerForPhoneId: mInstanceId: " + mInstanceId + " UNchanged");
@@ -718,6 +733,8 @@ public final class ImsSenderRxr extends ImsPhoneBaseCommands
implements ImsPhone
mWakeLock.setReferenceCounted(false);
mWakeLockTimeout = SystemProperties.getInt(
TelephonyProperties.PROPERTY_WAKE_LOCK_TIMEOUT,
DEFAULT_WAKE_LOCK_TIMEOUT);
+ mUnsolWakeLock = pm.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK, LOG_TAG + "_
UNSOL");
+ mUnsolWakeLock.setReferenceCounted(false);
mRequestMessagesPending = 0;
mRequestMessagesWaiting = 0;
sTestMode = SystemProperties.getBoolean("persist.qualcomm.imstestrunner", false) &&
@@ -769,6 +786,30 @@ public final class ImsSenderRxr extends ImsPhoneBaseCommands
implements ImsPhone
}
}

+
+ /**
+ * Holds a PARTIAL_WAKE_LOCK whenever an UNSOL event is received from RIL.
+ * A timer is also started to release it.
+ */
+ private void acquireUnsolWakeLock() {
```

```
+ Log.d(LOG_TAG, "acquireUnsolWakeLock");
+ synchronized (mUnsolWakeLock) {
+ mUnsolWakeLock.acquire();
+ mUnsolWakeLockTimeoutHandler.removeMessages(EVENT_UNSOL_WAKE_LOCK_TIMEOUT)
;
+ Message msg = mUnsolWakeLockTimeoutHandler.obtainMessage(
EVENT_UNSOL_WAKE_LOCK_TIMEOUT);
+ mUnsolWakeLockTimeoutHandler.sendMessageDelayed(msg,
UNSOL_WAKELOCK_TIMEOUT_MS);
+ }
+ }
+
+ private void releaseUnsolWakeLock() {
+ Log.d(LOG_TAG, "releaseUnsolWakeLock");
+ synchronized (mUnsolWakeLock) {
+ if (mUnsolWakeLock.isHeld()) {
+ mUnsolWakeLock.release();
+ }
+ }
+ }
+
public void send(IFRequest rr) {
Message msg;

@@ -1231,6 +1272,8 @@ public final class ImsSenderRxr extends ImsPhoneBaseCommands
implements ImsPhone
return;
}

+ acquireUnsolWakeLock();
+
switch (response) {
case ImsQmiIF.UNSOL_RESPONSE_IMS_NETWORK_STATE_CHANGED:
unsljLog(response);
--
1.8.2.1
```