Id : ds-1505
Cross ref : 1499
Result : Unknown

Reason :

Sink statement : Line 590 in the screenshot below

```
577     @Override
578     public void onTextChanged(CharSequence s, int start, int before, int count) {
579       String number = s.toString();
580       if (start == 0 && before == 0 && count == 0) {
581         return;
582       } else if (TextUtils.isEmpty(s) && before > 0) {
583         setCallLogMode();
584         contactsEntryAdapter.resetFilter();
585         list.setSelection(0);
586       } else if (number.equals("*#06#")) {
587         showDeviceId();
588       } else if (number.startsWith("*#*#") && number.endsWith("#*#*")) {
589         String secretCode = new StringBuilder(number).substring(4, number.length()-4);
590         sendBroadcast(new Intent( action: "android.provider.Telephony.SECRET_CODE",
591             Uri.parse("android_secret_code://" + secretCode)));
592       } else {
593         setContactsMode();
594         contactsEntryAdapter.getFilter().filter(s);
```

Source Statement : Line 92 in the below screenshot

```
89
90    public void unknownNumberDialog(final String number) {
91      AlertDialog.Builder builder = new AlertDialog.Builder(activityRef.get());
92      builder.setTitle(PhoneNumberUtils.formatNumber(number, Locale.getDefault().getCountry()));
93      String[] items = new String[]
94          {"Send a message",
```

The graph ds-1499_sink.drawio shows the sources of variables passed to the sink statement, the graph comes to point where there is unknown end to predict the exact source of the variable phone number.