Id : fd-66

Result : False Positive

Sink statement : Line 203 in below screenshot is the sink statement.

```
186 @   private void handleTriggerAlarm(Intent i) {
187       final long alarmid = i.getLongExtra(ALARM_ID, defaultValue: -1);
188       final DbUtil.Settings settings =
189         DbUtil.Settings.get(getApplicationContext(), alarmid);
190
191       PowerManager.WakeLock w = null;
192       if (i.hasExtra(AlarmTriggerReceiver.WAKELOCK_ID)) {
193         w = AlarmTriggerReceiver.consumeLock(
194             i.getExtras().getInt(AlarmTriggerReceiver.WAKELOCK_ID));
195       }
196
197       if (w == null)
198         Log.e(TAG, msg: "No wake lock present for alarm trigger " + alarmid);
199
200       if (activeAlarms == null) {
201         activeAlarms = new ActiveAlarms(getApplicationContext(), w, settings);
202       } else {
203         Log.i(TAG, msg: "Already wake-locked, releasing extra lock");
204         w.release();
205       }
206       activeAlarms.alarmids.add(alarmid);
207
```

All the variables to the sink statement are constants.

```
388
389     private static final String TAG =
390       AlarmNotificationService.class.getSimpleName();
```

Source Statement : Line 33 in following screenshot

```
30
31 @   public static Alarm get(Context context, long id) {
32       Alarm s = null;
33       final Cursor c = context.getContentResolver().query(
34           ContentUris.withAppendedId(AlarmClockProvider.ALARMS_URI, id),
35           new String[] {
36             AlarmClockProvider.AlarmEntry.TIME,
37             AlarmClockProvider.AlarmEntry.ENABLED,
38             AlarmClockProvider.AlarmEntry.NAME,
39             AlarmClockProvider.AlarmEntry.DAY_OF_WEEK,
40             AlarmClockProvider.AlarmEntry.NEXT_SNOOZE },
41           selection: null, selectionArgs: null, sortOrder: null);
42       if (c.moveToFirst())
43         s = new Alarm(c);
44       else
45         s = new Alarm();
46       c.close();
47       return s;
48     }
```