Gmail

[URGENT] Exposure Notification API failing to download diagnosis keys to devices

'via exposure-notifications-early-access 14. januar 2021 01:15
Za: **********@googlegroups.com

Dear Exposure Notifications developers,

We are aware of an issue with the `provideDiagnosisKeys()` method of the Exposure Notifications API which **affects all apps** using the API. The issue became widespread around 12 Jan 22:00 UTC, and it causes the `provideDiagnosisKeys()` to appear to hang. **Please, read this message carefully as it contains important information about how to solve this problem**.

Tl;dr;

- There is an issue currently affecting all users that prevent apps from matching the latest exposure data (since Jan 12th, 2021), but not from continuing broadcasting or scanning keys to devices in proximity
- A fix is rolling out and will reach over 90% of your users by 1/14/2021 AM, EST time, but you may need to review your app implementation to confirm whether this will resolve the issue completely or action is required (see below for details)
- Users should *not* manually clear cache or storage, or uninstall the app. This won't fix the issue, and it will incur exposure data loss on their devices

What is happening?

Calls to `provideDiagnosisKeys()` never return due to a defect originated on Jan 12th 2021, and in consequence matching does not occur. As a result, Exposure Notifications apps can't match the latest exposure data (since Jan 12th only). This is the only EN API function affected, and we can confirm that:

- Apps are handling exposure data in the device uninterrupted (broadcasting and scanning)
- Uploading keys is working correctly and was never interrupted (reporting a case through the app)

We have determined the root cause, and **a fix is already rolling out to users' devices**. Once the fix is installed in the device, calls to `provideDiagnosisKeys()` will have the expected behavior again. The rollout of the fix will reach over 90% of your users within 24 hrs.

How is this affecting your users in particular?

How your users experience this issue depends on your integration of the Exposure Notification API. This is true also for what action will be required to recover.

If your app periodically calls `provideDiagnosisKeys()`, it is likely to recover automatically. However, if a call to provideDiagnosisKeys() blocks or hangs, and the method is never called again, then users may need to reboot their devices to get the app back into a good state.

Some cases we have observed are:

Apps with timeouts for the key download process and retries to call `provideDiagnosisKeys()`
For these apps, the issue is not easily visible, and very likely will recover on its own. If your app has a timeout mechanism implemented for the `provideDiagnosisKeys()` call, the only symptom for users is probably just not seeing exposure notifications since the last successful download. They might notice that the last successful check happened more than 24 hours ago. Any potential exposures should be detected once their device receives the fix, as long as the app picks up all the unprocessed files correctly. This is the case, for example, of our reference implementation.

Apps with no timeout and visible notifications for the key download process
If your app shows a visible and/or persistent notification when pulling diagnosis key data from the server, users may notice it getting stuck. This might be the case if you are using a foreground service job and you don't have a timeout mechanism to abort the call when it doesn't return the data. This makes it apparent to the user that the download process is blocked, and it might alarm them. In these cases, rebooting the device might help eliminate the notification. Like in the previous case, any potential exposures should be detected once their device receives the fix, as long as the app picks up all the unprocessed files correctly.

Apps with no retries to call `provideDiagnosisKeys()`
If, once a call to `provideDiagnosisKeys()` blocks or hangs, the method is never called again, then users may need to reboot their devices to get the app back into a good state.

If you think your users might be affected in other ways and need additional guidance to estimate the impact, please reach out to your Google contact or .

Do developers need to take action?

This fix will not require any immediate code changes to your app. However, we are still investigating and every integration has its own particularities, so we encourage you to monitor your app's activity, inform your users of the situation if you have the means and review a few items, if possible:

- Check whether your app has a timeout mechanism that would allow it to gracefully recover from the unresponsive call to `provideDiagnosisKeys()`. If it's not the case, you might want to implement it for all API calls to avoid a similar issue in the future.
- Check whether your app will correctly pick up all the pending diagnosis key files from the server, regardless of when they were generated. This includes implementing a retry strategy that is conscious of quota limitations. If it's not the case, reach out to your Google contact or if you need help determining the best way to recover.

As always, our support team is available at for any question you may have. Please reach out if you need additional assistance.

Do users need to take action?
It's important to stress **users should *not* manually clear cache or storage**. This will not resolve the problem and will also remove all RPIs and TEKs. Here are some points you can share with your users to resolve their issues:

- **Explain that their app is still correctly handling local data, and uploading to the server when users request it.** New notifications won't show until the fix is in their phone, but once it is all their potential exposures should be accounted for unless they deleted their local exposure data.
- **Rebooting the device might help depending on app implementation**, especially if the app has a persistent notification that looks perpetual because it's waiting for a callback that will never arrive.

- **Discourage uninstall**, users will lose their exposure data if they do.
- **Discourage deleting local exposure data** via settings.
- **Discourage switching exposure notifications off.** This would stop the visible persistent notifications we are seeing from some apps, but users might not remember to switch back on once the fix is deployed and will stop broadcasting and scanning for new data.
- Switch WiFi off and on has helped some users get rid of the persistent notification; however, this doesn't solve the issue on the app, the fix still needs to be installed. This should happen automatically.
- The fix is actively being installed on devices and we expect to reach saturation in 24hrs. If users continue experiencing the issue after that period, they could try some of this advice.

Kind regards,

The Exposure Notification Team.