Dynamic Feature Modules - Doesn't extract native ".so" libs

WAI Bug P2 + Add Hotlist

STATUS UPDATE No update yet. Edit

DESCRIPTION em...@gmail.com created issue #1

Build: AI-201.7223.91.41.6507185, 202005182153,

AI-201.7223.91.41.6507185, JRE 1.8.0_242-release-1644-b01x64 JetBrains s.r.o, OS Windows 10(amd64) v10.0 , screens 1536x864

AS: 4.1 Canary 10; Kotlin plugin: 1.4-M2-release-Studio4.1-1; Android Gradle Plugin: 4.1.0-alpha10; Gradle: 6.5; NDK: from local.properties: 21.0.6011959-beta2, latest from SDK: (not found); LLDB found)Source: user_sentiment_feedback

1. Enabled in Module's manifest : android:extractNativeLibs="true"
2. Enabled in gradle.properties: android.bundle.enableUncompressedNativeLibs=false

The problem is facing only in Android 10 and i have problems with all Android 10 users suffering from that issue.

Some native libraries are executed from "exec" directly from libs dir.

Also, noticed that native libs are extracted in : user/0/com.xxxxx.xxxxx/splitcompat/xxx/native-libraries/modulename/libexample.so

But does this directory have execute permissions like normal libs dir in Android 10 and why it is not extracting them anymore like before ?

Using : api("com.google.android.play:core:1.7.3") classpath ("com.android.tools.build:bundletool:0.15.0")

It is happening when download the modules from play store, because when testing locally (debug mode) it is working OK. After uploading the .aab and downloading/installing modules, not extra

Also, native libs extracted in : user/0/com.xxxxx.xxxxx/splitcompat/xxx/native-libraries/modulename/libexample.so **are not executable(permission denied)** !

How to deal with such big problem ?

As i said before it worked OK, native libs was extracted to the main app libs folder and was able to execute, but after upgrade AS, Gradle and many many other plugins it is not working anymore a

Before, after dynamic module installed, it was able to extract bundled native libs to correct dir and i was able to execute them without any problem in Android 10(Q) like this :

```
Runtime.getRuntime().exec(context.applicationInfo.nativeLibraryDir + File.separator + "libexample.so")
```

Sorry for repeating again, but it is really important for the app's functionality and the app is constantly being affected with bad reviews because of it !

✓ Mentioned issues (1) ✓ Links (8)

Mentioned issues (1)

P3 [Android Q Beta] Apps can no longer execute binaries in their home directory (execute_no_trans) "https://issuetracker.google.com/128554619"

Links (8)

"To speed up the investigation on our side, could share the package name of your application with us and would you mind sharing an internally shared version of your app for which the bug is repro
"Free One : https://play.google.com/store/apps/details?id=com.eakteam.network... "
"Paid One : https://play.google.com/store/apps/details?id=com.eakteam.networkman..."
"Also, you can download it directly from the Play Store, also added your email at licensed users. https://play.google.com/apps/test/RQsVeSJlU9E/ahAJEhp-l0FCGu_UkNNWxUrNHqEEpYuA6aiKinepY9
"...stead of adding every native library to the base app you test out install-time modules with conditional delivery (only deliver to Android 10). See https://developer.android.com/guide/app-bundle/con

See all related links

COMMENTS

em...@gmail.com <em...@gmail.com> #2

UPDATE :

If keeping the app installed including dynamic feature module installed and upload new update to Play Console and updating the app directly again it is extracting and working OK.

But if installed the app for the first time(as new user), it is not extracting them and all new users are unable to use the app.

Weird....

al...@google.com <al...@google.com>

Assigned to ja...@google.com.

ja...@google.com <ja...@google.com> #3

Hi

Thanks a lot for reporting this issue. We will look into this issue.

To speed up the investigation on our side, could share the package name of your application with us and would you mind sharing an internally shared version of your app for which the bug
I think I remember your app is paid, so you can specify a list of allowed testers: Could you give jakobschneidertest1@gmail.com access to the internal shared version?

---

em...@gmail.com <em...@gmail.com> #4

Hello, thank you so much for replying !

I have 2 apps with the same issue :

1. Free One : https://play.google.com/store/apps/details?id=com.eakteam.networkmanager
2. Paid One : https://play.google.com/store/apps/details?id=com.eakteam.networkmanager.pro

You said : "I think I remember your app is paid", how did you remember that :D :D....

Anyway i have uploaded the app to the internal testing and added your email address to the testers list.

Also, you can download it directly from the Play Store, also added your email at licensed users. https://play.google.com/apps/test/RQsVeSJlU9E/ahAJEhp-l0FCGu_UkNNWxUrNHqEEpYuA6a

Please inspect with high priority only the paid app firstly(it is contains more dynamic feature modules than free one)

Extra notes for fast debugging saving time:

1. After installing the app and launching for first time, open the main navigation drawer
2. I suggest you to test modules with name : Iperf3, Nmap Scanner, VPN Client (there are more but i think this is enought)
3. After module/s are installed you will notice that they will not extract assets/nativeLibs or other resources at the main app directory (normally: /data/app/com.eakteam.networkmanager

Also i have noticed that if keep the app installed together with dynamic features, upload new version and update it, all previous dynamic features start to work OK, they extract everything as t

Sorry for my "not good English" and best regards. Waiting for a solution

---

em...@gmail.com <em...@gmail.com> #5

Also, as temporary solution, i'm thinking to add all the native libs to the base app module because in the first install it is extracting everything OK, but it can make my app download size +20 M

So, just tell me if you have any ETA to solve it from DFMs without needed to apply the temporary solution in my own.

Thanks again !

---

ja...@google.com <ja...@google.com> #6

Thanks, so far I cannot provide you with an ETA unfortunately (until we have found out the root cause).

In the meantime: You mentioned that this happened only after upgrading many plugins. If this is a problem for many users of yours it might be a good idea to roll-back to a previous version o

Another idea: Instead of adding every native library to the base app you test out install-time modules with conditional delivery (only deliver to Android 10). See https://developer.android.com/

---

em...@gmail.com <em...@gmail.com> #7

Thanks again for the reply.

But i cannot roll-back because it seems to happens since 1 or maybe 2 months before, but i have ignored the bug because of very low number of users (In my phone was working OK :) :)).Nov
remember all the changes, but if it is related to `Play Core Library`, i remember that some months before i have received an warning on the Play Console who says that i should upgrade the

Maybe it is causing that ? I Don't know.

Also, tested with `install-time modules with conditional delivery`, thanks.

Please keep me informed as soon as you can, it is really harming the users trust and the app's reputation.

Best Regards

---

em...@gmail.com <em...@gmail.com> #8

More info to help you find the root cause.

After playing around and testing different things, something new weird happened :

1. Installed the app from the Play Store (fresh/new install)
2. Choose and install DFM from app like before (issue still present)
3. Leaving the device idle/locked for maybe about 1 hour or maybe less
4. Unlocked it again and opened the app (just for some other purposes)
5. Noticed that the DFM installed before, now works OK and it was extracted everything as it should
6. Surprised from that event and choose to add this comment here :) :)

Also, uninstalled the app and tried again the same pattern as above just to be sure that i was not a coincidence, and yes, it react in the same way.... :O :O

It seems that it is extracting everything in background when the device is idle after some time, and not immediately as supposed to do.

So, to summarize we have 3 cases here :

1. When installed the app for the first time(as new user) and after this install DFM too, it is not working **(Root Bug)**
2. Keeping the app as it is with DFM installed(not working), publishing new update to the Play Store, updating the app to the new version, it is working OK and extracted everything for every
3. When installed the app for the first time(as new user) and after this install DFM too, it is not working **(Root Bug)**. **But**, without publishing new version or doing any change to existing app works as they should.

This seems strange to me, because when the DFM is downloaded and start installing, it supposed to extract everything immediately but unfortunately it doesn't...

Hope that this added info will help you more to find out the problem.

Again, thanks and best regards! Waiting for an update from you.

---

ja...@google.com <ja...@google.com> #9

Hi,

I think I can shed some light into what exactly is happening here:

1. Dynamic Feature Modules in practice .apk files in addition to the base APK. However in order to install additional .apk files for an app the app needs to be restarted (or not running in the
2. We have developed a way a to immediately dynamically load these .apk files so that they can be used without restarting the app. This feature is called SplitCompat
3. The behaviour you describe makes it clear that once DFMs are fully installed, they work fine. However the dynamic loading seems to have issues.

I think the crux of this issue is how native libraries are loaded in the app (`Runtime.getRuntime().exec(context.applicationInfo.nativeLibraryDir + File.separator + "libexampl` libraries are probably in a different folder.

Good news is that this should be fixable: We are aware of this special interaction and it should still be possible load the native libraries.
Could you try these steps to fix the issue:

1. Please make sure that you are correctly initializing SplitCompat, see guide here.
2. Instead of using `Runtime.getRuntime().exec(context.applicationInfo.nativeLibraryDir + File.separator + "libexample.so")` (this relies on a static path which might ch of the native library and load it.

Let me know if this change works for you!

---

em...@gmail.com <em...@gmail.com> #10

Hi again !

I am fully aware of `SplitCompat` and followed everything as it should!

Also, understand that DFM are fully installed, but extraction process nativelibs/assets etc. are not happening immediately (before it does !)

1. I am sure that have correct initialization of SplitCompat
2. Cannot use `loadLibrary` because as i said, native libraries are static executable, not managed by JNI, this is why i use static path, but in fact it is not static and cannot be change as yo

The change doesn't work for my case with `executable libraries`, but i am sure that is is working OK, if we have to deal with JNI. But this is not the issue of my case.

Everything from DFMs should be extracted immediately to the base app if instructed to do that from `manifest` or `gradle.properties`

Thanks

---

ja...@google.com <ja...@google.com> #11

Status: Won't Fix (Intended Behavior)

Hi,
Thanks for the extra clarification, the case where native libraries do not contain JNI and cannot be loaded over `System.loadLibrary()` is a special case.

Executing such binaries is not possible anymore from Android 10 onwards, more details about this can be found in this issue https://b.corp.google.com/issues/128554619. This is an Androi

In order to make your libraries work on Android 10 there are a two options that could work for you:

1. Implement a JNI interface for your affected libraries and load them via `System.loadLibrary()`
2. Include the affected libraries directly in your base APK or in install-time modules

---

ja...@google.com <ja...@google.com> #12

Sorry, I provided the wrong link, here is the correct one to the issue: https://issuetracker.google.com/issues/128554619

---

em...@gmail.com <em...@gmail.com> #13

My friend, i don't think you'r right...

While exec() no longer works on files within the application home directory, it continues to be supported for files within the read-only /data/app directory. In particular, it should be possible to exec() on the /data/app artifacts.

And, i said it works after leaving device sometime idle (something happening in the background), but it should do it immediately.

I am just saying that DFMs doesn't extract their artifacts as they should do to the /data/app directory, no matter if it is native executable or any asset or raw file.

Why they are extracting later (after idle time) ??? Why doesn't extract immediately like before ???

---

em...@gmail.com <em...@gmail.com> #14

Also, why it is ectracting fine everything if keep the `non-working` modules installed, and update the app later ???

The issue is definitely **not** `Won't Fix (Intended behavior)` and i think something goes wrong here, `it is a bug`

And also if supposing that you'r right about packaging the libs to main base module to exec(), why DFMs not ectracting them like base module. DFMs are supposed to make the app's downlo

Also, please take a look at comment #13

---

em...@gmail.com <em...@gmail.com> #15

Also, DFMs doesn't extract assests except libraries to base app. They can be opened by `assetmanager` but in some cases we need directly static file access and it is not working also.

Please review last 3 comments and suggest or review the bug it is very important for my app integration and functionality !

---

ja...@google.com <ja...@google.com> #16

Yes, you're correct. Sorry I realize it might not be clear what I was trying to say:

It will still be possible to use `exec()` like you did for any files in the read-only `/data/app` directory you have mentioned. This is where the native libs are extracted from the apk.
The reason why this works for DFMs only after some idle time is because DFMs are only transformed/installed as APKs during idle time once the app is not running. This is because Android
immediately then the app would have to be closed/restarted after each DFM is installed.

For this reason DFMs are dynamically loaded using SplitCompat. SplitCompat still supports using native libraries using `System.loadLibrary()` however it cannot circumvent the Android lin

---

em...@gmail.com <em...@gmail.com> #17

Thanks for the clarification, but as i said it was worked before, the DFMs was transformed immediately and everything was worked as it should. Now they can't.

So, is there any chance to change this behavior to transform them directly as APKs in the first time and if it is really needed to restart the app OK, will notify users about that, if it doesn't affec

Adding everything on `install-time` DFM really makes app bigger for Android 10+ users in following.

---

ja...@google.com <ja...@google.com> #18

I think the best solution forward would be to temporarily add them as `install-time` modules and then work on a way of packaging the native libraries with JNI.

DFMs cannot be immediately transformed, `exec()` worked previously on Android P and lower because the Android framework had a different restriction on `exec()`. The Android framework
Unfortunately there is no solution from our side that we can offer to make `exec()` work for DFMs immediately. My best suggestion for your app is what I described in comment#11