application crash when using own libnativehelper.so

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

Hi, I am an Android developper and I experience a crash using my open-source application nova video player: https://github.com/nova-video-player/aos-AVP https://play.google.com/store/apps/

The issue and logs are available here: https://github.com/nova-video-player/aos-AVP/issues/457

It appears to be a regression: same apk runs well on Android 11 and crashes on Android 12 beta 3. The issue is confirmed both on the emulator `sdk_gphone64_x86_64-userdebug 12 SPB3.21(`

The crash happens when the application loads a local shared `libnativehelper.so` library and complains on Android 12 that it does not find symbol `jniGetFDFromFileDescriptor` present in t

Note that changing name of the library to `libnvpnativehelper` fixes the issue in my application.

To recap and answer the requested questions in the desired format:

- Are you an Android developer? Yes
- Which Android Developer Preview build are you using? SPB3.210618.013
- Is this a regression from Android 11 to 12? Yes
- What device are you using? (for example, Pixel 4 XL): Pixel 3 XL and emulator
- What are the steps to reproduce the problem? Download following application apk it crashes on Android 12 beta 3 and not 11
- Issue Category: NDK / platform
- What was the expected result? No crash in the application navigating in SMB tab
- Can you provide the API document where this expected behavior is explained? expected behavior is to load correctly the shared library `libnativehelper.so` and the one from local to the a
- What was the actual result? Crash because wrong library potentially.
- Relevant logcat output.

```
07-15 21:42:56.676 20686 20686 E AndroidRuntime: FATAL EXCEPTION: main
07-15 21:42:56.676 20686 20686 E AndroidRuntime: Process: org.courville.nova, PID: 20686
07-15 21:42:56.676 20686 20686 E AndroidRuntime: java.lang.UnsatisfiedLinkError: dlopen failed: cannot locate symbol "jniGetFDFromFileDescriptor" referenced
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at java.lang.Runtime.loadLibrary0(Runtime.java:1077)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at java.lang.Runtime.loadLibrary0(Runtime.java:998)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at java.lang.System.loadLibrary(System.java:1656)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at com.archos.filecorelibrary.samba.SambaDiscovery.<clinit>(SourceFile:564)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at com.archos.mediacenter.video.browser.filebrowsing.network.SmbBrowser.SmbRootFragment.onAttach(Source
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.Fragment.performAttach(SourceFile:2922)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentStateManager.attach(SourceFile:464)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentStateManager.moveToExpectedState(SourceFile:275)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentManager.executeOpsTogether(SourceFile:2189)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentManager.removeRedundantOperationsAndExecute(SourceFile:2100)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentManager.execPendingActions(SourceFile:2002)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentManager.dispatchStateChange(SourceFile:3138)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentManager.dispatchActivityCreated(SourceFile:3072)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentController.dispatchActivityCreated(SourceFile:251)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.fragment.app.FragmentActivity.onStart(SourceFile:501)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at androidx.appcompat.app.AppCompatActivity.onStart(SourceFile:246)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.Instrumentation.callActivityOnStart(Instrumentation.java:1455)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.Activity.performStart(Activity.java:8076)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.ActivityThread.handleStartActivity(ActivityThread.java:3658)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.servertransaction.TransactionExecutor.performLifecycleSequence(TransactionExecutor.java
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.servertransaction.TransactionExecutor.cycleToPath(TransactionExecutor.java:201)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.servertransaction.TransactionExecutor.executeLifecycleState(TransactionExecutor.java:173
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.servertransaction.TransactionExecutor.execute(TransactionExecutor.java:97)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.ActivityThread$H.handleMessage(ActivityThread.java:2202)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.os.Handler.dispatchMessage(Handler.java:106)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.os.Looper.loopOnce(Looper.java:201)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.os.Looper.loop(Looper.java:288)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at android.app.ActivityThread.main(ActivityThread.java:7829)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at java.lang.reflect.Method.invoke(Native Method)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:548)
07-15 21:42:56.676 20686 20686 E AndroidRuntime:       at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:982)
```

- Link to captured Android bug report:
  - apk used to perform the test: https://drive.google.com/file/d/1kIapXE82J93MhETnIgYx19bGEHMFz4OX/view?usp=drive_web (Video-universal-API_21+-release.apk)
  - associated bugreport https://drive.google.com/file/d/1RXcg7lWHG6nIiC09tTgdrbSSkWJwd3by/view?usp=drive_web (bugreport-crosshatch-SPB3.210618.013-2021-07-18-18-05-43.zi

✓ Links (11)

Links (11)

"Hi, I am an Android developper and I experience a crash using my open-source application nova video player: https://github.com/nova-video-player/aos-AVP https://play.google.com/store/apps/deta
"Hi, I am an Android developper and I experience a crash using my open-source application nova video player: https://github.com/nova-video-player/aos-AVP https://play.google.com/store/apps/deta
"The issue and logs are available here: https://github.com/nova-video-player/aos-AVP/issues/457"
"apk used to perform the test: https://drive.google.com/file/d/1kIapXE82J93MhETnIgYx19bGEHMFz4OX/view?usp=drive_web (Video-universal-API_21+-release.apk)"
"associated bugreport https://drive.google.com/file/d/1RXcg7lWHG6nliC09tTgdrbSSkWJwd3by/view?usp=drive_web (bugreport-crosshatch-SPB3.210618.013-2021-07-18-18-05-43.zip)"

See all related links

COMMENTS

ad...@google.com <ad...@google.com>

Assigned to ad...@google.com.

ad...@google.com <ad...@google.com> #2

We have passed this to the development team and will update this issue with more information as it becomes available.

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

Reassigned to ar...@google.com.

> Note that changing name of the library to libnvpnativehelper fixes the issue in my application.

yeah, from the dynamic linker perspective i think this is working as intended, and you were just getting lucky before: if you have a library with the same soname as a library that's part of the O

looks like the ART folks renamed these in S as part of the work to make ART a mainline module:

```
# NDK API for libnativehelper.
AFileDescriptor_create;
AFileDescriptor_getFd;
AFileDescriptor_setFd;
```

so on the bright side, these are now public APIs going forward, but unfortunately we didn't do this 10 years ago so you could already rely on them being a public part of the OS :-(

i'll pass this bug to them, but i expect they'll say "working as intended", and the safe fix is to ensure that your soname for every shared library is globally unique. personally i'd include my FQDN

en...@google.com <en...@google.com> #4

(technically, what i said about sonames is only true from API 23... if you still care about older API levels, it's the *filename* rather than the soname that matters. in practice those are typically th
ndk-developers.md#correct-soname_path-handling-available-in-api-level-23)

ot...@google.com <ot...@google.com> #5

Status: Won't Fix (Intended Behavior)

Yes, this is working as intended.

There was a goal of limiting the APIs exposed by libnativehelper as it now ships in the ART module and exposes those APIs to the platform and the module can be independently updated fro

`jniGetFDFromFileDescriptor` pokes around private fields and ideally would be covered by hiddenapi but the interaction between native code and hiddenapi is sufficiently costly that it has be relied on for API level >= 30.

ph...@gmail.com <ph...@gmail.com> #6

I don't really understand, is libnativehelper.so in APK's linker namespace? I was under the impression it wasn't, so the only libnativehelper in APK's search path is APK's own libnativehelper, sc

As for not using jniGetFDFromFileDescriptor, ok, but please do tell what is the method recommended that works from sdk 17 to 31.

Also, I'm quite surprised by this answer, since I thought Android had a strict "don't break existing apps" policy, since this same APK works fine on API level 30, but anyway, it's fine for us to fix

ot...@google.com <ot...@google.com>

Status: Assigned (reopened)

ot...@google.com <ot...@google.com> #7

Status: Won't Fix (Intended Behavior)

In S, `libnativehelper.so` is part of the Android NDK and therefore available to apps. It exposes two groups of methods now for apps:

```
# NDK Methods (https://developer.android.com/ndk/reference/group/file-descriptor)
AFileDescriptor_create
AFileDescriptor_getFd
AFileDescriptor_setFd
```

```
# JNI Invocation API (response to https://github.com/android/ndk/issues/1320)
# https://docs.oracle.com/javase/8/docs/technotes/guides/jni/spec/invocation.html
JNI_CreateJavaVM            // NB always fails for apps, only one runtime per process
JNI_GetDefaultJavaVMInitArgs
JNI_GetCreatedJavaVMs
```

Before S, `jniGetFDFromFileDescriptor` et al were not supported API, it was a bug / hole being able to access these at all.

> As for not using jniGetFDFromFileDescriptor, ok, but please do tell what is the method recommended that works from sdk 17 to 31.

- API level < 31, you can use `jniGetFDFromFileDescriptor()` or your own implementation.
- API level >= 31, use `AFileDescriptor_getFd()`.

If the opportunity arises, we may implement a compat library to make this easier.

As Android releases go forward, we try to move apps on to supported APIs. For example, see Restrictions on non-SDK interfaces. These changes unavoidably cause some inconvenience t