Android Public Tracker 290806736

ANR WebViewChromium.init

+1 6 Hotlists (3) Mark as Duplicate

Comments (9) Dependencies Duplicates (0) Blocking (0) Resources (2)

Infeasible Bug P3 + Add Hotlist

STATUS UPDATE No update yet. Edit

DESCRIPTION ra...@haystack.tv created issue #1

We have been getting ANR reports from Google Play console, which point to following stack trace during instantiation of WebView.
(~93% of these ANRs were reported on Android 11 across varying set of Android TV devices)
The number of occurrences is in couple hundreds, so we do want to flag this and get some help in analyzing this together.

We cannot attach bugreport as we cannot replicate the issue, and these occurrences have happened to other users and reported on Play console.

ANR trace , main thread snapshot (entire trace is included in attach.)

```
"main" tid=1 Native
  #00  pc 0x000000000005e2e8  /apex/com.android.runtime/lib/bionic/libc.so (syscall+28)
  #01  pc 0x0000000000130bc3  /apex/com.android.art/lib/libart.so (art::ConditionVariable::WaitHoldingLocks(art::Thread*)+82)
  #02  pc 0x00000000002d68cf  /apex/com.android.art/lib/libart.so (art::JNI<false>::NewString(_JNIEnv*, unsigned short const*, int)+402)
  #03  pc 0x000000000227951d  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #04  pc 0x0000000001a2a61f  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #05  pc 0x000000000227925d  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #06  pc 0x0000000002f8c443  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #07  pc 0x0000000000d38681  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #08  pc 0x0000000000d38a9b  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #09  pc 0x0000000000d38a79  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #10  pc 0x0000000000d1c391  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so
  #11  pc 0x0000000000d1b033  /data/app/~~7r7jSVxDaHD-ldq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so (Java_J
  at J.N.MFiR_zHY (Native method)
  at org.chromium.android_webview.AwContents.<init> (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:375)
  at com.android.webview.chromium.k.run (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:111)
  at GC0.b (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:20)
  at FC0.run (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:3)
  at org.chromium.base.task.PostTask.d (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:11)
  at GC0.a (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:28)
  at com.android.webview.chromium.WebViewChromiumFactoryProvider.a (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:3)
  at com.android.webview.chromium.WebViewChromium.init (chromium-TrichromeWebViewGoogle.aab-stable-<US_SOCIAL_SECURITY_NUMBER>:332)
  at android.webkit.WebView.<init> (WebView.java:435)
  at android.webkit.WebView.<init> (WebView.java:355)
  at android.webkit.WebView.<init> (WebView.java:337)
  at android.webkit.WebView.<init> (WebView.java:324)
  at android.webkit.WebView.<init> (WebView.java:314)
  at com.haystack.android.tv.hstl.OverlaysWebview.<init> (OverlaysWebview.java:103)
  at com.haystack.android.tv.hstl.HstlSetup.initWebView (HstlSetup.kt:53)
  at com.haystack.android.tv.hstl.HstlSetup.init (HstlSetup.kt:43)
  at com.haystack.android.tv.ui.activities.MainActivity.setupHstl (MainActivity.java:365)
  at com.haystack.android.tv.ui.activities.MainActivity.onCreate (MainActivity.java:228)
  at android.app.Activity.performCreate (Activity.java:8000)
  at android.app.Activity.performCreate (Activity.java:7984)
  at android.app.Instrumentation.callActivityOnCreate (Instrumentation.java:1309)
  at android.app.ActivityThread.performLaunchActivity (ActivityThread.java:3422)
  at android.app.ActivityThread.handleLaunchActivity (ActivityThread.java:3601)
  at android.app.servertransaction.LaunchActivityItem.execute (LaunchActivityItem.java:85)
  at android.app.servertransaction.TransactionExecutor.executeCallbacks (TransactionExecutor.java:135)
  at android.app.servertransaction.TransactionExecutor.execute (TransactionExecutor.java:95)
  at android.app.ActivityThread$H.handleMessage (ActivityThread.java:2066)
  at android.os.Handler.dispatchMessage (Handler.java:106)
  at android.os.Looper.loop (Looper.java:223)
  at android.app.ActivityThread.main (ActivityThread.java:7656)
  at java.lang.reflect.Method.invoke (Native method)
  at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run (RuntimeInit.java:592)
  at com.android.internal.os.ZygoteInit.main (ZygoteInit.java:947)
```

Haystack_WebView_Init_ANR.rtf
25 KB Download

✓ Links (2)

Links (2)

“For steps to capture a bug report, please refer: https://developer.android.com/studio/debug/bug-report#bugreportdevice”
“We have found this post that seems related to our case.”

COMMENTS

ra...@google.com <ra...@google.com>

Assigned to cl...@google.com.

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

Reassigned to ra...@google.com.

This does not have a piece of information needed to symbolize the WebView stack trace (missing buildId). If its from the Play console, its probably cleaned up and we cant get it.

From what is available, the ANR happens when webview is starting up.

Without a full bug report we cant progress much further. Perhaps you have reproduction steps or a bug report from a user with the same issue?

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

Thanks, For us to further investigate this issue, please provide the following additional information:

Steps to reproduce
What steps are needed to reproduce this issue?

Android bug report (to be captured after reproducing the issue)
For steps to capture a bug report, please refer: https://developer.android.com/studio/debug/bug-report#bugreportdevice

Alternate method
Navigate to “Developer options”, ensure “USB debugging” is enabled, then enable “Bug report shortcut”. Capture bug report by holding the power button and selecting the “Take bug report” o

Note: Please upload the bug report to google drive and share the folder to android-bugreport@google.com, then share the link here.

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

So hi my name is Windell Hitlallthe developer of this account. I did not
allow any web view Chromium.init. to my phone or account.

On Tue, Jul 18, 2023, 3:03 AM <buganizer-system@google.com> wrote:

- Show quoted text -

ra...@haystack.tv <ra...@haystack.tv> #5

Regarding the bugreport, as mentioned in the initial description we have not been able to replicate this on our side, and the only reports of the ANR trace we have is from the Google Play cons
The issue seems to happen immediately on launching Haystack News app on AndroidTV/GoogleTV , and we have seen most of the occurrences on Android 11 devices.

Apart from the root cause of the ANR itself, based on the ANR trace can we get some guidance on if there are any things we can do to avoid the ANR from perspective of WebView. We canno

ra...@google.com <ra...@google.com>

Reassigned to pt...@google.com.

is...@haystack.tv <is...@haystack.tv> #6

from Firebase this is happening across the following device types. Hard to reproduce in an isolated environment.

Wed Aug 02 2023 12:31:21 GMT-0700 (Pacific Daylight Time).png
28 KB View Download

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

Reassigned to to...@google.com.

@torne maybey you know some trick to find the missing buildId to symbolize this? I dont see anything actionable otherwise

ju...@haystack.tv <ju...@haystack.tv> #8

Since the ANR is still occurring we can add any recommendation you may have to our incoming releases, to gather more information.

Please, any insights on how to get *user properties* or the `buildId` on our **ANR details** in the Play Store Console?

We have found this post that seems related to our case.

- They preload the Chromium engine when the main thread gets idle, to alleviate the WebView init when required.
- Do you think it's worth trying this way, or is it too risky dealing with internal APIs?

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

Status: Won't Fix (Infeasible)

The Play Store console doesn't provide that information - there's nothing else you can provide unless you can reproduce the issue.

We don't actually need the build ID to symbolize this, but symbolizing it is unfortunately not going to provide any useful information; ANRs during init are basically never actionable and in this
initialization successfully if given more time - there's just been too much happening on the UI thread, which can happen for a vast number of reasons and can't be diagnosed from an ANR trac
what matters is *everything* that was happening over the last 5 seconds, including before WebView initialization started.

At the end of the post I symbolized this anyway (mostly for reference for other folks on my team) but unfortunately it is indeed not useful.

The post you link to is simply wrong that the static methods do not initialize Chromium completely; `getDefaultUserAgent` *does* call `ensureChromiumStartedLocked`. There is no need to a
use any WebView APIs immediately during startup to function, and you're able to wait until later instead of touching WebView during `Activity.onCreate`, then yes, you can wait until the res

The most efficient way to initialize it is to call `WebSettings.getDefaultUserAgent()` on a background thread when you are ready, which will perform a part of the initialization on that backg
a significant amount of work to be done on the UI thread even with this approach, but it's slightly reduced and you can do it whenever you choose. Once you've called this, calling any other W

However to get the most benefit from this, you need to be very careful throughout your app to avoid calling any WebView-related API (basically the entirety of `android.webkit.*`) unnecessa
that also call WebView APIs and may have their own requirements about when this happens.

If you can reliably avoid initializing WebView while your app is starting up then the chance of ANRs occurring will be reduced (though it's hard to predict by how much) - creating an instance c
thread tasks which are doing *all* of these things:

1. Initializing your app's `ContentProvider`s (if you have any; notably some 3P libraries define one just to run their initialization code very early)
2. Initializing your app's `Application` class (`attachBaseContext` and `onCreate` and related) - also often involves calling 3P library init
3. Initializing your initial `Activity` (`onCreate` and related)
4. Loading the WebView implementation code
5. Running WebView's global initialization
6. Running the initialization for an individual instance of WebView (much cheaper than the global init but still measurable)

and often on slower devices the total time required to do all of these steps even in the *best* case can be 2 seconds or more, which means it doesn't actually take a lot to push it over the ANR t

If you avoid WebView APIs during steps 1-3 and delay it until later, othe UI thread events can be processed in between, and the total time for all the steps is no longer relevant for ANRs - only
background thread using `getDefaultUserAgent` and wait until that's complete before using it on the UI thread, then you *also* remove step 4 from consideration.

We're working on a new API to explicitly initialize WebView with documented behavior and specific recommendations for how to use it, because the information here is not well-explained els
give approximately the same benefits, it's just much less clear what's going on.

For Paulius/other internal folk's reference, when we have native stack frames like:

`/data/app/~~7r7jSVxDaHD-1dq8ir4HTw==/com.google.android.trichromelibrary_573519630-W6oCqA_LF-JeQiOFrE8ThA==/base.apk!libmonochrome.so`

the 573519630 is the versionCode of TrichromeLibrary, which is the same as the versionCode of WebView (i.e. this is a 114.0.5735.196 build for arm32), and the native library name is `libmo`
`114.0.5735.196/arm/Monochrome_symbols.zip`). The WebView native stack frames here symbolize to:

```
0x0227951d: _JNIEnv::NewString(unsigned short const*, int) at ./../../third_party/android_ndk/toolchains/llvm/prebuilt/linux-x86_64/sysroot/usr/include/jn
 (inlined by) (anonymous namespace)::ConvertUTF16ToJavaStringImpl(_JNIEnv*, base::BasicStringPiece<char16_t, std::Cr::char_traits<char16_t> > const&) at .
 (inlined by) base::android::ConvertUTF8ToJavaString(_JNIEnv*, base::BasicStringPiece<char, std::Cr::char_traits<char> > const&) at ./../../base/android/j
0x01a2a61f: base::android::(anonymous namespace)::GetClassInternal(_JNIEnv*, char const*, _jobject*) at ./../../base/android/jni_android.cc:55 (discrimina
0x0227925d: base::android::GetClass(_JNIEnv*, char const*) at ./../../base/android/jni_android.cc:157
 (inlined by) base::android::LazyGetClass(_JNIEnv*, char const*, std::Cr::atomic<_jclass*>*) at ./../../base/android/jni_android.cc:191
0x02f8c443: org_chromium_components_viz_service_frame_1sinks_ExternalBeginFrameSourceAndroid_clazz(_JNIEnv*) at ./gen/jni_headers/components/viz/service/s
 (inlined by) viz::Java_ExternalBeginFrameSourceAndroid_Constructor(_JNIEnv*, long long, float) at ./gen/jni_headers/components/viz/service/service_jni_he
 (inlined by) viz::ExternalBeginFrameSourceAndroid::ExternalBeginFrameSourceAndroid(unsigned int, float, bool) at ./../../components/viz/service/frame_sin
0x00d38681: android_webview::RootBeginFrameSourceWebView::RootBeginFrameSourceWebView() at ./../../android_webview/browser/gfx/begin_frame_source_webview.
 (inlined by) base::NoDestructor<android_webview::RootBeginFrameSourceWebView>::NoDestructor<>() at ./../../base/no_destructor.h:88
 (inlined by) android_webview::RootBeginFrameSourceWebView::GetInstance() at ./../../android_webview/browser/gfx/begin_frame_source_webview.cc:129
0x00d38a9b: android_webview::BrowserViewRenderer::UpdateBeginFrameSource() at ./../../android_webview/browser/gfx/browser_view_renderer.cc:584
0x00d38a79: android_webview::BrowserViewRenderer::BrowserViewRenderer(android_webview::BrowserViewRendererClient*, scoped_refptr<base::SingleThreadTaskRun
0x00d1c391: android_webview::AwContents::AwContents(std::Cr::unique_ptr<content::WebContents, std::Cr::default_delete<content::WebContents> >) at ./../../
0x00d1b033: android_webview::JNI_AwContents_Init(_JNIEnv*, long long) at ./../../android_webview/browser/aw_contents.cc:421 (discriminator 4)
 (inlined by) Java_J_N_MFiR_1zHY at ./gen/jni_headers/android_webview/browser_jni_headers/AwContents_jni.h:48 (discriminator 4)
```

So, yeah - this isn't blocked, it's just in the middle of setting up the WebView's internal state and would continue to make progress if the ANR hadn't killed the app.