Man-in-the-Disk: Android Apps Exposed via External Storage

By deugenio

Published: 2018-08-12 · Archived: 2026-04-20 02:07:09 UTC

Research By: Slava Makkaveev

Recently, our researchers came across a shortcoming in the design of Android's use of storage resources. Careless use of External Storage by applications may open the door to an attack resulting in any number of undesired outcomes, such as silent installation of unrequested, potentially malicious, apps to the user's phone, denial of service for legitimate apps, and even cause applications to crash, opening the door to possible code injection that would then run in the privileged context of the attacked application.

These Man-in-the-Disk attacks are made possible when applications are careless about their use of External Storage, a resource that is shared across all applications and does not enjoy Android's built-in Sandbox protection. Failing to employ security precautions on their own leaves applications vulnerable to the risks of malicious data manipulation.

In this post we will show how one WRITE_EXTERNAL_STORAGE permission can make it possible for an Android app to silently execute privileged code and install other apps.

Overview

There are two types of Android device storage:

- **Internal Storage:** This is an app's built-in memory which each application uses separately and is segregated by the Android Sandbox.
- **External Storage:** This is a partition in the Internal Storage or SD-card which is shared by all applications.

Before going any further, though, let's take a deeper look at Android's External Storage.

Many high privileged systems and popular apps use Android's External Storage as a temporary buffer when downloading data from the Internet. What's more, many apps hold working data permanently in the External Storage since the device can often be limited by using the Internal space only.

As all third-party apps installed on the device can observe data transferred by other apps via the External Storage, as well as overwrite a target file, various possibilities are available for a Man-in-the-Disk attack.

Firstly, a malicious app can crash a target app's native library which parses a storage located data file. This crash can lead to a code execution in the context of a privileged or popular app. Another attack is to hijack an app update flow, overwrite APK files and silently install a rogue app instead of a legitimate one. DoS and EoP vulnerabilities are also possible.

The Fuzzing Tool

Once we encountered an app that saves either data or configuration using the External Storage, we decided to use fuzzing techniques to try and search for vulnerabilities within the app. However, there are only Black Box Fuzzing available for Android, which would not be an efficient or sufficient, and therefore not really applicable, for our purposes. So in order to successfully test for vulnerabilities in the target apps our research team adapted an existing fuzzing tool (AFL) to run native Android libraries over the QEMU Emulator. This allowed us to run 'Grey Box Fuzzing'.

This tool consists of several parts including patched version of Android Runtime (libart.so), dalvikvm tool, GLib, QEMU and AFL. The assembly of these tools eventually allows to fuzz a user custom lib wrapped in a DEX file on an actual Android device or an ARM device emulator. The execution flow is as follows: AFL Fuzz Engine → QEMU ARM CPU emulator → dalvikvm tool → Android Runtime + adapter DEX.

In the case of some of the apps below, for example Yandex or Google Translate, the ‘Grey Box’ fuzzing allowed us to mutate the translation package files in an efficient way until a lib crash was reached. In one case it took just 15 minutes to reach the crash, with the other it took half a day.

Examples of a Man-in-the-Disk Attack

Let’s take several examples of how a Man-in-the-Disk attack can be used. These vulnerabilities show that even the biggest Android vendors and developers make dangerous mistakes when working with the External Storage.

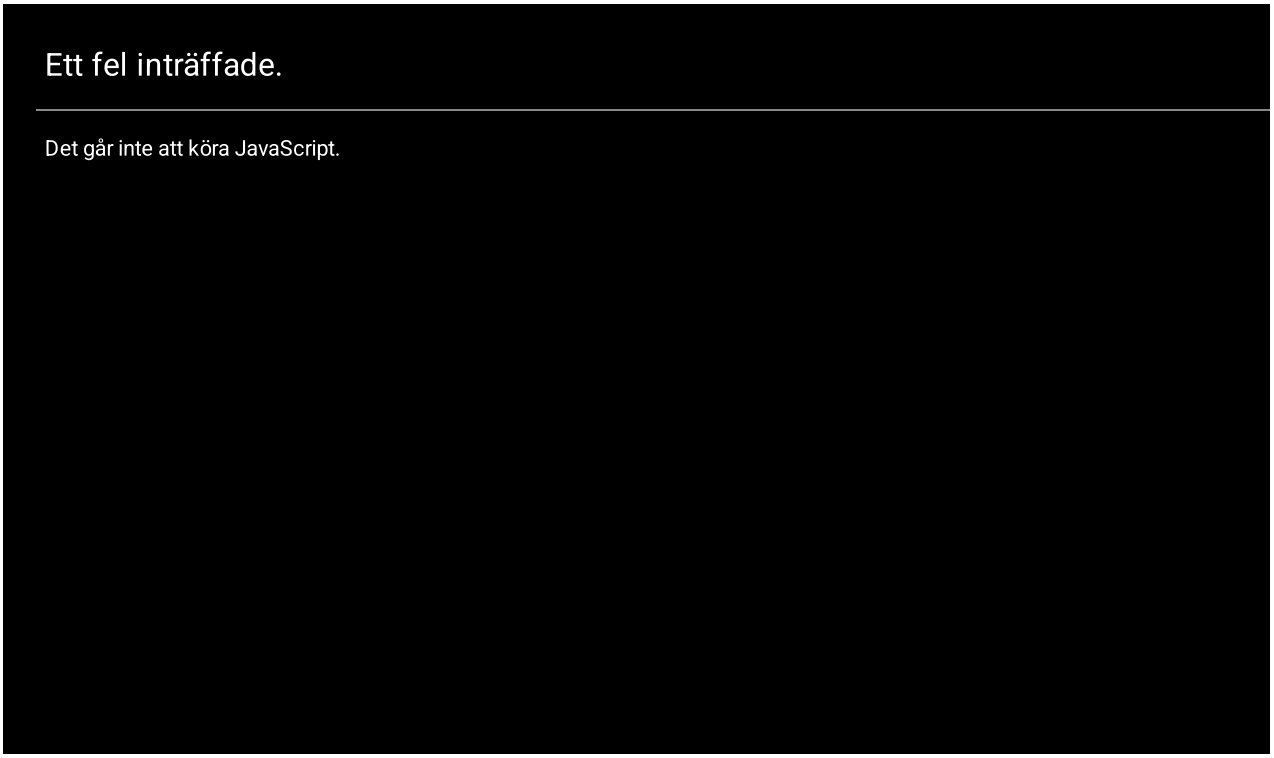
Google Translate (com.google.android.apps.translate)

Google Translate app downloads and holds off-line mode translation packages on external storage – */storage/emulated/0/Android/data/com.google.android.apps.translate/files/olpv3/v5/25/r11/* . These data files are processed by *libtranslate.so* app’s native lib upon entering the first letter for translation. The attacker can then generate a rogue binary, overwrite original files located in the external storage and crash the lib. This scenario does not require any file observation from the attacker side.

Video: A demonstration of a Man-in-the-Disk attack against Google Translate.

Google Voice Assistance (com.google.android.googlequicksearchbox)

The app downloads offline speech recognition languages to External Storage (*/storage/emulated/0/Android/data/com.google.android.googlequicksearchbox/files/download_cache/)* as an archive and then decompresses machine learning related binaries in the app’s Internal Storage directory (*app_g3_modes)*. The attacker can then overwrite downloaded files and crash *libgoogle_speech_jni.so* app’s native lib.



Video: A demonstration of a Man-in-the-Disk attack against Google Voice Assistant.

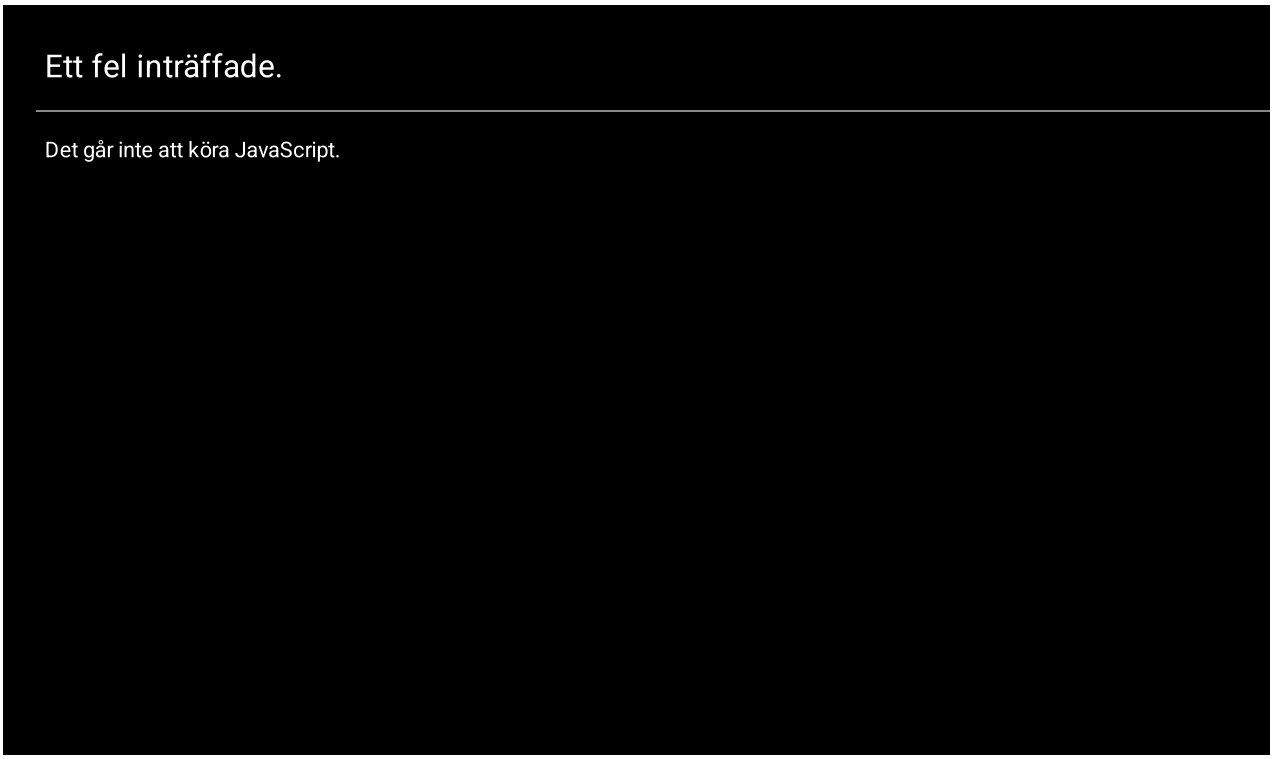
Xiaomi Browser (com.android.browser package)

The Xiaomi Browser downloads a self-updating APK file to External Storage */storage/emulated/0/Android/data/com.android.browser/files/update*, verifies SHA1 and installs the APK. The attacker can then overwrite the updated file right after verification since verification and installation are two separated and manually invoked actions.

Video: A demonstration of a Man-in-the-Disk attack against Xioami Browser.

Yandex Translate (ru.yandex.translate)

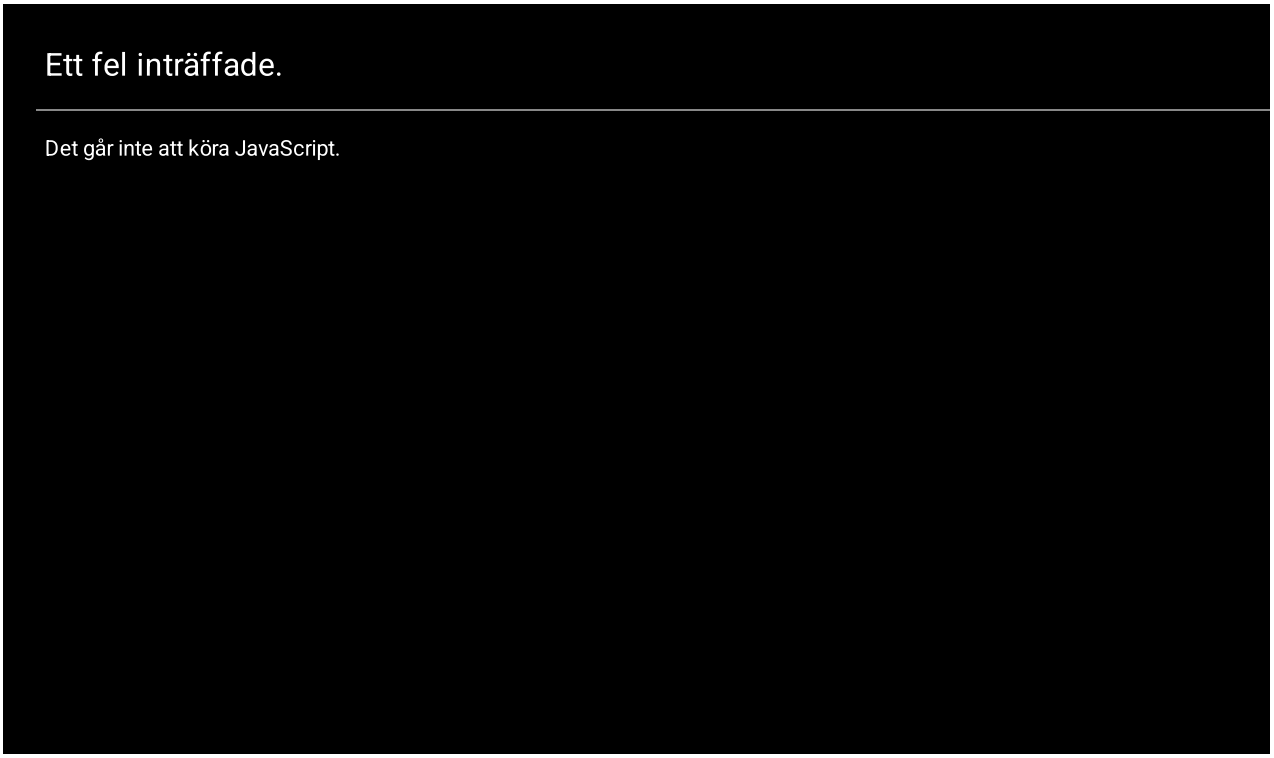
The Yandex Translate app downloads and holds offline mode translation packages on External Storage – */storage/emulated/0/Android/data/ru.yandex.translate/files/offline/translation/* . The same process follows as was seen with Google Translate, but in this case a different *libmobile-android.so* native lib is under attack.



Video: A demonstration of a Man-in-the-Disk attack against Yandex Translate.

Yandex Search (ru.yandex.searchplugin)

Yandex Search downloads and holds offline mode search packages on External Storage – */storage/emulated/0/Android/data/ru.yandex.searchplugin/files/edge_search_dicts/*. The attacker can then generate a rogue binary, overwrite the original file located in External Storage and crash *liboffline_search-data_reader.so* app's native lib.



Video: A demonstration of a Man-in-the-Disk attack against Yandex Search.

Source: https://research.checkpoint.com/androids-man-in-the-disk/