February 1st, 2024
Christoph T. Hebeisen - Director, Security Intelligence Research, Lookout, Inc.
Statement for the Record before the U.S.-China Economic and Security Review Commission
Current and Emerging Technologies in U.S.-China Economic and National Security Competition

Software Development Kits

Software Development Kits (SDKs) are software packages that enable a developer to include specific functionality into their code without having to develop and maintain that functionality themselves. For example, a developer who wants to accept credit-card payments in their iOS or Android app may include the Stripe SDK[1]. If social networking functionality through Facebook is desired, the developer may leverage Meta's Facebook SDK[2] and if they want to monetize their app by displaying advertising to app users, they may deploy Google Mobile Ads SDK[3]. The functionality provided by SDKs is invoked by the code of the app through an Application Programming Interface (API).

The abstraction of implementation details of functionality that is not part of the core purpose of an app is beneficial to app developers. It avoids re-implementation of the same functionality for many apps and the reuse of an SDK in a multitude of scenarios usually results in higher-quality code, at least in the case of well-maintained SDKs.

Much of the discussion in this statement applies to SDKs in general. However, this panelist's expertise is in mobile systems (Android and iOS) and examples as well as technical details and considerations below are specific to mobile apps and mobile SDKs.

SDKs as a Potential Security Threat

While not having to know or understand the implementation details of an SDK is a great advantage, SDKs contain code that is not developed or controlled by the creator of the app. The user may choose to trust an app's developer based on their reputation and install the app on their device. However, they have no way of knowing which SDKs developers are utilizing in the app.

SDKs therefore provide an opportunity for a malicious actor to place code inside third-party apps - either by creating a useful SDK with well-hidden malicious functionality already included or by adding malicious functionality to an already existing, well-established SDK. Note that SDKs created by Chinese software companies and developers are well-represented in the mobile app space.

[1] https://stripe.com/docs/libraries/android
[2] https://developers.facebook.com/docs/ios/
[3] https://developers.google.com/admob/android/sdk

The creators of SDKs are usually software companies, which act primarily in their own business interest. If an SDK contains malicious code and that code is discovered, it is likely to reduce their future revenue. As a result, they are unlikely to add malicious code to their SDK voluntarily (unless there is significant business benefit to them - see examples below). However, a nation state might be able to compel or pay a software company to integrate malicious code to spy on users or perform other attacks such as Denial of Service (DoS).

However, adding malicious code to the SDK may not be necessary to gain access to some sensitive pieces of data. Much of the data collected by an SDK legitimately is sent to cloud infrastructure. For example, an in-app advertising provider may decide to display ads based on the location of the user as well as their past buying behavior. For this purpose, they need a way to identify the user as well as their geolocation. An advertiser may have a legitimate interest in both pieces of data but the use of the same data for any other purposes could well be considered a violation of privacy and - in the hands of an adversary - a serious risk. Similarly, billing or analytics SDKs may, in the normal course of operations, collect sensitive information about a user. Neither the user nor the creator of the app using a third-party SDK have a way to verifiably determine who can access the collected data and for what purpose. Examples such as TikTok's tracking of journalists[4] and Cambridge Analytics's abuse of Facebook data[5] demonstrate how data originally collected for non-malicious reasons can be used for spying and political manipulation purposes.

Operations of an SDK are only limited by the constraints the operating system imposes on the host application via the app sandbox. For example, if an application has the permission to access the microphone or the precise location of the device, the SDK code has the same access, even if it is not required for its stated functionality. And while mobile operating systems prevent apps from accessing other apps' private data, an included SDK has full access to the private data of the app. As a result, app developers have to trust an SDK to be "well behaved."

Examples of Malicious Use of SDKs, Code, and Data

Despite the risk of negative business impacts, multiple Chinese companies have been found to use malicious code to further their interests:

- In 2019, the BeiTaAd advertising SDK (created by CooTek) was found to aggressively flood users with ads[6] in violation of Google Play store rules.
- In 2020, it was discovered that MIntegral, a popular Chinese advertising SDK for Android and iOS, used malicious code to attribute clicks on advertising displayed by third-party SDKs to their SDK[7].

[4] https://www.nytimes.com/2023/03/17/us/politics/tik-tok-spying-justice-dept.html

[5] https://www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html

[6] https://www.zdnet.com/article/440-million-android-users-installed-apps-with-an-aggressive-advertising-plugin/

[7] https://snyk.io/blog/sourmint-malicious-code-ad-fraud-and-data-leak-in-ios/

- In 2023, PinDuoDuo was discovered to contain malicious code that exploited vulnerabilities in the Android operating system[8] in a version of the app distributed in some parts of mainland China. PinDuoDuo is PDD Holdings' app for the Chinese market. Their online shopping app for the North American market is Temu.

While CooTek's business appears to have suffered significantly after their SDK was banned from Google Play, the MIntegral SDK is still available and popular on U.S.-based advertising mediation platforms. At the time of writing, Temu is the top shopping app on both the Apple App Store and Google Play.

To the best of our knowledge, all of these cases were aimed at increasing revenue rather than spying on users.

Types of SDKs

While there are many different types of SDKs for mobile applications, the most commonly used ones fall into the following categories:

- Advertising
- Analytics
- Billing
- Storage
- Social Media
- Identity

In the following sections we will discuss common types of Chinese SDKs we observe integrated mobile apps. Data on presence of SDKs in apps presented in this document is for January 16th, 2024.

Advertising SDKs

Advertising SDKs are used by developers to generate revenue from their (usually free) apps. This means that in contrast to other types of SDKs, the integration of Ad SDKs is often not driven by technical requirements but by the expected payouts to app creators. This process is facilitated by so-called ad mediation platforms. Examples of popular US-based ad mediation platforms are

- Google AdMob[9]
- AppLovin MAX[10]
- Unity Ads[11]

[8] https://arstechnica.com/information-technology/2023/03/android-app-from-china-executed-0-day-exploit-on-millions-of-devices/

[9] https://developers.google.com/admob/android/choose-networks

[10] https://dash.applovin.com/documentation/mediation/android/mediation-adapters

[11] https://unity.com/products/mediation

With ad mediation, developers are incentivized to add as many ad network integrations as possible to maximize monetization potential. Integrating with an ad provider may also require installation[12, 13] of the upstream ad provider's SDK.
Thus, app developers may end up bundling Chinese ad SDKs even in cases where the transaction was initially brokered by a non-Chinese third party marketplace.

Popular China-based ad networks that integrate with mediation platforms are

- MIntegral[14] (partners with AdMob and AppLovin)
- Bytedance Pangle[15] ("Ad Network of TikTok for Business", partners with AdMob and AppLovin)

Unsurprisingly, ad SDKs are the most common Chinese SDKs found in Google Play top 150 grossing apps. MIntegral and Pangle were each found to be present in 20 of these 150 apps. MIntegral was also found in four of the 100 top grossing apps on the Apple App Store. While most of the Android apps are games, including titles such as Township, Solitaire, Lily's Garden and Heart of Vegas, the list also includes Grindr (social media / dating). The iOS apps are more varied, including ReelShort (streaming), GoodNovel (e-books) and CapCut (video editing).

Analytics SDKs

Analytics SDKs are used to collect information on how users engage with an app and on bugs and other issues with the app containing the SDK. Well-known Chinese analytics SDKs are Bugly (by Tencent)[16] and UMeng Analytics[17]. While Bugly was present in nine of the 150 top-grossing apps on Google Play and one of the top 100 apps in Apple's App Store, at the time of writing UMeng does not have a presence in the top apps of either store. All of the apps containing Bugly in the two stores' top apps were games.

Billing SDKs

While there are a considerable number of Chinese billing SDKs, these are usually not observed in mobile apps available in official app stores. Apps that are not available through these channels are unlikely to gain much traction in the U.S. and other western countries. This strongly entices app developers to distribute their apps through Google Play / the Apple App Store. Historically, these marketplaces require most categories of in-app purchases to be conducted using the marketplace's own billing system (App Store rules for in-app purchasing[18],

[12] https://developers.google.com/admob/android/choose-networks
[13] https://dash.applovin.com/documentation/mediation/max/get-started-with-max
[14] https://www.mintegral.com/
[15] https://www.pangleglobal.com/
[16] https://bugly.qq.com/v2/
[17] https://www.umeng.com/analytics
[18] https://developer.apple.com/in-app-purchase/

Google Play payment policy[19]). Notable Chinese billing SDKs include WeChat Pay[20] and Alipay[21].

Data and Cloud Infrastructure

As of January 16th, 2024, both MIntegral and Pangle ad SDKs appear to use U.S.-based endpoints for their cloud infrastructure (operated by Amazon CloudFront and Akamai, respectively). However, the geolocation of the endpoint the SDK connects to has no bearing on the location of the ultimate processing and storage location of the data. Even if the data is stored and processed in the US, foreign actors may have access to query or pull data remotely. The geolocation of the endpoints used by an SDK should therefore by no means be taken as an indication that the data is kept in the U.S. (or some other place).

Recommendations

Two classes of threats posed by SDKs were introduced in this document.

- Malicious code contained in an SDK
- Malicious use of the data collected by an SDK

Both types of threats are difficult to defend against but each type can be targeted through a number of mitigation measures.

Malicious Code / Behavior by an SDK

By design, app creators use third-party SDKs without exactly understanding their code. It is unrealistic to expect that individual developers fully understand all aspects of third-party code they use and determine if the code is behaving in a malicious manner. However, as described earlier, many app creators use advertising mediation platforms that integrate third-party ad networks. Bringing app developers and ad networks together is the core part of their business so mediation platforms may be in a better position to vet ad SDKs. Regulation holding mediation platforms responsible for malicious code found in ad SDKs supported by their platforms would create an incentive to ensure that ad networks they offer are well behaved. Note, however, that audits of large code bases such as ad SDKs are expensive and require specialized reverse engineers. If such measures were used specifically for SDKs from particular countries, they could be seen as a trade barrier.

The security of U.S. users from malicious code in SDKs could be further strengthened through the development of technical protections built into the the platforms on which the apps in question are executed. As an example, Google is developing SDK Runtime[22] for Android, which isolates SDKs from the host app, protecting private app data from being accessed by a rogue

[19] https://support.google.com/googleplay/android-developer/answer/10281818?hl=en
[20] https://developers.weixin.qq.com/doc/oplatform/en/Mobile_App/WeChat_Pay/Android.html
[21] https://global.alipay.com/docs/ac/dws2/mobilesdkintegrationguide
[22] https://developer.android.com/design-for-safety/privacy-sandbox/sdk-runtime

SDK. Similar core security developments as well as research into other privacy-enabling technologies should be encouraged and promoted.

Abuse of Collected Data

As explained above, it is essentially impossible for an outsider to determine how data is used once it reaches the SDK's backend infrastructure, who can access it and from where. While it is probably impossible to completely prevent abuses of such data by an adversary (short of a heavy-handed and impractical blanket ban on any foreign SDKs), limiting the collected data to the minimum required for the business purpose of the SDK can help mitigate the damage that would result from such abuses.

App privacy labels are provided by both Apple and Google in their app stores. Both platforms provide mechanisms to facilitate the (mandatory) inclusion of use of data by third-party SDKs in privacy labels. However, both approaches ultimately fall short of being a full, reliable solution since they ultimately rely on self reporting. Strengthening of privacy legislation and penalties for knowingly making false claims about app or SDK data use will strengthen users' control over the privacy of their data.

SDK Transparency

The current situation, in which users are largely unaware of the SDKs contained in their apps, leaves them without the ability to make decisions about which SDKs they are comfortable loading on their devices and which ones they want to avoid. Mandatory disclosure of SDKs developers use in their apps would empower users to make their own decisions. Especially if malicious behavior or a privacy problem with a particular SDK has been reported, being able to scrutinize the "ingredient list" would empower users to vote with their feet, creating cascading incentives for app developers, ad mediation platforms and SDK developers to protect users from spying and other malicious behaviors.