Security of Third-Party Keyboard Apps on Mobile Devices

By Lenny Zeltser

Published: 2014-09-24 · Archived: 2026-04-06 01:12:04 UTC

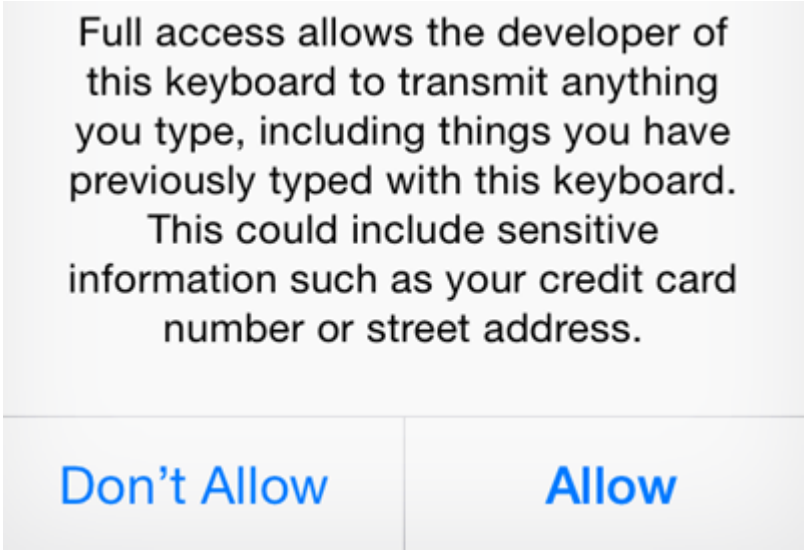
Third-party mobile keyboards with network access can capture keystrokes and transmit them to developers' servers, creating keylogger-like risks. Keyboard developers vary widely in their security transparency—some like Google clearly explain data practices while others provide only vague privacy policies that users cannot meaningfully evaluate.

Major mobile device platforms allow users to replace built-in keyboard apps with third-party alternatives, which have the potential to capture, leak and misuse the keystroke data they process. Before enabling the apps, their users should understand the security repercussions of third-party keyboards, along with the safeguards implemented by their developers.

Third-party keyboards received a boost of attention when Apple made it possible to implement such apps starting with iOS version 8, though this capability have existed on Android for a while. iOS places greater restrictions on keyboards than do Android operating systems; however, even Apple cannot control what keyboard developers do with keystroke data if users allow these apps to communicate over the network.

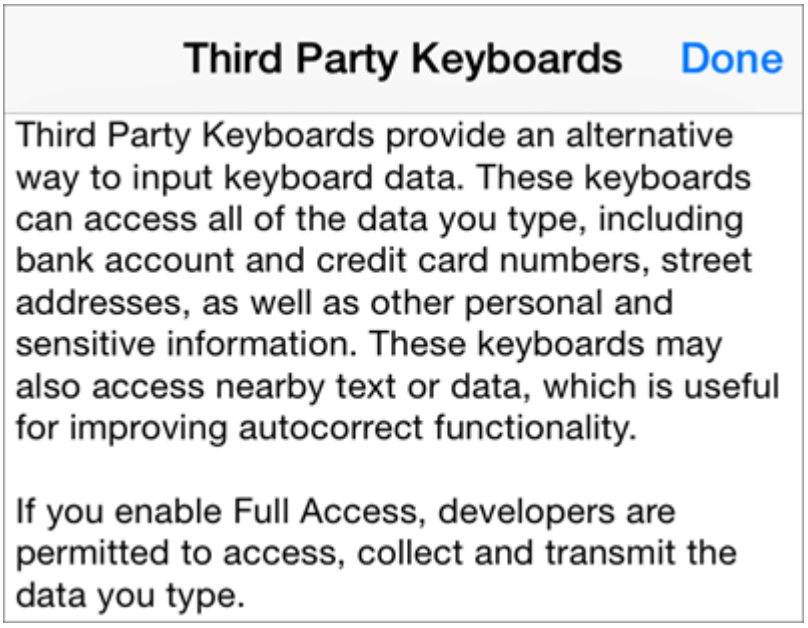
Granting Network Access to the Keyboard App

The primary security concerns related to keyboard applications are associated with their ability to transmit the user's keystrokes and potentially other sensitive data to developers' servers. On iOS, this is possible if the user grants the app "full access," as encouraged or required by the application.



The notion of "full access" is equivalent to the term "network access" that Apple's App Extension Programming Guide discusses when explaining how to build a custom keyboard. According to the guide, network access is disabled by default. Once enabled by the user, this capability allows the keyboard to access "Location Services

and Address Book, with user permission," "send keystrokes and other input events for server-side processing," and perform additional actions that would otherwise be unavailable.



A legitimate keyboard app might send keystrokes to its server to perform intelligent analysis that is best conducted in the "cloud," such as predict upcoming letters, store user typing patterns or perform large-scale lexicon analytics. Some third-party keyboards will function without their users granting them "full access;" however this will disable some of the features that might have drawn the users to the app in the first place.

Guidelines for Networked Keyboard Apps

Apple encourages keyboard developers to "establish and maintain user trust" by understanding and following "privacy best practices" and the associated iOS program requirements, guidelines and license agreements. For instance, if the keyboard sends keystroke data to the provider's server, Apple asks the developer to not store the data "except to provide services that are obvious to the user." Presumably, Apple's app review process catches blatant violations of such guidelines; however, once the keyboard transmits user data off the phone, there is little Apple can do to assess or enforce developers' security measures. (I haven't been able to locate keyboard-related security guidelines for the Android ecosystem, but I suspect they are not stronger than those outlined by Apple.) Users of third-party keyboards should not count on mobile OS providers to enforce security practices on server-side keyboard data processing. Instead, they should decide whether they trust the keyboard developer with their data, perhaps on the basis of their perception of the developer's brand and public security statements.

Potential for Inadvertent Data Leakage

Even if the keyboard developer intends to safeguard the data their app captures, there is the possibility and a bug in the implementation or design of the keyboard will inadvertently expose sensitive data. For example, in July 2016, some SwiftKey users reported that "that their keyboards were populating with other people's email addresses and searched phrases." This might have been caused by SwiftKey's language model improperly cross-pollinating the terms learned from one user to others. SwiftKey didn't offer much of an explanation for this issue beyond a terse statement. Another illustrative example of the risks posed by third-party keyboard comes from the

accidental leak of sensitive data about 31 million users of the ai.type mobile keyboard app. This breach reportedly contained risky information about the keyboard's users, which demonstrated the type of data third-party keyboards can capture and expose:

- Phone number, email address and full name
- Device characteristics, including model, IMSI and IMEI numbers
- Social network profile details
- Physical location information
- Address book contact records

Security and Privacy Statements from Keyboard Developers

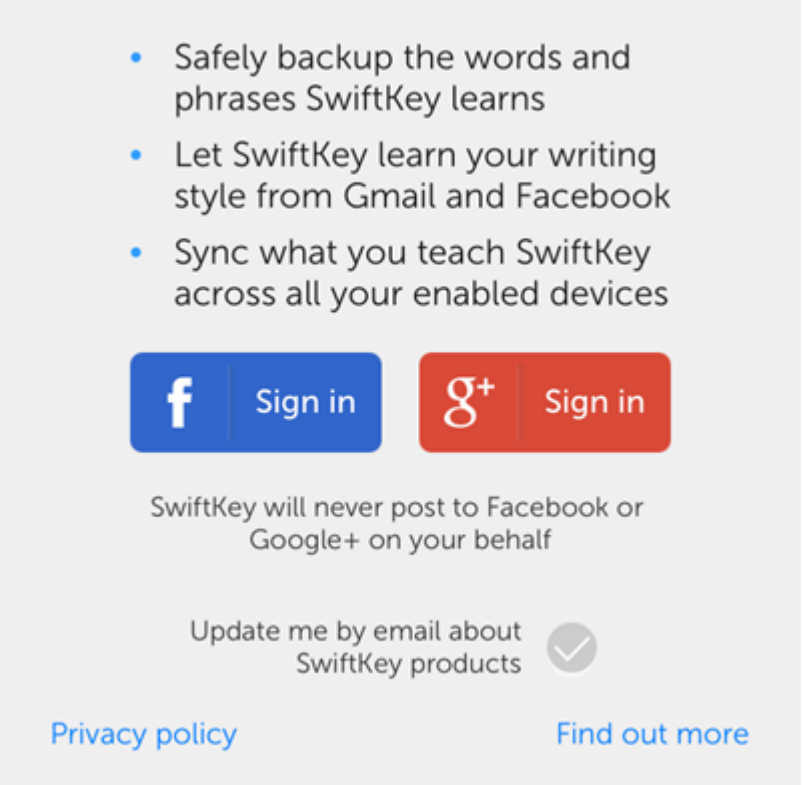
I looked at the security and privacy statements published by the developers of three popular keyboard apps that are available for mobile devices: SwiftKey, Gboard, Fleksy and Swype. Their developers differ in their explanation of how the safeguard users' data.

SwiftKey

SwiftKey, which is now owned by Microsoft, publishes a high-level overview of its data security and privacy practices. For users who opt into the SwiftKey Cloud feature of the product, the company collects "information concerning the words and phrases" that users utilize. The company calls this Language Modeling Data and explains that it is used to provide users with "personalization, prediction synchronization and backup." Fields that website or app developers designate as denoting a password or payment information are excluded from such collection.

SwiftKey states that the data transferred to SwiftKey Cloud is transmitted to their servers "over encrypted channels" and is stored in a "fully encrypted" manner. The company also points out that its data protection practices are governed by "stringent EU privacy protection laws" and mentions the (now weakened) Safe Harbor principles. The company also states that data of users who don't enable SwiftKey Cloud is not transferred out of their devices.

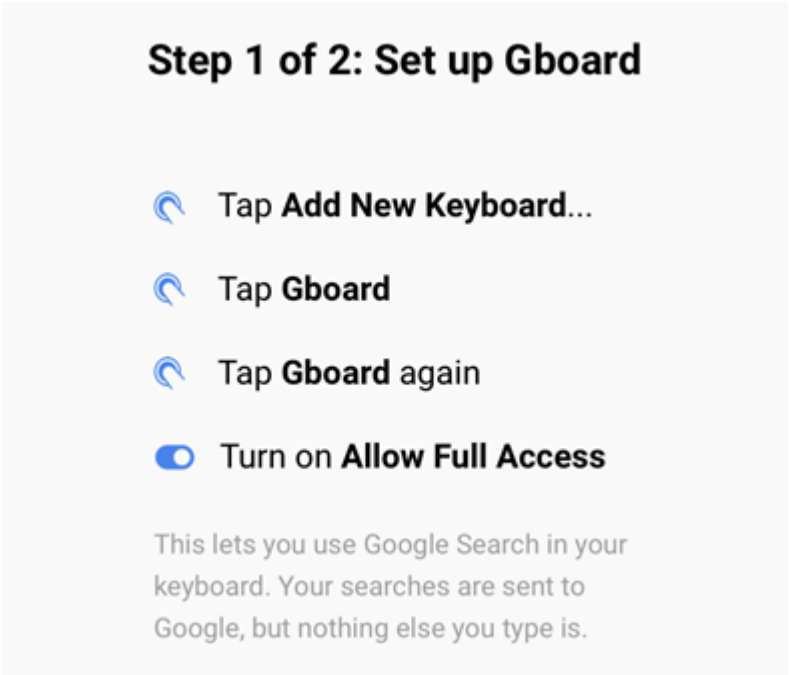
Users of the SwiftKey iOS app are instructed to grant the keyboard full access. Once this is accomplished, the user has the option of enabling SwiftKey Cloud "to enhance SwiftKey's learning" by signing in with their Facebook or Gmail credentials. To discover security implications of enabling this feature, the user had to dig into the details available by clicking "Privacy policy" and "Find out more" links.



Unfortunately, SwiftKey on iOS doesn't function at all until the user grants it full access. The user needs to trust SwiftKey that they won't send data off the mobile device unless SwiftCloud is enabled.

Gboard

The most attractive feature of Google's Gboard keyboard is probably the web-searching interface, which allows its user to submit search queries to Google without explicitly switching to a web browser.



The user needs to grant Gboard full access so that Google can receive the person's search terms. However, Google states that nothing else the person types is sent to Google. This is explained to the user from within the app during the setup process. Also, the statement is reiterated in the app's privacy policy, which is refreshingly easy to understand. It states:

> - "Your searches are sent to Google's web servers to give you search results.
> - Usage statistics are sent to let us know which features are used most often and to help us understand problems."

Glenn Fleishman confirmed that Gboard operates in a manner consistent with this statement. He used a network sniffer to examine what data Gboard transmits to Google's servers. When typing in Gboard, he observed "that no data was being sent at all between iOS and the Internet while I typed. I could then tap the G icon and perform searches, and watch data get sent back and forth."

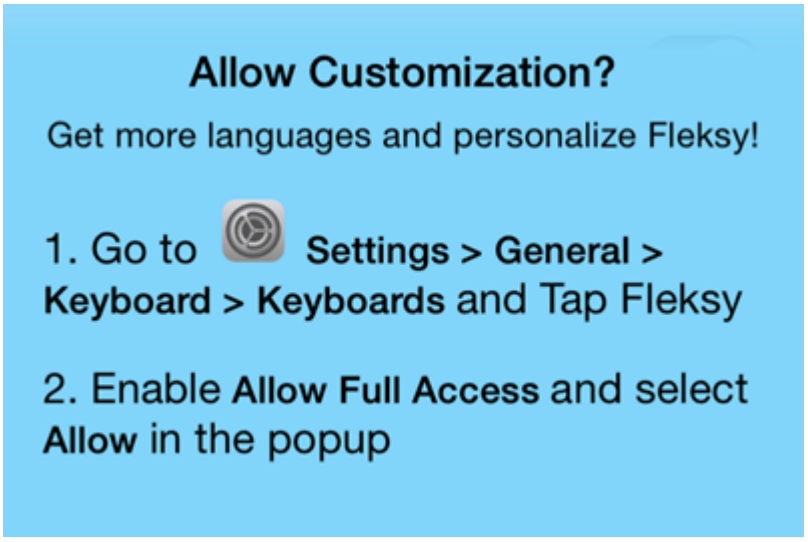
Fleksy

Fleksy used the term Language Modeling Data in its original Privacy Policy, explaining that it refers to data "such as common phrases or words that you use when typing with Fleksy." As of this writing, Fleksy's revised Privacy Policy doesn't use this term, and instead makes general claims such as the "Data concerning the User is collected to allow the Owner to provide its services" and purposes such as infrastructure monitoring, analytics, etc.

Though the policy at the first glance appears comprehensive, I couldn't find any details related to Fleksy's language modeling data or security measures beyond generic, vague statements. The company provided some clarification regarding the type of data its app uses and accesses in its FAQ.

The company's CEO Olivier Plante made a public commitment to protect users' privacy, stating that the company "purposefully built our technology and algorithms to remain processed locally, and never rely on server-side personal data processing." He pointed out that while most of Fleksy's competitors make money by mining user data "for various purposes such as advertising," personal voice assistants and so on, the company's business model is based on the purchases users make inside Fleksy's app.

When installing Fleksy's app, I was encouraged to enable "Customization" by granting the keyboard full access, stating that "keyboard apps require full access to sync your preferences."



Swype

Swype is another popular keyboard for mobile devices. It's developed by a company called Nuance. Unfortunately, I could not find any information about this app's security practices beyond a knowledgebase article for its Android keyboard, which stated that the company "does not collect personal information such as credit card numbers" and "suppresses all intelligent learning and personal dictionary addition functionality when used in a password field." The company's privacy policy contains generic statements such as:

> "Nuance may share Personal Information within Nuance to fulfill its obligations to you and operate its business consistent with this Privacy Policy and applicable data protection law."

Conclusions and Implications

Third-party keyboards for mobile devices offer features that improve upon some aspects of built-in keyboards. However, to enjoy such innovations, the users generally need to grant keyboard apps network access. This means that the users need to accept the following risks:

- The users need to be OK allowing keyboard developers to collect and store on their servers Language Modeling Data without fully understanding the meaning of this term or its privacy and security implications. The collection of such data is generally acknowledged by the keyboard developers, though they offer almost no details regarding how this data is safeguarded beyond referring to "encryption."
- The users need to trust the keyboard developer not to capture keystrokes and other sensitive data beyond Language Modeling Data. Doing this could be done on purpose by a malicious keyboard app or by accident by an otherwise benign application. In this case, the keyboard could act as a powerful keylogger for the mobile device.

Users might assume that the guardians of their mobile OS, be they Google, Apple, Samsung, etc., might protect them from malicious or accidental misuse of keystroke data and network access. However, such firms have no direct control over what happens once the data leaves the mobile device. Keyboard developers should provide additional details about their security practices to reassure users and consider how their apps might provide innovative features in a manner that minimizes users' risk and the apps' need for network access. It's unclear why

SwiftKey cannot function (at least on iOS) without network access, why Fleksy doesn't make it easier for users of its app to understand when data is sent to the company's servers and why Nuance doesn't discuss any of these topics for its Swype keyboard on its website. I discussed this topic on the Science Friday radio show in the second half of the segment called Tracking the Hidden Trail Left by Your Smartphone.
