The problem of key authenticity in secure messaging

Diana A. Vasile Martin Kleppmann Daniel R. Thomas
Alastair R. Beresford

March 30, 2019

Extended Abstract

Modern mobile messaging apps with end-to-end security, such as Signal, WhatsApp and iMessage, are now regularly used by over 1 billion people. These apps use public-key cryptography to encrypt messages on the sending device such that it can only be decrypted by recipient devices; any server infrastructure used to store and forward such messages cannot read or modify message contents. However, modern messaging apps have a common weak point in their security model: knowing whether a user has the right public keys for their communication partners. In the case of Signal, WhatsApp and iMessage, the discovery of public keys is performed using a *key server* or *key directory* operated by the app provider: when a device generates a keypair it sends its public key to the key server, and when a user wishes to communicate with a contact, the app looks up the public keys for the contact's devices using their phone number or email address.

A key server or key directory is an example of a *Public Key Infrastructure* (PKI), which links human-readable names (email addresses, phone numbers, domain names) with public keys, and requires an element of trust in the PKI provider. The PKI removes the need for users to manually manage keys – a task that has repeatedly been shown to be challenging for users – and effectively automate the security decision of whether to trust a particular public key.

Such automation of security decisions and key checking undoubtedly helps, and is one reason why the current generation of messaging apps with end-to-end security have seen widespread adoption, whereas PGP did not. Unfortunately, the security and privacy requirements of a user are not universal. For example, a human rights activist using a messaging app in a country with a repressive regime has a different threat model from a typical user communicating via social media in the United Kingdom. The effectiveness of specific attacks depends on how users behave, what they are trying to protect, and the degree to which they understand the security features of an app. In short, requirements and context matter. Consequently, an app cannot automate all security decisions; some decisions are necessarily deferred to the user.

Deferring security decisions to the user is hard to do safely because we cannot expect users to be cryptographers and security engineers: they do not have the knowledge to reason about and deal with security decisions appropriately.

Moreover, the PKI is a significant weak link in the end-to-end encryption ecosystem as deployed by messaging apps today: a compromised PKI allows an attacker to break end-to-end encryption by adding another "end" (sometimes called a *ghost user*) to the set of public keys that a user has registered with the PKI, allowing the ghost user to read all messages in a conversation. Neither user may be aware that this has happened. This approach has been proposed by GCHQ as the preferred way to provide law enforcement with access to end-to-end encrypted communication without inserting explicit back-doors into the actual encryption protocols.

This presentation will discuss the current attempts to detect ghost user attacks and compromised PKIs. However, these approaches are not perfect: false positives and false negatives can arise in the detection of such attacks. We propose several ways of achieving overall better system designs.