**ABSTRACT**

Messaging app developers are beginning to take the security and privacy of their users’ communication more seriously [3]. Unfortunately, a recent study has shown that the developers of many popular apps incorrectly use cryptography [2]. As a result, they make mistakes that may result in trivially broken encryption schemes. For example, the developers of Snapchat use a constant symmetric encryption key hardcoded into the app and it only takes 12 lines of Ruby to crack the encryption [1].

In this work, we propose ZIPR (Zero-Interaction PRi-vacy), a system that relieves developers from the task of using cryptography correctly. Designed for text-messaging apps, ZIPR automatically negotiates shared secret keys, and encrypts and decrypts messages as users of these apps chat with each other. No manual intervention is required by users for them to enjoy secure messaging.

There are two key ideas behind ZIPR. First, most text-messaging apps follow a basic UI scheme that contains (i) a text box for users to compose their message, (ii) a “send” button which they click on to send the message, and (iii) a list view to display sent and received messages. By intercepting events on these UI elements, ZIPR can manipulate the composed message before it is sent and before it is displayed. This allows the system to transparently encrypt and decrypt message data.

The second key idea is that ZIPR can reuse the communication channel defined by an app to negotiate a shared secret key between two users. This is done by piggybacking negotiation data on the messages users send to each other. A major advantage of this approach is that ZIPR can avoid the difficult task of establishing user identities. After all, a user of a text-messaging app is likely to carry out a conversation only with someone she knows, and both of them would have signed up for the chat service using some personal data such as their email addresses or phone numbers.

Developers use ZIPR by tagging UI elements; no changes to their source code are required. This is similar to HTTPS where web developers only need to configure their servers with SSL certificates to encrypt data transmission with their users. However, unlike HTTPS, the end-to-end encryption in ZIPR takes place between the two users carrying out a conversation and not between a server and a user. This ensures that even if the app servers are compromised, users’ messages would remain secure.

ZIPR is implemented in Android 4.3 and works with existing apps with very few modifications. In this demo, we show that our current prototype works with several apps including Whatsapp, Facebook Messenger, and Skype. These apps required only four, five, and three lines of modification to their UI XML definition files, respectively.

![Figure 1: Whatsapp running under ZIPR.](image)
1. REFERENCES

[1] Dmitri DB. 
https://security.stackexchange.com/q/52584/

An Empirical Study of Cryptographic Misuse in Android Applications. In Proceedings of CCS ’13, 
November 2013.

http://nyti.ms/1pbofVI