Secure Messaging Authentication Ceremonies Are Broken

Amir Herzberg | University of Connecticut
Hemi Leibowitz | Bar-Ilan University
Kent Seamons | Brigham Young University
Elham Vaziripour | Google
Justin Wu | Sandia National Laboratories
Daniel Zappala | Brigham Young University

People who use secure messaging apps are vulnerable to a hacked or malicious server unless they manually complete an authentication ceremony. In this article, we describe the usability challenges of the authentication ceremony and research to improve it. We conclude with recommendations for service providers and directions for research.

Messaging applications are a primary method for communication between individuals, to a large extent due to their simple, intuitive user interface and the ubiquity of smartphones. To protect the privacy of their users, some messaging applications incorporate end-to-end encryption, which encrypts messages between the sender and receiver so that their contents are not visible to anyone intercepting the traffic or even to the service provider. Messaging applications that incorporate end-to-end encryption are often known as *secure messaging applications*; they include WhatsApp, Facebook Messenger, iMessage, Telegram, and others. Many of these applications use an additional feature known as *forward security*, in which the encryption keys change from message to message, so that if an attacker is able to decipher one message it provides no hints about how to decipher previous or future messages.

Integrating end-to-end encryption and forward secrecy into a highly popular and usable messaging application seems to be the holy grail that the usable security community has looked for since the seminal paper "Why Johnny Can't Encrypt"[1] was published in 1999, which showed that users had difficulty using encrypted email even when significant effort had been spent to try to make it usable. The key advance made by secure messaging applications lies in automating all user interactions with encryption so that the messaging application appears no different from any insecure messaging system. Thus, these applications provide significant privacy benefits to users compared to many prior communication methods (for example, email and standard instant messaging). In particular, secure messaging applications protect users from passive attackers who seek to eavesdrop on connections, such as governments conducting surveillance.

However, the encryption in these applications typically uses provider-supplied keys for the communicating parties. Unless the parties perform an optional process for authenticating the keys they are using, an active attacker, such as a rogue or hacked provider, may supply incorrect keys. An attacker in this position can either impersonate anyone to another user of the service or conduct a man-in-the-middle attack, with the ability both to eavesdrop on and modify messages exchanged between users. Hence, to achieve true end-to-end security, it is crucial for users to correctly authenticate the keys they are using. Without doing this, users cannot know whether they have the right key for the person they are talking to or a false key provided by an attacker.

Digital Object Identifier 10.1109/MSEC.2020.3039727
Date of current version: 22 December 2020

The process by which users can verify their keys is known as an *authentication ceremony*.

At the same time, it is important to place active attacks in context. We know of no attacks on widely used secure messaging applications that have subverted the key server. Instead, successful attacks have included subverting SMS verification codes in Telegram and exploiting weaknesses in the phone calling implementation in WhatsApp. In addition, many ordinary users are not greatly concerned about security, since their conversations often don't include information they consider sensitive. Nevertheless, secure messaging applications are a tempting target for attackers, particularly because they are used by vulnerable groups such as journalists, members of political campaigns, and activists. As service providers close off other avenues of attack, authentication in secure messaging applications is one area that has received less attention.

In this article, we examine the tradeoffs secure messaging applications are making in their authentication designs, discuss their shortcomings, and make recommendations for improvements. Our analysis is applicable to the most popular secure messaging applications on the market, including the aforementioned, since all share architectural and user interface similarities. We first describe some of the technical details of secure messaging apps and how users can effectively authenticate their encryption keys. We then explain the usability shortcomings of authentication ceremonies in current applications and illustrate some of the research done to help users take better advantage of the privacy these applications offer. We conclude by providing recommendations for service providers and describing challenges for future research.

The Importance of Authentication

Whenever Alice and Bob communicate using a messaging application, both use some communication service operated by a service provider. While Alice and Bob may feel that they converse with one another directly, this is not the case. Any message that Alice sends to any other client is actually sent to the service provider, which then relays the message to its real destination. So, in practice, the communication between Alice and Bob is facilitated by two separate "client–server" connections: one between Alice and the provider and the other between the provider and Bob. Since the communication between the clients and the provider is encrypted, this model of communication is secure against eavesdroppers; however, this model does not provide protection from the provider itself. In other words, this communication model has a major security flaw "baked" into it: since all messages must go through the service provider, the service provider can read all the messages. A provider could use this position to mine messages for advertising purposes, conduct surveillance for a government, or enact censorship.

To mitigate this threat, secure messaging applications use end-to-end encryption. The application encrypts messages from Alice to Bob using an encryption key that is known only to her and to Bob, thus eliminating the threat of an honest but curious provider. For Alice and Bob to agree on such a key in a secure manner, they follow a key exchange protocol similar to the one performed at the beginning of a Transport Layer Security (TLS) connection. However, unlike a TLS connection, where at least one of the connection's endpoints is in possession of a public key certificate, clients normally do not have certificates; instead, the service provider assumes the role of a key directory and helps Alice and Bob exchange their public keys. Once Alice knows Bob's public key, she can securely establish a secret shared key with Bob, and no other party, including the service provider, can determine the value of the key or decrypt any of the messages.

A significant problem with this arrangement is that the aforementioned key exchange does not provide protection from rogue or compromised providers; such providers can still circumvent the protection provided by end-to-end encryption in the following way. Whenever Alice asks for Bob's public key, a rogue provider gives Alice a fake public key that the rogue provider controls. This is known as a *key substitution attack*. As a result, when Alice encrypts the secret key for Bob using the fake public key and sends it to Bob through the rogue provider, the provider can easily decrypt the secret key. Now, all that is left for the rogue provider is to reencrypt the key using Bob's real public key, and send it to Bob claiming that this message was sent by Alice. The result is that the rogue provider has positioned itself as a man-in-the-middle between Alice and Bob, tricking them into thinking that they are end-to-end encrypted, where in reality their messages are only client-server encrypted. The rogue provider can now eavesdrop on the message or modify messages without either party being aware of this attack.

Note that such an attack on TLS is futile because Alice can verify the authenticity of Bob's public key using a certificate Bob provides that is signed by a trusted certificate authority (CA). An adversary that controls such a CA can still launch the attack; however, this is a different problem and is the focus of current deployments of certificate transparency.

Unfortunately, currently there is no established automated mechanism to prevent or detect such attacks on secure messaging applications. The result is that current secure messaging applications are only opportunistically

encrypted end-to-end, that is, security guarantees hold only as long as you trust the provider. Instead, concerned users are offered an optional mechanism that they can perform to verify that their conversation was not compromised. This mechanism, known as an *authentication ceremony*, allows users to verify that the provider did not compromise the key agreement process, hence, their communication is truly end-to-end encrypted.

A typical authentication ceremony includes several methods for the user to verify the encryption keys. Figure 1 shows the ceremony from WhatsApp. In this instance, if the user is in the same location as their contact, they could scan a QR code from their contact's phone. The QR code typically encodes the fingerprints of the public keys for each user, and the application automatically compares the fingerprint of the locally stored keys with the representation in the QR code automatically, showing an indication of success or failure. In the more typical case where the user is not in the same location as their contact, they can make a phone call and compare the numeric representation of the key fingerprints.

Each application may use slightly different terminology and representation of the key fingerprints, as shown in Table 1. Moreover, some messaging applications do not encrypt all messages by default, and the user must know how to activate end-to-end encryption for each conversation from the menu system. Another difference is that some secure messaging applications use the Signal Protocol, which has been published and subjected to scrutiny,[2] whereas others use unpublished proprietary protocols whose security properties are less well known since research on them is sparser.

For the authentication ceremony to be effective, a number of important steps must be made by the user, all in the affirmative. Figure 2 illustrates these steps and the number of ways this process can go wrong. If a rogue

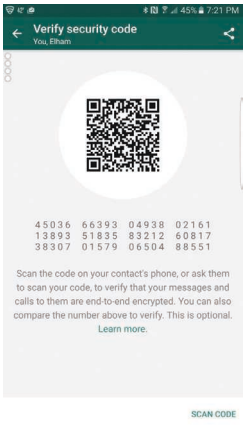


Figure 1. The authentication ceremony in WhatsApp.

provider attacks an existing connection, then it can only be thwarted if the user sees and understands a warning about their encryption keys changing, finds the authentication ceremony, successfully executes it, understands what failure (nonmatching encryption keys) means, and decides to stop communicating. Note that many of these depend on user understanding, which puts a premium on good design, following principles developed by the usable security community. Then, even if a user takes all the correct actions, they may still decide to continue communicating if, for example, they don't perceive their communication to be worth protecting. Finally, if a rogue provider attacks a new connection, then the user typically sees no warning and must be sufficiently aware of security risks and decide authentication is important to have a chance of thwarting the attack.

Finally, we note that a more detailed comparison of secure messaging applications was made by Unger et al.[3] in 2015. They used a framework to compare a

Table 1. An overview of secure messaging applications, the protocol they use to encrypt messages, the terminology they use to refer to the authentication ceremony, and the method they use to represent fingerprints of the encryption key.

Application	Protocol	All messages encrypted	Terminology	Representation
Signal	Signal	✓	Safety numbers	60-digit numeric
WhatsApp	Signal	✓	Security code	60-digit numeric
Facebook Messenger	Signal		Device keys	66-digit hexadecimal, one per device
iMessage	Proprietary	✓	No ceremony	None
Viber	Proprietary	✓	Secret identification keys	48-digit numeric
Telegram	Proprietary		Encryption key	8 × 8 pixel image or four emojis

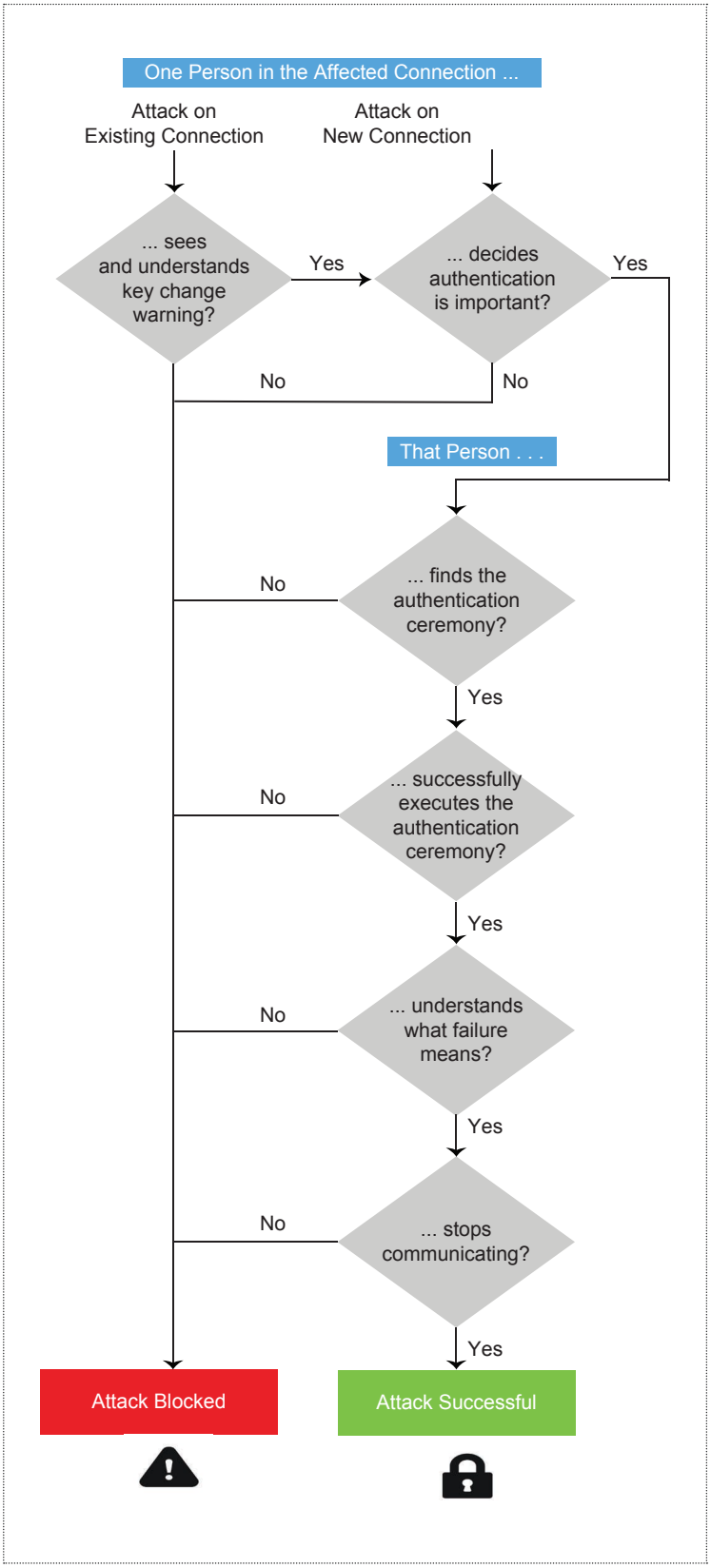


Figure 2. The security decisions needed for the authentication ceremony to be successful.

wide variety of messaging applications using a set of security, usability, and adoption properties. Since then, most secure messaging applications in use today, serving billions of users, have converged on the approach we describe here, which was pioneered by TextSecure (the predecessor to Signal), using a combination of a key directory, trust-on-first-use, and optional key verification. The primary exception is iMessage, which, as noted in Table 1, assumes the key server is trusted and does not provide a method for manually verifying key fingerprints.

The Usability of the Authentication Ceremony

A variety of papers have examined the usability of the authentication ceremony in secure messaging applications. The key questions these papers consider is whether users will be able to find and use the authentication ceremony, particularly when under attack. Generally, these papers have found that users do not understand the threats they face when using a secure messaging application nor the need for the authentication ceremony. They typically only find the ceremony when prompted to look for it and, even then, have a hard time using and understanding it.

A typical setup in these studies is to have users communicate normally through the application, initiate an attack (for example, by running a modified key server under the control of the study authors), and then observe how users react to the attack. One of the first studies of this sort[4] used an early version of Signal that showed the user the public keys for each party in the conversation, but without any instructions regarding how to compare these keys. A small number of people simply clicked "Accept," essentially allowing the attack to occur, and, of the rest, about half tried to complete the ceremony and less than half of them succeeded.

Two other studies have generalized these observations to additional secure messaging applications, including WhatsApp, Viber, Telegram, Signal, and Facebook Messenger.[5,6] Both studies conducted an initial phase where participants received high-level information about secure communication but no detailed instructions about the authentication ceremony and how to complete it. The studies observed similar results, with only 13% able to find and complete the authentication ceremony in the first study and only 14% in the second study. Both studies also then included a second phase where users received some general instructions about encryption or were given a tip to confirm that they shared the same encryption key. In both cases, the majority of participants (about three-fourths) were able to complete the authentication

ceremony. However, it took significant time to complete the ceremony, about an average of 70 s in the first study and over 11 min to find and complete the ceremony on average in the second.

Qualitative results from these studies revealed additional usability issues with the authentication ceremony. For example, many participants grew fatigued during the authentication process and complained about the length of the encryption key they needed to compare. Although the applications include some kind of warning about encryption keys changing, due to the attack, most users did not notice the message. Furthermore, users did not understand the meaning or importance of the key change or mistook it for a connectivity problem. A few users even attempted to send the key verification information through the messaging application itself.

Studies of this nature also reveal the mental models that users have when conceiving of authentication. During one study involving pairs of participants,[6] the participants were asked to exchange messages with each other while making sure they were communicating with each other, and no one else could read their messages (such as a service provider). The participants did not use the authentication ceremony to satisfy this request. Instead, they attempted to authenticate their partner using a variety of approaches, including a video call, asking questions that required specialized knowledge, and speaking in a second language that both participants knew. This indicated that users generally lack any understanding of encryption, cryptographic keys, and the concept of a man-in-the-middle attack. Using an authentication ceremony is not intuitive.

Toward a More Usable Authentication Ceremony

In addition to studying the usability of existing secure messaging applications, some research has investigated how to improve the usability and security of the authentication ceremony. This research stems from a concern that if users do not understand how to find or use the authentication ceremony, then they could be susceptible to attack. This question was investigated in several works, falling into three main approaches.

Improve the Representation of Public Keys in the Ceremony

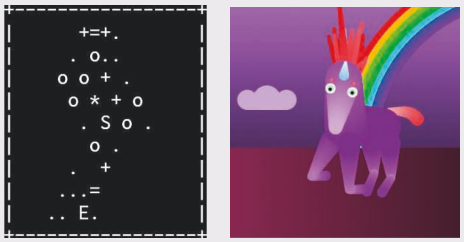
The first approach studies alternative representations for the fingerprints of the encryption keys used in the authentication ceremony. Research has investigated a variety of alternative representations, including numeric, hexadecimal, textual (words and sentences), and graphical, as shown in Table 2. The primary finding of two works[7,8] is that textual representations, which transform the fingerprint into a series of words or sentences, have the best success in helping users detect attacks. However, so far, most secure messaging apps continue to use numeric or hexadecimal representations of fingerprints, despite these results. The Signal application has stated a preference for a numeric format, because it is simpler to provide international localizations for numbers.[9] However, using sentence-based fingerprints could improve usability significantly and impact the vast majority of users if done for the most common languages, with numeric representation as a fallback.

Help Users Find and Use the Ceremony

The second approach seeks to redesign the user interface of secure messaging applications to help users find and use the authentication ceremony. Vaziripour et al.[10] modified the Signal user interface with these goals in mind, as shown in Figure 3, including both an explicit prompt for the authentication ceremony and streamlined instructions for users. Their user study showed that these changes significantly reduce the time for users to find and complete the ceremony. With these changes, 90% of study participants were successfully able to find and use the ceremony compared to 30% with the existing design of Signal.

Despite these positive results, this approach may not be appropriate for all users. Some users may not

Table 2. Example fingerprint representations that have been tested by usability researchers.

Representation	Example
Numeric	7748 5689 7453 6977 5604 5939 2765 8791 5022 4957 3805 0309
Hexadecimal	C10A 8BE2 6123 FA22 BB83 02E3 123 C 5AE6 21FB 41BC
Words	jumping crazy baggage help ripcord pardon board shelf sofa rain forward happy stay lunch trouble satisfy
Sentences	The basket ends your right cat on his linen. Her range repeats her nerve. The smile tells secretly. My clean cake pulls your waiting pocket.
Graphical	

(Source: Tan et al.[8])

consider themselves to be targets of attacks or they may not be discussing anything sensitive in a particular conversation and hence could be annoyed if regularly nudged to improve security when the risk is actually low. Indeed, the most typical reason for authentication keys changing is when one person reinstalls the messaging application. Thus this approach may be most helpful for users who are at high risk, such as journalists, members of political campaigns, or activists.

Design for Understanding

The third approach focuses on improving the awareness of users to the risk they face when authentication keys change and the responses they can take to mitigate this risk. This approach emphasizes user autonomy to make the choice that is relevant for them rather than forcing users toward maximal security. Wu et al.[11] followed this idea by using a risk communication framework borrowed from public health. As shown in Figure 4, they reframed the authentication ceremony as a privacy check and substantially redesigned the user interface of Signal to better help users understand why a privacy check is needed, what effort they may need to expend to conduct the privacy check, and what it means if the check succeeds. Their design also included a shield icon that shows the privacy status of their conversation, shown in Figure 5, which is currently hidden in secure messaging applications. This helps the user know whether each conversation partner has been authenticated, using a visual representation of whether the privacy check is undone, successfully complete, or failed.

Designing for user understanding means that compliance is not the primary concern. Rather, the priority lies in informing the user so they can make a choice suitable for their situation (for example, based on the sensitivity of their conversation). This work showed that user understanding of risk increased with the new design, with 50% of users understanding the purpose of the authentication ceremony (16% for Signal), 56% understanding the meaning of matching identifiers (27% for Signal), and 56% understanding the meaning of nonmatching identifiers (28% for Signal). More work is needed to further improve on the effectiveness of the design, but it is possible to improve comprehension of security mechanisms while promoting user choice in whether to activate them.

Recommendations

Secure messaging applications prioritize usability over security, thus enabling billions of users to regularly use an end-to-end encrypted messaging service. Usability is increased by automating user interactions with encryption and only warning users if public key fingerprints change. This trust-on-first-use model assumes the messaging service is not hacked or malicious, which is a reasonable threat model for most users. Moreover, participants from the literature have indicated regularly that they don't consider themselves an important target

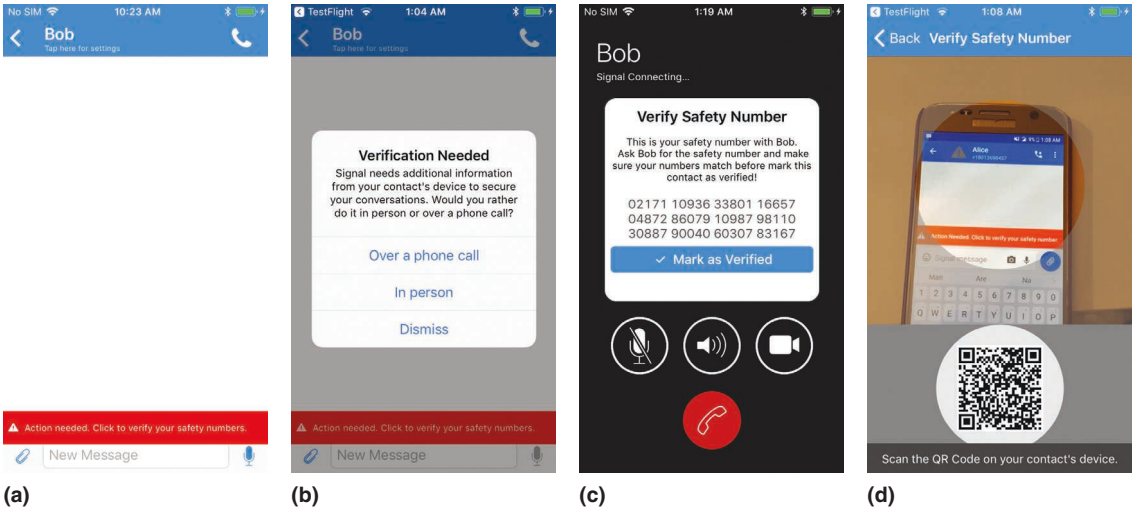


Figure 3. Vaziripour et al.[10] modified Signal (a) by adding an explicit prompt for the authentication ceremony, using a red bar at the bottom of every conversation that was not yet verified and the wording "Action needed: Click to verify your safety numbers." (b) They also split the authentication ceremony into two parts: a phone call option and an in-person option. (c) The phone call option uses an in-app phone call with safety numbers shown on the screen. (d) The in-person option uses a QR code scanner. (Source: Vaziripour et al.[10])

for an attacker and don't believe their conversations reveal anything important, so security is not a priority. Likewise, both automation of encryption and avoiding warning fatigued are backed by years of research in the usable security community.

However, the downside of these design choices is that users cannot find, understand, or execute the authentication ceremony when needed. The authentication ceremony as currently implemented is thus broken, since it is both not being used in practice and is unusable. Based on our review of the literature, we believe improvements to the authentication ceremony could significantly increase its usability. We make the following recommendations for service providers.

1. Reframe the authentication ceremony as a privacy check, as recommended by Wu et al.[11] This design follows best practices, based on Microsoft's NEAT guidelines and a wealth of literature from the usable security community. As a result, it leads to stronger user understanding of the purpose and meaning of the authentication ceremony.
2. Enable at-risk users to activate a high-security mode, which safeguards them through additional policy. This could include preventing users from exchanging messages until they perform the authentication ceremony, both at the start of each conversation and when safety numbers change. Preventing messages from being delivered is already a part of the Signal app in certain situations,[11] so this change is not expensive. The default settings that prioritize usability can be kept for most users, with the changes from the first recommendation enabling them to transition to stronger security when they need it.

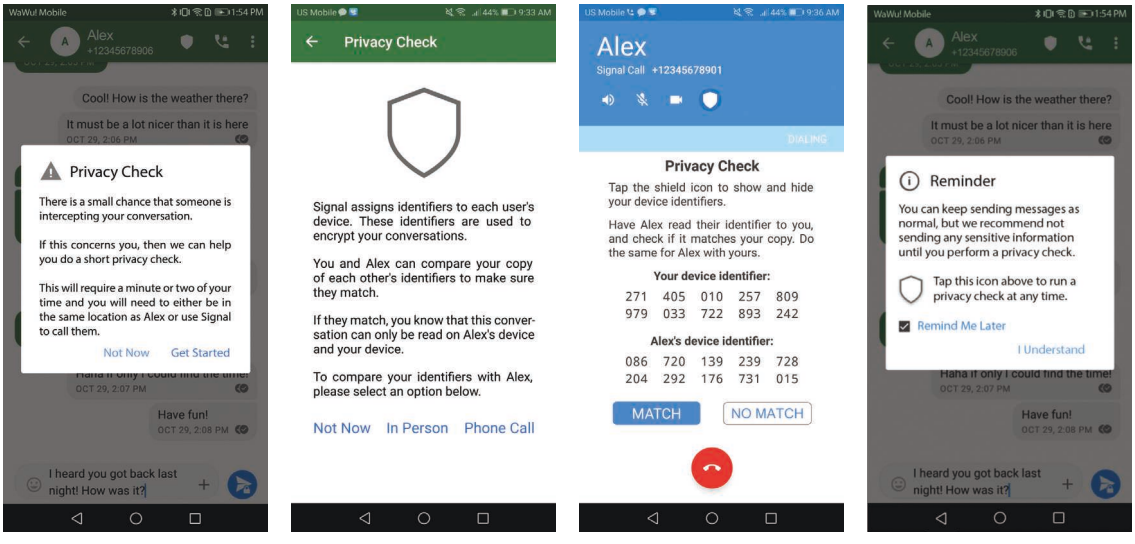


Figure 4. Wu et al.[11] reframed the authentication ceremony as a privacy check (a) with messages that communicate the likelihood of a risk, the action they can take to mitigate that risk, and how much of their time this will take. (b) If the user chooses to perform the privacy check, its purpose and what it accomplishes are concisely defined and, as with Vaziripour et al., the ceremony is split into an in-person and phone call option. (c) The safety number is split and renamed as device identifiers, following prior work showing users view encryption primarily as access control.[12] (d) Users can also choose not to do the authentication ceremony, and the application shows them how to get back to it later. (Source: Wu et al.[11] and J. Wu and D. Zappala.[12])

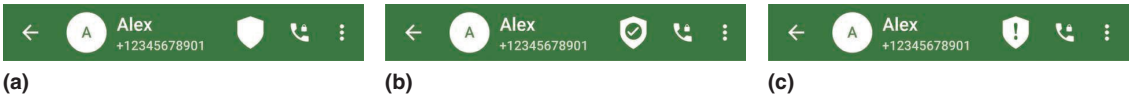


Figure 5. The privacy check icons designed by Wu et al.[11]: (a) the default state, (b) the matching identifiers, and (c) the nonmatching identifiers. (Source: Wu et al.[11])

3. Use textual representations of key fingerprints. While this creates extra work for developers, this will make it significantly easier for users to compare fingerprints and thus more likely that they will perform the ceremony when needed. Users have consistently complained about numeric representations, so developers should heed their concerns. The user interface could allow users to fall back to a numeric representations if they do not understand the textual one.

If a major provider would push such improvements, this could set the bar for the rest of the providers. The success of such a push would rest on how well an application could help users to understand the risks they face and the benefits of these changes in preventing attacks. It may be possible to use gamification to encourage users to use the ceremony. We note that some games have offered incentives for users to adopt two-factor authentication, so a straightforward rewards approach could reap benefits.

We especially call attention to the reality that not all users have the same risk profile. Users who are at risk have compelling needs for heightened security.[13,14] Likewise, residents of countries without strong rights for free speech are regularly at risk when communicating with friends and family. Service providers should at a minimum focus on helping these groups of users, since their app can cause significant harm for these users if they are victims of an attack.

We also advise the research community to investigate methods for automatically detecting and preventing key substitution attacks, which could eliminate the need for the authentication ceremony. A likely approach for detecting attacks is to deploy a system that audits key servers run by service providers. Similar to Certificate Transparency for the web, CONIKS[15] and Google's Key Transparency system could be used to verify that key servers advertise a consistent public key for each user. Research is still needed to demonstrate how to integrate this type of system with a secure messaging app and to design a user interface that helps users understand the consequences of an attack and take appropriate action. Moreover, while this approach helps service providers offer better assurances for their users, it still requires significant deployment effort and only provides detection of attacks.

Solutions that can prevent key substitution attacks are both more useful and more difficult to develop. One possible approach is to establish a system for issuing certificates to users to certify their public keys. Such a public system could allow Alice to learn Bob's public key without relying on or trusting the service provider. Further, such an open system would allow Bob to choose which CA to use when obtaining a certificate, and likewise would allow Alice to choose which authorities to trust when querying keys. Such a system would need some method of auditing authorities to detect misbehavior, as discussed above. Research would be needed to develop usable methods for issuing certificates to users at scale, including covering situations where a phone is lost or software is reinstalled. Significant development and user testing would also be needed to verify that this kind of system would be feasible. Finally, coordinating such a standardization and deployment effort among service providers is a daunting challenge. Providers may not be enthusiastic about delegating this operation outside of their control, since they are thriving while having built a "walled garden" in which their app serves only their users.

Any solution for improving secure messaging applications, such as those proposed above, carries with it fundamental tradeoffs. The framework developed by Unger et al.[3] provides a useful way to reason about these tradeoffs in terms of achievable properties, which could then be verified with user testing. We call for providers to work with researchers in developing detection and prevention mechanisms that would meet their needs and protect users.

We close with the hope that service providers recognize the importance of truth in advertising. End-to-end encryption only provides protection from active attackers if users authenticate each other. Application providers should ensure their users are aware that secure messaging applications are currently only secure if they trust the application provider or if they take additional steps to authenticate. Since users generally do not use the authentication ceremony, even when warned about a key change, trust-on-first-use has essentially devolved to simply always trusting the service provider. Trusting service providers may be appropriate for much of the general public, but those at risk should be guided toward learning how to authenticate. ■

Acknowledgments

Amir Herzberg was partially supported by an endowment from the Comcast Corporation. Kent Seamons and Daniel Zappala are supported in part by the National Science Foundation grant CNS-1816929. The opinions expressed in the article are those of the researchers themselves and not of their universities or sources of funding.

References

1. A. Whitten and J. D. Tygar, "Why Johnny can't encrypt: A usability evaluation of PGP 5.0," in *Proc. 8th USENIX Security Symp.*, 1999, pp. 169–184.

2. K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila, "A formal security analysis of the signal messaging protocol," *J. Cryptology*, vol. 33, no. 4, pp. 1914–1983, 2020. doi: 10.1007/s00145-020-09360-1.
3. N. Unger et al., "SoK: Secure messaging," in *Proc. IEEE Symp. Security Privacy*, 2015, pp. 232–249.
4. S. Schröder, M. Huber, D. Wind, and C. Rottermanner, "When SIGNAL hits the fan: On the usability and security of state-of-the-art secure mobile messaging," in *Proc. 1st European Workshop on Usable Security (EuroUSEC)*, 2016, pp. 1–7. doi: 10.14722/eurousec.2016.23012.
5. A. Herzberg and H. Leibowitz, "Can Johnny finally encrypt? Evaluating E2E-encryption in popular IM applications," in *Proc. Workshop on Socio-Technical Aspects Security Trust (STAST)*, 2016, pp. 17–28. doi: 10.1145/3046055.3046059.
6. E. Vaziripour et al., "Is that you, Alice? A usability study of the authentication ceremony of secure messaging applications," in *Proc. Symp. Usable Privacy Security (SOUPS)*, 2017, pp. 29–47.
7. S. Dechand, D. Schürmann, K. Busse, Y. Acar, S. Fahl, and M. Smith, "An empirical study of textual key-fingerprint representations," in *Proc. 25th USENIX Security Symp.*, 2016, pp. 193–208.
8. J. Tan, L. Bauer, J. Bonneau, L. F. Cranor, J. Thomas, and B. Ur, "Can unicorns help users compare crypto key fingerprints?" in *Proc. Conf. Human Factors Comput. Syst. (CHI)*, 2017, pp. 3787–3798. doi: 10.1145/3025453.3025733.
9. M. Marlinspike, "Safety number updates," Signal, May 2016. [Online]. Available: https://signal.org/blog/safety-number-updates/
10. E. Vaziripour et al., "Action needed! Helping users find and complete the authentication ceremony in Signal," in *Proc. Symp. Usable Privacy Security (SOUPS)*, 2018, pp. 47–62.
11. J. Wu et al., "Something isn't secure, but I'm not sure how that translates into a problem": Promoting autonomy by designing for understanding in Signal," in *Proc. Symp. Usable Privacy Security (SOUPS)*, 2019, pp. 137–153.
12. J. Wu and D. Zappala, "When is a tree really a truck? Exploring mental models of encryption," in *Proc. Symp. Usable Privacy Security (SOUPS)*, 2018, pp. 395–409.
13. L. Simko, A. Lerner, S. Ibtasam, F. Roesner, and T. Kohno, "Computer security and privacy for refugees in the United States," in *Proc. IEEE Symp. Security Privacy*, 2018, pp. 409–423. doi: 10.1109/SP.2018.00023.
14. C. Chen, N. Dell, and F. Roesner. "Computer security and privacy in the interactions between victim service providers and human trafficking survivors," in *Proc. USENIX Security Symp.*, 2019, pp. 89–104.
15. M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten, and M. J. Freedman, "CONIKS: Bringing key transparency to end users," in *Proc. USENIX Security Symp.*, 2015, pp. 383–398.

Amir Herzberg is the Comcast Professor for Cybersecurity Innovation in the Department of Computer Science and Engineering at the University of Connecticut, Storrs, Connecticut, 06268, USA. His research interests include network security (especially routing/Domain Name System/transport, denial of service, web), privacy and anonymity, applied cryptography, usable security, security for cyberphysical systems, and social, economic, and legal aspects of security. Herzberg received a Ph.D. in computer science from Technion, Israeli Institute of Technology, Haifa, Israel. Contact him at amir.herzberg@uconn.edu.

Hemi Leibowitz is a Ph.D. student in the Computer Science Department at Bar-Ilan University, Ramat Gan, 5290002, Israel. His research interests include usable security, anonymous communication, and key management. Contact him at leibo.hemi@gmail.com.

Kent Seamons is a professor in the Computer Science Department at Brigham Young University, Provo, Utah, USA, and director of the Internet Security Research Lab, Provo, Utah, 84602, USA. His research interests include usable privacy and security, authentication, and key management. Seamons received a Ph.D. in computer science from the University of Illinois. He is an associate editor of *IEEE Transactions on Dependable and Secure Computing*. Contact him at seamons@cs.byu.edu.

Elham Vaziripour is a software engineer at Google, Sunnyvale, California, 94089, USA. Her research interests include the usability of the authentication ceremony in secure messaging applications. Vaziripour received a Ph.D. in computer science from Brigham Young University. Contact her at elhamvaziripour@gmail.com.

Justin Wu is a usability researcher and developer at Sandia National Laboratories, Albuquerque, New Mexico, 87185, USA. His research interests include security and privacy. Wu received a Ph.D. in computer science from Brigham Young University. Contact him at justin.c.w.wu@gmail.com.

Daniel Zappala is a professor in the Computer Science Department at Brigham Young University, Provo, Utah, 84602, USA. His research interests include usable security, authentication, and risk communication. Zappala received a Ph.D. in computer science from the University of Southern California, Los Angeles. Contact him at zappala@cs.byu.edu.