# Social Authentication for End-to-End Encryption

Elham Vaziripour
Brigham Young University
elhamvaziripour@byu.edu

Mark O'Neill
Brigham Young University
mto@byu.edu

Justin Wu
Brigham Young University
justinwu@byu.edu

Scott Heidbrink
Brigham Young University
sheidbri@byu.edu

Kent Seamons
Brigham Young University
seamons@cs.byu.edu

Daniel Zappala
Brigham Young University
zappala@cs.byu.edu

## 1. INTRODUCTION

Over the last several decades, it has become increasingly important to secure data via end-to-end encryption. The Internet has evolved to provide security for connections, primarily using TLS (or SSL), but generally fails to provide true end-to-end encryption. While TLS and similar protocols encrypt data during transit, data at rest is often unprotected, residing in storage on a client or server machine in plaintext. Data in this state are susceptible to honest-but-curious service providers, hackers, physical theft, and coercive governments.

Generic public-key cryptography provides powerful mechanisms to enable end-to-end encryption, but providing good usability for these mechanisms is a challenging task for novice users—leading to the decades-long situation where "Johnny can't encrypt" [8, 7, 6]. The primary problems center on *user-to-user authentication* – authenticating users to each other by associating their identities with public keys. We have made significant progress authenticating web sites to users (via X509 certificates and associated authorities) and authenticating users to web sites (with passwords). Each of these have their challenges, but have at least been widely deployed. Authenticating users to one another, however, has seen relatively little adoption. Usable mechanisms for personal key management, key distribution, and key authentication are still largely open issues.

Significant progress has been made recently with the proliferation of secure messaging apps such as Signal and ChatSecure. These applications address the aforementioned issues in a variety of ways. First, operating primarily on mobile devices greatly mitigates key management problems, since users almost always have their mobile devices on their person. Second, these apps also store mappings between identities and keys and perform authentication on behalf of users, reducing the need for manual collection of key-identity pairs and authentication. During the first communication with another user, users are often advised to perform out-of-band validation of public keys (e.g. reciting public key fingerprints by voice through a phone call), but it is not clear how frequently this is done. Afterward, the verified keys are stored on their respective devices and future authentication proceeds automatically (and locally). Similar efforts have been made to provide usable, secure email. Efforts such as Private WebMail (PWM) [5] and Virtru handle generation and validation of keys on behalf of users by means of key escrow. There is some evidence that users prefer the convenience of automatic (but potentially less secure) methods rather than manual key exchange [1].

Despite the improvements these applications bring, some notable issues persist:

- **Key discovery:** Discovering the public key of a user may or may not be possible. For example, Signal uses phone numbers to look up identities for remote users, prohibiting contact with anyone whose phone number is unknown.
- **Key validation:** These apps rely on the user to verify the public key of their associates through some manual means. Such behavior is also not enforced, reducing the security of the system overall.
- **Generality:** These and similar applications are not general in two ways. First, these applications use specific mediums for communication rather than supporting key infrastructure for communication across arbitrary mediums (Signal uses SMS and ChatSecure uses XMPP). Second, the applications cannot be used from other devices such as laptops because their associated private keys rely on the mobile device only.
- **Trusted third parties:** Pwm and Virtru rely on a centralized server to verify that users own their respective email addresses and delivers private keys to users based on this third party authentication. While this assists greatly in application portability and key discovery, reliance on a trusted third party violates the true spirit of end-to-end encryption.

We need more work on user-to-user authentication solutions that are general and portable, automate key discovery, and bootstrap and automate key validation as much as possible. An identity and associated contacts should be portable across the devices and the applications a person uses. Key management should avoid reliance on trusted third parties, so that the security guarantees provided to users cannot be easily broken by governments or service providers.

This work was supported by the National Science Foundation under Grant No. CNS-1528022. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.


*Symposium on Usable Privacy and Security (SOUPS)* 2016, June 22–24, 2016, Denver, Colorado.

## 2. SOCIAL AUTHENTICATION

We propose that the issues of key discovery and validation can be solved by bootstrapping off the trust that already exists among users of online social networks (OSNs). Each OSN already provides users with long histories of posts, pictures, and personal communications from their contacts and provides authentication of its users (via password logon). By following a verified user on Twitter or accepting a friend request on Facebook, users are already making an authentication judgment. Thus if public keys were posted to and associated with users' various OSN accounts, an organic set of verified key-identity pairs could emerge. By querying keys for a user from multiple OSNs and checking for agreement, the application could enhance trust in a public key, as multiple OSNs would have be compromised or collude to present a believable false key. Such a system mitigates much of the manual key validation problem as users can rely on the robustness of multiple authorities vouching for the authenticity of a particular key. This mechanism also has an added benefit – users need only have some type of OSN account (or phone number) to be discovered, rather than forcing every user to have an account with a specific service or OSN.

Using OSNs for discovery and validation then opens the door for a generic key management platform that does not rely on trusted third parties to store private keys or to validate OSN accounts. A mobile application could be responsible for generating a keypair and posting, retrieving, and maintaining public keys on OSNs. The mobility of the application would allow the private key to be readily available, rather than stuck on a device that is not with you. In addition, the application could provide a crypto API that allows both local and remote (e.g. desktop) applications to encrypt, sign, verify, and decrypt arbitrary communications.

Two recent efforts provide some parts of this solution. Keybase provides a set of tools that allow users to generate PGP keypairs and post public keys to an OSN such as Twitter and Github, which implicitly verifies the authenticity of those keys to anyone who trusts those OSN accounts [4]. It also allows users to store a password-encrypted private key on the Keybase server for portability across devices. However Keybase falls short of providing a mobile application responsible for key management, automatic key discovery (by querying the services and OSN accounts associated with an identity), and automation of cryptographic operations. SafeSlinger provides a mobile application that automates key exchange among a group of users, but is primarily aimed at synchronous, in-person key exchange [2].

## 3. OPEN RESEARCH QUESTIONS

A wide range of open usability problems must be solved in order to provide social authentication:

- *Managing keys.* Users need methods for managing their public keys, including revoking keys when they are lost or stolen, and issuing new ones after expiration. Typical methods for coping with lost keys depend on a smartcard or a third party that can store private keys that are encrypted with a strong password. However, these methods introduce additional usability challenges, such as helping users to manage subkeys.
- *Inducting novices.* Our experience designing secure email systems indicates that novices need help transitioning to secure communication. Leveraging the users' OSN has the potential to ease the induction experience because it will include familiar identifiers and systems.
- *Authenticating strangers.* Authenticating people a user doesn't know well is a particular challenge. Perhaps a system should help people take different actions depending on the level of trust they have established. The web of trust has long been proposed as a way of helping determine the trustworthiness of someone who is known to your friends, but inferring trust is difficult [3] and little usability work has shed light on whether building a web of trust is viable.
- *Evaluating Resilience.* If someone's OSN account is compromised attackers may provide a fake public key for the compromised identity. The software will need to distinguish between this and regeneration of expired or lost keys (possibly by leveraging agreement among other OSN accounts for the compromised user). We also need to measure how vulnerable such a platform would be to Sybil and related attacks on OSNs that may provide inaccurate values of trust to users, directly or indirectly. Finally, standards are needed for measuring trust in both a user and a user's key.

We intend to develop a system that will allow us to evaluate solutions to these problems in both short and long term user studies.

## 4. REFERENCES


[1] W. Bai, D. Kim, M. Namara, Y. Qian, P. G. Kelley, and M. L. Mazurek. An inconvenient trust: User attitudes toward security and usability tradeoffs for key-directory encryption systems. In *Symposium On Usable Privacy and Security*, 2016.

[2] M. Farb, Y.-H. Lin, T. H.-J. Kim, J. McCune, and A. Perrig. Safeslinger: easy-to-use and secure public-key exchange. In *Proceedings of the 19th annual international conference on Mobile computing & networking*, pages 417–428. ACM, 2013.

[3] J. Golbeck and J. Hendler. Inferring binary trust relationships in web-based social networks. *ACM Transactions on Internet Technology (TOIT)*, 6(4):497–529, 2006.

[4] Keybase. `https://keybase.io/`.

[5] S. Ruoti, J. Andersen, S. Heidbrink, M. O'Neill, E. Vaziripour, J. Wu, D. Zappala, and K. Seamons. "we're on the same page": A usability study of secure email using pairs of novice users. In *Proceedings of the 34th annual ACM conference on Human factors in computing systems, ACM*, 2016.

[6] S. Ruoti, J. Andersen, D. Zappala, and K. Seamons. Why Johnny still, still can't encrypt: Evaluating the usability of a modern PGP client. *arXiv preprint arXiv:1510.08555*, 2015.

[7] S. Sheng, L. Broderick, C. A. Koranda, and J. J. Hyland. Why Johnny still can't encrypt: evaluating the usability of email encryption software. In *Symposium On Usable Privacy and Security*, pages 3–4, 2006.

[8] A. Whitten and J. D. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *Usenix Security*, volume 1999, 1999.