Identifying Close Friends on the Internet

Randy Baden, Neil Spring, Bobby Bhattacharjee
University of Maryland

ABSTRACT

Online Social Networks (OSNs) encourage users to create an online presence that reflects their offline identity. OSNs create the illusion that these online accounts correspond to the correct offline person, but in reality the OSN lacks the resources to detect impersonation. We propose that OSN users identify each other based on interaction and experience.

We believe that impersonation can be thwarted by users who possess *exclusive shared knowledge*, secret information shared only between a pair of OSN friends. We describe existing protocols that use shared secrets to exchange public keys without revealing those secrets to attackers. We present results from a user study on Facebook to show that users do share exclusive knowledge with their Facebook friends and attackers are rarely able to guess that knowledge. Finally, we show that friend identification can be extended using a web of trust built on the OSN friend graph.

1. INTRODUCTION

Online Social Networks (OSNs) have persuaded millions of users [6] to give their offline identities an online presence. While these OSN identities are convenient for online communication, they risk impersonation [1] and may provide personal information that threatens the security of other systems [10, 11]. Users, aware that their personal information is valuable, may choose only to allow their friends to see their information. However, even correct privacy settings can be foiled if someone has infiltrated their circle of friends. Users cannot trust that the person behind an online account is actually their offline friend, even if that account has the correct picture and profile information [4]. Solving the problem of OSN impersonation is critical to the future of privacy and security in OSNs.

We are approaching a point at which OSNs will become the bridge between offline identities and systems. Authenticatr [12] shows that these identities can be a valuable tool in system design. Unfortunately, an OSN provider is not equipped to authenticate user identities since the provider knows almost nothing about its users other than what they themselves supply, and that supplied data can be easily forged. Though we, the OSN users, are also not able to identify arbitrary OSN users, we are actually well-equipped to detect when an attacker is impersonating one of our friends.

Offline we have ways of identifying a friend—such as recognizing her appearance or voice—that are either difficult or impossible in online communication. Instead we can use *exclusive shared knowledge* for identification: we can identify a friend (either online or offline) by asking questions that only she can answer.

Once we identify our friend, we can ask her to provide or verify a public encryption key associated with her identity. By repeating this process with all of our friends, we bootstrap a public key infrastructure (PKI) that we can use on the OSN, a PKI that could be important for emerging OSN applications that require security or privacy. This distributed and decentralized PKI could be a necessary component of future distributed OSN research.

We face several challenges by verifying OSN identities with shared knowledge. We must guarantee that shared knowledge remains secret or we open ourselves up to impersonation attacks. Users may not share exclusive knowledge with all of their friends, so the PKI we create may be limited in scope. Lastly, an impostor may be able to guess the knowledge shared by a pair of users, so we must limit and, if possible, detect such attacks.

Our contributions are the following. We show that existing protocols can be used in an OSN to exchange keys without revealing shared knowledge. We perform a user study that shows that strangers have less than a 2% chance of guessing the answers to shared knowledge questions; this compares favorably to web-based security questions—another identification scheme based on personal information—which can be guessed 17% of the time by strangers [13]. We show that even when users only exchange keys with a few friends, we can discover the keys of many friends and friends-of-friends with a web of trust [14]. Finally, we show that the same web of trust detects 80% of all successful impersonation attacks.

We organize this paper as follows. In Section 2 we describe how to use exclusive shared knowledge to distribute public keys and show that we can avoid impersonation attacks with existing protocols. We describe our user study in Section 3 and show that shared knowledge exists and can be used to identify friends. We describe related work in Section 4 and conclude in Section 5.

2. EXCLUSIVE SHARED KNOWLEDGE

The strength of exclusive shared knowledge lies in its secrecy, so we must handle it delicately to prevent attacks. We seek a key exchange protocol in which one user can use shared knowledge to verify another user's offline identity, without either user revealing that knowledge in the process.

2.1 Design

One user, the *asker*, wishes to verify the identity of her friend, the *askee*. The users are communicating over an insecure channel and we assume their messages can be intercepted by a man-in-the-middle attacker, the *meddler*. Key exchange is asymmetric: in one instance of the protocol, the asker identifies the askee only. Symmetry is not required in OSNs that have directed friend relationships, such as Twitter. For symmetric OSNs like Facebook, we realize symmetry by repeating the asymmetric protocol with the asker and askee roles reversed.

We will apply exclusive shared knowledge in our protocol as follows. The asker formulates a question Q with answer A that relies on the exclusive knowledge shared between the asker and askee. At the end of the protocol, the asker will receive a public key PK with the guarantee that the person who sent the key used the answer A in the protocol, even though A is never communicated in any way.

2.2 Attacks

We first consider the askee impersonation attack. The meddler, though he does not know A, may make a guess G that could be equal to A, especially if the set of possible answers to Q is small. The meddler will attempt to use G to offer the fake key FK instead. If $G = A$, the asker will receive FK and be convinced that it belongs to the askee, meaning the impersonation is successful. However, if $G \neq A$, the asker will be unable to verify the askee's identity and may grow suspicious of an impersonation attempt.

A meddler who can prevent messages from being delivered could also prevent successful verification of the askee. The general problem of denial of service attacks is outside of the scope of this paper.

Alternatively, the meddler could attempt to impersonate an asker rather than the askee. The meddler chooses a question Q' and asks it of the askee. The askee does not reveal A' in the protocol, so the meddler can only learn A' if her guess G' is correct. A' is only useful information in a subsequent askee impersonation attack, and even then only if the asker chooses to use Q' as a question.

In either of these attacks, the meddler can impersonate a person who is not actually a user of the OSN by creating a fake account on the OSN with that person's information. The same attacks apply even when there is not a "real" asker or askee for the meddler to impersonate.

In order to maintain shared knowledge secrecy, the meddler must be unable to recover A from the protocol even with an offline dictionary attack. Therefore, any attempt to test whether $G = A$ must require the cooperation of either the askee or the asker, to limit the number of guesses a meddler may make.

2.3 Existing Protocols

Two existing protocols satisfy the requirements for our problem. Jablon [8] describes SPEKE, a protocol designed to establish a secure channel between a client and a server who share a common passphrase. As Jablon suggests, this can also be used with shared knowledge as the passphrase between two users. SPEKE is specifically designed to preserve the secrecy and require online verification of the passphrase. The secure channel in SPEKE can be trivially used to exchange a public key once the protocol is complete.

SPEKE achieves these properties by modifying the Diffie-Hellman protocol, replacing the ordinarily fixed primitive base with a primitive base given by a well-chosen function of the shared information. We omit further details of SPEKE.

Ellison [5] describes a multi-question protocol that also satisfies the properties we require. This protocol allows the asker to ask several questions before deciding that she is in fact communicating with the askee. Although it may prevent some honest users from exchanging keys successfully, askers and askees must limit the number of verifications they will perform to reduce the number of guesses a meddler may make.

2.4 Embedding SPEKE in an OSN

Facebook is one of many web-based OSNs, and we use it as an example of how one would augment an existing OSN to support SPEKE. Facebook provides private messages between users, which could be used as the communication channel in SPEKE.

Several steps of the protocol require local cryptographic operations that must not reveal certain information such as private keys or the answer. One can perform the SPEKE protocol on an OSN by embedding the protocol in a Firefox extension. SPEKE requires several messages, so the asker and askee must either visit the OSN simultaneously or they must interleave their visits to the OSN several times to complete the exchange. This solution is also appropriate for other OSNs; in particular, Persona [3] already relies on a Firefox extension for cryptographic operations.

In addition to key exchange with SPEKE and exclusive shared knowledge, we can increase a user's view of trusted public keys through a web of trust built on the OSN friend graph. Although users might hesitate to ascribe trust to all of their Facebook friends, they might be more willing to trust the friends they know well enough to identify through exclusive shared knowledge. We consider the benefit of using a web of trust in Section 3.2.2.

3. CAN USERS ASK GOOD QUESTIONS?

Since the security of our system relies on the ability of users to ask good questions, we perform a real-world user study to determine whether users can do so. This study presents a challenge that most user studies do not face: the results depend on getting data about both participants and their friends. Rather than bring individual users in for interviews, we perform our study directly on Facebook to take advantage of the existing friendship information that Facebook provides. We describe our user study, Bond Breaker[1], in this section.

Like many other viral Facebook applications, Bond Breaker is a social game. We wanted to ensure that users had the right goals while using Bond Breaker, so scoring in the game reflects desirable behavior in an actual system built for secure key exchange. We also believed that if the game was fun, it would encourage competition and convince users to invite their Facebook friends to participate.

3.1 Bond Breaker Game Rules

We present the rules to the users before they begin playing Bond Breaker. In Bond Breaker, users are rewarded for establishing *bonds*. A user establishes a bond by asking a question of a friend, providing an answer, and getting the friend to provide the same answer; this is analogous to successful (one-way) completion of the key exchange protocol in Section 2. Both the asker and the askee are rewarded for successfully establishing a bond, and they are also rewarded for establishing a bond in the other direction.

For example, Alice asks Bob, "Where did we meet for the first time?", and Alice and Bob answer, "a roller disco", forming a bond. Bob independently asks Alice, "What color is my bike?", and both answer "blue", forming another bond in the other direction.

As the name Bond Breaker suggests, we also encourage users to *break* bonds. A user is rewarded when she guesses the correct answer to a question that was not intended for her. A user may break a bond in this way even if the intended askee is unable to answer the question correctly, since this still corresponds to a successful attack in the actual key exchange protocol. A given asker and askee may establish only one bond at any given time: if that bond is broken, they may try to use a new question. Since we want to discourage users from asking and answering poor questions, we penalize the asker and askee whenever a bond is broken.

[1]Bond Breaker remains open and is available to use at http://bondbreaker.cs.umd.edu/

	Asker	Askee	Meddler
Creating a bond	+1 (each)	+1 (each)	-
Breaking a bond	-2 (once)	-1 (once)	+1 (each)

Table 1: Scoring in Bond Breaker. Askers and askees earn points for each bond they create and only lose points once per bond if the bond is broken. Each meddler earns points for breaking a bond, even if the bond was already broken by another meddler.

Continuing our previous example, Eve guesses the answers to Alice and Bob's questions. To the first, she guesses "high school", and fails to break their bond. To the second, she guesses "blue", breaking the bond from Bob to Alice. Unless Eve knows more information about Alice and Bob, Alice's question is good because there are many places to choose from and the answer is relatively obscure. Bob's question is not as good because the answer is easily guessed.

We reward and punish users based on a point system and include a leaderboard to give users incentive to earn points. We present our scoring rules in Table 1. The asker and askee are penalized once for having a bond broken, but arbitrarily many meddlers can earn points for breaking the same bond. We penalize the asker more than the askee when a bond is broken since the asker chose the question and has more at stake in the key exchange protocol; if the bond is established and then broken, the net result would be that the asker loses one point and the askee breaks even.

In our study, each meddler only gets one guess per question, corresponding to the requirement that askers and askees limit the number of answer verifications they will make. In practice, some users asked questions that the askee was unable to answer only because of slight formatting problems and then asked the same question again, giving meddlers an extra chance to answer. We used case-insensitive matching as the only transformation on answers and have not evaluated any other transformations. Any transformation that makes matching more lax will favor usability at the expense of security: the easier it is for friends to identify each other, the easier it will be for an attacker to guess the correct answer.

The rules that we have described provide an effective analogy between success in Bond Breaker and success in an actual system. Users are rewarded for asking questions of as many users as possible, but punished whenever those questions can be answered by a meddler; in a real system users would obtain benefit from learning many public keys and could incur substantial costs when meddlers convince them to use false public keys. By rewarding users for breaking bonds, we provide an

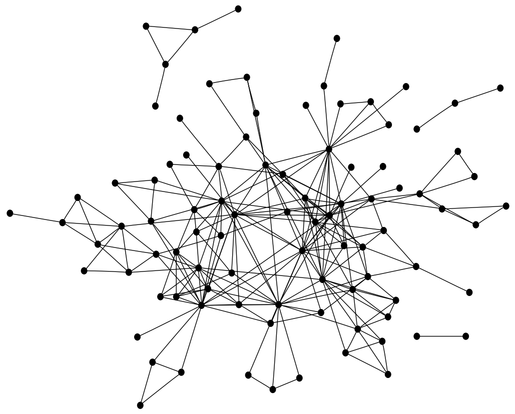

Figure 1: Friend graph for active users in Bond Breaker.

incentive to do so, just as meddlers in an actual system would have incentive to falsify public key information. We believe that Bond Breaker measures well the usability of shared knowledge for identity verification.

3.1.1 Data Collection

We opened Bond Breaker to the public on April 3rd, 2009 and collected data for three months. We primarily advertised by word-of-mouth, but also with flyers and a Facebook advertisement. In total, 171 people agreed to participate in Bond Breaker, but 70 of the participants did not ask, answer, or attempt to meddle in any of the questions. Of the remaining 101 active users, 92 chose to ask or answer at least one question while 9 chose only to meddle. In total, there were 225 questions, 200 answers from the askees, and 300 answers from meddlers.

The friend graph among participants in our study is not as densely connected as we would expect in a complete OSN friend graph. 41% of the active users had only one or two friends actively participating in the study. Through user feedback we found that this was a combination of the following: users did not want to bother their friends with what could be seen as spam invitations, the signup process required reading a detailed description of the rules, and users felt discouraged from using the application if none of their friends had signed up for it yet. In contrast, 14% of active users had ten or more friends participating; many of these users were connected to each other, forming the densely connected core in Figure 1. Most of our results do not depend on how densely connected the friend graph is, but we may be able to obtain more accurate results about the web of trust if we obtain a more complete friend graph in the future.

3.2 Results

We use the results of our user study to answer the following questions. Can users easily formulate answerable questions based on exclusive shared knowledge? How likely is an attacker to guess the answer to those questions? Finally, can we use these local verifications to bootstrap a PKI, similar to the PGP web of trust?

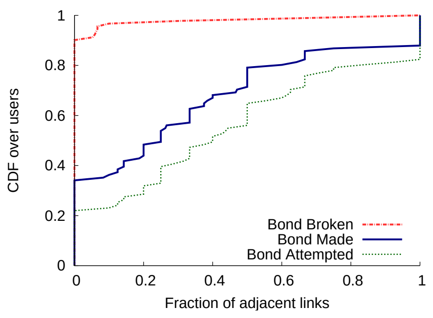

Figure 2: Fraction of friends to whom users asked questions, created bonds, and had bonds broken.

	Friend	Stranger	All
Unsuccessful	50%	44%	94%
Successful	5%	1%	6%
All	55%	45%	100%

Table 2: Breakdown of meddling attempts based on whether the meddler was a stranger or a friend and whether the meddler was successful or unsuccessful.

3.2.1 Question Success Rate

We first consider whether the participants were able to successfully use shared knowledge to establish bonds. Figure 2 shows that users had varying degrees of success in their ability to pose questions to friends: about a fifth of the users did not ask questions of any of their friends, another fifth asked questions of all of their friends, and the remainder were distributed nearly evenly. However, when users did pose a question, the friend answered correctly 69% of the time.

Unlike askees, meddlers are rarely able to answer questions, with only a 6% success rate. Table 2 shows that strangers meddled with nearly as many questions as friends. However, five out of six successful break attempts were by a friend of either the asker or askee. Though users may have spiteful friends who try to interfere with their efforts, we expect most attacks in practice to come from strangers. The ratio of successful attacks to attempts is only 9% for friends and 2% for strangers. To provide a point of comparison, web-based security questions can be answered 28% of the time by friends and 17% of the time by strangers [13]. Though this is the only comparison we can draw to actual systems, there are significant differences between the problems being solved and the experimental methodologies

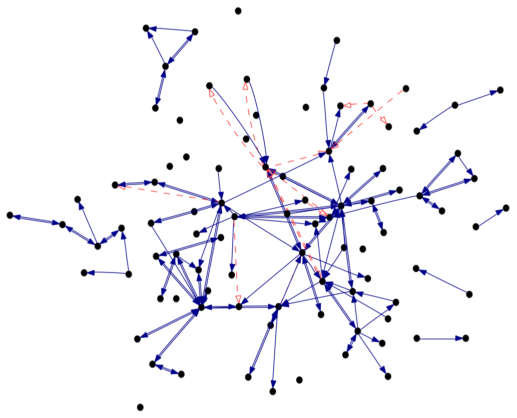

Figure 3: Bond graph in Bond Breaker; solid lines with filled arrows represent successful bonds and dashed lines with empty arrows represent broken bonds.

of these two studies, so this comparison is meant only to put the results in context.

From these results we conclude that users are only able to use shared knowledge with some of their friends, and it is usually difficult for a meddler to guess the answer to a shared knowledge question.

3.2.2 *Web of Trust*

Though we have demonstrated that many users only formed bonds with only a small fraction of their friends, we now show that users can learn the keys of other users in the OSN. Figure 3 shows the bonds and broken bonds between active users. Based on this graph, we define a user U's *web of trust* to be the set of users reached by breadth-first search on the directed bond edges beginning at user U. The web of trust may also be limited to a fixed number of hops away from U; a web of trust restricted to 2 hops would include U, any of U's friends to which U has established a bond (hop 1), and any user reachable from those users (hop 2). Restricting the web of trust sacrifices graph coverage for the sake of security. Our definition assumes that trust is related to hops in the friend graph, but in practice we advocate the use of explicit, user-defined trust information.

With a web of trust, the user can do two things: discover the identities of users she does not bond with first-hand, and detect when an attacker has falsified an identity. Since most OSN communication is between friends or between friends-of-friends (FoFs), we focus on learning the keys of those users.

We first show a CDF of the fraction of friends and friends-of-friends reachable via a web of trust in Figure 4. If we restrict the web of trust to 2 hops, meaning that the user trusts her friends to attest to keys belonging to her FoFs, 18% of users can identify more than half of their friends or FoFs in the OSN. However, if we do not restrict the web of trust, half of the users with at least one outgoing bond can reach at least 73% of their friends or FoFs. This suggests that the web of trust is a powerful tool in creating a PKI for friends and FoFs and that the study of trust in OSNs deserves further research.

We also discovered that 12 of the 15 unique broken bonds could be detected by the unrestricted web of trust. That is, for 80% of the broken bonds, there is a path of good bonds from the asker to the askee in the unrestricted web of trust. We can use this feature of the web of trust to reduce impersonation in an OSN. Our results use Facebook account ids to identify nodes in the web of trust, but an impostor could create a fake account to thwart this. Figure 5 shows an example of how the web of trust could fail: the path of good bonds (Carl to Bob to Alice) points to a different node in the friend graph than the broken bond (Carl to Ivy), so Carl will be oblivious to Ivy's attack if he cannot discover that the accounts created by Alice and Ivy refer to the same offline identity. In order to use the web

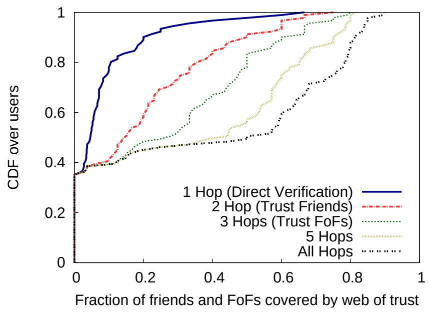

Figure 4: CDF of the fraction of friends and FoFs reachable through the web of trust, by web of trust restriction. 36% of the users have no outgoing bonds, so they gain nothing from the web of trust.

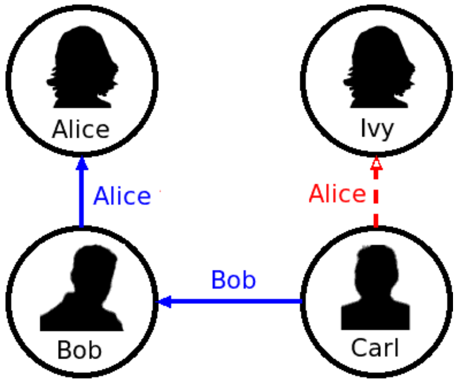

Figure 5: Carl incorrectly believes that the account created by Ivy belongs to Alice, and his web of trust does not detect his error.

of trust to detect impostors, bond edges must therefore encode information about offline identity to be able to match fake accounts to their real counterparts.

These results demonstrate that we can bootstrap a PKI that provides the keys of friends and FoFs in an OSN. This PKI is distributed and decentralized; we do not require a centralized authority and we can exchange keys entirely online, as opposed to the offline key signing parties of PGP. This PKI could become a critical tool for providing security and privacy in emerging OSN-based applications.

4. RELATED WORK

Persona [3] is a distributed OSN that provides user-defined privacy through cryptography, but relies on the existence of a PKI even as it eschews a centralized OSN provider. The distributed PKI we describe, in which the local user identifies her own friends, is a natural solution for the PKI in Persona as it is entirely online and does not require a central authority.

Toomim et al. [16] show that shared knowledge could be used as an alternative to group-based access control in OSNs. In their work, users protect OSN content by guarding it with a question; only users who can answer the question can access the content. In contrast, we use *exclusive* shared knowledge to verify identity rather than group membership, and exchange cryptographic keys on which access control can be built.

The third-party application *My Public Key* [9] provides mechanisms for encryption and signatures on Facebook, but the application does not verify user identity and keys could be falsified by a man-in-the-middle attacker.

Studies of OSN security show that current methods of identifying users are insufficient. Bilge et al. [4] describe an attack in which the attacker copies a victim's information from one OSN to another to impersonate the victim on the new OSN. This allows the attacker to befriend the victim's friends, learn information about them, and continue the attack on those new users. Key exchange with shared knowledge could be used to prevent such attacks; the attacker may have access to personal information, but not exclusive shared knowledge. Felt [7] also describes an exploit for hijacking Facebook accounts (which has since been patched). Our work could be used to detect hijackings and repair them.

Alexander et al. [2] describe a modification to off-the-record (OTR) instant messaging that allows users to authenticate each other using shared secrets. Stedman et al. [15] study how a small group of users interact with the modification. Our work instead considers shared knowledge in the broader arena of OSNs and quantitatively demonstrates the ability of users to employ shared knowledge.

5. CONCLUSION

Impersonation is a fundamental problem of OSNs. We have described how to use exclusive shared knowledge to allow users to take responsibility for identifying their own friends in an OSN in a completely online way. We have described a user study, Bond Breaker, that takes takes advantage of existing Facebook friend information to study our idea in a real setting. We have demonstrated through Bond Breaker that exclusive shared knowledge is a practical tool for identifying friends in an OSN and that users can establish a PKI among their friends and friends-of-friends with a web of trust.

Though Bond Breaker reveals the potential of exclusive shared knowledge, it does not test the extent of its use. In both the feedback we received from participating users and from our own experience with the Bond Breaker application, we observed that (1) users have trouble creating questions with difficult-to-guess answers, but (2) users *can* come up with many weak questions that collectively are a thorough test of the askee's shared knowledge. To facilitate the identification of friends, we should take advantage of the weak shared knowledge that users possess more abundantly. We believe that the use of multiple identifying questions may be able to bridge the gap between the results we have presented and a complete PKI, and leave this as a topic of future research.

6. REFERENCES

[1] ABC News. http://www.abcnews.go.com/Technology/Story?id=7960020.

[2] C. Alexander and I. Goldberg. Improved user authentication in off-the-record messaging. In *WPES*, 2007.

[3] R. Baden, *et al.* Persona: An online social network with user-defined privacy. In *SIGCOMM*, 2009.

[4] L. Bilge, *et al.* All your contacts are belong to us: automated identity theft attacks on social networks. In *WWW*. ACM, 2009.

[5] C. M. Ellison. Establishing identity without certification authorities. In *SSYM*. USENIX Association, 1996.

[6] Facebook Statistics. http://www.facebook.com/press/info.php?statistics.

[7] A. Felt. Defacing facebook: A security case study. White paper, UC Berkeley, 2007.

[8] D. P. Jablon. Strong password-only authenticated key exchange. *SIGCOMM Comput. Commun. Rev.*, 1996.

[9] My Public Key Facebook Application. http://apps.facebook.com/mypublickey/.

[10] PC World. http://www.pcworld.com/article/168462/.

[11] A. Rabkin. Personal knowledge questions for fallback authentication: security questions in the era of facebook. In *SOUPS*. ACM, 2008.

[12] A. V. Ramachandran and N. Feamster. Authenticated out-of-band communication over social links. In *WOSN*. ACM, 2008.

[13] S. Schechter, A. J. B. Brush, and S. Egelman. It's no secret: Measuring the security and reliability of authentication via 'secret' questions. In *SOSP*, 2009.

[14] W. Stallings. The pgp web of trust. *BYTE*, February 1995.

[15] R. Stedman, K. Yoshida, and I. Goldberg. A user study of off-the-record messaging. In *SOUPS*, 2008.

[16] M. Toomim, *et al.* Access control by testing for shared knowledge. In *CHI*. ACM, 2008.