FaceTrust: Assessing the Credibility of Online Personas via Social Networks

Michael Sirivianos
Duke University
msirivia@cs.duke.edu

Xiaowei Yang
Duke University
xwy@cs.duke.edu

Kyungbaek Kim
University of California, Irvine
kyungbak@uci.edu

ABSTRACT

Despite the large volume of societal interactions taking place on the Internet, it is still hard to assess the credibility of statements made by online users. The digital credentials issued by trustworthy certificate authorities partially address this problem, but a tedious registration and verification process as well as its high cost hinder the wide adoption of this solution. This paper presents FaceTrust a system that leverages online social networks to provide lightweight, flexible and relaxed credentials that enable users to assess the credibility of others and their assertions.

1. Introduction

Nowadays, rich social interactions take place online. Users read, shop, chat, or even play online. Yet the Internet has largely hidden the identity attributes of online users. "On the Internet, nobody knows you are a dog," says the famous Peter Steiner cartoon. When treading through the Internet jungle, what to believe and whom to believe remains a formidable challenge to Internet users. Unscrupulous users may easily become victims of online scams. There have been numerous incidents where scammers defrauded users [4, 18, 20] or even organizations [19] through email or online social networks to obtain sensitive information. Users with vested interest in a company have been caught creating fake positive reviews for the company's products or services [16, 17]. Pedophiles may lie about their ages in online chatrooms, and on the other hand, underage users may also lie about their ages to gain access to age-restricted websites.

This problem stems from the fact that there is currently no lightweight and effective way to assess the *credibility* of assertions/statements made by online personas. In this paper, we refer to credibility as a measure of the likelihood that a user's assertion is correct or true. The current paradigm to address this problem is to use digital credentials issued by trustworthy authorities. Users use them to authenticate at verifying online services. For example, Digital Certification Authorities such as VeriSign [10, 14] can validate a user's identity (name, and domain name) and issue a certificate that assures the user's identity.

These proposals, albeit effective, are heavy-weight and not easily extensible. They often require a tedious registration process that involves manual verification. Obtaining one is not only time consuming but also expensive. As an example, a class one digital certificate from a trustworthy CA costs about $20 [15]. In addition, a digital credential only certifies a limited subset of a user's attributes, such as the Internet domain or name. It does not quantify the general credibility of its holder's assertions, *e.g.*, online reviews or ratings in recommendation sites such as epinions.com.

This paper presents FaceTrust, a system that enables online users or services to assess the credibility of other online personas and their assertions. FaceTrust achieves this goal by mining and enriching information embedded in Online Social Networks (OSNs) such as Facebook [47]. FaceTrust extends an OSN to provide lightweight, extensible, and relaxed digital credentials through *social tagging*. We observe that OSNs already allow users to express a limited form of trust relationships using friend links. FaceTrust extends this ability by allowing users to tag friend links with how credible they consider their friends' assertions (such as their profiles). For instance, a user that wishes to obtain an age certificate from his OSN provider may state that he is above 18 years old in his profile. The OSN provider will request his friends to tag this assertion with a credibility value. The OSN provider analyzes the annotated social graph to obtain the overall credibility of a user's assertion. It can then issue a credential in the form of (assertion, credibility). Websites or other online users may use this OSN-issued relaxed credential to inform their interactions with this user.

FaceTrust's design aims to use social tagging to replace centralized, heavyweight, and restricted verification of user credentials. We face several main challenges in realizing this vision. First, what types of assertions is social tagging able to verify reliably(§2.1)? Second, how can an OSN provider extract the overall credibility information and export it to verifiers without violating a user's privacy (§2.2, 2.4)? Third, how can an OSN provider effectively and efficiently analyze a multimillion-node social graph to extract robust credibility assessments (§2.2, 2.3, 4.2)? Lastly, how can we evaluate the feasibility and performance of such a design (§4)? The main body of this paper describes our initial approaches towards addressing these challenges.

2. FaceTrust Design

Figure 1 presents an overview of FaceTrust. The FaceTrust design consists of three main components: a) an OSN provider that maintains the complete social graph and its users' profiles; b) online users that maintain an account with the OSN and attempt to access other online services or communicate with other users by presenting OSN-issued credentials; and

c) verifying services or users that regulate access to their resources or characterize user inputs based on their credentials. Next, we describe each component of FaceTrust and how they interact in more detail.

2.1 Social Tagging

In FaceTrust, OSN users annotate the social links between them and their friends with additional credibility information. By "social tagging", we refer to the process in which an OSN user assigns a credibility value to a friend or the assertions made by that friend. Tagged values are only known to the OSN and the taggers. Tagging is requested only for users that wish to obtain credentials from the OSN.

FaceTrust relies on social tagging to assess the credibility of online personas. Our assumption is that people usually do not lie on behalf of others. Therefore, the collective information gathered from a user's social acquaintances is likely to correlate positively with the truth. Of course, this assumption does not hold if one user can create multiple fake OSN accounts, an attack known as Sybil attacks [28], For clarity, we assume that each OSN account corresponds to a unique user for the moment, and describe how to mitigate Sybil attacks in § 2.3.

The FaceTrust design categorizes user assertions into different types such as age, address, profession, etc. A user stores his assertions of assorted types in his OSN profile. For instance, for the type age, a user may assert that he is older than 18; for the type profession, he may assert that he is a faculty member at Duke University. For each assertion type k made by a user j, j's friend i may tag a direct credibility score as d_{ij}^k. Moreover, the user i also tags his friend j with a transitive trust score t_{ij} that indicates how much i trusts j to correctly assess j's friends' assertions of all types.

In our current design, assertion types are mainly identity attributes present in a user's OSN profile, but they can be extended to include other attributes' of users such as fields of expertises. Both direct credibility and transitive trust scores can be discrete values such as "highly trustworthy," "fairly trustworthy," and "untrustworthy" for usability purposes. Presently, we make them continuous values between $[0,1]$ for ease of analysis.

We note that the FaceTrust design assumes that users are willing to tag their friends. There is abundant evidence that suggests such social tagging may be adopted by users. For example, the "Best-friends" [3] Facebook application enables users to tag and order their friends and has amassed $330K$ active users. It is our future work to conduct a usability study to validate the adoptability of social tagging.

2.2 Assessing Credibility

In the FaceTrust design, an OSN provider plays the role of inexpensive and relaxed credential-issuing authorities. By relaxed, we mean that unlike a conventional certificate authority, the OSN does not guarantee that an assertion is absolutely correct. Instead, each credential is associated with a credibility metric that either resembles "wisdom of crowds"

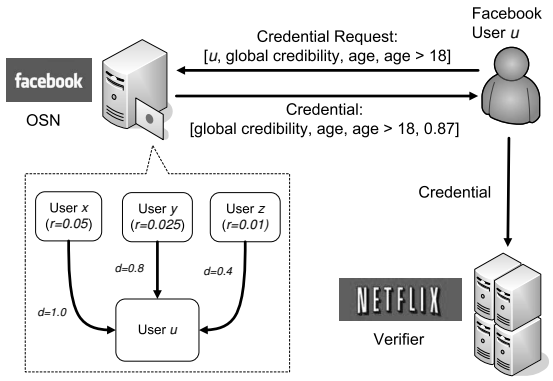


Figure 1: **FaceTrust architecture and an age verification example. We use d to denote direct credibility and r to denote SocialRank.**

(global credibility § 2.2.1), or the pairwise transitive trust between a verifier and a user (pairwise credibility § 2.2.2). An OSN provider analyzes an annotated social graph to assess the credibility of a user's assertion, and issues a credential that includes the assertion and its credibility score. For instance, in Figure 1, the OSN provider issues a credential that says the global credibility score of the user assertion "age $>$ 18" is 0.87. We describe how an OSN might obtain the global and pairwise credibility assessments in the next.

2.2.1 Global Credibility

Let F_j denote the set of friends a user j has. To compute a global credibility score (g_j^k) on a user j's assertion of type k (A_j^k), FaceTrust computes the weighted average among the direct credibility measures of all j's friends:

$$g_j^k = \sum_{i \in F_j} w_{ij} \cdot d_{ij}^k / \sum_{i \in F_j} w_{ij} \quad (1)$$

User ratings are weighted by w_{ij} because they are not equally credible, *e.g.*, a teenager's rating on another teenager's age assertion should carry less weight than those from his parents.

How can FaceTrust reliably determine the weights w_{ij}? To address this issue, we design a social graph analysis algorithm, SocialRank, to rank the trustworthiness of a user i among all other OSN users based on the transitive trust scores users assign to each other. SocialRank is similar to PageRank [22], which is an iterative algorithm that computes the likelihood that a random walk following the hyperlinks ends up at a page. In SocialRank, we replace PageRank's directed hyperlinks with directed and weighted social links that correspond to the transitive trust assignments from a user i to another user j. Let V be the set of nodes in the annotated social graph. Formally, a user i's social rank r_i at each iteration is computed as:

$$r_i = \frac{1-g}{|V|} + g \sum_{x \in F_i} \frac{r_x \cdot t_{xi}}{\sum_{y \in F_x} t_{xy}} \quad (2)$$

The parameter g is a damping factor, which we set to 0.9. To initialize the iterations, an OSN provider selects a few

(W) well-connected users and assigns $\frac{1}{W}$ social rank to each. The computation terminates when the difference between the current social rank vector's norm and the norm in the previous iteration is below a specified threshold.

Intuitively, the higher a user's social rank r_i is, the higher its weight w_{ij} should be. But we can not simply set w_{ij} to r_i, because a user with a large number of low-ranked friends may also obtain a high credibility score. To mitigate this issue, FaceTrust assigns a zero weight to a user i if its social rank r_i is in the bottom b percentile of the entire user population. This design assumes that the bottom b percent of users are untrustworthy. For the rest of the users, the weight w_{ij} of a user i in determining another user j's global credibility (Eq. 1) is set to its social rank: $w_{ij} = r_i$.

2.2.2 Pairwise Credibility

Global credibility has the drawback that a user's credibility is not directly determined by how much a verifier trusts the friends of the user. In certain cases, *e.g.*, a user j invites a user v to become his friend in an OSN, it may be desirable for the verifier v to use a trusted social path from itself to verify j's assertions. This is a robust technique for v to prevent manipulation from malicious users that collude to boost j's credibility. To this end, FaceTrust's design also includes an algorithm to compute the pairwise credibility p^k_{vj}, which assesses the credibility of a user j's assertion A^k_j from the verifier v's point of view.

Pairwise credibility relies on the assumption that trust is largely transitive [31,32]. There have been various proposals to model how trust propagates through a network with a discounting factor, *e.g.*, Guha et al. [32] employ matrix multiplications. Inspired by Credence [45], we use the maximum trust path between the verifier and the user. We choose this approach because the maximum trust path can be computed efficiently using Dijkstra's algorithm.

To compute p^k_{vj}, the OSN provider finds the path P_i from v to $i \in F_j$ such that $\Pi_{m \to n \in P_i}\, t_{mn} \cdot d^k_{ij}$ is the largest. The pairwise credibility p^k_{vj} is set to this product. This design assumes that the most useful credibility assessment is the maximum one. Alternatively, to compute p^k_{vj}, the OSN provider can find the friend i of a user j that the verifier trusts the most. That is, it finds the path P_i from v to $i \in F_j$ for which $\Pi_{m \to n \in P_i}\, t_{mn}$ is the largest. The pairwise credibility p^k_{vj} is set to $p_{vi} \cdot d^k_{ij}$. This alternative design assumes that the most useful credibility assessment is the one by the most trusted friend of j. Our implementation uses the first definition for its simplicity.

2.3 Mitigating Sybil Attacks

When malicious users create numerous fake online personas, FaceTrust's credibility assessment can be subverted. For instance, SocialRank is a variation of PageRank and shares its vulnerabilities, but Cheng et al. [25] have shown that PageRank is manipulable using Sybil strategies. A malicious user a with high pairwise credibility with another user u may create Sybils and assign high transitive trust or credibility to them and their assertions. As a result, all the Sybils of the attacker would gain high pairwise credibility with user u. Furthermore, an attacker may create Sybil friends that assign high credibility to it.

We therefore need a mechanism to combat Sybils when obtaining the pairwise or global credibility. Fortunately, there exist effective algorithms that that can mitigate Sybil attacks on social graphs. These algorithms take advantage of the feature that most social network users have a one-to-one correspondence between their social network identities and their real-world identities. Malicious users can create many identities or connect to many other malicious users, but they can establish only a limited number of trust relationships with real humans. Thus, clusters of attackers are likely to connect to the rest of the social graph with a disproportionately small number of edges. The first systems to exploit this property were SybilGuard and SybilLimit [49, 50], which bound the number of Sybil identities using a fully distributed protocol.

FaceTrust adapts the SybilLimit algorithm to determine an identity-uniqueness score q_j for each user j. The value of q_j is between 0 and 1, indicating the likelihood that an OSN user j corresponds to a unique user in the real life. To be Sybil-resistant, FaceTrust multiplies the identity-uniqueness score q_j to obtain the final global or pairwise credibility score of a user assertion A^k_j: $g^k_j \leftarrow g^k_j \cdot q_j$ or $p^k_j \leftarrow p^k_j \cdot q_j$.

First, we provide informal background on the theoretical justification of SybilGuard and SybilLimit. It is known that randomly-grown topologies such as social networks and the web are fast mixing small-world topologies [21, 33, 46]. Thus in the social graph $\mathcal{S}(\mathcal{V}, \mathcal{E})$, a walk of $\Theta(\sqrt{|V|}\log|V|)$ steps contains $\Theta(\sqrt{|V|})$ independent samples approximately drawn from the stationary distribution. When we draw random walks from a verifier user v and the suspect s, if these walks remain in a region of the network that honest (non-Sybil) users reside, both walks draw $\Theta(\sqrt{|V|})$ independent samples from roughly the same distribution. It follows from the generalized Birthday Paradox [50] that they intersect with high probability. The opposite holds if the suspect resides in a region of Sybil attackers that is not well-connected to the region of honest users.

SybilGuard replaces random walks with "random routes" and a trusted verifier user accepts the suspect if random routes originating from both users intersect. In random routes, each user uses a pre-computed random permutation as a one-to-one mapping from incoming edges to outgoing edges. Each random permutation, generates a unique routing table at each user. As a result, two random routes entering an honest user along the same edge will always exit along the same edge ("convergence property"). This property guarantees that random routes from a Sybil region that is connected to the honest region through a single edge will traverse only one distinct path, further reducing the probability that a Sybil's random routes will intersect with a verifier's random routes

SybilLimit [49] is a near-optimal improvement over the SybilGuard algorithm. In SybilLimit, a user accepts a suspect only if random routes originating from both users intersect at their last edge. For $o(\sqrt{|V|}/log|V|)$, attack edges, SybilLimit bounds the number of Sybils that are accepted for each attack edge to $O(\log|V|)$, while SybilGuard bounds it to $O(\sqrt{|V|}log|V|)$ For two honest users to have at least a shared last edge with high probability, the required number of the random routes from each user should be approximately $r = \Theta(\sqrt{|E|})$. The length of the random routes should be $w = O(\log|V|)$.

FaceTrustdetermines the uniqueness of a social network user's identity using a version of SybilLimit that can be efficiently computed in a large cluster of an OSN provider. This algorithm is executed solely by the OSN provider over the social graph $\mathcal{S}(\mathcal{V},\mathcal{E})$. At initialization time, the OSN picks a small set $V_v \subset V$ of random trusted verifier users, where $|V_v| = 100$. It also creates $r = 3sqrtE$ independent random permutations. All suspects share the same r permutations For each user $s \in V$, the OSN performs the following steps:

1. For each of the verifiers $v \in V_V$, pick a random neighbor of v. Draw along the random neighbor r random routes of length $w = O(\log|V|)$, for each instance of the r permutations. Store the last edge (tail) of each verifier's random route.
2. Pick a random neighbor of s and draw along it r random routes of length $w = 3\log|V|)$, for each instance of the r permutations. Store the last edge (tail) of each random route. We refer to steps (1) and (2) of the algorithm as *random routing.*
3. For each verifier $v \in V_v$, if one tail from s intersects one tail from v, the verifier v is considered to "accept" s. We refer to this step as *verification.*
4. Compute the ratio of the number of verifiers that accept s over the total number of verifiers $|V_v|$. That ratio is user's s identity uniqueness score id_s.

The OSN performs the above computations off-line and periodically to accommodate for social graph changes. The OSN stores the result of this computation for each user as a separate attribute.

2.4 OSN-Issued Credentials

For an open system such as the web to operate reliably, assesment of the credibility of user statements is of the utmost importance. Existing Certification Authorities (CA) are not suitable for open, large scale networks because they require users to pay a certification fee (∼$20 for class 1 certificate). Even when the certification process requires additional steps for authentication (e.g. class 3 digital certificate), in practice manual verification can also be error-prone [5]. In addition to CAs being expensive, they currently represent a monopoly for a very important Internet primitive, and we consider breaking this monopoly beneficial.

Moreover, in many realistic Internet settings absolute accuracy of the credentials is not required. It is critical to use accurate credentials issued by authentication schemes such as Kerberos [42] or OpenID [12] to control access to an email account. It is also important for a bank to obtain a highly trusted (and expensive) certificate using the standard PKI infrastructure [10] so that it prevents man-in-the-middle attacks. On the other hand, the FaceTrustcredential system is suitable for use cases such as providing a relying service (*e.g.*Netflix) with a measure of the probability that a user is old enough to download an R-rated film. Another example of a suitable FaceTrustuse case is for informing a user who posted a classifieds ad that the person responding to it does not reside in a country associated with an advance-fee fraud [1].

After an OSN obtains the credibility scores for a user j's assertion A_j^k, it can issue a credential for this assertion of j. As shown in Figure 1, a credential issued by an OSN will include the assertion type k, the assertion A_j^k, the type of credibility score, and the credibility score.

A credential must be authenticated by cryptographic primitives such as an OSN's public key signature. In the FaceTrust design, we use the *idemix* [23] anonymous unlinkable credential system because users may desire to preserve their anonymity and untraceability of online activities. The *idemix* system is based on an efficient non-transferable anonymous and unlinkable credential scheme introduced by Camenisch et al. [24]. An *idemix* credential does not reveal any identifying information of a user that possesses the credential, which is ideal for online verifications such as age checking. It also prevents one user from transferring his credentials to other users.

When a verifier v requests a credential from a user u with a global credibility score, it does not need to reveal any identity information to u. However, when a verifier v requests a credential with pairwise credibility from a user u, the verifier v must reveal its social network ID to the user u so that the OSN can compute the pairwise credibility p_{vu}^k. To prevent revealing its true identity, a verifier should use a pseudonym that by itself does not reveal anything about his real identity, *e.g.*, a Facebook 64-bit semantic-free user ID. In this design, a user u can remain anonymous and unlinkable, but the verifier v can be surveyed if he requests credentials from multiple users. We do not expect that this will become an issue for deployment, because the verifier can choose to request a credential with global credibility to remain unlinkable, or because the multiple users with whom a verifier v interacts may not be aware of each other.

Figure 2 presents an overview of the idemix system. Particular to the FaceTrustsetting, the role of the certificate issuer in assigned to the FaceTrustOSN provider, which extracts the credibility of user statements by analyzing the social network. The OSN provider in our scheme acts as a Pseudonymous Certification Authority, *i.e.*the user does not have to register with its real identity, but has the option to register only a few of its real attributes. The role of the verifier is assigned to the relying service. A user u can obtain a

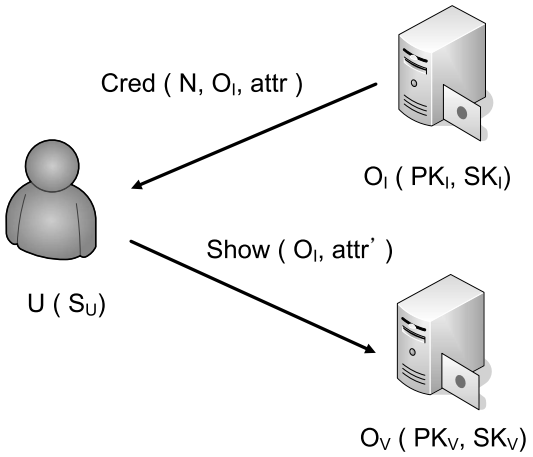


Figure 2: High level description of idemix anonymous credential scheme.

credential *cr* from the issuer, and subsequently present *cr* to a verifier relying service. A credential is always issued on a pseudonym *n* under which *u* is registered with the issuer . A user *u* has one master secret key s_u, which is linked to all the pseudonyms and credentials issued to that user.

The credential concerns an assertion made by *u*. The issuer assesses the credibility of the assertion and assigns a credibility value to it. The issuer of the credential has a public/private key pair. The issuer uses its private key to issue the credential *cr* to *u*. *cr* contains the pseudonym *n*, the assertion and its credibility as assessed by the issuer, which is the FaceTrustOSN provider.

The verifier also has a public/private key pair. When the user presents a credential to the verifier, she uses the public key of the verifier. The verifier then uses its private key and the public key of the issuer to verify the credential. At the core of the desirable nature of the protocol is the fact that *u* does not send to the verifier its pseudonym *n*. Instead *u* convinces the verifier that the credential is issued to her, using its master secret. Thus the protocol ensures the unlinkability between distinct presentations of the credentials as well as between the credential and the pseudonym used. Therefore, the user remains anonymous to the verifier.

The certifying OSN provider can either be the provider of existing popular OSN (e.g. Facebook) or a third-party OSN application (e.g. Facebook application [6]) provider. In the first case, the popular OSN provider has access to the complete social network, and augments the OSN application API to allow applications to query the identity trust of the OSN's users. OSN providers are incented to provide this service because it adds value to their service, making the service more attractive to subscribers. In the second case, a third-party deploys FaceTrust as an OSN application and has access only to the social network of users using FaceTrust. Although the FaceTrust-only social network is not as complete as the OSN provider's one, the fact that it is smaller makes its analysis less computationally expensive. In addition, it does not require the adoption of the service by the OSN provider. For ease of exposition, for the rest of the paper we refer to both the popular OSN provider and the third-party OSN application provider as *OSN*.

3. Sample Applications of FaceTrust

Next, we briefly describe how FaceTrust's global and pairwise credibility can be used in various scenarios. Unlike Certificate Authorities [10] or access control systems such as Kerberos [42] and Shibboleth [29], OSN-issued credentials can not be completely trusted. However, we argue that in many real world situations, they provide valuable and otherwise unavailable information for access control and identity verification.

3.1 Age Verification

A concrete example of FaceTrust credentials is age verification in online settings. Figure 1 shows an example. User *u* attempts to access an age-restricted movie at the Netflix website. At the same time, *u* is concerned with his anonymity and does not wish to reveal neither his real identity nor a linkable pseudonym to Netflix.

Since Netflix as an entity may not be a user of an OSN, it will demand an age crendential with a global credibility score. To obtain this credential, user *u* issues a credential request for global credibility of assertion type "age". This request typically happens before *u* attempts to access an online service. User *u*'s friends on the OSN who have seen this assertion can judge whether it is correct, and tag this assertion with a credibility score.

In Figure 1, there are three friends that have tagged *u*'s age credibility. These friends, users *x*, *y*, *z*, have social ranks equal to 0.05, 0.025 and 0.01 respectively. In addition, users *x*, *y*, and *z* assign direct credibility scores to *u*'s age assertion ("I am 21"): 1.0, 0.8 and 0.4, respectively. The global credibility of *u* is derived using Equation 1. At this point, the OSN provider considers that there is 0.87 probability that *u*'s age assertion is correct, assuming that *u*'s identity uniqueness has been determined to be 1.

The OSN compares the assertion for which the credential is requested ($age > 18$) with the assertion *u*'s friends have tagged. It determines that the requested assertion does not contradict the assessed one ($21 > 18$). Consequently, the OSN issues the credential depicted in Figure 1 and *u* presents it to Netflix. Netflix uses this credential to decide whether to allow *u* to download the movie.

3.2 Fraud Detection

FaceTrust can help a user to detect malicious intentions when he is contacted by an unknown user. Scammers commonly respond to online postings alleging to be prospective participants in legitimate transactions but in reality aiming to commit "advance-fee" fraud [11]. Misbehaving users may create fake OSN personas to defraud other users [18, 20].

Such attacks could be averted if scammers were unable to lie about their location, affiliation, age,

In this example, suppose a user v receives a message from u in response to his Craigslist posting offering a room to rent. Both users are members of an OSN that supports FaceTrust. User u is actually a scammer residing in a suspicious country. On the other hand, u claims in his message that he currently resides in US.

With FaceTrust, user v can require u to present a credential on the pairwise credibility regarding u's location: p_{vu}^{loc}. If v is not worried about other users tracking his activities (because other users may not know each other), he can release to u his semantic-free OSN ID. User u then requests from the OSN a pairwise credibility credential with respect to v. For this example, we assume that the maximum trust path is $v \rightarrow x \rightarrow y \rightarrow u$, and the transitive trust scores t_{vx} and t_{xy} are 1.0 and 0.5, respectively. Also, y's direct credibility rating on u's location d_{yu}^{loc} is 0.3. According to § 2.2.2, the pairwise credibility p_{vu}^{loc} is 0.15. Subsequently, the OSN issues the following credential for u:

$$[\textit{pairwise credibility}, \textit{location}, \textit{USA}, 0.15]$$

User u can then present the credential to v. Likely, the user v will be alarmed by u's low credibility and refrain from any further dealings with u.

4. Preliminary Evaluation

To gain a better understanding on how our initial design works, we would like to evaluate all of the following aspects of FaceTrust:

- Effectiveness: How well do credibility scores correlate with the truth?
- Computational feasibility: A social network may consist of several hundreds of millions of users. Will an OSN provider have sufficient computational resources to mine the social graph and derive credibility measures?
- Robustness: How robust is the design in withstanding various malicious attacks or incorrect user tagging?
- Usability: How often and how accurate will a user tag his friends to help them obtain credentials?

It will take a full system implementation and experimentation on a real-world OSN to answer these questions. The question regarding effectiveness is particular difficult, because trust is inherently subjective, and it might not even be feasible to obtain the ground truth. In this section, we describe our preliminary approaches to evaluate the effectiveness and feasibility of the design. We defer the robustness and usability study to future experimentations on an OSN such as Facebook.

4.1 Effectiveness

4.1.1 Effectiveness of Pairwise Credibility

FaceTrust's design uses the SocialRank algorithm to derive global credibility, and the maximum trusted path to derive pairwise credibility. As the SocialRank algorithm is similar to PageRank, which has been shown to be effective in ranking pages regarding their relevance to users, we assume that it is also effective in ranking users they are trustworthy. Real-world experimentation is required to validate this hypothesis, *i.e.*, to verify whether the highly ranked users are indeed more likely to make a correct rating than lowly ranked ones. We defer this study to future work.

As a first step, we evaluate the effectiveness of pairwise credibility (§ 2.2.2) in verifying identity assertions made by online personas, using a well-known and deployed social graph that embed identity assertions: the PGP Web of Trust (WoT) [7]. In WoT, each user x corresponds to a unique public key. Each edge between x and another user y corresponds to a signature by x on y's public key. The signature includes an annotation indicating the level of trust x places on y, both in terms of y's identity and in terms of y's ability to evaluate the identity of others.

Recall that pairwise credibility is derived from the maximum transitive trust. Thus, we use the WoT graph to study how effective the maximum transitive trust is in predicting the true trust level between two users. To do so, we randomly remove an edge between a node x and y ($x \rightarrow y$) in the WoT graph. We then compute the maximum transitive trust p_{xy} as described in § 2.2.2 between x and y, and compare p_{xy} with the original trust level t_{xy}. If they match, it indicates that transitive trust is effective in predicting the truth.

The WoT data we use are PGP key certificates stored at the Swiss keyserver [13]. The data set represents a snapshot of the WoT between July 19 2008 and July 21 2008, and includes 39625 keys and 398660 signatures. The trust levels specified in a certificate have four discrete values: unknown, untrusted, marginal, and full. We map the discrete trust levels to credibility values in $[0,1]$ to facilitate transitive trust computation. The level "unknown" and "untrusted" are both mapped to 0; the level "marginal" is mapped to 0.5; and the level "full" is mapped to 1.0. The percentages of full, marginal, and unknown/untrusted edges in the WoT graph are 29.47%, 6.8%, and 63.73%, respectively.

Each signature has a certificate level which corresponds to the signer's trust on the identity of the signee. There are 10 levels (0/1/2/3/a/b/c/d/?/!), but in practice only the first eight are used. We treat levels 0/1/2/3 and a/b/c/d in an identical manner because they are different only in terms of which ID of the key they represent - a key can be used with multiple user IDs if the user wants to use the key in multiple contexts. Level 0/a is uncertain, 1/b, is untrusted, 2/c is marginal, level 3/d is full. The number of signatures for 0/a, 1/b, 2/c and 3/d are 253187, 872, 27104 and 117497, respectively.

Table 1 depicts how well transitive trust predicts the original trust level for each removed edge $x \rightarrow y$. The results are averaged over 1000 randomly removed edges for each trust level. Note that if a node y has only one incoming edge $x \rightarrow y$, we do not remove that edge because y will be dis-

Orig. Trust Level	1.0	0.5	0.0
Exact (%)	92.6	22.9	68.1
Fair (%)	1.7	70.2	3.7
Wrong (%)	5.7	6.9	28.2

Table 1: Prediction performance of transitive trust.

connected from the WoT graph and we can not use transitive trust to predict the trust level on $x \rightarrow y$. In the table, "Exact" means that the maximum transitive trust p_{xy} between nodes x and y is exactly the same as the original trust value t_{xy} on edge $x \rightarrow y$. "Fair" means that the obtained pairwise trust value has fairly accurately predicted the original trust level. For the full trust level 1.0, a prediction above or equal 0.5 is considered a fair match; for the marginal trust level 0.5, a prediction above 0.5 is considered fair; for the untrusted/unknown trust level 0.0, any value below or equal to 0.5 is considered fair. Predictions in all other ranges are considered "Wrong."

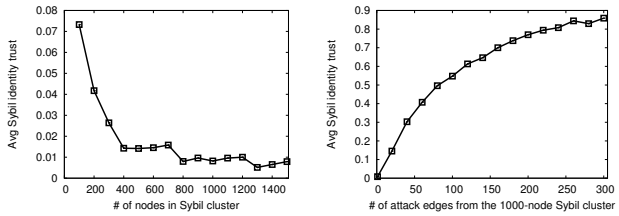
We observe that maximum transitive trust predicts the original trust level satisfactorily. For the original trust level 1.0 and 0.0, the transitive trust prediction has 92.6% and 68.1% exact matches respectively. For the original trust level 0.5, the transitive trust prediction yields 70.2% marginal matches. These results suggest that pairwise credibility computed using transitive trust is likely to predict the level of credibility that a user x itself would directly assign to a user y, if x and y were acquainted.

4.1.2 Effectiveness of Identity Uniqueness

We now evaluate how effective FaceTrust's identity uniqueness primitive is in mitigating Sybil attacks. In our evaluation, Sybil attackers form a single well-connected cluster. The honest region consists of a 100K-node sample of the Facebook [47] social network. This Sybil cluster has a random graph topology under which Sybil nodes have average degree 14. The Sybil cluster is connected to the honest region through *attack edges* between a Sybil node and an honest node. We vary the number of nodes in the Sybil cluster from 100 up to 1500, as well as the number of attack edges from 1 up to 300. We set the length w of random routes equal to 17, the number of routing tables r at each node equals 2600 and the number of verifiers l equal to 100. With these settings, the average identity uniqueness of 1000 randomly chosen honest nodes is 0.89 with 0.10 standard deviation.

In Figure 3(a), we observe that as the number of nodes in a Sybil cluster increases, the average identity uniqueness of Sybil nodes decreases and stabilizes at around 0.01. The reason is that as the Sybil cluster increases in size, the probability that random routes from Sybil nodes cross attack edges decreases. Figure 3(b) shows that when the number of attack edges increases, the average identity trust of Sybil nodes in the cluster increases logarithmically.

The above results suggest that the identity uniqueness scheme is effective; it assigns much lower identity uniqueness to Sybil nodes than to honest nodes. when the number of attack edges is not too high.



(a) (b)

Figure 3: **The effectiveness of identity uniquenesss. a) average identity uniqueness of all Sybil nodes as a function of the number of Sybils using one attack edge. In contrast, the average identity uniqueness of honest nodes nodes is ~ 0.89; (b) average identity uniqueness of all Sybil nodes as a function of the number of attack edges in the Sybil cluster with 1000 Sybil nodes.**

4.2 MapReduce Feasibility Evaluation

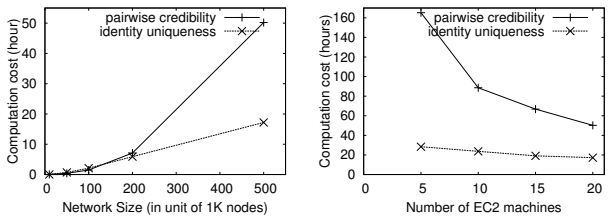
Next, we evaluate whether OSN providers that typically employ clusters of thousands of machines can efficiently analyze large social graphs and provide FaceTrust's credential service. To this end, we implement the algorithms to compute pairwise credibility and identity uniqueness for Sybil-resistance using MapReduce [27], and test the efficiency of the implementation using a 500K user crawled Facebook social graph sample, which we obtain from a previous study [30]. We also sample the crawled Facebook graph using the "forest fire" [36] sampling technique to create social graphs of different sizes to study the implementation's scaling factor. The size of the Facebook sample graph varies in 10K to 500K nodes and the average node degree varies in 8 to 30.

We have not evaluated the SocialRank algorithm because it is based on PageRank whose feasibility has been thoroughly proven in practice.

4.2.1 MapReduce Implementation

MapReduce is a programming framework for performing distributed processing of large data sets using a large number of machines (nodes). We use the EC2/S3 [2] cluster and the Hadoop [9] Java-based MapReduce framework. Our cluster uses at most 20 machines at any time. Below, we provide an overview of our MapReduce implementation.

Our MapReduce pairwise credibility implementation copies the complete credibility-annotated social graph for type k to all nodes in the cluster. The computation is performed over M MapReduce stages. We assign to each Map task a disjoint subset $D \subset V$, where $|D| = |V|/(\#nodes\ in\ cluster)$. At stage M, each Map task performs a single iteration of Dijkstra's algorithm on the trust graph. This iteration involves finding the pairwise credibility between all users in D and all other users in V that use at most M edges. The Map tasks then emits the pairwise credibility values. At stage M, the Reduce task is responsible for creating a compact representation of the pairwise credibility values between all users, creating a new trust graph. The new trust graph is then copied to all nodes in the cluster to be used in the $M+1$ stage. This process is repeated until pairwise credibility values cannot be



(a) (b)

Figure 4: **(a) Computation time to derive pairwise credibility for all pairs of users as a function of the number of users in social graphs of various sizes; (b) Computation time to derive pairwise credibility for all pairs as a function of the number of machines in the EC2/S3 cluster.**

obtained.

The MapReduce identity uniqueness implementation also copies the complete social graph to all nodes in the cluster. It assigns to each Map task a disjoint subset $D \subset V$. For each permutation, the Map task updates the routing tables once and performs random routing for all nodes in D, emiting the pairs ($d \in D$, list of r tails for d) (§ 2.3). Each Reduce task is also assigned a disjoint $D \subset V$ and a list of (verifier $v \in V_v$, r tails of v) pairs. It then performs verification and emits the identity uniqueness of each $d \in D$

4.2.2 Pairwise Credibility Cost

We evaluate the time needed to compute the pairwise credibility between all pairs of users. The complexity of this computation on a credibility-annotated social graph $S(V,E)$ is $O(|V|^2 log|V|)$. The computational cost increases proportionally to the square of the network size. Therefore it is CPU bound, rather than bandwidth or memory bound.

We now evaluate the time needed to compute the pairwise credibility between all pairs of users using the EC2/S3 Hadoop cluster. The assertion and assesment credibility values assigned to the edges are uniformly randomly distributed in $[0.0, 1.0]$.

Using our MapReduce implementation (§ 4.2.1), the cost of computing pairwise credibility values between all pairs in the credibility-annotated social graph $S(V,E)$ is $O(|V|^2 log|V|)$. Therefore the cost increases proportionally to the square of the network size. The computation cost also depends on the number of EC2 machines employed.

In Figure 4(a), we observe that the computational cost grows dramatically as the network size increases. In Figure 4(b), we observe that for $|V| = 500k$ the time needed to complete the parallel computation decreases almost linearly with the number of machines. With a 2000-node cluster, the computation time for a 500K graph is likely to be $\sim$ 30 minutes, shortened by 100-fold. In addition, an OSN only needs to perform the social graph analysis periodically (such as daily) as a background task because the social graph structure is likely to remain largely unchanged in a short period of time. Furthermore, OSN providers can partition multimillion-user social networks into subgraphs of more manageable size (*e.g.*1 million). Such partition could be achieved by exploiting the group structure of OSN fan clubs, interest groups and applications. Finally, the all-pair maximum trust algorithm could be optimized for sparse graphs to further reduce the computation time, a step we are currently pursuing.

Generating r permutations	6473
Random Routing using r routes	197
Verification using 100 verifiers	9.1

Table 2: Average times (msec) for computing the identity uniqueness of a suspect in a 500K node social network with the SybilLimit-like technique.

4.2.3 Identity Uniqueness Cost

We now evaluate the time needed to compute the identity uniqueness of all users in the social graph using the EC2/S3 Hadoop cluster. We aim at demonstrating the practicality and efficiency of identity uniqueness when employed by OSN providers. The identity uniqueness of nodes is computed as a background task at the OSN provider's datacenter. Therefore, it is critical for the scalability of the system that the cost of this computation is not prohibitively expensive.

Table 2 shows the average required time for the routing and verification portion of the identity uniqueness computation for one node in a 500K-node sample of the Facebook social network. We profile the identity trust computation implemented in Java on a Intel Pentium D, 3.40GHz CPU, 2048KB cache and 2GB RAM machine, running Linux 2.6.25-2-68.

The identity uniqueness computation consists of three parts: a) the cost of generating the random permutations at each node; b) the cost of drawing random routes from the suspect and the verifiers (random routing); and c) the cost of determining intersections between the verifiers' and the suspects' tails (verification).

In our MapReduce implementation the computation time is dominated by the random permutation generation, which costs $\Theta(|V| * \sqrt{|E|})$. This computation cannot be parallelized because permutations need to be re-computed at each node. On the other hand, random routing and verification are highly parallelizable. This means that as number of nodes in the cluster increases, the permutation cost becomes even more dominant.

The number of traversed edges during random routing for all nodes is $\Theta(|V| \log(|V|) \sqrt{|E|})$.

The verification compares r tails of a suspect with r tails of l verifiers and its cost in our implementation is $\Theta(l * log(\sqrt{|E|}) * \sqrt{|E|})$. Similar to random routing, the cost of the verification increases $\Theta(\sqrt{R} \times \log(\sqrt{R * |E|}) / \log(\sqrt{|E|})$ times, when the size of the social network increases R times. The cost of creating permutations increases linearly with the number of nodes, since the system generates r permutations for all nodes. The number of permutations r for the social graph $S(V,E)$ should be $\Theta(\sqrt{|E|})$. Consequently, the total cost of creating all permutations is $\Theta(|V| * \sqrt{|E|})$ (§ 2.3).

We profile the MapReduce computation of identity unique-

ness using the EC2/S3 cluster and Hadoop. We use social graphs of size varying in $10K$ to $500K$ nodes. Figure 4(b) shows that the cost of random routing decreases with the number of machines. Using Figure 4(a) we observe that the computation cost increases proportionally to the network size times the square root of the network size.

From the above, we conclude that OSN providers with the capacity to host million of users would be capable of performing the identity uniqueness computation. They would probably need to partition the network into smaller 1 million node social subgraphs and avoid performing this computation frequently.

5. Related Work

Overview: FaceTrust adapts several link analysis algorithms from previous work such as PageRank [22], transitive trust analysis algorithms [45], and Sybil-resistance algorithms [49]. However, our contributions are not the graph algorithms per se. Instead, they lie in the novel idea of using OSNs to provide lightweight, extensible, and relaxed credentials, and the overall design and preliminary evaluation of FaceTrust. A variety of systems have employed trust in social networks to improve system security [26, 38, 40, 41, 43, 44, 48]. To the best of our knowledge, this is the first work that proposes to use OSNs to provide credentials for online personas.

Social Web of Trust: To circumvent the expensive and less scalable identity verification by Certificate Authorities, Zimmerman proposed the PGP Web of Trust [51]. With PGP WoT, users do not rely on a trusted third party to verify their identity, instead they sign each other's public keys, assigning several levels of trust to them. Users perceive the trust levels of the signatures on public-key certificate in two ways [7]: a) as the signer's trust on the validity of the binding between the public-key and the claimed owner of that key; and b) as the signer's trust on the signees ability to assess the validity of other keys.

Users can derive the authenticity of a public key issued by a user with which they have no direct acquaintance by analyzing the corresponding trust graph. Users submit their key signatures to "key servers" and a user can query those servers for the trust paths between her and other users. Typically users analyze the trust graph using tools, such as the gpg command or using non-standardized guidelines such as [8].

FaceTrust's social tagging and credibility metrics are inspired by the PGP WoT. However, PGP's on-demand solution does not scale in the case of very large trust graphs with long trust chains. Instead of requiring users to sign keys, FaceTrustemploys social tagging using the intuitive OSN interface. In addition, the limited number of PGP trust levels (four) does not allow users to express nuances regarding how exactly they trust each other. On the other hand, FaceTrustusers can tag each other with respect to multiple types of trust using a continuous value between 0 and 1.

The concept of the web of trust has also been explored in [32, 35], where the goal is to derive trust values for any two users in the trust graph, which similar to FaceTrustthey describe ways to compute pairwise reputations/trust. Web-of-trust pairwise reputations systems should not be confused with EigenTrust [34] or PageRank [22], which produce global reputation values. Unlike FaceTrust, they do not describe a protocol that enables verifiers to assess the credibility of user assertions they do not defend against Sybil attacks.

Similar to FaceTrust's social rank, NodeRanking [39] uses the same underlying idea as PageRank. The main differences with FaceTrust are: a) we weigh links based on user annotations, while NodeRanking relies solely on the social network topology; b) NodeRanking derives global trust values, whereas pairwise credibility produces pairwise trust values; and c) we further process the ranking of users to derive absolute measures of their credibility (Equation 1).

Social Sybil Attack Defenses: FaceTrust's identity uniqueness is based on SybilGuard and SybilLimit [49, 50]. These systems defend against Sybil attacks by exploiting the fact that OSNs can be used for resource testing, where the test in question is a Sybil attackers ability to create and sustain social acquaintances. FaceTrust's Sybil defense is designed for centrally maintained online social networks (e.g. Facebook). It is conceptually simpler because it leverages information available to the OSN provider (i.e. the complete social graph topology). In addition, FaceTrust does not intend to completely ignore input from not well-connected nodes, but it assigns a low identity uniqueness to it.

A SybilGuard-like technique is used by Leshniewski et al. [37] to limit the Sybil attack in one-hop DHTs. SumUp [44] uses the social network of users to significantly limit the number of fake ratings submitted by Sybil users to no more that the number of edges that connect the Sybil region with the honest region. Similarly, Ostra [38] uses the social network of email senders and receivers to limit the amount of communication that is not flagged as wanted by its receivers. FaceTrust can also be used to build more trustworthy recommendation systems by using credentials that assess a user's ability to recommend/rate correctly.

Authentication via Social Networks: Authentictr [40] is an OSN substrate that provides an API for user authentication and secure communications among users and groups. It exploits a user's access control settings in his OSN account. However, its does not provide primitives for Sybil attack mitigation, and assesment of the credibility of generic user assertions.

Kaleidoscope [41] disseminates the addresses of censorship-preventing proxy relays over a social network, such that each user is aware of only a small subset of the proxies. In this way it ensures that no single censor can block a large number of proxies.

6. Conclusion

Despite the large volume of social interactions taking place

on the Internet, it is still hard to assess the credibility of statements made by online users. This paper presents FaceTrust, a system that leverages online social networks to provide lightweight, flexible, and relaxed credentials that enable users to assess the credibility of others and their assertions. In the FaceTrust design, OSN users explicitly tag transitive trust and credibility scores to their friends and their identity assertions available in their social network profiles. An OSN provider analyzes the annotated social graph to assess the credibility of a user's identity assertions, and issue credentials annotated by credibility scores. Our preliminary evaluation suggests that FaceTrust is feasible and could be effective in obtaining credible and otherwise unavailable identity information for online personas.

7. References

[1] Advance Fee Fraud. http://en.wikipedia.org/wiki/Advance_fee_fraud.

[2] Amazon elastic compute cloud (ec2).

[3] Best friends facebook application. http://www.facebook.com/applications/Best_Friends/6421213170.

[4] Craigslist scams. http://www.craigslist.org/about/scams.

[5] Erroneous VeriSign-Issued Digital Certificates Pose Spoofing Hazard. http://www.microsoft.com/technet/security/bulletin/MS01-017.mspx.

[6] Facebook developers api. http://developers.facebook.com/.

[7] GNU Privacy Handbook. http://www.gnupg.org/gph/en/manual.html#AEN385.

[8] GNU Privacy Handbook. Trust in a Key's Owner. In *http://www.gnupg.org/gph/en/manual.html#AEN346*.

[9] Hadoop. http://wiki.apache.org/hadoop/HadoopMapReduce.

[10] Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.

[11] Nigerian Advance Fee Fraud. www.state.gov/www/regions/africa/naffpub.pdf.

[12] Openid. http://openid.net/.

[13] Swiss keyserver Web of Trust.

[14] VeriSign, Inc. http://www.verisign.com.

[15] Verisign: Personal Digital Certificate Enrollment. https://personalid.verisign.com.au.

[16] Belkin's Amazon Rep Paying For Fake Online Reviews. http://hardware.slashdot.org/article.pl?sid=09%2F01%2F17%2F166226&from=rss, 2009.

[17] Carbonite Stacks the Deck With 5-Star Reviews. http://tech.slashdot.org/article.pl?sid=09/01/27/2349217, 2009.

[18] Fears of Impostors Increase on Facebook. http://www.cnn.com/2009/TECH/02/05/facebook.impostors/index.html?eref=rss_topstories, 2009.

[19] Nigerian Accused in Scheme to Swindle Citibank. www.nytimes.com/2009/02/21/nyregion/21scam.html?_r=3, 2009.

[20] Teen Accused of Sex assaults in Facebook Scam. http://www.msnbc.msn.com/id/29032437/, 2009.

[21] R. Albert, H. Jeong, and A.-L. Barabasi. Internet: Diameter of the World-Wide Web. In *Nature*, 1999.

[22] S. Brin and L. Page. The Anatomy of a Large-scale Hypertextual Web Search Engine. In *Computer Networks and ISDN Systems*, 1998.

[23] J. Camenisch and E. V. Herreweghen. Design and Implementation of the idemix Anonymous Credential System. In *ACM CCS*, 2002.

[24] J. Camenisch and A. Lysyanskay. An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In *EUROCRYPT*, 2001.

[25] A. Cheng and E. Friedman. Manipulability of PageRank under Sybil Strategies. In *NetEcon*, 2006.

[26] G. Danezis and P. Mittal. SybilInfer: Detecting Sybil Nodes using Social Networks. In *NDSS*, 2009.

[27] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In *OSDI*, 2004.

[28] J. R. Douceur. The Sybil Attack. In *IPTPS*, March 2002.

[29] M. Erdos and S. Cantor. Shibboleth Architecture DRAFT v05. *Internet2/MACE, May*, 2, 2002.

[30] M. Gjoka, M. Sirivianos, A. Markopoulou, and X. Yang. Poking Facebook: Characterization of OSN Applications. In *WOSN*, 2008.

[31] Gray, Seigneur, J.-M., Y. Chen, and Jensen. Trust propagation in small worlds. In *LNCS*, pages 239–254. Springer, 2003.

[32] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of Trust and Distrust. In *WWW*, 2004.

[33] J. Kaiser. It's a Small Web After All. In *Science*, 1999.

[34] S. D. Kamvar, M. Schlosser, and H. Garcia-Molina. The EigenTrust Algorithm for Reputation Management in P2P Networks. In *WWW*, 2003.

[35] R. Khare and A. Rifkin. Weaving a Web of trust. In *World Wide Web Journal*, 1997.

[36] J. Leskovec and C. Faloutsos. Sampling from Large Graphs. In *SIGKDD*, 2006.

[37] C. Lesniewski-Laas. A Sybil-proof One-hop DHT. In *Workshop on Social Network Systems*, 2008.

[38] A. Mislove, A. Post, P. Druschel, and K. P. Gummadi. Ostra: Leveraging Social Networks to Thwart Unwanted Traffic. In *NSDI*, 2008.

[39] J. M. Pujol and R. S. J. Delgado. Extracting Reputation in Multi Agent Systems by Means of Social Network Topology. In *International Conference on Autonomous agents and Multiagent Systems*, 2002.

[40] A. Ramachandran and N. Feamster. Authenticated Out-of-Band Communication Over Social Links. In *WOSN*, 2008.

[41] Y. Sovran, A. Libonati, and J. L. Pass it on: Social Networks Stymie Censors. In *IPTPS*, 2008.

[42] J. Steiner, C. Neuman, and J. Schiller. Kerberos: An Authentication Service for Open Network Systems. In *USENIX Winter Conference*, 1988.

[43] A. Tootoonchian, K. K. Gollu, S. Saroiu, Y. Ganjali, and A. Wolman. Lockr: Social Access Control for Web 2.0. In *WOSN*, 2008.

[44] D. N. Tran, B. Min, J. Li, and L. Subramanian. Sybil-Resilient Online Content Rating. In *NSDI*, 2009.

[45] K. Walsh and E. G. Sirer. Experience with an Object Reputation System for Peer-to-Peer Filesharing. In *NSDI*, 2006.

[46] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. In *Nature*, 1998.

[47] Facebook. http://www.facebook.com.

[48] S. Yardi, N. Feamster, and A. Bruckman. Photo-Based Authentication Using Social Networks. In *WOSN*, 2008.

[49] H. Yu, P. Gibbons, M. Kaminsky, and F. Xiao. A near-optimal social network defense against sybil attacks. In *IEEE S&P*, 2008.

[50] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. SybilGuard: Defending Against Sybil Attacks via Social Networks. In *ACM SIGCOMM*, 2006.

[51] P. Zimmerman. PGP Users Guide. December 1992.