Towards Social Profile Based Overlays

David Isaac Wolinsky, Pierre St. Juste, P. Oscar Boykin, Renato Figueiredo
University of Florida

Abstract

Social networking has quickly become one of the most common activities of Internet users. As social networks evolve, they request more information from the users and thus requiring the users to place more trust into the social network. Peer-to-peer (P2P) overlays can return ownership of information and system control to the user as they can be constructed in a way to not require third party proxies.

In this paper, we present a novel concept known as the structured social overlay that applies social networks to structured P2P overlays to provide ownership, scalability, reliability, and security. Each user's profile is assigned a unique private, secure overlay, where members of that overlay have a friendship with the overlay owner. The profile data is stored using the profile overlay's distributed data stores. To ensure privacy, the profile overlay employs the public key infrastructure, where the role of certificate authority (CA) is handled by the overlay owner and each member of the overlay has a CA signed certificate. Each member of the social network, joins a common public overlay, which provides services to discover friends and bootstrap connections into existing private overlays through a distributed data store. We define interfaces that can be used to implement this system as well as explore some of the challenges related to it.

1 Introduction

Social networking has become pervasive in daily life, though as social networks grow so does the wealth of personal information that they store. Users become more dependent on social networks as users surrender tasks such as communication and identity to the social network. Once information has been released to a social network, known as a user's profile, the user is at the mercy of the social network. If the social network engages in activites disagreeable to the user, such as recent issues with Facebook's Beacon program [9], the user has the option to leave the social network surrendering their identity and features provided by the social network, to accept the disagreeable activities, or to petition and hope that the social network changes its behavior.

Recent work [2] presents the coupling between social networks and P2P systems. Noting that a social network made up of social links is inherently a P2P system with the aside that they are currently developed on top of centralized systems. In this paper, we focus on the topic of topology, how the P2P system should be organized, by applying the social network to structured P2P overlays. We then explain how the use of this topology enables reliability, security, and robustness.

Structured P2P overlays provide a scalable, resilient, and self-managing platform for distributed applications. Structured overlays provide means by which users can easily create their own decentralized systems for the purpose of data sharing, interactive activities, and other networking enabled activities. In recent work [15], we have implemented mechanisms that allow users to create and manage their own private overlays using a common public overlay to assist in discovery and NAT traversing of the private overlay. In this paper, we further this work by an indepth discussion on how to apply this technique to social networks.

Social networks consist of users, each of whom typically has a a profile, a set of friends, and a private message inbox. The profile contains the users personal information, status updates, and public conversation with the friends. Friends are individuals which the user trust sufficiently to view the profile. The private message inbox allows users to send messages between each other without leaking any information to other friends. Using this model, we describe a common directory or public overlay which allows peers to provide services where peers can find friends and join overlays where there already exists an established relationship. Each user has their own profile overlay, where the members of the overlay are limited to the current friends of the profile owner. The profile overlay is secured by a public key infrastructure (PKI) with the profile owner being the certificate authority (CA). The profile information is stored information is stored in distributed datastores, allowing profile information to be accessed in scalable mechanisms regardless of the profile owner's online state.

Our proposed social network makes use of two types of overlays, a directory overlay and a profile overlay. In this paper, we explain how these overlays provide key features of social networks: finding and befriending peers, sending public and private messages, and sharing media.

The rest of this paper is organized as follows. Section 2 provides background and related work. Section 3 describes our multioverlay approach, explaining how to map social networks onto structured P2P overlays. In Section **??**, we present constraints that our proposed system introduces and provide potential solutions. We conclude the paper in Section 4.

2 Background

In this section, we review structured P2P overlays and distributed and decentralized social network techniques.

2.1 Structured P2P Overlays

Structured P2P systems provide distributed lookup services with guaranteed search time with a lower bound of $O(\log N)$, in contrast to unstructured systems, which rely on global knowledge/broadcasts, or stochastic techniques such as random walks [3]. Some examples of structured systems can be found in [13, 14, 7, 8, 11]. In general, structured systems are able to make these guarantees by self-organizing a structured topology, such as a 2D ring or a hypercube.

In the overlay, each node is given a unique node ID drawn from a large address space. Each node id must be unique otherwise address collisions will occur, which can prevent nodes from participating in the overlay. Furthermore, having the node IDs well distributed assist in providing better scalability as many shortcut selection algorithms depend on having node IDs uniformly distributed across the entire address space. A simple mechanism to ensure this behavior is to have each node use a cryptographically strong random number generator to generate the node ID. Another mechanism involves the use of a trusted third party to generate node IDs and cryptographically sign them [4].

As with all P2P systems, in order for an incoming node to connect with the overlay, the node must know of at least one active participant. A list of nodes that are running on public addresses is typically distributed with the application, available through some out-of-band mechanism, or possibly using multicast to findpools [13].

In dealing with ring based overlays, a node must be connected to closest neighbors in the node ID address space; optimizations for fault tolerance suggest that it should be between 2 to $\log(N)$ on both sides. Having multiple peers on both sides assist in stabilizing the overlay structure when experiencing churn, particularly when peers leave without warning.

Overlay shortcuts enable efficient routing in ring-structured P2P systems. Different shortcut selection methods include: maintaining large tables without using connections and only verifying usability when routing messages [13, 8], maintaining a connection with a peer every set distance in the P2P address space [14], or using locations drawn from a harmonic distribution in the node address space [7].

Most structured P2P overlays support decentralized storage/lookup of information by mapping keys to specific node IDs in an overlay. At a minimum, the data is stored at the node ID either smaller or larger to the data's node ID and for fault tolerance the data can be stored at other nodes. This sort of mapping and data storage is called a distributed hash table (DHT).

In [5, 10, 12], the authors discuss the concept a single overlay supporting services by additional overlays that use the underlying overlay to assist in discovery. In [15], we describe a reference implementation of a multiple overlay system that supports the use of a public overlay's DHT to store currently active peers in the private overlays. Whereby users could create their own overlays without having to create their own bootstrap network. In addition, our system provides both relay and hole-punching NAT traversal techniques and supports point-to-point PKI based security.

2.2 Social Networks

3 Social Overlays

In this section we introduce the components of our multi-overlay system, the public directory overlay and the private profile overlays. A directory overlay supports two features: 1) a directory for friend discovery and verification and 2) lists of peers currently active in each profile overlay. A profile overlay supports the features of a profile, private messages, and media sharing. In this section, we explain how to map social networking features to this multioverlay approach. First we explain how peers find each other, then there interaction in the private overlay, and finally how they connect to the private overlay.

3.1 Finding and Verifying Friends

In a traditional social network, a directory consists of many directory entries consisting of peer's public information, such as the user's name, user name, e-mail address, group affiliations, and friends. A directory can be searched using this information to find one or more matching directory entries. The user then makes a friendship request, which notifies the remote peer of this request. The request receiver can review the public information of the requestor prior to making a decision. If the receiver accepts the request, the peers are given access to each other's profiles. Whereupon, they can learn more information and if it turns out to be a mistake, the peers can unilaterally end the relationship.

To map this to our proposed social overlay, the directory entries can be inserted into the DHT. As discussed in previous work, the keys where the directory entries are stored at consist of a subset of the user's public information in lower-case format and hashed to an overlay address. The value stored at these keys is the user's certificate, which consists of its public information and an overlay address where the user expects to receive notifications. The overlay address can be used for asynchronous offline messaging, whose function we will explain shortly.

To bind public information to a certificate, we use a certificate of the format presented in Figure **??**. The

main portion of the certificate is similar to a self-signed x509 [6] certificate with public information such as user's name, user name, e-mail address, and group affiliations embeded into the certificate. A friend list is represented by many friend entries, for this we employ a technique similar to PGP, user's can acquire from their friends a signed message consisting of a hash of the peers certificate, the time stamp, and the friend's certificate hash signed by the friend. Since PGP does not provide a strong method for revocation the time stamp provides a slightly better means to decipher whether or not a friendship link is still active without accessing the public overlay of either peers. Peers should request new friend list entry within a certain period of time or it will appear that the friendship is no longer valid.

While looking for an individual, a peer may discover that many individuals have overlapping public information components, such as the user's name. Assuming all entries are legitimate, the overlay must support inserting multiple values at the same key, leaving the peer or the peer's DHT client to parse the responses and determining the best match by reviewing the contents of each certificate. Alternatively a technique like Sword [1] support distributing the data across a set of nodes and using a bounded broadcast to discover peers that match all information used for searching.

Upon discovering an individual with whom a peer would like a friendship, the peer will issue a friendship request. As stated earlier, the data stored in the directory has an overlay address, where a peer expects friendship requests to be inserted into the DHT. The friendship request consists of the self-signed certificate of the requesting peer, the public information of the request receiver, a time stamp, and a signature made from the private key associated with the self-signed certificate. Though because DHTs are soft state systems having leases, the requester must reinsert the request upon timeout and no response for the receiver.

Once a request has been inserted into the DHT, the receiver can come online and check for outstanding requests. If the receiver would like to add the user, he may do so conditionally or unconditially. An unconditional accept would cause the user to issue a request himself and also sign the request of the originating requester. Alternatively in the case of a conditional accept, the user would issue a request, wait for a reply, and investigate prior to signing the originating requesters request. Once a user has received signed certificate, they may access the remote peers profile overlay as discussed in 3.2, which is also responsible for activities such as revocation.

Since a DHT is a soft-state system that uses leases to remove expired data, requests and responses must be occassionally reinserted into the DHT. Alternatively approaches mentioned in [], that suggest methods of storing data in overlays using quotas could be used to ensure fair usage of the overlay.

3.2 The Profile Overlay

In a traditional social network, the profile or user-centric portion consists of a public message board for status updates or public messages, private messaging, data sharing, and maintenance of existing friendships. In this section, we explain how these components can be applied to a structured overlay dedicated to an individual profile.

The profile overlay consists of all the online friends of the profiler owner. For access control, we employ a PKI, where each member uses the signed certificate generated during the "finding and verifying friends" stage. All links are encrypted using symmetric security algorithms established through the PKI. Thus preventing uninvited guests from gaining direct access to the overlay. Because the profile owner also is the CA for all members of the overlay, they can easily revoke users from the profile overlay with ease similar to a centralized social network. In [**?**], we presented a broadcast mechanism for immediately revoking peers from a system and we also suggest using a DHT based revocation list for indirect revocation.

The message board of a profile can be stored in two ways, distributed via a data store as well as stored on the profile owners personal computing devices. While the distributed data store would provide the profile when the profile owner is offline, it also distributes the load for popular profiles. Since reliability in a profile may be suspect, each peer should always be a provider for all data in their profile, when they are online.

Similarly, private messages could be stored in this distributed data store, though since they are private, the message should be prepended by symmetric key used to encrypt the rest of the message. the symmetric key should be encrypted with the profile owner's public key, so that they are the only party that can read the message. Additionally, the body of the message encrypted should contain all relevant material such as the sender, time sent, subject, and the message itself. If the messages are stored in a well known location, the profile owner can either poll the location or, alternatively, use an event based system to notify them of new messages.

An ideal distributed data store would support database like access allowing to be stored chronologically and by type, i.e., media, status update, or public messages. The distributed data store should be built on top of the profile overlay so that only members of the profile store the personal data of the user. Since the material is public for the group and the groups links are all encrypted, the data could be distributed without encryption though all data should be signed so that each post has an identity attached to it. Posts that lack this identity should be ignored when viewing the profile. Only the profile owner and owners of posts made on the message board should

have the ability to delete the material.

3.3 Active Peers

The directory overlay should be used to assist in finding currently active peers in the profile overlays. By placing their node IDs at a well known though unique per profile overlay key in the DHT, active peers can bootstrap incoming peers into the profile overlay. We implemented and evaluated this concept in [15]. Because the profile overlay members all use PKI to ensure membership, even if malicious peers insert their ID into the active If malicious peers were to insert their ID into this list, it would be useless as the peer would only form connections with peers who also have a signed certificate for that profile.

4 Conclusion

References

[1] J. Albrecht, D. Oppenheimer, A. Vahdat, and D. A. Patterson. Design and implementation trade-offs for wide-area resource discovery. In *ACM Trans. Internet Technol.*, 2008.

[2] S. Buchegger and A. Datta. A case for P2P infrastructure for social networks - opportunities & challenges. In *WONS '09: The Sixth International Conference on Wireless On-demand Network Systems and Services*, 2009.

[3] M. Castro, M. Costa, and A. Rowstron. Debunking some myths about structured and unstructured overlays. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, 2005.

[4] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Security for structured peer-to-peer overlay networks. In *5th Symposium on Operating Systems Design and Implementaion (OSDI'02)*, December 2002.

[5] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. One ring to rule them all: Service discover and binding in structured peer-to-peer overlay networks. In *SIGOPS European Workshop*, Sept. 2002.

[6] R. Housley, W. Polk, , W. Ford, and D. Solo. *RFC 5280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, May 2008.

[7] G. S. Manku, M. Bawa, and P. Raghavan. Symphony: distributed hashing in a small world. In *USITS*, 2003.

[8] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the XOR metric. In *IPTPS '02*, 2002.

[9] J. C. Perez. Facebook's beacon more intrusve than previously thought. `http://www.pcworld.com/article/140182/facebooks_beacon_more_intrusive_than_previously_thought.html`, 2007.

[10] Randpeer development team. Randpeer. `http://www.randpeer.com`, May 2007.

[11] S. Ratnasamy, P. Francis, S. Shenker, and M. Handley. A scalable content-addressable network. In *In Proceedings of ACM SIGCOMM*, 2001.

[12] S. Ratnasamy, M. Handley, R. M. Karp, and S. Shenker. Application-level multicast using content-addressable networks. In *NGC '01: Proceedings of the Third International COST264 Workshop on Networked Group Communication*, 2001.

[13] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, November 2001.

[14] I. Stoica and et al. Chord: A scalable Peer-To-Peer lookup service for internet applications. In *SIGCOMM*, 2001.

[15] D. I. Wolinsky, K. Lee, T. W. Choi, P. O. Boykin, and R. Figueiredo. Virtual private overlays: Secure group communication in NAT-constrained environments. In *TR-ACIS-09-004*, December 2009.