Digital Object Identifier: 10.1109/MIC.2013.131

URL: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6685803

Privacy Preservation in Decentralized Online Social Networks

Lorenz Schwittmann, Matthäus Wander,
Christopher Boelmann, Torben Weis
University of Duisburg-Essen
Bismarkstr. 90, 47057 Duisburg, Germany
wimi@vs.uni-due.de

December 9, 2013

Keywords— Online Social Network, Privacy, Social Graph Obfuscation, Survey

Abstract

The desire for personal data of large online social networks arouses concerns from privacy advocates and leery users. Researchers have proposed decentralized architectures to create online social networks with technically imposed privacy preservation. This article surveys several approaches for their privacy benefit and discusses their architecture and suitability for mobile devices.

1 Introduction

Most commonly used online social networks (OSNs) have a centralized architecture. Centralized OSNs are characterized by being developed and deployed by only one service provider. With over 1 billion active users Facebook is the largest of these networks.

The centralized architecture and data aggregation improve the usability. Since one entity (i.e., the OSN service provider) maintains all user data, searching for users is trivial. Furthermore, updates and extensions to the OSN functionality and changes to the underlying architecture are eased since the whole OSN is managed by one entity.

From the user's perspective, however, there are also disadvantages arising from user data aggregation at a single entity. The apparent one is the access to private data by the service provider and any third-party the provider has granted access to. Another reason is the disclosure of the *social graph* which represents the interconnections of OSN users, their behavior within the OSN and their preferences. The social graph itself is private information and allows even the identification of users who did not sign up with their real name [XZL08].

> “You are who your friends are.” —Anderson and Stajano [AS13]

Researchers have investigated technical solutions for solving the privacy concerns that arose from centralized OSN. In this article we survey state of the art decentralized OSN systems and compare their characteristics.

Two such systems are Diaspora and OneSocialWeb which both use a federated architecture of independent servers. Another class of systems uses end-to-end client-side data encryption to hide the data content from the storage providers. This includes Persona, Vegas and SoNet. The third class of systems – PeerSoN, Safebook, LifeSocial and Cachet – utilize a distributed hash table (DHT) instead of federated servers.

The fundamental characteristic which we compare is the privacy benefit of the decentralized approach. Apart from personal content this includes more subtle data and metadata, e.g. the time at which the user was active or the social graph the user is interacting with. A closely related question to that of privacy benefit is the cost that it incurs. While some users may be willing to accept performance penalties to a certain degree, a privacy preserving system has to compete with its centralized forerunners in terms of efficiency and functionality. In March 2013, Facebook registered 751 million active mobile users. These users will unlikely give up mobile access if they switch to a privacy enhanced OSN. Therefore, a succeeding OSN needs to run efficiently, including on mobile devices with limited hardware resources.

2 Server Federations

A server federation is a decentralized system of loosely connected servers, each being run by an independent data-holding entity (Figure 1). This should prevent accumulation of large amounts of personal data in one place.

One famous representative of this class of OSNs is Diaspora [Dia13] which received a large crowdfunding and has been featured by the New York Times.[1] With about 380,000 users,[2] it is the largest decentralized OSN. Diaspora consists of independent servers (called *pods*) which communicate with each other. Users can either register on an existing pod or create their own. In the later case, the user stays in control of his data because the pod runs on a machine administrated by himself. Diaspora

[1]http://www.nytimes.com/2010/05/12/nyregion/12about.html
[2]https://diasp.eu/stats.html

has a fine grained access control scheme to share content with specific contacts or contact groups. The server-to-server protocol specifies transfer of encrypted and signed messages providing security comparable to SSL/TLS. End-to-end encryption does not take place.

This is similar to OneSocialWeb, which is another approach for decentralized online social networking.[3] It aims to enrich the Extensible Messaging and Presence Protocol (XMPP) with OSN features. XMPP itself is an instant messaging protocol with a federated architecture. OneSocialWeb leverages the existing XMPP messaging infrastructure with enhancements for creating profiles, expressing social relationships and sharing content.

Although Diaspora and OneSocialWeb allow users to set up their own servers, personal data will not always stay there. If friends from other servers request this information, it will be transfered to their servers and become accessible in plaintext to the service provider. Consider the current deployment status of Diaspora: the largest server hosts 70% of all users [BHG$^+$]. One service provider has thus access to the data of 70% of Diaspora users plus their friends' data stored on other servers. Therefore, a theoretical decentralized architecture itself is not enough to prevent aggregation of user data in one place. This shows that a user has to trust his service provider as well as the friends' service providers. Furthermore, one service provider can see every interaction of its users since every user is uniquely identifiable (i.e., *username@hostname*). This enables the service provider to determine the social graph of its hosted users.

This shows that merely distribution of user data on different servers combined with server-side enforced access control is still prone to attacks on users' privacy. The users' privacy depends on providers not disclosing their data without authorization. In the next section we discuss decentralized OSNs that use encryption to hide content from service providers.

3 Encrypted Data Storages

To cope with the issue of curious or malicious service providers, researchers suggested the use of encryption on the end hosts. With encryption and decryption taking place at the edges of the network, service providers become ordinary storage providers for opaque data blobs. Several challenges arise from using cryptographic user-to-user security in an OSN. First of all there is the well-known key exchange problem: How can users exchange cryptographic keys over an untrusted server without initial knowledge of authentic data?

This can be solved by an out-of-band method, e.g. face-to-face meeting. While this is generally cumbersome, methods like Bluetooth, NFC or QR-codes can make it more

[3]http://onesocialweb.org/

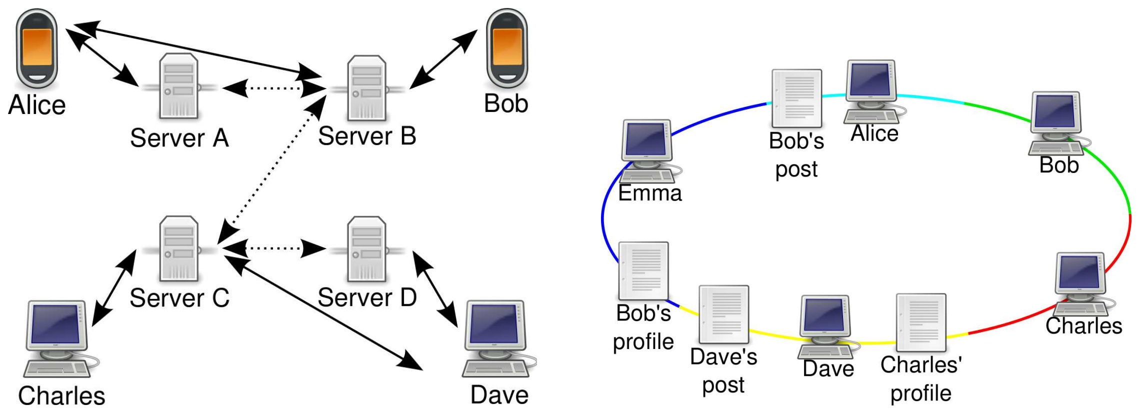

Figure 1: Examples of OSN architectures: A federated system with interconnected servers (left) and a DHT-based structured peer-to-peer system (right). Indicated by color, files are associated with their responsible peer.

comfortable, e.g. as used in Vegas [DMD]. Inviting new friends to a privacy enhanced OSN by mail is discussed in [DWM10]. An in-band authentication approach is the Socialist Millionares Protocol which prevents a malicious OSN provider from interfering with exchanged messages by using a shared secret. This shared secret can be a common information in natural language, e.g. location where two acquaintances met first [SBWW13].

As it is not possible to crawl a friend's list for new contacts, Vegas specifies a coupling mechanism. A user can initiate friendship authentication between two of his trusted friends and to guarantee for their correct identity. During this coupling procedure new keys are exchanged so the initiating user cannot read future messages exchanged between the coupled friends.

3.1 Encryption Schemes

In an unencrypted OSN it is the responsibility of the server to enforce access control lists. With user-to-user encryption, the end host becomes responsible to enforce access control via cryptography.

Traditionally, a hybrid encryption scheme is used. Every user has a public/private key pair. Messages are encrypted with a symmetric key, which is in turn encrypted with the public key of each recipient. The idea is to reduce the runtime of the computationally expensive asymmetric encryption and to be able to reuse the symmetric key for future messages. In particular, the runtime of the symmetric encryption does not depend on the number of recipients. To revoke a user's access to a group key, a new symmetric key must be rolled out to the remaining group members. This requires again to perform asymmetric cryptographic operations, but removing a user from a group is expected to happen less often than publishing new content.

Persona [BBS$^+$] uses a different encryption scheme called Attribute Based Encryption (ABE). With ABE, the symmetric content key is encrypted to be shared with predefined groups or combinations of groups ("co-worker $\wedge$ friend"). This allows to encrypt a new group key with one ABE operation for intersections of existing contact groups. This is more efficient than the traditional hybrid approach which would encrypt the group key n times for each of the n recipients. While saving space, it comes with a higher computational cost: ABE operations are about 100-1000 times slower than those of RSA.

A similar encryption scheme is Identity Based Broadcast Encryption (IBBE) [RMJM]. IBBE is more flexible than ABE as it addresses individual recipients instead of predefined groups. It also supports removing recipients from groups with no further cost. This differs from traditional hybrid approaches and ABE which must re-run the generation and roll-out new keys.

3.2 Metadata

At a glance, the idea of encrypted content seems to obsolete the necessity for traditional access control lists. However, world readable storages of encrypted data may still leak metadata like timestamps, size of data objects, data structures or header information. Data lengths can be used to identify object types (texts, images, videos, likes) and header information leak group communication patterns and allow social graph inference [GKB12]. A newly appearing data object implies OSN activity at a certain time, which can be of interest to employers for example. To prevent a third-party attacker from harvesting such metadata, server access control can be used in addition to end-to-end encryption.

Persona takes a different approach and does not offer public lists of data objects. Any user that can name a data object may retrieve it, but references to other data objects are encrypted together with the content. This hides metadata from public but still allows server providers to observe interactions in the social graph. Friends are granted write access by signing a token with their public key, which is a unique identifier. With the IBBE encryption scheme, each message contains identifiers of the recipients, including those who do not write back. This leads to the question whether the social graph can be hidden even from storage server providers.

3.3 Hiding the Social Graph

An approach to hide the social graph from server providers is aliasing in the SoNet OSN [SBWW13]. Users communicate exclusively with their storage service which is also used as proxy for accesses to other users' storages. Upon friendship establishment between Alice@ServerA and Bob@ServerB both sides generate random identifiers, i.e., ServerB sees Alice as j@ServerA and ServerA sees Bob as i@ServerB. These identifiers are unique per friendship, so if Charles@ServerB becomes friends with Alice, ServerB sees Alice as k@ServerA ($i \neq j \neq k \neq i$). Each storage server knows all aliases of their users and can therefore forward messages to the correct recipient. A storage cannot resolve aliases of other storages to cleartext usernames. Due to the uniqueness of aliases, the storage does not see if two of its users are friends with the same user on another storage.

Aliasing attempts to disguise user identities in an aggregated user base on a storage server but fails with imbalanced user distributions. Consider e.g. a storage server with 70% of all users on which all relations among each other are known to the provider, or communicating with a storage which has only one user. Colluding storage providers can as well unveil relations between their users. An ideal setup would be a homogeneous distribution of users on independent servers but this can not be enforced by technical measures.

The OSN Vegas takes a more rigorous approach to hide the social graph [DMD]. Viewing a friend's contact list is deliberately not a supported feature. Vegas uses public storage servers but with random file names and hidden directory structures. One user can run multiple storage servers and point each contact to a separate server. For each contact, different cryptographic keys are used to avoid identification by public keys. This hides the social graph from server providers and from the friends but leads to limitations which are discussed in the following section.

3.4 Performance and Limitations

Encryption and signing on end user devices necessarily imposes technical constraints. If storage providers are not able to read user data, they cannot easily assist the user at processing this data. This includes e.g. searching, creating image thumbnails, or computing summaries over large data sets. In a worst case scenario clients have to download all data, decrypt it, and process it locally. While there is research on homomorphic encryption allowing server-side operations on encrypted data it is currently limited to rudimentary operations and not practical for real applications. Nevertheless, the performance of basic OSN functionality does not need to be considerably inferior to OSNs without end-to-end encryption, even on mobile devices. In SoNet, the time overhead for the cryptographic operations of a typical group message was 28 ms on an average Android device[SBWW13]. While this shows feasibility for symmetric ciphers and classic asymmetric ciphers like RSA, more computational complex schemes like IBBE are not suitable for mobile devices.

All of the discussed encrypted OSN approaches prevent leakage of metadata to third-parties. SoNet attempts to obfuscate the social graph from server providers by communicating with aliases. The aliasing has a low network and CPU overhead but hides the graph only partially. Vegas effectively hides the social graph from server providers but does not support OSN features like discussion groups or comments readable by other contacts. Avoiding those communication types which reveal social relations consequently yields 1:1 messaging. Sending one message to n recipients is possible but expensive because it requires $2n$ asymmetric cryptographic operations. Posting a status message to 100 friends would take more than 2 seconds on a mobile device.

4 Peer-to-Peer Approaches

Peer-to-peer based OSNs have been proposed that differ from federated approaches by using shared resources of connected clients for storing user data that is encrypted and can only be accessed by users that have the necessary keys, e.g. friends. The idea is to waive dedicated third party servers and to have a system which scales automatically with the number of participating users. The common element is a distributed hash table (DHT) which is used to find data in a peer-to-peer system with logarithmic routing complexity.

Distributed hash tables assign every participating peer and every data object that has to be stored an ID within the DHT key space. Every peer is responsible for the data stored in a part of the key space (e.g. closest IDs), as sketched in Figure 1. The allocated hard disk space can thus be much higher than the data a user has actually stored in the DHT himself, which may lead to a perception of unfairness. As peers sign on and off, the key space responsibilities of peers have to be adjusted, which requires a constant exchange between peers. This poses an overhead compared to server federations, especially when the DHT is used as payload storage and not just as a data index.

This approach is used in LifeSocial [GGS$^+$] which stores all user content in a Pastry DHT. Data objects can contain actual payload data or references to other data objects. References represent a distributed linked list and allow efficient storing, e.g. a photo object being linked from different albums and from the user profile. However, this has implications for data retrieval. Each reference has to be resolved by a DHT query which will usually be routed to different peers. Since the assignment of data objects to peers is seemingly random, it does not take authorship or network proximity into account. Even if resolution of references is parallelized, the total time to fetch a subtree is still linear in its depth. While the authors evaluated performance for a small network of 30 nodes, practicability for a large network consisting of millions of peers remains unclear.

As this may be a major obstacle for practical deployment, Cachet [NJM$^+$12] attempts to improve lookup times by caching data at friends. In addition to maintaining a DHT network, Cachet establishes direct connections to friends to push data objects to them.

Friend connections are secured with SSL/TLS which is faster than decrypting many small objects individually. If a user is offline, common trusted friends can be used to retrieve content without costly DHT lookups. According to the Cachet authors, the time required to display an OSN newsfeed decreased from several hundred to around 10 seconds in their software prototype. While this is a significant improvement, it is still a perceptible delay compared to server-based systems.

In the related PeerSoN [BSVD] approach content is always transfered directly between peers. The DHT serves as index to track the list of peers which hold a copy of each object. Each peer downloading data from a friend can enter its address into the DHT and share the cached data. This keeps the DHT thin, similar to the proven trackerless BitTorrent DHT, but what does this mean for privacy?

4.1 Privacy and Security Implications

In PeerSoN, the online status of all users and their IP addresses are world-readable from the DHT. The social graph can be easily derived from peers which cache data for their friends. In LifeSocial, social relationships are not reflected in the network topology because data-to-peer allocations are random. Nevertheless, each user is identified uniquely by his public key. Each data object in the DHT contains the cleartext IDs of the recipients, allowing recovery of the social graph as well.

Cachet makes use of object references to hide the recipients of data objects. The public keys are not saved with the data objects but at its parent objects in encrypted form. Given a well-known object as entry point, only authorized friends see the recipients of the referenced objects. This yields a privacy level comparable to federated, world-readable servers.

The main privacy difference between a federated and a DHT-based online social network is the distribution of responsibility and power. In a server federation a user chooses a provider and trusts him to not delete his encrypted data or attempt de-anonymization by analyzing network traffic and metadata. This provider remains fixed until the user decides to move to another provider. In a DHT the responsibility and power is dispersed to a varying set of peers, assuming that the majority is trustworthy or at least not interested in decloaking the users identity. Due to the openness of the system, a DHT provides weak spots to increase the power that one peer should have. Examples include deleting or modifying data, denying access to data (DoS), choosing a peer ID to control a specific part of the DHT key space or placing oneself in DHT routing tables to control large parts of network traffic [UPS11].

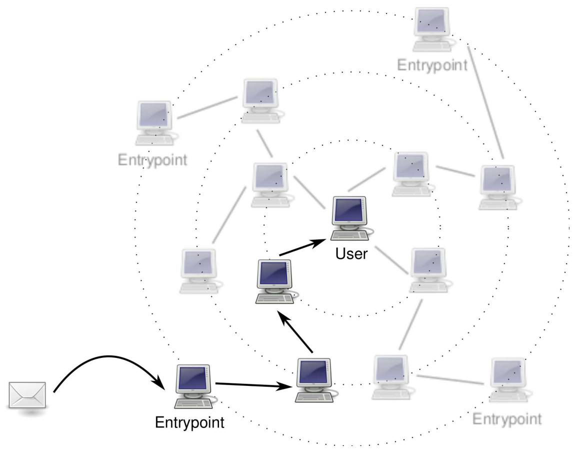

Figure 2: Message routing through matryoshkas circles. Lines between circles represent a friend relation.

4.2 Friend-to-Friend Networks

The use of existing real-life trust has been suggested in Safebook [CMO] as foundation for an online social network. Each user connects directly to trusted friends to store data and forward messages. This constitutes a friend-to-friend network with concentric circles around each user called *matryoshkas*. That way it is possible to route from the outermost to the innermost circle by only traversing strong trusted links as seen in Figure 2. The outermost peer signs into a Kademlia DHT as entry point for a user's matryoshka network. These paths are created randomly during bootstrapping. Using this scheme, it is possible to communicate with a peer without knowing his IP address or disclosing his identity. The peers on the innermost circle, which are connected directly to the user, are used to mirror his profile data. These mirrors also store comments from other users and present them to the user as soon as he is online.

The matryoshka network reflects the social graph but as social links are trusted connections, outsiders cannot disclose the graph. Like in other peer-to-peer-based systems, it is again the DHT that offers weak spots: adversaries can sign into the DHT as entry point for any existing user or attempt to control the DHT routing or key space. The principle to replicate data only to friends and not into the DHT puts constraints on the usability. Most friends can be expected to share a similar circadian rhythm and turn off their devices when they go to bed. Friends staying in other parts of the world will need to share online time to send messages and vacation pictures. While this can be alleviated by keeping some devices online around the clock, it questions the suitability of a pure peer-to-peer network when it is in fact a network of servers that keeps the system running.

4.3 Mobile Devices in P2P Systems

The effort to maintain a DHT or another type of peer-to-peer network causes a continuous basic system load. While this may be negligible for desktop devices, it renders peer-to-peer technology unsuitable for mobile devices whose battery will drain and data volume might be exceeded. This is an additional load besides the previously discussed cryptography on mobile devices. Mobile devices do not necessarily need to become part of the peer-to-peer network to access the DHT: desktop peers can act as gateways. This saves resources on mobile devices and increases the network quality since only the more reliable desktop peers maintain the DHT. On the other hand, it may lead to an unbalanced distribution of resource consumers versus contributors. As the number of mobile devices increases, the scalability advantage of peer-to-peer systems diminishes.

5 Conclusion

There are different understandings about the objectives of privacy preservation in online social networks. An overview of the features of the discussed systems is given in Table 1. Most authors agree on the need to encrypt content before passing it into the storage service. OSN system designs are divergent about hiding metadata like online status, IP address, recipients, message size and length. We conclude that peer-to-peer-based systems, while evading the need for dedicated storage servers, are especially prone to leak metadata. Compared with encrypted data storages this can be seen as a cost/privacy tradeoff. Together with functional constraints with regard to the growing mobile user population, federated approaches are a better fit for future privacy enhanced OSNs.

Server federations without encryption are the only class of OSNs which allows some kind of server side data utility. While these are a privacy threat this approach could provide a advertisement based business model for free OSN hosting. Privacy is only provided by choosing a trustworthy OSN provider.

Besides the challenges every new OSN has to face (e.g. convincing users of established OSNs to switch, although there are none/few of their friends), a privacy enhanced OSN has additional obstacles: Higher loading times (Cachet), increased resource requirements due to advanced encryption schemes, key exchange related actions (SoNet, Vegas), unsuitability for mobile devices and a reduced feature set (Vegas) are examples of downsides observed in this article.

Furthermore, systems without end-to-end encryption have to be discarded due to missing confidentiality. This confidentiality can be achieved by using hybrid encryption schemes which have been shown to be feasible for mobile devices. While the leakage of metadata to third parties can be prevented by read access control or disallowing directory listing, a partly unsolved problem persists in obfuscating the social graph in

	Diaspora	OneSocialWeb	Persona	Vegas	SoNet	PeerSoN	Safebook	LifeSocial	Cachet
End-to-end encryption	-	-	+	+	+	+	+	+	+
DHT-based	-	-	-	-	-	+	+	+	+
Federated Servers	+	+	+	+	+	-	-	-	-
Graph hidden	-	-	-	+	(+)	-	+	-	-
Replication	-	-	-	-	+	+	+	+	+
Mobile support	+	+	+	+	+	+	-	-	-
Hide activity from 3rd party	+	+	-	+	-	-	-	-	-

Table 1: Overview of characteristics.

front of the storage provider. Vegas proposes a "one storage per friend" solution which solves this problem but it results in poor multicast performance. On the other hand, SoNet is efficient but only hides the graph partially and under certain preconditions in terms of user distribution. A solution realizing both, generally hiding the social graph while being efficient, is yet to be found.

The authors

Lorenz Schwittmann is a PhD candidate at the University of Duisburg-Essen. His thesis focuses on privacy in distributed federated storages. He has an MSc in computer science from the University of Duisburg-Essen. Contact him at lorenz.schwittmann@uni-due.de.

Matthäus Wander is a PhD candidate at the University of Duisburg-Essen working on DNSSEC and related Internet security topics. Email: matthaeus.wander@uni-due.de

Christopher Boelmann is a PhD candidate in the Distributed Systems research group at the University of Duisburg-Essen. His research interests include self-stabilization in distributed systems, sensor-networks and distributed computing. Contact him at christopher.boelmann@uni-due.de.

Torben Weis is a professor at the University Duisburg-Essen, where he leads the Distributed Systems research group. His research focuses on architectures and protocols for distributed applications, peer-to-peer systems, and self-organizing distributed systems. Weis has a PhD in computer science from the Technical University of Berlin. Contact him at torben.weis@uni-due.de.

References

[AS13] J. Anderson and F. Stajano. Must social networking conflict with privacy? *Security Privacy, IEEE*, 11(3):51–60, 2013.

[BBS$^+$] Randy Baden, Adam Bender, Neil Spring, Bobby Bhattacharjee, and Daniel Starin. Persona: an online social network with user-defined privacy. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*.

[BHG$^+$] A. Bielenberg, L. Helm, A. Gentilucci, D. Stefanescu, and Honggang Zhang. The growth of diaspora - a decentralized online social network in the wild. In *Computer Communications Workshops, 2012 IEEE Conference on*.

[BSVD] Sonja Buchegger, Doris Schiöberg, Le Hung Vu, and Anwitaman Datta. PeerSoN: P2P social networking - early experiences and insights. In *Proceedings of the Second ACM Workshop on Social Network Systems Social Network Systems 2009, co-located with Eurosys 2009*.

[CMO] L.A. Cutillo, R. Molva, and M. Onen. Safebook: A distributed privacy preserving online social network. In *World of Wireless, Mobile and Multimedia Networks, 2011 IEEE International Symposium on a*.

[Dia13] Diaspora Inc. http://diasporaproject.org/, 2013.

[DMD] Michael Dürr, Marco Maier, and Florian Dorfmeister. Vegas - a secure and privacy-preserving peer-to-peer online social network. In *Social Computing, 2012 IEEE Fourth International Conference on*.

[DWM10] M. Dürr, M. Werner, and M. Maier. Re-socializing online social networks. In *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, pages 786 –791, dec. 2010.

[GGS$^+$] K. Graffi, C. Gross, D. Stingl, D. Hartung, A. Kovacevic, and R. Steinmetz. Lifesocial.kom: A secure and p2p-based solution for online social networks. In *Consumer Communications and Networking Conference, 2011 IEEE*.

[GKB12] B. Greschbach, G. Kreitz, and S. Buchegger. The devil is in the metadata - new privacy challenges in decentralised online social networks. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 333–339, 2012.

[NJM$^+$12] Shirin Nilizadeh, Sonia Jahid, Prateek Mittal, Nikita Borisov, and Apu Kapadia. Cachet: a decentralized architecture for privacy preserving social networking with caching. In *CoNEXT*, 2012.

[RMJM] F. Raji, A. Miri, M.D. Jazi, and B. Malek. Online social network with flexible and dynamic privacy policies. In *Computer Science and Software Engineering, 2011 CSI International Symposium on*.

[SBWW13] Lorenz Schwittmann, Christopher Boelmann, Matthäus Wander, and Torben Weis. SoNet – Privacy and Replication in Federated Online Social

Networks. In *Proceedings of the 2013 33rd International Conference on Distributed Computing Systems Workshops*, June 2013.

[UPS11] Guido Urdaneta, Guillaume Pierre, and Maarten Van Steen. A survey of dht security techniques. *ACM Comput. Surv.*, 43(2):8:1–8:49, February 2011.

[XZL08] Wanhong Xu, Xi Zhou, and Lei Li. Inferring privacy information via social relations. In *Data Engineering Workshop, 2008. ICDEW 2008. IEEE 24th International Conference on*, pages 525–530, 2008.