Exploring Decentralization Dimensions of Social Networking Services: Adversaries and Availability

Thomas Paul* Benjamin Greschbach† Sonja Buchegger† Thorsten Strufe*

*TU Darmstadt & CASED
Darmstadt
Germany
{thomas.paul, strufe}@cs.tu-darmstadt.de

†KTH Royal Institute of Technology
School of Computer Science and Communications
Stockholm, Sweden
{bgre, buc}@csc.kth.se

ABSTRACT

Current online Social Networking Services (SNS) are organized around a single provider and while storage and functionality can be distributed, the control over the service belongs to one central entity. This structure raises privacy concerns over the handling of large-scale and at least logically centralized collections of user data. In an effort to protect user privacy and decrease provider dependence, decentralization has been proposed for SNS. This decentralization has effects on availability, opportunities for traffic analysis, resource requirements, cooperation and incenctives, trust and accountability for different entities, and performance.

In this paper, we explore the spectrum of SNS implementations from centralized to fully decentralized and several hybrid constellations in between. Taking a systematic approach of SNS layers, decentralization classes, and replication strategies, we investigate the design space and focus on two issues as concrete examples where the contrast of extreme ends of the decentralization spectrum is illustrative, namely potential adversaries and churn-related profile availability. In general, our research indicates that hybrid approaches deserve more attention as both centralized as well as entirely decentralized systems suffer from severe drawbacks.

1. INTRODUCTION

Decentralizing Social Networking Services (SNS) has frequently been proposed to overcome shortcomings of centralized systems in recent publications. Removing the central provider has various advantages with respect to individual control, general service availability, privacy, attack resilience, and a decreased impact of misconfiguration and individual errors. Eliminating the single point of failure, or central bottleneck, hence is a valid motivation for the decentralization of service providers. Decentralization may increase the scalability and availability of services, helping to prevent service breakdowns or e. g., politically motivated service shutdowns.

The desire to protect the confidentiality of the vast amount of personally identifiable information that is stored in online social networks like *Facebook*, *twitter*, and *Google+* has been the primary driving force behind the numerous attempts to decentralize Social Networking Services. Centralized services act as gatekeepers that try to control the access of third parties to the personal information of a user. While denying it to unauthorized users, the providers can grant access to their affiliates, as well as to those users who have been authorized, and for this purpose they have full access to and full control over all data that is published within the system. Several cryptographic schemes [8, 2, 17, 9] have been proposed to overcome the omniscience of the provider. These approaches, however, can not achieve anonymous participation, confidentiality of communication acts, or anonymization of communication partners, let alone plausible deniability of participation, since they still rely on a centralized data store.

Several approaches to decentralize the Social Networking Service, or to integrate different SNSs, have been proposed, and some have been implemented. *Getsharekit*[1] and *sociallib*[2] are examples for general APIs that help integrating a few, large, existing providers. Shared information hence can be spread over several services, thus making it available to the designated receivers while keeping it partially hidden from overly curious providers. *Diaspora**[3] and *friend-of-a-friend* [18]

HotSocial '12, August 12, 2012, Beijing, China.

[1] http://www.getsharekit.com
[2] http://code.google.com/p/sociallib/
[3] http://www.diasporaproject.org

are closely related in that they propose to break up the provider into a few integrated, dedicated servers, using DNS for addressing. The entire profile of a user in this case is hosted on a single, remote provider. The providers, however, are decentralized and hence capable of accessing only the subset of profiles their registered users trusted them with. *LifeSocial* [6], *LotusNet* [1], *PeerSoN* [3], and *SafeBook* [5] are prominent, fully implemented or at least prototyped examples for systems that provide the service in an entirely decentralized, peer-to-peer (P2P) fashion.

Decentralizing the SNS may entail several side effects that have not satisfyingly been addressed by the proposed approaches so far. First, removing dedicated resources from the system, which then is characterized by high churn and very low reliability of the P2P-based service providers, makes guaranteeing availability of profiles very challenging. Second, applying redundancy and replication, that are the primary solutions to such lack of reliability, imposes high costs in terms of storage and communication overheads. Centralized servers, finally, are acting as de-facto anonymization mixes with respect to the data transmitted between the participants. Forwarding all messages over a series of decentralized devices, instead, simplifies the analysis of service requests and responses for the purpose of endpoint correlation, identification of individuals, or participation disclosure to third parties. The provider of centralized services has access to this information; decentralization removes this threat but can strengthen other adversaries.

The contributions of this paper are the following:

- We give an *overview of possible classes of decentralization*.
- Assuming the decision to decentralize the service provision we *analyze new vulnerabilities and attack surfaces* that evolve due to the service decentralization.
- We discuss the implications of the degree of decentralization on *profile data availability*.

The rest of the paper is divided into the six sections of (2) a brief introduction of definitions and the system layer model, (3) the description of different classes of decentralization, (4) adversary models, (5) a discussion of the consequences to security, (6) a discussion on availability and replica placement strategies, and (7) a conclusion and outlook to future work.

2. LAYERED SYSTEM MODEL

We define a *Social Networking Service* (SNS) as an Internet-based service that allows users to maintain profiles as their digital representations and explicitly declare connections between these profiles. A connection

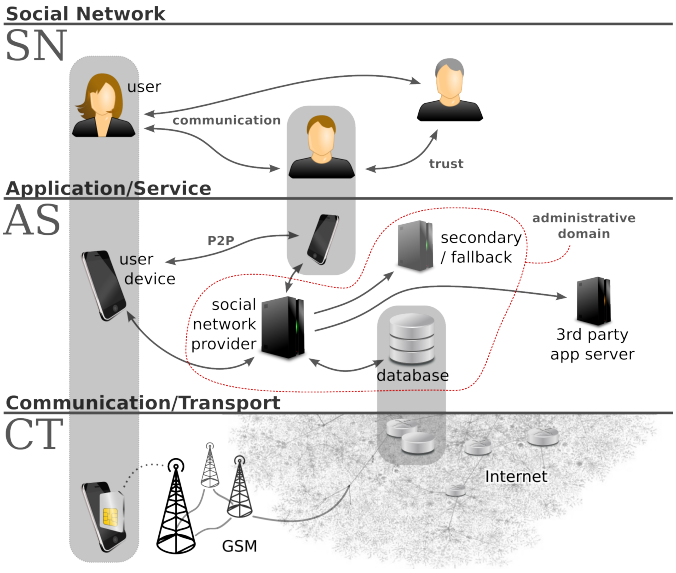

Figure 1: System model with three conceptional layers of a Social Networking Service.

between two profiles can represent friendship, trust, or other kinds of relations between the subjects.

To analyze the attack entry points and based on [4], we use a three-layer model to describe the Social Networking Service, as depicted in Fig. 1: The communication and transport (CT) layer, the application service (AS) layer and the social networking (SN) layer. The underlying CT layer is responsible for routing messages between network nodes and will usually comprise physical networks such as the Internet and mobile phone networks. The implementation of the SNS on top of a physical network forms the AS layer, where entities represent distributed applications, used and provided by the involved parties, such as clients representing members of the network, as well as primary and fallback servers of the SNS providers, their delegates, and their affiliates including third party application providers. The SN layer represents the network users and their real-world relations, such as friendships, trust relations and communication.

3. CLASSES OF DECENTRALIZATION

We distinguish three properties of possible architectural approaches in order to systematically analyze different proposed systems.

The (1) **decentralization** reflects the degree of decentralization of the network nodes. It can be (a) completely *centralized*, consist of (b) *distributed servers*, i. e., comprise several distributed centers, or be a (c) *peer-to-peer* approach, i. e., completely decentralized. The location of (2) **data integration** distinguishes between (a) *local* data integration, where isolated datasets are integrated locally at the client side (e. g., usage of several

services) and (b) *remote* data integration, e. g., all user data is integrated remotely at the server side. Finally, (3) **communication paths**, describe the difference between (a) *direct* communication, having the possibility of direct connections between members of the network (potentially after a prior lookup phase), and (b) *relayed* communication only, where all communication is recursively routed through servers or overlay nodes.

Table 1 shows the resulting classes of decentralization, when considering all meaningful combinations of the three properties. The location of data integration is only relevant for decentralized servers. For centralized systems it is necessarily located at the server side, for P2P systems usually at the client side. While we list a wide range of classes of decentralization, not all of these classes have been used in current proposals for decentralized SNS designs.

3.1 Centralized

A SNS that is based on an – at least logically – centralized network structure is the common case of today's popular applications, such as *Facebook*, *Google+* or others[4]. All traffic is either relayed or at least mediated by the central provider. This introduces a limited mix functionality: an adversary sniffing on the CT layer is unable to correlate communication endpoints, given higher layer confidentiality. Content is stored centralized and the provider is responsible for enforcing user-defined access control policies. All system internal entities are located in one administrative domain, but data access interfaces are usually provided for affiliates, such as third party application servers.

3.2 Peer Assisted Centralized

Peer-assisted centralized SNSs are a combination of a centralized network and a flat P2P system. One logically centralized server is used for registration, identity management and other tasks, that can profit from a centralized design. Content distribution, synchronous interactions and other suitable operations, however, can leverage direct connections between the peers, having the central server as a fallback solution. While this class so far has not been proposed for SNS designs, it represents distributed approaches like, e.g., skype, or other common audio/video conference systems.

3.3 Decentralized Servers

A system of decentralized servers aims to avoid the comprehensive data aggregation in the domain of one SNS provider. Hence it consists of multiple servers, each responsible for one or several clients. The servers host the profile data of assigned clients. Moreover, they act as message storage for incoming messages, that cannot be delivered to temporarily unavailable clients. Representing a proxy of their users, the servers are required to be online and available. *Diaspora**[5] is one example in this class, where several users can be hosted on each of the servers. In *Vis-a-Vis* [13], each server typically hosts a single profile.

Peer-assistance can be used for these approaches to mitigate the server load. Clients are allowed to exchange data directly, when both communication partners are online simultaneously.

3.4 Common Interface Decentralized Services

This approach consists of several different SNS, where a user maintains one identity in each service. The connection between the SNS is a common user interface for managing the integrated services. One example in this class is *onesocialweb*[6]. Here again, a peer-assisted version is conceivable, leveraging direct connections between users. These might, however, not cross SNS borders, since maintaining multiple identities would otherwise not make sense any more.

3.5 Pure P2P and Recursive Routing

In P2P systems, user data is usually hosted by the members themselves, while content replication is used to increase availability. Access control is typically realized by encrypting the content, allowing only the intended recipients to decipher it. While in some cases the overlay network serves only as pragmatic routing substrate, e. g., to implement a DHT, other approaches leverage trust relationships of users to form network edges. Trusted paths for package transport in combination with source address rewriting at each step [4] provide the possibility to hide the identities in the communication and transport layer, comparable to darknets.

In flat P2P systems, every node represents one subject that is member of the network. There are no servers and no explicit distinctions between nodes, although high-degree nodes can become more attractive attack targets. *SafeBook* [5] is one example in this class. Explicitly hierarchical P2P systems, such as *SuperNova* [14], distinguish between plain clients and dedicated *supernodes*. The latter are used for bootstrapping new members, managing directories or other tasks that rely on certain properties, such as high availability, performance, or security.

3.6 Pure P2P and Iterative Routing

This class is similar to Section 3.5, with the difference that direct links, that are not based on the overlay, are allowed, using addresses of the underlying network (e.g. IP addresses). Consequently, this approach does not provide anonymity with respect to the IP layer,

[4] e. g., linkedin.com, xing.com, myspace.com, netlog.com

[5] http://www.diasporaproject.org

[6] http://onesocialweb.org

Decentralization	Integr.	Comm.	Class
centralized	remote	relayed	3.1 Centralized (e. g., *Facebook*, *Google+*)
		direct	3.2 Peer-assisted Centralized
decentralized servers	remote	relayed	3.3 Decentralized Servers (e. g., *Diaspora**, *Vis-a-Vis*)
		direct	(see 3.3, peer-assisted version)
	local	relayed	3.4 Common Interface Decentralized Services (e. g., *onesocialweb*)
		direct	(see 3.4, peer-assisted version)
peer-to-peer	local	relayed	3.5 Pure P2P and Recursive Routing (e. g., *SafeBook*)
		direct	3.6 Pure P2P and Iterative Routing (e. g., *PeerSon*, *Persona*)

Table 1: Classification of architectural approaches based on system properties.

but message paths are shorter (fewer hops) and fewer nodes might have access to the messages (or the cipher text). Some decentralized SNS approaches, such as *PeerSoN* [3], *Persona* [2] or *Decent* [10], are based on this concept.

4. ADVERSARY MODELS

To characterize possible adversary models, we first discuss general goals of attackers. Then we address the attack surfaces and attack vectors on the three system layers from [4].

4.1 Adversary Goals

An adversary may want to affect one or more parts of security and privacy: confidentiality, integrity, service availability. Thus he may want to deanonymize a target user, get access to confidential information that is not addressed to him, may falsify information to influence subjects or he may attempt to prevent subjects from using the system by destroying the functionality.

4.2 Attack Surfaces and Vectors

Communication and Transport Layer.

On the CT layer, the natural attack surface for an adversary is the network communication. Network traffic can be sniffed locally, e. g., in a wireless network environment, or more globally, e. g., by a malicious ISP. Besides passively observing traffic, an adversary can intervene with the communication and filter or block traffic. Furthermore, an adversary might try to delay traffic, in order to disrupt time-critical synchronous communication.

An even more active adversary can try to manipulate or inject packages, although on the CT layer this might not be practical, because creating a meaningful packet requires perfect knowledge of several network parameters. Finally, an adversary can perform denial of service (DoS) attacks against the SNS network components.

All these attacks are not specific to the domain of SNS, but have to be taken into account when analyzing the security and privacy properties of these services. Furthermore, different SNS implementations expose different observable information on this layer.

Application and Service Layer.

On the AS layer, several attack surfaces are available for an adversary. Again, network traffic can be sniffed, manipulated, blocked and injected, this time, informed by routing characteristics of the AS layer. Adversaries can try to exploit high-value nodes, e. g., high traffic volume nodes or those responsible for routing traffic of a certain target user.

Stored data is another attack surface on the AS layer. Malicious storage nodes do have extended access to private content data they store (plain- or ciphertext, depending on the implementation) and can evaluate request logs to analyze access patterns for this content. Active attackers may even modify or delete stored content. External adversaries can at least crawl the network and harvest accessible storage objects.

Moreover, an adversary can exploit vulnerabilities of the identity management employed by the SNS. Possible attacks in this category comprise impersonation (using the existing ID of a target user to act on her behalf), spoofing (using a new, falsified ID), sybil attacks (creating and orchestrating a large amount of fake IDs), and eclipse attacks (surrounding a target user by adversary-controlled nodes). Adversaries with system-internal friendships to a target user (including indirect ones, e. g., friend-of-a-friend relations), can exploit the extended legitimate data access that comes with this status.

Finally, weaknesses of the protocols employed on the AS layer, can be exploited by attackers. Besides obvious security holes that allow for illegitimate actions, such as deleting a profile, other types of attacks might be possible, such as specialized DoS attacks (e. g., flooding of SNS requests). Attacks on APIs for interactions with third-parties, delegates or replicas also fall in this category.

Social Networking Layer.

On the SN layer, the users' vulnerability to social

engineering is the main attack surface for adversaries. It allows for, e. g., social pressure, phishing and password theft. Furthermore, an adversary can use background knowledge about a target user, that was acquired service-externally (friend adversary). It allows to interpret information that was collected inside the system or to mount inference attacks on sparse raw data, that on its own might not have been critical with respect to user privacy (e. g., sparse location data that together with background knowledge about preferred places of the user enable precise localization with high probability).

5. SECURITY DISCUSSION

In the following we compare the different classes of decentralization, described in Section 3, with respect to trust models and the different adversary threats, discussed in Section 4.

5.1 Trust Models

SNS affiliates are part of a broad spectrum of trust relationships with legal trust at one end, and interpersonal social trust at the other end.

A central SNS providing company is a single administrative domain that can be identified by the users and thus can be sued in the case of misbehavior. This allows for legal trust in the company to respect the law of at least the country where it has the registered office.

A decentralized SNS consists of more than one administrative domain, that store user data and they may not be a registered company, subject to the legal accountability that this status entails. In addition, the administrative domains may be situated in different countries with different law systems, further complicating legal trust. Therefore, trust in a decentralized SNS is rather based on technical mechanisms like cryptography and interpersonal as well as institutional reputation. Distributed server models are divided into several administrative domains, each comprising the machines related to one server. P2P approaches can even be seen as having one administrative domain for each network member. While increasing the number of domains allows for distributing trust on several parties (as all of them have to collude in order to harm the user in the same way as a single provider), it also requires more inter-domain communicating links, which are more exposed to adversaries.

5.2 Adversary Threats

Attacks on the social layer do not depend on the technical implementation of the SNS. Some of them can be related to properties of the user-interface (e. g., phishing), but the interface is in general independent of the degree of decentralization in the underlying system. For the following discussion we therefore focus on attacks on the communication and transport layer, as well as on the application and service layer.

Centralized systems (Section 3.1) expose very limited entry points for external adversaries, and by aggregating all user traffic on a small number of machines, the central provider performs a kind of traffic mixing, which makes it hard for external observers to infer sensitive information from traffic analysis. However, in a centralized system the provider hast to be trusted, not only not to misuse or sell the massively accumulated private user data, but also to protect it perfectly from unintentional leakages, attacks, and curious employees.

Peer-assisted centralized systems (Section 3.2) open up some more attack surfaces for external adversaries compared to a purely centralized system. As part of the traffic is routed directly between peers, the implicit mix property of centralized systems is lost: network sniffers can infer interactions between peers by monitoring communication partners and traffic volume. The central party still has an almost comprehensive view of the users' activities and remains a major threat to user privacy if it is not fully trusted.

Distributed Servers (Section 3.3) have the potential to combine the properties of centralized and decentralized approaches. Trust requirements are usually distributed over several servers, and users are free to choose a certain server based on experience and reputation. Traitor attacks (i. e., a server first behaves honestly to gain reputation and exploits that trust later) are still possible but mitigated since trust is not mainly based on recent behavior of the servers, but more on the reputation of the server maintaining parties (e. g., communities, companies or private persons).

Relying on a single one of the servers is not required since data can be stored redundantly (e. g., a complete copy of a user's profile at the user's device, or on fallback servers) to minimize data loss in case a single server turns malicious or is no longer maintained.

Pure P2P approaches (Section 3.5 and Section 3.6) do not have the requirement of trusting a central party. While this constitutes a major advantage for user privacy, these systems have to cope with other challenges [7]: The complete decentralization of the network content and functionality implies exposing all system- and metadata to anybody observing the network, including all participants on the path along which a message is forwarded.

Even though content is usually encrypted in this kind of systems, metadata about the content or data generated while managing the content, can reveal sensitive information with the potential to invade the users' privacy. This holds especially for the communication partner identification as well as for the frequency and volume of data exchange. Thus it can tell an observer qualitative information about the relation of the com-

municating members. The size of a storage object can indicate its content type (e. g., the difference between text posts and pictures), statistical information (e. g., the length of a post), or act as a fingerprint to track a specific content, even if it is re-encrypted under different keys when shared by different users.

Finally, the identity and relationship management, e. g., the distribution of cryptographic keys, can leak sensitive information to network-sniffing adversaries, such as the content audience of encrypted objects or friendship status changes of network members. Hierarchical P2P systems have the potential of hiding some of the system internals from outsiders, by entrusting supernodes with certain crucial tasks. The selection mechanisms for supernodes is usually based on automatic evaluations of node properties, such as availability. Therefore the design is vulnerable to sybil attacks or other approaches with the aim to tamper with the evaluation results in favor of adversary-controlled nodes.

Systems that employ recursive routing (Section 3.5) can facilitate communication anonymization, as identifiers of the communication endpoints can be hidden from external observers. They are characterized by a higher dependence of users on the forwarding nodes, and hence may be more vulnerable to insider attacks.

We conclude, that hybrid approaches, as described in Section 3.3, are less exposed to the metadata vulnerabilities of pure P2P approaches since servers exist, which act as implicit mixes. Furthermore, there is no central authority with total access to all user data, like in a central SNS. A user, joining an SNS based on a hybrid approach still needs to solve the trade-off to choose a server maintained by an authority whom he trusts, or engage in the challenge of providing and maintaining its own server, thus potentially abandoning the mixing of a server, which hosts multiple profiles.

6. PROFILE DATA AVAILABILITY

To allow users to access profile data of other users, it needs to be available, respecting reasonable delay. In this section, we give our notion of availability and provide an overview of how the different approaches aim to tackle this issue, since the profile accessibility and thus the availability of the data is one key issue when building an SNS.

For the scope of this paper, we define profile availability to be the fraction of time (baseline: 24/7) in total that a profile is available to others. Approaches based on dedicated storage resources potentially reach full-time (churn related) profile availability by definition. Another notion of availability, that is sometimes used for SNS, is to restrict the scope to the time when friends, who might need to access the data, are most likely to be online. This, however, leads to more optimistic estimations and does not take into account friendships attempted to be established during offline times or profile request from friends of friends.

In case of a centralized SNS, the server is the place to store the data and provide access at all times, whether a particular user is online or not. While this can be realized with a distributed set of servers or using content distribution networks, the control over the storage is in the hand of a single entity and we thus consider this a logically centralized setup. In contrast, in the case of fully decentralized approaches using non-reliable profile storage (e. g., P2P), more care needs to be taken to keep data accessible even when the owner of the data is not online. This can be achieved by other mechanisms, such as replication.

Different strategies exist to achieve data availability in a P2P-based, decentralized SNS under condition of churn. First, replicas can be **spread randomly across the network**. Second, the **friends' nodes may hold replicas of the profile data**, since relevance of data and good behavior are assumed more likely due to the trust relationship of being friends [5]. Finally, **nodes can be selected based on different metrics** to store replicas, aiming to achieve the highest possible profile availability while minimizing network traffic and storage overhead. In the following, we discuss these replica placement strategies in terms of their impact on availability.

6.1 Random Selection of Replica Nodes

Randomly storing profile copies at unrelated nodes may lead to a large number of necessary copies. Assuming uniformly random distributed online times, which represents a favorable assumption, leveling out deviations in user density over varying timezones, we make the following back-of-the envelope calculation to illustrate the worst case, when replicas are only available as long as the SNS session is active (in the remainder we use the term *online* as a shorthand for this). Given this restriction, the profile availability PA is calculated as:

$$PA = 1 - (1 - OF)^R \quad (1)$$

where OF is the fraction of time a node is online on average, and R is the number of replicas. This number of replicas also equals the number of profiles each node needs to store and serve (for n users, in total $R \cdot n$ profile copies have to be distributed over n nodes).

According to a recent study[7] *Facebook* users have 130 friends, and publish 90 items a month on average. This translates to an average of three profile updates per day. It further states that the 750 million users included had been online for 700 billion minutes a month in total,

[7] http://www.internetworld.de/Specials/Facebook/Zahlen-und-Fakten/Facebook-Nutzung-weltweit-Die-offizielle-Statistik

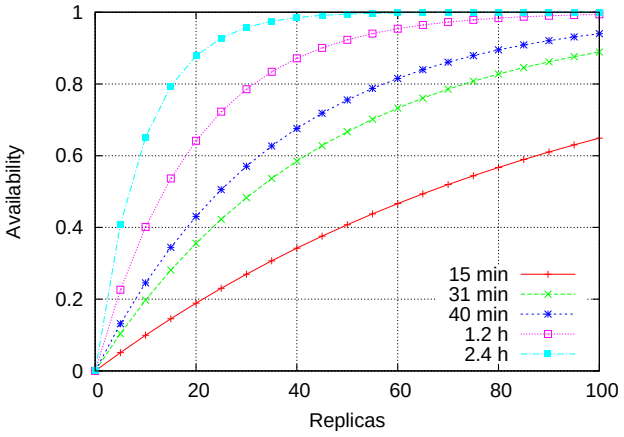

Figure 2: Availability of profiles depending on the number of replicas, for different SNS online times of replicating devices.

or 31 minutes per day and user. Schneider et al. [12] analyzed passively monitored network traffic of "tens of thousands of users at different ISPs" in 2008 and showed "that OSN sessions exhibit high variability, with many lasting a very short period of time and a few lasting for hours, with a mean of about 40 minutes". The study hence supports the numbers given in the first source, even if a user might not necessarily have exactly one session per day: both give a general order of magnitude for the time users utilize the service.

To give an idea about how many replicas are necessary in order to reach a certain availability, we plot the cases that nodes are part of the network for 2.4 hours a day (10% of the time), 1.2 hours (5%), 40 minutes, 31 minutes, and 15 minutes per day on average (Fig. 2).

Random peer selection for profile data replication in decentralized SNS in this light does not seem to be feasible for high availability requirements, especially if a profile is considered accessible only when one of the replicating nodes are engaged in an SNS session. Even considering simplified, favorable circumstances and realistic session times, profile data availability requires very high replication factors (cmp. (Fig. 2). Therefore, more sophisticated replica placement strategies have been proposed, as described in the following.

6.2 Friend Storage

Several approaches propose to chose a users friends for replication, since they are both assumed to be interested in the content and to cooperate in favor of the user [5, 11]. In case of replicating profile data at friends' nodes, it is not unlikely that the friends live in the same – or at least close – time zones and thus are offline at similar times. Furthermore, storing replicas at friends' nodes causes a bootstrapping problem. When a new user joins the network, she does not have any friendship connections. The profile will therefore not have enough replicas in the network to be sufficiently available to be found by other users.

Sharma et al. [15] conducted an empirical study of availability in friend-to-friend storage systems. They observe that "roughly 50% nodes can achieve at least 90% of coverage". Thus, if every node holds a copy of the data of all friends, only roughly 50% of the datasets can be held available for at least 90% of the total time.

6.3 Metric-based Replication Strategies

Tegeler et al. introduce Gemstone [16], a more sophisticated approach to select replicas for storing SNS content in a decentralized manner. Aiming at reducing the number of necessary copies to achieve a convenient availability, the authors suggest a selection strategy based on (1) the online time represented by the average online probability, (2) the social relation (binary friendship indicator), and finally, (3) an "online experience".

SuperNova [14] introduces "super nodes" and "storekeepers" to increase the availability of profile data. A node joining the network first uses a chosen super node to bootstrap and replicate the data until it has enough edges on its own. This approach differentiates between two kinds of edges: friends and storekeepers. To be storekeeper is a unidirectional connection between nodes, representing the willingness to replicate the profile data of another node. Storekeepers are a manually chosen subset of friends supplemented by asking the super node to convey additional storekeeping nodes beyond the own friendship horizon.

The discussed approaches are effective strategies of selecting replica nodes with respect to decreasing number of replicas while maximizing the profile availability and thus help to improve the availability of profile data in a P2P-based SNS. Nevertheless, the following issues indicate that more research is needed. So far, metric-based approaches fall short of the standard expectation of 24/7 availability in centralized systems or other approaches using dedicated services. Furthermore, the reliance on online time measured and advertised by replicator nodes themselves has some disadvantages: saving bandwidth and storage resources is a strong incentive to lie and the online time might be highly dynamic, thus resulting in the past not necessarily being a good estimator for the future behavior (e. g., weekend vs business days), and the online time might be considered as a private information. The "online experience" leads to preferring nodes as storage with similar temporal online patterns which is suboptimal for continuous data availability. Preferably using socially related nodes may cause privacy issues, since a social relation may create a strong interest in learning information about the related user by observing SNS usage patterns. Another

open issue for replication strategies is not only to maximize availability but also to take bandwidth needs into account, which is especially relevant in the presence of very popular profiles and temporal popularity peaks (e. g., caused by media attention).

7. CONCLUSION

This paper discusses six classes of decentralization for implementations of Social Networking Services. Identifying the central provision of an SNS as a potential threat to the privacy of the users, numerous proposals to decentralize the service have been made in the recent past. Common ground of these proposals is to aim at removing an omniscient central entity. The decentralization, while offering benefits with respect to reducing both the data exploitation surfaces for the service provider as well as the existence of a high value attack target for adversaries, comes with several, potentially undesired side effects, that so far have broadly been disregarded.

This paper introduces a classification of decentralization degrees for Social Networking Services, which can be used to formalize decentralized service architectures, and that helps to identify possible attack surfaces as well as classes of adversaries. It is subsequently applied to help identifying drawbacks of purely decentralized approaches, highlighting the fact that pure decentralization may introduce disadvantages regarding privacy as well as availability of profiles.

We have started to investigate hierarchical, hybrid architectures of distributed servers as they seem a promising alternative combining positive characteristics of centralized as well as fully decentralized approaches at the chance of avoiding their specific flaws.

8. REFERENCES

[1] Luca Maria Aiello and Giancarlo Ruffo. Secure and Flexible Framework for Decentralized Social Network Services. In *SESOC*, 2010.

[2] Randy Baden, Adam Bender, Neil Spring, Bobby Bhattacharjee, and Daniel Starin. Persona: an online social network with user-defined privacy. In *SIGCOMM*, 2009.

[3] Sonja Buchegger, Doris Schiöberg, Le Hung Vu, and Anwitaman Datta. PeerSoN: P2P social networking - early experiences and insights. In *Workshop on Social Network Systems*, 2009.

[4] Leucio-Antonio Cutillo, Mark Manulis, and Thorsten Strufe. Security and Privacy in Online Social Networks. *Handbook of Social Network Technologies and Applications*, 2010.

[5] Leucio-Antonio Cutillo, Refik Molva, and Thorsten Strufe. Safebook: Feasibility of Transitive Cooperation for Privacy on a Decentralized Social Network . In *WoWMoM*, 2009.

[6] Kalman Graffi, Sergey Podrajanski, Patrick Mukherjee, Aleksandra Kovacevic, and Ralf Steinmetz. A Distributed Platform for Multimedia Communities. In *International Symposium on Multimedia*, 2008.

[7] Benjamin Greschbach, Gunnar Kreitz, and Sonja Buchegger. The Devil is in the Metadata – New Privacy Challenges in Decentralised Online Social Networks. In *SESOC*, 2012.

[8] Saikat Guha, Kevin Tang, and Paul Francis. NOYB: Privacy in Online Social Networks. In *First Workshop on Online Social Networks*, 2008.

[9] Felix Günther, Mark Manulis, and Thorsten Strufe. Cryptographic Treatment of Private User Profiles. *Lecture Notes in Computer Science (LNCS)*, 7126, 2012.

[10] Sonia Jahid, Shirin Nilizadeh, Prateek Mittal, Nikita Borisov, and Apu Kapadia. DECENT: A Decentralized Architecture for Enforcing Privacy in Online Social Networks. In *SESOC*, 2012.

[11] Rammohan Narendula. The Case of Decentralized Online Social Networks. Technical report, EPFL, 2012.

[12] Fabian Schneider, Anja Feldmann, Balachander Krishnamurthy, and Walter Willinger. Understanding Online Social Network Usage from a Network Perspective. In *IMC*, 2009.

[13] Amre Shakimov, Harold Lim, Ramón Cáceres, Landon Cox, Kevin Li, Dongta Liu, and Alexander Varshavsky. Vis-a-Vis: Privacy-Preserving Online Social Networking via Virtual Individual Servers. In *ComsNets*, 2011.

[14] Rajesh Sharma and Anwitaman Datta. SuperNova: Super-peers Based Architecture for Decentralized Online Social Networks. Technical report, ArXiv e-prints, 2011.

[15] Rajesh Sharma, Anwitaman Datta, Matteo Dell'Amico, and Pietro Michiardi. An Empirical Study of Availability in Friend-to-Friend Storage Systems. In *Peer-to-Peer Computing (P2P), 2011 IEEE*, 2011.

[16] Florian Tegeler, David Koll, and Xiaoming Fu. Gemstone: Empowering Decentralized Social Networking with High Data Availability. In *Globecom*, 2011.

[17] Amin Tootoonchian, Stefan Saroiu, Yashar Ganjali, and Alec Wolman. Lockr: Better Privacy for Social Networks. In *CoNEXT*, 2009.

[18] Ching Man Au Yeung, Ilaria Liccardi, Kanghao Lu, Oshani Seneviratne, and Tim Berners-Lee. Decentralization: The Future of Online Social Networking. In *W3C Workshop on the Future of Social Networking*, 2009.