

# Decentralized Online Social Networks

Anwitaman Datta<sup>1</sup>, Sonja Buchegger<sup>2</sup>, Le Hung Vu<sup>3</sup>, Thorsten Strufe<sup>4</sup>, and Krzysztof Rzadca<sup>1</sup>

<sup>1</sup> NTU Singapore, [anwitaman@ntu.edu.sg](mailto:anwitaman@ntu.edu.sg), [krzadca@gmail.com](mailto:krzadca@gmail.com)

<sup>2</sup> Royal Institute of Technology (KTH), Stockholm, Sweden. [buc@kth.se](mailto:buc@kth.se)

<sup>3</sup> École Polytechnique Fédérale de Lausanne (EPFL), Switzerland.  
[lehung.vu@epfl.ch](mailto:lehung.vu@epfl.ch)

<sup>4</sup> TU Darmstadt, Germany. [strufe@cs.tu-darmstadt.de](mailto:strufe@cs.tu-darmstadt.de)

**Abstract.** Current Online social networks (OSN) are web services run on logically centralized infrastructure. Large OSN sites use content distribution networks and thus distribute some of the load by caching for performance reasons, nevertheless there is a central repository for user and application data. This centralized nature of OSNs has several drawbacks including scalability, privacy, dependence on a provider, need for being online for every transaction, and a lack of locality. There have thus been several efforts toward decentralizing OSNs while retaining the functionalities offered by centralized OSNs. A decentralized online social network (DOSN) is a distributed system for social networking with no or limited dependency on any dedicated central infrastructure. In this chapter we explore the various motivations of a decentralized approach to online social networking, discuss several concrete proposals and types of DOSN as well as challenges and opportunities associated with decentralization.

## 1 Introduction

Adapted from the original definition in [BE07], we define an online social network (OSN) as an online platform that (1) provides services for a user to build a public profile and to *explicitly* declare the connection between his or her profile with those of the other users; (2) enables a user to share information and content with the chosen users or public; and (3) supports the development and usage of social applications with which the user can interact and collaborate with both friends and strangers.

Current online social networks are extended in two main directions towards the capabilities of the provided services and the decentralization of the supporting infrastructures, as depicted in Fig. 1.

Along the first dimension, the features and services provided by online social networks have been extended significantly. Early social networking sites, from a simple online tool to manage personal and professional contacts, such as in

SixDegrees and Friendster, to an effective tool for sharing several kind of information and contents with a viral spread. Popular OSNs such as Facebook offer users even more services and applications, as third-parties are allowed to develop and plug their applications into the site. The OSN has come closer to being a full-fledged development platform for social applications. If a Web browser becomes an operating system for the next-generation computing devices as predicted by various technological experts, it is very likely that online social network service would be the user interface of that operating system, i.e., it provides users a portal to manage their Web-wide personal and social information sources.



**Fig. 1.** Classification and development trend of online social network services

Another trend of extending current online social network services is towards the decentralization of the backend infrastructure. Centralized social networking services are prone to some problems, some of which have even led to the demise of many early-generation OSN sites such as SixDegrees and Friendster [BE07]. The problems include both technical and social issues that emerge as a consequence of a centralized management of the services. On the technical side, the centralized management of a social network with rapid growth in user popularity has led to various performance scalability issues, most notably the frequent down-time of the micro-blogging service Twitter and similar slowness and unresponsiveness of Facebook to many users <sup>5</sup>. Along with the increasing popularity of the service comes the increasing cost of management and maintaining the infrastructures to ensure a smooth continuation and reasonable performance of the provided

<sup>5</sup> [http://www.watchmouse.com/en/SPI/2008/performance\\_social\\_networking\\_sites.php](http://www.watchmouse.com/en/SPI/2008/performance_social_networking_sites.php)

services. On the social side, the unlimited sharing capability of information has also led the social collisions in the sense that the fast growth of the networking sites, without proper privacy preserving schemes, leads to a collapse in social contexts, for instance, users had to face their bosses and acquaintances alongside their close friends and so information is shared to many unwanted contacts. The ruptures of trust between the users and the site providers were also among the key problems leading to the collapse of such early social networking services.

A decentralized online social network is an online social network implemented on a distributed information management platform, such as a network of trusted servers or a peer-to-peer systems. In contrast to centralized OSNs where the vendor bears all the cost in providing the services, a distributed or peer-to-peer OSN offers a cost-effective alternative. In fact, a P2P approach helps lower the cost of the provider drastically, e.g., the case of Skype [LPG<sup>+</sup>07]. Other benefits of a DOSN include better control of user privacy and the enhancement of innovative development [BD09a, SW09]. By decentralizing OSNs, the concept of a service provider is changed, as there is no single provider but a set of peers that take on a share of the tasks needed to run the system. This has several consequences: in terms of privacy (no central data collection, reduced economic incentive for advertisement) and operation, nor any central entity that decides or changes the terms of service whimsically. Moving from a centralized web service to a decentralized system also means that different modes of operations become possible: using one's own storage or cloud storage, delay-tolerant social networks, and local treatment of local content, to name some of them.

DOSNs combine *social* and *decentralized* elements. The path to a system with these properties vary: a DOSN can be achieved by adding one of these two properties to an existing system and thus transforming it, as by taking centralized OSNs and decentralizing them or by adding a social component to current decentralized applications that do not have a social component yet. It can also be achieved by adding both properties at the same time: decentralizing and adding a social component to current centralized applications. We discuss the first of these transformations, i.e., going from a generic OSN to a DOSN in Section 4 and decentralizing application specific OSNs in Section 5, and adding social functionalities to distributed systems in Section 6 on social distributed systems.

## 1.1 Scope of the Chapter

In this chapter, we provide an overview of current (mostly, ongoing) approaches to realize DOSNs. We summarize what each of these projects plan to achieve and what their current status is, as well as what key innovations have been achieved in these mostly nascent initiatives. We discuss several design choices and challenges first in general and then in the context of particular projects.

In addition to entire self-contained DOSN systems, there is a range of individual functionalities that can be combined and integrated in a DOSN. Generic mechanisms for distributed systems, such as cryptography for privacy, key management, storage distribution, numbers and location of replicas, incentives for cooperation and resource-sharing, topology, p2p substrate and administration tasks, trust needed for the program, etc. may be developed either within the context of DOSNs from the start or be adapted from other contexts. We touch upon several of these generic mechanisms in the following sections when needed, but an exhaustive treatment of these enabling techniques is out of the scope of this chapter.

## 2 Challenges for DOSN

In this section, we list some challenges for decentralized social networks. While many of them are technical in nature, some are trade-offs that depend on preference, such as whether privacy should be prioritized or search. Such trade-offs are closely connected to technical questions and are thus included.

Decentralizing the existing functionality of online social networks requires finding ways for distributed storage of data, updates propagation and versioning, a topology and protocol that enables search and addressing, i.e. a mechanism to find friends in the topology, robustness against churn, openness for third-party applications, and means for content revocation (by encryption and/or time), dealing with heterogeneity of user resources, demands, online behavior, etc. We discuss these challenges in more detail below.

**Storage.** Where should content be stored? Should they be stored exclusively at nodes run by friends, or be encrypted and stored at random nodes, or should the nodes be chosen using some other heuristics such as in a DHT or based on uptime history? As in file-sharing, there will be several answers to this question. The requirement for redundancy to provide availability of data depends to a large extent on the duration and distribution of time peers are online. These activity patterns are also influenced by the geographic distribution of the peers and shifted by time zones. The distribution of interested and authorized peers and the desired probability of availability are to be traded off with storage requirements, especially if the system should allow for storing of media files and not only links to websites where such media files can be found.

**Updates.** How can we deal with updates, e.g. status updates of friends? In peer collaboration systems, updates, e.g. of a workplace, are sent to a small group of peers via a decentralized synchronization mechanism. In P2P social networks, with distributed storage and replication - and a potential need for scalability, the requirements change. P2P publish/subscribe mechanisms are a possibility, but their security in terms of access control will have to be developed further. Unlike a traditional peer-to-peer environment, where many peers are involved, each of

the sub-networks will be much smaller (though larger than typical collaborative groups), making it relatively simpler to realize quorum systems and deal with updates.

**Topology.** Should nodes be connected according to their social connection? This would cluster friends in the overlay network, which would facilitate updates. As a downside, given the possibility of a relatively small set of friends, this would limit the availability and robustness of data access. How can we build a decentralized, p2p topology suitable for social networks? In pure file-sharing networks, the topology does not depend on whether the peers know each other and nodes exchange content with any other nodes in the network. At the other end of the spectrum, existing examples of decentralized social networks (in the widest sense) are mostly platforms for collaboration or media sharing and they tend to consist of collaborative groups that are relatively closed circles, e.g. using a “ring of trust” or darknets. In contrast, online social networking services have overlapping circles.

**Search, Addressing.** Related to the topic of updates above, how can users find their friends from the real social network in the P2P virtualization thereof, and conversely, how can they discover new friends by virtue of common interests. Over multiple sessions, peers may change their physical address. In a typical file sharing network, this is not an issue. One just needs to find some peer with the content it is looking for. However friends and trust links of a social network are essential, and so it is crucial to both be able to find back friends even if they have changed their physical address, and also authenticate the identity. Traditionally, peer identity is tied with an IP address [PGW<sup>+</sup>08a] which clearly is not sufficient. However, handling peer identity in a self-contained manner in a P2P system is also feasible [ADH04] (also partially in Skype [Sky04]). It may also be difficult to maintain a complete ring like in traditional structured overlays as an index structure, if the network is based on only social links. Recent advances in realizing distributed indexing with a ringless overlay [GGD<sup>+</sup>08] potentially holds the key to this issue. These mechanisms composed together potentially can help maintain social network links under churn. Another search issue is - how can users find out about information available concerning their interests? In social networks, tagging or folksonomies is the basic mechanism to annotate content. Recently, there have also been advances made in enabling decentralized tagging [GSS08], which paves another step towards realizing social networks on top of a P2P infrastructure. Note that there is a trade-off between privacy protection and search capabilities.

**Openness to New Applications.** One of the most alluring features of current online social networks is that they are open to third-party applications, which enables a constant change of what a social networking service provides to the users. There is a core functionality for maintaining social ties, such as profile information, connection to friends, status updates, internal messaging, posting on each other’s sites, events notification. In addition, third-party applications provide more and unpredictable ways of contacting users, finding out about other users’ interests, forming groups and group identities, etc. This openness

to extensions potentially provides great benefits for the users. The price for these benefits is the risk that comes with opening the service to untrusted third parties, extending the privacy problem from the single service provider to all application providers. In a decentralized environment, if some users choose to enable a third-party application, their choice should not affect other users or even users connected directly to them. How to draw this boundary is an open and challenging question.

**Security.** Keeping control over their data with the user implies the need for security support, so the classical requirements for security (confidentiality, access control, integrity, authentication, non-repudiation) apply, albeit modified for the context of decentralized social networks. The main questions for user control are in the domain of access control, e.g. how can we ensure that only authorized friends can access content. For distributed storage with other peers that the user not necessarily wants to access data, the content has to be encrypted, as done for example for file backup [ABC<sup>+</sup>02] or anonymous peer-to-peer file-sharing [DFM00, CSH00]. To manage access to encrypted data, key distribution and maintenance have to be handled such that the social network group can access data but be flexible enough to handle churn in terms of going offline and coming back, additions and removal to the user's social network. Group membership research has dealt with questions of key management and renewal and how to give access to new members of a group by issuing new keys in rounds [STW00]. Likewise darknets [Fra03] also share a key within the group. Such existing mechanisms are however grossly inadequate to meet the finer granularity of access control needs for social network features.

Even in the most simple scenario, where all members are allowed full access, if one wishes to realize control on membership itself, then sharing a secret key is not enough. Any member who already has the shared key can pass it on to new members. Therefore, keys and identities need to be combined for access control, but without access to a file system, mechanisms like access control lists are not feasible. In many online networks (for example Yahoo! Groups<sup>6</sup>), a smaller subset of members own and moderate the membership of a group. Thus even a minor variation of the basic groups like darknets, to realize a group where all members still have equal access to content, but only a subset of members control the membership itself, is non-trivial in a decentralized setting.

There is on top of that the need for a finer granularity of access control, determining who can read, write or modify and delete each shared object, and how to enforce such access control in a decentralized setting, while still guaranteeing non-repudiation as well as preventing impersonation and replay. Achieving such finer granularity of access control in a decentralized manner is, we believe one of the hardest security challenges, and the biggest hurdle in realizing a P2P infrastructure for social networking applications.

Other security issues like prevention of DDoS and Sybil attacks, enforcing cooperation and preventing free-riding or content pollution, and establishing trust are also of course long-standing issues in the community, but since they

---

<sup>6</sup> <http://groups.yahoo.com/>

have been in the spotlight for years now, we do not highlight these here. That of course does not mean that these are trivial, or even practically solved. However, in the social network context, some of these issues may actually get simpler to deal with [YKGF06].

For peer identities, one can take advantage of opportunistic networks and peer authentication by in-person contact, when friends meet in real life and exchange keys over their phones. For bootstrapping authentication, a central authority (trusted third party) seems hard to avoid.

**Robustness.** Against misbehavior: In a centralized system, one can turn to the provider in case of user misbehavior, there is usually a process defined for dealing with such complaints. In a decentralized system, there is no authority that can ban users for misbehavior or remove content. Robustness against free-riding: Without the monetary incentive offered by advertising, other incentives have to come in to make users shoulder the responsibilities for keeping up the infrastructure, providing storage and ensuring availability by staying online. Robustness and Trust: Once access to content is granted, it is difficult to revoke that right. When a user allows a friend to see a message, the friend can store the message and keep access to it even after a change of key. Trust has to be at least equal to assigned access rights, due to this difficulty.

**Limited Peers.** To take advantage of the decentralized nature of social networks, a mapping of physical social network to virtual and vice versa enables extensions to offering access via web browsers by phone applications and direct exchange of data in physical proximity. A major impedance to widespread adoption of a decentralized system for OSNs will be users' reluctance to install yet another software. Consequently, it will be essential to allow for two classes of users, a core network of users who run the decentralized infrastructure software as well as a web service front-end, and the other, who are essentially clients accessing this service. This of course throws open Pandora's box with lots of questions, including technological feasibility as well as game theoretic issues like incentives and fairness in such a two tier system. Another immediate benefit however of allowing such two-tier system is that users can then participate in the social network with resource constrained (e.g. mobile) devices, which they may use as an auxiliary, even when they contribute resources to the core of the system with their primary device.

**Locality.** Using direct exchange between devices, real-life social networks can be used to support the decentralized social networking application. In addition to such opportunistic networks between users, a distributed architecture also enables us to take advantage of geographic proximity and its correlation with local interests. For example, most access routers for home Internet access now come with USB slots where storage can be added or they already have unused storage on the device itself. These routers are typically always on and thus would provide some stability for availability of data of local interest. This local interest can arise from the locality of events but also from the locality of typical real-life social networks of friends and neighbors. How to best harness this locality remains to be seen.

## 2.1 Differences to other decentralized or P2P applications

While there are properties common to peer-to-peer approaches that we can take advantage of for decentralized social networks, in this section we focus on what makes the requirements for social networks different from other peer-to-peer services in order to point out where new solutions are needed.

Peer-to-peer storage has been done successfully for file-sharing. These results can, however not be directly applied to social networks: In P2P file-sharing, any copy of a music, video, media file, potentially present in high numbers, will do. These copies are usually not updated, although new versions or different content get added to the system. In social networks, information, such as the current status of a person, is updated often and it makes a difference which version gets downloaded, the value of outdated information is much less than that of timely information that enables users to react to content changes.

In most file-sharing systems, files are not encrypted. When they are [DFM00, CSHW00], mostly for reasons of plausible deniability, the keys to the files can be obtained. For peer-to-peer social networks privacy is even more important as it concerns personal information, so storing content unencrypted at other peers whom the user does not want to access personal information is not an option. Files should only be readable by peers that are specifically allowed to access them.

This access control has been also required for peer-to-peer collaborative work support (CSCW), albeit for smaller groups than the typical user base of online social networks. Work colleagues are added or removed from a working group at a lower rate than expected churn for social networks. For DOSN we need a way of dealing with dynamic relations, that is churn both in terms of online/offline behavior and of adding/removing friends and corresponding access rights.

This dynamic behavior coupled with a different distribution of interest add requirements to availability. In file-sharing, a file is potentially of interest to a large population of users, a Zipf-law distribution of file popularity and thus download availability has been observed. For social networks, this distribution is expected to be different and often limited to a number of friends that is not directly correlated with the network size. The information about a person is of interest to their social network, not typically the general public, there are different economies of scale and scope at play.

Who is interested and authorized to access in social networks also differs from peer-to-peer backup/storage systems, where there is typically one owner of data, so granting access rights and key management is much simplified. Peer-to-peer storage for file backup has been addressed by numerous systems such as Farsite [ABC<sup>+</sup>02] for individuals but these do not address the issues arising from social networks, nor utilize the opportunities of ingrained mutual trust.

For social networks, we need a large-scale peer-to-peer network with fine-grained access control for reading and writing, with changing files (versions), small number of interested peers compared to overall population, and enable a list of features including file-sharing, chat, news-feeds, public and private asynchronous messaging, search, notifications. This means that there are components

we can take from other peer-to-peer services, but need to modify and extend them as described above.

### 3 The Case for Decentralizing OSNs

Keeping user data centralized or even just distributed but connected allows the service providers of online social networks, third-party application providers and, in cases where there is no deliberate protection, indeed anyone to crawl the network and find out about content or at least about connectivity and access patterns. The information gathered can then be used for data mining, direct advertising, censorship, or other purposes. Moreover, a centralized depository or fully connected network is more susceptible to virus or malware spreading than mostly local social networks that can be partitioned.

An immediate advantage of a DOSN is rather straightforward: it is not centralized, not owned by a single entity. The central storage of user information and ownership by a company, along with commercial exploitation of this information e.g. for ad revenue, raises privacy concerns that could be better addressed by a decentralized approach, with encryption and appropriate key management.

Of course at this juncture it is legitimate to ask, why use a decentralized infrastructure for supporting social networks, when the good old client-server architecture works fine. One can give the traditional arguments that P2P scales well, since a growing user base naturally brings in more infrastructural resources. This definitely can be a good incentive for people with good ideas but little money to support and expand overnight if their popularity increases. Also, if popularity declines over time, there is less exposure. However, given the success of numerous upstart companies, which have managed to scale well to not just millions but even hundreds of millions of users, the traditional scalability argument alone does not justify the hassles of a decentralized infrastructure.

Privacy has become a major concern. Particularly privacy and protection from massive data-mining and “big-brotherly” treatment of the users by the social networking service providers. This is expected eventually to lead to a significant population of users, who while they would like to enjoy the benefits and fun of social networks, may also want to restrict access to their personal data not only from fellow users who happen to be strangers, but also from any provider or indeed the general public. This disaffected population is expected to be the early adopters of decentralized social networks with encryption.

Besides privacy and other related security concerns, a decentralized approach also enables content creators to execute greater control over their content, as well as avoid censorship either by the website owner, or censorship of the hosting website by a third party.

While some sites follow up with corrective measures because of users’ outcry, e.g. [Asp08], and one may also argue about legislative solutions to protect users’ privacy, there is no guarantee that in the future the users’ data will not be misused. The objective decentralized privacy-preserving OSNs is thus to aim for a system which makes it technologically harder (ideally, impossible) to violate

the users' privacy and large scale data mining, even while the users continue to enjoy the advantages of social networking.

Given the reality of privacy breaches by centralized online social network providers, as exemplified by Facebook's beacon application [Per07] among others [Gol07], there is a motivation for giving the control over data back to the users and not have one entity access to all personal data of the participants in the social network. With a peer-to-peer approach, decentralization is a given, and combined with appropriate encryption users can determine whom they allow access to their data.

Essentially, a decentralized approach seems promising to be the right technology to achieve both privacy and freedom of speech. For this reason, user-provided content and participatory media creation suit themselves better to a peer-to-peer rather than a client-server model. Another incentive for users to embrace such a model is to evade any constraints put by the service provider in the present or future (e.g. for the amount of storage space, or subscription fees, or service shut down).

Realizing an application layer Internet on top of diverse networking infrastructure, including the Internet, but also mobile - cellular as well as ad-hoc, and supporting Web 2.0 applications on top of such an application layer Internet, can also help making them ubiquitous.

By supporting the direct exchange of information between devices, be it between users that meet or between adjacent nodes of a city mesh network, a peer-to-peer infrastructure can take advantage of real social networks and geographic proximity. In contrast to a centralized web server, local connectivity already facilitates social networking without Internet access.

While access control by encryption for user data privacy would be possible in a centralized system, it does not go as far as a peer-to-peer approach in ensuring user control. First, whoever would be willing to provide the centralized service and infrastructure would also be able to cease to provide the service or change its terms. Second, due to the lack of data mining and advertising possibilities, there would be fewer incentives and means to provide a good service and all the servers necessary for a centralized solution. Third, a centralized service requires more trust by the users than a distributed system that limits the risk of privacy breaches by not providing a central repository of user data, so that only a small fraction of protected data may be exposed at any time, should the encryption be broken.

In addition to addressing the privacy aspects in general, there is an opportunity to support a non-commercialized self-organized service. Web-based centralized online social networks today bring together the social sphere of family and friends with the commercial sphere. This combination enables targeted advertising thanks to profile information and data mining and thus based on a person's revealed preferences and extends it to a more precise targeting by taking into account social information. We envision a peer-to-peer social network that separates these spheres and enables users to maintain their social network without commercial prompting by advertisement.

User control of data, as provided by a peer-to-peer and secured social network, has consequences beyond privacy and freedom from advertisement. One such consequence is that users can also exercise control over the content they create in terms of intellectual property. User control in this sense means control over who can access their content and what they are allowed to do with that content, e.g. access control can be combined by licensing models of the user's choice (e.g. creative commons licenses) allowing for flexible content rights, as opposed to the current practice of copyright for the online social network providers. Likewise, users also can enjoy freedom of speech, without fearing censorship or other obstacles (like a subscription fee), which a central service provider can impose at its whim.

Another aspect of control is how the social network can be accessed. Moving down the layers from application to network to physical access, there is another instance of peer-to-peer paradigm suitability that has been overlooked: decentralized access via various means (such as direct exchange, as in opportunistic networks) for ubiquitous social networks as opposed to those limited to the web.

Centralized web-based social networks do not match the inherent peer-to-peer nature of both social networks themselves and of participatory media creation. User-provided content and participatory media creation suit themselves better to a community-driven peer-to-peer rather than a client-server model. By mapping a peer-to-peer application to a peer-to-peer infrastructure, direct connections can be exploited such that locality can be taken into account. Peers can carry information for each other in a delay-tolerant fashion and use local access points for local information. We thus have a matching of the distributed nature of human social networks with a distributed service, and we can also match the service to a local environment and make it a ubiquitous service. By supporting the direct exchange of information between devices, be it between users that meet or between adjacent nodes of a city mesh network, a peer-to-peer infrastructure can take advantage of real social networks and geographic proximity. In contrast to a centralized web server, local connectivity already facilitates social networking without Internet access.

## 4 General purpose DOSNs

To give the readers a better overview of existing approaches towards the decentralization of online social network services, we propose a reference architecture of a general-purpose DOSN platform is given in Fig. 2. The other works will be reviewed based on the relation with this proposed architecture approach.

The reference architecture consists of six layers and provides an architectural abstraction of variety of current related approaches to decentralized social networking in the research literature. The lower layer of this architecture is the

physical communication network, which can be the Internet or a (mobile) ad hoc network (in case we consider a mobile online social network). The distributed or P2P overlay management provides core functionalities to manage resources in the supporting infrastructure of the system, which can be a distributed network of trusted servers or a P2P overlay. Specifically, this layer provides higher layers the capabilities of looking up resources, routing messages, and retrieving information reliably and effectively among nodes in the overlay.

On top of this overlay is the decentralized data management layer, which implements functionalities of a distributed or peer-to-peer information system to query, insert, and update various persistent objects to the systems.

The social networking layer implements all basic functionalities and features that are provided by contemporary centralized social networking services. Among these functionalities the most important ones are given in Fig. 2, namely the capability to search the system (Distributed search) for relevant information, the management of users and shared space (User account and share space management), the management of security and access control issues (Trust management, Access control and security), the coordination and management of social applications developed by third parties (Application management).

It is expected that the social networking layer exposes and implements an application programming interface (API) to support the development of new applications by freelance developers and other third-parties, as well as to enable the customization of the social network service to suit various preferences of the user. To enable better interoperability with available social network services, e.g., better portability of applications across OSN providers, this API should conform to existing API standards, e.g., OpenSocial<sup>7</sup>.

The top layer of the architecture includes the user interface to the system and various applications built on top of the development platform provided by the DOSN. The DOSN user is expected to provide the user the necessary transparency to use the DOSN as any other centralized OSN. Applications can be either implemented by the DOSN provider or developed by third-parties, and can be installed or removed from the system according to user's preferences.

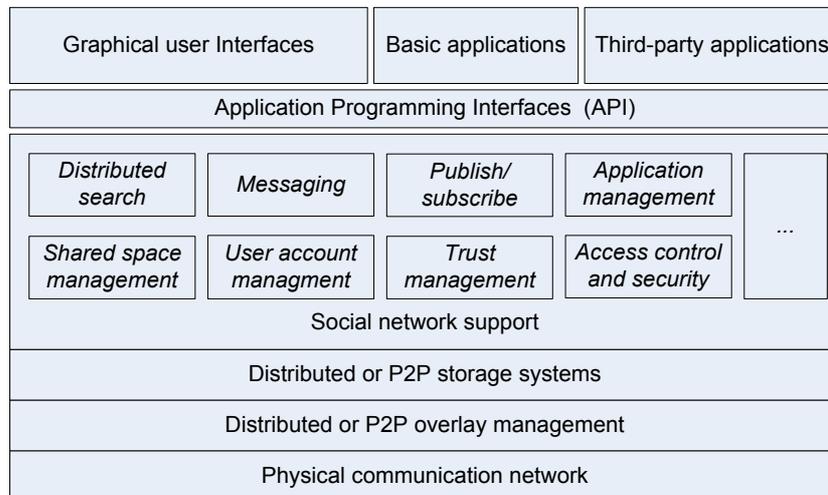
#### 4.1 Proposed DOSN approaches

This section introduces and explains several existing approaches to implement general-purpose decentralized online social networks. They range from academic proposals without any implementation to first systems that exist as demonstrators or are even in the course of public testing, as of the time of this writing.

**Safebook** Safebook [CMS09], an approach to provide a decentralized general purpose OSN, follows the main objective of protecting its users' privacy. It con-

---

<sup>7</sup> [www.opensocial.org](http://www.opensocial.org)



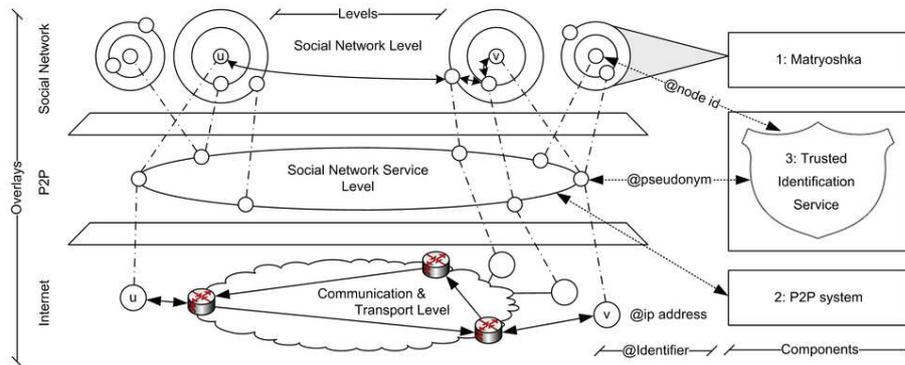
**Fig. 2.** The general architecture of a distributed online social network

siders adverse or erroneous behavior of a centralized service provider, possible adversaries which are misusing the functions of the social networking service, as well as external adversaries that could eavesdrop or modify data on the networking layer.

The main goals of offering the full set of services that centralized general purpose SNS usually implement, and of assuring the three security objectives of privacy, integrity, and availability, it is based on two simple assumptions: that decentralization and cooperation between friends will facilitate the implementation of a secure, and privacy preserving OSN. Considering the centralized storage to be a potential risk, safebook chooses a distributed implementation architecture. The social links between friends, family members, and acquaintances, which are represented as the core intrinsic knowledge of an OSN, are leveraged for multiple purposes. Requests from a user and for a user's profile are anonymized hop-by-hop on recursive routing paths traversing links of the OSN. Additionally, since friends are assumed to cooperate, they are leveraged to increase the availability of profiles.

Safebook consists of three major different components, the TIS, matryoshkas, and a peer-to-peer location substrate (see Fig. 3).

All users have to acquire certificates from a *Trusted Identity Service (TIS)* upon joining the OSN, which they are able to do by invitation, only. The TIS is a stateless, offline service. It does not store any information, but simply implements a cryptographic function to issue keys, identifiers, and certificates based on the identity of the requesting user. It currently is run at the institutions participating in the Safebook development (Institut Eurécom and TU Darmstadt), but may be further distributed to other trusted third parties. The TIS represents a powerful



**Fig. 3.** Main design components of safebook

entity in safebook, yet, it does not cause any threats, since it is only involved in the identification and certification of users, but does neither store nor retrieves or accesses any data of the users.

Safebook discerns between identified participation in the OSN, and anonymous participation to provide services to other users. In order to protect the identified participation, safebook introduces *matryoshkas*, which are specialized overlays encompassing each user. All contacts of a user represent the innermost shell of a user's matryoshka. Each user selected for a matryoshka in turn selects one of his contacts to be part of the next matryoshka shell, unless a predefined number of shells is reached. Current evaluation indicates that three to four shells represent a good trade-off between the resilience towards statistical identification attacks against the core, vs. the efficiency and performance of the system. The matryoshkas are used to anonymize requests, to hide the existence of the user in the center, and to increase the availability of the user's profile, by replicating it among the devices the matryoshka consists of.

For the purpose of locating other users' matryoshkas, safebook implements a peer-to-peer substrate. Considering response times of requests to be a main requirement, the developers have adapted Kademlia [MM02], an existing peer-to-peer system known to achieve fast lookup. Safebook implements the original routing structure and distance metric as Kademlia, simply switching for iterative to recursive routing and stochastically including access through contacts to protect the identity of the requesting user. The identifier used for the peer-to-peer substrate additionally is derived from the certificates, and hence determined by the TIS, which prevents denial of service attacks on the peer-to-peer overlay.

Safebook initially has been analyzed in formal models and large scale simulations. A first functional prototype to date is tested by the developers, which predict a first version for the public to be available by the end of 2010.

**FOAF** Yeung et al [mAYLL<sup>+</sup>09] presents another practical approach to decentralize management of current social networks. The framework enables users to export their FOAF<sup>8</sup> profiles, store them on dedicated trusted servers. Users query and manage these profiles through open Web-based protocols such as WebDAV<sup>9</sup> or SPARQL/Update<sup>10</sup>. A clear advantage of this approach is its compatibility to current social networking platforms and its entirely Web-based nature. The use of a set of trusted servers for storing user's data, however, raises some other security issues that necessitates further considerations. We believe this approach has a high potential of being adopted, as it is supported by the W3C consortiums.

Another related work is Social VPNs [FBSJW08]. Users may query different social networks to discover friends to build a Virtual Private Network (VPN) with them. The prototype integrates with Facebook, use the IP2IP virtual networks and IPsec security infrastructure to build a VPN platform for a number of interactive applications such as instant messaging, file sharing, etc.

NEPOMUK<sup>11</sup> is an on-going EU project with close relation to DOSN. The goal of NEPOMUK is to develop a middleware for sharing users' desktops with friends for online collaborations and sharing of knowledge by exploiting Semantic Web technologies.

**LifeSocial** Considering the immense increase of users that online social networks have experienced in the recent past, and which are expected for the future, too, LifeSocial [GPM<sup>+</sup>08] primarily aims at keeping social networking services scalable by distributing the load to their users' resources.

The main functional components of OSNs are data storage and interaction. Both are classic domains of peer-to-peer systems, and have very successfully been implemented as file sharing<sup>12</sup> and instant messaging (jabber), or telephony (skype) applications. With current social network providers being central entities, who have to provide the entire resources for storing the data uploaded to the OSN and for making it accessible, they are soon to become a bottle neck when the number of users increases further. LifeSocial hence proposes to distribute the service provision in a peer-to-peer fashion in order to mitigate the resource problem and to balance the service load to the resources at the users' devices.

LifeSocial is designed with the main premise to leverage on existing and proven components and to create a modular plugin-architecture to assure extensibility. It consequently is assembled using FreePastry<sup>13</sup>, a structured peer-

---

<sup>8</sup> <http://www.foaf-project.org/>

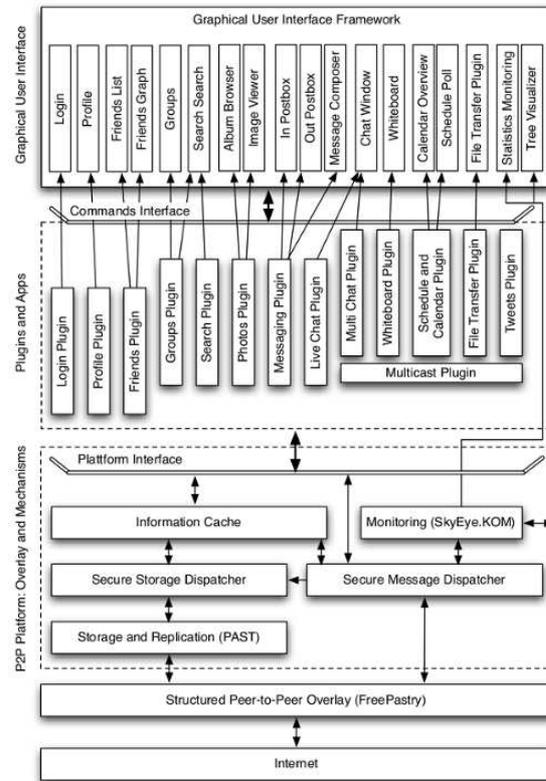
<sup>9</sup> <http://www.webdav.org/>

<sup>10</sup> <http://jena.hpl.hp.com/~afs/SPARQL-Update.html>

<sup>11</sup> <http://www.nepomuk.org>

<sup>12</sup> Gnutella [cli02], Kazaa [LCP<sup>+</sup>05], and Bittorrent [Coh03] are only some of the more prominent examples.

<sup>13</sup> <http://www.freepastry.org/>



**Fig. 4.** LifeSocial Plugin Architecture

to-peer overlay for data storage, and PAST<sup>14</sup> to achieve reliable replication of the data. LifeSocial implements its own access control scheme on top of these components. Some plugins are mandatory to implement a general purpose OSN: The whole system demands plugins for profile management, friend management, group management and photo albums as a minimum set. Additional plugins, such as a whiteboard and a chat system have been proposed.

The different components of LifeSocial, and the overall system, have been evaluated in simulation studies. It additionally is one of the few systems of which a prototype exists at the time of this writing (see Fig.: 4). The prototype consists of all mandatory plugins, as well as basic white-board and chat functions. It has been presented in different conferences and exhibitions and the system currently is tested between the group of its developers.

<sup>14</sup> <http://www.freepastry.org/PAST/default.htm>

**PeerSoN** PeerSoN<sup>15</sup> [BD09a, BSVD09] aims at keeping the features of OSNs but overcoming two limitations: privacy issues and the requirement of Internet connectivity for all transactions. To address the privacy problem, it uses encryption and access control coupled with a peer-to-peer approach to replace the centralized authority of classical OSNs. These two measures prevent privacy violation attempts by users, providers, or advertisers. Extending the decentralized approach, PeerSoN also enables direct exchange of data between devices to allow for opportunistic, delay-tolerant networking. This includes making use of ubiquitous storage to enable local services.

The main properties of PeerSoN are encryption, decentralization, and direct data exchange. In a nutshell, encryption provides privacy for the users, and decentralization based on the use of a P2P infrastructure provides independence from OSN providers. Decentralization makes it easier to integrate direct data exchange between users' devices into the system. Direct exchange allows users to use the system without constant Internet connectivity, leveraging real-life social networking and locality.

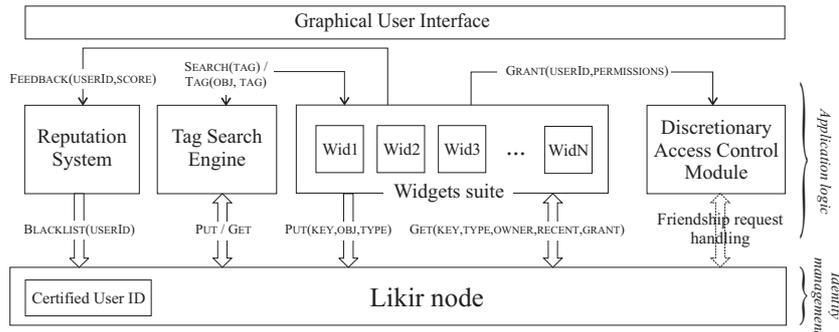
The current PeerSoN implementation replicates the following features of OSNs. In the category of social links, users (peers) can become friends and thus establish a social link between each other. Digital personal spaces are provided in that users can maintain their own profile and a wall, a space for items posted by themselves or their friends. Communications between users are directly peer-to-peer when both are online, and the implementation supports asynchronous messaging when this is not the case. PeerSoN uses a DHT as a lookup system and then lets peers connect directly; all data is encrypted and keys for accessing an object are encrypted for the exclusive use of authorized users and stored in a separate file associated with a particular object, such as a user's profile. The prototype implementation has been tested on PlanetLab and uses OpenDHT for the lookup service. Using OpenDHT facilitated the PeerSoN deployment on PlanetLab but will be replaced for the next iteration of the implementation.

**Likir** Likir [AMRS08] is a Kademlia-based DHT that is aimed to protect the overlay against attacks common to these systems, by embedding a strong identity notion at overlay level. This increases the reliability of the overlay and offers many identity-based services that well suit social applications [AR10]. Likir targets at the main goals of avoiding a centralized storage of personal information, offering reliability, and integrating the identities of users deep into the system.

The main motivation of Likir is to avoid a central data repository, for both the reason to avoid a single point of failure and aggregation of user data. The rationale is to use a reliable, identity-based DHT layer, to provide the main services. The applications built on top of this substrate consequently are relieved from identity management, as it is already provided by the underlying DHT (see Fig. 5). Likir is designed to achieve full confidentiality of data as well as anonymity of the users. It additionally provides access control in a granularity

---

<sup>15</sup> <http://www.peerson.net>



**Fig. 5.** Likir Architecture

to applications: “authorized disclosure” ensures that malicious applications installed at some individual’s device is unable to access data disclosed to other applications.

Likir builds its core properties on introducing cryptography in a plain DHT-based approach. All nodes are furnished with an identifier in the form of an OpenId by a certification service. Subsequent communication events consequently are encrypted and authenticated by both communicating parties. A supplemental access control scheme is integrated that requires all service providing nodes to check grants that are appended to each request before returning any data.

In addition to an analytical study on the bandwidth and computational overhead consumed by Likir, the authors have run large scale emulations on Planet-Lab. A prototype of the Likir middleware has been published and is available for download, including a simple chat application for the purpose of demonstrations.

## 5 Specialized Application Centric DOSNs

Besides decentralized counterparts of general purpose online social networking applications as described above, there are also initiatives to realize decentralized counterparts of niche applications such as video sharing, social bookmarks and libraries, and micro-blogging to name a few.

### 5.1 Social-based P2P File Sharing

Among works in this category, the most well-developed initiative is Tribler [PGW<sup>+</sup>08b], which is basically a P2P content sharing system that leverages the existing social relationships and taste similarities among its users for fast discovery and recommendation of digital contents.

Initiated from the EU project P2P-Next<sup>16</sup> whose goal is to provide end-to-end professional streaming content based on P2P technologies and low-cost commodity hardware devices, i.e., user computers, Tribler has been extensively developed and provides a rich set of functionalities for its users, e.g., discovery and recommendation of friends with similar preferences, as well as providing fast and low overhead delivering video-on-demand such as television programs. The system, implemented as an social-based extension of the BitTorrent engine, is freely available<sup>17</sup> and has been evaluated at small and medium scales. The most recently public trial of Tribler, reported in [MBP<sup>+</sup>09] in mid July 2008, which lasted in 9 days, consists a global deployment of nearly 5000 unique peers for live streaming a free video. The system has shown to yield a relative good performance, e.g., the pre-buffering time before playing back the video is from 3 to 10 times shorter than that of the two existing deployed P2P live stream systems reported in [XKL07, ZLLsPY05].

To facilitate the social group formation between users, Tribler uses a de-anonymized approach to manage the peer identities. During the registration phase, each participating user is given by email a secure, unique, permanent identifier (*PermID*). PermIDs are obtained via a public key generation scheme with challenge-response mechanism to prevent spoofing. Existing contacts of a user (a peer with a PermID) can also be imported from other social networks such as MSN and GMail. For bootstrapping the networks, any newly joined peer can contact one in a list of pre-known superpeers, from which to obtain a list of other peers already available in the systems. The bootstrapping of the networks is done via an epidemic protocol [WJF<sup>+</sup>08].

A peer in Tribler uses many types of caches to store locally any contextual information relevant to its interests and tasted. These so-called Megacaches (each less than 10MB) of a peer may store various information related to its friend lists, the altruism levels, the preferences of its friends, and meta-data of the files and contents posted in the network. With existing BitTorrent networks, the number of files injected per day is sufficiently small such that the caches of file meta-data can be fully replicated among all peers. Thus there is no need to perform network-wide searching for interested content, but only the browsing of the local meta-data cache.

Peers in Tribler are clustered into many groups, each of which contains those peers with similar preferences and interests, or taste buddies group. The interest commonality between two peers are given by the similarity in their preferences of in the same or related content. Fore example, each peer's preference can be defined as its zapping behavior profile, which is the percentage of length the online television program actually watched by the peer, compared to the length of the complete program on air. The formation of such taste buddies group is done via the epidemic protocol BuddyCast [WJF<sup>+</sup>08]. The BuddyCast protocol requires a peer to periodical connect to either an existing friend (the exploitation phase) or a randomly peer (the exploration) and exchange a BuddyCast message

---

<sup>16</sup> <http://www.p2p-next.org>

<sup>17</sup> <http://tribler.org/>

with the selected peer. The BuddyCast message usually consists of the number of taste buddies with top-10 preference lists, e.g., the type of TV programs the peer is interested in, a number of random peers, and the top-50 preference of the sending peer. The exploration-to-exploitation ratio can be adapted to limit the randomness of the exploration. To ensure that a BuddyCast message has a high probability to be replied, the random peer in an exploration phase is selected based on its freshness to increase the chance it is still online, i.e., newly joined peers are more likely to be selected.

The exchange of a BuddyCast message enables the two contacting peers to discover whether they have similar preferences. Overtime the social network of peers would also be clustered into different groups of users with related interests. Thus the BuddyCast protocol in fact implements a decentralized collaborative filtering technique to recommend programs and potential friends of interests to a user. This will also facilitate the content delivery to end-users, since the downloading of contents relies on the collaboration among users in such social groups.

The download of contents from the system is according to a collaborative downloading protocol 2Fast [GIEvS06], in which a peer asking its friends to help it in downloading. The assumption is that peers would behave altruistically in favor of their friends, thus a peer will be willing to help uploading the content which it is not interested in. Peers that are not in a social group would follow an uncooperative downloading scheme, which is the default Tit-for-Tat strategy in BitTorrent. The 2Fast protocol has been deployed for testing in a real environment and gives promising results for small-scale experiments. With around 1900 peers, 6% seeds to download a 1.2GB file, the download time decreased by a factor from 2 to almost 6 for different types of connections from the downloading peers. Larger scale experiments have not been carried out with more realistic workloads from existing BitTorrent networks or available video streaming systems, however to evaluate the overall system performance more thoroughly.

Recently, a light weight message exchange protocol using a reputation metric, BarterCast [MPES09], has also been proposed to prevent the free-riding problem in the system. Consider the social networks of peers as a graph and define the capacity of an edge from a peer  $u$  to a peer  $v$  as the total number of bytes (of data content)  $u$  has uploaded to  $v$  in the past. The subjective reputation  $R_i(j)$  of a peer  $j$ , as evaluated by a peer  $i$  is proportional to the difference in the max-flow of bytes from  $i$  to  $j$  with the max-flow of bytes in the opposite direction. Therefore, this  $R_i(j)$  represents the service that  $j$  has provided to other peers in the system from the viewpoint of  $i$ . A peer is classified by another peer as a sharer, neutral, or free-rider peer depending on whether its subjective reputation falls below a certain threshold. Given the reputation, a peer may rank or ban another peer from downloading certain pieces of content from it, and thus free-riding incentives are reduced or eliminated.

## 5.2 Shared bookmarks and collaborative search

Diki<sup>18</sup> is a social bookmarking service that allows users to encrypt and share their bookmarks with trusted friends via the Extensible Messaging and Presence Protocol (XMPP), a real-time Web-based communication protocol for Internet services<sup>19</sup>. User privacy is preserved via three design principles: to enable data exchange only between trusted friends, not storing any data centrally, and any stored data is encrypted.

User data is encrypted using the PGP private key encryption scheme, stored locally, and exchanged to trusted friends using the XMPP protocol. A user may establish his own XMPP server or use existing ones to communicate with the others.

Since the XMPP servers only handle the exchange of encrypted data between users, without knowing the content of the exchanges, the system ensures that the information, e.g., URLs a user shares with his or her friends is kept secret. In short, Diki provides a decentralized, privacy-preserving version of the highly popular social bookmarking service Delicious<sup>20</sup>.

Using such an XMPP protocol, it is completely possible to implement the decentralized, privacy-preserving, and secured versions of other existing social applications such as Twitter.

Shared bookmarks and other meta-information can in turn be used for implicit or explicit collaborative online search. Some recent initiatives such as Gossple<sup>21</sup> and COBS<sup>22</sup> pursue such an approach of leveraging peer-to-peer/hybrid infrastructure based social networks and interest communities to facilitate collaborative search.

## 5.3 Micro-blogging

FETHR [SW09] is a light-weight protocol enabling users to use any existing microblogging service to communicate with other users on top of HTTP with near real-time guarantee. Users who follow the FETHR protocol can subscribe to each other's updates (tweaks) and receive these tweaks in real-time. Published entries are signed digitally by the publisher and linked together in a chain in reverse chronological order. Each entry in the list contains the hash of the previous entry for detecting any tampering in the content of the tweak. A FETHR publisher also includes the hashes of prior tweaks from his or her friends, thus integrating the timelines of many participants into a directed acyclic graph of tweaks spanning the whole network. This provides a provable order of events happening, as well as enabling the reconstruction of conversation threads.

According to the FETHR protocol, the publishers (the micro-blogger) push the entire (assumedly small) contents of the messages (tweaks) to the subscribers.

---

<sup>18</sup> <http://www.pace-project.org/>

<sup>19</sup> <http://xmpp.org>

<sup>20</sup> <http://delicious.com/>

<sup>21</sup> [www.gossple.fr/](http://www.gossple.fr/)

<sup>22</sup> <http://code.google.com/p/socialcobs/>

This is different from Twitter: in Twitter the followers are simply notified of the new tweaks, whose contents are still stored on the servers.

The current implemented prototype, BirdFeeder<sup>23</sup>, can be considered as a distributed alternative to the existing so-called microblogging platform Twitter<sup>24</sup>. It aims to target the weaknesses of Twitter, whose centralized architecture is shown to be the main cause to its performance bottleneck and single point of failure.

FETHR is designed as an entirely new protocol to be applied on existing microblogging services, thus its success much dependent on the widely adoption of the service provider communities. Also, the implemented prototype is still in early phase, and there is little study and analysis on the security and efficiency of FETHR.

## 6 Social distributed systems

Besides the initiatives to emulate specifically online social networks, as described in the previous section, there is also a growing trend to develop *social distributed systems* (SocDS) which provide equivalent functionalities to non-social (and sometimes, also centralized) systems. Such initiatives try to leverage and translate the social trust into properties leading to system reliability. This trend of coupling real life social networks with distributed systems is gaining traction in recent times. For example, distributed hash tables realized using exclusively social links can provide robustness against Sybil attack. Likewise, P2P storage systems relying on social trust to do data backup provide resilience against free-riding address limitations of computational trust models and stand-alone algorithmic solutions. Such social distributed systems can be used for carrying out any task the non-social counterparts are typically used for, but additionally, and naturally, SocDS are ideal to be used as substrates and building blocks for distributed/decentralized online social networks.

Essentially, one can argue that with the use of SocDS, we have a two way symbiotic design. Distributed/peer-to-peer infrastructure for OSNs provide desirable properties and addressing the shortcomings and constraints of traditional OSNs which use centralized resources provided by OSN service providers. On the other hand leveraging on social networks to design robust distributed systems provide new systems design opportunities which are beyond the scope of traditional algorithmic solutions. Such distributed systems can be used for different applications, but are of-course naturally suitable to form the underlying substrates for decentralized online social networks.

We next discuss social distributed systems design for some basic building blocks such as indexing and routing using distributed hash tables (DHTs) and p2p storage & back-up systems. For each of these building blocks, there are non-social counterparts which have been studied in traditional systems research,

---

<sup>23</sup> <http://brdfdr.com/>

<sup>24</sup> <http://twitter.com/>

which we will point out whenever necessary. The benefits (as well as possible drawbacks) of the “social” counterparts will also be briefly explained.

## 6.1 Social DHT: SocialCircle

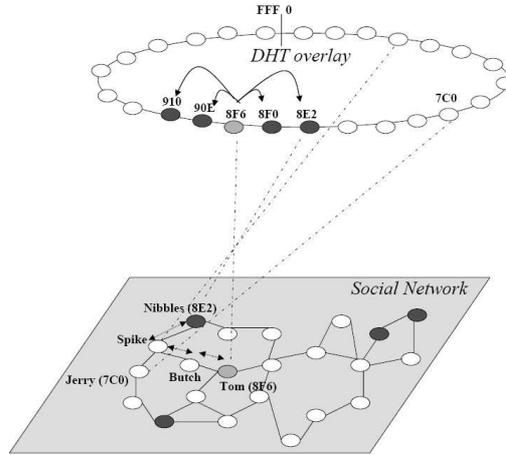
Structured overlays, e.g., Distributed Hash Tables (DHTs) provide essential indexing and resource discovering in distributed information systems. Typically, structured overlays are based on enhanced rings, meshes, hypercubes, etc., leveraging on the topological properties of such geometric structures. The ring topology is arguably the simplest and most popular structure used in various overlays. In a ring based overlay network like Chord [SMLN<sup>+</sup>03] nodes are assigned to distinct points over a circular key-space, and the ring invariant is said to hold if each node correctly knows the currently online node which succeeds it (and the one which precedes it) in the ring. The ring is both a blessing and a curse.

On the one hand, an intact ring is sufficient to guarantee correct routing. Hence, historically, all existing structured overlays over circular key space have considered it necessary de facto. Previous attempts have used social network links to bolster DHTs, e.g., Sprout [MGGM04], preferring social links whenever possible, but nevertheless requiring links to random/unknown nodes also. Such an approach still relies on using the untrusted links most of the time, but was arguably as good as it could get under the older paradigm of DHT designs, where a completely connected underlying graph and ring invariance were considered necessary.

In the recent years several radical DHT designs have been proposed, for example VRR [CCN<sup>+</sup>06] proposed for ad-hoc environments and Fuzzynet [GGD<sup>+</sup>08] designed specifically to avoid ring maintenance. Neither of these two rely on sanctity of a ring or fully connected underlying graph. SocialCircle [ZD09] adapts and hybridizes ideas from these two DHTs. In the description of SocialCircle below we also point out which of the features are derived from which of VRR or Fuzzynet respectively.

*Virtual ring routing* (VRR) is a DHT style overlay layer approach used to define the underlying network’s routing mechanism. It is implemented directly on top of the link layer and provides both traditional point-to-point network routing and DHT routing to the node responsible for a hashed key, without either flooding the network or using location dependent addresses. While traditional DHTs take for granted point-to-point communication between any pair of participating nodes, VRR extends the idea, using only link layer connectivity. Essentially this means that the VRR scheme relaxes the traditional DHT assumption of a completely connected underlying graph. Each node in VRR has an unique address and location independent fixed identifier, organized in a virtual ring, emulating Chord style network. Each node keeps a list of  $r/2$  closest clockwise and counter-clockwise neighbors for the node on the virtual ring. Such a set of neighbors is called the node’s virtual neighbor set (*vset*).

Typically, members in a node’s *vset* won’t be directly accessible to it through the link layer. Thus each node also maintains a second set called the physical neighbor set (*pset*), comprising nodes physically reachable to it through the link



**Fig. 6.** Sybil attack resistant *SocialCircle* DHT exploiting social connections. This example of DHT over Tom & Jerry’s social graph is adapted from the virtual ring figure in [CCN<sup>+</sup>06] for routing in ad-hoc networks.

layer. In *SocialCircle*, this idea is exploited by replacing VRR’s *pset* with the set of friends a node has - its social set *sset*.

Thus, instead of exploiting the physical layer connectivity as VRR does, *SocialCircle* builds the overlay over the *social plane* exploiting people’s social connections. In figure 6 the lower plane shows the social graph, while the upper plane shows the *SocialCircle* DHT. Adaption of VRR to exploit social links rather than physical neighbors provides a good abstraction, enabling *SocialCircle* to realize a Sybil attack [Dou02] resistant DHT, where end-to-end routing can be achieved following a web or trust of friends-of-friends.

Finally, each peer maintains a routing table, which comprises of routes to its *vset* neighbors using its *sset*. These routes can be established and maintained using different strategies typically inspired by mobile ad-hoc routing protocols. Like in VRR, nodes in *SocialCircle* also keep track of the routes that pass through them. The advantage of using the DHT abstraction to do the routing over social graph is same as the use of DHTs instead of using flooding based search in a typical peer-to-peer system. The DHT abstraction ensures efficiency and certainty of routing to the appropriate target.

Thus, in the example from figure 6, *Tom* with logical identifier *8F6* on the *SocialCircle* has *8F0*, *8E2*, *90E* and *910* in its *vset*. *Spike* has *Jerry*, *Nibbles* and *Butch* in its *sset* since they are his direct social connections.

*Tom* needs to maintain routes to all its *vset* nodes, and thus, for *8E2*, he will have a route through his *sset* entry *Butch*, who will route through his *sset* entry *Spike*.

So when *Tom* needs to route a message to *7C0*, then it will try to forward the message closest to the target on the SocialCircle, which happens to be *8E2*. While the message is being routed to *8E2* following the *sset* nodes at each peer, *Spike* will observe that the ultimate destination is *7C0*, for which it may already have a route passing through it, and will thus forward the message to *Jerry*, instead of sending it to the intermediate destination *Nibbles*. *Jerry* processes the routing request, and forwards it to the final destination *Quacker*, who happens to have the identifier *7C0* on SocialCircle.

VRR works in an opportunistic manner where the route is forwarded along the virtual ring, but discovers shortcuts, so that the search is still efficient. SocialCircle preserves the same benefits by routing over the social links. Each hop on the social link involves IP level routing, which may need several hops, just like any logical overlay hop of traditional DHTs.

While the routing in SocialCircle follows the ideas from VRR, it uses Fuzzynet's data-management ideas [GGD<sup>+</sup>08] for storing and retrieving key-value pairs in SocialCircle. The details of SocialCircle's data-management are not relevant here.

Given the exclusive use of social links to realize a DHT, SocialCircle is naturally resilient against Sybil attacks and also provides a means to more effectively use trust mechanisms to deal with free-riding and other anti-social/non-cooperative behaviors. It is also a natural primitive to be used in DOSNs, and specifically has been used to realize a directory service facilitating search for friends which is a common functionality required in online social networks.

Robustness against churn, scalability of the SocialCircle overlay with respect to the number of peers, and performance under partial knowledge of social networks (i.e., when an individual is yet to add all its friends) are unaddressed issues which will need further investigation to determine SocialCircle's usability in large-scale real life deployments.

## 6.2 Storage/back-up

A p2p storage (or back-up) system uses the storage space of its participants to increase the availability or the survivability of the data. Coupled with mechanisms for content sharing and privacy, a distributed storage system is a building block for a DOSN. For instance, it can ensure that, when a user is off-line, her profile is available through the replicas. At the same time, real-world relations between people can be used as a basis for replication agreements. As in any other p2p system, one of the key issues in a p2p storage system is to provide incentives for peers to act fairly: not to consume disproportional amount of other peers' bandwidth or storage and to provide stored data on a request. Basing replication agreements on real friendships can mitigate the need for more explicit incentives: in general, a friend is less likely to free-ride on our resources than a stranger (or, even if a friend free-rides, we are more likely to forgive her and the system).

Many distributed storage systems have the option of sharing stored objects among a group of users. OceanStore [KBC<sup>+</sup>00] is one of the first distributed storage infrastructures, focusing on storing objects. As all the objects are encrypted,

read sharing (i.e., many agents accessing an object) is accomplished through sharing of the encryption key. OceanStore also permits multiple agents to modify the same object through Access Control Lists (ACL), which are OceanStore object themselves (a write on an object is authenticated by a trusted server against an ACL associated with the object).

Wuala<sup>25</sup> is an on-line storage system that combines distributed and centralized elements. Data is stored both on users' machines and on Wuala's servers; the algorithm associating data with particular machines is most probably centralized. Wuala can act as a crude DOSN as it enables its users to share files in a group of "friends"; however other OSN features, such as profile information, are missing.

BlockParty [LD06] and FriendStore [TCL08] are p2p backup system in which data is stored only on designated peers corresponding to real-world "friendships" between users. In fact, the "friendship" relation acts as a proxy for a simple reputation system: it verifies user identities and provides off-system discouragement for free-riding. Moreover, [LD06] argues that friendship-based system will have less permanent node departure; and thus the data survivability will depend on (rare) hardware failures rather than permanent departures frequent in usual p2p systems. Friendship-based systems can be thus more forgiving to transient errors [TCL08] instead of assuming a permanent departure and, consequently, large-scale data transmissions, longer delays can be used.

The main disadvantage of friendship-based p2p backup systems is that, in general, systems that constrain choice of replicating nodes have lower survivability (see e.g., [GMP09] for analysis with locality constraints). In a friendship-based system, achieved data survivability strongly depends on survivability of machines of friends—and friendships are fostered rather based on the person's character, and not her computer's up-time, which is the important factor for the system's reliability.

Approaches mixing traditional algorithmic solutions, for example, uptime history based choice of peers in a storage system with real life social trust to build a robust system are still in their nascence, and with great potentials and research opportunities.

## 7 Delay-Tolerant DOSN

Once the functionality is distributed, social networks are no longer dependent on Internet connectivity for every transaction – in contrast to current web-based services. We therefore have the opportunity to take into account locality, both in terms of connectivity by direct exchange between devices, and in terms of content, such as local community interests and events. This way, social networking applications can benefit from local storage, connectivity, and delay-tolerant data transfer via social encounters. The local communities, in turn, can benefit from the social networking applications enabled by such a system, e.g., by finding neighbors with similar interests.

---

<sup>25</sup> <http://www.wuala.com>

Current online social networking services require the user to be connected to the Internet for every interaction, not only for real-time information but also for older information such as data posted by the user or her friends in the past. Since online social networks are part of the so-called Web 2.0, they run on dedicated web servers.

All information in the online social network is thus stored on logically central servers, even though they may be replicated or cached in different geographic regions using content distribution services. Due to such centralization, there is no distinction between information of global or exclusively local relevance.

We propose to implement online social network functionality in a distributed, delay-tolerant way. Intermittent Internet connectivity can be used to connect with the wider user community, while users can exchange data among each other in direct physical proximity during offline times. The need for constant Internet connectivity, which can be costly, is thus eliminated. When information is of local relevance only, it need not be transferred to a central server that is potentially far away. These needless long-distance transfers can be replaced by local storage. In addition, it becomes easier to take locality into account logically when keeping local information also local physically.

While portable user devices, such as phones, laptops, and personal digital assistants (PDAs) can be used to exchange data directly, also fixed devices can contribute resources. Schioeberg [Sch08] proposed to use storage on home routers, such as ADSL modems with WLAN capabilities to support peer-to-peer social networks. Many home routers now have unused storage or can at least be extended by USB sticks or external hard drives. Fixed devices that typically are switched on irrespective of user activity not only contribute resources but also increase availability and robustness of a system for delay-tolerant social networks.

Such delay-tolerant, local social networks allow us to build on other proposals and new opportunities. For example, Antoniadis et al. [ALGS<sup>+</sup>08] proposed to use local wireless networks to enhance communities such as neighborhoods in towns. Collectively, users would build wireless neighborhood networks by pooling their resources to support the creation and operation of the underlying communication network. They envision user participation and cooperation at several layers, physical, access, network, and application layers. They argue that *the design of communities suitable for this environment will encourage users to participate, enable trustworthy network creation, and provide a social layer, which can be exploited in order to design cross-layer incentive mechanisms that will further encourage users to share their resources and cooperate at lower layers*. The goal is to bridge the gap between online and offline communities.

The way we envision delay-tolerant social networks can be a vehicle to such fostering of communities. Beyond the features of current social networks that allow users to keep in touch and up-to-date with the friends they already have and, increasingly, the new ones they found thanks to the service itself, delay-tolerant social networks would allow users to benefit from locality. They could find others who live nearby and have similar interests, find or start events in

the neighborhood, organize or collaborate for creative or political collective action, found local marketplaces of ideas, goods, or services, edit local information repositories or wikis, to name just a few possibilities.

Local social networks could also be established to never connect to a wider collection of networks but form islands of social networks, effectively making censorship or data mining prohibitively difficult.

The possibilities of use of delay-tolerant social networks are of course not limited to the examples given above, once the technology is available, users may come up with novel and original applications, as has been the case with online social networks or indeed the advent of the Internet and the World-Wide Web itself. Delay-tolerant social networks can thus be seen as enablers for applications or uses not yet foreseen.

Taking a wider perspective, we contend that there is a feedback loop between society and technology, and there are interesting dynamics in both directions, raising questions such as the following. How can we develop and use technology to enhance people's lives and society as a whole and how can we take societal phenomena and changes into account to improve technology? Delay-tolerant social networks can serve as an example to allow us to explore these questions directly. First, by experimenting how local user communities can benefit from social networks that do not require Internet connectivity. Second, by analysing how user behavior, such as mobility and use of ubiquitous computing resources, can support distributed social networks.

## 8 Conclusion

Most of the early commercial initiatives for DOSN do not have a large user base due to the dominance of their existing centralized social application counterparts that offer equivalent functionalities. The major obstacles to the wide adoption of these decentralized social applications are their immaturity in features and the acceptance of existing users. Data portability issues also hinder the popularity of the new systems, even if these new ones offer much more security and privacy-preserving features. Another reason is the network effect problem: users of an existing social networking service do not want to switch to another one without their friends doing so, since maintaining these connections is important for them. Therein a chick-and-egg problem emerges: a newly developed decentralized social system is less appealing to new users due to its lack of benefits, while the system itself relies on a certain critical mass of participants before it can offer its users any significant values. While this is true also for centralized OSNs, the problem is exacerbated when involving a change away from web-based services. Additionally, various performance issues, mostly related to the availability, latency, and throughput in data access due to data encryption and replication, of these decentralized social applications have still yet to be investigated carefully to compare with their existing centralized approaches.

Despite the above challenges, we believe that development and research on DOSNs still are very important and have significant impact. Due to scalability

issues, major social networking providers may want to switch to use a decentralized infrastructure for their services. Furthermore, such decentralized alternatives are also less costly compared to centralized architectures. In this chapter, we made a case for using a decentralized infrastructure for social networks to address problems other than purely technical ones stemming from a centralized service-provider owned approach, such as privacy and access limitations. We listed a variety of research challenges and opportunities that result from decentralization, e.g. security issues to enable user control of data, storage, topology, search, and update management. Research into the area of DOSN has become very active, and we discussed several recent approaches in this paper, in terms of general-purpose DOSN, DOSNs with a focus on a specialized application, and the emerging trend of social applications. Once decentralized, the functionality of OSNs can be expanded by allowing for direct data exchange between devices, enabling ad-hoc social communities, data locality, and delay-tolerant social networks.

## References

- [ABC<sup>+</sup>02] A. Adya, W. Bolosky, M. Castro, R. Chaiken, G. Cermak, J. Douceur, J. Howell, J. Lorch, M. Theimer, and R. Wattenhofer. Farsite: Federated, available, and reliable storage for an incompletely trusted environment, 2002.
- [ADH04] Karl Aberer, Anwitaman Datta, and Manfred Hauswirth. Efficient, self-contained handling of identity in peer-to-peer systems. *IEEE Transactions on Knowledge and Data Engineering*, 16(7), 2004.
- [ALGS<sup>+</sup>08] P. Antoniadis, B. Le Grand, L. Satsiou, A. and Tassioulas, R.L. Aguiar, J.P. Barraca, and S. Sargento. Community building over neighborhood wireless mesh networks. *IEEE Technology and Society Magazine*, 27:48–56, March 2008.
- [AMRS08] Luca Maria Aiello, Marco Milanesio, Giancarlo Ruffo, and Rossano Schifarella. Tempering Kademlia with a robust identity based system. In *P2P '08: Proceedings of the 2008 Eighth International Conference on Peer-to-Peer Computing*, pages 30–39, 2008.
- [AR10] Luca Maria Aiello and Giancarlo Ruffo. Secure and Flexible Framework for Decentralized Social Network Services. In *SESOC 2010: IEEE International Workshop on SECurity and SOCial Networking*, 2010.
- [Asp08] Maria Aspan. Quitting Facebook Gets Easier, Feb. 2008. <http://www.nytimes.com/2008/02/13/technology/13face.html>.
- [BD09a] Sonja Buchegger and Anwitaman Datta. A case for P2P infrastructure for social networks - opportunities and challenges. In *Proceedings of WONS 2009, The Sixth International Conference on Wireless On-demand Network Systems and Services*, Snowbird, Utah, USA, February 2-4, 2009.
- [BD09b] Sonja Buchegger and Anwitaman Datta. A case for P2P infrastructure for social networks - opportunities and challenges. In *WONS 2009, 6th International Conference on Wireless On-demand Network Systems and Services, Snowbird, Utah, USA*, February 2009.
- [BE07] Danah Boyd and Nicole B. Ellison. Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1-2), November 2007.

- [BSVD09] 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*, Nürnberg, Germany, March 31, 2009.
- [CCN<sup>+</sup>06] M. Caesar, M. Castro, E.B. Nightingale, G. O’Shea, and A. Rowstron. Virtual ring routing: network routing inspired by dhts. In *SIGCOMM, Proceedings*, 2006.
- [cli02] clip2. The gnutella protocol specification v0.4. <http://rfc-gnutella.sourceforge.net/>, letzter Abruf: 19.04.2007, 2002.
- [CMS09] Leucio-Antonio Cutillo, Refik Molva, and Thorsten Strufe. Safebook: a Privacy Preserving Online Social Network Leveraging on Real-Life Trust. *IEEE Communications Magazine*, 47(12):94 – 101, December 2009.
- [Coh03] Bram Cohen. Incentives build robustness in bitorrent. In *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*, 2003.
- [CSWH00] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, pages 46–66, July 2000.
- [DFM00] Roger Dingledine, Michael J. Freedman, and David Molnar. The free haven project: Distributed anonymous storage service. In H. Federath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*. Springer-Verlag, LNCS 2009, July 2000.
- [Dou02] J.R. Douceur. The sybil attack. In *Peer-To-Peer Systems: First International Workshop, IPTPS, Revised Papers*. Springer, 2002.
- [FBSJW08] Renato J. Figueiredo, Oscar P. Boykin, Pierre St. Juste, and David Wolinsky. Social VPNs: Integrating overlay and social networks for seamless P2P networking. In *17th IEEE International Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises*, June 2008.
- [Fra03] Justin Frankel. Waste P2P Darknet, 2003. <http://waste.sourceforge.net/>.
- [GGD<sup>+</sup>08] Sarunas Girdzijauskas, Wojciech Galuba, Vasilios Darlagiannis, Anwitaman Datta, and Karl Aberer. Fuzzynet: Zero-maintenance Ringless Overlay. Technical report, 2008.
- [GIEvS06] P. Garbacki, A. Iosup, D.H.J. Epema, and M. van Steen. 2fast: Collaborative downloads in p2p networks (best paper award). In *6-th IEEE International Conference on Peer-to-Peer Computing*, pages 23–30. IEEE Computer Society, sep 2006.
- [GMP09] F. Giroire, J. Monteiro, and S. Pérennes. P2p storage systems: How much locality can they tolerate? Technical Report 7006, INRIA, 2009.
- [Gol07] Jennifer Golbeck. Quechup: Another Social Network Enemy!, Sept. 2007. Oreillynet.com.
- [GPM<sup>+</sup>08] Kalman Graffi, Sergey Podrajanski, Patrick Mukherjee, Aleksandra Kovacevic, and Ralf Steinmetz. A distributed platform for multimedia communities. In *IEEE International Symposium on Multimedia (ISM’08)*, page 6, Berkley, USA, Dec 2008. IEEE, IEEE Computer Society Press.
- [GSS08] Olaf Gorlitz, Sergej Sizov, and Steffen Staab. Pints: Peer-to-peer infrastructure for tagging systems. *IPTPS*, 2008.

- [KBC<sup>+</sup>00] J. Kubiawicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, C. Wells, et al. Oceanstore: An architecture for global-scale persistent storage. *ACM SIGARCH Computer Architecture News*, 28(5):190–201, 2000.
- [LCP<sup>+</sup>05] Eng Keong Lua, Jon Crowcroft, Marcelo Pias, Ravi Sharma, and Steven Lim. A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys & Tutorials*, 7(2):72 – 93, 2005.
- [LD06] J. Li and F. Dabek. F2F: Reliable storage in open networks. In *IPTPS, Proceedings*, 2006.
- [LPG<sup>+</sup>07] Nicolas Liebau, Konstantin Pussep, Kalman Graffi, Sebastian Kaune, Eric Jahn, André Beyer, and Ralf Steinmetz. The impact of the p2p paradigm. In *Proceedings of Americas Conference on Information Systems 2007*, Aug 2007.
- [mAYLL<sup>+</sup>09] 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 Position Papers*, 2009.
- [MBP<sup>+</sup>09] J.J.D. Mol, A. Bakker, J. Pouwelse, D.H.J. Epema, and H.J. Sips. The design and deployment of a bittorrent live video streaming solution. In *ISM 2009*. IEEE Computer Society, December 2009.
- [MGGM04] S. Marti, P. Ganesan, and H. Garcia-Molina. Dht routing using social links. In *The 3rd International Workshop on Peer-to-Peer Systems*. Springer, 2004.
- [MM02] Petar Maymounkov and David Mazieres. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *LNCS: International Workshop on P2P-Systems*, volume 2429, pages 53 – 65, 2002.
- [MPES09] M. Meulpolder, J.A. Pouwelse, D.H.J. Epema, and H.J. Sips. Bartercast: A practical approach to prevent lazy freeriding in p2p networks. In State University of New York Yuanyuan Yang, editor, *Proceedings of the 23rd IEEE International Parallel and Distributed Processing Symposium*, pages 1–8, Los Alamitos, USA, May 2009. IEEE Computer Society.
- [Per07] Juan Carlos Perez. Facebook’s Beacon More Intrusive Than Previously Thought, Nov 2007. <http://www.pcworld.com/article/id,140182-c,onlineprivacy/article.html>.
- [PGW<sup>+</sup>08a] J.A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D.H.J. Epema, M. Reinders, M. van Steen, and H.J. Sips. Tribler: A social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 20, February 2008.
- [PGW<sup>+</sup>08b] J.A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D.H.J. Epema, M. Reinders, M. van Steen, and H.J. Sips. Tribler: A social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 20:127–138, February 2008.
- [Sch08] Doris Schiöberg. A peer-to-peer infrastructure for social networks. Diplom thesis, TU Berlin, Berlin, Germany, December 17, 2008.
- [Sky04] Skype.com. Skype P2P telephony explained, 2004. <http://www.skype.com/intl/en/download/explained.html>.
- [SMLN<sup>+</sup>03] I. Stoica, R. Morris, D. Liben-Nowell, DR Karger, MF Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *Networking, IEEE/ACM Transactions on*, 11(1):17–32, 2003.

- [STW00] M. Steiner, G. Tsudik, and M. Waidner. Key agreement in dynamic peer groups. *IEEE Transactions on Parallel and Distributed Systems*, 11(8), 2000.
- [SW09] Daniel R. Sandler and Dan S. Wallach. Birds of a fethr: Open, decentralized micropublishing. In *8th International Workshop on Peer-to-Peer Systems (IPTPS '09) April 21, 2009, Boston, MA, 2009*.
- [TCL08] D.N. Tran, F. Chiang, and J. Li. Friendstore: cooperative online backup using trusted nodes. In *SocialNets '08: Proceedings of the 1st Workshop on Social Network Systems*, pages 37–42. ACM, 2008.
- [WJF<sup>+</sup>08] J. Wang, J.A.Pouwelse, J.E. Fokker, A.P. de Vries, and M.J.T. Reinders. Personalization on a peer-to-peer television system. *Multimedia Tools and Applications*, 36:89–113, 2008.
- [XKL07] Susu Xie, Gabriel Y. Keung, and Bo Li. A measurement of a large-scale peer-to-peer live video streaming system. *Parallel Processing Workshops, International Conference on*, 0:57, 2007.
- [YKGF06] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. In *SIGCOMM*, 2006.
- [ZD09] Lukasz Zaczek and Anwitaman Datta. Mapping social networks into p2p directory service. In *SocInfo, International Conference on Social Informatics*, 2009.
- [ZLLsPY05] Xinyan Zhang, Jiangchuan Liu, Bo Li, and Tak shing Peter Yum. Cool-streaming/donet: A data-driven overlay network for peer-to-peer live media streaming. In *in IEEE Infocom*, 2005.