Peer-to-Peer Netw. Appl.
DOI 10.1007/s12083-014-0258-2

A taxonomy of decentralized online social networks

Shihabur Rahman Chowdhury · Arup Raton Roy · Maheen Shaikh · Khuzaima Daudjee

Received: 27 January 2013 / Accepted: 27 February 2014

Abstract Despite their tremendous success, centrally controlled cloud-based Online Social Networks (OSNs) have inherent issues related to user privacy and control. These issues have motivated researchers to make a paradigm shift in the OSN architecture by proposing to replace centrally controlled OSNs with Decentralized OSNs (DOSNs) in a peer-to-peer setting. DOSNs give users more autonomy and the chance to participate in social networks without loosing control over their data. The various DOSN proposals have significant differences in their proposed services, architecture and extent of decentralization. In this survey, we study a number of proposals for peer-to-peer DOSNs, distil a set of criteria to compare them, and provide a taxonomy for their comparison.

Keywords Peer to peer system · Online social network

1 Introduction

Social networks result from social interactions between groups and individuals. The advent of social networking sites like Facebook, Twitter, and Flickr has taken social networking to Internet scale. These OSN sites attract a majority of the Internet users. According to *Nielsen's Social Media Report 2011* [1], around 80 % of active Internet users daily visit one of the OSN sites everyday. Most of the mainstream OSN sites provide free storage for users to upload and share their social content such as photos, videos, blogs, *etc*. The popularity of OSN sites along with their storage and sharing facilities are influencing the trend of content sharing over the Internet. Instead of using free or paid file hosting services (e.g., MediaFire, RapidShare, etc.), people are using OSN sites as a means for storing and sharing content with their social peers. Facebook is currently the largest OSN site with more than one billion active users [2]. It maintains more than 100 petabytes of online storage and stores more than 100 billion images [3].

However, all of the mainstream OSN providers have a cloud based infrastructure that is controlled by a single authority (e.g., Facebook's thousands of servers are under the control of Facebook authority only). That authority can gain ownership of the user's content to some extent [4] and use it for their business purpose. Despite the immense popularity and free services provided by these OSN sites, their centralized nature is bringing imbalance to the Internet's ecosystem in many ways. First, the centralized architecture requires the users to upload their social content to a cloud based storage, which is under the control of a central authority. While cloud based storage itself can be distributed over a large number of servers, these servers are controlled by a single authority, thereby providing a single system image to the end users. This poses serious threats to the user privacy and content ownership. For example, many OSN sites use their users' data to feed the advertisement industry. Second, centralized cloud based storages are also creating

S. R. Chowdhury (✉) · A. R. Roy · M. Shaikh · K. Daudjee
David R. Cheriton School of Computer Science,
University of Waterloo, Waterloo, ON, Canada
e-mail: sr2chowdhury@uwaterloo.ca

A. R. Roy
e-mail: ar3roy@uwaterloo.ca

M. Shaikh
e-mail: m7shaikh@uwaterloo.ca

K. Daudjee
e-mail: kdaudjee@uwaterloo.ca

Published online: 29 May 2014

data silos in the Internet [27]. Because of the closeness and lack of interoperability between OSN sites, users cannot share their data between different sites. For example, users cannot share their uploaded photos in Facebook with their friends in Google+ without re-uploading their photos in Google+. Because of the lack of interoperability there is also no common framework or API for developing social applications. Developers need to develop the same application using different APIs provided by different OSN providers to make their application available on different OSN platforms. Finally, OSN providers reserve the right to change the terms and conditions of usage at any time and the users have to agree with that.

The limitations posed by the centralized nature of OSNs have motivated the research community to develop alternative OSN architectures. The main theme of these alternative proposals is to give OSN users more autonomy in terms of storing and controlling the access rights of their contents. Most of these proposals advocate a decentralized peer-to-peer (P2P) architecture where an OSN infrastructure is formed through participation by a set of autonomous OSN users collaborating with each other. As a result, users can have more control over where their contents will be stored and how they will be accessed. This, in turn, gives users freedom to participate in any OSN without the need to migrate their data between different systems.

An early proposal [27] describes the limitations that centralized OSNs impose and proposes a decentralized architecture for federating the centralized OSNs as a first step to solve these problems. Marcon et al. conducted a measurement study to show the feasibility of sharing social content from users' homes [28]. PeerSon [15], the first complete architecture for a DOSN was proposed in 2009. In the following years, there have been a number of proposals for DOSN architectures [12, 16, 20, 32, 38, 40]. Apart from architectures, a good number of research effort has also been put towards particular issues for DOSNs [11, 13, 21–23, 37]. The decentralization offered by these systems results from the peer-to-peer nature of the proposed architecture. Therefore, we consider the terms decentralized and peer-to-peer equivalent and refer to these systems as Decentralized Online Social Network (DOSN) in rest of the paper. Different projects offer different degrees of decentralization and address different subsets of issues related to the live deployment of DOSN. This has motivated us to study a representative set of existing works in this field, distil a set of criteria for comparing these works, and finally compare and contrast the selected works based on the criteria. Prior work [17] surveys DOSN architectures in addition to some decentralized systems in the context of social P2P file sharing. In this survey, our primary focus is on systems that provide social networking features. Additionally, we include a discussion of more recent systems and, in contrast to [17], identify a set of criteria that we use to construct a taxonomy for the classification of DOSN architectures.

The rest of the paper is organized as follows. First, we briefly describe a number of DOSN proposals (Section 2). Second, we present a set of criteria for comparing different DOSN proposals along with specific issues to consider for each (Section 3). These issues serve as alternate design choices for implementing a DOSN. Third, we use these alternate options for each criteria to show how different proposals have adopted different classes of design choices to achieve particular goals (Section 4). Finally, we conclude in Section 5.

2 DOSN: the current picture

The privacy concerns and closed nature of existing OSNs have motivated researchers to come up with different proposals for DOSN. In this section, we present a brief discussion on a number of proposals found in the literature (Table 1). These projects present sufficient variation in their architecture, proposed services and design choices.

2.1 P2P social networking (PeerSon)

PeerSon [15] is one of the early prototypes of DOSN. This prototype was implemented based on the ideas developed in [14], where the authors discuss some challenges for decentralizing OSNs and provide some preliminary solutions for them. The prototype implementation supports user login procedure and file sharing service among social peers.

PeerSon proposes a two tier architecture that decouples user contents from the control infrastructure (Fig. 1). The lowest tier consists of users and their contents. Users can store their content on their own storage system and they can directly exchange contents among themselves. The upper

Table 1 List of DOSN systems

System Name	Reference	Year
P2P Social Networking (PeerSon)	[15]	2009
Featherweight Entangled Timelines over HTTP Requests (FETHR)	[35]	2009
SafeBook	[16]	2009
PrPl	[36]	2010
Cuckoo	[41]	2010
Vis-á-Vis	[38]	2011
SuperNova	[40]	2012
Cachet	[31]	2012

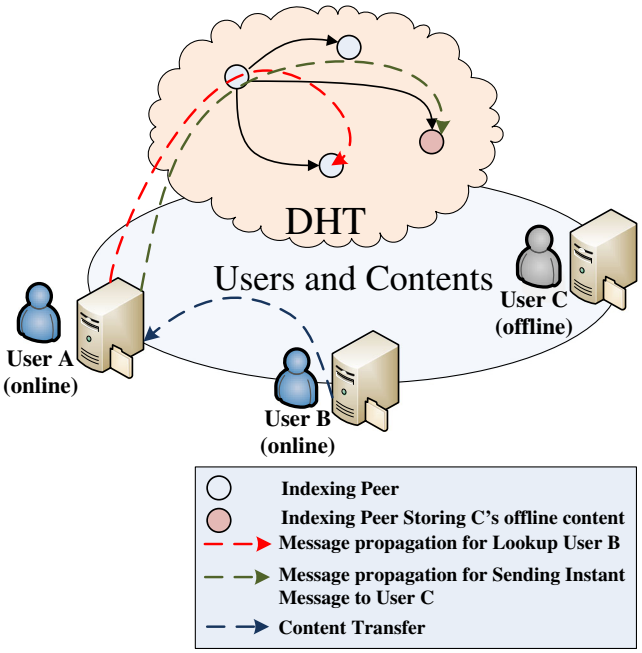


Fig. 1 PeerSon architecture

tier provides lookup services (e.g., find friends, find contents), stores users' meta-data, and keeps the updates for a user during that user's offline period. The authors propose to use a structured P2P overlay, i.e., a Distributed Hash Table (DHT) to implement the lookup service. The prototype implementation of PeerSon uses OpenDHT [34] service deployed on PlanetLab [5]. However, the implementation choice of the lookup service is not restricted to DHT, rather it is open ended. This open ended choice for implementation allows going from single authority controlled to completely decentralized lookup service. However, this lookup service does not store users' private data and contents; it stores only the meta-data necessary for serving lookup queries.

Identity management and privacy control in PeerSon assumes the existence of a Public Key Infrastructure (PKI). Users encrypt their content with the public key and distribute the key to the intended audience. Contents become accessible only to those who have the right key. In this way, users can have fine grained control over the access rights of their contents.

2.2 SafeBook

Safebook [16] proposes a three tier architecture for DOSN with the main focus on privacy, integrity and availability (Fig. 2). The lowest tier consists of the users and their social relationships. This tier handles data storage, content availability and communication privacy. A P2P overlay on top of the social networking tier provides the application services

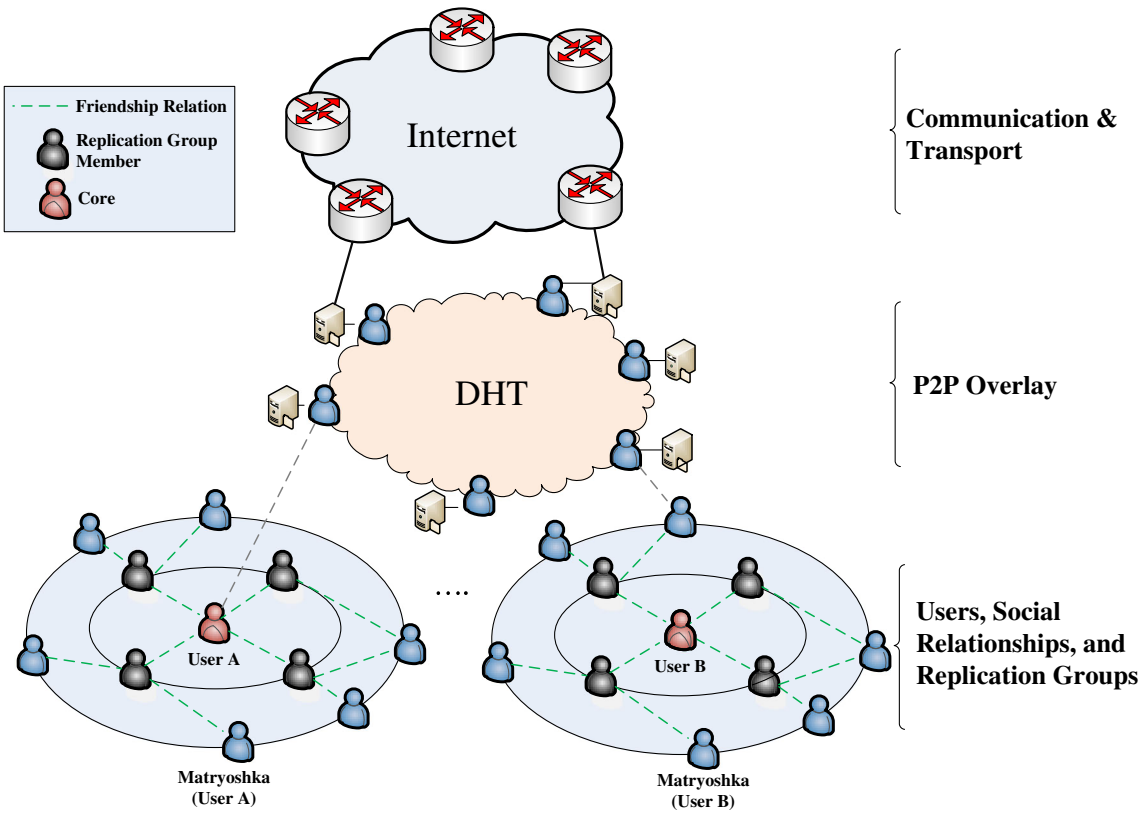


Fig. 2 Safebook architecture

(e.g., lookup service). The Internet sits in the topmost tier providing communication and transportation services.

At the core of operations in the social networking tier lies a per user logical concentric structure of other SafeBook users that is referred to as *Matryoshka* [16]. The centre or innermost user of a Matryoshka is known as the *core*. The innermost logical circle consists of the core's direct social peers (e.g., friends). Users in successive logical rings are logically connected by friendship or trust relationships. The innermost circle acts as the core's replication group to ensure high content availability for the core. The outermost circle handles the requests for accessing data at the core and forwards the requests to inner layers until it reaches the core. A user's data is kept encrypted in the replicas to ensure privacy and security. The Matryoshka ensures that the path to a user is through its trusted peers only.

The architecture of the P2P overlay in Safebook is similar to the KAD network [6] (an implementation of Kademila P2P Overlay [29]). This overlay provides users with various application layer services, e.g., lookup and identity management service. A service in the P2P overlay named Trusted Identification Service (TIS) manages user identities and provides identifiers and pseudonyms to new users. Identifiers are used in the social networking layer, whereas pseudonyms are used in the P2P overlay. Any communication to and from a user is done using the encryption and the decryption of the pseudonym's public and the private key, thus ensuring message integrity and confidentiality. After getting an identity, a new user starts the process of creating their Matryoshka by sending requests to their friends or trusted peers.

2.3 PrPl

PrPl [36] aims to build a DOSN with fine grained privacy control and also provides an API for social application development that can run across different administrative domains. PrPl allows the users to store their content (e.g., photos, music, status updates) in devices of their choice. A user's distributed storage is federated by a per user service named *Personal Cloud Butlers* [36]. This butler service keeps track of the location of the contents and gives a single system image of the underlying distributed storage to other users (Fig. 3). The butler service is also responsible for enforcing privacy control, storing a user's private data, maintaining list of friends, i.e., trust relationships, and communicating with other users' butler services to access their content.

PrPl relies on the OpenID [33] service for identity management. OpenID is a decentralized user centric identity management system with an underlying philosophy to provide a single virtual identity for each user. This system is responsible for ensuring the authenticity of the butler services. The butler services provide fine grained per user privacy control. The butler service provides its owner the provision to set per user access level to each individual's content. Thus, it becomes possible to allow/deny incoming access requests to individual content for individual users. PrPl assumes that this butler service and user's personal storage will be deployed on devices with relatively higher uptime, e.g., home gateways, set top boxes, gaming consoles, etc. Based on this assumption, PrPl does not provide any replication or caching mechanism to ensure high availability of the butler service and the user's storage system. Note that the butler service is a single point of contact for a user. Therefore, it is a potential scalability bottleneck for users with large numbers of friends.

One of the major contributions of PrPl is *SociaLite* [36], an expressive query language to develop social applications on top of the DOSN. SociaLite allows applications to query data from the butler services running across different administrative zones. A prototype implementation of a social music streaming application built on top of PrPl using the SociaLite query language has also been demonstrated [36].

2.4 SuperNova

SuperNova [40] proposes a DOSN architecture based on the concept of *super peers*. SuperNova allows users to be part of the OSN and to share their content while retaining

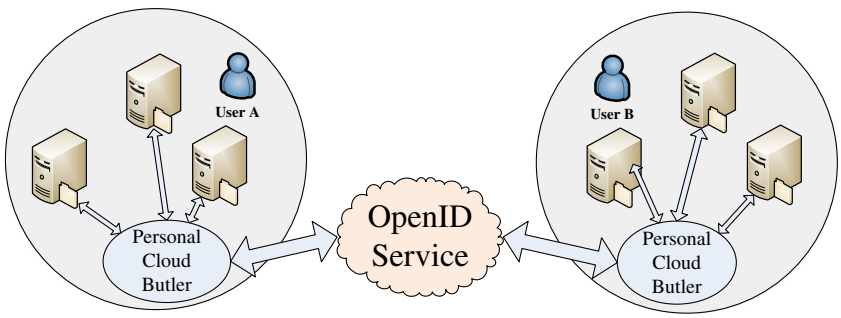


Fig. 3 PrPl architecture

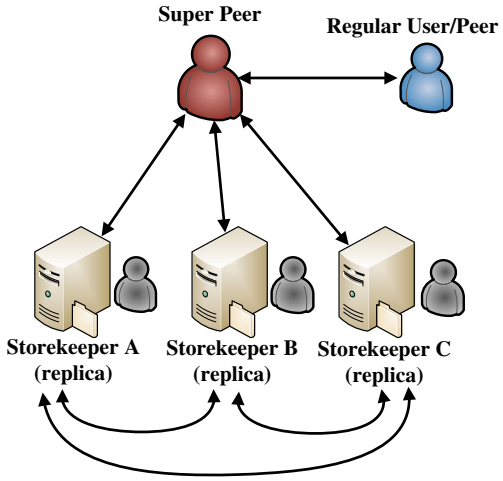


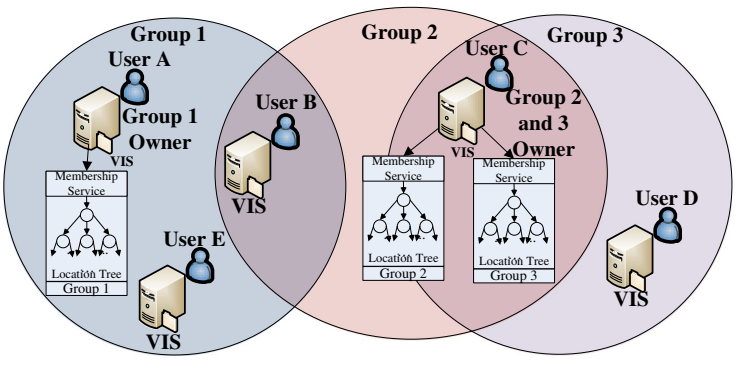
Fig. 4 SuperNova architecture

full content ownership. Moreover, users can impose public (accessible to all), private (accessible to none) or protected (accessible to a subset of friends) access on all of their contents. In SuperNova, super peers actively participate in the formation of the OSN's control infrastructure. SuperNova users can store their content on their own machines. However, a user's storage system might not be highly available, e.g., a cellphone which might not have 24×7 Internet connectivity. To deal with this issue, users can replicate their contents to a set of other users known as *Storekeepers* (Fig. 4). Super peers in the system track the availability pattern of users assigned to them. A super peer for users helps the users to choose their storekeepers by providing them with the availability pattern of other users. A user selects another user as its storekeeper when it can reach an agreement with the invited user. During a user's uptime, updates on its contents (e.g., comment on photos, videos etc.) are forwarded to that user only. The replicas are updated when the user goes offline and handle subsequent updates until the user becomes online again. To ensure data privacy, data is encrypted in the replicas. When a new user joins the system, its super peer provides temporary storekeeping service until the new user forms replication groups.

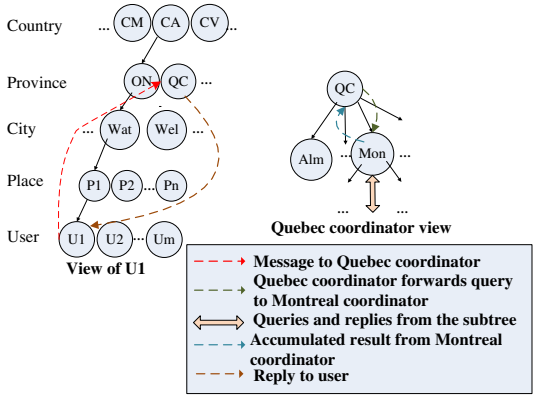
Super peers are the basic building blocks of the SuperNova architecture. They are responsible for providing lookup service (e.g., find potential replica, find friends), storage service, book-keeping service, recommendation service to a subset of the system's users, and to keep track of these users' replicas. Participation in the system costs them computational power, network bandwidth and storage space. The authors briefly discuss an advertisement based incentive for the super peers, whereby the super peers may put advertisements on the user profiles to gain monetary benefit. However, the proposed incentive models are not very concrete and a more concrete incentive model is required to strongly motivate users to provide super peer service at the cost of their resources.

2.5 Vis-á-Vis

Vis-á-Vis [38] provides a decentralized location-based social networking service. It proposes to use cloud computing facilities such as Amazon EC2 [7] instead of using personal workstations to achieve high availability of user services. The virtual machines running on the cloud, called *Virtual Individual Servers* (VISs) [38], provide storage and computation services for users. Figure 5a shows the architecture of Vis-á-Vis. Users store their personal data in their trusted VIS without any encryption. A user can become member of one or more groups and share geographical location with group members. A group is the unit of authoritative domain in Vis-á-Vis, i.e., a group is the smallest unit under the control of a single authority. Users have the freedom of choosing a different granularity level for location



(a) Vis-a-Vis architecture



(b) Search query for all users of Montreal city (figure adapted from [38])

Fig. 5 Vis-á-Vis

sharing, such as coarse granularity (e.g., city) or fine granularity (e.g., geographical coordinate). Users can also have different granularity levels for different groups. Groups are created and administered by a user. A group owner's public key is used to identify that group. The corresponding private key is stored in the group owner's VIS. On the other hand, the public key of the group and IP address of the corresponding VIS are broadcasted throughout the network so that any user of the network can request to join the group.

The initial architecture of Vis-á-Vis was based on a two tier DHT [39]. The upper tier was used for indexing all the VISs while the lower tier was used for indexing the groups in the system. This architecture changed later to support range queries for location-based services. For facilitating these services, information about each of the subdivisions of location covered by the group is stored in a tree structure known as *location tree* [38]. Each intermediate node of the location tree represents a subdivision of a region. A user's VIS becomes a leaf of the location tree according to the location it shares with the group. If a user shares a coarse location with a group then its VIS is stored randomly below the intermediate node corresponding to the user's shared location. Moreover, a coordinator (a VIS) is assigned to each intermediate node of the location tree for processing the search queries intended for the subtrees under it. The coordinator of a node is elected from all the coordinators of its children using a distributed consensus protocol such as Paxos [26] for a predefined lease period. Zookeeper [8] is also used as a distributed coordination service. For extending the lease period, each coordinator issues periodic lease-renewal messages using multicast. Each group also has membership service for supporting the admission policy of the group. In addition, membership service stores a reference to the group's location tree.

A Vis-à-Vis user stores a subgraph of the location tree for each of the groups that it belongs to. This subgraph contains the user's neighbours (which are sharing the same place), and all nodes and their siblings on the path from the user's node to the root of the location tree. Moreover, the VIS keeps information about the regional coordinators for all non-leaf nodes of this subgraph. Figure 5b shows the steps of the search query of a user U1, and the search initiator, for finding all the Montreal users. The query first goes to the coordinator of Quebec, the initiator's nearest regional coordinator whose geographical territory covers the location of the query. The coordinator of Quebec forwards the message to Montreal's coordinator. Montreal's coordinator, in turn, queries all of its subtrees, accumulates the response messages, and returns the aggregated message to the coordinator of Quebec. Finally, Quebec's coordinator returns the intended result to U1.

2.6 Cachet

Cachet [31] is a purely decentralized OSN where users collaborate with each other to store their contents without any centralized service. It is built on top of DECENT [24] and improves on DECENT's high latency of feed generation. Figure 6 shows its basic architecture along with the current

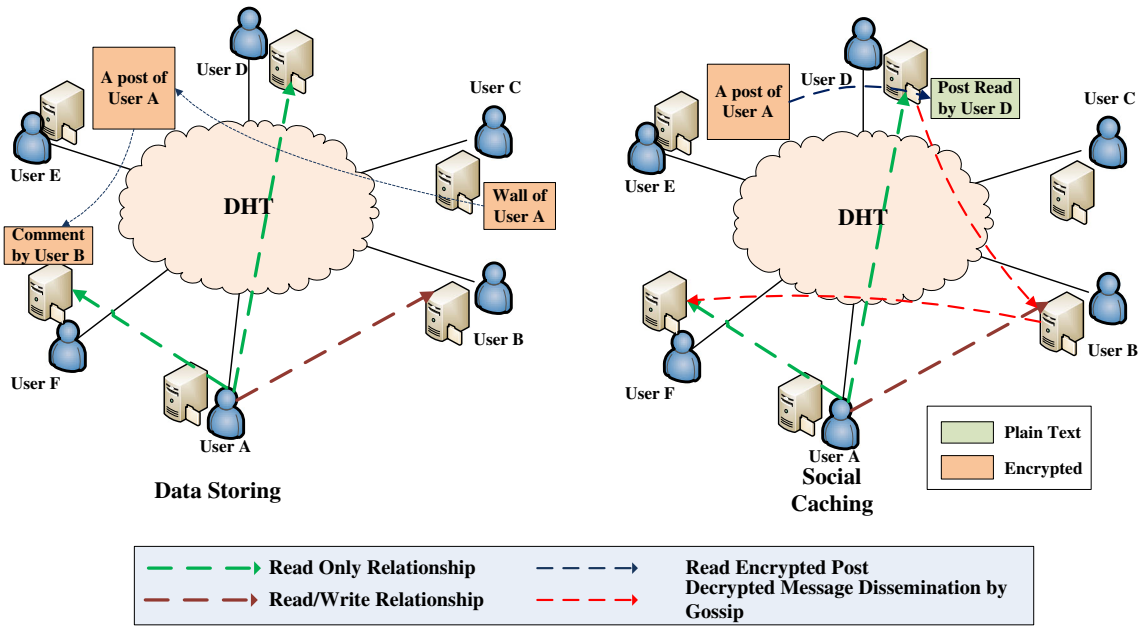


Fig. 6 Cachet architecture

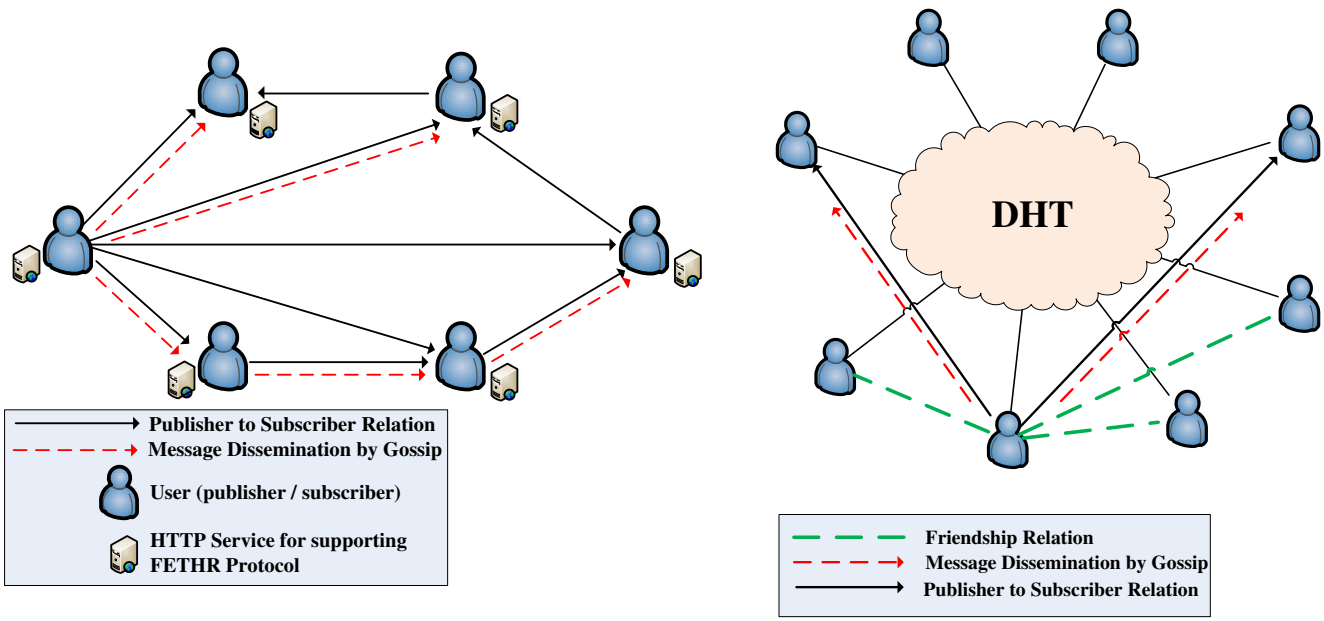
modification in content dissemination. In DECENT, users of the network form a DHT for storing container objects. A container object is the unit of access control in the system. This object stores the actual content and may contain references to other container objects. For example, a container object for a post by a user contains the corresponding text and also references to other container objects for each of the comments on the post. Moreover, sometimes content can take the form of references to other container objects. A typical example of this is a photo album in which the corresponding container object stores the references of the containers of all of its photos. All container objects use randomly generated identifiers for placement in the DHT and are stored with encryption in the storage nodes.

Owner-defined read, write, and append policies stored with the object metadata dictate the access control mechanism on the object. These policies can depend on user identity and/or the group for which the content is accessible. A user generates several encryption key pairs for each of the policies. Upon formation of a relationship, any user (user A) provides another user (user B) with an encryption key based on the mode of relationship. This encryption key governs accessibility of user B to either all or a subset of the contents of user A in the network. User B, in turn, also supplies a key to user A for accessing their content. However, the capabilities of the encryption keys exchanged by both users may not be the same, resulting in an asymmetric relationship among the users. An authorized user can locate a friend's data in DHT and decrypt it using the provided key by the owner. If permitted, any user can append to the content of their friend. Actual data of the append is stored as a separate container object in DHT and reference to this object is inserted in the container object of the previously read data. This newly created container object is also encrypted with a secret key from its owner and can be read by only those who are supplied with the corresponding public key.

The container level encryption technique in the basic architecture of Cachet requires costly decryption operations of several contents from a user's social contact during the generation of its news feed. To improve the response time, the authors augment the basic architecture of Cachet with social caching. In this method, each user node maintains a continuous secure connection with all of its online social contacts. Any node satisfying the attribute-based policy for a specific content can cache that content. Moreover, the node can provide this cached and decrypted data to other nodes, which also satisfy the policy. The latter feature also accelerates content retrieval when the owner of the content is offline. For maintaining a live connection with online users, a list called *presence list* is stored in a DHT [31]. The presence object, an entry in the presence list, is updated exclusively by the owner whenever it joins or leaves Cachet. For avoiding encryption, presence objects are also cached and disseminated using a gossip-based protocol.

2.7 Peer-to-peer microblogging

A number of architectures have been proposed in the literature for decentralized microblogging having similar features as found in Twitter. The first project of this kind was



(a) FETHR (b) Cuckoo

Fig. 7 Decentralized micropublishing architecture

Featherweight Entangled Timelines over HTTP Requests (FETHR) [35]. Later, Cuckoo [41], another decentralized microblogging system was released as an enhanced version of FETHR. In the rest of this section we describe these two systems in detail.

2.7.1 FETHR

FETHR [35] proposes a gossip-based decentralized micropublishing architecture (Fig. 7a) and is the first attempt to propose a DOSN similar to Twitter. The main contribution of this project is a HTTP-based communication protocol that allows the users to communicate with each other directly. FETHR supports Twitter-like features such as "following", and "tweeting".

A FETHR user can subscribe to other user's updates through simple HTTP GET and POST messages. Content publication follows a push based model where publishers push their updates directly to their subscribers. To mitigate the update propagation overhead of the broadcasters in the network, FETHR proposes gossip-based update propagation where updates are pushed to a subset of subscribers who in turn take the responsibility to push them to the rest of the network. Pushed contents at the subscribers increase the contents' availability since they can be served from there when the publisher goes offline. Content authenticity is ensured by including the hash of all prior events (publication, subscription) at a publisher into the content.

FETHR does not address a number of practical issues. For example, the HTTP-based communication protocol between users uses URL for communication. However, the process of resolving these URLs to user machine addresses, which can be behind NATs, or the user's overhead for managing and running HTTP services have not been addressed by FETHR.

2.7.2 Cuckoo

Cuckoo [41] proposes a decentralized micropublishing architecture and supports Twitter like services, e.g., subscriptions, microblog publications. But unlike its predecessor FETHR, Cuckoo has a structured architecture (Fig. 7b). Cuckoo creates a structured overlay (DHT) of its users to implement a lookup service. A newly joined user or an existing user can use this lookup service to find other users. However, this lookup process does not depend purely on the DHT, rather it follows a hybrid approach. The hybrid approach combines DHT's capability of guaranteed and efficient lookup of rare objects and of using flooding for fast lookup of popular objects. Cuckoo architecture adopts a push strategy to publish microblogs. The publishers use a gossip-based publication strategy similar to that of FETHR. The pushed contents in the subscribers act as a replica of the original content and increase the original content's visibility during the offline period of the publisher.

The main difference between Cuckoo and its predecessor, FETHR, is the DHT. FETHR does not provide any details regarding user management in a decentralized setting. Cuckoo addresses this issue by using a DHT. The DHT is formed by direct participation from the system's user, and thus, system operations do not rely on any centralized service and are not controlled by any centralized authority.

3 Comparison criteria

In this section, we present the set of criteria for comparing different DOSN proposals. For each criteria, we also discuss the specific issues that need to be taken into consideration for comparing them.

3.1 Architecture

The first criteria for comparison is the architecture of different DOSNs, i.e., how the different system components are organized to achieve the system's goals. In DOSN context, we particularly focus on the organization of the following system components:

- **Control:** The control part of the architecture consists of lookup services (both user and content lookup) and identity management services.
- **Storage:** The storage part of the architecture consists of storage of user's content and ensuring its high availability.

The organization of these components closely resembles that of P2P systems, and thus we have the following high level classification:

- **Structured:** Users directly participate to form a structured overlay or use services provided by a third party structured overlay. Any query in the system can be resolved using the structured overlay in a bounded number of steps.
- **Semi-structured:** A subset of all the users in the system (super peers) take responsibility for storing the index and managing other users. The super peers are responsible for providing the interface to the rest of the users for carrying out different system operations. User participation for providing super peer service can be voluntary or incentive-based.
- **Unstructured:** No user in the system maintains an index, and system operations are usually carried out using flooding or gossip-like communication between users.

Table 2 Different types of OSN service

Service	Type
Micropublishing	Read only
Commenting	Read-write
Multimedia Content Sharing	Read only
Newsfeed	Read only
Instant Messaging	Read-write

It is to be noted that a system can have different architecture for its different components, e.g., structured control and unstructured storage.

3.2 Types of service

Our set of DOSN systems provide different subsets of the services provided by the existing OSNs. Based on the type of operation a user can perform on other user's personal digital space, we have classified the services into the following two classes:

- **Read only services:** allow users of an OSN to view other users' content. It does not allow any user to write anything in other user's personal digital space. For example, the capability to only view another user's shared photos is a read only service.
- **Read-Write services:** allow users of an OSN to write into other user's personal digital spaces. For example, the capability to view and make a comment on other users' shared photos is a read-write service.

Table 2 gives a non-exhaustive list of different services that we shall take into account while comparing different DOSNs.

3.3 Social application development API

The existing OSNs provide developers with a set of APIs to develop social applications on their platform. The large user base of OSNs combined with the provided APIs have made them a popular platform for application development. These social applications also have a large active user base. For example, Zynga, the largest social games company reports having a monthly active user base of more than 230 million [9]. Even after decentralization, an application development API will remain an attractive feature of OSNs. Based on the types of services the APIs provide, we have the following classification of social application development API:

Social Graph Access API: allows querying of data from the underlying social graph, e.g., getting list of friends, finding mutual friends, etc.

User Meta-data Access API: accesses user's meta-data based on the user's privacy settings, e.g., user's demographic information

User Control API: allows automating social network activities, e.g., user authentication, content posting etc.

3.4 Availability architecture

In centralized OSNs, providers ensure system availability by providing dedicated servers and different fault tolerance mechanisms. However, in case of DOSNs, users have the freedom of storing their content on the device of their choice. These devices do not necessarily have high uptime and they may be quite prone to failures. For example, a user may choose to host some content from their cell phone, which might not have 24×7 Internet connectivity and might switch off if its battery dies. Therefore, ensuring 24×7 availability of users' contents stored in a decentralized setting is a challenging issue. Replication and caching are proven techniques to ensure availability when the original source of the content might not have high uptime. In the context of DOSN we consider the following issues regarding availability:

- **Availability Mechanism:** The DOSN proposals exhibit two major classes of techniques to ensure availability:
  - **Replication and Caching:** User content is replicated and/or cached to a set of other users so that they can act as a proxy when the content owner is not online.
  - **Stable Nodes:** The architecture assumes that the storage system is highly available. This assumption restricts user's choice of storage system. For example, home gateways and set top boxes have storage capabilities and they have higher uptime compared to a cellphone or laptop.

  Despite the above mechanisms, some of the DOSN proposals do not address the issue of ensuring high availability at all.
- **Replica Selection Policy:** If the DOSN supports replication of a users' contents then the choice of replica placement is another important issue to consider. In the context of DOSN the following policies for replica placement are possible:
  - **User selected** replicas based on trust relationship of a user with other users.
  - **System selected** replicas based on some parameters, e.g., user's availability pattern.
  - **User driven** replication, where replication is performed based on subscription from users.

– **Replica Synchronization:** If user content is replicated then keeping a single image of all the copies is also an issue.

3.5 Scalability

Scalability is measured as a system's ability to provide good or reasonable performance according to some metric under large loads. In the context of DOSN, individual users become part of the infrastructure and they have some storage, bandwidth and processing loads. In an OSN, users and their contents are the major load for the whole system. In a decentralized setting, the frequency of a user going online and offline, i.e., *churn*, also puts some overhead on the system components. Thus, the following load parameters are important for a DOSN:

– Number of users in the system
– Number of friends of a user
– Volume of contents of a user
– Volume of contents in the system
– Churn rate

To analyse the scalability of a system we have focused on the change of the aforementioned parameters on the following performance metrics for each user:

– Network Bandwidth
– Storage
– Update Overhead

3.6 Privacy control

Users can utilize privacy control mechanisms to control the degree to which their data is visible to other users of the OSN. An OSN service provides different privacy policies for the stored content in the network. The granularity level of visibility of a post can range from public to specific user. Enforcement of privacy can be done per post or on a global basis. The per post privacy enforcement obviously offers the greatest flexibility to users. Dynamic change in the privacy of content is also another concern that gives users the ability to modify the visibility of their contents at any time.

3.7 Security model

In OSN, a user shares a common platform with a vast number of unknown users for storing personal data and for communicating with other users. A number of user based studies [19, 42] have shown that OSN users tend to create social relationships through OSN sites even with weaker trust and privacy measures. The studies also show that OSN users tend to disclose private information regardless of privacy concerns. This situation is further complicated by third party social applications and the presence of Sybil users in the network. Krishnamurthy et al. [25] studied different ways for personal information leakage in social networks, and identified transmission of user data to third party servers as one of the potential causes of private information leakage in OSN. Under these circumstances, OSNs need to have robust security features to protect user's personal data while putting least possible responsibility on users to customize their privacy settings for maximum security. As many of the DOSNs store data in storage nodes of different administrative domains, the concern for territory of security widens in this decentralized scenario. In the context of DOSN, we focus on the following security issues:

– **User Authentication:** For identifying the legitimate users in the system, every user should go through the user authentication process. The system has to provide legitimate users with the credentials to communicate with it.
– **Confidentiality:** Disclosure of data to unauthorized personnel should be prevented by the system. No information regarding the messages exchanged among users or between a user and the system should be exposed to a third party. This end-to-end data confidentiality should be supported by secure communication. Moreover, encryption can be used to keep data on separate storage nodes of different administrative domains to protect them from being read by unauthorized parties.
– **Data Integrity:** Upon receipt of data, users should verify whether the content has been changed during transfer. If data are stored in a place other than the user's premises, the system should also consider the prevention of unauthorized modification of data at the storage node. Digital signature is one of the possible solutions to verify the integrity of the content.
– **Resilient to attack:** Social networking sites become an attractive target for attackers due to the huge amount of personal information that they store. Denial of Service (DoS) is a prevalent form of attack for disrupting the service of the system. Moreover, a user can create fake identities (Sybil attack [18]) for exploiting the shared resources unfairly and influencing other users to fulfill its intention. A DOSN feature for identifying and protecting against these forms of attack will make the system more trustworthy.

3.8 Business model

In traditional OSNs, users are offered free service with the illusion of infinite storage space by the provider. The service provider benefits financially by allowing the business

entities to show tailored advertisements to the users based on their pattern of interaction in the network. However, in the case of DOSN, the original idea was to establish an online community from voluntary participation of the users. A user in the network stores data of other users and does computation for them because he/she is also getting the same services from others in return. This voluntary model of DOSN can be augmented where users with higher storage and computation capacity and network bandwidth can offer service to the social network. Other than using a personal computing node, any user can avail this service with payment or by allowing advertisements depending on the agreement.

4 Comparative classification

In this section, we provide a comparison of the described systems with respect to the criteria presented in Section 3.

4.1 Structured vs. semi-structured vs. unstructured architecture

The control and storage of the surveyed DOSN systems can be classified into three categories as described in Section 3.1. Table 3 gives a classification of these systems with respect to their architecture.

Most of the proposals for DOSN use a structured P2P overlay for control. SuperNova proposes to use a super peer based semi-structured architecture, where each super peer is responsible for the management of a number of regular users. The super peer provides lookup service, tracks user uptime to recommend suitable users for replication, manages its users' replicas, and acts as a final backup in case all replicas of a user fails. FETHR has a decentralized control architecture. On the other hand, Cuckoo has a hybrid architecture; its lookup services take advantage of both structured and unstructured architectures. The structured lookup provides guaranteed lookup of rare items; whereas, the unstructured lookup provides fast discovery of popular items.

Storage and replica management in FETHR and Cuckoo is unstructured. Both systems use flooding or gossip-based protocol for the updates, which are not efficient in terms of network bandwidth. PrPl's storage architecture lies between structured and semi-structured. PrPl allows users to store the contents in a decentralized and unstructured manner. The decentralized storage is federated by a per user process, which resembles the responsibilities of a super peer. This storage structure gives users more flexibility for placing their content than that provided by other architectures. In the other systems, users form a structured overlay that manages their storage.

The main advantage of using structured or semi structured P2P overlay for control is its efficiency in terms of network bandwidth over unstructured overlay. An unstructured overlay has almost zero management overhead with more autonomy than the structured overlay. An interesting observation is that systems that propose rigorous privacy and security mechanisms stick to a structured architecture. This is due to the fact that with a structured approach, it is possible to give users sufficient autonomy while keeping control over privacy and security.

Table 3 Classification of DOSNs by architecture

System	Control	Storage
PeerSon	DHT	DHT
Safebook	DHT	DHT
PrPl	DHT	Between Structured & Semi-structured
Vis-à-Vis	Tree like Structured Overlay	Tree like Structured Overlay
SuperNova	Super peer	Super peer
Cachet	DHT	DHT
FETHR	Unstructured	Unstructured
Cuckoo	Hybrid (both structured & unstructured method)	Unstructured

4.2 Service types

Safebook, PrPl, FETHR, and Cuckoo support only read only services. From a users point of view, these cannot write anything to other users' personal digital spaces. Therefore, these proposals do not support services such as commenting on other users' content or instant messaging. Vis-á-Vis provides location based services and allows users to share their location with micropublishing. It also provides only read only services. All other DOSN proposals support read-write services and any user can write to other users' personal digital space based on the privacy level.

Note that FETHR and Cuckoo are tailored to support only micropublishing. Therefore, these two DOSNs do not support the sharing of multimedia content. The same applies for Vis-á-Vis. Table 4 gives a summary of different services provided by different DOSNs.

4.3 Social application development API

Some DOSN proposals have early prototypes whereas some exist only as a proof of concept. Thus, none of the projects discussed in the paper except PrPl provides a set of APIs

Table 4 Services provided by DOSNs

Service	PeerSon	Safebook	PrPl	Vis-à- Vis	SuperNova	Cachet	FETHR	Cuckoo
Micropublishing	✓	✓	✓	✓	✓	✓	✓	✓
Commenting	✓	×	×	×	✓	×	×	×
Multimedia Content Sharing	✓	✓	✓	×	✓	✓	×	×
Newsfeed	×	✓	✓	✓	✓	×	✓	✓
Instant Messaging	✓	×	×	×	✓	×	×	×

for application development. PrPl includes a query language, *SociaLite*. SociaLite provides two types of APIs to application developers, Social Graph Access API and User Meta-Data Access API. Developers can use these APIs to query relationship data from the underlying social graph and user data across different administrative domains. PrPl also demonstrates a social music sharing application developed using SociaLite.

4.4 Availability architecture

In a DOSN, the content is stored at user devices having relatively lower uptime compared to a dedicated cloud based storage. Therefore, ensuring content availability is a challenging issue in a DOSN. The authors in [30] performed trace based simulations to study the effect of various parameters on the content availability in DOSN. These parameters include user uptime, number of replicas and replica placement policy. Apart from global availability, the authors also consider the availability of users' contents to only their friends. They study three different replication strategies where the replicas are placed greedily to maximize a content's availability, at the most active friends of a user, and randomly. Their study shows that placing replicas at most active friends with a 40 % replication factor yields high availability of users' contents to their friends.

PeerSon acknowledges the availability issues in DOSN, however, it does not provide any mechanism to ensure high content availability in the system. The other projects adopt two approaches to increase availability of the contents. One approach is that the user level processes are assumed to be deployed on stable infrastructure that have relatively higher uptime. PrPl and Vis-à-Vis follow this model. PrPl assumes that the user level processes and storage will be deployed on home gateways, set top boxes, gaming consoles or user's trusted free or paid hosting services that are online most of the time. Vis-à-Vis is based on virtual servers running on Amazon EC2 infrastructure, which have very high uptime [10]. This approach eliminates the bandwidth and storage overheads associated with replica management. On the other hand, it restricts a user's option for hosting contents.

The other approach is to replicate a user's content to a number of other users that can serve as a proxy when the user goes offline. All other projects except PeerSon, PrPl, and Vis-à-Vis follow this approach. Two important issues related to replication are the replica selection policy and the replica synchronization mechanism. Of the proposed DOSNs, only Safebook allows a user to create a replication group based on the trust or friendship relationship with other users. Users follow an eager (synchronous) approach to keep the replicas synchronized. Replica placement in SuperNova and Cachet are decided by the system, i.e., by the super peers and DHT nodes, respectively. A similar DHT based replication scheme is proposed by the authors in [37]. However, their key contribution is the concept of β-availability groups, where at least β members of a replication group are expected to be online at a given instant. The structured overlay tracks the uptime for the users and acts as a matchmaking agent for forming the replication groups. Their simulation-based study shows that 2-availability groups can provide high resilience to failures while requiring reasonable overhead for group formation. In addition to replication, Cachet also uses caching to improve content access latency and content visibility when the original source is offline. SuperNova allows the update of replicas whenever the user or one of the replicas go offline.

The decentralized micropublishing architectures have a different replica placement policy. They use publish-subscribe model and replicate users' (publisher) content to only their subscribers. They follow a gossip-based approach to *eventually* propagate updates to the replicas. The eventual update propagation may cause the replicas to have a stale copy for some period of time. The gossip-based approach generates a large volume of network messages during update compared to the other approaches. Forsyth et al. also propose an update propagation and replication scheme for decentralized microblogging [21]. The authors propose to query for a microblog by random walk through the social graph and cache both the data and path information along the successful search path. Storing the path information allows future updates to be applied along the same path, therefore yielding consistent replicas. A recent study by Asthana et al. [12] investigates the optimal number of replicas and their placement in the context of

Table 5 Comparison of DOSNs by availability models

System	Availability Mechanism	Replica Placement	Replica Synchronization
PeerSon	Not Addressed	Not Addressed	Not Addressed
Safebook	Replication	User Selected	Eager
PrPl	Stable Nodes	No Replication	No Replication
Vis-à-Vis	Stable Nodes	No Replication	No Replication
SuperNova	Replication	System Selected (Placed by super peer)	Only During Failure
Cachet	Replication & Caching	System Selected (Placed by DHT)	Not Addressed
FETHR	Replication	User Driven	Eventual
Cuckoo	Replication	User Driven	Eventual

peer-to-peer microblogging. They also propose a gossip based algorithm for placing the microblogs at a fraction of the network to ensure its high visibility to other users. This work tries to find a balance between the number of users where a microblog is replicated and the number of users that needs to be queried to find a microblog, while keeping the bandwidth usage at minimum. According to their analysis a microblog needs to be replicated to about 20 % and 6 % users in a 10,000 and 100,000 user microblogging network, respectively, to ensure high availability of the post while requiring less bandwidth for future lookup.

A summary of different aspects of ensuring availability in DOSNs is presented in Table 5.

4.5 Scalability

The ability of a DOSN to scale to the number of users, contents, and churn is directly affected by its architecture, types of services, and availability model.

An unstructured architecture that uses gossip or flooding based techniques for query routing and update propagation generates a large volume of network messages. An increase in the number of users in the system will overwhelm the network with messages and the per user bandwidth requirement will increase as well. Since the uplink bandwidth of a user is fixed in practice, per user network bandwidth for both content dissemination and update propagation may become a scalability bottleneck in FETHR, Cuckoo and Cachet. However, FETHR and Cuckoo support micropublishing services only and short microblogs are stored at the users. Thus replication does not incur very high storage overhead. Therefore, an increase in the number of users or amount of content in the system will not have a significant effect on the storage overhead.

The structured and semi-structured architectures generate a much lower number of network messages than in unstructured architectures due to the informed nature of routing. These structured architectures generally exhibit a logarithmic relation between the number of messages and the number of users in the system. With an increase in the number of users there is little increase in the number of generated messages in the system. Hence, with an increase in the number of users in the system, per user network bandwidth does not become a scalability bottleneck for the control part of PeerSon, Safebook, PrPl, and Vis-à-Vis. However, network bandwidth can become a scalability bottleneck for the broadcasters, i.e., users with large number of friends or followers in the system. In the current DOSN proposals, systems with or without replicas have a single point of entry for a user's content. Therefore, network bandwidth can become a scalability bottleneck for users with large numbers of friends or followers.

When replication is used in a system, storage and update propagation overhead is incurred to some extent. In a super peer based semi-structured architecture such as SuperNova, the super peers have higher storage requirements compared to the users in the system because they provide backup for a large number of users in case of failures of all of the users' replicas. Therefore, the super peers may pose a potential scalability bottleneck if the super peer network is not scaled with the number of users in the system. Also, SuperNova's update-on-failure policy incurs high update overhead under high churn rates. Thus, SuperNova does not scale well with an increase in number of users and a high churn rate. On the other hand, although Safebook also uses replication, most of the Safebook users incur lesser storage overhead than SuperNova. This is because of its policy of replicating content to very close friends, which are small in number for most of the users. However, the broadcasters in Safebook may require replicating the content of a large number of users, suffering from scalability in terms of storage overhead. On the contrary, storage does not pose a scalability bottleneck for PrPl and Vis-à-Vis. In these architectures, it

Table 6 Scalability of DOSNs

System	Network Bandwidth	Storage	Update Overhead
PeerSon	Not Scalable for Broadcasters	Not Addressed	Not Addressed
Safebook	Not Scalable for Broadcasters	Less scalable with increasing content	Scalable
PrPl	Not Scalable for Broadcasters	Scalable	No Update Propagation
Vis-à-Vis	Not Scalable for Broadcasters	Scalable	No Update Propagation
SuperNova	Not Scalable for Boradcasters	Not Scalable for super peers as number of users increase	Not Scalable during high churn
Cachet	Not Scalable for Boradcasters	Scalable	Not Scalable
FETHR	Not Scalable for Broadcasters	Scalable	Not Scalable
Cuckoo	Not Scalable for Broadcasters	Scalable	Not Scalable

is the users responsibility to manage storage for increased content volume and there is no replication overhead. Scalability of different DOSN systems are summarized in Table 6.

4.6 Privacy control: per user to public

Vis-à-Vis is a location-based service offering only range searches based on geographical location. The visibility of a user is within the group(s) to which the user belongs. FETHR and Cuckoo offer a micropublishing service where a post of a user is visible only to its subscribers. Cachet and SuperNova offer the best privacy control policy among all the DOSN proposals. All three granularity levels of content visibility, i.e., public, group, and individual can be used in both Cachet and SuperNova. This visibility level can be enforced on a per post basis. In PrPl, a post of a user has a dedicated viewer; thus it supports a per post privacy enforcement policy. Safebook allows privacy control on group basis. However, PeerSon does not address the privacy control issues. Privacy control features of different DOSNs are summarized in Table 7.

4.7 Security

PeerSon assumes the existence of a PKI infrastructure to ensure confidentiality and data integrity. PKI also provides the required keys for data encryption for both storage and communication. Similarly, SafeBook uses TIS to verify a user's legitimacy and provides private-public key pair to the user. The message transferred through the network is encrypted with the receiver's public key. The sender also signs the message with his private key. For resisting Sybil attack [18], Safebook requires face-to-face meeting or presentation of identity proof (e.g., passport) to the TIS authority. On the other hand, Cachet uses symmetric key and attribute based encryption techniques. The data is stored in the DHT with encryption. For verifying the originality of the content, the signature created by the owner is stored along with the content. However, no identification service is discussed in the proposal of Cachet. The butler service of PrPl architecture stores data in encrypted form in the storage node. OpenID service acts as the identification and certification authority for each PrPl user. SuperNova stores data in super peers and storage nodes

Table 7 Privacy of DOSNs

Privacy Issue	PeerSon	Safebook	PrPl	Vis-à-Vis	SuperNova	Cachet	FETHR	Cuckoo
Content visibility	Not Addressed	Group	Individual	Group	All	All	Group	Group
Enforcement mechanism	Not Addressed	Group	Post	Group	Post	Post	Group	Group

Table 8 Security of DOSNs

System	User Authentication	Confidentiality	Data Integrity	Resilient to attack
PeerSon	Using PKI	Message transfer with encryption	Sign with private key	Not Addressed
Safebook	Using TIS	Message transfer with encryption	Sign with private key	Sybil attack resilient
PrPl	OpenID	Data stored with encryption	Using digital certificate	Not Addressed
Vis-à-Vis	Not addressed	Message transfer with encryption VIS stores raw data	Not addressed	Not addressed
SuperNova	Not Addressed	Data stored with encryption	Not Addressed	Not Addressed
Cachet	Not addressed	Encryption in storing and transfer	With digital signature	Not addressed
FETHR	Not Addressed	Not Addressed	Using hash chaining & digital signature	Not Addressed
Cuckoo	Not Addressed	Not Addressed	Not Addressed	Not Addressed

using encryption to ensure privacy. In FETHER, the publisher includes the hash of all prior events into messages to guarantee message authenticity. Moreover, data integrity is ensured with hash chaining. Cuckoo does not address the security issues. Finally, Vis-à-Vis stores the data in trusted VISs without any encryption, but the data undergoes encryption before transmission.

Almost all of the DOSN proposals consider encryption to ensure security of users' stored and communicated data. However, encryption usually incurs both communication and computational overhead. Bodriagov et al. [13] propose a set of criteria for evaluating encryption schemes for DOSNs, and also evaluate a number of encryption schemes based on these defined criteria. An interesting outcome of the study is that none of the encryption schemes used in the proposed DOSN architectures can achieve all the efficiency criteria defined by the authors. To solve this issue, the authors propose to use Broadcast Encryption (BE) schemes.

The security features of different DOSN proposals are summarized in Table 8.

4.8 Business model

All of the DOSN proposals except SuperNova adopt the voluntary business model. In the voluntary model, a user either stores content in the leased node (PrPl, Vis-à-Vis) or participates using his/her personal machine (PeerSon, SafeBook, Cachet, FETHER, Cuckoo). There is no monetary incentive involved in participation. On the other hand, the super peer and storekeepers of SuperNova can have a financial benefit by offering computing and storage facility to ordinary users.

5 Conclusion

In this survey, we have studied different aspects of eight DOSN proposals. While this set is not meant to be exhaustive, it provides a wide coverage of different design choices. We have distilled a set of criteria and specific issues related to these criteria to compare different DOSN proposals. Finally, we have provided a comparative analysis of the DOSN proposals based on our proposed set of criteria. Our survey can provide a rich knowledge base for researchers and system designers in the fertile area of peer-to-peer based online social networks.

Acknowledgments We would like to thank the anonymous reviewers and Natalie Borsuk for comments that helped to improve the presentation of the paper.

References

1. http://cn.nielsen.com
2. http://newsroom.fb.com/News/One-Billion-People-on-Facebook-1c9.aspx
3. http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/
4. http://www.facebook.com/about/privacy
5. http://planet-lab.org
6. http://en.wikipedia.org/wiki/Kad_Network
7. http://aws.amazon.com/ec2
8. http://hadoop.apache.org/zookeeper
9. wearsocial.net/blog/2010/06/rise-social-gaming/zynga
10. http://aws.amazon.com/ec2-sla/
11. Aiello LM, Ruffo G (2012) Lotusnet: tunable privacy for distributed online social network services. Comput Commun 35(1):75–88
12. Asthana H, Cox IJ (2013) A framework for peer-to-peer microblogging. In: 5th International workshop on peer-to-peer systems and online social networks, (HotPOST 2013)

13. Bodriagov O, Buchegger S (2013) Encryption for peer-to-peer social networks. In: Security and privacy in social networks. Springer, pp 47–65
14. Buchegger S, Datta A (2009) A case 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, USA
15. Buchegger S, Schiöberg D, Vu L-H, Datta A (2009) Peerson: P2P social networking: early experiences and insights. In: Proceedings of the second ACM EuroSys workshop on social network systems, SNS '09, pp 46–52
16. Cutillo L, Molva R, Strufe T (2009) Safebook: a privacy-preserving online social network leveraging on real-life trust. Commun Mag IEEE 47(12):94–101
17. Datta A, Buchegger S, Vu L-H, Rzadca K, Strufe T (2010) Handbook of social network technologies and applications. Decentralized online social networks. Springer
18. Douceur J (2002) The sybil attack. In: Peer-to-Peer Systems (Lecture Notes in Computer Science), vol 2429. Springer Berlin, Heidelberg, pp 251–260
19. Dwyer C, Hiltz SR, Passerini K (2007) Trust and privacy concern within social networking sites: a comparison of facebook and myspace. In: AMCIS, p 339
20. Famulari A, Hecker A (2013) Mantle: a novel dosn leveraging free storage and local software. In: Advanced Infocomm Technology, pp 213–224. Springer
21. Forsyth S, Daudjee K (2013) Update management in decentralized online social networks. In 5th International workshop on peer-to-peer systems and online social networks (HotPOST 2013)
22. Han L, Nath B, Iftode L, Muthukrishnan S (2011) Social butterfly: social caches for distributed social networks. In: Proceedings of SocialCom/PASSAT, pp 81–86
23. Han L, Punceva M, Nath B, Muthukrishnan SM, Iftode L (2012) SocialCDN: caching techniques for distributed social networks. In: 2012 IEEE International conference on peer-to-peer computing
24. Jahid S, Nilizadeh S, Mittal P, Borisov N, Kapadia A (2012) DECENT: a decentralized architecture for enforcing privacy in online social networks. In: 2012 IEEE international conference on pervasive computing and communications workshops (PERCOM Workshops), pp 326–332
25. Krishnamurthy B, Wills CE (2008) Characterizing privacy in online social networks. In: Proceedings of the first workshop on Online social networks. ACM, pp 37–42
26. Lamport L (1998) The part-time parliament. ACM Trans Comput Syst 16(2):133–169
27. man Au Yeung C, Liccardi I, Lu K, Seneviratne O, Berners-lee T (2009) Decentralization: the future of online social networking. In: W3C workshop on the future of social networking position papers
28. Marcon M, Viswanath B, Cha M, Gummadi KP (2011) Sharing social content from home: a measurement-driven feasibility study. In: Proceedings of the 21st international workshop on Network and operating systems support for digital audio and video, NOSSDAV '11, pp 45–50
29. Maymounkov P, Mazires D (2002) Kademlia: a peer-to-peer information system based on the xor metric. In: Peer-to-peer systems (Lecture Notes in Computer Science), vol 2429. Springer Berlin, Heidelberg, pp 53–65
30. Narendula R, Papaioannou TG, Aberer K (2012) Towards the realization of decentralized online social networks: an empirical study. In: 2012 32nd International conference on distributed computing systems workshops (ICDCSW). IEEE, pp 155–162
31. Nilizadeh S, Jahid S, Mittal P, Borisov N, Kapadia A (2012) Cachet: a decentralized architecture for privacy preserving social networking with caching. In: The 8th international conference on emerging networking experiments and technologies
32. Pouwelse JA, Garbacki P, Wang J, Bakker A, Yang J, Iosup A, Epema DH, Reinders M, Van Steen MR, Sips HJ (2008) Tribler: a social-based peer-to-peer system. Concurr Comput Pract Experience 20(2):127–138
33. Recordon D, Reed D (2006) Openid 2.0: a platform for user-centric identity management. In: Proceedings of the second ACM workshop on digital identity management, DIM '06, pp 11–16
34. Rhea S, Godfrey B, Karp B, Kubiatowicz J, Ratnasamy S, Shenker S, Stoica I, Yu H (2005) OpenDHT: a public DHT service and its uses. In: Proceedings of the 2005 conference on applications, technologies, architectures, and protocols for computer communications, SIGCOMM '05, pp 73–84
35. Sandler D, Wallach DS (2009) Birds of a fethr: open, decentralized micropublishing. In: Proceedings of the 8th international conference on Peer-to-peer systems
36. Seong S-W, Seo J, Nasielski M, Sengupta D, Hangal S, Teh SK, Chu R, Dodson B, Lam MS (2010) Prpl: a decentralized social networking infrastructure. In: Proceedings of the 1st ACM workshop on mobile cloud computing & services: social networks and beyond, MCS '10, pp 8:1–8:8
37. Shahriar N, Chowdhury SR, Sharmin M, Ahmed R, Boutaba R, Mathieu B (2013) Ensuring β-Availability in P2P Social Networks. In: 5th International workshop on peer-to-peer systems and online social networks (HotPOST 2013)
38. Shakimov A, Lim H, Cáceres R, Cox LP, Li KA, Liu D, Varshavsky A (2011) Vis-à-vis: privacy-preserving online social networking via virtual individual servers. In: Proceedings of COMSNETS, pp 1–10
39. Shakimov A, Varshavsky A, Cox LP, Cáceres R (2009) Privacy, cost, and availability tradeoffs in decentralized osns. In: Proceedings of the 2nd ACM workshop on online social networks, WOSN '09. ACM, New York, pp 13–18
40. Sharma R, Datta A (2012) Supernova: super-peers based architecture for decentralized online social networks. In: Proceedings of COMSNETS, pp 1–10
41. Xu T, Chen Y, Zhao J, Fu X (2010) Cuckoo: towards decentralized, socio-aware online microblogging services and data measurements. In: Proceedings of the 2nd ACM international workshop on hot topics in planet-scale measurement, HotPlanet '10, pp 4:1–4:6
42. Young AL, Quan-Haase A (2009) Information revelation and internet privacy concerns on social network sites: a case study of facebook. In: Proceedings of the fourth international conference on Communities and technologies, pp 265–274

Shihabur Rahman Chowdhury is currently a PhD student at the David R. Cheriton School of Computer Science, University of Waterloo. He received B.Sc. degree in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET). His research interests are in future Internet architecture, peer-to-peer systems, and software defined networking.

Arup Raton Roy received the B.Sc. degree in Computer Science and Engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2009. He is currently pursuing his M.Math. degree in Computer Science at University of Waterloo, under the supervision of Prof. Raouf Boutaba. His research interests include data center, cloud computing, complex network, and wireless communication.

Maheen Shaikh received her B.E degree in Software Engineering from Mehran University of Engineering And Technology (MUET) Pakistan. She is a recent graduate of Master of Health Informatics from David R. Cheriton School of Computer Science, University of Waterloo. Her Research Interest includes Cloud Computing in Health Care IT, Electronic Health Records, Clinical Decision Support and distributed databases. She is a recipient of various scholarships and awards during her undergrad studies at MUET.

Khuzaima Daudjee is a faculty member in the David R. Cheriton School of Computer Science at the University of Waterloo. His research interests are in distributed systems, and data management.