Cadros: the Cloud-Assisted Data Replication in Decentralized Online Social Networks

Songling Fu[§], Ligang He[#*], Xiangke Liao[§], Chenlin Huang[§], Cheng Chang[#], Bo Gao[*]

[§]School of Computer Science, National University of Defense Technology, Changsha, China
[*]Department of Computer Science, University of Warwick Coventry, UK
[#]School of Information Science and Engineering, Hunan University, Changsha, China
e-mail: songling.fu@gmail.com, liganghe@gmail.com

***Abstract*— Online Social Network (OSN) services are very popular nowadays. In order to protect the data privacy, Decentralized Online Social Network (DOSN) services have been proposed. In DOSN, the data published by a user and the data replicas are only stored in the friend circle of the user. Although full replication can improve Data Availability (DA), pure DOSNs may not be able to deliver sustainable data availability. This paper proposes a Cloud-Assisted scheme, called Cadros, to improve the DA in DOSN. This paper conducts quantitative analysis about the storage capacity of Cadros as the result of integrating the Cloud into DOSN, and further models and predicts the level of DA that Cadros can achieve. Extensive simulation experiments have been conducted to verify the effectiveness of Cadros.**

Keywords—Decentralized Online Social Network; Cloud; Data Availability; Data Replication; Erasure Coding

I. INTRODUCTION

In the last decade, Online Social Networks (OSNs), such as Facebook [18], Twitter and Sina Weibo [19], have gained extreme popularity with more than a billion users worldwide. OSNs allow a user to publish the data to all friends in his friend circle.

Currently, the OSN platforms are typically centralized, where the users store their data in the centralized servers deployed by the OSN service providers. The service providers can utilize and analyze these data to know the users' private information, such as interest and personal affairs, and in the worst case may sell these information to the third party. Therefore, the current Centralized Online Social Networks (COSNs) have raised the serious concerns in privacy [14, 15, 30].

In order to address the data privacy issue, an obvious solution is to encrypt the user data stored in the centralized server [1,2,16,29]. Typically, this solution works in the following way[1]. The user data are first encrypted with the secret key, and the secret key is then encrypted with the public keys of the corresponding friends. After a friend receives the encrypted data and secret key, it first decrypts the secret key with his own private key, and then the user data are decrypted with the secret key. However, the disadvantage of this encryption solution is that a user may have a large number of friends and a user may add or delete the friends over time. It is not practical to manage this many keys. Another obvious downside of this approach is that encrypting and decrypting the user data and the secret keys incur high overhead.

Therefore, Decentralized Online Social Networks (DOSNs) have been proposed recently as a promising solution to protect data privacy [1-4]. Although the DOSN products [17] are not as popular and mature as the OSN products [18], DOSN is indeed under active research and development. In DOSNs, in order to protect the data privacy the centralized servers are bypassed and the data published by a user are stored and disseminated only among the friend circle of the user [4]. Although DOSNs can help protect the data privacy, maintaining Data Availability (DA) becomes a big challenge [11, 29, 32]. This is because if a friend of the user is offline, the data stored in the friend cannot be accessed by other friends.

In order to achieve good data availability in DOSN, the data replication approach has been widely used [6]. Full replication is a popular replication approach. In this approach, a certain number of copies (e.g., *k* copies) are created for each data item published by a user and these data replicas are stored across the user's friends in the DOSN. By doing so, if a friend is offline, the data in this offline friend can be accessed through the replicated data stored in other friends. Consequently, data availability is improved.

Although data replication helps improve DA, the following characteristics of DOSN have negative impact on its data availability.

First, the friends in DOSN are highly volatile [22,28]. Further, the studies [11, 31] show that the online/offline states of individual friends show high correlation, which indicates that many friends in a friend circle may go offline in the same time duration. When this happens, there may not be enough online friends to contribute the sufficient storage to save the data (and their replicas) published by the user.

Second, in a typical DOSN, the data published by a user are distributed among the friends in his own friend circle. Some friend circles may be small (e.g., with tens of friends), which may also cause the situation during certain periods where there are not enough online friends to offer the adequate storage for the published data.

Third, the increasingly more data are being generated on the OSNs nowadays. On the other hand, current users often use the mobile devices, such as smart phones, to access the OSN

The corresponding author is Dr. Ligang He.

978-1-4799-5066-9/14 $31.00 © 2014 IEEE
DOI 10.1109/SCC.2014.15

IEEE computer society

services. The storage capacity in the mobile devices is much more limited than the desktop computers used in the “old fashioned” style of accessing OSNs. Adding even more strain, a mobile device owner typically only sets a small fraction of total storage capacity in his device to be used by the OSN client app installed in the device.

There is now a dilemma. On one hand, using the centralized server to store the published data raises the data privacy concern. On the other hand, using the friend nodes as the only storage facility may raise the data availability concern although data replication helps improve data availability. In order to further improve data availability while guaranteeing data privacy, this paper proposes a hybrid data replication and storage approach which combines the DOSN with the centralized server.

Nowadays, the Cloud becomes a popular storage platform. The Cloud is very suitable to be used as the centralized server in this work, because 1) it is available all the time and 2) the storage capacity offered by the Cloud can scale up and down according to the users’ demands. Thus, this work utilizes the Cloud as the centralized server to improve the data availability in Decentralized Online Social Networks (Cadros).

Due to the complexity of using encryption to protect DA, Cadros employs the erasure coding technique [20] to prevent the Cloud service provider from knowing the content of the stored data. In the erasure coding technique, the original data are split into m data segments, which are then encoded into n new data segments. Any r data segments of the n encoded segments can be used to reconstruct the original data. Thus, if the number of data segments stored in the storage facility offered by a Cloud service provider is less than r, the Cloud service provider cannot reconstruct and know the original data.

The erasure coding technique can also be regarded as a data replication technique, and its redundancy degree is $({}^{n}/_{m})$. Therefore, Cadros effectively employs two data replication techniques. Namely, all data replicas generated by full replication are stored in the friend circle, while less than r data segments generated by erasure coding are stored in the Cloud. Erasure coding can save storage space when ${}^{n}/_{m}$ is less than the number of data replicas generated for each data item in full replication (k), which is typical case. The first contribution of this work is to conduct the quantitative analysis about the amount of data that Cadros can store as the result of combining the Cloud and erasure coding with DOSN.

In order to help achieve the desired data availability, it is very useful to analyze and predict the user and the friends’ behavior in the DOSN, and act upon the analyses and make judicious replication and storage decisions in advance. The second contribution of this work is to analyze the probabilistic behavior of the friend circle in the DOSN and predict the values of two metrics at a future time point: i) the storage capacity that the friend circle can contribute and ii) the amount of data that the friends request to update at a future time point. Further, this work models the relation between the above two metric values and DA, and consequently predicts the level of DA that the Cloud-assisted DOSN system can achieve at a future time point.

The rest of this paper is organized as follows. Section II discusses related work. Section III analyzes the storage capacity that Cadros can provide. Section IV presents the methods to model and predict the DA. Section V presents the experimental results. Finally, Section VI concludes the paper.

II. Related Work

A. DOSN

To address the data privacy problem in COSNs, several decentralized approaches have been proposed [1-4].

Graffi et al. [1] advocate that online social networks will be the next main application field for the p2p paradigm. They proposed a secure and P2P-based solution for secure online social networks called LifeSocial.KOM. Buchegger et al. [2] proposed a decentralized, peer-to-peer approach coupled with encryption. Yeung et al. [3] adopted a decentralized approach by using the URIs as the identifiers throughout, which can provide the same (or even higher) level of user interaction as with many of the current popular OSN sties. Tandukar et al. [4] also proposed a decentralized OSN. With this approach, users can maintain the control over their data to protect their data privacy, and forward the social data selectively to reduce the irrelevant data among the users. None of these approaches only stores the data published by a user in his friend circle.

B. Data Availability

The existing work in improving data availability mainly focuses on designing smart data replication and data storage policies.

Shakimov et al. [5] propose three schemes for storing the data in DOSNs: the cloud-based scheme, the desktop-based scheme, and the hybrid scheme combining the above two. The approach proposed by Koll et al. [6] exchanges the recommendations among the socially related nodes in order to effectively distribute a user’s data replicas among the eligible nodes carefully selected in the OSN. In the approach developed by Olteanu et al. [7], the preferences are given to the nodes when it comes to selecting the nodes for storing the data (and their replicas) published by a user. The online friends of the user have the highest priority. When all friends are offline, the data are then stored in the nodes which are not in the user’s friend circle.

Buchegger et al. designed a two-tiered DOSN architecture (PeerSoN) [2]. One tier serves as a look-up service which is implemented by OpenDHT. The second tier consists of the peers and contains the user data. When a user is offline, his all data will be stored across the whole network. Cutillo et al. [8] propose a P2P-based DOSN (Safebook), in which each node is accessible through the so-called shells. The profile data is mirrored and stored in a subset of a node’s direct contacts, which forms the so called innermost shell. Tegeler et al. [29] propose an approach called Gemstone. Gemstone protects the user’s privacy by encrypting all data using ABE, and stores the user’s data in the so-called Data Holding Agents (DHAs). If a DHA itself is offline, the data have to be passed to the offline DHA’s DHAs.

C. Cloud-assisted p2p systems

In 2006, Li and Dabek [9] argue that a node should choose its neighbors (the nodes with which it shares resources) based on existing social relationships instead of randomly when deploying a distributed storage infrastructure in peer-to-peer systems. The system is called the F2F storage system, in which nodes restrict themselves to sharing storage and network resources only with their friends. The authors argue that the F2F system provides the incentives for the nodes to cooperate with each other, which results in a more stable system. Based on this idea, they later proposed a cooperative online backup system named Friendstore [10], which allows the users to back up the data into the trusted nodes (i.e., their friends and colleagues).

In 2011, Sharma et al. [31] argued that the limitation of storing data only on friends can come to the detriment of data availability, and showed that the problem of obtaining maximal availability while minimizing redundancy is NP-complete. In 2012, Gracia-Tinedo et al. [11] showed that pure F2F storage systems present a poor QoS, mainly due to the availability correlations, and proposed a hybrid architecture called F2Box to combine F2F storage systems and the cloud storage services. F2Box uses erasure coding to replicate the data and allow users to adjust the amount of redundancy according to the availability patterns exhibited by friends. Compared with F2Box, our work uses the combination of full replication and erasure coding and proposes the scheme to minimize the overhead caused by erasure coding while satisfying the desired level of QoS.

In 2011, Liu et al [12] present Confidant, a decentralized OSN designed to support a scalable application framework for the OSN data without promising users' privacy. A user's data are replicated on the trusted servers controlled by his/her friends. FS2You [13], presented by Sun et al. in 2009, is a large-scale and real-world online storage system with peer assistance, which can dramatically mitigate server bandwidth costs. FS2You stores the data in the cloud and the peers. Peers are allowed to request help from the cloud but only when some conditions hold. FS2You can achieve a reasonable and balanced tradeoff between the p2p storage system and the cloud storage system.

In 2013, Mega et al [16] proposed a cloud-assisted dissemination approach in social overlays. In this approach, Updates to the user's profile are always performed first on the profile store, which is encrypted and hosted in the cloud, and are then disseminated via the social overlay. When updates from the user are not received by a friend for a long time, the cloud serves as an external channel to verify their presence.

III. Analyzing the Storage Capacity of Cadros

Although it is assumed that the Cloud server can provide the unlimited storage capacity, in order to protect the data privacy the number of segments stored in the Cloud server for a data item must be less than r under erasure coding. Moreover, both full replication and erasure coding can be used to replicate the data and the redundancy degrees of these two techniques are different. This section conducts the quantitative analysis about the amount of published data that Cadros is able to store, as the result of combining the Cloud with DOSN.

Assume that k copies are generated for each data item under full replication, and that in erasure coding, the original data are split into m segments, which are then encoded into n new segments, and any r encoded segments can be used to reconstruct the original data.

D denotes the amount of published data stored by Cadros. D_f and D_e denote the parts of published data that are stored using full replication and erasure coding, respectively. Then Eq. (1) holds.

$$D = D_f + D_e \qquad (1)$$

Since k copies of replicas are generated under full replication, the amount of data replicas generated by full replication is $\left(k \cdot D_f\right)$. All these data replicas have to be stored in the friend circle for the sake of data privacy.

After applying erasure coding, the amount of data replicas generated is $\left(\frac{n}{m} \cdot D_e\right)$. Assume that the actual number of the segments stored in the cloud server is x. Then $(n-x)$ is the number of segments stored in the friend circle. The amount of data replicas stored in the friend circle can be calculated by

$$\frac{n}{m} \cdot D_e \cdot \frac{n-x}{n} = \frac{n-x}{m} \cdot D_e.$$

SS denotes the total storage space contributed by the friends in the DOSN. Then Eq. (2) holds.

$$SS = k \cdot D_f + \frac{n-x}{m} \cdot D_e \qquad (2)$$

Eq. (2) can be transformed to Eq. (3).

$$D_f = \frac{1}{k} \cdot SS - \frac{n-x}{k \cdot m} \cdot D_e \qquad (3)$$

Substituting D_f in Eq. (1) for that in Eq. (3), Eq. (1) becomes Eq. (4).

$$D = \frac{k \cdot m - (n-x)}{k \cdot m} \cdot D_e + \frac{1}{k} \cdot SS \qquad (4)$$

From Eq. (4), we can draw the following conclusions.

i) Typically, $k > \frac{n}{m}$, i.e., the redundancy of full replication is greater than that of erasure coding. In Eq. (4), therefore, $k \cdot m - (n-x) > 0$, which means that D increases linearly as D_e increases. Thus, when D_e is 0, i.e., all data are replicated using full replication, Eq. (4) obtains the minimal value, which is $\left(\frac{1}{k} \cdot SS\right)$. When D_e is D , i.e., all data are replicated using erasure coding, Eq. (4) obtains the maximum value, which is $\left(\frac{m}{n-x} \cdot SS\right)$.

ii) Since $k \cdot m - (n-x) > 0$, D increases as x increases. Since x must be less than r, the maximum of Eq. (4) is $\left(\frac{m}{n-r+1} \cdot SS\right)$.

Therefore, the range of the amount of published data that Cadros is able to store is

$$\left[\frac{1}{k}\cdot SS, \frac{m}{n-r+1}\cdot SS\right] \quad (5)$$

It is straightforward to know that when the friend circle is the only storage facility and only full replication is used to generate the data replicas, the amount of published data is $\left(\frac{1}{k}\cdot SS\right)$.

IV. MODELLING AND PREDICTING THE DATA AVAILABILITY

Section III analyzes the amount of published data that Cadros can store given the storage capacity of the friend circle SS . This section presents the method of modeling and predicting the level of Data Availability (DA).

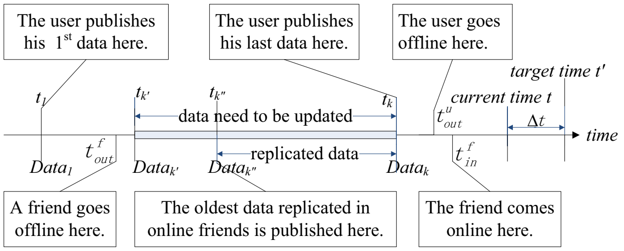

The user publishes his 1st data here.
The user publishes his last data here.
The user goes offline here.
target time t'
current time t
t_1
$t_{k'}$
$t_{k''}$
t_k
data need to be updated
t^u_{out}
Δt
time
t^f_{out}
replicated data
t^f_{in}
$Data_1$
$Data_{k'}$
$Data_{k''}$
$Data_k$
A friend goes offline here.
The oldest data replicated in online friends is published here.
The friend comes online here.

Fig. 1. Illustration of the data availability problem

In the DOSN system assumed in this paper, all nodes logs in and out the OSN service dynamically, and the online and offline duration of these nodes follow certain probability distributions. The user publishes the data following certain probability process (e.g., Poisson process). When a friend logs in, he tries to update the data published by the user after the friend logs out last time.

The DA problem is illustrated in Fig. 1. In Fig. 1, the user publishes the data at a series of time points along the time line. Assume t_1 is the first time point when he publishes the data, $Data_1$, after he comes online, and t_k is the last time point the user publishes the data, $Data_k$, before he goes offline at the time point t^u_{out}. Now consider one of the friends in the user's friend circle. Assume that the friend goes offline at time point t^f_{out} just before the user publishes $Data_{k'}$ (and after the user publishes $Data_{k'-1}$), and then comes online at time point t^f_{in} after the user goes offline. Therefore, $Data_{k'}$ to $Data_k$ are the data that the friend missed when he is offline and consequently need to update when he comes online. Since the user is already offline, the friend can only update the missed data from other online friends where the data replicas are stored, or reconstruct the missed data from the data segments stored in the Cloud server and/or other online friends. Note that if the friend comes online before the user goes offline, the friend can update all missed data from the user directly and the data availability is not a problem under this circumstance.

When a friend comes online, assume that the total amount of data that the friend tries to update is D_{up}. Out of D_{up}, the amount of data that are stored in Cadros is D_{st}. The level of DA for the friend is defined as Eq. (6).

$$DA = \frac{D_{st}}{D_{up}} \quad (6)$$

The data replication frameworks typically work in the following way [6, 10]. When the user publishes a data item, the data replicas are created and stored in the storage pools, which are either the online friends or the Cloud server in this work. If the storage capacity is unlimited, the newly published data will just be added. If the storage capacity is limited and the storage space is already full, the oldest data in the storage space will be replaced with the new data. Therefore, the storage capacity will determine the period of the data that are stored in the pool, which is called the time window of the stored data in this paper. The time window affects the *DA*.

For example, in Fig. 1, if the storage space can only store the data published from t_k back to $t_{k''}$, i.e., the time window of the stored data is $[t_{k''}, t_k]$, then the data published earlier than $t_{k''}$ are not available to the friend who goes offline at t^f_{out} and comes online at t^f_{in}.

One of the aims of this work is to model and predict the *DA* at a future time point t'. Eq. (6) shows that in order to predict the *DA*, we need to predict D_{st} and D_{up}. This section presents the methods to predict D_{st} (Section IV.A) and D_{up} (Section IV.B), respectively.

A. Predicting D_{st}

According to Eq. (5) in Section III, the amount of data that Cadros can store depends on SS, the total storage capacity that the friend circle can provide (all other parameters in Eq. (5) are constants). Therefore, in order to predict D_{st} at time t', we have to predict SS at t', which is denoted by $SS(t')$. The focus of this subsection is predict $SS(t')$, given the current time t.

The existing work [21-25] has extensively studied the patterns of the user behaviors in OSNs, such as the accessing frequency, online and offline durations, and the total time spent on OSNs. These patterns can be expressed by probability distributions. In this paper, we assume that the probability distributions for online and offline durations are already known.

Given the current time t, it can be determined that which friends are online or offline. For an online friend v_i at time t, we can know the time point at which v_i logged in (i.e., became online), which is denoted by $t^{fon}_{in\_i}$ ("fon" means "online friend", "in" means "login"). Note that in all notations in this paper, the superscript represents the role (e.g., "u" for "user", "$foff$" for "offline friend") and the subscript represents the action of the role or status (e.g., "in" for "login", "out" for "logout", "up" for "update data", "off" or "on" for being in the "offline" and "online" state).

The probability that the online friend v_i does not change to offline before t' equals to the probability that v_i will only log out after t' (i.e., v_i's logout time, denoted by $t^{fon}_{out\_i}$, is greater than t'). The probability, denoted by $p\left(t^{fon}_{out\_i} > t'\right)$, in turn equals to the probability that v_i's online duration is greater than $\left(t' - t^{fon}_{in\_i}\right)$ under the condition that v_i's online duration is no less than $\left(t - t^{fon}_{in\_i}\right)$, which can be computed using the conditional probability shown in Eq. (7), where $t^{fon}_{on\_i}$ is the time duration of friend v_i being online continuously and $F^{fon}_{on\_i}$ is the probability distribution function of $t_{on\_i}^{fon}$. The condition of $\left(t_{on\_i}^{fon} \geq t - t_{in\_i}^{fon}\right)$ in Eq. (7) reflect the fact that v_i has been staying online for the duration of $\left(t - t_{in\_i}^{fon}\right)$.

$$p\left(t_{out\_i}^{fon} > t'\right) = p\left(\left(t_{on\_i}^{fon} > t' - t_{in\_i}^{fon}\right) | \left(t_{on\_i}^{fon} \geq t - t_{in\_i}^{fon}\right)\right)$$
$$= \frac{p\left(t_{on\_i}^{fon} > t' - t_{in\_i}^{fon}\right)}{p\left(t_{on\_i}^{fon} > t - t_{in\_i}^{fon}\right)}$$
$$= \frac{1 - F_{on\_i}^{fon}\left(t' - t_{in\_i}^{fon}\right)}{1 - F_{on\_i}^{fon}\left(t - t_{in\_i}^{fon}\right)} \quad (7)$$

Similarly, the probability that an offline friend v_j becomes online at t' equal to the probability that it logs in before t', which can be computed using (8), where $t_{off\_j}^{foff}$ is the time duration of the offline friend v_j being offline continuously and $F_{off\_j}^{foff}$ is the probability distribution function of $t_{off\_j}^{foff}$.

$$p\left(t_{in\_j}^{foff} \leq t'\right)$$
$$= p\left(\left(t_{off\_j}^{foff} \leq t' - t_{out\_j}^{foff}\right) | \left(t_{off\_j}^{foff} \geq t - t_{out\_j}^{foff}\right)\right)$$
$$= \frac{p\left(t - t_{out\_j}^{foff} \leq t_{off\_j}^{foff} \leq t' - t_{out\_j}^{foff}\right)}{p\left(t_{off\_j}^{foff} \geq t - t_{out\_j}^{foff}\right)}$$
$$= \frac{F_{off\_j}^{foff}\left(t' - t_{out\_j}^{foff}\right) - F_{off\_j}^{foff}\left(t - t_{\mathrm{out\_j}}^{foff}\right)}{1 - F_{off\_j}^{foff}\left(t - t_{out\_j}^{foff}\right)} \quad (8)$$

S_i denotes the storage capacity contributed by friend v_i. Eq. (7) and Eq. (8) calculate the probability that friend, either online or offline at the current time t, will be online at time t'. Therefore, the expectation of the storage capacity contributed by the friend circle, i.e., $SS(t')$, can be calculated by Eq. (9).

$$SS(t') = \sum_{i=1}^{N_{on}} \left(S_i \cdot p\left(t_{out\_i}^{fon} > t'\right)\right) + \sum_{j=1}^{N_{off}} \left(S_j \cdot p\left(t_{in\_j}^{foff} \leq t'\right)\right)$$
$$= \sum_{i=1}^{N_{on}} \left(S_i \cdot \frac{1 - F_{on\_i}^{fon}\left(t' - t_{in\_i}^{fon}\right)}{1 - F_{on\_i}^{fon}\left(t - t_{in\_i}^{fon}\right)}\right)$$
$$+ \sum_{j=1}^{N_{off}} \left(S_j \cdot \frac{F_{off\_j}^{foff}\left(t' - t_{out\_j}^{foff}\right) - F_{off\_j}^{foff}\left(t - t_{out\_j}^{foff}\right)}{1 - F_{off\_j}^{foff}\left(t - t_{out\_j}^{foff}\right)}\right) (9)$$

Further, from Eq. (5) in Section III, $D_{st}(t')$ can be determined.

B. Predicting D_{up}

A friend needs to update the data through Cadros only when both of the following situations occur.

i) The friend v_j is offline at time t but comes online at time t', and $t_{out\_j}^{foff}$ is later than t_{out}^{u}.

ii) The user is offline at time t' (otherwise, the friend can update the data directly from the user).

Situation ii) can be further divided into two cases: 1) the user is online at the current time t, but becomes offline at time t', and 2) the user is offline at t, and remains offline at t'. We now present the methods to calculate $\mathrm{D_{up}}$ for both cases.

Case 1: the user is online at the current time t

As discussed in Fig. 1, when the friend v_j who is offline at the current time t and comes online at t', v_j needs to update the data only when $t_{out\_j}^{foff}$ is earlier than t_{out}^{u},. The time window of these data is $\left[t_{out\_j}^{foff}, t_{out}^{u}\right]$. $tw_{up\_j}(t', t_{out}^{u})$ denotes the length of the time window of the data that the offline friend v_j has to update when v_j comes online at t' and the user's last logout time is t_{out}^{u}. $tw_{up\_j}(t', t_{out}^{u})$ can be calculated using Eq. (10).

$$tw_{up\_j}(t', t_{out}^{u}) = t_{out}^{u} - t_{out\_j}^{foff} \quad (10)$$

Note that t_{out}^{u} is unknown at t and can be any time point in the time interval $[t, t']$. Therefore, the expectation of $tw_{up\_j}(t', t_{out}^{u})$, denoted by $tw_{up\_j}(t')$, can be calculated using Eq. (11), where $f(t_{out}^{u})$ is the probability density function (pdf) of t_{out}^{u}.

$$tw_{up\_j}(t') = E\left[tw_{up\_j}(t', t_{out}^{u})\right]$$
$$= \int_{t}^{t'} tw_{up\_j}(t', t_{out}^{u}) \cdot f(t_{out}^{u}) \cdot dt_{out}^{u}$$
$$= \int_{t}^{t'} \left(t_{out}^{u} - t_{out\_j}^{foff}\right) \cdot f(t_{out}^{u}) \cdot dt_{out}^{u} \quad (11)$$

$x_{pu}(t)$ denotes the number of times that the user publishes the data in the time duration t. $x_{pu}(t)$ is a discrete random variable. $f\left(x_{pu}(t)\right)$ denotes the probability density function of $x_{pu}(t)$. a denotes the average size of the data published by the user each time. The methods of analyzing and obtaining $f\left(x_{pu}(t)\right)$ and a have been presented in the literature [22]. $s_{pu}(t)$ denotes the total size of the data published by the user during t. Clearly, $s_{pu}(t) = a \cdot x_{pu}(t)$. Therefore, the probability density function of $s_{pu}(t)$, denoted by $f\left(s_{pu}(t)\right)$, can be determined by Eq. (12) and the expectation of $s_{pu}(t)$ can be calculated by Eq. (13).

$$f\left(s_{pu}(t)\right) = a \cdot f\left(x_{pu}(t)\right) \quad (12)$$
$$E\left[s_{pu}(t)\right] = a \cdot E\left[x_{pu}(t)\right]$$
$$= a \cdot \sum_{x=1}^{+\infty} x_{pu}(t) \cdot f\left(x_{pu}(t)\right) \quad (13)$$

Substituting the time duration t in Eq. (13) with $tw_{up\_j}(t')$ in Eq. (11), we can obtain $D_{up\_j}(t')$, i.e., the amount of the data which the offline friend v_j needs to update when he comes online at time t', which is expressed in Eq. (14).

$$D_{up\_j}(t') = E\left[s_{pu}\left(tw_{up\_j}(t')\right)\right] \quad (14)$$

Case 2: the user is offline at time t

The procedure for calculating $D_{up\_j}(t')$ for Case 2 is the same as that for Case 1. Only Eq. (11) in Case 1 should be re-derived, because t_{out}^{u} is known if the user is offline at time t, i.e., t_{out}^{u} is a constant. Therefore $tw_{up\_j}(t')$ should be calculated using Eq. (15).

$$tw_{up\_j}(t') = t_{out}^{u} - t_{out\_j}^{foff} \qquad (15)$$

C. Predicting DA

With the results of $D_{st}(t')$ and $D_{up\_j}(t')$, the level of DA that the offline friend v_j can achieve when he comes online at *t'*, denoted by $DA_j(t')$, can be calculated using Eq. (16).

$$DA_j(t') = \begin{cases} 100\% & D_{st}(t') \geq D_{up_j}(t') \quad (a) \\ \dfrac{D_{st}(t')}{D_{up\_j}(t')} & D_{st}(t') < D_{up\_j}(t') \quad (b) \end{cases} \qquad (16)$$

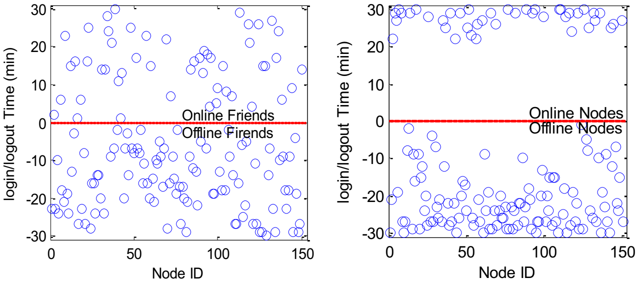
Since $D_{st}(t')$ can be any value in $\left[\frac{1}{k}\cdot SS, \frac{m}{n-r+1}\cdot SS\right]$ according Eq. (5), $DA_j(t')$ calculated by Eq. (16) falls in the corresponding range.

V. Evaluation

A discrete simulator has been developed in this work to simulate an OSN. There are *N* users in the simulated OSN. Some users act as the friends of another user and update the data published by the user. The online and offline durations of the users in the simulated OSN follow the Power-Law distribution (PL) or the Exponential distribution (Exp), as observed in the literature [21,26]. The user publishes the data following the Poisson process. Based on the methods presented in this paper, the replicas of the published data are created using either full replication or erasure coding and are stored in either online friends or the simulated Cloud Server.

In order to evaluate the prediction results, the experimental scenario is designed as follows. A friend contributes the storage capacity that is randomly taken from the range of $[S_{min}, S_{max}]$. The user and his friends log in and out following the specified probability distribution during the experiment interval $[0, l]$. The current time is set to be *m*-th minute ($m < l$). The online or offline states of all friends at time *m* as well as the latest login or logout time before time *m* are collected. The collected data, combing with the specified probability distributions of the friends' online and offline durations, are used to predict the total storage capacity contributed by online friends (i.e., *SS*) and the amount of the published data that have to be stored (i.e., *D*) at the future time points (i.e., the time points later than *m*). The predicted data are then compared against the data gathered from the actual simulation running. For example, Fig. 2(a) shows the first login/logout time of each friend when the number of the friends of a user is set to be 150, while Fig. 2(b) shows the latest login and logout time of each friend when the current time is set to be 31st min. A point above the red line (i.e., when y=0) in Fig. 2(b) represents the latest login time of a friend who is online at 31st min, while a point below the red line represents the latest logout time of a friend who is offline at 31st min.

Unless stated otherwise, the experimental parameters used in the simulator and the performance evaluation take the values shown in Table I. These values are chosen based on those used in the literature [21, 22, 26, 27].



(a) the initial states

(b) the states at 31st min

Fig. 2. The states of all friends at current time point

TABLE I. THE EXPERIMENTAL PARAMETERS

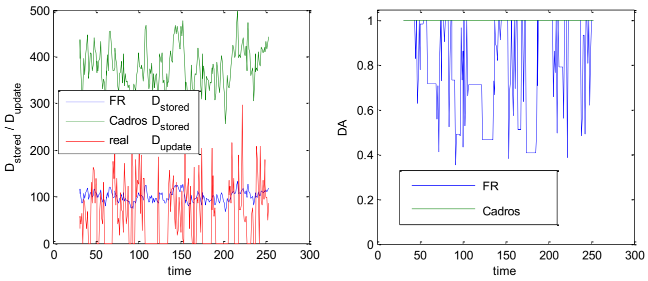
notations	default value	descriptions
N	150	The number of the user's friends.
a	1	The average size of published data
λ_{on}^{pl}	2.5	The parameter of the online time duration which follows power-law distribution
λ_{off}^{pl}	2.1	The parameter of the offline time duration which follows power-law distribution
λ_{pu}^{ps}	1	The parameter of the number of times that the user publishes data which follows Poisson distribution
k	3	Redundancy degree of full replication
m	5	The number of the original data segments needed before encoding in erasure code
n	8	The total number of the data segments after encoding in erasure code
r	5	The minimum number of the data segments needed to the original data in erasure coding
x	4	The maximum number of the data segments stored in cloud in erasure code
p_t	99%	The desired level of data availability
t	31	The current time point
S_i	[1,10]	The storage capacity contributed by friend v_i

A. Analyzing the Storage Capacity of Cadros

1) Compare storage capacity and data availability

The aim of combining the Cloud with DOSN is to increase the storage capacity of the system and thus to improve the data availability. Fig. 3(a) compares the real amount of data updated by offline friends (real D_{update}, shown as red line), the storage capacity achieved by Cadros (Cadros D_{stored}, shown as green line) and that obtained in the case where the data are replicated only using full replication and stored only in friend circle without the Cloud (FR D_{stored}, shown as blue line). In the experiments, the storage capacity available at each time point during the simulation run is recorded. We can see from Fig. 3(a) that, compared with FR, Cadros can not only store much more data but also satisfy the need of data update all the time, while FR cannot satisfy the need all the time.

Fig. 3(b) compares the data availability achieved by Cadros and that achieved by FR. We can see from Fig. 3(b) that the data availability of Cadros is 100% all the time, while the data availability of FR is not always 100%.



(a) storage capacity

(b) data availability

Fig. 3. Comparing the storage capacity and data availability between Cadros and Full Replication (FR) without the Cloud

2) Compare the overhead between Cadros and Erasure Coding (EC)

In Cadros, the data are partitioned in the way that the overhead caused by erasure coding is minimized while satisfying the desired level of DA. Fig. 4 compares the overhead caused by Cadros and that by pure erasure coding, i.e., all data are replicated using erasure coding. It can be seen from Fig. 4 that Cadros incurs much lower overhead in almost all cases. In many cases, the overhead is 0. This is because in those time points, the data replicas can be fully stored in the friend circle without the need to encode the data and store them in the Cloud. These results indicate that Cadros is able to judiciously determine the suitable data partition points and use erasure coding or full replication to replicate different portions of data.

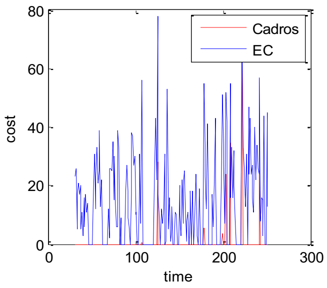
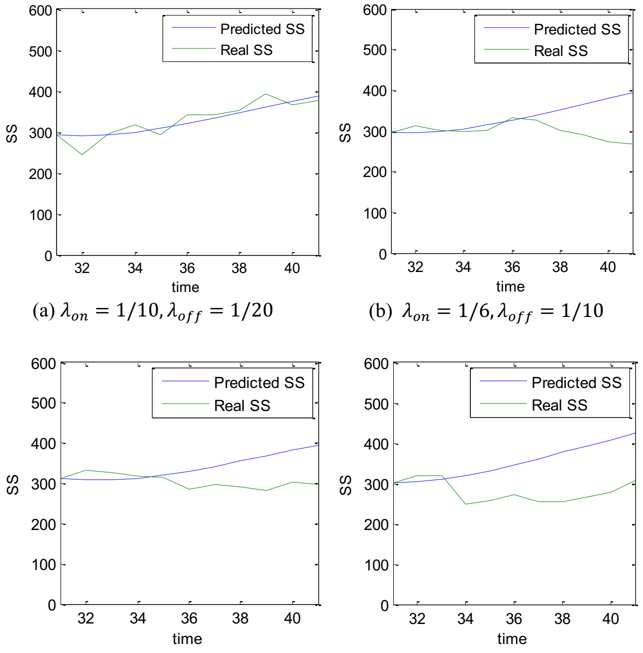


Fig. 4. Comparing the overhead between Cadros and Erasure Coding (EC)

B. DA Model

1) Prediction accuracy for SS

Fig. 5 shows the experimental results for the accuracy of predicting *SS*. In the experiments, the current time *t* is set to be 31st min and then SS at future time points is predicted. And then the actual values of SS are gathered as the simulation progresses. Fig. 5(a), (b), (c) and (d) show the results under different λ_{on} and λ_{off} (i.e., online and offline durations). It can be seen from Fig. 5(a) that compared with its actual values, the prediction of *SS* is fairly accurate in the first 10 minutes, which shows the effectiveness of the proposed prediction method. By comparing Fig. 5(a), (b), (c) and (d), we can see that the length of the accurate prediction decreases as the settings of λ_{on} and λ_{off} change from Fig. 5(a) to (d). These results indicate that the online and offline durations have impact on the prediction accuracy. After carefully analyzing the changing trend of λ_{on} and λ_{off}, it appears that the minimum value between the online and the offline durations (i.e., $\min(1/\lambda_{on}, 1/\lambda_{off})$) determines the length of accurate prediction. The less the value of $\min(1/\lambda_{on}, 1/\lambda_{off})$, the shorter the length of the accurate prediction. The reason for this is because when $\min(1/\lambda_{on}, 1/\lambda_{off})$ is smaller, the friends are more dynamic and consequently, it is more difficult to obtain the accurate prediction for future time points.



(a) $\lambda_{on} = 1/10, \lambda_{off} = 1/20$

(b) $\lambda_{on} = 1/6, \lambda_{off} = 1/10$

(c) $\lambda_{on} = 1/3, \lambda_{off} = 1/11$

(d) $\lambda_{on} = 1, \lambda_{off} = 1/15$

Fig. 5. The accuracy of *SS* prediction

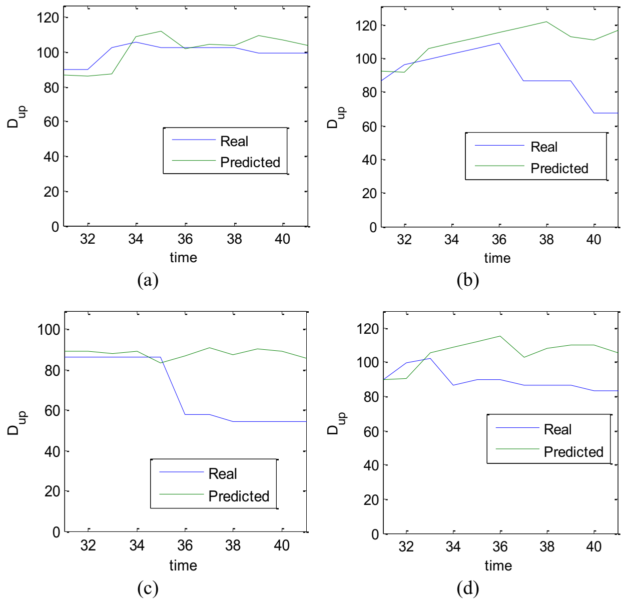
2) Prediction accuracy for D_{up}

In the same simulation runs that generate the results in Fig. 5, the actual values of D_{up} are also collected and compared with the predicted counterparts. These results are plotted in Fig. 6, where the experimental settings in Fig. 6(a)-(d) are the same as those in Fig. 5(a)-(d). It can be seen that the D_{up} prediction is rather accurate in most cases.

VI. Conclusions

This paper proposes a Cloud-assisted data replication and storage service, called Cadros, for DOSN, aiming to improve the data availability of DOSN. This paper first conducts the quantitative analysis about the storage capacity as the result of combining the Cloud with DOSN. Further, this paper models and predicts the level of DA that Cadros is able to achieve.

Erasure coding can save storage space. However, it incurs the overhead for coding and reconstructing the data. More data are stored using erasure coding, higher overhead. Therefore, one of our future work is to develop a data partition scheme in terms of the replication techniques, i.e., decide the portion of published data that should be stored using full replication or erasure coding, so that the erasure coding overhead is minimized. Furthermore, the data availability model only indicates that the hybrid system has the capacity to achieve such a certain level of data availability. It still depends on the underlying data placement strategies to realize the data availability. Therefore, another future work is to develop the placement strategies for data replicas so that the predicted DA can be realized.



(a) (b) (c) (d)

Fig. 6. The accuracy of D_{up} prediction; the experimental settings are the same as those in Fig. 5.

ACKNOWLEDGMENT

We would like to thank the users and the developer community for their help with this work. The work reported in this paper was supported by China HGJ Project (No. 2013ZX01040-002), China Open Fund Project (No. KJ-13-105), and China NSFC No.61304119, No.61303191, No.61272482, No.61272483 and No.61379146.

REFERENCES

[1] Graffi K, Gross C, Mukherjee P, et al. LifeSocial.KOM: A P2P-based platform for secure online social networks[C]//Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on. IEEE, 2010.

[2] Buchegger S, Schiöberg D, Vu L H, et al. PeerSoN: P2P social networking: early experiences and insights[C]// the Second ACM EuroSys Workshop on Social Network Systems. ACM, 2009: 46-52.

[3] Yeung C A, Liccardi I, Lu K, et al. Decentralization: The future of online social networking[C]//W3C Workshop on the Future of Social Networking Position Papers, 2009.

[4] Tandukar U, Vassileva J. Selective propagation of social data in decentralized online social network[M]//Advances in User Modeling. Springer Berlin Heidelberg, 2012: 213-224.

[5] Shakimov A, Varshavsky A, Cox L P, et al. Privacy, cost, and availability tradeoffs in decentralized OSNs[C]// the 2nd ACM workshop on Online social networks. ACM, 2009: 13-18.

[6] David Koll, Jun Li, Xiaoming Fu, With a Little Help From my Friends: Replica Placement in Decentralized Online Social Networks, Technical Report, University of Goettingen, Germany, January 2013.

[7] Olteanu A, Pierre G. Towards robust and scalable peer-to-peer social networks[C]//Proceedings of the Fifth Workshop on Social Network Systems (WOSN). ACM, 2012.

[8] Cutillo L A, Molva R, Strufe T. Safebook: A privacy-preserving online social network leveraging on real-life trust[J]. Communications Magazine, IEEE, 2009, 47(12): 94-101.

[9] Li J, Dabek F. F2F: Reliable Storage in Open Networks[C]//IPTPS. 2006.

[10] Tran D N, Chiang F, Li J. Friendstore: cooperative online backup using trusted nodes[C]//Proceedings of the 1st Workshop on Social Network Systems. ACM, 2008: 37-42.

[11] Gracia-Tinedo R, Sánchez-Artigas M, Garcia-Lopez P. F2BOX: Cloudifying F2F Storage Systems with High Availability Correlation[C]//Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on. IEEE, 2012: 123-130.

[12] Liu D, Shakimov A, Cáceres R, et al. Confidant: Protecting OSN data without locking it up[M]//Middleware 2011. Springer Berlin Heidelberg, 2011: 61-80.

[13] Sun Y, Liu F, Li B, et al. Fs2you: Peer-assisted semi-persistent online storage at a large scale[C]//INFOCOM 2009, IEEE, 2009: 873-881.

[14] Krishnamurthy B, Wills C E. Characterizing privacy in online social networks[C]//Proceedings of the first workshop on Online social networks. ACM, 2008: 37-42.

[15] Zhang C, Sun J, Zhu X, et al. Privacy and security for online social networks: challenges and opportunities[J]. Network, IEEE, 2010, 24(4).

[16] Mega G, Montresor A, Picco G P. Cloud-assisted dissemination in social overlays[C]//Peer-to-Peer Computing (P2P), IEEE, 2013: 1-5.

[17] Diaspora, https://joindiaspora.com/

[18] Facebook, https://www.facebook.com/

[19] Sina Microblog, http:// weibo.com/

[20] Weatherspoon H, Kubiatowicz J D. Erasure coding vs. replication: A quantitative comparison[M]//Peer-to-Peer Systems. Springer Berlin Heidelberg, 2002: 328-337.

[21] Jin L, Chen Y, et al. Understanding user behavior in online social networks: A survey[J]. IEEE Communications Magazine, 2013: 144-150.

[22] Benevenuto F, Rodrigues T, Cha M, et al. Characterizing user behavior in online social networks[C] //Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference. ACM, 2009: 49-62.

[23] McGlohon M, Akoglu L, Faloutsos C. Statistical properties of social networks[M]//Social Network Data Analytics. Springer US, 2011: 17-42.

[24] R. E. Wilson, S. D. gosling, L. T. Graham, A Review of Facebook Research in the Social Sciences, Perspectives on Psychological Science, 7(3): 203-220, May 2012.

[25] Mislove A, Marcon M, Gummadi K P, et al. Measurement and analysis of online social networks[C]//Proceedings of the 7th ACM SIGCOMM conference on Internet measurement. ACM, 2007: 29-42.

[26] Barabasi A L. The origin of bursts and heavy tails in human dynamics[J]. Nature, 2005, 435(7039): 207-211.

[27] Zhou T, Han P, et al. Towards the understanding of human dynamics[J]. Science matters: humanities as complex systems, 2008: 207-233.

[28] Stutzbach D, Rejaie R. Understanding churn in peer-to-peer networks[C] //Proceedings of the 6th ACM SIGCOMM conference on Internet measurement. ACM, 2006: 189-202.

[29] Tegeler F, Koll D, Fu X. Gemstone: Empowering Decentralized Social Networking with High Data Availability[C] //Global Telecommunications Conference (GLOBECOM 2011). IEEE, 2011: 1-6.

[30] Krishnamurthy B, Wills C E. Privacy leakage in mobile online social networks[C]//Proceedings of the 3rd conference on Online social networks. USENIX Association, 2010.

[31] Sharma R, Datta A, et al. An empirical study of availability in friend-to-friend storage systems[C]//Peer-to-Peer Computing (P2P), IEEE, 2011: 348-351.

[32] Rzadca K, Datta A, et al. Replica placement in p2p storage: Complexity and game theoretic analyses[C]//Distributed Computing Systems (ICDCS), 30th International Conference on. IEEE, 2010: 599-609.