Privacy, Cost, and Availability Tradeoffs in Decentralized OSNs

Amre Shakimov
Duke University
Durham, NC
shan@cs.duke.edu

Alexander Varshavsky
AT&T Labs
Florham Park, NJ
varshavsky@research.att.com

Landon P. Cox
Duke University
Durham, NC
lpcox@cs.duke.edu

Ramón Cáceres
AT&T Labs
Florham Park, NJ
ramon@research.att.com

ABSTRACT

Online Social Networks (OSNs) have become enormously popular. However, two aspects of many current OSNs have important implications with regards to privacy: their centralized nature and their acquisition of rights to users' data. Recent work has proposed decentralized OSNs as more privacy-preserving alternatives to the prevailing OSN model. We present three schemes for decentralized OSNs. In all three, each user stores his own personal data in his own machine, which we term a Virtual Individual Server (VIS). VISs self-organize into peer-to-peer overlay networks, one overlay per social group with which the VIS owner wishes to share information. The schemes differ in where VISs and data reside: (a) on a virtualized utility computing infrastructure in the cloud, (b) on desktop machines augmented with socially-informed data replication, and (c) on desktop machines during normal operation, with failover to a standby virtual machine in the cloud when the primary VIS becomes unavailable. We focus on tradeoffs between these schemes in the areas of privacy, cost, and availability.

Categories and Subject Descriptors

C.0 [**General**]: System Architectures; D.4.6 [**Operating Systems**]: Security and Protection—*Access Controls*

General Terms

Design, Performance, Reliability, Security

Keywords

Cloud computing, online social networks, privacy, replication, utility computing, virtual machines

1. INTRODUCTION

The popularity of Online Social Networks (OSNs) has experienced unprecedented growth. For example, Facebook [9] attracted over 130 million unique visitors in June 2008, compared to 52 million a year earlier [7]. It had more than 200 million users as of May 2009. Odnoklassniki [14] and Vkontakte [20] have acquired more than 60 million users from the Russian-speaking sector of the Internet alone.

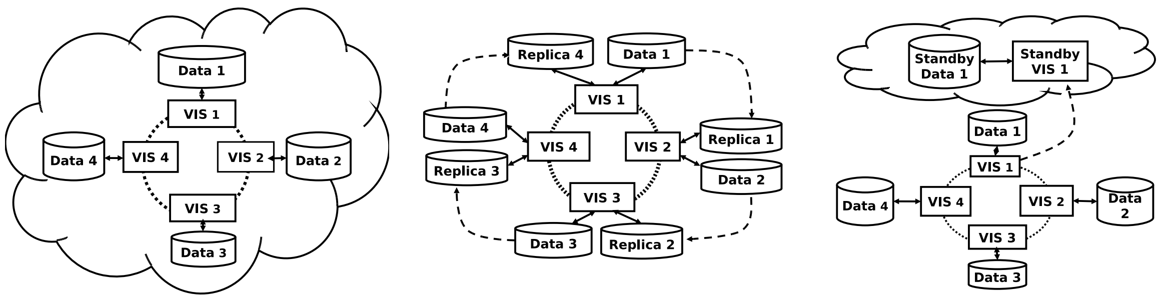
As a result of this popularity, OSN services have amassed large amounts of information about their users, including personal profiles, friend relationships, daily activities, and photos. For instance, Facebook has become the largest photo-sharing service on the Internet, outgrowing even dedicated photo services like Flickr and Photobucket [7].

Unfortunately, two characteristics of current OSNs raise important privacy concerns. One, most OSNs concentrate the data of all their users under a single administrative domain. This concentration makes them vulnerable to large-scale privacy breaches from intentional and unintentional data disclosures. For example, sensitive user data may be divulged directly to users' social connections [19]. Two, the terms of service of many OSNs grant service providers rights to users' data. For example, these rights commonly include a license to display and distribute all content posted by users in any way the provider sees fit [9][11].

Recent work [5] [8], including our own [18], has proposed the use of decentralized OSNs as more privacy-preserving alternatives to the prevailing model. Decentralized OSNs distribute users' personal data across multiple administrative domains and thus reduce the chance of large-scale privacy breaches. In addition, they operate in a peer-to-peer fashion that gives users more control over their own data.

Decentralized OSNs present tradeoffs that bear further study. In particular, most centralized OSNs are free to users. Free services have obvious appeal, but they give rise to commercial pressures on providers to share users' data in ways that may diminish user privacy. In contrast, peer-to-peer OSNs generally require that users pay for some of the computing resources they use. We feel it is important to explore alternatives to the OSN status quo even though these alternatives may cost money to users, especially as popular awareness of privacy issues grows and the price of computing drops.

In this paper, we present three architectural alternatives for decentralized OSNs and compare their privacy, cost, and availability tradeoffs. Those alternatives are:



(a) Cloud (b) Desktop with socially-informed replication (c) Hybrid of desktop and cloud

Figure 1: Alternative approaches to decentralized OSNs. In all three, users store their own personal data in their own Virtual Individual Servers (VISs), which communicate in a peer-to-peer fashion. The approaches differ in where VISs and data reside, which presents privacy, cost, and availability tradeoffs.

- Cloud-based decentralization.
- Desktop-based decentralization with socially-informed replication of user data.
- A hybrid of cloud- and desktop-based decentralization.

Figure 1 shows a visual representation of the three architectures. In all three, each user stores her own personal data in her own *Virtual Individual Server (VIS)*. VISs self-organize into peer-to-peer overlay networks, one overlay per social group with which the VIS owner wishes to share information. This structure supports many of the social networking features provided by popular OSNs, such as forming groups, finding friends, exchanging messages, etc. We use the word "virtual" because VISs can take the form of virtual machines, which offer important manageability and other advantages when compared to physical machines.

The three architectures differ in the placement of VISs and personal data. In cloud-based decentralization, VISs are hosted by a utility computing infrastructure such as Amazon Elastic Compute Cloud (EC2) [2]. The main advantage of this scheme is its high availability. Unfortunately, hosting a VIS in a cloud is currently costly. Continuously running a virtual machine at EC2 costs upwards of US$75 per month.

In desktop-based decentralization, VISs run on desktop-class machines owned by OSN users. This solution has lower cost, but suffers from lower availability due to the high churn of desktop machines [13]. To increase availability, we propose a novel socially-informed replication scheme. The main insight behind this scheme is that users may be willing to replicate some of their personal data on machines belonging to social connections who would in any case have access to the data through normal OSN operations.

Hybrid decentralization is a combination of desktop-based and cloud-based decentralization. During normal operation, a user's data is served from a VIS running in the user's own desktop machine. When this machine becomes unavailable, a standby VIS is resumed inside the utility computing infrastructure. This solution could combine high availability with low cost because the cloud-hosted VIS would provide a stable backup while remaining quiescent most of the time.

The rest of this paper discusses these three solutions in further detail. It also presents our ongoing implementation and evaluation efforts, and surveys related work.

2. CLOUD-BASED DECENTRALIZATION

In a recent paper, we proposed Vis-à-Vis [18], a decentralized approach to online social networking. In Vis-à-Vis, each person stores his own data on a personal virtual machine instance called a Virtual Individual Server (VIS). VISs self-organize into structured overlay networks that represent OSN groups, with one overlay per group. The same VIS can belong to multiple overlay networks, just as one person can belong to multiple social groups. VISs communicate with each other as peers and are free to reside anywhere in the Internet.

In Vis-à-Vis, VISs run in the cloud, more specifically in a paid utility computing environment such as Amazon EC2 [2]. We believe that individual consumers will adopt virtualized utility computing for many of the same reasons that enterprises have: it unburdens them from maintaining their own high-availability hardware without forcing them to give up control of their data, software, and policies. In contrast to free OSN services, paid utility computing services allow users to retain full rights to the content and applications that users place on these services [4].

Vis-à-Vis is based on a two-tier Distributed Hash Table (DHT) structure composed of a set of highly available VISs. The top-tier DHT, called the Meta Group, is used to advertise and search for public OSN groups. The lower-tier DHTs correspond to OSN groups and are maintained by the VISs of the group members. This requires VISs to maintain routing state for the Meta Group and every group of which they are members. Figure 2 shows a Vis-à-Vis network of eight VISs and three groups. Group 1 is composed of VISs A, B, E, and H; Group 2 is composed of A, C, D, F, and H; and Group 3 is composed of B, G, and H. All VISs are members of the Meta Group.

Vis-à-Vis offers a number of attractive features:

- It supports open and restricted groups.
- It supports public and secret groups.
- It scales to both very small and very large groups.
- It supports common OSN operations such as finding groups of interest, joining a group, leaving a group, and searching for information within a group.

Experimental results using our prototype implementation indicate that Vis-à-Vis is a viable alternative to the centralized OSN architecture. The latency of common OSN operations grows slowly, if at all, with the size of the corresponding OSN group. Similarly, the memory required by a VIS to participate in Vis-à-Vis is manageable and grows with the size and number of OSN groups to which a user belongs. It is important to note that the highly available nature of cloud-hosted VISs has the potential to mitigate some of the scalability problems seen in more general DHT deployments. For example, the frequency of DHT maintenance messages can be kept low since nodes can be expected to fail rarely.

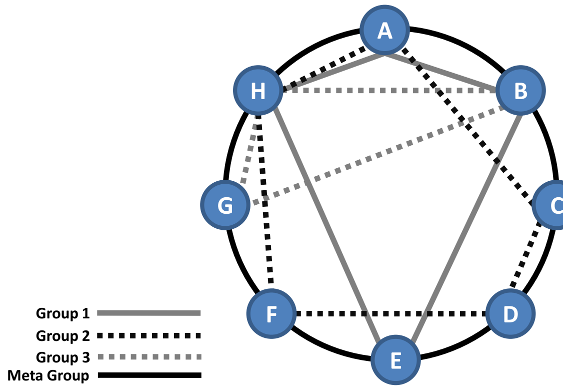


Figure 2: Example Vis-à-Vis Network with eight Virtual Individual Servers and three groups.

The main drawback of Vis-à-Vis is its cost. At the moment, Amazon EC2 charges ten cents per hour for a default virtual machine with 1.7 GB of memory, 1 virtual core, and 160 GB of persistent storage. Data-transfer fees vary depending on the location of the machines with which the virtual machine is communicating. Even without network-usage fees, running a virtual machine for one month costs close to US$75.

3. DESKTOP-BASED DECENTRALIZATION

A cheaper alternative to cloud-based decentralization is to place VISs on desktop machines belonging to OSN users. These machines may reside, for example, at homes or offices. A desktop machine consumes significantly less than US$75 of energy per month and can share a flat-rate Internet connection with other machines. In addition, hosting personal data on machines that users physically control is more privacy-preserving than delegating control to a third-party service provider, regardless of the terms of service.

The main drawback of this approach is the low availability of desktop machines. Desktop computers are prone to failures, reboots, power-offs, and network disconnections. Providing reasonable availability under the churn characteristic of these machines requires data to be replicated. However, because of its sensitive nature, OSN data must be replicated carefully.

One strategy is to replicate encrypted data on potentially untrusted hosts and distribute decryption keys to authorized clients. Consider FARSITE [1], a representative peer-to-peer file system. In FARSITE, each user has a public-private key pair and each replicated file is encrypted using a unique symmetric key. Each file's symmetric key is encrypted with the public keys of all users authorized to access the file and embedded in the file's meta-data. To access a file, authorized users can decrypt the file's symmetric key with their private key, and then use the symmetric key to decrypt the file's content.

The drawbacks of applying this scheme to a decentralized OSN is twofold: (1) relying on a centralized PKI to manage public keys undercuts the cost advantage of desktop-based decentralization, and (2) each user's private key must be accessible to any client software used to access OSN data, including web browsers and mobile-phone applications.

An alternative approach is to apply a *socially-informed replication scheme*. Under such a scheme, users can take advantage of the trust embedded in the social network to replicate cleartext data on the VISs of friends or relatives. To ensure that only authorized users can access sensitive data, replica sites would be trusted to enforce owner-specified access-control policies. The primary appeal of this approach is that it has the potential to greatly simplify key management and support a wider range of protection mechanisms than relying on encryption alone.

For example, by default in Facebook whenever a person is tagged in a photograph, all of their friends are automatically granted read access to the image and the album that contains it. To support this level of sharing under a FARSITE-like encryption scheme, the image owner would have to encrypt the symmetric key for the album with the public key of their own friends as well as all the keys of the friends of anyone tagged in the image. If a tagged person's friend list changed, the image owner would have to subsequently add or delete keys from the album's meta-data. On the other hand, under socially-informed replication, access to an album replica could be regulated in more general terms, such as granting read-write access to "friends of Alice." If Alice is tagged in an image, her friends would only have to present a credential testifying to their friendship with Alice in order to access the album.

A non-trivial challenge of replicating OSN data on friends' machines arises from friends' non-overlapping social connections. For instance, a user may not want to store her private communication with a boyfriend on her brother's desktop. Therefore, any pair-wise shared secrets between friends cannot be replicated on desktop machines of third parties without adequate encryption. Furthermore, aggregrated data such as the Facebook news feed that collects updates from a user's friends can only be replicated at hosts that are authorized to access each of these data feeds. In general, we believe that using social information to appropriately replicate sensitive data represents a challenging new research problem.

4. HYBRID DECENTRALIZATION

The main disadvantage of hosting VISs in desktop machines is the low availability of these machines. Although socially-informed replication mitigates this problem, nodes holding replicas may still experience correlated failures. For example, if a user replicates her data on desktop machines that are located in the same household, all replicas may be down if the house loses electricity.

A hybrid decentralization architecture allows us to trade off cost for availability. The main idea behind the hybrid approach is to use a cheap desktop machine for normal operation, and fall back to a more expensive but highly-available *standby VIS* that runs in the cloud only when both primary and replica copies of data are unavailable. To simplify the following discussion, we will assume that socially-informed replication is not in use when the primary VIS fails, though we believe the scheme can be extended to handle replicas as well.

We envision that in the course of normal operation, all updates to the primary copy of data will also be propagated to a highly-available storage facility in the cloud, such as Amazon's Simple Storage Service (S3) [3]. Accessing storage is far less expensive than executing a virtual machine. On failure of the primary VIS, the standby VIS can resume using the copy of the data stored in the cloud. Techniques for efficiently migrating virtual-machine state [17] can be used to propagate updates from a primary, desktop VIS to cloud storage.

A key challenge is identifying how to detect the failure of a primary VIS and resume its standby counterpart. Assuming that the utility allows virtual machines to be instantiated remotely, we envision three ways in which a failure of the primary VIS could potentially be detected. First, a utility provider could offer an inexpensive subscription-based service that would periodically probe the primary VISs of subscribed users and instantiate their standby VISs when needed. Second, any node that tries and fails to contact

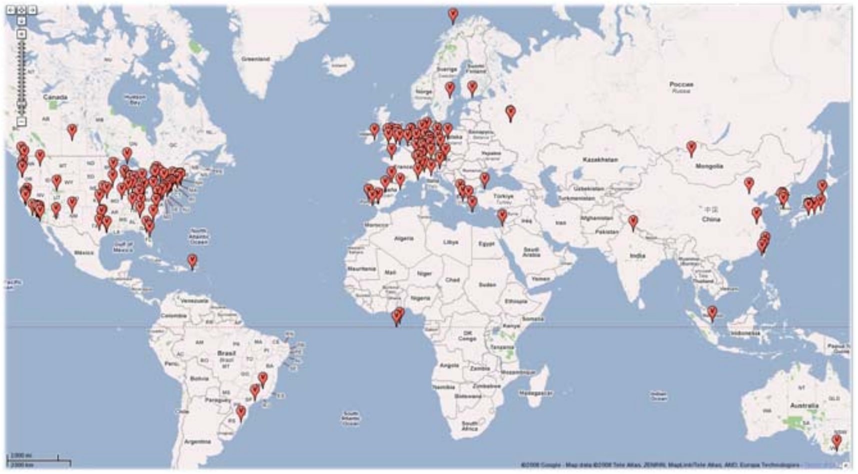


Figure 3: Geographic distribution of the 120 Virtual Individual Servers in our PlanetLab experiments.

a primary VIS may initiate the process of starting the standby VIS. In such cases, the utility may need to verify that the primary VIS is actually unavailable before starting the standby VIS. Third, VISs belonging to the same social group could collectively monitor each other and start the standby VIS of any failed VISs in the group. An attractive feature of this technique is that it can be piggybacked on the keep-alive messages already in use to maintain the underlying DHT.

5. INITIAL EVALUATION

Our current cloud-based Vis-à-Vis prototype provides basic OSN features: joining and leaving groups, creating groups with different privacy policies, sending messages, searching for friends, and browsing profiles. VISs are organized using an implementation of the Pastry DHT [16], and use Scribe [6] to provide multicast functionality for groups that require it. Our Vis-à-Vis prototype allows flexible group admission policies ranging from one in which anyone can join, which we call an *Open Group*, to a restricted one in which a majority vote of the group's members must approve admission, which we call a *Closed Group*.

VISs are implemented as a Xen virtual-machine image that uses Apache Tomcat for simple administrative tasks, messaging, and other basic functionality. Users' data is stored in XML files. We have successfully run VISs in a variety of virtualized computing environments, including Amazon EC2, Emulab, PlanetLab, and experimental utility computing facilities at AT&T Labs and Duke University. In this section we report experiments run on Emulab and PlanetLab.

5.1 Experimental Setup

Emulab [21] is a network testbed that provides resources for conducting experiments on distributed systems. All of the resources are located on a single site, the University of Utah. We provisioned 105 Emulab nodes for our initial experiments. On each node we installed a pre-built VIS image containing our Vis-à-Vis prototype software, Apache Tomcat 6.0.18, and Java JDK 1.5. We used a simple topology where all nodes are connected to a single 100Mbps network switch, without artificial latency and losses. We used the experimental results from this simple topology as a baseline with which to compare our results from PlanetLab, which capture many complexities of real-world networks.

PlanetLab [15] is a world-wide research network for the deployment of distributed systems. Unlike with Emulab, PlanetLab nodes are geographically distributed. PlanetLab experiments thus capture the variable bandwidth, latency, and packet losses of the live Internet. In addition to network variability, PlanetLab experiments also capture the effects of load placed on the test nodes themselves by the many experiments running on PlanetLab at any one time. For example, during our experiments the average CPU load on each node was greater than 10, as reported by the Linux *top* command.

For our PlanetLab experiments we provisioned 120 VISs distributed throughout the globe, as seen in Figure 3. On each VIS we installed the same software that we installed on our Emulab VISs, namely our Vis-à-Vis prototype software, Apache Tomcat 6.0.18, and Java JDK 1.5.

5.2 Vis-à-Vis Performance Characteristics

The time Vis-à-Vis takes to complete OSN operations is important to the Vis-à-Vis user experience. As a result, we measured the latency of the following example operations as we varied the group size: joining an Open Group, joining a Closed Group, and searching for information in a group. For each of these sample operations, we created a Meta Group consisting of all available VISs. Every operation was conducted 5 times on a group size of 20, 40, 60, 80, and 100 VISs randomly chosen from our provisioned VISs. These group sizes are meaningful because our prior work on characterizing groups within Facebook found an average group size of close to 250 [18].

Figures 4, 5, and 6 report the average of these latency results along with their standard deviations. It should be noted that OSN latency requirements are relatively lax because OSNs are meant to support social interactions. Thus, it is acceptable for a group-join operation to take many seconds to complete, even hours or days when human approval of a join request is involved.

5.2.1 Joining an Open Group

For this experiment, a node chosen randomly from the Meta Group tried to join an Open Group via a randomly-chosen member of the Open Group. We measured the time between when the candidate node sent the join request and when it became a member of the group. The results are shown in Figure 4.

This figure shows that join latency for this type of group does not grow appreciably as the group size grows from 20 to 100. The Emulab results are quite stable, and even with 100 nodes the latency

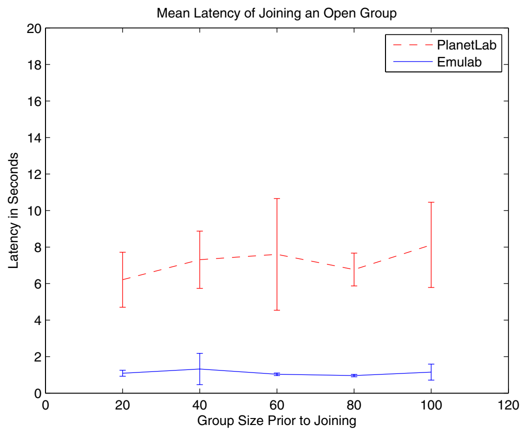


Figure 4: Mean latency of joining an Open Group in Emulab and PlanetLab. Error bars show the standard deviation.

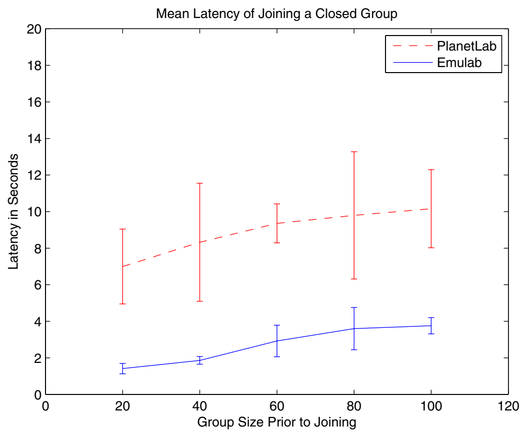


Figure 5: Mean latency of joining a Closed Group in Emulab and PlanetLab. Error bars show the standard deviation.

is still less than 2 seconds. The PlanetLab results exhibit more variability due to the varying network characteristics and testbed loads discussed earlier. Moreover, the mean latency of joining an Open Group of size 100 is only at most 2 seconds longer than the mean latency of joining an Open Group of size 20. Overall, these results are promising and suggest that the join operation will scale well to larger group sizes.

5.2.2 *Joining a Closed Group*

For this experiment we randomly picked one node from the Meta Group and had it send a join request to a randomly chosen member of a Closed Group. The Closed Group required all members to vote on whether to admit a new candidate node to the group. The Closed Group also required all members to be notified as to whether the join request was granted or not. Because both of these configuration choices require that all group members be contacted before the join operation can complete, our Closed Group can be thought of as the worst case for join latency. Figure 5 shows the measured join latency, not counting the human think time that might be required to allow group members to enter their votes.

As shown in Figure 5, the latency of joining a Closed Group grows slowly with the size of the group. In both the Emulab and PlanetLab experiments, the mean latency of joining a Closed Group of size 100 is at most 2 seconds longer than joining a Closed Group of size 20. This growth is due to those configuration choices that require all members to be contacted as part of the join operation. Groups created with less burdensome configurations will perform better, as in the Open Group experiments.

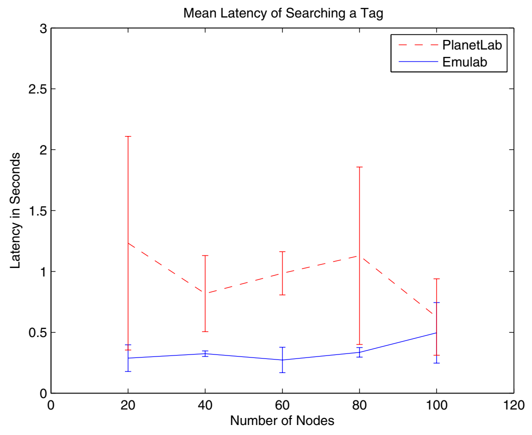


Figure 6: Mean latency of searching for a data item in Emulab and PlanetLab. Error bars show the standard deviation.

5.2.3 *Finding and Retrieving Information in a Group*

The goal of this experiment was to measure the latency of searching for a keyword within a previously created group. We measured the time from when a randomly chosen member of a group submitted a query containing a keyword to when it received the data associated with this keyword. More concretely, during our experiment the randomly chosen node searched for all data associated with randomly generated and previously advertised keywords.

Figure 6 shows that search latency is not greatly affected by group size, as desired. As before, the Emulab results were very stable because they were taken in a controlled laboratory environment, while the PlanetLab results exhibit greater variability because of external factors. In both testbeds, the mean latency of searching for a data item is less than 1.5 seconds.

In the future we plan to compare the performance of the three alternative architectures for decentralized OSNs that we have discussed in this paper, both to each other and to a centralized OSN implementation.

6. RELATED WORK

PeerSon [5] and [8] are two recent proposals for peer-to-peer OSNs with similar goals and structures as Vis-à-Vis. PeerSon relies on encryption to protect users' replicated data, and uses a two-tier architecture in which the the first tier is organized as a DHT for lookups. The second tier allows users to exchange data and is implemented as a strongly-connected overlay. [8] also utilizes a DHT for lookups and storing social groups' information. In this scheme, logical "matryoshkas" store data and are organized as concentric "rings of friends". Profiles are replicated in the innermost layer of a user's matryoshka to increase availability of the data. The primary difference between these projects and Vis-à-Vis is that neither relies on a compute utility to improve service availability. In addition, neither considers using a socially-informed replication scheme to simplify key management.

NOYB [10] takes a different approach to the privacy threats of centralized OSN services by encrypting some of the data that users hand to the services. This approach is appealing because it may allow users of existing OSNs to retain their existing profiles, while Vis-à-Vis requires users to divest themselves from existing OSNs. However, NOYB has several drawbacks compared to Vis-à-Vis. First, retaining any presence in a centralized OSN leaves users open to the service directly notifying a user's friends of potentially sensitive activity in other corners of the Internet [19]. Vis-à-Vis users are not vulnerable to such attacks because their VISs control all data sent along their social links. Second, NOYB requires additional key-managing software to be installed on any client machine accessing encrypted profile data, including public kiosks and mobile phones where it may not be convenient. Vis-à-Vis only requires clients to have a web browser. Finally, it is unclear how well NOYB generalizes to non-textual information, while Vis-à-Vis can secure arbitrary data types.

Like NOYB, flyByNight [12] uses encryption to ensure that sensitive data is never posted to OSNs in unencrypted form. flyByNight is also appealing because it allows users to continue using existing OSNs and the social state that users have accumulated there. However, flyByNight remains vulnerable to several types of attack from within the OSN, which Vis-à-Vis avoids by doing away with centralized OSNs.

7. CONCLUSION

In this paper we discussed and examined tradeoffs between three different decentralized architectures for Online Social Networks. All three alternatives offer improved privacy with respect to centralized OSN architectures by avoiding large-scale privacy breaches and allowing users to retain full rights to their personal data.

In the first approach, a user keeps her data in a personal virtual machine residing in a utility computing environment, e.g. Amazon EC2. This approach provides high availability in exchange for higher cost to the user. Improved privacy comes partly from the difference in terms of service between paid utility computing services and free OSNs.

The second approach uses desktop-class computers augmented with a socially informed data replication strategy. The idea is to store user's data on desktops of close friends and relatives such that they could serve incoming requests if the primary host has failed. This approach is cheaper than the first. However, it incurs considerably more complexity and may not provide the desired availability because trust issues restrict the number of viable replica sites and socially-linked sites may fail concurrently.

The third approach is a hybrid of the first two: most of the time the user's data is served from the user's desktop machine, but the scheme fails over to a personal virtual machine running in the cloud when the desktop machine becomes unavailable. This approach has the potential of combining low cost and high availability.

We feel it is important to explore such privacy-preserving alternatives to prevailing OSNs. We plan to more fully evaluate the above privacy, cost, and availability tradeoffs in our ongoing work.

8. REFERENCES

[1] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, Jon, J. Howell, J. R. Lorch, M. Theimer, and R. P. Wattenhofer. Farsite: Federated, available, and reliable storage for an incompletely trusted environment. In *Proc. of OSDI*, 2002.

[2] Amazon Elastic Compute Cloud (EC2). `http://aws.amazon.com/ec2/`.

[3] Amazon Simple Storage Service (S3). `http://aws.amazon.com/s3/`.

[4] Amazon Web Services Customer Agreement. `http://aws.amazon.com/ec2/agreement/`.

[5] S. Buchegger and A. Datta. A Case for P2P Infrastructure for Social Networks - Opportunities and Challenges. In *WONS 2009, 6th International Conference on Wireless On-demand Network Systems and Services, February 2-4, 2009, Snowbird, Utah, USA*, Feb.

[6] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. Scalable application-level anycast for highly dynamic groups. In *Proc. of NGC*, 2003.

[7] comScore. `http://www.comscore.com`.

[8] L. A. Cutillo, R. Molva, and T. Strufe. Privacy preserving social networking through decentralization. In *WONS 2009, 6th International Conference on Wireless On-demand Network Systems and Services, February 2-4, 2009, Snowbird, Utah, USA*, Feb 2009.

[9] Facebook. `http://www.facebook.com`.

[10] S. Guha, K. Tang, and P. Francis. NOYB: Privacy in online social networks. In *Proc. of WOSN*, 2008.

[11] LinkedIn. `http://www.linkedin.com`.

[12] M. M. Lucas and N. Borisov. flyByNight: Mitigating the privacy risks of social networking. In *Proc. of WPES*, 2008.

[13] J. Mickens and B. Noble. Exploiting availability prediction in distributed systems. In *Proc. of NSDI*, 2006.

[14] Odnoklassniki. `http://www.odnoklassniki.ru`.

[15] PlanetLab. `http://www.planet-lab.org`.

[16] A. I. T. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proc. of Middleware*, 2001.

[17] M. Satyanarayanan and et al. Pervasive personal computing in an Internet Suspend/Resume system. *IEEE Internet Computing*, 6(4), 2007.

[18] A. Shakimov, H. Lim, L. P. Cox, and R. Cáceres. Vis-à-Vis: Online Social Networking via Virtual Individual Servers. Duke University Technical Report TR-2008-05, October 2008.

[19] L. Story and B. Stone. Facebook retreats on online tracking. The New York Times. November 2007.

[20] Vkontakte. `http://www.vkontakte.ru`.

[21] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An Integrated Experimental Environment for Distributed Systems and Networks. *Proc. of OSDI 2002*.