ABnews: A fast private social messaging system using untrusted storage and attribute-based encryption

Yasushi Shinjo, Sota Naito, Xiao Kunyao, Akira Sato
Department of Computer Science
University of Tsukuba
1-1-1 Tennoudai, Tsukuba, Ibaraki 305-8573, Japan
Email: yas@cs.tsukuba.ac.jp

***Abstract*—Centralized social networking services (SNSs) or online social networks (OSNs) inherently have privacy concerns. Ciphertext-policy attribute-based encryption (CP-ABE) is an effective and promising tool for protecting privacy in centralized and distributed systems. However, a large effort is required to develop new practical social applications using CP-ABE. Furthermore, such applications often have performance problems because of the substantial computation time needed for ABE. This paper describes the design and implementation of the ABnews system, a Usenet-like social messaging system using ABE and untrusted storage. ABnews is fast because it eliminates the heavy computation of ABE from the interactive access to messages. ABnews inherits application programs from Usenet, and allows Usenet's rich newsreaders to be utilized without any modification. Furthermore, ABnews provides overlay social applications, including a private blogging service, a presence service, and one-to-one direct messaging, on top of bulletin board systems. The ABnews system has been implemented using the cpabe toolkit and Google Drive, which holds encrypted messages. Experimental results show that the overhead of CP-ABE is negligible when implementing such a messaging system on a current cloud storage service.**

I. Introduction

People use social networking services (SNSs) or online social networks (OSNs) every day. Because current popular SNSs have a centralized architecture, they have technical and social problems [14], [39]. One significant technical problem is the potential lack of scalability. Notable social problems include those related to user privacy and trust. Users must trust the central servers when they upload sensitive and private data. It is not trivial for casual users to maintain access control lists (ACLs) of the resources in central servers to protect their privacy [31]. Some recent events have proved the existence of such privacy concerns in widely-used centralized SNSs [21]. In addition to privacy and trust issues, users must consider the possibility of losing access to their data in the centralized servers if these servers shut down.

To address the problems associated with centralized servers, researchers are working on building decentralized SNSs (DSNSs) or decentralized OSNs (DOSNs) [14], [39]. After ciphertext-policy attribute-based encryption (CP-ABE) was introduced [7], [8], researchers began implementing decentralized SNSs using CP-ABE [4], [34], [50]. CP-ABE has proven to be an effective and promising tool for fine-grained access control to distributed shared data. However, there are still problems to solve in implementing social applications with CP-ABE. First, a large effort is required to develop new practical social applications that have good user interfaces. For example, Persona [4] requires modifications to Web browsers, and this occasionally prevents people from using Persona. Second, existing decentralized SNSs with ABE are attempting to produce user experiences very similar to those of centralized SNSs, and such reproduction is often cumbersome. For example, Persona implements Facebook Wall, and Cachet implements Facebook Newsfeed. These applications are cumbersome because of the substantial computation time needed for ABE and the large number of accesses to distributed hash tables (DHTs).

In this paper, we describe the design and implementation of a social messaging system that is suitable for use with ABE. The contributions of this paper are as follows.

- We allow users to exchange private messages using existing rich applications including mobile ones without any modification.
- We eliminate the heavy computation of ABE from the interactive access to messages.
- We show the implementation of overlay social applications, including a private blogging service, a presence service, and one-to-one direct messaging by making maximum use of the fine-grained access control of ABE.
- We take back control of our data from centralized SNSs.

Concretely, we realize a Usenet-like messaging system by using untrusted storage and attribute-based encryption (Section II). We call this system the Attribute-Based news system, or ABnews for short. ABnews inherits application programs from Usenet, and allows Usenet's rich newsreaders to be utilized without any modification. Users can exchange messages by using standard newsreaders that have good user interfaces. Furthermore, ABnews provides overlay social applications, including a private blogging service, a presence service, and one-to-one direct messaging, on top of BBSs (Section III).

ABnews is very fast because it accesses plaintext messages

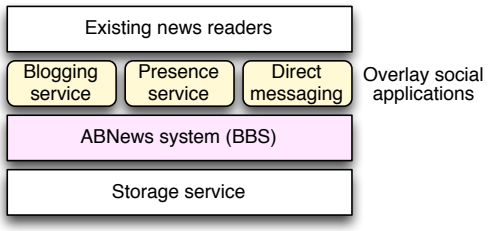

Fig. 1. Layered architecture of the ABnews system.

in a local spool, and performs neither decryption nor Internet access when users access these messages interactively. We have implemented the ABnews system using the cpabe toolkit [7], [8] and Google Drive [19], which holds messages encrypted with ABE (Section IV). Experimental results show that the overhead of CP-ABE is negligible when implementing such a messaging system on a current cloud storage service (Section V).

II. ABnews System Design

In this section, we describe the design of our ABnews system. We build an SNS using four layers, as shown in Fig. 1. We implement the ABnews system on top of a cloud storage service. ABnews provides a BBS service as Usenet does. ABnews inherits existing rich newsreaders from Usenet. On top of the BBS service, we implement other overlay social applications, including a private blogging service, a presence service, and one-to-one direct messaging.

In this section, we briefly describe Usenet and its important ideas that are used in our news system. Next, we discuss the goals, implementation environment, and threat model of our news system. Finally, we describe the base social application, a BBS. In the next section, we will describe the overlay social applications.

A. Usenet

Usenet is a worldwide distributed BBS on the Internet. In Usenet, a message is called an *article*. Its main goal is to multicast an article posted on a node to all nodes worldwide. A software suite that transfers and manages Usenet messages is called a news system; examples include A-News, B-News, C-News, and InterNetNews (INN) [44]. In ABnews, we reuse the same message format.

The articles in Usenet are classified into hierarchical newsgroups. The major top-level newsgroup hierarchies are called the Usenet Big 8: `comp.*`, `humanities.*`, `misc.*`, `news.*`, `rec.*`, `sci.*`, `soc.*`, and `talk.*`. In addition to the Big 8, it has local hierarchies, such as `aus.*` (Australia) and `can.*` (Canada). In ABnews, we reuse the ideas of newsgroups and top-level hierarchies.

A Usenet node usually runs a server called an NNTP daemon (NNTPD) [16]. Using NNTP (Network News Transfer Protocol), a node can exchange articles with other nodes. These nodes construct an unstructured peer-to-peer (P2P) network. An NNTPD also provides articles to local newsreaders. Usenet has rich newsreaders such as Mozilla Thunderbird and Emacs Gnus for PCs and NewsTap for iOS. In addition, users can access its articles using Web browsers. For example, SquirrelMail, a Web mail application, has a plugin that allows access to Usenet articles via the NNTP. In ABnews, we run our simple NNTPD to serve local newsreaders; however, our NNTPD does not communicate with other NNTPDs.

While Usenet was designed for distributing text messages, modern newsreaders can post HTML messages that include pictures (as in email). ABnews also allows HTML messages with pictures.

The current Usenet has the following problems in the context of SNSs. First, they are not designed for exchanging private articles among a small group of people. This is because it is very difficult to limit the distribution of articles within a small number of people. Second, news systems are difficult to manage, even for experts. This is because news systems are designed to achieve a high throughput. Administrators must tune many parameters and monitor the activities of the news systems.

B. Goals of ABnews

The goals of ABnews are summarized as follows:

- The ABnews system provides a BBS service that targets small groups of friends who trust one another; in contrast, Usenet provides an open worldwide BBS. When a user posts an article in a node, the news system delivers the article to the nodes of her/his friends. In addition, these friends can post follow-up articles to comment on the first article.
- The ABnews system supports NNTP access from LAN PCs [1]. Users can access articles by using standard newsreaders, including Mozilla Thunderbird and Emacs Gnus for PCs and NewsTap for iOS.
- The ABnews system provides fine-grained access control to articles with CP-ABE. A user can post an article with an access policy that specifies who can read the article. A user can read an article if he/she satisfies the corresponding access policy of the article.
- The ABnews system provides overlay social applications on top of the base BBS service. The overlay social applications include a blog similar to Twitter and Facebook Timeline, a presence service to distribute IP addresses, and one-to-one direct messaging similar to email.
- Users have control of their own data. Published messages are stored in their PCs. While ABnews uses a shared storage service on the Internet, the storage contains only encrypted data. When a storage service shuts down, it should be easy for a user to switch to another storage service with no data loss.

In ABnews, we classify users into two types: administrators and regular members. Each top-level newsgroup hierarchy has an administrator who manages the hierarchy's newsgroup list.

[1] Internet Service Providers (ISPs) cannot not block this NNTP access in a LAN.

The administrator also maintains the list of members and distributes ABE secret keys to these members. A single user can manage multiple hierarchies. A user can join multiple hierarchies that are managed by other users.

C. Environment

Each user must have a PC to run the ABnews system. Unlike in Usenet, a user can turn off her/his PC.

ABnews uses a storage service as a long-term messaging queue. The storage service must provide the following operations: reading, writing, and listing files. If the storage service allows appending files and disallows general writing, this is preferable to allowing general writing.

ABnews uses the following functions of CP-ABE:

Setup Generate an attribute public key (APK) and master secret key (AMSK).

Keygen Generate an attribute secret key (ASK) for a user with given attributes, the APK and the AMSK.

Encrypt Encrypt a plaintext under the given decryption policy using the APK.

Decrypt Decrypt a ciphertext using an APK and ASK.

D. Threat model

In the ABnews system, we assume the following.

First, the storage provider is a passive adversary, and can read all data, but it does not actively attack, as in Persona [4] and Priv.io [50]. The storage provider must provide the file access service, and keep the stored files. ABnews encrypts all the data before it is uploaded to the storage provider, and decrypts the data after downloading. In this paper, we do not address other attack types such as user tracking and metadata collection by storage providers. For example, the storage provider can track users with IP addresses, and ABnews cannot hide social graphs against such tracking.

Second, there are no malicious users in a small community and regular members trust the administrator of a hierarchy. We believe that a hierarchy administrator can have a strong sense of ethics and hold quality in high regard in a small community. We can find such a LAN administrator and a *benevolent dictator* in an open-source development project [32]. Regular members are relatively polite in a small community. All members keep ABE secret keys in secret. Section III-A will analyze a case in which everyone has her/his own individual hierarchy.

E. BBS

ABnews provides a BBS service as Usenet does. This section describes the BBS service of ABnews. Fig. 2 shows how ABnews distributes articles using a storage service. In this figure, Alice is a hierarchy administrator, and Bob and Carol are hierarchy members. First, Alice creates the hierarchy, and invites Bob and Carol to the hierarchy. All users (Alice, Bob, and Carol) can post articles to the hierarchy's newsgroups with access policies. A user can read an article if he/she satisfies the corresponding access policy of the article.

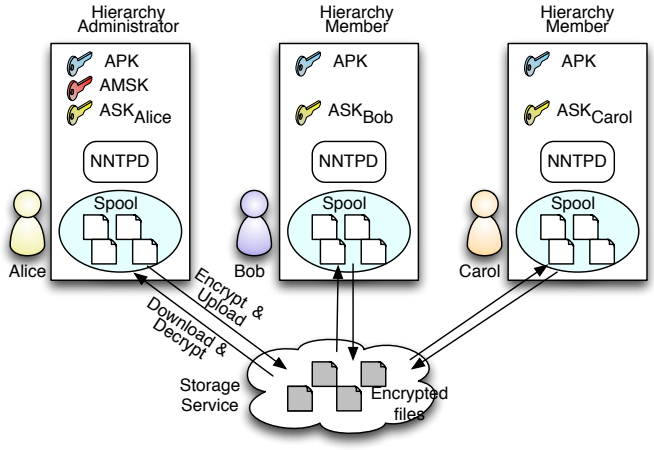

Fig. 2. Distributing articles through a storage service in ABnews.

Creating a hierarchy: To create a hierarchy, an administrator performs the following:

1) Allocates a directory in a storage service.
2) Generates an APK and AMSK.
3) Creates one or more newsgroups under the top-level hierarchy.

In an SNS for a small group of people, we can create the following newsgroups as examples:

- `prairie-waters-es-parents.*` for parents who have children in the Prairie Waters Elementary School.
- `bragg-creek-tennis.*` for a tennis circle in the Bragg Creek community.
- `north-mount-hs-alumni.*` for alumni who graduated from the North Mount High School.

Inviting a regular member: A hierarchy administrator invites a new regular member, as follows:

1) The administrator defines the attributes for the new member, and generates a unique ASK for the new member.
2) The administrator gives the new member the permission to access the storage service.
3) The administrator passes the APK, the ASK, and the information to access the storage service to the new member.

In an SNS for a small group of people, we can use following attributes as examples:

- `all` for all the members in a hierarchy.
- `admin` for the hierarchy administrator.
- `academic_year_2017` for the parents who have children enrolling in academic year 2017.
- `monday_players` for the tennis circle members who play tennis on Mondays.
- `a37th` for the 37th alumni who graduated from a high school.

We can include identity attributes in ASKs for renewing ASKs.

Posting an article: A user can post an article with a newsreader that communicates with an NNTPD running on the user's PC. At this time, the user must specify two headers: the `Newsgroups:` header for a newsgroup in the hierarchy, and the `X-Policy:` header for describing an access policy.

When the NNTPD receives a posted article, the ABnews system performs the following:

1) Retrieves the APK that corresponds to the newsgroup.
2) Encrypts the article with the APK and the policy in the `X-Policy:` header.
3) Uploads the encrypted article to the corresponding storage server.
4) Places the plaintext article in the local spool.

Reading articles: A user can read articles with a newsreader that communicates with an NNTPD. This NNTPD serves articles from one or more hierarchies in a local spool.

The ABnews system obtains articles from the storage service and saves them to the spool, as follows:

1) Downloads encrypted articles from a storage server.
2) Decrypts the encrypted articles with the user's ASK and the APK. If the ABnews system cannot decrypt an article, it drops the article.
3) Places decrypted plaintext articles into the local spool.

F. Canceling articles

ABnews allows articles to be canceled through a control message as Usenet does [1]. A control message is a special article that has the header field `Control:`. A control message to cancel a message has the header field `Control: cancel` *<message-id>*. ABnews distributes control messages in the same manner that it distributes regular messages. A control message also has the header `X-Policy:`, and is encrypted before uploading to the storage server. When a node of the ABnews system receives a cancel control message, it deletes the article that has the header `Message-Id:` *<message-id>* from the local spool. After that, the user can no longer access the canceled article.

G. Other features

The ABnews system removes old plaintext articles in a local spool as Usenet does. Each user decides the expiration periods of articles in a local spool. Similar to Usenet, ABnews can also archive old articles.

The ABnews system also removes old encrypted articles in a storage service. A typical expiration period of encrypted articles is one month. Before encrypted articles expire, members should turn on their PCs and download these files.

ABnews inherits some useful functions from Friend News System [42]. ABnews provides a newsgroup mapping function to solve name conflict problems. A user can join two hierarchies that have a same name but are managed by different administrators. ABnews uses digital signatures to prevent unauthorized message modifications and identity spoofing attacks.

III. Overlay social applications

In Section II, we have described the base BBS service of the ABnews system. ABnews provides overlay social applications on top of the base BBS service. In this section, we describe three overlay social applications: a private blogging service, a presence service, and one-to-one direct messaging. In Fig. 3, three users, Alice, Bob, and Carol are friends and using these three overlay applications. Each user is an administrator of her/his hierarchy and invites the other users.

A. Private blogs over BBS

In ABnews, we implement a private blog application (which is similar to Twitter and Facebook Timeline) over the BBS service, as follows.

- Each user creates her/his individual top-level hierarchy, and a newsgroup for her/his blog under the top-level hierarchy. For example, Alice creates the newsgroup hierarchy `alice.*` and the newsgroup `alice.blog`.
- Each user invites her/his friends to her/his individual hierarchy. Typically, two friends invite each other.
- Each user posts articles to her/his blog newsgroup. For example, Alice posts her messages to the newsgroup `alice.blog`.
- A user posts a comment as a follow-up article to the friend's blog newsgroup. For example, Bob reads an article in `alice.blog`, and posts his comments to the same newsgroup with the same subject. Many popular newsreaders support threads and collect original articles and follow-up articles into a single thread.

When a user invites a friend to her/his individual hierarchy, the user can give the friend the following attributes:

- `family`, `parents`, `children`, `relatives`, etc.
- `colleague`, `boss`, etc.
- `friend`, `closed_friend`, etc.

Using these attributes, a user can post articles to the blog newsgroup with the following policies, given as examples:

- `X-Policy: (family)` for messages to family members.
- `X-Policy: (family or colleague)` for messages to family members or colleagues.

Note that a user can post an article to another user group, even if he or she is not a member of that group. For example, suppose that Bob has the attribute `family` and Carol has the attribute `colleague` in Alice's blog newsgroup. Alice posts a blog article with `X-Policy: (family or colleague)`. Both Bob and Carol can read the article because they satisfy this access policy. Bob can post a comment article as a follow-up article with the same policy, `X-Policy: (family or colleague)`. He can do this even though he does not have the attribute `colleague` in CP-ABE.

Facebook's Newsfeed application collects messages in friends' Timelines. We can provide a similar function by using the virtual newsgroup facility, without developing new social applications. *A virtual newsgroup* is a collection of

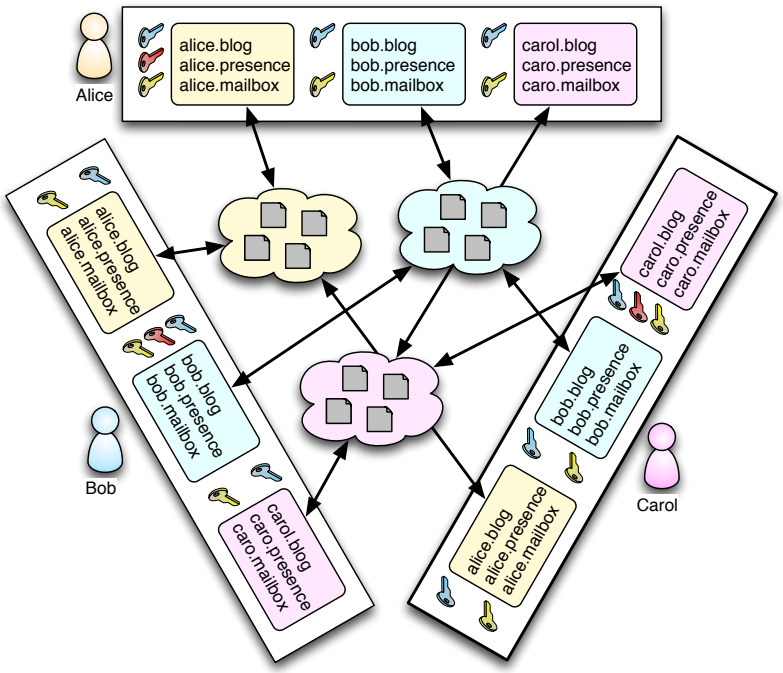

Fig. 3. Three overlay applications `*.{blog,presence,mailbox}` in the ABnews system.

other newsgroups. Newsreaders Gnus [2], nn [3], and vin [4] have such functionality. We can collect multiple blog newsgroups into a single virtual newsgroup. For example, Alice subscribes two friends' blog newsgroups: `bob.blog` and `carol.blog`. She can create a virtual newsgroup `friends.newsfeeds` that collects all messages in `bob.blog` and `carol.blog`.

B. Presence

Social VPNs [18], [25], [45], voice over IP, and other peer-to-peer real-time applications often require the current IP addresses and port numbers of friends. This information is referred to as presence information.

We implement an application that returns presence information over the private blog application described in Section III-A. For example, Alice can create the newsgroup `alice.presence` to post her presence information. Alice's friends can obtain her latest presence information from the newsgroup. A user can restrict access to her/his presence information. For example, Alice can post an article that contains presence information with the policy `X-Policy: (family)`, to allow access to family members only.

C. One-to-one direct messaging

We implement a one-to-one direct messaging application over the base BBS service of ABnews.

- Each receiver is a hierarchy administrator and creates a mailbox newsgroup for receiving messages under her/his individual hierarchy.
- When a sender sends a direct message to a friend, the sender posts an article to the friend's mailbox newsgroup. At this time, the sender specifies the policy only the hierarchy administrator satisfies.

For example, suppose that Bob is a friend of Alice, and Bob has his ASK and the APK of Alice's individual hierarchy. Bob posts an article to the newsgroup `alice.mailbox` with the policy `X-Policy: (admin)`. Because Alice satisfies the policy, she can decrypt and read the article. Other users who are regular members of the hierarchy `alice.*` cannot read the article because they do not satisfy the policy.

IV. Implementation of the ABnews system

We have implemented the ABnews system using the cpabe toolkit [7], [8] and Google Drive [19], which holds messages encrypted with ABE. In this section, we describe its implementation.

Fig. 4 shows the components of the ABnews system running in a node. This news system consists of the NNTPD, Login command, Uploader, Downloader, and Administration tool. The NNTPD interacts with a user through a newsreader. The Login command performs the login to Google Drive and obtains tokens. The Uploader and Downloader perform ABE encryption and decryption with the cpabe toolkit, and communicate with Google Drive to store and retrieve encrypted files. Users use the administration tool to manage ABE keys. These programs are written in the Ruby language. We use

[2] https://www.gnu.org/software/emacs/manual/html_node/gnus/Virtual-Groups.html

[3] https://www.opennet.ru/docs/FAQ/soft/newsreaders/nn-getting-started.html

[4] ftp://ftp.delegate.org/pub/cosmos/README

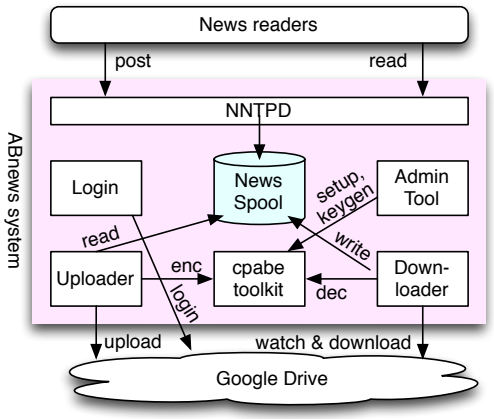

Fig. 4. Components of the ABnews system on a PC.

the google-drive-ruby library to interact with Google Drive [5]. We reuse some components of Friend News System [42] to implement the ABnews system.

A. NNTPD

The core of the ABnews system in a node is the NNTPD. This is a simple NNTPD written in the Ruby language. This NNTPD serves plaintext articles in a local spool. Unlike an NNTPD in Usenet, the NNTPD of ABnews does not communicate with other instances running in other nodes. Instead, the NNTPD of ABnews performs the following:

- It uploads posted articles to the storage service with Uploader.
- It receives articles from the storage service with Downloader.

B. Login

The Login command performs the login to Google Drive. This Login command asks the user for the authorization to access directories in Google Drive based on the OAuth protocol [22]. If the user grants the access with a Web browser, Google Drive returns the OAuth authorization code. The Login command obtains an access token and refresh token with the authorization code, and saves them for the Uploader and Downloader described below.

C. Uploader

The Uploader program is executed when a user posts an article. This programs takes the following arguments:

- The plaintext file that contains the article.
- The APK of the newsgroup hierarchy of the article.
- The ABE access policy of the article.

First, this program executes the `cpabe-enc` command of the cpabe toolkit with these arguments, and obtains an encrypted file. Next, this program uploads the encrypted file to Google Drive.

[5]https://github.com/gimite/google-drive-ruby

D. Downloader

The Downloader program periodically monitors the directory in Google Drive. When this program finds new files, it downloads them. Next, this program decrypts the downloaded encrypted files by executing the `cpabe-dec` command of the cpabe toolkit with the APK and the ASK. Next, this program places the decrypted plaintext files into the local spool.

E. Administration tool

The administration tool of ABnews performs miscellaneous tasks.

When a hierarchy administrator creates a top-level hierarchy, the tool generates an APK and an AMSK by executing the `cpabe-setup` command of the cpabe toolkit. Next, the tool allocates a directory in Google Drive to store encrypted files.

When a hierarchy administrator invites a new regular member, she/he receives the user ID of the regular member in Google Drive. Next, the administrator defines the attributes of the new regular member. Next, the administrator executes the tool to generate a unique ASK by executing the `cpabe-keygen` command of the cpabe toolkit. This command takes the following arguments: the APK, AMSK, and the attributes. The tool gives the permissions of the directory in Google Drive to the new regular member. Finally, the tool returns the ASK, APK, and the information to access the directory. The hierarchy administrator passes these pieces of information to the new regular member. When the new regular member receives these pieces of information from the hierarchy administrator, the tool saves them to the ABnews configuration file.

V. Evaluation

In this section, we evaluate the ABnews System.

A. Achievement of goals

Section II-B shows the goals of the ABnews system. This news system provides a BBS service for small groups of people using a storage service. An administrator manages a top-level newsgroup hierarchy with an ABE public key and an ABE master secret key, invites regular members, and distributes ABE secret keys. When a user posts an article in a node with an access policy, the news system delivers the article to the nodes of friends who satisfy the access policy of the article. Users can access articles by using existing rich newsreaders via NNTP. The ABnews system provides overlay social applications *person*.{`blog`,`presence`,`mailbox`} on top of the base BBS service. Users have control of their own data because posted messages remain on their PCs, and only encrypted messages go to the storage service. The results indicate that we have achieved all our goals.

In ABnews, a user has to download encrypted files even if he or she cannot decrypt them. This causes a technical problem and a social problem. The technical problem is that this generates unnecessary traffic, because users download encrypted files they cannot decrypt. We believe that the impact

of such unnecessary traffic would be minor in current Internet access lines. The social problem is that a user notices that his/her friends exchange messages without him/her. We believe that most users of ABnews will accept this because they themselves use this function. In the future, we aim to add cover traffic to realize differential privacy, as in Vuvuzela [46]. This will also mitigate the social problem.

The article canceling process in ABnews is not perfect as in Usenet. If a cancel control message is lost in a node, its corresponding target article remains in the node. Because users of Usenet accept this limitation, we believe that most users of ABnews will also accept this limitation. We can supplement this imperfection of canceling with a popular Web-based technique. Users can place encrypted files with ABE into their Web sites or storage services, and send the URLs of the encrypted files to friends using ABnews. Users can remove these encrypted files and encrypt these files again with other ABE keys and policies later. People are using a similar technique in email.

The current implementation of ABnews depends on centralized storage services, which can perform traffic analysis. We can avoid this traffic analysis by running users' own storage services or using decentralized storage services, such as BitTorrent Sync over the DHT [37].

An administrator allows regular members to upload files to the storage service. An administrator protects the storage from outside the community by using OAuth [22], random URLs as *capabilities* or other methods.

B. Comparisons with conventional SNSs using ABE

Some SNSs use ABE to implement fine-grained access control to distributed shared data [4], [34], [50]. Among these SNSs, our implementation most closely resembles Persona [4]. Persona is a privacy-enhanced decentralized SNS application that provides flexible, user-defined access control with ABE. In Persona, users exchange messages through shared storage systems. The data in shared storage systems are encrypted by ABE. Persona proposes an abstraction called *Doc*, which is a generic multi-reader multi-writer application and stores encrypted data to a shared storage service. Persona implements social applications, such as wall, chat, presence, and newsfeed, on top of the Doc abstraction. Because Persona is Web-based, it requires on-demand access to encrypted Web resources that are stored in remote storage services. In Persona, a user accesses encrypted Web resources that are stored in remote storage services using a Firefox Web browser equipped with Persona's extension.

In addition to a storage service, Priv.io [50] uses Amazon Simple Queue Service (SQS) [2] to implement inter-user communication. Priv.io requires developing new social applications in JavaScript that perform ABE computation in Web browsers.

In ABnews, in contrast to these SNSs, a user uses existing newsreaders to access plaintext messages that are stored on a local PC. Therefore, ABnews is extremely fast and does not require new social applications to be developed. Furthermore, users can read messages in their PCs while they are offline. Unlike Priv.io, ABnews uses a storage service as a long-term messaging queue with multiple readers.

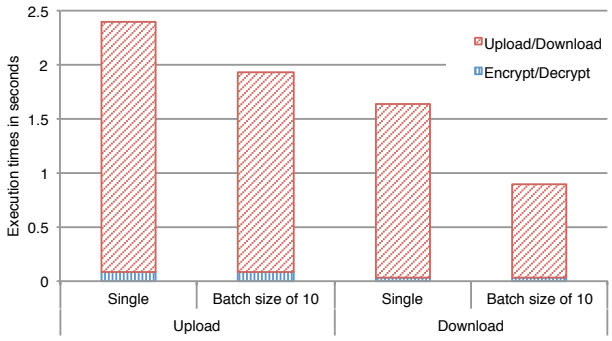

Fig. 5. Execution times of uploading and downloading an article.

Cachet [34] uses DHTs to store encrypted data. Cachet uses a social caching approach to avoid slow DHT accesses and ABE decryption. In this social caching approach, a user obtains plaintext data from friends' nodes, and the fine-grained access control with ABE does not work in those nodes. In ABnews, a user obtains plaintext data from her/his local node, and the access control works in the local node.

Sweets [26] is our Twitter-like application for Android mobile devices. Sweets performs data synchronization among devices in a P2P manner. Sweets uses ABE for access control in indirect replication as well as conventional ACLs in direct replication. In Sweets, a node must send a single message to multiple peers, which are often offline. In ABnews, a node sends a single message to a single untrusted server, which is always online.

C. Performance

In this subsection, we evaluate the scalability of ABnews in a typical use case: an SNS for a small group of people. ABnews provides a lightweight and comfortable user experience, because users access plaintext data in local nodes. Current cloud storage services have enough performance and scalability. Therefore, in this section, we measure the performance of encryption, decryption, uploading, and downloading in a single PC node and discuss the scalability of ABnews.

In our experiments, we used a PC equipped with an Intel Core i7 2.8 GHz processor and 16GB of main memory. Its operating system was Mac OS X 10.11 El Capitan. It was connected to a campus network at our university, which is connected to a national academic backbone network for universities.

We posted an article with an image and measured the uploading time and downloading time. The total size of the article was 100 KB. The article has five attributes in the policy header. We repeated the experiments 10 times, and obtained the average execution times.

Fig. 5 shows the experimental results. When a user is online, the Uploader program uploads a posted article to the storage service immediately. It took approximately 2.4 seconds to

encrypt and upload an article. This corresponds to posting an article when a user is online. When a user is offline, posted articles are spooled in a local PC. When the user becomes online, these files will be uploaded to the storage service. In Fig. 5, when the local spool had ten files and the Uploader program encrypted and uploaded these ten files (Batch size of 10), it took approximately 2 seconds per an article.

The Downloader program downloads an article periodically when a user is online. It took approximately 1.6 seconds to download and decrypt an article. When the storage had ten files and the Downloader program downloaded and decrypted these ten files (Batch size of 10), it took approximately 0.9 seconds per an article. We noticed that the encryption and decryption times were negligible compared with the uploading and downloading times. This is because the cpabe toolkit uses a hybrid encryption scheme. Furthermore, we can perform downloading and decryption in parallel if the server has multiple new files.

Because ABnews targets small groups of friends, the number of friends in a group is limited. If a user has 100 friends, and each friend posts 10 articles a day, the user uploads 10 article and download 1000 articles per day. This means that downloading limits the scalability of ABnews in this use case. It takes 17 minutes to receive these articles at a rate of one second per article, and consumes 100 MB of the user's Internet access line per day. This is sufficient for such SNSs if the user is connected to a fast Internet access line. We can also schedule downloading at midnight.

VI. Related work

We have compared ABnews with existing SNSs using ABE in Section V-B. Compared to these other SNSs, ABnews is faster and does not require new social applications to be developed. Other decentralized SNSs can be classified into three types:

- Those using federated servers; examples include Diaspora [9], OneSocialWeb [35], Vegas [15], and SoNet [38].
- Those using peer-to-peer (P2P) architecture; examples include PeerSoN [10], Safebook [13], and LifeSocial [20]. Several P2P decentralized SNSs use distributed hash tables (DHTs).
- Those using social VPNs [18], [25], [45].

Unlike these decentralized SNSs, ABnews provides fine-grained access control with ABE and reuses the rich newsreaders of Usenet without any modifications. When users run storage servers by themselves, ABnews has an advantage over the federated-server approach in that the servers contain encrypted data.

In social VPNs, users can reuse local area network (LAN) applications and intranet applications as social applications. Friend News System is our messaging system that runs on our social SoftEther VPN [41], [42], and provides similar functions to ABnews. ABnews has the following advantages over Friend News System: ABnews provides fine-grained access control with ABE, and overlays social applications that make maximum use of this access control. Furthermore, ABnews performs access control without depending on external entities. Friend News System depends on existing centralized SNSs, such as Facebook, to identify friends and realize access control.

Virtual private social networks (VPSNs) provide private messaging spaces over public centralized SNSs such as Facebook [12]. With this method, members of a private social network publish pseudo-information in the centralized SNS servers, and exchange real information with private Extensible Messaging and Presence Protocol (XMPP) servers [36]. These members can get real information with a special Web browser that maps the pseudo-information onto the real information. In our messaging system, we need not publish the pseudo information in a centralized SNS.

The paper [40] proposes federating individual users' servers running in cloud-based virtual machines. A virtual machine in this system is a good platform on which to run ABnews' NNTPD.

Numerous researchers are working on the attribute and user revocation issues in ABE [23], [24], [29], [30], [47]–[49]. Most of them are focused on algorithms and the computational time of encryption. They neither implement social applications nor evaluate the total performance of social applications. If they provide the similar API as the cpabe toolkit, we can replace it with one that supports revocations.

UsenetDHT implements a backing store of Usenet articles by using a DHT [43]. The goal of UsenetDHT is to reduce the NNTP traffic and disk usage among collaborating nodes. Our goal is to provide a social messaging system for small groups. A friend-to-friend (F2F) network is a P2P network where users only make direct connections with their friends or other users they know. The paper [27] describes the design of Usenet over an F2F network to achieve the same goal as UsenetDHT.

SUNDR [28], Depot [33], and SPORC [17] allow online collaborative applications via untrusted servers. BSTORE [11] and DepSky [6] encrypt data in storage servers. These systems do not provide SNS functionality. The papers [3], [5] build SNS functionality over untrusted networks. In this paper, we describe the provision of better access control with ABE.

VII. Conclusion

In this paper, we have described the design and implementation of ABnews, a fast social messaging system using untrusted storage and attribute-based encryption. ABnews allows Usenet's rich newsreaders to be reused without any modification. Furthermore, ABnews provides overlay social applications, including a private blogging service, a presence service, and one-to-one direct messaging, on top of bulletin board systems. We have implemented the ABnews system using the cpabe toolkit and Google Drive, which holds encrypted messages. Experimental results show that the overhead of ABE is negligible when implementing such a messaging system on a current cloud storage service.

In the future, we plan to use an ABE implementation that supports revocation. In addition, we plan to add cover traffic to realize differential privacy.

ACKNOWLEDGMENT

This work was partially supported by JSPS KAKENHI Grant Number 16H02805.

REFERENCES

[1] R. Allbery and C. Lindsey, "Netnews architecture and protocols," *RFC 5537*, 2009.

[2] Amazon Web Services. Accessed: 2016-08-01. [Online]. Available: https://aws.amazon.com/

[3] J. Anderson, C. Diaz, J. Bonneau, and F. Stajano, "Privacy-enabling social networking over untrusted networks," in *Proceedings of the 2Nd ACM Workshop on Online Social Networks*, 2009, pp. 1–6.

[4] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: An online social network with user-defined privacy," in *the ACM SIGCOMM 2009 Conf. on Data Comm.*, 2009, pp. 135–146.

[5] A. Barenghi, M. Beretta, A. D. Federico, and G. Pelosi, "Snake: An end-to-end encrypted online social network," in *2014 IEEE Intl Conf on High Performance Computing and Communications, 2014 IEEE 6th Intl Symp on Cyberspace Safety and Security, 2014 IEEE 11th Intl Conf on Embedded Software and Syst (HPCC,CSS,ICESS),*, 2014, pp. 763–770.

[6] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "DepSky: dependable and secure storage in a cloud-of-clouds," *Trans. Storage*, vol. 9, no. 4, pp. 12:1–12:33, Nov. 2013.

[7] J. Bethencourt, A. Sahai, and B. Waters. The cpabe toolkit in advanced crypto software collection. Accessed: 2016-08-01. [Online]. Available: http://acsc.cs.utexas.edu/cpabe/

[8] ——, "Ciphertext-policy attribute-based encryption," in *2007 IEEE Symposium on Security and Privacy (SP '07)*, May 2007, pp. 321–334.

[9] A. Bielenberg, L. Helm, A. Gentilucci, D. Stefanescu, and H. Zhan, "The growth of Diaspora – a decentralized online social network in the wild," in *IEEE Infocom Workshops*, 2012, pp. 13–18.

[10] S. Buchegger, D. Schiöberg, L.-H. Vu, and A. Datta, "PeerSoN: P2P social networking: Early experiences and insights," in *the Second ACM EuroSys Workshop on Social Network Systems*, 2009, pp. 46–52.

[11] R. Chandra, P. Gupta, and N. Zeldovich, "Separating web applications from user data storage with BSTORE," in *Proceedings of the 2010 USENIX conference on Web application development*, 2010, pp. 1–13.

[12] M. Conti, A. Hasani, and B. Crispo, "Virtual private social networks and a Facebook implementation," *ACM Trans. Web*, vol. 7, no. 3, pp. 14:1–14:31, Sep. 2013.

[13] L. A. Cutillo, R. Molva, and T. Strufe, "Safebook: A privacy-preserving online social network leveraging on real-life trust," *IEEE Communications Magazine*, vol. 47, no. 12, pp. 94–101, Dec. 2009.

[14] A. Datta, S. Buchegger, L. Vu, T. Strufe, and K. Rzadca, "Decentralized Online Social Networks," *Handbook of Social Network Technologies and Applications*, pp. 349–378, 2010.

[15] M. Durr, M. Maier, and F. Dorfmeister, "Vegas – a secure and privacy-preserving peer-to-peer online social network," in *ASE/IEEE International Conference on Social Computing and International Conference on Privacy, Security, Risk and Trust*, 2012, pp. 868–874.

[16] C. D. Feather, "Network news transfer protocol (NNTP)," *RFC 3977*, 2006.

[17] A. J. Feldman, W. P. Zeller, M. J. Freedman, and E. W. Felten, "SPORC: Group collaboration using untrusted cloud resources," in *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, 2010, pp. 337–350.

[18] R. J. Figueiredo, P. O. Boykin, P. S. Juste, and D. Wolinsky, "Social VPNs: Integrating overlay and social networks for seamless P2P networking," in *IEEE 17th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2008, pp. 93–98.

[19] Google Drive APIs. Accessed: 2016-08-01. [Online]. Available: https://developers.google.com/drive/

[20] K. Graffi, C. Gross, D. Stingl, D. Hartung, A. Kovacevic, and R. Steinmetz, "LifeSocial. KOM: A secure and P2P-based solution for online social networks," in *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, 2011, pp. 554–558.

[21] G. Greenwald, *No place to hide: Edward Snowden, the NSA, and the US surveillance state*. Hamish Hamilton, 2014.

[22] D. Hardt (ed.), "The OAuth 2.0 authorization framework," *RFC 6749*, 2012.

[23] J. Hur, "Improving security and efficiency in attribute-based data sharing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 10, pp. 2271–2282, Oct 2013.

[24] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, 2011, pp. 411–415.

[25] P. S. Juste, D. Wolinsky, P. O. Boykin, M. J. Covington, and R. J. Figueiredo, "SocialVPN: Enabling wide-area collaboration with integrated social and overlay networks," *Computer Networks*, vol. 54, no. 12, pp. 1926 – 1938, 2010.

[26] R. Lai and Y. Shinjo, "Sweets: A decentralized social networking service application using data synchronization on mobile devices," in *Collaborate Computing: Networking, Applications and Worksharing: 12th International Conference (CollaborateCom 2016)*, 2016, pp. 188–198. [Online]. Available: https://doi.org/10.1007/978-3-319-59288-6_17

[27] J. Li and F. Dabek, "F2F: Reliable storage in open networks," in *The 5th International Workshop on Peer-to-Peer Systems (IPTPS'06)*, 2006.

[28] J. Li, M. N. Krohn, D. Mazières, and D. Shasha, "Secure untrusted data repository (SUNDR)," in *Proceedings of the 6th USENIX Conference on Operating Systems Design and Implementation*, 2004, pp. 121–136.

[29] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 1, pp. 131–143, 2013.

[30] X. Liu, Y. Xia, Y. Xiang, M. M. Hassan, and A. Alelaiwi, "A secure and efficient data sharing framework with delegated capabilities in hybrid cloud," in *2015 International Symposium on Security and Privacy in Social Networks and Big Data (SocialSec)*, Nov 2015, pp. 7–14.

[31] Y. Liu, K. P. Gummadi, B. Krishnamurthy, and A. Mislove, "Analyzing Facebook privacy settings: User expectations vs. reality," in *the 2011 ACM SIGCOMM Conf. on Internet Measurement*, 2011, pp. 61–70.

[32] J. Ljungberg, "Open source movements as a model for organising," *European Journal of Information Systems*, vol. 9, no. 4, pp. 208–216, 2000.

[33] P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, and M. Walfish, "Depot: Cloud storage with minimal trust," in *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, 2010, pp. 307–322.

[34] S. Nilizadeh, S. Jahid, P. Mittal, N. Borisov, and A. Kapadia, "Cachet: A decentralized architecture for privacy preserving social networking with caching," in *The 8th International Conference on Emerging Networking Experiments and Technologies*, 2012, pp. 337–348.

[35] T. Sahama, J. Liang, and R. Iannella, "Impact of the social networking applications for health information management for patients and physicians," *Proceedings of the 24th European Medical Informatics Conference (MIE2012)*, vol. 180, pp. 803–807, 2012.

[36] P. Saint-Andre, "Extensible messaging and presence protocol (XMPP): Core," *RFC 6120*, 2011.

[37] M. Scanlon, J. Farina, and M. T. Kechadi, "Bittorrent sync: Network investigation methodology," in *2014 Ninth International Conference on Availability, Reliability and Security (ARES)*, 2014, pp. 21–29.

[38] L. Schwittmann, C. Boelmann, M. Wander, and T. Weis, "Sonet–privacy and replication in federated online social networks," in *IEEE International Conference on Distributed Computing Systems Workshops*, 2013, pp. 51–57.

[39] L. Schwittmann, M. Wander, C. Boelmann, and T. Weis, "Privacy preservation in decentralized online social networks," *IEEE Internet Computing*, vol. 18, no. 2, pp. 16–23, 2014.

[40] A. Shakimov, H. Lim, R. Cáceres, L. P. Cox, K. Li, D. Liu, and A. Varshavsky, "Vis-a-vis: Privacy-preserving online social networking via virtual individual servers," in *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)*, 2011, pp. 1–10.

[41] Y. Shinjo, N. Kainuma, X. Kunyao, D. Nobori, and A. Sato, "Magic mantle using social vpns against centralized social networking services," in *14th annual conference on privacy, security and trust*, 2016.

[42] Y. Shinjo, X. Kunyao, N. Kainuma, D. Nobori, and A. Sato, "Friend news system: A modern implementation of usenet over social vpns," in *Proceedings of International Conference on Social Computing and Networking*, 2014, pp. 432–440.

[43] E. Sit, R. Morris, and M. F. Kaashoek, "UsenetDHT: A low-overhead design for Usenet," in *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI'08, 2008, pp. 133–146.

[44] H. Spencer and D. Lawrence, *Managing Usenet*. O'Reilly & Associates, Inc., 1998.

[45] P. St Juste, K. Jeong, H. Eom, C. Baker, and R. Figueiredo, "TinCan: User-defined P2P virtual network overlays for ad-hoc collaboration," *ICST Transactions on Collaborative Computing*, vol. 14, no. 2, Aug 2014.

[46] J. van den Hooff, D. Lazar, M. Zaharia, and N. Zeldovich, "Vuvuzela: Scalable private messaging resistant to traffic analysis," in *Proceedings of the 25th Symposium on Operating Systems Principles*, 2015, pp. 137–152.

[47] F. Wang, J. Mickens, N. Zeldovich, and V. Vaikuntanathan, "Sieve: Cryptographically enforced access control for user data in untrusted clouds," in *the 13th USENIX Conference on Networked Systems Design and Implementation*, 2016, pp. 611–626.

[48] H. Yiliang, J. Di, and Y. Xiaoyuan, "The revocable attribute based encryption scheme for social networks," in *2015 International Symposium on Security and Privacy in Social Networks and Big Data (SocialSec)*, Nov 2015, pp. 44–51.

[49] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, 2010, pp. 261–270.

[50] L. Zhang and A. Mislove, "Building confederated web-based services with priv.io," in *Proceedings of the First ACM Conference on Online Social Networks*, 2013, pp. 189–200.