プライバシを保護するための VPN を用いたソーシャルアプリケーション実行環境

海沼直紀[†] 新城靖[†] 登大遊[†]
肖焜瑶[†] 佐藤聡[†] 中井央[†]

現在広く利用されている Social Networking Service(SNS) は中央サーバを用いて実装されており、プライバシ侵害などの問題がある。この問題に対処するために分散型 SNS が提案されている。分散型 SNS とは、中央サーバを用いることなく、友人間の交流を支援するソーシャルアプリケーションを利用するため仕組みである。分散型 SNS の実装方法の 1 つとして、友人間の PC を VPN で接続するソーシャル VPN という概念が提案されている。ソーシャル VPN では、NFS などの LAN 上で動作するアプリケーションを SNS アプリケーションとして利用することができる。

この論文では、集中型 SNS を補完し、連携して動作しつつ、分散型 SNS と同様にプライバシ保護を可能にするためのソーシャル VPN について述べている。このソーシャル VPN では、パブリックな情報は、集中型 SNS で、プライベートな情報は、友人間の PC を接続する VPN で交換する。このソーシャル VPN は、ブラウザを用いて利用者認証を行うことで SNS と VPN のユーザ認証を連携させる。さらに、VPN のユーザ認証と VPN 内の Web サーバのユーザ認証を連携させる。このソーシャル VPN を SoftEther VPN に集中型 SNS を用いたユーザ認証機能を付加することで実装している。また LAN 用のアプリケーションを仮想計算機で実行する。

1. はじめに

SNS(Social Networking Service) は世界中で多くのユーザが利用しており、有用なコミュニケーションツールとして普及している。そのような Web ベースの SNS の多くは集中型 SNS であり、その上のアプリケーションの多くは中央サーバを用いて Web アプリケーションとして実装されている。例えば、Facebook では、Facebook のグループ内のユーザ同士で写真の共有ができる Web アプリケーションがある。

集中型 SNS では、中央サーバに個人情報が集約される。そのため、プライバシと信頼の問題が指摘されている[1),2)]。ユーザは、プライベートな情報を中央サーバにアップロードする時、中央サーバを全面的に信頼しなければならない。2013 年には、Edward Snowden 氏により、米国政府機関が集中型 SNS に対して大規模なプライバシ侵害行為を行っていることが告発された[3)]。コンピュータに詳しくない一般ユーザにとって、自分の資源に対するアクセス制御リスト (access control list) を維持し、プライバシを保護することは簡単ではない[4)]。

このような集中型 SNS の問題に対処するために分散型 SNS または分散型 OSN(Decentralized Online Social Network) が提案されている[1),2)]。分散型 SNS とは、中央サーバを用いることなく、友人間の交流を支援するソーシャルアプリケーションを利用するための仕組みである。分散型 SNS の実装方法は、主に次の 2 種類に分類される[2)]。

- サーバを連邦化 (federation) するもの
  ユーザは信頼する組織が設置したサーバ、または、個人で設置したサーバにデータを保存する。例として、Diaspora[5)], OneSocialWeb[6)], Persona[7)] が挙げられる。
- P2P(peer-to-peer) ネットワークや DHT(Distributed Hash Table) を使うもの
  個人のコンピュータで、SNS アプリケーションを動作させる。例として、PeerSoN[8)] や Safebook[9)], LifeSocial[10)] が挙げられる。

この他に**ソーシャル VPN** を用いて実装する分散型 SNS が存在する[11)]。この実装方法では VPN で接続する。その際、ユーザ認証に用いるパスワード等を各ユーザに配布する代わりに SNS のユーザ認証サービスを用いる。SNS ユーザは構築されたネットワークで NFS などの LAN 用アプリケーションを利用できる。NFS を使えば、ユーザはプライベートな文書や写真を個人の PC に置き、友人だけにアクセスさせることが可能となる。ソーシャル VPN を実装した例として、

† 筑波大学
University of Tsukuba

フロリダ大学によるものが挙げられる[11]。

分散型 SNS では、個人情報は個人の PC や信頼できる連邦化されたサーバ上に保存される。そのため集中型 SNS のプライバシ侵害の問題点を解決できる。しかし、集中型 SNS の全ての機能を分散型 SNS で置き換えることはできない。例えば、分散型 SNS では海外に住んでいて連絡先が不明な昔の友人と偶然出会えるという事ができなくなる。その理由は、分散型 SNS で友人関係を結ぶには、公開鍵を交換する必要がある。その公開鍵を交換するために電子メールやインスタント・メッセンジャが使われることが多いからである。これらのツールは、連絡先が不明な昔の友人との間では使えない。また、集中型 SNS には、個人の意見を全世界に広く伝えるには有用なツールである。

そこで本研究の目的を、集中型 SNS と分散型 SNS を連携させて利用することと定める。具体的には、パブリックな情報は集中型 SNS で、プライベートな情報は、SNS メンバの PC と PC を接続する VPN で交換できるようにする。これにより、集中型 SNS の利点と分散型 SNS の利点を共に利用することを可能にする。

集中型 SNS とソーシャル VPN による分散型 SNS を連携させるためには、ユーザ認証を連携[12),13)] させることが重要である。しかしながら、従来のソーシャル VPN の実装では、次のような問題があり、利便性が低かった。

- ユーザ認証のために、何度もパスワードを入力する必要がある。たとえば、分散型 SNS の Web アプリケーションを利用する場合、VPN のユーザ認証と Web アプリケーションでのユーザ認証のために 2 回ユーザ名とパスワードを入力する必要がある。(集中型 SNS も使う場合には、そのためにさらにもう 1 度ユーザ名とパスワードを入力する必要がある。)
- セキュリティを高めるためにソーシャル VPN によるネットワークをインターネットから隔離した場合、そのネットワークでは、SNS のユーザ名とパスワードを使ったユーザ認証が行えない。そのため、別途ユーザ名とパスワードを安全に配布する必要があるが、この手間は非常に大きい。

このような問題点を解決し、利便性を高めるために、本研究では、次のようなソーシャル VPN を用いたソーシャルアプリケーション実行環境を実現する。

(1) 集中型 SNS におけるユーザ認証と VPN のユーザ認証を連携させる。
(2) VPN のユーザ認証と本実行環境内で動作している Web アプリケーションのユーザ認証を連携させる。VPN 内でしか通信ができない場合でもこの連携を可能にする。

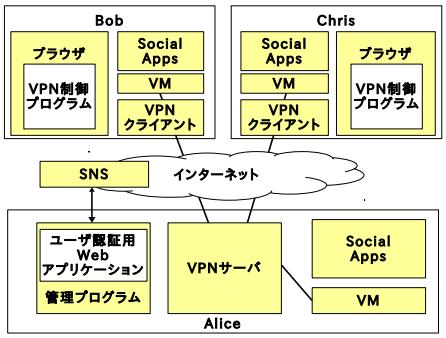

図 1 ソーシャルアプリケーション実行環境の外観

2. 実装するソーシャルアプリケーション実行環境の概要

2.1 本研究の目標

本研究の目標は、互いに信用している少人数のグループが LAN 用のアプリケーションをソーシャルアプリケーションとして実行できる環境を提供することである。この実行環境は集中型 SNS が持っている全世界に対する発信機能を生かしながら、分散型 SNS が持つプライベートな通信機能を提供する。本研究ではこの環境を実現するために、我々が開発している VPN である SoftEther VPN に Facebook, Twitter, および Google+によるユーザの認証機能を付加し、SNS のグループ内のユーザ間で VPN を簡単に導入できるようにする。

本実行環境では、全てのユーザは、自分の PC 上に VM を構築し、その VM をソーシャル VPN で接続し、その中でソーシャルアプリケーションを実行する。その利点の 1 つとして、VM で実行可能な OS であれば、任意の OS を利用できることが挙げられる。例えば、NetBSD のように、SoftEther VPN のクライアントが提供されていないような OS でも利用できる。またもう 1 つの利点として、アプリケーションを安全に利用できることが挙げられる。例えば、アプリケーションに悪意のあるコードが含まれていた場合、被害を最小限に抑えることができる。また VM にアプリケーションをインストールしておき、そのディスクイメージを配布することもできる。

VM 上で動くアプリケーションは DNS サーバを用いて SNS の ID を用いて通信相手を指定可能にする。例えば、通信相手が Alice ならば `alice.social-vmvpn`

```
/home/alice/share *.social-vmvpn(rw,\
sync,no_subtree_check)
/home/alice/secret bob.social-vmvpn(rw,\
sync,no_subtree_check)
```

図 2 公開ディレクトリの設定

```
bob@bobvm:~$ sudo mount -t nfs alice.\
social-vmvpn:/home/alice/secret /mnt
```

図 3 公開ディレクトリをマウントするコマンド

というホスト名を指定することができる。

2.2 想定する利用方法

本実行環境のユーザはハブユーザと一般ユーザに分類される。**ハブユーザ**は、LAN の管理者に当たり、実行環境の中心となる VPN サーバ、および、その管理プログラムを実行する。**一般ユーザ**は、LAN の一般ユーザに当たり、VPN クライアント、および、その制御プログラムを実行する。

図 1 で、ハブユーザである Alice と一般ユーザである Bob と Chris は SNS の友達同士であり、この 3 人は SNS においてあるグループに所属している。この実行環境で LAN 用のアプリケーションをソーシャルアプリケーションとして利用する。以下では、NFS を例として想定する利用方法を簡単に述べる。その他のアプリケーションについては、7.1 節で述べる。

本実行環境で SNS ユーザは NFS を利用して PC 間で容量制限や帯域制限のないファイルの共有が可能になる。SNS ユーザは友達が NFS で公開しているディレクトリをマウントするために SNS のユーザ名を含むホスト名を利用できる。NFS にはアクセス制御の機能があり、NFS サーバは NFS クライアントに対してアクセス可能なディレクトリと読み書き許可等のオプションを設定することができる。NFS を用いてファイルを共有するには、まず、NFS サーバとなる VM の `/etc/exports` というファイルに公開するディレクトリ、公開する範囲、およびオプションを図 2 のように記述する。図 2 では`/home/alice/share` というディレクトリを `social-vmvpn` のドメインを持つユーザに読み書き可能で公開し、`/home/alice/secret` というディレクトリを Bob のみに読み書き可能で公開する設定にしている。

NFS のクライアントでは図 3 のように公開されているディレクトリをマウントする。図 3 では Alice が公開している`/home/alice/secret` ディレクトリを Bob の VM 上の`/mnt` ディレクトリにマウントしている。

2.3 使用する VPN ソフトウェア

本ソーシャルアプリケーション実行環境の実現で使用している VPN ソフトウェアは SoftEther VPN である。SoftEther VPN[14] は、イーサネット通信の仕組みをソフトウェアで実装することによって、VPN を実現しているソフトウェアである。スイッチングハブや LAN カードもソフトウェアで実装されている。スイッチングハブに相当するものが仮想ハブ、LAN カードに相当するものが仮想 LAN カードである。

SoftEther VPN では、ユーザは、VPN サーバに接続するための仮想 LAN カードと接続設定をクライアント PC 上で作成する。**接続設定**とは、ユーザ名、使用する仮想 LAN カード、接続するサーバのホスト名または IP アドレスとポート番号、接続するハブ、ユーザの認証方法などを記したものである。VPN の管理者はサーバ側で仮想ハブを作成し、接続してくるユーザ名とそのユーザの認証方法を登録する。

SoftEther VPN でのユーザの認証方法には広く利用されているパスワード認証の他に、署名済み証明書認証と呼ばれる方法がある。これは、サーバに信頼する認証局を登録し、クライアントがその認証局で署名された X.509 証明書とその証明書に対応する秘密鍵を保持していることを確認する認証方法である。本実行環境の実現では署名済み証明書認証を利用する。

SoftEther VPN ではクライアントに IP アドレスを割り当てる必要がある。これには 2 種類の方法がある。1 つは VPN サーバに内蔵されている DHCP(Dynamic Host Configuration Protocol) サーバを利用する方法、もう 1 つは仮想ハブの外部に独立した DHCP サーバを接続する方法である。後者の方法では仮想ハブと同一のセグメントにある既設の DHCP サーバにより、クライアントに IP アドレスを割り当てることができる。本実行環境の実現では後者の方法を利用する。

SoftEther VPN のユーザインタフェースは GUI によるユーザインタフェースとコマンドラインによるユーザインタフェースがある。前者は Windows にのみ実装されている。本実行環境の実現では後者を使用する。この方法では通常は VPN 接続を行うために多くのコマンドを打たなければならない。本実行環境の実現では、Web ブラウザを用いて簡単に VPN クライアントを実行する機能を実現する。

3. 管理プログラムの実装

この章では、ハブユーザが実行する管理プログラムの実装について述べる。管理プログラムとは、本実行環境のサーバの運用に必要なプログラムの設定を行うプログラムである。ハブユーザはこのプログラムを実行するだけでサーバを簡単に設定できる。管理プログラムは以下の要素から構成される。

ユーザ認証用 Web アプリケーション　ユーザ認証を行う Web アプリケーションである。詳しくは 3.1 節で述べる。

仮想ハブ　SoftEther VPN のサーバである。

認証局　SoftEther VPN クライアントを認証するための証明書を発行する。OpenSSL[15] を利用する。

証明書発行プログラム　クライアントに証明書を発行する。詳しくは 3.2 節で述べる。

DNS サーバ　集中型 SNS のユーザ名を含むホスト名が登録される。BIND9[16] を利用する。

DHCP サーバ　クライアントに VPN 内で使える IP アドレスを割り当てる。ISC-DHCP[17] を利用する。

DNS 登録プログラム　集中型 SNS のユーザ名を含むホスト名を DNS に登録する。詳しくは 3.3 節で述べる。

3.1 ユーザ認証用 Web アプリケーション

ユーザ認証用 Web アプリケーションとは集中型 SNS のユーザ認証により証明書発行プログラムへアクセスできるようにする Web アプリケーションである。一般ユーザはまずこのアプリケーションを Web ブラウザでアクセスして実行する。このアプリケーションは次のことを行う。

(1) 一般ユーザを集中型 SNS のユーザ認証画面にリダイレクトさせる。
(2) ユーザ認証が完了した一般ユーザのユーザ情報を SNS のサーバから取得する。
(3) そのユーザ情報からユーザ名を取り出し、データベースに乱数をキーとして保存する。
(4) 一般ユーザに証明書発行プログラムへの乱数を含む URL、VPN サーバの IP アドレス、そして、VPN サーバの仮想ハブ名が書かれた Web ページを生成する。

3.2 証明書発行プログラム

証明書発行プログラムとは 2.3 節で述べた SoftEther VPN の署名済み証明書認証のために使用する証明書を発行するプログラムである。このプログラムは一般ユーザ側で実行される VPN 制御プログラムからの要求に応じて実行される。このプログラムは次のことを行う。

(1) ハブユーザのユーザ認証用 Web アプリケーションが取得したユーザ名を VPN サーバに登録する。
(2) 公開鍵・秘密鍵の組と証明書を作成する。また、ユーザ名、VPN サーバのアドレス、および仮想ハブ名を書いたテキストファイルも作成する。
(3) これらの秘密鍵、証明書、およびテキストファイルを ZIP ファイルにまとめて 4 章で述べる VPN 制御プログラムに送信する。

3.3 DNS 登録プログラム

DNS 登録プログラムとは DNS サーバに SNS のユーザ名を含むホスト名を登録するプログラムである。DHCP サーバは一般ユーザに IP アドレスを割り当てたときにこのプログラムを動かす。この時、DHCP サーバはゲスト OS の MAC アドレスと割り当てた IP アドレスを DNS 登録プログラムに渡す。MAC アドレスと IP アドレスを受け取ったこのプログラムは次のことを行う。

(1) MAC アドレスをもとに VPN サーバから対応する VPN セッションを探す。
(2) VPN セッションからユーザ名を取得する。このユーザ名は SNS のユーザ名である。
(3) SNS のユーザ名を含むホスト名を生成し、ユーザに割り当てた IP アドレスを用いて、DNS サーバにホスト名の正引きと逆引きを登録する。例えば SNS のユーザ名が Alice の場合、ホスト名は alice.social-vmvpn となる。

3.4 ハブユーザがすること

ハブユーザは、次の操作を 1 回だけ行う。

(1) SNS のグループに向けて本実行環境の紹介、本実行環境を使うために必要なプログラムとソーシャルアプリケーションを含む VM のディスクイメージを配布するページ、およびユーザ認証用 Web アプリケーションへのリンクを投稿する。
(2) ユーザ認証用 Web アプリケーションを SNS に登録する。
(3) SNS への登録の結果得られたアプリケーションの ID とシークレットキーを管理プログラムにセットする。
(4) 管理プログラムを実行し、ユーザ認証用 Web アプリケーションをアクセス可能にする。

また、ハブユーザは次の操作をソーシャルアプリケーションを使うときに毎回行う。

(1) VM を実行する。VM のネットワークインタフェースを、VPN サーバに直接☆接続する。
(2) VM の中でソーシャルアプリケーションを実行する。

☆ SoftEther VPN が持っているローカルブリッジ機能を用いる。VPN クライアントを用いない。

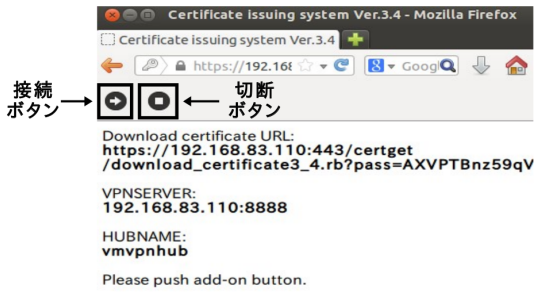

図 4 Web ブラウザを用いた VPN 制御プログラムのユーザインタフェース

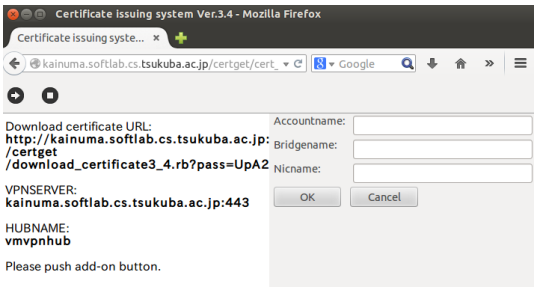

図 5 設定ファイルの作成フォーム

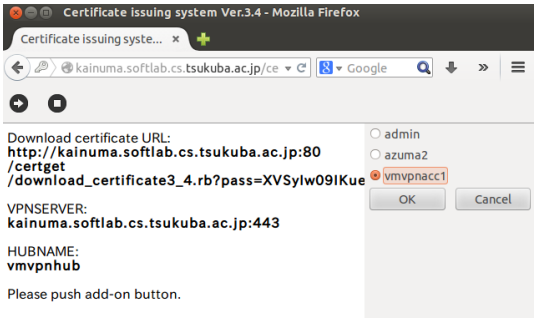

図 6 切断する接続設定を選択するフォーム

4. Web ブラウザインタフェースを持つ VPN 制御プログラム

この章では一般ユーザが実行する VPN 制御プログラムについて述べる。VPN 制御プログラムとは VPN クライアントのパラメタを設定し実行するプログラムである。一般ユーザはこの VPN 制御プログラムを Web ブラウザインタフェースを通して実行することにより、集中型 SNS とユーザ認証の連携ができる。Web ブラウザを使うことによりユーザは確かに SNS のサーバでユーザ認証がなされたということを確認できるという利点も生じる。また、VPN の設定も Web ブラウザから行えるため、VPN のコマンドラインによるユーザインタフェースを実行する必要もなくなる。

一般ユーザのプログラムは以下の要素から構成される。

VPN クライアント SoftEther VPN のクライアントである。

VPN 制御プログラム VPN 接続を行うプログラムである。詳しくは 4.1 節で述べる。

Firefox のアドオン VPN 制御プログラムを起動する。詳しくは 4.2 節で述べる

4.1 VPN 制御プログラム

VPN 制御プログラムは、ハブユーザからの証明書の取得、VPN クライアントの設定、VPN サーバへの接続、およびブリッジの作成を行うプログラムである。一般ユーザは集中型 SNS でユーザ認証を終えた後、Firefox のアドオンを通して VPN 制御プログラムを実行する (図 4)。アドオンの接続ボタンが押されると、このプログラムは次のことを行う。

(1) Web ページに書かれた URL、VPN サーバの IP アドレスと仮想ハブ名を受け取る。
(2) その URL を用いて 3.2 節で述べた証明書発行プログラムを実行する。
(3) 証明書発行プログラムから秘密鍵、証明書、およびテキストファイルを含む ZIP ファイルを受け取る。
(4) これらの秘密鍵、証明書、および、テキストファイルから読み取ったユーザ名、VPN サーバの IP アドレス、および仮想ハブ名を 2.3 節で述べた接続設定に設定する。
(5) 仮想 LAN カードを作成し、仮想 LAN カード名も接続設定に設定する。
(6) VPN クライアントを VPN サーバに接続する。
(7) ブリッジを作成してそれに仮想 LAN カードを接続する。

4.2 Firefox のアドオン

Web ブラウザによるユーザインタフェースを実装するために Firefox のアドオンを作成した。このアドオンは図 4 のような証明書取得のための URL、VPN サーバの IP アドレス、そして、VPN サーバの仮想ハブ名が書かれた Web ページで実行する。

VPN 接続を開始するとき、一般ユーザは Web ブラウザ上のアドオンの接続ボタンを押す (図 4)。初めて VPN サーバに接続するときには、Web ブラウザに図 5 のようなフォームが現れる。ここには 2.3 節で述べた接続設定の名前、ブリッジの名前、仮想 LAN カードの名前を入力する。OK のボタンを押すと、設定ファイルが作成され、VPN 制御プログラムが実行される。設定ファイルには接続設定の名前とブリッジの名前が

書かれている。2 回目以降は VPN 制御プログラムはこの設定ファイルを読み込むことにより、どの接続設定とブリッジ名を VPN 接続に使えばいいのか知ることができる。2 回目以降にアドオンの接続ボタンが押されると、図 5 のようなフォームは現れず、VPN 制御プログラムが実行される。

また、このアドオンは VPN 接続の終了も可能である。この機能は任意のページで実行可能である。

VPN 接続を終了するとき、一般ユーザは Web ブラウザ上のアドオンの切断ボタンを押す。するとアドオンは VPN サーバに接続中の接続設定を探す。接続中の接続設定が複数ある場合は Web ブラウザにどの接続設定を切断するのか決めるフォームを表示する (図 6)。切断する接続設定を選択し OK ボタンを押すと、アドオンは VPN 制御プログラムを起動する。VPN 制御プログラムは接続設定が使用している仮想 LAN カードが接続しているブリッジを探す。最後に VPN 制御プログラムは VPN 接続を切断し、ブリッジも削除する。

4.3 一般ユーザがすること

一般ユーザは、1 回だけハブユーザが投稿した記事のリンクから本実行環境を使うために必要なプログラムとソーシャルアプリケーションを含む VM のディスクイメージをダウンロードしてインストールする。

次に、一般ユーザは次の操作をソーシャルアプリケーションを使うときに毎回行う。

(1) ハブユーザが設置したユーザ認証用 Web アプリケーションにアクセスする。
(2) SNS のユーザ認証画面に遷移するのでそこでユーザ名とパスワードを入力する。すると (1) のアプリケーションに戻る。
(3) ユーザ認証用 Web アプリケーションで Web ブラウザのアドオンである VPN 制御プログラムを実行する。すると VPN 接続が完了する。
(4) VM を実行する。VM のネットワークインタフェースを、VPN 接続がなされたブリッジに接続する。
(5) VM の中でソーシャルアプリケーションを実行する。

(2) の操作は既に Web ブラウザで SNS にログインしている状態なら必要ない。また初めてこの本実行環境を使う場合、SNS アプリケーションの使用許可が 1 回だけ必要となる。

5. VPN サーバと Web アプリケーションにおけるユーザ認証の連携

1 章では、従来のソーシャル VPN の実装にはユーザ認証においていくつかの問題があることを述べた。その問題のうち、集中型 SNS のユーザ認証と VPN のユーザ認証の繰り返しの問題は、4 章で述べた Web ブラウザインタフェースを持つ VPN 制御プログラムにより解決された。この章では、残りの問題、すなわち、Web アプリケーションに対するユーザ認証の繰り返しの問題とパスワード配布の問題を解決する方法を示す。

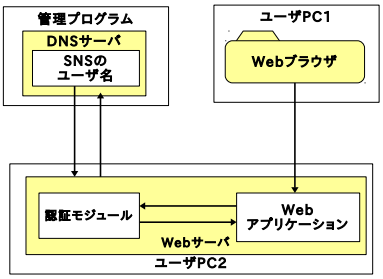

図 7 Apache HTTP Server の認証モジュールの動作

```
AuthType Vpn
AuthVpnDomain social-vmvpn
Require user bob chris
```

図 8 Apache HTTP Server の認証モジュールの使用方法

本研究では、これらの問題を解決するために、VPN のユーザ認証と Web アプリケーションのユーザ認証を連携させる。そのために Apache HTTP Server の認証モジュールを作成する。本実行環境では 3 章で述べたように、SNS のユーザ名で接続相手を指定するため、DNS サーバに SNS のユーザ名を含むホスト名を登録している。この SNS のユーザ名を取り出してユーザ認証を行うために認証モジュールを作成した。このモジュールを用いたユーザ認証は次のように行われる (図 7)

(1) ユーザは Web ブラウザを用いて他のユーザの Web サーバで実行されている Web アプリケーションにアクセスする。
(2) 認証モジュールは getpeername システムコールでユーザの IP アドレスを取得する。
(3) 認証モジュールは取得した IP アドレスを用いて DNS に問い合わせる。DNS はホスト名を返す。
(4) 認証モジュールはホスト名からユーザ名を取り出し、ユーザ名を環境変数 `REMOTE_USER` に入れ、CGI や PHP の Web アプリケーションを実行する。

この認証モジュールを利用するためには `.htaccess` ファイルに図 8 のように記述する。この記述は Basic

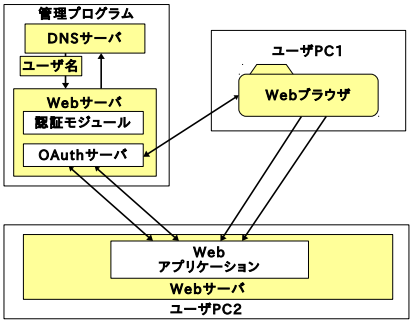

図 9 OAuth サーバを用いたユーザ認証

認証に似ている。この記述により `social-vmvpn` というドメインを持つ VPN 内の Bob と Chris のみがこのページにアクセスできるようになる。

本認証モジュールを Apache HTTP Server にインストールするには apxs(APache eXtenSion tool) を使用する。これは Apache HTTP Server に拡張モジュールをインストールするために使用するツールである。

6. OAuth サーバによるユーザ認証の連携

6.1 Apache 認証モジュールの問題点

5 章で述べたように、ユーザ認証の連携のために、認証モジュールは DNS からユーザ名を取得する。しかし、この機能を利用するためには Apache HTTP Server に認証モジュールを導入する必要がある。そのため各ユーザの所持する Web サーバに認証モジュールを導入できない場合はこの機能を使えないという問題がある。

6.2 解 決 手 法

この問題点を解決するために、OAuth[18] サーバを用いてユーザ名を Web アプリケーションに返すようにする。これにより OAuth サーバにのみ 5 章で述べた認証モジュールを導入すれば他の SNS のメンバーは特別なモジュールの導入を行うことなくユーザ認証の連携を行うことができるようになる。

OAuth サーバを用いたユーザ認証の連携の様子を図 9 に示す。DNS は図 9 のように OAuth サーバに近いところに設置する。OAuth サーバを用いたユーザ認証の連携は次のように行われる。

(1) ユーザが Web ブラウザを用いて Web サーバにアクセスする。このとき、OAuth サーバにリダイレクトされる。
(2) OAuth サーバは `REMOTE_USER` からユーザ名を取り出し、ユーザ名をアクセストークンでアクセス可能にする。
(3) ユーザは Web サーバにリダイレクトされる。このとき OAuth サーバから code というパラメータを渡される。そして Web アプリケーションが実行される。
(4) Web アプリケーションはユーザから送られた code を使用して、OAuth サーバからアクセストークンを受け取る。そしてアクセストークンからユーザ名を取得する。

この OAuth サーバを用いることにより既存の OAuth に対応した Web アプリケーション本体を大きく変更することなく本実行環境で利用することができる。

6.3 OAuth サーバの実装

OAuth サーバは PHP のライブラリである oauth-server-php[19] を用いて実装した。この OAuth サーバは 3 つの Web アプリケーションから成る。

authorize.php Web アプリケーションの使用許可をユーザに求める。普通はここでユーザ名とパスワードの入力が求められるが、本 OAuth サーバではユーザ名を 5 章で述べた `REMOTE_USER` から取得するので必要ない。ユーザは単に使用許可を与えるだけである。許可が与えられれば、ユーザの情報 (ユーザ名等) をデータベースに保存する。

token.php Web アプリケーションから code を受け取りアクセストークンを返す。

resource.php Web アプリケーションからアクセストークンを受け取り、データベースからユーザの情報を取り出し、それを返す。

7. 評　　価

7.1 使用できるアプリケーション

本研究で構築した実行環境では NFS の他に次のようなアプリケーションがある。

7.1.1 Friend News System

本実行環境で、Friend News System が正常に動作した。Friend News System[20] とはネットニュースのように記事のやりとりを行うアプリケーションである。従来のネットニュースは記事を全世界に配信しているが、このアプリケーションは SNS の少人数グループ内で記事を交換する。記事の交換には本研究のような通信路を用いるので安全であるが、更にデジタル署名を用いて悪意のあるユーザが記事を改変することを防ぐことができる。このアプリケーションではメンバーリストという電子メールのメーリングリストと似ている仕組みを用いて配布範囲を指定することができる。またこのアプリケーションは NNTP を利用したニュースリーダを使用して記事の投稿や閲覧をすることができる。

これにより Thunderbird などのニュースリーダで簡単な設定をすることでメッセージの投稿や閲覧をすることができる。さらにタグという従来のネットニュースでの Newsgroups に似ている仕組みを用いて、投稿する記事に好きなタグをつけたり、購読したい記事をタグで選択することができる。送信者の身元は From のヘッダフィールドに置かれるが、これは本実行環境から容易に得られる。

7.1.2 インスタントメッセンジャー Empathy

本実行環境で、インスタントメッセンジャー Empathy が正常に動作した。Empathy[21] はテキスト、音声、ビデオチャット、ファイル転送を様々なプロトコルを用いて行うことのできるインスタントメッセンジャーである。デフォルトでは People Nearby☆という同一の LAN 内にいるユーザ同士でチャットやファイル転送ができるサービスを利用する。Empathy はマルチキャスト DNS を利用してユーザを自動的に発見する。ユーザが発見できたとき、そのユーザが相手先の一覧に現れる。その一覧からチャットをしたいユーザを選択してメッセージをやり取りできる。

7.1.3 ゲーム Frozen-Bubble

本実行環境で、対戦型ゲーム Frozen-Bubble が正常に動作した。Frozen-Bubble[22] は画面上のバブルを撃ち落とすパズルゲームである。このゲームではネットワーク対戦ができる。対戦をするには、まず一方のユーザが Frozen-Bubble を起動して対戦ゲームモードを起動する。するとこのユーザはサーバとなり対戦相手を待つ。もう一方のユーザはコマンドライン上で Frozen-Bubble を起動する。この際に本実行環境での対戦相手の名前を指定する。すると、指定した相手のゲームサーバに接続して対戦が可能となる。

7.1.4 ウィキ DokuWiki

6 章で述べた OAuth 認証機能を使ってウィキ DokuWiki が正常に動作した。DokuWiki[23] は PHP 言語で実装されているウィキである。その大きさは 15 万行である。このアプリケーションには oauth plugin[24] というプラグインがある。このプラグインも PHP 言語で実装されている。その大きさは 4 千行である。このプラグインは Facebook や Google+などの OAuth プロバイダを利用したユーザ認証の連携ができる。この oauth plugin には DoorkeeperAdapter.php というスクリプトがある。このスクリプトには 4 行ハードコーディングされた部分がある。その部分を本実行環境で使用できるように変更した。

7.2 機能評価

2 章で述べたように、本研究の目標は、ソーシャル VPN を実現し、互いに信用している少人数のグループが LAN 用のアプリケーションをソーシャルアプリケーションとして実行できる環境を提供することである。実現したソーシャルアプリケーション実行環境は、集中型 SNS が持っている全世界に対する発信機能を生かしながら、友人間のプライベートな通信機能を SoftEther VPN を使って提供する。この実行環境では、LAN 用のファイル共有、ゲーム、チャット等のアプリケーションが利用可能である。また、本研究室で実現している Friend News System の通信基盤としても動作している。さらに、本実行環境では、VM を用いることで、LAN 用のアプリケーションの配布をディスクイメージの配布という形で容易にしている。このように本実行環境は 2 章で述べた目標を達成したと言える。

本研究では、集中型 SNS とソーシャル VPN でユーザ認証が連携できないという問題を、3 章で述べたように、Web ブラウザのアドオンを用いて解決した。また、VPN 内部で動作する Web アプリケーションと VPN のユーザ認証が連携できないという問題を、3 章と 4 章で述べたように、Apache の認証モジュール、および、OAuth サーバにより解決した。このように、本研究は 1 章で述べた従来のソーシャル VPN におけるユーザ認証の問題を解決した。

本ソーシャルアプリケーション実行環境は、従来のソーシャル VPN における XMPP(eXtensible Messaging and Presence Protocol) サーバと同様に、次のような制約がある。

- ハブユーザは、24 時間、VPN サーバと管理プログラムを動作させることが望まれる。高い可用性を実現するためには、グループ内の複数のユーザがハブユーザとなることが望まれる。
- ハブユーザが利用している IP アドレス（刻々と変化する可能性がある）を、他の一般利用者に公告する必要がある。それには、SNS 内の特定のページに現在の IP アドレスを記述する方法や、Dynamic DNS を使う方法がある。

これらの問題は、従来の Social VPN における XMPP サーバでも発生する。その解決方法も、従来の方法と似ている。

本ソーシャルアプリケーション実行環境は、Facebook, Twitter, および Google+の Web アプリケーショ

☆ People Nearby は、Telepathy-Salut (http://telepathy.freedesktop.org/wiki/) と呼ばれる、mDNS (multicust DNS)、リンクローカル XMPP (Extensible Messaging and Presence Protocol) 等の技術を用いて、サーバレスでチャットを行う仕組みを用いている。

ンとして動作するので、それらの SNS が提供している高度な API が利用可能である。たとえば、Facebook のグループ機能や Twitter のリスト機能を利用することができる。この点は、XMPP を利用している、従来の Social VPN よりも優位性がある。ただし、本実行環境では、ハブユーザはそれら SNS の開発者用ページで、Web アプリケーションの登録を行わなければならない。これは、慣れていないユーザには負担となる。

本ソーシャルアプリケーション実行環境では、VM を用いる。その利点としては、VM で実行可能な OS であれば、任意の OS を利用できることが挙げられる。また、アプリケーションを安全に利用することができるという利点も挙げられる。さらに、アプリケーションの配布も、ディスクイメージの形で簡単に行うことができる。VM により作られた隔離された環境でソーシャルアプリケーションを実行することもできる。VM を用いることの問題点は、性能が低下することである。また、通常の実行環境と隔離されているので、通常のアプリケーションとソーシャルアプリケーションで情報を共有する時に利便性が低下する。

Facebook, Twitter, および Google+ は、ユーザ認証に OAuth 2.0 や 1.0 プロトコルを用いている。6 章で述べた OAuth サーバは、これらのプロトコルに対応しているので、これらの SNS 用に開発されたアプリケーションをイントラネットのアプリケーション、すなわち、本実行環境のソーシャルアプリケーションとして利用できる。ただし、SNS アプリケーションの多くは、特定の SNS、たとえば、Facebook 用にハードコーディングがなされていることが多い。その場合は、本実行環境で動作させるためには、ハードコーディングされている部分を修正する必要がある。

本ソーシャルアプリケーション実行環境では、VPN で接続された VM に対して、DHCP で IP アドレスを割り当て、DNS サーバに SNS のユーザ名を含むホスト名を登録する。この機能は、従来の Social VPN でも提供されていた機能である。本研究では、7.1 節で述べたアプリケーションを用いてこの機能の有用性を再確認した。さらに、本研究では、この機能を新たに Web サーバ Apache 、および、OAuth に対応した任意の Web アプリケーションで利用可能にした。この機能も、有用性が高い。

本実行環境は、従来の Social VPN と同様に、オンラインのユーザ間の通信しか行えない。オフラインのユーザ間の情報交換を行うためには、7.1.1 項で述べた Friend News System を用いる必要がある。

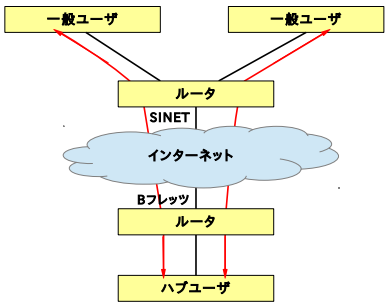

図 10 物理的ネットワーク構成

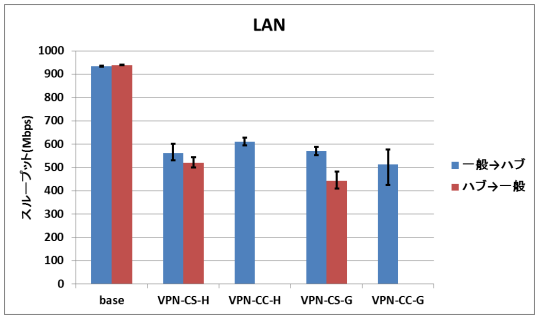

図 11 LAN 上での TCP のスループット

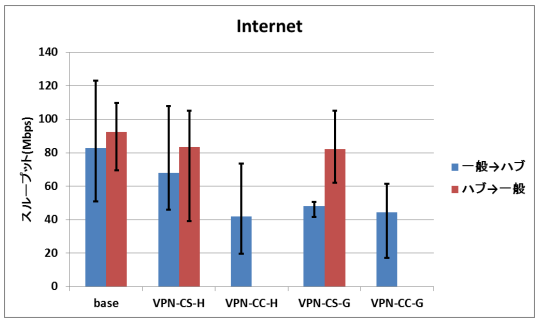

図 12 インターネット上での TCP のスループット

7.3 性　　能

本ソーシャルアプリケーション実行環境は、SoftEther VPN を用いて実現している。SoftEther VPN では、一般ユーザ間の通信であっても必ずハブユーザのコンピュータで実行されている VPN サーバを経由する。このような通信形態がアプリケーションに影響を与え

表 1 ping の結果

	LAN	インターネット
base	0.2ms	19.3ms
VPN-CS-H	1.2ms	18.8ms
VPN-CC-H	1.3ms	38.1ms
VPN-CS-G	1.5ms	20.3ms
VPN-CC-G	1.9ms	38.1ms

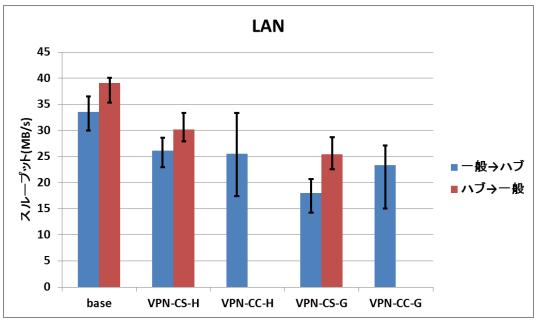

図 13 LAN 上での NFS のスループット

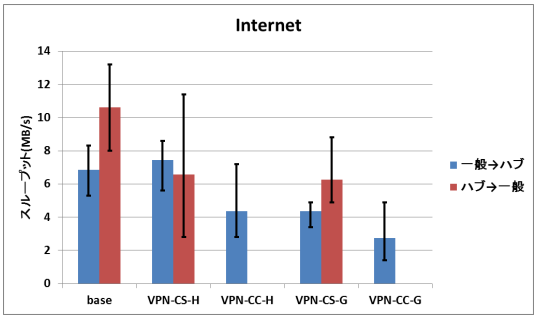

図 14 インターネット上での NFS のスループット

るのかどうか、および、現在のインターネット環境で LAN 用のアプリケーションをソーシャルアプリケーションとして利用可能かどうかを調べる実験を行った。

実験に利用したコンピュータの論理的なネットワーク構成は図 1 と同じである。物理的なネットワーク構成を図 10 に示す。参考のために、インターネットではなく LAN においても同じ実験を行った。実験に用いたネットワークは、以下の通りである。

- ハブユーザのインターネット接続: NTT フレッツ・光ネクスト ギガファミリー・スマートタイプ。最大 1Gbps。インターネット・サービス・プロバイダは、Interlink。
- 一般ユーザのインターネット接続: 筑波大学の学内ネットワークを経由して SINET で接続。

実験に用いたコンピュータの CPU は、Intel Core i7-3820 3.60GHz、メモリは、32GB、ハードディスクは、Serial ATA 3.0 接続 (Westan Digital Blue 500MB) である。オペレーティング・システムは、Ubuntu 12.04 または 14.04 である。これを、LAN には 1Gbps で接続した。ハブユーザのインターネット接続には、ブロードバンドルータ MICRO RESEARCH MR-GL2000 を用いた。SoftEther VPN のバージョンは、サーバが PacketiX VPN Server 4.0 Version 4.02、クライアントが SoftEther VPN Client 4.10 である。暗号化アルゴリズムは RC4-MD5 を使用した。通信プロトコルは UDP を使用した。また、VM を用いた時と用いない時の両方で実験を行った。

表 1、および、図 11-図 14 に結果を示す。これらの表、および、図では次の記号を用いている。

- base: VPN も VM も用いていない状態。
- VPN-CS-H: VPN を用いている。VPN のクライアントとサーバの間（一般ユーザとハブユーザの間）。VM を用いていない (Host)。
- VPN-CC-H: VPN を用いている。VPN のクライアントとクライアントの間（一般ユーザと一般ユーザの間）。VM を用いていない (Host)。
- VPN-CS-G: VPN-CS-H で、VM を用いている (Guest)。
- VPN-CC-G: VPN-CC-H で、VM を用いている (Guest)。

表 1 に、ping を用いて測定した通信遅延を示す。ping コマンドで 10 回、遅延を測定し、その平均値を求めた。LAN の結果を見ると、VPN を用いると、1 ミリ秒程度の遅延が発生する。さらに、VM を用いると、0.4 ミリ秒から 0,5 ミリ秒の遅延が加算される。これらの遅延は、インターネットの遅延の変動よりも小さい。インターネットでは、一般ユーザとハブユーザの間 (-CS-) の遅延が 18 ミリ秒から 20 ミリ秒、一般ユーザ間 (-CC-) の遅延が、38 ミリ秒であった。以上のことから、VM を利用したことによる通信遅延オーバヘッドは、今回実験に用いた高速なインターネット上でソーシャルアプリケーションを実行する場合には大きな影響がないと言える。

図 11 と図 12 に、iperf を用いて測定した TCP のスループットを示す。実験では、iperf コマンドを 10 回実行した。図 11 と図 12 は、その平均スループットと、エラーバーとして最大値、および、最小値を示している。VPN も VM も用いない場合 (base)、LAN は、一般ユーザからハブユーザの方向が平均 934Mbps、ハブユーザから一般ユーザの方向が平均 939Mbps であった。インターネットでは、一般ユーザからハブユーザの方向が、平均 82.84Mbps、ハブユーザから一般ユーザの方向が、平均 92.26Mbps であった。インターネットでは、方向により大きな差があった。また同じ方向でも、実験の度にスループットが大きく変化した。この原因は、他のインターネットユーザによる通信の影響を大きく受けているためだと思われる。

このような下位層のネットワークにおいて、VPN、および、VM を用いてソーシャルアプリケーション実行環境を構築した。その場合、インターネットでは、一般

ユーザとハブユーザの間 (VPN-CS-G) では、ハブユーザから一般ユーザの方向が 48Mbps、一般ユーザからハブユーザの方向が 82Mbps、一般ユーザ間 (VPN-CC-G) では、約 44Mbps のスループットが得られた。

このような環境で、7.1 節で述べた次のアプリケーションを実行したところ、いずれも問題なく動作した。

- Friend News System
- Empathy
- Frozen-Bubble
- DokuWiki

図 13 と図 14 に NFS の性能を示す。この実験では、次のようなコマンドを実行してスループットを測定した。

```
dd if=/dev/zero/ of=file \
bs=1000 count=10000 conv=fsync
```

図 14 に示すように、インターネットではハブホストと一般ユーザの間 (VPN-CS-G) では、4〜6MBytes/s (32〜48Mbps)、一般ユーザ間では、約 3MBytes/sec (24Mbps) のスループットが得られた。この一般ユーザ間での速度は、100KB の写真を、1 秒間に 30 枚表示できる速度である。すなわち、ハブユーザを経由する通信形態であってもこの実験で用いた高速なインターネット接続が利用可能なら、写真を表示するようなソーシャルアプリケーションを NFS で実現しても十分な性能が得られると思われる。

8. 関連研究

従来のソーシャル VPN の実装[11] との比較については、7.2 節で述べた。この章では、それ以外の関連研究について述べる。

ソーシャルネットワークのプライバシー問題に対処する Virtual private social networks[25] という研究がある。この研究では Facebook のような既存の Web ベースのソーシャルネットワークには偽の情報を表示して、内部のユーザにだけ本来の情報を表示する。この研究では本来の情報を XML ファイルに保存し、そのファイルを通信相手に XMPP サーバ経由で送る。内部のユーザが他の内部のユーザの Facebook ページにアクセスした時に特別な Web ブラウザで XML ファイルに含まれた、本来の情報を表示する。本研究ではユーザ認証には Facebook 等を用いるが、Facebook に偽の情報を出す必要はない。内部の情報は各ユーザが実行している Web サーバや NFS サーバに置くことができる。また本研究では LAN 上で動作する Web 以外のアプリケーションも利用できる。

分散型 SNS の実装方法は 1 章で述べたように、大きくサーバの連邦化と P2P ネットワークの利用に分類される。P2P ネットワークを用いる分散型 SNS のうち、任意のノードを用いるのではなく、友人のノードだけを用いるものは、F2F ネットワークと呼ばれる。たとえば、文献 26) は、F2F ネットワークを用いるバックアップとネットニュースというアプリケーションの設計を示している。本実行環境の実現では友人のノードだけを結ぶので、F2F の考え方に近い。ただし、ハブユーザが存在するので、その部分は、サーバの連邦化による実装におけるサーバと似ている。

本ソーシャルアプリケーション実行環境は、集中型 SNS をユーザ認証に用いているので、完全な分散型 SNS とは言えない。ユーザが本実行環境を使えば、分散型 SNS と同等のプライベートな通信やファイル共有を利用できる。それと同時に、ユーザは集中型 SNS の情報発信機能を利用できるという利点もある。ただしユーザは、集中型 SNS と本実行環境を明示的に使い分ける必要がある。このことは、ユーザにとって新たなツールを使わなければならないという負担が増えることを意味する。しかしながら、現在の集中型 SNS では、ユーザが適切にアクセス制御の設定を行うことは非常に難しいという指摘もある[4]。この観点では、公開用とプライベート用でツールを使い分けることには利点があるとも言える。

9. おわりに

本論文では、集中型 SNS と連携可能なソーシャル VPN について述べた。本ソーシャル VPN では、パブリックな情報は集中型 SNS で、プライベートな情報は、SNS メンバの PC と PC を接続する VPN で交換できるようにする。これにより、集中型 SNS の利点と分散型 SNS の利点 を共に利用可能にする。

従来のソーシャル VPN の実装には、ユーザ認証が連携できず利便性が低いという問題があった。本研究では、この問題を解決し利便性を向上させた。まず、SNS サーバと VPN のユーザ認証の連携の問題は Web ブラウザを使用したインターフェースにより解決した。Web アプリケーションのユーザ認証の問題は VPN のユーザ認証と連携させることにより解決した。Apache HTTP Server の認証モジュール導入の問題は OAuth サーバにより解決した。また、アプリケーション実行には VM を用いるようにする。これにより安全にアプリケーションを使用できる。

現在、SoftEther VPN に SNS でユーザ認証できる機能の実装、Web ブラウザ上での VPN 接続の開始と終了、Apache HTTP Server の認証モジュールによるユー

ザ名の取得、および OAuth サーバによるアクセストークンからのユーザ名の取得ができている。また実験により、VPN ハブを通して通信をしても、高速なインターネット接続ならいくつかの LAN 用アプリケーションを問題なく動作させられることがわかった。

今後は VPN サーバの分散化を行いたいと考えている。現在の実装では VPN サーバが集中化している。このため VPN サーバが停止した場合、VPN 通信ができなくなる。また誰が VPN サーバを動かし続けるのかという問題もある。この問題を解決するために、VPN サーバをオンデマンドで実行できるようにする。VPN サーバは各ユーザが持ち、必要なときに必要なサーバだけが起動するようにする。また、友人間で誰が VPN サーバを動かすのかを決定する機能を作成する。

参 考 文 献

1) Datta, A., Buchegger, S., Vu, L., Strufe, T. and Rzadca, K.: Decentralized Online Social Networks, *Handbook of Social Network Technologies and Applications*, pp. 349–378 (2010).
2) Schwittmann, L., Wander, M., Boelmann, C. and Weis, T.: Privacy Preservation in Decentralized Online Social Networks, *IEEE Internet Computing*, Vol. 18, No. 2, pp. 16–23 (2014).
3) Greenwald, G.: *No Place to Hide: Edward Snowden, the NSA, and the US Surveillance State*, Metropolitan Books (2014).
4) Liu, Y., Gummadi, K. P., Krishnamurthy, B. and Mislove, A.: Analyzing Facebook Privacy Settings: User Expectations vs. Reality, *the 2011 ACM SIGCOMM Conf. on Internet Measurement*, pp. 61–70 (2011).
5) Bielenberg, A., Helm, L., Gentilucci, A., Stefanescu, D. and Zhan, H.: The growth of Diaspora – A decentralized online social network in the wild, *IEEE Infocom Workshops*, pp. 13–18 (2012).
6) Sahama, T., Liang, J. and Iannella, R.: 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).
7) Baden, R., Bender, A., Spring, N., Bhattacharjee, B. and Starin, D.: Persona: An Online Social Network with User-defined Privacy, *the ACM SIGCOMM 2009 Conf. on Data Comm.*, pp. 135–146 (2009).
8) Buchegger, S., Schiöberg, D., Vu, L.-H. and Datta, A.: PeerSoN: P2P Social Networking: Early Experiences and Insights, *the Second ACM EuroSys Workshop on Social Network Systems*, pp. 46–52 (2009).
9) Cutillo, L.A., Molva, R. and Strufe, T.: Safebook: A Privacy-preserving Online Social Network Leveraging on Real-life Trust, *IEEE Communications Magazine*, Vol. 47, No. 12, pp. 94–101 (2009).
10) Graffi, K., Gross, C., Stingl, D., Hartung, D., Kovacevic, A. and Steinmetz, R.: LifeSocial. KOM: A secure and P2P-based solution for online social networks, *Consumer Communications and Networking Conference (CCNC), 2011 IEEE*, IEEE, pp.554–558 (2011).
11) Figueiredo, R., Boykin, P., Juste, P. and Wolinsky, D.: Integrating Overlay and Social Networks for Seamless P2P Networking, *Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE '08. IEEE 17th*, pp.93 –98 (2008).
12) Cahill, C., Canales, C., Le Van Gong, E. H. A., Madsen, P. and Maler, N. E.: Liberty Alliance Web Services Framework: A Technical Overview, *Liberty Alliance* (2008).
13) Morgan, R., Cantor, S., Carmody, S., Hoehn, W. and Klingenstein, K.: Federated Security: The Shibboleth Approach., *The Educause Quarterly*, Vol. 27, No. 4, pp. 12–17 (2004).
14) SoftEther VPN プロジェクト: `http://ja.softether.org`. 2014-10-17.
15) OpenSSL Project: OpenSSL: The Open Source toolkit for SSL/TLS, `https://www.openssl.org/`. 2014-10-15.
16) Internet Systems Consortium: BIND name server software, `http://www.isc.org/downloads/bind/`. 2014-05-21.
17) Internet System Consortium: ISC DHCPD Dynamic Host Configuration Protocol Daemon, `http://www.isc.org/downloads/dhcp/`. 2014-08-12.
18) Hardt, D. and Microsoft: The OAuth 2.0 Authorization Framework, *Internet Engineering Task Force Request for Comments: 6749* (2012).
19) Shaffer, B.: OAuth2 Server PHP, http://bshaffer.github.io/oauth2-server-php-docs/. 2014-10-17.
20) Shinjo, Y., Kunyao, X., Kainuma, N., Nobori, D. and Sato, A.: Friend News System: A Modern Implementation of Usenet over Social VPNs, *The 7th IEEE International Conference on Social Computing and Networking* (2014). (To appear).
21) GNOME.org: Apps/Empathy - GNOME Wiki!, `https://wiki.gnome.org/Apps/Empathy`. 2014-10-17.
22) Cottenceau, G., Younes, A., Bidan, M.L., Kim, D.J. and Amblard-Ladurantie, A.: Frozen Bubble - the official home, `http://www.frozen-bubble.org/`. 2014-10-17.
23) Gohr, A. and the DokuWiki Community: DokuWiki

open source wiki software, https://www.dokuwiki.org/start?id=dokuwiki. 2014-04-26.
24) Gohr, A.: OAuth Plugin for DokuWiki, https://www.dokuwiki.org/plugin:oauth. 2014-10-14.
25) Conti, M., Hasani, A. and Crispo, B.: Virtual private social networks, *Proceedings of the first ACM conference on Data and application security and privacy*, pp. 39–50 (2011).
26) Li, J. and Dabek, F.: F2F: Reliable Storage in Open Networks, *The 5th International Workshop on Peer-to-Peer Systems* (2006).