自律的なソーシャルVPNの設計と実装

田中 正規[1]　新城 靖[1]　佐藤 聡[1]　中井 央[1]

概要：現在広く利用されている SNS(Social Networking Service) は，ユーザのプライバシ侵害の問題が指摘されている．この問題を解決するために分散型 SNS が提案されている．その実装手法の 1 つに，友達間の PC を VPN で接続するソーシャル VPN がある．従来のソーシャル VPN には，友達関係のグループ化やユーザを発見し友達関係を構築する機能が，特定の外部サービスに依存しているという問題がある．
本稿では，これらの問題を解決する自律的なソーシャル VPN の設計と実装について述べる．本研究におけるソーシャル VPN では，ユーザは自身の視点で友達との関係性に応じたグループを作成し，そのグループにドメイン名を割り当てることができる．これにより，ユーザはソーシャルアプリケーションでドメイン名を用いたアクセス制御を利用できる．更に友達のノードを介することで，特定の外部サービスに依存せずにソーシャル VPN 上のユーザを発見できる．このソーシャル VPN を，P2P(Peer-to-Peer) 型 VPN ソフトウェアである TincVPN を用いて実現する．

1. はじめに

近年，コミュニケーションや情報発信の手段として SNS(Social Networking Service) の利用が進んでいる．多くの SNS は集中型 SNS であり，プロファイルや投稿コンテンツ等から構成されるユーザのデータは，運営組織の管理下にある中央サーバで管理される．そのため，集中型 SNS のユーザに対するプライバシ侵害の問題が指摘されている [1]．集中型 SNS においては，ユーザは自身のデータの送信先である中央サーバを信頼する．しかし 2013 年には，米国政府機関が集中型 SNS に対して不当な監視を行っていたことが告発されている [2]．そのため集中型 SNS では，自身のプライバシを保護しつつサービスを利用することは難しい．このような問題を解決するために，分散型 SNS が提案されている [3]．分散型 SNS においては，ユーザのデータを自身または信頼できる友達のストレージに保存することで，プライバシ問題に対処している．

本研究ではソーシャル VPN という実装手法を対象とする．ソーシャル VPN とは，友達関係にあるユーザの PC 間を VPN で接続した分散型 SNS である．ソーシャル VPN では，ユーザは既存の LAN 用アプリケーションを自身の PC 上で動かすことで，そのままソーシャルアプリケーションとして利用できる．また友達の PC に対して，友達の名前を含むドメイン名でアクセスできる．

従来のソーシャル VPN には次のような問題点がある．まず，ソーシャル VPN 上のアカウントが，特定の外部サービスに依存しているという問題がある．先行研究における実装では，Facebook や Twitter 等のサービスを利用することで，VPN 接続に必要な IP アドレスの交換およびユーザ認証を実現している [4]．そのためユーザはソーシャル VPN の利用において，このような外部サービスを全面的に信頼する必要がある．外部サービス上のアカウントが削除された場合，ユーザはソーシャル VPN のサービスが利用できなくなる．

従来のソーシャル VPN では，友達関係を外部サービスの機能を用いてグループ化できる．しかしそのようなグループは，ソーシャルアプリケーションのアクセス制御には利用できない．そのためアクセス制御の設定では友達を列挙する必要があり，友達の数の増加に従って設定が煩雑化するという問題がある．

また，従来のソーシャル VPN には，ユーザを発見し友達関係を構築する機能が不十分であるという問題もある．集中型 SNS とアカウントを連携することで，集中型 SNS のユーザを発見する機能や友達候補のユーザのレコメンデーションをそのまま利用することが可能である．しかし異なる外部サービスに登録したユーザ間では，相手の発見および友達関係の構築ができない．

以上の問題を解決するため，本研究では自律的なソーシャル VPN の実現を目的とする．ここで自律的とは，特定の外部サービスに依存することなく次のような要件を満たすことである．

(1) ユーザは自ら定義した友達関係のグループをソーシャ

[1] 筑波大学
University of Tsukuba

ルアプリケーションのアクセス制御に利用できる
(2) ユーザはソーシャル VPN 上でユーザを発見し新たに友達関係を構築できる

これらの要件を，本研究では次の機能を提供することで実現する．

- 独立性の高い手段による友達の公開鍵および IP アドレスの取得
- 仮想的なドメイン名による友達関係の管理
- 友達のノードを介したユーザの発見および友達関係の構築

本研究ではこれらの手法に基づいて，P2P 型 VPN ソフトウェアである TincVPN を用いてソーシャル VPN を実装する．そして実装したソーシャル VPN が，上で述べた自律の要件を満たすことを示す．

本稿の構成は次の通りである．まず 2 章では，自律的なソーシャル VPN の概要について述べる．3 章では，Linux による本研究のソーシャル VPN の実装について述べる．4 章では，本研究のソーシャル VPN において利用可能なソーシャルアプリケーションの例について述べる．5 章では，実装したソーシャル VPN の評価について述べる．6 章では，本研究に関連する研究について述べる．7 章では，まとめと今後の課題について述べる．

2. 自律的なソーシャル VPN の概要

本研究ではソーシャル VPN の手法の内，ソーシャルルータ [5] の方式を用いる．この方式においては，ソーシャルアプリケーションを実行する PC と，SNS の基本機能を提供するルータは分離される．このルータに，友達間の通信に必要なルーティング機能とアクセス制御のためのパケットフィルタ機能を実装し，インターネットと LAN の間に設置することでソーシャル VPN を利用可能にする．このルータをソーシャルルータと呼ぶ．ソーシャルルータを導入することで，ユーザは複数の PC や携帯端末を，特定のプログラムをインストールすることなくソーシャル VPN に参加させることが可能である．

本研究のソーシャルルータでは，登録する友達に対して，プライベート IP アドレスのサブネットを自動的に割り当てる．ユーザの PC からは，そのサブネットに属する IP アドレスが友達の PC に割り当てられているように見える．

ソーシャルルータを用いて 2 つの LAN を VPN で接続すると，IP アドレスが衝突することがある．この問題を解決するために，本研究では TwiceNAT[6] を用いる．TwiceNAT とは，IP パケットの送信元アドレスと宛先アドレスの両方を変換する NAT(Network Address Translation) である．

ユーザはソーシャル VPN を利用する際に，ニックネーム，コミュニティ，連絡先，住んでいる地域等のプロファイルのデータおよびソーシャル VPN に参加させる PC のデータをソーシャルルータに登録する．本研究のソーシャルルータは Web ユーザインタフェースを提供する．そのスクリーンショットを図 1 に示す．図 1 では，左サイドカラムにフレンドリスト，メインカラムにニックネーム，ユーザ ID，自己紹介およびデバイスリストが表示されている．デバイスリストは，ユーザが自身のソーシャルルータに対して登録した PC の名前と IP アドレスのリストである．フレンドリストについては 2.2 節および 2.3 節で述べる．

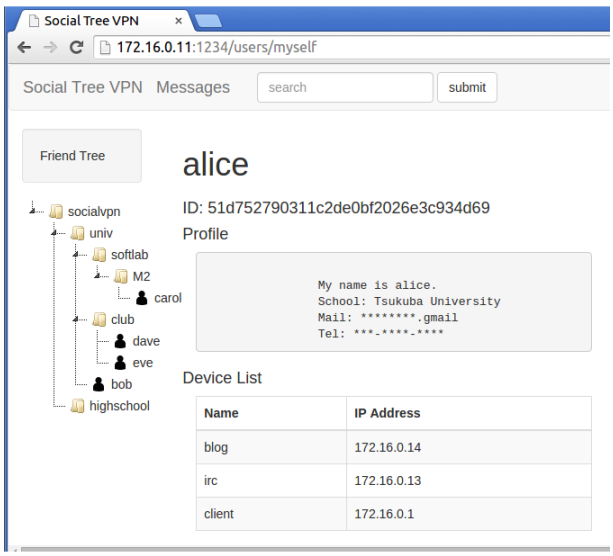


図 1 Web ユーザインタフェースのトップページ

2.1 独立性の高い手段による友達の公開鍵および IP アドレスの取得

本研究のソーシャルルータはユーザを公開鍵により識別する．各ユーザは 1 つの公開鍵を持つ．2 人のユーザが友達関係にあるとは，互いに相手の公開鍵を持ち，互いに友達であると設定していることを意味する．

ソーシャルルータに友達を登録する時，ユーザは次の情報を与える．

- ニックネーム
- 友達の公開鍵
- 友達の IP アドレス，またはそれを取得するための手段
- 友達を分類するグループ

IP アドレスは，ユーザが家庭用ルータ等を通してインターネットに接続している場合，不定期に変化することが想定される．こうした変化に対して，本研究では IP アドレスの登録と参照を行える独立性の高いサービスを利用する．分散型 SNS においては，このようなサービスをランデブ・ポイント [7] と呼ぶ．本研究ではランデブ・ポイントとして，Resilio Sync[8] 等の独立性の高いファイルストレージサービスや DDNS(Dynamic DNS) サービスを利用する．ユーザはランデブ・ポイントの種類と識別子を，友達と互

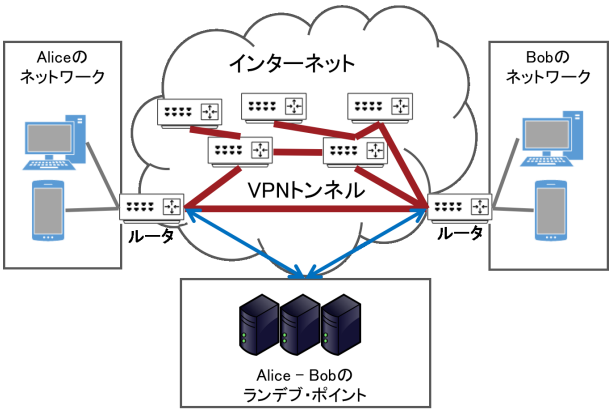


図 2 ソーシャルルータのネットワーク構成

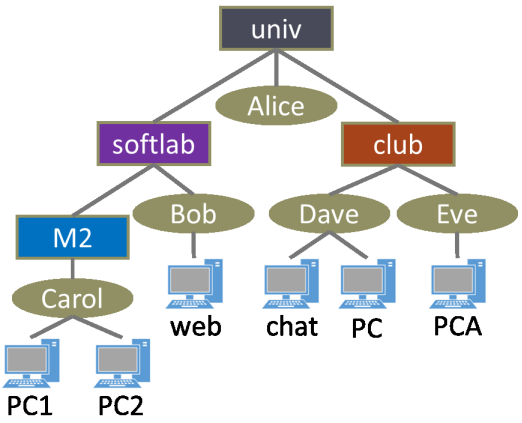


図 3 仮想的なドメイン

いに交換する．そしてそのランデブ・ポイントを利用することで，不定期に変化する IP アドレスを交換する．また本研究室の蛸井らにより開発されている共通ランデブ・ポイント・インターフェース [7] を利用することで，様々なランデブ・ポイントを統一的に利用可能にする．

従来の手法では，ソーシャル VPN は特定の外部サービスを利用することで VPN 接続時のユーザ認証を行う．それに対して提案手法では，VPN 接続時のユーザ認証に登録した友達の公開鍵を使うため，特定の外部サービスに依存することはない．

本研究で実現するソーシャルルータのネットワーク構成を図 2 に示す．ソーシャル VPN 全体では，ソーシャルルータをノード，ソーシャルルータ間で接続された VPN をエッジとして，非構造化 P2P ネットワークを構成する．ユーザは自身を中心として，ネットワーク中のノードを自身，友達，それ以外のユーザの 3 種に大別する．そして友達に対して，自身との関係性に基づいたドメイン名を独自に割り当て，友達の PC にそのドメイン名でアクセス可能にする．

2.2 仮想的なドメイン名による友達関係の管理

本研究では仮想的なドメイン名による友達関係の管理手法を提案する．提案手法では，ユーザは自身との関係性に基づいて友達をグループ化し，そのグループを自身の DNS サーバ上に構築される階層構造の中間節とする．ユーザは友達の PC にアクセスする時，IP アドレスではなく仮想的なドメイン名を用いることができる．またソーシャルアプリケーションのアクセス制御の設定において，仮想的なドメイン名を指定することで，グループに対するアクセス制御を可能にする．

このような仮想的なドメイン名の例を図 3 に示す．図 3 における Carol の持つ PC である PC1 の FQDN は，PC1.Carol.M2.softlab.univ である．この表現により，PC1 を持つ Carol を univ グループかつ softlab.univ グループかつ M2.softlab.univ グループに分類していることを記述している．

```
/home/report *.softlab.univ.socialvpn(rw)
/home/manual *.club.univ.socialvpn(rw)
```

図 4 グループを指定した公開ディレクトリの設定

仮想的なドメイン名を利用したアクセス制御の例としては，NFS(Network File System) を利用した友達間でのファイル共有が挙げられる．ユーザが NFS サーバを動作させる時，PC 上の`/etc/exports` というファイルでマウント可能なホストのドメイン名を指定する．この時ワイルドカードを用いることで，グループに対するアクセス制御の設定が行える．このようなアクセス制御の設定例を図 4 に示す．図 4 の例では，`/home/report` というディレクトリを softlab.univ.socialvpn グループに，`/home/manual` というディレクトリを club.univ.socialvpn グループに読み書き可能で公開している．

各ユーザは，このような仮想的なドメイン名を，自身のソーシャルルータ上の DNS サーバで管理し，他のユーザに参照させない．そのため各ユーザは，このドメイン名を自由に設定することができ，かつどのグループに分類したかという情報を他のユーザに対して秘密にできる．

階層構造はアクセス制御に加えて，ユーザが友達関係を管理する場合においても有用である．図 1 の左サイドカラムで示す通り，本研究のソーシャルルータの Web ユーザインタフェースは，友達関係を階層構造で表示している．これにより，ユーザは一般的なディレクトリ操作と似たユーザ体験で，友達関係を確認できる．また階層構造を用いたグループ化により，あるグループ以下に属する友達に対する一括操作も可能である．具体的には，ソーシャルアプリケーションの通信を許可するポート番号の設定の一括操作を可能にしている．例えば大学関係者にのみアクセスを許可したい Web サービス等がある場合，ユーザは univ.socialvpn ドメインと 80 番ポートを指定する．

2.3 友達のノードを介したユーザの発見および友達関係の構築

提案手法では，ユーザはフレンドリストを含む自身のプロファイルをソーシャル VPN 上のユーザに公開する．各ユーザは友達のフレンドリストを介して，友達の友達のプロファイルにアクセスできる．そのプロファイルのフレンドリストから，更に友達の友達の友達のプロファイルにアクセスできる．このような再帰的なアクセスにより，ユーザは自身のソーシャルルータを起点として，ソーシャル VPN のネットワークを辿って他のユーザを発見できる．またユーザを発見するまでに辿った経路を記録することで，そのユーザへメッセージを送ることも可能である．

ユーザのプロファイルには，公開鍵を含む友達関係を構築するためのデータも含まれる．フレンドリストを介してユーザを発見した時に，このデータも併せて取得することが可能である．またフレンドリストを介してメッセージを送る際に，自身のプロファイルも含めることが可能である．

本研究では，友達のフレンドリストに含まれるプロファイルを，自身のソーシャルルータに収集することで，キーワード検索によるユーザの発見を可能にする．この機能は，特定のコミュニティに関連する友達の友達と新たに友達関係を構築したい場合に有用である．例えば筑波大学に関連するユーザを発見したい場合，”筑波大学”をキーワードとして収集したプロファイルを検索することで，筑波大学の学生や教員等のユーザを発見することが可能である．

以上の手法に基づいて，2 人のユーザ Alice と Bob がソーシャル VPN 上で友達関係を構築する手順は次の通りである．

(1) Alice はフレンドリストを介した探索や収集したプロファイルへの検索により，友達候補 Bob のプロファイルを取得する
(2) Alice は自身のソーシャルルータに，Bob の公開鍵等の友達関係の構築に必要なデータを登録する
(3) Alice は Bob に対して，友達になることを希望するメッセージを，自身のプロファイルと併せて送信する
(4) Bob はメッセージを受信後，メッセージが送られてきた経路と，それを通して得られた Alice のプロファイルを確認し，友達になるかを判断する
(5) Bob は自身のソーシャルルータに Alice の公開鍵等の友達関係の構築に必要なデータを登録する

以上の通り，フレンドリストを介した友達の発見およびメッセージの送受信機能により，友達関係の構築および VPN 接続を，ソーシャル VPN 上のコミュニケーションだけで完結させることができる．

なお，場合によっては，あるユーザと友達関係にあることを他のユーザに秘匿したいことがある．本研究のソーシャルルータでは，ユーザが互いに友達関係を公開するか否かを設定できる機能を提供している．この機能により，友達関係にある 2 人のユーザは，他のユーザに対して友達関係を秘匿しつつソーシャルアプリケーションで交流できる．

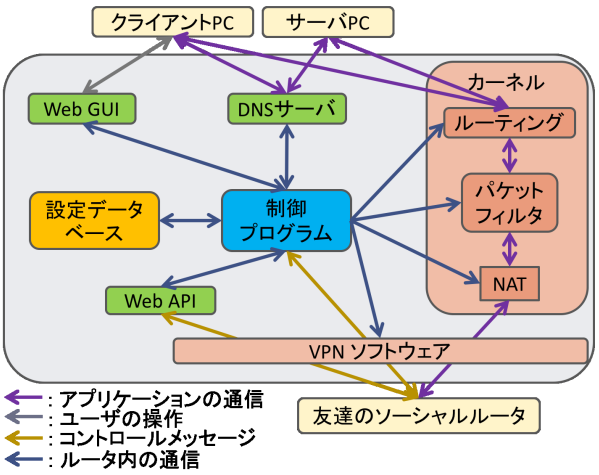


図 5 ソーシャルルータの構成

3. Linux におけるソーシャルルータの実装

本研究のソーシャルルータは Linux 上で実装している．ソーシャルルータは，次の要素から構成される．

- 外部とのインタフェース
  - Web ユーザインタフェース
  - Web API
  - DNS サーバ
- 制御プログラム
- VPN ソフトウェア
- カーネル内のネットワーク機能
  - パケットフィルタ
  - ルーティング
  - NAT
- 設定データベース

各構成要素間の，通信および制御の流れの概略を図 5 に示す．図 5 には次のエンティティが描かれている．

- クライアント PC
  ユーザはクライアント PC を用いて Web ユーザインタフェースにアクセスし，自身のソーシャルルータを操作する．また友達のサーバ PC に対して，クライアント PC でアクセスする．クライアント PC はソーシャルルータ上の DNS サーバ対して問い合わせることで，友達のサーバ PC の IP アドレスを得てアクセスする．
- サーバ PC
  ブログやチャットサーバ等のソーシャルアプリケーションは，ソーシャルルータに登録したサーバ PC 上で動作する．友達のクライアント PC からアクセスされた時，サーバ PC はソーシャルルータ上の DNS サーバに対して問い合わせることで，友達のクライアント PC の逆引きを行う．

- 友達のソーシャルルータ
  このソーシャルルータと友達のソーシャルルータは，互いにコントロールメッセージを送受信する．コントロールメッセージとは，SNS の基本機能を実現するための通信である．具体的には，友達のプロファイルの取得，ユーザの探索，ユーザ間のメッセージ送信の機能を実現するための通信である．

次の節以降で，ソーシャルルータの構成要素について詳しく述べる．

3.1 外部とのインタフェース

外部とのインターフェースには，クライアント PC がアクセスする Web ユーザインタフェース，クライアント PC およびサーバ PC がアクセスする DNS サーバ，友達のソーシャルルータがアクセスする Web API がある．Web ユーザインタフェースおよび DNS サーバは LAN に，Web API は VPN を通して友達のソーシャルルータにのみ公開されている．これらのインターフェースは，ソーシャル VPN と関連のない外部からのアクセスを許可しない．

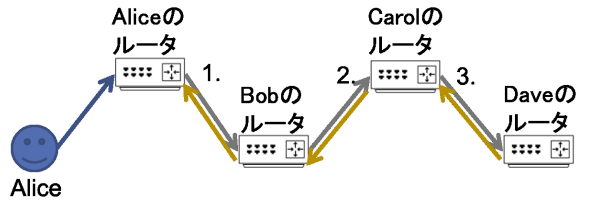
ソーシャルルータは Web API を通して，互いにコントロールメッセージを送受信する．コントロールメッセージには，要求メッセージとそれに対する応答メッセージがある．

ソーシャルルータの要求メッセージは次の種類がある．

- `GET /registrant_info`
  ユーザのデータを取得する．ユーザのデータはプロファイル，デバイスリスト，TwiceNAT の設定に用いる IP アドレスが含まれる．
- `GET /friend_list`
  ユーザのフレンドリストを取得する．フレンドリストには各友達のプロファイルが含まれる．
- `POST /messages`
  ユーザへメッセージを送信する．

ソーシャルルータは要求メッセージを，別の友達のソーシャルルータに中継することができる．中継した要求メッセージに対する応答メッセージが送られてくると，ソーシャルルータは要求メッセージを送ってきた友達のソーシャルルータに返す．

要求メッセージの中継の概略を図 6 に示す．図 6 では，Alice のソーシャルルータが Bob および Carol を経由して，Dave のソーシャルルータに要求メッセージを送信している．要求メッセージには path および via パラメータを渡す．パラメータ path は，ソーシャルルータへの連鎖的なアクセスを行うための経路を示す文字列である．またパラメータ via は，経由したソーシャルルータを示す文字列である．path および via パラメータは，UUCP(Unix to Unix Copy) と類似の表記方法として，区切り文字「!」で連結されたユーザ ID のリストが渡される．ユーザ ID はユーザの公開鍵のハッシュ値から生成され，ソーシャル VPN 全体におけるソーシャルルータの識別子として用いられる．各ソーシャルルータはメッセージを中継する時，path パラメータの先頭の自身の ID を取り去り，via パラメータに自身の ID を追加する．そしてこの先頭のユーザを，次のリクエスト先のユーザとして指定する．via パラメータは送信元のユーザから宛先ユーザまでの経路を保存する．この経路は受信したメッセージに返信する際等に用いる．



要求メッセージ
1. GET /friend_list?path=Carol!Dave&via=Alice
2. GET /friend_list?path=Dave&via=Alice!Bob
3. GET /friend_list?path=&via=Alice!Bob!Carol

図 6 フレンドリストへの連鎖的なアクセス

3.2 制御プログラム

制御プログラムは，ソーシャルルータの設定およびコントロールメッセージの送受信を行う中心的なプログラムである．

制御プログラムは，Web ユーザインタフェースからのアクセスにより設定データベースを更新する．必要ならば続けて外部コマンドを実行することで，VPN ソフトウェアおよびカーネル内のネットワーク機能の設定を更新する．

制御プログラムは，DNS サーバからのアクセスにより設定データベースからドメイン名または IP アドレスの情報を取得する．この時制御プログラムは，設定データベースに保存されているデータを，DNS サーバにおける応答メッセージに整形する．

友達のソーシャルルータが Web API へ要求メッセージを送信した時，自身へのリクエストであれば，制御プログラムは設定データベースからユーザのプロファイルを取得する等の処理を行う．他のソーシャルルータへのリクエストであれば，制御プログラムは該当する友達のソーシャルルータの Web API に対して要求メッセージを送信する．

制御プログラムは定期的に友達のソーシャルルータの Web API に要求メッセージ `GET /registrant_info` および `GET /friend_list` を送り，取得したデータを設定データベースに保存する．取得するデータは，フレンドリストを含む友達のプロファイルである．設定データベースに収集されたこれらのデータは，友達のプロファイルページの表示やキーワード検索時に参照される．

3.3 VPN ソフトウェア

本研究のソーシャル VPN を，P2P 型 VPN ソフトウェアの TincVPN[9] を用いて実装する．制御プログラムは TincVPN の設定ファイルを出力する．その設定ファイルを元に, TincVPN は友達のソーシャルルータとの間で VPN 接続を行う．接続後は友達毎に 1 つの仮想ネットワークインターフェースが生成される．本研究の実装では，各ソーシャルルータは TincVPN のインスタンスを 1 つ生成し，ソーシャル VPN 全体で単一のネットワークを構成する．

TincVPN では，仮想ネットワークインターフェースへの仮想 IP アドレスの割り当ては，VPN 接続の前に行う必要がある．本研究では，IP アドレスの割り当てを行う外部サービスを想定しないため，各ソーシャルルータは自律的に仮想 IP アドレスを決定する必要がある．本研究の実装では仮想 IP アドレスの衝突を防ぐため，自身のユーザ ID から IPv6 ユニークローカルアドレスを生成し，仮想ネットワークインターフェースへの割り当てを行う．更に IPv4 で動作するソーシャルアプリケーションのルーティングを行うため，友達間では IPv6 トンネリング [10] により IPv4 パケットのカプセル化と転送を行う．

制御プログラムは，外部コマンドとして tinc コマンドを実行して，TincVPN の設定ファイルを更新する．友達の登録時は，友達の IP アドレスと公開鍵を含む設定ファイルを出力し，友達のソーシャルルータ上の TincVPN に接続する．友達の削除時は，該当する VPN を切断し，友達の設定ファイルを削除する．

3.4 カーネル内のネットワーク機能

本研究の実装では，Linux カーネル内のネットワーク機能の設定を変更することで，PC 間の通信を制御する．

ルーティングは，友達の登録時に割り当てたサブネット宛のパケットを，3.3 節で述べた IPv6 トンネリングにおける仮想ネットワークインターフェースから送信するよう経路を設定する．

パケットフィルタは，Netfilter の Input チェインと Forward チェインを用いる．Input チェインでは，LAN 側からの通信および TincVPN の待ち受けポートへの通信のみを許可する．Forward チェインでは，LAN 側からの通信および友達に公開するソーシャルアプリケーションのポートへの通信のみを許可する．

NAT は，登録した PC の実際の IP アドレスと，友達のソーシャルルータにおいて自身に対して割り当てられたサブネットの IP アドレスの変換を行う．友達が自身にどのサブネットを割り当てているかという情報は，3.1 節で述べた通り，ソーシャルルータ間のコントロールメッセージにおいて確認する．

制御プログラムは，外部コマンドとして iptables および iproute2 コマンドを実行して，友達毎にルーティングおよびパケットフィルタの設定を更新する．またグループに対するパケットフィルタの一括操作においては，制御プログラムはまず指定されたグループに属する友達のデータを設定データベースから取得する．そして取得した友達のデータ毎に外部コマンドを実行して設定を更新する．

4. ソーシャルアプリケーションの例

ソーシャル VPN では，TCP/IP で動作する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できる．本章では，動作を確認したソーシャルアプリケーションの具体例について述べる．

4.1 NFS を利用したファイル共有

2.2 節で述べた通り，ユーザは NFS サーバを動作させることで，友達に対してその PC 上のディレクトリを公開することができる．友達のディレクトリをマウントする場合，ユーザは NFS クライアントで友達の PC のドメイン名を用いて NFS サーバを指定する．

4.2 IRC(Internet Relay Chat) を利用したプライベートチャット

ユーザは IRC サーバを動作させることで，友達のみが参加できるプライベートチャットを提供できる．また IRC サーバの管理者は，特定のドメイン名を持つユーザのみ入室を許可するよう設定することができる．これにより，グループ限定のチャットルームを開設することが可能である．

4.3 プライベートブログ

ユーザは WordPress 等のブログプラットフォームを導入した Web サーバを動作させることで，友達のみが閲覧できるプライベートブログを開設できる．WordPress はプラグインによる機能拡張の手段を提供しているため，本研究ではドメイン名によるアクセス制御設定を行うプラグインを実装した．これを用いることで，友達がアクセスした時にアクセス権限のない記事を除外して表示することが可能である．

WordPress においては，データの入出力をトリガとしてプラグインの関数を実行する仕組みがある．このプラグインでは，記事を取得するクエリの実行が完了した時に実行される posts_results のフックを使って，アクセス権限のない記事のフィルタリングを実施する．

まずユーザは記事の投稿時に，追加したメニューボックスから，アクセスを許可するドメイン名の一覧を記述する．その後ブログに友達がアクセスした時，posts_results でプラグインの関数 `check_domain_access_allow` が実行される．`check_domain_access_allow` には，記事を取得するクエリの実行結果の配列である posts が引数として渡される．この関数は，まず `gethostbyaddr` 関数によりリモートホス

図 7 Carol が Alice のブログにアクセスした場合

図 8 Dave が Alice のブログにアクセスした場合

トのドメイン名を得て，更にそれから友達のドメイン名を取得する．続けて posts に対して繰り返し処理を行い，配列の要素である記事毎に，友達のドメイン名と許可するドメイン名とを比較する．友達のドメイン名が含まれていない場合，アクセス権限がないことを意味するので，その記事を posts から取り除く．最後に，アクセス権限がない記事を全て取り除いた posts を `check_domain_access_allow` の返り値とする．

図 1 のフレンドリストで示される友達を持つ Alice が，記事の投稿時にドメイン名によるアクセス制御を設定した際の結果を図 7 および図 8 に示す．Alice は友達 Carol と Dave に対して，次のようなドメイン名を与えている．

Carol

carol.M2.softlab.univ.socialvpn

Dave

dave.club.univ.socialvpn

Alice は，各記事に対して次のような ACL を設定している．

(1)「大学の学食のメロンパンがおいしい」
univ.socialvpn

(2)「論文の進捗状況」
M2.softlab.univ.socialvpn

(3)「合気道部の引き継ぎ」
club.univ.socialvpn

図 7 に，Carol が自分の PC で Alice のブログにアクセスした画面を示す．その PC は，pc1.carol.M2.softlab.univ.socialvpn というドメイン名を持つ．Carol がブログにアクセスすると，記事 (1) および記事 (2) が表示される．

図 8 に，Dave が自分の PC で Alice のブログにアクセスした画面を示す．その PC は，pc1.dave.club.univ.socialvpn というドメイン名を持つ．Dave がブログにアクセスすると，記事 (1) および記事 (3) が表示される．

以上の通り，本研究で実装したプラグインの導入により，友達関係に基づくアクセス制御をプライベートブログにおいて容易に実現することができる．

5. 評価

本研究で実装したソーシャル VPN が，1 章で述べた次の要件を，特定の外部サービスに依存することなく満たすか評価する．

(1) ユーザは自ら定義した友達関係のグループをソーシャルアプリケーションのアクセス制御に利用できる
(2) ユーザはソーシャル VPN 上でユーザを発見し新たに友達関係を構築できる

本研究ではユーザに仮想的なドメイン名を割り当て，逆引きによりこの情報を取得可能にしている．ソーシャルアプリケーションのサーバはアクセスしてきたクライアントを IP アドレスで識別し，ソーシャルルータ上の DNS サーバに問い合わせることで，その友達のドメイン名を取得できる．この情報はドメイン名の階層構造に基づき，どのグループに分類されているかも併せて示している．ドメイン名に対するアクセス制御は，4 章で述べた通り，サーバアプリケーションに対するアクセス制御の手段として利用可能である．そのため，従来のソーシャル VPN における友達毎のアクセス制御と比較して，本研究のソーシャル VPN の方が設定が容易である．また階層構造を用いた友達管理により，2.2 節で述べた通り，ソーシャルルータのパケットフィルタの設定もグループに対する一括操作で行える．そのため，ポート単位でのアクセス制御においても，仮想的なドメイン名による友達関係の管理は有効である．これらの機能は特定の外部サービスに依存しない．以上より，提案手法および実装は要件 (1) を満たす．

本研究で提案するソーシャル VPN は，ユーザを中心とする非構造化 P2P ネットワークを構成し，ネットワークのノードには各ユーザのソーシャルルータが対応している．公開鍵等の友達関係の構築に必要なデータは自身のソーシャルルータ上で管理する．そのためソーシャルルータを連鎖的にアクセスすることで，ソーシャルグラフを辿ってユーザを発見し，そのユーザの友達関係の構築に必要なデータを取得できる．また友達のフレンドリストに含まれるプロファイルを，自身のソーシャルルータに収集することで，キーワードによるあいまい検索も可能である．ユーザは，友達関係を構築したいユーザのメールアドレスや電話番号等の Out of Bands な連絡手段を知っている場合，それらを使って公開鍵等の友達関係の構築に必要なデータを交換できる．またメッセージ機能により，ソーシャル VPN 上のコミュニケーションによる友達関係の構築もできる．これらの機能は特定の外部サービスに依存しない．以上より，提案手法および実装は要件 (2) を満たす．

現在までに実装したソーシャル VPN には次のような制約がある．第 1 に，ユーザはソーシャルアプリケーションを動作させるサーバ PC を維持管理する必要がある．そのため集中型 SNS におけるソーシャルアプリケーションと比較して，可用性を維持することが難しい．ただしソーシャルルータとサーバ PC は分離しているため，サーバ PC がダウンしても分散型 SNS としての基本機能は影響を受けない．

第 2 に，現在の実装では，SNS ユーザ全体を対象とする高度なキーワード検索を行うことができない．そのため集中型 SNS における検索機能と同等の機能性は有さない．

第 3 に，現在の実装では，コントロールメッセージを中継する場合，中継するソーシャルルータに対してそのコントロールメッセージを秘匿することができない．そのためソーシャル VPN 上のユーザにメッセージを送信した場合，中継するソーシャルルータのユーザにもそのメッセージが読まれる可能性がある．また現在の実装においては，中継されるデータが改竄された場合，それを検出できない．この問題への対策としては，中継するソーシャルルータでコントロールメッセージへのデジタル署名を行うことが挙げられる．また中継するメッセージの経路を記録することで，ソーシャル VPN 上のソーシャルグラフに関する情報を収集される可能性もある．2.3 節で述べた通り，本研究のソーシャルルータでは，友達関係にある 2 人のユーザが秘匿することに合意している場合，その友達関係を非公開にすることができる．しかし多くのユーザが友達関係を非公開にすると，ユーザを発見し新たに友達関係を構築する機能が有効に機能しなくなる．

6. 関連研究

6.1 ソーシャル VPN とその他の分散型 SNS との比較

ソーシャル VPN 以外の分散型 SNS の実装手法は，主に次の 2 種に大別される [3].

- サーバの連邦化
  複数の独立した管理者が管理するサーバ群を基盤とする SNS．ユーザは自身が信頼する組織，または自身が管理するサーバにデータを保存する．実装例としては Diaspora[11]，Vegas[12] が挙げられる．
- DHT(Distributed Hash Table) の利用
  ユーザの利用する PC により構成される，DHT ネットワークを基盤とする SNS．ユーザは DHT ネットワーク上にデータを保存する．実装例としては PeerSoN[13]，Safebook[14] が挙げられる．

これらの実装手法に対するソーシャル VPN の利点は次の通りである．まず一般的な SNS ユーザにとって，ソーシャルアプリケーションの利用が容易であるという点が挙げられる．サーバの連邦化および DHT の利用による手法では，ソーシャルアプリケーションはインターネットから接続可能なホストで動作している必要がある．しかしソーシャルアプリケーションが外部サービスとして動作している場合，ユーザは利用する際にその外部サービスを信頼する必要がある．ユーザが独自にソーシャルアプリケーションを設置する場合においても，インターネットからアクセス可能なホストでアプリケーションを実行する必要がある．そのようなホストをユーザが適切に管理することは難しい．それに対してソーシャル VPN では，ユーザは自身の PC 上で既存の LAN 用アプリケーションを実行することで，ソーシャルアプリケーションを利用できる．この時，友達のみがソーシャルアプリケーションへアクセス可能であり，友達以外のインターネットからのアクセスを許可しない．

またユーザが自身のデータを，自身の PC 上で管理可能な点も挙げられる．サーバの連邦化および DHT の利用による手法では，ユーザのデータは自分以外の管理者のサーバまたは PC 上で管理される．それに対してソーシャル VPN では，ユーザは自身の PC 上で自身のデータを管理し，それを NFS やブログを通して友達にアクセスさせることが可能である．

6.2 ソーシャル VPN

Figueiredo らによる Social VPN[15] では，連携する外部のサービスとして XMPP サーバを利用する．XMPP におけるコンタクトリストに相互に登録することで，各ユーザは互いに公開鍵を含む証明書を交換する．そして XMPP を介して互いに IP over P2P（IPoP）[16] によるネットワー

ク上の識別子を交換し，VPN 接続を確立する．

また海沼らによる Social Softether VPN[4] では，外部のサービスとして Facebook，Twitter および Google+を利用している．Social Softether VPN のユーザは，これらのサービスとアカウントを連携させることで，外部サービス上の友達間で VPN 接続を行うことが可能になる．また Facebook のグループ機能や Twitter のリスト機能を利用することで，VPN 接続する友達を限定することも可能である．

本研究で実装したソーシャル VPN は先行研究と比較して，特定の外部サービスに依存することなくソーシャル VPN 上のユーザを発見し，友達関係を構築できる点が異なる．また，Figueiredo らによるソーシャル VPN では，集中型 SNS の提供する外部サービスを信頼しない場合，独自に XMPP サーバを設置することでソーシャル VPN を利用することはできる．しかし一般的なユーザにとって，インターネットからアクセス可能な XMPP サーバを適切に管理するのは容易ではない．また友達との関係性に応じて定義した仮想的なドメイン名で友達を識別できる点も，先行研究とは異なる．

6.3 分散型 SNS における検索機能

Greschbach らは，DHT(Distributed Hash Table) を用いた分散型 SNS における検索機能を提案している [17]．この検索機能においては，氏名や住所等のユーザの属性とそれに対応する値のペアを DHT のキーとして表現し，ユーザの識別子をバリューとして対応づける．例えば氏名が田中，住所がつくばで検索する場合，”氏名:田中:住所:つくば”が DHT のキーとなる．このように DHT へのクエリをユーザの検索手段として利用可能にしている．

この手法に対して，本研究で実装するソーシャル VPN では，自身のソーシャルルータに収集した友達の友達のプロファイルに対してキーワード検索が可能である．そのため DHT へのクエリとは異なり，キーワードによるあいまい検索が可能である．またフレンドリストを経由してユーザを発見する機能を提供している点も，先行研究とは異なる．

7. おわりに

本稿では，自律的なソーシャル VPN の設計と実装について述べた．従来のソーシャル VPN には，ソーシャルアプリケーションにおいて，ユーザが定義した友達のグループをアクセス制御に利用できないという問題があった．また異なる外部サービスに登録したユーザ間での相手の発見および友達関係の構築ができないという問題もあった．本研究では，特定の外部サービスに依存しない自律的なソーシャル VPN を実装することでこれらの問題を解決した．

本研究のソーシャル VPN では，自身と友達のルータ間を VPN で接続する．ユーザは友達に対し，自身の視点でその関係性に基づくグループを作成し，そのグループにドメイン名を割り当てることができる．これにより，ソーシャルアプリケーションでグループを用いたアクセス制御を可能にした．またソーシャルアプリケーションとして，NFS を利用したファイル共有，IRC を利用したプライベートチャットおよび WordPress 等を用いたプライベートブログが利用可能であることを確認した．

本研究では，友達のソーシャルルータを連鎖的にアクセス可能にすることで，プロファイルの再帰的な取得およびメッセージの送受信を可能にした．更に，自身のソーシャルルータに収集された友達のフレンドリスト中のプロファイルを，キーワードで検索可能にした．これにより，ソーシャル VPN 上でユーザを発見し友達関係を構築することを可能にした．

今後の課題としては，コントロールメッセージの暗号化が挙げられる．またインターネット上で多数のソーシャルルータを動作させた時の性能を評価する．

参考文献

[1] Schwittmann, L., Wander, M., Boelmann, C. and Weis, T.: Privacy Preservation in Decentralized Online Social Networks, *IEEE Internet Computing*, pp. 16–23 (2014).
[2] Greenwald, G.: *No place to hide: Edward Snowden, the NSA, and the US surveillance state*, Macmillan (2014).
[3] Datta, A., Buchegger, S., Vu, L.-H., Strufe, T. and Rzadca, K.: Decentralized Online Social Networks, *Handbook of Social Network Technologies and Applications*, Springer, pp. 349–378 (2010).
[4] 海沼直紀，新城靖，登大遊，肖焜瑶，佐藤聡，中井央：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，情報処理学会第 26 回コンピュータシステム・シンポジウム論文集 (2014).
[5] 櫻井孝一，海沼直紀，新城靖，佐藤聡，須藤侑一，肖焜瑤，中井央：安全な家庭向けソーシャルルータの実現，情報処理学会研究報告. 2013-05-127(3)，pp. 1–7 (2013).
[6] Srisuresh, P. and Holdrnge, M.: IP Network Address Translator (NAT) Terminology and Considerations, *RFC2663* (1999).
[7] 蛸井博，新城靖，佐藤聡，中井央：分散型 SNS のための共通ランデブ・ポイント・インターフェースの提案，情報処理学会第 28 回コンピュータシステム・シンポジウム論文集 (2016).
[8] Resilio Sync: https://www.resilio.com/ Accessed: 2016-10-31.
[9] TincVPN: http://www.tinc-vpn.org/ Accessed: 2016-10-15.
[10] Conta, A.: Generic Packet Tunneling in IPv6— Specification, IETF Request for Comment (RFC) 2473 (1998).
[11] Bielenberg, A., Helm, L., Gentilucci, A., Stefanescu, D. and Zhang, H.: The growth of Diaspora-A decentralized online social network in the wild, *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pp. 13–18 (2012).
[12] Dürr, M., Maier, M. and Dorfmeister, F.: Vegas–A Secure and Privacy-Preserving Peer-to-Peer Online Social

Network, *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, IEEE, pp. 868–874 (2012).

[13] Buchegger, S., Schiöberg, D., Vu, L.-H. and Datta, A.: PeerSoN: P2P social networking: early experiences and insights, *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, ACM, pp. 46–52 (2009).

[14] Strufe, T.: Safebook: A privacy-preserving online social network leveraging on real-life trust, *IEEE Communications Magazine*, p. 95 (2009).

[15] Figueiredo, R. J., Boykin, P. O., Juste, P. S. and Wolinsky, D.: Integrating Overlay and Social Networks for Seamless P2P Networking, *IEEE 17th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises 2008 (WETICE'08)*, pp. 93–98 (2008).

[16] Ganguly, A., Agrawal, A., Boykin, P. O. and Figueiredo, R.: IP over P2P: enabling self-configuring virtual IP networks for grid computing, *arXiv preprint cs/0603087* (2006).

[17] Greschbach, B., Kreitz, G. and Buchegger, S.: User Search with Knowledge Thresholds in Decentralized Online Social Networks, *IFIP PrimeLife International Summer School on Privacy and Identity Management for Life*, pp. 188–202 (2013).