Vol.2013-OS-127 No.2
Vol.2013-EMB-31 No.2
2013/12/3

ソーシャルルータを用いたネットニュースシステムの実現

肖 焜瑤[1] 新城 靖[1] 櫻井 孝一[1] 佐藤 聡[1] 須藤 侑一[1] 中井 央[1]

概要：本研究は複数の SNS(Social Networking Service) ユーザの間で記事を交換し共有する協調 SNS アプリケーション Friend News System を実現する。Friend News System はソーシャルルータとよばれる SNS ユーザ間で安全な通信路提供する仕組みを利用して、従来のネットニュースと類似の仕組みを構築し、信頼する小人数のグループ内で記事を交換可能にする。Friend News System は、ソーシャルルータを利用するので、中央サーバを介することなく電子掲示板を実現することかできる。また、従来のネットニュースとは異なりデジタル署名を用いることで、記事の完全性を確保することかできる。さらに、タグを用いることで、ユーザが自由に記事を分類して閲覧したり、保存期間の設定などの管理をすることができる。

1. はじめに

近年ネットワークの発展とともに、多人数で協調して作業をすることが増えてきた。協調アプリケーションを利用すれば、複数のユーザがそれぞれのパソコンを操作しながら全体で一つの作業をすることが可能である。協調作業のために多くのアプリケーションが開発されているが、その多くが中央サーバに依存している。これにより、潜在的なスケーラビリティの低下、サーバの障害によるアクセス不能、サービス終了に伴う保存データの喪失およびプライバシーの問題が起こる可能性がある [1]。2013 年 6 月には Microsoft 社、Facebook 社などからアメリカ政府当局がプライバシを侵害するかたちで個人情報を収集していた事実が判明し、問題になっている。特に SNS(Social Networking Service) では小さなグループで中央のサーバを経由しないで機密性のあるデータの共有やメッセージの交換をしたいという要求がある。

本研究室ではこのような中央サーバの問題を解決するため、ソーシャルルータを提案して開発している [2]。ソーシャルルータとは、家庭用ルータを拡張したもので、利用時に Facebook などの SNS のアカウントを用いて認証を行い異なる LAN に存在するユーザの計算機同士を簡単に通信可能にする仕組みである。これにより中央サーバを利用せず、通信を行うことを可能になる。

本研究ではソーシャルルータを用いて、複数のユーザの間で記事の転送や管理が可能な協調 SNS アプリケーション Friend News System を実現する [3]。これはソーシャルルータを利用して、従来のネットニュースと類似の仕組みを構築し、信頼する小人数のグループ内で記事を交換可能にする。Friend News System は、ソーシャルルータを利用するので、中央サーバを介することなく記事を交換することができる。また、従来のネットニュース [4] とは異なりデジタル署名を用いることで、記事の完全性を確保することができる。

単純なソーシャルルータ上のアプリケーションでは、SNS が提供する通信路を使って安全な通信路を利用できるが、通信相手がオフラインの時には何の操作も行えないという問題がある。Friend News System では、オフラインのユーザが存在しても、共通の友人を介して記事を配信できるようにする。

2. ネットニュース

2.1 ネットニュースの概要

ネットニュースは、インターネットの初期から利用されている分散型掲示板である [4]。その掲示板を実現するソフトウェアの集まりを**ニュースシステム**と言う。ネットニュースの目的は 1 台から投稿された記事をその他のすべてのコンピュータにマルチキャストして読み出せるようにすることである。ネットニュースに参加しているコンピュータは、お互いそれほどの信用し合う必要がない。また、中央管理の必要もない。さらに回線の障害に対して頑強であり、UUCP(Unix to Unix CoPy) のように間欠的に接続されるネットワークでも動作する。ニュースシステムの例として、BNEWS、CNEWS や INN などのソフトウェアが存在する。

ネットニュースに基づいて生まれた最初の電子コミュニティは Usenet である。Usenet は階層化されたニュースグ

[1] 筑波大学

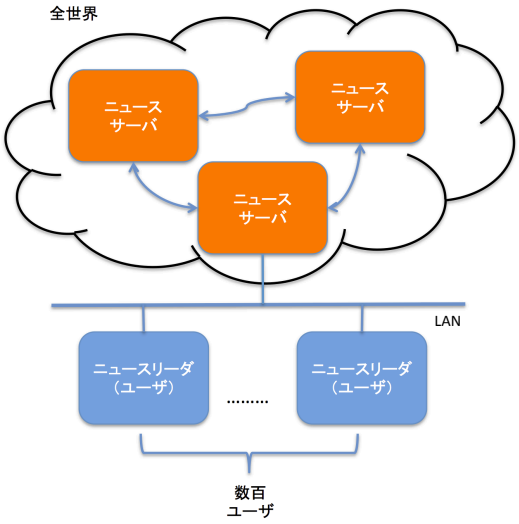

図 1 Usenet のネットワーク

ループで構成されている。Usenet はさまざまな種類の通信ネットワークを利用し、協調してニュースの記事を交換する。例えば Usenet Big8 は以下のようなニュースグループがある。

comp.* コンピュータに関連した話題を議論する。
news.* ネットニュース自身について議論する。
sci.* 科学に関連した話題を議論する。
humanities.* 人文科学について議論する。例えば、文学や哲学。
rec.* 娯楽について議論する。例えば、ゲームや趣味。
soc.* 社会や文化について議論する。
talk.* 宗教や政治について議論する。
misc.* 雑多な事柄について議論する。他の階層に馴染まないすべての話題。

日本語のものとして、fj.*, japan.* などの階層もある

図 1 は従来のネットニュースのネットワークである。従来のネットニュースではニュースサーバ間で相互に記事を転送して、記事を全世界に配る。サーバとユーザは一対多で、一つのノード（ニュースサーバ）は数百個のユーザに保存している記事を提供できる。2007 年に全世界で 40,000 人の利用者による記事のアクセスで 20 Gbit/s のトラフィックを生成していたという記録がある [5][6]。1995 年には、1 日 200 万の記事が新たに投稿され、それは 1 日 450MB に達していたという記録もある [7]。現在、筑波大学のニュースサーバは、次のようなテキストの階層の記事を交換している。

comp.*, fj.*, gnu.*, misc.*, news.*, rec.*, sci.*, soc.*, talk.*, tnn.*, okinawa.*

これらの階層では、1 日に、約 20,000 記事が交換されている。データ量は、1 日で、約 20MB になる。

2.2 ネットニュースの問題点

従来のニュースシステムは高い性能を出すことを第一に設計されたので、設定と管理が非常に難しい。一般のユーザに対してユーザインターフェイスは提供されていない。したがって、普通のユーザはニュースサーバを利用して記事を読み書きすることはできるが管理することはできない。

Usenet のニュースグループの管理は非常に厳密なので、ニュースグループを作成するには news.gourps で議論を行い、投票により賛同者を集める必要がある。ユーザは自由にニュースグループ を作成することはできないので、自由に記事を分類することができない。

Usenet は安全な通信路を利用していないので、通信が盗聴される恐れがある。そのため、小さいグループで機密性のある記事を交換することができない。

Usenet は記事の配布範囲を指定する機能があるが、現状ではほとんど実装されていない。従来のネットニュースでは、Distribution ヘッダにより記事の配布範囲を狭めることができる。しかしながら、ほとんどの記事は、Distribution ヘッダを持っていない。この場合、全世界への配布を意味する world が指定されているとみなされる。また、world 以外の Distribution も定義可能であるが、事実上有効に機能していない。その理由は、あるグループに属するすべてのサーバで正しく設定しない限り、記事の配布が止まらないからである。1 つのサーバで設定に誤りがあれば、そのサーバを通じて全世界に記事が配布されてしまう。

Usenet はコントロールメッセージと言う特別な記事がある。例えばコントロールメッセージを使って、記事をキャンセル、すなわち他のサーバ の記事を削除することができる。コントロールメッセージを乱用すれば、大きいな被害が発生する可能性がある。

3. Friend News System

Friend News System は従来のネットニュースと同じように、メッセージを他のノードへマルチキャストする。そして、データやリンクを冗長化することで、サーバやリンクの障害に頑強なものにする。Friend News System は、SNS における時系列のメッセージ配信と似ている。たとえば、Facebook の Feed、Twitter の Timeline に類似している。これらの SNS における記事の配信と異なり、記事は中央サーバではなくユーザのローカルコンピュータに保存される。Friend News System は 2.2 節で述べた従来のネットニュースの問題点を解決し、簡単で軽量なニュースシステムを提供することを目指す。

従来のネットニュースでは、記事を全世界に対して配信する。これに対して、Friend News System は記事をユーザの SNS のフレンドリストを元に作成した少人数グループのみ配信する。Friend News System のネットワークのイメージは図 2 のようになる。ユーザが一つのノード（ニュース

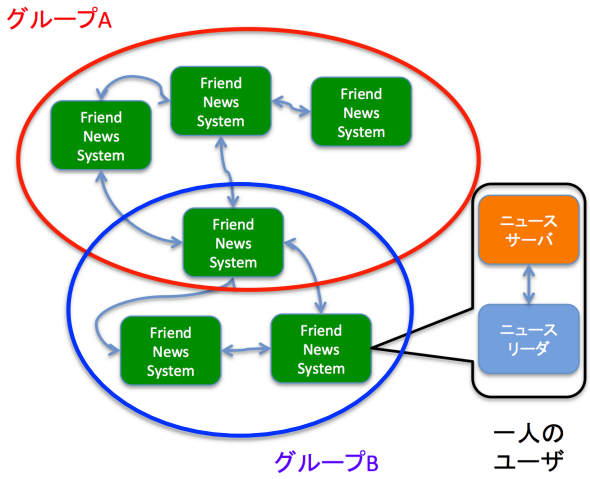

図 2 Friend News System のネットワーク

```
Date: Tue, 29 Oct 2013 15:00:21 +0900
From: alice <alice@comp.alice.socialrouter>
Message-ID: <fb9ab529-faf0-4cd4-8260-13a7b289a804@alice>
Subject: Let's go fishing
Tag: fishing,ibaraki,lake
Path: mobile.bob.socialrouter!comp.alice.socialrouter
Signature: From,Subject,Tags,Message-ID
Distribution: fishing
Msg-Sign: OyY4Sv8flMcuwKeiM2IDNvcfsib0WzO3Z3xP9wjS3jl6IQ1
    MnGlYKTjhDBIiXztnYiOIQrvrpakogpOu9WYGmOXRuFr39KO+GF
    Pe7covb3uM+fu+UQkww8CTcGKgvR29Rm7otYLKeOgI1M3+1KS
    rLOgf8fgcwUgXodI56ZLuQlun3Y+vT7EutHCgAM6pEgr90Hp8hCF
    hEZt1GMp1fm79Loqc6gpEdpLetOvSZZsMPq4D5CFMoaVAb/xmr
    edSvDA8HQ2SuZHhYi/+cFUsn1HvBjp4/5UVtalNt9c+DVWqrqP/p
    5Pel4N4ZIBLCuC1YISJ5s/HYzOp7t328EL79a43sw==
```

図 3 Friend News System 記事の例

サーバ）に対応している。記事は自分のフレンドリストにいる人としか交換しない。

Friend News System のユーザは自分のコンピュータ上でニュースシステムを動作させ管理できる。管理や設定のユーザインターフェイスとしては Web ブラウザを利用可能にする。Friend News System は現在の NNTP を利用したニュースリーダを使って投稿や閲覧することを可能にする。これにより、ユーザは NNTP に対応したニュースリーダを使うことによって、簡単な設定でメッセージの投稿や閲覧を可能にする。例えば、Thunderbird では、メニューからネットニュースを選び、NNTP サーバのホスト名として localhost を指定すればすぐに Friend News System を利用することができる。

Friend News System はニュースグループの代わりにタグを使って、記事を管理する。ユーザは自由にタグ作成や削除することができる。これにより各ユーザは自分の視点で記事の分類や管理することができるようになる。

Friend New System はソーシャルルータが提供する安全な通信路を利用しているので、機密ある記事の交換が可能である。更にデジタル署名を用いて、たとえ記事を中継する途中に時悪意のあるユーザがいたとしても、そのユーザによる記事の改変を防ぐことができる。

FriendNewSystem ではメンバーリストと言う仕組みを用いて、従来のネットニュースでは実装されていない配布範囲の指定機能を実現する。メンバーリストとは特定の話題に関心を持つグループで情報交換をする仕組みである。電子メールのメーリングリストと類似な仕組みである。

3.1 記事の構造

Friend News System の記事の形式は RFC2822[8] に従って定義する。記事の構造は電子メールや従来のネットニュースの記事と同様にヘッダとボディの 2 つの部分から構成されている。記事の例を図 3 に示す。その特徴はニュースグループではなく、タグを保持するフィールド Tags があること、およびデジタル署名用のフィールドがあることである。さらに、従来のネットニュースでは有効に利用されていない Ditribution ヘッダを利用して、配布範囲が指定できる。メンバーリストについて、5.2 節で述べる。

From ヘッダは送信者の身元を示すヘッダである。 従来のネットニュースの From ヘッダには記事の著者の電子アドレスが置かれる。Friend News System はソーシャルルータを利用しているので、ソーシャルルータ得られるドメイン名と著者の名前を組み合わせたものを入れている。

Message-ID は従来のネットニュースと同じようにニュース記事を区別する識別子である。従来のネットニュースは一般的に日付や時刻と送信者の電子アドレスのに基づいて ID を生成していた。Friend News System は 128 ビットの UUID(Universally Unique Identifier) を生成して From ヘッダの内容と合わせってユニークな ID を生成する。

Msg-Sign ヘッダはデジタル署名を保持するヘッダである。Signature ヘッダに記述されているヘッダフィールド値とボディに SHA1(Secure Hash Algorithm) 関数で 160 ビットのハッシュ値を生成する。次に、そのハッシュ値に対して、RSA 秘密鍵で署名し、Base64 でエンコードし、Msg-Sign ヘッダのフィールドに保存する。

3.2 タグ

従来のネットニュースは Newsgroups ヘッダを利用し、記事の話題を示す。Newgroups は自由に作成することができない。

Friend News System で記事の著者は、記事を投稿する時に自分の観点でその記事にタグを設定する。従来のネットニュースのニュースグループとは異なり、記事の著者はユーザは自由にタグを付加することができる。たとえば、図 4 のよにユーザ A は comp と sci 2 つのタグを利用している。この仕組みは、Twitter で用いられているハッシュタグの仕組みと似ている。

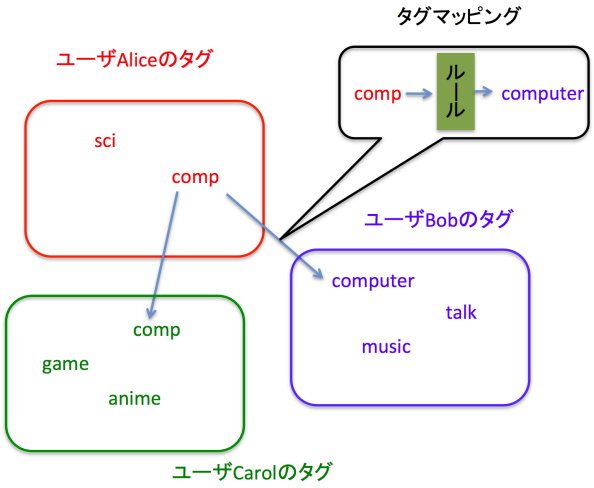

図 4 タグの構成

一方、記事を受信したユーザは、記事に含まれているタグをそのままニュースグループとして利用することもできる。それに加えて、記事を受信したユーザは、自分の好みのタグにマッピングすることができる。ユーザは事前に正規表現のルールを従って、マッピングルールを作成する。たとえば、図 4 のようにユーザ B がユーザから comp タグを付ける記事を自動的に computer タグへ変換する。

Friend News System では、記事を受信したユーザは、タグマッピングの他に、記事のヘッダフィールドを用いてタグを付加することもできる。たとえば、Alice が作成した記事には、from.alice というタグを付加することができる。複数の釣り仲間の友達から受け取った記事に自動的に fishing というタグを付加することができる。

Friend News System は分類していない記事を junk と言う特別なタグに所属させる。マッピングができない記事は junk に所属させる。

3.3 peering

従来のニュースシステムでは、2 つのサーバの間で記事を交換するように設定する。このことを、peering と呼んである。INN では、送信側は、newsfeed ファイル、受信側は、incoming.conf を設定する。newsfeed ファイルには、送信する階層を書く。

Friend News System ではユーザのメールアドレス、ユーザの公開鍵、ソーシャルルータから得られるニュースサーバのドメイン名、そしてユーザ送信側のタグを用いて peering を行う。設定は以下のようになる。

- 送信側は受信側に自分の公開鍵を渡す。
- 送信側と受信側は各自ソーシャルルールに Friend News System のサーバを動かすコンピュータを登録する。
- 送信側は fnsfeed ファイルに転送するタグを登録する。

3.4 コントロールメッセージ

コントロールメッセージとは、制御情報を含んだ記事である。Friend News System のサーバがコントロールメッセージを受けとると、単に記事をファイルとして保存だけではなく特定の行動をとる。コントロールメッセージによって要求される行動の実行には、ローカルな管理上の制限が課せられる。それに基づいて要求を拒否したり、あるいは実行したりすることがある。本研究では、従来のネットニュースのコントロールメッセージの仕組みを改良する。その特徴としてはデジタル署名、および、ソーシャルルータから得られた通信相手の情報を参照して、記事の送信者の身元を確認する。

Friend News System に以下のようなコントロールメッセージがある。

cancel 記事を削除するコントロールメッセージである。

newtag タグを作るコントロールメッセージである。他のサーバへ転送しない。

rmtag タグを削除するコントロールメッセージである。他のサーバへ転送しない。

newml メンバーリストを作成するコントロールメッセージである。

updateml メンバーリストを更新したり削除したりするコントロールメッセージである。

ニュースサーバは記事の Control ヘッダフィールドの内容からコントロールメッセージの制御情報を識別して、要求される行動を実行する。

ニュースサーバはコントロールメッセージ cancel を受け取ると、元の記事の著者が送信した場合のみ実行する。コントロールメッセージに付けられた署名を著者の公開鍵で検証する。検証に成功した場合、記事を削除する。検証に失敗した場合、キャンセルは実行されない。

Friend News System ではタグの管理を管理モジュールで操作できるが、ニュースリーダでタグを管理したい場合がある。その場合は newtag と rmtag コントロールメッセージを用いて操作する。

newml と updateml はメンバーリストの管理に用いる。メンバーリストについては 5.3 節で述べる。

4. Friend News System の構成

Friend News System はニュースサーバ、フィードサーバ、システム管理モジュールとソーシャルルータから構成される。Friend News System はソーシャルルータを利用して相互通信を行う。システムの全体的の構造を図 5 に示す。

4.1 ニュースサーバ

ニュースサーバはシステムの中心として、ローカルホストからの投稿や他のサーバから受信した記事をローカル

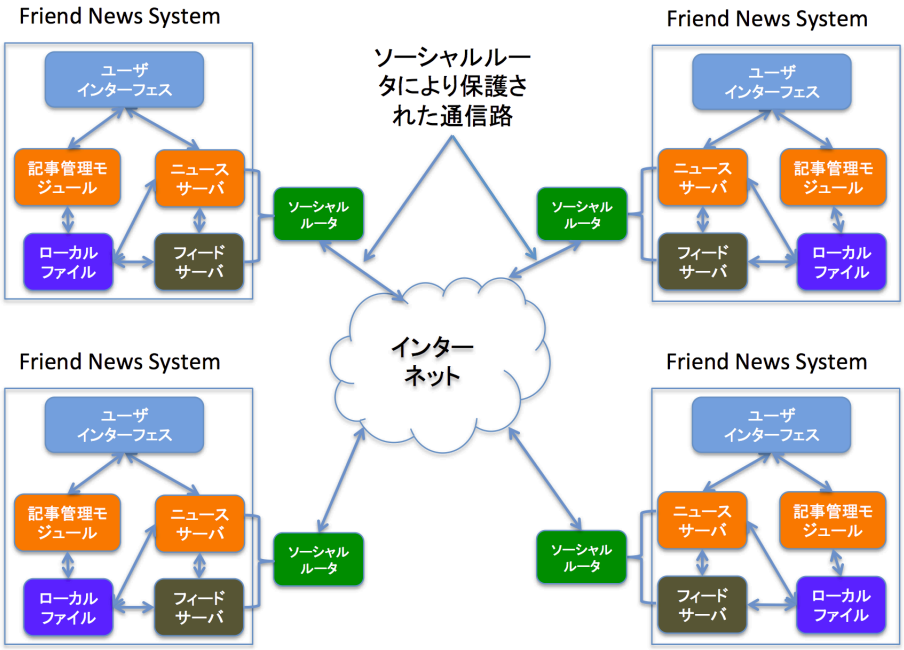

図 5 Friend News System の構成

ファイルに保存し、フィードサーバに送る。そして、保存された記事のうち、期限が切れている記事を削除したりアーカイブしたりする。

4.2 フィードサーバ

フィードサーバは記事を他のニュースサーバへ転送する役割を果たす。同じコンピュータで運行しているニュースサーバから渡された記事をルールに従って、ほかのニュースサーバに転送する。転送するルールについては 5.1 節で述べる。

4.3 システム管理モジュール

Friend News System のシステム管理モジュールはエキスパイア機能、アーカイブ機能、履歴管理機能、および署名管理機能を持つ。ユーザはシステム管理モジュールの Web インターフェイスを経由して、ローカル記事を管理したりシステムを設定することができる。

従来のニュースシステムは Expire ヘッダにある期限をヒントにして、システム管理者が定めたポリシーで記事を削除する。Friend News System でも同様に期限が切れている記事に対して、自分の必要に応じて削除やアーカイブを行う。

Friend News System はデジタル署名を用いて、記事の完全性を確保する。Friend News System のユーザは事前に公開鍵を交換する必要がある。ユーザは、PGP (Pretty Good Privacy) の方法と同様に、直接面談したり、通信相手の信頼性を確保できる通信路を利用して、公開鍵を交換する。

サーバは受信した記事の署名が正しければ、履歴にメッセージ ID を書いて記事をファイルに保存する。 正しくなければ、一時的なファイルとして保存して、ユーザによる例外処理を行わせる。

システム管理モジュールインターフェイスとして、Ruby on Rails で構築される Web インターフェイスを提供する。

4.4 ソーシャルルータ

ソーシャルルータでは SNS メンバの計算機が、自分の計算機と接続しているルータの向こう側に存在するように見える。ソーシャルルータは DNS サーバを含み、各 SNS メンバの計算機には特別のドメイン名と IP アドレスを割り当てる。IP アドレスから SNS のメンバのユーザ名を知ることができる。加えてソーシャルルータには強力なアクセス制御機能が存在する。TCP/IP のポート単位で通信を制御できるため、SNS メンバは許されたサービスにのみ接続することができる。HTTP のみを許可したり、NNTP のみを許可したりできる。

ソーシャルルータは本研究室で開発を行なっている [2]。ソーシャルルータのアプリケーションとしては本研究で述べるもの以外に SQL データベースに関するものがある [9]。このアプリケーションは各ユーザが自身の計算機に持つデータベースをソーシャルルータを利用した通信を用いて SNS を利用する別のユーザにアクセスさせることを可能にする。

5. 記事の配布

Friend News System では、ユーザはローカルサーバに

記事を投稿する。サーバは、投稿された記事の形式を解析する。解析の結果、形式に問題がなければ、サーバが記事に対してデジタル署名を付ける。その後、隣接の他のサーバに記事を配布する。

5.1 記事転送ルール

ニュースサーバは受信した記事をすべて転送するのではなく、ルールに従って転送する。例えば、隣のサーバのユーザ B がコンピュータに関わる記事には興味がある場合、ニュースサーバ comp タグの記事をユーザ B に転送する。この場合、設定ファイルのユーザ B の設定として comp を記述する。

5.2 メンバーリストを用いた配布範囲の決定

Friend News System は配布範囲を実装して、転送者の指定を可能にする。このために、メンバーリストと言うリストを保持して、配布範囲を指定する。

メンバーリストは、メール・リーダ UCB Mail[12] が持つ alias 機能と似ている。UCB Mail では、ユーザは、次のような alias を設定ファイル ~/.mailrc に記述することができる。

alias fishing alice@example.com bob@example.org

ユーザは、この疑似的なアドレス "fishing" をメールの To: に指定してメールを送信できる。UCB Mail は、To に指定された疑似的なアドレスを本来のアドレスに展開してから MTA (Mail Transfer Agent) に渡してメールを送信する。

Friend News System のメンバーリストは UCB Mail の alias と類似の仕組みを利用する。メンバーリストの例を図 6 に示す。メンバーリスト fishing には 3 名のメンバーが存在する。alice はメンバーリストの管理者なので、コントロールメッセージを使ってメンバーリストを管理することができる。bob と carol はメンバーリストを管理することができない。メンバーリストの情報は alice、bob、carol の各ユーザが保持する。投稿時には、Distribution ヘッダにメンバーリスト fishing を参照して、ニュースサーバがメンバーリストのメンバー検索を行い、転送する人のソーシャルルータのドメイン名を取得して転送する。記事を受信したサーバも同様にメンバーリストを参照して記事を転送する。

5.3 メンバーリストの管理

メンバーリストはコントロールメッセージを通じて配布する。メンバーリストの管理者は自分のニュースサーバに newml と言うコントロールメッセージを投稿する。すると、ニュースサーバがこのコントロールメッセージを転送する。投稿する時点の時刻がメンバーリストのタイムスタンプとなる。メンバーリストが更新される場合、更新権限

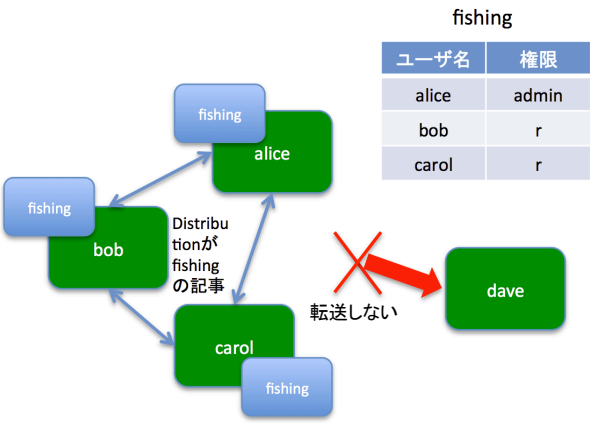

図 6 メンバーリストの例

を持つメンバーが updateml というコントロールメッセージを投稿する。このコントロールメッセージを投稿する時点の時刻がタイムスタンプとなる。コントロールメッセージを受信するニュースサーバは受け取ったコントロールメッセージのメンバーリストのタイムスタンプとローカルメンバーリストのタイムスタンプを比較して、タイムスタンプが新しい場合メンバーリストを更新する。

5.4 ニュースサーバ間の通信

Friend News System ではサーバ間の通信プロトコルを RFC977 の NNTP(Network News Transfer Protocol)[10]、RFC2980 の Common NNTP Extensions[11] に基づいて設計する。Friend News System で用いるプロトコルはクライアントサーバモデルに基づいている。Friend News System のサーバがサーバとクライアント両方の役割を果たすことになる。記事を転送する時には、送信するノードがクライアント、受信するノードがサーバとして動作する。

Friend News System の通信はソーシャルルータが提供する通信路を利用してなされる。転送元のニュースサーバはソーシャルルータから得られた転送先のニュースサーバの IP アドレスに接続して通信を行う。本研究で実装するメッセージを転送する時のコマンドを応答コード例は以下のようになる。

(1) 送信側のフィードサーバ（以後は送信側）が" IHAVE" コマンドと記事の ID を受信側のニュースサーバ（以後は受信側）に送信して、受信側に自分が転送したい記事があることを伝える
(2) 受信側が受けた記事の ID が履歴に含まれていないのことを調べ、記事が既に受信済みかどうかを確認する。記事が既に存在すれば、コード 437 を返信してこの記事を受信しないと伝える。記事が存在しなければ、コード 335 を返信して記事を受信すると伝える。
(3) 送信側は返信コードにしたがって動作する。コード 335 の場合、送信側が記事を送信する。コード 437 の

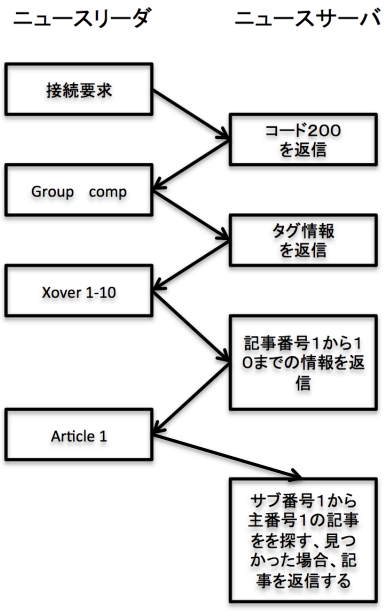

図 7 ニュースリーダとニュースサーバの間で送受信される要求メッセージと応答メッセージの例

場合、次の記事の転送に進む。

(4) 受信側が記事を受信する。記事を全部受信した後、記事の署名を検証する。検証が成功すれば、 受信側がコード 235 を返信して、送信が成功したことを伝える。その後、この記事を他のサーバへ転送する。

(5) 送信側は返信コードにしたがって動作する。コード 235 の場合、送信完了し、次の記事の転送に進む。

(6) 受信側がオフラインの時や途中で通信が切れ、コード 235 が得られないことがある。その場合、送信側は、その記事の ID をサーバごとのファイルに保存し、ある時間をおいてから再度記事の転送を試みる。

6. ローカルの NNTP クライアントへの対応

ニュースリーダとニュースサーバ間の通信は TCP/IP を用いる。ポート番号としては NNTP の 119 を利用する。Friend News System の NNTP サーバは既存の NNTP のコマンドのうち、ニュースリーダモード (mode reader) で使えるコマンドを実装する。これにより、Thunderbird 等の既存のニュースリーダを変更することなくそのまま利用可能にする。

6.1 タグとニュースグループの対応

Friend News System は、3.1 節で述べたようなヘッダを持つ記事を交換する。それには、Tags ヘッダは含まれているが、Newsgroups ヘッダは含まれていない。

Friend News System のサーバは、NNTP でアクセスしてきたクライアントに対しては仮想的にニュースグループが存在しているかのごとく見せかける。このニュースグループは、記事の Tags ヘッダに含まれているタグに加えて、3.2 節で述べたマッピングルールも適応されたものも利用可能である。

具体的には、次のことを行う。

- NNTP クライアントから list コマンドを受け取ると、サーバは、ニュースグループとして利用しているタグの一覧を返す。
- NNTP クライアントから group コマンドでニュースグループ名を受け取ると、サーバは、対応したタグが指定されたものとして扱う。
- NNTP クライアントから article コマンドを受け取ると、サーバは、保存している記事をクライアントに渡す前に、Newsgroups ヘッダ、Xref ヘッダ、Followup-To ヘッダ等を生成して付加する。
- NNTP クライアントから post コマンドを受け取ると、サーバは、Newsgroups ヘッダにニュースグループ名をタグに変換して Tags ヘッダへ埋め込む。

6.2 記事番号

従来のネットニュースでは各記事には所属ニュースグループごとに記事番号がつける。複数のニュースグループに所属する場合各ニュースグループごとに記事番号を与える。記事番号はニュースリーダにより既読未読の管理に使われる。

本研究ではすべての記事を all と言うタグに所属しているとみなす。記事を受信する順番に従って、主となる記事番号を割り付ける。受信や投稿した記事のタグがシステムのタグに含まれていない場合、junk と言う特殊なタグに所属させる。ユーザがタグを作成する場合、タグごとにサブ記事番号を生成する。そして、ニュースシステムはサブ記事番号と主記事番号との対応関係を管理する。

6.3 サーバと NNTP クライアントの対話の例

ニュースリーダとニュースサーバの間で送受信される要求メッセージと応答メッセージの例を図 7 に示す。

(1) ニュースリーダがローカルホストのポート番号 119 に接続する。ニュースサーバが接続先がローカルホストの時のみ code 200 を返信する。

(2) ニュースリーダが group コマンドを用いて、タグを選択する。ニュースサーバがタグの最初の記事番号、最後の記事番号、記事の数などの情報を返信する。

(3) ニュースリーダが xover コマンドを用いて、選択したタグ内の一定範囲の記事の概要を要求する。ニュースサーバはその範囲内存在する記事のタイトル、ID、送信などの情報を返信する。

(4) ユーザが記事の本文を見る場合、ニュースリーダが article コマンドを用いて、記事を要求する。ニュースサーバはニュースリーダから受け取れた記事番号をもとに記事を探して返信する。

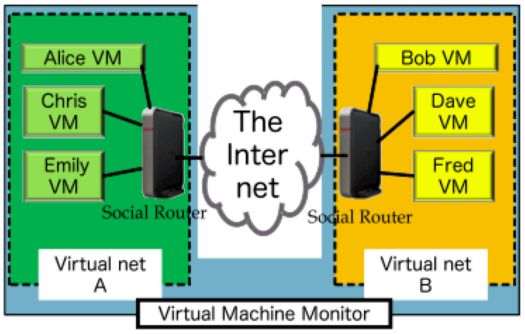

図 8 ソーシャルルータを用いた実験ネットワーク

7. 実装

システムを構築する環境を以下に示す。

- 開発 OS: Linux Ubuntu12.04LTS。
- 開発言語: Ruby1.9.3。

ニュースサーバとフィードサーバは Ruby で作成した。システム管理モジュールは Ruby on Rails で構築した。

ソーシャルルータを用いた実験環境を図 8 に示す。 実験環境はひとつの実機計算機上に仮想計算機モニタを導入し、その上に仮想ネットワークと仮想計算機を配置することによって実現している。まず仮想計算機モニタ上に 2 つの独立した LAN を構築する。それぞれの LAN にひとつずつソーシャルルータを配置し、The Internet のゲートウェイとする。それぞれの LAN に実験用の仮想計算機を配置する。これらの仮想計算機はそのままでは異なるネットワークの仮想計算機と通信することはできない。これらの仮想計算機がブラウザ上で Facebook の認証を通過した時、初めて異なる LAN に属していてもお互いの SNS アカウントが友人関係ならば仮想計算機同士で通信することが可能になる。ソーシャルルータ及び各仮想計算機には外部から制御するための通信路が存在する。これらの通信路を使用して異なる LAN 同士の仮想計算機が通信することはない。

8. 関連研究

8.1 UsenetDHT

UsenetDHT と言うネットニュースを DHT(Distributed Hash Table) で実装する研究がある [5][6]。この研究は DHT を用いて、メッセージを幾つかのノードに保存する。従来のネットニュースでは、あるメッセージを全てのノードにコピーして保存していたのに対して、UsenetDHT では、DHT の仕組みによりあるメッセージを少数のノードにだけ保存する。これにより、ネットワークのトラフィックを減し、ストレージを節約する。

本研究ではソーシャルルータを利用しているので、DHT より実装が簡単になっている。たとえば本システムではノードの発見、ノードの参加、ノードの離脱等の処理をしなくてもよい。SNS のコンタクトリストに含まれた友人間でメッセージを交換するので、悪意のあるユーザへの対応は DHT を用いる方法よりも容易である。

8.2 ネットニュースを使った電子メール型メッセー ジングサービス

ネットットニュースを使って、メッセージを転送するメールシステムの研究がある [13]。この研究ではメールサーバが受信したメッセージを暗号化して対応するニュースグループに投稿して、NNTP を利用して、他のサーバに配送する。ユーザはアクセス可能なサーバにアクセスしてメールを読む。この研究は災害の時メールサーバが利用できない場合一時的に利用することを目的としている。システムは既存のニュースシステムとメールシステムをそのまま利用している。

本研究では日常的なメッセージ交換を目的として Friend News System を提案している。本システムは既存のニュースシステムを再利用したものではなく、独自のニュースシステムを開発している。従来のニュースシステムと同様に障害に対して頑強であり、SNS の友人の間でメッセージを交換することも可能である。さらに、Ruby でニュースシステムを開発することで、軽量で管理しやすいものを目指している。

9. まとめ

本研究ではソーシャルルータにより提供される通信路を用いたニュースシステム Friend News System について述べ た。その特徴は、ソーシャルルータを利用して中央サーバを利用せずに記事を交換する点にある。記事を交換するプロトコルは NNTP を参考にして設計している。現在、ニュースサーバとフィードサーバの基本的な部分は完成し、システム管理モジュールの発信者を用いたタグ付けやメンバリストの更新を実装している。

今後の課題はユーザの公開鍵を効率的に入手する方法を実現することである。サーバは受信した記事の著者の公開鍵を保持していない場合がある。その場合、従来のネットニュースで定義されている sendme コントロールメッセージと類似のコントロールメッセージを実装し利用して、隣のノードから鍵を要求できるようにしたいと考えている。

参考文献

[1] Datta, A., Buchegger, S., Vu, L.H., Strufe, T.,Rzadca, K.: ”Decentralized Online Social Networks”, Handbook of Social Network Technologies and Applications, pp 349-378, 2010.
[2] 櫻井孝一, 新城靖, 佐藤聡, 須藤侑一, 肖焜瑶, 中井央: ”安全な家庭向けソーシャルルータの実現”, 情報処理学会システムソフトウェアとオペレーティングシステム研究会, 2013-OS-127, 2013.
[3] 肖焜瑶, 新城靖, 佐藤聡, 中井央, 板野肯三: ”分散型 Web

ブラウザにおけるネットニュースシステムの実現”, 情報処理学会第 24 回コンピュータシステム・シンポジウム, ポスターセッション, 2012.
[4] Spencer, H. Lawrence, D. :”Managing Usenet”, Ferguson, P. Clairemarie Fisher O’Reilly, 1998.
[5] Emil Sit, Frank Dabek, James Robertson: ”Usenet-DHT: A Low Overhead Usenet Server”, Lecture Notes in Computer Science Volume 3279, pp 206- 216, 2005.
[6] Emil Sit, Robert Morris, Frans, M. Kaashoek: ”Usenet-DHT: A Low-Overhead Design for Usenet”, USENIX Networked System Design and Implementation, 2008.
[7] http://web.archive.org/web/20080208153343/ http://www.tlsoft.com/arbitron/
[8] Resnick, P. : ”Internet Message Format”, RFC2822, 2001.
[9] 須藤侑一, 新城靖, 櫻井孝一, 佐藤聡, 肖焜瑶, 中井央: ”ソーシャルネットワークを利用した SQL データベースの相互利用”, 情報処理学会システムソフトウェアとオペレーティングシステム研究会, 2013-OS-127, 2013.
[10] Kantor, B. Lapsley, P. : ”Network News Transfer Protocol”, RFC977, 1986.
[11] Barber, S. Academ Consulting Services: ”Common NNTP Extensionsi”, RFC2980, 2000.
[12] UNIX 4.2 BSD Reference Mannual, 1982.
[13] 石橋由子, 桝田秀夫: ”ネットニュースシステムを使った電子メール型メッセージングサービスの提案”, 情報処理学会インターネットと運用技術研究会, 2013-IOT-20, 2013.