送信者 ID の動的把握に基づくメッセージ受信制御方式

阪 本 裕 介[†1]　中 山 雅 哉[†2]

各コミュニケーションツールはスパムの問題に悩まされている．スパムの判別の際に正常な通信をスパムと誤判別することの被害は大きい．その誤判別を防ぐためにはホワイトリストによって保護することが有用である．本研究では，メッセージの到着時に相手の ID を確認することで，送信者ごとにメッセージを保護するかどうかを決定する受信制御方式を提案する．提案手法を実装し受信制御に要する時間計測の結果，提案手法は各コミュニケーションツールで実用的に利用できることがわかった．

A Receiver-side Message Control Mechanism Based on Reactive Retrieval of Sender-ID Information

Yusuke Sakamoto [†1] and Masaya Nakayama[†1]

Each communication tools has a problem of spam messages. Spam detecting methods sometimes classify valid messages as spam messages, and the damage by this is large. It is effective to protect valid messages from the false classification by whitelist methods. We propose a mechanism to control messages by retrieving sender-ID information when receiving messages. As a result of the implementation and the evaluation of our proposed method, we concluded that it would be practical.

1. はじめに

インターネット上において電子メールやインスタントメッセンジャー（以下，IM），IP 電話などの様々なコミュニケーションツールが利用されている．これらは広く利用される一方で，スパムの問題を抱えている．スパムを識別して排除する研究は多数存在するが，その多くは意図した相手からの通信であってもそれをスパムと判別することがある．このような誤判別は，実際のコミュニケーションに支障をきたすだけでなく，自分が送信したメッセージがスパムとして排除されているのでないかと心配するユーザが存在するという間接的な問題の原因ともなっている．

意図した相手からのメッセージを誤判別から保護するための手法としてホワイトリスト方式がある．中でも，メッセージの送信に使われた電子メールアドレスや IM のアカウント名，電話番号といった ID（以下，コンタクト ID）を元に正当な送信者かどうかを判別するホワイトリストは広く用いられている．しかしながら，コミュニケーションツールやコンタクト ID の相手に応じた使い分けやコンタクト ID の変更といった利用形態を考慮すると，現在のホワイトリストが本来の目的を達成できているとは言えない．事前に交換したものと異なるコンタクト ID からメッセージが送信される場合は，正当なユーザからのメッセージであってもそれをホワイトリストでは保護することができない．

我々は，意図した相手からのメッセージを正しく保護するために，相手のコンタクト ID をメッセージの到着時に動的に把握する受信制御方式の提案を行い，理論的にその有効性を示してきた[12)]．本論文では，提案方式を実装し，その実装下での情報取得にかかる時間の計測および見積りを行うことで，具体的な効果について検証を行った．その結果，情報取得に必要な時間については，今後改良を行うことで実用面において問題が生じない程度まで軽減することが可能であると予測することができた．これにより，提案手法を各コミュニケーションツールで利用することが可能であり，実用性があると結論付けられた．

本論文の構成は以下の通りである．2 章では，コミュニケーションツールの利用形態とスパムの現状について触れ，正当なメッセージを保護するという本研究の目的を述べる．3 章では，正当なメッセージを保護するという目的をもつ関連研究について触れる．4 章では，Web 上でコンタクト ID 情報を管理し，それを正当なメッセージの判別に役立てるという本研究の提案手法について述べる．5 章と 6 章では，提案手法の実装と，その実装下で行った各種の時間評価について述べる．最後に 7 章で本論文の結論と今後の課題について述べる．

†1 東京大学大学院新領域創成科学研究科基盤情報学専攻
Department of Frontier Informatics, Graduate School of Frontier Sciences, The University of Tokyo
†2 東京大学情報基盤センター
Information Technology Center, The University of Tokyo

2. 背景と目的

現在インターネット上では電子メール，IM，IP 電話といった様々なコミュニケーションツールが利用されている．これらは広く利用されているが，受信者の意図を無視して一方的にメッセージを送りつけるスパムが問題となっている[2]．電子メールにおいてはスパムメールが全電子メールの 40%を占めるなど大きな問題となっている[5]．IM においては，スパムメール同様，広告メッセージなどを送りつける SPIM が確認されている．IP 電話においても，相手に一方的に電話をかけて録音された広告メッセージなどを送る SPIT(Spam over Internet Telephony) が確認されている．現在，SPIM や SPIT はそれほど大きな問題とはなっていないが，IM や IP 電話の利用者の増加に伴って今後の増加が予想されている[8]．

スパムを識別し排除しようとする研究は多く存在するが，それらには，正当な相手からのメッセージをスパムと判別してしまう誤判別の問題がある．メッセージ本文の解析によりスパムを判別する手法では，電子メールにおいてはそのような誤判別が発生する確率は低いが，短文がやりとりされる IM においては確率が数%程度まで上昇することが報告されている[9]．また，電話をとる前に判別する必要がある SPIT ではメッセージの内容を判別に用いることができないため，誤判別の確率が上昇することが予想される．正当なメッセージがスパムとして排除されてしまうとその被害は大きい[7]．また，実際に排除されていなくても，自身の送ったメッセージがスパムとして排除されているのではないかと心配するユーザが存在するなど，誤判別問題による間接的な被害も無視できない[6]．

正当なメッセージを誤判別から保護する手法としてホワイトリスト方式がある．正当なメッセージを送ってくると考えられる相手の情報をホワイトリストに記載し，その情報に合致するメッセージが来た場合はそれを正当なものとして扱う手法である．ホワイトリストに記載する情報としては相手のコンタクト ID や IP アドレスなどいろいろなものが考えられるが，各コミュニケーションツールにおいてユーザの識別にコンタクト ID が利用されていることを考慮すると，コンタクト ID を元に判別を行うホワイトリストが汎用性の高いものだと言える．

しかしながら，現在のコミュニケーションツールの利用形態の下では，従来のホワイトリスト方式ではその効果を十分に発揮できていない．ユーザはコミュニケーションツールや各ツールにおけるコンタクト ID を相手や目的に応じて使い分けている．仕事の連絡には仕事用の電子メールアドレスを用い，個人の趣味のコミュニケーションには個人的に所有している電子メールアドレスを用いる，といった例が挙げられる．また，親しい相手とは電子メールアドレスも IM のコンタクト ID もお互いに交換してコミュニケーションを行うが，そうでない相手とは電子メールアドレスだけを交換し IM によるコミュニケーションは行わないといった例も挙げられる．こういった使い分けは利用者の意図に応じたコミュニケーション手段の制御を可能にして，利便性を向上させており，一般的な利用形態だと言える．

このとき 1 人のユーザが複数のコンタクト ID を所有しており，事前に相手に教えたものとは異なるコンタクト ID を用いてその相手にメッセージを送信することが起こり得る．事前に教えられたコンタクト ID を受信者がホワイトリストに記載していたとしても，このメッセージをホワイトリストによって保護することは不可能である．その結果，正当なユーザからのメッセージを誤判別から保護するという目的でホワイトリストを利用しているにも関わらず，コンタクト ID が異なるせいでその目的を達成できなくなっている．

この問題は，送信者のコンタクト ID が変更になった場合にも生じる．ユーザのコンタクト ID は，所属組織の変更などといったユーザの環境の変化によって変更されることがある．コンタクト ID が変更になったときそのことを相手のユーザに適切に通知しておかなければ，新しいコンタクト ID を用いてコミュニケーションを図った際に受信者はそのメッセージを正しくホワイトリストによって保護することができない．そのため，この場合も正当なユーザからのメッセージを保護するというホワイトリストの目的を達成できない．

正当なユーザからの通信をコンタクト ID の差異に関わらず誤判別から保護できるようにすることは，メッセージの誤判別による直接的な被害を抑えるとともに，各コミュニケーションツール利用時のユーザの不安を解消する意味でも重要な課題である．そのために，本研究ではコンタクト ID の使い分けや変更といった背景に対応した，メッセージの受信制御方法の実現を目的とする．

3. 関 連 研 究

コンタクト ID の差異によらずに正当なユーザを判別し，そこからのメッセージを正当なものと判別するための関連研究として，各コンタクト ID の信頼度を割り当てる手法[1)3)]，SNS（Social Networking Service）からユーザ同士の近さを取得して判別に利用する手法[11)] などがある．

コンタクト ID に信頼度を割り当てる手法としては，電子メール向けに Chirita らが MailRank[3)] を，IP 電話向けに Balasubramaniyan らが CallRank[1)] をそれぞれ提案している．

MailRank においては，各ユーザのメールクライアントアプリケーション中のアドレス帳の内容や送信先履歴などを参考にして各電子メールアドレスに信頼度が割り振られる．このとき，スパム送信者は自分からメールを送信する頻度は高いけれども，他のユーザからメールを受け取ることは少なく，メールの送信先と

して指定されることが少ないため，彼らのメールアドレスには低い信頼度しか割り当てられない．ある閾値を設け，それ以上の信頼度をもつ電子メールアドレスを用いたメッセージを正当なものとして受け取るようにすることで，事前に交換していない電子メールアドレスを用いたメールであっても信頼度が高ければ正当なものとして処理するようにできる．しかしながら，1 人のユーザが所有するメールアドレスすべてに高い信頼度が割り当てられていなければ，低い信頼度しか割り当てられていない一部のメールアドレスを用いたメールが正当なものとして判別されず，コンタクト ID の差異に依らずに正当なものと判別するという目的は達成できない．特に，コンタクト ID が変更になった直後にはそのコンタクト ID に高い信頼度を割り当てることは難しいため，変更後のコンタクト ID から送信されたメッセージを正当なものと判別することが達成できていない．

CallRank においては，過去の通話時間に基づいて各電話番号に信頼度が割り当てられる．こちらも MailRank と同様，1 人のユーザが所有するすべての電話番号に高い信頼度が割り当てられていなければ，一部の電話番号からのメッセージは正当なものと判別されない可能性がある．

横山らは SNS からユーザ同士の趣味的な近さを取得する API を開発し，それらがコミュニケーションツールにおけるメッセージの正当性判別に利用できると提案している[11]．各ユーザが SNS 上に自身のコンタクト ID 群を記載し，提案された API を用いてその情報を取得できるようにする．コミュニケーションツールを介してメッセージを受け取ったユーザは，その API を用いて SNS からコンタクト ID 群の情報を取得し，送信者が自分と近い相手かどうかを把握する．送信者が自分に近い相手であればそのメッセージはスパムである可能性は低いとし，正当なものとして受信するということが提案されている．これにより，コンタクト ID に関わらず，SNS 上におけるユーザ間の距離が近いユーザからのメッセージを正当なものとして判別することが可能になっている．しかし，横山らが情報を取得する対象として用いた SNS では，相手に応じて異なるコンタクト ID 群を公開する機能はなく，相手に応じてツールやコンタクト ID を使い分けるというコミュニケーションツールの利用形態への対応が不十分であると言える．送信者のコンタクト ID 情報を取得するための場として，コミュニケーションツールの現在の利用形態にあったものを準備する必要がある．

4. 提 案 手 法

意図した相手からのメッセージを相手のコンタクト ID に関わらず正しく受信することが本研究の目的である．そのためには，メッセージの送信に用いられたコンタクト ID がどのユーザのものかを正確に把握する必要がある．

	ID: A	ID: B
http://bob.example.com/	公開	非公開
http://carol.example.net/	非公開	公開

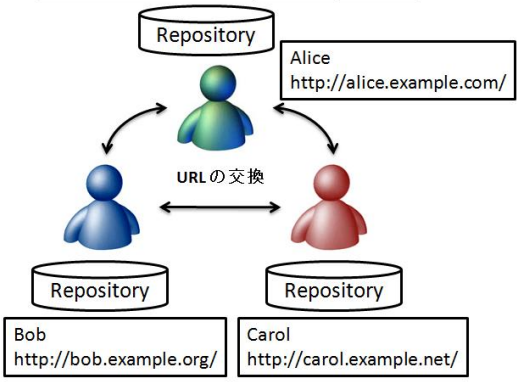

図 1 コンタクト ID の管理

本研究において我々は，各ユーザが Web 上で自身のコンタクト ID を管理し，その情報をメッセージの受信者が取得することで，ユーザごとの受信制御を可能にする手法を提案してきた[12]．本章では，その提案手法についてまとめる．

4.1 Web 上でのコンタクト ID 管理

本研究の目的を達成するためには，受信者がメッセージの送信に用いられたコンタクト ID の所有者を把握する必要がある．コンタクト ID の使い分けという利用形態を考えると，コンタクト ID 情報を取得する際に受信者ごとに異なる情報を取得させるようにすることが重要である．そのために本研究では Web 上で各ユーザが自身のコンタクト ID 群を管理することを提案する．

本研究で提案するコンタクト ID の管理手法は**図 1** の通りである．各ユーザは自身のコンタクト ID を管理する機構 (以下，リポジトリ) を 1 つずつ Web サイトの形で所有する．その後リポジトリの URL を，すでにコミュニケーションツールを用いて連絡を取り合っているユーザや，今後そうしたいユーザと交換する．このとき，リポジトリの URL とユーザは 1 対 1 に対応しており，リポジトリの URL を用いてユーザを識別することが可能である．

ユーザは自身のリポジトリにおいて自身のコンタクト ID の管理を行う．具体的には，コンタクト ID 一覧の作成，ならびに各コンタクト ID をどのユーザに対して公開するかの決定を行う．公開先ユーザを指定する際には，ユーザ間で交換した URL を用いる．各リポジトリの所有者による公開先設定後，所有者以外のユーザがリポジトリにアクセスすると，そのユーザに対して公開が許されているコンタクト ID 群が取得できるようにする．これにより，相手に応じてツールやコンタクト ID を使い分けるという利用形態に対応したコンタクト ID の管理場所を準備できる．

リポジトリにアクセスしてきたユーザに適切なコンタクト ID 情報を表示するためには，そのユーザの認

証が必要となる．この際の認証には，OpenID 認証が適していると考えられる．OpenID 認証は Web での利用が想定され，URL を認証の際のユーザ識別子として用いるシングルサインオン認証方式である[10]．OpenID 認証の詳細な方式については付録 A.1 で説明する．OpenID 認証を用いることにより，各ユーザのリポジトリの URL を認証の際の識別子としても利用することが可能になる．すなわち，認証のための識別子を各ユーザがリポジトリの URL の他に管理しなくてよいという利点がある．

アクセスしてきたユーザを認証し公開が許可されているコンタクト ID 群を返すことで，各リポジトリの管理者が相手ごとにコミュニケーションツールやコンタクト ID を使い分けることが可能になる．これにより，既存の SNS でコンタクト ID 情報を管理する手法[11] に比べ，コミュニケーションツールの利用形態により対応したコンタクト ID 管理が実現できる．

4.2 メッセージ受信時のコンタクト ID 情報取得

コンタクト ID ごとではなく送信者ごとに受信制御を行うためには，どのユーザからのメッセージであれば正当なものとして処理するかというリストを記述できるようにする必要がある．そして，記述されたリストに基づいて実際にメッセージの受信制御を行う必要がある．

リストの記述の際には，正当なものとして処理したいユーザのリポジトリの URL を用いることを提案する．リポジトリの URL はユーザと 1 対 1 に対応しており，URL の記載によってユーザごとの受信制御リストを作成することが可能である．

ユーザはコミュニケーションツールを介してメッセージを受け取った際に，リストに記載されているリポジトリ群からコンタクト ID 情報を取得していく（**図 2**）．そのメッセージの送信に用いられたコンタクト ID と一致するものが取得した情報の中に存在した場合，そのメッセージは誤判別から保護されるようにする．このようにすることで，事前に交換していないコンタクト ID を用いてメッセージを送信したとしても，そのメッセージの送信者がリポジトリで管理する自身のコンタクト ID の公開設定を適切に保っていれば，受信者がそのメッセージを保護することが期待される．したがって，コンタクト ID に変更が生じてもユーザ間のコミュニケーションに支障をきたさないようにできると言える．

各コンタクト ID に信頼度を割り振る手法[1)3)] では，未交換のコンタクト ID を用いてメッセージを送信した場合，信頼度が高くなければそれを保護することができなかった．提案手法ではメッセージの送信に用いられたコンタクト ID が未交換であっても，自分がリストに記載した正式なユーザのものかどうかを識別できる．このため，信頼度を割り振る手法よりも，利用可能範囲が広いと言える．

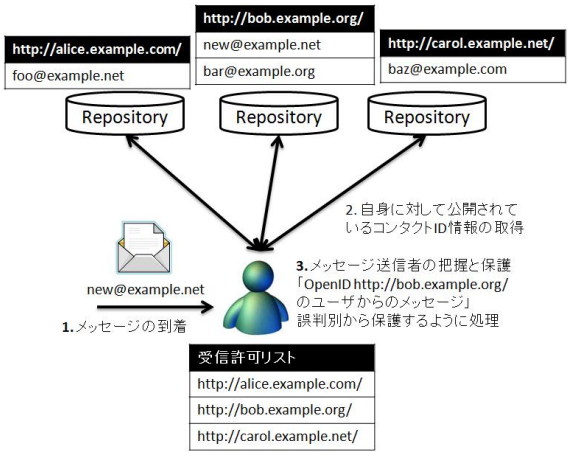

図 2 提案手法におけるメッセージ受信時の動作

5. 実　　装

本論文で我々は，リポジトリを Web サーバ上で動く CGI プログラムとして実装した．また，リポジトリ群からコンタクト ID 情報を取得するアプリケーションを，ローカルホスト上で動作するデーモンプロセスとして実装した．いずれも実装に用いた言語は Perl である．

5.1 リポジトリの動作

リポジトリにアクセスした際に，そのリポジトリの所有者は自身のコンタクト ID 群の公開先を設定でき，それ以外のユーザは所有者が公開を許しているコンタクト ID の情報を取得できるようにする必要がある．アクセスしてきたユーザが誰かを識別するために，リポジトリはアクセスしてきたユーザに対して OpenID の入力を受け付けるフォームを表示し，OpenID 認証を求める．入力された OpenID およびその認証結果に応じて，リポジトリはユーザの識別を行う．入力された OpenID がリポジトリの URL と同一で認証も成功している場合は，そのユーザはリポジトリの管理者であるとみなされる．入力された OpenID がリポジトリの URL とは異なるが認証は成功している場合は，コンタクト ID 情報を取得しにきたユーザであるとみなされる．認証が失敗している場合は，上記のいずれでもないとして，再度 OpenID 認証を求める．

管理者は，リポジトリにおいて，管理したい自身のコンタクト ID 群の追加，コンタクト ID を公開するかしないかを設定する対象ユーザの追加と削除，各コンタクト ID の公開先の設定を Web ブラウザ経由で行うことができる．そのユーザインタフェースは**図 3** である．画面左上のフォームにおいて，自身のコンタクト ID を，それをどのアプリケーションで利用しているかを指定して追加できる．画面右上のフォームにおいて，自身のコンタクト ID 群の公開設定を行う相手を，相手の OpenID を用いて追加できる．画面中央の表においては，どの相手に対してどのコンタクト

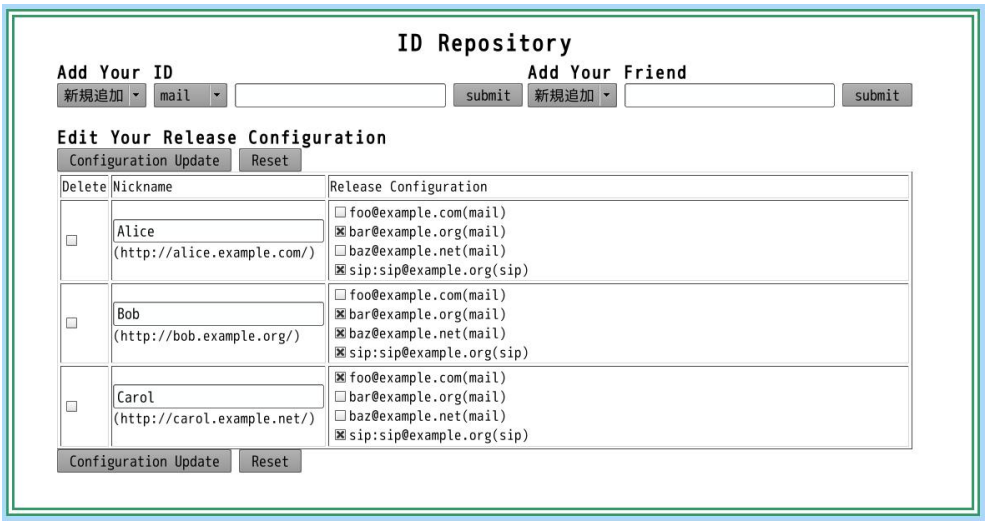

図 3　リポジトリの管理者画面

ID 群を表示するかを決定できる．公開相手の指定は相手の OpenID を用いて行うが，その際には，その OpenID の持ち主の名前をニックネームとして入力し OpenID の持ち主が誰かを管理者自身にとってわかりやすい形で記録しておくことが可能である．なお，各コンタクト ID やそれらの公開先設定といった情報を記録する際には，データベースソフトの 1 つである SQLite[⋆1] を用いた．

管理者以外のユーザは，OpenID 認証が成功している場合，リポジトリから自身に公開されているコンタクト ID 情報を取得することができる．例えば，図 3 で表されるリポジトリに対して http://alice.example.com/ という OpenID を持つユーザが Web ブラウザを用いてアクセスし，その認証が成功している場合，Web ブラウザに mailto:bar@example.org，sip:sip@example.org が表示される．この際，mailto: など，各コンタクト ID が利用されているツールに合わせて，IANA[⋆2] で定められている URI スキーム名が付与されて表示される．

リポジトリは管理者が最後に設定を行った日時をタイムスタンプとして記録する．このタイムスタンプはリポジトリの index.html の中に記載され，OpenID 認証を介せずにそのリポジトリにアクセスした全ユーザに対して公開される．このタイムスタンプの果たす役割については 5.2 節で説明する．

5.2　コンタクト ID 取得ツールの実装

5.1 節において，リポジトリからのコンタクト ID 情報取得を行う際に，Web ブラウザ経由でのアクセスを例に挙げた．しかしながら，リポジトリから取得したコンタクト ID 情報を各コミュニケーションツールから利用できるようにするためには，Web ブラウザに依存しない情報取得手段が必要である．また，メッセージの受信後にその送信者の確認を行う提案手法においては，送信者の確認に必要な時間をできるだけ短縮するような実装モデルを考えることも重要である．さらに，提案手法の汎用性を高めるためには，多くのコミュニケーションツールから提案手法を利用できるように，各ツールへの変更を小さく抑えることも重要である．

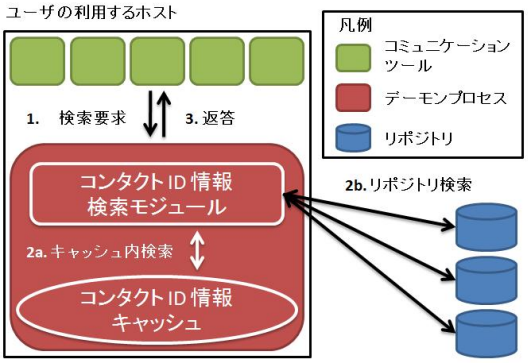

図 4　デーモンプロセスの動作

本研究ではその考えの元で，コンタクト ID 情報の取得を行う部分をローカルホスト上で動くデーモンプロセスとして実装した．このデーモンプロセスは各コミュニケーションツールからの検索要求にしたがって各リポジトリからコンタクト ID 情報の取得を行う．また，リポジトリから取得した情報を自身の中に一時的に保存しておくキャッシュ機構も持つ．このキャッシュには，どのリポジトリからどのコンタクト ID 情報を取得したかという情報と，各リポジトリが記録しているタイムスタンプが保存される．

なお，リポジトリからコンタクト ID 情報の取得を行うためには，情報取得対象となるリポジトリの URL のリスト，OpenID 認証を行うための情報が必要であるが，これらはデーモンプロセスの設定ファイルとし

⋆1 http://www.sqlite.org/
⋆2 Internet Assigned Numbers Authority, http://www.iana.org/

て事前に各ユーザが準備するものとする．

このデーモンプロセスの動作概要は**図 4** の通りである．

(1) デーモンプロセスは通常時は TCP の 49125 番ポートで各コミュニケーションツールからのコンタクト ID 情報検索要求を待つ．コミュニケーションツールは，メッセージが届いた際にデーモンプロセスに対し，メッセージの送信に用いられたコンタクト ID の所有者検索を要求する．この検索要求には，コミュニケーションツールの種類とコンタクト ID が含まれている．

(2) デーモンツールはコミュニケーションツールから受け取った検索要求に基づいて，そのコンタクト ID の所有者が誰かを検索する．
検索の際，まず自身の持つキャッシュ内に要求に合致するコンタクト ID 情報がないかを検索する．キャッシュ内に合致するものがあれば，ここで検索を終了する．
キャッシュ内に合致するものがなければ，情報取得対象のリストに記載されているリポジトリからコンタクト ID 情報の取得を行う．その際，時間短縮のため，まずリポジトリからタイムスタンプを取得し，キャッシュ内のタイムスタンプと比較を行う．キャッシュ内のタイムスタンプと取得したものが同じ場合，そのリポジトリからのコンタクト ID 情報取得は行わない．キャッシュ内のタイムスタンプよりも取得したものの方が新しい場合，またはキャッシュ内にそのリポジトリのタイムスタンプが存在しない場合は，そのリポジトリからコンタクト ID 情報取得を行う．リポジトリからの情報取得にはリポジトリへの OpenID 入力および OpenID 認証が必要となるが，その際にデーモンプロセスは設定ファイルから自身の OpenID 情報や，その認証に必要な情報を読み込み，対処する．
ここで取得されたタイムスタンプやコンタクト ID 情報とキャッシュ内の情報に差異があれば，キャッシュ内の情報を更新する．

(3) 検索要求されたコンタクト ID と一致するものがキャッシュまたはリポジトリから取得した情報に存在した場合は，コミュニケーションツールに対してその所有者の OpenID（そのコンタクト ID が管理されていたリポジトリの URL）を返す．要求されたコンタクト ID の所有者が全リポジトリからの情報取得後も判明しなかった場合には，コミュニケーションツールに対し，”Not Found”という文字列を返す．

本実装の利用により，提案手法を用いるために各コミュニケーションツールに加える必要のある変更は以下の 2 つである．1 つ目は，届いたメッセージから送信に用いられたコンタクト ID を抜きだし，それをデーモンプロセスに検索要求として渡すことである．2 つ目は，デーモンプロセスから OpenID が返ってくればそのメッセージを正当なものだとして誤判別から保護するように処理し，”Not Found” が返ってくればそれを特に保護する必要のないメッセージだとして処理することである．各コミュニケーションツールにデーモンプロセスの行う検索機能を追加するよりも少ない変更で済むと言える．

また，本実装においては，あるコミュニケーションツールからの検索要求に答える際に作られたキャッシュが，他のコミュニケーションツールからの検索要求に答える際にも利用できる．キャッシュ内の情報で検索要求に答えられる可能性が高くなると，リポジトリから情報を取得する頻度を下げることができる．これにより，要求に答えるのに必要な時間の短縮が見込める．

表 1 マシンスペック

	CPU	メモリ	OS
ホスト A	Athlon 64X2 4600+	4GB	Linux 2.6.24
ホスト B	Athlon 64X2 4600+	4GB	Linux 2.6.24
ホスト C	CoreDuo T2500	2GB	Linux 2.6.24

6. 評　　価

提案手法ではメッセージの受信後に送信者の確認を行うため，その確認に必要な時間の見積りが提案手法の可用性を考察する上で重要である．その見積りを行うために，5 章の実装下で時間の計測を行った．

6.1 コンタクト ID 情報取得に必要な時間

複数のリポジトリからコンタクト ID 情報を取得するために必要な時間の見積りを行うため，1 つのリポジトリからの情報取得に必要な時間の計測を行った．その際に用いたマシンのスペックについては，**表 1** の通りである．ホスト A，B，C は同一データリンク上にあり，各ホスト間の RTT は 1[ms] 以下であった．

ホスト A，B では Web サーバとして Apache[⋆1] を動作させ，ホスト A ではリポジトリを，ホスト B では OpenID 認証に必要な OpenID Provider として SimpleID[⋆2] を動作させた．SimpleID は OpenID Provider のオープンソースな実装の 1 つであり，用いられているプログラミング言語は PHP である．

ホスト C では 5.2 節のデーモンプロセスを動作させた．デーモンプロセスに検索要求を送る部分については，コミュニケーションツールに追加するのではなく，それ専用のクライアントアプリケーションを別途作成することで時間の計測を行った．このクライアントアプリケーションは，ローカルホストで動作するデーモンプロセスに対して TCP で接続し，検索要求を送り，それに対する返答を受け取る．コミュニケーションツールに提案手法を利用する機能を追加する際にはそれぞれのツールに合わせたプログラミング言語で実装を行う必要があるが，このクライアントアプリケーションは C と Perl を用いて実装した．2 種類の

⋆1 http://www.apache.org/
⋆2 http://simpleid.sourceforge.net/

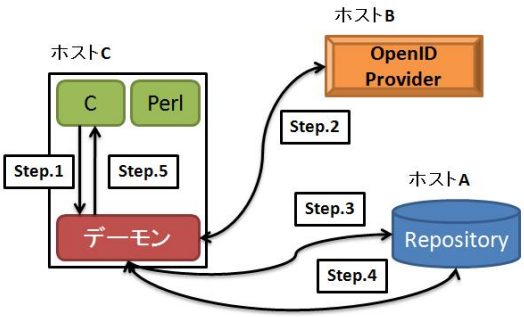

図 5 計測時の環境

言語で実装したのは，利用するプログラミング言語の差が，デーモンプロセスとの通信に必要な時間にどの程度の影響を与えるかを予測するためである．

計測時の動作は**図 5** の通りである．

(1) クライアントアプリケーションがデーモンプロセスに対してコンタクト ID の検索要求を出す．

(2) 要求を受け取ったデーモンプロセスは OpenID Provider に，自身の OpenID とその認証に必要なパスワードを送信し，認証状態を確立させる．なお，次に行うタイムスタンプの取得には認証状態の確立は必要ではないが，今回の計測ではタイムスタンプの取得の前にこの動作を行った．実際の動作ではタイムスタンプを比較しコンタクト ID 情報の取得が必要だと判断した段階で，この動作を行う．

(3) デーモンプロセスはリポジトリにアクセスをして index.html を取得し，そこからタイムスタンプを抜き出す．抜き出したタイムスタンプと自身のキャッシュ内にあるタイムスタンプを比較する．実際の動作では比較結果に応じてコンタクト ID 情報の取得を行うかどうかを判別するが，今回は比較結果に関わらずコンタクト ID 情報取得を行うようにして計測を行った．

(4) デーモンプロセスがリポジトリに自身の OpenID を入力し，そのリポジトリからコンタクト ID 情報を取得する．取得したコンタクト ID 情報とクライアントアプリケーションからの検索要求を比較し，比較結果をクライアントアプリケーションに返す．

(5) クライアントアプリケーションは，デーモンプロセスからの返答を受け取る．

C で実装したクライアントアプリケーションがデーモンプロセスに検索要求を出しそれに対する答えを取得するまでの時間（図 5 における Step.1 から Step.5）を 100 回，Perl で実装したクライアントアプリケーションも同様に 100 回計測した．その際，デーモンプロセスが検索要求を受け取りそれに対しての答えを返すまでの時間（図 5 における Step.2 から Step.4）も合わせて計測し，両者の差を取ることでデーモンプロセスとの通信に必要な時間（図 5 における Step.1 と Step.5 の和）を求めた．

デーモンプロセスが，ホスト B で動作する OpenID Provider において認証状態を確立するのに必要な時間については 100 回の計測を行った．

C と Perl による計 200 回の計測の結果は**表 2**，**図 6**，**図 7**，**図 8** の通りである．

図 6 は，図 5 における Step.2 に必要な時間の計測結果の累積相対度数分布である．全試行の内 95%以上が 160[ms] 以内に終了し，平均では 150[ms] かかっている．

図 7 は，図 5 における Step.3 に必要な時間の計測結果の累積相対度数分布である．全試行の 95%以上が 50[ms] 以内に終了し，平均では 42[ms] がかかっている．デーモンプロセスがリポジトリから情報取得を行う際，タイムスタンプに更新がなければこの程度の時間で 1 つのリポジトリからの情報取得が完了すると言える．

図 8 は，図 5 における Step.3 に必要な時間と Step.4 に必要な時間の和を累積相対度数分布の形で表したものである．全試行の内 95%以上が 560[ms] 以内に終了し，平均では 550[ms] かかっている．リポジトリのタイムスタンプに更新がありコンタクト ID 情報の取得を行う場合，OpenID の認証に必要な時間に加えて，1 つのリポジトリにつきこの時間が必要だと言える．

クライアントアプリケーションとデーモンプロセス間の通信は，今回の実装の範囲ではプログラミング言語によらず数 [ms] で終了することがわかった．各コミュニケーションツールに提案手法を利用する機能を追加する際にも，この程度のオーバーヘッドで可能であることが予想される．

以上の結果により，N 個のリポジトリからの情報取得に必要な時間の見積りを行った．今回の実装下においては，一度 OpenID の認証状態を確立すればそれを元に複数のリポジトリからコンタクト ID 情報を取得することが可能である．そのため，N 個のリポジトリすべてからコンタクト ID 情報を取得する場合でも OpenID 認証の確立は一度だけ行えばよく，表 2，図 6 の通りとなる．したがって，$n(0 < n \leq N)$ 個のリポジトリにおいてタイムスタンプの更新があった場合，全リポジトリからの情報取得が 95%以上の確率で完了するのに最低限必要な時間 T[ms] は式 (1) のように見積もられる．N 個のリポジトリにおいてタイムスタンプが更新されているものがなければ，式 (2) のように見積もられる．

$$T = 160 + 560n + 50(N - n) \quad (1)$$

$$T = 50N \quad (2)$$

見積りに使用したデータは各ホストが同一データリンク上にある場合のデータであり，各ホスト間の RTT の増加などによってさらに長い時間がかかることが予想される．

6.2 時間短縮に向けての並列アクセスの可能性

6.1 節では 1 つのリポジトリからの情報取得についての時間計測を行い，その結果を用いて複数リポジトリからの情報取得に必要な時間の見積りを行った．し

表 2 計測結果の平均および標準偏差

	平均 [ms]	標準偏差 [ms]
OpenID 認証	150	27
タイムスタンプの取得と比較	42	3.6
コンタクト ID 情報取得	550	7.9
デーモンプロセスとの通信（C）	2.9	1.8
デーモンプロセスとの通信（Perl）	3.1	3.9

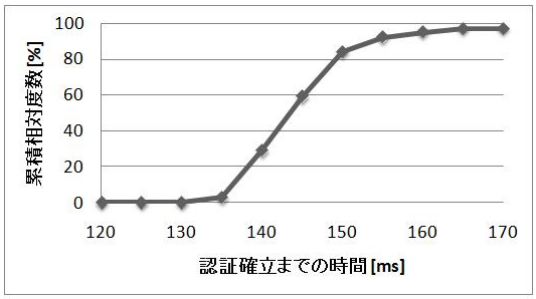

図 6 OpenID 認証確立に必要な時間

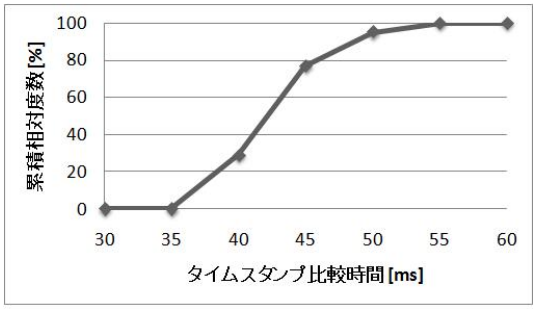

図 7 タイムスタンプの取得と比較に必要な時間

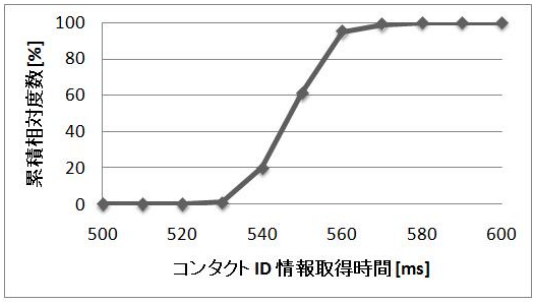

図 8 コンタクト ID 情報の取得に必要な時間

かしながら，複数リポジトリからの情報取得の際には，リポジトリへの並列アクセスにより時間の短縮を図ることができると考えられる．

OpenID の認証状態を確立するためのアクセスやコンタクト ID 情報の取得のためのアクセスはすべて HTTP が用いられる．並列アクセスによる効果を確認するために，20 の Web サーバに並列アクセスを行いそれによって全サーバからのレスポンスを処理するのにかかる時間が短縮されるかどうかを確認した．これら 20 の Web サーバそれぞれの平均応答時間は一番早いもので 15[ms]，一番遅いもので 17[ms] であった．これらにマルチスレッド環境でアクセスした結果，シングルスレッド環境でアクセスした場合に比べて，全サーバからのレスポンスを処理するのにかかる時間は短縮された．これにより，並列アクセスによって HTTP サーバからの情報取得の時間は短縮可能であることがわかった．

この計測に用いた Web サーバのレスポンス時間と提案手法の実装の各処理に必要な時間には差がある．そのため提案手法の実現の際に最適なスレッド数についてはさらに調査が必要であるが，並列アクセスにより提案手法の時間短縮が可能であることが予測できた．

情報取得にかかる時間が 2 秒から 4 秒程度なら不快感なく待つことができるという報告[4] があるが，以上の結果と式（1）（2）より，数十程度のリポジトリ数であればその時間内に情報取得が可能であると予測できる．今後，その時間内に情報取得できるリポジトリ数を増やしていく必要性はあるが，現在でも数十人程度のユーザ間では提案手法の実用性はあると言える．

6.3 アクセス集中による影響

6.1 節では 1 つのリポジトリに対して 1 つのホストからアクセスを行うことで評価を行った．しかしながら，実際の環境では複数のホストがリポジトリにアクセスすることが想定される．我々は，リポジトリに対するアクセスの集中度合いとそれによる情報取得時間への影響を明らかにするために，シミュレーションおよび計測を行った．

提案手法ではメッセージの到着時にリポジトリにアクセスを行うため，アクセスの頻度はメッセージの到着間隔に依存する．ここでは，メールクライアントアプリケーションにおける新着メッセージ確認間隔の設定が分単位であることが多いことなどを考慮し，到着間隔を 1 分だと仮定した．その仮定の下で 50 人，100 人，150 人，200 人のユーザがリポジトリにアクセスを行う場合，あるユーザがリポジトリにアクセスする際に他のユーザが同一リポジトリにアクセスしている確率は，シミュレーションの結果，**図 9** のようになった．図 9 の横軸は同時にアクセスしている他のユーザの数を表すので，同時にアクセスしているユーザ数はそれに 1 を足した数になる．いずれの場合も，今回の仮定の下では同時アクセス人数が 5 人以下である確率が 95%以上を占めた．

ホスト C でクライアントアプリケーションを並列動作させてリポジトリへのアクセスを集中させ，それによる情報取得時間への影響を示したものが，**図 10** である．各同時アクセス数ごとに 100 回の計測を行った．アクセスが集中するにつれて情報取得にかかる時間が増加している．図 9 の結果と合わせると，情報取得にかかる時間の期待値は 50 人の場合で 581[ms]（アクセス集中を考慮しない場合と比較して 5%増），100 人の場合で 637[ms]（同，16%増），150 人の場合で 705[ms]（同，28%増），200 人の場合で 776[ms]（同，

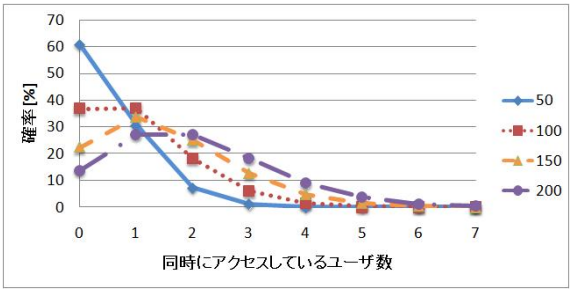

図 9 リポジトリにアクセスするユーザの重複確率

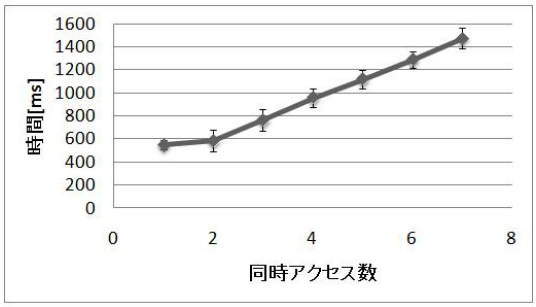

図 10 同一リポジトリに同時にアクセスしている人数

41%増）となった．

アクセス集中による影響は，現時点で実用性があると思われる数十人規模であれば数%であり小さいと言えるが，今後利用規模の拡大に伴い無視できないほど大きくなってくることがわかった．利用規模の拡大を図る際には，コミュニケーションツールの利用形態に合わせたアクセス頻度の調整などを行い，アクセスの分散を図ることが必要になってくると言える．

7. 結論と今後の課題

本研究において我々は，相手のコンタクト ID の把握をメッセージの到着時に行うことでそのメッセージの送信者を把握する手法を提案してきた[12]．具体的には，Web 上に各ユーザが自身のコンタクト ID の公開設定を管理するリポジトリを所有し，ここから他のユーザが OpenID 認証を通じてコンタクト ID 情報を取得し，メッセージの判別に利用するという手法を提案した．これにより，従来のホワイトリストにおいて未解決であった，現在のコミュニケーションツールの利用形態への対応を可能にした．これは，各コンタクト ID への信頼度割り当てという手法[1)3)] において未解決であった変更直後のコンタクト ID への対応や，SNS からの情報取得という手法[11)] において未解決であった複数コンタクト ID の使い分けへの対応を可能にするものである．

本論文では提案手法の実装を行った．コンタクト ID を管理するリポジトリを Web サーバ上で動く CGI として実装した．リポジトリから情報を取得し，それを各コミュニケーションツールに提供するための機能をデーモンプロセスとして実装した．デーモンプロセスとして実装することで，提案手法を利用するために各コミュニケーションツールに追加しなければならない機能を削減することができた．また，取得した情報をデーモンプロセスがキャッシュとして管理し，それを各コミュニケーションツールから利用できるようにすることで情報取得に必要な時間の短縮が可能になった．

この実装の下で，リポジトリからの情報取得に必要な時間の計測および見積りを行い，提案手法の実用性について具体的に検証した．その際，リポジトリからの情報取得の際にアクセスの並列化が図れるかどうかも検証した．その結果，複数リポジトリからの情報取得に必要な時間を並列アクセスにより短縮することが可能であり，数十人規模のユーザ間であれば現在の実装でも実用的であることがわかった．これにより，我々は，提案手法およびその実装が今後のコミュニケーションツールの利用において実用的であるということを示した．

今後の課題としては，まず，複数リポジトリからの情報取得にかかる時間のより詳細な見積りを行うことが挙げられる．リポジトリを複数台用意し，その元での情報取得，および並列アクセスの評価を行うことにより提案手法がどの程度の人数のユーザ間で利用可能かを明らかにし，その人数を増やしていくことが課題である．特に，待ち時間に関する報告[4)] を考慮し，その時間内に情報取得を完了できるリポジトリ数を増やすことが重要であると考えている．また，デーモンプロセスに検索要求を送りそれに対する返答を受け取り，それに応じてメッセージを処理する機能を各コミュニケーションツールに追加することも今後の課題である．さらに，本稿での実装ではリポジトリにおけるコンタクト ID の公開設定は各ユーザが手動で行うようにしていたが，メッセージの交換状況に応じて公開設定を自動で行うような機能追加も，提案手法の利便性を向上させる上で重要であり，今後の課題と言える．

付　　録

A.1 OpenID 認証

OpenID 認証は，Recordon らによって提案されたシングルサインオン認証方式である[10)]．OpenID 認証では，URL をユーザの識別子として利用できる．OpenID 認証において，ユーザからの要求に対して OpenID の発行と，その後の認証を行う機関を OpenID Provider（以下，OP） と呼ぶ．OpenID 認証を利用して Web サービスを提供する機関を Relying Party（以下，RP）と呼ぶ．

OpenID 認証を利用する際に，ユーザは OP から OpenID の発行を受ける必要がある．ユーザはその発行の際に，パスワードや公開鍵といった，今後その OpenID 認証に必要な情報を OP に登録する．OpenID として URL を発行されたユーザは，その URL でアクセスできる Web サイト下の文書に自身の OP の URL

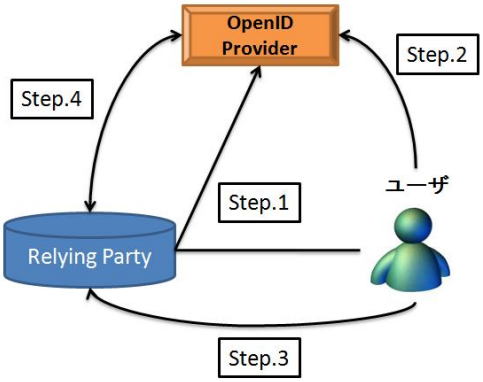

図 11 OpenID 認証

を記載する．

OpenID 認証の流れは**図 11** の通りである．RP は，アクセスしてきたユーザに対して OpenID を入力させるフォームを表示する．入力を受けた RP はその OpenID 下の文書を解析して OP の URL を取得し，ユーザはそこへ転送される（Step.1）．OP に転送されたユーザは，OP に対して OpenID の認証に必要な情報を入力する（Step.2）．入力を行ったユーザは RP へ転送される（Step.3）．ユーザに再度アクセスされた RP は，OP に対してその OpenID の認証結果を問い合わせる（Step.4）．認証結果に応じて RP はユーザにサービスを提供する．一般に，認証結果が真であればサービスが提供され，偽であれば提供されない．

参　考　文　献

1) Balasubramaniyan, V., Ahamad, M. and Park, H.: CallRank: Combating SPIT Using Call Duration, Social Networks and Global Reputation, *Conference on Email and Anti-Spam* (2007).
2) Cerf, V.G.: Spam, spim, and spit, *Commun. ACM*, Vol.48, No.4, pp.39–43 (2005).
3) Chirita, P., Diederich, J. and Nejdl, W.: MailRank: using ranking for spam detection, *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp.373–380 (2005).
4) Culotta, A., Bekkerman, R. and McCallum, A.: A study on tolerable waiting time: how long are Web users willing to wait?, *Behaviour & Information Technology*, Vol. 23, pp. 153 – 163 (2004).
5) Evett, D.: Spam Statistics 2006, *TopTenREVIEWS http://spam-filterreview. toptenreviews. com/spam-statistics. html* (2006).
6) Fallows, D.: Spam: How it is hurting email and degrading life on the Internet, *Pew Internet and American Life Project* (2003).
7) Hershkop, S. and Stolfo, S. J.: Combining email models for false positive reduction, *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, New York, NY, USA, ACM, pp.98–107 (2005).
8) Leavitt, N.: Instant Messaging: A New Target for Hackers, *IEEE Computer*, Vol.38, No.7, pp. 20–23 (2005).
9) Liu, Z., Lin, W., Li, N. and Lee, D.: Detecting and filtering instant messaging spam - a global and personalized approach, pp.19–24 (2005).
10) Recordon, D. and Reed, D.: OpenID 2.0: a platform for user-centric identity management, *DIM '06: Proceedings of the second ACM workshop on Digital identity management*, New York, NY, USA, ACM, pp.11–16 (2006).
11) Yokoyama, T., Kashihara, S., Okuda, T., Kadobayashi, Y. and Yamaguchi, S.: A Generic API for Retrieving Human-Oriented Information from Social Network Services, *Applications and the Internet Workshops, 2007. SAINT Workshops 2007. International Symposium on*, pp.33–33 (2007).
12) 阪本裕介，中山雅哉：OpenID を用いたユーザ指向型ホワイトリスト作成手法の提案，情報処理学会研究報告, 2008-GN-67，pp.73–78 (2008).