2009－GN－71 (23)
2009／3／19

コンタクト ID 変更に対応したユーザ主導型コミュニケーション手段選択方式の提案と評価

阪本裕介[†1] 中山雅哉[†2]

現在インターネット上では様々なコミュニケーションツールが利用されている．電子メールアドレスなどのコンタクト ID や各種ツールの使い分けおよび変更は，相手とのコミュニケーション手段の制御を可能にする．その一方で，そのような利用はメッセージ送信者の判別を困難にし，メッセージがスパム判別手法により誤判別される原因となる．我々は，メッセージ到着時におけるコンタクト ID 確認によってメッセージ保護の有無を決定する受信制御方式を提案する．提案手法の実装およびその動作時間計測の結果，実用的な待ち時間内にコンタクト ID の確認が完了することがわかり，提案手法に実用性があることがわかった．

Proposal and Evaluation of an Adaptive Communication Method for Dealing with Dynamic Contact ID Updates

Yusuke Sakamoto [†1] and Masaya Nakayama[†2]

Users select their Contact IDs (e.g. E-mail Addresses) to communicate to others depending on partners. Selective use is widely used because it enables users to control their communication methods. However it causes false positives in spam detection methods, because identifying the senders of messages becomes difficult. We propose a method to protect valid messages based on checking the owner of Contact IDs. We implemented the middleware for checking the sender, and measured its searching time. Our result shows that our middleware provides a good performance in searching a sender.

1. はじめに

現在，インターネット上では電子メールやインスタントメッセンジャーなど様々なコミュニケーションツール（以下，単にツールと呼ぶ.）が利用されている．ユーザがこれらを利用する際には，各ツールごとに電子メールアドレスやインスタントメッセンジャーのアカウントといった ID が必要になる．

ユーザは相手に応じて異なるツール，異なる ID を使い分けることで，コミュニケーション手段の選択を行っている．このような利用方法はユーザの利便性を高める一方で，相手と未交換の ID を用いる場合はそのメッセージがスパムだと判別される危険性を有している．このような誤判別はコミュニケーションに大きな損失を与える[9] ため，誤判別を防ぎながら，コミュニケーション手段をユーザが自由に選択できる手法の提供が望まれている．

既存の手法では，複数のツールで総合的にスパムの誤判別を防ぐ方式や，ひとりのユーザが複数 ID を使い分けるケースには適用できないものがほとんどであったのに対し，我々は，この問題を解決するためのメッセージ受信者が主導してアクセス制御を行う新しい方式の提案を行ってきた[10)11)12)]．この方式では，各ユーザが利用する各ツールとその ID（以下，本稿ではコンタクト ID と呼ぶ.）を Web 上で管理し，他のユーザからメッセージを受信した際にユーザがアクセスを許可する相手のコンタクト ID 情報を動的に確認して，受信したメッセージの正当性判別を行う方式で前述した誤判別を軽減することが可能となる．文献 11)12) ではコンタクト ID の管理を CGI プログラムとして実装して，実用的な時間で動作することを示してきた．

本稿では，各種ツールから共通して，受信メッセージの正当性判別を行うことができるようなミドルウェアを実装するとともに，ツールの一例として Mozilla Thunderbird（以下，Thunderbird と呼ぶ.）のアドオンを作成して，提案手法の効果を調べることにした．

また，提案手法の適用範囲を明らかにするため，各

†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

ユーザのコンタクト ID 変更頻度を変化させた時の検索時間の計測と考察を行うことにした．

2. 背景と目的

現在多くの人々はインターネット上の様々なツールを利用したコミュニケーションを行っている．このため，近年は，スパムと呼ばれる受信者が希望していないメッセージを無作為に送信する行為が頻発するようになってきた[1]．そこで多くのツールではコンタクト ID を用いた受信制御が行われる様になってきた．例えば，多くの MUA(Mail User Agent) では，アドレス帳に登録された相手から送られてくる電子メールをホワイトリストとして扱い，スパムと判別しないような受信制御を行うことでメッセージの保護を可能にしている．

最近では，ひとりのユーザがコンタクト ID を複数所有し，コミュニケーション相手に応じて使い分けることが一般的に見られるようになってきた．職場でのコンタクト ID とプライベートでのコンタクト ID を別々に用意し，利用目的ごとに使い分ける例が挙げられる．このようなコンタクト ID の使い分けはコミュニケーション手段の制御を可能にし，利便性を向上させる一方で，同じ通信相手に通常と異なるコンタクト ID でメッセージを送信する場面も多くなる．この場合，通常やりとりするコンタクト ID を用いた場合はホワイトリストとして扱われるメッセージも，相手にとって未知のコンタクト ID が用いられることでスパムとして判別されることになる．

この状況は，自身が利用するコンタクト ID を変更した場合にも生じる．通信相手が，新しく変更したコンタクト ID をホワイトリストとして扱うように適切な更新を行わなければ，送信したメッセージがスパムとして判定される危険性がある．

正当メッセージの保護が不可能になることによる影響は大きい．正当なメッセージがスパムとして誤判別される実際の確率は数%と低いが[4)5)6)]，その誤判別を不安に感じるユーザは全体の数十%に上る[3)]．

以上の点を考慮し，本稿では，コンタクト ID の使い分けや変更に対応したメッセージの受信制御方式の実現を目指す．その際に，ホワイトリストのメンテナンスコスト軽減を考え，受信制御条件が相手のコンタクト ID 仕様状況に応じて自動的に変更されること，その方式が多くのツールから利用可能な統一的受信制御基盤となることを目的とする．

3. 関連技術

コンタクト ID の使い分けや変更を管理し，目的の相手に正しく通知することを可能にするアプリケーションが存在する．Ripplex[*1] はリプレックス株式会社が開発した統合アドレス帳アプリケーションであり，ユーザのホスト上で常駐し各 Ripplex 間でコンタクト ID 情報を同期することでアドレス帳の情報を最新に保つことを可能にする．Ripplex においては各コンタクト ID の細かな公開制限とコンタクト ID 変更時の通知が可能であるため各ユーザの受信制御条件を適切に更新できるが，Ripplex で管理可能なコンタクト ID の種類は限られており，統一的受信制御基盤として利用するには不十分である．

4. 実　　装

本章では，提案手法におけるリポジトリの実装，およびリポジトリ群からコンタクト ID 情報を取得してメッセージの正当性判別を行うミドルウェアの実装について述べる．また，ミドルウェアを利用してメッセージの正当性判別を行うツールの一例として Thunderbird のアドオンを作成したので，それについての報告も合わせて行う．なお，提案手法の内容については文献 10)12) に譲る．

4.1 リポジトリの実装

我々は，Web 上で動く CGI プログラムとしてリポジトリを実装した．各ユーザは 1 つずつリポジトリを所有する．各リポジトリにおいて所有者は自身のコンタクト ID 情報を管理でき，所有者以外のユーザは自身に対して公開が許されているコンタクト ID 情報を取得できるようにする必要がある．リポジトリではそのためのユーザ識別を OpenID 認証[8)] を用いて行っている．そのため，リポジトリにはユーザから OpenID の入力を受け付けるフォームがある．ユーザが入力した OpenID がリポジトリの URL と一致しており，その OpenID 認証も成功している場合，そのユーザはそのリポジトリの所有者であると見なされる．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，そのユーザは所有者がそのユーザに対して公開設定をしているコンタクト ID を取得できる．

所有者は**図 1** のようなインタフェースを用いて各コンタクト ID をどのユーザに公開するかを設定することができる．リポジトリは所有者による設定内容と，その設定内容の最終更新日時 (以下，タイムスタンプと呼ぶ．) を記録する．設定内容に関しては，SQLite[*2] を用いてすべてテキスト形式で記録される．そのため，テキスト形式で表現できるコンタクト ID であればどのようなものでもリポジトリで公開先の管理が可能であり，対応ツールを増やすことが Ripplex に比べて容易である．

タイムスタンプは，OpenID 認証を行わずとも，リポジトリにアクセスした全ユーザに公開されるようにした．タイムスタンプは，4.2 節で述べる検索動作において時間短縮を図る目的で利用される．

---

*1 http://www.ripplex.com/

*2 SQLite: http://www.sqlite.org/

図 1 リポジトリの所有者画面
Fig. 1 User interface of a repository

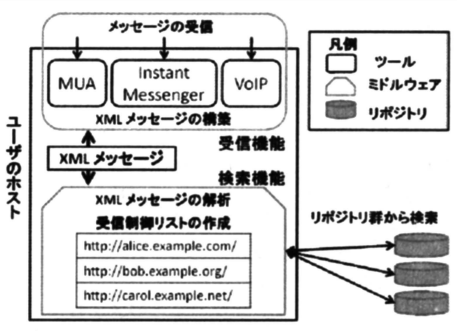

図 2 メッセージ受信機能と検索機能の分離
Fig. 2 Separating receiving function and searching function

4.2 コンタクト ID 検索ミドルウェアの実装

提案手法では，ツールを介してメッセージを受信した際に，そのメッセージに用いられたコンタクト ID 情報とリポジトリ群におけるコンタクト ID 情報を比較し，メッセージの正当性判別を行う．そのためには，メッセージからコンタクト ID 情報を抜き出し，それに関する検索をリポジトリ群から行う必要がある．

我々はリポジトリ群から情報取得を行う機能を各ツールに追加するのは非効率的であると考え，その機能を**図 2** のようにミドルウェアとして分離して実装した．これにより，提案手法の利用において各ツールに必要な変更の軽減を図った．

検索を行うためには，リポジトリからコンタクト ID 情報を取得しなければならず，そのユーザの OpenID ならびにパスワードなどのその OpenID 認証に必要な情報が必要である．また，検索を行うリポジトリ群はメッセージの宛先となっているコンタクト ID ごとに異なるため，そのコンタクト ID 情報も必要である．そして，検索対象としてメッセージの送信に用いられたコンタクト ID 情報も必要である．これらのメッセージをミドルウェアに伝える際のメッセージは，汎用性を考慮して一般的なフォーマットである XML 形式と

```
<?xml version="1.0" encoding="UTF-8"?>
<message>
<method>SEARCH</method>
<auth>
<openid>http://example.com/</openid>
<user>User Name</user>
<passwd>Password</passwd>
</auth>
<to>
<id>mailto:from@example.jp</id>
</to>
<query>
<id>mailto:to@example.com</id>
</query>
</message>
```

図 3 検索要求 XML メッセージ
Fig. 3 XML Message for Searching ID

した．そのメッセージの具体例は**図 3** である．method 要素には検索要求を表す SEARCH が指定され，auth 要素には OpenID ならびにその認証情報が含まれる．本実装内で利用した OpenID Provider[*1]では，ユーザの認証をユーザ名とパスワードの組を用いて行うため，図 3 の auth 要素には認証情報として両者が含まれている．to 要素としてそのメッセージの宛先となっているコンタクト ID が指定され，query 要素としてメッセージの送信に用いられたコンタクト ID 情報が指定される．

ミドルウェアはローカルホスト上で TCP 49152 番ポートでツールからのメッセージを待つ．検索要求を受け取ったミドルウェアは，XML の構文解析を行って各情報を抜き出す．

構文解析を終えたミドルウェアは，検索のためにアクセスを行うべきリポジトリ群を知る必要がある．ミドルウェアは検索要求内の OpenID を元にそのユーザのリポジトリへアクセスし，各コンタクト ID の公開状況を取得する．取得した公開状況から宛先コンタクト ID が公開されている相手を確認し，その一覧を受信制御リストとして作成する．

受信制御リスト作成を終えたミドルウェアは，キャッシュ内の検索を行う．検索の効率を図るため，ミドルウェアは過去にリポジトリから取得した情報をキャッシュとして保有している．このキャッシュには，各リポジトリから取得したタイムスタンプおよびコンタクト ID 情報が入っている．なお，以下に述べる情報取得動作によってキャッシュは適宜更新される．

キャッシュ内に検索要求に合致するコンタクト ID が

*1 SimpleID: http://simpleid.sourceforge.net/

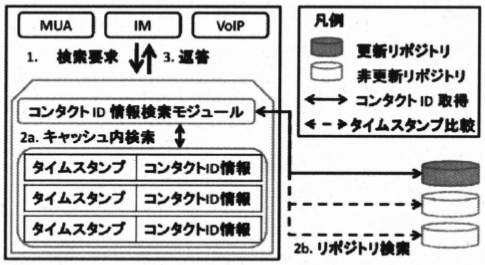

図 4 キャッシュ内検索失敗時のミドルウェアの動作
Fig. 4 Middleware action when the result of searching in cache is failure

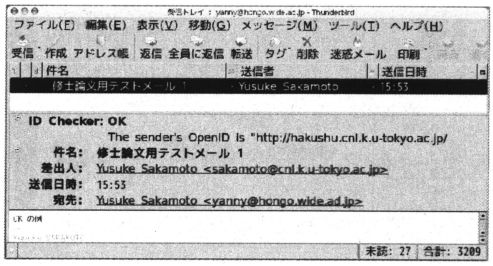

図 5 ミドルウェアを利用する Thunderbird アドオン
Fig. 5 Thunderbird add-on accessing our middleware

見つかった場合は，そのキャッシュの内容が有効性を確認する必要がある．この際，キャッシュ内のタイムスタンプを用いることで検索時間の短縮を図った．ミドルウェアは合致したキャッシュに対応する 1 つのリポジトリにアクセスを行い，タイムスタンプを取得し，キャッシュのタイムスタンプとの比較を行う．両者が一致すればキャッシュは最新で有効だと考えられるため，ミドルウェアは即座にそのリポジトリの URL をツールに返す．この URL は，ツールが受信したメッセージの送信者の OpenID に該当する．両者が異なればキャッシュは古く無効だと考え，ミドルウェアはそのリポジトリから OpenID 認証を介したコンタクト ID 情報取得を行う．取得した情報の中にも検索要求に合致するものがあれば，ミドルウェアはツールに対してそのリポジトリの URL を返す．取得した情報の中に合致するものがなければキャッシュ内検索は失敗だとして次に述べる全検索を行う．

キャッシュ内の検索が失敗した場合は，ミドルウェアは**図 4** のように受信制御リストにある全リポジトリにアクセスを行う．全リポジトリにアクセスをする場合も，まず各リポジトリのタイムスタンプの取得を行い，変更が見られるリポジトリからのみ OpenID 認証を介したコンタクト ID 情報取得を行う．これにより，全リポジトリにアクセスをする場合も検索時間短縮を図った．各リポジトリから情報を取得する中で検索要求に合致するものが見つかれば，その時点で即座にツールに対して見つかったリポジトリの URL を返す．全リポジトリにアクセスをしても検索要求に合致するものが見つからなければ，“Not Found” という特定の文字列をツールに返す．

なお，全リポジトリへアクセスを行う際は，並列アクセスを行うことで検索時間の短縮を図った．ミドルウェアはスレッドを 5 つ生成し，これらを用いて並列アクセスを行った．

4.3 コミュニケーションツールへの実装

ミドルウェアと通信を行って正当性判別を行う機能を既存のツールである Thunderbird にアドオンとして追加した．本アドオンは，ユーザが Thunderbird のユーザインタフェースにおいて電子メールメッセージを選択するとその From: ヘッダと To: ヘッダ，および事前にアドオン内に設定した OpenID 情報等から図 3 のような検索要求を作成し，ミドルウェアに渡す．ミドルウェアはそれを受け取って検索を行う．ミドルウェアによる検索の後に，本アドオンは検索結果を受け取り，それを Thunderbird のヘッダ表示部に ID Checker という項目として表示する．

ミドルウェアが検索要求に対して何らかの URL を返した場合，すなわちメッセージが正当であると判別された場合の表示例は**図 5** である．ID Checker の項目に青文字で “OK” と表示され，その下にミドルウェアから受け取った URL，すなわちそのメッセージの送信者の OpenID が表示される．ミドルウェアが “Not Found” という文字列を返した場合，ID Checker の項目には赤文字で “NG” と表示される．

本アドオンの実装には Javascript を用いた．そのコード量は約 250 行であるが多くはユーザインタフェースの記述部分であり，検索要求作成などミドルウェア利用機能は約 100 行で実現されている．ミドルウェアの検索機能は約 1,100 行の Perl スクリプトで実現されている．実装に用いた言語が異なるため単純な比較はできないが，検索機能の追加に比べ，ミドルウェア利用機能の追加の方が提案手法の利用が容易であると考えられ，ミドルウェアによる検索機能の分離が有効であると結論付けられる．

5. 評　　価

提案手法においてはメッセージの受信時にその正当性判別を行うため，その時間が実用的な範囲内であることが重要である．我々はミドルウェアの検索動作にかかる時間を計測し，その結果を元に提案手法の利用可能範囲についての考察を行うことにした．なお，本稿では実用的な検索時間として 4 秒という値を用いる[2]．

ミドルウェアの検索動作にかかる時間を測定した．測定において**表 1** の各ホストを利用した．ホスト X でミドルウェアを，ホスト A ではリポジトリを，ホスト

表 1 マシンスペック
Table 1 Machine specification

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

表 2 ミドルウェアの動作時間
Table 2 Individual action proccesing time

	平均 [ms]	標準偏差 [ms]
検索要求の構文解析	2.95	0.05
タイムスタンプ比較	8.43	5.04
コンタクト ID 情報取得	76.2	2.7

B では OpenID Provider を動作させた．リポジトリの実装には Perl を利用しているので，ホスト A では動作の高速化を図るために Web サーバの Apache 内で mod_perl を用いた．各ホストは同一データリンク上に存在し，各ホスト間の RTT は 1[ms] 以下であった．

ミドルウェアの各動作にかかった時間は**表 2** の通りである．計測はそれぞれ 100 回行った．リポジトリに更新があるかどうかでリポジトリへのアクセス時間が大きく異なる（8.43[ms] と 76.2[ms]）こともわかる．

以降では，キャッシュ内検索失敗時にかかる時間についての考察を行う．キャッシュ内検索が失敗した場合には多くのリポジトリにアクセスを行う必要があり，検索時間が増加する．ツールの利用における各リポジトリの更新頻度を見積り，その際の検索時間を考慮する必要がある．

各ユーザが独立に自身のリポジトリ更新を行うとすると，キャッシュの全更新から時間 T が経過した時の更新リポジトリ数はポアソン分布に従う．T は，実際の利用においてはキャッシュの全更新から次のメッセージ到着までの時間を意味する．受信制御リスト中の全リポジトリ数を N_{all}，更新リポジトリ数を x とするとその分布 $f(x)$ は式 1 で表せる．式 1 において $P(T)$ はある時間 T が経過した時のあるリポジトリの更新確率であり，p を単位時間あたりの更新確率とすると式 2 のように表せる．ポアソン分布の性質より，更新リポジトリの期待値は $N_{all}P(T)$ となり，非更新リポジトリ数の期待値は $N_{all}(1-P(T))$ となる．

$$f(x) = \frac{e^{-N_{all}P(T)}(N_{all}P(T))^x}{x!} \quad (1)$$

$$P(T) = \sum_{k=1}^{T}(1-p)^{k-1}p \quad (2)$$

ユーザの利用モデルを仮定し，キャッシュ内検索失敗時の検索完了時間の見積りを行う．ここでは，新しいコンタクト ID の利用開始後 1 週間程度するとスパムが届き始めるという報告[7] を考慮し，ユーザが 1 週間に 1 回あるいは 1 日に 1 回，自身のコンタクト ID を変更するという利用モデルを仮定する．前者はスパムが届き始めるとそのコンタクト ID を変更するというモデル，後者はスパムが届くこと自体を避けるために頻繁にコンタクト ID を変更するというモデルである．それぞれ p は $1/(60\times24\times7) = 1/10080$, $1/(60\times24) = 1440$ である．T には，60 分という値を用いた．

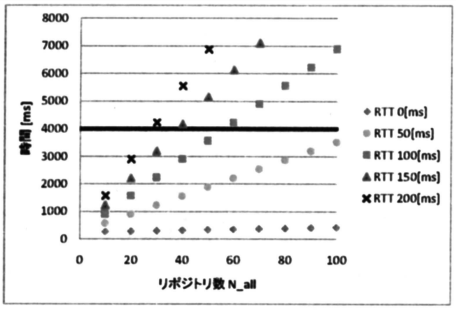

図 6 1 週間に 1 回のコンタクト ID 変更環境における検索時間の変化
Fig. 6 Change of searching time under the model where each user changes his/her Contact IDs once a week

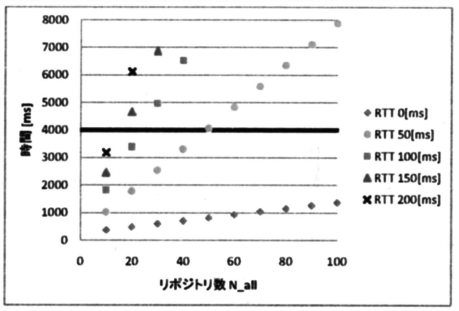

図 7 1 日に 1 回のコンタクト ID 変更環境における検索時間の変化
Fig. 7 Change of searching time under the model where each user changes his/her Contact IDs once a day

見積り結果は**図 6** および**図 7** である．ある N_{all} の内，$N_{all}P(60)$ が更新リポジトリ，$N_{all}(1-P(60))$ が非更新リポジトリであるとして，ミドルウェアがそれらへのアクセスを完了するまでの時間を図示したものである．本稿で目的とする時間を明確にするため，4 秒を表す線を強調して示している．提案手法の利用においてはインターネット上にリポジトリが分散することが想定されるため，tc[*1] を用いて表 1 のホスト X - ホスト A 間およびホスト X - ホスト B 間に仮想的に遅延を挿入して表 2 と同様の計測を行い，図 6，図 7 にはその結果を用いている．系列は，ここで挿入した遅延ごとに分けている．また，並列アクセスによる時間

*1 http://lartc.org/

短縮効果やスレッド生成のオーバヘッドもミドルウェアの動作に影響を及ぼす．それらについても別途計測し，図6，図7に反映している．

図6より1週間に1回の頻度で変更を行う環境では，RTTが50[ms]であれば100程度，RTTが100[ms]の環境であれば60程度，RTTが200[ms]であっても30程度のN_{all}に対応できることがわかる．図7より1日に1回の変更を行う環境では，RTTが50[ms]であれば50程度までのN_{all}に対応できるが，RTTが100[ms]を超えるような環境では10〜20程度のN_{all}にしか対応できないことがわかる．

N_{all}は1つのコンタクトIDを用いてコミュニケーションを行う相手ユーザ数であり，数十程度あれば現在のコミュニケーションツールの利用に問題がないと思われる．そのため，各ユーザのコンタクトID変更頻度が1週間に1回であるモデル，つまりスパムが届き始めたらコンタクトIDを変更するといった利用モデルにおいては提案手法は十分実用的であると考えられる．しかしながら，コンタクトIDの変更頻度が1日に1回程度になると，実用的な時間内で検索が完了できるN_{all}の数は少なくなり，インターネット上におけるリポジトリの配置によっては10程度までしか対応できなくなる．そのような利用モデルにおいては提案手法は実用的でなくなるため，検索時間短縮に向けた更なる改善などが必要となる．

6. 結　　論

本稿では，我々が行ってきた提案の実装及び評価を行った．我々は，コンタクトIDの公開先管理場所であるリポジトリを実装し，メッセージの正当性判別のためにリポジトリから情報取得を行うミドルウェアを実装した．ミドルウェアによってコンタクトID検索機能と各ツールの分離が実現された．

また，ミドルウェアを利用して送信者判別を行うツールの一例としてThunderbirdのアドオンを作成した．本アドオンは，ユーザが選択した電子メールに関する検索要求メッセージを構築してミドルウェアに渡し，その検索結果をユーザに提示するものであった．本アドオンの実装におけるコード量の比較から，実装における検索機能の分離により，提案手法を利用するツールの作成が容易であると結論付けた．

我々はミドルウェアの動作時間計測に基づいた提案手法の実用性評価を行った．新しいコンタクトIDの利用開始後1週間程度するとスパムが届き始めるという報告を考慮してユーザの利用モデルを想定し，各モデルにおける提案手法の実用性について考察を行った．その結果，1週間に1回の頻度でコンタクトIDの更新が行われるモデルにおいては，RTTが50[ms]の範囲であれば100程度，RTTが100[ms]の環境であれば60程度，RTTが200[ms]であっても30程度までのリポジトリの検索が実用的な時間内に可能であることがわかった．これらは1つのコンタクトIDを用いてコミュニケーションを行っている相手の数としては妥当であり，このモデルでは提案手法は実用的であることがわかった．しかしながら，変更頻度が1日に1回であるモデルでは，RTTによっては実用的な時間内に検索が完了するリポジトリ数が10〜20程度に減少することがわかり，検索時間短縮に向けた改良が必要になるということがわかった．

今後の課題としては検索時間短縮に向けたキャッシュ更新手法の検討や，提案手法および実装のユーザビリティ面での評価などが挙げられる．

参 考 文 献

1) Cerf, V.G.: Spam, spim, and spit, *Commun. ACM*, Vol.48, No.4, pp.39–43 (2005).
2) Culotta, A. et al.: A study on tolerable waiting time: how long are Web users willing to wait?, *Behaviour & Information Technology*, Vol.23, pp.153 – 163 (2004).
3) Fallows, D.: Spam: How it is hurting email and degrading life on the Internet, *Pew Internet and American Life Project*, pp.2008–01 (2003).
4) Falomi, M. et al.: Simulation and Optimization of SPIT Detection Frameworks, *Proc. of GLOBECOM2007. IEEE*, pp. 2156–2161 (2007).
5) Hershkop, S. et al.: Combining email models for false positive reduction, *Proc. of KDD2005. ACM* pp.98–107 (2005).
6) Liu, Z. et al.: Detecting and filtering instant messaging spam - a global and personalized approach, *Proc. of NPSec2005* , pp.19–24 (2005).
7) Prince, M. et al.: Understanding How Spammers Steal Your E-Mail Address: An Analysis of the First Six Months of Data from Project Honey Pot, *Proc. of CEAS 2005* (2005).
8) Recordon, D. et al.: OpenID 2.0: a platform for user-centric identity management, *Proc. of DIM2006. ACM*, pp.11–16 (2006).
9) Vaughan-Nichols, S.: Saving private e-mail, *Spectrum, IEEE*, Vol. 40, No. 8, pp. 40–44 (2003).
10) 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，情処研報，2008-GN-67，pp.73–78 (2008).
11) 阪本裕介，中山雅哉：IDに基づく受信制御の自律管理方式とその効果，第7回情報科学技術フォーラム (2008).
12) 阪本裕介，中山雅哉：送信者IDの動的把握に基づくメッセージ受信制御方式，インターネットコンファレンス2008，pp.57–66 (2008).