6J-03　エージェントを用いた柔軟で軽量なワンストップサービスシステム*

江藤 秀一　宇田 隆哉　岩田 善行　重野 寛　松下 温 †
慶應義塾大学理工学部‡

1　はじめに

近年、ネットワーク化の進歩は目覚ましく、日々の生活をより快適にするために、様々なサービスがコンピュータネットワークを通して遠隔地の利用者に瞬時に提供されるようになってきている。日本でも行政サービス向上の観点からネットワーク化が推進され、ワンストップサービスが注目されている [1]。

情報ネットワークを利用し、一つの窓口で一括して行政手続きを行うことのできる「ワンストップ行政サービス」は、国民の利便性の向上や効率的な行政の実現のためにもきわめて重要である。「ワンストップ行政サービス」[2] は、住居地にとらわれず、自分にもっとも都合のよい窓口一ヵ所（郵便局 24000ヵ所、コンビニエンスストア 48000ヵ所、あるいは行政書士事務所 36000ヵ所など）に設置された情報端末により、住民票の写し、登記簿謄抄本、課税証明、所得証明、印鑑証明などの申請手続きや交付が可能となる行政サービスで、わが国以外でも同様の構想が進められている。具体的なプロジェクトとして、いろいろなものが挙げられるが、本研究では国民がもっとも直接的に恩恵を受けるであろう住所変更ワンストップサービスに着眼した。

しかし、この構想を実現する際、いくつかの問題点が生じる。その中でも、個人情報保護が最大の課題である。低コストに利便性を得ようとした場合、サービス全体を通しての安全性が甘くなり、利用者情報の漏洩や改竄に繋がる恐れがある。今日では、共通鍵暗号方式や公開鍵暗号方式のように、利用者間で交換されるデータを暗号化技術で保護しようとする仕組みがスタンダードとして確立されつつある。しかし、暗号 [3] を使用して安全性を追求すればするほど、システム全体の負荷は大きくなりネットワークのトラフィックは増加する。そしてこれらはシステム運用のコストを引き上げ、コストの増加はサービスの利用料金に反映されてくる。そこで、本研究では、このような大規模ネットワークでの情報の管理において、エージェントを利用し、セキュリティが必要な箇所に必要最小限の負荷でデータを保護する、低コストで高いセキュリティを維持できるシステムの構築を目指す [4][5]。

2　引越し手続きの現状

引っ越しの諸手続きを以下の表 1 に表す。

表 1: 引っ越しの手続き

手続き	必用なもの
転出届け	印鑑
転入届け	印鑑、転出証明書
印鑑登録	印鑑登録証、登録印
国民健康保険	印鑑、国民健康保険証
児童手当て	印鑑、所得証明書
国民年金	印鑑、国民年金手帳
運転免許証	免許証、住民票
転校手続き	印鑑、在学証明書、教科書給与証明書

引っ越しの際に必要な手続きは、意外に多くわずらわしいものである。その点、ワンストップサービスは、一つの窓口で一括して行政手続きを行うことにより我々国民の利便性を大きく向上させるものである。

3　エージェントによる通信システム

3.1　モバイルエージェントの利点と問題点

本システムではエージェント [7] を使用し、ワンストップサービスを実現している。モバイルエージェントは、プログラムコードを保持したままホスト間を移動できる（マルチホップという）ため、ホストを次々と経由して処理をする場合にトラフィックの軽減を図れるという利点を持つ。また、エージェントがインテリジェントである場合には、各ホストにおいて必要な処理のみを自動的に行うことができ、エージェント固有

*Flexible and Inexpensive Onestop Service System with Agents
†Shuichi Eto, Ryuya Uda, Yoshiyuki Iwata, Hiroshi Shigeno, Yutaka Matsushita
‡Faculty of Science and Technology, Keio University

の実装を各ホストにおいて予め実装することなく、セキュリティ面の向上を図れる。しかし、問題点としてエージェント通信路の安全性・オーナシップ・実行権限・アクセス権利・ライフサイクル・コントロールなどが挙げられる。

エージェントが移動する通信路が安全でない場合、公開ネットワーク上で運営されるサービスにシステムを用いることは非常に危険である。単に暗号化により安全性を確保することはシステムの負荷やネットワークのトラフィックを増大させることにもなり、モバイル機器との連携や大規模なシステムでは、従来方式での安全性の確保は困難であると言える。また、オーナーシップとは各エージェントの持ち主や出身となるドメインを特定することであり、オーナシップの管理は実行権限やアクセス管理の基盤となる。オーナシップの特定は各々のエージェントの認証によって得られる。しかし、モバイルエージェントの認証は非常に難しい性質を持っている。

3.2 マルチホップ時の認証

図1において、マルチホップ時の認証について概略を説明する。

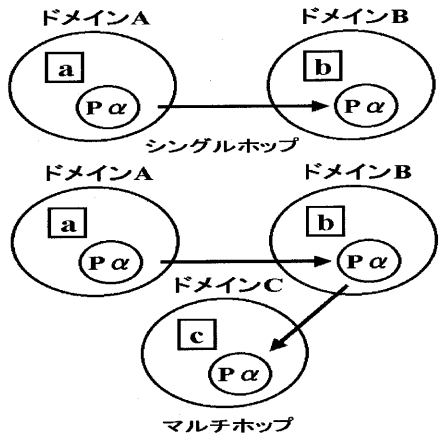

図 1: エージェントの認証

通常、モバイルエージェントにおける認証は、ドメインの認証に基づくものである。図1のシングルホップの例では、ドメインAから発信されたエージェントPαは、ドメインBを経てドメインCへ到着する。ドメインBは、ドメインAとドメイン間での認証を行うため、ドメインAから発信されたエージェントはドメインAのものであることを認証できる。しかし、マルチホップでこのエージェントを受け取るドメインCは、ドメインBを認証することは出来るが、通信を行わないドメインAを認証することは出来ない。そこで、ドメインBから来たエージェントが、本当にドメインAから発信されたものであるかどうかは確認することができない。

本研究では、このようなエージェント通信路の安全性や認証に着眼しそれを解決する方法として多段ハッシュ方式を提案する。

4 多段ハッシュ方式による高速な認証方法

4.1 多段ハッシュ方式の概要

本研究では、MD5によるハッシュを多段に用いてエージェントがマルチホップする場合の認証を行うシステムを提案する。これにより、公開鍵暗号方式を使わずに高速な認証を行うことが可能となるため、負荷の高い状態においては非常に有効であると考えられる。

ここでは、差分方式によってエージェントの処理を行うことを前提としている。図2に、本システムにおける差分方式の簡単な例を示す。

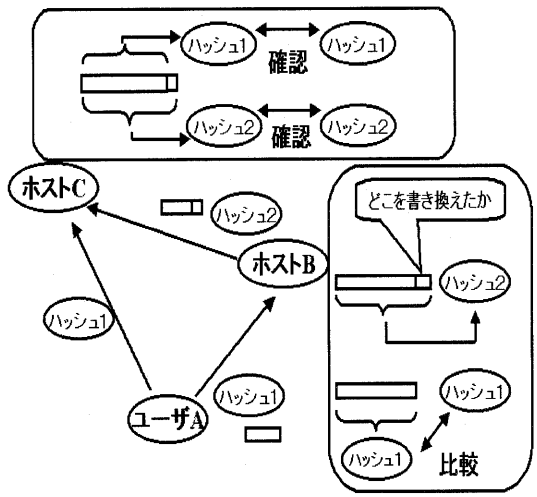

図 2: 差分方式によるエージェントの認証

図2では、ユーザAは、あるデータをホストBに処理してもらった後に、ホストCにも処理をしてもらいたいと思っている。そこで、ユーザAは、自分自身が発信するエージェントのハッシュをブロードキャストでホストBとホストCに送信する。ホストBは、ユーザAから受け取ったデータ部は破壊せずに、エージェントの後部に、どのデータをどのように処理したかのプログラム部を追加し、全体のハッシュをホストCに送信する。ホストCは、ユーザAとホストBからそれぞれのハッシュを受け取るため、受け取ったエージェ

ントのどこに不正があったかを同様のハッシュを計算することにより判断できる。

ここで、4 者間をエージェントがマルチホップする場合には、図 3 のようになる。

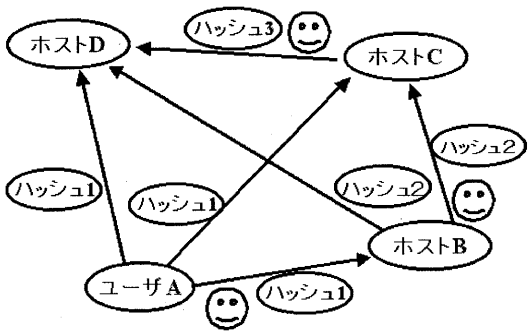

図 3: エージェントのマルチホップ

このように、エージェントが複数のホストをマルチホップしていく場合にはハッシュが多段となるが、これによって不正ホストの発見が可能となり、システム全体の早期リカバリにも繋がる。モバイルエージェントが処理をする場合には、本来、エージェント自体が内包するデータ部を更新するが、今回の差分方式では、受信データ部を更新せずに処理内容をログとして付加することでハッシュをとることを可能にしている。この差分方式が本研究の最大の特徴である。

5 システムの実装

5.1 システムとその概要

本システムの概要を図 4 に示す。

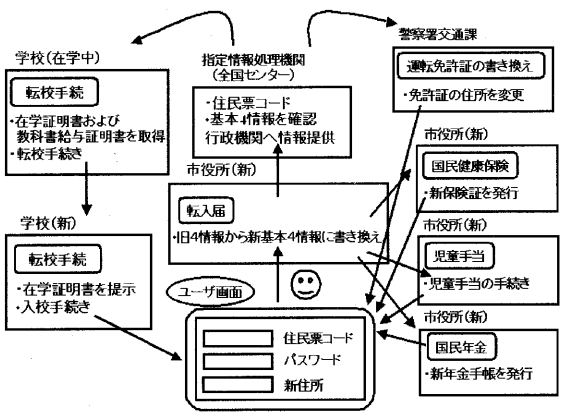

図 4: システムの概要

本システムは、全国共通の相互に重複しない住民表コード・パスワード・新住所をユーザがユーザ画面に入力するだけで、エージェントが引っ越しに関する諸手続を行うものである。エージェントは、新市役所において住民基本台帳を新しいものに書き換え、さらに国の各行政機関に行く際には、指定情報処理機関 (全国センター) よりその中の新 4 情報と住民票コードを受取り各機関をまわることとなる。また、ワンストップサービスでは、IC カードの採用が予定されている住民基本台帳カードを使った特例がある。転入届の特例により、このカードを添えることで転出証明書が不要となる。よって郵便で予め転出届を出せば、窓口に出向く通常の転出届は不要となる。このように、ワンストップサービスでは、一度の手続で今までの面倒だった処理を簡略に効率よく行うことができる。

5.2 多段ハッシュの内容

前章で提案した多段ハッシュ方式を用いるにあたり、各機関における本システムでのそのハッシュの内容を下の図 5 に示す。

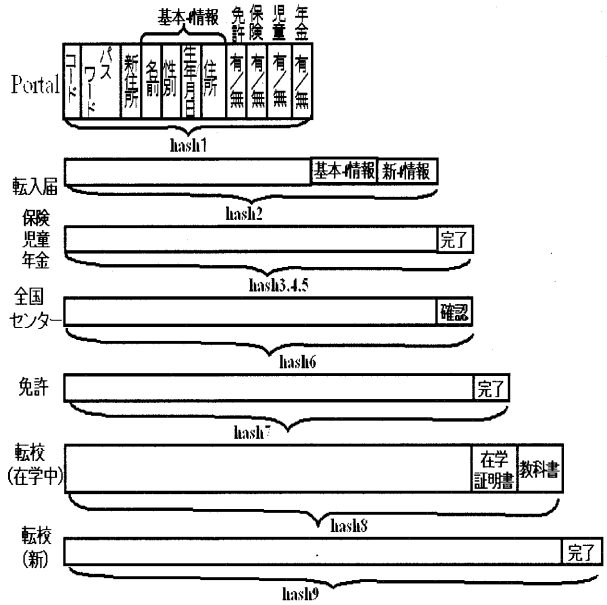

図 5: 各サーバーとそのハッシュ

以上のようにしてハッシュをとることにより, マルチホップの認証という問題に対して、任意のホストがデータの改竄を行い、改竄を他のホストが行ったように見せかける行為を防止することができる。

5.3 開発環境

今回実装で使用したのは、FIPA 準拠である Java エージェント開発環境の JADE(Java Agent Development Framework) である [6]。JADE は、セキュリティ機能を内包しておらず、ACL のメッセージ通信およびその他のデータ転送においても安全性を確保していな

い。今回のような様々な機関と協力してネットワーク上でサービスなどを行うには、セキュリティは必要不可欠であり、安全性の確保できないプラットフォームは問題がある。

上記をふまえ、本研究では、JADEプラットフォームにACLメッセージ交換経路を暗号化するモジュールを実装した。暗号化通信に際しては、各種暗号化方式を用いて実装し、またC言語での実装も行った。

5.4 JADEに実装した暗号化方式

今回、JADEに実装した、暗号化通信に用いることの出来る暗号化方式は、DES、CAST-256、Twofishによる共通鍵暗号方式である。また、公開鍵暗号方式に関してはRSAを実装した。

6 評価

[1]JADE上での暗号化通信における性能評価
[2]JADEとC++での暗号化通信における性能評価
[3] 多段ハッシュ方式の評価

6.1 JADE上での暗号化通信評価

JADEに実装した暗号化通信モジュールを用いて通信を行い、各暗号化モジュールの性能を測定した。

(1) 送信側がメッセージを暗号化して送信
(2) 受信側がメッセージを復号化して確認通知を送信
(3) 送信側が確認応答を受け取り通信終了

以上に掛かる時間を測定するために、(1)〜(3)を10回繰り返し行い、平均時間を算出した。DES、CAST-256、Twofishおよび暗号化無しの場合において、ファイルサイズを変化させ測定した。なお、IIOPによるシークのオーバーヘッドは、別途測定して省いており、転送のみに掛かる時間の性能評価となっている。この評価を図6に示す。

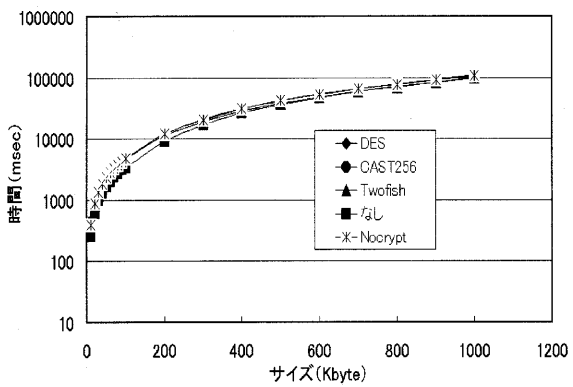

図 6: JADEでの暗号化性能比較

6.2 C++との暗号化通信性能評価

暗号化および復号化は、CPUの能力を非常に必要とするため、Javaによるシステム性能全体の低下が予想される。そのため、C言語により、同様の暗号化モジュールを用いた通信システムを実装し、JADEで実装した場合との性能差を測定した。現在、Microsoft Visual C++ 6.0を用いて、DES、CAST-256、Blowfish、MD5, Nocryptの暗号化通信システムについての評価を図7に示す。

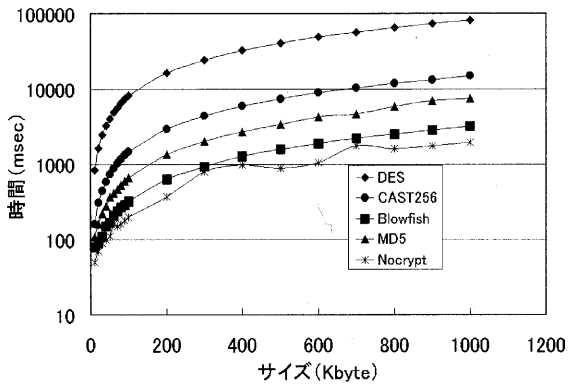

図 7: C++での暗号化性能比較

各暗号モジュールごとの送信サイズによる付加の増加の違いを考察する．図7をみると，これらの各モジュールはファイルサイズの増加に対してほぼ比例して時間が増加していることが分かる．また，ファイルサイズが小さいときには時間もとても減少している．しかし，例えば図14をみても分かるようにRSAの場合ファイルサイズがごく小さいときにも共通鍵暗号方式やハッシュ関数ほど，時間は減少しない．これは，RSAが暗号化以外のオーバーヘッドが大きいためであると考えられる．また，各暗号モジュールによりかなりの速度の違いが現れているが，これらはモジュールがC++に最適化されているかどうかも関係している．

6.3 多段ハッシュ方式の評価

この実装に関しては、マルチホップする場合とシングルホップする場合のトラフィックの差、また暗号化通信時のシステム負荷によるパフォーマンスの差を比較評価した。実装するシステムの概略を以下に示す。多段ハッシュシステムに関しては，前章のシステムと概要で述べた図4と同様である．

各経路において，それぞれそのバックボーンのサーバは太い回線とし．各サーバーと発信元ホストとの回線は細いものとする．本研究においてはエージェントを用いることにより、ネットワークトラフィックの軽

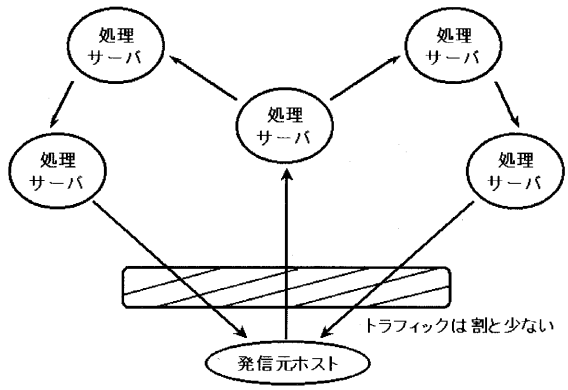

図 8: 2 分型

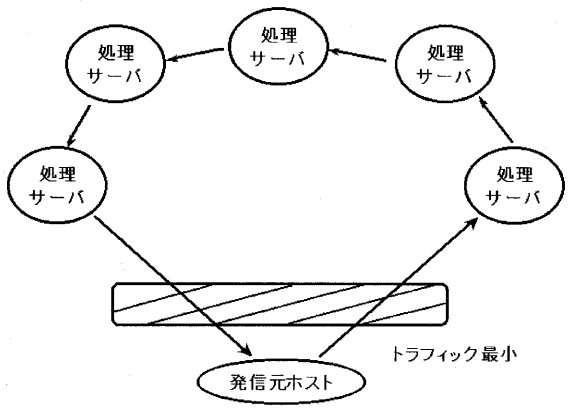

図 9: 一周型

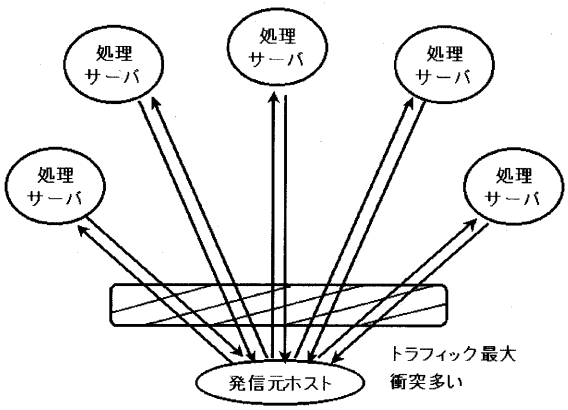

図 10: 直接同時接続型

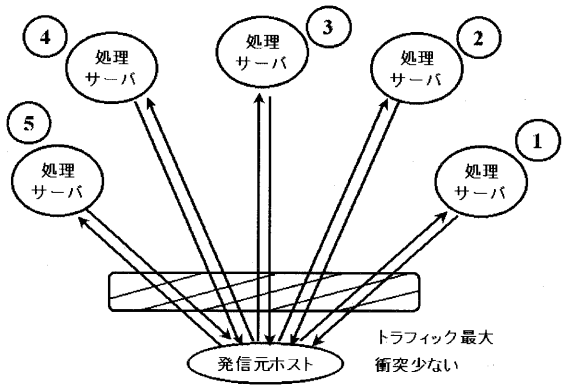

図 11: 直接順次接続型

減も図れると期待されることから、通信負荷の面からの評価も行った。ここで述べるトラフィックは図 8〜図 11 の斜線長方形内を通った転送量，つまりホストと各サーバーの情報転送量のことである．

図 12 をみると分かるように，今回評価を行った 5 つの経路のうち，最も応答が早かったものは一周型である．その理由として，一周型は絶対に衝突をおこすことがなく，細い回線を通る際には他の経路に比べトラフィックを少なく最小におさえることができることがあげられる．これらは，徐々に負荷を大きくしていった際も同様の結果を示している．

そして，今回実装したワンストップサービスに関しては，エージェントは図 4 のようにいくつかのサーバーに分割して進むことから，たくさんのサーバーに処理を投げる分だけ時間をロスするため，一周型や，ニ分型よりは遅くなる．しかし，ワンストップサービスは 8 つのサーバーを回っているにも関わらず，5 つのサーバーを回る同時型とほぼ同じ値を示していることから十分な評価を得たと考えられる．

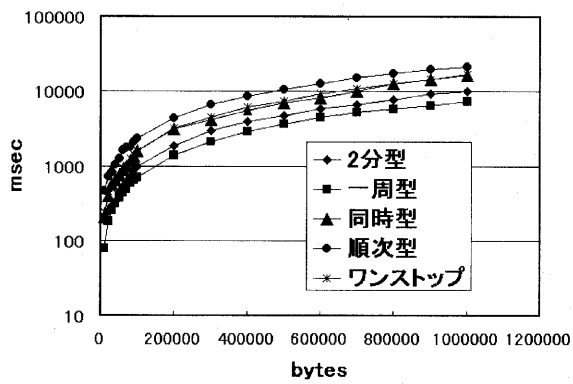

図 12: 性能評価 (速度)

さらに，トラフィックの観点からワンストップサービスをみてみる．図13をみると，一周型や二分型よりもトラフィックが多くなっていることが分かる．これは，ワンストップサービスが一周型や二分型に比べてホストに戻る経路が多いことに起因する．一周型の場合はホストに戻ってくる経路は当然1つ，二分型の場合には2つである．しかし，ワンストップサービスではその経路は5個にもなることからこのような結果になったことと思われる．しかし，それでも直接同時型・順次型よりもトラフィックは少ないことから，これも評価に値するといえるだろう．

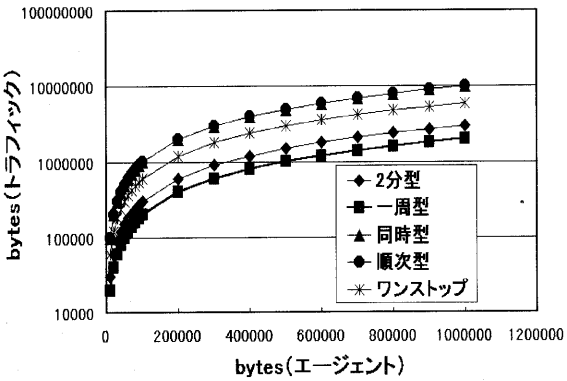

図 13: 性能評価 (トラフィック)

また，ハッシュ関数であるMD5を用いて暗号化したものと公開鍵暗号方式のRSAを用いて暗号化したものを各経路において比べた図14〜図18をみてみると，どの経路を回った際も，約100Kbyteのところまではハッシュ関数であるMD5が公開鍵暗号方式のRSAを上回っており，100Kを越えるとRSAがMD5を逆転した形となっていることが分かる．これには，RSAの鍵の生成時間が関係しているものと思われる．RSAは，その鍵の生成に一定時間を必要とする．それは，暗号化のサイズの大小に大きく影響を受けず，サイズが大きくなってもその暗号化の時間はあまり増加しない．一方，ハッシュはデータを暗号化する部分の処理以外に要する時間が少ない．したがって，ある一定のサイズのものを暗号化するまではハッシュ関数が向いており，その値を越えると公開鍵暗号方式が向いているということになる．したがって，本研究のように処理単位におけるサイズが非常に小さなデータを扱うサービスにおいて，ハッシュ多段方式による認証は非常に有効であると考えられる．

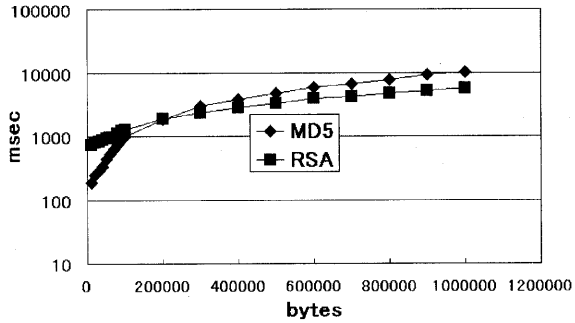

図 14: 二分型

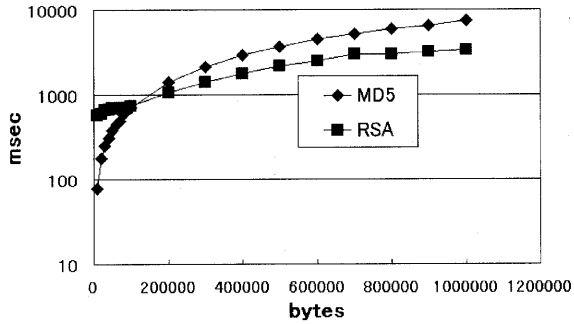

図 15: 一周型

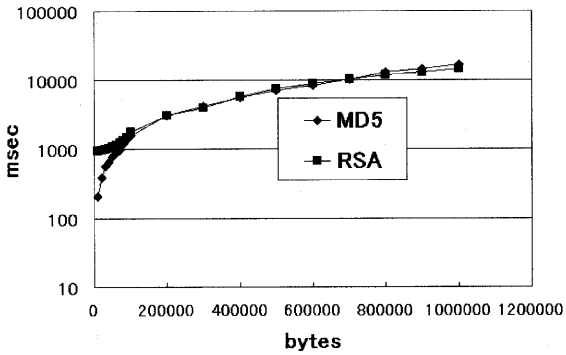

図 16: 直接同時接続型

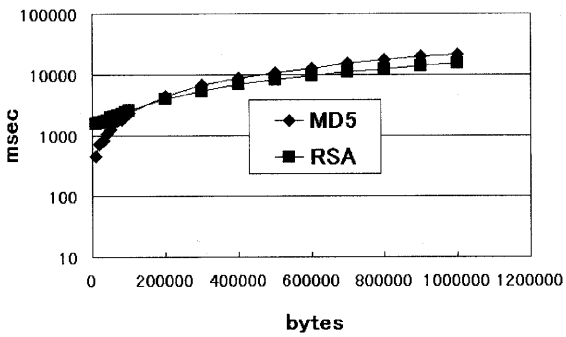

図 17: 直接順次接続型

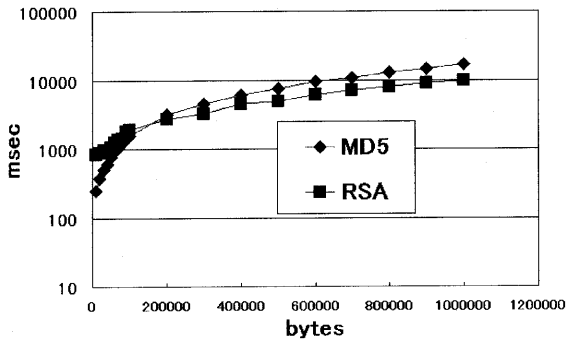

図 18: ワンストップ型

7 動作実験環境

サーバ（メッセージ受信側）
CPU: Pentium III 600MHz
RAM: 128MB
OS: Windows NT 4.0 Workstation

クライアント（メッセージ送信側）
CPU: Pentium III 600MHz
RAM: 128MB
OS: Windows NT 4.0 Workstation

ネットワーク速度
100Mbps Ethernet（非スイッチング Hub）

8 まとめ

本研究では、ワンストップサービスといった大規模ネットワークでの情報の管理において、エージェントを利用し、低コストで高いセキュリティを維持できるシステムの構築・実装を行いそれを評価した。その際、エージェントの通信路の安全性や認証の問題を解決する方法として、多段ハッシュ方式を提案・実装した。このような複数のホストで連続的に処理を行う必要のあるサービスにおいて、本システムは非常に有効であることが確認できた。今後の課題として、モバイルエージェントのマルチホップにおける実行権限・アクセス権利・ライフサイクル・コントロール等の問題を検討していきたい。

参考文献

[1] 大橋 有弘, “改正住民基本台帳法の意義と住民基本台帳ネットワーク構築への課題” 行政 & ADP 2000 年 1 月号

[2] http://www.mha.go.jp/news/970617.html

[3] http://www.cgc.co.jp/text/security6.html

[4] 佐々木 良一, “インターネット時代の情報セキュリティ”, 共立出版株式会社, 2000

[5] http://www.ibm.co.jp/developerworks/java/000915/j_jw-0428-security.html

[6] http://Fabio Bellifemine, Giovanni Tiziana, “JADE PROGRAMMER'S GUIDE”, 2000 CSELT S.p.A.

[7] Prithviraj Dasgupta, Nitya Narasimhan, Louise E. Moser, P.M. Melliar-Smith, “MAgNET:Mobile Agents for Networked Electronic Trading” , IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING VOL.11, NO.4, JULY/AUGUST 1999,pp.509-525