

メンバ間公平性保証方式における ハードウェアを用いた偽証防止方法

谷口幸久[†] 石原進[‡] 水野忠則[‡]

[†] 静岡大学大学院情報学研究科 [‡] 静岡大学情報学部

インターネットの普及に伴い、ネットワークゲームやオークションといったネットワークアプリケーションの需要が高まっている。このようなネットワークアプリケーションの遅延差による不平等性を解消するため、メンバ間公平性保証方式 ICEGEM (Impartial Communication Environment for GamE Members) が提案されている。しかしながら、この方式にはクライアントが虚偽の応答時間を返答した時これを発見できないという問題が存在する。そこで、本論ではこの問題を解決するため、専用ハードウェア SIC (Secure Ice Card) を使用した偽証防止方法を提案する。クライアントはネットワークインターフェースカード (NIC) として SIC を利用するが、実際は SIC は到着したパケットをアプリケーション層で監視し、ICEGEM による順序制御を行うパケットに対しては、クライアントプログラムに代わってクライアントの応答時間を測定する。

Hardware-based method of preventing from perjury for ICEGEM

Yukihisa Taniguchi[†] Susumu Ishihara[‡] Tadanori Mizuno[‡]

[†] Graduate School of Information, Shizuoka University

[‡] Faculty of Information, Shizuoka University

With the spread of the Internet, the demand of network applications like network games and auctions are increasing. In order to cancel the unequal nature by the delay difference between hosts, we have been proposed a fairness guarantee method ICEGEM (Impartial Communication Environment for GamE Members). However, in this method, when a client reports false reaction time to the server, the server cannot discover this client's cheating. In order to solve this problem, we propose a method of preventing for perjuries which utilize an exclusive hardware SIC (Secure Ice Card) in this paper. Clients use SIC as a Network Interface Card. SIC supervises arrived packets on application layer. SIC measures the reaction time of the client instead of client programs for the packets requiring order control by ICEGEM.

1 はじめに

インターネット人口は増えつつあり、オンラインゲームなどのインターネットを用いたアプリケーションの需要はますます高まると考えられる。ネットワークゲームの中にはユーザ登録数が数十万人にも及ぶものも登場している。また、ネットワークゲームへの参加方法は、アナログ電話回線によるものから、ADSL, FTTH などのより高速な方法へと変化している。インターネット環境においてホスト間の遅延は、このような端末へのラスト1ホップの接続速度の違いに加え、ホップ数の違いや遅延の揺らぎなどにより一定ではない。このため、ネットワーク対戦ゲームやネットワークオークションの入札などでは、サーバからクライアントへ送られたメッセージの、クライアント側の応答時間と、サーバ側でのクライアントからのメッセージ到着時刻の間にずれが生じる。例えば早押しクイズでは、サーバからの問題提示メッセージに十分早く応答したにもかかわらず、ネットワーク上の遅延により、サーバへ応答メッセージの到着が遅れたために、サーバ側では処理を受け付けられないという問題がある。この不公平性を解消するための機構として、メンバ間公平性保証方式

(ICEGEM: Impartial Communication Environment for GamE Members) が提案されている [1]。この方式は、ユーザの応答時間のみに基づいて順序の制御を行うことにより公平性を保証する。しかしながら、この方式はクライアントプログラム自身がクライアントの応答時間の測定を行うため、クライアントプログラムが虚偽の情報をサーバへ送信 (偽証) した際に改竄を検知できないという問題が存在する。この問題に対し、筆者らは、[2] において、クライアントの使用しているプログラムが正規のプログラムであるかどうかをサーバが検証する、鍵付きハッシュ関数による正当性検証の方法を提案した。しかしながら、この方式においてはクライアントプログラムが自分自身を証明するための秘密情報を全て知り得てしまう。すなわち、クライアントは改竄したプログラムを使用しているにもかかわらず、正規プログラムのハッシュ値をサーバに送信することにより、サーバによるチェックを回避することが可能である。このことより、クライアント自らが応答時間を測定する方式では偽証の防止は困難であると考える。よって、本稿においては、クライアントではなく、信頼できるハードウェアを用いることによって

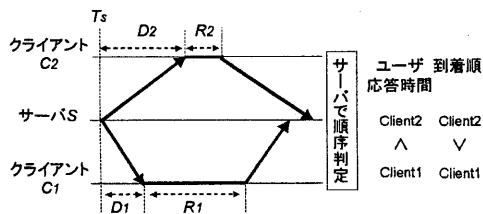
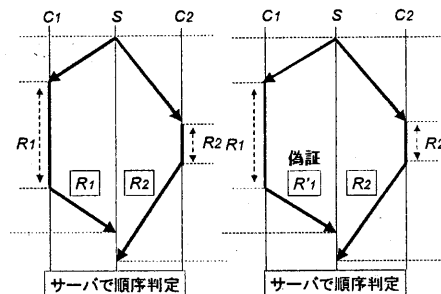


図 1 到着時間とユーザ応答時間の逆転



(a)共に正規クライアントの場合 (b)C1が改竄された場合

クライアントの偽証を防止する方式を提案する。

図 2 クライアントからのメッセージの改竄

2 メンバ間公平性保証方式 ICEGEM

2.1 応答時間に基づく順序制御

サーバからのメッセージに対し複数のクライアントが応答するアプリケーションにおいて、各ホスト間の遅延が一定ではないインターネット環境では、各クライアントに与えられる環境は公平ではない。例えば、ネットワークゲーム等において、サーバへクライアントからの反応が到着する順序でサーバが順序判定を行った場合、ホスト間の遅延差により実際の反応順序と到着順序が入れ替わる場合があり、公平性が保たれない。図 1 において、サーバからのメッセージ送信時刻を T_S 、サーバ-C₁ 間の遅延を D_1 、サーバ-C₂ 間の遅延を D_2 、C₁ のユーザ応答時間を R_1 、C₂ のユーザ応答時間を R_2 とする。簡単のため、遅延は変動しないものとする。サーバからのメッセージは C₁ と C₂ 共に時刻 T_S に送信されている。C₁ と C₂ にメッセージが到着する時刻はそれぞれ $T_S + D_1$ 、 $T_S + D_2$ となる。その後クライアントからの反応メッセージがサーバへ届く時刻は、それぞれ $T_S + R_1 + 2D_1$ 、 $T_S + R_2 + 2D_2$ となる。ここで、 $D_1 < D_2$ 、 $R_1 > R_2$ 、かつ $2(D_2 - D_1) > (R_1 - R_2)$ とすると、 $T_S + R_1 + 2D_1 < T_S + R_2 + 2D_2$ が成立し、 $R_1 > R_2$ にもかかわらず、サーバへは C₁ のパケットが先に到着する。そのため、サーバは $R_1 < R_2$ と判断するため、 D_1 と D_2 の差によって、C₁ と C₂ の間に不公平性が生じている。

上記の問題を解決するため、ICEGEM では各クライアントプログラム C_n が自分自身の応答時間 R_n を測定し、応答メッセージに付加する。以上によって、サーバが各 R_n を知ることが可能となり、クライアントの応答時間に基づく順序制御を行うことが可能となる。

2.2 ICEGEM の応用範囲

ICEGEM は、 R_n によって勝敗が決まるアプリケーション全般、例えばネットワークゲームやオンラインカジノ、オークションに適用が可能である。しかしながら、金銭の授受が行われるオークションやオンラインカジノに対して ICEGEM を適用しようとした場合、不正の発生を防止する必要がある。

2.3 クライアントによる偽証

ICEGEM では、クライアントプログラム自身が R_n を測定しサーバに送信するが、あるクライアントプログラム C_k が偽の応答時間 R'_k を送信した場合（偽証）に、サーバは偽証を検出できないという問題が存在する。具体例として、図 2 において、 $R_1 > R_2$ 、 $R'_1 < R_2$ となる場合を考える。サーバメッセージに対して C₁ は R_1 後に応答したにもかかわらず、C₁ は、図 2(b) のように応答時間 R'_1 をサーバへ送信する。このとき、サーバは偽りである R'_1 というメッセージを信用し、C₂ の応答時間 R_2 と比較する。このとき、サーバは $R_1 \neq R'_1$ を検出できず、 $R_1 < R_2$ とみなす。よって、このような偽証が発生した際、サーバが偽証を検出できないため、正しい順序制御をサーバが行うことができない。

2.4 偽証メッセージの発生理由

サーバへクライアントからの偽証メッセージが到着する理由として、以下の 3 点が考えられる。

i). 経路上における第三者による情報の改竄

クライアントからの応答メッセージがサーバへ送信された後に、経路上においてパケット内容が改竄されることにより、偽証が発生する。

ii). あるクライアントの実装ミス

クライアントの OS などの環境によって使用する

プログラムが異なるものとする。このとき、ある C_k で実装ミスにより用いられた場合、偽証が発生する。

iii). クライアントプログラムの変更

ファイルシステムの異常や、サーバからのダウンロードの失敗などによりプログラムが破損し、正しく動作しなかった場合、もしくは、悪意ある第三者が C_k の改竄を行った場合、偽証が発生する。

上記の問題のうち i) は、既存の暗号化方式、例えば IPsec によるパケットレベルでの暗号化や、応答時間に電子署名を施すといった方法で防止可能である。ii) に対しては、サーバ自身が動作確認を行ったクライアントプログラムをセッション開始時にクライアントに対して送信することによって防止が可能である。iii) は、クライアントプログラムが正当であるか否かを定期的にサーバが検証することにより防止可能である。

3 鍵付きハッシュ関数による正当性検証

ICEGEM における偽証の問題を解決するためには、不特定多数への配布が可能であり、かつサーバがプログラムの正当性を検証できるものでなくてはならない。電子透かしを用いてファイルの改竄を検知する方式が多数提案されている。特に、実行ファイルの改竄を検証する手法として、[3] が提案されている。[3] においては、プログラム内に冗長なコードを加えることにより、電子透かしを埋め込むことが可能である。しかしながら、この方式では、一つのオリジナルプログラムから複数の冗長化プログラムが作成された場合、パターンマッチングによって冗長化された部分が判明するため、不特定多数への配布には不適切である。本節では、クライアントプログラムが自分自身を評価することによって正当性の検証を行う方式を提案する [2]。この方式は ICEGEM のみならず、分散システムなどにおけるファイル同期確認や不正使用者によるファイルの改竄チェック等にも応用が可能であると考えられる。

3.1 概要と前提条件

本方式では、クライアントプログラムの正常な動作を検証するため、サーバがクライアントに対してクライアントプログラムの配布を行い、サーバとクライアントがプログラムを共有する。クライアントプログラムは自分自身のハッシュ値を定期的に計算し、サーバは自身が持つクライアントプログラムのハッシュ値と

比較することによって、正当性の検証を行う。ただし前提として、サーバとクライアントはそれぞれ公開鍵と個人鍵、あるいは各ホスト固有の ID を所持するものとする。またサーバ-クライアント間の通信はパケットレベルで暗号化されているものとし、第三者によるパケット傍受やなりすましは行われぬものとする。

3.2 サーバによるプログラムの送信

クライアントは、クライアントプログラム使用に先立ち、サーバへ自身の ID の送信とクライアントプログラムの要求を行う。クライアントは要求を受けたサーバがクライアントプログラムを正しく受信した後に、その使用が可能となる。またサーバは、クライアントの ID に基づいてそのクライアント専用のハッシュ関数用の秘密鍵 k を作成する。以下において、サーバ上におけるクライアントプログラムを p 、クライアントが使用するクライアントプログラムを p' とする。

3.3 サーバによるプログラムチェック

サーバは、まず k を安全な方法で p' へ送信する。その後、 p' へハッシュ値の要求を行い、同時にタイムスタンプ T_j を送信する。 p' は T_j を加味した自分自身のハッシュ値 h_c を HMAC-MD5[4] 等の鍵付きハッシュ関数の鍵として k を用いることよって作成し、サーバへ送信する。一方でサーバも同様に p と k と T_j からハッシュ値 h_s を求め、 h_c と比較を行う。その結果、ハッシュ値が等しければ、サーバは p' を正当なプログラムであるとみなす。

3.4 ハッシュ値の改竄対策

鍵付きハッシュ関数の利用により、 k に使用期限を設けることで h_c の偽造防止が可能である。また、ハッシュ値は t_j により毎回異なった値となるため、再送攻撃も防止できる。

3.5 問題点

提案方式について、検討を行った。その結果、提案方式の問題点は大きく分けて 2 つ存在した。

i). ハッシュ計算に伴う負荷の発生

サーバにおいて、クライアント数の増加に伴い多数の負荷が集中する。またクライアントにおいても、ICEGEM を 3D ゲームなど複雑な処理を必要とするアプリケーションに適用する際に、負荷によってアプリケーションの動作に支障をきたす可

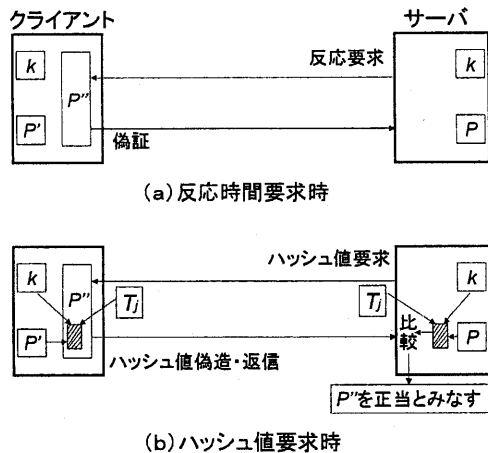


図 3 偽造プログラムの動作

能性がある。

そのため、例えばクライアントの反応頻度や一度のセッション時間、クライアント数などによって、それぞれ要求頻度を变化させる方式が必要である。

ii). クライアント自身による改竄

提案方式では h_c をクライアントが作成するため、第三者によるクライアントプログラムの改竄には効果が期待できる。しかしながら、クライアント自身がクライアントプログラムの改竄を行い、かつサーバへ正しいハッシュ値を送信しようとした場合、クライアントは図 3 に示すように h_c の偽造が可能である。具体的な偽造法について以下に示す。

- クライアントは正規のプログラム p' と、偽証を行うプログラム p'' を持つ。
- 通常の応答時間を含む応答メッセージ送信においては、 p'' が偽証を行う。
- プログラム正当性の検証プロセスにおいては、 p'' は p' を用いることによりハッシュ値 h_c' を作成する。
- サーバは h_c' と h_s を比較し結果が等しいため p'' を正当であるみなす。

クライアントは、 p' を逆コンパイルすることによって、どのように p' が k や T_j を使用するかを知りうる。さらに、クライアントが p'' に管理者権限を与えることによってこれらの秘密情報を p'' は知ることが出来る。よって、提案方式はサーバによって他の端末のプログラムの正当性を検証する方式であるため、外部からの攻撃によるファイル改竄を発見することは可能であるが、クライアント自身による改竄が問題となる ICEGEM では、この方式は効果が無いという結論に達した。

4 応答時間測定カード SIC

4.1 概要

SIC(Secure ICEGEM Card) は、クライアントの応答時間を測定するハードウェアである。クライアントはネットワークインターフェースカード (NIC) として SIC を使用する。SIC は、ICEGEM 使用時以外は NIC として振舞うが、実際はアプリケーション層で動作し、各パケットを監視するものとする。サーバからの問題提示メッセージを受信した際に、SIC がクライアントに代わって応答時間の測定を行う。このとき、図 4 に示すように応答時間の測定に必要な情報は SIC のみが知りうるものとし、クライアントプログラムはサーバからのメッセージの受信と、クライアントの反応を送信するのみとする。そのため、クライアントによる応答時間の改竄を防止でき、サーバの応答時間による順序制御が妨げられない。ICEGEM による順序制御を行うクライアントに SIC の利用を義務付けることにより、応答時間の信頼性が保証される。

4.2 応答時間測定方法

応答時間の計算には、パケットがクライアントに到着した時刻 T_S と、パケットがクライアントから送信された時刻が T_C が必要となる。しかしながら、 T_S をパケットに書き込む場合、クライアントが T_S を改竄する恐れがある。クライアントに応答時刻に関するデータを与えてはならない。そこで、SIC は T_S を記憶するために到着時刻保持テーブルを持つものとする。この到着時刻保持テーブルに保存された T_S と T_C を対応付けるため、各パケットには ID を割り振る。あるパケット ID_n に対してサーバへの返信を行う際に、クライアントは同じパケット ID_n を用いる。以上によって、パケットがクライアントから返信された際にクライアントへの到着時刻を取り出す。なお、テーブル内のデータは、参照されるか、もしくは一定時間経過すると消去される。

4.3 内蔵鍵

サーバが、クライアントが本当に SIC を用いているのか否かを判別するために、SIC に秘密鍵を内蔵する鍵はクライアントごとに異なるようにする。クライアントはセッションの開始時に ID を送信し、その後 SIC が自動的に内蔵の鍵を用いて応答時間を暗号化する。一方サーバでは SIC 製造の段階でクライアントの鍵を保持し、管理する。クライアントからの暗号化された

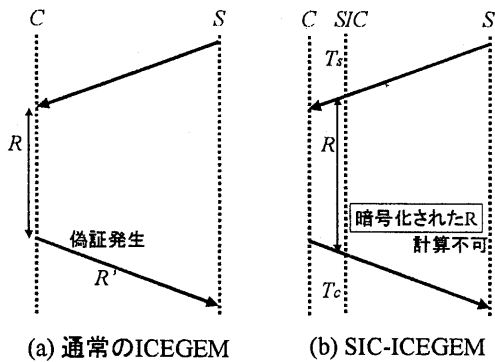


図 4 ICEGEM と SIC-ICEGEM の比較

応答時間に対し、ID に対応した鍵を使用することで復号が可能である。このことによって、別クライアントへのなりすましや、不正プログラムによる SIC のエミュレート、および経路上における改竄を防止する

4.4 検討事項

パケットフォーマット：SIC が、ICEGEM を使用するパケットか否かを判断するために、独自に ICE プロトコルが必要である。ICEGEM を利用するパケットは、パケット ID、送信元サーバアドレス、クライアント応答時間、実データ、及び ICEGEM を表す識別子を含む。サーバは識別子と送信元サーバアドレス、パケット ID、実データをクライアントへ送信する。SIC は到着したパケットのフォーマットを調べることにより、ICEGEM を使用するパケットか否かを判別する。クライアントも同様にパケットに識別子とパケット ID、送信元サーバアドレス、実データを書き込み、サーバへ送信する。SIC が測定した応答時間は、パケットの応答時間フィールドへと書き込まれる。それまでは応答時間フィールドには何も書き込まれない。

同一パケット ID の複数到着：ICEGEM においては UDP を使用しているため、再送処理によって同一の ID を持ったパケットが複数に到着することはない。しかしながら、ルーティングによっては、あるクライアント C_i へ同一 ID を持ったパケットが複数個到着する可能性が考えられる。このような際に、先に到着したパケットの到着時刻 T_{S1} を、遅れて到着したパケット ID の到着時刻 T_{S2} で上書きするか否かが問題となる。なぜなら、先に SIC に到着したパケットが C_i にも到着していた場合、応答時間は $T_{S1} - T_C$ で求める必要があり、そうでなければ、応答時間は $T_{S2} - T_C$ で求められるから

である。しかしながら、SIC とクライアントは密接しているため、SIC-クライアント間でのパケットロスはないと考える。そのため、複数パケットが到着した際は T_{S1} を保持することとする。

IPSec との整合性：IPSec で暗号化された場合、SIC はそのデータを参照することができない。そのため、IPSec との整合性を保つためには、SIC がセキュリティゲートウェイの内側で動作するか、SIC 自体がセキュリティゲートウェイとして動作する必要がある。

同一鍵の複数サーバへの適用：SIC に内蔵される鍵を複数のサーバにおいて使用する場合、その鍵の正当性を保証するための第三者機関を利用することにより、鍵を複数サーバにおいて使用できると考える。第三者機関による認証については、既存の鍵管理方式が利用できる。

5 SIC プロトタイプの作成と評価

SIC を用いる上で懸念されるのは、クライアント-SIC 間の遅延によって実際のクライアントの応答時間と SIC が測定する応答時間にずれが生じることである。そこで、SIC のプロトタイプを、UNIX 上で動作するアプリケーションとして作成し、クライアントの応答時間と測定アプリケーションが測定した応答時間の差を測定した。

5.1 概要

SIC プロトタイプ (以下 SICp) はクライアントとサーバ間のマシンにおいて動作し、応答時間の測定を行う。サーバ、クライアントはパケット送信時に相手の IP アドレスとポートではなく、SICp の IP アドレスとポートを指定する。サーバとクライアント、測定アプリケーションはそれぞれ別のマシンで動作させ、(図 5) に示すように接続した。また通信プロトコルには ICEGEM 同様 UDP を用いた。各プログラムの動作を以下に示す。

サーバ：サーバは SICp に対して ICE 層を含むパケットを SICp へ送信する。この時、パケットにはパケット ID、送信元サーバ IP アドレス、データが含まれる。データは 1 桁から 4 桁までのランダムな数字とした。サーバはパケットを送信した後、クライアントからの返答を受信するか、10 秒の経過の後に次のパケットを送信する。

クライアント：クライアントはサーバからのメッセージを受信し、データに含まれる数値の 2 倍の値を

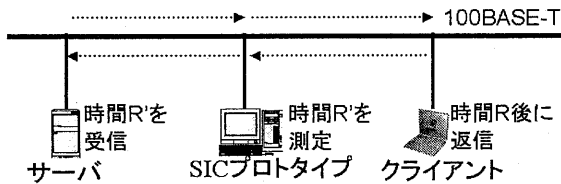


図 5 接続形態

求め、その値と、クライアントの IP アドレス、及びパケット ID を SIC_p へ送信する。ただし、ユーザの反応をエミュレートするため値を求めてから送信まで sleep 関数によって 1 秒間の遅延を行う。ただし sleep による待機は一定ではないため、遅延時間を厳密に求め、 L に保存する。

SIC_p : SIC_p はサーバとクライアントの IP アドレスを知るものとする。SIC_p はパケットを受け取り、IP アドレスを調べる。IP アドレスがサーバのものと同じであれば到着時刻とパケット ID を到着時刻保持テーブルに記憶し、クライアントへ送信する。IP アドレスがクライアントのものと同じであれば、到着時刻保持テーブルから到着時刻を取り出し、応答時間を求めた後、応答時間フィールドに応答時間を書き込み、サーバへ返信する。この時、測定応答時間を L' として保存する。

5.2 測定結果

サーバ・クライアント間のパケット送受信を 1000 回測定し、 L と L' の差を求めた。その結果を (図 6) に示す。その結果、約 20% は誤差 1ms 以下であったが、残り 80% は 1ms~2ms の誤差が発生した。また、わずかながら 2ms~3ms の誤差も測定された。今回の測定環境が LAN 環境であったことを考えると、クライアントに直接 NIC として取り付けた場合にはこの誤差はさらに短くなると考える。しかしながら、誤差は各 SIC ごと不規則に発生するため、クライアント数が多数になった際クライアント間の SIC の誤差に差が生じ、不公平性が起こる可能性がある。SIC を実際に NIC として使用した際に、どの程度不平等性が発生し、その結果どの程度影響が現れるかは、実装に基づく必要がある。参加人数や、アプリケーションへの反応頻度による影響の度合いを含めて詳細に検討する予定である。

6 まとめ

メンバ間公平性保証方式において、クライアントによる偽証を防止する方式を提案した。まずハッシュ鍵を

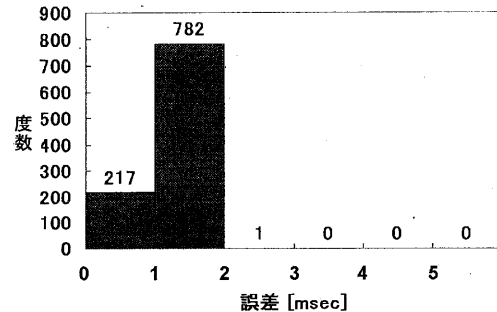


図 6 測定結果

用いてサーバが定期的にクライアントプログラムの正当性を提案した。しかしながら、この方式ではクライアントプログラムが自分自身の秘密情報を全て知りうるため、偽証が発生する恐れが存在する。そこで、ハードウェアである SIC を使用する方式を提案し、プロトタイプを作成した。その結果、クライアントプログラムに全く処理を行わせることなく、誤差 2ms 以内でクライアントの応答時間を測定することができた。この誤差は、人間の応答時間に比べると十分に小さいと考えるが、多数のプレイヤーによって頻繁に応答時間による順序制御を行うアプリケーションでは、順序が完全には公平に制御されない可能性がある。

今後の課題は、SIC の、IP ネットワーク上で透過な SIC の実装、および応答時間の誤差による順序逆転の頻度の評価である。

参考文献

- [1] 石川貴士, 石原進, 井手口哲夫, 水野忠則: メンバ間公平性保証方式の同期機構の特性評価, 情報処理学会研究報告, モバイルコンピューティングとワイヤレス通信, Vol. 2000, No. 15, pp. 81-88, (2000. 12).
- [2] 谷口幸久, 石川貴士, 石原進, 水野忠則: メンバ間公平性保証方式における偽証防止に関する一検討, 情報処理学会全大 1S-6, (2000. 3)
- [3] 中村直己, 西垣正勝, 曾我正和, 田窪昭夫: プログラムの冗長化に関する検討, 情報処理学会研究報告, コンピュータセキュリティ, Vol. 2000, No. 68, pp. 41-48, (2000. 7)
- [4] Cheryl Madson Cisco Systems, Inc. Rob Glenn NIST: The Use of HMAC-MD5-96 within ESP and AH, RFC 2403, (1998.11)