LM-009

P2P ネットワークのための分散ハッシュ型認証手法
A Hash-based Distributed Authentication Method for P2P Network

武田 敦志 * 北形 元 † 松島 悠 ‡ 木下 哲男 § 白鳥 則郎 †
Atushi Takeda Gen Kitagata Yu Matsushima Tetsuo Kinoshita Norio Shiratori

1. まえがき

P2P ネットワークはサーバとなる計算機を必要とせず，従来のサーバ・クライアントモデルのネットワークと比べて利便性などの点で優れている．一方，運用上の問題として，P2P ネットワークに参加する計算機端末 (以降，ノードと呼ぶ) の認証問題がある．本稿におけるノードの認証とは，通信データに付加された電子署名と対象のノードの公開鍵を用いてノードの本人性を認証することを指す [1]．ノードの本人性を認証する有用な既存手法として，ノード利用者と認証局管理者の社会的な信頼を基にノードの認証を行う PKI(Public Key Infrastructure)[2] がある．この手法では認証局と呼ばれるサーバで公開鍵などの認証情報を集中管理する．しかし，全てのノードがネットワークへの参加と離脱を繰り返す P2P ネットワークでは，永続的なサービスを提供できるノードが存在しないため，全てのノードから信頼される特定のノードで公開鍵などの認証情報を集中管理することは難しい．

そこで本稿では，P2P ネットワークに参加しているそれぞれのノードが相互に認証するための効率的な端末認証手法 HDAM (Hash-based Distributed Authentication Method) を提案する．提案手法の基本は，参加しているノード同士を認証するために，信頼の輪と分散ハッシュテーブル (DHT) を用いて公開鍵を分散管理することにある．具体的には，P2P ネットワークに参加するノード間で信頼の輪を形成することにより，それぞれのノードが管理する公開鍵の数を減少させ，ノードに必要となるメモリ量を従来手法より大幅に減少させる．また，DHT を効果的に用いて公開鍵を効率的に分散管理することにより，ネットワークへの参加時，ネットワークからの離脱時，及び公開鍵の更新時に送受信する通信データ量を従来手法より大幅に減少させる．コンピュータシミュレーションにより提案手法の評価を行った結果，参加ノード数が 500 の場合，提案手法は計算機端末に必要なメモリ量を従来手法に比べ最大で 95%削減可能であり，同様に送受信する通信データ量を従来手法より最大で 80%削減可能であることを確認した．

2. 関連研究

サーバを必要としないノード間の相互認証手法として，信頼の輪と呼ばれるノード間の相互関係を用いたノード認証手法がある [3]．この手法は，信頼の輪を利用することにより，認証済みのノードを介して新たな公開鍵を収集することが可能としている．しかし，この手法は計画的な信頼の輪を形成することが出来ないため，任意のノードを認証するためには全てのノードの公開鍵を入手する必要がある．そのため，各ノードは公開鍵を管理するために多くのメモリ量を必要とし，公開鍵を入手するための多くの通信データ量を必要とする．

また，無線アドホックネットワークにおいて，各ノードがそのネットワークに参加する全てのノードの公開鍵を自律的に収集する手法が提案されている [4]．この手法では，各ノードが自律的に全ノードの公開鍵を収集することにより，サーバを利用せずにノード間の認証を実現している．しかし，この手法は全ての公開鍵を無計画に収集するため，各ノードは多くのメモリ量と通信データ量を必要とする．一方，ノードが必要とするメモリ量と通信データ量を削減するために，無線アドホックネットワークのルーティング情報を利用して計画的に公開鍵を収集する手法が提案されている [5]．この手法では，ルーティング情報と信頼の輪という概念を活用することにより，公開鍵のオンデマンドで効率的な収集を実現している．しかしながら，この手法は無線アドホックネットワークのルーティングプロトコルに依存しており，無線アドホックネットワーク以外での実現が難しい．

本稿で提案する認証手法は，分散ハッシュテーブル (DHT) を効果的に用いて計画的な信頼の輪を自動的に形成することにより，計算機ネットワークの種別を制限せずにオンデマンドで効率的なノード間の認証を実現する．

3. 分散ハッシュ型認証手法 HDAM の提案

3.1 HDAM の概要

公開鍵をそれぞれのノードで効率的に分散管理することが出来れば各ノードで管理する公開鍵の数を減らすことが可能となる．また，各ノードで管理する公開鍵の数が減少すれば，各ノードに必要とされるメモリ量や通信データ量が減少する．前述したように，P2P ネットワークに参加しているノード間で形成された信頼の輪を利用すれば，公開鍵を分散管理することが可能となる．しかし，信頼の輪を用いて認証を行うためには，任意のノードの公開鍵を取得するための情報を知る必要があるが，P2P ネットワークでは各々のノードがネットワークへの参加・離脱を繰り返すため，この情報を特定のノードで集中管理することは難しい．また，公開鍵の分散の方法によっては，特定のノードに公開鍵が集中する問題もある．

そこで本稿では，信頼の輪と分散ハッシュテーブル (DHT) を用いて，P2P ネットワークに参加している任意のノード間の認証と公開鍵の効率的な分散管理を実現する HDAM (Hash-based Distributed Authentication Method) を提案する．HDAM では，DHT を効果的に用いて信頼の輪を形成することにより，公開鍵の効率的で安全な分散管理を実現する．これにより，HDAM は特定のノードに公開鍵が集中する問題を解決し，各ノードに必要とされるメモリ量及び各ノードが送受信する通信データ量を従来手法より減少させる．

*東北文化学園大学コンピュータサイエンス学科
†東北大学電気通信研究所/大学院情報科学研究科
‡(株) 新日鉄ソリューションズ
§東北大学情報シナジーセンター/大学院情報科学研究科

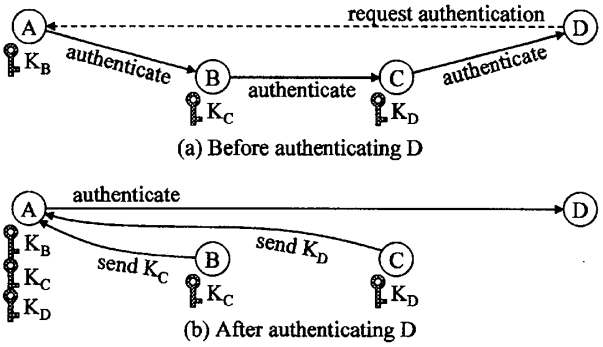

図 1: 信頼の輪を用いたノードの認証

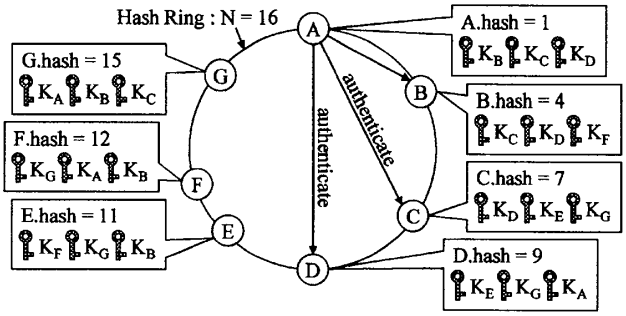

図 2: HDAM による公開鍵の分散管理

3.2 信頼の輪を用いた認証

本稿におけるノードの認証とは，通信データに付加された電子署名と対象ノードの公開鍵を用いてノードの本人性を認証することを指す．2 つのノード A,B が存在し，A が B の公開鍵 K_B を保持している状態を“A が B を認証している状態”と呼ぶ．本稿では，A が認証しているノードの集合を $A.trust$ と表す．

図 1 に信頼の輪に基づくノードの認証手順を示す．ノード A,B,C,D が存在し，$B \in A.trust$, $C \in B.trust$, $D \in C.trust$ であり，D が A に対して認証要求を行った場合を考える (図 1(a))．この時，A は D の公開鍵 K_D を保持していないため，D を認証することが出来ない．そこで，以下の手順に従って公開鍵 K_D を入手する．

1. B から K_C を入手する．$C \in A.trust$ となる．
2. C から K_D を入手する．$D \in A.trust$ となる．

以上の手順により A は D の公開鍵 K_D を入手し，K_D を用いて D を認証する (図 1(b))．このように，認証済みのノードから間接的に公開鍵を入手して新たなノードの認証を行う認証手法を信頼の輪を用いた認証と呼ぶ．

3.3 DHT を用いた公開鍵の分散管理

図 2 に HDAM による公開鍵の分散管理の例を示す．ここで，$i.hash$ はノード i のハッシュ値，K_i はノード i の公開鍵，N はハッシュ値がとりうる最大の値 (最大ハッシュ値) を表す．HDAM では，ノードの識別子から一方向ハッシュ関数で求めたハッシュ値 ($i.hash$) を基に，1 から N までの指標を円形に配置したハッシュリング上に各ノードを仮想的に配置する．そして，各ノードはハッシュリングにおいて自身の位置から正の方向に $2^k (k = 0, 1, 2, \cdots)$ 以上離れたノードのうち最も近い位置に配置されたノードの公開鍵を管理する．図 2 の場合，A が管理する公開鍵は以下の 3 個となる．

- 正の方向に $2^1 (2^0)$ 以上離れたノードの中で最も近い位置に配置されたノードである B の公開鍵 K_B
- 正の方向に 2^2 以上離れたノードの中で最も近い位置に配置されたノードである C の公開鍵 K_C
- 正の方向に 2^3 以上離れたノードの中で最も近い位置に配置されたノードである D の公開鍵 K_D

上記の公開鍵を A が管理することにより，認証関係は $\{B, C, D\} \subseteq A.trust$ となる．最大ハッシュ値が N のとき，各ノードが管理する公開鍵の最大数は $log_2 N$ となる．

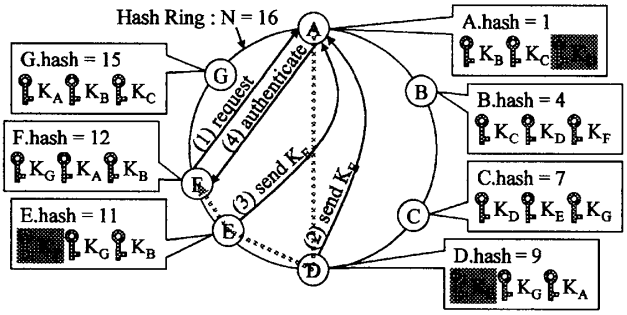

図 3: 認証手順

3.4 信頼の輪と DHT を用いた端末認証手順

ノード n がノード d の公開鍵を保持していない時に d から n への認証要求が行われた場合，以下の手順により n は d の公開鍵を入手し，d を認証する．

1. n は自身が認証しているノードの中から，ハッシュリング上で最も d に近い位置に配置されたノード n^t に対し，d の公開鍵 K_d を要求する．
2. n^t が公開鍵 K_d を保持している場合，n^t は n へ K_d を送信する．これにより，n は d を認証する．
3. n^t が公開鍵 K_d を保持していない場合，n^t が認証しているノードの中からハッシュリング上で最も d に近いノード n' の公開鍵 $K_{n'}$ を n に送信する．n は n' を認証し，手順 (1) に戻る．

図 3 にノード間の認証手順の例を示す．この例では，前述した手順に従い，ノード A がノード F を認証する．

1. F が A に対して認証要求を送信する．
2. A が D に対して F の公開鍵 K_F を要求する．この要求に対して，D は A に対して E の公開鍵 K_E を返す．$E \in A.trust$ となる．
3. A が E に対して F の公開鍵 K_F を要求する．この要求に対して，E は A に対して F の公開鍵 K_F を返す．$F \in A.trust$ となる．
4. A が F を認証する．

以上の手順により，A は公開鍵 K_F を入手し，F を認証する．最大ハッシュ値が N のとき，認証に必要となる通信データ量は $O(log_2 N)$ となる．

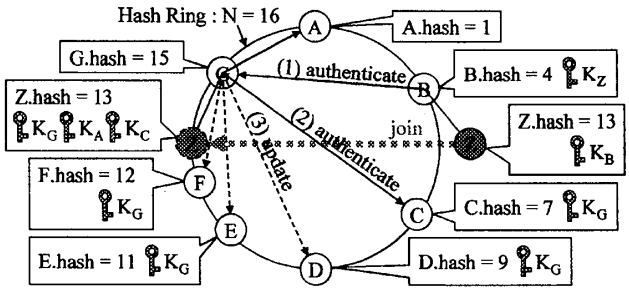

図 4: ノードの参加手順

3.5 ノードの参加手順

ノード n が P2P ネットワークに参加するときの前提条件として，n はすでにネットワークに参加している任意のノード g と相互に認証しているものとする．n がネットワークに参加する手順を以下に述べる．

1. n はハッシュリング上で自身の正側に隣接するノード $n.successor$ の公開鍵を g から取得する．
2. n が認証すべきノードを計算し，それらの公開鍵を $n.successor$ から取得する．
3. $n.successor$ は自身の公開鍵を配布したノードに対して，信頼の輪の再構築を通知する．
4. 再構築の通知を受けたノードは自身が認証すべきノードを再計算し，それらのノードの公開鍵を取得する．n の公開鍵は $n.successor$ から取得する．

図 4 にノードがネットワークに参加する際の例を示す．この例では，ノード Z がノード B を介してネットワークに参加する．以下，本例をもとに処理手順を述べる．

1. Z はハッシュリング上で自身の正側に隣接するノード G の公開鍵 K_G を B から取得する．
2. Z は自身が認証するノード G, A, C の公開鍵を G から取得する．
3. G は自身の公開鍵を保有しているノード F, E, D, C に対して，信頼の輪の再構築を通知する．この通知をうけたノードのうち，F, E, D は Z を認証するために Z の公開鍵 K_Z を G から取得する．

以上の手順により Z はネットワークへ参加する．最大ハッシュ値が N のとき，ノードがネットワークに参加するために必要な通信データ量は $O(log_2 N)$ となる．

3.6 ハッシュ値の重複への対応

それぞれのノードに割り当てられるハッシュ値はノードの識別子を基にした一方向ハッシュにより決定するため，P2P ネットワークに参加している複数のノードのハッシュ値が重複する可能性がある．そこで HDAM では，複数のノードのハッシュ値が重複した場合，これらのノードの関係が階層構造となるように配置する．具体的には，ハッシュ値が重複したノード群の中で最初にネットワークに参加したノードを親ノードと呼び，このノードは通常通りハッシュリング上に配置する．一方，親ノードと重複したハッシュ値を持ったノードは子ノードと呼ばれ，ハッシュリング上には配置せず，その認証はすべて親ノードを介して行う．

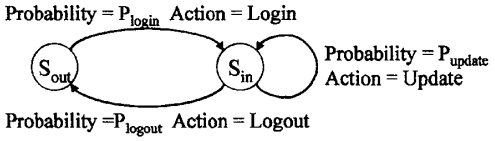

図 5: ノードエージェントの状態遷移図

3.7 ノードの離脱手順

ノードがネットワークから離脱する場合，離脱ノードはハッシュリング上で隣接するノードに離脱通知を行い，この通知に従って隣接ノードは離脱ノードが存在しない信頼の輪を再構築する．また，予告なくノードが離脱した場合，隣接ノードがノードの離脱を検知し，隣接ノードは離脱ノードが存在しない信頼の輪を再構築する．最大ハッシュ値が N のとき，ノードの離脱に必要な通信データ量は $O(log_2 N)$ となる．

3.8 公開鍵の更新

信頼の輪を安全に運用するためには一定期間ごとに公開鍵を更新する必要がある．HDAM では，各ノードが自身の古い公開鍵を所有しているノードに対して新しく生成した公開鍵を送信することにより，公開鍵の更新を行う．最大ハッシュ値が N のとき，公開鍵の更新に必要な通信データ量は $O(log_2 N)$ となる．

4. 特性評価

4.1 シミュレータの概要

HDAM の特性を評価し，提案手法の有効性を示すために，Java を用いて P2P ネットワークの動作シミュレータを設計・実装した．このシミュレータは，ネットワークに参加するノードをエージェント (ノードエージェント) として実現したもので，ノードエージェント間でメッセージを送受信することにより，ネットワークへの参加時，ネットワークからの離脱時，及び公開鍵の更新時に行われるノード間の通信をシミュレートする．

図 5 にノードエージェントの状態遷移図を示す．ノードエージェントは，ネットワークに参加していない状態 (S_{out}) とネットワークに参加している状態 (S_{in}) を持つ．ノードエージェントは状態 S_{out} の時に確率 P_{login} で状態 S_{in} に移行し，状態 S_{in} の時に確率 P_{logout} で状態 S_{out} に移行する．また，状態 S_{in} の時に確率 P_{update} で公開鍵を更新する．状態 S_{in} への移行時には 3.5 で述べた手順に基づいてメッセージの送受信を行う．また，送受信する全てのメッセージには電子署名が付加されており 3.4 で述べた認証手順に基づいてノードの認証を行う．

以上の動作を行うシミュレータを用いて HDAM の特性評価を行った．この評価では，ハッシュリング上に配置される親ノードを対象に，このノードが管理する公開鍵数とこのノードが送受信するメッセージ数を計測した．ここで，公開鍵数はノードが必要とするメモリ量を示しており，メッセージ数はノードが送受信する通信データ量を示している．ノードエージェントの設定パラメータは $P_{login} = 1.0, P_{logout} = 0.25, P_{update} = 0.15$ とした．

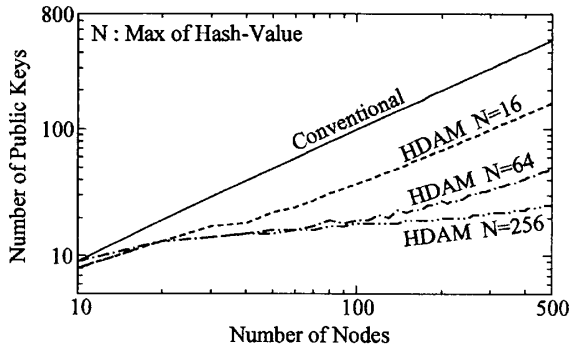

図 6: ノード数と公開鍵数の関係

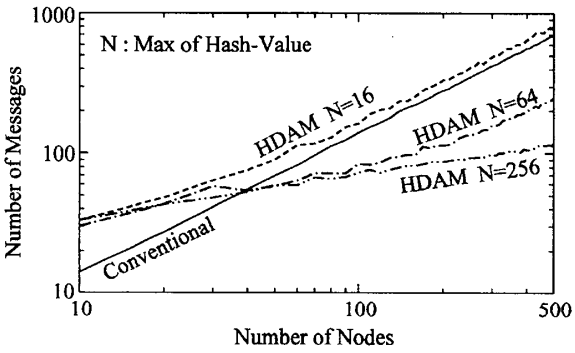

図 7: ノード数とメッセージ数の関係

4.2 従来手法との比較

提案手法の有効性を示すために，HDAM と従来手法との比較を行った．比較対象の従来手法はすべてのノードの公開鍵をそれぞれのノードで独自に管理する手法とした [3][4]．この従来手法をシミュレータ上に実装し，HDAM と従来手法を比較評価した．

図 6 にネットワークに参加しているノードの数と 1 個の親ノードが管理する公開鍵の数の関係を示す．ここで，N は最大ハッシュ値を表す．また，実線は従来手法を示し，それ以外の線は提案手法である．図 6 より，HDAM の親ノードが管理する公開鍵の数は従来手法に比べて大幅に減少していることが分かる．特に，$N = 256$, ノード数 $= 500$ のとき，HDAM の親ノードが管理する公開鍵の数は従来手法に比べ 95%以上削減されている．このことより，HDAM は各ノードに必要となるメモリ量が従来手法に比べて大幅に少なく，効率の高い手法といえる．

次に，ネットワークに参加しているノード数と 1 個の親ノードが送受信するメッセージ数の関係を分析する．図 7 に，ネットワークに参加しているすべてのノードが図 5 に示した状態遷移を繰り返し行い定常状態に達したとき，1 個の親ノードが 1 ステップで送受信するメッセージ数の平均を示す．ここで，N は最大ハッシュ値を表している．また，実線は従来手法を示し，それ以外の線は提案手法である．図 7 よりノード数が 40 以下のとき，HDAM における親ノードが送受信するメッセージ数は従来手法に比べて大きくなることがわかる．これは，従来手法は信頼の輪の構築を必要としないのに対し，HDAM はこれを構築するためである．しかしノード数が 40 以上のとき，HDAM の親ノードが送受信するメッセージ数は従来手法に比べて小さく，ノード数の増加に従ってその差が大きくなることがわかる．特に，$N = 256$, ノード数 $= 500$ のとき，HDAM の親ノードが送受信するメッセージの数は従来手法に比べ 80%以上削減されている．これは，従来手法では公開鍵を送受信するためのメッセージ数がノード数に比例するのに対し，HDAM では公開鍵を送受信するためのメッセージ数がノード数 p に対し $O(log_2 p)$ となることによる．

これらの評価結果より，40 個以上のノードで構成される P2P ネットワークで HDAM を用いた場合，それぞれのノードに必要とされるメモリ量とそれぞれのノードが送受信する通信データ量は従来手法を用いた場合に比べて少なく，この差はノード数が増加するに従い大きくなることがわかった．この結果より，HDAM は従来手法と比べてスケーラビリティに優れているといえる．

5. むすび

本稿では，信頼の輪と分散ハッシュテーブルを用いて P2P ネットワークに参加しているノードを相互に認証する手法 HDAM を提案した．HDAM では，それぞれのノードが管理する公開鍵の数と送受信する通信データ量が従来手法に比べて減少され，かつ，ネットワークに参加しているすべてのノードが確実に認証可能となる．コンピュータシミュレーションにより HDAM を評価した結果より，P2P ネットワークに参加しているノード数が 40 以上の場合，ノードが必要とするメモリ量と通信データ量を従来手法より削減可能であり，ノード数が 500 の場合はこれらを大幅に削減可能であることを検証した．

謝辞

本研究の一部は科研費（19200005）の助成を受けたものである．

参考文献

[1] S. Farrell and R. Housley. Rfc 3281: An internet attribute certificate profile for authorization, 2002.

[2] R. Housley, W. Polk, W. Ford, and D. Solo. Rfc 3280: Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile, 2002.

[3] Simson Garfinkel. *PGP : Pretty Good Privacy*. Oreilly and Associates Inc., 1994.

[4] Srdjan Capkun, Levente Buttyan, and Jean-Pierre Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2, No.1:52–64, 2003.

[5] 北田 夕子, 荒川 豊, 竹森 敬祐, 渡辺 晃, and 笹瀬 巌. 無線アドホックネットワークに適したルーティング情報を用いたオンデマンド公開鍵分散管理方式. **電子情報通信学会論文誌**, J88-D1, No.10:1571–1583, 2005.