社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－DPS－126 (21)
2006－CSEC－ 32 (21)
2006／3／16

DHTを用いた新しいSelfish Node対策手法の提案

荻野 剛[†] 金子 伸一郎[†] 上田 真太郎[†]
重野 寛[†] 岡田 謙一[†]

近年, 無線インターフェースを標準装備した機器が普及しつつあり, それに伴いモバイルコンピューティングが急速に発展している. そして今後ユーザのコミュニティにおいて相互協力的にパケットを中継する形態のアドホックネットワークが一般的になると考えられる. この中において非協力的で利己的に振舞うノード (Selfish Node) の存在によりネットワークの公平さが失われ, 最終的にネットワークが利用不能となる恐れがある. これはアドホックネットワークにおいてセキュリティ面から見て重要な問題である. そこでこのようなSelfish Nodeをネットワークから追い出すのではなく貢献させるような環境をつくることでこの問題を解決することを目指すSelfish Node対策手法を提案する.

Proposal of novel countermearsure against the Selfish Node by using the DHT in Ad Hoc network

Takeshi Ogino Shin-ichiro Kaneko Shintaro Ueda Hiroshi Shigeno Ken-ichi Okada

With the rapid deployment of mobile electronic devices with wireless interfaces, mobile computing networks are growing. Therefore Ad Hoc networks where packets are forwarded among users will become more common. However, the existence of selfish nodes which act uncooperatively, will effect the fairness of the Ad Hoc network and even may eventually make the network fall. This is a nontrival issue in Ad Hoc networks in the terms of security perspectives. Therefore in this paper we will propose a method where selfish nodes are not expeled from the network but stimulated to cooperate with the other nodes.

1. はじめに

今後, P2Pやアドホックネットワークなどの特定の中央機関を利用しないネットワーク網の拡大が見込まれている. それに伴い新しいセキュリティ上の問題も想起されてきている. 今まで公開ネットワークにおいて通信をする上で問題となっていた, データの改ざん, 成りすまし, 事後否認などだけではなく, 新たにノードの相互協力に関する問題が大きなトピックとして浮上してきている [1].

特定のインフラストラクチャーのないアドホックネットワークでは各ノードでパケットフォワーディングして通信を行う必要がある. しかしアドホックネットワークを構成するノードはノートパソコンや携帯電話, PDAなどのモバイル端末でありリソースの限られた機器であることが多いため, パケットフォワーディングによる電力消費や帯域の減少は無視することが出来ない. そこで自身のリソースの消費を避けるため, 自身からはパケットを発信するが他人から回ってきたパケットはフォワーディングしない, ネットワークに非協力的なSelfish Nodeの問題が注目されている. Selfish Nodeは悪意ある攻撃者とは違いネットワークの破壊自体を目的とするわけではなく, 人のリソースのみを使ってネットワークを利用することを目的としている. このSelfish Nodeの問題の大きなポイントとして, Selfish Nodeによるルートパスの分断及び通常のノードとSelfish Nodeのリソース消費量の不公平さの2点が挙げられる.

この Selfish Node の問題は, 一般ユーザが安易な考えで行う攻撃であると考えられるため, 単純に攻撃者をネットワークから排除する手段をとるのはノードの数を減少させ, 結果的にネットワークに打撃を与えることになりかねない. そこで, パケットフォワーディングを回避しようとする Selfish Node を追放するのではなく, ネットワークにより貢献させる環境を作ることでこの問題を解決する必要がある. これによりパケットが故意に途中でドロップされることなく目的ノードまで達し, また各ノード間でのリソース消費量の不公平さの無いアドホックネットワークを実現する.

以下, 本稿では第2章でSelfish Nodeの問題の背景と関連研究について述べ, 第3章でSelfish Node対策手法を提案する. 第4章では今後予定している評価方法について述べる. 最後にそれによって予想される結果と意義についても述べる.

2. 背景と関連研究

2.1 Selfish Node

アドホックネットワークとはアクセスポイントを必要とせずに複数のノードが無線を用いて接続することで構成されるネットワークのことである. 中央集権的なサーバを含まずにノード間でパケットを送受信して通信を行う. その際にワンホップでは届かない範囲にいる相手と通信するためにはルート上のノードが送信者のパケットを受信者までフォワーディングする必要があるが当然パケットフォワーディングすることでリソースを消費することになる. ここで自身のリソースをセーブするため自

[†]慶應義塾大学大学院理工学研究科

身が通信するときには他のノードの構成しているネットワークを利用するが, 他のノードが通信するときには協力を拒否するノードのことを Selfish Node[2] と呼ぶ. このような Selfish Node の挙動は結果としてネットワークに障害をもたらすことになるがネットワークの破壊を目的として行われるものではなく, 自身のリソース保護のためにのみ行われる攻撃であるため, 自己のリソースを消費してまでネットワークに打撃を与えようとする積極的攻撃者とは違う. Selfish Node が得られる利点として

- 自身の消費電力の抑制
- 利用可能な帯域の増加

の 2 点が挙げられる.

また、ネットワークに対して及ぼす影響として

- 通信相手までのホップ数の増加
- 通信効率の悪化、または通信不能

の 2 点が挙げられる.

Selfish Node はネットワーク知識の薄い一般の利用者であっても簡単に実行可能であるため, 対策が必要である.

2.2 関連研究

現在提案されている Selfish Node への対策手法は大きく以下の 3 つのアプローチに分けることが出来る.

1. ノードの挙動を監視する手法
2. レピュテーション を用いる手法
3. インセンティブプライシングを用いる手法

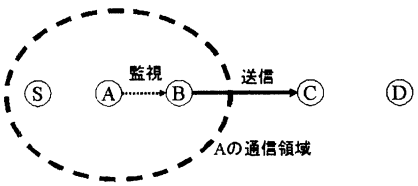

図 1: Watchdog

2.2.1 ノードの挙動を監視する手法

S.Marti の Watchdog & Pathrater [2] という手法では各ノードが他のノードのトラフィックを監視することで Selfish Node を判別して, ルーティング時に意図的に Selfish Node を避けるという 2 つの段階を用いて Selfish Node に対抗している.

まず Watchdog を用いて Selfish Node を特定する. 図 1 において Watchdog の働きを示す. S がメッセージの送信ノード, D が受信ノード, A, B, C が中継ノードである. S から送信されてきたパケットを A は直接 C まで送信することは出来ないが, B から C へのトラフィックは監視することが出来る. そこで A は隣の B にパケットフォワーディングした後, B が次の C にパケットフォワーディングするかどうか監視することが出来る. また, 各リンク毎でメッセージの暗号化がされていない場合は, メッセージやヘッダーの内容の改ざんの有無も判断することできる.

次に Pathrater を用いて Selfish Node を避けるルーティングを行う. Pathrater はネットワークに存在する全てのノード上で動いており, Watchdog で得られた情報から各ノードの信頼度についてレーティングを行う. 信頼度は各ノードのパケットフォワーディング率を元に計算しておりパケットフォワーディングを行ったときは信頼度を 1 上げ, 行わなかったときには 1 下げる. ルーティング時に選択ルートが複数ある場合には中継ノードの信頼度の平均値を算出し, 最も平均値の高いルートを選択する. これにより, より確実に目的ノードまでメッセージを送信することが出来る.

この手法の利点としてパケットフローを直接監視するため, RREQ (Route Request) の中継拒否などによる単純な攻撃には効果的である. ただし, 各ノードが常にネットワークを監視するためネットワークに負担がかかる. また, パケット衝突により監視が失敗したり自然パケットロスによる誤検知などの問題がある. パケットを目的のノードまで確実に送信することについては効果があるが, それにより Selfish Node をルーティング上から省くため Selfish Node のリソースのセーブに協力して利益を与えてしまうという問題もある.

2.2.2 レピュテーション を用いる手法

前述した Watchdog & Pathrater ではルート上から Selfish Node を取り除くことで確実にメッセージが届くルーティングを目指したが, これでは Selfish Node と通常のノード間でのリソース消費量の不公平さが大きくなる.

S.Buchegger の提案した CONFIDANT[3] という手法では各ノードのレピュテーションを評価し, 不正な挙動を行うノードに対しては罰を与えることで Selfish Node をネットワークから排除する対策をとっている. この手法は大きく 1)Monitor 2)Trust Manager 3)Reputation System 4)Path Manager の 4 つの機構に別れている. 以下でそれらの役割について述べる

1)Watchdog を用いて近隣ノードのパケットフローを監視する. その挙動に応じて近隣ノードの信頼度を設定する. 2) 各ノードが絶対に信頼できるノードを friend と呼び, そのノードを確保する. friend の friend もまた friend として出来上がった friend のリストを friend list とする. friend はこのアドホックネットワークでの通信以外の方法で取得する必要がある. 3)Monitor した結果から得られた信頼度の情報を friend list 上の friend 間で共有する. 共有された信頼度を レピュテーションとする. 4) レピュテーションの結果から一定の閾値以下のノードを Selfish Node と判断しルーティング情報から削除する. ノード自体も無視して, そのノードが送信者

のパケットは一切中継しないようにする.

以上の手法を用いることで Selfish Node は不正とみなされた挙動を行うとネットワークから孤立してしまうため, 結果的に Selfish Node にパケットフォワーディングを促すことが出来る.

この手法の利点として複数のノードと情報を共有することでより多くの Selfish Node の存在を検知できる. また Selfish Node に対して罰を与えるため, S.Marti の手法のように Selfish Node に利益を与えることにはならない. しかし, 絶対に信頼出来るノードである friend の確保の方法が確立されておらず, その friend が確保できない場合は虚偽の報告に脆弱である. また Watchdog のシステムをそのまま使用しているため Watchdog の欠点である自然パケットロスによる誤検知や, パケット衝突による監視の失敗などの問題についても対処出来ていない. また Watchdog & Pathrater と共通の問題点として, Selfish Node の問題は一般ユーザが犯しやすい問題であるため Selfish Node のネットワークからの排除はネットワーク自体にもダメージを与えることになり, ネットワーク全体のパフォーマンスを下げてしまうという点が挙げられる.

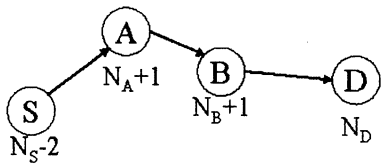

図 2: プライシング

2.2.3 インセンティブプライシングを用いた手法

プライシング手法とはネットワークに貢献したノードには報酬を与え, 利用しただけのノードからは徴収するという手法である. ネットワークを利用するためには必ず他のノードのためにも働く必要があるので各ノードのネットワークに対する貢献を促すことになる. Selfish Node の隔離ではなく積極的な参加を強制することで Selfish Node もパケットフォワーディングを行うようになる. それにより故意のパケットロスによるルートパスの分断及び通常のノードと Selfish Node 間でのリソース消費量の不公平さの両方の問題を解決することが出来る. プライシングを用いた手法の中でも耐タンパ性のあるモジュールを用いる手法と, ある特別なノードを用いる手法の 2 つのアプローチが考えられている.

L.Buttyan[4] は耐タンパ性のあるモジュール [6] を用いてプライシングを行う手法を提案している. 前提条件としてネットワークに参加している全ノードは耐タンパ性のあるモジュールを持っており, そこには nuglet counter というポイントが保存されている. まず, ノードが送信元となって n ホップ先のノードにパケットを送りたいとき, nuglet counter が n 以上あれば送信することが出来る. 送信後 nuglet counter は n 減る. 次にノードが中継ノードとしてパケットフォワーディングをしたとき nuglet counter は 1 増える. 図 2 においてノード S がノード D までパケットを送信したとき, ノード S の nuglet counter N_S は 2 減り途中の中継ノードの nuglet counter は 1 ずつ増える.

S.Zhong の提案した SPRITE[5] という手法では CCS(Credit Clearence Service) と呼ばれる特別なノードに全てのノードのポイントを管理させることでプライシングを行っている. ノードはパケットを受信したときそのパケットを受信したことを証明する receipt を生成する. その後ノードは CCS に receipt を提出してパケットフォワーディングしたことを申請する. CCS はノードから送られてきた receipt を用いて各ノードのポイントの決済を行う.

インセンティブプライシングを用いた手法の利点としてノードのネットワークへの貢献を促すことに高い効果があり, コンセプトが完全に Selfish Node の問題に一致していると言う点が挙げられる. 欠点としては特殊なハードウェアやアドホックネットワークには不自然な存在である特別なノードが必要という前提条件の困難さが挙げられる.

プライシング手法の困難さはポイントの管理手法に大きく依存しており, 手法自体の目的と方法にはズレがないと考えられる. そこで, ポイントの報酬と徴収の管理手法さえ確立されれば, この問題に対して非常に有効になる. そこでこれらの問題点を解決するための手法を第 3 章で提案する.

3. 提案手法

本章では本研究において DHT を用いたポイントの管理を行う Selfish Node 対策手法を提案する. パケットフォワーディングを促すためのインセンティブプライシングの応用, 及びポイントの安全な管理の実現を具体的な目的とする.

3.1 パケットフォワーディングを促すためのインセンティブプライシングの応用

通常, インセンティブプライシングは仮想的なポイントを用いて行われる. しかし, このポイントはそれぞれのアドホックネットワーク毎で設定されている場合もあり, 短期間しかネットワークに滞在しないユーザに対してはパケットフォワーディングを促す対価として効果が薄い. そこで, パケットフォワーディングに対する報酬の与え方に実利的な価値を持たせることであらゆるユーザに対してよりパケットフォワーディングに対するインセンティブを持たせる必要がある. 既存手法ではポイントの増減は送信できるパケット数の増減であったが本手法では使用可能な帯域をリアルタイムに増減させるなどと言った別の形で報酬を与える.

3.2 ポイントの安全な管理の実現

インセンティブプライシングにおいて最も困難な課題が安全なポイントの管理手法の確立である. ポイントの管理を各ノード自身に任せると Selfish Node は自身のポイントを不正に改ざんする恐れがある. よっていかにポイントの改ざんを防ぐかということがプライシング手法におけるセキュリティ上の最大の論点になる. しかし

関連研究で述べたような手法は前提条件があまりにも困難すぎるため一般的に普及させるのが難しい．そこで，本研究ではポイントの改ざんそのものを防止するのではなく，Selfish Node にとって改ざんする動機をなくすことで安全なポイントの管理を目指す．

3.2.1 ポイント管理者の定義

ポイントをノード自身が管理した場合，前述した Selfish Node による有利な改ざんが行われる恐れがある．そこで，各ノードのポイントを自身以外のノードに管理させる方法を用いる．このポイントを管理するノードを管理されるノードのポイント管理者と呼ぶ．ただし，この方法では悪意あるノードがポイント管理者になった場合，故意に他のノードのポイントを改ざんしてネットワークを破壊しようとする攻撃にもつながると考えられる．

この問題についてはひとつのノードのポイントに対し複数のノードがそのポイント管理者となって管理し，多数決をとることで回避出来る．しかし Selfish Node は自身のリソースのみに興味があり，ネットワークを破壊すること自体は望んでいないため今回は考慮する必要がないものとする．

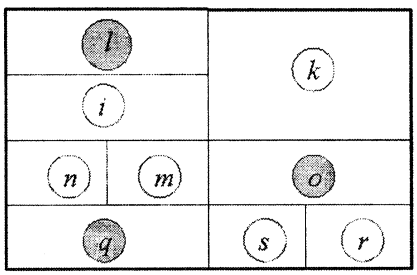

図 3: Distributed Hash Table

3.2.2 ポイント管理者の割り当て

前述のポイント管理者を選択するにあたり CAN[7] や Chord[8] といった DHT(Distributed Hash Table) を用いる．DHT とはノードとコンテンツをハッシュ空間上に割り当てて情報を管理する手法であり，分散環境でのデータの管理に優れている．本提案ではノードの IP アドレスをハッシュ化したものを DHT のハッシュ空間上のアドレスに割り当てる．このハッシュ空間をカバーするノードがその上に存在するアドレスのノードのポイント管理者となる．これにより，あるノードのアドレスを知っているノードはそのノードのポイント管理者に問い合わせることが出来るようになる．

たとえば図 3 においてノード i のアドレスを特殊なハッシュ関数にかけて座標に変換した値が，q がカバーする座標上にあった場合，ノード q が i のポイント管理者となる．今回，DHT を用いることで誰が，誰のポイントを管理しているのか直接把握することが出来ず Selfish Node は自分の有利なように改ざんを行えないため，不正な操作をする動機を下げることが出来る．

3.2.3 ポイントの管理

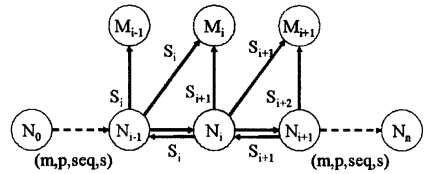

図 4: 提案手法

ポイントの獲得に関しては S.Zhong[5] が考案したレシートを用いる．図 4 において概要を示す．ここではノード N_i の公開鍵及び秘密鍵を PK_i, SK_i，それを用いた署名と検証を $sign_{SK_i}()$, $verify_{PK_i}()$ と表記する．以下，レシートの生成手順を示す．

まずノード N_0 がペイロードデータ m のパケットを送信したいとき N_0 は署名 s を生成する．ここで s は以下の式で表すことが出来る．

$$s = sign_{SK_0(MD(m),p,seq)} \quad (1)$$

式 (1) において $MD()$ はハッシュ関数，p は DSR プロトコル [9] を用いて探索された送信者から受信者までのルートパス，seq はパケットのシーケンス番号を示す．N_0 は次のノードに対し (m, p, seq, s) を送信し，seq を 1 増加する．

N_i がパケットを受信したとき N_i はまず 1)N_i がルートパス上に含まれているか確認 2)seq が増加しているか確認 3)$verify_{PK_0}((MD(m), p, seq, h)s)$ で署名を検証の 3 つの確認を行う．3 つの確認のうち一つでも失敗した場合はパケットをドロップする．全て確認出来たら $(MD(m), p, seq, s)$ を $receipt$ として保存する．N_i がそのメッセージの受信者ではなくパケットフォワーディングする場合は (m, p, seq, s) を次のノードに送信する．

ノード N_i は パケットフォワーディングされたことを証明するため $S_i = sign_{SK_i}(receipt)$ を N_{i-1} に送信する．その後 N_i は生成した $receipt$ と N_{i+1} から送信されてきた $S_{i+1} = sign_{SK_{i+1}}(receipt)$ をパケットフォワーディングした証明として自身のポイント管理者 M_i に提出する．また，N_{i+1} から S_i+1 が送信されたことを証明するため N_i+1 のポイント管理者 M_{i+1} に対して S_i+1 を送信する．ポイント管理者 M_i は N_{i-1} から送信された $verify_{PK_i}(S_i(receipt))$ が $TRUE$ かつ $verify_{PK_i+1}(S_{i+1}(receipt))$ が $TRUE$ のとき，N_i のポイントを増加させる．最終受信者のポイント管理者は送信者のポイント管理者に対し，何ホップかかったのか申告する．送信者のポイント管理者はかかったホップ数に応じたポイントを減少させる．以上の手順によりプライシングを行う．

4. 評価予定項目

評価方法としてはシミュレーションによる評価を予定している. 仮想的に 50〜200 のノードをアドホックネットワークで接続する環境を想定する. シミュレータには ns2(Network Simulator) を用いる.

本手法では既存手法で困難であった前提条件の回避が最も大きな特徴となっている. 本手法の効果を実証するため, オーバヘッドとパフォーマンスについて評価する.

オーバヘッドについてはまず本手法を適用した場合の CPU の処理時間及び必要メモリと帯域について測定し, 本手法を適用しない場合に比べて十分現実的な値であることを示す.

本手法のパフォーマンスについてはパケット到達性及び各ノードのパケットフォワーディング回数の分散について測定する. パケット到達性の測定ではランダムに選ばれた 2 ノード間でパケットが到達する確率を測定し, Selfish Node によるルートパスの分断に対する効果を示す. また通常のノードのパケットフォワーディング回数の平均と Selfish Node のパケットフォワーディング回数の平均を測定し比較することで Selfish Node と通常のノード間でのリソース消費量の不公平さを緩和し Selfish Node をよりネットワークに貢献させていることを示す. また今後, ノードの残りリソース量に応じたポイント管理手法の研究も視野に入れて, 各評価項目とノードのリソースとの関係についても評価することを検討している.

評価を行う上でのパラメータとして, ネットワーク全体のノード数, Selfish Node が占める割合, 最大ホップ数, Selfish Node がパケットフォワーディングしない頻度, 通常ノードが自らパケットを発信する頻度, Selfish Node が自らパケットを発信する頻度を予定している.

5. 予想される結果と意義

パケットフォワーディングを促すためのインセンティブプライシング手法の応用では, 消極的な攻撃しか行わない Selfish Node をネットワークから排除しないことでネットワークを本来の形に保ち, また積極的にパケットフォワーディングを促すことでネットワークを活性化する. ポイントの安全な管理の実現では特別な機器やノードの存在を必要とせずにポイントを管理するため幅広いユーザに利用してもらえる. また Selfish Node は自身の管理するポイントを改ざんしたとしても得るものがないため改ざんする動機を持たない. これにより, プライシング手法において用いられるポイントを安全に管理することが出来る.

6. おわりに

アドホックネットワークにおいて今後、重要視されるであろう Selfish Node の問題について考察し, 対策手法を提案した. 既存研究の最も大きな課題である, 困難な前提条件を廃し, よりユーザが公平にネットワークを使用する手法を提案した. 今後の課題としてはポイント管理者の割り当てにおいて, ノードの出入りによるポイント管理の強度が DHT の強度に依存しているためより柔軟な DHT の構築が必要である.

参考文献

[1] M. Conti, E. Gregori, and G. Maselli, "Towards Reliable Forwarding in Ad Hoc Networks," In *Proceedings of the Eighth International Conference on Personal Wireless Communications* (*PWC* 2003), pp.790-804, September 2003.

[2] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," In *Proceedings of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp.255-265, 2000.

[3] S. Buchegger and J.-Y. Le Boudec, "Performance analysis of the CONFIDANT protocol", In *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp.226-236, June 2002.

[4] L. Buttyan and J.P. Hubaux, "Stimulating Cooperation in Self-organizing Mobile Ad Hoc Networks," *ACM/Mobile Networks and Applications* (*MONET*), Vol.8, No.5, pp.579-592, October 2003.

[5] S. Zhong and J. Chen, "SPRITE: Cheat-Proof, Credit-based System for Mobile Ad-Hoc Networks," In *Proceedings of Twenty – Second Annual Joint Conference of the IEEE Comupter and Communications Societies*, Vol.3, pp.1987-1997, April 2003.

[6] IBM, IBM 4758 PCI Cryptographic Coprocessor, Secure Way Cryptographic Products, June 1997.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-Addressable Network,", In *Proceedings of the* 2001 *Conference on Applications, Technologies, Architectures, and Protocls for Computer Communications*(SIGCOMM '01), pp.161-172, August 2001.

[8] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," In *Proceedings of the* 2001 *Conference on Applications, Technologies, Architectures, and Protocls for Computer Communications*(SIGCOMM '01), pp.149-160, August 2001.

[9] D. Johnson and D. Malt, Y. Hu, and J. Jetcheva, "The Dynamic Source Routing Protocol for Mobile Ad hoc Networks," Internet-Draft, draft-ietf-manet-dsr-07.txt, Feburary 2002.