IPv6用ニックネームプロトコル

市澤 浩史, 後藤 幸功, 村山 優子

岩手県立大学大学院ソフトウェア情報学研究科

Nickname Protocol for IPv6

Hiroshi Ichisawa, Yukinori Goto, Yuko Murayama

Graduate School of Software and Information Science, Iwate Prefectural University

概 要

Internet Protocol version 6(IPv6) には，IPv6 自身のプラグアンドプレイ機能により，ユーザが IP アドレスを設定する必要がない．ただし，名前解決までは行われない．名前解決には Domain Name System(DNS) を用いるのが一般的だが，一時的なネットワークを構築する場合に DNS サーバを運用するには困難である．また，仮りに DNS サーバを運用する場合は，自動で割り振られた IPv6 アドレスは，アドレス長が 128bit であるため，打ち間違いを起こしやすい．本稿では，IPv6 を用いたネットワークを運用する際に用いる名前解決の一つの方法としてニックネームプロトコルを提案する．ニックネームプロトコルはネットワーク上の各ホストが，IP マルチキャストを用いて，自らのニックネームを公開する．本稿では，IPv6 のためのニックネームプロトコルについて述べる．

1 はじめに

Internet Protocol version 6(IPv6)[1, 2] の標準機能には Plug-and-Play 機能がある．Plug-and-Play 機能とは，ホストに対して自動的に IP アドレスが割り当てられる機能である．IPv4 を用いてネットワークに接続する場合，Dynamic Host Configuration Protocol(DHCP)[3] サーバのような特別なサーバをネットワーク上に用意しないかぎり，ネットワークにホストを接続するユーザは，自ら IP アドレス，ネットマスクやルータのアドレスなどを設定しなければならなかった．しかし，IPv6 の Plug-and-Play 機能によって，DHCP サーバのような特別なサーバがネットワーク上に存在しなくとも，IP アドレスを設定する手間を省くことができる．IPv6 の Plug-and-Play 機能は，一時的に構築される Ad-hoc ネットワークには向いている機能である．一時的に構築される Ad-hoc ネットワークは，一般的なネットワーク環境と異なり，管理者が存在しないことが多い．そのため，DHCP サーバや Domain Name System(DNS)[4, 5] サーバなどのインターネット上で伝統的によく利用されるサーバを利用することが出来ない．このような Ad-hoc ネットワークであっても IPv6 の Plug-and-Play 機能により即座に Ad-hoc ネットワーク上の各ホスト間でネットワークが構築される．しかし，IPv6 の Plug-and-Play 機能だけでは円滑なコミュニケーションを行うことが困難である．その原因は名前解決 [6] の問題である．IPv6 アドレスは，もともと IP アドレスの枯渇を懸念して設計されたプロトコルである．膨大なアドレス空間を確保するため，IPv6 のアドレス長が 128bit である．IPv4 のアドレス長 32bit と比較すると非常に長い．IPv6 の長いアドレスを表記するために 16 進数が用いられる．それでも，最長 32 文字のアドレスとなる．例えば

ABCD:EF01:2345:6789:ABCD:EFFF:FE01:2345

のようなアドレスとなる．心理学には Magical Number Seven[7] という法則が存在する．複数個の事柄に関してなんらかの意味付けがされていない場合に，人間のワーキングメモリの許容量は 7(±2) の事柄を認識するのが限界であることを示した法則である．Plug-and-Play 機能によってホストに割り当てられる IPv6 のアドレスは，人間が聞いて即座に記憶できないアドレスであると言える．つまり，IPv6 のアドレスだけでは円滑なコミュニケーションを行うには困難であり，名前解決が必須である．しかし，DNS

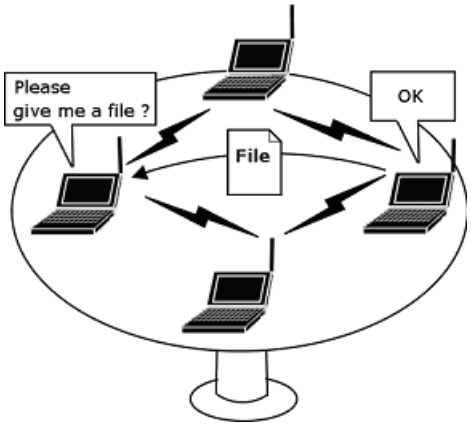

図 1: MANET の一例

サーバを一時的に設置することが困難な Ad-hoc ネットワーク上で円滑なコミュニケーションを行うためには，DNS サーバのような中央サーバを必要としない他の名前解決の方法が必要である．そこで本稿では，IPv6 のために設計されたニックネームプロトコルを提案する．ニックネームプロトコルが利用される環境はホスト数が多くとも 20 台程度を想定しており，単純かつ非階層型な名前が最適であると判断した．そこでインターネット上で伝統的に用いられる Fully Qualified Domain Name(FQDN) のような階層型の名前ではなく，短い名前 (ニックネーム) を用いることでホストが接続しているローカルネットワーク上で用いる非階層型な名前解決を提供する．ニックネームの公開には，中央サーバを必要とせず，IP マルチキャストを用いる．そして，ニックネームのデータベースはローカルネットワーク上の各ホストに分散して保存される．IPv6 とニックネームプロトコルを同時に利用することにより，一時的に構築された Ad-hoc ネットワークでも円滑なコミュニケーションを行える．このニックネームプロトコルの仕様と実装について述べる．

2　ニックネームプロトコルのモデル

図 1 のように 2 から 5 人程度のメンバがカフェテリアに集まり，各自ラップトップパソコンを持ち込んでミーティングを行う時がある．このような場合，無線 LAN のアドホックモードの機能を用いて，各ホスト間で一時的なネットワークを構築できる．このようなネットワークを Mobile Ad-hoc Network(MANET)[8] と呼ぶ．ホットスポットではない環境でも，メンバは MANET 上でファイル交換やスケジュールの同期などの作業などが行える．ただし，MANET を構築して，円滑なコミュニケーションを行うには二つの問題がある．一つはアドレスの設定である．IPv4 では IP アドレスの自動設定機能である Plug-and-Play 機能は存在しなかったために，ユーザが IP アドレスやネットマスクの設定を行う必要があった．IP アドレスを利用しない小規模ネットワーク用のプロトコルである Apple Talk[9] や NetBIOS[10] などを利用する方法も考えられるが，実装されていないクライアント用の Operation System(OS) もあり，確実に利用できるわけではない．Apple Talk や NetBIOS と比較して最も普及している IP を用いる方が確実にネットワークを構築することができる．IPv6 を用いれば，標準で Plug-and-Play の機能を持つので，IP アドレスを設定する手間を省略することができる．もう一つの問題は名前解決である．IPv6 にはアドレスの自動設定機能は存在するが，名前解決を行う機能は存在しない [11]．MANET は管理者の存在しないネットワークであるため，インターネット上で伝統的に利用されている DNS サーバのような管理者が管理を行っているサーバを用意するには困難である．また，MANET はトポロジも一般的なネットワークと比べて変化しやすく，安定したサーバクライアント型のサービスを行えない．そのため名前解決には，中央サーバを利用しない解決方法が必要である．サーバを利用しない解決方法として，伝統的な名前解決手法である IP アドレスと名前の静的な対応付けがある．IP アドレスと名前を対応付けする設定ファイルに各ホストの所有者であるユーザがそれぞれ静的に IP アドレスと名前を対応付けすることにより名前解決を行う．DNS が存在する以前によく利用されていた解決方法であり，現在普及率の高いクライアント用の OS として存在する Windows や MacOS や Linux などの OS にはすべて対応付け用のファイルが存在している．ネットワーク上にある全てのホストの IP アドレスと名前を対応付けファイルに記述してゆけば名前解決は可能だが，ここにも問題がある．自動で割り当てられた IPv6 のアドレスはアドレス長が 128bit ととても長く，書き間違えを起こしやすい．自らが所有しているラップトップパソコンに自動設定された IPv6 のアドレスを言葉で伝える時，非常に伝えにくい．提案するニックネームプロトコルは，DNS のような管理された中央サーバを必要とせず，IP マルチキャストを用いて伝統的な静的かつ非階層型な名前解決を支援するプロトコルである．ニックネーム

プロトコルを利用することで手軽に名前解決が行えるので，MANET のような一時的に構築されたネットワーク上での円滑なコミュニケーションを行える．

3 既存の名前解決手法との違い

インターネット上では，伝統的な名前解決の手法として DNS サーバを用いた階層型の名前解決を行ってきた．しかしながら，DNS は管理者によって管理されたネットワークで利用するシステムであり，一時的に構築されるために管理者が存在しない Ad-hoc ネットワークで利用するには困難である．MANET の場合はさらにトポロジの変化についても考える必要があり，DNS のような中央サーバを必要とするサーバクライアント型のサービスを提供するには困難である．そのため，小規模ネットワーク用の DNS 以外の別の名前解決手法が必要である．小規模ネットワークで利用する他の名前解決手法としては，NetBIOS over TCP/IP(NetBT) や MulticastDNS[12, 13] がある．NetBT は非 IP ネットワークとして 1984 年に IBM が開発した NetBIOS を TCP/IP 上で利用できるように改良したものである．現在の Windows にはこの NetBT が実装されており，この機能を利用して Microsoft Network を構築している．主にファイル共有サービスやプリンタ共有サービスなどに利用されている．NetBT では NetBIOS Nam と言う名前を利用する．この NetBIOS Name は名前の長さが 15 文字までと制限がある．MulticastDNS は IEFT の Zero Configuration Networking(zeroconf) ワーキンググループと DNS Extensions(dnsext) ワーキンググループが提案する小規模ネットワークのための名前解決手法である．DNS のメッセージを IP マルチキャストを用いて送信することで名前解決を行う．DNS Service Discovery(DNS-SD)[14] と併用して利用することで，ネットワーク上のサービスを探しだし，そのサービスを利用することを目的としている．現在は主に MacOS の Bonjour[15] に利用されており，プリンタを探しだすときなどに利用されている．これらシステムは，名前解決以外にネットワーク上のサービスを提供したり，検索したりすることを目的としている．特に zeroconf に関わりがある MulticastDNS は，Service Location Protocol(SLP)[16] や Jini[17] や Intentional Naming System(INS)[18] などのシステムと同じように，サービスロケーションを支援する目的が大きい．これらのシステムに比べ，本稿で提案するニックネームプロトコルは Service Location を目的とするプロトコルではなく，ローカルネットワーク内の非階層型な名前解決だけに特化したシステムである．そのため，行っている作業が単純であるため，実装が容易である利点がある．

4 ニックネームプロトコル

提案するニックネームプロトコルについて詳述する．はじめにニックネームの仕様について述べ，その後，メッセージタイプとニックネームプロトコルの機能について述べる．

4.1 ニックネームの仕様

IPv6 のためのニックネームには，以下のような特徴がある．

- ニックネームの長さは 1〜255 文字
- 特定のアプリケーションに依存しない
- DNS など他の名前解決手法と併用して利用できる

1 節で説明しているように，ニックネームプロトコルが利用される環境はホスト数が多くとも 20 台程度を想定している．このような環境では伝統的な FQDN のような階層型の名前ではなく，非階層型で短い名前が最適であると判断した．階層型の名前は，大規模なネットワークで名前を管理する場合に効果を発揮する．名前を階層化することにより，ネットワークを論理的に分類できる利点がある．また名前の管理するデータベースも階層化することにより，名前の管理を容易することができる．しかしながら，一時的に構築する Ad-hoc ネットワークの場合には，階層型の名前を利用する必要はない．本稿で対象としている Ad-hoc ネットワークの構成はルータによって階層化されたネットワークではなく，また規模も小さいネットワークである．そのため論理的に分類するほどホストがネットワーク上になく，分類する必要性が低い．Ad-hoc ネットワークには管理者が存在しないので，データベースを階層化して管理する利点も同様に低い．このようなニックネームを各ホストは IP マルチキャストを用いてネットワーク上に公開する．各ホストはそれぞれニックネームのためのデータベースを持つ．ネットワーク上に公開されたニックネームはすべてそのデータベースに記録する．同時に IP アドレスと対応付けを名前を行う対応付け

表 1: N6 で用いるメッセージ

名前	説明
ADD	ニックネームの新規登録
DELETE	ニックネームの削除
LISTQUERY	リストの要求
ERROR	エラー情報の送信

ファイルへの設定も行われる．ニックネームプロトコルは伝統的な静的な名前解決を支援するシステムであるため，当然名前は特定のアプリケーションに依存せず，DNS など他の名前解決手法と併用して利用することができる．

4.2 メッセージタイプ

ニックネームプロトコルは 4 つのメッセージを利用してニックネームを公開する．

ADD メッセージ ADD メッセージは，公開したいニックネームを伝えるメッセージである．メッセージの内容はニックネームと IPv6 アドレスとホストを特定するためのハッシュ値を組みにした情報である．新しくニックネームを公開するホストは ADD メッセージを IP マルチキャストを用いて送信し，各ホストに伝える．また，ADD メッセージは定期的に送信する．登録されているニックネームが有効であることを各ホストに伝えるキープアライブとしての役割も持つ．

DELETE メッセージ DELETE メッセージは，削除するニックネームを伝えるメッセージである．メッセージの内容は削除対象のハッシュ値である．自らが公開しているニックネームを削除したいホストは DELETE メッセージを IP マルチキャストを用いて送信する．

LISTQUERY メッセージ LISTQUERY メッセージは，現在のニックネーム状況を他のホストに問い合わせる時に利用するメッセージである．LISTQUERY メッセージを受信したホストは，自分が保持しているデータベースの情報を ADD メッセージを用いて，LISTQUERY メッセージを送信したホストに伝えなければならない．

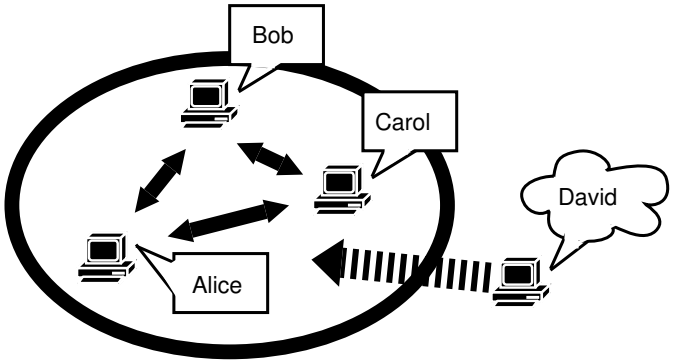

図 2: ニックネームプロトコルのモデル

4.3 ニックネームプロトコルの機能

ニックネームプロトコルは，ネットワーク上のホストがお互いに非階層型かつ短い名前 (ニックネーム) を公開することによって，お互いの名前を認識し，名前解決を行う．前提として図 2 のように，ネットワーク上には現在 Alice，Bob，Carol のホストが 3 台存在しており，各ホストはニックネームプロトコルを用いて名前解決を行っているとする．その後 David が新たにニックネームを公開する．その後，Bob が名前の公開を停止する．このような場合を例にして説明する．

ニックネームの登録 ネットワークに新しく接続するホストは，これから接続するネットワーク上で現在利用可能なニックネームのリストを持っていない．このホストが初めに行わなければならないことは，ニックネームのリストを取得する事である．自らのデータベースにニックネームの状況を記録する目的と同時に，公開したいニックネームが既に利用されていないか確認するためである．図 2 の例では，新たに David と名乗る予定のホストが，David という名前が既に利用されていないか確認を行わなければならない．そのため LISTQUERY メッセージを用いてリストの要求を行う．LISTQUERY メッセージはネットワーク上の全てのホストに伝わるが，この要求に対する返答を行うには 1 台のホストのみである．この 1 台は通常ネットワーク上で最後にニックネームを公開したホストが担当する．この理由は最後に ADD メッセージを送信をしたホストがネットワーク上で最も生存している可能性が高いホストだからである．ただし，返答を行うはずのホストが落ちている可能性も考えられるので，LISTQUERY メッセージ内に ADD メッセージで公開した順番の後ろから 2

番目のホストなどリスト返信要求先を変更することもできる．LISTQERY メッセージを送信するホストはリストがもらえるまで，リストを返信してもらう対象を変えながら数回繰り返す．新規ニックネーム公開ホストは受信したニックネームのリストを自らのデータベースに反映させる．同一セグメント内に同じニックネームを持つホストが存在してはいけない．そのため，ニックネームの公開は，先にニックネームを公開していたホストにニックネームの優先権がある．先ほど受信したリストから，あらかじめ公開したいニックネームが既に利用されていないかどうか確認する．その後，公開したいニックネームが既に利用されていなければ，ニックネームと IPv6 アドレスを組みにして IP マルチキャストを用いて送信する．各ホストは，新たに公開されたニックネーム情報を受信し，それぞれ自らのデータベースに登録する．例では，Alice が David に対してリストを送信したとする．リストを受信した David はネットワーク上に Alice と Bob と Carol の 3 台のホストの存在を確認する．その後 David は自分に名前をこの 3 台のホストに伝える．3 台のホストは David の存在を確認し，それぞれのデータベースに情報を書き加える．

キープアライブ　同一セグメント上の各ホストはお互いのデータベースの情報に誤りが無いことを証明するために定期的にニックネームのキープアライブを行わなければならない．各ホストは一定間隔ごとに ADD メッセージを送信する．ADD メッセージを受信したホストは，各自のデータベースの受信したニックネーム情報に対して生存確認を示す印をつけておく．例では 4 台のホストがそれぞれ定期的に ADD メッセージを送信しあい，お互いの存在を確認し続けている．

ニックネームの削除　ニックネームを削除する場合は，ニックネームを公開する時と同様に，ホストはニックネームの削除情報をマルチキャストで送信する．DELETE メッセージを受信したホストは，各自のデータベースの受信したニックネーム情報に対して生存確認を示す印を外しておく．例では Bob が DELETE メッセージを送信する．Bob 以外の 3 台はそのメッセージを受信し，生存確認の印を外しておく．

データベースチェック　各ホストは同一セグメント内のニックネームの状況が記録されたデータベースを持つ．各ホストはキープアライブの間隔よりも長い一定間隔ごとに．データベースの内容をチェックする．このチェックの際に，生存確認を示す印のなかったニックネームの情報をデータベースから削除する．このように DELETE メッセージを受信してすぐデータベースの書き換えを行わない理由は，頻繁に各ホストが保持しているデータベースの書き換えを防ぎ，できる限りデータベースが不安定な期間を短くするためである．図 2 の例は，Bob 以外のホストがそれぞれのデータベースチェックの時に，Bob の情報に生存確認の印が無いことを確認して，データベースから Bob の情報を消す．

5　N6:プロトコルの実装

本稿では Ruby 言語 [19] を用いてニックネームプロトコルのプロトタイプを実装した．プロトタイプの名前を N6 と名付けた．Ruby 言語を採用した理由は以下の 3 点である．プロトタイプの実装であるため，実行速度を求めないこと．インタプリタ型の言語であるためデバッグを行い易いこと．そして，オブジェクト指向の機能を持つ言語であるため，設計仕様に沿った実装を行い易いことである．N6 は，Ruby が動作する FreeBSD や Linux 上での動作を確認している．

N6 の動作を説明する．説明の前提として，同一セグメント上に N6 を起動しているホストは存在していないとする．ニックネームの公開の流れを図 3 に示す．ユーザ A は，公開したいニックネームを入力する．ホスト A は LISTQUERY メッセージを IP マルチキャストを用いて送信する．ホスト A は標準設定で 3 回，LISTQUERY メッセージを送信し，その間にリストの受信を待つ．しかし，現在同一セグメント上で N6 を起動しているホストはホスト A だけであるため，ホスト A はリストを受信できない． この後に ADD メッセージをマルチキャストを用いて送信する．公開されたニックネームは，各ホストが持つデータベースと Unix において名前登録に利用する /etc/hosts の二つにニックネーム情報が書き込まれる．/etc/hosts は IP アドレスと名前を静的に関連付けるための設定ファイルである．

ユーザ B が新たにニックネームを公開する場合もユーザ A が行った公開の流れと同様の流れである．ただ，ホスト A は既に N6 を用いてニックネームを公開しているので，ホスト B からの LISTQUERY メッセージを受けたホスト A は，ニックネーム情報のリストを ADD メッセージを用いてホスト B に送

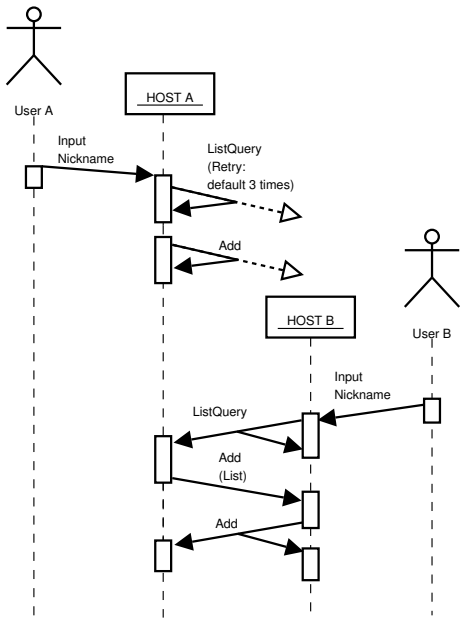

図 3: ニックネーム公開の流れ

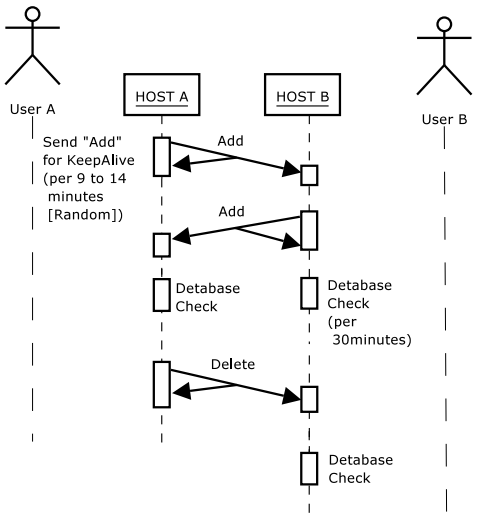

図 4: ニックネーム削除の流れ

らなければならない．ホスト B はホスト A から受信したリストから，ユーザ B が公開したいニックネームが既に使われていないかどうか確かめる．確かめて，利用されていなければ，ADD メッセージでニックネームの公開を行う．ニックネームの削除の流れを図 4 に示す． N6 では，各ホストは 30 分間隔でデータベースチェックを行う．このデータベースチェックまでに一度も ADD メッセージの受信を確認できなかったニックネームはデータベースから削除される．自らが公開するニックネームが有効である事を証明するために，各ホストは ADD メッセージを，次回のデータベースチェックまでに 2 回から 3 回程度送信しなければならない．そのため ADD メッセージの送信間隔は 1 分間隔で 9 分から 14 分の間の時間から任意に選択して送信する．任意に時間を設定する理由は同一セグメント内の ADD メッセージを一定時間内に集中させないためである．

6 評価

一般的な Ethernet を用いて実験的に作成したネットワーク内でニックネームプロトコルを使用し，メッセージの発生間隔とメッセージの量について測定を行った．図 5 から図 8 は，各ホストのデータベース更新間隔である 30 分内に発生するトラフィックの分布である．各ホストは既にお互いにニックネームを公開し終わり，キープアライブを行っている状態である．ニックネームプロトコルが用いられる環境は，同時に接続するコンピュータの台数が 20 台以下を想定している．そこでネットワーク上にホストが 5 台から 20 台ある状態を計測した．

表 2: トラフィックの総量とパケットの総数

	5 台	10 台	15 台	20 台
Traffic	1013byte	2018byte	2951byte	3872byte
Packet	11	22	32	43

30 分内に 2 カ所，トラフィックが発生している箇所がある．この箇所がニックネームのキープアライブの為に発生するトラフィックである．今回計測した中では一分間に発生した最大トラフィック量は約 450byte 程度であった．表 2 でも表されているように，30 分間に 5 台のホストにつき，約 1Kbyte のトラフィックが発生する．仮に 10Mbps 程度のネットワーク環境であったとしても，ネットワークへの影響は低い．

7 考察

本稿で提案するニックネームプロトコルを実装した N6 と 3 節で取り上げた NetBT と MulticastDNS の 2 つの既存の小規模ネットワーク向け名前解決手法との比較を行う．表 3 に比較を行った表を挙げる．いずれの解決手法も小規模なネットワークを対

表 3: 小規模ネットワーク用名前解決手法の比較

名前	N6	MutilcastDNS	NetBT
接続ホスト数	2 から 20 台程度	2 から 20 台程度	2 から 20 台程度 (サーバを利用した場合は別)
サーバの有無	無し	無し	利用する場合もある
IPv4 対応	×	○	○
IPv6 対応	○	○	×
文字数の制限	255 文字まで	255 文字まで	15 文字まで
解決方法	対応付けファイル	DNS メッセージ	NetBIOS メッセージ (もしくは対応付けファイル)
ブラウズ機能	○	×	○

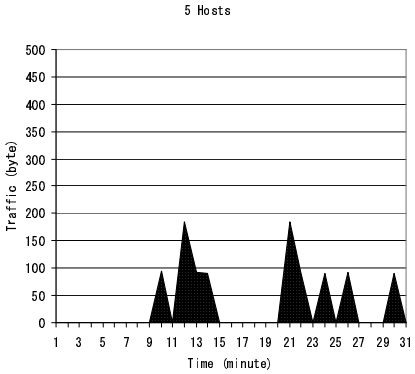

図 5: ホスト台数が 5 台の場合の 30 分内に発生するトラフィックの分布

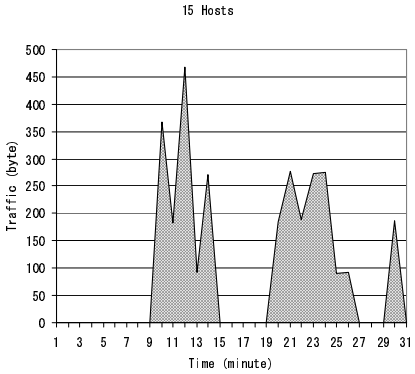

図 7: ホスト台数が 15 台の場合の 30 分内に発生するトラフィックの分布

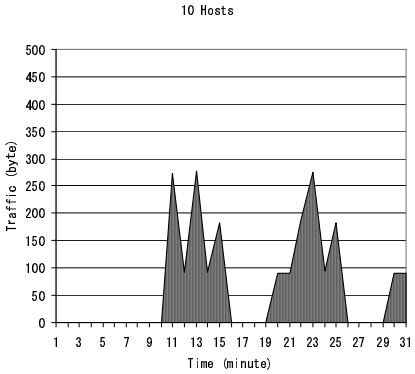

図 6: ホスト台数が 10 台の場合の 30 分内に発生するトラフィックの分布

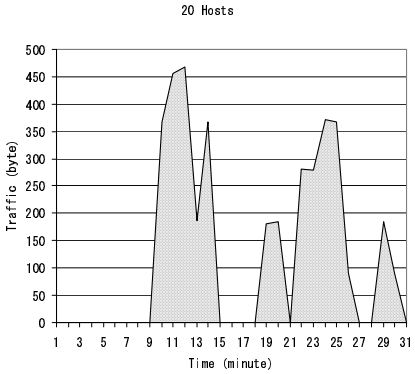

図 8: ホスト台数が 20 台の場合の 30 分内に発生するトラフィックの分布

象としているので，接続ホスト数はさほど変わらない．ただし NetBT は別途 Windows Internet Name Service(WINS)[20] サーバを用意することで，名前管理可能な台数も利用範囲も大幅に拡張できる．NetBT は三つの手法の中で唯一サーバを利用する場合がある手法である．対応する IP に関しては，MulticastDNS が IPv4，IPv6 共に利用可能となっている．ただし，ホストに IPv4 と IPv6 両方のアドレスが振られている場合，IPv4 アドレスが優先して返答される．N6 は IPv6 の問題点を指摘して制作しているプロトコルであるので IPv6 にしか対応していない．逆に NetBT は現在のところ IPv6 での操作が定義されておらず，IPv6 での名前解決を行うことはできない．利用可能な名前の文字数の長さは，NetBT が他の二つに比べ，圧倒的に短い．これは NetBIOS Name の仕様によるものである．NetBT では，場合によっては自分の付けたい名前を付けられない場合がある．名前解決の方法としては，N6 は伝統的な静的な対応付けを支援するプロトコルであるため，各 OS に付属している対応付けファイルを利用して名前解決を行う．MulticastDNS は DNS を用いた名前解決と同様に DNS メッセージを用いて解決する．DNS と MulticastDNS の違いは DNS メッセージをユニキャストで送信するかマルチキャストで送信するかの違いだけである．NetBT は NetBIOS 独自のメッセージを用いて解決する．もしくは NetBIOS 専用の対応付けファイルを利用して名前解決を行う．N6 は対応付けファイルを利用して名前解決を行っているため，DNS メッセージよりも優先される．そのため，もし N6 と同時に MulticastDNS もしくは DNS を用いた場合は，N6 によって公開されたニックネームが優先される．最後に公開された名前の一覧を表示するブラウズの機能について述べる．N6 はあらかじめ接続中のホストの一覧が記録してあるリストを取得する．そのリストから，ホストの一覧はいつでも確認できる．NetBT はキャッシュ済みのホストの情報を一覧にして表示する機能が実装されている．IPv6 を利用してネットワークを構築する場合は，N6 か MulticastDNS を利用した名前解決が可能である．

8 まとめ

本稿では IPv6 を用いた小規模ネットワークで利用するニックネームを用いた名前解決手法であるニックネームプロトコルを提案し，プロトタイプである N6 の実装を行った．IPv6 には標準機能として IP アドレスの自動設定を行う機能である Plug-and-Play 機能を持つ．この Plug-and-Play 機能を用いることで，IPv4 を用いた場合に比べ，IP アドレスやネットマスクなどの設定を行う必要がなくなった．そのため，一時的なネットワークの構築が容易になった．しかしながら，IPv6 の Plug-and-Play 機能だけでは円滑なコミュニケーションを行うには困難である．IPv6 アドレスは非常に長く，人間が聞いて即座に覚えられないアドレスである．一時的に作成したネットワークには管理者が存在しないことがある．管理者が存在しない環境で DNS の運用を行うには困難である．また MANET ではトポロジが変化しやすく，伝統的なサーバクライアント型のサービスを行えない．そこで中央サーバを必要としない名前解決が必要である．この問題を解決するためにニックネームプロトコルを提案した．ニックネームプロトコルでは伝統的な IP アドレスと名前を対応付けする静的な名前解決をサポートするシステムである．ニックネームプロトコルは，利用される環境は 20 台程度を想定しており，単純かつ非階層型な名前を用いて名前解決を行うシステムである．ニックネームを用いることで非階層型な名前解決が可能となった．ニックネームプロトコルと IPv6 の Plug-and-Play 機能によって円滑なコミュニケーションを行うことができる．N6 がネットワークに与える影響を測るため，メッセージの発生間隔とメッセージの量について測定を行った．N6 が，データベース更新間隔である 30 分間にトラフィックにあたえる影響は，1 台あたり 200byte 程度であった．ニックネームプロトコルが想定するネットワーク接続台数が 20 台の環境で，さほどネットワークに与える影響は低い．現在は IPv6 を専門にした名前解決の手法であったが，IPv4 への応用を考えている．また，スケーラビリティの向上や，ニックネーム公開範囲のグループ化や，さらなる IPv6 に特化した機能を追加することを目的として，ニックネームプロトコルの仕様を改良する予定である．

参考文献

[1] S. Deering and R. Hinden; Internet Protocol, Version 6 (IPv6) Specification; RFC 2460, December 1998.

[2] T. Narten and R. Draves; Privacy Extensions for Stateless Address Autoconfiguration in IPv6; RFC 3041, January 2001.

[3] R. Droms; Dynamic Host Configuration Protocol; RFC2131, March 1997.

[4] P. Mockapetris; DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION; RFC1035, November 1987.

[5] P.V. Mockapetris and K. Dunplap; Development of the Domain Name System; Proceedings of SIGCOMM'88, August 1988.

[6] John F. Shoch; Inter-Network Naming, Addressing, and Routing; In Proceedings of the Seventeenth IEEE Conference on Computer Communication Networks, pages 72-79, Washington, D.C., 1978.

[7] Miller, G. A.; The magical number seven, plus or minus two: some limits on our capacity for processing information; Psychological Review; vol. 63, pp. 81-97, (1956).

[8] Internet Engineering Task Force. MANET WG Charter; http://www.ietf.org/html.charters/manet-charter.html

[9] AppleTalk Network System Overview Addison-Wesley, Reading, MA; 1990.

[10] PROTOCOL STANDARD FOR A NetBIOS SERVICE ON A TCP/UDP TRANSPORT:CONCEPTS AND METHODS; RFC1001, March 1987.

[11] P. Huck, M. Butler, A. Gupta and M. Feng; A self-configuring and self-administering name system with dynamic address assignment; ACM Transactions on Internet Technologies, Volume 2, Issue 1:14 - 46; February 2002.

[12] S. Cheshire and M. Krochmal; Multicast DNS; http://files.multicastdns.org/draft-cheshire-dnsext-multicastdns.txt, June 2005.

[13] M. Watanabe, K. Takaya, A. Seo, M. Hashimoto, T. Izumida and A. Mori; A scheme of service discovery and control on ubiquitous devices; Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters; pp. 322-323, (2004).

[14] S. Cheshire and M. Krochmal; Requirements for a Protocol to Replace AppleTalk NBP; http://files.dns-sd.org/draft-cheshire-dnsext-nbp.txt, June 2005.

[15] Bonjour Network Services; http://developer.apple.com/documentation/Cocoa/Conceptual/NetServices/

[16] E. Guttman and C. Perkins; Service Location Protocol, Version 2; RFC2608, June 1999.

[17] Jini.org; http://www.jini.org/

[18] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan and J. Lilley; The design and implementation of an intentional naming system; Operating Systems Review, 35(5):185-201; December 1999.

[19] Y. Matsumoto; Ruby in a Nutshell.First Edition; November 2001.

[20] Windows Internet Naming Service (WINS): Architecture and Capacity Planning; http://www.microsoft.com/ntserver/techresources/commnet/WINS/WINSwp98.asp; September 1998.