NTMobileにおける組み込み機器向けトラフィック削減手法の提案

杉原 史人[1,a)] 内藤 克浩[2] 鈴木 秀和[3] 渡邊 晃[3] 森 香津夫[1] 小林 英雄[1]

概要：Machine to Machine (M2M) 通信は，今後の情報化社会を支える通信サービスとして注目されている．IPv4 アドレスの枯渇は知られており，今後は IPv4 と IPv6 アドレスが M2M デバイスの普及とともに利用されると考えられる．しかし，IPv4 と IPv6 間には互換性がないことから，M2M 通信における接続性と相互接続性は新たな課題になると考えられる．著者らは IPv4/IPv6 の混在環境において通信の接続性と移動透過性を同時に実現する Network Traversal with Mobility (NTMobile) を提案してきた．NTMobile では，IPv4 ネットワークで頻繁に利用される NAT の変換表内にある NTMobile の接続情報を維持するために，User Datagram Protocol (UDP) ベースの Keep-Alive パケットを定期的に送信していた．既存方式では，一定時間内にパケットの送信を繰り返す必要があるため，組み込み機器などで利用される安価な無線パケット通信サービスを利用することが困難である．そこで，本研究では NTMobile に NTMobile Notification Service (NNS) を導入する．NNS では，NTMobile ノードと NNS Server 間で Transmission Control Protocol (TCP) のコネクションを構築することにより，端末とサーバー間の通信を維持する．また，TCP を用いた場合，NAT の変換表の接続情報の維持時間が UDP と比較して長いことから，Keep-Alive パケットの送信頻度を大幅に間引くことが可能となり，接続情報維持を目的としたトラヒックの大幅な削減が可能となる．本研究では，提案方式を Linux 上に実装を行うことで，提案方式の有効性を実験により評価を行う．

1. はじめに

近年の無線通信技術の発達に伴い，すべてのモノがインターネットに接続され，情報交換を行う仕組みの Internet of Things (IoT) 及び機器同士が通信ネットワークを介して接続され，自律的に情報交換を行う Machine-to-Machine (M2M) は，今後の情報化社会を支える仕組みとして注目されている．IoT や M2M の基盤技術には Internet Protocol (IP) が利用されることが予想される．IPv4 アドレスはすでに枯渇しており，IPv6 の普及し始めている [1]．しかし，IPv6 を使用するためには IPv6 に対応した機器を用意する必要があるため，IPv6 への移行は容易ではなく，しばらくは IPv4 と IPv6 の共存環境が続くと考えられる [2]．IPv4 と IPv6 間の接続性を確保する技術は，「トンネリング」「トランスレータ」「デュアルスタック」の 3 方法に大別される．トンネリングには ISATAP，Teredo，6rd などがあり [3], [4], [5]，トランスレータには NAT-PT，NAT64/DNS64 などがある [6],[7]．また，接続性を確保したうえで移動透過性を実現するものとして DSMIPv6 がある [8]．DSMIPv6 は IPv4/IPv6 混在環境を想定した方式であり，標準化も終えている有望な方式である．しかし，DSMIPv6 を用いたとしても，グローバルアドレス確保の困難性やホームエージェントが通信ボトルネックとなる可能性が高いなどの課題が，特に IPv4 において残されている．

著者らは IPv4/IPv6 の混在環境において，確実な接続性と移動透過性を実現可能な技術として，Network Traversal with Mobility (NTMobile) を提案してきた [9], [10], [11], [12]．NTMobile では，NTMobile の機能を実装した端末 (以下 NTMobile ノード) が仮想 IP アドレスを用いて通信を行うことにより，実 IP アドレスが変化した場合にも，接続性と移動透過性を実現可能である．また，仮想 IP アドレスを用いた IP データグラムを User Datagram Protocol (UDP) トンネルを通して交換しているため，NAT などが存在する場合でも，NAT 外部からの

[1] 三重大学大学院工学研究科電気電子工学専攻
Department of Electrical and Electronic Engineering, Mie University, Tsu, Mie 514-8507, Japan
[2] 愛知工業大学情報科学部
Faculty of Information Science, Aichi Institute of Technology, Toyota, Aichi 470-0392, Japan
[3] 名城大学大学院理工学研究科
Graduate School of Science and Technology, Meijo University, Nagoya, Aichi 468-8502, Japan
[a)] fsugihara@com.elec.mie-u.ac.jp

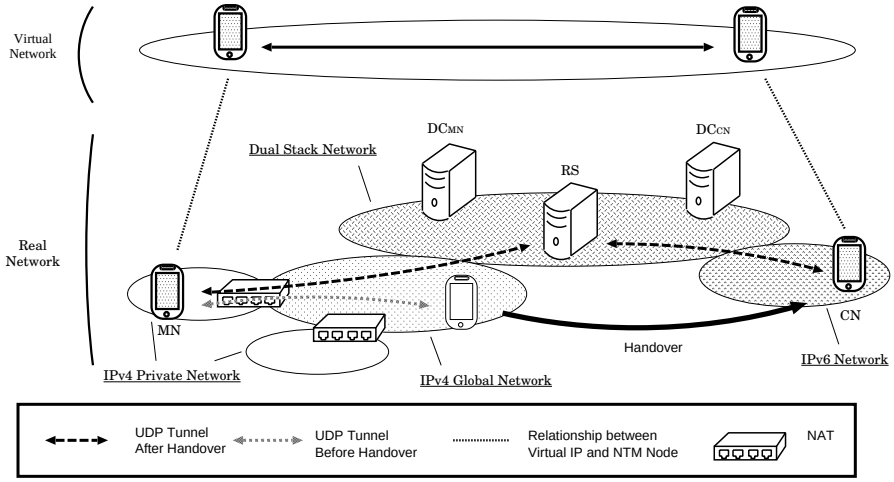

図 1 既存 NTMobile のシステムモデル

接続性も担保可能である．また，NTMobile ノードが直接通信可能な場合には，UDP トンネルを直接構築することにより，冗長な経路を防いでいる．そして，各 NTMobile ノードが利用するプロトコルバージョンが異なるなど，直接通信が不可能な場合には，リレー機能を持つサーバーを経由して UDP トンネルを構築する．

IPv4 ネットワークでは，アドレス数とセキュリティの観点からインターネットと内部ネットワークの間に Network Address Translation (NAT) と呼ばれるアドレス変換装置を設置し，内部ネットワークにおいてプライベートアドレスを利用することが一般的である．内部ネットワークに接続した端末からから外部ネットワークであるインターネットに接続することは可能である．しかし，NAT を利用した内部ネットワークは外部ネットワークであるインターネットから隠蔽されるため，インターネットに接続した端末から内部ネットワークに接続することができない．この問題は NAT 越え問題と呼ばれ，エンドエンド間の接続性をそこなう要因となっている．

NTMobile では，IPv4 ネットワークで頻繁に利用される NAT の変換表内にある NTMobile の接続情報を維持するために，UDP ベースのパケットを定期的に送信していた．この方式では，一定時間内にパケットの送信を繰り返す必要があるため，組み込み機器などで利用される安価な無線パケット通信サービスを利用することが困難である．また，ネットワーク上のトラフィックを増大させる要因にもなる．そこで，本研究では，NTMobile に NTMobile Notification Service (NNS) を導入する．NNS では，NTMobile ノードと NNS Server 間で TCP のコネクションを構築することにより，端末とサーバー間の通信を維持する．また，TCP を用いた場合，NAT の変換表の接続情報の維持時間が UDP と比較して長いことから，パケット送信頻度を大幅に間引くことが可能となり，接続情報維持を目的としたトラヒックの大幅な削減が可能となる．本研究では，提案方式を Linux 上に実装を行うことにより，提案方式の有効性を実験により評価を行う．

2. NTMobile の概要

図 1 に NTMobile のシステム概要を示す．NTMobile は，NTMobile ノードのほかに NTMobile ノードのアドレス情報の管理やトンネル構築の処理を行う Direction Coordinator (DC)，NTMobile ノードが直接通信を行うことができない場合に通信を中継する Relay Server (RS) から構成される．NTMobile ノードには NTMobile ネットワーク内で，一意に識別可能な仮想 IP アドレスを割り当てることにより，アプリケーションは仮想 IP アドレスに基づいた通信を行う．そのため，実 IP アドレスの変化した場合にも，アプリケーションは仮想 IP アドレスを利用して通信を継続可能である．仮想 IP アドレスを用いた IP データグラムは NTMobile のカーネルモジュールによりカプセル化され，UDP トンネルを通して交換される．トンネルの通信経路は 2 種類あり，原則としては両ノード間で直接トンネルを構築し，プロトコルバージョンが異なる場合などの両ノード間が直接通信ができない場合は，RS を経由するトンネルを構築することで通信が可能となる．NTMobile ではアプリケーションは仮想 IP アドレスを利用することから，相手側のネットワークを意識することがなく，相互通信が可能となる．また，NTMobile は汎用的な技術であることから，Linux 搭載の PC，スマートフォンへの実装を終え，現在組み込み機器への実装を進めている．

- Direction Coordinator (DC)
  NTMobile ノードの位置情報などを管理し，NTMobile ノードにトンネル構築に関わる各種処理の指示を出す装置である．各 DC は自身に割り当てられた仮想 IP アドレス空間を管理し，重複のないように NTMobile ノードに対して仮想 IP アドレスを割り当てる．また，DC は自身のデータベースに NTMobile ノードの Fully

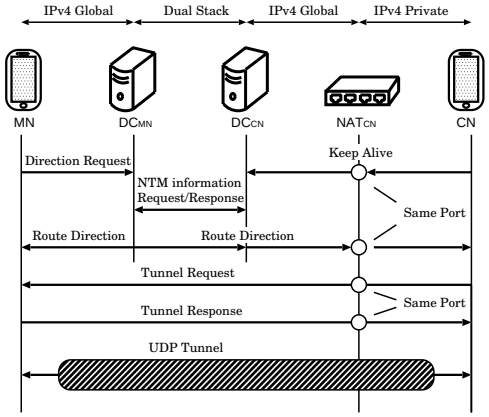

図 2 直接トンネル構築時の手順

Qualified Domain Name (FQDN)，実 IP アドレス，仮想 IP アドレス，NAT の外側の実 IP アドレスとポート番号を記録している．DC は Domain Name System (DNS) の機能をもっており，DC ごとに異なるドメインを管理する．NTMobile では DC を探索するために DNS の機構を利用しているため，分散配置が可能である．NTMobile ノードは IPv4 および IPv6 を利用する可能性があるため，DC はデュアルスタックネットワークに設置されることを想定する．

- Relay Server(RS)
  NTMobile ノードが NAT 越え問題や IPv4/IPv6 ネットワークの混在により，直接通信を行うことができない場合に，通信の中継を行う．また，RS は DC により管理されており，DC の指示により中継処理を行うため，多数の RS を用いた分散処理も実現可能である．DC 同様に，RS は IPv4 および IPv6 間の中継も行うため，RS はデュアルスタックネットワークに設置されることを想定する．
- NTMobile ノード
  アプリケーションが利用する仮想インタフェースを持ち，仮想インタフェースには DC から割り当てられる仮想 IP が設定されることにより，アプリケーションは仮想 IP を用いた通信を行う．また，仮想 IP を用いた IP データグラムは UDP によるカプセル化されることで，実 IP アドレスの変化を隠蔽している．NTMobile ノードは実 IP アドレスが変化した場合には，自身の DC に新たな実 IP アドレスの登録を行うことで，移動透過性と接続性を実現している．

2.1 トンネル通信

NTMobile で行われるトンネル通信には，直接トンネルを構築する場合と RS にトンネルを構築する場合の 2 種類がある．図 2,3 に 2 種類のトンネル通信を行う様子を示す．基本的には直接トンネルを構築し，プロトコルやアドレス空間の違いにより直接トンネルを構築するのが困難な場合には RS を経由するトンネルを構築する．両者のトンネル構築処理では，トンネル構築の要求を送信する先が違うのみで，ほかの手順は同じである．NTMobile ノードは起動時に DC に実 IP アドレスなどを登録するための登録要求である Registration Request を送信する．DC は NTMobile ノードからの要求を処理し，登録完了を示す Registration Response を NTMobile ノードへ返す．その後，NTMobile ノードは通信要求が発生するか，受け取るまで待機する．アプリケーション側から通信要求があると，相手との通信をするための経路を知るための要求である Direction Request を DC に送信する．DC は通信相手を探索することで経路通知である Route Direction を通信要求の発生した NTMobile ノードと通信相手に送信する．経路通知を受信した両ノードはトンネルを構築するための Tunnel Request を通信相手または RS に送信する．Tunnel Response が返信されることでトンネル構築を確認し，トンネルを使用した通信を開始する．

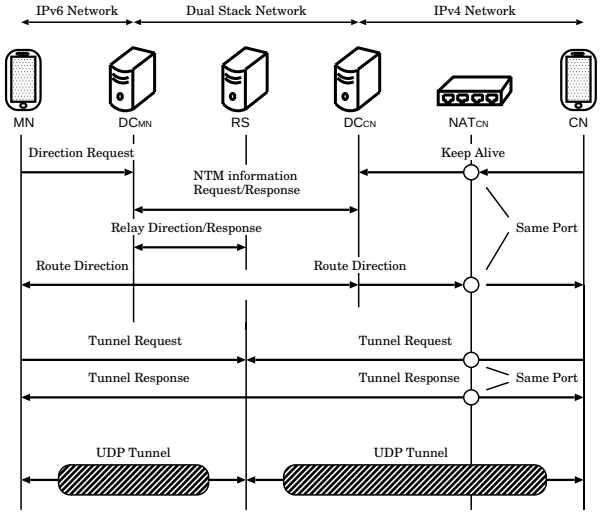

図 3 RS 経由のトンネル構築時の手順

3. NAT 越え問題

IPv4 ネットワークでは，IP アドレスの枯渇とファイヤーウォール機能を導入するために，NAT が頻繁に利用されている．近年の機器は NAPT と呼ばれるアドレスとポート番号を変換する方式が主流であり，変換テーブルを用いて内部ネットワークと外部ネットワークの通信を中継する．変換テーブルは有限領域であるため，一般的には何らかの規則で登録されている情報を削除していく．

変換表内コネクション情報のタイムアウト時間は任意に設定可能であるが，RFC5382 で NAT の変換テーブルについて記述があり，TCP の変換テーブルの保持時間を 2 時間 4 分にすべきとしている．これは，TCP の Keep-alive の周期が 2 時間に設定されている点とも整合する．しかし，商用ルータの実装などを確認すると，変換テーブル領域の都合から 1 時間などのより短い時間を設定するものもあり，

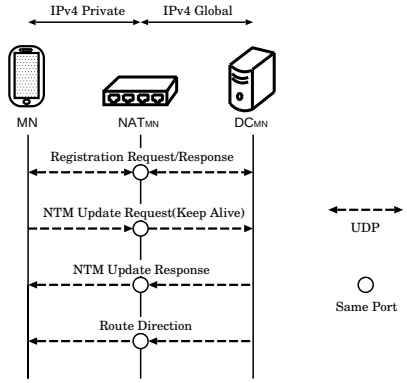

図 4　通信相手先の呼び出し手順 (既存方式)

NTM Header

NTMobile ID			
Version	Msg. Type	Flags	Count
Transaction ID		Sequence No.	
Meg. Length		Reserved	
Sender's Node ID / Path ID *			

* Only if Msg. Type is capsulated packet

図 5　NTMobile のヘッダフォーマット

Ethernet	IP	UDP	NTM Header
14 bytes	20 bytes	8 bytes	32 byte

図 6　NTMobile Update のパケットフォーマットとサイズ

コネクション切断の問題を生じることが知られている．

UDP については，RFC4787 によると変換テーブルの保持時間を 5 分としている．しかし，商用ルータの実装では，1 分などのより短時間の設定のものもある．そのため，TCP 以上に頻繁に通信を行わない場合，変換テーブルからコネクション情報が削除される．

上記の通り，NAT ルータは枯渇しているグローバル IP アドレスへの対応とセキュリティ対応で有効である一方で，特に UDP を用いた通信には，変換テーブル内のコネクション情報維持のために，多大なトラヒックを生じさせるという問題もある．

4. Keep-Alive を用いる既存 NTMobile

図 4 に既存の NTMobile で採用している Keep Alive 方式を示す．DC は NTMobile ノードへの通信要求が発生した場合，該当 NTMobile ノードを呼び出す必要がある．NTMobile ノードが NAT 配下のネットワークに属している場合，NAT 越え問題があるため DC と NTMobile ノードの間にある NAT の変換テーブルを維持する必要がある．そこで，NTMobile ノードは，定期的に NTMobile で利用する UDP ポートを利用して，Keep Alive 動作を行う．一般に NAT における UDP の情報保持時間は数十秒から数分と短く，30 秒ほどの間隔での Keep Alive 動作が必要である．既存方式の Keep Alive で利用される NTM ヘッダとパケットフレームを図 5,6 に示す．既存の Keep Alive 方式では NTMobile ノードは NTMobile ヘッダを送信し，DC では NTMobile ヘッダ中のメッセージタイプを確認することで Update パケットの判定を行う．一回の Keep Alive で利用されるパケットは 70byte と小さいため，ブロードバンド回線などを利用している場合には，大きな負荷にはならない．一方で，組み込み機器は 3G 網などの無線携帯網を利用することもあり，パケット課金方式が利用されていることも多い．そのため，既存の NTMobile の Keep Alive 方式では，高額な通信費が必要となり，実用上のデメリットとなり得る．また，組み込み機器は数十から数万個の単位で用いられるため，ネットワーク上のトラフィックを増大させることでネットワークに過剰な負荷をかける可能性も考えられる．

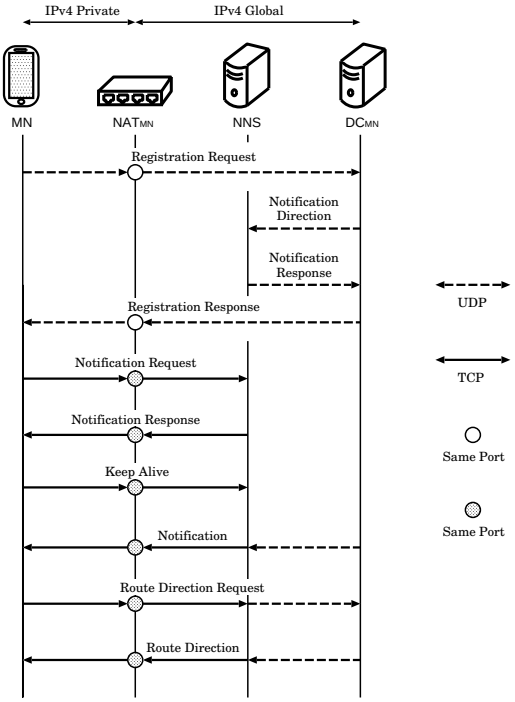

図 7　通信相手先の呼び出し手順 (提案方式)

TCP keep alive packet

Ethernet	IP	TCP
14 bytes	20 bytes	26 bytes

TCP keep alive response packet

Ethernet	IP	TCP
14 bytes	20 bytes	20 bytes

図 8　TCP における keep-alive のパケットフォーマットとサイズ

5. 通知技術を用いる NTMobile

図 7 に提案方式の変換テーブル維持方式を示す．本研究では，NAT における情報保持時間が比較的長い TCP を利用する NNS を新たに NTMobile に導入する．NNS の導

入により，NTM ノードからの UDP による定期的な Keep Alive 動作は必要なくなる上，NAT における TCP の情報保持時間は 1 時間近いため，情報保持に必要なトラヒックを大幅に削減可能である．また，提案方式では，NNS Server が各 NTMobile ノードとの TCP セッションを維持し続ける必要があることから，NNS の機能と DC の機能を論理的に分離することにより，複数の NNS を用いて多数の NTM ノードを管理可能な設計とする．

- Registration Request
  NTMobile ノードは IP アドレスなどの自身のネットワーク環境を登録するために Registration Request を自身の DC に向けて送信する．提案方式では，同時に NTMobile ノードの通知タイプも通知する．そのため，NTMobile ノードは既存の Keep-Alive 方式と TCP を用いた通知方式を任意に選択可能である．
- Notification Direction/Response
  DC は Registration Request により TCP を用いた通知方式を依頼された場合，該当 NTMobile ノードを担当する NNS Server に向けて通知機能を有効にする Notification Direction を送信する．Notification Direction には，DC と該当 NTMobile ノード間の通信で利用する共通鍵が含まれており，NNS Server と NTMobile ノード間の通信で利用される．NNS Server は通知機能の準備が完了したことを確認する Notification Response を DC に返信する．なお，Registration Request を受信した際，DC は鍵の有効期間が切れる前に，新たな鍵への更新処理を行う場合がある．この場合，Notification Direction を新たに送信することで，共通鍵の更新を行う．
- Registration Response
  DC は NNS Server の通知機能の準備したことを確認した後，NTMobile ノードに Registration Response を返信する．
- Notification Request
  NTMobile ノードは Registration Response 内に記載されてる NNS Server の FQDN を用いて，NNS Server に TCP を用いた通知用のコネクション作成を依頼する Notification Request を送信する．
- Notification Response
  NNS Server は通知用のコネクション作成が正常に完了したことを示す，Notification Response を作成した TCP コネクションを用いて返信する．その後，TCP の Keep-alive 機能を用いて，TCP コネクションの維持を行う．

図 8 は TCP の Keep-Alive 機能で利用するパケットフォーマットを示し，通信開始側から 60byte のパケットが送信され，54byte のパケットが返信されるため，1 回の Keep-Alive 動作により 114byte のトラヒックが発生する．

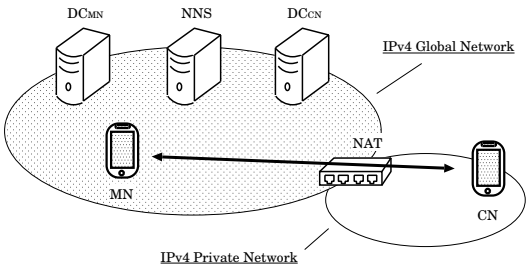

図 9 測定環境

表 1 DC，NNSS, MN, CN(Raspberry Pi) の性能諸元

	DC, NNSS, MN	CN(Raspberry Pi)
OS	Linux	Linux
Distribution	Ubuntu 12.04	Raspbian
Kernel version	kernel 3.2.0	Linux kernel 3.2.27
CPU	Core i7 870	ARM1176JZF-S
Clock	2.93 GHz (仮想 2 コア)	700 MHz
Memory	1 GB	512 MB

表 2 端末間の RTT

	RTT [ms]		
通信ペア	min	avg	max
MN - DCmn	0.217	0.483	0.64
MN - Raspberry Pi	1.64	1.97	3.15
DCmn - DCrpi	0.179	0.475	0.820
NNSS - DCrpi	0.153	0.425	0.715
Raspberry Pi - DCrpi	1.61	2.06	2.44
Raspberry Pi - NNSS	1.70	2.00	2.66

6. 実装と評価

NNS の機能はアプリケーション空間で動作するデーモンとして実装を行い，ソケット通信を用いて DC と連携して動作する方式を実装した．NTMobile で利用されるサーバーは予め公開鍵証明書を保持しており，サーバー間の通信は SSL の相互認証機能を用いて暗号化通信が可能である．また，暗号化通信を用いて，メッセージ交換で利用する共通鍵も交換しており，サーバー間のセキュアな通信を実現可能である．NNS デーモンと DC デーモンも同様に，公開鍵証明書を用いたセキュアな通信を利用する．

なお，NAT ルータにおける TCP コネクションの変換テーブル保持時間はデフォルトでは 1 時間のことが多いが，一部の実装では 30 分ほどと短縮されている．そのため，本実験では，Android OS や iOS でも採用している 20 分を利用した．また，UDP の Keep-Alive 間隔は 20 秒として，実験を行った．評価は，NNS Server を利用することによる遅延時間の増加と Keep-Alive に関するトラヒック削減効果について検証した．

測定環境のモデルを図 9 に，測定に用いた機器の性能諸元を表 1 に示す．また，測定環境の端末間の遅延時間を表 2 に示す．遅延時間の測定には IPv4 のグローバルネット

表 3　経路構築の遅延時間増加

通信経路	Delay time [ms] min	avg	max
DCrpi → RPi	76.4	96.9	158.6
DCrpi → NNSS → RPi	87.3	107.7	198.5

表 4　keep-alive にかかるコスト

	1time	1Hour	1Day
既存方式	140 byte	25.2 Kbyte	604.8 Kbyte
提案方式	114 byte	342 byte	8208 byte

ワークに接続した NTMobile ノードから NAT 下の IPv4 プライベートネットワークに接続する NTMobile ノードへの A レコードの問い合わせを行うプログラムを用いた．既存方式における遅延時間と提案方式における遅延時間の差分により，NNS Server を導入したことによる遅延時間を算出する．

6.1　遅延時間の測定

表 3 に測定した遅延時間を示す．結果から NNS Server を導入したことによる遅延が 10ms 程度増加していることがわかる．この 10ms の内訳は DC から NNS Server への転送時間や NNS Server から NTMobile ノードへの転送時間，NNS Server での処理時間からなっている．測定環境では遅延の少ない有線環境における測定を実施した．一方，3G や LTE,Wi-Fi などの無線通信を使用した場合には数 10 から数 100ms 程度の遅延が発生するため，NNS Server を導入することによる遅延時間への影響は小さいと考えられる．

6.2　トラヒック削減効果

提案方式では TCP の Keep-Alive 機能を利用するため，1 回の Keep-Alive 動作により，114byte のトラヒックが発生する．既存方式では，UDP を用いて NTMobile 独自の Keep-Alive メッセージを交換しているため，1 回の Keep-Alive 動作に 140byte のトラヒックが発生する．

表 4 に既存方式と提案方式の Keep-Alive 動作で利用する通信量を示す．結果より，同一時間のコネクションを維持するために必要とする通信量を，提案方式では既存方式と比較して約 98%削減可能であることが確認できる．

7.　まとめ

本稿では，NTMobile に TCP を用いた通知サービス (NTMobile Notification Service) を新たに提案することにより，NTMobile ノードが Keep-Alive 用に送受信する制御信号を大幅に削減可能であることを示した．また，提案方式を組み込みボード上に実装することにより，提案方式が適切に動作することを実験により確認した．評価では，NNS を実装した環境における遅延時間の測定をし，NNS Server を導入したことによる遅延時間の増加が通信に大きな影響を与えないことを確認した．

謝辞　本研究は科研費 (23700075, 26330103)，総務省 SCOPE2013 の助成を受けたものである．記して謝意を表する．

参考文献

[1] APNIC: IPv4 exhaustion details. `http://www.apnic.net/community/ipv4-exhaustion/ipv4-exhaustion-details`.
[2] Internet Society: IP Addressing Issues. `http://www.internetsociety.org/ip-addressing`.
[3] F. Templin, T. Gleeson and D. Thaler: Intra-Site Automatic Tunnel Addressing Protocol (ISATAP). RFC 5214, IETF(2008).
[4] C. Huitema: Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs), RFC 4380, IETF(2006).
[5] W. Townsley and O.Troan: IPv6 Rapid Deployment on IPv4 Infrastructures (6rd), RFC 5969, IETF(2010).
[6] G. Tsirtsis and P. Srisuresh: Network Address Translation - Protocol Translation (NAT-PT), RFC 2766, IETF(2000).
[7] M. Bangnulo, P. Matthews and I. van Beijnum: Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers, RFC 6146, IETF(2011).
[8] Soliman, H.: Mobile IPv6 Support for Dual Stack Hosts and Routers, RFC 5555, IETF(2009).
[9] 鈴木秀和, 上醉尾一真, 納堂博史, 西尾拓也, 内藤克浩, 渡邊晃: IPv4/IPv6 混在ネットワークにおいて通信接続性と移動透過性を実現する NTMobile の研究, マルチメディア, 分散, 協調とモバイル (DICOMO2012) シンポジウム論文集, pp. 2391–2401, Jul.2012.
[10] 上醉尾一真, 鈴木秀和, 内藤克浩, 渡邊晃: IPv4/IPv6 混在環境で移動透過性を実現する NTMobile の実装と評価, マルチメディア, 分散, 協調とモバイル (DICOMO2012) シンポジウム論文集, pp. 1169–1179, Jul.2012.
[11] 鈴木秀和, 上醉尾一真, 水谷智大, 西尾拓也, 内藤克浩, 渡邊晃: NTMobile における通信接続性の確立手法と実装, 情報処理学会論文誌, Vol. 54, No. 1, pp. 367–379, Jan.2013.
[12] 内藤克浩, 上醉尾一真, 西尾拓也, 水谷智大, 鈴木秀和, 渡邊晃, 森香津夫, 小林英雄: NTMobile における移動透過性の実現と実装, 情報処理学会論文誌, Vol. 54, No. 1, pp. 380–393, Jan.2013.
[13] Braden, R: Requirements for Internet Hosts – Communication Layers, RFC 1122, IETF(1989).
[14] S. Guha, K. Biswas, B. Ford, S. Sivakumar and P. Srisuresh: NAT Behavioral Requirements for TCP, RFC 5382, IETF(2008).
[15] F. Audet and C. Jennings: Network Address Translation (NAT) Behavioral Requirements for Unicast UDPs, RFC 4787, IETF(2007).