情報処理学会論文誌 Vol.60 No.1 16–26 (Jan. 2019)

エンドツーエンド通信をアプリケーションレベルで可能にする通信ライブラリの実現と評価

納堂 博史[1,a)] 鈴木 秀和[1] 内藤 克浩[2] 渡邊 晃[1,b)]

受付日 2018年3月30日, 採録日 2018年10月2日

概要：現存する IP ネットワークには，NAT 越え問題や移動透過性などいくつかの制約がある．これらの制約を回避するため，サービス提供者はインターネット上にサーバを設置するクライアント/サーバ型でシステムを構築する場合が多い．しかし，この方法はサーバの設置に起因する管理面や性能面での課題が存在する．IP ネットワークにかかわる制約の原因は TCP/IP の基本にかかわるものであり，制約を除去するためにこれまではカーネルを改造し，TCP/IP プロトコル自体に手を加える方法が検討されてきた．しかしカーネルを改造する方法は，開発者，利用者とも負担が大きく，普及させるのは困難である．本論文ではこれらの課題を解決するため，エンドツーエンド通信を実現する通信ライブラリをアプリケーションレベルで提供する方式を提案する．本通信ライブラリを利用すると，ユーザは接続されるネットワークを意識する必要がなくなり，あたかも巨大な LAN に接続されているように見える．そのため，クライアント/サーバ通信モデルを前提としてきたこれまでの通信システムの実現方法を 1 から見直すことができる．本通信ライブラリは様々な OS への移植が可能で，Android や iPhone などのスマートフォンでも利用できる．また，本通信ライブラリは C 言語のほか，Java，Ruby などの言語でも利用できる．

キーワード：モバイルネットワークプロトコル，NAT 越え問題，IPv4/IPv6 非互換性，移動透過性，エンドツーエンド通信

Realization of Communication Library that Enables End to End Communication in the Application Layer and Its Evaluation

Hiroshi Nodo[1,a)] Hidekazu Suzuki[1] Katsuhiro Naito[2] Akira Watanabe[1,b)]

Received: March 30, 2018, Accepted: October 2, 2018

Abstract: There are several restrictions in existing networks, such as the NAT traversal problem and mobility. In order to avoid the restrictions, most service providers set servers on the Internet and provide client/server model services. However, this method brings other issues on the side of management burden of servers, and performance degradation. Restrictions of IP networks derive from TCP/IP basics, and therefore conventional technologies have tried to solve the problems by changing kernel of each OS. However, the method accompanying kernel changes give developers and users large burden, and it is difficult to spread the technologies. In order to solve the issues, we propose the communication library by which we can avoid all restrictions of IP networks on the application level. By using the library, users do not aware of the restrictions, and networks can be seen as a big LAN. Therefore, we can review the premise of conventional client/server model, and reconsider the realization method of communication systems. The communication library can be transplanted to other platforms such as Android, and iOS. Also the library can be used from Java and Ruby not only from C language.

Keywords: mobile network protocol, NAT traversal problem, IPv4/IPv6 incompatibility, mobility, end-to-end communication

[1] 名城大学
Department of Information Engineering, Nagoya, Aichi 468–8502, Japan
[2] 愛知工業大学
Faculty of Information Science, Toyota, Aichi 470–0392, Japan

[a)] hiroshi.noudou@wata-lab.meijo-u.ac.jp
[b)] wtnbakr@meijo-u.ac.jp

© 2019 Information Processing Society of Japan

1. はじめに

モバイルネットワークが充実し，IoT が普及してきたことにより，インターネットトラフィックは飛躍的に増大した．インターネットをはじめとする現在のネットワーク基盤は，ほとんど IP ネットワークで実現されていることから，今後の通信インフラは，IP ネットワークを前提に発展していくものと考えられる．

しかし，IP ネットワークには以下のような課題が存在する．最大の課題は IPv4 グローバルアドレスの枯渇により導入されたプライベートアドレスに起因するもので，グローバルアドレス空間からプライベートアドレス空間に対して通信の開始ができないという制約がある（以下 NAT 越え問題）．この制約は IPv4 ネットワークが存続する限り続くため，解決することが強く望まれる．次に，IPv4 グローバルアドレスの枯渇により，IPv6 アドレスが徐々に使われていくと考えられるが，IPv4 と IPv6 には互換性がなく直接通信を行うことができない．今後 IPv6 アドレスしか取得できない端末が出現したとき，IPv4/IPv6 間で効率良く通信できる手段を提供できる必要がある．さらに，通信中にネットワークを切り替えると IP アドレスが変化するため通信を継続できない場合がある．文献 [1] によると，第 5 世代移動通信網と複数の自営網が連携して，周波数資源を動的に融通しあうことが必須であることが指摘されている．すなわち独立したネットワーク間で通信中にネットワークを切り替えることができる移動透過性技術が求められている．

これらの課題を回避するため，現在の通信システムはほとんどがクライアント/サーバ通信モデル（以後 CS モデルと略す）で実現されている．CS モデルはサーバをグローバルアドレス空間に置き，クライアント側からサーバに対して通信を開始するのが前提である．そのため，クライアントがどのようなアドレス空間に存在しても，サーバとの接続性が保証できるという利点がある．CS モデルは，現在のネットワークの課題を容認しつつ，ユーザの要求を満たすことができる最適の方法といえる．しかし，CS モデルはサーバのセキュリティ対策や二重化対策など，管理負荷が大きいという課題がある．また，サーバが処理ネックになる可能性があり，通信遅延が増大するという課題がある．

CS モデルの課題を解決する方法として，カーネルを改造することによりエンドツーエンド通信モデル（以下 E2E モデル）を提供する技術が複数提案されている．ここでいうエンドツーエンド通信とは，エンド端末がどのようなアドレス空間に接続していようと，つねに最適な通信経路での相互通信が可能で，かつ移動透過性を有する通信のことである．アプリケーションから見るとネットワーク全体があたかも巨大な LAN のように見え，ネットワークの制約をいっさい意識する必要がない．しかし，カーネルを改造する方法では，ユーザに高度な知識を要求することになる．また，サービスを提供する開発者側にとっても，カーネルのバージョンアップにつねに追従しなければならないという負担がある．

そこで本論文では，E2E モデルの実現を容易にするために，E2E モデルを実現する通信ライブラリをアプリケーションレベルで提供することを提案する．

E2E モデルが実現可能な既存技術として，DSMIPv6（Dual Stack Mobile IP version 6）[2]，HIP（Host Identity Protocol）[3], [4]，NTMobile（Network Traversal with Mobility）[5], [6], [7] があげられる．これらの技術は，NAT 越え問題の解決，IPv4/IPv6 間相互通信，移動透過性を同時に可能とするものである．

DSMIPv6 は IPv6 対応の移動透過性技術 MobileIPv6 [8] をベースに，IPv4 が混在する環境に機能を拡張した方式である．しかし DSMIPv6 は，IPv6 をベースにした技術であり，IPv4 ネットワークにおいては MobileIPv4 [9], [10] の課題をそのまま引き継いでいる．たとえば，すべての移動端末に IPv4 グローバルアドレスが必要となり，アドレス枯渇問題に逆行するという課題がある．

HIP は IP アドレスから通信識別子としての役割を分離し，HI（Host Identity）と呼ぶ新たな通信識別子を導入することによって，通信接続性と移動透過性を確保することができる．しかし，NAT 越え技術として移動通信を考慮していない ICE [11], [12], [13] を利用しているため，NAT をまたがる移動が複雑で，シグナリング時間が大きくなるという課題がある [14]．

DSMIPv6 と HIP に共通する課題は，カーネルに改造を加える必要があることであり，これらの技術が普及しない大きな要因となっている．

NTMobile はシステム内で一意となる仮想アドレスを各エンド端末に割り当て，すべての通信を実アドレスでカプセル化するのが特徴で，DSMIP や HIP で述べたような技術的な課題は存在しない．サービス提供者はインターネット上に NTMobile をサポートする装置群を設置する必要がある．しかし，エンドユーザはそのことを意識する必要はなく，エンドノードに NTMobile を適用することにより E2E モデルに移行することができる．NTMobile は当初はカーネルでの実装を行ってきた経緯があり，このままでは普及が難しいという課題があった．しかし，NTMobile は将来アプリケーションへ移植することを意識して仕様を策定してきた経緯がある．NTMobile の通信パケットは，仮想アドレスによる通信と，実アドレスによるカプセル化機能が明確に分離されており，NTMobile の機能をアプリケーションで実現することが可能である．

本論文では NTMobile のシグナリング処理と，パケット生成処理をアプリケーションレベルで実現し，通信ライブラリとして提供する [15]．この通信ライブラリを NTMobile

© 2019 Information Processing Society of Japan

framework library（以下 NTMfw と略す）と呼ぶ．具体的には，NTMfw は仮想アドレスによるパケット生成と，NTMobile 専用のヘッダの付与を行う．LINUX カーネルが上記パケットを実アドレスにより UDP カプセル化を行うことにより，カーネルをいっさい改造することなく NTMobile の通信を実現することができる．

NTMfw を LINUX 端末に実装し，IPv4 グローバルアドレス，IPv4 プライベートアドレス，および IPv6 アドレス空間の間で相互の通信が最適経路で確立できることを確認した．また，通信中にネットワークをどのように切り替えても，通信が継続できることを確認した．以上により，現状の IP ネットワークの制約，すなわち NAT 越え問題の解決，IPv4/IPv6 間の相互通信，移動透過性の実現をアプリケーションレベルで解消し，E2E モデルが実現できることを確認した．

NTMfw は C 言語で記述されており，LINUX，Android，iOS などで共有することができる．本論文では LINUX での動作を確認したので報告する．また，Java ラッパー，および Ruby ラッパーを実装し，C，Java，Ruby から呼び出して利用できることを確認した．

以下，2 章ではエンドツーエンド通信の利点と，関連技術として DSMIP と HIP を示す．3 章で NTMobile の概要，4 章で実現した NTMfw を示す．5 章で動作検証と測定結果について，最後に 6 章でまとめる．

2. エンドツーエンド通信の有用性と関連技術

本章では，まず CS モデルと E2E モデルを比較し，エンドツーエンド通信の有用性を述べる．次に E2E モデルを実現する関連技術として DSMIP と HIP を取り上げ，その課題について詳しく述べる．

2.1 クライアントサーバモデルとその課題

現在主流となっている CS モデルの構成は**図 1** のとおりである．CS モデルはインターネット上にサーバを設置することが前提となる．クライアントは携帯電話網を含め，ほとんどがプライベートアドレス空間に存在する．CS モデルはクライアント側から通信開始することが前提であるため，クライアントとサーバは必ず通信を確立でき，NAT 越え問題は発生しない．IPv4/IPv6 間の相互通信も，サーバが IPv4 と IPv6 を変換することにより実現可能である．移動透過性の処理においても，サーバ側のアドレスが固定であることから対応しやすいという利点がある．このように CS モデルは，IP ネットワークの課題をある程度解決することができる．

しかし，CS モデルには管理面と性能面において以下のような課題がある．管理面では，アプリケーションサーバが固定のグローバルアドレスを持つことから，攻撃者の標的にされやすく，常時万全のセキュリティ対策が必要である．アプリケーションサーバは稼働率を向上させるための二重化対策が必須となる．性能面では，すべての情報をサーバに集中させるため，サーバ周辺のトラフィックが増えるうえ，サーバ自体が性能ネックとなりやすい．アプリケーションによっては，サーバを経由することにより好ましくない通信遅延が発生する．

2.2 エンドツーエンドモデルとその有用性

次に本論文で実現を目指す E2E モデルの構成を**図 2** に示す．E2E モデルはネットワークに接続するすべての装置が，NAT の存在，IPv4/IPv6 の違いを意識することなく，かつ移動透過性を実現できるモデルである．すなわち任意の装置間で直接通信ができ，通信中に任意の場所に移動しても通信が切れることはない．

E2E モデルが容易に実現できれば，ユーザはシステム構築の方法を 1 から考え直すことができる．たとえば，エンド装置で実行した処理結果をエンド装置間で直接交換できる．この場合，サーバを経由する必要がないため性能的に有利であるうえ，サーバ自体が不要であるためサーバのセキュリティ対策が不要である．また，エンド装置は一般にプライベート IP アドレスであるため，E2E モデルでは攻撃が成立しにくい．さらに，サーバを介さず直接通信できれば，これまでサーバに起因していたトラフィックの集中，処理ネック，通信遅延などの課題を大きく軽減できる可能性がある．

ここで，E2E モデルは CS モデルを否定するものではなく，CS モデルを包含することができる．CS モデルが適したシステムはそのまま利用し，サーバが介在する必要のない情報を E2E 通信に切り替えることができる．サーバは

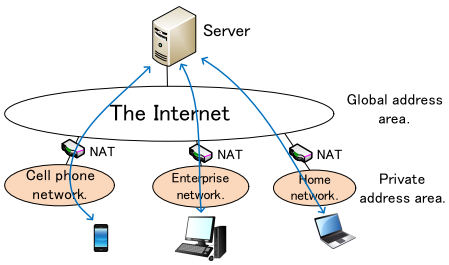

図 1 CS モデルの構成
Fig. 1 Configuration of CS model.

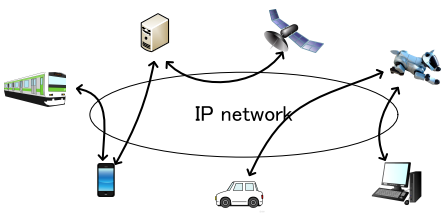

図 2 E2E モデルの構成
Fig. 2 Configuration of E2E model.

© 2019 Information Processing Society of Japan

エンドツーエンド通信の 1 エンドとなるので，E2E モデルの中にサーバが存在してもかまわない．これまでグローバル空間上のクラウドサーバが担っていた機能を，現場に近い複数のローカルサーバが分担し，相互に連携することも可能である．このときローカルサーバが異なるプライベートアドレス空間に分散設置されていてもかまわない．攻撃者は NAT 配下に存在するローカルサーバにアクセスできないため，クラウドサーバを使う方法に比べると安全性が高まる．

ここで E2E モデルを実現する方法がカーネルの改造をともなう方法であると，端末の機種が限定されるうえ，カーネルの頻繁なバージョンアップに追従する必要がある．特にスマートフォンでは，インストール時にルート権限の取得が必要になり，以下のような様々な課題がある．たとえば，ウイルス対策アプリが無効となり，ウイルス感染のリスクが高まる．メーカのサポート対象外となり，不具合が発生したときの対処が難しくなる．ルート化に失敗すると復活できなくなる恐れがある．これらの理由から，E2E モデルの普及を目指すには，E2E モデルをアプリケーションレベルで実現することが必須である．

2.3 エンドツーエンド通信を実現する既存技術

エンドツーエンド通信を実現するための既存技術として，DSMIP，HIP，NTMobile がある．ここでは DSMIP と HIP の概要とその課題を述べる．NTMobile については 3 章で述べる．

2.3.1 DSMIPv6

DSMIPv6 は，Mobile IPv6 を IPv4/IPv6 混在環境に拡張したものである．すなわち，IPv6 の移動透過性技術をベースに，NAT 越え，IPv4/IPv6 間通信を可能にした方式である．DSMIPv6 の構成を**図 3** に示す．IPv4/IPv6 デュアルスタックネットワークに設置する HA（Home Agent）と移動端末 MN（Mobile Node）がトンネル経路を構築する．MN はホームネットワークで取得する HoA（Home Address）と移動先のネットワークで取得する CoA（Care of Address）の 2 種類の IP アドレスを持つ．MN がどのようなネットワークに移動しても HoA は変化せず，CoA のみが変化する．MN の上位アプリケーションは HoA を用いて通信しており，CoA の変化を隠蔽することができる．IPv6 オンリーのネットワークでは経路最適化により直接通信を行うが，それ以外の通信はすべて HA が介在することによりあらゆる通信を可能にする．

DSMIPv6 は IPv4 ネットワークに対応するため，HA を経由して MobileIPv4 の技術をそのまま利用できるようにしている．そのため，MobileIPv4 に関わる課題がそのまま継承されている．MobileIPv4 では，MN の HoA をグローバルアドレスで割り当てる必要がある．これは IPv4 グローバルアドレスの枯渇に逆行しており現実的な方式とはいえない．また，IPv4 空間では経路最適化が定義されておらず，必ず HA を経由した冗長経路となる．

MobileIPv6 は IPv6 オプションを利用しており，MobileIPv4 は IPinIP 処理を行うことから，カーネル内の IP 層を改造するのが前提の仕様となっている．したがって，DSMIP をアプリケーションレベルで実現するのは難しい．

2.3.2 HIP

HIP は，IP アドレスが持つ端末識別子と位置識別子の役割のうち，端末識別子を分離し，端末識別子として HI（Host Identifir）を導入する．エンド端末における HIP のレイヤーモデルを**図 4** に示す．IP 層と TCP/UDP 層との間に新たに HIP 層を定義する．HIP 層においては，IP アドレスと HI のマッピングを管理し，上位層では HI を用いて通信を識別する．HI はエンド端末の公開鍵から生成するため，エンドツーエンドのセキュリティが確実で強固であるという特長がある．IP アドレスは位置識別子の役割のみを担うので，移動によって IP アドレスが変化しても，HI は変化せず移動透過性を実現できる．

HIP は既存技術を可能な限り流用するという方針を持ち，NAT 越え技術として ICE を改造する形で実現してい

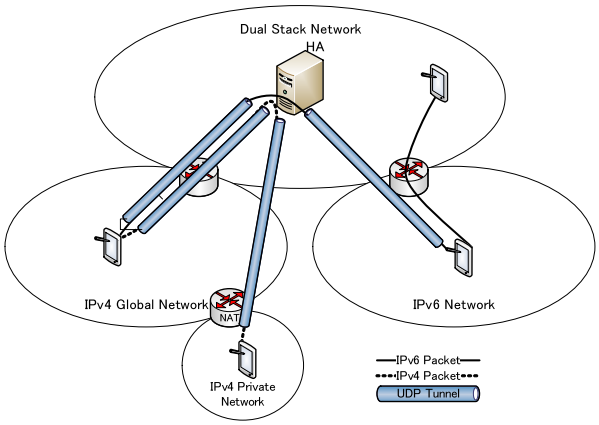

図 3 DSMIPv6 の構成
Fig. 3 Configuration of DSMIPv6.

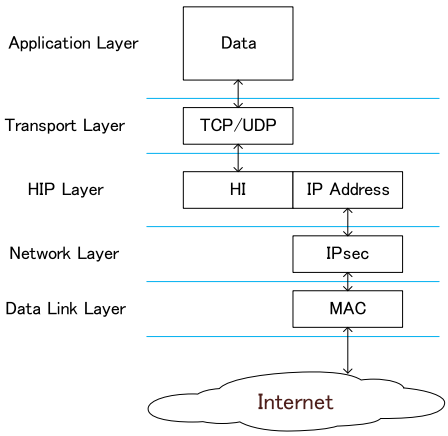

図 4 HIP のレイヤーモデル
Fig. 4 Layer model of HIP.

© 2019 Information Processing Society of Japan

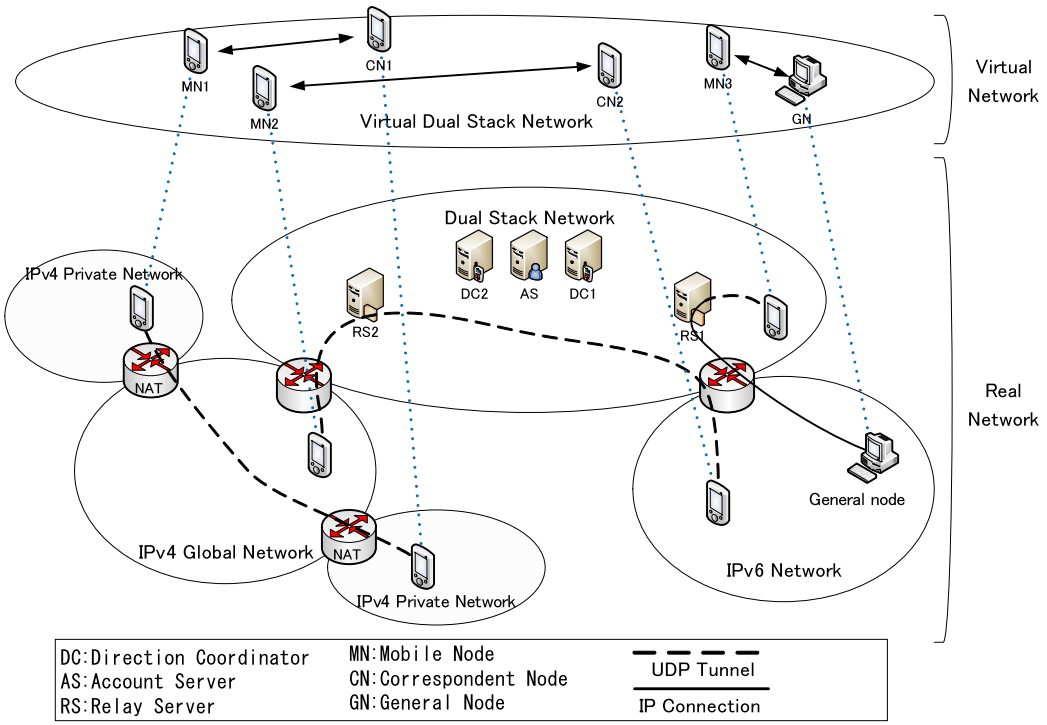

図 5 NTMobile の概要
Fig. 5 Overview of NTMobile.

る．しかし，ICE はもともとは移動透過性を考慮していない技術であり，通信経路上に NAT が介在する場合の移動処理は複雑で時間を要する [14]．

HIP は IP 層とトランスポート層の間に HIP 層を設ける構造上，カーネルの改造が必須であり，HIP 普及の 1 つの妨げとなっている．

3. NTMobile

本章では，NTMobile の構成と基本仕様について述べる．NTMobile は通信接続性と移動透過性を同時に解決することを目的として策定された技術で，DSMIP や HIP で述べたような課題は存在しない．また，仮想アドレスによる通信と実アドレスによるカプセル化は完全に独立しており，通信ライブラリをアプリケーションで実現するために適した構造となっている．本章では，NTMobile の仕組みについて述べ，これをアプリケーションで実現する方法について述べる．

3.1 NTMobile の構成

図 5 に NTMobile の構成を示す．NTMobile は下記に示す機器で構成される．

- DC（Direction Coordinator）
  NTM 端末の実 IP アドレスや仮想 IP アドレスの関係を管理し，UDP トンネルの構築指示を出す．
- AS（Account Server）
  ユーザの登録と認証を行う．NTM 端末の認証や，DC と NTM 端末間などにおける通信の暗号化に用いる共通鍵の配布を行う．
- RS（Relay Server）
  NTM 端末間で直接通信ができない場合にカプセル化パケットを中継する．両エンド端末が異なる NAT 配下に存在する場合，一方が IPv4 ネットワーク，もう一方が IPv6 ネットワークに存在する場合がこれに相当する．
- NTM 端末
  NTMobile の機能を有するエンド端末である．

NTM 端末を除く装置群はすべて公開鍵証明書を所持しており，暗号化通信に用いる共通鍵は，あらかじめ公開鍵証明書を用いて安全に共有しておく．NTM 端末は，あらかじめ AS にユーザ ID と認証情報を登録しておく必要がある．NTM 端末は，起動時に AS に TLS（Transport Layer Security）にてログインする．AS は NTM 端末を認証すると，DC と NTM 端末に共通鍵を配布する．NTM 端末どうしの通信に用いる共通鍵は，通信開始時に DC が生成して，そのつど NTM 端末に配布する．このようにして，NTMobile のシグナリングを含むすべての制御メッセージは，共通鍵を用いたパケット認証と暗号化がなされる．

3.2 NTMobile の通信方式

NTM 端末は，立ち上げ時に，DC に対して実 IP アドレスを登録し，DC からは仮想 IPv4/IPv6 アドレスの配布を受ける．以降，定期的に DC とキープアライブを実行するこ

© 2019 Information Processing Society of Japan

とにより，NTM 端末と DC 間のコネクションを維持する．NTM 端末がスマートフォンのときは APNS（Apple Push Notification Service）や GCM（Google Cloud Messaging）の Notification サービスを利用することにより，DC とのキープアライブを省略できる．

以下，イニシェータ側の NTM 端末を MN（Mobile Node），レスポンダ側の NTM 端末を CN（Corresponding Node）と記述する．MN が通信を開始するときは，CN に対する名前解決（DNS 要求）がトリガとなり，NTMobile のシグナリング処理が開始される．NTMobile のシグナリングとは，DC が MN と CN に対して通信経路を指示する動作である．MN はまず自らを管理する DC（DCmn）に CN の FQDN を添えて経路指示要求を送信する．DCmn は，DNS サーバ機能により CN を管理する DC（DCcn）を探索する．DCcn が見つかると，そこから CN の実 IP アドレスと仮想 IP アドレス，NAT が存在する場合は NAT の IP アドレスを取得する．DCmn は MN と CN の存在する位置関係から適切な通信経路を決定し，MN と CN に対して経路指示を行う．CN に対する経路指示は DCcn 経由で行う．MN と CN は指示に従って MN–CN 間での直接通信による UDP トンネル経路を構築する．MN と CN はこのトンネル経路を利用して仮想アドレスによるセッションを確立する．なお，MN と CN が直接通信できない場合は，RS 経由の UDP トンネルを構築する．直接通信ができない場合とは，両エンド端末がいずれも NAT 配下に存在する場合，一方が IPv4 端末，もう一方が IPv6 端末の場合である．RS は複数設置でき，DC がそのつど適切な RS を選択する．

4. NTMobile framework library

NTMobile framework library（NTMfw）は NTMobile の機能をすべてアプリケーションで実現するアプリケーションライブラリである．本章では NTMfw の動作を詳しく記述し，その実装方法について述べる．

4.1 NTMfw の位置づけ

NTMfw は，一般のアプリケーションがカーネルの通信ライブラリを利用するのと同じ要領で NTMfw を呼び出すことにより利用できる．NTMobile のシグナリング処理は，NTMfw が実行するため，アプリケーションはこれを意識する必要がない．シグナリング処理はアプリケーションからのアドレス解決指示（DNS 要求）をトリガにして実行される．DC と NTM 端末によるシグナリング処理によりトンネル経路が生成されると，その後の一般通信はすべて UDP カプセル化通信となる．カプセル化処理については，カプセルの内側のヘッダ生成を NTMfw が担当し，外側のヘッダ生成を LINUX カーネルが担当する．

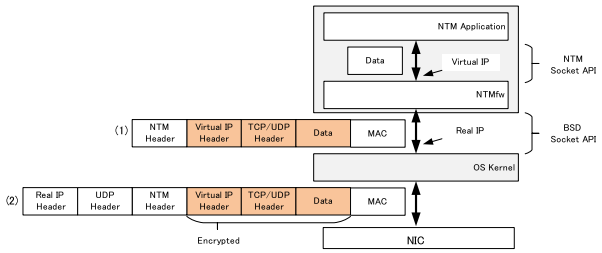

図 6 NTMfw のカプセル化通信の実現方法

Fig. 6 Realization method of capsuled communication in NTMfw.

4.2 NTMfw の動作

NTMfw のカプセル化処理に着目してその実現方法を**図 6** に示す．NTMfw は，仮想 IP プロトコルスタックの機能を包含しており，このプロトコルスタックの持つ仮想ネットワークインタフェースに仮想 IP アドレスが割り当てられる．アプリケーションは，NTMobile 用の NTM ソケット API を利用してデータの送受信を行う．NTMfw 自身は OS 標準の BSD ソケット API を利用してパケットの送受信を行う．

NTM アプリケーションが送信したデータは，仮想 IP プロトコルスタックの処理により，仮想 IP アドレスを用いて TCP/UDP ヘッダおよび IP ヘッダが付与される．このパケットには，さらに NTMobile 通信であることを示す NTM ヘッダが付与され，暗号化，MAC（Message Authentication Code）付与などの処理を経て，ソケット API を用いて OS に渡される（図 6 (1)）．MAC はメッセージの内容が正しいことを示す認証コードである．カーネルは上記パケットを UDP でカプセル化する（図 6 (2)）．これら一連の処理により，パケットはトンネル経路を経由して相手端末に届けられる．パケットの受信処理は上記と逆の手順により実現される．

NTMfw は DNS 要求をトリガとして，NTMobile のシグナリング処理も実行する．アプリケーションは NTMobile のシグナリング動作を意識する必要はない．また，NTMfw は通信中に自らの IP アドレスを監視している．IP アドレスの変化を検出すると，これをネットワークが切り替わったものとして認識し，DC との間で再度シグナリングを実行する．この動作により実 IP アドレスが変化した場合においても，アプリケーションは実 IP アドレスの変化に気づくことなく通信が継続される．

4.3 NTMfw のモジュール構成

NTMfw のモジュール構成を**図 7** に示す．NTMfw は次のモジュールから構成される．

- NTM ソケット API
  BSD ソケット API に代わってアプリケーションに提供するソケット API である．NTM ソケット API の名称は，たとえば ntmfw_sendto のように，BSD ソケット

© 2019 Information Processing Society of Japan

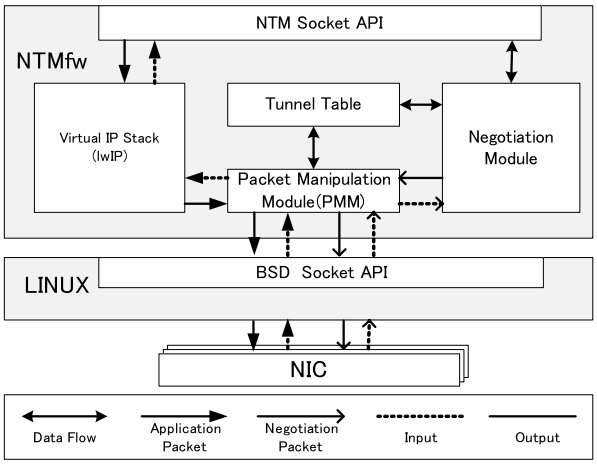

図 7 NTMfw のモジュール構成
Fig. 7 Module configuration of NTMfw.

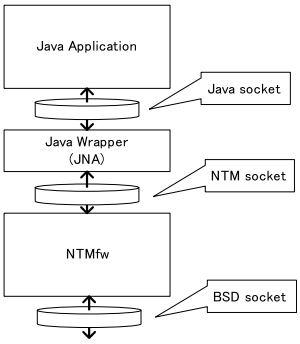

図 8 Java ラッパー
Fig. 8 Java wrapper.

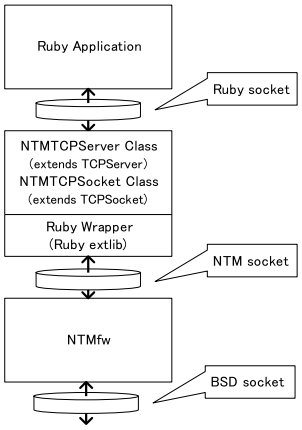

図 9 Ruby ラッパー
Fig. 9 Ruby wrapper.

API の名称の前に ntmfw_を付与し，機能的には BSD ソケットと互換性を持つ．ただし，NTMobile の初期化処理（電源立ち上げ時の登録処理）は NTMobile 特有の処理として必要である．そこで，アプリケーション側から NTMobile の初期処理を指示できるように，ntmfw_init 関数を新たに定義した．初期化処理により，AS との認証処理，および DC との通信に利用する共通鍵を取得する処理を実行する．また，名前解決を担う API（例：ntmfw_getaddrinfo）については，その引数から CN の FQDN を抽出して，ネゴシエーションモジュールに指示し，DC との間のシグナリングを開始する．

- ネゴシエーションモジュール
  NTMobile の初期化処理とシグナリング処理（通信開始時および移動時のトンネル生成）を行う．移動処理を実行するために，netlink ソケットを利用して，通信中の実アドレスの変化を検出する．変化を検出すると，ネットワークが切り替わったものと判断し，移動処理にかかわるシグナリングを開始し，新たなトンネル経路を生成する．
- パケット処理モジュール
  通信パケット，およびシグナリングパケットの生成，解析処理を行う．通信パケットに対しては，NTMobile ヘッダの付与，暗号化/復号処理，MAC 認証処理を実行する．BSD Socket API を用いて実アドレスによるパケットの送受信を行う．
- 仮想 IP プロトコルスタック
  NTMfw はアプリケーション層にて TCP/IP プロトコルを実行する．これには lwIP（A lightweight TCP/IP stack）[*1]を利用し，NTMfw が呼び出すライブラリとして実装した．lwIP は組込みシステム向けに提供されているオープンソースの TCP/IP スタックである．

[*1] http://savannah.nongnu.org/projects/lwip/

- トンネルテーブル
  通信相手ごとに FQDN，仮想 IPv4/IPv6 アドレス，実 IPv4/IPv6 アドレス，共通鍵，通信識別子などを格納したテーブルである．

4.4 ラッパーの機能

NTMfw を C 言語以外からも利用可能とするため，Java ラッパーおよび Ruby ラッパーを設計し実装した．Java ラッパーの実装構成を**図 8** に示す．Java アプリケーションから NTMfw を利用するため，JNA（Java Native Access）を用い，C 言語で記述された NTM ソケット API を呼び出すラッパークラスを定義した[*2]．開発者は，ラッパークラスのメソッドを利用してデータの送受信を行うことにより，NTMobile の機能を利用することができる．

Ruby ラッパーの実装構成を**図 9** に示す．Ruby アプリケーションから NTMfw を利用するため，Ruby 拡張ライブラリを作成し，C 言語で記述された NTM ソケット API を Ruby から利用できるようにした．また，Ruby で TCP 通信を行う際に利用されるクラス[*3]を継承し，NTMfw を利用した TCP 通信を行うラッパークラスを定義した．開発者は，TCPServer クラスまたは TCPSocket クラスを用いる代わりに，NTMTCPServer クラスまたは NTMTCPSocket クラスを用いることにより，NTMobile の機能を利用することができる．

4.5 NTMobile サービスの準備

NTMobile サービスを提供するには以下の準備が必要である．以下，AS，DC，RS を準備する者を NTMobile サービス提供者，アプリケーションを提供する者をアプリケーション提供者，それらを利用するユーザをエンドユーザとする．

[*2] https://github.com/java-native-access/jna
[*3] サーバアプリケーションは TCPServer クラスを，クライアントアプリケーションは TCPSocket クラスを利用する．

© 2019 Information Processing Society of Japan

- NTMobile サービス提供者の作業
  インターネット上に AS，DC，RS を準備する．ドメイン名を確保し，サーバ名とグローバル IP アドレスを取得する．各装置の公開鍵証明書を取得し，相互の認証関係を指定する．ユーザのアカウント登録を受け付け，ユーザ端末が立ち上がったときの認証を行う．DC，AS，RS に対しては万全のセキュリティ対策と二重化対策が必須である．
- アプリケーション提供者の作業
  アプリケーションの提供方法として，既存アプリケーションを利用する方法と，新規アプリケーションを利用する方法がある．既存アプリケーションを利用する場合は，当該プログラムのソケット API の名称を一律 NTMfw 用の名称に書き換える．また，初期化起動処理をアプリケーション起動時の処理に追加する．新規アプリケーションを利用する場合は，当該アプリケーションを新たに開発する必要がある．ソケット API は，NTMfw と C ソケットで機能互換があるので開発負荷は一般のアプリケーションを新規開発する場合とほぼ同等である．ただし，初期化起動処理は別途必要である．ユーザどうしの認証に関しては，小規模なシステムであればユーザが NTMobile のアカウントを保持していることをもって信頼関係にあると見なすことができる．しかし，ユーザ数が多く複数のアプリケーションが混在する場合には，アプリケーションごとにユーザの認証機能を別途考慮する必要がある．
- エンドユーザの作業
  NTMfw をライブラリとして組み込んだアプリケーションをエンド端末にインストールする．AS にてユーザアカウントを取得する．AS の FQDN を端末に登録する．

以上の準備により，エンドユーザはエンドツーエンドの通信が可能となる．NTMobile サービス提供者とアプリケーション提供者はそれなりの準備を要するが，エンドユーザは導入が容易である．

5. 評価

以上の構成よりなる NTMfw を実装し，動作検証および性能評価を行った．また既存技術との定性的な比較を行った．

5.1 通信接続性の検証

NTM ソケット API を用いた C 言語の試験プログラムを作成し，MN と CN 間でパケットが送受信可能であることを確認した．試験プログラムは，UDP および TCP の IPv4 ソケットと IPv6 ソケットを生成し，各ソケットで送受信を行う．

機能確認のため，Windows10 内に，仮想マシンにより NTMobile のネットワークを構築した．仮想マシンはすべて Ubuntu14.04LTS とした．ネットワーク接続は，MN および CN を IPv4 グローバルネットワーク，IPv4 プライベートネットワーク，IPv6 ネットワークのいずれかに接続し，すべての組合せで接続性試験を行った．IPv4 グローバルネットワークに接続する場合は IPv6 を無効化してブリッジ接続し，IPv4 プライベートネットワークに接続するときは NAT 経由の接続とした．IPv6 ネットワークに接続する場合は IPv4 を無効化してブリッジ接続した．なお，簡単のため DC と RS は 1 台のみとした．

表 1 通信接続性試験結果
Table 1 Test results of connectivity.

		CN		
		IPv4G	IPv4P1	IPv6
MN	IPv4G	◎	◎	○
	IPv4P2	◎	◎	○
	IPv6	○	○	◎

◎：direct ○：via RS

通信接続性試験結果を**表 1** に示す．表中の分類は実 IP アドレスの分類を示しており，G はグローバルアドレス，P1，P2 はプライベートアドレスを示す．P1 と P2 は異なる NAT 配下であり，異なるプライベートネットワークである．アプリケーションが利用する仮想 IP アドレスは IPv4，IPv6 のどちらでもかまわない．◎は直接通信，○は RS 経由での通信が成功したことを示す．異なるプライベートアドレスどうしの通信では，最初に RS 経由の通信を確立し，その後 MN と CN 間で NTMobile の経路最適化を実行する [17]．NAT が Cone 型の場合は経路最適化に成功するが，表 1 ではそれが成功したことを示している．

また，Java ラッパーと Ruby ラッパー間で通信試験を行った．それぞれのラッパーを用いて開発した Java プログラムおよび Ruby プログラムを用い，一方で Java を，もう一方で Ruby を実行して試験を行い，TCP と UDP それぞれの通信が実現できることを確認した．

5.2 パケット処理時間とその内訳

UDP パケットの処理時間に着目し，NTM 端末における処理時間の内訳を調査した．測定は getrusage 関数を利用し，各処理に要した時間を測定した．測定マシンは，OS：Ubuntu14.04，CPU：2 コア 2.8 GHz，メモリ：8 GB で，この中に AS，DC，RS，MN，CN を仮想マシンで生成した．受信処理時間の測定については，受信処理をループさせ，パケットを受信した直後からの処理時間が分かるテストプログラムを作成した．

表 2 に 1,024 バイト長のパケットにおける送信処理および受信処理について時間測定した結果を示す．表中の数字は 1,000 回の平均である．NTMfw 処理にかかわる機能

© 2019 Information Processing Society of Japan

表 2 UDP パケットの処理時間（単位 μ 秒）
Table 2 Process time of an UDP packet (micro sec.).

項目			処理時間	
送信処理	LINUX		160	3,116
	NTMfw	仮想 IP	1,250	
		暗号化	1,227	
		MAC 生成	106	
		その他	372	
受信処理	LINUX		226	3,106
	NTMfw	仮想 IP	1,020	
		復号処理	1,117	
		MAC 検証	92	
		その他	651	

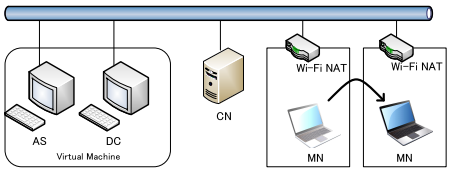

図 10 実験ネットワークの構成
Fig. 10 Configuration of the experimental network.

は，MAC 生成/認証，暗号/復号，仮想 IP スタック，その他（NTM ヘッダの付与/除去，パケット振り分け処理など）に分けて測定した．LINUX にかかわる処理は，UDP カプセル化/デカプセル化処理にかかわる部分であり，NTMfw を利用しない場合は，パケット処理時間はこの部分のみとなる．すなわち，LINUX 処理時間以外が NTMfw を利用したことによる処理時間の増加に相当する．NTMfw の内訳を見ると，暗号/復号処理，仮想 IP 処理が多くの時間を要していることが分かる．これらの処理については汎用のライブラリを利用している部分であり，短縮することが難しい．暗号/復号処理についてはハードウェア化することによる時間短縮が考えられる．

5.3 移動透過性試験

NTMfw による移動透過性の試験のため，MN と CN を実機に置き換え移動試験を行った．実験ネットワークおよび各機器の接続を**図 10** に示す．AS と DC を仮想マシンで構築し，MN と CN をそれぞれ別の物理マシンに構築した．MN と CN のマシンは，Ubuntu14.04，2 コア 2.8 GHz，メモリ 8 G バイトとした．実験ネットワークは 1 Gbps の IPv4 ネットワークとし，2 台の Wi-Fi 対応 NAT とそれに接続する CN を準備した．

MN–CN 間でトンネル通信を行っている途中で，MN の NAT を切り替えた．切替試験を 10 回繰り返し，wireshark を用いて MN の通信パケットをキャプチャすることにより，移動処理にかかわる時間を測定した．

移動処理の試験結果を**図 11** に示す．MN の Wi-Fi NAT を手動で切り替えると，直後に旧 Wi-FiNAT に向けて離脱

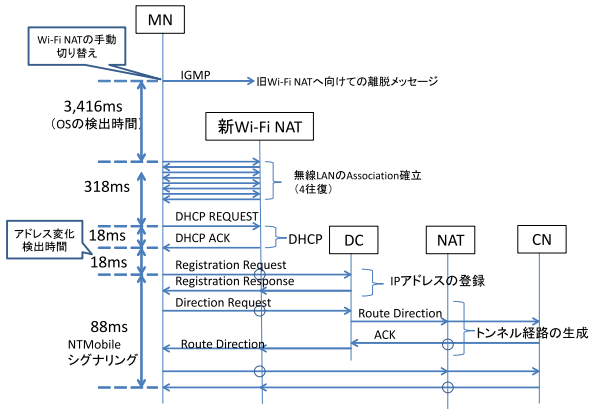

図 11 移動処理の試験結果
Fig. 11 Experimental results of the mobile process.

を知らせる IGMP が送信される．このパケットを起点として，以下のシーケンスを Wireshark で観測し，各処理の時間を測定した．測定結果は 10 回の移動処理の平均であり，それぞれの時間を図 11 内に記載した．NAT の切り替え後，MN と新 Wi-FiNAT 間で認証をともなう無線 LAN 特有のシーケンスが走り，無線 LAN のアソシエーションを確立する．次に 1 往復の DHCP シーケンスにより新 IP アドレスを取得する．DHCP は一般には 2 往復（DISCOVER/OFFER/REQUEST/ACK）であるが，Wi-Fi の場合は，2 回目以降の接続時に前のアドレスを記憶しており，1 往復で終了する．この IP アドレス変化を netlink で検出し，NTMobile のシーケンスが始まる．移動時のシーケンスは，DC へのアドレス登録，および新トンネル経路の生成よりなる．

図 11 より分かるように，ほとんどがネットワーク切り換え後，Association 確立シーケンスが開始するまでの時間，すなわち LINUX が Wi-Fi の移動を検知するまでの時間で占められている．DHCP による新アドレス取得時間，および netlink によるアドレス変化検出時間はきわめて短時間で終了しており，その後のシグナリング時間も短時間で終了していることが分かる．今回の試験ネットワークは，通信遅延のほとんどない環境で行っているが，実際には NTMobile のシグナリング処理にネットワーク遅延が加算される．しかし，インターネットの遅延を約 15 m 秒と仮定しても，NTMobile の移動処理開始後は十分高速に移動処理が実現できているといえる．今後は LINUX カーネル側にて接続断時間をさらに短縮可能であるかどうかの検討が必要である．

5.4 既存技術との比較

エンドツーエンド通信を実現する既存技術として DSMIP と HIP を取り上げ，提案方式（NTMfw を用いた NTMobile）と比較した．

表 3 に既存技術と提案方式の比較結果を示す．IPv4 の

© 2019 Information Processing Society of Japan

表 3 既存技術と提案方式の比較結果
Table 3 Comparison against the conventional methods.

	DSMIP	HIP	提案方式
IPv4 の NAT 越え	×	○	○
移動透過性	○	△	○
IPv4/IPv6 相互通信	○	○	○
カーネルの改造	×	×	○
既存アプリケーション	○	○	△

NAT 越えに関して，DSMIP は HA をグローバルアドレス空間に設置する必要があることから，MN の IPv4 ホームアドレスがグローバルアドレスである必要があり，IPv4 アドレス枯渇問題に逆行する．また，通信経路は必ず HA を経由する冗長経路となることから×とした．DSMIP は Mobile IPv4 における課題をそのまま引き継いでおり，IPv4 が中心の現在のネットワークでは現実的な方式ではないといえる．移動透過性について，HIP は NAT の存在する環境において移動処理に時間を要することから△とした．IPv4/IPv6 相互通信についてはすべての方式とも可能である．カーネルの改造について，DSMIP と HIP では必須であることから×とした．既存アプリケーションについては，DSMIP，HIP はそのまま利用できるが，提案方式は通信ライブラリ用にソケット名称を書き換える必要があり△とした．

本項目を×ではなく△とした理由は以下のとおりである．提案方式では既存アプリケーションをそのままでは使えないが，アプリケーション提供者がソケット API の名称の書き換えと，NTMobile の起動処理を追加することにより，エンドユーザは通常のアプリケーションとして利用できる．一方，カーネルを改造する方法はエンドユーザに所定の知識が必要である．また，スマートフォンユーザは 2.2 節で述べたように実質利用できないといえる．さらに，カーネルがバージョンアップしたとき，サービス提供者はカーネルの内容を解析し直さなければならない場合があり負担が大きい．上記をふまえ，アプリケーションを改造する方式は，カーネルを改造する方法に比べ優位であると判断し，差別化するために△とした．

以上の比較より，提案方式は E2E 通信の普及に資する方式といえる．

5.5 具体的なアプリケーション事例

エンドツーエンド通信により実現可能となるアプリケーションとして以下のようなものが考えられる．

IP 電話はエンドツーエンド通信が適している有用なアプリケーションである．直接通信のため遅延も少ないので，サーバを経由する場合に比べて会話品質が向上する．移動処理に関しては，移動前と移動後のネットワークに制約がないという利点がある．すなわち事業者が異なっていてもよいし，プライベートなネットワークであってもかまわない．さらに，IPv4 から IPv6 への移動であってもよい．このことから NTMobile を利用した IP 電話は携帯電話の完全代替になりうる可能性がある．エンドツーエンドの暗号化とパケット認証により安全性が高いという利点もあげられる．

次に既存の CS モデルにおけるセキュリティの向上と処理効率の向上に貢献できる．サーバをプライベート空間に置くことができるので，攻撃者からの対象になりにくく安全性が向上する．複数のサーバが処理を分担し，互いに連携することができる．サーバを分散させることにより，トラフィクネックや処理ネックの解消が可能になる．サーバをエンド装置の近くに置くことにより通信遅延を減らす効果もある．

E2E モデルは OS を改造しない限り同一 LAN 内でしか実現できなかった．提案方式により，E2E モデルを広域のネットワークをまたがってセキュアに実現できる．遠隔地の PC やスマートフォンからプライベート空間にあるセンサ情報を直接読んだり，機器を直接制御するようなアプリケーションを実現することが可能である．E2E モデルを前提とした新たな発想によるアプリケーションが出現してくることが期待できる．

6. まとめ

現状のネットワークには NAT 越え問題，IPv4/IPv6 の非互換性，異なるネットワークをまたがる移動透過性が難しいという課題がある．これらの課題を回避するため，本研究では NTMobile をアプリケーションで実現する通信ライブラリを提案した．NTMobile は移動透過性に有用なカプセル化処理を，アプリケーションとカーネルが分担して実現できるパケット構成となっているため，NTMobile の機能をユーザランドに移植することができた．NTMfw は移植性を考慮して C 言語で実現したが，Java，Ruby からも利用できるようにラッパーを開発した．

NTMfw を LINUX 上で試作し，動作検証および性能測定を行った．その結果エンドツーエンド通信をアプリケーションレベルで実現できることを確認した．Android，iOS は LINUX と同様の BSD ソケットにより通信インタフェースを提供しているため，今後はこれらの OS への移植を行う．またスループットの向上に向けた実装方式の検討が必要である．

参考文献

[1] 総務省：第 5 世代移動通信システム実現に向けた研究開発～複数移動通信網の最適利用を実現する制御基盤技術に関する研究開発 (2015)，入手先 ⟨http://www.soumu.go.jp/main_content/000349194.pdf⟩．
[2] Soliman, H.: Mobile IPv6 Support for Dual Stack Hosts and Routers, RFC5555, IETF (2009).

© 2019 Information Processing Society of Japan

[3] Moskowitz, R., Heer, T., Jokela, P. and Henderson, T.: Host Identity Protocol Version 2 (HIPv2), RFC7401, Updated by RFC8002, IETF (2015).
[4] Nikander, P., Gurtov, A. and Henderson, T.: Host Identity Protocol (HIP): Connectivity, Mobility, Multi-Homing, Security, and Privacy over IPv4 and IPv6 Networks, *IEEE Communications Surveys and Tutorials*, Vol.12, No.2 (2010).
[5] 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 (2013).
[6] 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 (2013).
[7] 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，情報処理学会論文誌，Vol.54, No.10, pp.2288–2299 (2013).
[8] Perkins, C., Johnson, D. and Arkko, J.: Mobility Support in IPv6, RFC6275, IETF (2011).
[9] Perkins, C. (Ed.): WiChorus: IP Mobility Support for IPv4, Revised, RFC5944, IETF (2010).
[10] Levkowetz, H. and Vaarala, S.: Mobile IP Traversal of Network Address Translation (NAT) Devices, RFC3519, IETF (2003).
[11] Rosenberg, J.: Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols, RFC5245, IETF (2010).
[12] Rosenberg, J., Mahy, R., Matthews, P. and Wing, D.: Session Traversal Utilities for NAT (STUN), RFC5389, IETF (2008).
[13] R. Mahy, P. Matthews and J. Rosenberg: Traversal Using Relays around NAT (TURN): Relay Extensions to RFC5766, IETF (2010).
[14] Maenpaa, J., Andersson, V., Camarillo, G. and Keranen, A.: Impact of Network Address Translator Traversal on Delays in Peer-to-Peer Session Initiation Protocol, *Proc. IEEE GLOBECOM2010* (2010).
[15] 納堂博史，杉原史人，鈴木秀和，内藤克浩，渡邊 晃：NTMobile の実用化に向けた統合的枠組の検討，情報処理学会研究報告，Vol.2015-MBL-77, No.20, pp.1–8 (2015).
[16] 三宅佑佳, 鈴木秀和, 内藤克浩, 渡邊 晃：NTMobile における最適なリレーサーバ選択手法の提案と実装，情報処理学会研究報告，Vol.2015-MBL-77, No.20, pp.1–9 (2015).
[17] 納堂博史，鈴木秀和，内藤克浩，渡邊 晃：NTMobile における自律的経路最適化の提案，情報処理学会論文誌，Vol.54, No.1, pp.394–403 (2013).

納堂 博史 （正会員）

1989 年生．2012 年名城大学理工学部情報工学科卒業．同年大口町役場入庁．2017 年名城大学大学院修士課程修了．在学時代，モバイルネットワークに関する研究に携わる．修士（工学）．

鈴木 秀和 （正会員）

2004 年 3 月名城大学理工学部情報科学科卒業．2009 年 3 月同大学大学院理工学研究科電気電子・情報・材料工学専攻博士後期課程修了．2008 年 4 月日本学術振興会特別研究員．2010 年 4 月名城大学理工学部助教．2015 年 4 月より同大学理工学部准教授および東北大学電気通信研究所共同研究員を兼任．ネットワークセキュリティ，モバイルネットワーク，ホームネットワーク等の研究に従事．博士（工学）．IEEE，ACM，WCTRS，電子情報通信学会各会員．

内藤 克浩 （正会員）

1977 年生．1999 年 3 月慶應義塾大学理工学部電気工学科卒業．2004 年 3 月名古屋大学大学院工学研究科情報工学専攻博士課程後期課程修了．2004 年 4 月三重大学工学部電気電子工学科助手．2007 年 4 月同大学助教．2011 年 9 月カリフォルニア大学ロサンゼルス校客員研究員．2014 年 4 月愛知工業大学情報科学部准教授．2016 年情報処理学会・長尾真記念特別賞受賞．博士（工学）．IEEE，情報処理学会，電子情報通信学会各会員．主として無線ネットワーク，モバイルコンピューティングの研究に従事．

渡邊 晃 （正会員）

1974 年慶應義塾大学工学部電気工学科卒業．1976 年同大学大学院工学研究科修士課程修了．同年三菱電機株式会社入社後，LAN システムの開発・設計に従事．1991 年同社情報技術総合研究所に移籍し，ルータ，ネットワークセキュリティ等の研究に従事．2002 年名城大学理工学部教授，現在に至る．博士（工学）．電子情報通信学会，IEEE 各会員．

© 2019 Information Processing Society of Japan