Android向けNTMobileの移動機能の実装と評価

黒宮 魁人[1]　田中 久順[1]　鈴木 秀和[1]　内藤 克浩[2]　渡邊 晃[1]

概要：NTMobile（Network Traversal with Mobility）は，バージョンの異なる IP アドレス間の通信や NAT 越え問題を解決したうえで，移動しながらの通信を可能とする移動透過性を実現することが可能な技術である．NTMobile は当初 LinuxPC での実装された．また，現在は Android 端末に NTMobile を使用するためのアプリケーションが提供され，一般アプリケーションに NTMobile によるバージョンの異なる IP アドレス間の通信や NAT 越えの機能の確認を実施した．しかし，移動に関わる部分が実装されておらず，一般アプリケーションに移動透過性を実現できていない．そこで本研究では，スマートデバイス向けに提供された NTMobile の移動に関わる処理の提案と実装を行い，Android に NTMobile による移動透過性を確認した．

Implementation and Evaluation of Mobility Functions of NTMobile for Android

KAITO KUROMIYA[1]　HISAYOSHI TANAKA[1]　HIDEKAZU SUZUKI[1]　KATSUHIRO NAITO[2]
AKIRA WATANABE[1]

1. はじめに

スマートデバイスの普及により，モバイルデータトラフィックが爆発的に増加しており，2015 年には 3.7EB/月であったモバイルデータトラフィックが 2020 年までに 30.6EB/月まで増加すると予測されている [1] そのため，モバイルデータ通信網から IP ネットワークにデータオフロードすることが求めらている．しかし，現在の IP ネットワークには以下のような課題がある．IP ネットワークは IPv4 アドレスと IPv6 アドレスが混在しており，バージョンの異なるアドレス同士で直接通信することができない．また，IPv4 ネットワークではインターネット側から NAT（Network Address Translation）配下の端末に通信を開始することが出来ない（NAT 越え問題）．さらに，IP ネットワークでは IP アドレスが位置識別子と通信識別子の 2 つの役割を担っているため，端末の移動に伴う IP アドレスの変化によってそれまで行なっていた通信が切断される．そのため，移動の激しいスマートデバイスにストレスなくデータオフロードをさせるためには移動透過性の実現が必須である．

これらの課題を解決する技術として筆者らは NTMobile（Network Traversal with Mobility）を提案している [2] [3] [4] [5]．NTMobile とは，IPv4 / IPv6 混在環境において NAT 越え問題を解決しつつ移動透過性を実現する通信技術である．まず，NTMobile では NTMobile を導入した端末（NTM 端末）に，位置に依存しない仮想 IP アドレスを割り当てる．NTM 端末に実装されたアプリケーションは仮想 IP アドレスを利用して通信を行うが，実際の通信は端末の実 IP アドレスを用いてカプセル化 / デカプセル化を行う．このため，実 IP アドレスが変化してもアプリケーションは IP アドレスの変化に気づくことなく通信を継続することが可能である．また，NTM 端末とグローバル空間に設置された通信経路指示装置に対して定期的 KeepAlive を行うことで，端末が NAT 配下に存在していても通信経路指示を受けることが可能である．更に，IPv4 と IPv6 間の通信や，通信を行う両端末が NAT 配下に存在するような直接通信ができない環境であっても，独自の中継装置を利用してカプセル化通信を行う．このように現在のインターネットの課題を解決するにあたり NTMobile は有用性の高

[1] 名城大学
Meijo University
[2] 愛知工業大学
Aichi Institute of Technology

い技術であるが，スマートデバイスに実装するためには大きな課題が存在する．

NTMobile ではカーネル空間にてパケットのカプセル化 / デカプセル化を実行するため，NTMobile による通信を行うために端末の管理者権限を取得する必要がある．このため端末の root 化が推奨されていないスマートデバイスに普及させることが難しい．

この課題を解決するため，Android の VPNService が提供するパケットのカプセル化 / デカプセル化を利用し，NTMobile を実現する方法（VPNService 型 NTMobile）が提案されている [6]．VPNService 型 NTMobile は一般アプリケーションを使用して NAT 越えや異なるバージョンの IP アドレスによる相互通信などの一部の機能を確認済みである．しかし，NTMobile 通信を実行するライブラリが，アドレス変化検出を異なる OS で共通で実現できないため，移動透過性が実現できていない．

そこで本研究では VPNService 型 NTMobile のアドレス変化検出を NTMobile 通信を行うライブラリから独立させることで移動透過性の実現を行う．

以降，2 章で NTMobile について述べ，3 章では提案方式について説明し，4 章で実装について説明し，5 章で評価について述べた後，最後に 6 章でまとめる．

2. NTMobile

本章では，NTMobile の動作と実装方法について説明する．

2.1 NTMobile 概要

図 1 に NTMobile の構成を示す．NTMobile は NTM 端末の他に，端末情報の管理や通信経路の指示，仮想 IP アドレスの割り当てを行う DC（Direction Coordinator）及び IPv4/IPv6 間の通信や，NTM 端末が異なる NAT 配下に存在する際に通信の中継を行う RS（Relay Server）によって構成される．DC，RS は Dual Stack Network に配置されていることが前提である．また，ネットワークの規模に応じて，複数台設置することにより負荷分散が可能である．NTM 端末は起動時に DC に登録処理を行うことにより，DC から仮想 IP アドレスの割り当てを受ける．NTM 端末のアプリケーションは，割り当てられた仮想 IP アドレスを用いて通信を行う．NTMobile では，IP アドレスの位置識別子の役割を実 IP アドレスが持ち，通信識別子の役割を仮想 IP アドレスが持つ．通信を行う際は仮想 IP アドレスを利用して作成したパケットを実 IP アドレスにて UDP でカプセル化する．これにより NTM 端末の実 IP アドレスが変化しても通信識別子としての役割を持つ仮想 IP アドレスは変化しないため，移動透過性を実現できる．また，NTM 端末と DC は一定時間ごとに行われる Keep Alive によって，NTM 端末が NAT 配下に存在しても DC からの

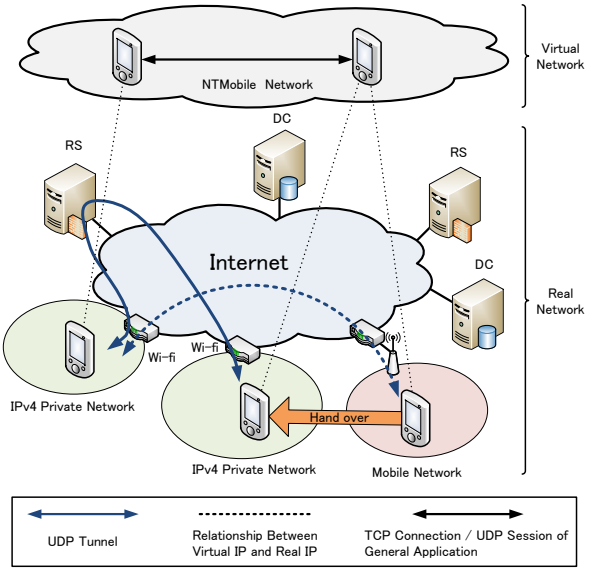

図 1 Configuration of NTMobile

指示はいつでも受信できる．NTM 端末同士は原則直接通信をするように設計されているが，IPv4/IPv6 間の相互通信の場合，もしくは NTM 端末が異なる NAT 配下に存在する場合は RS が通信の中継を行う．ただし，後者の場合，NAT の種類によっては通信を RS で中継せずに直接通信に切り替えることができる [8]．

2.2 NTMobile の動作シーケンス

以降の説明では，通信開始側の NTM 端末を MN（Mobile Node），通信相手側の NTM 端末を CN（Correspondent Node）とする．**図 2** に NTMobile の通信開始時のシーケンスを示す．図 2 では，MN，CN は異なる NAT 配下に存在している．また，簡略化のため DC，RS は 1 台としている．前提として，DC には MN と CN の端末情報が既に登録されており，定期的な Keep Alive が行われている．

通信開始時，MN は DC に経路指示要求として CN の FQDN を含む Direction Request を送信する．DC は受信した Direction Request と，登録済みの CN の情報を確認する．図 2 のネットワーク構成では，MN と CN が共に NAT 配下であることから RS を経由した通信経路を構築する必要があると判断する．DC は RS に対して通信の中継指示である Relay Direction を送信し，RS は ACK を返信する．次に DC は MN と CN に経路指示として Route Direction を送信する．Route Direction を受信した CN は RS からのパケットを受信可能とするため RS に対して Hole Punch を送信する．MN は UDP によるトンネルを構築するために Tunnel Request を RS を経由し CN に送信する．Tunnel Request を受信した CN は Tunnel Response を RS を経由して MN に返信する．これにより MN と RS 間の UDP トンネルと RS と CN 間の UDP トンネルが構築される．

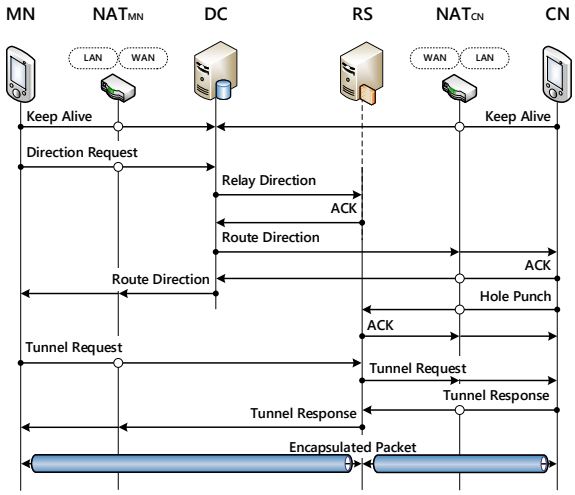

図 2 Sequence at the start of NTMobile's communication

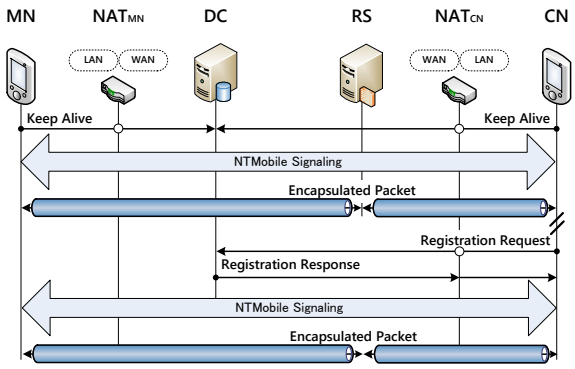

図 3 Sequence of NTMobile's movement processing

2.3 NTMobile の移動通信シーケンス

図 3 に移動に係る NTMobile の通信シーケンスを示す．図 3 では，カプセル通信を行っている途中で CN がアドレス変化した場合を想定したシーケンスである．NTMobile では，アドレスの変化を検出した場合，DC に実 IP アドレスの登録作業（Registration）を実行した後に再度シグナリングを行い新たなトンネル経路を生成する．

2.4 NTMobile Framework（NTMfw）

NTMobile 通信を提供する C で記述された通信ライブラリとして NTMobile Framework（NTMfw）がある．NTMfw では仮想 IP アドレスパケット生成のために lwip（A Lightweight TCP / IP stack）を利用している．lwip を利用することで，NTMobile がカーネルで行なっていた処理をユーザ空間で実行することが可能となる．NTMfw の処理は全てユーザ空間にて実行されるように設計されているため，端末の root 権限を必要としない．NTMfw をスマートデバイスのアプリケーションに組み込むことで，root 化をしていない一般的なスマートデバイスであっても NTMobile 通信が可能となる．しかし，アプリケーション

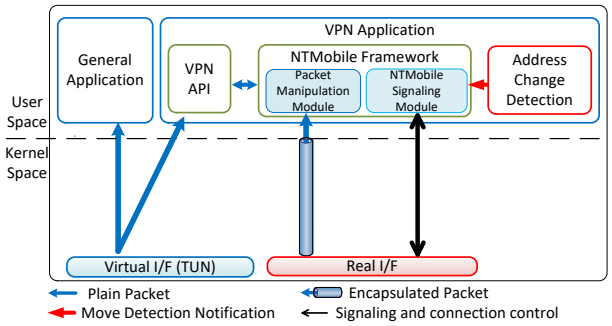

図 4 VPNService 型 NTMobile にアドレス変化検出を実装した際のモジュール図

毎に NTMfw を組み込む必要があり，ストアで提供されている一般アプリケーションに適用することが出来ないという課題も存在する．

2.5 VPNService 型 NTMobile

VPNService 型 NTMobile は，NTMfw をスマートデバイスアプリケーションとして利用するために実装されたモデルである．この実装方式は，VPN 通信を行うための API（以降，VPN API）である VPNService と NTMfw を利用している．VPNService 型 NTMobile は VPN API によるカプセル化 / デカプセル化機能を利用することで，既存のアプリケーションに対して NTMobile 上での通信を実現することが可能である．Java アプリケーションとして実装されており，C で記述された NTMfw を呼び出すために JNA（Java Native Access）を使用する．VPNService 型 NTMobile では NTMobile の NAT 越えやエンドツーエンドの直接通信を確認済みである．しかし，NTMfw が OS によって異なるインタフェースの名前を共通で検出することが出来ないため，端末の IP アドレスが変化しても移動処理を実行することが出来ないという課題が残されている．

3. 提案方式

本章では，VPNService 型 NTMobile にアドレス変化検出を実現する方法について説明する．

3.1 モジュール構成

図 4 にアドレス検出処理を独立させた VPNService 型 NTMobile のモジュール図を示す．

General Application はスマートデバイスにて一般的に使われるアプリケーションである．また VPN Application は，VPN API と NTMobile Framework を一つのアプリケーションとして実装したものである．VPN Application では，まず VPN Application 起動時に VPN API が仮想インタフェースである TUN の作成を行う．その後，一般アプリケーションは TUN インタフェースを経由してパケットの送受信を行う．NTMobile Framework の中に含まれている

NTMobile Signaling Module は，VPN Application 起動時のアドレス登録処理や NTMobile によるシグナリングを行う．また，Packet Manipulation Module では，NTMobile によるカプセル化パケットの送受信処理を行う．提案方式では，VPN Application の中に NTMfw とは独立してアドレス変化検出モジュールを追加した．アドレスが変化した際には NTMobile Signaling Module にアドレスが変化したことを伝える通知を送信する．その後，通知をトリガとして図 3 に示したシーケンスを実行することで VPNService 型 NTMobile の移動透過性を実現する．

4. 実装

本章では，アドレス変化の検出について実装を行なったので報告する．

4.1 Address Change Detection

4.1.1 動作概要

アドレス変化検出モジュールは VPN Application にてアドレスの変化を検出する．これを実現するために Android が提供する Connectivity Manager を使用する [10]．Connectivity Manager は，Android 端末の接続状況が変化すると，CONNECTIVITY_ACTION（“android.net.conn.CONNECTIVITY_CHANGE”）をブロードキャストする．そのため，VPN Application からアドレス変化検出を実行するためには，CONNECTIVITY_ACTION を受信するレシーバ (Connection Receiver) を作成する．Address Change Detection では，CONNECTIVITY_ACTION をトリガとして NTMfw の Signaling Module に通知を送信する．これらの処理は Java によって記述されているが，NTMfw は C で記述されているため JNA（Java Native Access）を経由する必要がある．そのため，NTMfw に Android 用の端末移動時の処理を呼び出す関数を追加し，共通ライブラリの作成を行った．また，CONNECTIVITY_ACTION を受信時に，Java 側から loadLibrary メソッドを利用し端末移動時の処理を呼び出した．

4.1.2 Connection Receiver の動作

Connection Receiver は Broadcast Receiver を継承させた，CONNECTIVITY_ACTION を受信するレシーバである．実装を行うにあたり Connection Receiver の内部に Wi-Fi に接続を切り替えた時，3G / LTE 通信（Mobile 通信）に接続を切り替えた時，端末がオフラインとなった場合に Connectivity Manager が送信するブロードキャストを監視するリスナーを作成した．アプリケーションを起動している状態で，Wi-Fi に接続を切り替えた時もしくは Mobile 通信に接続を切り替えた時は，NTMfw に記述されている移動処理を実行させる．

5. 評価

本章では，アドレス変化の検出についての動作検証と性能測定について述べる．

5.1 動作検証

VPN Application が端末移動時に正しく移動処理を実行できるかを検証するため，インターネットを模擬した研究室のローカル環境を利用して以下の実験を行った．実験を行う際，Google Play ストアから Android 向けアプリケーションとして提供されている Network Analyzer [11] を使用した．MN，CN 共に NAT 配下に設置した状態で，Network Analyzer を利用して定期的に Ping を送信する．この状態で MN，又は CN の接続先を変化させた．動作検証の結果，Android 端末が移動した際に通信が途切れず VPN Application に移動透過性が実現できていることを確認できた．

5.2 性能測定

提案方式にて移動処理にかかる時間を測定した．測定で使用した機材の仕様を**表 1** に示す．また，性能測定を行うにあたり使用したネットワーク構成の図を**図 5** に示す．ここで NEC 社製のルータを Wi-Fi，Buffalo 社製のルータを Wi-Fi2 とする．

図 6 に測定結果を示す．その結果 Wi-Fi から Mobile に通信を切り替える時間は，525[msc] であり，Wi-Fi から Wi-Fi2 に通信切り替える時間は 5305[msec] であった．NTMobile のシグナリングに要する時間は 122.85[msec] であった．測定結果より端末の接続を切り替えるための時間が処理時間の大部分を占めているため，提案方式が与える影響は軽微であると考えられる．

表 1 各機材の性能の仕様

	MN, CN	DC, RS
OS	Android 7.1.1	Ubuntu 14.04LTS
CPU	NVIDIA Tegra K1@2.50GHz	Intel(R) Xeon CPU E3-1240 v5@3.50GHz
Memory	2GB	1GB

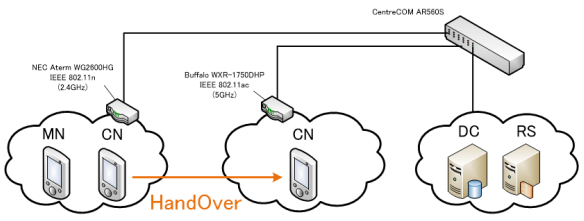

図 5 評価測定時のネットワーク構成

Wi-Fi から Mobile への切り替え時間が短い理由は，ス

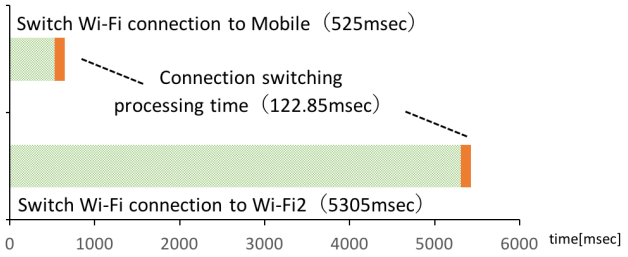

図 6 ネットワーク切り替え処理にかかった時間

マートデバイスが Wi-Fi を利用して通信している間も，Mobile 通信を行うための IP アドレスを保持しているため，短時間で IP アドレスを切り替えることが出来た結果であると推測できる．また，Wi-Fi から Wi-Fi2 に通信を切り替える際には Wi-Fi に IP アドレスをリリースした後に Wi-Fi2 から新しい IP アドレスの割り当てを受ける必要があるので，通信の切り替え処理に多くの時間を費やしたと考えられる．

6. まとめ

本稿では，VPNService 型 NTMobile に移動透過性を実現する方式についての実装と評価を行なった．

参考文献

[1] Cisco virtual networking index : 全世界のモバイルデータトラフィックの予測、2015 〜 2020 年アップデート. https://www.cisco.com/c/ja_jp/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html.

[2] 鈴木秀和, 水谷智大, 西尾拓也, 内藤克浩, 渡邊晃. Ntmobile における相互接続性の確立手法と実装. マルチメディア，分散，協調とモバイル（DICOMO2011）シンポジウム論文集, 第 2011 巻, pp. 1339–1348, 2011.

[3] 鈴木秀和, 上酔尾一真, 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.

[4] 内藤克浩, 上酔尾一真, 水谷智大, 西尾拓也, 鈴木秀和, 渡邊晃. 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.

[5] 上酔尾一真, 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. 52, No. 9, pp. 2549–2561, 2011.

[6] 山田貴之, 鈴木秀和, 内藤克浩, 渡邊晃. Ipv4/ipv6 混在環境に対応した vpnservice 型 ntmobile の性能評価. マルチメディア，分散，協調とモバイル（DICOMO2015）シンポジウム論文集, 第 2015 巻, pp. 1792–1799, 2015.

[7] Firebase cloud messaging - google. https://firebase.google.com/?hl=ja.

[8] 納堂博史, 鈴木秀和, 内藤 克浩晃. 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.

[9] Google developers japan: ios で firebase cloud messaging をデバッグする. https://developers-jp.googleblog.com/2017/02/debugging-firebase-cloud-messaging-on.html.

[10] Connectivitymanager — android developers. https://developer.android.com/reference/android/net/ConnectivityManager.html.

[11] Network analyzer - google play のアプリ. https://play.google.com/store/apps/details?id=net.techet.netanalyzerlite.an&hl=ja.