Vol.2012-DPS-151 No.19
Vol.2012-MBL-62 No.19
2012/5/22

NTMobileのAndroid端末への実装と評価

上醉尾 一真[1]　鈴木 秀和[1]　内藤 克浩[2]　渡邊 晃[1]

概要：スマートフォンなどの携帯端末の普及や無線技術の発展により，移動しながら通信を行うことや外出先からホームネットワーク内の情報機器へ通信を行うことに対する要求が高まっている．しかし，これらの要求を満たすためには，移動透過性の実現やNAT越え問題と呼ぶ課題を解決しなければならない．著者らは，移動透過性とNAT越えをIPv4とIPv6が混在した環境で実現するNTMobile（Network Traversal with Mobility）を提案している．本稿ではNTMobileをAndroid OSを搭載した携帯端末へ実装し，IPv4環境においてハンドオーバなどの動作検証および性能評価を行った．動作検証の結果，NAT配下端末に対する接続性の確保およびNATを跨がった移動をした際に通信を継続可能であることを確認した．

キーワード：移動透過性，NAT越え，Android

Implementation and Evaluation of NTMobile to Android Terminal

Kamienoo Kazuma[1]　Suzuki Hidekazu[1]　Naito Katsuhiro[2]　Watanabe Akira[1]

Abstract: The demand for mobile communication and remote access to home networks are getting much attentions with the spread of mobile terminals such as smartphones and the development of wireless technologies. However, it is necessary to solve the NAT traversal problems and realize mobility at the same time. We have proposed Network Traversal with Mobility (NTMobile) that can provide NAT traversal and mobility in IPv4 and IPv6 networks. In this paper, we have implemented NTMobile mechanisms in an Android terminal, and evaluated the communication performance in an IPv4 network. As the result, we have confirmed that the NTMobile terminal can establish a connection to the corespondent terminal behind an NAT router, and the communication maintains its connection ever when its IP address changes.

Keywords: Mobility, NAT Traversal, Android

1. はじめに

Android OS[*1]を搭載したスマートフォンをはじめとする高性能な携帯端末の普及に伴い，移動しながら通信を行いたいという要求が高まっている．スマートフォンには3GやWi-Fiなどの異なる無線インタフェースが搭載されており，必要に応じてインタフェースを切り替えて通信を行うことができる．しかし，IPネットワークでは通信端末のインタフェースに割り当てられたIPアドレスを用いて通信を管理しているため，端末の移動やインタフェースの切り替えに伴いIPアドレスが変化すると通信を継続することができない．このような問題を解決する技術を移動透過性技術と呼び，現在までに様々な移動透過性技術が提案されてきた [1].

また，既存のIPv4ネットワークではNATを導入してプライベートネットワークを構築することが一般的であり，IPv4グローバルアドレスが枯渇した今日では，LSN（Large Scale NAT）のようにキャリアレベルでもNATが導入され始めている．しかし，グローバルネットワーク側の端末からプライベートネットワーク側の端末に対する接続性を確保できない，NAT越え問題と呼ぶ課題があり，エンドツーエンドの接続性というインターネット本来の理念

[1] 名城大学大学院理工学研究科
Graduate School of Science and Technology, Meijo University
[2] 三重大学大学院工学研究科
Graduate School of Engineering, Mie University

Vol.2012-DPS-151 No.19
Vol.2012-MBL-62 No.19
2012/5/22

を損なう要因となっている．IPv4 グローバルアドレスの枯渇問題を本質的に解決するためには，アドレス空間の広い IPv6 への移行が必須である．しかし，IPv4 と IPv6 には互換性がないため，既存の IPv4 ネットワークを即座に IPv6 へ移行する事は困難である．そのため，当面の間は IPv4 と IPv6 が混在した環境が続くことが想定される．今後の IP ネットワークにてシームレスな通信を実現するためには，NAT 環境や IPv4 と IPv6 の混在環境にて相互接続性の確保や移動透過性を実現する必要がある．

IPv4 と IPv6 の混在環境において移動透過性を実現する技術として，DSMIP（Dual Stack Mobile IPv6）が提案されている [2]．DSMIP は移動端末に対して，ホームネットワークで取得する HoA（Home Address）と訪問先ネットワークから取得する CoA（Care of Address）の二種類のアドレスを割り当て，アプリケーションが HoA を用いた通信を行うことにより，端末の移動に伴う CoA の変化を隠蔽する．移動端末はホームネットワークに設置した HA（Home Agent）との間にトンネルを構築し，アプリケーションが生成したパケットはトンネルを用いて HA へ送信され，HA から通信相手端末へ転送される．移動端末が NAT 配下に接続している場合には常に HA を経由した冗長な通信を行うか，訪問先のネットワークに特殊な NAT が必要になるなどの課題がある．

著者らは IPv4 と IPv6 が混在した環境において移動透過性を実現する NTMobile（Network Traversal with Mobility）を提案している [3–6]．NTMobile は NAT 越えの技術を兼ね備えており，NAT に改造を加えることなく NAT 配下の移動端末（以後 NTM 端末）に対する接続性を確保することができる．NTMobile では NTM 端末に仮想 IP アドレスを割り当て，アプリケーションが仮想 IP アドレスに基づいた通信を行うことにより，端末の移動に伴う実 IP アドレスの変化を隠蔽し，アプリケーション間の通信を継続する．また，NAT の有無や IPv4 と IPv6 ネットワーク間の通信など，通信端末が接続しているネットワークに応じて最適な経路でトンネルを構築し，アプリケーションが生成したパケットを転送する．端末間に構築されるトンネルは，特定の状況を除き，エンドツーエンドで構築されるため経路が冗長になりにくいという特徴がある．

NTMobile は IPv4 環境における Linux PC 向けの実装と評価が先行している．しかし，実際に使用される携帯端末としては Android 端末などのスマートフォンが一般的であり，NTMobile でも Android 端末への実装を想定している．Android OS には携帯端末向けに変更が加えられた Linux カーネルが搭載されているため，Linux PC とは異なる挙動をする可能性がある．そこで本稿では NTMobile を Android 端末へ実装し，IPv4 環境における動作検証および性能評価を行った．

以下，2 章で NTMobile について概説し，3 章で実装について述べ，4 章で動作検証および評価を行い，5 章でまとめる．

2. NTMobile

2.1 概要

NTMobile で想定するネットワークを**図 1** に示す．NTMobile は DC（Direction Coordinator），NTM 端末，RS（Relay Server）によって構成される．DC や RS はグローバルネットワークに設置し，ネットワークの規模に応じて分散設置することができる．

DC は仮想 IP アドレスの割り当て管理や，NTM 端末に対してトンネル構築などの指示を出す装置である．NTM 端末に割り当てられる仮想 IP アドレスは一意なアドレスであり，各 DC は自身に割り当てられたアドレス空間から重複が起きないよう割り当てを行う [5]．また，DC は Dynamic DNS の機能を包含しており，NTM 端末の A レコードや AAAA レコードに加えて，NTMobile 専用レコード（以下 NTM レコード）を登録することにより，NTM 端末のアドレス情報を管理する．NTM レコードには IPv4 用と IPv6 用の 2 種類が定義されており，NTM 端末を一意に識別する Node ID，NTM 端末の FQDN（Fully Qualified Domain Name），実 IP アドレス，仮想 IP アドレス，NAT の外側の実 IP アドレス，アドレス情報を管理している DC の実 IP アドレスが記載されている．

NTM 端末は移動先のネットワークから割り当てられる実 IP アドレスと，DC から割り当てられる仮想 IP アドレスの 2 つのアドレスを保持している．NTM 端末の使用するアプリケーションは仮想 IP アドレスに基づいた通信を

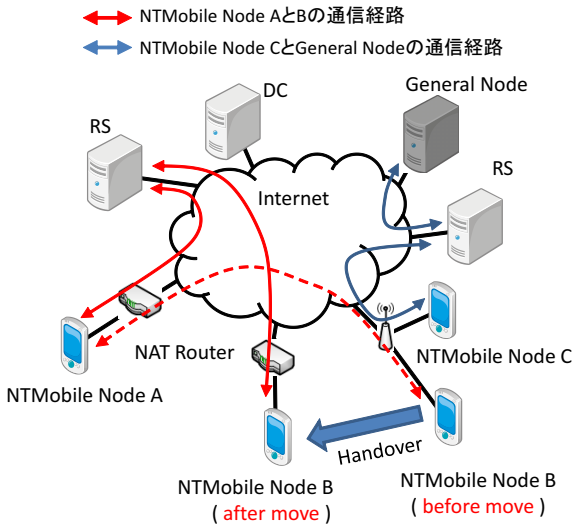

図 1 NTMobile の概要
Fig. 1 Overview of NTMobile.

*1 Android OS とは，米 Google 社が発表した Linux カーネルを採用した携帯端末向けの OS である．

行うことにより，NTM 端末の実 IP アドレスが変化しても通信を継続することができる．なお，仮想 IP アドレスに基づくアプリケーションパケットは，NTM 端末間に構築される UDP トンネルによって転送される．図 1 における NTM 端末 A と移動前の NTM 端末 B のように，どちらか一方の端末がグローバルネットワークに接続している場合には，エンドツーエンドでトンネルが構築される．

RS は，図 1 における NTM 端末 A と移動後の NTM 端末 B のように通信端末が異なる NAT 配下に位置する場合や，NTM 端末 C のように NTMobile 未実装の一般端末と通信を行う場合に，通信の中継を行う装置である．ただし，前者のケースにおいては，NAT の種類のよってはエンドツーエンドの通信に切り替えることが可能である [7].

RS をデュアルスタックネットワークに設置することにより，IPv4 と IPv6 間の通信を実現することができる．アプリケーション間では IPv4 または IPv6 のいずれかの仮想 IP アドレスに基づいた通信が行われるため，端末が接続しているネットワークの違いに影響されることがない．そのため，通信中に通信経路が IPv4 から IPv6 へ変化しても，通信を継続する事ができる．

DC と各端末は信頼関係があることを前提としており，NTMobile で使用されるメッセージは各端末間で共有している暗号鍵を用いて暗号化される．また，NTM 端末間や NTM 端末と RS の間で行われるトンネル通信は，トンネル構築時に DC より配布される共通鍵を用いて暗号化される．

本稿では IPv4 環境において NTMobile の動作検証および性能評価を行ったため，以後は IPv4 ネットワークに接続した NTM 端末同士が通信を行う際の動作を中心に記述する．

2.2 動作シーケンス

以後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．また，端末 X の Node ID を NID_X，実 IPv4 アドレスを $RIP4_X$，仮想 IPv4 アドレスを $VIP4_X$ とし，アドレス情報を管理している DC を DC_X とする．MN と CN の通信時に用いる Path ID を PID_{MN-CN} とする．Path ID は NTM 端末間の通信を一意に識別するための識別子である．

2.2.1 端末情報の登録

NTM 端末は，端末起動時および端末移動時に実 IP アドレスなどの端末情報を DC に登録する．MN は FQDN や NID_{MN}，$RIP4_{MN}$ など，NTM レコードに記載する情報を記載した Registration Request を DC_{MN} へ送信する．DC_{MN} は Registration Request を受信すると，DNS サーバに登録されている MN のリソースレコードを更新し，MN へ応答を返す．なお，Registration Request の IP ヘッダに格納されている送信元 IP アドレスが $RIP4_{MN}$ と異なる場合，MN が NAT 配下に存在すると判断し，NAT の外側の実 IP アドレスとして送信元 IP アドレスを登録する．

登録処理が完了した後，MN と DC_{MN} は定期的にメッセージを交換することにより，制御メッセージ用の通信経路を確保する（Keep Alive）．これにより，MN が NAT 配下に接続している場合であっても DC_{MN} は常に MN へ制御メッセージを送信することができる．また，CN についても同様の処理を行い，DC_{CN} へアドレス情報を登録する．

2.2.2 名前解決処理

NTMobile では，DNS による名前解決をトリガーとして NTMobile 特有のネゴシエーションを実行する [3]．MN は名前解決処理を検出すると，DNS リゾルバにより CN の A レコードや NTM レコードなどの問い合わせを行う．このとき，NTM レコードを取得できなかった場合には，通信相手が NTMobile 未実装端末であると判断する．MN は DNS サーバからの応答を一時待避し，取得した情報をもとに 2.2.3 項で説明するトンネル構築処理を実行する．

トンネル構築完了後，MN は待避していた DNS サーバからの応答に含まれる $RIP4_{CN}$ を $VIP4_{CN}$ に書き換え，DNS リゾルバへ渡す．これにより，MN のアプリケーションは通信相手の IP アドレスとして $VIP4_{CN}$ を認識することになる．

2.2.3 トンネル構築

NTM 端末は通信相手の名前解決完了後，DC の指示に従ってトンネルを構築する．**図 2** に IPv4 グローバルネットワークに接続した MN が，IPv4 プライベートネットワークに接続した CN との間にトンネルを構築するまでの様子を示す．MN は DC_{MN} へ Direction Request を送信し，トンネル構築の指示を要求する．Direction Request には PID_{MN-CN}，および MN と CN の NTM レコードの情報が記載されている．DC_{MN} はこのメッセージに記載された情報から，CN のみがプライベートネットワークに存在す

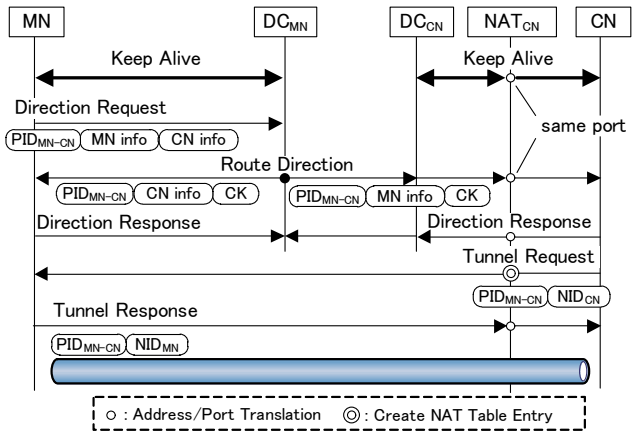

図 2 NTM 端末間のトンネル構築手順
Fig. 2 Tunnel establishment procedure between NTM nodes.

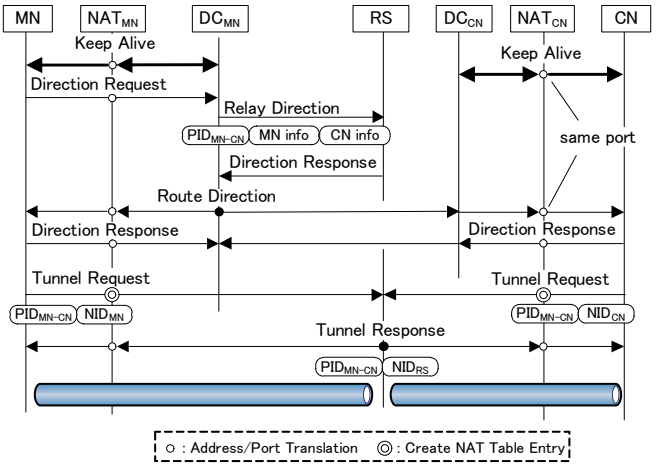

図 3 RS を経由したトンネル構築手順
Fig. 3 Tunnel establishment procedure via RS.

ることを認識すると，MN と CN 間で直接トンネルを構築するために，DC$_{CN}$ 経由で CN へ Route Direction を送信し，MN へ Tunnel Request を送信するよう指示する．また，DC$_{MN}$ は MN に対して，CN から送信される Tunnel Request を受信するよう指示する．Route Direction には PID_{MN-CN}，通信相手の実 IP アドレスと仮想 IP アドレス，トンネルの構築先の実 IP アドレス，トンネル通信時の暗号化に用いる共通鍵 CK などが記載されている．NAT 配下に接続した CN から MN へ Tunnel Request を送信することにより，NAT$_{CN}$ に CN と MN が通信を行うためのマッピング情報が生成され，MN と CN の間に NAT を跨がったトンネルを構築することができる．NTM 端末はカーネル空間にトンネルテーブルを保持しており，構築したトンネルの Path ID や通信相手の仮想 IP アドレス，トンネルの構築先の実 IP アドレス，暗号化に用いる共通鍵などを登録する．

MN と CN が異なるプライベートネットワークに接続している場合には，両端末が送信する Tunnel Request が通信相手側の NAT により破棄されてしまうため，エンドツーエンドでトンネルを構築することができない．そこで，**図 3** に示すように MN と CN は RS との間にトンネルを構築し，RS を経由した通信を行う．DC$_{MN}$ は MN が送信した Direction Request の内容から MN と CN が異なる NAT 配下に存在することを認識すると，RS に対して MN と CN の通信を中継するよう指示を記載した Relay Direction を送信する．Relay Direction には PID_{MN-CN} や MN と CN のアドレス情報が記載されており，これを受信した RS は DC$_{MN}$ へ応答を返し，MN と CN から送信される Tunnel Request を待機する．RS からの応答を受信した DC$_{MN}$ は，MN と CN に対して RS へ Tunnel Request を送信するよう指示する．MN と CN が RS へ Tunnel Request を送信することにより，RS との間に NAT を跨がったトンネルを構築することができ，以後は RS を経由した通信が開始される．

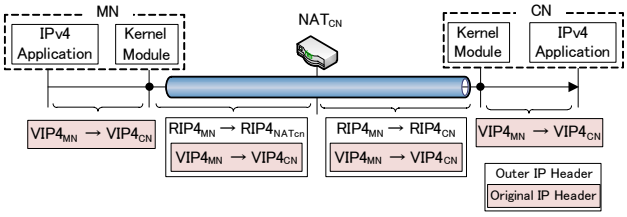

図 4 トンネル通信時のアドレス遷移
Fig. 4 Address transition in tunneling communication.

2.2.4 トンネル通信

MN が CN へトンネル通信を行う様子を**図 4** に示す．アプリケーションレベルでは仮想 IP アドレスに基づいた通信が行われるため，アプリケーションが生成したパケットには仮想 IP アドレスが記載される．MN は宛先アドレスである $VIP4_{CN}$ をキーにトンネルテーブルを検索し，該当エントリに従ってカプセル化および暗号化を行い，$RIP4_{NATcn}$ へ送信する．カプセル化の際には IP ヘッダと UDP ヘッダ，Path ID などを記載した NTM ヘッダが付加される．NAT$_{CN}$ は MN からのパケットを受信すると，マッピング情報に従ってアドレス／ポート変換を行った後，$RIP4_{CN}$ へ転送する．CN はカプセル化されたパケットを受信すると，NTM ヘッダに記載されている Path ID をキーにトンネルテーブルを検索し，該当エントリに従ってデカプセル化および復号処理を行う．その後，抽出したアプリケーションパケットを上位アプリケーションへ渡す．

以上により，MN と CN の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ことができる．また，NAT によるアドレス／ポート変換はカプセル化パケットの外側 IP ヘッダと UDP ヘッダに対して行われるため，MN と CN のアプリケーションは NAT に影響されることなく通信を行うことができる．

2.2.5 ネットワーク切り替え時の動作

NTM 端末の移動や無線インタフェースの切り替えにより実 IP アドレスが変化した際には，通信相手端末との間にトンネルを再構築する．このとき，通信相手の実 IP アドレスなどの情報は取得済みであるため，名前解決処理を省略して 2.2.3 項にて説明したトンネル構築処理を実行する．NTM 端末の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っているため，実 IP アドレスが変化してもその影響を受けることなく，通信を継続することができる．また，これと並行して 2.2.1 項にて説明した登録処理を行い，DC に登録したアドレス情報を更新する．常に最新のアドレス情報を DC へ登録することにより，NTM 端末に対する到達性を確保する．

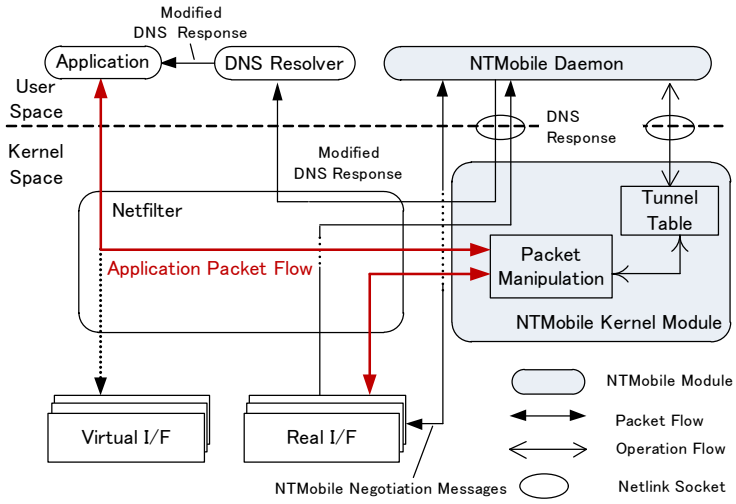

図 5 NTMobile 端末のモジュール構成
Fig. 5 Module configuration of NTMobile node.

3. 実装

3.1 NTMobile 端末の実装

NTMobile は Linux カーネルを搭載した Android 端末へ実装することを想定しており，これまでに Linux PC へ実装を行ってきた．**図 5** に NTM 端末のモジュール構成を示す．NTM 端末はカプセル化などを行うカーネルモジュール，ネゴシエーションを行うデーモンプログラム（以後 NTM デーモン），および仮想インタフェースを実装することにより動作する．アプリケーションの名前解決処理に対する DNS サーバからの応答メッセージは，カーネル空間において Netfilter によりフックし，Netlink Socket を用いて NTM デーモンへ渡される．その後，NTM デーモンは名前解決およびトンネル構築を実行する．

アプリケーションが送信したパケットは Netfilter の `NF_INET_POST_ROUTING` にてフックされ，カーネルモジュールでカプセル化および暗号化が行われる．その後，このパケットを Netfilter の `NF_INET_LOCAL_OUT` へ戻すことにより，実インターフェースからネットワークへ送信する．また，カプセル化されたパケットを受信した際には Netfilter の `NF_INET_PRE_ROUTING` にてフックし，デカプセル化および復号処理を行う．その後，抽出した元パケットを受信キューへ渡し，仮想インタフェースより受信する．一般的にカプセル化処理はオーバヘッドが高いことが知られているが，NTMobile ではパケットをカーネル空間にて直接操作することにより，カプセル化に伴うスループット低下を抑制している [4].

本稿では，Android OS を搭載したスマートフォンである Samsung 社の Galaxy S2 へ上記 NTMobile モジュールを実装した．NTMobile のカーネルモジュールのインストールや NTM デーモンを実行する際には root 権限が必要となるため，root 権限取得済みのファームウェアに更新した．

通常，Android 端末で使用されるアプリケーションは Dalvik と呼ぶ仮想マシン上にて動作するが，本稿では NTM デーモンをネイティブプログラムとして実装した．Galaxy S2 を含む多くの Android 端末は CPU として ARM プロセッサを採用しており，通常の Linux PC とは異なる CPU アーキテクチャである．そのため，NTM デーモンを ARM アーキテクチャ向けにクロスコンパイルし，Galaxy S2 へ実装した．

Galaxy S2 はカーネルのソースコードが公開されており，任意にカーネルの機能を有効にするなどの変更を加えることができる[*1]．NTMobile を実装する際には，カーネルのソースコードに変更を加える必要はないが，パケットのフックやカーネル空間とユーザ空間にて通信を行う際に，Netfilter や Netfilter Queue，Netlink の機能を必要とする．また，仮想インタフェースとして TUN デバイス，トンネル通信時の暗号化および認証に AES-CBC，HMAC-MD5 を使用している．そのため，Galaxy S2 に NTMobile を実装するにあたり，以下の機能を有効にしてカーネルを再構築した．なお，Linux カーネルのバージョンは 2.6.35.7 である．

- Network packet filtering framework
- Netfilter NFQUEUE over NFNETLINK interface
- “NFQUEUE” target Support
- AES cipher algorithms
- MD5 digest algorithm
- CBC support
- Universal TUN/TAP device driver support

[*1] `https://opensource.samsung.com/`

	RTT [ms]
DC_{MN} ～ DC_{CN}	0.28
DC_{MN} ～ RS	0.36
MN ～ DC_{MN}	4.45
MN_{moved} ～ DC_{MN}	31.18
CN ～ DC_{CN}	12.88
MN ～ CN	79.69
CN ～ RS	13.91
MN_{moved} ～ RS	32.04

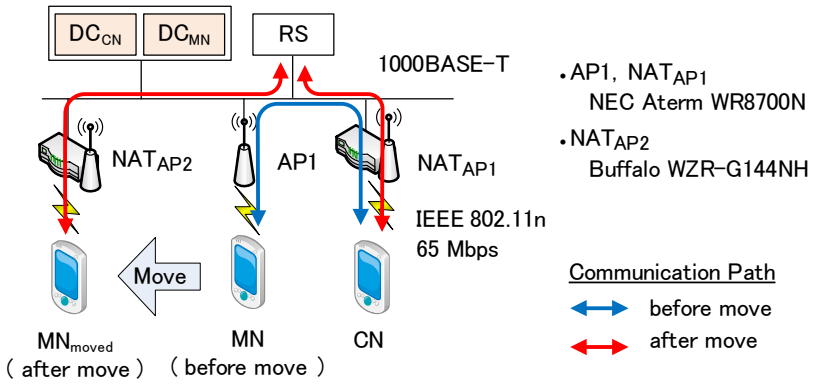

図 6 測定環境
Fig. 6 Evaluation environment.

表 1 装置仕様
Table 1 Device specifications.

	DC (Virtual Machine)	RS	MN, CN
Hardware	DELL PowerEdge R415	HP s5550j	Sumsung Galaxy S2 (SC-02C)
OS	Ubuntu 10.04 32bit	Ubuntu 10.04 32bit	Android 2.3.7
Linux Kernel	2.6.32.36	2.6.32.36	2.6.35.7
CPU	AMD Opteron 2.6GHz	Intel Core i7 2.93GHz	Samsung Exynos Orion Dual-core 1.2GHz
Memory	256MB	10GB	512MB

3.2 DC と RS の実装

DC と RS には，トンネル構築などのネゴシエーションを行うデーモンプログラムを実装した．また，DC は NTM 端末の情報を動的に管理するために Dynamic DNS サーバの機能を搭載した．DNS サーバのプログラムには，NTM レコードを登録できるよう改良した bind9[*1] 使用した．

RS にはデーモンプログラムに加えて，カーネル空間にトンネル通信の中継処理を行うためのカーネルモジュールを実装した．カーネルモジュールはパケットの転送機能に加えて，転送先の実 IP アドレスなどの転送情報を記載したリレーテーブルを保持する．カプセル化されたメッセージを受信した際には，NTM ヘッダに記載されている Path ID をキーにリレーテーブルを検索し，転送情報を取得する．その後，カプセル化パケットの宛先の IP アドレスやポート番号を転送先端末の情報に書き換え，送信元の IP アドレスを自身の IP アドレスに書き換え送信する．

4. 動作検証と評価

NTMobile を実装した Android 端末を用いて，提案方式による接続性の確立およびハンドオーバ時の動作検証と性能評価を行った．

4.1 測定環境

図 6 と**表 1** にネットワーク構成と各装置の仕様を示す．また，測定環境の特性を明確にするために，各端末間の平均 RTT（Round-Trip Time）を測定した．RTT の測定には ping を使用し，1 秒間隔で 64 バイトのパケットを 100 回送受信した際の平均値を示す．なお，2 台の DC は仮想マシンにより構築した．各アクセスポイント（AP）は市販のブロードバンドルータであり，NAT_{AP1} と NAT_{AP2} の配下はプライベートネットワークとして構築した．AP1 はルータ機能を無効にして，一般的なアクセスポイントとして動作させた．MN と CN は各 AP へ IEEE 802.11n にて接続し，暗号化および認証機能には WPA2-AES を使用した[*2]．また，ネゴシエーション時に使用する暗号化アルゴリズムとして AES-CFB，トンネル通信時には AES-CBC[*3] を設定し，認証アルゴリズムは HMAC-MD5，鍵長 128bit とした．

4.2 ネゴシエーションによるオーバヘッドの測定

通信開始時に発生するオーバヘッドを明らかにするために，AP1 に接続した MN が NAT_{AP1} に接続した CN の A レコードを問い合わせ，MN と CN 間に通信経路が確立されるまでに要する時間を測定した．測定用に A レコードの問い合わせのみを行う Android 端末向けのネイティブプログラムを作成し，A レコードの問い合わせ前と問い合わせ結果取得後のタイムスタンプの差分から，通信開始時に発

*1 http://www.isc.org/software/bind
*2 NAT_{AP2} は IEEE 802.11n ドラフト版準拠．
*3 Linux カーネルでは CFB を標準サポートしていないため，本稿では CBC を使用した．

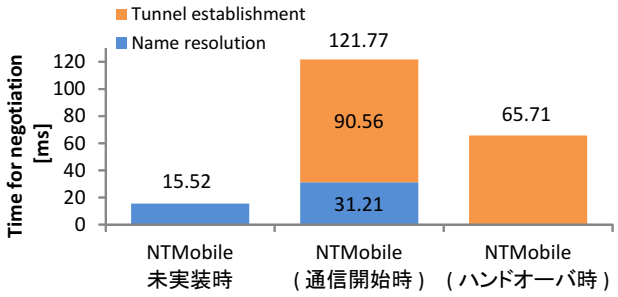

図 7 ネゴシエーションによるオーバヘッドの測定結果
Fig. 7 Results of overhead time by negotiation.

生するオーバヘッドを算出した．なお，NTMobie のネゴシエーション処理に要した時間を測定するために，NTM デーモンにイベントごとのタイムスタンプを出力するようプログラムを追加し，その差分から名前解決処理とトンネル構築処理に要した時間を算出した．

また，上記環境にて MN と CN 間にトンネルが構築された後，手動で MN の接続先を AP1 から NAT_{AP2} へ切り替え，ハンドオーバ時に発生する NTMobile の処理によるオーバヘッドを測定した．

100 回測定した平均値を**図 7** に示す．NTMobile 未実装時の測定結果は名前解決に要した時間である．ただし，この場合は NAT 配下端末に対する接続性を確保することができないため，A レコードの問い合わせのみが行われ，15.52 ms で処理が完了した．それに対して NTMobile 実装時には，通信開始時に行う名前解決処理に 31.21 ms を要した．これは A レコードの問い合わせを行った後に，NTM レコードなどを DNS サーバへ問い合わせるためである．

トンネル構築処理に要した時間は，通信開始時には 90.56 ms，ハンドオーバ時には 65.71 ms であった．通信開始時には MN と CN との間にエンドツーエンドでトンネルが構築され，ハンドオーバ後には RS との間にトンネルが構築されるため，通信開始時とハンドオーバ後とでは Tunnel Request/Response の送信相手が異なる．MN と CN 間，および MN_{moved} と RS 間では平均 RTT に 47 ms 程度の違いがあるため，トンネル構築処理に要した時間の差はこの違いにより生じたものと考えられる．

NTMobile を実装した Android 端末では，通信開始時に 120 ms 程度のオーバヘッドが発生することが分った．Akamai 社が行った調査によると，Web サイト閲覧時にエンドユーザが許容できる待ち時間は 2 秒程度とされている[*1]．Web サイトの閲覧は短時間で通信が完了するため移動透過性の必要性は低いが，スマートフォンの一般的な用途のひとつである．このような使用用途においても NTMobile によるオーバヘッドは十分に許容できる値であり，多くのアプリケーションでは実用上問題ないと考えられる．

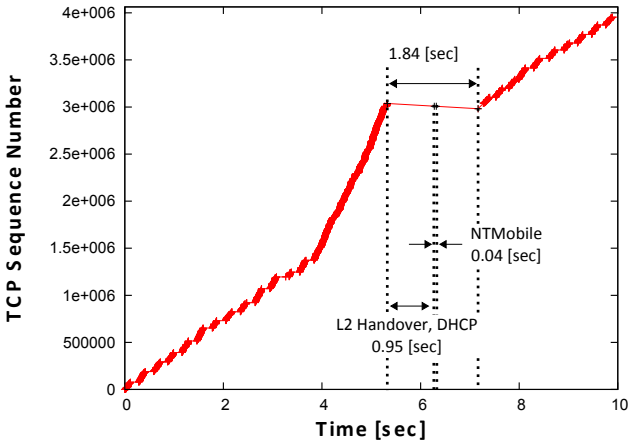

図 8 ハンドオーバによる TCP シーケンス番号の変化
Fig. 8 Changes of TCP sequence number by handover.

4.3 通信切断時間の測定

端末のハンドオーバ時には，L2 ハンドオーバや DHCP による IP アドレスの取得に起因する通信切断時間が発生する．これに伴い，パケットロスが発生してアプリケーション間の通信に影響が及ぶ可能性がある．そこで，ハンドオーバ時に発生する通信切断時間を測定した．AP1 に接続した MN から NAT_{AP1} に接続した CN へ iperf[*2]による TCP 通信を行い，通信中に MN の接続先を手動で AP1 から NAT_{AP2} へ切り替えた場合の TCP のシーケンス番号の変化を観測した．TCP のシーケンス番号の変化から，通信切断時間を明らかにする．パケットの送受信を観測するために，MN と CN へ LAN アナライザアプリケーションである Shark for Root[*3]をインストールした．MN 側では実インタフェースにて送受信されたパケット，CN 側では仮想インタフェースにて受信したパケットをキャプチャした．なお，キャプチャしたデータの解析には Wireshark[*4]を使用した．

ハンドオーバによる TCP のシーケンス番号の変化を**図 8** に示す．5.32 秒から 7.16 秒までの 1.84 秒間，アプリケーションによる通信が中断された．このとき，MN の接続先を AP1 から NAT_{AP2} へ手動で切り替えてから IP アドレスを取得するまでに約 950 ms，NTMobile によるトンネル再構築処理に要した時間は約 40 ms であった．また，トンネルの再構築が完了してから約 850 ms 後に，MN のアプリケーションが CN との通信を再開した．通信再開後に CN が受信したパケットの Wireshark による解析結果には，TCP の再送タイムアウト（RTO）として 1.84 秒が記載されていた．再送ごとに RTO が 2 倍されていくことから，このパケットが送信される 920 ms 前である 6.24 秒の

[*1] http://www.akamai.com/html/about/press/releases/2009/press_091409.html
[*2] http://sourceforge.net/projects/iperf/
[*3] https://play.google.com/store/apps/details?id=lv.n3o.shark
[*4] http://www.wireshark.org/

時点で MN から CN へパケットが送信されたと考えられる．この時点では DHCP による IP アドレスの取得処理が完了していないため，MN が送信したパケットは消失する．その後，RTO に従って 920 ms 後に再送が行われたため，7.16 秒の時点で通信が再開されたと考えられる．

文献 [8], [9] によると，無線 LAN の L2 ハンドオーバは 50〜400 ms，DHCP による IP アドレスの取得は 500 ms 程度で完了する．NTMobile によるトンネル構築処理が平均 65〜90 ms 程度で完了することから，通信中断時間のほとんどが L2 ハンドオーバや IP アドレスの取得に要する時間であることがわかる．そのため，ハンドオーバ時の通信中断時間を削減するためには L2 ハンドオーバや IP アドレス取得の高速化が重要である．

上記課題の解決策として，DHCP 処理の高速化 [9, 10] や異なる無線システム用いたシームレスハンドオーバ技術 [11] などが有用であると考えられる．また，Wi-Fi や 3G などの異なる無線インタフェースを使用し，ハンドオーバ前にネゴシエーションを完了することにより，シームレスハンドオーバを実現することも可能である [12]．この方法によると，ハンドオーバ時に発生する通信切断時間やパケットロスをなくすことができる．

5. まとめ

本稿では，移動透過性と NAT 越えを実現する NTMobile を Android 端末へ実装し，動作検証および性能評価を行った．動作検証により，NAT 配下端末に対する接続性および NAT を跨がった移動をした際にも通信を継続可能であることを確認した．また，NTMobile によるオーバヘッドは多くのアプリケーションにおいて許容できる値であり，Linux PC だけでなく一般的なスマートフォンでも実用可能であることを示した．ハンドオーバ時には L2 ハンドオーバや IP アドレス取得，トンネル構築処理に起因する通信中断時間が発生するため，シームレスハンドオーバ技術などを用いて通信中断時間をなくすための検討が必要である．

今後は IPv6 向けの実装を完了し，IPv4 と IPv6 が混在した環境において動作検証および性能評価を行う予定である．

参考文献

[1] Le, D., Fu, X. and Hogrefe, D.: A Review of Mobility Support Paradigms for the Internet, *IEEE Communications Surveys*, Vol. 8, No. 1, pp. 38–51 (2006).
[2] Soliman, H.: Mobile IPv6 Support for Dual Stack Hosts and Routers, *RFC 5555,IETF* (2009).
[3] 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，Vol. 2011, No. 1, pp. 1339–1348 (2011).
[4] 内藤克浩，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，DICOMO2011 論文集，Vol. 2011, No. 1, pp. 1349–1359 (2011).
[5] 西尾拓也，内藤克浩，水谷智大，鈴木秀和，渡邊　晃，森香津夫，小林英雄：NTMobile における端末アドレスの移動管理と実装，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，Vol. 2011, No. 1, pp. 1139–1145 (2011).
[6] 上醉尾一真，鈴木秀和，内藤克浩，渡邊　晃：IPv6 ネットワークにおける NTMobile の検討，情報処理学会研究報告，Vol. 2011-MBL-59, No. 9, pp. 1–8 (2011).
[7] 納堂　博, 鈴木秀和, 内藤克浩, 渡邊　晃：NTMobile の経路最適化の検討, 情報処理学会研究報告，Vol. 2011-MBL-61, No. 33, pp. 1–8 (2011).
[8] Arunesh Mishra, Minho Shin, W. A.: An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process, *ACMSIGCOMCom- puter Communication Review*, Vol. 33, No. 2, pp. 93–102 (2003).
[9] Forte, A. G., Shin, S. and Schulzrinne, H.: Improving Layer 3 Handoff Delay in IEEE 802 . 11 Wireless Networks, *WICON '06 Proceedings of the 2nd annual international workshop on Wireless internet*, No. 12, pp. 1–8 (2006).
[10] 小川猛志，　伊東匡：DHCP をベースとしたシームレスハンドオーバ方法の研究，電子情報通信学会論文誌，Vol. J88-B, No. 11, pp. 2228–2238 (2005).
[11] IEEE 802.21: MEDIA INDEPENDENT HANDOVER SERVICES, http://www.ieee802.org/21/.
[12] 福山陽祐，鈴木秀和，渡邊　晃：IPv4 移動体通信において携帯電話網と無線 LAN 間をシームレスに移動する方式の提案，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，Vol. 2011, No. 1, pp. 1115–1120 (2011).