目　次

概　要

電車内や自動車内にネットワークを構築し，そのネットワーク自体が移動しても，ネットワーク内の端末と外部端末との通信を継続できるネットワークモビリティの要求が高まっている．これまでネットワークモビリティを実現する技術はいくつか提案されているが，ネットワーク内のアドレスがグローバルアドレスであり，特殊サーバの設置が必要となる方式がほとんどで普及の妨げとなっている．本稿では特殊サーバの設置が不要でネットワーク単位の移動透過性を実現する Mobile Network to Peer Communication（Mobile NPC）について述べる．Mobile NPC は，我々がこれまでに提案してきた端末のみで移動透過性を実現した Mobile PPC と，端末・ルータのみの改造で NAT 越えを実現した NAT-f を組み合わせることで，ネットワーク単位の移動透過性を実現するものである．Mobile PPC と NAT-f との組み合わせた Mobile NPC を FreeBSD に実装して評価した結果，移動端末の集団をネットワークとして集約した効果と中継によるオーバヘッドがほとんどないことを示した．

1. はじめに

無線 LAN やインターネットの急速な普及により,ユビキタス社会を実現するために移動しながら通信ができる環境が要求されている．しかし，インターネットでは通信中の端末が移動すると IP アドレスが変化するために一般には通信が継続できない．そこで，移動によって IP アドレスが変わっても通信を継続できる移動透過性の研究が行われている[1]．また，電車内や自動車内にネットワークを構築し，そのネットワーク自体が移動するというケースも考えられる．この場合は，ルータが複数の端末に代わって移動透過性機能を実行し，ネットワーク内のアドレスはそのまま維持する方法が検討されている．これはネットワークモビリティと呼ばれ，移動にかかわる制御情報を減らし，かつ通信の中断時間を最小限にできる効果がある．

端末単位の移動透過性を実現する代表的な既存技術として，IPv4 対応の Mobile IP[2]と IPv6 対応の Mobile IPv6[2]がある．また，これらの技術を利用してネットワークモビリティを実現させた技術として, IPv4 Network Mobility Basic Support Protocol（以下 IPv4 NEMO）[4]と IPv6 対応の Network Mobility Basic Support Protocol（以下 NEMO）[5]がある．しかし IPv4 NEMO と NEMO は Mobile IP の技術を利用しているため，Home Agent（HA）のような特殊なサーバの設置が必要で，HA を介して通信を行うための通信経路の冗長や，トンネル化によるヘッダオーバヘッドなど，Mobile IP と同様の課題がある．Mobile IPv6 は経路最適化機能が追加されたことにより, 端末間でのエンドエンドの直接通信ができるようになったが，NEMO では必ず HA を経由する通信として定義されている.このような経路の冗長は,性能の劣化をもたらし，HA の多重化を考慮する必要があるなどの課題があって，今後の P2P 通信には適していない．

ところで，端末単位の移動透過性をエンドエンドで実現する既存技術として Location Independent Networking for IPv6（LIN6）[6]，Mobile IP with Address Translation（MAT）[7], および Mobile Peer to Peer Communication（Mobile PPC）[8]がある．これらの技術を応用してネットワーク単位の移動透過性を実現させた技術として χ LIN6-NEMO[9]と MAT-MONET[10]が提案されている．しかし，χ LIN6-NEMO は Mapping Agent（MA）と呼ぶ特殊なサーバを設置し，必ず MA を介した通信を行う形態となるため，ネットワークモビリティ実現時には通信経路の冗長にかかわる課題が解決できていない．また，MAT-MONET はネットワークモビリティ実現時においてもエンドエンドの通信が可能であり経路の冗長は発生しない．しかし，MAT-MONET は DNS の改造と IP address Mapping Server（IMS）と呼ぶ特殊なアドレス管理装置が必要

になるという課題がある．また，χLIN6-NEMO や MAT-MONET は IPv6 対応の技術であり，IPv4 の世界では利用できないという基本的な課題がある．既存のネットワークモビリティ技術として，唯一 NEMO は IPv4 にも対応しているが，前述のように Mobile IP と同様の課題がある他，移動ネットワーク内はグローバルアドレスでなければいけないという前提がある．IPv4 ではアドレス枯渇の問題が叫ばれており，移動ネットワーク内はプライベートアドレスを使用できることが望ましい．プライベートアドレスはアドレスの取得が不要で，アドレス管理が容易であるというメリットがある．ところが，ネットワーク内がプライベートアドレス空間であると，外部ネットワークから内部ネットワークのアドレスが隠蔽された形となり，外部端末から内部端末への通信開始ができない．これは NAT 越え問題と呼ばれており，IPv4 の汎用性を損なう一つの要因となっている．プライベートアドレスの使用が可能なネットワークモビリティ技術が現状では存在しない理由は，NAT 越え問題を解決できないことが原因となっている．現在主流の IPv4 に対応し，かつプライベートアドレスが利用できるネットワーク単位の移動透過性を実現できれば有用である．

そこで，本論文では特殊なサーバが不要で，かつネットワーク内にプライベートアドレスを使用可能な Mobile Network to Peer Communication（Mobile NPC）を提案する．Mobile NPC は Mobile PPC と NAT-f（NAT-free protocol）[11]を融合させた技術である．一般に移動透過性を実現するには，通信中に一方の端末が移動しても通信を継続できる通信継続性と，相手端末がどこへ移動しても通信の開始ができる移動ノード到達性の２つの機能を満たす必要がある．Mobile PPC は，特殊なサーバを利用することなく端末単位の通信継続性をエンドエンドで実現できる技術であり，この技術をベースに Mobile NPC の通信継続性を実現する．NAT-f は特殊なサーバを利用することなく NAT 越えを実現できる技術であり，この技術をベースに移動ノード到達性を実現する．

Mobile NPC を FreeBSD 上に実装し，ネットワークモビリティの実現を確認した．評価の結果，スループットの低下はほとんどなく，移動時にかかる時間が十分小さく，移動端末をネットワークに集約したことによる効果を示すことができた．

以下，2 章で Mobile PPC と NAT-f について記述し，3 章で Mobile NPC の詳細と DDNS サーバに登録する方法を記述する．4 章で Mobile NPC の実装，5 章で評価の結果を考察，6 章で既存技術と比較する．最後に 7 章でまとめる．

2. Mobile PPC と NAT-f の概要

以下に Mobile PPC と NAT-f の概要を述べる.

2.1. Mobile PPC

Mobile PPC は移動ノード到達性と通信継続性を明確に分離した点に特徴がある. 移動ノード到達性にはすでに普及している Dynamic DNS（DDNS）[12]サーバを利用することとし, 通信継続性に係る機能を Mobile PPC で実現する.

Mobile PPC の処理を図 1 に示す. Correspondent Node（CN）と Mobile Node（MN）は既に通信を行っている. CN と MN は IP 層内にアドレス変換テーブル（Connection ID Table : CIT）を保持する.

MN が通信中に移動して, IP アドレスが変わると, MN と CN は, 移動の通知（CIT Update : CU）と応答（CU reply）によるネゴシエーションを行い, CIT を更新する.

CIT 更新後は全パケットに対し, CN と MN の IP 層内で CIT に基づくアドレス変換を行う. この方式により, IP 層において正しくルーティングされるようにアドレス変換し, 上位ソフトウエアに対してはアドレスの変化を隠蔽することができる.

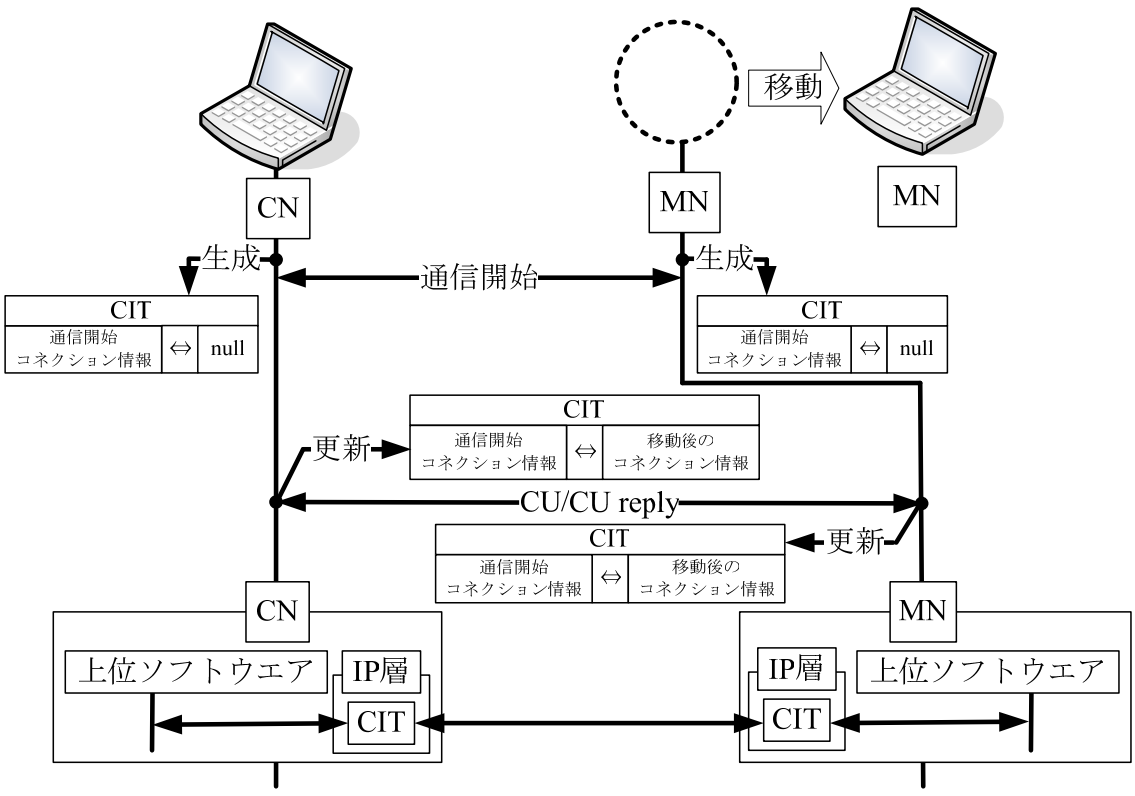


図 1　Mobile PPC の処理

2.2. NAT-f

NAT-f はグローバルアドレスとプライベートアドレス混在の環境下において移動ノード到達性を実現した技術と位置づけられる．図 2 に NAT-f の処理を示す．DDNS サーバはワイルドカード機能を有効にし，NAT-f ルータのホスト名とグローバルアドレスを登録しておく．

CN は NAT-f ルータの Fully Qualified Domain Name（FQDN）に NAT-f ルータ配下の端末 Private Node（PN）の Private Host Name（PHN）を付加して DDNS サーバに名前解決を行う．DDNS サーバはワイルドカード機能より NAT-f ルータのグローバルアドレスを応答する．応答を受信した CN は IP 層で応答パケットの NAT-f ルータのグローバルアドレスを仮想アドレスに書き換えて上位ソフトウエアに渡す．仮想アドレスとは CN の上位ソフトウエアが NAT-f ルータ配下のどの PN と通信するのかを区別する仮想的なアドレスである．

次に CN と NAT-f ルータは通信に先立つネゴシエーション処理を実行する．これにより NAT-f ルータは強制的に NAT テーブルを生成する．ここで NAT-f

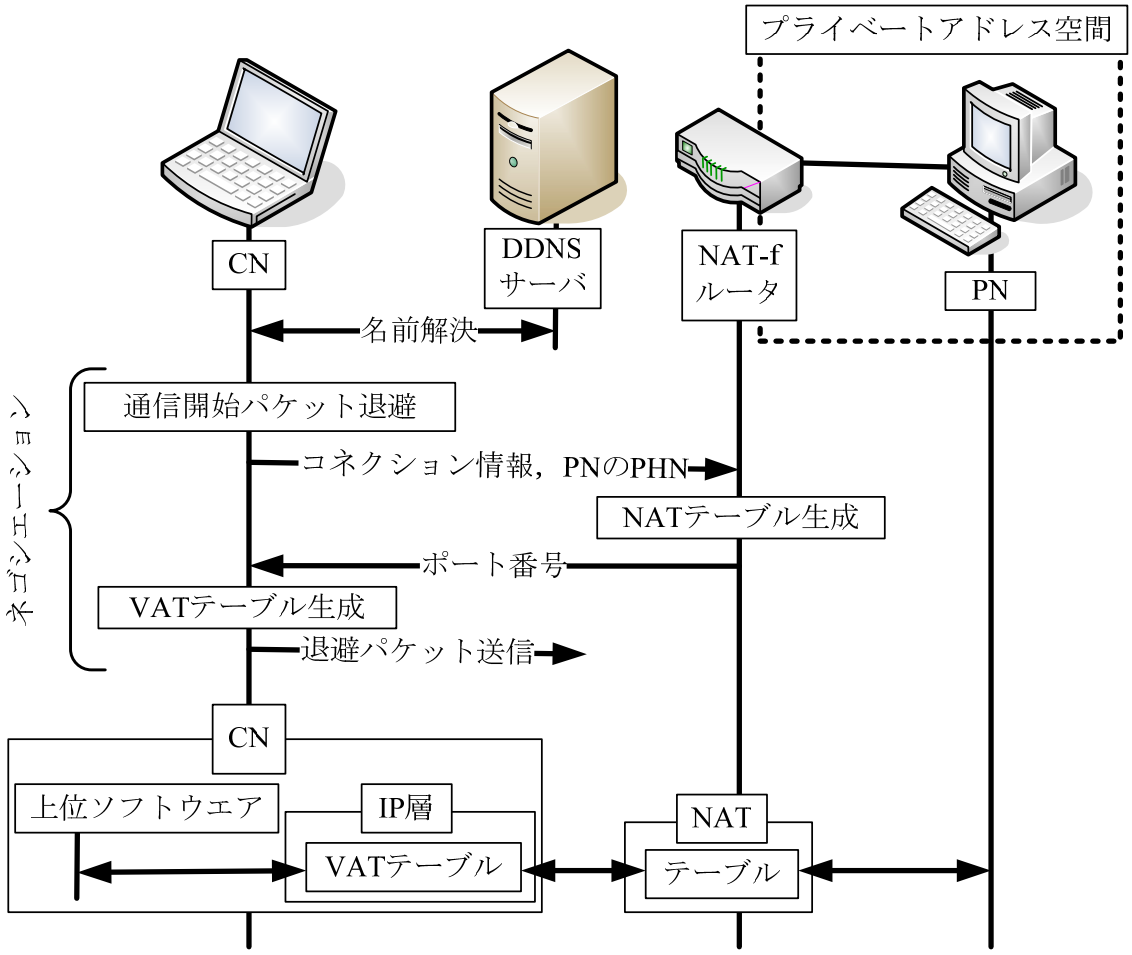


図 2　NAT-f の処理

ルータは NAT が生成したテーブルのポート番号を CN に通知する．CN が通知を受信すると，このポート番号に合わせたポート変換テーブル Virtual Address Translation（VAT）テーブルを IP 層内に生成する．以降の通信は CN 内の VAT テーブル及び NAT-f ルータ内の NAT テーブルでアドレス・ポート番号が変換されることにより通信が行われる．

このようにして CN 側から NAT-f router 配下に存在する端末 PN に対して通信を開始することができる．

3. Mobile NPC の提案

3.1. 概要

Mobile NPC の移動ネットワークは，Mobile PPC と NAT-f を組み合わせた Mobile NPC，Network Address Port Translation（NAPT）[13]を実装した Mobile NPC Router（MPR）によりインターネットと接続される．移動ネットワーク内は IPv4 のプライベートアドレス空間とし，複数の一般端末 Private address Terminal（PT）が存在できる．Mobile PPC と NAT-f を組み合わせたことで同様の機能である Mobile PPC の CIT と NAT-f の VAT を統合して Port Address translation Table（PAT）を定義した．Mobile NPC は通信を開始する方向により処理と生成される PAT が異なるという特徴がある．以下に PT から CN に向けて通信を開始する場合と CN から PT に向けて通信を開始する場合を述べる．

3.2. PT から CN に通信開始

PT から CN に向けて通信を開始する場合，通信開始，移動時，移動後のアドレス変換処理を以下に示す．

3.2.1. 通信開始処理

PT は通信開始パケットを CN に送信する．受信した MPR は送信元アドレスを PT のアドレスから MPR のグローバルアドレスに変換する NAPT テーブルを生成する．生成後，パケットの送信元アドレスを MPR のグローバルアドレスに変換して IP 層に渡す．2.1 で述べたように MPR と CN の IP 層内で CIT と同様に PAT が生成されて通信が開始される．

3.2.2. 移動情報の通知とアドレス変換処理

通信中に移動ネットワークが移動して MPR のグローバルアドレスが変わると，2.1 節で述べた Mobile PPC の手順に従い PAT を更新する．このとき

NAPT テーブルにはいっさい影響がない．PAT の内容は，MPR 移動前後のアドレスを変換する旨記述される．また MPR が移動して取得したアドレスを DDNS サーバに登録する処理は 3.4.2 節で述べる．

移動後のアドレス変換処理を示す．例として PT から CN へパケットを送信する場合を述べる．PT からのパケットを受信した MPR は NAPT テーブルを参照して，送信元アドレスを MPR のグローバルアドレスに変換し，MPR の IP 層へ渡す．2.1 節で述べたように IP 層で PAT を参照してアドレス処理が行われてパケットを MPR から CN に送信する．逆方向のパケットは逆の変換を行う．

3.3. CN から PT に通信開始

CN から PT に向けて通信を開始する場合，DNS 名前解決，ネゴシエーション，移動時，移動後の変換処理を以下に示す．

3.3.1. DNS 名前解決とネゴシエーション処理

2.2 節の DNS 名前解決で述べたように MPR の FQDN に通信相手 PT の PHN “bob” を付加して DNS 名前解決の処理を行う．DNS 名前解決後のネゴシエーション処理を図 3 に示す．上位ソフトウエアは DNS 名前解決で取得したアドレス V1 宛でパケットを送信する．IP 層にパケットが渡されると PAT を生成し，通信開始パケットを退避する．MPR のアドレスと退避させたパケットの CN のアドレス，ポート番号，プロトコル番号，通信相手 PT の PHN“bob” の情報を MPR に送信する．受信した MPR は受信した情報から強制的に NAT

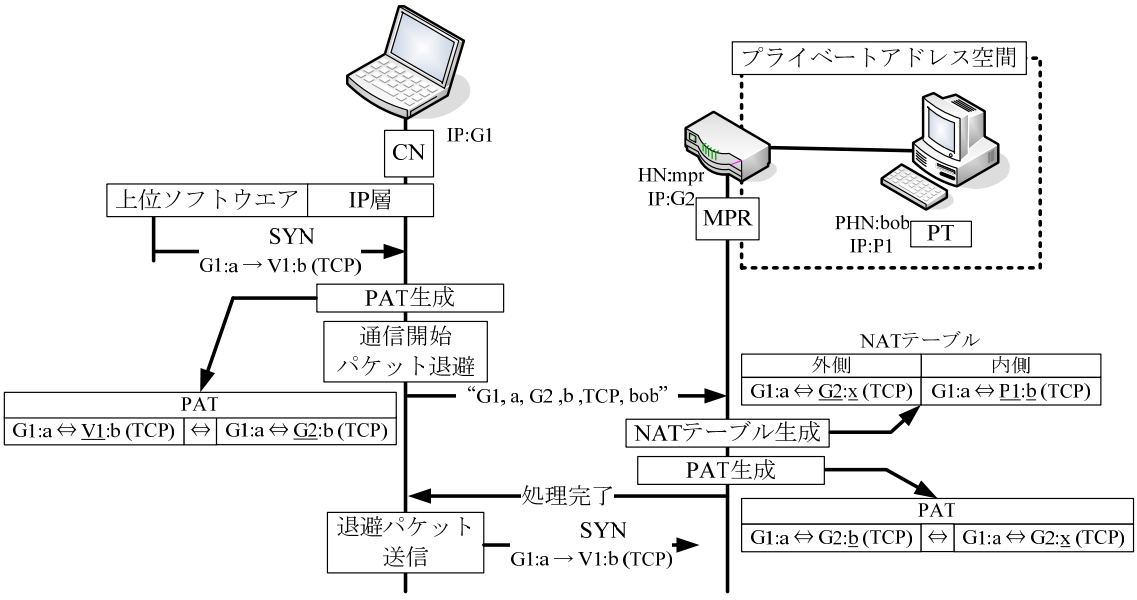


図 3　ネゴシエーション処理

テーブルを生成する．生成後，NAT テーブルの外側のポート番号 x から b に変換する PAT を生成する．その後，処理完了を CN に送信する．受信した CN は退避したパケットを送信する．

ネゴシエーション処理後の CN から PT への通信は CN の IP 層で PAT を参照して宛先の仮想アドレス（V1）を MPR のグローバルアドレス（G2）に変換して MPR に送信する．受信した MPR では PAT を参照して宛先ポート番号を b から x に変換して NAT に渡す．NAT では強制的に生成した NAT テーブルを参照して宛先アドレス：ポート番号を G2：x から P1：b に変換して PT に送信する．PT から CN への通信は上記と逆の変換を行う．このようにして CN は MPR 配下の端末 PT と通信開始することができる．

3.3.2. 移動情報の通知と変換処理

通信中に移動ネットワークが移動して MPR のグローバルアドレスが G2 から G3 に変わると，3.2.2 節同様に PAT を更新する．移動後の変換処理を図 4 に示す．CN の上位ソフトウエアが PT に送信すると IP 層に渡される．IP 層で PAT を参照して宛先の仮想アドレス（V1）を MPR 移動後のアドレス（G3）に変換して MPR に送信する．受信した MPR は IP 層で PAT を参照して宛先アドレス：ポート番号を G3：b から G2：x に変換して NAT に渡す．NAT は NAT テーブルを参照して宛先アドレス：ポート番号を G2：x から P1：b に変換して PT に送信する．PT から CN への通信は上記と逆の変換を行う．

このようにして通信開始が PT から CN または CN から PT でも通信中の CN

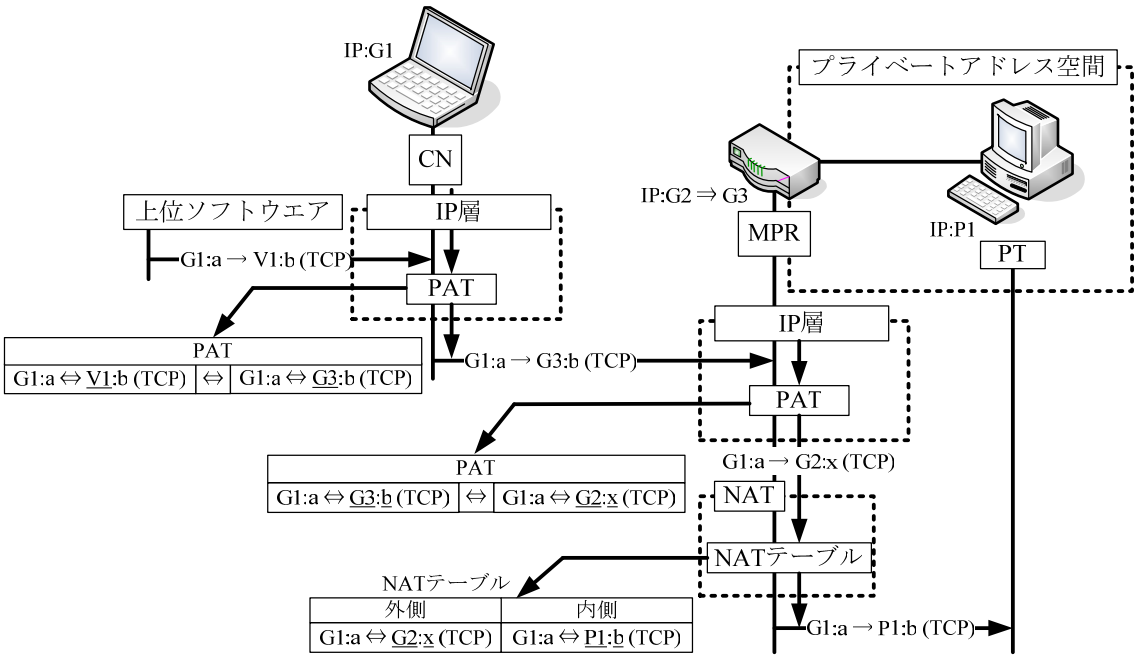


図 4　移動後の変換処理（CN ⇒ PT）

と PT はネットワークが移動しても，CN の上位ソフトウエアと MPR の NAT はアドレスの変化に気づかず，コネクションを維持することができる．

3.4. DDNS 登録

Mobile NPC は移動ノード到達性を実現するために既存の DDNS サーバをそのまま利用する．このため CN または MPR は現在のアドレスを DDNS サーバに登録し，登録の際には認証を行う必要がある．また CN または MPR が移動してアドレスが変わると，登録したアドレスを移動後のアドレスに更新する必要がある．以下に認証の方式の選定とアドレスを DDNS サーバに登録する方法を示す．

3.4.1. 認証

DDNS Update の認証を行う既存技術として SIG[14]と TSIG[15]がある．SIG は公開鍵，TSIG は共有秘密鍵を用いて認証を行う．また TSIG には共有秘密鍵の更新を自動で行うことができる TKEY[16]がある．そこで認証には処理が軽く，鍵の更新を自動で行うことができる TSIG を利用する．

3.4.2. 登録方法

CN または MPR の FQDN を登録している DDNS サーバにアドレスを登録するする処理はアプリケーションとして実装する．DDNS サーバの登録には BIND（Berkeley Internet Name Domain）[17]に付属する nsupdate コマンドを実行する．以下に CN または MPR の初期立ち上げ時と移動時の処理を述べる．

(1) 初期立ち上げ時の処理

CN または MPR の初期立ち上げ時にアプリケーションが実行される．アプリケーションは現在割り当てられているアドレスを取得し，nsupdate コマンドを実行する．nsupdate は FQDN，取得したアドレス，共有秘密鍵で署名したデータを含んだ DDNS Update パケットを DDNS サーバに送信する．受信した DDNS サーバは共有秘密鍵を用いて署名データの認証成功後，Resource Records（RR）を追加する．RR がすでに登録されている場合は RR を更新する．

(2) CN 移動時の処理

CN 移動時の処理を図 5 に示す．CN と CN の FQDN を登録している DDNS サーバは認証用の共有秘密鍵 kCN を事前に保持する．CN が移動してアドレスが Y0 から Y1 に変わると，アプリケーションはアドレスの変化を検知する．

検知するとアプリケーションは nsupdate コマンドを実行する．nsupdate は bob.example1.com，CN のアドレス Y1，共有秘密鍵 kCN で署名したデータを含んだ DDNS Update パケットを DDNS サーバに送信する．受信した DDNS サーバは kCN を用いて署名データの認証成功後，RR のアドレスを Y0 から Y1 に更新する．

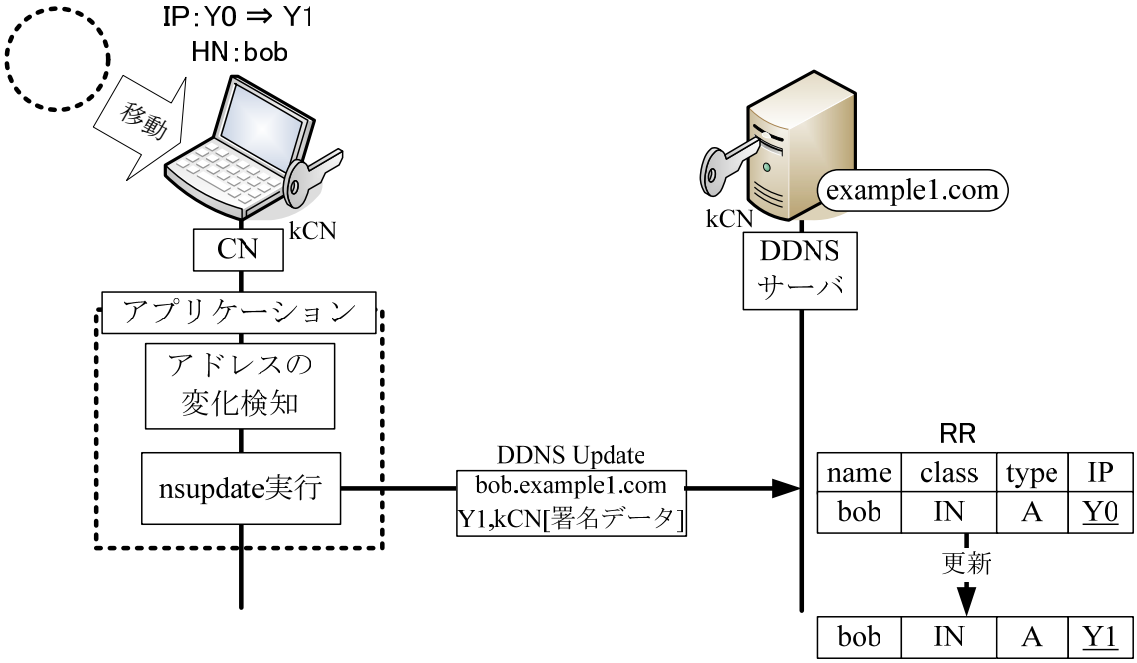


図 5　CN 移動時の処理

(3) MPR 移動時の処理

MPR 移動時の処理を図 6 に示す．MPR と MPR の FQDN を登録している DDNS サーバは認証用の共有秘密鍵 kMPR を事前に保持する．MPR が移動してアドレスが X0 から X1 に変わると，アプリケーションはアドレスの変化を検知する．検知するとアプリケーションは nsupdate コマンドを実行する．nsupdate は mpr.example2.com，MPR のアドレス X1，共有秘密鍵 kMPR で署名したデータを含んだ DDNS Update パケットを DDNS サーバに送信する．受信した DDNS サーバは kMPR を用いて署名データの認証成功後，RR の IP アドレスを X0 から X1 に更新する．MPR のワイルドカードの RR も同様に更新する．

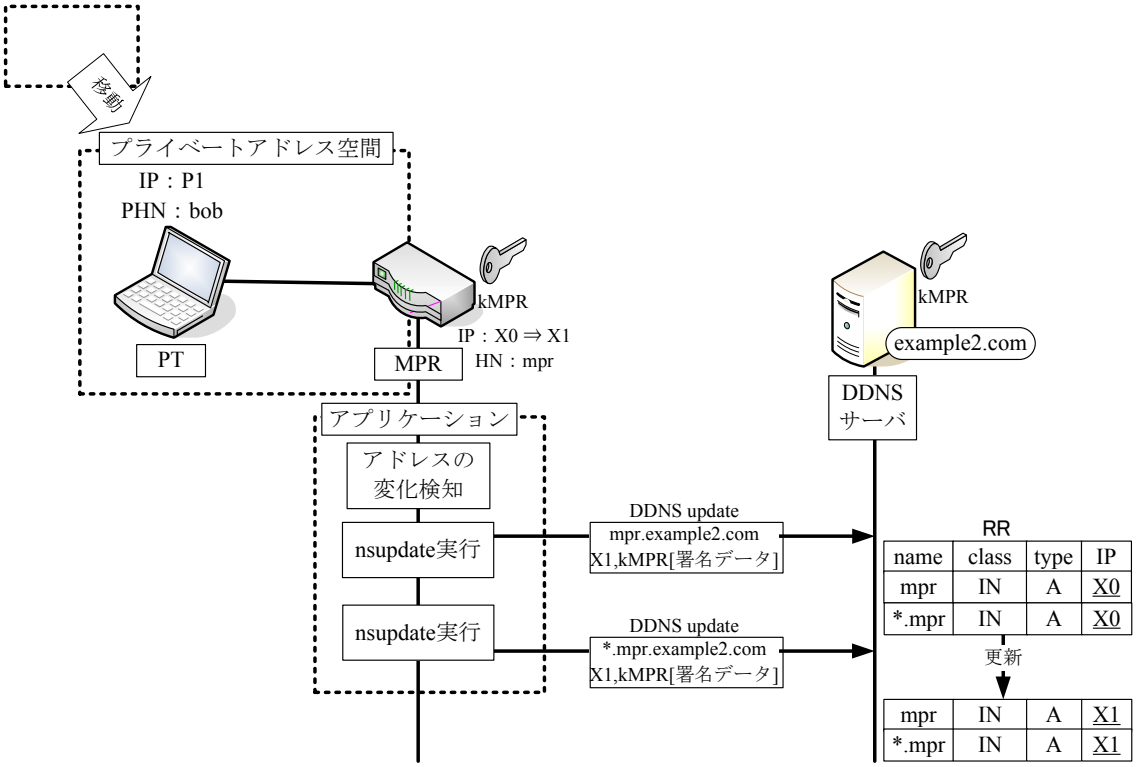


図 6　MPR 移動時の処理

4. Mobile NPC の実装

Mobile NPC を FreeBSD5.3 上に実装し動作検証を行った. NAPT は FreeBSD に標準にインストールされている natd を利用した．本章では Mobile NPC のモジュール構成と natd の改造について記述する.

4.1. モジュール構成

各モジュール構成を図 7 に示す. 既存の処理に変更を与えないようにパケット受信時におよびパケット送信時にパケット判定モジュールを呼び出す. パケット判定モジュールはグローバル側のインターフェースの場合に PAT モジュール，NAT-f モジュールを呼び出す．Mobile PPC モジュールと NAT-f モジュールは変更と削除したモジュールのみ記述した.

Mobile PPC モジュールの移動管理モジュールを CIT から PAT を利用するように変更し，CIT 操作モジュールとアドレス変換モジュールを削除した．移動してアドレスが変わったときに移動管理モジュールが呼ばれ，PAT を参照して CU/CU reply を行い，PAT を更新する.

NAT-f モジュールのネゴシエーションモジュールを VAT から PAT を利用するように変更し，VAT 操作モジュールを削除した．また宛先が仮想アドレスで通信開始されるときにネゴシエーションモジュールが呼ばれネゴシエーシ

ョンを行い，変換する PAT を生成する．

PAT モジュールはパケット変換モジュールと PAT 操作モジュールがある．パケット変換モジュールはパケットを送信・受信する度に呼ばれる．新しい通信開始時に PAT を生成し，ポート番号やアドレス変換が必要なパケットの場合は変換処理を行う．

natd はソケットバッファを介してアドレス変換を行う．

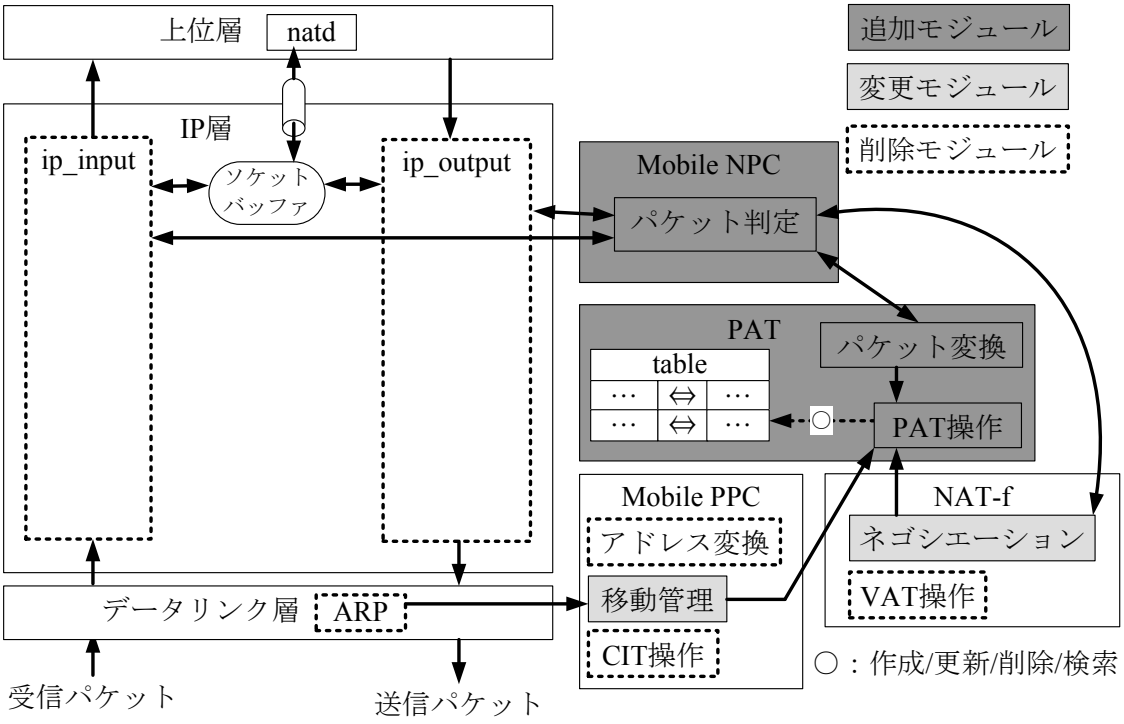


図 7　モジュール構成

4.2. natd の改造

natd ではグローバル側のインターフェースに割り当てられているアドレスを常に監視しており，そのアドレスが新しいアドレスに変わると保持しているアドレス変換テーブルをすべて削除する．このテーブルを削除しないように改造することでアドレスが変化しても通信中のパケットは古いアドレスでアドレス変換される．

natd はプライベートアドレスからグローバルアドレスに変換した後，移動後のグローバルアドレスでチェックサムの差分計算を行う．しかし，変換されたグローバルアドレスは移動前のアドレスであるためチェックサムの値が異なり破棄される問題が発生する．そこで natd がアドレス変換とチェックサムの計算を行った後に，チェックサムの値が正しいかチェックを行いチェックサムの値が正しくない場合は再計算するように改造した．

5. 評価

5.1. 評価環境

図 8 に示す評価環境で動作検証を行った．機器構成を表 1 に示す．CN1～3 及び MPR に Mobile NPC を実装した．NIC はすべて 100base-TX である．MPR の移動は手動で LAN ケーブルをルータ 1 からルータ 2 またはルータ 2 からルータ 1 につないだ後にアドレスの取得を行った．アドレスの取得には FreeBSD に標準にインストールされている dhclient コマンドを実行してルータ 1 またはルータ 2 からアドレスの取得した．

MPR の配下の PT が，CN と通信中に MPR または CN が移動してアドレスが変わっても PT と CN の通信を継続することを確認した．また CN からプライベートアドレス空間の端末 PT に通信開始することができ，通信中に MPR または CN が移動してアドレスが変わっても CN と PT の通信を継続することを確認した．

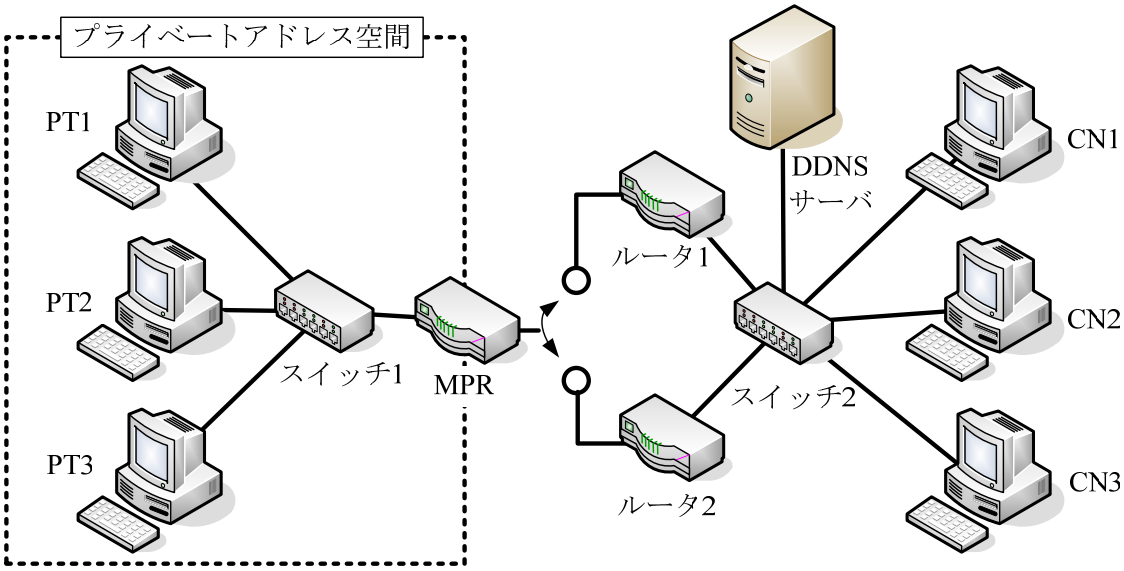


図 8　評価環境

表 1　機器仕様

	CPU	Memory	OS
MPR	Pentium4 3.0GHz	1GB	FreeBSD5.3
CN1～3	Pentium4 3.0GHz	1GB	FreeBSD5.3
PT1～3	Pentium4 3.4GHz	1GB	Windows XP Professional
ルータ 1,2	Pentium4 3.0GHz	1GB	FreeBSD5.3
DDNS サーバ	Pentium4 2.4GHz	256MB	FreeBSD5.3

5.2. スループット

Mobile NPC を実装した場合に MPR の処理が中継性能に与える影響を調査するため PT から CN への通信と CN から PT への通信のスループットを測定した．測定には Iperf[18]を用い 30 秒間の通信を 10 回試行してその平均をとった．

MPR に Mobile NPC を実装しなかった場合と実装した場合のスループットを測定した．表 2 は PT 側から通信を開始し，通信端末のペア数を 1 から 3 まで増加させた場合，表 3 は CN 側から通信を開始し，通信端末数を 1 から 3 まで増加させた場合の測定結果である．表 2 における Mobile NPC 未実装時は IP フォワードの設定により CN から PT への通信開始を可能とした．

これらの結果から MPR が Mobile NPC 機能を保持したことによる通信のオーバヘッドの増加はほとんどないことが分かる．

ここで CN から PT への通信より PT から CN への通信のスループットが高い理由は，PT から送信されるパケットのチェックサム計算を PT の NIC で行っているからである．

表 2　スループット（PT ⇒ CN）

通信端末のペア数	未実装	実装
1	94.9	94.7
2	47.5	47.5
3	31.7	31.7

単位：Mbps

表 3　スループット（CN ⇒ PT）

通信端末数	未実装(IP フォワード)	実装
1	94.1	94.1
2	47.1	47.1
3	31.3	31.4

単位：Mbps

5.3. 通信切断時間

MPR が移動してから通信の継続が可能になるまでの通信切断時間を図 9 に示す．測定には Read Time Stamp Counter（RDTSC）[19]を用いた．

測定結果は MPR の移動を 10 回行ったときの平均である．通信切断時間は 9.6sec で内訳を次に示す. 内訳は MPR の移動の LAN ケーブルの切り替え(L2 ハンドオーバ)に 3sec, dhclient の動作に 5.1sec, gateway への ARP request/reply に 1.5sec，CU/CU reply に 442 μ sec であり，CU/CU reply の処理時間は，ほとんど無視できる．dhclient の動作の内訳は DHCP Discover を送信するまでの Interval が 4sec，DHCP Discover を送信してから 2 重アドレスチェック用の

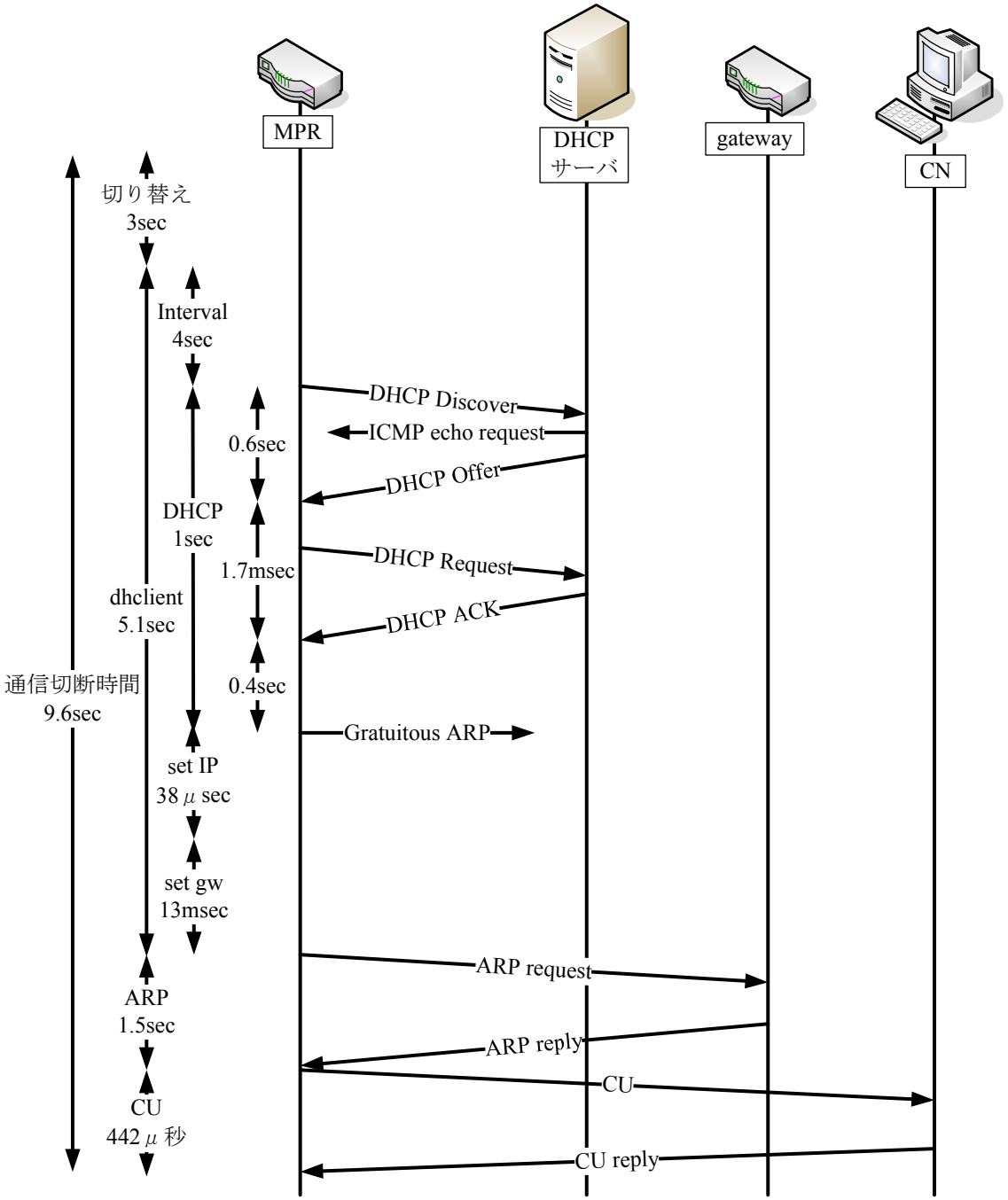


図 9　通信切断時間

Gratuitous ARP を送信するまでの DHCP の動作に 1sec，DHCP 動作後 DHCP から取得したアドレスをインターフェースに設定するまで 38μsec（set IP），アドレス設定後 DHCP のオプションで取得したデフォルトゲートウエイがルーティングテーブルに追加させるまでに 13msec（set gw）であった．DHCP の動作の内訳は DHCP Discover を送信してから DHCP Offer を受信するまでに 0.6sec, DHCP Offer を受信してから DHCP ACK を受信するまでに 1.7msec，DHCP ACK を受信してから２重アドレスチェック用の Gratuitous ARP を送信するまで 0.4sec であった．

この結果から通信切断時間の 9.3sec 中 LAN ケーブルの切り替え（L2 ハンドオーバ）の 3sec と dhclient の動作の Interval の 4sec の計 7sec がほとんどの割合を占めかつ通信切断時間中で無駄な時間である．また gateway への ARP request/reply に 1.5sec も要したのは，ARP request パケットは実際にはすぐに送信されずに退避された状態で２重アドレスチェック用の Gratuitous ARP のタイムアウト後に送信するからである．

MPR の移動を手動で LAN ケーブルの切り替え（L2 ハンドオーバ）を行ったが実際は無線 LAN の L2 ハンドオーバの時間は 50～400msec[20]である．また dhclient は同時に複数のクライアントのアドレス要求で要求がロックされないように乱数時間分一時退避する．このためすぐ DHCP Discover を送信せずに Interval が発生する．そこで dhclient コマンド実行後にすぐ DHCP Discover を送信するように改造すれば Interval をなくすことできる．これらから通信切断時間を 2.6sec～3sec に短くすることができる．また，実運用において数秒オーダーの通信切断を許容しないアプリケーションが存在すると考えられる．文献[21]では無線 LAN カードを 2 枚搭載することで，パケットロスを回避する方式を提案している．そこで MPR のグローバル側に無線 LAN カードを 2 枚搭載することで通信切断時間を無くす方法が考えられる．

5.4. CU 処理時間

5.3 節において CU 処理時間は，ほとんど無視できる時間であった．しかし、MPR が移動すると，通信中のすべての CN に対して CU/CU reply を行う必要がある．そこで通信中の CN が増加したときの CU 処理時間を測定し，結果を図 10 に示す．RDTSC で測定し，通信中の CN を３台まで増加し，それぞれの CN に対して CU/CU reply を 10 回行ったときの平均である．通信中の CN が 2 台の場合は 1 台の場合の処理時間から約 80μsec 増加，3 台の場合は約 110μsec 増加した．

この結果から通信中の CN が増加しても処理時間の増加はわずかである．CU 処理時間がこのように短時間で実現可能なのはシーケンスがシンプルで

あることと処理をカーネル内で実行していることによる．また通信中のすべての CN に対して CU の処理が完了することなく，CU が完了した CN は通信を継続できる．以上の結果から Mobile NPC による集約効果は大きいと判断できる．

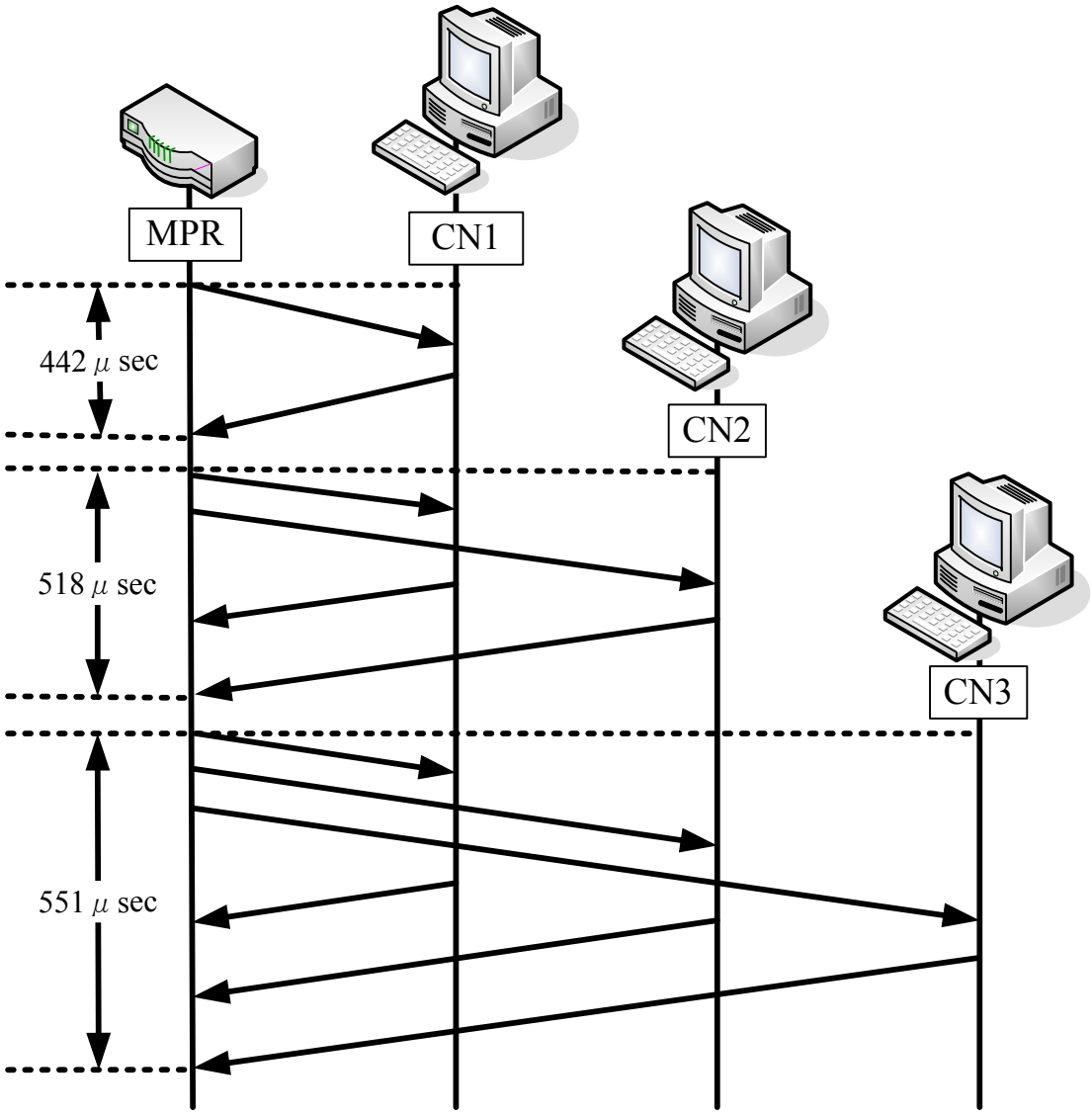


図 10　通信切断時間

6. 既存技術との比較

NEMO，χLIN6-NEMO，Mobile NPC の比較を表 4 に示す．NEMO は HA，χLIN6-NEMO は MA を経由して通信するため通信経路の冗長が発生する．NEMO は HA と MR 間で双方向トンネル化，χLIN6-NEMO は未実装の CN が MR 配下の固定ノードと通信する場合に MA と MR 間のトンネル化によるヘ

ッダオーバヘッドが発生する．χLIN6-NEMO は CN が MR 配下の固定ノードと通信する場合に CN に実装が不要だが，MN の場合には実装が必要となる．Mobile NPC は CN に実装が必要となるが，未実装の CN に対して通信継続する方法として文献[22]を提案している．NEMO には HA，χLIN6-NEMO には MA という特殊サーバを設置する必要がある．χLIN6-NEMO は IPv6 に基づいているため IPv4 に対応できない．Mobile NPC は現在主流の IPv4 を対象としている．χLIN6-NEMO は縮退アドレスモデルであるためアドレスに制約がある．NEMO とχLIN6-NEMO の移動ネットワークはグローバルアドレスであるため指定されたアドレスしか割り当てることができない．Mobile NPC は移動ネットワーク内のアドレスを自由に割り当てることができ，アドレスに制約がないため移動ネットワーク内のアドレス管理が容易である．

表 4　既存技術の比較

	NEMO	χLIN6-NEMO	Mobile NPC
通信経路	△	△	○
ヘッダオーバヘッド	△	○*	○
CN への実装	○	○*	△
特殊サーバ設置	×	×	○
IP v4	○	×	○
IP v6	○	○	×
アドレス制約	○	×	○
アドレス管理・制限	×	×	○

*：通信条件により異なる

7. まとめ

本論文では Mobile PPC と NAT-f を組み合わせた Mobile NPC と NAPT を実装させた MPR を用いることによりネットワーク単位の移動透過性を実現できることを示し，その実装方法と評価結果を述べた．Mobile NPC は特殊なサーバを必要とせず，常に最適な通信経路で通信が行われ，ヘッダオーバヘッドが発生しない．ネットワーク移動時の性能を測定し，移動端末の集団をネットワークとして集約した効果が大きいことを示し，中継によるオーバヘッドがほとんどないこと確認した．また端末やネットワークが移動して取得したアドレスを DDNS へ登録する方法を述べた．

CN は NAT-f により MPR 配下に移動した MN に通信の開始やその逆の MN

から CN への通信を開始することができる. しかし Mobile NPC の移動ネットワークはプライベートアドレスであるため通信中の MN が MPR 配下に移動した場合に MN は通信継続することができない．そこでグローバルアドレス空間に存在する MN がプライベートアドレス空間に移動しても通信継続する方法として文献[23]を提案している．また Mobile NPC は通信中に移動しても通信の継続はできるが，アプリケーションで移動前のアドレスを保持した状態で移動してアドレスが変わった後に保持したアドレスを使ってソケットをオープンしようとしてもそのアドレスはすでに使われていないためソケットをオープンできない．また宛先端末が移動した後，アプリケーションは移動前のアドレスを保持しているので宛先を移動前のアドレスで通信開始する場合に宛先アドレスが移動前であるためパケットが移動後の端末に到達できずに通信開始することができない．これらの課題があるためファイル交換によく使用される FTP[24]の利用に制限がある.

今後は，DDNS 登録の実装，移動前アドレスでソケットのオープン可能にする方法や宛先を移動前のアドレスで通信開始しても移動後の端末に到達できる方法の検討を行う予定である.

謝辞

本研究の遂行にあたり，ご助言とご指導をいただきました名城大学理工学部情報工学科 渡邊晃教授に厚く御礼申し上げます.

本論文を副査していただきました名城大学理工学部情報工学科 小川明教授，柳田康幸教授，宇佐見庄五講師に厚く御礼申し上げます.

最後に本研究の遂行にあたり名城大学理工学部情報工学科渡邊研究室の皆様から貴重なコメントをいただき心より感謝いたします.

参考文献

[1] 寺岡文男：インターネットにおけるノード移動透過性プロトコル，電子情報通信学会論文誌，Vol.J87-D1, No.3, pp.308−328 (2004).

[2] Perkins, C.: IP Mobility Support for IPv4, RFC3344, IETF (2002).

[3] Johnson, D., Perkins, C. and Arkko, J.: Mobility Support in IPv6, RFC3775, IETF (2004).

[4] Leung, K., Dommety, G., Narayanan, V. and Petrescu, A.: IPv4 Network Mobility (NEMO) Basic Support Protocol, Internet-Draft, IETF (2006).

[5] Devarapalli, V., Wakikawa, R., Petrescu, A. and Thubert, A.: Network Mobility (NEMO) Basic Support Protocol, RFC3963, IETF (2005).
[6] Ishiyama, M., Kunishi, M., Uehara, K., Esaki, H. and Teraoka, F.: LINA: A New Approach to Mobility Support in Wide Area Networks, IEICE Transactions on Communications, Vol.E84-B, No.8, pp.2076–2086 (2001).
[7] 相原玲二，藤田貫大，前田香織，野村嘉洋：アドレス変換方式による移動透過インターネットアーキテクチャ，情報処理学会論文誌， Vol.43, No.12, pp.3889–3897 (2002).
[8] 竹内元規，鈴木秀和，渡邊 晃：エンドエンドで移動透過性を実現する Mobile PPC の提案と実装，情報処理学会論文誌， Vol.47, No.12, pp.3244-3257 (2006).
[9] Banno, A., Oiwa, T. and Teraoka, F.: χLIN6-NEMO: A network mobility protocol based on LIN6, IEICE Trans. Commun., Vol.E89-B, No.4, pp.1070-1079 (2006).
[10] 藤田貴大，野村嘉洋，西村浩二，前田香織，相原玲二：MAT によるモバイルネットワークの実現，マルチメディア，分散，協調とモバイル（DICOMO2003）シンポジウム 2003 論文集, pp.105-108 (2003).
[11] 鈴木秀和, 渡邊 晃:アドレス空間透過性を実現する NAT-f の実装と評価，マルチメディア，分散，協調とモバイル（DICOMO2006）シンポジウム論文集(I)， Vol.2006, No.6, pp.453-456 (2006).
[12] Vixie, P., Thomson, S., Rekhter, Y. and Bound, J.: Dynamic Updates in the Domain Name System (DNS UPDATE), RFC2136, IETF (1997).
[13] Srisuresh, P. and Egevang, K.: Traditional IP Network Address Translator (Traditional NAT), RFC3022, IETF (2001).
[14] Eastlake, D.: DNS Request and Transaction Signatures (SIG), RFC2931, IETF (2000).
[15] Vixie, P., Gudmundsson, O., Eastlake 3rd, D. and Wellington, B.: Secret Key Transaction Authentication for DNS (TSIG), RFC2845, IETF (2000).
[16] Eastlake 3rd, D.: Secret Key Establishment for DNS (TKEY), RFC2930, IETF (2000).
[17] BIND : http://www.isc.org/index.pl?/sw/bind/
[18] Iperf : http://dast.nlanr.net/Projects/Iperf/
[19] Intel Corp.: Using the RDTSC Instruction for Performance Monitoring (1998). http://developer.intel.com/drg/pentiumII/appnotes/RDTSCPM1.htm.
[20] Mishra, A., Shin, M. and Srbaugh, W.: An Empirical Analysis of the IEEE802.11 MAC Layer Handoff Process, ACM SIGCOM Computer

Communication Review, Vol.33, No.2, pp.93–102 (2003).
[21] Kanemoto, A., Sejimo, M. and Watanabe, A.: Proposal of a packet-lossless handover in Mobile PPC, IEEE International Region 10 Conference (TENCON2006) (2006).
[22] 葛谷章一, 瀬下正樹, 渡邊 晃 : プロキシを利用した Mobile PPC の検討, 電子情報通信学会総合大会 (2007).
[23] Enomoto, K., Suzuki, H., Sakamoto, J. and Watanabe, A.: Researches on Mobile Communications over a Private Address Area and a Global Address Area, International Symposium on Information Theory and its Applications (ISITA) 2006 (2006).
[24] Postel, J. and Reynolds, J.: File Transfer Protocol (FTP), RFC959, IETF (1985).

研究業績目録

1. 坂本 順一, 鈴木 秀和, 竹内 元規, 渡邊 晃, “Mobile P2P を利用した移動ネットワークの提案”, 平成 16 年度電気関係学会東海支部連合大会講演論文集, Sep.2004.
2. 坂本 順一, 鈴木 秀和, 竹内 元規, 渡邊 晃, “Mobile PPC を利用したネットワーク単位の移動通信の提案”, 第 67 回情報処理学会全国大会講演論文集, Mar.2005.
3. 坂本 順一, 鈴木 秀和, 竹内 元規, 渡邊 晃, “Mobile PPC を利用したネットワーク単位の移動透過性の提案”, マルチメディア, 分散, 協調とモバイル(DICOMO2005)シンポジウム論文集, Vol.2005, No.6, pp.133-136, Jul.2005.
4. 坂本 順一, 鈴木 秀和, 竹内 元規, 渡邊 晃, “ネットワーク単位の移動透過性を実現する Mobile NPC とその実装”, 平成 17 年度電気関係学会東海支部連合大会講演論文集, Sep.2005.
5. 坂本 順一, 鈴木 秀和, 渡邊 晃, “ネットワーク単位の移動透過性を実現する Mobile NPC の実装と評価”, マルチメディア, 分散, 協調とモバイル(DICOMO2006)シンポジウム論文集(II), Vol.2006, No.6, pp.821-824, Jul.2006.
6. 榎本 万人, 坂本 順一, 鈴木 秀和, 渡邊 晃, “異なるアドレス空間を跨る移動通信の検討”, 平成 17 年度電気関係学会東海支部連合大会講演論文集, Sep.2005.

7. 榎本 万人，鈴木 秀和，坂本 順一，渡邊 晃，“プライベートアドレス空間とグローバルアドレス空間を跨る移動通信の検討”，第 68 回情報処理学会全国大会講演論文集，Mar.2006.
8. 榎本 万人，鈴木 秀和，坂本 順一，渡邊 晃，“プライベートアドレス空間とグローバルアドレス空間を跨る移動透過性の実現”，マルチメディア，分散，協調とモバイル（DICOMO2006）シンポジウム論文集(II)，Vol.2006，No.6，pp.813-816，Jul.2006.
9. K. Enomoto, H. Suzuki, J. Sakamoto and A. Watanabe, "Researches on Mobile Communications over a Private Address Area and a Global Address Area", International Symposium on Information Theory and its Applications (ISITA) 2006，Oct.2006.

付録 A

略語一覧

ACT：Access Control Table
CIT：Connection ID Table
CN：Correspondent Node
CU：CIT Update
DDNS：Dynamic DNS
FQDN：Fully Qualified Domain Name
HA：Home Agent
IMS：IP address Mapping Server
LIN6：Location Independent Networking for IPv6
MA：Mapping Agent
MAT：Mobile IP with Address Translation
Mobile NPC：Mobile Network to Peer Communication
Mobile PPC：Mobile Peer to Peer Communication
MN：Mobile Node
MPR：Mobile NPC Router
NAPT：Network Address Port Translation
NAT-f：NAT-free
NEMO：Network Mobility
NRT：Name Relation Table
PAT：Port Address translation Table
PHN：Private Host Name
PN：Private Node
PT：Private address Terminal
RDTSC：Read Time Stamp Counter
RR：Resource Records
VAT：Virtual Address Translation

付録 B

1. その他の DDNS 登録方法

3.4.2 節では CN または MPR の初期立ち上げ時と移動時の登録方法を述べた．以下 MN と MPR の場合について述べる．

1.1. MN の移動時の処理

MN の移動時の処理を図 11 に示す．以降 MN と MN の FQDN “bob.example.com” を登録している DDNS サーバはすでに認証用の共有秘密鍵 kMN を保持しているとする．MN が移動して IP アドレスがグローバルアドレス Y0 からグローバルアドレス Y1 に変わると application でアドレスの変化を検知する．アドレスの変化を検知した application は nsupdate コマンドを実行する．nsupdate は bob.example.com，MN の IP アドレス Y1，共有秘密鍵 kMN で署名したデータを含んだ DDNS update パケットを MN の FQDN を登録している DDNS サーバに送信する．受信した DDNS サーバは kMN を用いて署名データの認証成功後，RR の IP アドレスを Y0 から Y1 に更新する．

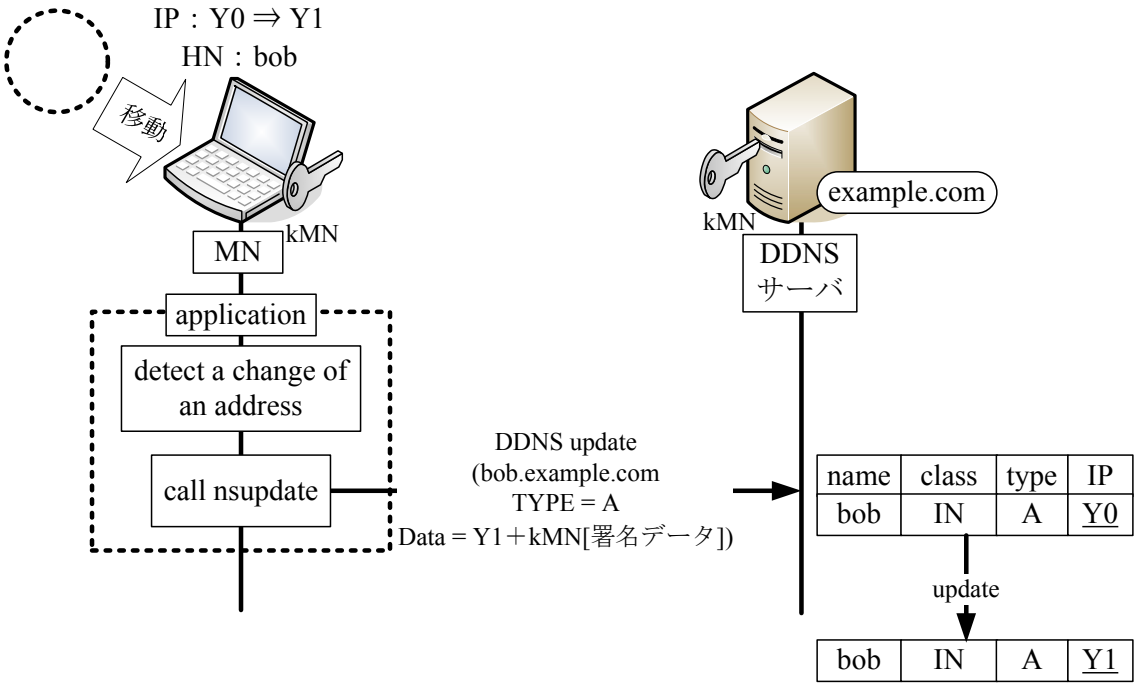


図 11　MN の移動時の処理

1.2. MPR 配下の MN が移動時の処理

MPR 配下に MN が移動した時の処理を図 12 に示す. MPR 配下に MN が移動すると IP アドレスがグローバルアドレス Y1 からプライベートアドレス P0 に変わると application でアドレスの変化を検知する．アドレスの変化を検知した application は nsupdate コマンドを実行する. nsupdate は bob.example.com, IP アドレス P0，共有秘密鍵 kMN で署名したデータを含んだ DDNS update パケットを MN の FQDN を登録している DDNS サーバに送信する．受信した MPR は DDNS update パケットの IP アドレスが MPR のグローバル側のアドレス X0 と異なるため，MN に MPR のグローバル側のアドレス X0 を含んだパケットを送信する．受信した MN は nsupdate コマンドを実行し，nsupdate は bob.example.com，MPR の IP アドレス X0，共有秘密鍵 kMN で署名したデータを含んだ DDNS update パケットを MN の FQDN を登録している DDNS サーバに送信する．受信した DDNS サーバは kMN を用いて署名データの認証成功後，RR の IP アドレスを Y1 から X0 に更新する.

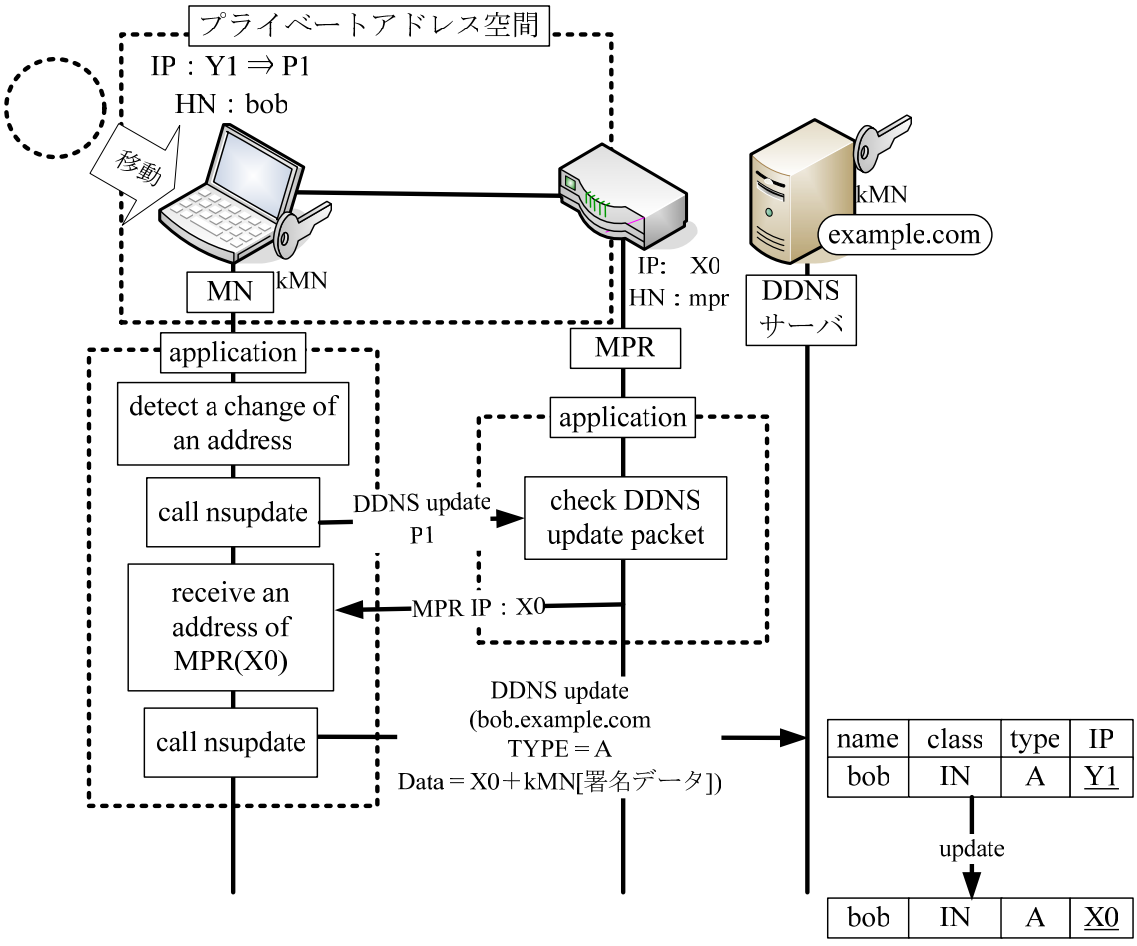


図 12　MPR 配下に MN が移動時の処理

1.3. MPR 配下に MN 存在時に MPR の移動時の処理

MPR 配下に MN が存在する時に MPR の移動する時の処理を図 13 に示す. MPR と MPR の FQDN “mpr.example1.com” を登録している DDNS サーバはすでに認証用の共有秘密鍵 kMPR を保持しているとする. MPR が移動して IP アドレスがグローバルアドレス X0 からグローバルアドレス X1 に変わると application でアドレスの変化を検知する. アドレスの変化を検知した application は nsupdate コマンドを実行する. nsupdate は mpr.example1.com, MPR の IP アドレス X1, 共有秘密鍵 kMPR で署名したデータを含んだ DDNS update パケットを MPR の FQDN を登録している DDNS サーバに送信する. 受信した DDNS サーバは kMPR を用いて署名データの認証成功後, RR の IP アドレスを X0 から X1 に更新する. その後 MPR はプライベートアドレス空間側に MPR の IP アドレス X1 を含んだパケットをブロードキャストする.

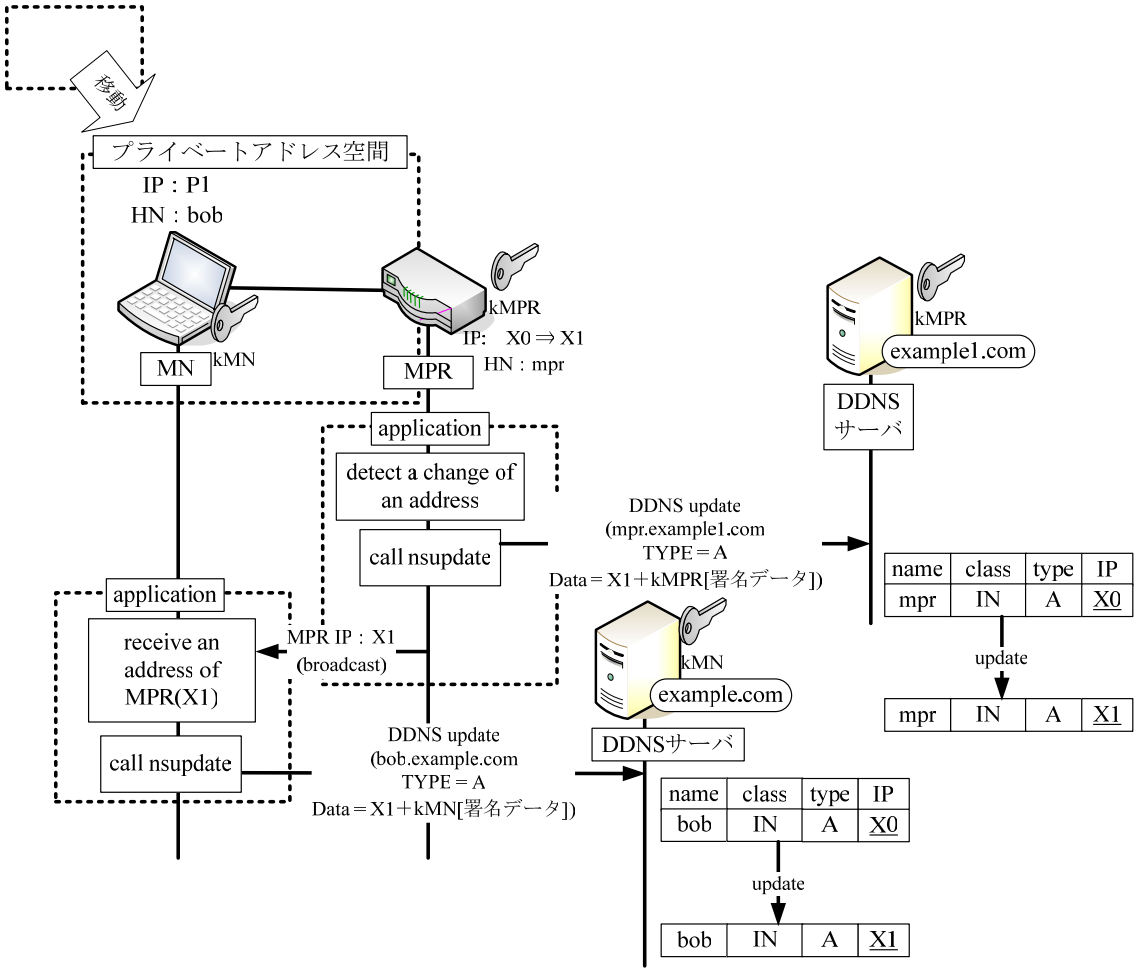


図 13 MPR 配下に MN 存在時に MPR の移動時の処理

受信した MPR 配下のすべての MN は nsupdate コマンドを実行し, nsupdate は bob.example.com，MPR の IP アドレス X1，共有秘密鍵 kMN で署名したデータを含んだ DDNS update パケットを MN の FQDN を登録している DDNS サーバに送信する．受信した DDNS サーバは kMN を用いて署名データの認証成功後，RR の IP アドレスを X0 から X1 に更新する．

1.4. MPR 配下の MN が MPR 外に移動時の処理

MPR 配下の MN が MPR 外に移動すると MN の IP アドレスはプライベートアドレスからグローバルアドレスに変わる. 取得した IP アドレスはグローバルアドレスであるから付録 B の 1.1 節と同様の処理を行う．

1.5. MPR 配下の MN が別の MPR 配下に移動時の処理

MPR 配下の MN が別の MPR 配下に移動すると MN はプライベートアドレスを取得するため付録 B の 1.2 節と同様の処理を行う．

1.6. MN の初期立ち上げ時の処理

MN の初期立ち上げ時は application を実行する．application は現在割り当てられているアドレスを取得し，登録する DDNS サーバに移動の登録と同様の DDNS update を行う．

2. 実装の詳細

実装における PAT，Mobile PPC，NAT-f モジュールの追加した点と変更の詳細を以下に述べる．

PAT には次の機能を実装した．PAT で生成されるテーブルの削除には無通信状態がある時間続いた場合に削除するタイマー機能とパケット監視することで TCP のソケットクローズ時と TCP のコントロールフラグに RST がある場合にテーブルを削除する機能を実装した. また通信開始パケットが PAT による変換後のアドレスとポート番号が衝突した場合に未使用のポート番号に変換することで衝突を回避する機能を実装した．

Mobile PPC には次の機能の追加と変更を行った．機能の追加には CU がロスした場合に CU の再送を行う機能と連続移動の対応，変更には CU の送信を Gratuitous ARP のタイムアウト後から Gratuitous ARP のタイムアウト後に CU の送信先のゲートウエイに ARP request を送信してゲートウエイからの ARP reply を受信した後に CU を送信することと CIT から PIT を使用するように変更した．

NAT-f には次の変更を行った. ポート番号変換を CN から MRP に行うこと

と VAT から PIT を使用するように変更した．

Mobile NPC はパケット受信時に呼び出される関数 ip_input にはグローバルアドレス側，およびプライベートアドレス側から受信するパケットの両方が Mobile PPC のモジュールに渡されてしまう．パケット送信時に呼び出される関数 ip_output も同様である．そこで入力・出力パケットのアドレスがグローバルアドレスの場合のみ PAT・Mobile PPC・NAT-f を呼び出すように改造した．

3. その他の評価

3.1. NAT-f ネゴシエーションのオーバヘッド

測定を図 8 に示す評価環境と表 1 に示す機器構成で行った．CN が MPR 配下の端末 PT に通信を開始する前に行う NAT-f ネゴシエーションの処理時間を図 14 に示す．RDTSC で測定し NAT-f ネゴシエーションを 10 回行ったときの平均である．Application が通信開始パケット（SYN）を送信すると

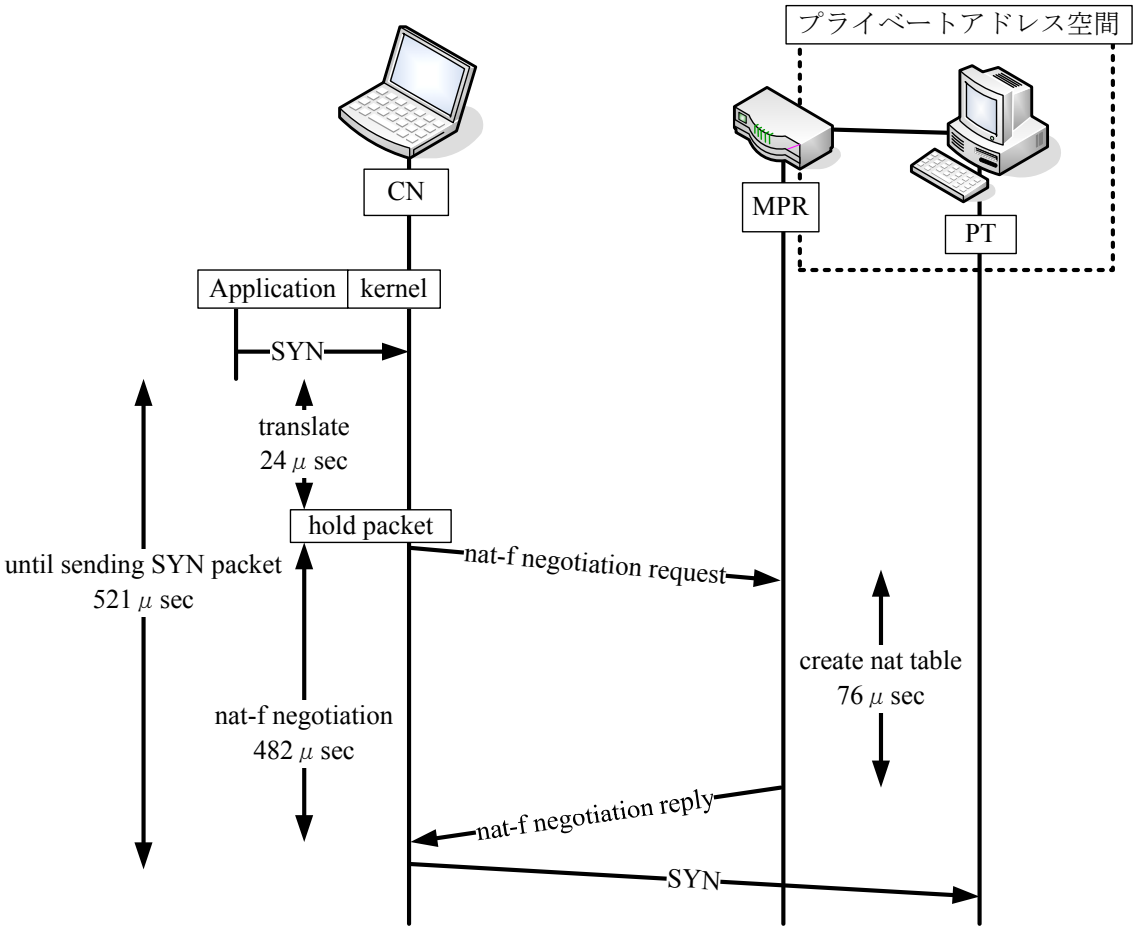


図 14　NAT-f ネゴシエーション処理時間

kernel に渡される．kernel で仮想アドレスから実アドレスに変換する処理に 24μ秒であった．CN は変換したパケットを退避させ，MPR に NAT-f negotiation request を送信する．MPR が受信してから擬似パケットで NAT テーブルを作成し，NAT-f negotiation reply を送信するまでに 76μ秒であった．NAT-f negotiation の時間は 482μ秒であり，通信開始パケット（SYN）が実際に送信されるまでの時間は 521μ秒であった．この結果から NAT-f ネゴシエーションのオーバヘッドはほとんどないことが分かる．

3.1. シーケンス番号

測定を図 8 に示す評価環境と表 1 に示す機器構成で行った．Iperf を用いて PT から CN に 15 秒間通信を行った．およそ 4 秒後に MPR の移動を手動で LAN ケーブルの切り替えで行い，dhclient コマンドを実行してアドレスを取得した．そのときのシーケンス番号の変化のグラフを図 15 に示す．切り替えは 3.7sec の時点で行われ，通信切断時間は 8.6sec であった．通信切断時間の内訳は切り替え時間に 3.1sec，dhclient 動作時間に 3.8sec，CU を送信す

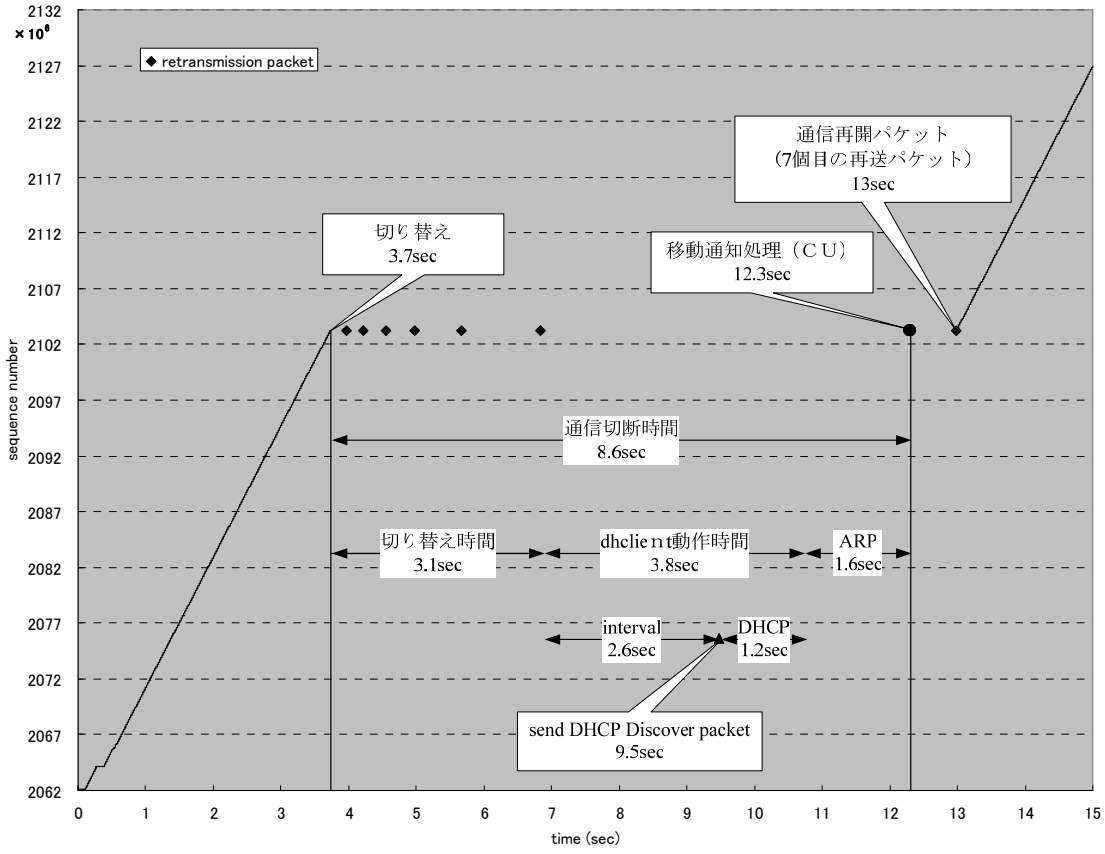


図 15　シーケンス番号

るゲートウエイへの ARP 要求/応答に 1.6sec であった. dhclient 動作時間の内訳は interval に 2.6sec，DHCP に 1.2sec で 9.5sec の時点で dhclient は DHCP Discover を送信した．12.3sec の時点で移動通知処理（CU）が行われた．7 個目の再送パケットにより 13sec の時点で通信が再開されたことが分かる．5.3 節で述べたように切り替え時間と interval をほぼ 0sec にすれば通信切断時間は DHCP と ARP の 2.8sec になり，切り替えから 2.8sec 後の 6.5sec の時点で通信継続が可能になる. このことからおよそ 7sec の時点で送信する 6 個目の再送パケットにより通信が再開する.

3.2. FTP の性能評価

図 16 に示す評価環境で動作検証を行った. CN の OS は FreeBSD で Mobile PPC を実装し，CPU は Pentium 2.4GHz，メモリが 256M，NIC は 100BASE-T である．MPR の OS は FreeBSD で Mobile NPC と NAPT を実装し，CPU は Pentium 1.7GHz，メモリが 512M，グローバル側は IEEE802.11g，プライベート側の NIC は 100BASE-T である. PT の OS は Windows XP Professional で CPU は Pentium 2.8GHz，メモリが 1024M，NIC は 100BASE-T である.

Mobile NPC を実装した場合に MPR の処理が中継性能に与える影響を調査するため PT が CN に対して FTP 接続のダウンロードにかかる時間を測定し

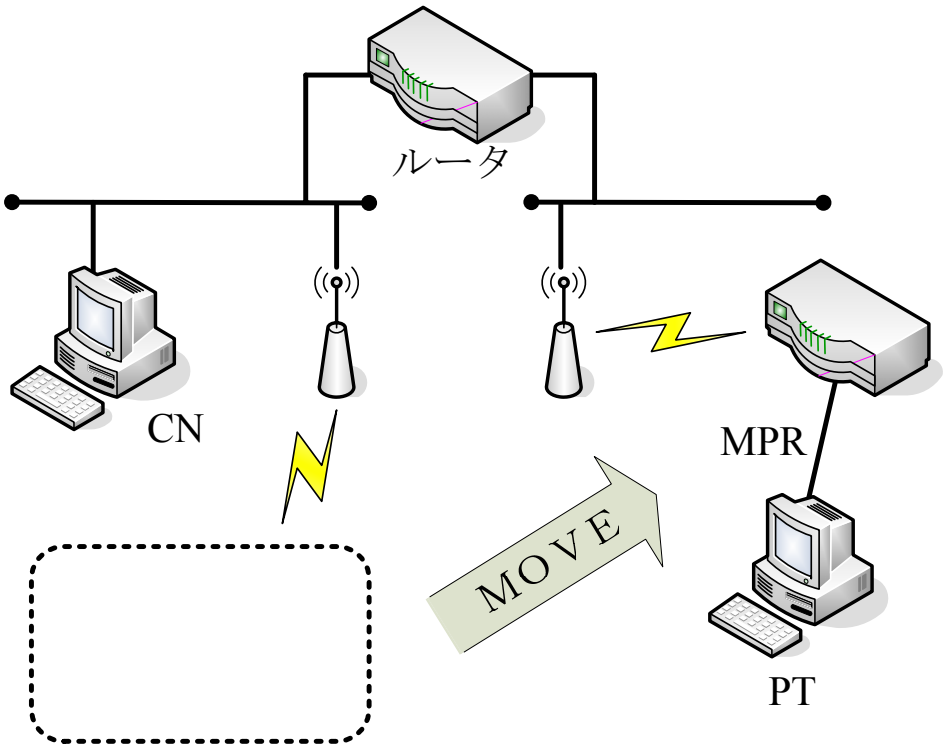


図 16　評価環境

た．MPR に（1）Mobile NPC を未実装の状態，（2）Mobile NPC を実装しアドレス変換なしの状態（移動前），（3）Mobile NPC を実装しアドレス変換ありの状態（移動後）でそれぞれ PT が CN から 100MByte のファイルを FTP でダウンロードした時間を表 5 に示す．ケース（1）とケース（2）はほぼ同じ性能であり，ケース（3）は約 2%のオーバヘッドが発生した．この結果により MPR が Mobile NPC 機能を保持したことによるオーバヘッドの増加はほとんどないことが分かる．

表 5　FTP のダウンロード時間

状態	ダウンロード時間	割合
Mobile NPC 未実装	129[秒]	1.00
アドレス変換なし	129[秒]	1.00
アドレス変換あり	131[秒]	1.02

付録 C

1. はじめに

FreeBSD における NAT の使用方法と扱い方を示す．FreeBSD には，IP 層で動作する ipnat とアプリケーション層で動作する natd が存在するが，このドキュメントではアプリケーション層で動作する natd の使用方法と扱い方を説明する．

2. FreeBSD の設定

FreeBSD に NAT の機能を追加するためには，カーネルをコンパイルする必要がある．

以下に手順を示す

2.1. カーネルの設定準備

```
# cd /usr/src/sys/i386/conf
# cp GENERIC MYKERNEL
```

・ MYKERNEL は任意名

2.2. ipfw と divert の導入

```
# vi MYKERNEL
```

以下の２行を追加（vi の使用法は FreeBSD 基本操作ガイドのドキュメントファイルなど参照）

```
# options IPFIREWALL
# options IPDIVERT
```

・不要なデバイスをコメントアウトすると起動時間が速くなります．（デバイスの詳細はハンドブックを参照）

2.3. カーネルコンパイル

```
# /usr/sbin/config MYKERNEL
# cd ../../compile/MYKERNEL
# make depend
# make
# make install
```

・ MYKERNEL を変更せずに、２回目以降コンパイルする場合は

```
# cd /usr/src/sys/i386/compile/MYKERNEL
```

make
make install
を実行
カーネルのソースフォルダ
全般：/usr/src/
主な IP 関連：/usr/src/sys/netinet/
natd のソースフォルダ
/usr/src/sbin/natd/

3. コンパイル後の設定

3.1. natd の設定

natd -interface lnc0
lnc0 はグローバルアドレスが割り当てられるインターフェース名

起動時に自動で設定したい場合
vi /etc/rc.conf
以下 2 行を追加
natd_enable=”YES”
natd_flags=”-f /etc/natd.conf”

vi /etc/natd.conf
以下 2 行を追加
dynamic
interface lnc0

3.2. Gateway の設定

sysctl net.inet.ip.forwarding=1

3.3. ipfw の設定

/sbin/ipfw -f flush
/sbin/ipfw add divert natd all from any to any via lnc0
/sbin/ipfw add pass all from any to any
lnc0 はグローバルアドレスが割り当てられるインターフェース名

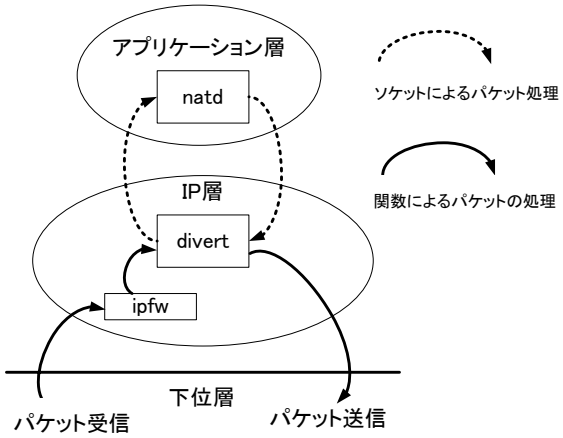
起動時に自動で設定したい場合

vi /etc/rc.conf
以下 2 行を追加
firewall_enable=”YES”
firewall_script=”/etc/rc.firewall”

4. NAT パケットの概要図

以下に用語を解説する.

ipfw は firewall の動作を行うモジュールである. divert は natd のパケットの取り出しをサポートするソケットである. ipfw と divert は標準で FreeBSD に導入できる.



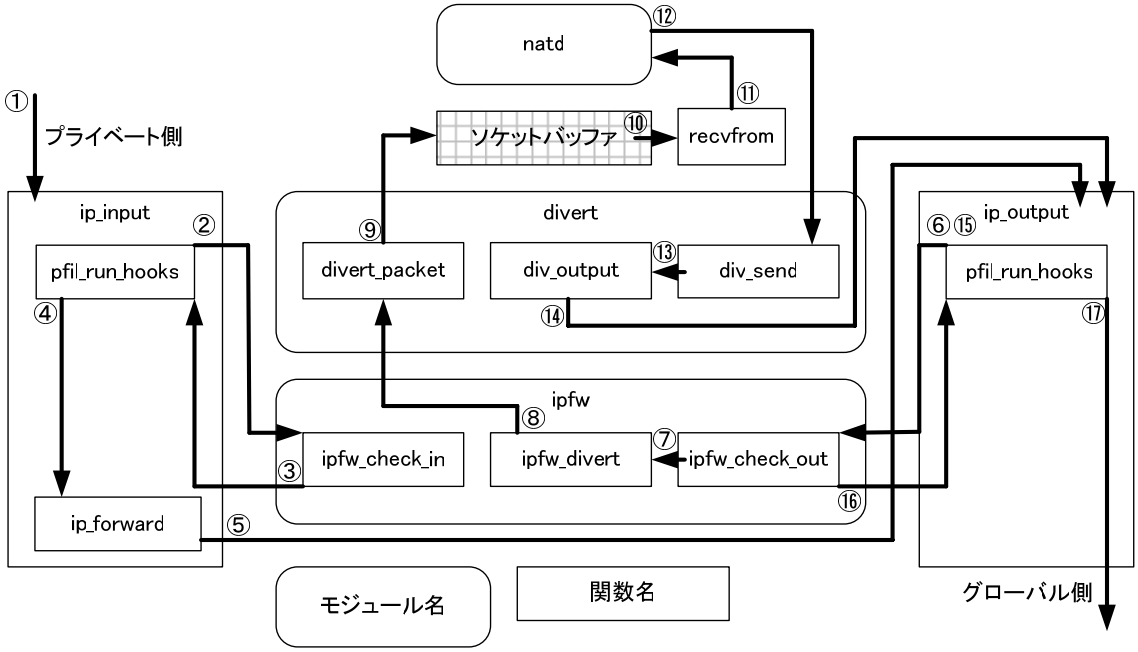
右図に NAT パケットの概要図を示す. 受信したパケットが下位層から IP 層の ipfw に渡される. ipfw はそのパケットを divert に渡す. 次に natd は divert からソケットを介してパケットを取り出し, natd はテーブル生成やパケットのアドレス変換などを行った後, ソケットを介して divert に渡す. Divert はパケットを下位層に渡し, パケットが送信される.

5. FreeBSD の IP 層における NAT Packet の処理

プライベート側からグローバル側へと逆方向のへのパケットの処理の説明を行う.

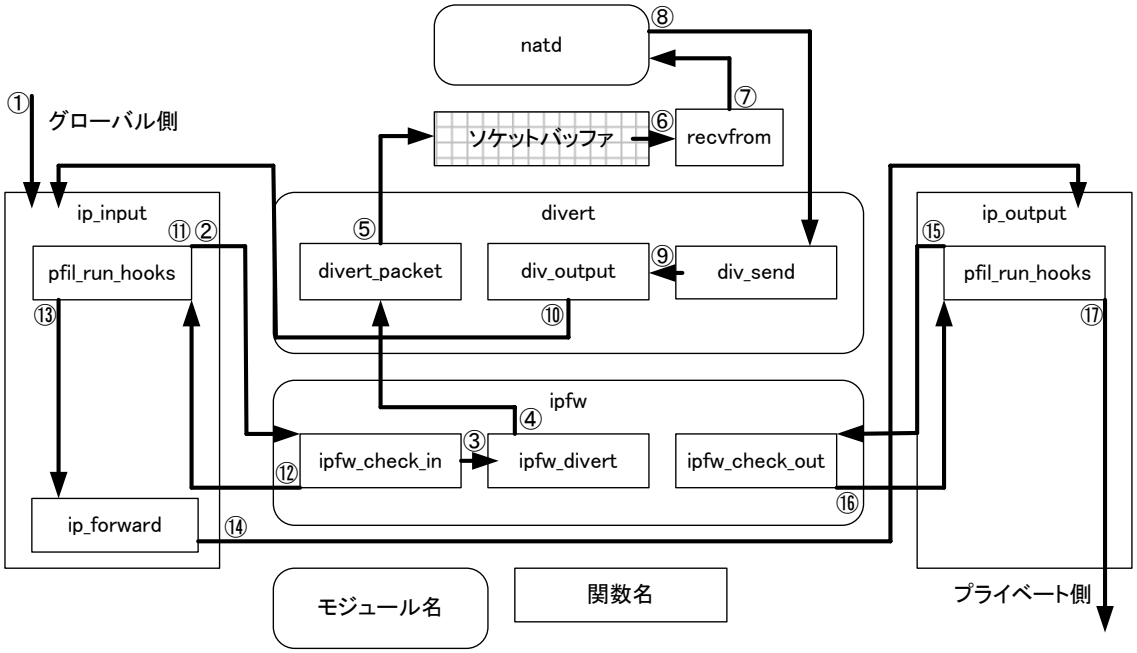
5.1. プライベート側からグローバル側へ

- プライベート側からのパケットが ip_input に渡される. ①
- pfil_run_hooks が call されると, ip_input 用の ipfw_check_in を call する. ②
- ipfw_check_in では, firewall の処理を行う. firewall を通過したパケットは ip_input に戻る. ③
- 他宛のパケットの場合, ip_input は ip_forward を call する. ip_forward は ip_output を call する. ④⑤
- pfil_run_hooks が call されると, ip_output 用の ipfw_check_out を call する. ⑥
- ipfw_check_out では, firewall の処理を行わずに ipfw_divert を call する. ⑦
- ipfw_divert は divert_packet を call してパケットをソケットバッファに退避する. ⑧⑨
- natd は, recvfrom を call して, ソケットバッファに退避されていたパケットを取り出す. ⑩⑪
- natd は, アドレス変換テーブルを参照して取り出したパケットの送信元アドレスをプライベートからグローバルに変換する. 通信開始のパケットの場合は, アドレス変換用テーブルを生成する. ⑫
- natd で変換されたパケットは div_send に渡されて, div_output を call する. div_output は ip_output を call する. ⑬⑭
- ⑥と同様の処理を行う. ⑮
- ⑦の処理を行わずに firewall の処理を行う. firewall を通過したパケットは ip_output に戻る. ⑯
- グローバル側へパケットが送信される. ⑰


natd
⑫
⑪
①
プライベート側
ソケットバッファ
⑩
recvfrom
ip_input
②
divert
⑨
⑥⑮
ip_output
pfil_run_hooks
divert_packet
div_output
⑬
div_send
pfil_run_hooks
④
⑭
⑰
ipfw
⑧
⑦
ipfw_check_in
ipfw_divert
ipfw_check_out
③
⑯
ip_forward
⑤
モジュール名
関数名
グローバル側

5.2. グローバル側からプライベート側へ

- グローバル側からのパケットが ip_input に渡される．①
- pfil_run_hooks が call されると, ip_input 用の ipfw_check_in を call する．②
- ipfw_check_in では，firewall の処理を行わずに ipfw_divert を call する．③
- ipfw_divert は divert_packet を call してパケットをソケットバッファに退避する．④⑤
- natd は，recvfrom を call して，ソケットバッファに退避されていたパケットを取り出す．⑥⑦
- natd は，アドレス変換テーブルを参照して取り出したパケットの宛先アドレスをグローバルからプライベートに変換する．⑧
- natd で変換されたパケットは div_send に渡されて，div_output を call する．div_output は ip_output を call する．⑨⑩
- ②と同様の処理を行う⑪
- ③の処理を行わずに firewall の処理を行う. firewall を通過したパケットは ip_input に戻る⑫
- 他宛のパケットの場合，ip_input は ip_forward を call する．ip_forward は ip_output を call する．⑬⑭
- pfil_run_hooks が call されると，ip_output 用の ipfw_check_out を call する．⑮
- ipfw_check_out では，firewall の処理を行う．firewall を通過したパケットは ip_output に戻る．⑯
- プライベート側へパケットが送信される．⑰



6. NAT における ICMP の扱い

ICMP は，IP 層であるためポート番号がないために，通常 NAT では動作しない．しかし，natd では一部の ICMP が動作する．

ICMP ヘッダ

以下の表に動作する ICMP を示す．

ICMP タイプ	メッセージ
0	エコー応答(Echo Reply)
8	エコー要求(Echo)
13	タイムスタンプ要求(Timestamp Request)
14	タイムスタンプ応答(Timestamp Reply)
3	到達不能(Destination Unreachable)
4	発信抑制(Source Quench)
11	時間超過(Time Exceed)
12	パラメータ異常(Parameter Problem)

識別子(2 バイト)　順序番号(2 バイト)

未使用(4 バイト)　元データグラムの先頭 64bit (8 バイト)

・ ICMP タイプ（0,8,13,14）の ICMP データの共通部分は識別子と順序番号である．

- ICMP タイプ（3,4,11,12）の ICMP データはすべて同じである.
- 元データグラムの先頭 64bit（8 バイト）: エラー発生したパケットの先頭 8 バイトのデータ部分になる
  - ICMP の場合
    - ＋ICMP データ 4 バイト
  - TCP の場合
    - 送信元ポート（2 バイト）＋宛先ポート＋シーケンス番号
  - UDP の場合
    - 送信元ポート（2 バイト）＋宛先ポート＋セグメント長＋チェックサム