モバイルコンピューティングと
ワイヤレス通信
高度交通システム
（２００１．９．７）
18－23
6－23

移動体通信プロトコルMobile IPv6の実用性の検証

小柴　晋[†]　湧川隆次[††]
植原啓介[††]　村井純[†††]

移動体計算機がインターネット上のネットワーク間を移動しても、一つのIPアドレスで通信を継続することを支援する移動体通信プロトコルとしてMobile IPv6やLocation Independent Network for IPv6が提案されている。しかし、現状では移動体通信プロトコルの実装を十分に評価した事例がなく、利用者がどの実装を利用するかを選択するための指標が存在しない。本稿では現在公開されているMobile IPv6及びLocation Independent Network for IPv6実装の性能を測定し、個々の実装の優れている点や実装が不十分であると考えられる点を整理した。結果、どの実装においても移動体通信プロトコルとして十分使用することが可能であることが確認できた。

Verification of Mobile IPv6 Practicability as a Mobility Support Protocol

Susumu Koshiba,[†] Ryuji Wakikawa,[††] Keisuke Uehara[††]
and Jun Murai[†††]

Mobile IPv6 and Location Independent Network for IPv6 are proposed to provide reachability to a host by sole IP address while the host is moving between sub-networks on the Interenet. However, there are no exemplum of evaluating the mobility support protocols and users do not have a reference to select which package to use. In this paper, disclosed Mobile IPv6 and Location Independent Network for IPv6 are tested and organized the characteristics of both implementations. For a result, all the implementations are proven to be usable.

1. はじめに

1.1 背　景

固定計算機に加えて移動する機器がインターネットに接続されつつある。今後は、Internet Protocol version6(IPv6)の導入や、広域無線通信網の整備により、計算機の継続した通信が必要となる。しかし、既存のインターネットアーキテクチャは、移動しない計算機を前提に構築されている。そのため、携帯電話や自動車等の移動する計算機をインターネットに接続する際には、様々な制約や問題が起こる。

移動の際に起こるIPアドレスの変化などの問題を解決すべく考案されたのが、Mobile IPv6(MIP6)[1]やLocation Independent[2] Network for IPv6(LIN6)などの移動体通信プロトコルである。これらのプロトコルにより、利用者やアプリケーションは、移動を意識すること無く通信を継続することができる。

しかし、ユーザーは円滑な移動体通信を行うために利用状況に応じて使用するプロトコルを選択する必要がある。現在、移動体通信プロトコルには同一仕様で複数の実装が存在しているものがあり、それぞれの機能・性能においてどのような違いがあるのかは測定されていない。このこのため、利用者が選択するための指標が存在しない。

1.2 目　的

本稿では、移動体通信プロトコルであるMIP6の性能評価を行い、その実用性を評価することを目的とする。実用性の評価において本稿では実際の利用環境において想定される通信環境について性能評価を行い、そこで得られた知見をもとに、それぞれの実装の現状を把握し、状況・用途に応じた実用性を検証する。さらに、今後の移動体通信プロトコルの実装にあたって改善されなければならない点を整理する。

† 慶應義塾大学 総合政策学部
Keio University Faculty of Policy Management
†† 慶應義塾大学 政策・メディア研究科
Keio University Graduate School of Media and Governance
††† 慶應義塾大学 環境情報学部
Keio University Faculty of Environmental Information

2. 移動体通信プロトコルについて

本章では、移動体通信環境に対する具体的な要求について述べる。さらに、本稿にて評価した移動体通信プロトコルである MIP6 と LIN6 の概要及び実装状況について触れる。

2.1 移動体通信に対する要求

既存の移動体通信環境では、高速移動に伴う IP アドレスの頻繁な変化に対応することができず、移動体計算機に必要とされる通信環境を提供することができない。そこで、以下のような要求を満たす移動体通信環境を構築することが今後の移動体通信に必要となる。

(1) 移動透過性：
移動による IP アドレスの変化をアプリケーション層から隠蔽することによって通信の継続を可能とすること。

(2) 着信可能性：
移動体計算機を常に一意のアドレスにてパケットを受信可能な状態にすること。

(3) 常時接続性：
1 つのデバイスではカバーできない場所での通信を、他の通信デバイスを用いることにより接続を保つこと。

上記のような要求を満たすために現在考案されているのが本稿で取り扱う移動体通信プロトコルである。

2.2 Mobile IPv6

MIP6 は先に述べた移動体通信環境に対する要求のうち、移動透過性及び着信可能性を提供することを目指したプロトコルである。

まず始めに、本稿にて使用される MIP6 の用語説明を**表 1** に記す。

MIP6 では、移動に影響を受けないアドレス (Home Address) と実際利用可能なアドレス (Care of Address, CoA) の対応をネットワーク層で管理する。これにより、トランスポート層やアプリケーションは、MN が実際にどのネットワークにいるのかを意識することなく、Home Address を用いて通信できる。

MN は、新しいネットワークに移動すると新たな CoA を取得する。その CoA を Binding Update(BU) と呼ばれるメッセージを用いて通信相手に知らせる。通信相手は、BU を受け取ることで移動後の MN との接続性を確保できる

2.3 Mobile IPv6 実装状況

本稿では MIP6 の測定に慶應義塾大学湘南藤沢キャンパス (以下、SFC) の InternetCAR プロジェクトで実装が進められている MIP6 の実装 (以下、SFC-MIP6) と日本電気株式会社 (以下、NEC) のネットワーク研究所の MIP6 の実装 (以下、NEC-MIP6) を使用した。どちらも IETF より Internet Draft として公開されている仕様にもとづいて実装されている。現在最新の仕様は draft-ietf-mobileip-ipv6-14 であるが、本稿で測定された実装はどちらも draft-ietf-mobileip-ipv6-13 にもとづいて実装されている。

表 1 Mobile IPv6 用語説明
Table 1 Descriptions of Words for Mobile IPv6

単語 (略称)	説明
Correspondent Node (CN)	MN と通信をする相手ノード。CN は移動体ノードでも固定ノードでも良い。
Mobile Node (MN)	ネットワーク間を移動するノード。
Care of Address (CoA)	MN が移動先リンク取得する IP アドレス。
Home Address	MN に割り当てられた IP アドレス。移動の影響を受けないため、MN を一意に識別するために用いられるアドレス。MN と通信相手ホストはこのアドレスを用いて通信を行う。
Home Link	MN の持つ Haddr と同じ prefix を持つネットワーク。
Foreign Link	MN の Home Link 以外のリンク。
Binding	MN の Haddr と CoA の関連を示すもの。Haddr と CoA の関連と共に有効期限などの情報のことを指す。MIP6 では全てのノードで必要に応じて MN の binding をキャッシュできなくてはならない。
Home Agent (HA)	MN の Home Link 上のルーター。MN 宛のパケットを代わりに受け取り、MN の CoA へ向けて転送する。

2.4 LIN6

LIN6 は、MIP6 とは別の方法で移動透過性を保証するプロトコルである。**表 2** に、LIN6 の用語説明を記す。

表 2 LIN6 用語説明
Table 2 Descriptions of Words for LIN6

単語 (略称)	説明
ノード識別子	IPv6 アドレスの下位 64bit を指す。
位置指示子	アドレス構造の 128bit 全体であるインターフェースアドレス。
汎用識別子	上位 64bit が固定値の LIN6 専用の prefix、下位 64bit がノード識別子のアドレス。
Mapping Agent(MA)	ノード識別子と位置指示子の関係を保持しているエージェント。

LIN6 では、従来の IPv6 アドレスの他に、LIN6 アドレスと呼ばれる 64bit のノード固有なノード識別子を利用する。このままでは従来の IPv6 アプリケーションが利用できないため、LIN6 Prefix を上位 64bit に

連結し通信することで、移動透過性が保証される。

パケットを移動体に配送するためには、通信相手は移動体のネットワーク上での位置を知る必要がある。LIN6では、この接続点をLIN6アドレスで表す。LIN6アドレスは、現在のnetwork prefixにLIN6 IDを連結したものである。したがって、LIN6汎用IDを持つノードが現在どのようなLIN6アドレスを持っているかという情報(Mapping)を管理しなければならない。

Mapping Agent(MA)は、このMapping情報を管理するAgentである。LIN6を使用するノードは、ネットワーク上を移動するたびに自分のMapping Agentに対してMappingを通知する。

2.5 LIN6実装状況

LIN6は現在、慶應義塾大学の寺岡らによって提案[3]されている移動体通信プロトコルであり、他の実装は存在しない。また、仕様に関しても同研究室を中心に議論がなされ、決定されている。標準化に関しては、IETFなどで議論することによって標準化へ向けて活動を行っている。

3. 移動体通信プロトコル性能評価

本章ではMIP6実装ごとの性能比較を行い、更にMIP6とLIN6の移動体通信プロトコルとしての性能を比較する。評価内容は以下の通りである。

- MIP6の実装の違いによる処理コストの差を測定し、実装に性能の違いを整理し、評価する。
- LIN6との性能比較を行い、移動体通信プロトコルとしてのMIP6の性能を検証する。
- MIP6における処理が異なるネットワーク環境を構築し、それぞれの環境においてMIP6の動作確認及び通信処理速度を比較する。

本稿の測定に使用した測定環境を図1に示す。

3.1 ヘッダ処理コスト

本稿では、それぞれの移動体通信プロトコルにおけるヘッダ処理コストを測定するためにpentium performance counterをそれぞれのカーネル内の関数に入れて使用した。

MIP6では、通信を行っているMNからの全パケットにおいてIPv6 Home Address宛先オプションを挿入し、送信処理を行っている。通信相手は受信したパケットにIPv6 Home Address宛先オプションが含まれていたら、それを処理してから上のレイヤへデータを渡す。この一連の処理を行って始めてMobile IPv6は移動透過性や着信可能性を提供できる。また、CNからのパケットにはrouting headerを付加することによってMNまでパケットを送信する。また、LIN6

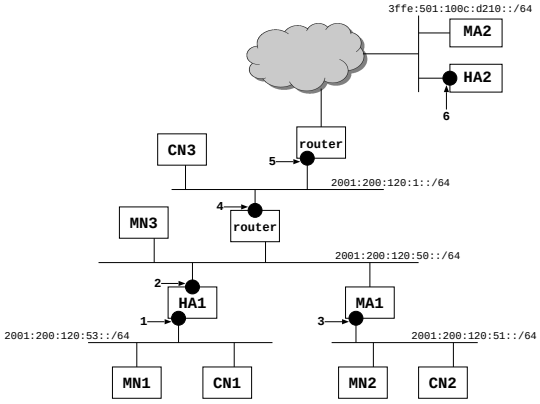

図1 測定トポロジー
Fig.1 Mesurement Topology

ではMIP6と異なりLIN6固有のIPv6 prefixを使用することによってLIN6パケットか否かを判断する。宛先アドレスの上位64bitがLIN6 prefixであった場合のみLIN6汎用識別子からLIN6 IDを抽出する処理を行い、mapping機構を用いてLIN6アドレスの生成を行う。LIN6アドレス宛のパケットに対しては、LIN6汎用識別子をLIN6 IDから取得し、上位層へ処理を渡す。これらの関数を使用することによって生じる計算処理コストは、MIP6及びLIN6での通信において常に発生し、通信処理コストを算出する上で欠かせない。

測定環境としては、図1のCN1とMN3を使用した。この環境ではICMP Echo Requestを送信した結果、RTTは全て0.5ms以下であった。また、MIP6、LIN6共にoutput関数はMN3にて、input関数の処理コストはCN1にて測定を行った。MN及びCNで使用した計算機のスペック☆は同一である。測定は、ICMP Echo Requestを1秒毎に50回送信し、平均を取った。表3に測定結果を示す。

表3 Pentium Counterを用いた処理コスト
Table 3 Process Cost taken with Pentium Counter

Protocol	input	output
KAME IPv6 Stack	81.9μs	7.29μs
SFC-MIP6	102μs	10.6μs
NEC-MIP6	105μs	11.7μs
LIN6	85.5μs	7.88μs

表3の結果から分かるようにMIP6、LIN6共に通常のIPv6での通信と比較してもそれほど大きな処理コストはかかっていない。この結果から、どちらの移

☆ CPU: Celeron 500Mhz, Memory: 256Mbyte

動体通信プロトコルのおけるヘッダ処理コストも、普通の IPv6 での通信とほとんど変わらないということが分かった。

また、SFC-MIP6 と NEC-MIP6 の処理コストを比較した結果、NEC の実装の方が処理コストがかかっていることが分かった。これは、NEC-MIP6 が KAME の IPv6 スタックに極力手を加えないようにしているのに対し、SFC-MIP6 は KAME の IPv6 スタックに直接手を加えていることに起因する。さらに、LIN6 の処理コストが MIP6 と比較してより小さかった理由は、LIN6 はカーネル内での処理を減らすように設計されているため、ここで測定した input 関数や output 関数での処理は MIP6 と比べて少ない。

ヘッダの処理を考慮した場合、通信速度にどれほど影響するかを測定した。測定方法として ICMP Echo Message を CN から MN に対して 100 回送信し、それぞれのプロトコル実装に関して最小 RTT、最大 RTT、そして RTT の平均を求めた。ICMP のデータ長は 64byte、送信間隔は 1 秒間隔である。**表 4** に測定結果を示す。

表 4 ICMP を用いた Round Trip Time
Table 4 Round Trip Time Using ICMP Echo Message

Protocol	最小 RTT	平均	最大 RTT
KAME IPv6 Stack	0.668 ms	0.745 ms	0.948 ms
SFC-MIP6	0.724 ms	0.784 ms	0.842 ms
NEC-MIP6	0.751 ms	0.824 ms	0.910 ms
LIN6	1.163 ms	2.492 ms	4.559 ms

表 4 の結果から、それぞれの移動体通信プロトコルを用いた通信は、オーバーヘッドが生じていることが分かる。実際の通信においては、SFC-MIP6 が最もプロトコル使用による RTT の差が小さかった。これは先述した実装方針の違いが影響していると思われる。また、LIN6 に関しては、MIP6 と比較して RTT にぶれがあった。LIN6 は MIP6 と違い、カーネル内での処理とは異なり、アプリケーション層での処理に依存するところが多い。よって、ICMP Echo Message を利用して RTT を測定した場合、アプリケーション層での処理が必要な分 MIP6 と比べて RTT の値が大きくなっている。

3.2 Home Address 及び Mapping 登録コスト

Binding Update 及び Mapping Update を送信してから、Ack が返って来るまでの時間を測定した。MIP6 には、全てのノードが piggyback と呼ばれる、binding 情報と共にデータを送信するための機構を持つ。そのため、今回測定を行った実装は、HA が piggyback するために最大 1 秒待つ。待ち時間内に MN に対するパケットが送信されて来た場合、そのデータに Binidng Ack を付加してから MN へ送信する。このように、piggyback することによって効率的な通信が行えるという設計であったが、最近では piggyback の待ち時間を管理するためのシグナルにかかるコストなどを考えると決して効率的ではないことが分かり、piggyback 機構を廃止する動きも強くなってきている。LIN6 は、piggyback 機構がない。よって測定結果は MN が Mapping Update を送信してから MA より Ack が返ってくるまでの時間である。

測定環境として、図 1 における MN3 と、HA1 を利用した。測定方法は、それぞれのプロトコルで Home Address 及び Mapping 登録処理を 10 回行わせた結果の平均値を取った。また、IPv6 の RTT は同じ環境で ICMP Echo Message を 10 回送信した数値の平均を取ったものである。

表 5 にそれぞれの実装が Home Address 及び Mapping を登録するためにかかった時間を示す。

表 5 Home Address/Mapping 登録コスト
Table 5 Home Address/Mapping Registration Time

Protocol	Time
KAME IPv6 RTT	0.111 s
SFC-MIP6	0.706 s
NEC-MIP6	0.405 s
LIN6	0.192 s

以上の結果から、どの移動体通信プロトコルも登録にかかる時間は平均して 1ms 以内に収まっている。

3.3 移動検知にかかる時間

Home Address 及び Mapping の登録にかかる時間以外に、移動検知にかかる時間も考慮しなければ移動後通信が再開できるまでの時間を算出することはできない。よってそれぞれの移動体通信プロトコルが RA 受信後、Binding Upadate 及び Mapping Update を送信するのにかかる時間を測定した。しかし、MIP6 では、全てのノードが piggyback をするため、移動検知にかかる時間は LIN6 と比較すると大きくなる。今後、piggyback 機構が廃止になった時にまた測定してみるとより正確な時間が分かるはずであるが、今回は piggyback 機構にかかった時間も含めて測定を行った。

測定方法は、MN が移動をし、RA を受信してから Binding Update または Mapping Update を送信するまでの時間をそれぞれ 10 回ずつ測定し、その平均を求めた。**表 6** にそれぞれのプロトコルにおける移動

検知にかかる時間を示す。

表 6　移動検知にかかる時間
Table 6　Time Required to Detect Movement

Protocol	Time
SFC-MIP6	0.761 ms
NEC-MIP6	0.914 ms
LIN6	1.585 ms

この結果からどの移動体通信プロトコルも RA 受信後 binding update 送信までに約 1ms かかる。

この結果から、piggyback 機構を持たない LIN6 の方が MIP6 よりも移動検知にかかる時間が大きいことが分かった。これは、アプリケーション層で移動検知を行う機構を提供していることが原因だと考えられる。また、SFC-MIP6 と NEC-MIP6 に至ってはどちらも piggyback 機構を持っているので移動検知にかかる時間はタイミングに大きく依存してしまい、正確な時間を計ることは難しい。

3.4　Interface Switch 機構の処理コスト

本章では、現在 SFC-MIP6 のみの実装である Interface Switch 機構の性能評価を行う。現在、一つのデバイスで広域接続性を得ることは不可能である。よって、移動体計算機は複数のインターフェースを切り替えながら使用しなければ広域接続性を得ることができない。つまり、移動体通信プロトコルが複数のインターフェースを動的に切り替えながら通信の継続を計らなければ移動体計算機に常時接続性を提供することは不可能である。

本稿では、SFC-MIP6 に実装されている Interface Switch 機構を用いることにより、ネットワーク間を同じデバイスで移動した場合と通信インターフェースを切り替えた場合にかかる処理コストを比較する。

測定環境として、移動体計算機を IEEE の 802.11b に対応した無線 LAN デバイスと携帯電話に接続して、それらを切り替えるのにかかる処理コストを測定した。また、測定には図 1 における CN1 と MN3 を利用した。このネットワークでは RA の送信インターバルは 7 秒である。CN1 からは、MN3 に対して ICMP Echo Message を ping6 に"-f"オプションをつけて連続して送信させることによって切り替えにかかるコストを測定した。**図 2** に Interface Switch 機構の処理にかかる時間を示す。

図 2 の結果から、Interface Switch 機構において通信インターフェースをを切り替えるのにかかる時間は無線 LAN から携帯、携帯から無線デバイスどちらの場合も約 7 秒であることが分かった。しかし、切り替えの処理を行ってから通信が再開するまでの時間は RA を受信するタイミング依存であり、今回はどちらも 7 秒であったが本来なら毎回変化することが予想される。また、RA を受信してから通信インターフェースを切り替え、Binding Update を送信するまでの間は 2 秒以下であることが今回の測定で分かった。

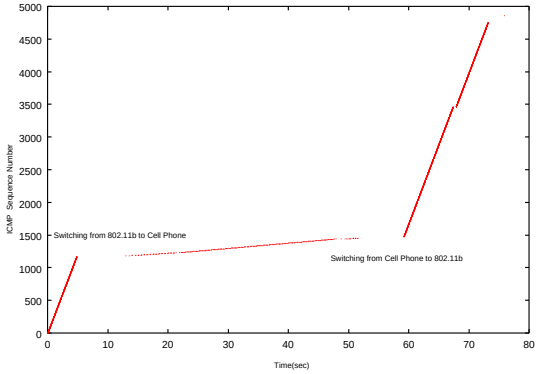

図 2　Interface Switch の処理コスト
Fig. 2　Time Intervals Caused by Interface Switch Function

表 7 に RA 送信インターバルと Interface Switch 機構を利用してから通信が再開されるまでの時間を示す。測定結果として 5 回同じ操作を行って測定した数値の平均を示す。

表 7　Interface Switch 後、通信が再開されるまでの時間と RA 送信インターバルの関係
Table 7　Relationship between Latency of Network Interface Switch and Interval of RAs

Interval of RAs (sec)	Latency of Interface Switch (sec)
4	4.05/5.78
5	5.82/8.03
6	6.63/8.08
7	7.65/8.57
10	11.73/11.78

表 7 における Latency 部分は"携帯電話から無線 LAN デバイス/無線 LAN デバイスから携帯電話" の順に数値が並んでいる。これからも分かるように無線 LAN デバイスのように通信速度が早いデバイスへ切り替えた場合は、通信速度の遅いデバイスへ切り替える場合よりも通信が再開されるまでの時間が短い。Interface Switch 機構における通信が再開されるまでの時間は、切り替え後の通信デバイスの通信速度に依存する部分が大きい。また、Interface Switch 機構は、RA を受信するタイミングに依存しているが、どの場合においても RA を受信してから BU 送信までの時

間は 2 秒以下である。

3.5 性能評価まとめ

本章では、それぞれの移動体通信プロトコルの性能評価を行った結果をまとめる。まず、ヘッダ処理コストであるが、これに関しては SFC-MIP6 が最も処理時間が短かった。MIP6 においては、全てがカーネル内で処理されているため、実装方針や実装方法などの影響を受けやすい。LIN6 に関してはカーネル内での処理が極めて少なく、ほとんどの処理をアプリケーション層にて行っているため、今回の測定では pentium performance counter を利用したカーネル内での処理時間は極めて小さかった。

Home Address/Mapping 登録処理コストであるが、これは MIP6 が piggyback してしまうためそのタイミング次第で結果が大幅に左右されてしまう。しかしどのプロトコルも Binding の管理はハッシュ関数にておこなっているため、HA 自体の性能は Binding の数が増えてもそれほど変わらないはずである。また、処理時間も今回の測定結果からは見えないが先行研究の結果[4] から binding/mapping のエントリーが 1000 以上になった場合でも処理コストは数 micro second に押えられる。登録処理にかかる RTT と比較すると極めて小さい。今後 piggyback 機構が廃止された場合、もう一度 MA と HA の性能の比較を行ってみる必要がある。

Interface Switch 機構に関しては、通信が再開されるまでの時間は RA 依存だが、移動検知後に Binding Update を送信するまでに約 2 秒かかってしまう。これではまだ十分とは言えないため今後 fasthandoff などの技術を用いて移動検知にかかる時間を短縮する必要がある。

これらの結果から、移動体通信プロトコルを使用した場合 CN から MN へ送信した最初のパケットが通る際にかかるコストは式 (1) で求めることができる。

$$\mathsf{T} = \mathbf{M_{t1}} + \mathbf{2x_p} + \mathbf{2H} + \mathbf{C} + \mathbf{M_{t2}} \qquad (1)$$

$\mathbf{M_{t1}}$ = Time for a Packet from MN to Reach HA
$\mathbf{M_{t2}}$ = Time for a Packet from MN to Reach CN
$\mathbf{H}$ = Time for a Packet from HA to Reach MN
$\mathbf{C}$ = Time for a Packet from CN to Reach HA
$\mathbf{X_p}$ = Time for Piggybacking
T = Total Time Required

4. ネットワーク環境別通信処理コスト測定

本章では、実際に MIP6 における処理が異なるネットワークモデルにおいて、MIP6 及び LIN6 の動作確認をすると共に環境別の処理コストを測定する。MIP6 において処理が異なる環境とは CN、MN がそれぞれ Home Link または Foreign Link にいるときの組合せである。

ここで測定された値は測定した環境で大きく変化するものであり、ここでの測定結果は一例にしか過ぎない。よって、全ての測定において通常の IPv6 にて通信を行った場合と比較することによって、それぞれの移動体通信プロトコルを使用することによって生じる通信オーバーヘッドを評価する。以下に MIP6 において処理が異なる通信形態を示す。

- CN,MN 共に Home Link にいる場合
  CN、MN 共に Home Link にいる場合、MIP6 の処理は完全に通常の IPv6 経路制御にもとづいて行われる。
- CN は Home, MN は Foreign Link にいる場合
  この場合、HA は Home Link に MN 宛のパケットは自分に送るように Home Link 内に広告する。Home Link 内のノードから MN 宛にパケットが送られて来た場合、それを傍受し、MN の CoA に転送する。
- CN,MN 共に Foreign Link にいる場合
  どちらも Foreign Link にいる場合は、普通の IPv6 経路制御にもとづいて送られて来たパケットを HA が傍受し、Home Address 宛先オプションが含まれていた場合は MN の CoA にパケットを転送する。

4.1 CN,MN 共に Home Link の場合

移動計算機と通信相手ノードが同じ Home Link にいる状態を想定する。これは、MIP6 が実社会にて使用された場合、社内ネットワークにどちらも接続していて、互いに通信している状態と考えることができる。この状態においては、 MIP6 を使用していない通常の IPv6 通信と変わらない。

図 1 における CN1 と MN1 を利用して測定を行った。表 8 にそれぞれの移動対通信プロトコルの処理にかかった時間を示す。内容としては、MN が Binding Update を送信してから Binding Acknowledgement が HA から戻ってくるまでと、CN との通信が開始されるのにかかった時間を MN が Binding Update を送信してから何秒か、更に通常の IPv6 の通信と比較した時に通信環境によって処理コストが変化するかを調べるために CN から送信されている ICMP Echo Message の RTT を測定した。また、LIN6 においては Mapping Update を MN が送信してから CN から

の通信が開始されるまでの時間を測定した。

表 8　CN,MN が共に Home Link にいる場合の処理コスト
Table 8　MIP6 Process Overhead when both CN and MN are at Home Link

Protocol	Haddr 登録	CN 通信開始	RTT
KAME IPv6	N/A	N/A	0.139 ms
SFC-MIP6	347.0 ms	1.140 ms	0.146 ms
NEC-MIP6	N/A	1.863 ms	0.144 ms
LIN6	0.458 ms	3.148 ms	0.145 ms

表 8 から分かるように、このプロトコルも CN、MN 共に Home Link にいる場合、通常の IPv6 での通信と比較しても RTT において差はない。しかし、LIN6 はアプリケーション層で処理を行っているためか、通常の IPv6 経路制御でパケットが届いているはずだが、通信開始まで MIP6 と比較して倍近く時間がかかっている。

4.2　CN は Home Link、MN は Foreign Link にいる場合

次に、図 1 における CN1 と MN3 を用いて CN が Home Link、MN は Foreign Link にいる場合の測定を行った。この通信形態は、今まで社内で通信していた移動体計算機が社外へ出て通信を始める状態と考えられる。

この場合の処理には、HA が正常に Proxy NDP を使用して MN の Home Address 宛のパケットを傍受し、MN の CoA へ転送できていなければならない。つまり、HA は自分の Home Link に MN の Home Address 宛のパケットは自分宛であることを広告し、Home Link 内にいるノード全てが Home Agent に MN 宛のパケットを送信できるようにしなければならない。

表 9 に測定結果を示す。測定内容は表 8 における測定方法と同一である。

表 9　CN は Home Link, MN は Foreign Link にいる場合の処理時間
Table 9　MIP6 Process Overhead when CN is at Home Link and MN is at Foreign Link

Protocol	Haddr 登録	CN 通信開始	RTT
KAME IPv6	N/A	N/A	0.265 ms
SFC-MIP6	737.5 ms	604.1 ms	0.626 ms
NEC-MIP6	405.1 ms	428.3 ms	1.364 ms
LIN6	137.7 ms	404.3 ms	1.151 ms

この結果から、MN が Foreign Link にいる場合、通信が開始されるまでの時間が大幅に長くなることが分かった。これは、HA が Binding Update を受信してから自分の Home Link に MN 宛のパケットは自分宛であることを広告しなけらばならないのと、傍受したパケットを MN へ転送しなければならないためである。また、LIN6 においても同じネットワークにいないため、MA に Mapping を問い合わせるのにかかった時間が加算される。

4.3　CN,MN 共に Foreign Link にいる場合

最後に、CN、MN 共に Foreign Link にいる場合について測定した。図 1 における CN3 と MN3 を用いて測定を行った。

ここでは、通常の IPv6 経路制御にもとづいて HA まで届けられてきたパケットを HA が傍受し、Home Address 宛先オプションがあった場合、パケットを宛先の MN へ転送することができているかを確認することができる。

測定結果を**表 10** に示す。

表 10　CN,MN 共に Foreign Link にいる場合
Table 10　MIP6 Process Overhead when both CN and MN are in the Foreign Link

Protocol	Haddr	CN	RTT
KAME IPv6	N/A	N/A	0.124 ms
SFC-MIP6	480.3 ms	982.6 ms	0.186 ms
NEC-MIP6	403.2 ms	642.6 ms	0.635 ms
LIN6	284.1 ms	942.2 ms	0.606 ms

全体的に MN が Home Link にいる時と比較した場合、通信が開始されるまでの時間が長い。また、登録にかかる時間に関しては piggyback 問題があるため数値がばらついている。

4.4　移動中の処理コスト

本章では、MN を移動させた際に CN との通信が再開されるまでにかかった時間を測定した。測定を行った環境は、図 1 における CN3 に対して MN は移動しながら CN との通信を継続させる。MN は MN1 to MN2 to MN3 to MN1 の順序で移動させた。CN はその間、MN に対して ICMP Echo Message を 1 秒間隔で送信し続ける。

表 11 にそれぞれが移動した際に RA 受信後通信を再開されるまでに所用した時間を示す。

この測定結果から、MN が移動した後通信が再開されるまでの時間は十分に短いということが分かった。また、MIP6 のどの実装も通信が再開されるまでの時間は大差無く、piggyback 機構によって生じる待ち時間などを考慮するとほとんど測定誤差であるということができる。しかし、LIN6 においては MIP6 のようにカーネル内に NDP Cache を更新する機能を実装し

表 11　移動中の処理コスト
Table 11　Time Required for MN to Restart Communication After Movement

Protocol	MN1-MN2	MN2-MN3	MN3-MN1
SFC-MIP6	1.830 s	1.429 s	3.830 s
NEC-MIP6	4.743 s	1.638 s	2.188 s
LIN6	15.877 s	9.263 s	20.758 s

ていないため、通信が再開されるまでにかなりの時間がかかった。アプリケーションでリンクのステータスを見張り、変化に応じて NDP Cache を更新させるアプリケーションを独自に用意すればより早くなるとのことだが、今回の測定ではあえてそのようなことをせずに性能評価を行った。結果、LIN6 は移動後通信を再開させるのに MIP6 と比較して長い時間を要した。

5. 評価まとめ

今回の測定を通して、移動体通信プロトコルとしての MIP6 と LIN6 の処理にかかる時間などを把握することができた。それぞれの実装の利点や問題のある点を以下に整理する。

5.1 SFC-MIP6

SFC-MIP6 は KAME の IPv6 スタックに直接手を加えているため、他の実装と比較して処理速度が速い。また、SFC-MIP6 は現在唯一 Interface Switch 機構を実装している MIP6 であり、この機構の存在によって移動体計算機が得られるメリットは大きい。よって通信を継続させながら移動体計算機を利用できる範囲を、他の移動体通信プロトコル実装と比較してより広域にすることができるプロトコルであると言える。

問題点としては、現在対応している OS が FreeBSD4.2-RELEASE のみであることと、、現時点では IKE に対応していないことである。

5.2 NEC-MIP6

今回の測定では純粋に MIP6 の処理にかかる時間を測定するため敢えて IPsec を使用せずに測定を行った。しかし、本来 NEC の実装には現在 IKE 機構が組み込まれていて、IPsec を利用して MIP6 での通信を行う環境が整ってきている。

問題点としては、現在 SFC のようにポリシーを設定して通信を行うことができないため、柔軟な通信を行うことができないことである。

5.3 LIN6

LIN6 はほとんどの処理をアプリケーション層にて行っているため挙動が安定していた。また、MIP6 と比較した場合使用方法が簡単であるため、現在 LIN6 を利用して通信を行っている人が増えて来ている。

LIN6 はアプリケーション層にて主な処理を行っているため、カーネル内で全て処理している MIP6 と比較して通信速度にぶれがあった。さらに、MIP6 と比較した場合 NDP Cache 周りの実装が存在しないため、移動後の通信の再開にかなり時間がかかってしまう。また性能とは異なるが LIN6 を用いてグローバルな通信を行う場合、あらかじめ DNS に手で登録してもらわなければならない点も Binding の処理のみでグローバルな移動体通信環境を得ることができる MIP6 と比較して時にデメリットとなる。

6. おわりに

移動体通信プロトコル全体として複数インターフェース切り替え機構の存在は必要である。また、セキュリティー対策も必要である。NEC の実装のように IKE のサポートを行い始めなければ実用性が低くなる。

今回は測定を行わなかったが、どの移動体通信プロトコルも Agent を使用しているため、高速で移動された場合動作しなくなる可能性がある。これは、Binding Update による Binding Cache の更新がアドレスの変化についていけなくなった場合である。さらに、現在 Mobile Networks に関しても研究がなされているが、それぞれのプロトコル上に実装し、ネットワーク単位で移動させることができなければ移動体通信プロトコルとしての利点を生かし切ることはできない。

7. 謝　　辞

本稿を執筆するに当たって多くの助言を下さった慶應義塾大学徳田・村井・楠本・中村・南合同研究室の皆様、また個人的に多数の有益な御助言を下さった NEC ネットワーク研究所の百瀬剛氏と慶應義塾大学寺岡研究室の國司光宣氏に心から感謝致します。

参 考 文 献

1) C.Perkins.: "Mobility Support in IPv6," Internet-Drafts(draft-ietf-mobileip-ipv6-13.txt)
2) 國司光信、石山政治、植原啓介、江崎浩、寺岡文男、:"縮退アドレスモデルに基づく IPv6 上移動透過プロトコルの設計と実装",:マルチメディア・分散・協調とモーバイル (DICOMO) ワークショップ,Jul. 2000.
3) 石山政浩、植原啓介、國司光信、江崎浩、寺岡文男、:"縮退アドレスモデルに基づく IPv6 上移動透過プロトコル",:プログラミング及び応用のシステムに関するワークショップ, Mar. 2000.
4) 湧川隆次、:"Mobile IPv6 を用いた複数インタフェース通信機構の設計と実装",:慶應義塾大学政策・メディア研究科 修士論文, 2000.