固定ノードとIPv4ノードを考慮したLIN6の改良

國司光宣[†] 石山政浩[††] 寺岡文男[†††]

Mitsunobu KUNISHI ,[†] Masahiro ISHIYAMA [††]
and Fumio TERAOKA[†††]

本稿では，IPv6 上の移動透過保証プロトコルである LIN6 を，移動しない固定ノードに対して適用する場合に生じる，グローバルユニークな LIN6 ID の割り当て問題を解決するため，stationary LIN6 アドレスと呼ばれるアドレスを導入する．stationary LIN6 アドレスを用いることにより，固定ノードに対してグローバルユニークな LIN6 ID を割り当てることなく，既存の LIN6 移動ノードと移動透過な通信をすることが可能となる．

さらに本稿では，IPv4 ノードについても考慮し，IPv6 to IPv4 トランスレータである faith に，stationary LIN6 アドレスを適用する．stationary LIN6 アドレスを導入した faith トランスレータは LIN6 トランスレータと呼ばれ，LIN6 トランスレータを利用することにより，IPv6 網中を移動する LIN6 移動ノードは，LIN6 を解釈できない IPv4 ノードと移動透過な通信をすることが可能となる．

1. はじめに

本稿では，IPv6 上の移動透過保証プロトコルである LIN6 を移動しない固定ノードに対して適用する場合について考察する．我々は，現在までに IPv6 上で移動透過保証プロトコル LIN6 について研究を行ない，実装を行なってきた．現在の LIN6 の方式では，移動透過な通信を行なうためには，すべてのエンドノードに対してグローバルユニークな LIN6 ID の割り当てが必要である．一方，固定ノードに対して移動ノードと同様に LIN6 ID を割り当てることを考えると，移動しないにもかかわらず，グローバルユニークな LIN6 ID を必要とし，LIN6 ID のアドレス空間として広大な空間が必要となる．このことは，LIN6 の規模拡張性や普及の面で問題となると考えられる．

そこで本稿では，固定ノードへの LIN6 ID を割り当てることによって LIN6 ID を浪費してしまうという問題を解決するために LIN6 の改良を行なう．この改良では，stationary LIN6 アドレスと呼ばれるアドレスを新たに導入する．stationary LIN6 アドレスは，LIN6 を解釈できるが移動しない固定ノードに対して割り当てるアドレスであり，このアドレスを導入することにより，固定ノードへ LIN6 ID を割り当てる必要が無くなる．その結果，LIN6 ID の浪費を抑えることが可能となる．

さらに本稿では，IPv4 ノードについても考慮し，IPv6 to IPv4 トランスレータである faith[?)] に導入した stationary LIN6 アドレスを適用する．stationary LIN6 アドレスを導入した faith トランスレータを，LIN6 トランスレータと呼ぶ．LIN6 トランスレータは，LIN6 を解釈できない IPv4 ノードと，IPv6 網中を移動する LIN6 ノードの中間に配置することにより，LIN6 移動ノードが，通常の IPv4 ノードと移動透過な通信を実現する環境を提供できる．

2. LIN6 概要

まず，本稿で改良を行なう IPv6 上の移動透過保証プロトコル LIN6 の概要について説明する．LIN6 は，現在の移動透過性が保証できない原因はネットワークアーキテクチャそのものにあると考え，ネットワークアーキテクチャから再考して構築された移動透過保証プロトコルである[?)]．

2.1 LIN6 の基本概念

LIN6 では，ネットワークアドレスが位置に関する情報とノード自体を識別する情報を，分離すること無く保持していることを問題として提起している．この問

† 慶應義塾大学 大学院 理工学研究科 計算機科学専攻
Graduate School of Science and Technology, Keio University
†† 東芝 研究開発センター
R&D Center, Toshiba Corporation
††† 慶應義塾大学 大学院 理工学研究科 計算機科学専攻/ソニーコンピュータサイエンス研究所
Graduate School of Science and Technology, Keio University/Sony Computer Science Laboratories

題は，**ネットワークアドレスの二重性**と呼ばれ，LIN6 ではこの問題を解決するため，ネットワークアドレスが保持している位置指示子とノード識別子という 2 つの情報を概念的に分離する．位置指示子とノード識別子の概念を分離することにより，ネットワーク層より上位層では，ノード識別子を用いて位置に依存しないコネクションを確立し，ネットワーク層では，位置指示子を用いて経路制御を行なうことで，移動透過性を保証する．LIN6 のノード識別子は **LIN6 ID** と呼ばれ，EUI-64 形式を利用したグローバルユニークな識別子として定義される．また，ネットワークの位置情報を示すものとして，LIN6 では既存の IPv6 のネットワークプレフィクスを利用する．

ここで，LIN6 で利用されるアドレスを図 1 に示す．また，送受信におけるアドレスの変換手順を図 2 に示す．　図 1(a) は，既存の IPv6 ユニキャストアドレスの構造を示したものであり，上位 64bit がノードのネットワークへの接続点を示すネットワークプレフィクス，下位 64bit がノードが接続したネットワークで一意となるようなインタフェース識別子という構造である[?]．図 1(b) は，**LIN6 アドレス**と呼ばれるネットワーク上の位置を示すネットワークプレフィクスと LIN6 ID の情報を含んだアドレスである．LIN6 アドレスはパケットを配送するために用いられる．図 1(c) は，**LIN6 汎用識別子** と呼ばれる 64bit の LIN6 ID を 128bit に拡張した識別子である．LIN6 汎用識別子は，あらかじめ決められた 64bit の固定値に対して LIN6 ID を埋め込んでいるため，位置に依存しないアドレスとなる．LIN6 汎用識別子を用いてコネクションを確立することにより，移動の際にもコネクションが継続でき，このアドレスを利用して発呼することにより位置に依存しない発呼が可能となる．

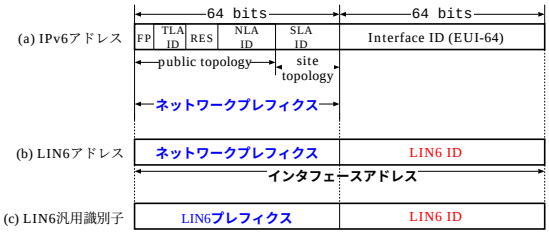

図 1　IPv6 アドレス, LIN6 アドレス, LIN6 汎用識別子

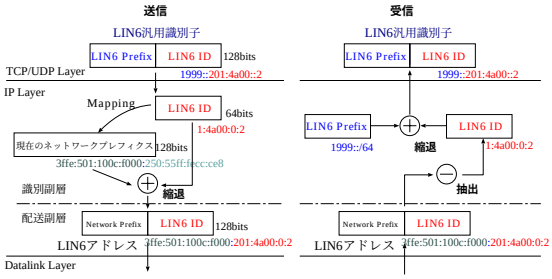

図 2　LIN6 のアドレス変換手順

LIN6 では，通信を開始する際に LIN6 ID と現在のネットワークプレフィクスとの対応づけを取得しなければならない．LIN6 では，この対応関係を **mapping** と呼び，mapping を管理する機構として **Mapping Agent(MA)** を導入する．MA は Mobile IPv6 における HA のようにホームアドレスと CoA の管理情報を基にパケットを中継するのではなく，LIN6 ID とネットワークプレフィクスという動的な情報の管理を行ない，要求に応じてネットワークプレフィクスを通知するという役割を担う．

2.2　LIN6 の通信手順

LIN6 では図 3 で示すような手順で通信を行なう．まず，移動ノードは MA に対して現在のネットワークプレフィクスを登録する (図 3(0))．通信ノードが通信を開始する場合には，まず DNS に対して移動ノードの mapping を管理している MA を問い合わせる (図 3(1))．DNS には，あらかじめ移動ノードの LIN6 ID とそれを管理する MA のアドレスという静的な対応情報を登録しているため，DNS から移動ノードの mapping を管理する MA を取得することが可能である (図 3(2))．次に通信ノードは，移動ノードの mapping を管理している MA に対して，現在のネットワークプレフィクスを要求する (図 3(3))．MA は通信ノードに対して事前に登録されているネットワークプレフィクスを通知する (図 3(4))．この様な手順を踏むことにより，通信ノードは移動ノードの現在のネットワークプレフィクスを得ることが可能であり，この情報を基に LIN6 アドレスを構築し通信を開始できる (図 3(5))．

移動ノードが移動した際には，MA に対してネットワークプレフィクスを更新するとともに，通信をしていた通信ノードに対しても新しいネットワークプレフィクスを通知する．この通知により，移動した際もパケットロスを最小限に抑えた通信をすることが可能である．

3.　LIN6 ID の割り当ての問題

LIN6 ID はグローバルユニーク性を保つため，図 4 で示すように，上位 24bit は我々が管理している OUI，下位 40bit は割り当てられる ID という構造になって

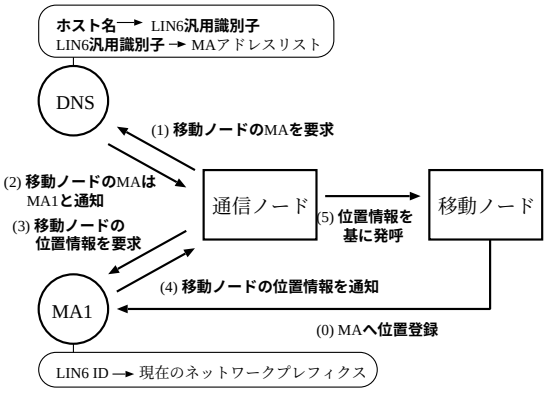

図 3 LIN6 の通信手順

いる．また，各エンドノードに対しての LIN6 ID の割り当ては，現在のところ我々が管理し，申請があり次第割り当てるという方式を取っている．

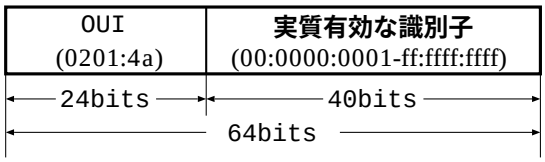

図 4 LIN6 ID の構造

LIN6 の移動透過性をグローバル環境で保証するためには，全てのエンドノードに対してグローバルユニークな LIN6 ID の割り当てを受けなければならなかった．つまり，現在の LIN6 の方式では，本来ならば割り当てが必要無いはずの移動しない固定ノードに対しても，LIN6 ID を割り当てなければ移動透過な通信することが出来なかった．このことは，LIN6 の規模拡張性の問題が生じるだけでなく，LIN6 の普及の妨げにもなると考えられる．

4. stationary LIN6 アドレス

本稿では，固定ノード (stationary node) に対するアドレス割り当ての問題を解決する手法として，stationary LIN6 アドレスと呼ばれるアドレスの方式を提案する．stationary LIN6 アドレスは，固定ノードが LIN6 ID の割り当て無しに LIN6 移動ノードと移動透過な通信を行なうためのアドレスである．

4.1 stationary LIN6 アドレスの構造

まず，stationary LIN6 アドレスが保持しなければならない情報について考える．stationary LIN6 アドレスは固定ノードに対して割り当て，かつ LIN6 移動ノードと移動透過な通信が可能となるようなアドレス構造でなければならない．このような要件をみたす stationary LIN6 アドレスは図 5 で示すような構造となる．

stationary LIN6 アドレスの上位 64bit は，位置に依存したネットワークプレフィクスである．stationary LIN6 アドレスは固定ノードに対して割り当てるため，この値は変化しない．一方，stationary LIN6 アドレスの下位 64bit は，LIN6 移動ノードがパケットを受信した際に，アドレスの構造だけで stationary LIN6 アドレスかどうかを判別しなければならないため，LIN6 ID に似た特別な構造となる．

図 5 に示した stationary LIN6 アドレス構造は，図 1(b) で示した LIN6 アドレスに非常に似た構造となっているが，下位 64bit の扱いが異なる．LIN6 アドレスと stationary LIN6 アドレスの判別方法については次節で述べるが，stationary LIN6 アドレスの下位 64bit は先頭 24bit に関しては特別な構造であるが，残り 40bit に関しては接続したネットワークの中で一意であれば自由に割り当て可能な値である．

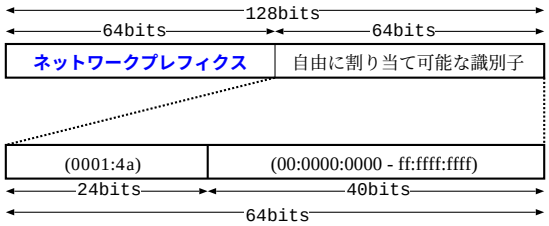

図 5 LIN6 stationary アドレスの構造

4.2 stationary LIN6 アドレスの判別方法

まず，stationary LIN6 アドレスと通常の IPv6 アドレスの判別方法について述べる．stationary LIN6 アドレスと通常の IPv6 アドレスの判別方法は，LIN6 アドレスと通常の IPv6 アドレスの判別方法と同様に，受信したアドレスから抽出した下位 64bit の先頭 24bit が我々の取得した OUI かどうかで判別する．stationary LIN6 アドレスと LIN6 アドレスはこの OUI の構造までは同型であるため，他の方法で判別しなければならない．

stationary LIN6 アドレスと LIN6 アドレスとの判別は，EUI-64 で定められている universal bit の反転の有無によって行なう．下位 64bit を調べた際，universal bit が立っている場合には，下位 64bit は LIN6 ID であるため LIN6 アドレスと判断し，立っていない場合には 128bit 全体で stationary LIN6 アドレスであると判断する．ネットワーク層ではこの判別を行

ない，判別の結果が LIN6 アドレスの場合には，通常の LIN6 の通信手順によって，LIN6 汎用識別子へ変換が行なわれる．判別の結果が stationary LIN6 アドレスの場合には，位置情報である上位 64bit は変化しないため，LIN6 汎用識別子への変換は行なわれない．

4.3 LIN6 ノードと stationary LIN6 ノードの通信手順

グローバルな LIN6 ID の割り当てを受けた LIN6 移動ノードと，独自で stationary LIN6 アドレスを割り当てた stationary LIN6 ノードとの通信について例を挙げて説明する．

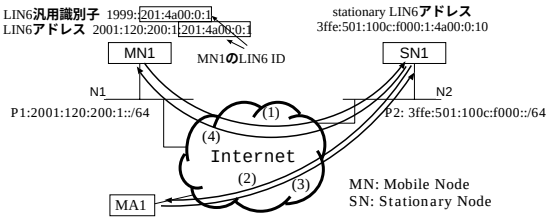

図 6 LIN6 ノードと stationary LIN6 ノードの通信

図 6 に LIN6 移動ノード (MN1) と，stationary LIN6 ノード (SN1) との通信例を示す．まず，N1 に接続している LIN6 移動ノードから，N2 に接続している LIN6 stationary ノードに対して発呼を行なう．この場合，LIN6 移動ノードで生成される IP ヘッダの構成は，上位層では送信元アドレスは MN1 の LIN6 汎用識別子 (1999::201:4a00:0:1)，送信先アドレスは SN1 の stationary LIN6 アドレス (3ffe:501:100c:f000:1:4a00:0:10) となる．このパケットは，ネットワーク層で LIN6 汎用識別子である送信元アドレスのみ MN1 の LIN6 アドレス (2001:200:120:1:201:4a00:0:1) へ変換され，ネットワークへ送信される (図 6(1))．

MN1 から送信されたパケットを受け取った SN1 では，次のような処理が行なわれる．まず，パケットの IP アドレスの下位 64bit を抽出し，LIN6 アドレスかどうか判別する．この例の場合，送信先アドレスは universal bit が立っていない (先頭 24bit が 0001:4a) ため stationary LIN6 アドレス，送信元アドレスは universal bit が立っている (先頭 24bit が 0201:4a) ため LIN6 アドレスであると判別する．stationary LIN6 アドレスと判別された送信先アドレスは，LIN6 汎用識別子へと変換されず，そのまま上位層へと渡される．一方，LIN6 アドレスと判別された送信元アドレスは LIN6 汎用識別子へと変換され，上位層へと渡される．

SN1 から MN1 への返信の際には，MN1 の LIN6 汎用識別子を LIN6 アドレスへと変換する必要があるため，通常の LIN6 の通信と同様の手順を踏む．SN1 は，MN1 の MA である MA1 を DNS から取得し，MA1 から MN1 の現在のネットワークプレフィクスを取得する (図 6(2), (3))．この手順により MN1 の LIN6 汎用識別子を LIN6 アドレスへと変換しパケットを返信する (図 6(4))．

ここで，上位層で確立されるコネクションについて考察すると，コネクション識別子は MN1 の LIN6 汎用識別子と SN1 の stationary LIN6 アドレスとなる．MN1 の LIN6 汎用識別子は位置に依存しないアドレスであり，SN1 の stationary LIN6 アドレスは移動しないアドレスであるため，MN1 が移動した際にもコネクションは継続可能である．

この例では，LIN6 移動ノードから stationary LIN6 ノードに対する通信例を挙げたが，逆に stationary LIN6 ノードから LIN6 移動ノードに対する通信も可能である．その場合，SN1 はまず MA1 から MN1 の現在のネットワークプレフィクスを取得し，MN1 に対して発呼するという手順になる．

4.4 stationary LIN6 アドレスの利点

stationary LIN6 アドレスを導入した場合の利点について考察する．ここでは例として，stationary LIN6 アドレスを導入したノードを MA として利用することを考えてみる．stationary LIN6 アドレスを導入した MA は，導入しない MA に比べ，次のような利点を持つ．stationary LIN6 アドレスを導入した MA は，LIN6 移動ノードを LIN6 汎用識別子によって一意に識別することが可能である．したがって，IPsec[7] の Security Association(SA) を事前に MA と移動ノードに作成しておけば，IPsec を利用した認証が可能となる．事前に確立した SA は，LIN6 移動ノードが移動した際にも変化しないため，LIN6 移動ノードが移動するたびに更新する必要もない．

5. faith トランスレータへの応用

本稿では，IPv4 ノードについても考慮にいれ，IPv6 to IPv4 トランスレータである faith に提案した stationary LIN6 アドレスを適用する．

5.1 faith 概要

実際にどのように利用するかを説明する前に，まず faith の概要について説明する．faith は，IPv6 通信のみ可能なホストから，IPv4 ネットワークに接続するために用いられる，TCP relay の技術を利用したトランスレータである． 図 7 を用いて，IPv6 ノードが IPv4 ノードと通信する様子を説明する．faith では，**faith**

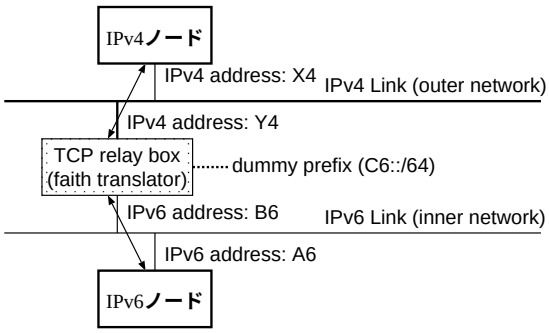

図 7 faith を用いた通信

prefix と呼ばれる dummy network prefix(C6::/64) を割り当てる．この例では，A6 というアドレスを持つ IPv6 ノードから X4 というアドレスを持つ IPv4 ノードに対して通信を開始する場合を考える．IPv6 ノードから IPv4 ノードに対して通信を開始する場合，この faith prefix を利用し，C6::X4 というアドレスに対して通信を開始する．faith では，このパケットを受信し，IPv6 ネットワーク側である A6 と C6::X4 間のコネクションを確立する．次に，C6::X4 の下位 32bit を抽出することにより IPv4 ノードの IPv4 アドレスを取得し，faith の IPv4 アドレスである Y4 を用いて IPv4 ノードのアドレスである X4 とコネクションを確立する．このように faith では，IPv6 ネットワーク側のコネクションと IPv4 ネットワーク側のコネクションを分離し，2 本のコネクションを確立することにより，IPv6 ノードから IPv4 ノードへのコネクションを透過的に確立している．

5.2 LIN6 トランスレータ

stationary LIN6 アドレスと faith を利用して構築したトランスレータを本稿では **LIN6 トンランスレータ**と呼ぶ．LIN6 トランスレータは，faith を用いて IPv4 との通信を行なう際のアドレスが，faith prefix + IPv4 アドレスという構造になることを利用する．LIN6 トランスレータを LIN6 を解釈できない IPv4 ノードと，IPv6 網中を移動する LIN6 ノードの中間に配置することにより，LIN6 移動ノードが，通常の IPv4 ノードと移動透過な通信を実現する．

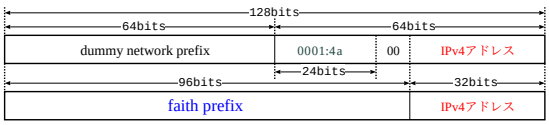

図 8 LIN6 トランスレータにおける faith prefix

LIN6 トランスレータで用いる faith prefix の構造を図 8 に示す．まず，128bit のアドレス空間の上位 96bit は faith prefix，下位 32bit は IPv4 アドレスとして考える．上位 96bit の faith prefix の内部は，64bit の dummy network prefix と 24bit の特別な値，そして 8bit の 0 という構造である．この構造において，24bit の特別な値の構造を stationary LIN6 アドレスの形式である 0001:4a とすると，LIN6 トランスレータを介した通信をする際のアドレスは，[network prefix (64bit)] + [0001:4a00(32bit) + IPv4 アドレス (32bit)] という構造になる．このアドレス構造は，本稿で導入した stationary LIN6 ノードのアドレスの構造として捉えることが出来る．

ここで，LIN6 移動ノードから，LIN6 トランスレータを介して，通常の IPv4 ノードに対して通信を行なった場合に確立されるコネクションに着目し，LIN6 トランスレータを導入したことによって生じる効果を説明する．まず，IPv6 網側の LIN6 移動ノードと LIN6 トランスレータの間のコネクションは，LIN6 移動ノードと stationary LIN6 ノード間で確立されるコネクションであると考えることができる．次に LIN6 トランスレータの IPv4 側と IPv4 ノードの間は，通常の IPv4 のコネクションが確立される．LIN6 移動ノードと LIN6 トランスレータ間のコネクションは，前述の通り LIN6 移動ノードが移動した際も継続可能なコネクションである．さらに LIN6 トランスレータは固定ノードであるため，LIN6 トランスレータと IPv4 ノード間のコネクションも継続できる．つまり，結果として LIN6 移動ノードと IPv4 ノード間で移動透過な通信を行なうことが可能となる．

この方式の利点は，LIN6 の通信は LIN6 トランスレータで終端されるため，IPv4 ノード側で LIN6 を解釈する必要がないにも関わらず，IPv6 網側では移動透過な通信が出来るという点である．

5.3 通 信 手 順

LIN6 トランスレータを用いて IPv4 ノードと通信を行なう場合の通信手順を図 9 に示す． ここでは具体的な例として，10.0.0.1 の IPv4 アドレスを持つ IPv4

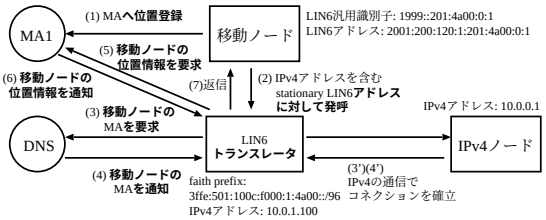

図 9 LIN6 トランスレータを用いた通信手順

ノードに対して，LIN6 移動ノードが LIN6 トランスレータを介して通信する場合を考える．LIN6 移動ノードが IPv4 と通信を開始するためには，送信先アドレスを faith prefix + IPv4 アドレスとなるように指定し通信を開始する．この場合の送信先アドレスは 3ffe:501:100c:f000:1:4a00::10.0.0.1 となり，LIN6 トランスレータへパケットが送信される．IPv6 のネットワーク側では，LIN6 移動ノードと stationary LIN6 ノードとの通信で説明したような手順により，LIN6 のコネクションが確立される (図 9(3)-(7))．一方，IPv4 のネットワーク側では，LIN6 移動ノードから送信されたパケットの送信先アドレスの下位 32bit を抽出し，抽出した IPv4 ノードのアドレスである 10.0.0.1 と LIN6 トランスレータの IPv4 アドレスである 10.0.1.100 間で通常の IPv4 のコネクションが確立される (図 9(3')(4'))．この通信で確立される 2 本コネクションについて考えると，1 本は IPv6 側で確立される LIN6 移動ノードの LIN6 汎用識別子 (1999::201:4a00:0:1) と IPv4 アドレスが埋め込まれた faith のアドレス (3ffe:501:100c:f000:1:4a00::10.0.0.1) の組で識別されるコネクション，もう 1 本は LIN6 トランスレータの IPv4 アドレス (10.0.1.100) と IPv4 ノード (10.0.0.1) の組で識別されるコネクションである．

5.4 totd の利用

LIN6 トランスレータを介して IPv4 ノードと通信するため必要な情報について分析してみると，IPv4 ノードの IPv4 アドレスと LIN6 トランスレータの dummy network prefix が必要である．これらを通信の度に逐次調べてから通信を開始するのは非効率的であり，現実的ではない．そこで本稿では，この問題を回避するために faith と連動して動作する totd と呼ばれる DNS Proxy を利用する．

totd は，A レコードと AAAA レコードを変換する DNS Proxy である．totd は，FQDN から IP アドレスへの変換要求を受けた際，A レコードしか持たない場合も AAAA レコードへ変換して返答する．AAAA レコードへの変換は，まず faith prefix を付加し，A レコードとして返って来た 32bit のアドレスを AAAA の下位 32bit に埋め込むことにより実現する．つまり，totd を介して得られた AAAA レコードは，faith を経由して IPv4 ノードと通信が可能となるような形式となっており，LIN6 トランスレータの場合でも faith で利用する場合と全く同じ方法で導入することが可能である．

LIN6 トランスレータで totd を利用する場合，totd の faith prefix に 96bit の faith prefix である dummy network prefix + 0001:4a00 という構造を設定しておき，nameserver として totd が動作しているノードを指定する．このような設定を行なうことにより，IPv4 ノードの FQDN を指定して通信を開始すると，totd により，IPv4 ノードの IPv4 アドレスが埋め込まれた stationary LIN6 アドレスが AAAA レコードとして返され，stationary LIN6 アドレスへ発呼が行なわれる．さらに stationary LIN6 アドレスは，LIN6 トランスレータのアドレスであるため，faith により TCP relay され，LIN6 トランスレータを介して IPv4 ノードと通信できる．

5.5 LIN6 トランスレータの利点

LIN6 トランスレータを導入する利点について考察する．ここでは，既存のトランスレータでは実現されておらず，LIN6 トランスレータによって実現可能となる機能について考察する．LIN6 トランスレータは，stationary LIN6 アドレスを利用しているため，stationary LIN6 アドレスの導入の利点で述べたように LIN6 移動ノードを汎用識別子によって一意に識別し，SA を事前に作り込むことによって IPsec の認証を用いることが可能となる．既存のトランスレータは，位置に依存せずノードを識別することが不可能であったため，トランスレータの利用を制限するためには，IP 層より上位層で何らかの認証が必要であった．しかし，LIN6 トランスレータでは，IP レベルで移動ノードを識別することが可能となるため，IP レベルで認証することが可能となる．今後の IPv6 への移行期にトランスレータは重要な技術であり，さまざまな場面で利用されることを考慮すると，IP レベルでの認証が出来るということは大きな利点であると考えられる．

6. 実　　装

stationary LIN6 アドレスは，既に配布している LIN6 のソースコードに実装されている☆．現在 stationary LIN6 アドレスの動作が確認されているプラットホームは，NetBSD 1.5.1, FreeBSD 4.3, BSD/OS 4.2, OpenBSD 2.9 である．これらのプラットホームで stationary LIN6 アドレスを設定し，さらに faith, totd の設定を行なえば，LIN6 トランスレータとしても動作する．

7. 結　　論

本稿では，stationary LIN6 アドレスと呼ばれるアドレスを導入し，移動しない固定ノードに対して適用

☆ http://www.lin6.net/

する場合に生じる，グローバルユニークな LIN6 ID の割り当ての問題を解決した．stationary LIN6 アドレスを用いることにより，固定ノードに対してグローバルユニークな LIN6 ID を割り当てることなく，既存の LIN6 移動ノードと移動透過な通信が可能となった．さらに本稿では，IPv4 ノードについても考慮し，stationary LIN6 アドレスを利用した，LIN6 トランスレータと呼ばる faith トランスレータを提案した．LIN6 トランスレータを利用することにより，IPv6 網中を移動する LIN6 移動ノードは，LIN6 を解釈できない IPv4 ノードと移動透過な通信をすることが可能となった．

LIN6 トランスレータを設置することにより，LIN6 移動ノードは実質的に IPv6 アドレスの割り当てだけ受けられれば，何の不自由も無く IPv4 ノードと通信を行なうことが可能となる．したがって，LIN6 トランスレータは，LIN6 の普及だけでなく，IPv6 の普及にも繋がる応用となる．

今後の課題としては，Mobile IPv6 でも議論されているように移動ノードの認証についても検討を重ねなければならない．また，高速ハンドオフ処理など，LIN6 におけるマイクロモビリティに関する研究については，Mobile IPv6 のようなトンネルを多用するような手法ではなく，ネットワークプレフィクスを書き換えるなどの方法で進めていきたいと思う．