IPv4-to-v6 トランスレータの評価と改良

2007MI080 伊藤 徹　　2007MI242 田財 彰久
指導教員　後藤 邦夫

1　はじめに

近年，経済発展が著しい BRICs 諸国を中心に，インターネット普及率が急激に増加している．そのため，従来のインターネットプロトコルにあたる IPv4 を利用したアドレス資源の枯渇問題がより深刻化し，IPv4 から IPv6 への円滑な移行を促進させる技術が求められている．

本研究では，その技術を実現するために，IPv4 のネットワークから IPv6 のネットワークへ通信することが出来る IPv4-to-v6 トランスレータ（以下トランスレータ）の性能評価と改良をする．

2010 年度森，畑佐の卒業研究「IPv4-to-v6 トランスレータの実現とネットワークエミュレータ上での評価」でトランスレータの基本構造の作成と性能評価をし，さまざまな課題を挙げた [2]．過去の評価はエミュレータ上で実施されたものであり，実用的な通信環境における評価はされていない．

そこで，本研究では既存システムを実ネットワークに則した環境で再評価し，新しい課題を示す．過去の課題を含めた改良を行い，より完成度の高いトランスレータを実装することを目的とする．

なお，実験の環境構成と評価を伊藤が担当し，課題の改善を田財が担当した．

2　システム概要

本節では先行研究で使用したネットワークエミュレータ GINE の概要，利用想定について説明する．

2.1　GINE(GOTO's IP Network Emulator) の利用

GINE は本研究室で開発中のネットワークエミュレータである．先行研究の GINE の利用目的は，実験における IPv4 及び IPv6 の仮想ホストを生成することと，データリンク層でフレームをやり取りする機能を使用することだった．フレームのやり取りには，PFPaket, FrameQueue と呼ばれる GINE クラスを使用する [3]．

PFPacketin でフレームを受け取り，enqueue でフレームを FrameQueue に渡す．FrameQueue 内でフレームの書き換えをし，FrameQueue を Timer で処理する．dequeue で書き換えたフレームを PFPacketout に渡す．PFPacketout で指定したインターフェースにフレームを渡す．この一連の処理は，3.1 節で説明する変換ゲートウェイ (TG) に組み込まれる．

2.2　利用想定

本研究では，実ネットワークに則した環境で再評価するために，利用想定を定義した上で実験をする．以下の図 1 のような状況を想定する．

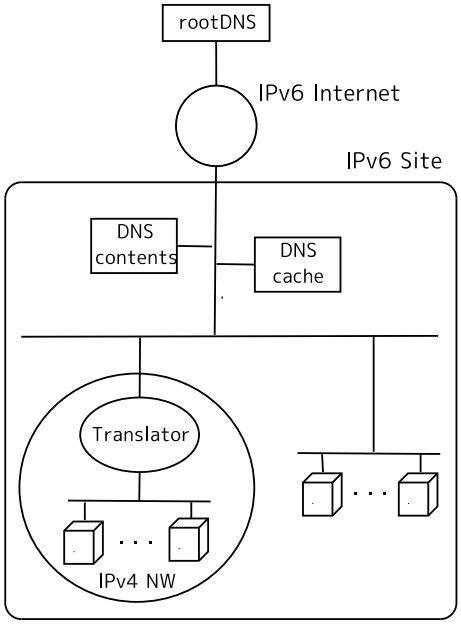

図 1　利用想定

図 1 では，インターネットでは IPv6 が普及し，とある IPv6 インターネットの LAN 内に IPv4 ネットワークが孤立している状況を想定する．IPv6 ネットワーク同士の通信は可能であるが，IPv4 ネットワークは他のネットワークと通信ができない．そこでトランスレータを取り入れることで，IPv6 ネットワークとの通信を可能にする．

DNS 正引きは，DNS で重要な役割を果たす rootDNS サーバから，自ドメイン DNS サーバを含める DNS コンテンツサーバに階層的に問い合わせる．DNS キャッシュサーバは，問い合わせ情報を一時的に保存することで以降の問い合わせを高速化する．

また，利用アドレスは IPv4，IPv6 とも割り当てられたグローバルアドレスを想定する．

3　トランスレータの構成と通信構成

本節では，先行研究のトランスレータの構成とネットワーク例について述べる．

トランスレータは以下の 3 つ要素で構成する．

- 変換ゲートウェイ (TG)
- 改造 DNS
- 変換表

先行研究では，TG と改造 DNS をスレッドとして作成した．TG と改造 DNS をマルチスレッドで動かし，変換表をオブジェクトとして生成して管理することで，通

信の高速化を実現した．

3.1 変換ゲートウェイ (TG)

TG は IPv4 と IPv6 のパケットを相互に変換する機能である．IPv4 と IPv6 は互換性のないプロトコルであるため， IPv4 と IPv6 間で通信するためにプロトコルやパケットを各ネットワークで理解できる形に変換する必要がある．

TG は主に以下の処理をする．

- IP ヘッダ変換
- チェックサム計算
- 上位層ヘッダ変換

3.2 改造 DNS

改造 DNS は，DNS レコードの書き換えと，仮の IPv4 アドレス生成をする．先行研究の改造 DNS は，主に IPv4host から聞かれる FQDN(Fully Qualified Domain Name) に対して処理をする．IPv4 に対応する A レコードを，IPv6 に対応する AAAA レコードに書き換えた後，IPv6 側 DNS に IPv6 アドレスを聞きにいく．返ってきた IPv6 アドレスから仮の IPv4 アドレスを生成し，AAAA レコードを A レコードに書き変えた後，IPv4host に DNS パケットを返す．

3.3 変換表

変換表は仮 IPv4 アドレスと IPv6 アドレスを一つのオブジェクトとして登録・管理する [1]．単純変換出来ない IPv6 アドレスが変換表に渡された時，対応する仮 IPv4 アドレスがあるかどうか調べる．該当するオブジェクトがある場合は，IPv6 アドレスに対応する仮 IPv4 アドレスを返す．該当するオブジェクトが無い場合は，改造 DNS を用いて仮 IPv4 アドレスを生成し，変換表に登録する．

3.4 通信構成

先行研究では，TG，改造 DNS，変換表の 3 つを組み込んだトランスレータを用いて，図 2 のような 1 対 1 のネットワーク例を定めた．本研究でも引き続き，このネットワーク例に基づいてトランスレータを実装する．

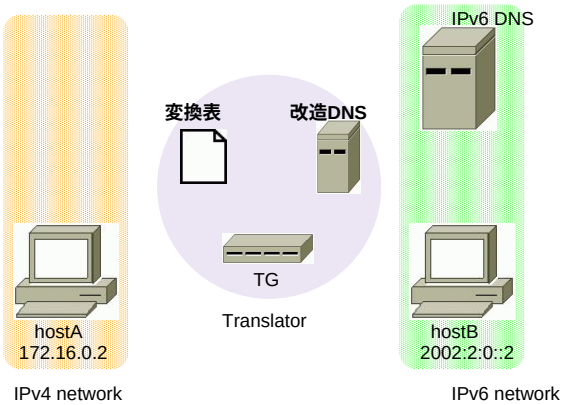

図 2　ネットワーク構成

送信元の hostA は IPv4host，宛先の hostB は IPv6host であるとする．IPv6 側の DNS には，hostA と hostB のドメイン名に対応した IPv6 アドレスが記録されているものとする．

なお，本来の DNS サーバは複数台の DNS が存在し，正引きで階層的な問い合わせが発生する．本研究では，IPv6 側 DNS を複数ドメインの情報を提供可能な仮想 rootDNS とし，実装を簡略化するものとする．

表 1 に各 host のアドレス情報を示す．

表 1　IPv4 アドレスと IPv6 アドレスの関係

host 名	IPv4 address	IPv6 address
hostA. mydomain	172.16.0.2 (実 IPv4)	2001:0:ffff ::172.16.0.2 (変換用 IPv6)
hostB. anydomain	10.x.y.z (仮 IPv4)	2002:0:2::2 (実 IPv6)

hostA の実 IPv4 アドレスを 172.16.0.2，hostB の実 IPv6 アドレスを 2002:0:2::2 とする． hostA 側で，IPv4 と相互変換が出来る変換用 IPv6 アドレスを定めるものとする． IPv6 アドレスプールを 2001:0:ffff/48 とし，末尾に実 IPv4 アドレスを埋め込んだ 2001:0:ffff::172.16.0.2 を変換用 IPv6 アドレスとする．

hostB 側の実 IPv6 アドレスを使って，仮 IPv4 アドレスを生成する．IPv4 アドレスプールを 10.0.0.0/8 とし，ホスト部に実 IPv6 アドレスの MD5 digest ハッシュ値の末尾 24bit を埋め込んだ 10.x.y.z を仮 IPv4 アドレスとする．

4 実験の構成

本節では，実機を用いた実験の構成について説明する．実験の構成は 2.2 節の利用想定に基づいて構築するものとする．トランスレータと DNS サーバは動作や性能を測るために実機を用いる．ルータ，IPv4host，IPv6host の部分はネットワークエミュレータ GINE を用いて仮想マシンを用意する．実験に用いるアドレスについては IPv4 の部分はプライベートアドレス，IPv6 の部分はドキュメント用に用意されたグローバルアドレスを使用する．

トランスレータ，DNS，仮想ホストの生成に使用する PC のスペックは以下の表 2 に示す．

表 2　PC スペック

PC	Power Edge 840
OS	Ubuntu 10.04 Jaunty Jackalop 64 bit
CPU	Intel(R) Xeon(R) CPU X3220 @ 2.40GHz (Quad Core)
メモリ	2.0GB
NIC	Broadcom BCM5721

実験の構成は以下の図 3 に示す.

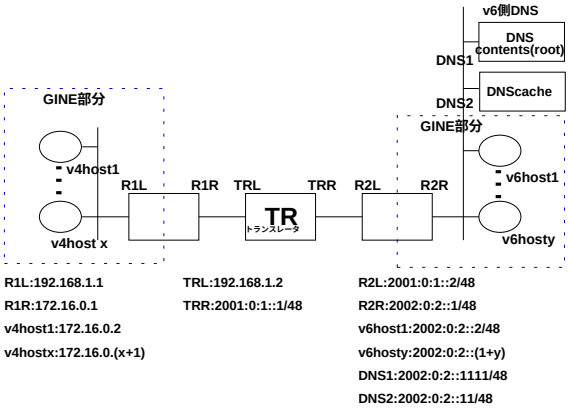

図 3　実験構成

5　評価

本節ではトランスレータと DNS の性能評価について述べる.

5.1　トランスレータの評価

4 節で構築した実機を用いた実験環境を使用し，iperf を用いて UDP，TCP でのスループットを 10 回測定し，平均値を算出する．また，先行研究で構築した仮想ネットワーク上でスループットを測定した結果と比較する.

UDP で iperf を行う場合，帯域幅を指定しないと送信するトラフィックが 1Mbps となるので，オプションで帯域幅を指定する.

5.2　1 対 1 の性能評価

1 対 1 の測定結果の比較を表 3 で示す.

表 3　本研究 (実機)

	プロトコル	最大	最小	平均
先行研究	TCP	368	291	328
	UDP	232	231	231
本研究	TCP	843	831	838
	UDP	748	724	736

(スループット単位：Mbps)

測定結果から，仮想ネットワーク上に比べ実機を用いると，スループットが大きく上昇した事が確認された．先行研究では仮想ネットワーク上で全ての処理を単体で行うため，CPU 負担が大きくなりスループットが低下したと考えられる．そこで，新しい課題として CPU 負担がかかった状態でもスループットを向上させる事が挙げられる.

6　トランスレータの改善

本節では，先行研究のトランスレータから改善した項目について説明する.

本研究ではスループットの改善と，IPv6host から IPv4host の通信を主な課題として扱う.

6.1　スループットの改善

スループットの改善とは，通信速度を上げてパフォーマンスを向上させることを目的とする．2.1 節で述べた通り，先行研究のトランスレータでは，FrameQueue を用いてフレームのやり取りをしていた．FrameQueue 内にフレームを enqueue して，フレームを溜めてから変換処理をするため，処理に無駄が生じる．そこで，本研究ではスループットを改善する目的で，TG に以下の変更を加える.

- FrameQueue にフレームを溜める処理の廃止
- フレームの変換処理をマルチスレッド化

これらの新機能を追加した TG を利用した通信形態を図 4 に示す.

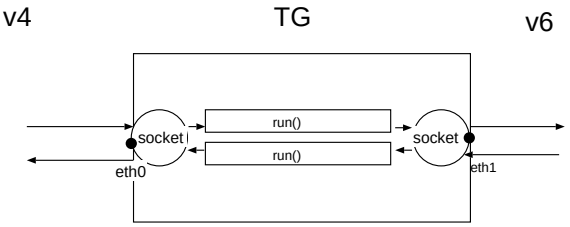

図 4　フレーム変換処理の簡略化

このように，FrameQueue と PFPacket を省略し，フレームの変換処理をマルチスレッド化することにより，スループットの改善が期待できる.

新しく実装した TG を用いて，先行研究の仮想的に構築した実験環境と，4 節で構築した実機を用いた実験環境で再評価する．また，本研究で実装した TG を新 TG，先行研究で実装した TG を旧 TG とする．iperf を用いて UDP，TCP でのスループットを 10 回測定し，平均値を算出する.

始めに，先行研究で構築した仮想ネットワークを使用し，旧 TG と新 TG 上でスループットを測定し，比較する.

測定結果は表 4 に示す.

表 4　旧 TG と新 TG の比較 (仮想ネットワーク)

	プロトコル	最大	最小	平均
旧 TG	TCP	368	291	328
	UDP	232	231	231
新 TG	TCP	585	569	574
	UDP	609	591	601

(スループット単位：Mbps)

測定結果から，TCP では 200Mbps，UDP では 350Mbps ほど増加しており，仮想ネットワーク上で CPU 負荷が大きい状態でもスループットが改善されていることが確認された.

次に，4 節で説明した実機を用いた実験環境を使用し，旧 TG と新 TG でスループットを測定し，比較する.

測定結果は表 5 に示す.

表 5　旧 TG と新 TG の比較 (実機)

	プロトコル	最大	最小	平均
旧 TG	TCP	843	831	838
	UDP	748	724	736
新 TG	TCP	871	862	868
	UDP	749	746	748

(スループット単位：Mbps)

測定結果より，スループットは TCP，UDP ともにほとんど変わらないことが確認された.

6.2　IPv6host から IPv4host の通信

IPv6host から IPv4host の通信とは，IPv6host から開始される通信を指す．IPv6host が DNS 正引きで IPv4host のアドレスを調べ，パケットを送信する．最終的に，IPv6host にパケットが返ってきた段階で通信終了とする.

先行研究の IPv4host から開始される通信と比べた大きな違いは，アドレス生成の段階が異なることが挙げられる．IPv4host から開始される通信では，DNS 正引き段階で IPv6host の実アドレスを問い合わせ，仮 IPv4 Address を生成する．本研究で実装する IPv6host から開始される通信では，DNS 正引き段階では IPv6host の実アドレスを扱わないため，パケット送信段階で仮 IPv4 アドレスを生成し，変換表に随時登録する.

なお，IPv4host の変換用 IPv6 アドレスは変換表に静的に登録する.

実際の通信手順について通信シーケンス図 5 を用いて説明する.

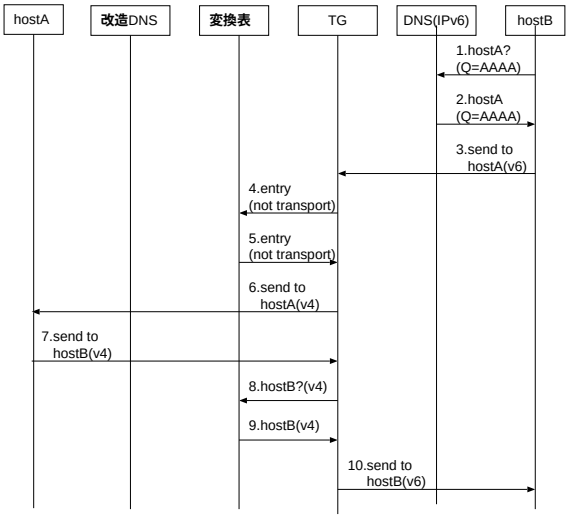

図 5　通信シーケンス図

各シーケンス番号に該当する処理の詳細について説明する．アドレスの記述は，アドレス形式 (Src/Dst) で統一する．例えば，Source のアドレス形式が実 IPv4 アドレスの場合，実 IPv4(Src) のように記述する.

- Source:hostB Destination:hostA
  1. hostB が IPv6 側 DNS に hostA.mydomain の変換用 IPv6(Dst) を問い合わせる.
  2. IPv6 側 DNS が変換用 IPv6(Dst) を hostB に返す.
  3. hostB が hostA に対してパケットを送信する.
  4. TG が実 IPv6(Src)，変換用 IPv6(Dst) を変換表に渡す.
  5. 変換表で実 IPv6(Src) に対応する仮 IPv4(Src) を作成・登録する．また，変換用 IPv6(Dst) の末尾 32bit を抜き取り，それを実 IPv4(Dst) とする．変換後，二つのアドレスを TG に返す.
  6. TG がヘッダ変換したパケットを hostA へ送信する.
- Source:hostA Destination:hostB
  7. 折り返し，hostA が hostB に対してパケットを送信する.
  8. TG が変換表に実 IPv4(Src) に対応する変換用 IPv6(Src)，仮 IPv4(Dst) に対応する実 IPv6(Dst) を問い合わせる.
  9. 変換表は変換用 IPv6(Src)，実 IPv6(Dst) を TG に返す.
  10. TG がヘッダ変換したパケットを hostB へ送信する.

7　おわりに

本研究では完成度の高いトランスレータを実装するために，再評価して発覚した新しい課題と，過去の課題を含めた改良を試みた．結果として，実機を用いたトランスレータの性能評価をしたこと，Framequeue や PFPacket を使用しないフレーム処理を追加したこと，スループットが向上したこと，IPv6host から IPv4host の通信が可能であることを示した.

今後の課題は以下の通りである.

1. ハニーポット利用時の TCP 終了処理.
2. DNS 逆引き処理の実装.
3. DNS 手抜き処理の修正.

参考文献

[1] 角川宗近：ネームサーバとヘッダ変換ゲートウェイを用いた IPv4 と IPv6 の相互互換，奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻修士論文 (1997).

[2] 森 知恵, 畑佐宏輝：IPv4/IPv6 トランスレータの実現とネットワークエミュレータ上での評価, 卒業論文, 南山大学 数理情報学部 情報通信学科 (2010).

[3] Ihara, A., Murase, S. and Goto, K.: IPv4/v6 Network Emulator using Divert Socket, *Proc. of 18th International Conference on Systems Engineering(ICSE2006)*, Coventry, UK, pp.159–166(Sep.2006).