ユーザポリシーに従ったインターフェイスセレクションの提案
The Interface Selection According to User Policy

木ノ下 稔† 知念 賢一† 篠田 陽一§

† 北陸先端科学技術大学院大学 情報科学研究科
§ 北陸先端科学技術大学院大学 情報科学センター

概 要

エンドホストでは、接続するネットワークが複数ある時、宛先 IP アドレスだけでなく、使用するネットワークの性質を考慮して、アプリケーション毎に制御が行いたい。このような場合、エンドホストでは経路表やパケット操作ツールを用いるのが一般的である。しかし、これらを用いても、状況に応じてアプリケーション毎にネットワークやインターフェイスを切替えるなど、柔軟に IP パケットを操作することは困難である。

そこで、本稿では宛先ポート番号を元にインターフェイスを選択できるネットワーク制御システム、Port base Interface Selection を既存のシステムに拡張として実装した。本システムにより上記問題の解決を行いその有用性を示す。

1 はじめに

エンドホストのネットワーク接続の形態は単一接続から、複数接続へと変化した。例えば、ノート PC では、IEEE802.3 のみならず、802.11a/b/g といった無線デバイスから Bluetooth まで付属している。他にも、ホームルータ、携帯電話など、複数のネットワークインターフェイスを持つ機器が増えてきている。

エンドホストのネットワークの制御機構は、使用するネットワークが複数あった場合には経路表と宛先 IP アドレスによって選択される。エンドホストでは、多数の IP パケット操作ツール (以下パケットフィルタ) が混在していることから、宛先 IP アドレス以外に、ポート、アプリケーションやネットワークの特性等、柔軟な IP パケット操作がしたいという要求があることが伺える。このような要求を本稿ではユーザポリシーと呼ぶ。エンドホストでユーザポリシーを満たすためには、次のような方法を用いて実現するのが一般的である。

1) 宛先 IP アドレスを経路表に追加する。
2) パケットフィルタを用いて IP パケットのヘッダを書き換える。

1) の方法では、Dynamic Host Configuration Protocol(DHCP) や多くの Peer to Peer(P2P) ソフトの宛先 IP アドレスは動的であることが多い。これらの IP アドレスの変化を経路表に反映しつづけるのは困難である。

2) の方法では、パケットフィルタの設定記述能力や機能の限界により、すべてのユーザポリシーを柔軟に満たすことはできない。

今後、Internet Protocol Version 4(IPv4)、Internet Protocol Version 6(IPv6) の混在した環境において、同様の方法で解決し続けるのは困難である。

そこで本稿では、エンドホストでユーザポリシーを反映する場合の問題点を述べ、その問題点を解決し柔軟にユーザポリシーを反映するシステム、PISelect(Port base Interface Selection) を提案する。また、その設計とプロトタイプ実装についてまとめる。

2 ユーザポリシー

本章ではユーザポリシーについて述べる。Internet は宛先指向のルーティングであり、Internet Services Provider(ISP) やキャリアが持つバックボーンネットワーク上のルータ (コアルータ) は、Autonomous System(AS) や IP アドレスを元にネットワーク全体を考慮したルーティングがされている。

エンドホストではコアルータとは機能も環境も異なり、制御の考え方も異なる。多くの場合、エンドホストでは、ネットーワーク全体を考慮したルーティングをする必要はなく、エンドホスト環境のみを考慮してネットワークを制御できればよい。そして、宛先 IP アドレスだけではなく、ネットワークの性質やアプリケーションの特性など、エンドホスト環境を考慮したルーティングができるべきである。

ネットワークの性質やアプリケーションの特性を含めた IP アドレス以外の要素でルーティングを行うことは、現在の IP アドレスのみでルーティングしているシステムでは困難である。この問題の顕著な例はエンドホスト上でのマルチホーム [1] である。

例えば、図 1 のようなマルチホーム接続のエンドホストがある。A のネットワークは広帯域であるが遅延が大きく、B のネットワークは狭帯域ではあるが遅延が少ないとする。エンドホストでは、使用するアプリケーションや用途によって、利用するネットワークを指定できると効果的である。このように、コアルータ

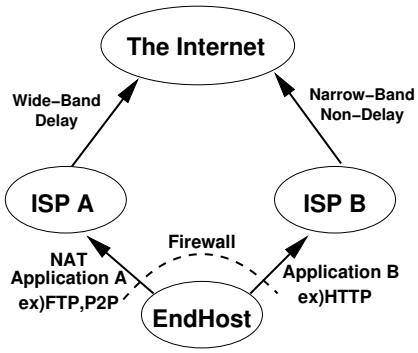

図 1: エンドホストでのマルチホーム接続の例

とは異なり、エンドホストではユーザ自身にとって都合の良いネットワーク制御が求められている。

3 エンドホストでのマルチホーム接続適応の問題点

前章で述べたユーザポリシーをマルチホーム環境のエンドホストに適用する上での問題点を述べる。

マルチホームとは一般的に複数のネットワークへの到達性があることを指す。現在のマルチホーム技術はコアルータを前提とした技術が多く、そのままではエンドホストへの対応が困難である。また、エンドホストでユーザポリシーを満たすために、経路表の追加やパケットフィルタの操作では、次のような問題と要求がある。

(1) 経路表の維持
(2) 柔軟なインターフェイスの切替え
(3) IP アドレスによるルーティングの限界
(4) パケットフィルタの静的動作

各問題点について述べる。

3.1 経路表の維持

バックボーンルータでは経路情報を Border Gateway Protocol(BGP) や Open Shortest Path First(OSPF) 等の経路制御プロトコルにより経路を動的に保持している。しかし、エンドホストに、これらの経路制御プロトコルが広告されることは極めて少ないため、動的に正常な経路を保持し続けることが困難である。また、多くのエンドホストの経路表にはデフォルトルートといくつかの静的経路しか入っておらず、これらでユーザポリシーを満たすことは困難である。

3.2 柔軟なインターフェイスの切替え

エンドホストでは、ネットワークの状態や特性を考慮して、柔軟にインターフェイスを選択したい。例えば、A、B 2 つのインターフェイスがあった時に、FTP は A のインターフェイス、HTTP は B のインターフェイスといった具合に選択したい。また、B のインターフェイスが障害の時には A のインターフェイスを使い、B のインターフェイスが復旧した時には、もう一度 B のインターフェイスを使用するといった要求が考えられる。

3.3 IP アドレスによるルーティングの限界

Internet 上のルーティングは宛先指向である。そのため、経路表でユーザポリシーをシステムに反映するためには、ユーザがアプリケーションに対する宛先 IP アドレスを常に周知しておかなければならない。IP アドレスには動的な IP アドレスと静的な IP アドレスとがあるが、どちらの場合においても問題は発生する。

- 宛先が動的な IP アドレスの場合

  宛先ホストが移動するようなモビリティー要素を持つ場合には、宛先 IP アドレスが同一かは保証されない。特に P2P のようなソフトウェアでは、接続する度に宛先 IP アドレスが変化する。
- 宛先が静的な IP アドレスの場合

  宛先 IP アドレスが固定である場合には経路表に経路を追加すれば制御できる。しかし、次々に加えていくと、経路表が繁雑になり見通しが悪くなる。また、インターフェイスの切替え時には、すべての経路表を書き換えなければならず、非常に手間が掛かり、非効率的である。

エンドホストでは動的、静的どちらの場合においても、常にユーザポリシーを満たした形で IP アドレスを経路表に反映するのは無理である。

また、宛先指向のルーティング以外にもソースルーティングがある。しかし、ソースルーティングは宛先まで各ルータがその技術をサポートしていなければならない。そのため、インターネットやネットワーク上の全てのマシンが他のマシンへの最良経路を知っている、または、見つけることができるという前提条件が必要であり、これらの 2 つの前提条件をデータ送信時に期待することはできない。

3.4 パケットフィルタの静的動作

エンドホストのマルチホーム接続では、経路表を利用する他にパケットフィルタを利用して、制御する方法もある。

代表的なパケットフィルタには bpf [3] や netfilter [4] があり、bpf は BSD 系の OS で使用されているパケットフィルタ、netfilter は Linux 系の OS で使用されているパケットフィルタである。両者は細かな部分で有する機能は異なるが、IP パケットを操作するという点において共通であるため、本稿では同一の技術として取り扱う。

パケットフィルタの問題点は、パケットフィルタの設定ファイルに記述された固定的な操作しかできないため、IP アドレスやインターフェイスの変化に対して瞬時に対応できない点である。ユーザポリシーを満たすためには、システムの変化に対応するような動作が必要である。

4 提案システムの設計と実装

3 章でマルチホーム接続のエンドホストにユーザポリシーを適応させる場合の問題点を述べた。本章ではこれら問題点を解決するために、ユーザポリシーを反映させる新たなネットワーク制御機構を設計実装する。

マルチホーム接続のエンドホストにユーザポリシーが反映できない根本的な原因は、宛先経路を決める経路表やその制御にユーザポリシーを追加できないことである。そこで、ユーザポリシーの追加による経路表の拡張を行う。まずユーザポリシーを構成する要素について述べ、次に経路表の拡張と実装について述べる。

4.1 ユーザポリシーの構成要素

本節ではユーザポリシーの要素について具体的に述べる。ユーザポリシーの構成要素に次の 3 点を挙げる。また、これら要素についての妥当性を検討する

- IP アドレス
- ポート番号
- アプリケーション特有の要素

IP アドレス　ユーザは宛先 IP アドレス、送信元 IP アドレスを元に送出するインターフェイスを選択することが考えられる。例えば、A、B 2 つのネットワークがあり、目的とする宛先 Z は A からは B よりも最適経路であることがあらかじめわかっている場合には、ユーザは明示的に B のネットワークへ接続しているインターフェイスではなく、A のネットワークへ接続しているインターフェイスから送出したい。

ポート番号　ポート番号も IP アドレスと同様の要求がある。ただし、ポート番号の場合は宛先ポートにより、アプリケーションが決まることが多いため、送信元ポートよりも宛先ポートの方が重要である。ユーザは宛先ポート番号を元に送出するインターフェイスを選択したい。

アプリケーションの特徴　ネットワークアプリケーションにはアプリケーション毎に特徴がある。例えば、短時間で多くのデータを送受信するアプリケーションや、連続的に通信を行いリアルタイム性を要求するアプリケーションまで様々である。ユーザはこのような特徴あるアプリケーションをその特徴にあわせたネットワークで使用したい。

4.2 システムの要求仕様

3 章で述べた問題点と 4.1 節で述べたユーザポリシーの要素からユーザポリシーを反映するための新たなネットワーク制御機構に必要な要求事項をまとめる。

a) IP アドレスによる IP パケット操作
b) ポート番号による IP パケット操作
c) アプリケーション毎の制御
d) インターフェイスの選択
e) 外部変化によるシステムの切替え

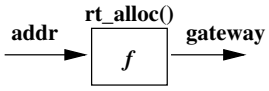

図 2: 基本形 rt_alloc() $f(\text{addr})$

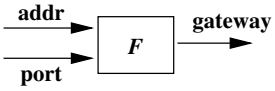

図 3: 引数の追加拡張 $F(\text{addr}, \text{port})$

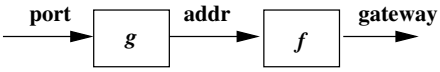

図 4: 段数の追加拡張 $f(g(\text{port}))$

これらの要求事項のうち a) と e) は経路表により実現されている。残りの b)、c)、d) の機能を PISelect で追加拡張させなければならない。

ここで、b) と c) の関係について着目する。ネットワークアプリケーションで使用されているいくつかのポート番号は、Well-known ポートと呼ばれ OS で予約されており、例えば HTTP は 80 番ポート、ssh は 22 番ポートが使用されている。これらポート番号とプロトコルの組合せは必ずしも一対一ではないが、多くの OS において、この関係が無視されていることは稀れである。そのため、多くの場合、ポート番号が決まれば、使用するアプリケーションが決まると考えてよい。そこで本稿では、アプリケーションとポート番号は一対一であるという前提条件で設計をする。

4.3 経路表の拡張

要求事項の残り 3 つ [b)、c)、d)] を PISelect で OS に追加する。現在のルーティングモジュールは宛先アドレスが決まると送出するインターフェイス、又はゲートウェイが決まる (図 2)。新たにユーザポリシーをシステムに反映するために次の 2 つの手法が考えられる。

i 引数の追加拡張

ポート番号と IP アドレスの組合せからインターフェイスを選択する新たなルーティングモジュール F を作成する (図 3)。

ii 段数の追加拡張

ポート番号を入力に IP アドレスを出力するモジュール g を作成する。関数 g の結果をルーティングモジュールに入力値として渡す (図 4)。

すでに手法 i は、Windeos2000/XP において実装されており、metric というパラメータを付加させて経路表を拡張している。metric は経路に対する信頼性の高さやコストを算出するのに使われる値で、3.2 節で述べた柔軟なインターフェイスの切替えや経路表内に同一

表 1: ポートテーブル

	ポート	Alias アドレス
1	80	192.168.0.1
2	25	172.16.0.1
3	25	192.168.0.1

表 2: Alias アドレス

インターフェイス	Alias アドレス
eth1	192.168.0.1
eth2	172.16.0.1

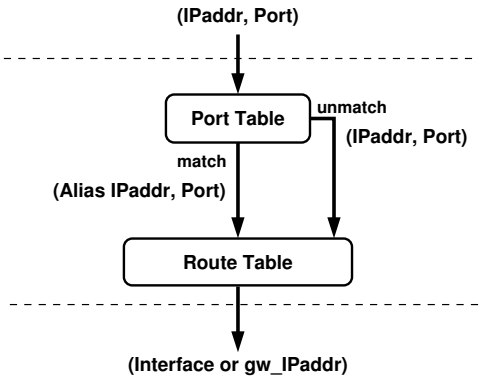

図 5: システムの流れ

宛先、異なるインターフェイスがあった場合に効果を発揮する。

そこで、PISelect では、このうち手法 ii を選択した。この手法を選択した理由は他の手法 i に比べ、1) 今までのルーティングモジュールがそのまま使えて後方互換が優れている、2) ポート番号毎に addr 入力が変更可能なため、ポート毎に経路が操作できる。という 2 点が上げられる。

4.4 プロトタイプの実装

本提案システムを、多くのアーキテクチャで動作し広く使われている NetBSD-1.6.2RELEASE [6] に拡張として実装し IPv4 を利用して動作検証を行った。

PISelect ではインターフェイスセレクションを行うためにポートテーブルを設けた。ポートテーブルはポート番号と IP アドレスで構成する表である。PISelect では、まず、インターフェイスに Alias アドレスをつける。この Alias アドレスは、任意のアドレスで、loopback アドレスでもプライベート IP アドレスでもよい。そして、ポート番号と Alias アドレスからポートテーブルを作成する。インターフェイス eth1 に Alias アドレス 192.168.0.1、eth2 に Alias アドレス 172.16.0.1 をつける。この時、ポートテーブルの例を表 1、Alias アドレスを表 2 に示す。ポートテーブルはポート番号と Alias アドレスの組合せであるが、Alias アドレスはインターフェイスを特定するためだけにあるので、ポートテーブルは Alias アドレスを介して、インターフェイスを導出することとなる。

次にシステムの制御の流れを図 5 に示す。上位レイヤから入力されたデータは、まず、ポリシーチェックにかけられる。ポリシーチェックはポートテーブルに一致するようなルールが存在しているか検索する。上記の表 1、表 2 を用いて説明すると、宛先ポート番号 80 の IP パケットが到着すると、そのルールはポートテーブルに存在するため、下側の流れ (match) になる。逆にルールに存在しない宛先ポート番号 22 の IP パケットが到着すると右側 (unmatch) の流れになる。また、宛先ポート番号 25 に関するルールは 2 つ存在している。この時のシステムはまず、172.16.0.1 の Alias アドレスが付いたインターフェイス eth2 へ送出を試みる。しかし、eth2 のインターフェイスの障害や Alias アドレスが存在していない等で eth2 へ出力できない場合は、192.168.0.1 の Alias アドレスの付いたインターフェイス eth1 へと出力を試みる。それでも出力できない場合は、IP パケットヘッダ等を変更せずに経路表からデフォルトルートへと IP パケットは出力される。

5 評価

3 つの既存システムと提案システムの比較を行う。

A: 既存システム 1 (経路表のみ)
B: 既存システム 2 (経路表の拡張)
C: 既存システム 3 (経路表とパケットフィルタの併用)
D: PISelect (経路表とポートテーブルの併用)

比較項目は次の通りである。

- パケット操作の粒度
- システムの外部変化に対する追従
- 送出インターフェイスの選択
- ユーザポリシーの反映度

まず、パケット操作の粒度について述べる。既存システム 1 では経路表による宛先 IP アドレスのみでしか操作できず操作粒度は粗い。既存システム 2 も metric の付加により、異なるパラメータでの指標ができる以外は既存システム 1 と同様である。既存システム 3 ではパケットフィルタの性能にも左右されるが、多くのパケットフィルタが有している機能、宛先 IP アドレス、送信元 IP アドレス、宛先ポート、送信元ポートの操作を含め、パケットフィルタによってはそれ以上の操作が可能であり、ユーザの好みに応じた粒度で操作できる。PISelect では IP アドレスに加え、ポート番号ごとに操作テーブルを持っているため、既存システム 1 よりも細かな操作ができるが、既存システム 3 まで細かくできず、操作粒度はやや粗い。

次にシステムの外部変化に対する追従について述べる。既存システム 1 と既存システム 2、PISelect のどの

表 3: 既存システムと PISelect の機能比較

	既存システム 1	既存システム 2	既存システム 3	PISelect
	経路表のみ	経路表の拡張	経路表とパケットフィルタの併用	経路表とポートテーブルの併用
IP パケット操作の粒度	粗い	粗い	細かい	やや粗い
システムの外部変化に対する追従	有	有	無	有
送出インターフェイスの選択	不可	限定的に可能	不可	可能
ユーザポリシーへの反映性	低い	少し高い	やや高い	高い

システムも経路表を参照しているため、システムの変化に対して追従性がある。特に既存システム 2 の実装の Windows2000/XP では、同一経路でも、metric 毎に経路が持てる。通常は metric の低いインターフェイスから選択されるが、metric の低いインターフェイスが障害等により切断された場合には、経路表からこのインターフェイスに関する経路が削除され、自動的に次の metric の低いインターフェイスへと切替えられる。これは、インターフェイスの状態を検知して、経路を自動的に切替えるため、e) を保持したまま d) の機能を付加させる点において優れている。一方、既存システム 3 では経路表を参照しているが、経路表の参照後に、パケットフィルタによって静的に書き換えられているためシステムの変化に対して追従できない。

また、インターフェイスの選択可能/不可能では、既存システム 1 は送出するネットワークは選択できるが、インターフェイスを選択できない。そのため、経路表に同一宛先、異なるインターフェイスがある場合には、最長一致でも 2 つの経路が一致してしまう。多くの場合、最初の一致が選択されるため、意図していないインターフェイスから送出される可能性がある。既存システム 2 の実装を持つ Wintows2000/XP の経路表では、metric 値により、この問題が解決されている。同一宛先、異なるインターフェイスが存在した場合には、metric の低いルートが選択される。この同一宛先、異なるネットワークの時のみインターフェイスの選択が可能となる。しかし、同一宛先、異なるインターフェイスの経路は、インターフェイスの metric を変えない限り、毎回、同じインターフェイスが選択される欠点がある。そのため、自動切替えは可能だが、インターフェイスの選択はできない。また、経路表の拡張だけでは IP アドレスベースなため、アプリケーション毎の切替えができない。既存システム 3 では、パケットフィルタの多くは、設定したルールを適応させるインターフェイスを選択することはできるが、送出するインターフェイスの選択はできない。PISelect では、ポートテーブルによって、送出インターフェイスの選択が可能となる。

さらに、ユーザポリシーの反映度に関して、上記 3 項目も含め、要求仕様の項目を満たしているかを比較する。既存システム 1 は b)、c)、d) を満たさず、ユーザポリシーは反映されにくい。既存システム 2 の実装例である Windows2000/XP では既存システム 1 と比べ d) が限定的ながら可能という点以外は同じとなる。既存システム 3 ではパケットフィルタの持っている機能にもよるが多くの場合 d)、e) は満たしていない。PISelect では要求仕様であげた項目を満たしており、ユーザポリシーの反映度は高い。

これまで述べた項目を表 3 にまとめる。

6 今後の展望

本稿では経路表の拡張として、1) 経路表の入力パラメータの拡張。2) 多段を増やして、複数のテーブルによる拡張。2 つの拡張を述べ、既存のシステムを含めた検討を行った。また、PISelect においては、アプリケーションとポート番号を一致するものという前提で設計実装した。しかし、ネットワークアプリケーションには、ポートとプロトコルの関係が一意でないアプリケーションは存在する。この問題と同様に、Stream Control Transmission Protocol(SCTP) や FTP、P2P など、複数の IP アドレスや複数のポートを使用するアプリケーション、動的に IP アドレスやポートが変化するようなアプリケーションについても考えなければならない。この問題を解決するためには、OS におけるアプリケーションを定義する必要があり、プロセス、プロトコル、ファイルディスクリプタ等も考慮した管理する必要がある。そのため、IP アドレスベースのルーティングを拡張するだけでは、解決できないことが伺える。

また、エンドホストにおける現在の経路表の役割を検討する。Internet の当初のネットーワークのモデルでは、コアルータや Internet Exchenge(IX) は存在せず、各計算機が宛先を経路表から導出する方式であった。しかし、現在のネットワークモデルは、接続するネットワーク毎にバックボーンやコアルータ等の役割が分担されており、エンドホストの経路表にはデフォルトルートと幾つかの静的経路しか入っていない。そのため、計算機から出力される IP パケットのほとんどはデフォルトルート宛に送出され、経路表自体は当初のような重要な意味を持っていない。このことから、エンドホストでは経路表を利用すること自体が非効率である。

一方、パケットフィルタは、上記であげたような処理が可能なように思われるが、現在のパケットフィルタの設計は IP パケットベースで処理されるため、動的に変化する IP アドレスやポート等をアプリケーションベースで制御するのは困難である。

そこで著者らは、経路表に頼らないエンドホストの環境に特化したアプリケーションベースの制御方式を現在検討中である。

7 まとめ

本稿では、エンドホストでユーザポリシーを反映する場合に、マルチホーム接続を具体的な適応例として考え、その場合に問題となる点をまとめた。そして、その解決手法として宛先ポートによりインターフェイスを選択する PISelect を提案し設計実装をした。PISelect により、エンドホストでは宛先ポートに応じて、インターフェイスを選択することが可能となる。既存のシステムと PISelect の提供する機能を比べ、エンドホストにおける PISelect の有用性が示せた。結果と検討からエンドホストにおいてユーザポリシーを満たすためには、複数のアドレスやポートを使うような複雑なアプリケーションやプロセス、ファイルディスクリプタといったアプリケーション毎の制御が必要である。

参考文献

[1] T.Bates, and Y.Rekhter, RFC 2260:Scalable Support for Multi-homed Multi-provider Connectivity, IETF, January 1998.

[2] J.Hagino, and H. Snyder, RFC3178: IPv6 Multihoming Support at Site Exit Routers, IETF, October 2001.

[3] BSD. bpf(4): NetBSD Programer's Manual.Steven McCanne, of Lawrence Berkeley Laboratory, implemented BPF in Summer 1990.

[4] netfilters Home page, http://www.netfilter.org/

[5] IP Filter Home page , http://coombs.anu.edu.au/avalon/

[6] NetBSD Home page, http://www.netbsd.org/

[7] W.Richard Stevens and Wright Gary R, Tcp/Ip Illustrated Volume2:The Implementation, Addison-Wesley, January 1995

[8] Implementing a Clonable Network Stack in the FreeBSD Kernel. Marko zec.Appeared in Proceedings of the 2003 USENIX Annual Technical Conference.

[9] Daniel Robbins. Dynamic iptables firewalls, http://www-106.ibm.com/developerworks/linux/library/l-fw/?n-l-4191

[10] dyfw(Dynamic Firewall Tools), Gentoo Linux Inc, http://www.gentoo.org/proj/en/dynfw.xml

[11] Thomas Graf, Greg Maxwell , Remco van Mook (Virtu Secure Webservices)Martijn van Oosterhout, Paul B Schroeder, Jasper Spaans, Pedro Larroy:Linux Advanced Routing & Traffic Control, http://lartc.org/