WIDE Technical-Report in 2007

USAGI プロジェクト
送信元アドレスに基づく経路選択の設計
wide-tr-usagi-polroute-00.pdf

WIDE Project : http://www.wide.ad.jp/

If you have any comments on this document, please contact to ad@wide.ad.jp

Title: USAGI プロジェクト 送信元アドレスに基づく経路選択の設計
Author(s): USAGI プロジェクトコアメンバ (usagi-core@linux-ipv6.org)
Date: 2007/01/01
X-Id: $Id: wide-tr-usagi-polroute.txt,v 1.2 2007/01/05 10:37:16 yoshfuji Exp $

■ポリシー・ルーティングとその Linux 実装

インターネット上で次ホップ(Next Hop)の選択を行う経路選択(Routing)は、宛先アドレス(および、あまり使われていないが、サービス型(Type of Service; ToS)フィールドに基づいて行われることを原則としている。しかし、場合によっては、それ以外の要素、たとえば、パケットの送信元アドレスなどによって特定の経路を選択したいことがあり、ポリシー・ルーティング(Policy Routing)と呼ばれる。

Linux におけるポリシー・ルーティングは、IPv4 に関してはかなり古くから存在しており、ip rule コマンドにより優先度付で設定されるルーティング・ポリシー・データベース(Routing Policy Database; RPDB)に、アクション並びに経路表を関連づける方法で実装されている(図1)。

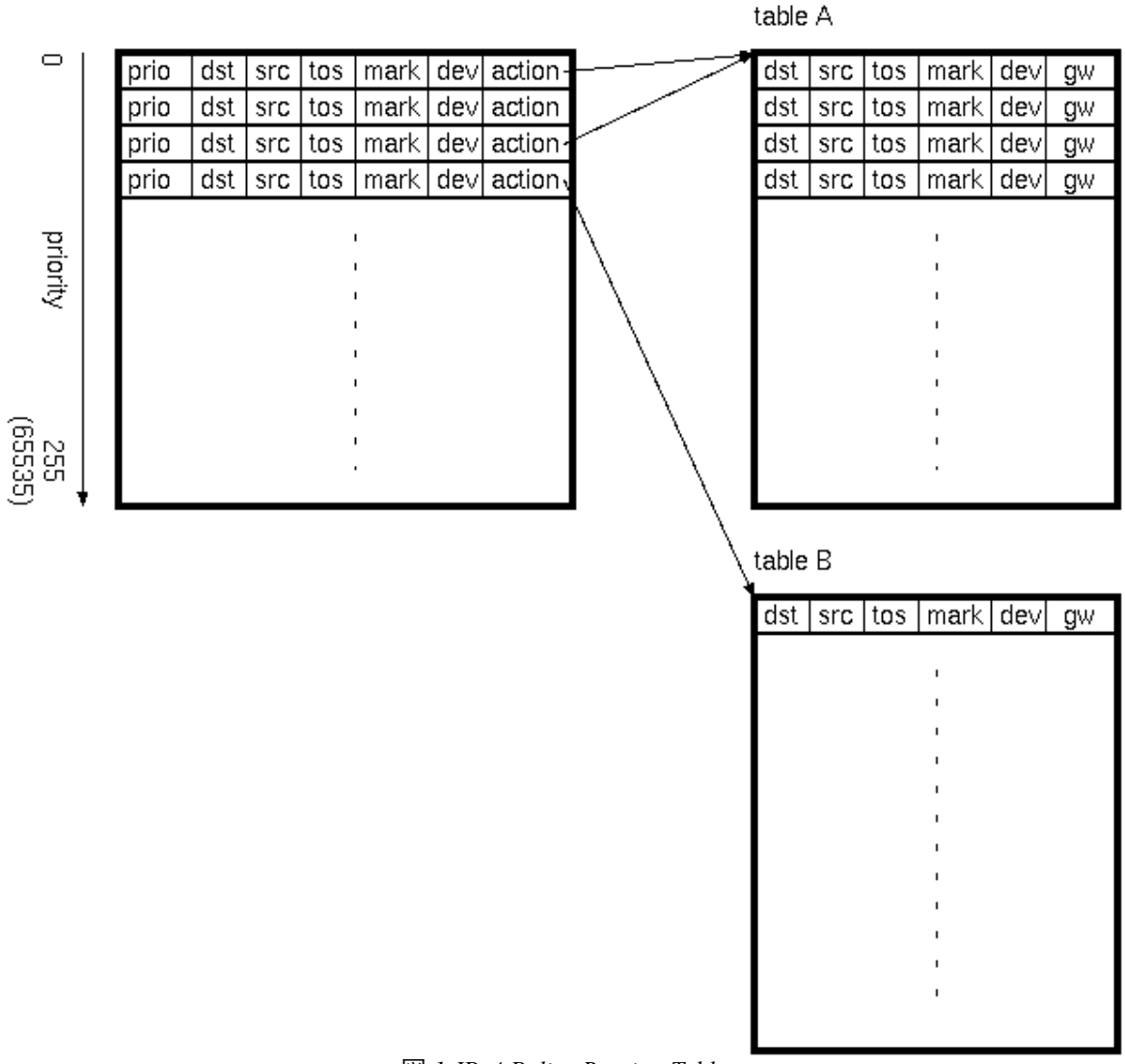

図 *1:IPv4 Policy Routing Table*

個々の経路ルールに設定できる項目には次のものがある:
- 送信元アドレス(from)
- 宛先アドレス(to)
- サービスタイプ(tos)
- Netfilter モジュールなどによるマーキング(mark)
- 入力デバイス(iif)

この概念そのものは IPv4 に特化したものではなく、IPv6 にも援用できるものであるが、実際には、機能の断片だけが実装された状態で長い時間が経過していた。しかし、Linux における Mobile IPv6 機能がこの機能を利用することから、ポリシー・ルーティング機能の整理と汎用的実装が重要な課題となった。

■パケット処理における送信元アドレス選択と経路選択の相互作用

Mobile IPv6 における移動ノード(Mobile Node; MN)がポリシー・ルーティングを活用することを例として、IPv6 のポリシー・ルーティングは、ホストでの利用も積極的にされることが想定される。この際、問題となるのが、ポリシー・ルーティングを行うノード自らが送信を行うパケットである所謂自発パケットに関係する、送信元アドレス選択と経路選択との相互作用である。

送信元アドレスは、アプリケーションなどの上位層から指定されないことがしばしばであり、その際には、カーネルが適切なものを選択する。IPv6 におけるその方法は、RFC3484 Default Address Selection for Internet Protocol version 6 (IPv6)として規定されている(注: IPv4 でも同様の議論は可能であるが、同文書の範囲外である。)。この中で、経路選択との相互作用に関しては、経路選択、特に、出力インタフェース選択が、送信元アドレス選択に先だって行われることを前提としているとされており、実際、従来の Linux IPv6 実装でも、送信元アドレスの選択は、経路選択が完了したのちに行われるように実装されている。

一方、ポリシー・ルーティングでは、検索すべき経路表の選択基準である RPDB の検索において、宛先アドレスに加え、送信元アドレスが考慮の対象となる。ここで、Mobile IPv6 では、送信元アドレスとしてはホーム・アドレス(Home Address; HoA)を優先し、また、Linux 実装では HoA を送信元アドレスとするパケットを特別扱いしてトンネル経由でホーム・エージェント(Home Agent; HA)に送るためにこの機能を用いている。しかし、従来の手順では、経路選択を行う段階では、送信元アドレスは未だ選択されておらず、上位層から指定されていない限り、未指定アドレスのままである。アプリケーションが、パケットの送信時に送信元アドレスを適切に選択していることは期待できないため、カーネルは未指定な送信元アドレスに関しても適切に HoA を選択し、それに基づいて経路を選択するように構成する必要がある。

■Linux における IPv6 経路表の実装と送信元アドレスに基づく経路選択の設計

Linux における IPv6 経路表の実装は図 2 のような構成である(注: 点線は、経路表を検索する際にメモリを参照する流れの概略を示す)。経路表は宛先アドレス用と送信元アドレス用の 2 段 Radish Tree で構成されており、rt6_info 構造体のリスト構造により、同一な宛先アドレスと送信元アドレスの組で複数の異なる経路を保持することができる。

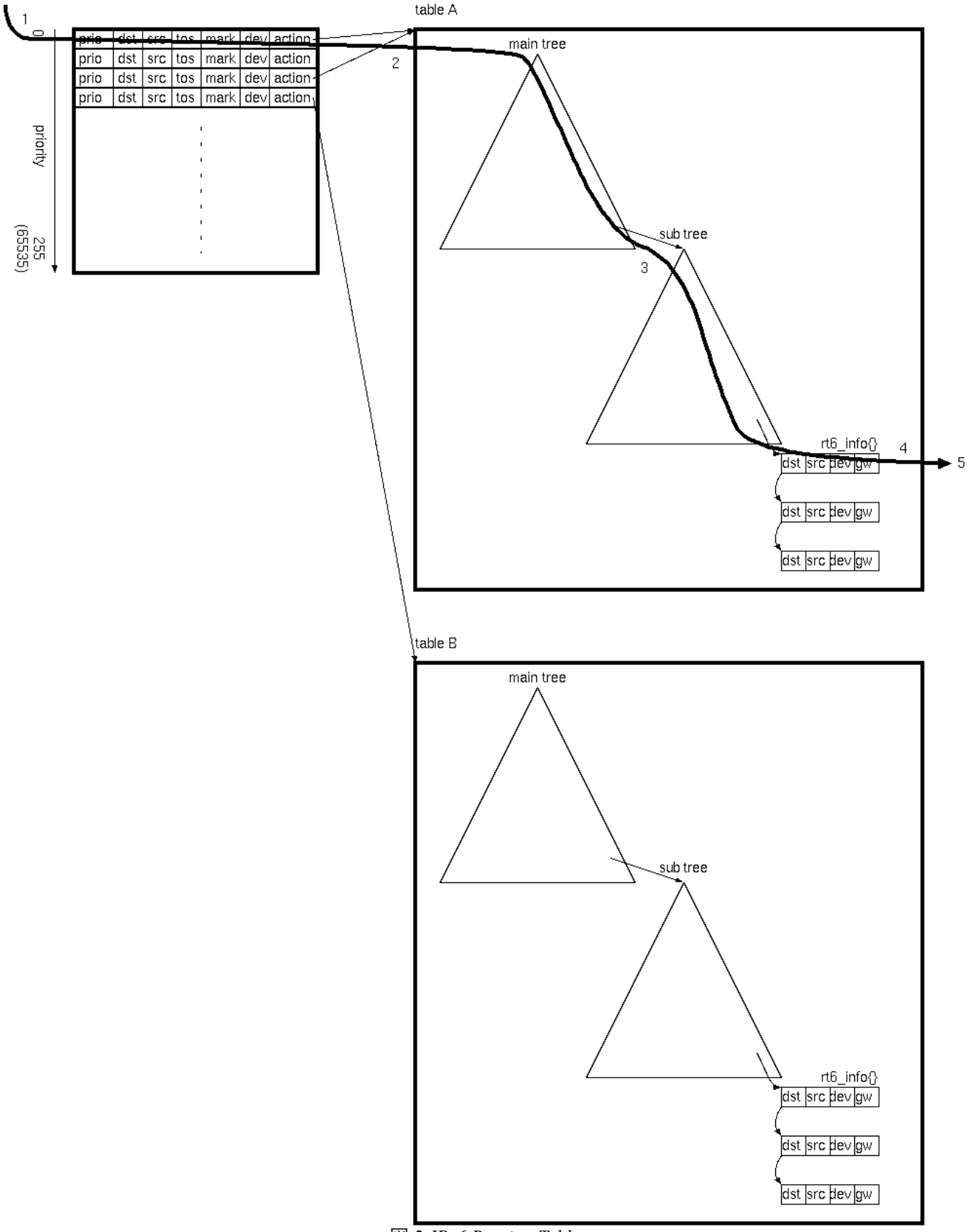

図 *2:IPv6 Routing Table*

上位層によって指定されていない場合における、送信元アドレスの選択については、現時点で、図の(1)-(5)の候補がある。

(1)RPDB を検索する前
(2)宛先アドレスに対応する Main Tree を検索する前
(3)宛先アドレスが検索された後、送信元アドレスに対応する Sub Tree を検索する前
(4)宛先及び送信元アドレスの検索を終了し、対応する複数の経路の比較ごと
(5)経路選択が完了した後

これらの組み合わせにより、いくつかの方式が検討された。
(a)(1): RPDB の前で行う方法。
　この場合、送信元アドレス選択は出力インタフェース選択の結果を反映できない。
(b)(2)及び(5): RPDB 検索において、個々のルールのアクションが経路表を検索するものである場合に限って、一旦送信元アドレスを無視した上、(2)において、ルールに送信元アドレスが含まれていればそのルールに制限された送信元アドレス選択を行い、そうでなければ、(5)において送信元アドレスを補完し、最終的に RPDB との整合性を確認する方法。
　(a)と同様、送信元アドレス選択は出力インタフェース選択の結果を反映できない。
(c)(3)及び(5): (b)と同様だが、(2)のかわりに(3)において、Sub Tree が存在する場合を追加的条件とするもの。
　送信元アドレスが RPDB や経路に関係していなければ要求を満たすことができるが、そうでない場合は、選択の順序が規定と合致しておらず、出力インタフェースの選択の結果を反映できない。
(d)(3)および(4): (c)と同様だが、(3)において全ての関連する出力インタフェースの可能性を網羅して検索し、もっとも送信元アドレスとの一致長が長く、送信元アドレスの評価の高いものを選択する方式。
　送信元アドレスが RPDB や経路に関係していなくても要求を満たすことができる可能性が高いが、実装がかなり複雑で実装速度が遅くなる。また、複数の選択肢があった場合にどう選択するかも課題である。
(e)(5): RPDB 選択時、未指定の送信元アドレスは、送信元アドレスに関連しない RPDB エントリのみを有効として扱う方法。
　結果として出力されるパケットにおいて、送信元アドレスが RPDP のルールと、一見、矛盾する可能性がある。この方法で Mobile IPv6 実装が動作可能か検討する余地がある。
(f)その他

なお、RPDB 検索で一旦送信元アドレスを無視する場合を、個々のルールのアクションが経路表を検索するものである場合に限る理由は、そもそも、経路選択とはパケットを終点にまで正しく届けることが目的であるから、アクションとしてパケットの破棄を指定することは必ずしも正統的な方法でなく、Netfilter を用いるべきであるが、もしパケットの破棄が指定される場合にも送信元アドレスを無視すると、送信元アドレスが指定されていないとほぼ常にそのルールに合致してしまい、事実上パケットの送信が困難になるためである。

■今後の展開

前章で掲げたいずれの方式も少なからず問題があると予想されているが、未だ絶対的な結論には達していない。今後は、いくつかの実装を通じてさらなる検討を行う予定である。