第 7 部

ポリシールーティング

第 1 章

はじめに

インターネットは年々拡大を続け、初期の ARPA-NET と地域ネットワークという構成から、複数のネットワークの相互接続という形に変化した。また、最近ではネットワークの商用利用も盛んになり、各組織が複数のプロバイダに接続することも増えてきた。相互接続した各々のネットワークは単一な組織や均一な技術によって作られたものではないため、使用目的や、遅延、バンド幅、課金方法など、さまざまな点で違いがある。このような環境では、発信者から宛先への経路が複数存在しうるが、従来の hop-by-hop ルーティングでは、あるコストについて最適な経路を選択するだけであった。複数のプロバイダと接続している場合でも、双方の自由な使い分けについてはかなり困難であった。

これに対し、使用し得る経路が複数あれば、ユーザの都合のよい経路を選択できるようにしたい。そのための経路制御手法が最近注目されてきている政策的経路制御 (Policy Routing) [51] である。それは、従来のようにひとつのコストついての最適な経路を計算するものではなく、各ノードの異なる主張や方針を考慮に入れて経路を決定するものである。

昨年度の報告書では、政策的経路制御で要求されているポリシの具体的な事例を示し、次に、網の政策、通信の始点、終点の意思という三種類の要求によるモデル化と、そのモデルに基づいたいくつかの通信の始点、終点の意思についての事例を示し、ポリシのいくつかを解決する経路制御手法として終端サイトの意思による経路制御 (Routing by Preference) を提案した。次に IDPR や SDRP などの IETF で研究されている経路制御プロトコルの紹介を行なった。

その後、研究には特に進展が見られなかったため、今年度の報告書では、IETF での研究の紹介をメインに行なう。第二章にて SDRP/ERP, Nimrod について紹介し、第三章にて、これからの研究の方向性について考える。

第 2 章

関連プロトコル

2.1 SDR(Source Demand Routing)

SDR の目的は BGP/IDRP で提供されたノード選択を補い、パケット発信者による経路選択を支援することである。

SDR は 4 つの要素からなる。

- プロトコル制御メッセージとユーザデータグラムのカプセル化のパケットフォーマット
- ユーザのデータと制御メッセージの処理と転送
- ルーティング情報の配布と収集
- 設定と使用

現状では、経路情報の収集については BGP/IDRP から得ることが決まっているほかは、具体的な方法そのものは曖昧であり、管理者が与えておくというイメージである。将来的には自動的な手法も提案されるだろう。

以下、IETF SDR WG の活動について、昨年度の報告書からの進展を紹介する。

1994 年夏の IETF にて IPng が決まったことによる、IPv6 上での SDR についてのドキュメントが公開された。

最近は研究の方向が Multicast や IPv6 に向かっているようである。

Multicast の研究グループでは、SDRP を経路指定できるトンネリングプロトコルとして使用しつつあるようである。

2.1.1 draft-ietf-sdr-sdrp-05.txt

draft-ietf-sdr-sdrp-05.txt[52] は SDR のパケットフォーマット、パケットフォーワーディングについてかかれたドキュメントである。昨年度の報告書での調査と比べると若干拡張され、完成度が上がっている。

ほぼ内容の packet-forwarding-spec-101.txt という draft が WG 内で公開されている。

2.1.2 draft-ietf-sdr-route-construction-01.txt

draft-ietf-sdr-route-construction-01.txt [53] はルーティング情報の配布と収集について書かれているドラフトである。経路情報の収集のアイデアと、それを使ってプロバイダ選択に応用するアイデア、そして IDRP と整合させる場合の問題点について書かれている。

経路情報収集の方法としては以下の二つが考えられている。

- RIB Query
  IDRP Speaker に対して Query を行ない、部分経路を得る
- Path Explorer
  IDRP の中間ノードでの、要求があり次第開始される経路計算で、ソースの preference を満たすような探索を行なう。

2.1.3 draft-ietf-sdr-erp-01.txt

draft-ietf-sdr-erp-01.txt[54] では、SDRP を IPv6 上にのせ、拡張した Explicit Routing Protocol(ERP) を提案している。1994 年 10 月に最初に公開され、1995 年 1 月に更新されている。以下、ERP について簡単に説明する。

従来の hop-by-hop ルーティングでは実現できない様々なルーティング要求がある。そのうち、ERP は、IPv6 のルーティングヘッダにより、発信元やパケットの経路上のいくつかのルータにてそこからのパスを細かく指定する機構である。

従来のソースルーティングとの違いは、従来のソースルーティングでは発信者が経路を指定し、受信者に届くまでソースルーティングヘッダがついたままであるが、ERP では、ERP ヘッダはスタッカブルであり、経路上のどのルータでも ERP ヘッダを付加して経路を指定でき、また経路指定部分が終ったら ERP ヘッダを削除する。経路上のルータで、ある行き先のパケットについては指定した経路を経由するように ERP ヘッダを付加するような経路制御を行うことができる。

詳細については SDRP とよく似ているが、特に拡張されている一点のみ説明する。

長期間にわたって小さいパケットをやりとりするような場合、全データパケットに全ソースルートをのせるような実装ではトラフィック的に非常に不利になる。それに対し、ERP では最初に Route Setup を行ない、経路上の ERP ルータにはソースアドレスとフロー毎にユニークな Flow ID をキーとしてソースルートを持たせ、各パケットには Flow ID のみ持たせるような仕組みも組み込まれている。この場合、ルータのステートテーブルの大きさには限界があるので、Route Setup が失敗した場合は従来と同じようにパケットにソースルート (経路) をのせることになる。

ドラフトに記載されている用途としては、Provider selection, 移動体、仮想ネットワーク、PIM などが示してある。

2.2 スケール問題に適応性のある経路制御アーキテクチャNimrod

この節では、IETF の Nimrod ワーキンググループの目的や活動予定を紹介し、draft の概要を紹介する。Nimrod アーキテクチャの詳細については、付録の日本語訳を参照していただきたい。

以下、WG のチャーターをもとにまとめる。

2.2.1 IETF Nimrod ワーキンググループ

Nimrod ワーキンググループの目的は、非常に巨大なインターネットに適応する柔軟な新しい経路制御とアドレスアーキテクチャを設計し、仕様化し、インプリメントし、テストすることである。

そのアーキテクチャは、ドメイン内、ドメイン間を含んだ経路制御全般に対する一つの均質な骨組みを提供する。ポリシー要求への考慮だけではなく、名前付けの抽象的な階層、ネットワーク接続点 (NAP) からの出発、実際の接続性を含む。こういった新しい名前付けは様々に抽象化できる可変長なものになる。

全体の設計とアルゴリズムがどのようなインターネットワークプロトコルファミリでの使用に適しているとしても、最初の詳細なプロトコル設計とインプリメンテーションはいまのところ IPv4 で開発することになっている。しかし、別のパケットフォーマットへのサポートに変更するかもしれないし、別のパケットフォーマットへのサポートを追加するかもしれない。現在の変更が加えられていない IPv4 ホストとの相互運用性は IPv4 のソース、デスティネーションアドレスを終端識別子として再評価することで達成されるだろう。

Nimrod アーキテクチャを設計するときには、移動体、マルチキャスト、資源割り当てへのサポートという点に実質的な成果がつくられるだろう。

ワーキンググループは以下のような予定で活動する。

Jun 94 経路制御とアドレスアーキテクチャの基本の批評と議論を完了する

Feb 94 Nimrod の詳しい紹介を含むドキュメントを作成する

Jul 94 基本の経路制御とアドレスアーキテクチャとプロトコルの設計を含む Internet Draft を発行する

Sep 94 完全な基本経路制御とアドレスアーキテクチャを表現する最初のバージョンのプロトコル仕様を作成し、Internet-Draft して発行する

Nov 94 核心の仕様に含まれない部分の見本アルゴリズムを含む詳細な手法の設計を終え、Internet-Draft として発行する

Dec 94 Internet-Draft としてクラスタリング戦略と設定について述べた利用ガイドを発行する

Jul 95 インターネットでフィールド実験ができるような、最初のプロトタイププロトコルの実装を終える

Oct 95 プロトタイププロトコルと実装の最初のフィールド実験を終える

Jul 96 運用実験に基づくプロトコルと見本アルゴリズムのパフォーマンスの評価のあと、プロトコル仕様と見本アルゴリズムの改訂版を発表する

2.2.2 draft-ietf-nimrod-routing-arch-00.txt

draft-ietf-nimrod-routing-arch-00.txt[55] では、現在の経路制御に関する問題点をあげ、Nimrod の目標や特徴を述べている。

Nimrod の目的は、次に述べる通りである:

1. internetwork 全体にわたって知られていなければならない経路指定情報の量を制御するメカニズムを提供して、任意のサイズの動的 internetwork に対応すること
2. サービス提供者と人とユーザにより求められる複数の制約の中で、サービスに対して差別化を施した経路制御を提供すること
3. internetwork の加速度的な拡大に対応できること

つまり、スケーラブルでありながら、ユーザのサービスに対する要求にも答えるというなんでもできる経路制御プロトコルである。我々が昨年度に研究していた拡張 Routing by Preference に近いが、考え方としてはより実用的である。

Nimrod で提供された基本経路選択機能は、通常の経路制御システムと同じである。

1. ルート発生と選択に必要な情報を、集め、整理し、そして分散させる。
2. この情報に基づくルートを生成して、そして選択する。
3. ルータ内でパケットを選択されたルートに沿って進めることに必要な情報を確立する。
4. 選択されたルートに沿ってパケットを進める。

Nimrod では集合化 (clustering) を行なって経路制御における要素の数を減らし、抽象化によって実体に関する情報量を減らし、internetwork を複数のレベルでの抽出の実体の集合として表現することができる。また、情報配送を限定したり、情報をキャッシュしたり、情報を進めることを制限することで情報量を全体的に減らし、スケール問題に対して適応性をつける。

2.2.3 draft-ietf-nimrod-mobility-01.txt

draft-ietf-nimrod-mobility-01.txt[56] は、Nimrod での移動体について議論したドキュメントである。Nimrod は移動体のサポートについて解決する必要があり、インターネットワークでの既存の移動体サポートについて利点と欠点を分類し、比較した。そして、Mobile-IP プロトコルと名付けられる IETF で開発されている案を利用する Nimrod での移動体サポートについて概要する。

2.2.4 draft-ietf-nimrod-multicast-01.txt

draft-ietf-nimrod-multicast-01.txt[57] は Nimrod でのマルチキャストについて議論したドキュメントである。Nimrod ではマルチキャストのサポートについて解決すべき問題点が残されており、インターネットワークでの既存のマルチキャストサポートについて利点と欠点を分類し、比較した。そして、Protocol Independent Multicast (PIM) プロトコルと呼ばれる IETF で現在開発されている案を利用する Nimrod でのマルチキャスト対応について概要する。

第 3 章

まとめ

政策的経路制御として考えないといけない基本的な要素には、中継網の政策とセッションの利用者の要求がある。従来の hop-by-hop ルーティングでは、そのうちの中継網の政策を実現するためにネットワークトポロジと経路制御プロトコル (BGP,IDRP,RIP) 等を用いる。また、利用者の要求をかなえる点に重点をおく研究テーマとしてリソースリザベーションと QoS(Quority of Service) があり、両者を考えた例として IDPR や拡張 Routing by Preference、Nimrod がある。

本ワーキンググループでは、最初は hop-by-hop ルーティングでの経路制御について議論していたが、その後、政策的経路制御として中継網の政策とユーザの要求の両方を満たすもののみを考えるようになってきた。

IDPR にはドメインレベルでの操作を行なうことになっていて一般ユーザからみるとあまりうれしくないとか、実装が重くて使われなかったという問題がある。拡張 Routing by preference の場合はいろいろと細かいことを行なおうと考えていたが、経路制御情報での量や計算量を減らすことをあまり考えていなかったため、行きづまってしまった。Nimrod では、ドラフトにどうしても動かすという決意みたいなものがみえていて、まず経路情報を減らすところから書きはじめられている。

今後の研究としては、Nimrod の研究を追い、Routing by Preference のときの知識などをあわせながらそれ以上のものをつくっていくような方向と、ATM などの QoS パラメータを実装できるメディアを使い、リソースリザベーションを含めた経路制御を実装してみるような方向が考えられる。

第 4 章

付録

以下に Nimrod の Internet Draft(draft-ietf-nimrod-routing-arch-00) の粗訳をのせる。

4.1 Nimrod

Nimrod はスケール問題に対して適応性のある経路制御アーキテクチャである。Nimrod の目的は、次に述べる通りである:

1. internetwork 全体にわたって知られていなければならない経路指定情報の量を制御するメカニズムを提供して、任意のサイズの動的 internetwork に対応すること

2. サービス提供者と人とユーザにより求められる複数の制約の中で、サービスに対して差別化を施した経路制御を提供すること

3. internetwork の加速度的な拡大に対応できること

これらのゴールに適合する Nimrod アーキテクチャを設計した。
このアーキテクチャーは以下のような特長を持つ。

1. internetwork 連結性の表現と複数のレベルにおけるマップの形の抽出の働き

2. マップとトラフィック・サービスの要求事項に基づくユーザーに制御されたルート発生と選択

3. 確立したパスに沿った、ユーザの指示によるパケット転送

Nimrod は、一つの経路指定領域の範囲内と複数の経路指定領域間の両方で経路指定に適用できる総合的経路指定アーキテクチャである。
一般の internetwork 経路指定アーキテクチャが internetwork のサイズの拡大と相違を扱うよう設計されているように、Nimrod は TCP/IP と OSI の環境に等しく適用できる。

4.2　概要

4.2.1　Internetworking 環境の制約

Nimrod アーキテクチャを開発する際に、我々は最初に経路指定のために密接な関係を持つ internetwork 環境の制約のリストを整理した。

以下に列挙したこのリストは、現在のインターネットの観察と、将来 5 から 10 年までの internetworks についての予言を含む。

1. インターネットは、$O(10^9)$ ネットワークを含むまでに成長する。

2. internetwork ユーザーの数は、限られていないかもしれない。

3. internetwork 資源の収容力は堅実に増加している、しかし、それはこれらの資源に対する需要である。

4. Routers とホストは有限の処理容量と有限のメモリを持つ、そして、ネットワークは有限の転送容量を持つ。

5. Internetworks は、通信媒介の異なる型を含む—有線、無線電信、地上、衛星、共有 multiaccess とポイント・ツー・ポイント—これらはスループット・遅れ・エラー・損失配送・プライバシーの観点において異なるサービス特徴をもつ。

6. ネットワーク、ルータ、ホストとプロセスなどの Internetwork の要素は、移動可能である。

7. サービスのプロバイダは、これらのサービスへのアクセスにおける、提供されたサービスと制約を指定する。制限とは例えば、サービスがいつ利用できるか、サービスはいくらくらいかかるか、どのユーザがそのサービスに加入してもよいか、そしてどんな目的のためか、そしてユーザはサービス保証を受けるためにどのようにトラフィックを制御しなければならないか、ということである。

8. ユーザーは、セッションの中で広く変化しうるトラフィックサービス要求を指定する。この仕様は、次のような見地から決められる。要求品質、ユーザがこれらサービスに払おうと思うような金額、ユーザがこれらサービスを使いたい回数、ユーザが使おうと思う提供者。

9. ユーザトラフィックセッションは、m 個のソースと n 個の行き先を含みうる $(m, n \geq 1)$。

10. サービス提供者とユーザは、相互依存的な関連を持つ。

すなわち、ユーザが特別サービス要求のあるいっそう多くのアプリケーションを開発するにつれ、サービス提供者はこれら要求に適応するサービスによって応じ、そしてユーザは、これらサービスの優位性を得るいっそう多くのアプリケーションを開発する。

11. . 様々な、そして特別サービスのサポートは、その部分の上にこれらのサービスを提供しているサービス提供者とこれらのサービスを要請しているユーザについてのいっそう多くの処理、記憶とバンド幅転送を必要とする。

    これにより、多くの経路制御関連機能は、routers とホストによっては実行されず、むしろ経路制御情報を行う独立の装置が自分たちのために経路情報を処理し、たくわえ、そして分配することにより行われるだろう。

12. 特殊化されたサービスを必要としているユーザ (例えば、高く保証されたスループット) は、通常これらのサービスに対してより多くを払い、そしてそれらを得るためにいくらかの遅れを負う気持ちがある。

13. サービス提供者は、彼らがいっそう管理するのが難しいので、複雑なプロトコルをそのネットワークにもたらすのに気が進まない。

14. ベンダは、開発に長い時間がかかるので、その製品の中で複雑なプロトコルを実現するのに気が進まない。

まとめると、これらの制約から、大きくかつ変化する internetwork の中で、成功した経路制御アーキテクチャは特別な特徴、たとえばサービス特有の経路制御と構成部分の可動性を、単純な手続きと最小の internetwork 資源の消費でサポートしなければならない、

4.2.2 基本経路選択機能

Nimrod で提供された基本経路選択機能は、通常の経路制御システムと同じである。

1. ルート発生と選択に必要な情報を、集め、整理し、そして分散させる。

2. この情報に基づくルートを生成して、そして選択する。

3. ルータ内でパケットを選択されたルートに沿って進めることに必要な情報を確立する。

4. 選択されたルートに沿ってパケットを進める。

Nimrod の経路制御機能へのアプローチは、「link-state」パラダイムによるマップ配送、ルート発生と選択についてトラフィックの始点と終点への局地化、始点・終点間の経路の確立によるパケット転送経路の明確化を含む。

link-state マップ配送により、各サービス提供者がそれがその経路制御情報の分配している制限と限定している配送を通して提供するサービスに対する制御を持つことができる。経路制御情報の配送の限定により、internetwork にわたって維持された経路制御情報の量を減らし、そしてある種の経路制御情報を秘密にしておく。しかしそれはまた、internetwork にわたって矛盾している経路指定情報データベースとなる。我々は、経路制御情報データベース不一致が、秘密保護をしているかどうかにかかわらず、大きい internetwork でしばしば起こると予想する。理由は、いくらかの装置が経路制御情報の完全なセットをその internetwork のために維持することができないだろうからである。これらの装置は、分配された経路制御情報のいくつかだけを選び出してそのデータベースの中に記憶する。

マップとトラフィックサービス要求に基づくルート発生と選択は、ユーザか、あるいはいっそうありうるだろう、装置によって完全に制御されうる、そしてルータの中でグローバルな調整を必要としない。このようにこれらの装置は特定のユーザのニーズに適合した経路を生成できる、そして、そのユーザはルートを生成するコストを払う。これらのアルゴリズムが internetwork の中で各ロケーションで同じである必要はないので、局所コントロールによるルート生成は、拡大と新しい経路制御の実験が可能である。

パケット転送は、パスに従って、ユーザーまたはそのために従っているデバイスによって完全に制御されるかもしれない。これらのパスは、そのマップが許す限り、詳細に指定されてもよい。これによりパケットは、パスの中のルータが矛盾している経路制御情報を持つとき、ループ内を転送されるのを回避する。

理由は、転送のパスは一つの装置によって計算されるルートであり、一つの装置で管理されている情報に基づいているからである。

Nimrod アーキテクチャと Inter-Domain Policy Routing(IDPR) が共通の link-state 経路情報配送、ルート生成の局所化とパス指向のメッセージ転送を共有する点に注意する。

4.2.3　Scalability Features

以下に Nimrod の資源消費量を抑制するメカニズムの簡単な概要を提供する — このメカニズムの任意の使用は調節可能な経路制御アーキテクチャを保証しない点に注意。

集合化 (clustering) と抽出 (abstraction)

Nimrod アーキテクチャは、internetwork を複数のレベルでの抽出の実体の集合として表現することができる。

集合化は、経路制御に見える実在の数を引き下げる。抽出は、経路制御に見える実体を特徴づけるための情報量を引き下げる。

集合化は、いくらかのあらかじめ決められた基準に従ってホスト、ルータとネットワークのような internetwork 要素を集合化することから始まる。

これらの要素は、それらの間の「同じ管理者によって管理される」といった関連、ある

いは、「各ルータでたくわえられる情報の予想される転送量を最小にする」といった目的を満足するために、集合化される。

Nimrod は、特別な集合形成アルゴリズムに依存しない。

既存の集合を集めることにより、新しい集団を形成してもよい。

実体の集合化を繰り返すことにより、他の全てを含む一つの集合を頂点とする集合の階層が形成される。

階層の各レベルで同じ集合化アルゴリズムが適用される必要はない。

集団の範囲内のすべての要素は、少くとも 1 つの関係、すなわち connectivity を満たさなければならない。

すなわち、集団の範囲内のすべての要素が使用できるならば、いかなる二つの要素も完全に集合内を通る少なくとも 1 つのルートによって接続されていなければならない。

集団が形づくられたら、集団特徴の描写を減らすために、接続性とサービスの情報が抽出される。抽出手続の例としては、サービスのその集団または表現の中で平均の値の観点からその要素の働き (小さい一部分によって提供される) の排除などがある。Nimrod は、特別な抽出アルゴリズムに依存しない。同じ抽出アルゴリズムが各集団に適用されなくてもよい、そして、複数の抽象アルゴリズムが一つの集団に適用されてもよい。

集合化と抽出のアルゴリズムは、ホスト、ルータとネットワークの物理的な組織から独立している。Nimrod の経路制御は、その internetwork の物理的な実現によってではなく、集合化と抽出による実体の階層によって行われる。実際は Nimrod は internetwork の物理的な要素にすら気づかなくてもよい。

情報配送を限定する

各集合は、経路制御情報の分配する部分と、それを配る相手の実在のセットを決定する。以下が例である。各集合は、経路情報を自動的に (共通の親により集合化されている) 兄弟に伝達するかもしれない。要求に答えて集合は、集合あるいは特定のユーザにあてはまる情報の特定の部分を伝達するかもしれない。集合は、サービスへの普遍的なアクセスを提供する集合からの経路制御情報を保持するだけかもしれない。

実行可能なルートのローカルな選択

複数の制約を満たすルートの生成は通常 NP 完全問題であり、計算量の多い手続きである。Nimrod では、特別な制約のルートを必要とする実体だけがそのようなルートの発生と選択による計算の負荷を引き受ける必要があると仮定する。さらに Nimrod のアーキテクチャでは、個別の実体が自身によるルート生成と選択のアルゴリズム、その機能に向ける資源の量を選ぶことができる。

Caching

Nimrod アーキテクチャは、消費される資源と将来情報を得る際に生じる遅れの量を減らすために、獲得した経路情報の caching を促進する。特定経路の生成の副産物として生成された経路の集合は、cache 可能な経路情報の例である。後から来るこれらの経路に対するリクエストは、経路 cache ですぐ応答できる。しかしいかなる cache 法と同様、cache された情報は古くなるかもしれず、その使用が不十分な質の経路を招くという結果に終るかもしれない。それゆえ、情報を cache するかどうか、そしてどれくらい長く cache するのかどうかを決めるにあたっては、経路情報の有用性の期待される持続期間が考慮されなくてはならない。

forwarding 情報の制限

Nimrod アーキテクチャは、各 router が維持する forwarding 情報量を収容するため、二つの異なるアプローチをサポートする。最初のアプローチは、一つのパス (または multicast の場合、木) にわたり、似たサービス要求を持つ複数の traffic flow をマルチプレックスすることである。二番目のアプローチは、アクティブな traffic flow に関してだけ、forwarding 情報を導入・保持することである。

Nimrod により、サービスプロバイダとユーザは internetwork 内の forwarding 情報の量に対する責任を共有する。ユーザはパスの確立に対する制御を持ち、サービスプロバイダはパスの維持に対する制御を持つ。このアプローチは forwarding 情報がこれに対する要求と独立に router の中で確立されてしまう、現在の Internet のそれと異なっている。

4.3　アーキテクチャ上の Overview

Nimrod は、階層的な、マップに基づく経路制御アーキテクチャで、広範囲にわたるユーザ要求を支持し、非常に大きな動的な internet に対応するよう設計された。トラフィックの記述と要求 (サービス要求の品質と、使用上の制約についての要求) が与えられたとき、Nimrod の主な機能は、そのトラフィックについて経路を選ぶときにどれくらい internetwork に関する情報が要求されるかを、調節可能な形態の中で管理することである。 言い換えれば、internetwork について経路の情報の量と、計算された経路の品質とのトレードオフを管理することである。Nimrod は、一組のプロトコルと分配されたデータベースとして実現される。以下のセクションでは、 Nimrod の中で使われる基礎的なアーキテクチャ上の概念を記述する。そのプロトコルとデータベースは、他の文書の中で指定される。

4.3.1　Endpoints

Nimrod の中の基礎的な実体は、endpoint である。endpoint は、internetwork 層のユーザを代表する (例: トランスポート接続)。各々の endpoint は、少くとも 1 個の endpoint 識

別子（EID）を持つ。与えられた EID は、一つの endpoint に対応する。EID は、地球的にユニークな、相対的に短い「親コンピュータ的」ビット列である—例えば 64 ビットの複合。EIDs は、全然位置的な意味を持たない。管理の楽のために EIDs は階層的に組織されるかもしれない、しかし、これは必要でない。

> 多分、実際問題として EID は我々が endpoint なラベル（EL）を呼ぶことができる二番目の形式を持つだろう。ELs は、無制限の長さの ASCII ストリングである — キーとして分配されたデータベース (DNS の名前と非常によく似た) の中で使われるために組み立てられる endpoint についての情報— 例えば、どのように到達するのか —は、キーとして endpoint のラベルを使用しているこの分配されたデータベースを調べあげることによって得られることができる。

4.3.2 マップ

経路制御のために使われる基礎的なデータ構造は、マップである。マップは、internetwork の異なる点の間の利用できる connectivity を表す。異なるマップは、物理的なネットワークの同じ地域を詳細の異なるレベルにおいて表現することができる。

マップは、ノードと弧から成るグラフである。ノードの属性は、それらと関連する属性に含まれる。弧は、属性を持たない。弧は、ノードの属性としてマップに現れる。Nimrod は、属性を指定して、マップを記述するための言語を定義する 。

マップが、ルートを生成するためにルータに使われる。一般に、異なるルータが一貫したマップを持つことは必要でない。

この文書の中で我々はルータについてのみ話す。「ルータ」とは、経路制御に関連した機能 (例:転送、経路計算、パス準備) を実現する物理的な装置を意味する。与えられた装置は、ルータと呼ばれるために、これらすべての機能ができる必要はない。後で—たとえば、プロトコル定義の中で—これらの機能を分けることは明白に便利かもしれない。それから、我々は転送エンジン、パス生成エージェント、ルートサーバ等について話すかもしれない。

> Nimrod は、異なるルータの経路データベースが一貫していないとき、経路ループが生じないように設計された。一貫性要求は、その internetwork の同じ地域を詳細の異なるレベルにおいて表現するのを許さない。また、経路データベース一貫性要求を保証することは、Nimrod が対応するよう設計された非常に大きい internet の中では難しいだろう。

接続性仕様

2 個の点の間の接続性とは、利用できるサービスとその使用における制約を意味する。接続性仕様は、ノードと関連する属性の中にある。以下は、接続性仕様の非公式の例である:

- 「これらの 2 個の点の間に、best-effort サービスは制限なしで存在する」。

- 「これらの 2 個の点の間で、10 ミリ秒の遅れが保証されたトラフィックの流れは、転送レートが 1Mbyte/sec 以下で、低い (指定された)burstiness を持つ。

- 「これらの 2 個の点の間に best-effort サービスは、提供され、その流量が始まるのと、同じくらい長く、そして組織を研究する運命にある」。

  接続性仕様の定義は、2 個の点の間ばかりでなく、点の集合の間でもできる。
  Nimrod は、接続性仕様を定義するための言語を含む 。

4.3.3　ノードと弧

ノードは、物理的なネットワークの領域を代表する。ノードによって表現されるネットワークの領域は、望みのまま大きくまたは小さくできる:ノードは、大陸またはホストの内側に走っているプロセスを表現することができる。さらにセクション 4 で説明されるように、そのネットワークの領域は、1 以上のノードによって同時に表現されることができる。

弧は、無向である。弧は、2 個の区別できる端点を持つ:head と tail (弧はしばしば視覚化されて、そして矢印で引かれる；その弧の頭はその矢の頭に対応する)。弧のその頭と尾は、各々ノードに接続している。

2 個のノードの間の弧の存在は、トラフィックがそれらの 2 個の点の間にその弧によって示される方向に流れることができることを示す。2 個の与えられたノードの間に一つの弧だけが、両方向にあることができる。弧は、常に異なるノードを結ぶ。

4.3.4　BTE

マップの区別できる構成要素 (ノードと接続性仕様) は、基本位相的実体 (basic topological entities, BTE) と呼ばれる。

4.3.5　ロケータ

ロケータは、マップの中の基本位相的実体（BTE）を確認するバイナリの桁のストリングである。異なる BTE は、必然的に異なるロケータを持つ。与えられた BTE は、一つのロケータだけを割り当てられる。ロケータは、BTEs を確認し、BTE がそのネットワーク中のどこにあるかを指定する。ロケータは、BTE にパスを指定しない。

この文書ではロケータは、ノードの構造をたどるために":"を含む ASCII ストリングとして書かれる (例 a:b:c)。データベースの中またはパケットの中のロケータの表現が必然的に何かコロンと等しいものを持つということを、これは意味しない。

与えられたネットワークの物理的な要素が一つ以上の BTE を実現するかもしれない (例えば、2 個の異なるノードの一部であるルータ)。したがって、この物理的な要素が一つ以

上のロケータと関連付けられたり、この物理的な要素が実現する BTE が各唯一の一つのロケータを持つかもしれない。

ノードは、接頭辞としてそのノードのロケータを持つようなロケータを所有すると言われている。内部のマップを持つノードの中で、この内部のマップの中のすべての BTE のロケータは、オリジナルのノードのロケータを接頭辞に持つ。特にノードの内部のマップに現れているノードのロケータは、そのノードのロケータを接頭辞に持つ。

弧は、ロケータを持たない。

ノードと関連するすべての接続性仕様のロケータは、また、ノードのロケータによって接頭辞を付けられる。

すべての経路制御マップ情報はロケータにより表され、そして、経路制御選択はロケータに基づく。EID は、経路制御決定を下すのには使われない—セクション 5 を見よ。

4.3.6 ノードの属性

以下は、Nimrod によって定義されるノードの属性である。

弧

弧は、その tail ノードの属性としてマップに現れる — "from" ノード。ノードと関連するあらゆる弧は、オリジナルのノードがデータを送信することができる先の近隣のノードを示す。ノードを与えられたとき、「隣接のソース」は、オリジナルのノードへデータを送信することができるノードである。それは、オリジナルのノードを head とする弧を持つノードである。同様に、「隣接の行き先」は、オリジナルのノードがデータを送信することができる先のノードである。それは、オリジナルのノードの 1 つの弧の head のノードである。

内部のマップ

その属性の一部としてノードは、内部のマップを持つことができる。ルータは、ノードの内部マップ、または必要ならノードの他の属性を得ることができる–そのノードの代表者からの情報を要求することにより。(そのノードと関連するルータは、そのような代表者であることができる。) ノードの代表者は、原則として、異なる内部マップで異なる要求に答えることができる—例えば、セキュリティのために。これは、そのネットワークの中の異なるルータが同じノードのために異なる内部のマップを持つかもしれないことを意味する。

マップを与えたとき、マップのノードの 1 つをそのノードの内部に置き換えることによって、ルータはいっそう詳細なマップを得ることができる。このプロセスは、再帰的に続けられることができる (多分ルータは、その現在の関心のマップの地域において、ノードを広げる)。Nimrod は、特定の目的のために使われることを意図した標準の内部のマップを定義する。ノードの「詳細なマップ」は、オリジナルのノードによって代表されるそのネットワークの領域について、いっそう多くの情報を与える。典型的には、それはそのネット

ワークの物理的な実現にオリジナルのノードよりもより近い。このマップのノードは、彼ら自身マップを詳述することができた。

通過接続性

与えられたノードにおいてこの属性は、隣接のソースと隣接の行き先の間に利用できるサービスを指定する。この属性は、ルータがノードを通してトラフィックを発送しようとするとき、要請され、使われる。概念的には、トラフィック接続性属性は、ロケータによってインデックスを付けられるマトリックスである:隣接のソースのロケータと隣接の行き先のロケータ。そのようなペアによってインデックスを付けられたエントリは、与えられたノードを横切って利用できる隣接の行き先へ行っている隣接のソースから入っているトラフィックのためのサービスの接続性仕様を含む。

この属性の実際のフォーマットは、マトリックスである必要はない 。この文書は、この属性についても次の二つの属性についてもフォーマットを指定しない。

Inbound 接続性

与えられたノードにおいてこの属性は、隣接のソースからそのノード内の点との接続性を示す。この属性は、ルータがそのノード内の点にトラフィックを発送しようとするとき、要求されて、そして使われる。しかし、ノードの詳細なマップは持たず、得ることもできないし、得ようと思わない。inbound 接続性属性は、接続性仕様がロケータの対の間で利用できることを示す。そのペアの最初の要素は、隣接のソースノードのロケータであり、その二番目は、そのノードによって所有されたロケータである。

Outbound 接続性

与えられたノードにおいてこの属性は、そのノード内の点から隣接のソースへの接続性を示す。この属性は、ロケータの対の間で接続性仕様が利用できることを示す。そのペアの最初の要素は、そのノードによって所有されたロケータであり、二番目は、隣接のデスティネーションのロケータである。

Transit、Inbound、Outbound の接続性属性は、「抽象的なマップ」として知られている。

4.4　物理的実現

ネットワークは、ルータ、ホスト、通信リンクといった物理的な要素の構成によりモデル化される。リンクは、ポイントツーポイント—例えば T1 接続—あるいはマルチポイント—例えばイーサネット、X.25 ネットワーク、IP のみのネットワーク等—のどちらでもよい。

ネットワークの物理的な表現は、一つ以上の Nimrod マップと関連させることができる。Nimrod マップは、物理的なネットワークだけでなくさらに要素 (ロケータ割当て) の構成

されたクラスタ化、そして構成された接続性のための機能である。

Nimrod は、あらかじめ定義された「もっとも低いレベル」を持たない: たとえば、物理的に CPU の内側に実現されるマップを定義して広報することは可能である。このマップの中でノードは、たとえば、プロセスまたはグループを表現してもよい。このマップのユーザは、知る必要もなければ、気にかける必要もない。

4.4.1 接近

接頭辞を共有しているロケータは、マップの隣接する領域に割り当てられなければならない。すなわち、接頭辞を共有するロケータを割り当てられたマップの 2 要素 (BTE) は、互いに、それら自身その接頭辞によりロケータを設定された要素によって接続されていなければならない。この要求 D の主な結果は、「あなたがあなたとともにあなたのロケータをとることはできない」ことである。

これの例として (図 4.1参照)、2 個の提供者として x.net と y.net をとり (これらの指定は、ロケータではなく DNS 名である)、これらはロケータ A と B を持つ 2 個のノードとして Nimrod マップに現れる。

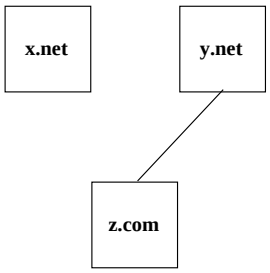

図 4.1: 提供者を切り替えた後の接続性

会社 z.com(これもまたロケータではない) は、x.net に本来接続していると仮定する。z.com の中で要素に対応しているロケータは、この例では A-prefixed である。会社 z.com は、提供者を変えることを決める — x.net へのその物理的な接続を切断する。このセクションの中で記述された接続性要求は、提供者の変更が起こったあとで z.com の中の要素に、この例では接頭辞 B のロケータが割り当てられ、そして y.net を通して接頭辞 A のロケータを予定しているデータを受けることができない、ということを意味する。

接近要求は、経路情報交換を単純化する:

z.com が y.net を通して接頭辞 A のロケータを受けることが許されたならば、ノード B を含むマップが接頭辞 A のロケータの集団がノード B の内部に存在することについての情報を含むことが必要である。

同様に、ノード A を含むマップは、z.com に割り当てられた接頭辞 A のロケータのセットが A の範囲内で見つけられないという情報を含まなければならない。

以上のようなことが起これば起こるほど、Nimrod の階層的な姿が「平らな経路制御」へと破壊されていく。

接近要求は、また、次のように表されることができる。「EID は安定している；ロケータは、短命である。」

接近要求は、Nimrod の移動体での実現へのいくらかのアプローチを除外する。

たとえば可動性のホストは、その新しいロケーションからその「ホームの」ロケータを通知することができない。

4.4.2　例

図 4.2は、物理的なネットワークを示す。

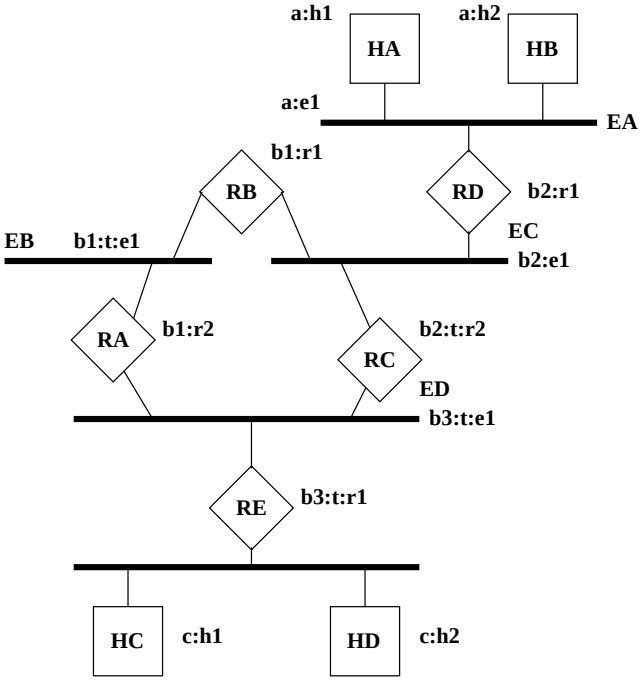

図 4.2: 物理的なネットワークの例

ホストは四角形として、ルータはダイヤモンドとして、通信リンクは線として描かれる。

示されたネットワークは、以下の構成要素を持つ:

5 個の ethernets — EE を通しての EA；

5 個のルータ — RE を通しての RA；

4 人のホスト — HD を通しての HA。

ルータ RA、RB、RC は、バックボーンイーサネット EB、EC、ED を相互に連結させる。

ルータ RD は、イーサネット EA とホスト HA、HB から成っているネットワークに、バックボーン EC を結ぶ。

ルータ RE は、イーサネット EE とホスト HC、HD から成っているネットワークに、バックボーン ED を相互に連結させる。

設定されたロケータは、対応する物理的な実在に比べより低いケースに現れる。

図 4.3は、そのネットワークの Nimrod マップを示す。

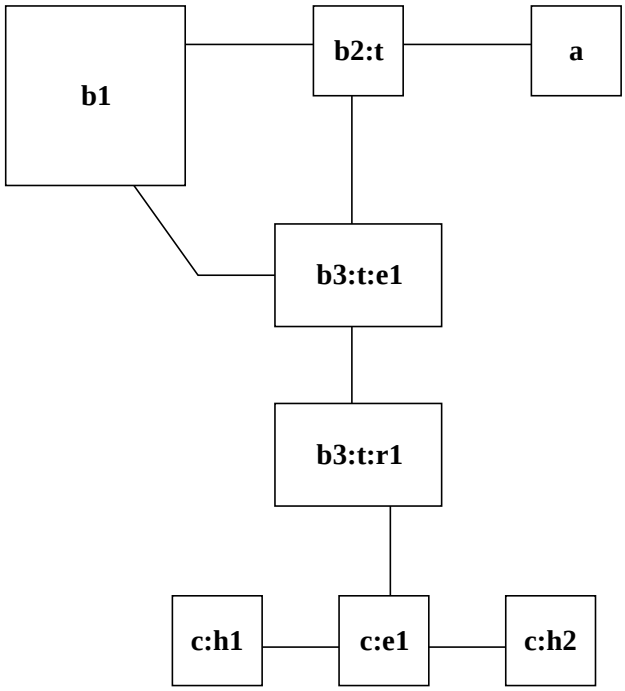

図 4.3: Nimrod マップ

そのマップのノードは、四角形として表現される。

ノードを結んでいる線は、対立している方向に 2 個の弧を表現する。

そのネットワークの異なる領域は、異なる詳細で表現される。

バックボーン b1 は、一つのノードとして表現される。

接頭辞”a”を付けられたロケータをもつネットワークの領域は、一つのノードとして表現される。

接頭辞”c”を付けられたロケータをもつネットワークの領域は、全面的な詳細として表現される。

4.4.3 複数のロケータ割当て

物理的な要素は、一つ以上の BTE の一部を形づくるあるいは実現することができる。

この意味において、それらが 1 つ以上のロケータを割り当てられることができると言える。

物理的なネットワークを示す図 4.4を考慮する。

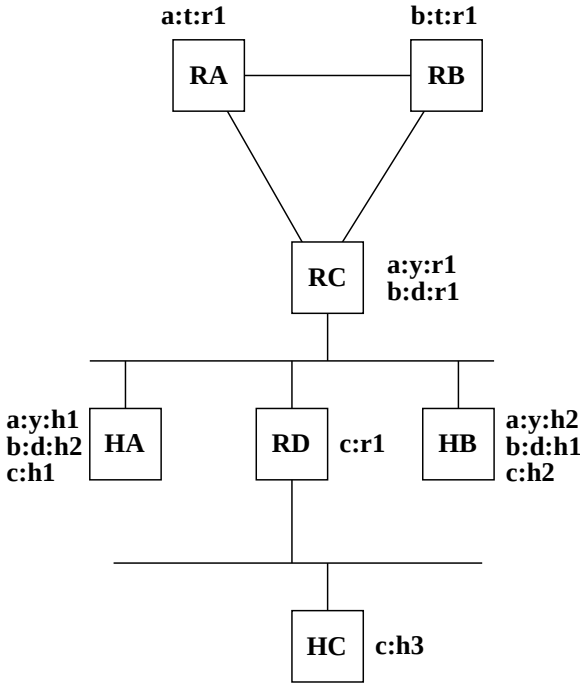

図 4.4: Nimrod マップ

このネットワークは、ルータ (RA、RB、RC と RD)、ホスト (HA、HB と HC) と通信リンクによって構成される。

ルータ RA、RB と RC は、ポイントツーポイントリンクで接続されている。

その図の最下部の 2 本の水平な線は、イーサネットを表現する。

その図は、また、ホストとルータに設定されたロケータを示す。

図 4.4において、RA と RB は、各々一つのロケータを割り当てられた (それぞれ a:t:r1、b:t:r1)。

RC は、ロケータ a:y:r1 と b:d:r1 を割り当てられた；

そのうち一方は RA のロケータの接頭辞を共有し、もう一方は接頭辞を RB のロケータと共有する。

ホスト HA と HB は、各々3 個のロケータを割り当てられた。

ホスト HC は、一つのロケータを割り当てられた。

点の間にどの通信パスが確立されたかに基づいて、異なる Nimrod マップが作られる。

このネットワークに対する可能な Nimrod マップは、図??において与えられる。

ノードと弧は、ネットワークに構成されるクラスタ化と接続性を表現する。

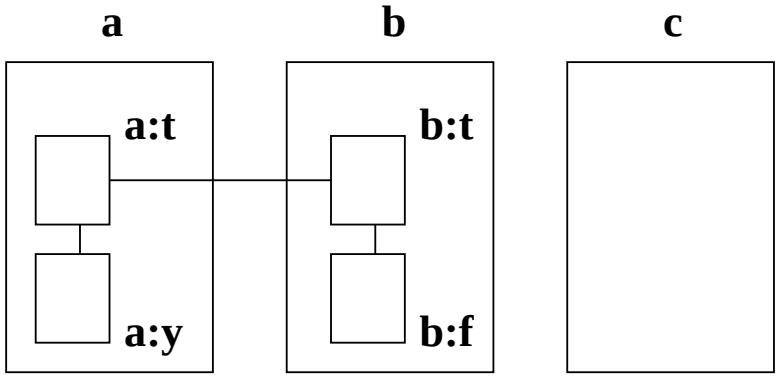

図 4.5: Nimrod マップ

a:y と b:d が同じハードウェアの上に定義されるけれども、そのマップが彼らの間の接続がないことを示すことに注意:この接続は構成されなかった。

”b:d”の接頭辞を付けられたロケータにアドレスされたノードの ‘a’ に与えられたパケットは、その行き先の方へ導かれる前に、ノードから彼らに加わっている弧を経たノードの b に伝わる。

同様に、そのマップは、c ノードと他の 2 個の最高のレベルのノードの間の接続を示さない。

希望する場合、これらの接続は確立されてもよい、そしてそれは経路情報の交換を設定することを必要とする。

これらの接続が確立されたとき、図 4.6はそのマップを示す。

厳密に言えば、Nimrod ノードは、重ならない:それらは、はっきりした実在である。

ただ我々が前の例の中で見えたように、物理的な要素は 1 以上のロケータを与えられることができて、そして、その意味で、一つ以上のノードの実現に参加する。

すなわち、2 個の異なるノードは、同じハードウェアの上に定義されるかもしれない。

この意味において、Nimrod ノードは重なると言うことができる。

しかし、この重複に気がつくためには物理-マップの対応を知っていなければならない。

2 個のノードが物理的な資産を Nimrod マップだけを見ることによって共有するとき、知ることは可能でない。

> Nimrod-aware でないネットワークの隣接する領域は、そのネットワークの領域が提供する接続性を記述する協調通過接続性仕様をもつノードとして表現されることができる。
>
> これの例は、Nimrod ルータを通して Nimrod internetwork に接続されている IP のみのネットワークである。

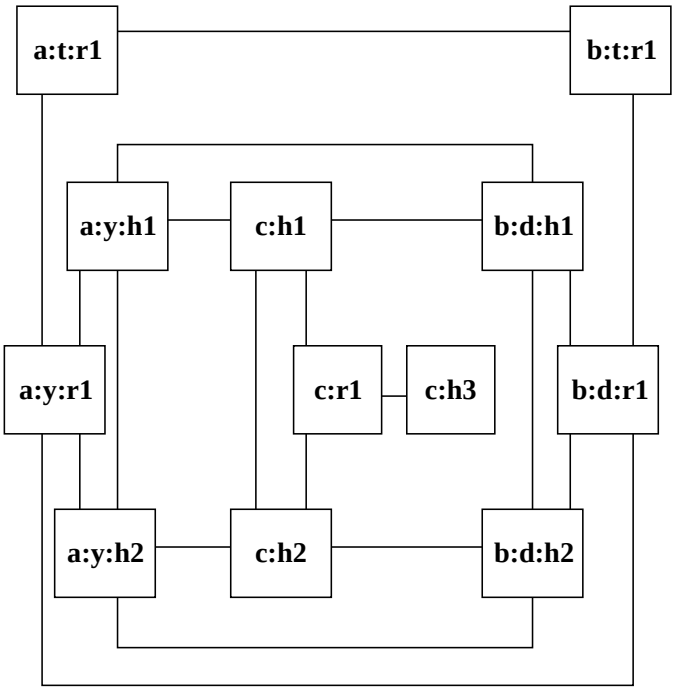

図 4.6: Nimrod マップ II

このネットワークに接続されている Nimrod-aware ホストは、このノードに接続されているノードとして表現される。

Nimrod ホストへと前もって送られるようになっている Nimrod パケット、または、「ネットワークの向こう側」の Nimrod ルータは、たとえば、IP パケットの内側にカプセル化されてもよい。

このネットワークに接続されている IP のみのホストは他の IP のみの集合から届くことができる、たとえば、Nimrod によって使われているフォーマットの IP パケットの中にパケットをカプセル化することにより。

Nimrod インターネットに IP ネットワークを接続させる Nimrod ルータは、IP のみのホストにむけるパケットをカプセルからはずす。

IP のみのホストには、たとえば、そのホストにパケットを送る方法を知っている Nimrod ルータのロケータによって接頭辞を付けられたロケータを与えられる。これによりそれらホストを協調する Nimrod「ルータ」に組み込む。

他の処置は、可能である:

たとえば彼らは IP ネットワークを表現するノードのロケータで接頭辞を付けられたロケータを与えてもよい。

最初のケースにおいて「ルータ」の範囲内でそのルータだけは、どのようにパケットを IP ホストへ転送するか知っている必要がある；

しかし、これはこのルータを失敗の唯一の点にする。

二番目のケースでは、このノードに接続されているすべての Nimrod ルータは、どのように IP パケットを IP のみのホストへ転送するか知っている必要がある。

パケット転送を単純化するために IP のみのホストのためのロケータは、そのホストの IP アドレスを含むかもしれない。

4.5 転送

Nimrod は、パケットフォーマットを特定しない。同じネットワークで同時に考えられるところで、異なるフォーマットで Nimrod を使うことは、可能である。このセクションは、packet-forwarding メカニズムについて Nimrod の要求を指定する。

Nimrod は、4 つの転送モードをサポートする:

1. 接続性定義鎖 (CSC) モード:このモードでは、パケットは接続性定義ロケータのリストを運ぶ。そのパケットは、特定されたサービスを使っている接続性定義を所有するノードを通り抜けることを要求される。リストされた接続性定義で関連させられたノードは、そのマップの中で連続のパスを定義しなければならない。このモードの要求のいっそう詳細な記述は、セクション 5.3 で与えられる。

2. 接続性定義シーケンス (CSS) モード:このモードでパケットは接続性定義ロケータのリストを運ぶ。そのパケットは、特定された順序でリストされた接続性定義の各一つを所有するノードを、順に通り抜けることになっている。そのノードは、隣人であってはならない。このモードは、CSC モードの一般化として見ることができる。CSCs はロケータの鎖であると言われていて、CSSs はロケータのシーケンスであることに注意せよ。この相違は、CSCs の中の接近要求を強調する。このモードの詳細な記述が、セクション 6 にある。

3. フローモード:このモードの中でパケットヘッダは、パスに沿ってルータの中で以前にセットされた状態にインデックスを付けるパス id を含む。状態が確立された時のパケット転送は、相対的に単純である:ルータの状態の指示に続く。Nimrod は、この状態を設定するためのメカニズムを含む。このモードのいっそう詳細な記述は、セクション 5.4 で見つけられることができる。

4. データグラムモード:このモードでは、すべてのパケットヘッダはソースと行き先ロケータを運ぶ。このモードは、CSS モードの特別ケースとして見ることができる。セクション 5.5 で示されるように、転送は以下の手続きにより行われる。

IPV4 のルーズソースルートの間の CSC モードと、IPV4 のストリクトソースルートとルートの間の CSS モードには、明らかな類似がある。

これらのモードのすべてでパケットヘッダは、また、そのソースと行き先のためにロケータと EID を運ぶ。普通の操作の中で転送は EID をアカウントに利用しない、レシーバーだけがする。EID は、そのレシーバーで多重化しないために、そしてある種の本当のエラー状態を発見するために運ばれる。たとえば、EID がそのレシーバーで未知であるならば、そのパケットに含まれたソースのそのロケータと EID は、そのソースへ戻るためにエラーメッセージを生成するために使われてもよい (通常通り、多分、このエラーメッセージはそれ自身、他のエラーメッセージの原因であるのを許されてはならないだろう)。転送はエラー状態に応答するためにソースロケータと EID を使うことができる、例えば、ソースに対しパス id に対する状態が見つけられないことを示すために。

パケットは、マップの中でノード間を動くものとして見ることができる。パケットのヘッダは、暗黙のうちにまたは明示的に行き先ロケータを示す。データグラム、CSC または CSS 転送モードを使うパケットの中で、行き先ロケータは、そのヘッダの中で明示的に示される。フロー転送モードを使うパケットにおいては、行き先ロケータは、パス id とネットワーク内の分配された状態によってほのめかされる (それはまた、明示的に含まれるかもしれない)。マップが与えられると、連携する行き先ロケータが属しているこのマップの中で、パケットはそのノードへ移動する。行き先ノードが「詳細な」内部のマップを持つならば、行き先ロケータは、この内部のマップの中でそのノードの人が所有していなければならない (さもなければエラーである)。そのパケットは、このノードへ行く (同様、再帰的に)。

4.5.1　政策

CSC と CSS モードパケットは、そのパケットが横切らなければならないノードに結び付いた接続性定義を定義することによって政策を実現する。厳密に言えば、パケットヘッダに含められた政策情報はない。すなわち、原則として、ヘッダを見ることによりルート選択をする時においてどんな基準が使われたかを決定することは可能ではない。パケットヘッダは、ルート発生プロセスの結果を含むだけである。同様に、フローモードパケットの中で、政策は選ばれたルートの中で内在する。

データグラムモードパケットは、行き先とソースのロケータを選択することにより、政策経路制御の限られた形式を示すことができる。この選択が存在するために、そのソースまたは行き先の端点はそれらと結び付いたいくつかのロケータを持たなければならない。このタイプの政策的経路制御により、例えば、提供者の選択ができる。

4.5.2 信頼

ノードが Nimrod マップ情報として広報する外部の特徴づけを満たす限り、その内部のマップを漏らさないノードが、その管理者が決めるように内部に経路を動かすことができる。したがって、広報された Nimrod マップは、ノードの実際の能力と一致していなければならない。たとえば、物理的なネットワークと広報された Nimrod マップを示す図 4.7で示されたネットワークを考える。ホストとルータから成る物理的なネットワークは、イーサネットで相互に接続されている。

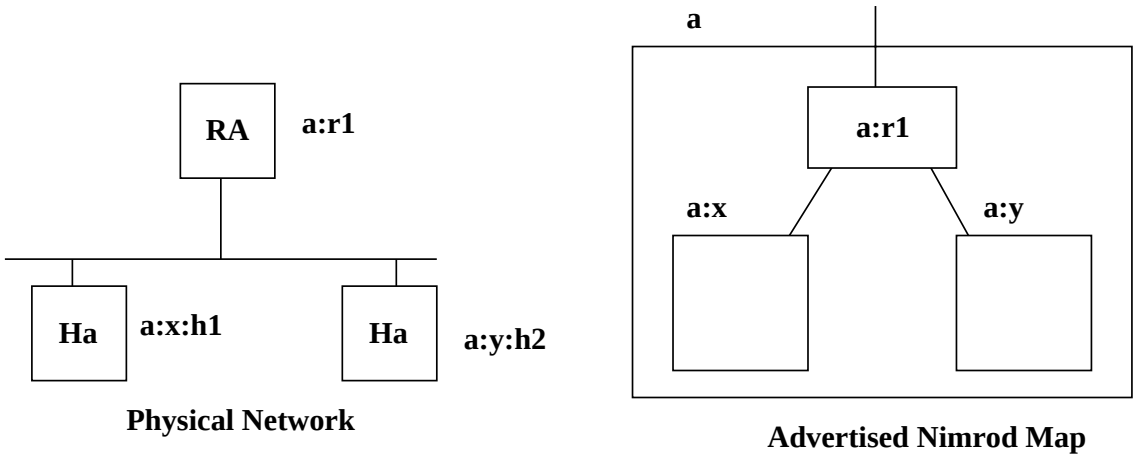

図 4.7: ミスリーディングを起こすマップの例

その図の中で示されるように、このノードは、ロケータを割り当てることと示されたマップを広報することによって構成しているノードに小分けすることができる。そのマップは、ノードの a:y によって注目されるこれらのノードなしでパケットをノードの a:x へ送信することは可能なことを意味するように見える。しかし、これは実際には実施可能でない。

一般に、ルータの得たマップにどれほどの信頼が持てるのか尋ねることは、合理的である。ノードが「信頼でき」、そしてそのノードから受けられた情報が認証されたとしても、誤る可能性は常にある。これらは、Nimrod に特有でない難しい問題である。多くの研究と標準グループがこの問題について言及している。我々は、それについて利用可能になれば、これらのグループの結果を Nimrod に編入する予定である。

4.5.3 接続性定義 (CSC) モード

CSC パケットに対する経路制御は、パケットヘッダで運ばれたロケータのリストによって指定される。そのロケータは、彼らがそのパスに沿って現れるという順序で、特定されたパスを作る接続性定義に対応する。これらの接続性定義は、ノードの属性である。CSC パケットによって示されたルートは、物理的な実在ではなく接続性定義の観点から特定さ

れることに注意せよ:CSC ヘッダの中のロケータは、物理的なパスを指定しないでネットワークの 2 点間の一種のサービスに対応できる。

CSC モードパケットのヘッダに連続的に現れる 2 個の接続性定義ロケータが与えられたとき、最初の接続性定義に対応しているノードから二番目の接続性定義に対応しているノードに行っている弧が存在しなければならない。

CSC モードヘッダの中で参照される最初の接続性定義は、outbound 接続性定義でなければならない; 同様に、CSC モードヘッダの中で参照される最後の接続性定義は、指示接続性定義でなければならない; その残りは、通過接続性定義でなければならない。

4.5.4　フローモード

フローモードパケットのヘッダは、path-id フィールドを含む。このフィールドは、中間のルータの中で確立された状態を確認する。このヘッダはまた、そのソースと行き先のためのロケータと EIDs を含むことができる。Nimrod は、中間のルータのフロー関連状態を修正し設定するプロトコルを含む。これらのプロトコルは、要請されたルートを識別しないし、またフロー (例えば、バンド幅、ディレイなど) によって要請される資源を記述しない。セットアップの試みの結果は、セットアップの確認あるいは失敗の通知である。フローモードの中のソース指定ルートは、CSC の観点から特定される。

4.5.5　データグラムモード

現実的な経路制御アーキテクチャは、遠隔翻訳データベースの中のルックアップのような一つのパケットから成るユーザ処理を意味する、データグラムトラフィックの最適化を含まなければならない。ネットワークトラフィックの多くがそのようなデータグラム処理から成るならば、これまでの 2 つの前のモードのいずれも容認できないオーバーヘッドを含む。既存の IPV4 の hop-by-hop メカニズムと同じくらい効率的なメカニズムが必要とされる。Nimrod は、そのようなメカニズムを持つ。

そのスキームは、データグラムネットワークの中でルータと実際のパケットの間での状態を分割する方法により特徴づけられる。ほとんどのパケットは現在、転送プロセスと連携した少量の状態だけを含む (転送状態)—ホップカウント。Nimrod が提案する、パケットの中で転送状態の量を大きくすることは、役に立つ性質をもつシステムを生産することができる。その方式は，部分的に SIP の効率的なソース経路制御メカニズムと PIP のロケータポインタメカニズムからも影響を受けている.

Nimrod データグラムモードは、プリセットフローモード状態を使って、厳密にループのないパスを得る (ソースルートを用いずに)。データグラムモードの中でそのパケットは、ローカルに有用な path-id フィールドに加えて、以下を含む:

- ソース、行き先ロケータ

- ロケータへのポインタ

そのポインタは、ソースロケータのもっとも低いレベルから始め、そして、そのロケータを上昇させ、そして、行き先ロケータへ持っていき、そして、下降させる。そのパケットの中のこれらの特別なフィールドに加えて、すべてのルータは、前もって設定されるフローの最小のセットを含まなければならない、あるいは抽象階層の中で重大な場所にある確実なルータにこれらのフローを需要に応じて準備させなければならない。

「前もって設定される」フローは、実際に前もって設定される必要はないが、しかし、要求があり次第つくられることができる。そのシステムのために、働くために設定することができなければならないフローの最低限のセットがある。実際のトラフィックの要求がある前に設定されるそれらのフローのいずれかも、全くローカルな決定である。効率的に動くために、データグラムが要求するとき、ラウンドトリップの遅れを避けるためにフロー設定要求とともにデータパケットは送られてもよく、フローはそれを実際に扱うために設定される。これらのフローを「データグラムモードフロー」または「DMF」と呼び、それらはいずれも実際に必要とされるまで、つくられる必要はない。

そのメカニズムの実際の操作は、かなり単純である。ソースロケータを、行っている間、各「活発な」ルータ (たとえば、フロ-の一部として扱うために、どこにそのパケットを送るか実際に決定を下すもの) は、ソースロケータの「次のより高い」レベルの目的にパケットを連れていって、ポインタを進め、そしてその DMF に沿ってパケットを発送する、そのような DMF を選択する。その DMF の終わりに着くとき、そのプロセスは、パケットが 2 個のロケータの最も少ない共通の交差であるルータに届くまで繰り返す。(例えば A:P:Q:R と A:X:Y:Z に対して、これはパケットが A に到達したときである。)

それからそのプロセスは、行き先ロケータの次のより低い目的にパケットを連れていく DMF を選択している各活発なルータで逆転する。つまり、A は A:X へのフローを選択し、そしてそれが A:X に着いたならば、A:X は A:X:Y へのフロー選択する、等々。

結果として生ずるパスがループがないことをプロセスが保証することは、簡単にわかる。選択された各フローは、必然的にそのパケットをその行き先により近くしなければならない (各フロー選択がそのロケータを通して単調に進められていたポインタになったから)、そしてそのフロー自身がそのパスが設定されることに先立ち選択されるとき、ループにならないことを保証する。

そのシステムが DMF の最小のセット以上のものを保つならば (上限は内部のルータの中の一つの境界ルータまで、そして下限は各境界ルータにおける一レベル低い各目的まで)、

そしてテーブルを効率的なルックアップのためにソートされてるよう保つと (例えば hop-by-hop データグラムのための現在の経路制御テーブルと多くの点で同じ)、経路制御はいっそう効率的であることができる。例として上のケース (A:P:Q:R から A:X:Y:Z へのパケット) を使って、A:P:Q が A:X:Y に実際に隣人であり、そして A:P:Q から A:X:Y への直接フローを維持するならば、

そのパケットが A:P:Q に届くとき、その経路の残りを上下する代わりに、そのポインタ

は、A:X:Y の行き先ロケータの値をつけられることができ、そしてそのパケットはそこで直接送られる。トラフィックモニタリング、そして分析 (再び、全くローカルなアルゴリズム使う) は、何度も作られるデータベースの中に結果が得られ、その中でどの DMF が最小のセット以上・以下のものについて保つ価値があるか示す。

トラフィックモニタリングはまた、必要とされる実際のトラフィックに先立って設定するために役に立つ DMFs の必要な最小のセットによるフローを示す。しかし、再びすべてのこれらのセットは、そのシステムの操作を全体として乱さないで、ローカルな、徐々に増大する経路の中で、変更することができる。

これらの新しい転送状態フィールドは、end-end 承認システムにも、いかなる既存の (状態を転送している) ホップカウントフィールドにもおおわれていない。これは、そのパケットがそのネットワークを横切ることによりこれらのフィールドの内容が変わるという事実によって起こる問題を止める。

これらのパケット転送はとても効率的であることができる (おそらくさらに標準の hop-by-hop よりいっそう)。非活動的なルータの中で、全くソフトウェアの関わり合いのないハードウェア処理を可能にする方法で、パケットとフローを連携させられる。活発なルータの中で、次の DMF をルックアップするプロセスは、現在の経路制御テーブルをルックアップするのと同様に高価であり、

そして主な相違は、ルックアップがパケットに保管されなければならないということによる、これは大した差とならない。

4.6　接続性指定シーケンスモード

接続性指定シーケンスモードは接続性指定ロケータのリストで経路を指定する。連続ロケータの隣接制限はない。

> CSS および CSC モードはデータグラムおよびフローモードを組み合わせたものとみることができる。それゆえ、ある意味では、Nimrod の基本 forwarding モードは、これら最後の二つしかない。

4.7　リナンバリング

この節では、いかにして Nimrod ネットワークをリナンバリングするかの例を示す。図 4.8はリナンバリング処理中のネットワークを表している。

図は物理的ネットワークと関連付けられたロケータを示す。ネットワークは三つのイーサネットに接続するルータ RA によって形成されている。図には、5 つのホスト “HA” から “HE” が示されている。各ホストの右には、二つのロケータを示している。最初に示すロケータは、古いナンバリングに対応しており、二番目は新しいナンバリングに対応して

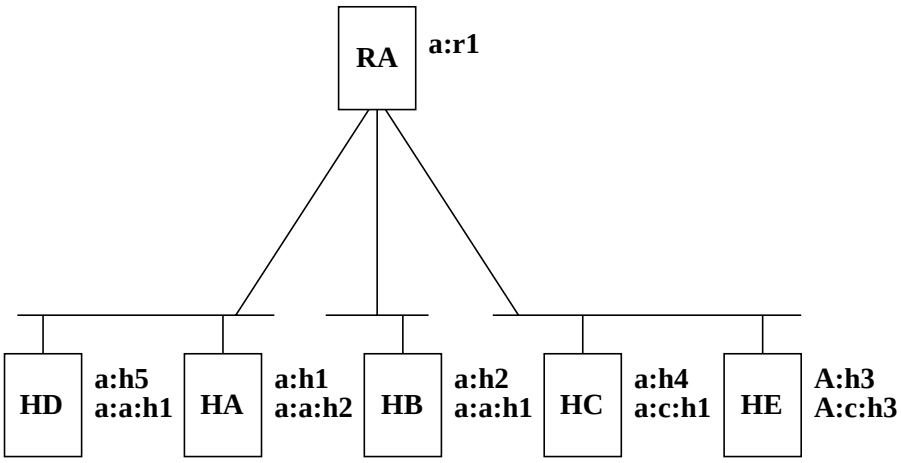

図 4.8: リナンバリング

いる。リナンバリングは、いわば RA の仕事を簡単にするために、新しい階層レベルを追加することからなる。ネットワークエレメントは一つ以上のロケータを持ち得るから、ロケータの二つの集合が同時にアクティブになりうる。初めは、一つ目のロケータ集合がアクティブである。すなわち、ルータ RA はヘッダ中のロケータを与えられれば、どちらのイーサネットへパケットを向けるべきか分かる。(ホストのうち一つへ向けられたパケットが与えられれば、ルータはロケータの「ホスト部」に基づいて、三つのインタフェースのうち一つを採る。例えば、ロケータ a:h1 の “h1” のように。)

たぶん、RA は DNS(またはそれと同等なもの) がロケータの新しい集合を知らされる前に、メッセージをそちらへ forward しはじめようとするだろう。もしロケータが非活性化されたあとで、RA へ古いロケータを含むパケットが与えられたら、エラーメッセージが生成される。古いロケータが再割当される可能性は存在する。もしパケットが間違った endpoint へ受けとられたら、この状況はパケットヘッダに含まれる destination EID を見ることによって、検出できる。

上に述べたリナンバリング手法は、DNS(またはそれと同等なもの) を安全に、相対的・動的に更新できなくてはならないということを暗に意味している。しかし、リナンバリングはほとんどの場合、頻繁なものではなく、注意深く計画されたものであるから、この更新メカニズムの負荷は、許容できるものだと期待している。

このリナンバリング手法の二番目の実装は、保安性と、ホストやルータのロケータを簡単に更新するための、必要条件である。

4.8 保安上の配慮

このドキュメントでは保安上の配慮は述べられていない。

4.9　著者のアドレス

Isidro Castineyra
BBN Systems and Technologies
10 Moulton Street
Cambridge, MA 02138
Phone: (617) 873-6233
Email: isidro@bbn.com

Martha Steenstrup
BBN Systems and Technologies
10 Moulton Street
Cambridge, MA 02138
Phone: (617) 873-3192
Email: msteenst@bbn.com