社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE
電子情報通信学会 IN 研究会 2012-10-11

[招待講演] ネットワーク仮想化基盤における仮想ノード間とスライス-外部ネットワーク間の接続機能

金田 泰
日立製作所 中央研究所
〒244-0817
神奈川県横浜市戸塚区吉田町 292
Yasusi.Kanada.yq@hitachi.com

白石 圭
日立製作所 通信ネットワーク事業部
〒212-8567
神奈川県川崎市幸区鹿島田 890
shiraishi_kei@itg.hitachi.co.jp

中尾 彰宏
東京大学大学院
情報学環・学際情報学府
〒113-0033 東京都文京区本郷 7-3-1
nakao@iii.u-tokyo.ac.jp

あらまし 複数の仮想化ノード (VNode) によって構成されるネットワーク仮想化基盤においては，多数のスライス (仮想ネットワーク) を同時に，他のスライスから隔離された状態で動作させることができる．スライスがふくむ仮想ノード (ノードスリバー) どうしは仮想リンク (リンクスリバー) によって自由に最高 10 Gbps で接続でき，そのなかでは Ethernet や IP にしばられない任意のプロトコルが使用できる．VNode においては，GRE トンネルを使用して仮想リンクを IP ネットワーク上に実装している．また，ネットワーク収容装置 (NACE, NC) を使用してパケット・データ・フォーマットを外部ネットワーク用に変換することにより，スライスを外部の VLAN ネットワークに最高 10 Gbps で接続できる．この論文では，スライス開発者からこれらの接続がどのように参照・定義され，またそれがどのように変換され物理的な通信が実現されるかをを記述する．

キーワード ネットワーク仮想化基盤，仮想化ノード，VNode，スライス，仮想ネットワーク，ノードスリバー，リンクスリバー，ネットワーク収容．

[Invited Talk] Inter-Virtual-Node and Slice-to-External-Network Connection Function in the Network Virtualization Platform

Yasusi Kanada
Central Research Laboratory, Hitachi, Ltd.
Totsuka-ku Yoshida-cho 292, Yokohama 244-0817, Japan
Yasusi.Kanada.yq@hitachi.com

Kei Shiraishi
Telecommunications & Network Systems Division, Hitachi, Ltd.
Saiwai-ku Kashimada 890, Kawasaki 212-8567
shiraishi_kei@itg.hitachi.co.jp

Akihiro Nakao
Graduate School of Interdisciplinary Information Studies,
The University of Tokyo
Bunkyo-ku Hongo 7-3-1, Tokyo 113-0033
nakao@iii.u-tokyo.ac.jp

Abstract On the network virtualization platform that consists of multiple virtualization nodes (VNodes), multiple slices (virtual networks) can be operated simultaneously while isolated from other slices. Virtual nodes (node slivers) can be connected freely by using virtual links (link slivers) at up to 10-Gbps, and arbitrary protocol, which is not constrained by Ethernet or IP, can be used on the slice. VNodes implement virtual links on IP networks by using GRE tunnels. In addition, by using a network accommodation equipment (NACE, NC) that translates the packet data format to an external format, slices can be connected to external VLAN networks at up to 10-Gbps. This paper describes how such connections are referenced and defined by slice developers and how they are translated and physical communication is realized.

Keywords Network virtualization platform, Virtualization node, VNode, Slice, Virtual network, Node sliver, Link sliver, Network accommodation.

1. はじめに

新世代ネットワークのプロジェクトにおいては IP にとらわれない新プロトコルを開発し，IP 上では困難だったさまざまなアプリケーションの実現をはかろうとしている．そのベースとなっているのが仮想化ノード・プロジェクトにおいて開発されているネットワーク仮想化技術と仮想化ノード (VNode) である．このプロジェクトは情報通信研究機構 (NICT) が設定した場において東大，NTT，NEC，富士通，日立が協力して，共同研究および委託研究としてすすめてきた．このプロジェクトにおいては，独立かつ自由に設計された機能を実装した複数のスライスすなわち仮想ネットワークがひとつの物理ネットワーク上で同時に動作できる環境を実現することをめざしている [Nak 10]．実装されるべきネットワーク機能としては IP にかわるべき “clean slate” [Fel 07] のプロトコルやアプリケーションに特化された機能などがある．ここで開発された VNode は研究開発用テストベッド・ネットワーク JGN-X に導入され，委託研究メンバーを中心に使用されつつある．

この論文においては仮想化基盤において VNode 上に生成される仮想ノード (ノードスリバー) どうしの接続と，スライスを外部ネットワークに接続する (外部ネットワークをスライスに収容する) ための接続について，それがスライス開発者からどのように参照・定義され，またそれがどのように変換され通信が実現されるかを記述する．すなわち，2 章においては仮想化基盤とスライスについて概説し，2 種類の接続について記述する．3 章においてはそのうち仮想ノードど

うしの接続の実現法についてのべ，4 章においては外部ネットワークとの接続についてのべ，5 章でしめくくる．

2. 仮想化基盤におけるスライスとその生成・実行

この章においては，仮想化基盤およびそのうえで定義されるスライスについて概説し，スライス定義上での 2 種類の接続についてスライス開発者の視点で記述する．

2.1 仮想化基盤における役割

まず，仮想化基盤にかかわるひとの役割について説明する．仮想ネットワークの生成と運用にかかわる役割は，運用者，開発者，一般ユーザという 3 者にわけられている [Kan 10][Kat 12].

- **運用者 (オペレータ)** は**ポータル** (Web インターフェース) を使用して実ネットワークを管理する．開発者は運用者がシステムに登録する．
- **開発者 (デベロッパ)** はポータルを使用して仮想ネットワークを生成し管理する．すなわち，スライスを生成し，一般ユーザを管理する．仮想ネットワーク上で一般ユーザにサービスを提供する．
- **一般ユーザ**は仮想ネットワークを使用する．端末の設定と端末に関する情報は一般ユーザがポータルに登録する．

この報告においては開発者の視点を中心にする．Chowdhury ら [Cho 09] の用語では運用者は**インフラストラクチャ・プロバイダ**に対応し，開発者は**サービス・プロバイダ**に対応するとかんがえられる．これらのくべつは将来のインターネットにのぞまれていることだが，仮想化されたネットワークにおいてはそれが自然に実現される．なお，前記の 3 つの役割はかならずしもことなるひとにわりあてられているとはかぎらない．たとえば，JGN-X における実験などの場合には開発者と一般ユーザが同一であることが多いとかんがえられる．

2.2 スライスとスリバー

仮想化基盤に関してこれまで使用されてきた用語としてスライス，スリバーなどがあるが，これらは資源管理の視点からの用語である．用語そのものを変更すると混乱をまねくとかんがえられるので，ここではそれらの用語はそのまま使用し，その意味を開発者の視点で定義しなおす．また，ここではわかりやすい説明をめざすので，厳密な定義はめざさない．

仮想化基盤上につくられる抽象化・構造化された仮想ネットワークを**スライス** (slice) とよぶ．ネットワーク仮想化について論じるときに無視できないプロジェクトないしテストベッドとして PlanetLab [Tur 07] や GENI [Due 12] があり，スライスということばはそこからきている．しかし，それらにおいてはスライスということばは単純な資源の集合を意味しているので，スライスの意味には差がある．仮想

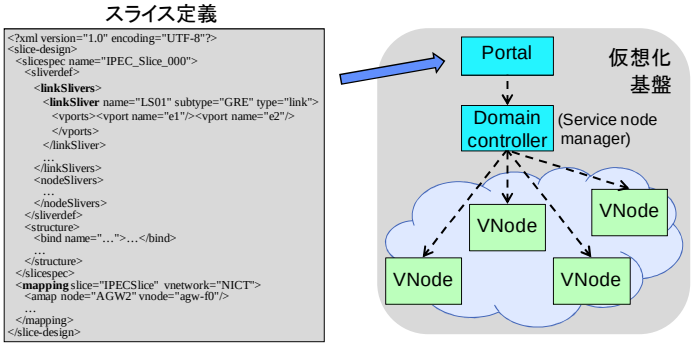


図 1 仮想化基盤へのスライス定義の投入

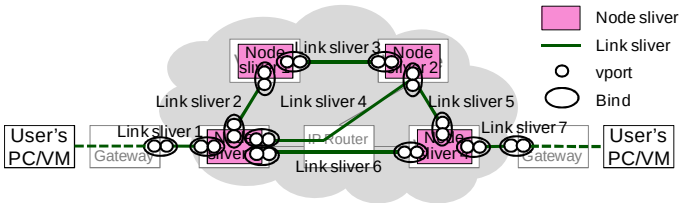


図 2 スライス設計図の例

化基盤においては，開発者は XML にもとづくスライス定義言語によって特定の構造や性質をもつスライスを定義 (記述) して，ポータルから投入できる (図 1)．スライスが使用する資源やその量 (帯域など) を指定すれば他のスライスから隔離され，干渉をうけないようにする**リソース・アイソレーション機能**が実装されている [Kan 12e].

スライスは複数の仮想ノードとそれらをつなぐ仮想リンクとで構成されるが，仮想化基盤において，これらはつぎのように呼ばれる (図 2) [Nak 10][Nak 12].

- **ノードスリバー** (node sliver) は仮想のノードであり，プロトコル処理やノード制御などを実行するのに使用する．初期状態ではなんの機能ももたず，プログラムを使用しないかぎりはパケットはスイッチングもルーティングもされない．ノードスリバーは 1 個または複数個の**スローパス** (slow path) や**ファストパス** (fast path) によって構成される．スローパスは Linux などの OS を搭載した仮想マシン (VM) であり，ファストパスは通常，OS を使用しない (ベアメタルという) 高速処理用の特殊なアーキテクチャをもつノードである．仮想化基盤ではネットワーク・プロセッサを搭載したファストパスを利用することもできる．
- **リンクスリバー** (link sliver) はノードスリバー間を結合する 2 層 (L2) の仮想リンクであり，単純にその一方の端点から入力されたパケットを他方の端点につたえる．IP や Ethernet などにしばられないので，任意のプロトコルが使用できる．通常はことなる物理ノード内にあるノードスリバーを結合する．リンクスリバーは複数の物理ノードにまたがって存在するのに使用する．リンクスリバーは物理リンクと対応している必要はなく，物理リンクによって直接結合されていない物理ノード間に張ることもできる．

開発者は**スライス定義**において，まず特定の性質をもつノードスリバーとリンクスリバーとをそれぞれ独立に記述し，つぎにそれらの結合のしかたを記述する．すなわち，GENI などにおいては仮想リンクの定義において仮想ノードのポートを指定するが，仮想化基盤においてはリンクスリバーをノードスリバーからは独立したオブジェクトとしてあつかい，ノードスリバーとリンクスリバーのを**結合** (bind) 操作によって結合する (図 2 と次節の例を参照) ようになっている．

ノードスリバーがふくむスローパスやファストパスの内部構造や動作はスライス定義とはべつにあたえられる．スローパスの場合は開発者がノードスリバー内の VM の管理ポートの IP アドレスをうけとり，それにログインしてネットワーク・インターフェースの設定，プログラムのロード・実行などをおこなう．ファストパスの場合は通常，スライス定義において指定されたプログラムがロードされ実行される．

2.3 スライス定義の例

この節ではスライスおよびその XML による記述の例をあげる．**図 3** に図示したスライスは 3 個のノードスリバーをふくみ，そのうちの 2 個がそれぞれネットワーク収容装置 (Network ACcommodation Equipment, NACE, NC) 経由で (疑似) データセンタのネットワーク (外部ネットワーク) と接続され，のこりの 1 個がアクセス・ゲートウェイ (AGW) を経由して端末 (PC) に接続される [Kan 12b].

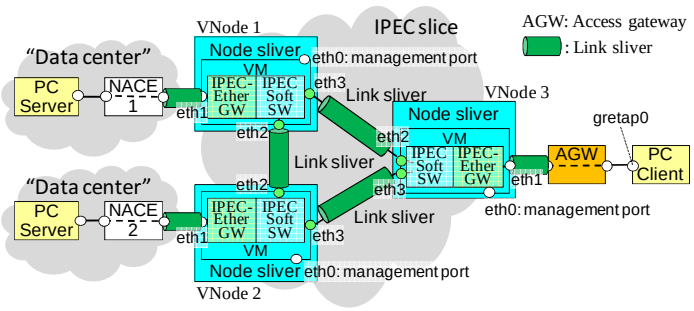


図 3 スライスの構成例

データセンタにおいては Ethernet および IP の使用が前提されているとかんがえられ，また NACE においても Ethernet パケットだけをあつかう．しかし，スライス内部では Ethernet および IP にしばられないパケットを使用することができるし，スライス内で閉じた OpenFlow [McK 08] を実現することもできる ("OpenFlow In A Slice", OFIAS [Du 12])．実際，Kanada et al. [Kan 12b] ではこのスライス上で IP-Ether-Chimera (IPEC) [Kan 12a][Kan 10] という非 IP プロトコルを使用している．端末はインターネットなどを経由して AGW からはなれた場所に設置し接続することを想定しているため，AGW-端末間は IPSec によってセキュリティが確保される．

各ノードスリバー内には 1 個ずつスローパス (VM) を生成する．この VM のブート・イメージの URL をスライス定義において指定する (図 4 参照)．なお，図 3 の各 VM 内のプログラム (箱) はスライス定義とはべつにあたえるものであり，次節において説明する．また，ここに付記した各 VM の eth0, eth1 などのインターフェース名も 2.4 節の説明用であり，スライス定義においては記述しない．

図 3 に対応する XML によるスライス定義を**図 4** に部分的にしめす．このスライスには 3 個のノードスリバー，2 個の NACE，1 個の AGW，3 個のリンクスリバーがふくまれるが，それぞれ 1 個だけをしめしている．スライス定義の末尾で仮想ノード (ノードスリバーと仮想化された AGW) と実ノードとを対応づけているが，この定義 (mapping) は省略できる．省略時は管理系が対応づけする．

ハッチングした部分が「接続」に関係する部分である．すなわち，第 1 がリンクスリバー定義，第 2 がネットワーク収容のための (仮想的な) ノードスリバーの定義，そして第 3 が結合の定義である．以下結合以外の 2 つについて説明する．

第 1 の**リンクスリバー**においては，GRE トンネル [Far 00] によって実装されるため "GRE" が指定されている．将来 VLAN など他の実装法がとりいれられたときには，この部分をかきかえることになる．

第 2 の**外部ネットワークとの接続**は現在のところ VLAN による接続にかぎられている．スライス上のデータ・フォーマットも Ethernet に限定されるが，ネットワーク層 (L3) において IP 以外のプロトコルを使用することは可能である．外部ネットワークの 1 個の VLAN をタグなしで 1 個のスライスに **1 対 1 収容**する方法と，外部ネットワークの複数の VLAN をタグつきで 1 個のスライスに**多対 1 収容**する方法がある [Kan 12b]．図 3 は 1 対 1 収容におけるスライス定義をふくんでいる．<param key="PORT" value="1/7" /> というタグがポート (NACE の物理インターフェース) として "1/7" を使用することを指定している．また，<param key="VLANID" value="100" /> というタグが VLAN ID として 100 を使用することを指定している．これに対して多対 1 のときは VLAN ID として "all" を指定する．

ネットワーク収容の指定においては，"Pipe" という名称の仮想のノードスリバーを使用するなど，やや複雑な設定が必要である．これは将来の拡張にそなえたものである (4.1 節参照)．

```
<?xml version="1.0" encoding="UTF-8"?>
<slice-design>
  <slicespec name="IPEC_Slice_000">
    <sliverdef>
      <linkSlivers><!-- 以下はリンクスリバーの定義 -->
        …
        <linkSliver name="LS01" subtype="GRE" type="link">
          <vports><vport name="e1"/><vport name="e2"/></vports>
        </linkSliver>
        …
      </linkSlivers>
      <nodeSlivers><!-- 以下はノードスリバーの定義 -->
        …
        <nodeSliver name="Node1" type="prog">
          <vports><vport name="vp1"/><vport name="vp2"/>
              <vport name="vp3"/></vports>
          <hierarchy>
            <sliverdef>
              <nodeSlivers>
                <nodeSliver name="SP00">
                  <vports><vport name="vip1"/><vport name="vip2"/>
                      <vport name="vip3"/></vports>
                  <instance subtype="KVM" type="SlowPath_VM">
                    <resources><!-- ノードスリバーの計算資源 -->
                      <resource keyword="cpu" value="1"/><!-- CPUの個数 -->
                      <resource keyword="arch" value="x86_64"/>
                      <resource keyword="memory" value="2048"/>
                    </resources>
                    <params><!-- 以下はあらかじめ用意された VM イメージ -->
                      <param keyword="bootImage"
                        value="http://…/KVM_Ubuntu910Server32.img"/>
                    </params>
                  </instance>
                </nodeSliver>
              </nodeSlivers>
              <linkSlivers>…</linkSlivers>
            </sliverdef>
            <structure>… </structure>
            <params>…</params>
          </hierarchy>
        </nodeSliver>
        …
        <!-- 以下はネットワーク収容の定義 -->
        <nodeSliver name="NACE" type="prog">
          <vports><vport name="vp1"/></vports>
          <hierarchy>
            <sliverdef>
              <nodeSlivers>
                <nodeSliver name="Pipe">
                  <vports><vport name="vp1" /></vports>
                  <instance type="pipe">
                    <interfaces>
                      <interface name="VLAN100" type="VLAN" >
                        <params><param key="VLANID" value="100" />
                          <param key="PORT" value="1/7" />
                        </params>
                      </interface>
                    </interfaces>
                  </instance>
                </nodeSliver>
              </nodeSlivers>
            </sliverdef>
          </hierarchy>
        </nodeSliver>
        …
        <nodeSliver name="AGW" type="agw">
          <vports><vport name="vp1"/></vports>
        </nodeSliver>
      </nodeSlivers>
    </sliverdef>
    <structure><!-- 以下はノードスリバーとリンクスリバーとの結合の定義-->
      …
      <bind name="w101"><!-- w101 は Node1 と LS01 を結合する -->
        <vport portname="vp1" slivername="Node1"/>
        <vport portname="e1" slivername="LS01"/>
      </bind>
      …
    </structure>
  </slicespec>
  <!-- 以下はノードスリバーの物理ノードへの配置 (マッピング) -->
  <mapping slice="IPEC_Slice_000" vnetwork="NICTtestbed">
    <amap node="AGW2" vnode="agw-f0"/>
      …
    <amap node="Node2" vnode="rp-nh0"/>
  </mapping>
</slice-design>
```

図 4 スライス定義の例 (一部省略)

2.4 開発者にとっての接続とその実装

前節のスライス定義によってスライスを生成したとき，ノードスリバーどうしの接続やスライスと外部ネットワークとの接続が開発者にどうみえるかを説明する．まず，**ノードスリバーどうしの接続**についてのべる．ノードスリバーがふくむ全 VM において eth0 は管理用のインターフェースであり，**Z プレーン** [Kat 12] という開発者用のネットワークに接続される．開発者はスライスを生成したあと，このインターフェースを使用して VM を設定する．すなわち，eth0 から ssh などの手段でログインして他のインターフェース (eth1, eth2 など) に必要な設定をおこない，eth0 経由でプログラムをロードし実行する．

図 3 のノードスリバーについていえば，それらがふくむ VM はそれぞれ 3 個のネットワーク・インターフェースをもち，リンクスリバーが接続されている．これらのインターフェースには eth1, eth2, eth3 という名称があたえられる．スライス上で IP を使用するときも IP アドレスは自動的につけられるわけではないので，たとえばつぎのようなコマンドを使用して IP アドレスを設定する必要がある．

```
ifconfig eth1 192.168.1.1
```

上記のようにインターフェースの名称は開発者が指定するわけではないので，どのリンクスリバーがどのインターフェースに対応しているかを開発者が確認する必要がある．IP を使用するときも IP アドレスをつけるまえに接続を確認する必要があるが，そのためには **IP に依存しない導通テストツール** (たとえば [Kan 12d]) を使用するのが容易である．

VM 上で動作させるプログラムは，IP ベースならば IP ルーティングにしたがってリンクスリバーを選択する．非 IP であればインターフェース名 eth1, eth2, … によって，そのインターフェースに接続されたリンクスリバーを選択し，同時にパケット転送先のノードスリバーを選択する．eth1 などという名称からは，ここでは Ethernet プロトコルを使用しなければならないような印象をあたえるが，promiscuous mode を使用すればリンクスリバー上で任意のプロトコルが使用できる (最初のビットから自由にフォーマットをきめられる) [Kan 10] (仮想化基盤上ではこのパケットを GRE トンネル (GRE/IP) によって転送するが，開発者がこの実装法を意識する必要はほとんどない)．

つぎに，**外部ネットワークとの接続**についてのべる．前記のようにこのとき通常は物理ポートと VLAN ID とを指定するので，それにしたがって開発者が NACE から外部ネットワークへの配線をする (または既存の配線にしたがってそれらをスライス定義に指定する) 必要がある．VLAN ID やポートの指定や接続にエラーがあっても自動的には検出されないので，注意ぶかく指定・接続する必要がある．たとえば，スライス定義においてキーワード “VLANID” のかわりに “VLAN” を指定するなどしても，仕様上，エラーは検出されず VLAN ID の設定はされないので，動作しない．

3. ノードスリバー間接続の実現

前記のようにノードスリバーどうしの接続においては，開発者がリンクスリバーを指定するときはインターフェース名を使用する．この章においてはそれがどのようにして GRE トンネルのパラメタにマップされ，高速なデータ変換が実現されるかをしめす．その説明のためにまず VNode の構成を記述する．

3.1 VNode の構成

VNode は 3 種類の要素によって構成されている (**図 5**) [Nak 12].

- **VNode マネジャ** (VNode Manager, VNM) は VNode 全体の管理をおこなう．仮想化基盤全体を管理するドメイン・コントローラからの指示にもとづいて，プログラマやリダイレクタを制御する．

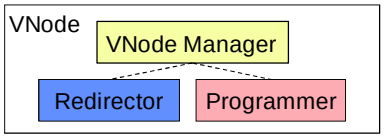


図 5 VNode の構成

- **プログラマ** (Programmer) はノードスリバーを実装する．すなわち VNode マネジャからの指示にしたがってノードスリバーに対応するスローパスやファストパスを生成したり，他のプログラマから送信されたパケットをリダイレクタ経由で受信してノードスリバーを構成するスローパスやファストパスに転送したりする．
- **リダイレクタ** (Redirector) はリンクスリバーを実装する．すなわち VNode マネジャからの指示にしたがって GRE トンネルを生成・削除したり，パケットを他の VNode や他の種類のノードから受信してそれを加工してプログラマに転送したり，プログラマからのパケットを加工して他の VNode に送信したりする．

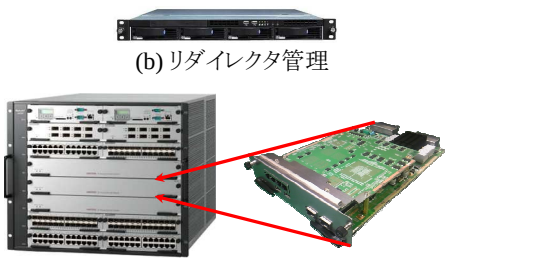
転送処理をおこなうリダイレクタとノード内処理をおこなうプログラマとを分離したことがこのモデルの特徴のひとつである．これらを分離することによってプログラマとリダイレクタとのあいだに明確なインターフェース (内部インターフェース) が定義され，第 1 に VNode の外部インターフェースと内部インターフェースとを独立にあつかうことが可能になる (後述) とともに，第 2 にプログラマとリダイレクタとを独立に発展 (進化) させられる [Nak 12] ようになっている．

リダイレクタはさらに 3 個の要素から構成される (**図 6**) [Kan 12c].

- **リダイレクタ本体 (スイッチ部)**: リダイレクタ本体はシャーシ型の Ethernet スイッチである．データ・プレーンにおいては，VNode や他の IP ルータなどで構成されるネットワークに対しては IP ネットワーク機能によりパケット転送をおこなう．また，プログラマ-リダイレクタ間は VLAN 機能によりパケット転送をおこなう．
- **リダイレクタ管理部 (リダイレクタ制御ソフトウェア)**: スライスを設定する機能にくわえて，リダイレクタ全体の装置管理機能をもつ．VNode マネジャが制御 API を使用してリダイレクタを管理する．現在の実装においては，リダイレクタ管理部はスイッチにそとづけされたサーバを使用して実現されている．
- **サービス・モジュール・カード** (SMC): SMC はリダイレクタ本体に内蔵できる，ネットワーク・プロセッサを搭載したカードである．SMC は内部データ形式と外部データ形式とのあいだの変換機能をもつ．SMC は複数枚，搭載することができる．

3.2 リダイレクタを中心とするマッピングの記述

リダイレクタは仮想リンクであるリンクスリバーを物理リンクにマップする．すなわち，他の VNode またはプログラマからパケットを受信すると，それに対応するリンクスリバーが接続された (他の VNode またはプログラマ内の) ノードスリバーの対応するポートのアドレスを付


(b) リダイレクタ管理

(a) リダイレクタ本体　(c) SMC (Service Module Card)

図 6 リダイレクタの構成

加して送信する．

プログラマとリダイレクタとの独立な進化を可能にするため，VNode 外部と内部のデータ表現およびマッピングは独立にきめられ，それらをつなぐためにパケットの受信時にデータ変換をおこなっている．すなわち，リダイレクタは，VNode を構成する他の要素である VNode マネジャやプログラマと協力することにより，つぎの 3 つの機能をはたしている．これらの機能に関する詳細は Kanada et al. [Kan 12c] が記述している．

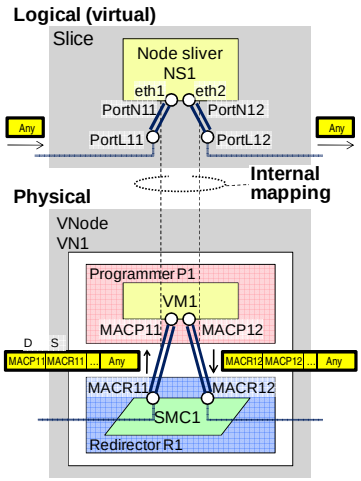


図 7　スライス上の属性と仮想化基盤の内部属性との対応

- **内部モデルマッピング — リンクスリバー (モデル) とプログラマ-リダイレクタ間パス (実装) との対応づけ**：リダイレクタはプログラマと協力してリンクスリバーとプログラマ-リダイレクタ間をむすぶ Ethernet または VLAN による内部パスとを 1 対 1 対応させる (図 7)．ノードスリバーから出力されるパケットにおいては開発者が指定したインターフェース (図 7 の eth2) はノードスリバーのポート (PortN12) に対応するので，それが内部パスの MAC アドレスの対にマップされ，それをふくむ MAC ヘッダがプログラマにおいてパケットに付加されてリダイレクタに転送される．一方，ノードスリバーに入力されるパケットにおいては，リンクスリバーの反対側でパケットを出力する際にきめられたポートに対応する MAC アドレス (MACR11, MACP11) が入力パケットに付加されてリダイレクタからプログラマに転送される．このアドレスはノードスリバーのポート (PortN11) に対応するので，この MAC ヘッダがはがされたパケットがそのポートに対応するインターフェース (eth1) にとどけられる．
- **外部モデルマッピング — リンクスリバー (モデル) と VNode 間のパス (実装) との対応づけ**：リダイレクタは，VNode マネジャと協力して，リンクスリバーと VNode 間のパスとを 1 対 1 対応させる (図 8)．ノードスリバーから出力されたパケットには，指定されたリンクスリバーのポート (PortL12, PortL21) に対応する VNode 間パスの IP アドレスおよび GRE キーをふくむ GRE ヘッダが付加され，IP ネットワーク上で転送される．宛先の VNode に到達すると，その IP アドレスと GRE キーに対応するポート (PortL21) に対

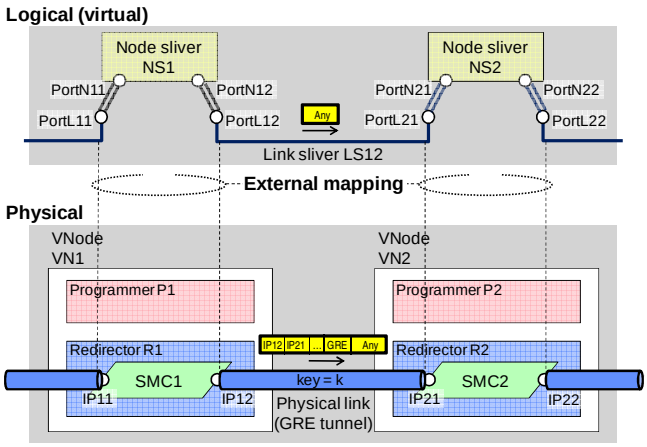


図 8　スライス上の属性と仮想化基盤の外部属性との対応

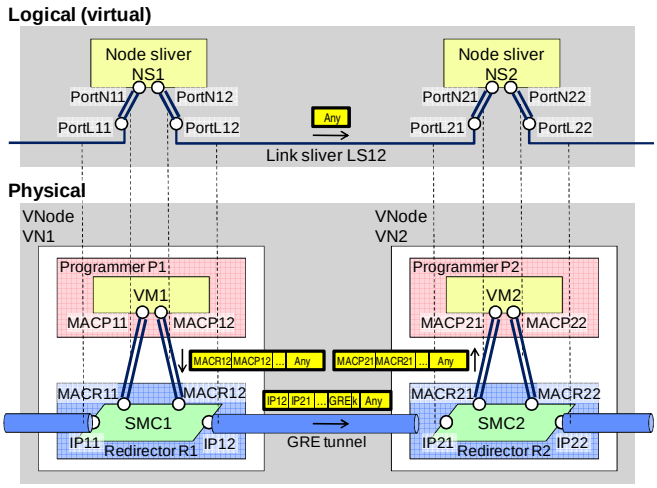


図 9　スライス上の通信と仮想化基盤上の通信との対応

応するプログラマに転送される．

- **外部表現-内部表現間のデータ変換**：上記のように，VNode の内部と外部とでデータ表現 (データ形式) はことなっている．そのため，それらのあいだの変換が必要である (図 9)．外部から入力されたパケットは内部のデータ表現に変換してプログラマに転送する必要があり，プログラマからとどいたパケットは外部のデータ表現に変換して外部に出力する必要がある．外部からのパケットについては IP アドレスと GRE キーをキーとして対応する MAC アドレスを検索するしてパケット・ヘッダをつけかえる．また，内部からのパケットについては MAC アドレスをキーとして IP アドレスと GRE キーを検索してパケット・ヘッダをつけかえる．この変換もリダイレクタの機能であり，1 台の SMC によって双方向の合計で最高 10 Gbps のスループットを実現している．

上記のように，リダイレクタにおいては外部モデルマッピングと内部モデルマッピングとを独立に指定することができ，両者をつなぐために変換をおこなっている．このように 2 つのマッピングを分離することによって，つぎの 2 つの利点が生じている [Kan 12c].

- **VNode 実装の隠蔽**：プログラマおよびリダイレクタの実装を VNode 外部にみせないことによって，それら，とくにプログラマの実装を変更したり，あらたな実装を追加したりしても，VNode 外部に影響をあたえないようにすることができる．
- **VNode 間パス実装がプログラマに影響をあたえないこと**：VNode 間のパスを実装するために，GRE トンネル，VLAN，光パスなど，さまざまな方法を使用することができる (ただし，現在までに実装されているのは GRE トンネルだけである)．このように VNode 間パスの実装が変化しても，あるいはさまざまな種類のパスが追加されても，リダイレクタがその差を吸収することができるので，プログラマには影響をあたえないようにすることができる．

一方，2 つのマッピングの分離により，つぎのような欠点も生じる．

- **外部表現と内部表現の変換オーバーヘッド**：マッピングの分離により外部表現と内部表現の変換が必要になる．この変換が高速にできないとボトルネックが生じる可能性がある．この変換は外部表現が GRE トンネルのような場合だけでなく，内部表現と外部表現がともに Ethernet である場合にも通常は必要になる．すなわち，外部表現における MAC アドレスと内部表現における MAC アドレスはことなる (これらを同一にすると上記の利点がうしなわれる) ので，変換が必要である．このような変換機能は通常

の Ethernet スイッチにはないので，あらたな実装が必要である．

4. 外部ネットワーク接続の実現

外部ネットワークへの接続においては，開発者が指定した物理ポートや VLAN ID がそのまま使用されるので，マッピングは直接的である．したがって，ここでは接続のために使用する NACE の構造とデータ変換の方法だけについてのべる．

4.1 NACE の構成

ネットワーク収容装置 (NACE) は仮想化基盤と外部のネットワークとを接続する装置であるから，端末を接続するためのアクセス・ゲートウェイ (AGW) と同様に**ゲートウェイ**の一種である．しかし，その実装においては VNode をベースとし，その構造も VNode にちかい．図 4 にしめした VNode の構成において VNode マネジャとリダイレクタが機能拡張され，プログラマだけが**疑似プログラマ**におきかえられている [Kan 12b]．

疑似プログラマは通常のプログラマのようなデータ・プレーン処理の機能をもたず，管理プログラムだけで構成されている．NACE はもともと VNode の一種として企画された．VNode はプログラマビリティをもつべきである．しかし現在の NACE にはプログラマビリティがなく，仮想化基盤と外部ネットワークのパケット形式のちがいを吸収する以外は単純にパケットを通過させる機能しかもたない．そのため，この機能を “Pipe” という仮想的なノードスリバーとみなして，将来 NACE がプログラマブルになったときにはそのかわりに他のプログラマブルなノードスリバーを指定すればよいようにしている．

4.2 NACE におけるデータ変換

NACE においては仮想化基盤におけるパケットの外部表現つまり GRE ヘッダつきのフォーマットと外部ネットワークにおける VLAN によるフォーマットとの変換をおこなう (**図 10** に 1 対 1 収容時の変換とマッピングをしめす [Kan 12b])．リダイレクタが内蔵する SMC をこの変換用に使用している．ネットワーク・プロセッサを使用することによって最高 10 Gbps での外部ネットワーク収容を実現している．

5. 結論

ネットワーク仮想化基盤においては，開発者が XML によって記述したスライス定義をポータルから投入することによってスライスを生成することができる．この論文においては開発者がどのように接続先を参照・指定し，それがどのように VNode および NACE で変換され通信が実現されるかを記述した．すなわち，第 1 にスライス定義がふくむノードスリバーにおいてはインターフェース名によってパケット転送先を自由に選択し，Ethernet や IP にしばられない任意のプロトコルを使用して最高 10 Gbps で通信できる．プログラマ-リダイレクタ間ではリンクスリバーが VLAN パスにマップされ，VNode 間ではそれが GRE トンネルにマップされる．また，第 2 に NACE を使用すれば開発者がスライス定義上で指定した物理ポートと VLAN ID によって外部ネットワークと接続することができ，最高 10 Gbps で通信できる．これらの高速通信は SMC の使用により実現されている．

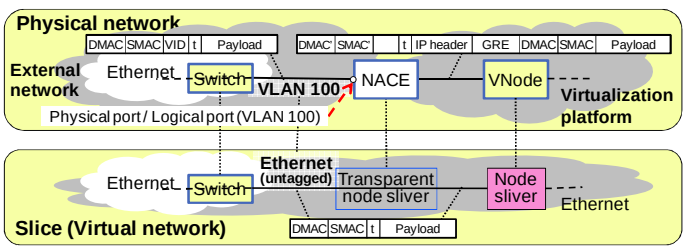


図 10 NACE におけるスライス上の通信と基盤上の通信との対応

今後の課題としては，リダイレクタにおいて増設 SMC が使用可能なアーキテクチャを実現し，また内部データ形式と外部データ形式との対応の単純化をはかることによってプログラマと外部とのスループットを向上させること，リダイレクタとプログラマとの内部インターフェースの単純化をはかることなどがある．

謝辞

VNode は東大，NTT，NEC，富士通研，日立による仮想化ノード・プロジェクトにおいて開発したものであり，その参加者各位に感謝する．とくに，この報告の内容は NTT の 高原 厚，高橋 紀之，NEC の 元木 顕弘，林 偉夫，富士通研の 的場 一峰，阿比留 健一，アラクサラの 木谷 誠 の各氏が中心となって開発した機能に依存しているので感謝する．この報告の内容は情報通信研究機構 (NICT) の委託研究「新世代ネットワークを支えるネットワーク仮想化基盤技術の研究開発」(課題ア) の研究開発成果をふくむ．

参考文献

[Cho 09] Chowdhury, N. M. M. K. and Boutaba, R., “Network Virtualization: State of the Art and Research Challenges”, *IEEE Communications Magazine*, Vol. 47 , No. 7, pp. 20–26, July 2009.

[Du 12] Ping Du, Maoke Chen, and Nakao, A., “OFIAS: A Platform for Exploring In-Network Processing”, *TridentCom 2011*, *LNICST 90*, pp. 142–151, Springer, 2012.

[Due 12] Duerig, J., Ricci, R., Stoller, L., Strum, M., Wong, G., Carpenter, C., Fei, Z., Griffioen, J., Nasir, H., Reed, J., and Wu, X., “Getting Started with GENI: A User Tutorial”, *ACM SIGCOMM Computer Communication Review*, Vol. 42, No. 1., pp. 72–77, January 2012.

[Far 00] Farinacci, D., Li, T., Hanks, S., Meyer, D., and Traina, P., “Generic Routing Encapsulation (GRE)”, RFC 2784, IETF, March 2000.

[Fel 07] Feldmann, A., “Internet Clean-Slate Design: What and Why?”, *ACM SIGCOMM Computer Communication Review*, Vol. 37, No. 3, pp. 59–74, July 2007.

[Kan 10] 金田 泰, 中尾 彰宏, “仮想化ノードを使用した非 IP プロトコル開発法と経験”, 電子情報通信学会インターネットアーキテクチャ (IA) 研究会, 2010-12-17.

[Kan 12a] Kanada, Y. and Nakao, A., “Development of A Scalable Non-IP/Non-Ethernet Protocol With Learning-based Forwarding Method”, *World Telecommunication Congress 2012* (*WTC 2012*), March 2012.

[Kan 12b] Kanada, Y., Shiraishi, K., and Nakao, A., “High-performance Network Accommodation into Slices and In-slice Switching Using A Type of Virtualization Node”, *IARIA 2nd International Conference on Advanced Communications and Computation* (*Infocomp 2012*), October 2012.

[Kan 12c] Kanada, Y., Shiraishi, K., and Nakao, A., “Network-Virtualization Nodes that Support Mutually Independent Development and Evolution of Components”, *IEEE International Conference on Communication Systems* (*ICCS 2012*), November 2012.

[Kan 12d] “ネットワークで IP がつかえないときの導通テスト・ツール”, http://www.kanadas.com/program/2012/05/_ip.html

[Kan 12e] Kanada, Y., Shiraishi, K., and Nakao, A., “Network-resource Isolation for Virtualization Nodes”, *17th IEEE Symposium on Computers and Communications* (*ISCC 2012*), July 2012.

[Kat 12] 片山 陽平, 山本 猛仁, 山田 一久, 中尾 彰宏, “ネットワーク仮想化基盤における仮想ネットワーク管理モデルに関する一検討”, 電子情報通信学会 ネットワーク仮想化時限研究会, 第 3 回, 2012-3, http://www.ieice.org/~nv/06-nv20120302-katayama.pdf

[McK 08] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J., “OpenFlow: Enabling Innovation in Campus Networks”, *ACM SIGCOMM Computer Communication Review*, pp. 69–74, Vol. 38, No. 2, April 2008.

[Nak 10] Nakao, A., “Virtual Node Project — Virtualization Technology for Building New-Generation Networks”, *NICT News*, No. 393, pp. 1–6, Jun 2010.

[Nak 12] Nakao, A., “VNode: A Deeply Programmable Network Testbed Through Network Virtualization”, *3rd IEICE Technical Committee on Network Virtualization*, March 2012, http://www.ieice.org/~nv/05-nv20120302-nakao.pdf

[Tur 07] Turner, J., Crowley, P., Dehart, J., Freestone, A., Heller, B., Kuhms, F., Kumar, S., Lockwood, J., Lu, J., Wilson, M., Wiseman, C., and Zar, D., “Supercharging PlanetLab — High Performance, Multi-Application, Overlay Network Platform”, *ACM SIGCOMM Computer Communication Review*, Vol. 37, No. 4, pp. 85–96, October 2007.