情報処理学会論文誌 Vol.56 No.1 378–390 (Jan. 2015)

推薦論文

OpenFlowと協調する仮想マシン環境におけるリアルタイム通信基盤の設計

鈴木 健一[1,a)] 宮田 宏[1] 佐藤 未来子[1] 並木 美太郎[1]

受付日 2014年4月7日, 採録日 2014年10月8日

概要：近年，ネットワーク QoS（Quality of Service）の重要性が増しており，特に，仮想マシン上で動作する音声通話や機器制御アプリケーションなどのリアルタイム通信に対して，End-to-End のリアルタイム性を保証することが求められている．しかし，仮想化のオーバヘッドやネットワークの輻輳などによりパケットの転送が遅延し，リアルタイム性が損なわれるという課題がある．そこで本研究では，リアルタイム通信制御機構を追加した仮想マシンモニタとプログラマブルな特性を持つ OpenFlow の協調動作により，これらの課題を解決するリアルタイム通信基盤「RTvNIC システム」を提案し，仮想マシン間の End-to-End のリアルタイム通信を実現する．RTvNIC システムでは，仮想 NIC（Network Interface）に QoS 補助機能を付加し，さらに，仮想マシンモニタと OpenFlow が連携することで，各通信のデッドライン時間に応じた動的なネットワーク QoS 制御を行う．提案方式について実装，評価を行い，転送時間のデッドラインをそれぞれ 5 ms と 10 ms に設定した 2 つのフローに対して，擾乱フローがある状態でも最大 3.6 ms と 8.7 ms の遅延時間に抑え，仮想マシン間のリアルタイム通信の保証を実現した．

キーワード：仮想マシン，OpenFlow，QoS，リアルタイム通信

Design of a Real-time Network in Virtual Machine Environment with OpenFlow

Kenichi Suzuki[1,a)] Hiroshi Miyata[1] Mikiko Sato[1] Mitaro Namiki[1]

Received: April 7, 2014, Accepted: October 8, 2014

Abstract: It is important for real-time network such as VoIP, video conference, and industrial network to control QoS. Guaranteed real-time communication for applications on Virtual Machine (VM), especially, is an important issue when the delay-sensitive applications creates in network. Network delay caused by overhead of virtualization will seriously impact the performance of applications using real-time communication on VM. We introduce design and implementation of RTvNIC System on VMM for real-time network communication. To achieve real-time guarantees, VMM and OpenFlow manage bandwidth and priority control for avoiding deadline miss. The evaluation shows that the worst-case communication delay of two flows which have each deadline 5 ms and 10 ms between VMs via Ethernet kept within 3.6 ms and 8.7 ms by this system.

Keywords: Virtual Machine, OpenFlow, QoS, real-time communication

1. はじめに

近年，仮想マシンモニタ（VMM）層にリアルタイム性を持たせ，仮想マシン（VM）上でリアルタイムアプリケーションを動作させる研究 [1], [2] の進展にともない，音声通話やテレビ会議システム，制御通信などリアルタイム通信を必要とするアプリケーションへの仮想マシン技術の適用が求められている．遅延を対象としたリアルタイム通信で

[1] 東京農工大学
Tokyo University of Agriculture and Technology, Koganei, Tokyo 184–8588, Japan
[a)] ksuzuki@namikilab.tuat.ac.jp

本論文の内容は 2013 年 7 月のマルチメディア，分散，協調とモバイル（DICOMO2013）シンポジウム 2013 にて報告され，マルチメディア通信と分散処理研究会主査により情報処理学会論文誌ジャーナルへの掲載が推薦された論文である．

は，パケットがある一定時間（デッドライン時間）以内に宛先 VM へ到達することが必要である．

通信のデッドライン保証に必要なネットワークの品質を確保する QoS（Quality of Service）の代表的な技術として，DiffServ（Differentiated Services）[3] や RSVP（Resource Reservation Protocol）/IntServ（Integrated Services）[4], [5] があげられる．しかし，従来までの DiffServ や RSVP/IntServ の運用手法は主に静的な設定をされたネットワーク構成を想定しており，VM 環境で頻繁に発生する VM の生成や消滅，マイグレーションにともなうネットワーク構成の変化に追随して QoS を変更することが難しい．

さらに VM 環境特有の課題として，仮想化にともなうオーバヘッドにより VMM 層において通信の遅延が発生するという課題がある [6], [7], [8]．同一物理マシン上で複数の VM が動作するため，VMM は物理 NIC（Network Interface）や物理 CPU など物理的なリソースを各 VM に割り当てるための機能を持つが，その割当てのスケジューリングにおいて発生する待ち時間により通信が遅延する．VMM 層の課題に対しては，先行研究において，Cheng ら [1] が各 VM に必要なリアルタイム性に応じて仮想 CPU スケジューリングとパケットスケジューリングを行うことで送信パケットのジッタを低減させる方式を提案している．しかし，この方式は送信パケットのジッタ保証を目的としているため，各パケットの End-to-End のデッドライン保証については課題がある．

そこで，本研究では，ネットワーク構成や QoS を動的に変更できる SDN（Software Defined Network）技術の 1 つである OpenFlow [9] と VMM の協調動作により，これらの課題を解決することを目的とする．本研究で提案する方式では，VMM がゲスト OS の送受信するパケットをすべて制御できることを利用し，仮想 NIC に QoS 補助機能を持たせる．さらに，VMM と OpenFlow とが連携することで動的なネットワーク QoS 制御を行う．本論文では，これらの機能を備えたリアルタイム通信基盤「RTvNIC システム」を提案し，異なる物理マシン上で動作する VM 間での End-to-End のリアルタイム性を保証する．

2. RTvNIC システムの概要

本章では，本論文で提案する VM 間のリアルタイム通信を実現する通信基盤である RTvNIC システムの技術課題と目標，およびシステムの概要を述べる．

2.1 課題と目標

VM 間のリアルタイム通信を実現するための課題として，まず，VMM 層における送信パケットに対するパケットスケジューリング遅延の問題があげられる．複数の VM が同一物理マシン上で動作しているため，VMM では，ゲスト OS の送信したトラフィックの合計が物理 NIC の帯域幅を超える場合，パケットスケジューリングが発生する．既存技術ではパケットスケジューリングにおいて各パケットの遅延時間は考慮されないため，転送時間のデッドラインを超える可能性がある．また，パケット受信時の仮想 CPU スケジューリング遅延の課題がある．VMM 層がネットワークからパケットを受信してから，パケットの宛先 VM の仮想 CPU に実行権が割り当てられ，受信処理を開始するまでの待ち時間により，パケットの遅延が発生する．これらの遅延は，物理 NIC の帯域がボトルネックとなり発生するため，将来的に物理 NIC の帯域が大きくなった場合でも，VM からのトラフィックの増加とともに発生する可能性があり，本質的な課題である．

また，End-to-End のデッドライン保証においては，パケット送信側 VM がネットワーク内における転送遅延時間を見積もれないため，End-to-End でデッドラインを守れる保証がないという課題がある．DiffServ による QoS 制御が行われても，あるフローの転送遅延時間はネットワークの輻輳状態の影響を受けて変化するため送信前に見積もることは難しい．このため，ネットワークを介した End-to-End の通信のリアルタイム性を確保できる保証がない．

本論文で提案する RTvNIC システムでは，VMM の仮想 NIC に対してリアルタイム通信制御機能を付加した“リアルタイム仮想 NIC（RTvNIC）”と OpenFlow の連携によりこれらの課題を解決する．RTvNIC は，VMM 層の内部におけるパケットと VM の優先制御により，VMM 層のパケット遅延の課題を解決する．OpenFlow では，RTvNIC からの要求に応じた QoS 確保により，End-to-End の通信のリアルタイム性保証の課題を解決する．さらに，デッドラインミス検出時には VMM と OpenFlow の連携による通信の優先度変更により，同一優先度の他の通信によるパケット遅延の課題を解決する．これらにより，異なる物理マシンで動作する VM 間の通信のリアルタイム性を保証する．

2.2 全体構成

RTvNIC システムの全体構成を**図 1** に示す．RTvNIC システムは，異なる物理マシンで動作する VM 間の通信のリアルタイム性を保証するために VMM 層のリアルタイム通信制御機構とネットワークの OpenFlow コントローラにより，VMM 層とネットワークの双方の通信制御を行う．リアルタイム通信制御機構は，次の 2 つの制御部を持つ．

- RTvNIC 制御部
- OpenFlow 制御部

RTvNIC 制御部では，VMM 層におけるリアルタイム性を確保するため，ゲスト OS が送信するパケットに対してデッドラインに応じたパケットスケジューリングを行うことで送信遅延時間を制御する．また，パケットの受信時

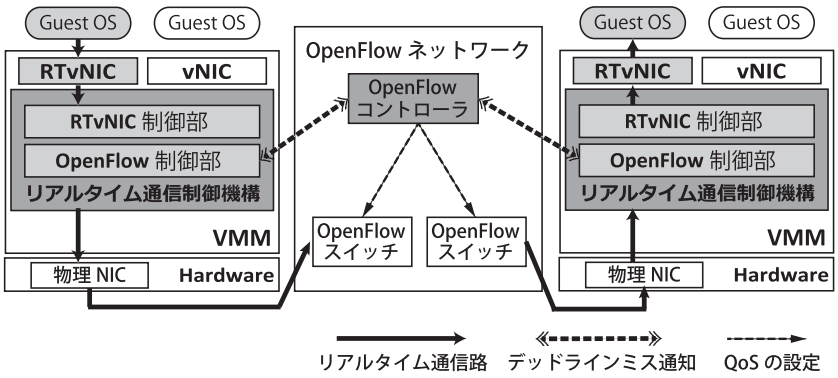

図 1 RTvNIC システムの全体構成
Fig. 1 The overall structure of the RTvNIC System.

には，パケットの宛先 VM に対する実行待ち時間を少なくするため，RTvNIC を持つ VM を優先する仮想 CPU スケジューリングを行う．さらに，受信パケットのデッドライン時間以内の到着を判定し，デッドラインミスを検出した場合には，次のデッドラインミスを回避するために OpenFlow 制御部にデッドラインミス時の情報を通知する．

OpenFlow 制御部は，OpenFlow コントローラと RTvNIC 制御部が QoS に関する情報を通知する際の仲介の役割を担う．RTvNIC 制御部は OpenFlow 制御部を介して，RTvNIC 制御部で検出したデッドラインミス情報など QoS に関する情報を OpenFlow コントローラと交換する．

OpenFlow コントローラでは，各リアルタイム通信の資源予約，優先制御，帯域制御とネットワークの状況の監視を担う．OpenFlow コントローラは，VMM の OpenFlow 制御部や OpenFlow スイッチと QoS 制御のための情報を交換する．それらの情報を用いて，OpenFlow コントローラがリアルタイム通信のために帯域の確保，リアルタイム通信のパケットに対する DiffServ に基づいたマーキングによる優先度の付与，デッドラインミスに応じた優先度を設定の切替えを行うことで，VM 間の QoS 保証を行う．この詳細については，3.5 節で示す．

これらの機能を VMM と OpenFlow コントローラの連携により活用することで，VMM 層と OpenFlow におけるパケット優先制御を動的に行う．連携はリアルタイム通信の開始要求とデッドラインミスを契機に行われる．ゲスト OS が VMM に対してリアルタイム通信の開始を要求したときは，VMM が OpenFlow コントローラに対してリアルタイム通信の経路決定と，経路上の OpenFlow スイッチに対する QoS 設定を行うよう通知する．次に VMM 層がリアルタイム通信のパケットのデッドラインミスを検知したときは，OpenFlow コントローラに対してデッドラインミスを起こしたフローに設定された帯域やスケジューリングのパラメータなどの再設定を要求する．さらに，デッドラインミスを起こしたパケットの送信元 VMM に対しても通知を行い，パケットを送信したゲスト OS が指定したデッドラインミスハンドラを起動することができる．

3. RTvNIC システムの設計

本章では，RTvNIC システムの設計方針，VMM と OpenFlow の連携のインタフェースとリアルタイム性の保証手法について述べる．

3.1 設計方針

RTvNIC システムの設計方針を述べる．本研究の目標である End-to-End のリアルタイム性を確保するため，宛先と送信元 IP アドレスの組でフローを区別し，そのフローごとにリアルタイム性を確保する．また，フローの区別には必要に応じて受信側のポート番号も対象とする．これにより，同じ DSCP（Differentiated Services Code Point）値を持つパケットをすべて同様に扱う Diffserv では実現できない，各フローごとのリアルタイム性を確保する．IP アドレスは仮想 NIC と 1 対 1 で対応しているため，2 つの仮想 NIC 間の通信のリアルタイム性を確保することと同じである．IP アドレスを用いて通信を区別することから，RTvNIC システムでは IP/Ethernet プロトコルを用いたパケットを対象とする．

音声通信では，一定の割合でデッドラインミスが発生することが許容されるソフトリアルタイムであるが，制御通信などはデッドラインミスが致命的な損害が生じるハードリアルタイムである．このようにアプリケーションにより求めるリアルタイム性が異なるため，通信の利用目的に合わせ保証するリアルタイム性を選択できるようにする．デッドラインに応じてパケットごとの転送遅延を測定するためには，VMM 間の時刻同期が必要となるが，本研究では NTP，IEEE1588 PTP，GPS など目的とする精度を持った既存技術の利用を想定する．

また，RTvNIC システムを用いない VM や OS も，RTvNIC システムを用いた VM と通信可能な設計にすることで，相互接続性を確保する．ただし，それらの通信はリアルタイム性の保証の対象外とする．

各 VM の仮想 CPU スケジューリングに対してもリアルタイム性を提供する必要があるが，Cheng ら [1] の EDF と Credit アルゴリズムを用いたランキューの手法を流用し，VM ごとに必要な実行権割当てのリアルタイム性を確保することで解決する．詳細は 3.2 節および 3.4 節で述べる．

3.2 リアルタイム通信路の設計と VM スケジューリング

RTvNIC システムにおけるリアルタイム通信は，「リアルタイム通信路」と呼ばれる論理的な通信路によって実現する．リアルタイム通信路の両端はリアルタイム向け仮想 NIC（RTvNIC）であり，送受信側の RTvNIC にそれぞれ割り当てられた IP アドレスと受信側のポート番号の組により識別される．RTvNIC システムでは，このリアルタイム通信路をリアルタイム性を確保するフローの単位として扱い，この通信路上で転送される片方向のパケットに対して End-to-End のリアルタイム性を保証する．ゲスト OS は，このリアルタイム通信路の設定を VMM へ要求することで通信のリアルタイム性を確保することができる．なお，リアルタイム通信路上のパケットをリアルタイムパケットと呼び，リアルタイム性を確保しないノンリアルタイムパケットと区別する．

RTvNIC システムでは，リアルタイム通信路の保証するリアルタイム性として，ソフトリアルタイムとハードリアルタイムに対応するために，ゲスト OS 上のアプリケーションが必要に応じて次の 2 つのリアルタイム性保証を選択できるようにする．

(1) デッドライン保証型

パケットのデッドライン時間を保証する通信路を提供する．動画像や音声の配信など一定量のデッドラインミスやパケットロスが許容されるソフトリアルタイムなパケットを対象とし，最悪遅延時間を保証するのではなく，できるだけ多くのパケットがデッドライン時間以内に到着することを目標としている．デッドライン保証型のリアルタイム通信路の設定には，最低帯域，デッドライン時間，デッドラインミス処理を指定する．転送遅延ができるだけ少なくなるようにスケジューリングされるが，デッドラインに対して余裕のない通信路は，他の余裕がある通信路よりも優先的にパケットスケジューリングされる．

(2) 資源予約型

最低帯域と最悪遅延時間が予測可能な通信路を提供する．モーション制御などデッドライン時間以内にパケットが到着しないと，致命的なダメージが発生する，もしくはパケットの価値がなくなるようなハードリアルタイムなパケットを対象としている．資源予約型のリアルタイム通信路の設定には，最低帯域，最大バースト長，デッドライン時間，デッドラインミス処理を指定する．この情報をもとにネットワーク上の帯域に空きがあるか，最悪遅延時間の理論値は要求されたデッドライン時間以内であるかを確認したうえで，ネットワーク資源を確保する．ネットワーク資源に余裕がない場合，資源予約型リアルタイム通信路は設定できない．この場合，リアルタイム通信路は確保されずゲスト OS に対してエラー通知を行う．

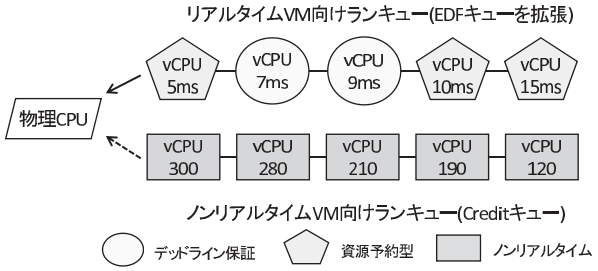

図 2 VM 実行のリアルタイム性を保証するダブルランキュー
Fig. 2 The double runqueue design for assigned physical CPU core to both real-time and non-real-time VMs.

パケットスケジューリングとネットワークにおける遅延はこの 2 種類のリアルタイム通信路で解決するが，2.1 節で述べたように，VM の仮想 CPU への適切な実行権の割当ても必要であるため，Cheng ら [1] の手法のランキューを拡張する．本研究で適用する VMM のランキューを**図 2** に示す．

Cheng ら [1] は，VMM の 1 つである Xen [10] を対象として仮想 CPU スケジューリングによる I/O 遅延を削減するため，EDF と Credit アルゴリズムを用いた 2 つのランキューを持つ仮想 CPU スケジューラを提案している．この手法では，リアルタイム性を必要とする VM に対しては，パケット送信処理のための周期的な実行とパケット受信などの外部イベントの発生時の実行のリアルタイム性を確保するために，EDF ランキューを用いることで実行待ち時間を削減した．また，リアルタイム性を必要としない VM は Credit ランキューを用い，実行時間の公平性を確保した．

本システムでは，この EDF ランキューを拡張し，リアルタイム VM 向けランキューとして，デッドライン保証型と資源予約型の RTvNIC を持つ VM をこのランキューで管理することで，2 つの型のリアルタイム VM に対するパケット送受信処理の実行待ち遅延時間を保証する．また，Credit ランキューを拡張し，ノンリアルタイム VM 向けランキューとして，通常の仮想 NIC を持つリアルタイム性の必要ない VM をこのランキューで管理することで，実行時間の公平性を確保する．

3.3 RTvNIC の設計

本研究で用いる VMM は，既存 VMM の仮想 NIC に対して，パケットの優先制御機能とデッドラインミス判定をするためにパケットへのマーキング機能，パケットの転送

遅延測定機能，ゲスト OS との協調機能を追加したものである．本節では，RTvNIC へ追加したこれらの機能について述べる．

3.3.1 優先制御とデッドラインミス判定のためのマーキング機能

ネットワーク内においてパケットを処理する際に，リアルタイムパケットとノンリアルタイムパケットを区別する必要がある．リアルタイム通信路の定義からは，宛先・送信元 IP アドレスを確認することで区別可能である．しかし，ネットワーク上のすべてのスイッチがすべてのリアルタイム通信路情報を持ち，そのつど IP アドレスを見て判定する場合，スイッチが保持するデータ量が増加し，1 パケットの処理にかかる負荷が大きくなる．そこで，RTvNIC システムでは IP ヘッダの TypeOfService（ToS）フィールドにリアルタイム通信路ごとに固有の値を識別子として書き込み，これを利用することでこの問題を解決する．

本設計では，特に，このリアルタイム通信路ごとに固有の識別子を DiffServ（Differentiated Services）の DSCP 値と対応付けることにより，ネットワークを転送する際に，既存の DiffServ による QoS を流用できる形にした．送信側 VM から送信されたパケットの IP ヘッダ内の送信元 IP アドレス，宛先 IP アドレスを参照し，現在設定されているリアルタイム通信路の送信元・宛先 IP アドレスの組と一致していれば，IP ヘッダの DSCP フィールドにリアルタイム通信路ごとに固有に割り当てられた DSCP 値を設定することで，DiffServ ネットワーク上での優先制御を実現する．

3.3.2 デッドラインミス判定のための転送遅延時間の管理

パケットの転送遅延測定機能で計測した時刻情報に基づき，リアルタイムパケットが送信元 VMM から宛先 VMM に届くまでの転送遅延時間が，指定されたデッドライン時間以内であることを確かめることで，End-to-End での通信のリアルタイム性が確保されていることを確認することができる．このパケットのデッドライン判定を行うためには，各パケットの送信側 VMM での送信時刻を宛先 VMM へ通知する必要がある．パケットのデッドライン時間を送信パケットと別のパケットで通知する方法では，通知パケットの到着遅延やロスによりデッドライン時間判定ができなくなる可能性がある．このため，デッドライン時間は送信パケットに付加して，宛先 VMM への通知を行う．

パケットに付加するデッドライン時間は相対デッドライン時間と絶対デッドライン時間が考えられる．相対デッドライン時間を付加する場合は，送信パケットがネットワーク上の各スイッチを通るたびに，付加された相対デッドライン値からスイッチ内に滞在していた時間を減算し，値が 0 以下になったときデッドラインミスをしたと判定する．この方法では，送信元と宛先の時刻同期が必要ない点で優れているが，各遅延時間を減算して間接的にデッドライン時間を求めているため，実際のパケット転送時間とずれが生じる可能性がある．一方，絶対デッドライン時間を付加する場合は，送信側・受信側の時刻を同期させ，同期した時計の時刻とパケットに付加された時刻を比較することにより，デッドラインを判定する方法である．この方法は実際のパケット転送時間を直接計測することができるが，計測の精度は時刻同期の精度に依存する．

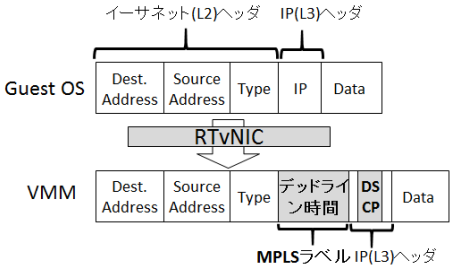

図 3 リアルタイムパケットフォーマット
Fig. 3 The format of the real-time packet.

本研究では，計測誤差の発生要因が多く誤差の見積りが難しい相対デッドライン方式ではなく，絶対デッドライン方式を利用することとし，絶対デッドラインをパケットに付加し，同期した時刻によりデッドライン判定を行う．

パケットへのデッドライン時間の付加は，MPLS（Multi-Protocol Label Switching）のラベリングを流用する．このときのパケットフォーマットの構造を**図 3** に示す．パケットの L2 ヘッダと L3 ヘッダの間に MPLS ヘッダを挿入するフレームモードを利用する．送信側 VMM において，リアルタイムパケットに対して，MPLS ヘッダの Label フィールド（20 bit）に絶対デッドラインを書き込んだ MPLS ラベルを挿入することで，デッドライン時間の付加を行う．また，このとき IP ヘッダにおける DSCP 値の変更も行う．なお，RTvNIC システムで付加した MPLS ヘッダはデッドライン時間を送信側 VMM から受信側 VMM へ通知するためのもので，この値によるラベルスイッチングは行わない．MPLS によるラベルスイッチングが必要な場合は，デッドライン時間を含む MPLS ラベルを挿入されたパケットに対して，さらに任意のラベルを付加することで対応する．

MPLS ヘッダは 32 bit（4 byte）の長さであるため，リアルタイムパケットのパケット長はすべて元のパケットサイズより 4 byte 大きくなる．RTvNIC システムの MPLS ラベリングは VMM によって，ゲスト OS に対して隠蔽されているため，ゲスト OS が送信する Ethernet フレームに MPLS ラベルを付加すると，Ethernet の最大フレームサイズである 1,522 バイトを最大 4 byte 超過する可能性がある．本システムでは，VMM が Ethernet フレームの最大フレームサイズを超過するパケットを受け取った場合，VMM がゲスト OS に対して，ICMP プロトコルの PATH MTU Discovery（type 3:code 4）を送ることで，ゲスト OS の送信する最大フレームサイズを 1,518 byte に設定するこ

表 1 リアルタイム通信路確保：要求メッセージ
Table 1 The request messages for reservation of the real-time virtual circuit.

Name	Description
dst_ip	宛先 IP アドレス
src_ip	送信元 IP アドレス
dst_port	宛先ポート番号（オプション）
min_rate	最低帯域
max_burstlen	最大バースト長
deadline_time	パケット転送のデッドライン時間
deadline_handler	デッドラインミスハンドラ
rtpath_type	確保するリアルタイム通信路の型

表 2 リアルタイム通信路確保：応答メッセージ
Table 2 The response messages for reservation of the real-time virtual circuit.

Name	Description
result	リアルタイム通信路の確保結果
rtpath_id	確保したリアルタイム通信路の識別番号

表 3 デッドラインミス通知メッセージ
Table 3 The notification messages of deadline miss.

Name	Description
rtpath_id	リアルタイム通信路の識別番号
exceed_time	デッドラインを超過した時間
ip_id	デッドラインミスをしたパケットの IP ヘッダの Identification 値

とで，MPLS ラベルの挿入に対応する．

3.3.3 ゲスト OS と VMM 間の協調

本項では，ゲスト OS が VMM に対してリアルタイム通信開始要求を行うためのインタフェースについて述べる．リアルタイム通信の開始にあたり，ゲスト OS は**表 1** に示すパラメータを持つ，リアルタイム通信路確保要求メッセージを VMM に対して発行する．VMM は OpenFlow ネットワークにリアルタイム通信の開始の可否について問合せを行った後，その結果を**表 2** に示す応答メッセージとしてゲスト OS に通知する．

リアルタイム通信開始後，送信したパケットがデッドラインミスを起こした場合，VMM からゲスト OS に対して**表 3** のデッドラインミス通知メッセージが発行される．このメッセージによりゲスト OS はデッドラインミス処理を行うことができる．

3.4 リアルタイム性の保証

VMM からリアルタイム通信路設定要求メッセージを受け取った OpenFlow コントローラは，要求されたリアルタイム性を保証できるだけネットワーク資源に余裕があるか確認し，余裕がない場合はリアルタイム通信路の設定が失敗したことを VMM へ通知する．要求された通信路が帯域を確保できないなどリアルタイム性を保証できないときは要求を拒否することにより，リアルタイム性を確保できないフローが生じないようにする．また，VMM からデッドラインミス発生通知メッセージを受け取った OpenFlow コントローラは，デッドラインミスをしたリアルタイム通信路に対して，QoS の再確保を行う．

アプリケーションに必要なリアルタイム性に応じて，3.2 節で述べたデッドライン保証型と資源予約型のどちらか一方のポリシを選択する．このポリシを実現するために，VMM におけるパケットスケジューリングに対して，デッドライン保証型では EDF（Earliest Deadline First）アルゴリズム [11]，資源予約型では WFQ（Weighted Fair Queuing）を適用する．

デッドライン保証型リアルタイム通信路を実現するためのパケットスケジューリングアルゴリズムとして，デッドライン時間が最も近いパケットから優先的に送信する EDF アルゴリズムを用いる．VMM 層のパケットスケジューリングでは，パケットが任意の時刻に到着し，ある程度のデッドラインミスが許容できるという特徴を持つため，動的な優先度の変更が可能な EDF アルゴリズムを選択した．EDF パケットスケジューリング対象のあるフローがデッドラインミスをした場合，そのフローの優先度を上げることで，遅延を削減し，デッドラインミスを回避する．

リアルタイム性を保証しないノンリアルタイムパケットについては，遅延時間が大きいと TCP 通信のスループットが低下するなど悪影響が考えられるため可能な限り遅延を小さくすることが望ましい．このため，デッドラインミスが発生しない範囲においてはリアルタイムパケットよりもノンリアルタイムパケットを先に送信する．さらに，リアルタイム通信が優先されるため，リアルタイム通信の帯域が増加するとノンリアルタイム通信の帯域が圧迫されてしまうという問題がある．この問題については，各リアルタイム通信に適切な帯域制限を設けることで対処する．

また，資源予約型リアルタイム通信路を実現するためのパケットスケジューリングアルゴリズムとして，WFQ アルゴリズムを用いる．パケットスケジューリングにおけるあるフローの最大遅延時間は一般に式 (1) で表される．

$$Delay = \frac{(b-M)}{R} \times \frac{(p-R)}{(p-r)} + \frac{(M+C_{tot})}{R} + D_{tot} \quad (1)$$

ただし，

r: Token Bucket rate
b: Token Bucket Size
p: Peak Rate
M: Maximum Packet Size
R: Service Rate
C_{tot}: Total Rate-Dependent Error Term
D_{tot}: Total Rate-Independent Error Term
$0 \le r \le R \le p, 0 \le M \le b, 0 \le C_{tot}, 0 \le D_{tot}$

これより，WFQ を用いているフローがデッドラインミスをした場合，そのフローの Service Rate（式 (1) の R）を増加させることで遅延を削減し，デッドラインミスを回避する．

仮想 CPU のスケジューリングのリアルタイム性については，3.2 節で述べたように，送受信処理を行うための実行権割当ての遅延時間を保証するため，デッドライン保証型，資源予約型をともにリアルタイム VM 向けランキューで管理する．このスケジューリングにおいて，デッドライン保証型と資源予約型がスケジューリングで競合した場合は，より厳密なリアルタイム性を保証する必要のある資源予約型の仮想 CPU を優先する．

3.5 VMM と OpenFlow の連携

本節では，ネットワークのリアルタイム性の保証を実現するための OpenFlow と VMM の連携とそのインタフェースについて述べる．

3.5.1 本システムにおける OpenFlow の連携内容

提案手法における OpenFlow の役割は，リアルタイム通信に対する優先度と帯域の調停と，デッドラインミス情報の収集である．本システムは，OpenFlow コントローラ，VMM の RTvNIC 制御部と OpenFlow 制御部から構成されているが，OpenFlow コントローラは調停とデッドラインミス情報の収集を行う．VMM では OpenFlow コントローラで監視された情報をもとに，RTvNIC の制御を行う．OpenFlow 制御部は OpenFlow コントローラと RTvNIC 制御部のインタフェースをつかさどり，OpenFlow コントローラからの DSCP 値を受信して RTvNIC 部に渡す．また，RTvNIC 制御部から QoS およびデッドラインミス情報を受信し，OpenFlow コントローラに送信する．RTvNIC 制御部は，OpenFlow コントローラからの DSCP 値をもとに，3.3 節で述べた VMM 内のパケット転送の QoS 制御を行う．

OpenFlow コントローラは調停のために，リアルタイム通信の受付制御，優先制御，帯域制御を行う．受付制御では，リルタイム通信の過剰な設定によりリアルタイム性が阻害されることを防ぐため，OpenFlow コントローラは，現在設定されている帯域情報をもとに，RTvNIC 制御部からの新規リアルタイム通信の受け入れの可否を判断する．優先制御では，各パケットに対してデッドラインミス時間に応じた優先制御を実現するため，OpenFlow スイッチを流れるパケット中の DSCP 値を，リアルタイム通信を識別するタグとして扱い，OpenFlow スイッチに対して DSCP 値に対応した優先度を設定する．帯域制御では，輻輳によるリアルタイム性の阻害を防ぐため，優先制御と同じ機構を用いて，OpenFlow スイッチに対して DSCP 値に対応した帯域制限を設定する．

また，OpenFlow コントローラは，デッドラインミス情報を収集する役割も担っている．OpenFlow コントローラでは，VMM からデッドラインを超過した場合に送られてくるデッドラインを超過した情報と DSCP 値を収集し，デッドライン処理に用いる．デッドライン処理については，次項で述べる．

表 4 リアルタイム通信路設定：要求メッセージ
Table 4 The request messages to create the real-time virtual circuit.

Name	Description
qos_param	ゲスト OS からのリアルタイム通信路確保要求メッセージの内容

3.5.2 VMM と OpenFlow のデッドライン制御

前項で述べた機能により OpenFlow と VMM はリアルタイム通信路の確保時とデッドラインミス時に情報のやりとりを行う．調停によるリアルタイム通信路の確保では，まず，VMM が OpenFlow コントローラに対してデッドライン時間など要求するリアルタイム性を通知する．OpenFlow コントローラではその要求を満たすリアルタイム通信路を確保可能か判定し，可能であれば経路上のスイッチに帯域予約を設定する．不可能であれば，確保失敗通知を送信側 VMM へ返す．

デッドラインミス時には，デッドラインミス判定を行っている VMM が OpenFlow コントローラとデッドラインミスをしたパケットの送信 VMM へデッドラインミスの発生を通知する．一度デッドラインミスが発生したネットワークの QoS 設定のままでは，デッドラインミスが頻発する可能性があるため，3.4 節で述べたリアルタイム性保証より，デッドライン保証型では EDF スケジューリングの優先度を上げ，資源予約型では帯域を大きくする．また，OpenFlow コントローラへの要求によりネットワーク上での通信の優先度を上げることで通信路の転送遅延を削減しデッドラインミスが再発しないように，ネットワークの QoS の変更を行う．

3.5.3 VMM と OpenFlow 間で連携するためのインタフェース

ここでは，VMM と OpenFlow が連携し，ネットワーク資源の確保とデッドラインミス対処を行うためのインタフェースを述べる．まず，ゲスト OS からリアルタイム通信路確保要求メッセージを受け取った VMM は，**表 4** のリアルタイム通信路設定要求メッセージを OpenFlow コントローラに対して通知する．OpenFlow コントローラは，要求されたリアルタイム通信路が設定可能かを確認した後，その結果を**表 5** の応答メッセージとして通知する．また，受信側 VMM でデッドラインミスを検知した場合，受信側 VMM から OpenFlow コントローラと送信側 VMM に対して**表 6** のデッドライン発生通知メッセージを発行する．これらのメッセージは「制御通信型」として，最高優先度で

転送される．

3.6 本方式のソフトウェア層への適用

本提案手法を適用した場合に，ゲスト OS 上のアプリケーションがリアルタイム通信を利用するために必要な，VM 上のゲスト OS，ゲスト OS 上のアプリケーション，OpenFlow に関する変更について述べる．

本提案方式では，ゲスト OS 全体で同一のリアルタイム制御を行うとき，ゲスト OS とアプリケーションに修正を加えずにリアルタイム通信を利用できる．QoS 設定やデッドラインミスに対応する処理などのリアルタイム通信の制御は VMM に隠蔽されており，QoS の設定値や処理方法を VM の起動時のパラメータとして設定することで，特定の VM 間でアプリケーションのリアルタイム通信を実現する．

アプリケーションごとのリアルタイム通信の制御は，ゲスト OS とアプリケーションに変更を加えることで実現できる．ゲスト OS では，アプリケーションと VMM が，表 1 で示すリアルタイム通信に関する情報を交換するためのシステムコールを提供するように修正する．アプリケーションはこのシステムコールを用い，リアルタイム通信に関する情報を VMM に伝えることで，リアルタイム通信を利用できる．次に，アプリケーションが必要に応じてデッドラインミスに対応した処理を行えるように，表 3 で示すデッドラインミス情報を通知するためのアップコールを提供するように修正する．また，アプリケーションでは，上記のゲスト OS の変更で述べたシステムコールとアップコールを利用するよう変更する．

OpenFlow コントローラについては，SDN として 3.5.1 項で述べた機能を実装する必要がある．また，プロトコルに関しては，3.3.2 項で述べたように標準のパケットフォーマットを流用するため変更は不要であるため，OpenFlow スイッチはそのまま利用できる．

表 5 リアルタイム通信路設定：応答メッセージ
Table 5 The response messages to create the real-time virtual circuit.

Name	Description
result	リアルタイム通信路の設定結果
rtpath_id	設定したリアルタイム通信路の識別番号

表 6 デッドラインミス発生通知メッセージ
Table 6 The deadline miss messages.

Name	Description
rtpath_id	リアルタイム通信路の識別番号
exceed_time	デッドラインを超過した時間
ip_id	デッドラインミスをしたパケットの IP ヘッダの Identification 値

4. 実装と評価

本章では，RTvNIC システムの実装と評価について述べる．RTvNIC システムの評価として，パケットスケジューリングに EDF を用いて QoS を確保した OpenFlow ネットワークを実装し，パケットのリアルタイム性の評価を行った．

4.1 実装

RTvNIC システムの実装には，VMM として Linux のカーネルモジュールである KVM/QEMU-KVM(1.2.0) を利用した．このうち，QEMU-KVM のエミュレートする仮想 NIC である e1000 にパケットへの MPLS ラベルの挿入と DSCP 値の付加を実装し，これを RTvNIC とした．OpenFlow コントローラには Trema(0.4.6) を利用した．OpenFlow スイッチには仮想スイッチソフトウェアの Open vSwitch(1.9.0) を利用し，PC ルータとして利用した．VMM において，EDF パケットスケジューリングを実現するために，Linux カーネルの機能である Traffic Control を改変し，EDF パケットスケジューラと WFQ パケットスケジューラを実装した．さらに，実運用においては一定量のデッドラインミスやパケットロスが許容されるようなリアルタイム通信では，パケットがデッドラインミスするごとに優先度の変更を行うと，他のフローとの調停に支障が出る可能性がある．このため，優先度変更の実行タイミングをパラメータ化して，アプリケーションで要求するタイミングに応じて優先度変更を実行できる機能を備えた．

仮想 CPU スケジューラは理想的なスケジューリングが行われる状態を想定し，RTvNIC を持つ VM の仮想 CPU を高優先度とする優先度スケジューラを実装した．仮想 CPU スケジューリングによる擾乱を避けるため，各 VM の仮想 CPU 数は 1 つとし，各仮想 CPU をそれぞれ別の物理 CPU コアにバインドして実験を行った．また，評価に用いた PC の仕様を**表 7** に示す．

4.2 評価

次の 4 つの実験を行い RTvNIC システムの有効性を評価した．

- 実験 1：リアルタイムパケットスケジューリングの評価
- 実験 2：VM 間通信の転送遅延時間の評価
- 実験 3：動的な優先度変更によるデッドラインミスの

表 7 評価環境
Table 7 Evaluation environment.

CPU	Intel Core i7 980（6core/3.40 GHz）
Memory	24 GB
NIC	1 Gbps

低減効果の評価

- 実験 4：RTvNIC システムによるオーバヘッドの評価

次項から評価の詳細について述べる．

4.2.1 リアルタイムパケットスケジューリングの評価

VMM 層の内部のリアルタイムパケットスケジューリングについて，EDF と WFQ を適用したときのパケット遅延の特徴を検証するため，次に示す実験 1 を行った．実験 1 では，ゲスト OS の送信したパケットが，VMM 層の送信キューにエンキューされてから，リアルタイムパケットスケジューラによってデキューされるまでのパケットのキュー内滞在時間を計測した．ゲスト OS からは下記に示す，デッドラインを設定した 2 つのフローと輻輳状態とするためのノンリアルタイムフローを送信した．

- フロー A：2 Mbps，Deadline 5 ms
- フロー B：2 Mbps，Deadline 10 ms
- フロー NRT：1.1 Gbps，Deadline 指定なし

フローの生成には，ネットワークのベンチマークである iperf による UDP パケット通信を用い，帯域は先行研究 [1] の評価において動画像の帯域として用いられている 2 Mbps に設定した．なお，UDP パケットのサイズは 1,502 byte である．評価結果を**表 8** に示す．

EDF パケットスケジューリングはデッドラインをミスしない範囲で可能な限りフロー NRT を先に送るため，転送遅延時間はデッドライン時間に近いが，デッドラインミスが発生しない範囲で転送を行えた．WFQ は帯域を割り当てているため，フロー NRT の量にかかわらず一定以下の遅延時間となる．WFQ を用いた場合は，遅延時間の保証のために帯域の予約を行っていることから帯域の効率的な利用が難しい面があるため，WFQ を適用するフローは障害発生を知らせる緊急のエラー通知など厳密な遅延時間保証を行う必要のある少数のフローが対象となる．

以後の評価は，より多くのフローに適用可能で効率的な帯域の利用とデッドライン保証の両立が見込める EDF により実現される，デッドライン保証型のリアルタイム通信路を対象として評価を行った．

表 8 実験 1：送信パケットの送信キュー内滞在時間
Table 8 The experiment 1: The waiting time of packets in the send queue.

	EDF		WFQ	
遅延時間	フロー A	フロー B	フロー A	フロー B
最大 (us)	4,412	9,303	1,047	7,639
最小 (us)	3,043	7,032	0.4	0.5
平均 (us)	3,201	8,749	274	1,640

4.2.2 VM 間通信の転送遅延時間の評価

提案手法が VM 間の通信のリアルタイム性保証に対して有効であることを示すため，デッドライン保証型のリアルタイム通信路を対象として VM 間の通信の転送遅延時間について評価を行った．

(1) 評価方法

評価に用いた VMM・ネットワーク構成を**図 4** に示す．VM の構成は，各物理マシンに RTvNIC を持つ VM を配置し，この VM 間で OpenFlow ネットワークを介したリアルタイム通信を行う．本節の評価では，RTvNIC1 から RTvNIC4 まで，vNIC1，vNIC2 を利用し，RTvNIC5 は用いない．OpenFlow ネットワークには 1 台の OpenFlow スイッチを配置し，VMM においては EDF スケジューリングによる QoS を実施する．

この構成を用い，End-to-End の転送遅延時間として，送信側の RTvNIC がパケットを送信したときから，受信側の RTvNIC に到達するまでの転送遅延時間を計測する実験 2 を行った．この実験では，送信側 VM から受信側 VM に対して iperf による UDP 通信を行った．計測フローは実験 1 で示したフローを利用し，フロー A は RTvNIC1 から RTvNIC3 へ，フロー B は RTvNIC2 から RTvNIC4 へ向かうよう設定した．また，輻輳状態を起こす NRT フローは vNIC1 から vNIC2 に対して送信する．この評価では，計測フローに対して RTvNIC システムのリアルタイム性保証を適用する場合としない場合で計測した．

なお，この構成では End-to-End のパケットの遅延時間を計測するために，物理マシン A と B の間で NTP（Network Time Protocol）による時刻同期を行った．時刻同期精度を高めるため，物理マシン A と B の間に NTP による時刻同期専用に LAN を構築した結果，同期誤差は最大でも

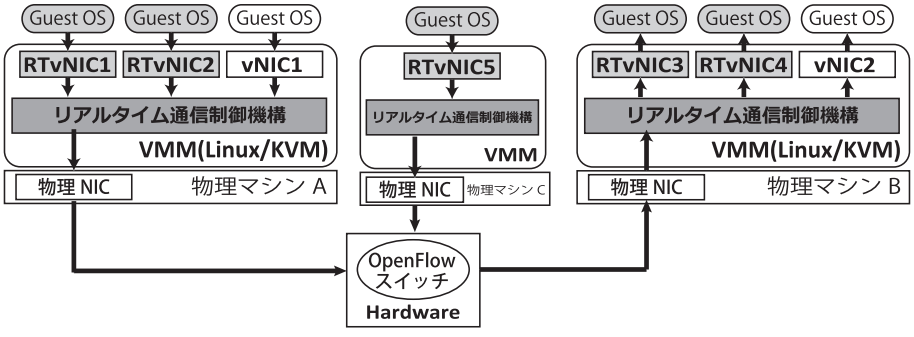

図 4 評価用ネットワーク構成
Fig. 4 The network environment for evaluation.

表 9 実験 2：別物理マシン構成の VM 間通信の遅延時間
Table 9 The experiment 2: End-to-End delay time between VMs on the different physical machines.

	RTvNIC あり		RTvNIC なし	
遅延時間	フロー A	フロー B	フロー A	フロー B
最大 (ms)	3.6	8.7	13.9	14.0
最小 (ms)	3.1	8.1	6.3	7.8
平均 (ms)	3.4	8.4	13.5	13.5

表 10 実験 2：同一物理マシン構成の VM 間通信の遅延時間
Table 10 The experiment 2: End-to-End delay time between VMs on the same physical machine.

	RTvNIC あり		RTvNIC なし	
遅延時間	フロー A	フロー B	フロー A	フロー B
最大 (ms)	2.1	8.5	49	38
最小 (ms)	0.9	1.7	4.0	5.5
平均 (ms)	1.5	3.2	9.8	10.2

400 ns 程度の収まったため，本評価で用いるには十分な精度である．

さらに，時刻同期が理想的に行われた場合を想定した評価を行った．物理 NIC を 2 つ持つ物理マシンを利用し，同一物理マシン上に RTvNIC1 から RTvNIC4，vNIC1 から vNIC2 を持つ VM をすべて配置し，物理 NIC を介して送信されたパケットが外部の OpenFlow スイッチを介して，もう一方の物理 NIC にループバックして戻ってくる構成で，別物理マシンを利用した実験と同様にフロー A，フロー B，フロー NRT を用いて計測を行った．

(2) 実験結果と考察

実験 2 の計測結果を**表 9**，**表 10** に示す．別物理マシン構成での実験では，最大転送時間について，デッドラインをそれぞれ 5 ms と 10 ms を設定したフロー A とフロー B に対して，RTvNIC を適用しない場合ではフロー A で 13.9 ms，フロー B で 14.0 ms とデッドライン時間を超えているのに対して，RTvNIC を適用した場合では，フロー A で 3.6 ms，フロー B で 8.7 ms となり，最大転送時間をデッドライン時間以内に抑えることができた．このとき，RTvNIC を適用しない場合のパケットスケジューリングである FIFO アルゴリズムではデッドラインを超えるスケジューリングが存在していたのに対し，RTvNIC を適用した場合の EDF では，各パケットのデッドラインを考慮したスケジューリングを行ったことにより，デッドラインを守るパケットスケジューリングとなっている．別物理マシンと同一物理マシンのいずれも同様の結果となったことから，NTP の誤差から見ても十分信頼できる結果である．

実験 2 の RTvNIC を適用した場合のパケットは，デッドライン時間を管理の MPLS ラベル（4 byte）を付加したため，実験 2 の RTvNIC を適用しないパケットと比較して 4 byte フレームサイズが大きい．このため，パケット長が長くなったことにより物理 NIC のパケット送信処理時間は増加しているはずであるが，1 Gbps の帯域を持つ物理 NIC が 4 byte 分のパケットを送信するのにかかる時間は，4 byte/1 Gbps = 0.032 us であるため，最大遅延時間増加のオーバヘッドは小さい．

4.2.3 動的な優先度変更によるデッドラインミスの低減効果の評価

提案手法のデッドラインミスに応じた優先度の変更を評価するため，実験 3 を行った．デッドラインミスが発生した場合，VMM と OpenFlow の協調動作によるネットワークの通信の優先度の動的な変更で End-to-End の通信のデッドラインミスを低減させることができることを示す．

実験 3 では，4.2.2 項の評価で示したフローに加え，新たに 1 つのリアルタイム通信を追加した場合を想定して，提案手法の優先度変更について基礎的な評価を行う．実際のネットワークでは，大規模なイベントなどによるリアルタイム通信のアクセス集中によりトラフィック量が増大している状況など QoS 確保の難しくなる異常系についても対応が必要となる．提案手法では，新規リアルタイム通信に対する受付制御と優先制御によりこの問題に対応する．リアルタイム通信の設定数の増加によりリアルタイム通信の予約帯域の合計が物理帯域を超えてしまう状況では，すでに確保済みのリアルタイム通信を優先し，新規リアルタイム通信の設定要求を拒否するという受付制御により QoS を保証する．また，ネットワークの輻輳により QoS のための制御通信パケットがロスする状況では，パケットロスが発生しないよう制御通信パケットをつねに最高優先度で送信するという優先制御により，QoS 制御を維持する．

(1) 実験方法

実験 3 では，4.2.2 項の評価の構成に加えて，RTvNIC5 から RTvNIC4 へ向かうフロー C：1 Gbps を用いる．フロー C は，QoS 制御されたフロー A，B より高優先度に設定されたフローである．このため，フロー C の投入時にはフロー C が優先され転送される．

計測項目は実験 2 と同様に，RTvNIC 間のパケットの転送遅延時間を計測する．評価の手順として，まず，実験 2 の状態で計測を開始する．次に，フロー C を追加することで OpenFlow スイッチにおいて輻輳によるパケット遅延が発生し，デッドラインミスが多発する状態とする．このとき，デッドラインミスの発生を検知した VMM が OpenFlow に対して，デッドラインの発生しているフロー A とフロー B の優先度を上げるよう要求が出される．この優先度変更によるパケット遅延時間の変化を評価した．なお，4.1 節で述べた優先度実行のタイミングはパラメータとして，本実験ではフロー C が加わって一定の状況になった 20 秒を設定した．

(2) 実験結果と考察

実験 3 の計測結果について，最大転送遅延時間を 100 ms

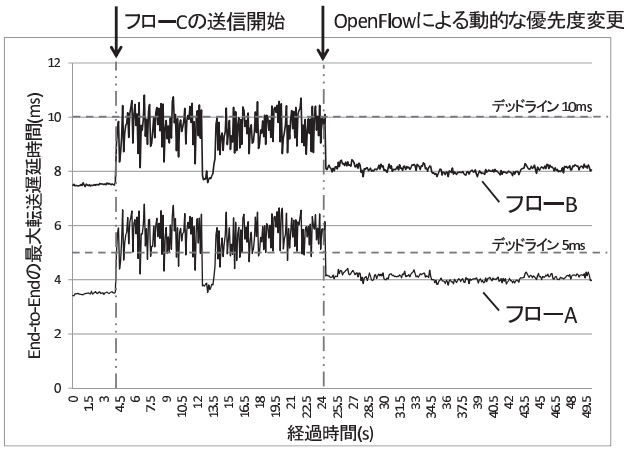

図 5 実験 3：動的な優先度の変更の評価
Fig. 5 The experiment 3: Evaluation for the dynamic priority changes.

表 11 リアルタイム情報の付加/除去の処理時間
Table 11 Overhead time to add and delete the real-time information to packets.

	付加	除去
最大処理時間 (us)	80	72
最小処理時間 (us)	23	14
平均処理時間 (us)	32	23

表 12 1 パケットあたりの EDF スケジューラの処理時間
Table 12 Overhead time for EDF scheduling for one packet.

最大処理時間 (us)	160
最小処理時間 (us)	92
平均処理時間 (us)	101

ごとに平均をとってグラフにしたものを**図 5** に示す．計測開始時は実験 2 と同様のフロー A とフロー B，フロー NRT を送信している状態で，経過時間 4 秒後にフロー C の送信を開始した．経過時間 24 秒後に，VMM が OpenFlow に対してフロー A とフロー B の優先度を上げるよう要求を行った．優先度の高いフロー C の影響によりフロー A とフロー B のデッドラインミスが増加したが，デッドラインミスに基づく優先度の変更により，フロー A とフロー B の優先度を上昇させたことで，デッドラインミスを低減させることができた．

4.2.4 RTvNIC システムのオーバヘッドの評価

RTvNIC システムによるオーバヘッドを検証する実験 4 を行った．本実験では，パケット遅延の増加要因となる，パケットへの MPLS ラベルの挿入/除去と EDF パケットスケジューリングの処理時間を計測した．計測結果を**表 11**，**表 12** に示す．また，受信パケットのジッタについても計測を行った．計測結果を**表 13** に示す．ジッタとしては送信側と受信側の双方のジッタが存在するが，今回は RTvNIC システムによるパケットスケジューリングと帯域制御による影響を見るために，それらのジッタを含む受信側の到着パケットに対するジッタを計測した．RTvNIC システムはミリ秒オーダのリアルタイム性の保証を目指しているため，このパケットマーキングと EDF パケットスケジューリングの処理時間，ジッタは十分に小さい．

表 13 受信パケットのジッタ
Table 13 The jitter of received packets.

	フロー A	フロー B
ジッタ (us)	548	10,026
分散	580	9,656

4.3 評価結果のまとめ

実験 1 では，EDF と WFQ のパケットスケジューリングにおけるパケット遅延を検証し，それぞれデッドラインを守る送信ができたことから，EDF と WFQ 双方でリアルタイム性確保に対する有効性を示した．実験 2 では，別物理マシンの VM 間で End-to-End のパケット転送において，EDF のパケットスケジューリングによる異なるデッドラインを持つ 2 つのフローの転送遅延時間を計測した．この結果，2 つのフローともデッドライン時間以内に転送できたことから，EDF パケットスケジューリングの End-to-End の遅延に対する有効性を示した．実験 3 では，擾乱フローを追加し，ネットワークの遅延によりデッドラインミスが多発する状態において，フローの転送遅延時間を計測した．この結果，VMM と OpenFlow の協調動作による通信の優先度変更によりデッドラインミスが低減したことを示し，デッドラインミスに対処する本手法の有効性を示した．最後に，実験 4 では，RTvNIC システムのオーバヘッドを検証し，保証するデッドライン時間に対して非常に小さいオーバヘッドでリアルタイム通信制御を実現したことを示した．以上より，提案している RTvNIC システムによる VMM と OpenFlow のパケット制御で，VM 間の End-to-End のリアルタイム通信の保証を実現した．

5. 関連研究

仮想マシン環境での End-to-End の QoS 保証の手法について，Hyper-V [12] や vSphere [13]，Xen [14] においては，仮想 NIC に対する帯域制限と仮想スイッチによる仮想ネットワークの構成を組み合わせることにより，End-to-End で QoS 確保を実現している．しかし，これらの手法では，QoS として帯域の保証を目的としており，デッドラインを考慮しない転送がされるため，通信のリアルタイム性が保証されない可能性がある．これに対して，本提案手法では各通信のデッドラインと遅延の状況をもとに動的に優先制御と帯域制御を行うことで，通信のデッドラインを守り，リアルタイム性の保証を実現している．

Sharma ら [15] の研究では，多項式時間アルゴリズムを用い，多数の多様な QoS 要求に対する経路設定と帯域予約を最適化する手法について提案している．しかし，これ

を仮想マシン環境に適用する場合，VMM における遅延が課題となる．また，Cheng ら [1] の研究では，各 VM に必要なリアルタイム性に応じて仮想 CPU スケジューリングとパケットスケジューリングを行うことで，送信パケットのジッタを低減させる手法を提案している．しかし，ジッタ保証を目的としているため，デッドラインについては保証ができないという課題がある．これらの先行研究に対して，本提案手法では，VMM において各通信のデッドラインをパラメータとした優先制御を行うことで，VMM 層のパケット転送のデッドライン保証を実現し，VM 間の End-to-End のリアルタイム通信を可能としている．

6. おわりに

本論文では，OpenFlow と VMM が協調動作することにより，異なる物理マシン上で動作する VM 間のリアルタイム通信を実現する通信基盤である RTvNIC システムを提案した．また，OpenFlow と VMM の協調動作のインタフェースの設計と，通信のリアルタイム性を確認，保証するための機構の設計を示した．そして，パケットスケジューラとして EDF と WFQ アルゴリズムを用いたスケジューラを実装し，RTvNIC システムによるリアルタイム通信の実験を行った．実験の結果，ノンリアルタイムパケットで輻輳させたネットワーク中でも，RTvNIC システムによりリアルタイム性を確保したパケットが優先的に送信され，設定したデッドライン時間以内に到達することを確認した．この結果より，RTvNIC システムによるパケットスケジューリングとネットワークにおける優先制御によるパケット遅延時間の低減効果を示した．さらに，VMM と OpenFlow の協調動作によるネットワーク QoS の動的な変更により，パケットに設定したデッドラインが保証されることを示した．

今後は，リアルタイム通信のフロー数や仮想マシン数のスケーラビリティ，ノンリアルタイム通信への影響，VM に対する仮想 CPU のコア数や割当て方による挙動の変化の検証など，実運用を想定した環境で，提案手法の適用可能な範囲を明らかにする評価，分析を行っていく．

参考文献

[1] Cheng, L., Wang, C. and Di, S.: Defeating network jitter for virtual machines, *Proc. Utility and Cloud Computing* (*UCC*), *2011 4th IEEE International Conference*, pp.65–72 (2011).

[2] 太田貴也，Daniel Sangorrin，本田晋也，高田広章：組込みマルチコア向け仮想化環境における性能低下抑止手法，情報処理学会研究報告，EMB，組込みシステム，Vol.2012-EMB-27, No.12, pp.1–8 (2012).

[3] Blake, S., Black, D., Carlson, M., et al.: An architecture for differentiated services, IETF RFC Standard 2475 (1998).

[4] Zhang, L., Deering, S., Estrin, D., et al.: RSVP: A new resource reservation protocol, *Network*, Vol.7, No.5, pp.8–18, IEEE (1993).

[5] Braden, R., Clark, D. and Shekner, S.: Integrated services in the internet architecture: An overview, RFC 1633 (1994).

[6] Guo, Z. and Hao, Q.: Optimization of kvm network based on cpu affinity on multi-cores, *Proc. Information Technology, Computer Engineering and Management Sciences* (*ICM*), *2011 International Conference*, Vol.4, pp.347–351 (2011).

[7] Liu, J.: Evaluating standard-based self-virtualizing devices: A performance study on 10 gbe n ics with sr-iov support, *Proc. Parallel Distributed Processing* (*IPDPS*), *2010 IEEE International Symposium*, pp.1–12 (2010).

[8] Tang, Y. and Li, J.: Von/kvm: A high performance virtual overlay network integrated with kvm, *Proc. Apperceiving Computing and Intelligence Analysis* (*ICACIA*), *2010 International Conference*, pp.129–132 (2010).

[9] McKeown, N., Anderson, T., Balakrishnan, H., et al.: Openflow: Enabling innovation in campus networks, *SIGCOMM Comput. Commun. Rev.*, Vol.38, No.2, pp.69–74 (2008).

[10] Barham, P., Dragovic, B., Fraser, K., et al.: Xen and the art of virtualization, *Proc. 19th ACM Symposium on Operating Systems Principles* (*SOSP '03*), pp.164–177 (2003).

[11] Liu, C.L. and Layland, J.W.: Scheduling algorithms for multiprogramming in a hard-real-time environment, *J. ACM*, Vol.20, No.1, pp.46–61 (1973).

[12] Paul Schnackenburg: Virtualization: What's new with Hyper-V, Microsoft TechNet, available from ⟨http://technet.microsoft.com/en-us/magazine/dn235778.aspx⟩ (accessed 2014-07-28).

[13] VMware: vSphere 5.5 document center, available from ⟨http://pubs.vmware.com/vsphere-55/index.jsp#com.vmware.vsphere.networking.doc/GUID-35B40B0B-0C13-43B2-BC85-18C9C91BE2D4.html⟩ (accessed 2014-07-28).

[14] Citrix: Xen: Virtual network QoS settings, XenServer Administrator's Guide, available from ⟨http://docs.vmd.citrix.com/XenServer/4.0.1/reference/reference.html⟩ (accessed 2014-07-28).

[15] Sharma, S., Katramatos, D., Yu, D. and Shi, L.: Design and Implementation of an Intelligent End-to-End Network QoS System, *Proc. International Conference on High Performance Computing, Networking, Storage and Analysis* (*SC '12*), Article 68 (2012).

推薦文

本論文では，仮想マシン間でのエンドツーエンドのリアルタイム性を保証するために，仮想マシンモニタ層にリアルタイム通信機能を付加し，ネットワーク制御には OpenFlow を利用した，リアルタイム通信基盤が提案されている．仮想化ハイパーバイザと OpenFlow を組み合わせた QoS 保証環境という新規性の高い提案であり，さらに，通信基盤を実装し，設定したデットライン時間内にパケットが到達することも確認されており，有用性の高い研究である．以上より，本研究会からの推薦に値する．

（マルチメディア通信と分散処理研究会主査　勝本道哲）

鈴木 健一 （学生会員）

2013 年東京農工大学工学部情報工学科卒業．現在，同大学大学院修士課程在学中．仮想マシン環境におけるネットワーク通信の QoS 保証に関する研究に従事．

宮田 宏 （学生会員）

1988 年北海道大学理学部卒業．2001 年横河電機（株）入社，IPv6 検査ツールの開発等に従事，2011 年東京農工大学大学院技術経営研究科修了．2012 年より東京農工大学大学院工学府電子情報工学専攻博士後期課程在学中．現在，産業用通信分野の研究に従事．IEEE，電子情報通信学会，計測自動制御学会各会員．

佐藤 未来子 （正会員）

1990 年東京農工大学大学院工学研究科修了．同年（株）日立製作所入社，サーバシステムの設計・性能評価等に従事．2006 年東京農工大学大学院工学教育部電子情報工学専攻博士後期課程修了．工学博士．2010 年東京農工大学工学部・工学府特任助教．並列計算機，省電力，オペレーティングシステムに関する研究に興味を持つ．IEEE，電子情報通信学会各会員．

並木 美太郎 （正会員）

1984 年東京農工大学工学部数理情報工学科卒業．1986 年同大学大学院修士課程修了，同年 4 月（株）日立製作所基礎研究所入社．1988 年東京農工大学工学部数理情報工学科助手，1993 年 11 月電子情報工学科助教授，1998 年 4 月情報コミュニケーション工学科助教授．2007 年 4 月大学院共生科学技術研究院教授．博士（工学）．オペレーティングシステム，言語処理系等のシステムソフトウェア，並列分散処理，組込みシステム，コンピュータネットワーク等の研究・開発・教育に従事．ACM，IEEE，電子情報通信学会，ソフトウェア科学会各会員．