Análise de Desempenho de Mecanismos de Encaminhamento de Pacotes em Redes Virtuais*

Ulisses da Rocha Figueiredo, Antonio Gonzalez Pastana Lobato, Diogo Menezes Ferrazani Mattos, Lyno Henrique Gonçalvez Ferraz e Otto Carlos Muniz Bandeira Duarte

[1]Grupo de Teleinformática e Automação
Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brasil

{ulisses,antonio,menezes,lyno,otto}@gta.ufrj.br

Resumo. *A técnica de virtualização de redes é essencial para criar ambientes de experimentação para a Internet do Futuro, assim como também é uma efetiva proposta pluralista para a Internet, na qual diversas redes diferentes executam em um mesmo substrato físico. Embora a virtualização de servidores seja um sucesso, a virtualização de redes ainda apresenta enormes desafios e, ente eles, o isolamento e o desempenho são os principais. Este artigo analisa as principais arquiteturas de virtualização de redes usadas em computadores pessoais, focando nas plataformas Xen e OpenFlow. São analisados os mecanismos de comunicação ponte e roteador nativos do Xen, o Open vSwitch como comutador e no modo OpenFlow, o OpenFlow em NetFPGA, a comutação com recursos virtualização de dispositivos de entrada e saída com acesso direto ao hardware e a proposta de virtualização híbrida XenFlow, que combina o plano de dados OpenFlow com o plano de controle Xen. Os resultados mostram taxas de encaminhamento similares do OpenFlow implementado pelo Open vSwitch e do Linux nativo e um desempenho superior ao se usar encaminhamento pelo hardware provido por NetFPGA, agindo com um comutador OpenFlow.*

Abstract. *The network virtualization technique is essential to create environments for experimentation of Future Internet proposals, and it is also an effective pluralist proposal for the Internet, in which several different networks run on the same physical substrate. While server virtualization is a success, network virtualization still presents challenges, and, among them, isolation and performance are key challenges. In this paper we analyze the major network-virtualization architectures based on personal computers, focusing on OpenFlow and Xen platforms. We analyzed native Xen mechanisms for networking, bridge and router; Open vSwitch, as an OpenFlow switch, NetFPGA running OpenFlow, input and output device virtualization with direct access to hardware; and we also analyze the the XenFlow hybrid virtualization technique, which combines OpenFlow data plane with Xen control plane. The results show that Open vSwitch, implementing OpenFlow, shows similar transfer rates than native Linux, and NetFPGA hardware switching, running OpenFlow, performances better than native Linux.*

1. Introdução

O projeto original da Internet não atende às exigências das aplicações atuais, como segurança, gerenciamento, mobilidade e monitoramento [Mattos and Duarte 2012,

*Este trabalho foi realizado com recursos da FINEP, FUNTTEL, CNPq, CAPES, FAPERJ e UOL.

Fernandes and Duarte 2011, Moreira et al. 2009]. As novas propostas para a Internet do Futuro, além de serem modeladas, analisadas e simuladas, precisam também ser testadas em ambientes reais com escala similar à atual Internet. No entanto, os provedores de serviço de Internet evitam efetuar experimentos que alterem o núcleo da rede, pois isso pode impactar e causar falhas em serviços já estabelecidos. Portanto, a experimentação de novas aplicações, serviços e protocolos requer um ambiente que se aproxime das condições realistas da rede de produção convencional e, ao mesmo tempo, seja isolado o suficiente para protegê-la de um mau funcionamento dos testes.

A virtualização é a técnica usada para prover um ambiente pluralista de redes virtuais para a experimentação de propostas para a Internet do Futuro [Feamster et al. 2007, Mattos et al. 2012a]. A virtualização separa a função desempenhada por um elemento de rede de sua realização física e, portanto, diversas redes virtuais executam simultaneamente sobre um mesmo hardware [Chowdhury and Boutaba 2010]. Assim, diversos protocolos e serviços inovadores podem ser testados no núcleo da rede, pois cada rede virtual se comporta como uma fatia de rede isolada das demais. Cada rede virtual é dedicada a um experimento e tem sua própria pilha de protocolos e arcabouço de controle. As diversas redes virtuais são isoladas e podem ter diferentes requisitos de qualidade de serviço (QoS) [Keller and Rexford 2010, McKeown et al. 2008].

Xen [Pisa et al. 2011] e OpenFlow [Mattos et al. 2011a, Mattos et al. 2011b] são as técnicas de virtualização fundamentais usadas para a construção do ambiente de experimentação proposto. Xen [Egi et al. 2007] é uma ferramenta de virtualização de computadores, em que as máquinas virtuais se comportam como roteadores. O OpenFlow [McKeown et al. 2008], por outro lado, provê elementos de rede programáveis, no qual o plano de controle da rede é executado em um nó independente e centralizado, interagindo com os elementos de encaminhamento através de uma interface de programação de aplicação (ıApplication Programming Interface - API). Este artigo propõe a análise e a experimentação de ferramentas de virtualização de redes. As ferramentas analisadas são o Xen e o OpenFlow. Para tal , os experimentos propostos consideram as configurações de encaminhamento de pacotes para máquinas virtuais no Xen, nos modos roteador e ponte nativos, além de considerar o encaminhamento com o comutador por *software* Open vSwitch, tanto usando-o como uma ponte Ethernet, quanto como um comutador OpenFlow. Por fim, também é analisado o impacto do uso de técnicas de aceleração de encaminhamento de dados por *hardware*, seja usando placas de redes programáveis NetFPGA [Naous et al. 2008], seja através de placas de rede que fornecem acesso direto ao *hardware* a máquinas virtuais [Division 2011].

A análise de desempenho proposta nesse artigo avalia ainda o sistema híbrido de virtualização de rede XenFlow [Mattos et al. 2011c]. O sistema XenFlow mostrou ter desempenho similar ao encaminhamento obtido pelo Open vSwtich sendo controlado pela aplicação padrão de comutação do controlador POX para OpenFlow. O desempenho obtido pelo sistema XenFlow comprova que o paradigma de separação de planos de controle e dados [Pisa et al. 2010, Fernandes et al. 2010] é capaz de fornecer desempenho à virtualização de redes e permite que o roteamento seja implementado diretamente em um plano de encaminhamento OpenFlow.

O restante do artigo está organizado da seguinte forma. A plataforma de testes usada nos experimentos é discutida na Seção 2. Os modelos de virtualização de redes analizados são apresentados na Seção 3. A Seção 5 discute os resultados dos experimentos. A Seção 6 conclui o artigo.

2. A Plataforma de Experimentação de Redes

A plataforma de testes usada para a experimentação das tecnologias de encaminhamento é o FITS (*Future Internet Testbed with Security*)[1]. O FITS segue uma arquitetura de *software* extensível capaz de implantar um ambiente de experimentação para propostas de Internet do Futuro. O principal objetivo do FITS é proporcionar uma infraestrutura aberta, compartilhada e de propósito geral para testes, permitindo avaliar o desempenho de propostas inovadoras. Outro objetivo da plataforma de testes é oferecer à comunidade de pesquisadores em redes mecanismos para que sejam desenvolvidas soluções para a próxima geração da Internet. Essa plataforma de testes oferece uma abordagem pluralista, na qual as redes virtuais são espalhadas em diversos nós físicos [Mattos et al. 2012b]. O FITS também permite escolher entre o encaminhamento de pacotes através das máquinas virtuais ou adotar uma abordagem de separação de planos, em que o encaminhamento de pacotes é realizado pela máquina física, de acordo com as informações de controle oriundas da máquina virtual. Assim, na abordagem de separação de planos, os pacotes de dados são encaminhados para seu destino final sem passar pela máquina virtual. A abordagem de separação de planos aumenta a capacidade de encaminhamento da rede virtual e também permite à máquina virtual liberar recursos para executar outros processos, ao invés de processar o encaminhamento de pacotes.

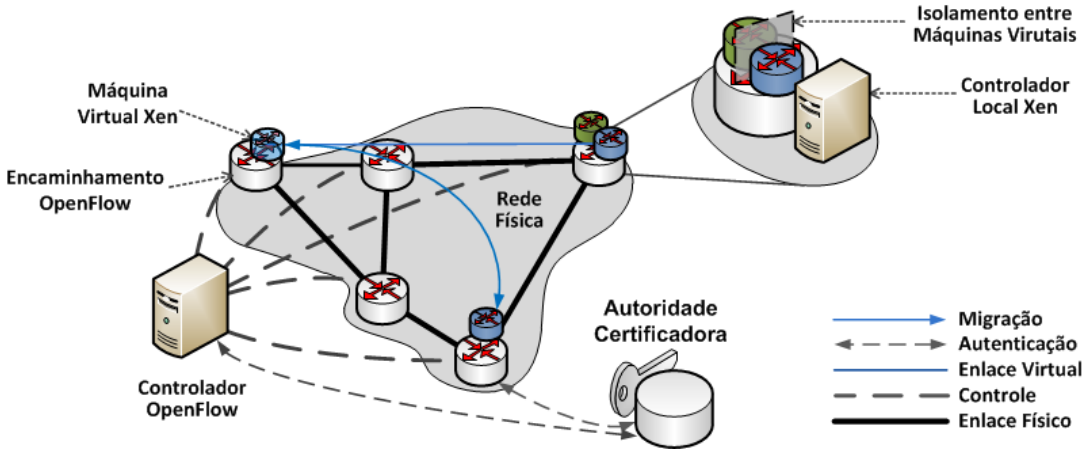


Figura 1. Arquitetura da plataforma de testes FITS. A plataforma de testes implementa o plano de controle em máquinas virtuais Xen, enquanto o encaminhamento de pacotes é realizado pelo OpenFlow.

O encaminhamento padrão da plataforma de testes FITS é realizado pelo comutador por *software* Open vSwitch. O Open vSwitch realiza a comutação dos pacotes entre máquinas virtuais e a rede física. Contudo, o Open vSwitch é um comutador programável que permite também o controle e o gerenciamento dos fluxos em redes virtuais. Uma das possíveis interfaces de interação com o Open vSwitch é através da API OpenFlow. Portanto, a integração de máquinas virtuais Xen e a rede OpenFlow é possível através do uso do Open vSwitch.

Este artigo apresenta uma análise do desempenho de diversas formas de encaminhamento de pacotes em redes virtuais. A ideia básica do artigo é usar um conjunto de nós da plataforma de testes FITS para avaliar as diferentes técnicas de encaminhamento.

[1]http://www.gta.ufrj.br/fits.

3. Os Modelos de Virtualização de Rede

As duas principais tecnologias avaliadas são a virtualização de máquinas provida pelo Xen e a comutação programável provida pelo OpenFlow. A comutação provida pelo OpenFlow é ainda avaliada quando implementada em *hardware*, através de uma placa de rede programável NetFPGA [Naous et al. 2008]. A vantagem da utilização da plataforma de virtualização Xen é a criação de um ambiente de virtualização de redes com controle distribuído e com amplo suporte à inovação, pois as máquinas virtuais (roteadores virtuais) são isoladas umas das outras e o administrador da rede virtual pode desenvolver seus mecanismos de controle sem restrições. A vantagem da utilização do OpenFlow é que, por ser um comutador programável, a migração de fluxos é trivial, consequentemente a migração dos enlaces virtuais torna-se uma atividade trivial, permitindo o remapeamento simples de topologias lógicas sobre a topologia física. Além disso, o encaminhamento com base em recursos OpenFlow garante o uso de mecanismos de encaminhamento especialmente desenvolvidos para essa função, o que contribui para um maior desempenho no encaminhamento de pacotes

3.1. O Hipervisor Xen

O Xen é uma plataforma de virtualização de computadores pessoais bastante empregada na consolidação de servidores[2]. A arquitetura do Xen é baseada em uma camada de virtualização, localizada sobre o *hardware*, denominada Monitor de Máquina Virtual (VMM – *Virtual Machine Monitor*) ou hipervisor, como pode ser visto na Figura 2. Sobre o hipervisor são executados os ambientes virtuais, chamados de máquinas virtuais, ou domínios não privilegiados (Domínio U), que acessam recursos de forma independente, como CPU, memória, acesso a disco e à rede. Cada ambiente virtual está isolado dos demais, isto é, a execução de uma máquina virtual não afeta a de outra máquina virtual as quais, inclusive, podem ter sistemas operacionais distintos. Há ainda um ambiente virtual privilegiado, denominado Domínio 0, que detém a exclusividade do acesso aos dispositivos físicos e, portanto, provê o acesso às operações de Entrada/Saída dos demais domínios, além de executar operações de gerência do hipervisor. Já os demais domínios, referenciados como Domínio U ou domínios desprivilegiados, não possuem acesso direto ao *hardware*. Sendo assim, os domínios desprivilegiados possuem dispositivos (*drivers*) virtuais, que se comunicam com o Domínio 0 para acessarem os dispositivos físicos.

A virtualização da interface de rede no Xen é feita demultiplexando os pacotes que entram pela interface física para os domínios não privilegiados e, de forma similar, multiplexando os pacotes que saem desses domínios para as interfaces físicas de rede. A virtualização das operações de entrada e saída nas interfaces de rede se dá da seguinte forma. Os Domínios Us possuem acesso a dispositivos virtuais de entrada e saída, que são controlados por dispositivos (*drivers*) virtuais que fazem requisições ao Domínio 0 para acessarem os dispositivos físicos. Ao contrário dos Domínios Us, o Domínio 0 tem acesso direto aos dispositivos de entrada e saída, através dos controladores de dispositivos (*drivers*) nativos. Dessa forma, ao receber uma requisição de um Domínio U, o Domínio 0 executa a requisição diretamente sobre o controlador de dispositivo (*driver*) nativo. A comunicação entre os dispositivos virtuais dos domínios desprivilegiados e o Domínio 0 é realizada através de uma dupla de interfaces: interface *back-end* e interface *front-end* [Fernandes et al. 2010]. Cada domínio desprivilegiado tem interfaces virtuais,

[2]A consolidação de servidores consiste em instalar diferentes servidores em máquinas virtuais isoladas hospedadas por uma mesma máquina física.

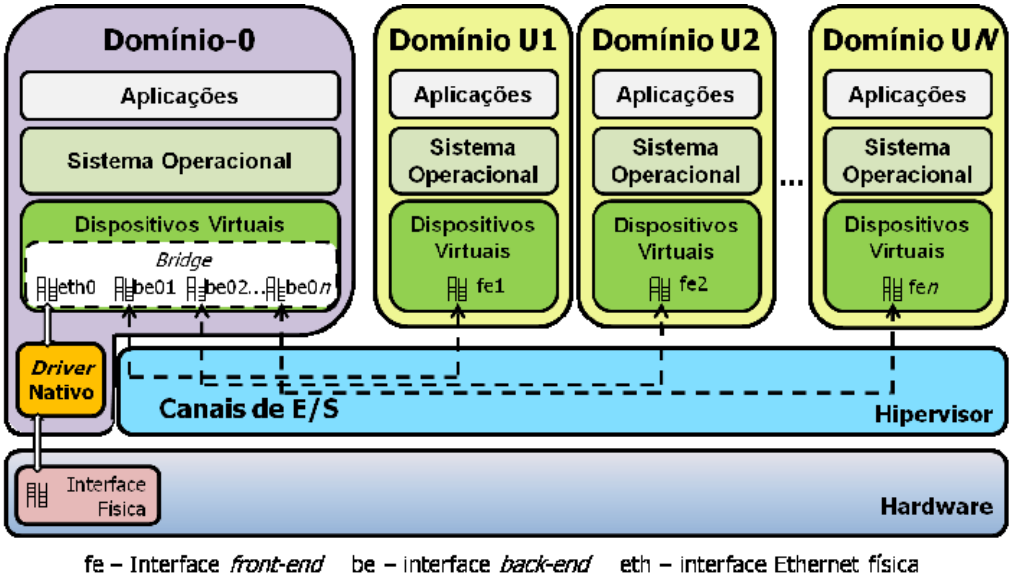
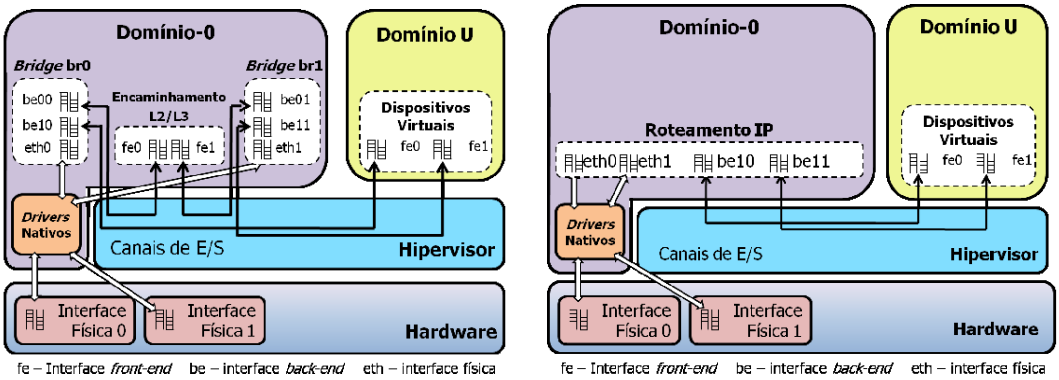


Figura 2. Arquitetura da plataforma de virtualização Xen.

chamadas *front-end*, que são utilizadas para todas as comunicações de rede. Essas interfaces virtuais são tratadas pelos sistemas operacionais dos Domínios Us como se fossem interfaces físicas reais. Para cada interface *front-end* criada nos domínios desprivilegiados, é criada uma interface *back-end* no Domínio 0. As interfaces *back-end* atuam como representantes das interfaces dos domínios desprivilegiados no Domínio 0. As interfaces *back-end* e *front-end* se comunicam através de um canal de E/S (*I/O Channel*). A troca de pacotes entre as interfaces dos domínios Us e o Domínio 0 é realizada de forma eficiente e, portanto, sem cópia de memória. Um mecanismo empregado pelo canal de E/S remapeia a página física que contém o pacote no domínio de destino.

Por padrão, no Xen, a conexão entre as interfaces *back-end* e as interfaces físicas de rede pode ser realizadas de duas maneiras. A primeira, também presente no padrão do Xen, é através do modo comutado (*Bridge*), mostrado na Figura 3(a). Nesse modo, são instanciadas pontes (*bridges*) no Domínio 0 e as interfaces *back-end* e as interfaces reais são associadas a elas. Uma ponte (*bridge*) é um comutador por *software*. Assim, o encaminhamento do pacote para interface *back-end* correta é realizado através do encaminhamento do pacote pela interface que responde ao endereço MAC de destino do pacote. Vale ressaltar que são necessárias tantas pontes (*bridges*) quanto o número de interfaces físicas . O segundo modo de interconexão das interfaces reais e as *back-end* é o modo roteado (*Router*), mostrado na Figura 3(b). No modo roteado, o Domínio 0 passa a se comportar como um roteador. Dessa forma, para cada pacote que chega, o Domínio 0 verifica o endereço IP de destino e encaminha o pacote de acordo com as rotas definidas em suas tabelas de roteamento. Assim, o encaminhamento do pacote para a interface *back-end*, ou física, correta, depende somente da definição correta da rota no Domínio 0 [Fernandes et al. 2010].

A virtualização da rede é alcançada no Xen através da instanciação de diversas máquinas virtuais, que correspondem a elementos virtuais de rede sobre um mesmo *hardware* físico, pois o Xen permite a execução de múltiplas máquinas virtuais simultaneamente sobre a mesma máquina física. Um exemplo de virtualização de redes usando Xen é o caso em que os elementos de rede virtuais instanciados são roteadores virtuais.



(a) Modo comutado (*Bridge*). (b) Modo roteado (*Router*).

Figura 3. Virtualização do recurso de rede no Xen.

Nesse caso, como a camada de virtualização do Xen está abaixo dos sistemas operacionais, cada roteador virtual pode ter o seu próprio sistema operacional e cada um detém os seus próprios planos de dados e controle isolados dos demais roteadores. Nessa arquitetura de rede virtual, um roteador virtual pode ser instanciado, configurado, monitorado e desativado sob demanda. O roteador virtual pode ser migrado, ainda em funcionamento, usando o mecanismo de migração ao vivo do Xen [Clark et al. 2005].

3.2. A Comutação OpenFlow

A ideia básica do OpenFlow é dividir a rede em dois planos, o de controle, responsável pela execução dos algoritmos de controle da rede, e o plano de dados, responsável pelo encaminhamento e pela aplicação de políticas a cada pacote. Para tanto, o OpenFlow explora o fato de que a maioria dos fabricantes de comutadores *Ethernet* e de roteadores implementa uma tabela de fluxos que permite a execução de aplicações de *firewall*, *Network Translation Table* (NAT), qualidade de serviço (QoS – *Quality of Service*), e também a coleta de estatísticas diretamente nos equipamentos [McKeown et al. 2008]. No entanto, cada fabricante tem uma tabela de fluxos diferente. Assim, a proposta do OpenFlow é definir um conjunto de funcionalidade comuns que deve executar em comutadores e roteadores, definindo uma interface padrão de controle do plano de dados. Dessa forma, os comutadores e roteadores da rede ficam responsáveis somente pelo plano de dados. Já o plano de controle é executado em outro nó, logicamente centralizado, que detém uma visão global da rede. O plano de controle centralizado executa os algoritmos de controle e decide as ações a serem tomadas pelo plano de dados. As informações são trocadas entre o plano de dados e o plano de controle segundo o protocolo OpenFlow.

O OpenFlow implementa a virtualização do plano de dados. A arquitetura do OpenFlow é baseada na separação física das funções de encaminhamento e de controle da rede. O OpenFlow associa o encaminhamento eficiente, já que a função de encaminhamento é mantida em *hardware* especializado, com uma interface de controle simples e que permite o desenvolvimento de novas aplicações. Essa arquitetura possibilita que as funções da rede sejam definidas por aplicações expressas em *software*, enquanto o encaminhamento é realizado por *hardware* especializado. A função de encaminhamento, desempenhada pelo plano de dados, é executada por elementos especializados da rede que apresentam uma tabela de fluxos compartilhada. É através dessa tabela de fluxos

compartilhada que o plano de dados é virtualizado. Já a função de controle, exercida pelo plano de controle, é centralizada em outro elemento da rede, o chamado controlador. O controlador executa funções de controle para a rede virtual.

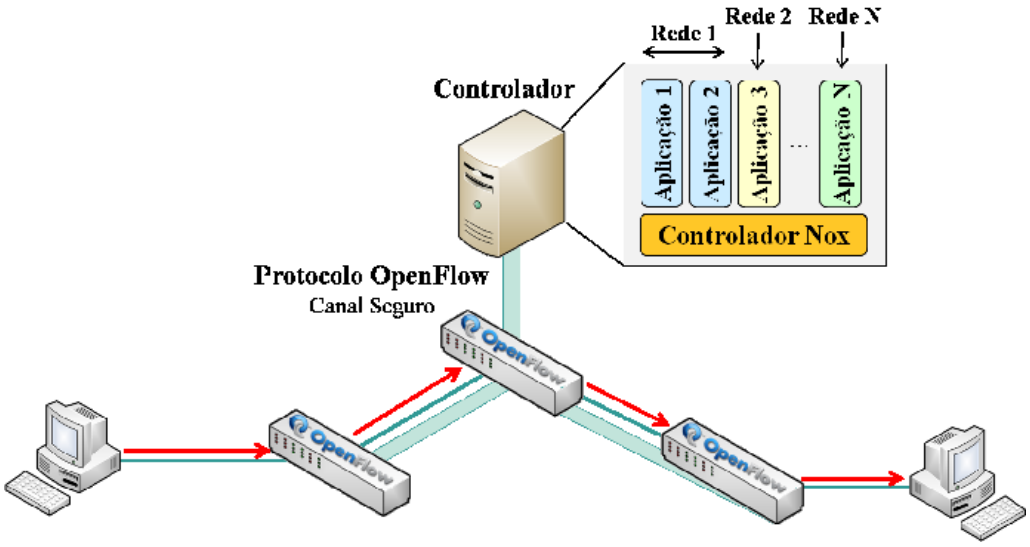


Figura 4. Arquitetura de uma rede OpenFlow. Os comutadores OpenFlow comunicam-se com o controlador através do protocolo OpenFlow em um canal seguro. O controlador executa as aplicações de controle de cada rede virtual.

A Figura 4 mostra a organização de uma rede OpenFlow e a comunicação dos comutadores com o controlador. A comunicação entre os elementos de rede e o controlador é definida pelo protocolo OpenFlow. A comunicação é estabelecida através de um canal seguro definido entre o controlador e cada elemento OpenFlow. O protocolo OpenFlow define funções para configurar e monitorar os elementos. O encaminhamento é definido com base em fluxos. Um fluxo é uma sequência de pacotes com um conjunto de características comuns. Quando um pacote chega ao elemento encaminhador, o elemento verifica se o pacote se adéqua a algum fluxo já definido. Em caso positivo, as ações definidas para aquele fluxo são aplicadas ao pacote. Em caso negativo, o pacote é encaminhado para o controlador, que extrai as características do fluxo, a partir do pacote, e cria um novo fluxo, introduzindo-o na tabela de fluxos do elemento OpenFlow. Uma das possíveis ações que o controlador pode definir para um fluxo é que ele siga o processamento normal, seja de comutação na Camada 2, seja roteamento na Camada 3, como se não existisse o protocolo OpenFlow.

A implementação do OpenFlow pode ser feita em diversos equipamentos da rede. A sua especificação concilia recursos de hardware disponíveis em equipamentos de rede comuns, como tabela de fluxos, estruturas de filas e memória endereçada por conteúdo, com os requisitos de redes programáveis, em especial a separação do plano de controle do plano de dados. Entre as opções de implantação do OpenFlow, há a proposta do Open vSwitch, que concilia as vantagens de redes programáveis com a virtualização de sistemas. O Open vSwitch [Pfaff et al. 2009] é um comutador de software projetado tanto para ser usado em ambientes virtualizados, para transformar um computador comum em um comutador programável. O Open vSwitch é um comutador por software que fica no hipervisor, ou no domínio de gerenciamento do sistema virtualizado, como o domínio 0 do Xen [Egi et al. 2008], e provê a conectividade entre máquinas virtuais e as interfaces

físicas. No caso de um sistema sem virtualização, o Open vSwitch age de maneira similar a *bridge* nativa do Linux, encaminhando os pacotes entre as interfaces de rede do sistema. O seu modelo de operação é muito semelhante ao de um comutador Ethernet padrão, no entanto o novo modelo de comutador exporta uma interface para a manipulação do encaminhamento e o gerenciamento do estado de configuração, em tempo de execução. Essas características são desejadas para poder fazer a distribuição lógica de comutadores e para suportar a integração com os ambientes virtuais. Pela interface remota, um processo pode ler e escrever variáveis de configuração do comutador, assim como pode manipular o caminho definido para o encaminhamento de um fluxo. Nesse sentido, a interface para a manipulação de fluxos implementa um super-conjunto do protocolo OpenFlow, já que agrega ainda algumas funções específicas para a virtualização.

4. O Sistema Híbrido XenFlow

XenFlow [Mattos et al. 2011c] é um sistema de virtualização híbrido baseado nas plataformas Xen e OpenFlow. O XenFlow oferece tanto a opção de controle distribuído ou de controle centralizado da rede. No XenFlow, as funções do roteador virtual são divididas em dois planos, o plano de controle e o plano de dados. O plano de controle, que é executado dentro da máquina virtual criada com a plataforma Xen, é responsável por todas as tarefas de controle, como a atualização da tabela de roteamento. Por outro lado, o plano de dados é criado utilizando-se o OpenFlow que é responsável pelo encaminhamento dos pacotes. A principal diferença do encaminhamento XenFlow para o encaminhamento de pacotes no Xen é que, ao contrário do que acontece no Xen, o roteamento de pacotes no XenFlow ocorre no plano de dados, ou seja , após o fluxo de encaminhamento de pacotes, ao invés de encaminhar cada pacote para a máquina virtual, ele executa as funções relativas ao roteamento e encaminha os pacotes na interface de saída [Mattos et al. 2011c, Mattos and Duarte 2012]. No XenFlow, cada máquina física está associada a um comutador OpenFlow que liga as máquinas virtuais Xen ao resto da rede e cada máquina virtual Xen funciona como um controlador do comutador.

As duas principais vantagens do XenFlow são: a realização da migração de topologias virtuais sem perdas de pacote, herdada da característica de separação de planos do OpenFlow, podendo mapear um enlace lógico sobre um ou mais enlaces físicos, e a possibilidade de fazer o controle distribuído da rede, tal qual nas redes convencionais, herdada do Xen. O XenFlow é inovador em relação ao alcance da migração de um roteador virtual, sendo ele aumentado em relação a outras propostas na literatura [Wang et al. 2008, Pisa et al. 2010, Clark et al. 2005], que se restringem ao nó físico de destino ter os mesmos vizinhos que o nó físico de origem. A facilidade provida pelo XenFlow de mapear um enlace lógico sobre um, ou mais enlaces físicos, elimina essa restrição. Outra vantagem do sistema proposto é o controle distribuído da rede, que é realizado pelos protocolos de roteamento executando em máquinas virtuais.

4.1. O Encaminhamento Assistido por *Hardware*

Outra possibilidade para o encaminhamento de pacotes em redes virtuais é através das modernas técnicas de acesso direto ao *hardware* de rede pelas máquinas virtuais, como o PCI-SIG *Single Root I/O Virtualization* (SR-IOV) [Division 2011], e através do uso de placas de rede programáveis, como a NetFPGA (*Network Field Programmable Array*). Com a tecnologia SR-IOV, um único dispositivo PCI *Express* (PCIe) apresenta-se como múltiplos dispositivos virtuais. Cada dispositivo virtual é dedicado a uma máquina virtual que terá acesso direto à instância do dispositivo virtual no *hardware*. Logo, o

sistema evita mudanças de contexto, controle e classificação de tráfego. Além disso, evitam-se cópias extras de memória em relação ao uso de comutadores por *software* para a multiplexação de acesso aos recursos de E/S.

Outra possível implementação do plano de encaminhamento de pacotes em redes virtuais é a placa NetFPGA programada com o OpenFlow para a comutação de pacotes por *hardware*, atingindo taxas de até 10 Gb/s [Naous et al. 2008]. A implantação do protocolo OpenFlow em placas NetFPGA permite que essas placas implementem a API OpenFlow e, então, possam ser controladas por um controlador OpenFlow. Dessa forma, uma aplicação OpenFlow pode controlar uma placa NetFPGA sem que haja a necessidade de implementar um novo programa específico para a placa NetFPGA.

5. Experimentos e Resultados

Dois ambientes de testes distintos foram implantadas para testar o desempenho dos mecanismos de encaminhamento. Ambos foram implementados em linguagem Python, sendo que um utiliza as funções do módulo do *kernel* do Linux para geração de pacotes, `pktgen`[3], e o outro utiliza-se do aplicativo de testes para servidores *Web* `HTTPerf`[4], responsável por gerar demandas de conexão a um servidor HTTP. Foram realizados três experimentos distintos para cada mecanismo de encaminhamento. O primeiro experimento utilizou o `pktgen` para medir a relação entre a taxa de pacotes enviada e a recebida. O segundo usa o `pktgen` em conjunto com o `ping` para medir o atraso com um fluxo sujeito a um tráfego de fundo variável. Já o terceiro experimento usou a ferramenta `HTTPerf` para avaliar a taxa de conexões a servidor web, permitindo assim, avaliar a taxa de fluxos por segundo cada mecanismo de encaminhamento de pacotes é capaz de atender.

Para todos os experimentos, foi implementada uma topologia com três máquinas físicas. Uma geradora de pacotes, uma agindo como encaminhadora de pacotes e a terceira agindo como receptora de pacotes. As máquinas geradora e receptora de pacotes são servidores, cada um com dois processadores Xeon X5570, 2.93 GHz, cada processador com 8 núcleos, logo, ao todo cada servidor conta com 16 núcleos, 48 Gb de memória RAM e interface de rede Gigabit Intel. A máquina encaminhadora conta com um processador Intel Core i7 950, com 8 núcleos, 3.07 GHz, 4 Gb de memória RAM e duas interfaces de rede Intel Gigabit e uma interface de rede com quatro portas Ethernet NetFPGA de 1 Gb/s. A razão pela qual essa topologia foi escolhida é que, desta forma, é possível testar diversas formas de encaminhamento e roteamento de pacotes. As máquinas foram interligadas através de cabos Ethernet, com saturação dos enlaces a 1 Gb/s. Os resultados apresentados nessa seção são médias de 10 rodadas de cada experimento, com intervalo de confiança de 95%.

O objetivo do primeiro experimento é mensurar a taxa máxima de encaminhamento de pacotes para cada modelo de encaminhamento adotado. Para tanto, foi usado o `pktegen` que é uma ferramenta de testes que permite a geração de um fluxo de pacotes na rede a altas taxas, pois trata-se de um módulo do *kernel* do Linux. Ao se verificar a taxa de pacotes recebida, é possível medir a eficiência do mecanismo de encaminhamento testado. Um contador é iniciado tanto na máquina geradora quanto na receptora, sendo então calculadas as taxas de pacotes gerados e recebidos por segundo. Com esse tipo de experimento, é possível avaliar a taxa de pacotes que chega com sucesso ao ou-

[3]http://www.linuxfoundation.org/collaborate/workgroups/networking/pktgen.

[4]http://www.hpl.hp.com/research/linux/httperf/.

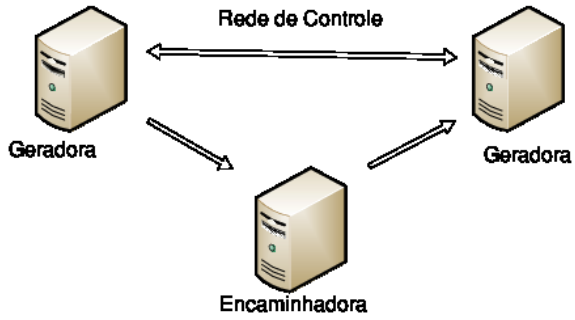


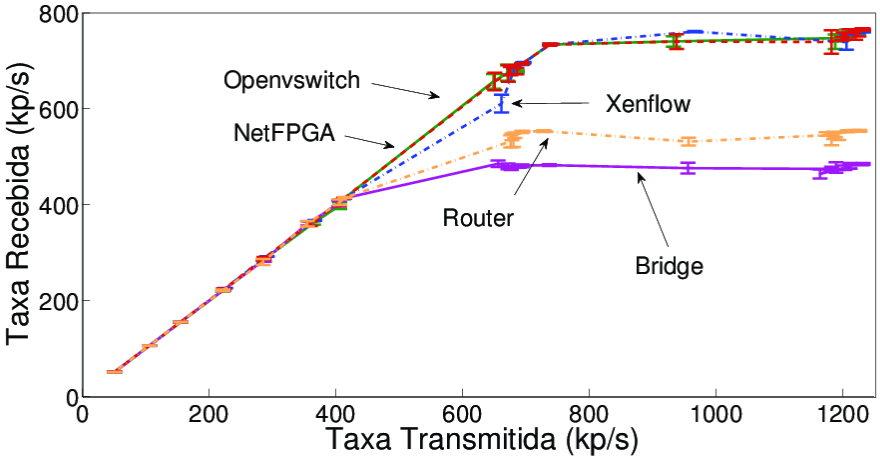
Figura 5. A topologia com uma máquina geradora, uma encaminhadora e a receptora.

tro nó da rede e também verificar a quantidade pacotes que o gerador é capaz de gerar. Desta forma, a análise fica mais precisa, visto que o número de pacotes gerados oscila em função das limitações `pktgen`, como o escalonamento do processo de geração de pacotes e limitações de desempenho que impeçam a geração de uma maior taxa de pacotes.

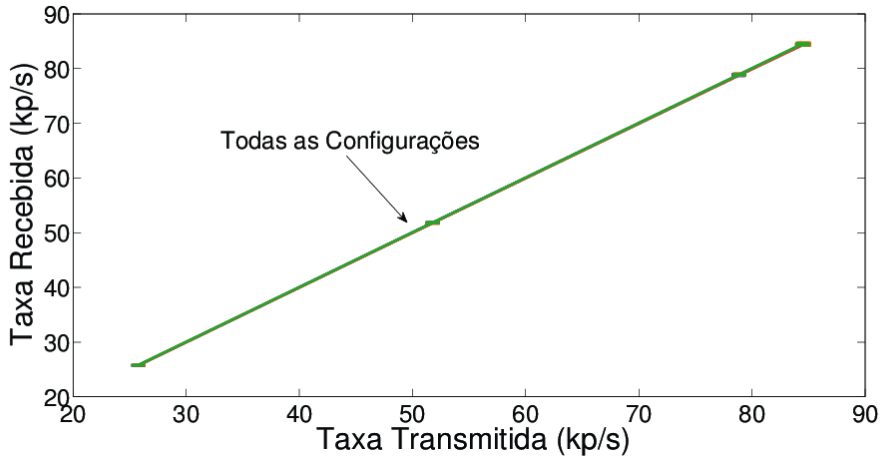
No primeiro teste, foram utilizados pacotes UDP de 64 bytes e 1500 bytes[5], pois desta forma é possível testar o desempenho do mecanismo de encaminhamento tanto para um fluxo de numerosos pacotes pequenos, quanto para um fluxo menos intenso quanto a taxa de pacotes, porém com maior volume de dados, para pacotes grandes. A variação da taxa de pacotes enviada foi até o máximo que foi possível ser gerado. Pelos resultados apresentados na Figura 6, é possível perceber que todos os mecanismos conseguiram atingir valores muito próximos da taxa de saturação de enlace, exceto a *bridge* nativa do Linux, agindo no modo *bridge* do Xen, e o modo roteador do Xen. Tanto no modo roteador, quanto no modo *bridge*, a taxa recebida foi limitada, já que o roteador físico encaminha todos os pacotes para o roteador virtual, máquina virtual do Xen, onde os pacotes são processados para calcular o próximo salto de sua rota, causando perda de desempenho. Contudo, o modo roteador apresenta desempenho melhor, pois o encaminhamento do modo roteador baseia-se nos mecanismos nativo do Linux para encaminhamento de pacotes IP implementados diretamente no *kernel* do sistema operacional, enquanto o modo *bridge* é implementado como um módulo do *kernel* e realiza o encaminhamento de pacotes entre o roteador físico e o virtual com base no encaminhamento da Camda 2.

Para medir a variação do atraso em função de um fluxo de fundo, mais uma vez foi usado o `pktgen`. O `pktgen` permite gerar um tráfego de pacotes de 64 bytes, variando durante o teste, até a capacidade máxima de geração de 1.2 M pacotes/s/footnoteO limite de 1.2M pacotes/s é a limitação da configuração de *hardware* e *software* utilizados nos experimentos.. Em paralelo com esse tráfego, foi medido o tempo médio que pacotes *ICMP Echo Request* e *Echo Reply*, da aplicação `ping`, levaram para serem executados para cada taxa de fluxo de fundo. Os resultados mostrados na Figura 7(a) revelam que a variação do tráfego de fundo causa uma variação do tempo de atraso, entretanto, essas oscilações não seguem um padrão específico. É possível perceber que o mecanismo de encaminhamento assistido por *hardware*, NetFPGA, apresentou um tempo reduzido de atraso. Isso ocorre, pois como utiliza *hardware* dedicado a funções de rede, a NetFPGA evita o atraso do envio de pacotes ao sistema operacional para decisão de encaminhamento de pacotes. Os pacotes são encaminhados diretamente pelo *hardware* de rede. O modo

[5] 64 bytes é o tamanho mínimo do conteúdo de um quadro Ethernet, enquanto 1500 bytes é o tamanho mais comum de MTU (*Maximum Transmission Unit*).



(a) Taxa transmitida e recebida para pacotes de 64 bytes.



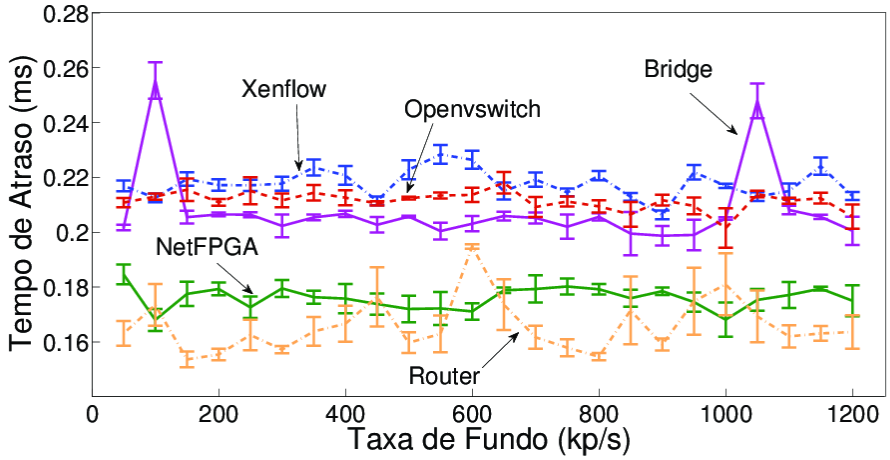
(b) Taxa transmitida e recebida para pacotes de 1500 bytes.

Figura 6. Relação entre taxa de pacotes UDP enviados e recebidos.

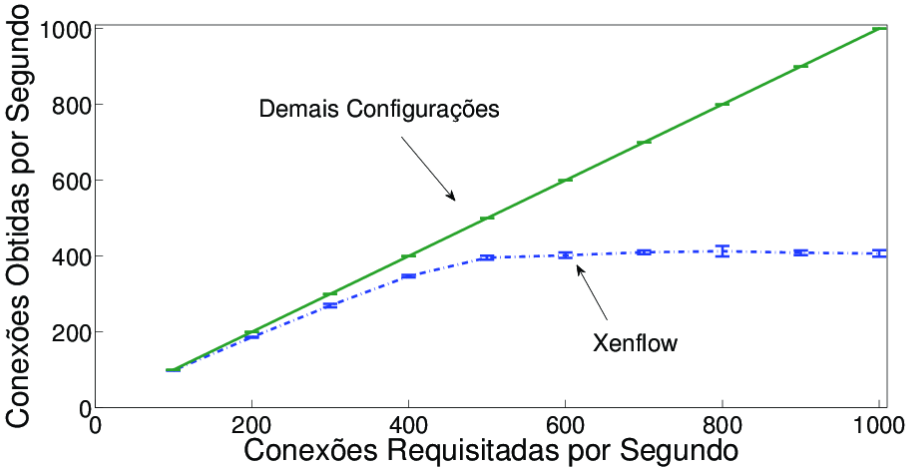
roteador também apresentou um atraso reduzido em relação aos outros mecanismos.

O terceiro experimento realizado verifica a limitação de cada mecanismo de encaminhamento para a definição de fluxos por segundo. O experimento é realizado medindo quantas conexões HTTP são estabelecidas em função da quantidade de conexões requisitadas, cada conexão HTTP estabelece dois fluxos no mecanismo de encaminhamento, um entre o cliente e o servidor e o outro no sentido inverso. Esse teste visa medir a limitação de cada mecanismo de encaminhamento para a resposta à alta carga de requisição de novos fluxos por segundo. Na topologia do teste, um dos computadores faz o papel de cliente e estabelece conexões com o servidor HTTP em outro computador, a conexão é realizada através da máquina encaminhadora. Para realizar as conexões HTTP, foi utilizada a ferramenta `HTTPerf` que é usada para medir a performance de servidores *web*. A maioria dos mecanismos de encaminhamento estabeleceu o número de conexões requisitadas, com exceção do XenFlow. A limitação do XenFlow ocorre, pois o XenFlow introduz dois atrasos no mecanismo de encaminhamento. O primeiro atraso deve-se ao fato de o primeiro pacote de cada fluxo ser enviado ao controlador e, portanto, gera um atraso quando comparado aos modos *bridge* e roteador do Xen. O segundo atraso in-

troduzido pelo XenFlow é o atraso relativo ao cálculo do próximo destino do pacote na rede, pois no XenFlow, ao invés de serem definidos dois fluxos para o pacote ir à máquina virtual e sair da máquina virtual para a interface de saída, só um único fluxo é definido encaminhando o pacote diretamente no plano de dados. A separação de planos apresenta a desvantagem da redução taxa de fluxos por segundo suportada pelo XenFlow, mas, como ressaltado pela Figura 6(a), também apresenta a vantagem de a taxa de encaminhamento de pacotes com o XenFlow ser superior aos modos *bridge* e roteador do Xen.



(a) Atraso de ida e volta em pacotes ICMP.



(b) Taxa de fluxos atendidos.

Figura 7. Atraso do ping com tráfego de fundo e limitação da taxa de definição de fluxos por segundo no mecanismo de encaminhamento.

6. Conclusão

Este artigo analisou o desempenho de diversas técnicas de virtualização de redes. A ferramenta de virtualização Xen implementa dois modos nativos para virtualização de redes, o modo roteador e o modo ponte. O modo roteador realiza o encaminhamento IP entre máquinas virtuais, enquanto o modo ponte realiza o encaminhamento na camada de enlace, agindo assim como uma ponte Ethernet. A técnica de encaminhamento de pacotes OpenFlow também é analisada. Os resultados mostram taxas de encaminhamento

similares do OpenFlow, implementado pelo Open vSwitch, e do XenFlow. Os resultados também revelam que o desempenho de encaminhamento é superior ao se usar encaminhamento pelo *hardware* provido por NetFPGA, agindo com um comutador OpenFlow. Por outro lado, observa-se que a maior complexidade envolvida no encaminhamento através de máquinas virtuais Xen limita a sua taxa de encaminhamento de pacotes. Contudo, ao usar a comutação híbrida XenFlow, que combina a flexibilidade do Xen e o desempenho do OpenFlow, a taxa de encaminhamento alcançada é superior à obtida pelos mecanismos nativos do Xen.

7. Referências

[Chowdhury and Boutaba 2010] Chowdhury, N. M. K. and Boutaba, R. (2010). A survey of network virtualization. *Comput. Netw.*, 54(5):862–876.

[Clark et al. 2005] Clark, C., Fraser, K., Hand, S., Hansen, J., Jul, E., Limpach, C., Pratt, I., and Warfield, A. (2005). Live migration of virtual machines. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pages 273–286. USENIX Association.

[Division 2011] Division, I. L. A. (2011). PCI-SIG SR-IOV primer, an introduction to SR-IOV technology. Technical report, Intel.

[Egi et al. 2008] Egi, N., Greenhalgh, A., Handley, M., Hoerdt, M., Huici, F., and Mathy, L. (2008). Towards high performance virtual routers on commodity hardware. In *Proceedings of the 2008 ACM CoNEXT Conference*, pages 1–12. ACM.

[Egi et al. 2007] Egi, N., Greenhalgh, A., Handley, M., Hoerdt, M., Mathy, L., and Schooley, T. (2007). Evaluating Xen for router virtualization. In *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*, pages 1256–1261. IEEE.

[Feamster et al. 2007] Feamster, N., Gao, L., and Rexford, J. (2007). How to lease the Internet in your spare time. *ACM SIGCOMM Computer Communication Review*, 37(1):61–64.

[Fernandes et al. 2010] Fernandes, N., Moreira, M., Moraes, I., Ferraz, L., Couto, R., Carvalho, H., Campista, M., Costa, L., and Duarte, O. (2010). Virtual networks: Isolation, performance, and trends. *Annals of Telecommunications*, pages 1–17.

[Fernandes and Duarte 2011] Fernandes, N. C. and Duarte, O. C. M. B. (2011). Provendo isolamento e qualidade de serviço em redes virtuais. In *XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC'2011*.

[Keller and Rexford 2010] Keller, E. and Rexford, J. (2010). The platform as a service model for networking. In *Proceedings of the 2010 internet network management conference on Research on enterprise networking*, pages 4–9. USENIX Association.

[Mattos et al. 2011a] Mattos, D., Fernandes, N. C., Costa, V., Cardoso, L., Campista, M. E. M., Costa, L. H. M. K., and Duarte, O. C. M. B. (2011a). OMNI: OpenFlow MaNagement infrastructure. In *2011 International Conference on the Network of the Future (NoF'11)*, pages 52–56, Paris, France.

[Mattos and Duarte 2012] Mattos, D. M. F. and Duarte, O. C. M. B. (2012). QFlow: Um sistema com garantia de isolamento e oferta de qualidade de serviço para redes virtualizadas. In *XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC'2012*.

[Mattos et al. 2011b] Mattos, D. M. F., Fernandes, N. C., Cardoso, L. P., da Costa, V. T., Mauricio, L. H., Barretto, F. P. B. M., Portella, A. Y., Moraes, I. M., Campista, M. E. M., Costa, L. H. M. K., and Duarte, O. C. M. B. (2011b). OMNI: Uma ferramenta para gerenciamento autônomo de redes openflow. In *Salão de Ferramentas do XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos- SBRC'12*.

[Mattos et al. 2011c] Mattos, D. M. F., Fernandes, N. C., and Duarte, O. C. M. B. (2011c). XenFlow: Um sistema de processamento de fluxos robusto e eficiente para migração em redes virtuais. In *XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC'2011*.

[Mattos et al. 2012a] Mattos, D. M. F., Ferraz, L. H. G., Costa, L. H. M. K., and Duarte, O. C. M. B. (2012a). Evaluating virtual router performance for a pluralist future internet. In *Proceedings of the 3rd International Conference on Information and Communication Systems*, ICICS '12, pages 4:1–4:7, Irbid, Jordan. ACM.

[Mattos et al. 2012b] Mattos, D. M. F., Mauricio, L. H., Cardoso, L. P., Alvarenga, I. D., Ferraz, L. H. G., and Duarte, O. C. M. B. (2012b). Uma rede de testes interuniversitária a com técnicas de virtualizacao híbridas. In *Salão de Ferramentas do XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC'12*.

[McKeown et al. 2008] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74.

[Moreira et al. 2009] Moreira, M., Fernandes, N., Costa, L., and Duarte, O. (2009). Internet do futuro: Um novo horizonte. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC 2009*, pages 1–59.

[Naous et al. 2008] Naous, J., Gibb, G., Bolouki, S., and McKeown, N. (2008). NetFPGA: reusable router architecture for experimental research. In *Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow*, pages 1–7. ACM.

[Pfaff et al. 2009] Pfaff, B., Pettit, J., Koponen, T., Amidon, K., Casado, M., and Shenker, S. (2009). Extending networking into the virtualization layer. *Proc. HotNets*.

[Pisa et al. 2010] Pisa, P., Fernandes, N., Carvalho, H., Moreira, M., Campista, M., Costa, L., and Duarte, O. (2010). Openflow and xen-based virtual network migration. In Pont, A., Pujolle, G., and Raghavan, S., editors, *Communications: Wireless in Developing Countries and Networks of the Future*, volume 327 of *IFIP Advances in Information and Communication Technology*, pages 170–181. Springer Boston.

[Pisa et al. 2011] Pisa, P. S., Couto, R. S., Carvalho, H. E. T., Neto, D. J. S., Fernandes, N. C., Campista, M. E. M., Costa, L. H. M. K., Duarte, O. C. M. B., and Pujolle, G. (2011). VNEXT: Virtual NEtwork management for Xen-based Testbeds. In *2011 International Conference on the Network of the Future (NoF'11)*, pages 41–45, Paris, France.

[Wang et al. 2008] Wang, Y., Keller, E., Biskeborn, B., van der Merwe, J., and Rexford, J. (2008). Virtual routers on the move: live router migration as a network-management primitive. *ACM SIGCOMM Computer Communication Review*, 38(4):231–242.