OpenFlow 1.3 Software Switch

Eder Leão Fernandes[1], **Christian Esteve Rothenberg**[2]

[1]Diretoria de Redes Convergentes
Fundação CPqD – Campinas – SP – Brasil

[2]Departamento de Engenharia de Computação e Automação Industrial
Universidade de Campinas (UNICAMP) – Campinas – SP – Brasil

ederlf@cpqd.com.br, chesteve@dca.fee.unicamp.br

Abstract. *OpenFlow is a new and fast evolving technology that enables the implementation of Software Defined Networks. This papers describes a tool designed for rapid and inexpensive experimentation of OpenFlow networks. The software switch is intended to be easy to prototype new features due to the implementation in a traditional language and for his simple code. These characteristics contributed to the adoption of the tool by an active community and has become a good choice for both OpenFlow newcomers as well as more advanced experimenters.*

Resumo. *OpenFlow é uma tecnologia nova e de rápida evolução que habilita a implementação de Redes Definidas por Software. Este artigo apresenta uma ferramenta desenvolvida para experimentação rápida e de baixo custo. O software switch é voltado para a prototipagem de novas funcionalidades e novas versões do protocolo pelo seu código simples implementado em uma linguagem tradicional. Essas características ajudaram a criar uma comunidade ativa e fizeram da ferramenta uma boa escolha tanto para o primeiro contato com OpenFlow como para experimentações mais avançadas.*

1. Introdução

O modelo atual das redes de computadores, embora um sucesso, ao se pensar na Internet e nos serviços disponíveis hoje, sofre com vários problemas para se adaptar às novas demandas e cenários da Internet do Futuro [Shenker 1995] [Feldmann 2007]. Por isso, nos últimos anos, a área de redes de computadores tem visto a ascensão do modelo de Redes Definidas por Software. A separação do plano de controle do plano de dados da rede, apesar de não ser uma proposta realmente nova, ganhou força após a criação do protocolo *OpenFlow*, com o qual é possível programar o encaminhamento dos pacotes através de um agente externo, além de poder controlar todos os nós da rede a partir de um único ponto [McKeown et al. 2008].

Apesar de existirem outras tecnologias habilitadoras das Redes Definidas por Software, o protocolo *OpenFlow* segue como uma das principais apostas, principalmente pela indústria, visto a criação de uma organização para desenvolver o padrão [ONF 2013]. Na academia, apesar de mais tímidos, vários trabalhos começam a utilizar o *OpenFlow* para o avanço da pesquisa em redes.

Para rápida experimentação e desenvolvimento de aplicações *OpenFlow* foram desenvolvidas várias ferramentas, como *software switch*, controlador [Gude et al. 2008]

e plugin do *Wireshark* para capturar as mensagens trocadas no canal de controle. Porém todas essas ferramentas estavam disponíveis apenas para OpenFlow 1.0. Com o cenário em mente e a rápida evolução das versões do protocolo pela *Open Networking Foundation*, foi identificada a necessidade de evolução das ferramentas para a versão mais atual, pois as novas funcionalidades habilitam um número ainda maior de experimentos, permitindo a antecipação de soluções para equipamentos não existentes à epoca.

O resto do artigo está organizado da seguinte forma: A Seção 2 fala sobre o breve histórico do trabalho e a motivação para o desenvolvimento da ferramenta. A Seção 3 descreve a arquitetura em alto nível. Na Seção 4 são apresentadas as funcionalidades do *OpenFlow* 1.3 implementadas no *software switch*. A Seção 5 traz os resultados obtidos no projeto. A Seção 6 apresenta a documentação e a proposta de demonstração no Salão de Ferramentas do SBRC 2014. A Seção 7 discute trabalhos futuros e a Seção 8 é a conclusão do artigo. A última Seção traz os e agradecimentos aos colaboradores desse trabalho.

2. Histórico, Objetivos e Motivação

Na época em que o trabalho de implementação do *software switch* foi iniciado, não havia outras opções disponíveis. As ferramentas existentes suportavam até a versão 1.1 do *OpenFlow* [Kis 2011], num momento em que haviam vários esforços para adição de novas funcionalidades. Um exemplo é o protocolo *IPv6*, de grande interesse aos primeiros usuários, devido à situação de esgotamento do *IPv4*. O primeiro trabalho, então, foi adicionar ao *switch* de referência da versão 1.1 suporte à *IPv6* e aos seus cabeçalhos de extensão [Denicol et al. 2011]. O resultado desse primeiro trabalho foi muito importante para a sequência do *software switch*, pois a implementação de umas das principais mudanças da versão 1.2, o *OpenFlow Extended Match* (OXM), foi facilitada devido à implementação anterior utilizar o modelo do *Nicira Extended Match* do *Open vSwitch* (OVS) [Pfaff et al.], o qual foi a base do OXM. Logo após a divulgação da especificação da versão 1.3, o trabalho continou até culminar na implementação descrita nesse artigo.

Para desenvolver a ferramenta foram consideradas duas características essenciais: (i) usabilidade simples, com integração ao Mininet [Lantz et al. 2010] e (ii) fácil de estender (linguagem e código não devem ser de difícil modificação). Esses requisitos foram identificados no *software switch* de referência da versão 1.1, o qual possui um código simples e é uma das opções de simulação no Mininet.

2.1. Análise das ferramentas existentes

Além desse trabalho também há outros *software switches OpenFlow* em constante desenvolvimento. Do ponto de vista da implementação e experimentação todos possuem vantagens e desvantagens.

Open vSwitch. É o *software switch* mais utilizado pela comunidade. Possui grande desempenho, pois disponibiliza, além do modo usuário, um modo *kernel*, sendo propício até para ambientes de produção. O suporte à *OpenFlow* está sendo feito de modo que um único programa suporte todas as versões, por isso ainda tem muitas funcionalidades parciais de cada versão. O fato de ser mais difícil programar no nível do *kernel* e também por não ser um *switch* com foco apenas em *OpenFlow*, tornam o OVS uma escolha mais difícil para rápida prototipagem de novas funcionalidades do protocolo.

LINC. O LINC é um *software switch* escrito em *Erlang*. É um programa de nível do usuário e suporta *OpenFlow* 1.2 e 1.3, com integração ao Mininet. Uma grande vantagem do LINC para os outros é o suporte ao protocolo de gerência dos *switches OpenFlow* chamado OF-CONFIG. O maior desafio para o LINC está na linguagem, pois exige bom conhecimento em Erlang.

Trema Edge Switch. Este *software switch* faz parte do código do controlador Trema, desenvolvido pela NEC. O interessante desse *switch* é poder criar topologias virtuais utilizando apenas uma ferramenta, diferente das outras que usam o *Mininet*. O código ainda está em fase de transição para o repositório principal do Trema.

3. Arquitetura

A arquitetura do *software switch* segue a especificação do *OpenFlow 1.3*. A Figura 1 mostra os principais componentes da ferramenta.

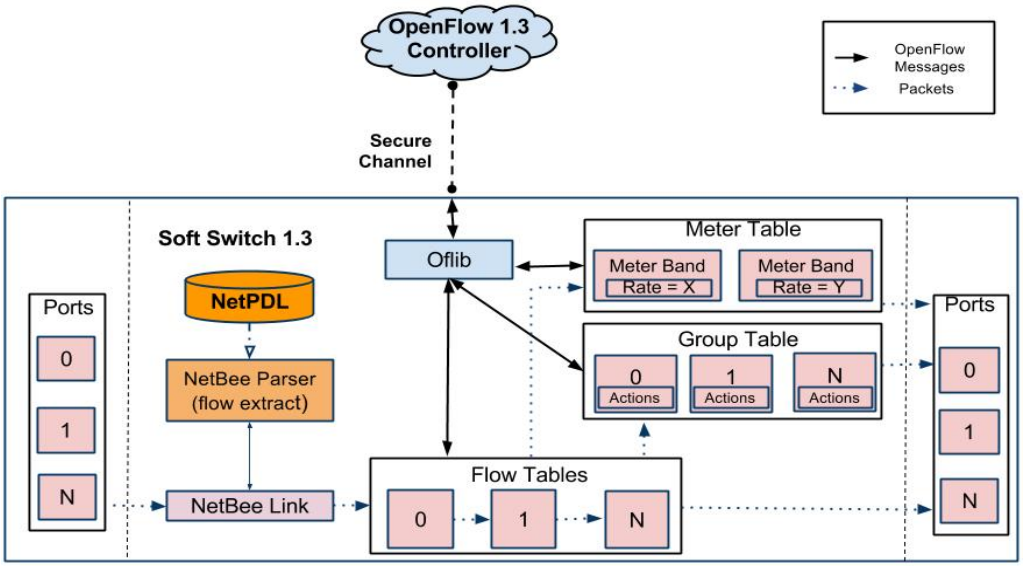


Figura 1. Arquitetura do Software Switch 1.3.

Ports: São as portas do *software switch*, nas quais os pacotes são recebidos e enviados. A ferramenta pode utilizar portas físicas e virtuais do equipamento em que está sendo executado. Na Figura 1 as portas estão refletidas para demonstrar o fluxo dos pacotes.

NetPDL: É uma linguagem definida em XML para descrever protocolos de rede [Risso and Baldi 2006]. A ferramenta utiliza um arquivo com os protocolos suportados pela especificação do *OpenFlow* 1.3, o qual é utilizado na decodificação dos pacotes pela biblioteca *NetBee*.

NetBee Parser: Biblioteca desenvolvida para vários tipos de processamento em pacotes, como decodificação e filtragem, por exemplo [NetGroup 2004]. A biblioteca constrói uma árvore em memória a partir do arquivo *NetPDL*, a qual é usada pelo componente NetBee Link para a decodificação.

NetBee Link: Utiliza a biblioteca NetBee para decodificar os pacotes que chegam pelas interfaces do *software switch*. Os campos de protocolo do pacote são convertidos na estrutura de *match* a qual será enviada para as tabelas de fluxos.

Flow Tables: As múltiplas tabelas, presentes desde a versão 1.1, é o componente no qual os fluxos são instalados e verifica-se se nenhum pacote possui uma entrada correspondente. Se existir uma entrada, o pacote é processado por alguma das instruções descritas na especificação, caso contrário, o mesmo é descartado.

Group Table: A tabela de grupos foi adicionada ao OpenFlow 1.1 para proporcionar novas formas de encaminhamento. Cada grupo possui um conjunto de ações chamados *Actions Buckets*. Os pacotes chegam à tabela de grupos depois de um pacote ser processado por uma ação de grupo.

Meter Table: Funcionalidade adicionada na versão 1.3 do protocolo, a *Meter Table* é uma tabela utilizada para limitar a banda por fluxo. Os pacotes são enviados para ela através de uma instrução contida em algum fluxo das tabelas de fluxos, para serem processados por estruturas chamadas *Band*. Um *Band* possui uma taxa de limite e é acionado no caso do fluxo de pacotes ultrapassar esse valor. A *Meter Table* pode ter múltiplos *Band*, sendo que no caso do fluxo ultrapassar o limite de vários é escolhido aquele com a a maior taxa próxima da banda passante atual.

Oflib: É a biblioteca utilizada para converter as mensagens *OpenFlow* do formato de rede para um formato interno, mais eficiente e fácil de trabalhar (geralmente sem o *padding* explicíto nas estruturas apresentadas na especificação) , e do formato interno para o de rede.

Secure Channel: Reponsável pela conexão entre o *software switch* e o controlador. Apesar do nome, no momento é possível realizar apenas conexões TCP não seguras.

4. Funcionalidades do *OpenFlow* 1.3 implementadas

A especificação 1.3 do *OpenFlow* adicionou significativas funcionalidades ao protocolo. Quase todas foram implementadas, com exceção apenas de conexões auxiliares utilizando o protocolo UDP, pois a especificação não é clara a respeito de como o *switch* deve lidar com mensagens perdidas, uma vez que no UDP não há reenvio de mensagens. Abaixo, segue a lista de modificações e comentários a respeito de como foram implementadas no software switch.

Table miss. No *OpenFlow 1.3* não há mais opções para determinar o comportamento das tabelas, no caso de não existir um fluxo instalado para algum tipo de pacote. A implementação segue o padrão de descartar esses pacotes, deixando a responsabilidade para o usuário do *switch* instalar o fluxo especial, chamado de *table-miss*, para evitar o descarte.

Cabeçalhos estendidos do IPv6. O trabalho realizado no trabalho para estender o *OpenFlow 1.1* para IPv6 contava com os cabeçalhos estendidos, porém a versão 1.3 verifica apenas a presença desses campos. Portanto foi necessário reescrever o código antigo do parsing para atender esse novo cenário, principalmente a verificação da ordem dos cabeçalhos.

Flow Meter. Foram implementados os dois tipos descritos na especificação: *Drop*, no qual a limitação é feita com o descarte dos pacotes que ultrapassem o limite de banda, e *DSCP Remark*, em que a precedência de descarte do campo DSCP do pacote é aumentada, possibilitando a criação de um serviço diferenciado.

O mecanismo de medida e limitação da banda por fluxo foi implementado utilizando o algoritmo de *Token Bucket*.

Filtragem de eventos por conexão. Essa funcionalidade permite ao controlador configurar o *switch* para não enviar determinados tipos de mensagem assíncrona. Foi implementada a filtragem tanto para os papéis de mestre quanto para escravo, habilitando ao desenvolvedor da aplicação especializar o controlador para receber ou não mensagens específicas.

Conexões auxiliares. O *switch* pode abrir conexões auxiliares com os controladores para melhor explorar o parelelismo de algumas operações. Com um identificador auxiliar o *switch* identifica a conexão principal e pode ser configurado para receber tipos especificos de mensagens, como *packet-in*.

Adição dos campos *MPLS BOS* e *PBB*. Foram adicionados os campos de *MPLS Bottom of the Stack* e o *I-SID* do protocolo *Provider Backbone Bridging*. O suporte à PBB também exigiu a implementação das operações de *Push* e *Pop*, para adicionar e remover o cabeçalho respectivamente.

Há outras mudanças menores, como desabilitar os contadores dos fluxos e a adição do campo cookies no packet-in, mas, pela simplicidade, não serão destacadas aqui.

5. Resultados e Casos de Uso

Vários resultados podem ser contabilizados até o momento.

Desenvolvimento de comunidade open source. A popularidade na comunidade é considerada um desses resultados. A Figura 2 apresenta a quantidade de visitas e visitantes no período de 1 a 14 de Janeiro e mostra também o número de pessoas acompanhando o repositório (42) e *forks* (56) do código. Até o momento foram realizados 219 commits, de 12 contribuintes diferentes [1] e já foram postados 82 *Issues* com informação de *bugs*, sugestões e perguntas. Estes números demonstram constante interesse da comunidade em utilizar e contribuir para o projeto.

Integração no Mininet. Com a popularidade, a ferramenta acabou sendo integrada ao *script* de instalação do emulador *Mininet*, alcançando um número ainda maior de usuários.

Roteador Wireless. Outro resultado relevante é a adaptação do código para o sistema operacional *OpenWrt*, permitindo utilizar *OpenFlow 1.3* em roteadores com esse sistema, habilitando testes em um ambiente mais real e também com interfaces sem fio.

Outros resultados. Como resultado indireto considera-se outros artefatos gerados, devido à necessidade de testar o *software switch*. A adaptação do controlador NOX, do *plugin* do *Wireshark* e do *framework* de testes OFtest para *OpenFlow* 1.3 fazem parte de um ambiente com todos os recursos necessários para experimentar o *OpenFlow* 1.3.

5.1. Casos de uso

Dentre os resultados, consideram-se os casos de uso conhecidos, pois como um projeto aberto e com foco na experimentação, o emprego da ferramenta em variados cenários são fatores do grau de sucesso alcançado.

[1] A verdade é que há mais, os 12 usuários são contribuições diretas via *Github*.

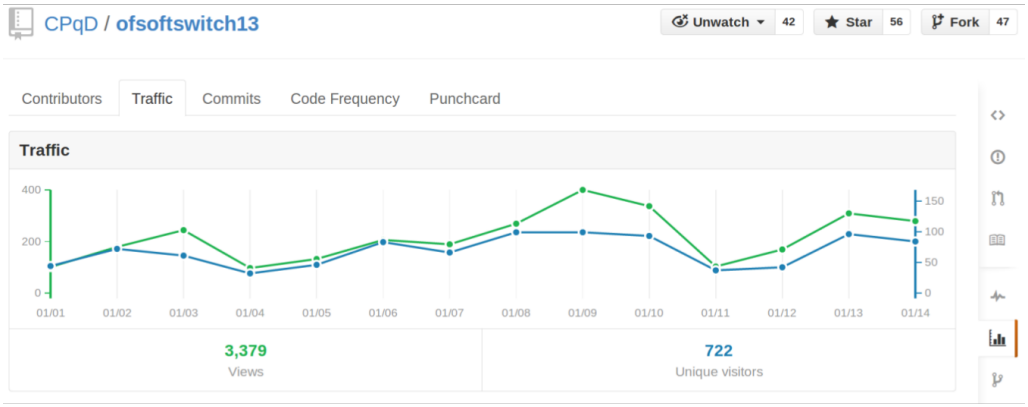


Figura 2. Estatísticas de acesso do of13softswitch no repositório *Github*.

Plataforma para incubação e padronização de novas funcionalidades. Membros da ONF têm empregado o *software switch* para prototipar em código aberto e validar experimentalmente novas propostas funcionalidades do protocolo *OpenFlow* [Tourrilhes 2013], requisitos necessários para novas extensões e funcionalidades serem aceitas na especificação do padrão *OpenFlow*.

Acadêmico. A ferramenta tem sido utilizada para ministrar cursos de introdução à *OpenFlow* e *SDN*. Além disso, aparece em dissertações de mestrado seja para adicionar novas funcionalidades [Shahmir Shourmasti 2013] ou como ferramenta de apoio [Arora 2013].

Indústrial. Empresas estão utilizando a ferramenta para testes de interoperabilidade com controladores, provas de conceito e até criando portes comerciais para implementações *hardware*, principalmente na linha de *Network Processors*.

6. Documentação, Código e Demonstração

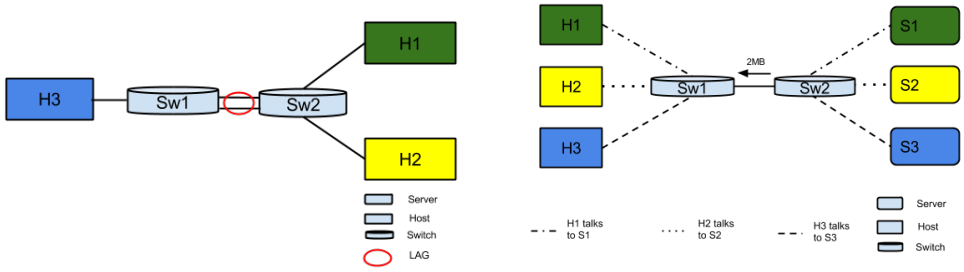
O *software* é de código aberto, licença BSD, e está disponível no *Github*, na página https://github.com/CPqD/ofsoftswitch13. O uso da ferramenta de issues do *Github*, no endereço *https://github.com/CPqD/ofsoftswitch13/issues* é encorajado para reportar *bugs* ou discutir sobre as funcionalidades do *software switch*. Contribuições podem ser submetidas realizando um *fork* do código e utilizando o recurso de *pull request* do *Github*.

A documentação está no endereço *https://github.com/CPqD/ofsoftswitch13/wiki* e inclui além de tutorial do *OpenFlow* 1.3, os comandos para utilizar o *Dpctl* e como compilar o *switch* para um roteador com o sistema operacional *OpenWrt*.

A demonstração focará em funcionalidades menos exploradas das versões mais novas do *OpenFlow*. Para isso serão criadas duas aplicações para o controlador Ryu [Open Source Software Computing Group 2012], as quais controlarão diferentes topologias no *Mininet*:

Link Aggregation (LAG) Esse cenário demonstra uma das aplicações do uso de tabelas de grupos do *OpenFlow*. O cenário da Figura 3(a) mostra os *switches* Sw1 e Sw2 conectados através de uma agregação de *links*. A demonstração focará na redundância

criada pelo *LAG*, mostrando que, ao derrubar um dos *links*, os *hosts* H1 e H2 ainda serão capazes de se comunicar com H3.



(a) Link Aggregation (b) Limitação da banda por fluxo

Figura 3. Topologias da demonstração

Limitação de Banda por fluxo. Essa aplicação demonstrará como pode ser realizada a limitação da banda no cenário de uma rede doméstica. A topologia dessa demonstração é a da Figura 3(b), em que há seis *hosts*: H1, H2 e H3 são os clientes da rede, enquanto S1, S2, S3 representam os servidores de onde os usuários obtêm conteúdo. Os clientes estão ligados a Sw1 enquanto os servidores estão conectados a Sw2. Entre os dois *switches OpenFlow* há um *link* com velocidade de 2MB. Inicialmente todos os usuários compartilham essa mesma banda sem reserva alguma, demonstrado utilizando o programa IPERF entre os clientes e servidores. Em seguida, após a instalação de alguns fluxos com instrução de *Meter*, o trafêgo para H2 e H3 será limitado para 500KB/s, garantindo a H1 o total de 1MB do *link* entre Sw1 e Sw2.

7. Trabalhos Futuros

A atualização do código para as revisões, já foram lançadas até a versão 1.3.3, é uma das prioridades para o futuro próximo. Além disso, testes além do OFTest, como os de certificação do *Ryu* e da plataforma de testes *Twister*, mostram falhas em algumas funcionalidades, portanto a correção desses problemas também deve ser priorizada.

No longo prazo, o objetivo é transformar o *software switch* em uma plataforma de prototipação mais modular, de modo que uma nova funcionalidade possa ser adicionada sem a necessidade de modificações em vários pontos do código. Para isso será necessário remodelar toda a arquitetura, buscando um *design* fácilmente extensível. Outra mudança é a remoção da biblioteca *Netbee* e utilizar outro mecanismo para decodificar os pacotes, pois vários usuários relataram problemas devido a velocidade de atualização não acompanhar a dos sistemas operacionais. Por fim, novas versões do protocolo sempre estão em pauta, e a implementação da versão 1.4 seria um passo natural para manter o compasso com a versão mais recente.

8. Conclusão

O *software switch* da versão 1.3 do *OpenFlow* é uma ferramenta bem estabelecida como projeto *Open Source*, com uma boa base de usuários e contribuidores. Com características que o tornam a opção mais fácil para prototipar novas funcionalidades, como código mais simples em linguagem C e a facilidade de uso, tem sido a escolha da ONF para prototipação e validação de novos recursos. Os experimentos na academia, utilizado em

mestrados e aulas, e na indústria reforçam a utilidade da ferramenta no desenvolvimento do protocolo *OpenFlow* e na experimentação de aplicações com requisitos cobertos apenas pela versão 1.3 em diante.

9. Agradecimentos

Agradecemos ao Centro de Inovação da Ericsson Brasil, pelo suporte financeiro e a manutenção do projeto com o código aberto, e a Ericsson *Research* pela colaboração técnica. E a comunidade de usuários do projeto, pelas contribuições.

Referências

Arora, D. (2013). Proactive routing in scalable data centers with paris.

Denicol, R. R., Fernandes, E. L., Rothenberg, C. E. R., and Kis, Z. L. (2011). On ipv6 support in openflow via flexible match structures. *OFELIA/CHANGE Summer School*.

Feldmann, A. (2007). Internet clean-slate design: What and why? *SIGCOMM Comput. Commun. Rev.*, 37(3):59–64.

Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., and Shenker, S. (2008). Nox: Towards an operating system for networks. *SIGCOMM Comput. Commun. Rev.*, 38(3):105–110.

Kis, Z. L. (2011). OpenFlow 1.1 software switch. `https://github.com/TrafficLab/of11softswitch`. [Online; acessado 14-Janeiro-2014].

Lantz, B., Heller, B., and McKeown, N. (2010). A network in a laptop: Rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, pages 19:1–19:6, New York, NY, USA. ACM.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74.

NetGroup (2004). The NetBee Library. `http://www.nbee.org/`. [Online; acessado 14-Janeiro-2014].

ONF (2013). Open Network Foundation Website. `https://www.opennetworking.org/`. [Online; acessado 14-Janeiro-2014].

Open Source Software Computing Group (2012). Ryu SDN Framework. `http://osrg.github.io/ryu/`. [Online; acessado 15-Janeiro-2014].

Pfaff, B., Pettit, J., Koponen, T., Amidon, K., Casado, M., and Shenker, S. e.a.: Extending networking into the virtualization layer. In *In: 8th ACM Workshop on Hot Topics inNetworks (HotNets-VIII).New YorkCity,NY(October 2009*.

Risso, F. and Baldi, M. (2006). Netpdl: An extensible xml-based language for packet header description. *Comput. Netw.*, 50(5):688–706.

Shahmir Shourmasti, K. (2013). Stochastic switching using openflow.

Shenker, S. (1995). Fundamental design issues for the future internet. *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, 13:1176–1188.

Tourrilhes, J. (2013). Prototyping on SoftSwitch. `https://github.com/jean2/ofsoftswitch13`. [Online; acessado 15-Janeiro-2014].