ESTUDO E TESTE DO PROTOCOLO DE REDE OPENFLOW

LIMA, Gustavo de Oliveira(1); ASSIS, Luiz Luan Beijo de (1); RAMOS, Celso de Ávila (2);

(1)Acadêmicos de Ciência da Computação da UNIFENAS; (2) Orientador.

Abstract. Openflow is an open protocol that allows you to remotely control forwarding tables of network switches, routers and other access points. Using low-level primitives, researchers can build new networks with high-level properties. The Openflow enables you to implement security on networks that do not present this as default, also allows the implementation of scalable networks, hosts, and networks mobility more efficient as energy and new technologies for long distance. With Openflow is possible, control flows by choosing the routes of the packages and the processing that they will receive. It becomes possible to create and experiment of new protocols, new models and new addressing schemes.

Keywords: OpenFlow,Protocol, Network.

Resumo. *Openflow* é um protocolo aberto que permite controlar remotamente tabelas de encaminhamento dos *switches* de rede, roteadores e de outros pontos de acesso. Utilizando primitivas de baixo nível, pesquisadores podem construir novas redes com propriedades de alto nível. O *Openflow* permite implementar segurança em redes que não apresentam isso como padrão, também permite a implementação de redes escaláveis, mobilidades de *hosts*, e redes mais eficientes quanto à termos energéticos e novas tecnologias de longa distância. Com o *Openflow* é possível, controlar os fluxos escolhendo as rotas dos pacotes e o processamento que eles iram receber. Com isso torna-se possível a criação e o experimento de novos protocolos, novos modelos de segurança e novos esquemas de endereçamento.

Palavras-chave: OpenFlow; Protocolo; Rede.

1 INTRODUÇÃO

Muitos dos protocolos e das arquiteturas de rede utilizadas ainda hoje são antigos ou ultrapassados e muitas vezes não são capazes de atender a demanda. Uma das causas são os diferentes componentes de *hardware* relacionados a redes que são operados por *softwares* individuais.

Com o passar do tempo, tornou-se uma necessidade que esses *softwares* controladores fossem substituídos por um *software* em um só protocolo padrão e que a rede fosse controlada remotamente por uma estação de controle. Nasce o termo Redes definidas por *software*, *OpenFlow* é uma delas.

O *OpenFlow*, é uma proposta tecnológica que, baseada no modelo de controle de rede logicamente centralizado, permite a pesquisadores executarem seus experimentos em redes utilizadas no dia-a-dia, sem interferir no tráfego de produção. *OpenFlow* é um protocolo de comunicação que permite total controle sobre as tabelas de fluxos presentes no *switches*. O conceito de fluxo é o do bloco fundamental que habilita os pesquisadores a “setar” o plano de encaminhamento na rede conforme os objetivos definidos pelas novas propostas de arquiteturas e de protocolos de rede. O *OpenFlow* também define um novo elemento de rede, o controlador.

O *OpenFlow* é uma forma de executar experimentos nas redes que usamos todos os dias. A contribuição mais importante do paradigma do *OpenFlow* é a generalização do plano de dados. Isso quer dizer que qualquer modelo de encaminhamento de dados que se baseie na tomada de decisão fundamentada em algum valor do campo de cabeçalho dos pacotes pode ser suportada. As entradas correspondentes na tabela de fluxos podem ser interpretadas como decisões em *cache (hardware)* do plano de controle *(software)*. Um fluxo na rede é, portanto, nada mais que a mínima unidade de controle para criar sistemas escaláveis usando *OpenFlow*. Trata-se de um compromisso pragmático, no qual permite a pesquisadores realizarem experimentos em suas redes sem exigir que os fabricantes exponham o funcionamento interno dos equipamentos.

1.1 Objetivos

Este trabalho tem como objetivo geral apresentar o protocolo *OpenFlow* e seus princípios básicos sobre os controles. E especificamente, simular uma rede virtual definida por software, fazendo um comparativo de velocidade entre controles.

1.2 Justificativas

Um tema que desperta grande interesse, o *OpenFlow* é uma nova proposta que promete quebrar paradigmas com uma nova forma de transmissão de fluxos, além do mais, é uma proposta de estudos da RNP - Rede Nacional de Pesquisa, o que comprova a grande importância do assunto.

.

1.3 Hipóteses

O presente trabalho buscará mostrar o funcionamento de uma rede controlada por *software*, virtualmente, visando um resultado satisfatório se comparado às redes atuais. Com os conhecimentos e o material disponível, a simulação possui grandes chances de sucesso.

2 REFERENCIAL TEÓRICO

2.1 *OpenFlow*e redes definidas por *software*

A infraestrutura das redes de pacotes é composta, atualmente, por equipamentos proprietários, fechados e de alto custo, cujas arquiteturas básicas são concebidas a partir da combinação de circuitos dedicados, responsáveis por garantir

alto desempenho (*Application Specific Integrated Circuit* – ASIC), ao processamento de pacotes. A infraestrutura é complementada por uma camada de *software* de controle, responsável pelo suporte a uma extensa pilha formada por um número elevado de protocolos. Torna-se, no entanto, evidente a necessidade de especialização da lógica de controle de acordo com cada tipo e objetivo de rede (ROTHENBERG, 2010).

Qualquer mudança de configuração avançada do equipamento ou da especialização da lógica de controle e de tratamento dos pacotes ou, ainda, a inserção de novas funcionalidades estão sujeitas a ciclos de desenvolvimento e de testes restritos ao fabricante do equipamento, resultando em um processo demorado e custoso (HAMILTON, 2009).

Em contraste com o desenho da arquitetura da *Internet*, caracterizada por um Plano de Controle (PC) distribuído, os avanços na padronização de APIs *(Application Programming Interface)* independentes do fabricante do equipamento, permitem mover grande parte da lógica de tomada de decisão dos dispositivos de rede para controladores externos, que podem ser implementados com o uso da tecnologia de servidores comerciais, um recurso abundante, escalável e barato. Essa "lobotomia" da inteligência do equipamento da rede para controladores logicamente centralizados possibilita a definição do comportamento da rede em *software* não apenas pelos fabricantes do equipamento, mas também por fornecedores ou pelos próprios usuários, como, por exemplo, por operadores de rede (OPENFLOW, 2010; IETF, 2011).

De acordo com Juniper Networks (2010), uma análise da arquitetura atual dos roteadores permite observar que se trata de um modelo formado basicamente por duas camadas bem distintas: o *software* de controle e o *hardware* dedicado ao encaminhamento de pacotes. O primeiro, encarregado de tomar as decisões de roteamento, transfere essas decisões para o plano de encaminhamento através de uma API proprietária. A única interação da gerência com o dispositivo ocorre através de interfaces de configuração (*Web*, SNMP - *Simple Network Management Protocol*, CLI - *Command Line Interface*, por exemplo), limitando o uso dos dispositivos às funcionalidades programadas pelo fabricante.

É coerente pensar que se a arquitetura é, atualmente, composta por duas camadas autocontidas, estas não precisam estar fechadas em um mesmo equipamento. Para isso, basta que exista uma forma padrão de se programar

remotamente o dispositivo de rede, permitindo que a camada de controle possa ser movida para um servidor dedicado e com alta capacidade de processamento. Desse modo, mantém-se o alto desempenho no encaminhamento de pacotes em *hardware,* aliado à flexibilidade de se inserir, de se remover e de se especializar aplicações em *software* por meio de um protocolo aberto para programação da lógica do equipamento. Com esse propósito, nasceu o consórcio *OpenFlow* (MCKEOWN et al., 2008), dando origem ao conceito de *software defined networking* – as redes definidas por *software* (GREENE, 2009).

2.2 Funcionalidade do *OpenFlow*

O *OpenFlow* foi proposto pela Universidade de Stanford para atender à demanda de validação de novas propostas de arquiteturas e de protocolos de rede (incluindo as abordagens *clean slate*) sobre equipamentos comerciais. *OpenFlow* define um protocolo-padrão para determinar as ações de encaminhamento de pacotes em dispositivos de rede, como, por exemplo, comutadores, roteadores e pontos de acesso sem fio (ROTHENBERG, 2010).

De forma pragmática, a especificação *OpenFlow* (OPENFLOW, 2010) procura reutilizar as funcionalidades do *hardware* existente (por exemplo, *Access ControlList*-ACL - em *switches* e roteadores para implementar serviços como NAT – *Network AddressTranslation*, *firewall* e VLANs - *Virtual Local Area Network*) por intermédio da definição de um conjunto simples de regras e das ações associadas: encaminhar, descartar, enviar para o controlador, reescrever campos do cabeçalho, etc.

2.3 Premissas básicas sobre *OpenFlow*

O *Openflow* é composto pela tabela de fluxos onde cada entrada possui uma ação associada à mesma, esta tabela é composta por cabeçalho, ações e contadores. Além da tabela de fluxos o *OpenFlow*possuí o *SecureChannel* (Canal

Seguro) que utiliza protocolo SSL para maior segurança nas comunicações.O protocolo em si permite gerenciar e controlar a rede e a comunicação entre *switch* e controlador evitando-se, assim, a necessidade de um *switch* programável (MACAPUNA, 2010).

Para cada pacote que chega ao *switch,* uma ação é associada. Ele pode ser encaminhado para uma determinada porta, pode ser encapsulado e transmitido para o controlador, pode ser descartado ou encaminhado para as camadas 2 e 3 (isso permite que o tráfego experimental não interfira no tráfego de produção) (MACAPUNA, 2010).

2.4 Controlador

O controlador é o *software* responsável por tomar decisões e por adicionar e por remover as entradas na tabela de fluxos, de acordo com o objetivo desejado. O controlador exerce a função de uma camada de abstração da infraestrutura física, facilitando a criação de aplicações e de serviços que gerenciem as entradas de fluxos na rede. Esse modelo assemelha-se a outros sistemas de *software* que proveem abstração do *hardware* e a funcionalidade reutilizável. Dessa forma, o controlador *OpenFlow* atua como um Sistema Operacional (SO) para gerenciamento e para controle das redes, além de oferecer uma plataforma com base na reutilização de componentes e na definição de níveis de abstração (comandos da API). Novas aplicações de rede podem, contudo, ser desenvolvidas rapidamente (GUDE et al., 2008).

2.5 Fatores do sucesso do *OpenFlow*

O *OpenFlow* pode ser incorporado em equipamentos de rede comerciais sem modificação de *hardware* mediante a atualização do *firmware,* garantindo desempenho. O protocolo separa o plano de controle do plano de dados e permite a utilização de controladores remotos baseados em servidores com sistema operacionais (ROTHENBERG, 2010).

Também oferece um serviço de encaminhamento multicamada orientada a fluxos definidos por qualquer combinação de mais de 20 cabeçalhos-

padrão. Uma rede *OpenFlow* permite a definição de fatias de rede com garantia de isolamento entre os diferentes controladores permitindo o tráfego experimental (definido pelo usuário) e o tráfego normal (conforme protocolos antigos) operem e paralelo. Além disso o protocolo é compatível com *Internet* atual que pode continuar em uma ou mais fatias de rede (ROTHENBERG, 2010).

Todos os argumentos citados apontam que o *OpenFlow* é uma tecnologia inovadora que abre uma série de novas oportunidades de desenvolvimento tecnológico na área de redes de pacotes (ROTHENBERG, 2010).

2.6 Mercado e aplicações

Empresas como HP (*Hewlett-Packard*) e Nec *Extreme* começaram a desenvolver protótipos com suporte ao *OpenFlow*. Surgiram novas empresas (Nicira*, Big Switch Networks*) e operadoras como DeustcheTelekom e Docomo, além de serviços de Nuvem como Google, Facebook e Amazon começaram a investir no *OpenFlow* (ROTHENBERG, 2010).

Dentre os aspectos promissores da tecnologia *OpenFlow,* vale destacar as redes corporativas (novos mecanismos de controle de acesso de segurança); *backbones* (convergência de redes de pacotes e circuitos); redes de celulares (uso transparente de diversas redes de acesso separação do provedor de infraestrutura do provedor de serviços); *data centers* (conservação de energia, engenharia de tráfego); redes domésticas e redes de ensino e pesquisa (ROTHENBERG, 2010).

3 MATERIAIS E MÉTODOS

Para realização do trabalho, foi utilizado a *internet* para a pesquisa bibliográfica. Para os experimentos, foi utilizado a máquina virtual *Mininet*, que possui k*ernel Linux,* possibilitando, a criação de redes virtuais e experimentação de protocolos através de redes definidas por *softwares,* tornando possível também, a criação de redes virtuais com *hosts*, *switches* e controlador. O *software* para

simulação escolhido para realização do trabalho foi o *Vmware*. Também foi utilizado o *Putty* para acesso remoto à máquina virtual, isso foi necessário devido a utilização de várias janelas, e o *Xming* que permite que aplicativos gráficos do *Linux* sejam exibidos no *Windows*.

Dentro da simulação, foi utilizado o *Wireshark* para captura de pacotes e análise de protocolo. Durante os experimentos, foi utilizado o controle *Beacon* baseado em *Java*.

4 DESENVOLVIMENTO

Como guia desse experimento foi utilizada a documentação do *Mininet*.

A documentação *Mininet* contém uma riqueza de informações úteis, incluindo como usar *Mininet* e muito mais. DOCUMENTATION MININET.2013.

A fase de desenvolvimento resumiu-se em, Instalação da máquina virtual; configuração do *Putty*, demonstração da tabela de fluxos, utilização do *Wireshark* para capturas de pacotes, demonstração das ações do ping, utilização do controle ptcp. B*enchmarking* básico de velocidade, instalação e demonstração do controle *Beacon*. O experimento foi descrito de maneira didática de maneira que futuramente alguém possa realizar outros testes e quem sabe aprimorar os resultados.

5 RESULTADOS E DISCUSSÃO

Um ambiente simulado, caracterizado pelo uso de máquinas virtuais, é sem dúvida uma alternativa muito útil na informática para a realização de experimentos, porém alguns experimentos, principalmente os mensuráveis, podem apresentar resultados diferentes dos resultados reais. Isso porque em um ambiente simulado não existe equipamentos reais interconectados com uma interface de rede, tornando desconhecido qualquer dificuldade ou problema entre a configuração desses equipamentos em um ambiente real.

Em análise dos resultados, foi observado que, quando o teste de velocidade era realizado junto com o *Wireshark* aberto, um pequeno atraso acontecia na velocidade. Sendo assim, foram separadas duas tabelas com resultados dos testes, a TAB. 1 mostra os resultados dos testes de velocidade com o *Wireshark*, enquanto a TAB. 2 mostra os resultados dos testes de velocidade sem o *Wireshark*.

TABELA 1 - Teste de velocidade com *Wireshark*:

Controle	**Faixa de Velocidade Atingida**	Média
ptcp:	Entre 1,65gbps até 1,80gbps	1,725gbps
User Switch	Entre 365mbps até 425mbps	395mbps
Beacon	Entre 1,81gbps até 1,97gbps	1,89gbps

TABELA 2: Teste de velocidade sem *Wireshark*.

Controle	**Faixa de Velocidade Atingida**	**Média**
ptcp:	Entre 1,88gbps até 2,55gbps	2,215gbps
User Switch	Entre 418mbps até 447gbps	432,5gbps
Beacon	Entre 1,86gbps até 2,60gbps	2,23gbps

É perceptível, por meio dos resultados, estabelecer que o controle *Beacon* é quase que equivalente ao controle *ptcp*. Ambos são baseados em *Learning switch.* Vale ressaltar que o controle *Beacon* está executando na própria máquina, utilizando outra interface de rede, portanto outro *hardware* e o *ptcp* se encontram na própria máquina virtual e está sendo executado localmente com o *hardware* da máquina virtual.

O controle *Beacon* pode ser utilizado remotamente. Para complementar, o experimento foi realizado um teste remoto com uma máquina externa, um *desktop core* 2 *duo* com 4gb de memória. A TAB. 3 mostra os resultados deste teste.

TABELA 3 - Resultado do teste remoto

Situação	Faixa de Velocidade	Média
Com Wireshark	1,80 a 1,93gbps	1,865 gbps
Sem Wireshark	1,77 a 2,01	1,89 gbps

Mesmo estando em outra máquina, o controle *Beacon* consegue ter um bom desempenho e, com o *Wireshark* aberto, a rede conseguiu ter um desempenho equivalente.

Conhecendo as interfaces de redes atuais, percebe-se que a velocidade conseguida durante os testes é somente possível utilizando-se o padrão IEEE 802.3 *Ethernet* 10 *gigabit*. Vale notar, entretanto, que a rede na qual foram feitos os testes é virtual, não emula uma interface real de rede, apenas a conexão. Isso quer dizer que a comunicação entre o *switch* e o *host* é quase que instantânea, não apresentando perdas como acontece nos meios de transmissão de um ambiente real, mas mesmo assim isso pode mostrar a capacidade de uma rede definida por software.

Além disso, o *OpenFlow* depende muito do *software* de controle, se esse *software* controlador for mal implementado ou até mesmo for mal configurado pode comprometer toda rede, podendo apresentar resultados bem inferiores aos esperados.

Foi procurado um meio de se estimar a velocidade real. Em uma rede *Ethernet* comum de 100mbps, poder-se-iam dividir os resultados obtidos nos testes por 100 e estimar a velocidade em uma rede real, mas esse resultado seria impreciso, pois não se tem conhecimento se há um limite de velocidade no *Mininet*. Então, teria que ser tomado como base que a velocidade máxima que um *switch* virtual pode alcançar seria 2,6gbps que é o maior resultado obtido nos testes. Sendo assim, poderia se estimar a velocidade de comunicação real, utilizando regra de três, caso a rede fosse *Ethernet*. Usando como exemplo o resultado a velocidade de comunicação com *Beacon* em outra máquina, a velocidade real estimada seria de 71,73mbps conforme mostra a FIG. 1.

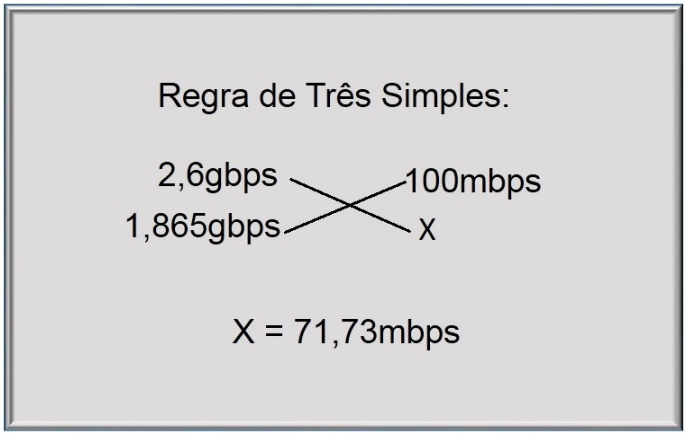

FIGURA 1 - Estimativa de velocidade

Considerando que a outra máquina está conectada via *Wireless* com roteador com padrão IEEE 802.11n e a velocidade máxima de comunicação com o controlador seria de 150mbps (velocidade do roteador). Por isso pode-se concluir que o controlador não interfere tanto na comunicação do *switch* com o *host*. No exemplo, ele identifica para qual porta de saída e para qual estação os pacotes devem ir. Isso é possível por meio da tabela de MAC que armazena o MAC's de todos os *hosts*. O controlador fica responsável por adicionar o fluxo na tabela do *switch*, e o *switch* fica responsável apenas por encaminhar os pacotes.

O conceito de redes definidas por *software* lembra muito a topologia de estrela em que todas as estações dependem de uma só estação de controle (redes centralizadas). Sendo assim, o protocolo *OpenFlow* fica dependente da estação de controle. Com a análise do conteúdo estudado, foi percebido que isso pode ser contornado facilmente utilizando-se:

- Servidores de Reserva;
- *Backup,* configuração do *switch.* Caso a estação de controle falhe, o *switch* passa a funcionar normalmente utilizando protocolos antigos até que a estação seja reparada;
- Controles distribuídos (*clusters*) para redes de grandes escalabilidades.

Pode-se observar que os estudos referentes ao protocolo *OpenFlow* estão em fase inicial e expandindo cada vez mais, mas ainda há muito o que ser estudado. Entre as promessas do *OpenFlow,* vale destacar o conceito que se refere à segurança, à maneira como se pode implantá-la ou até mesmo melhorá-la.

Experimentar isso em uma rede real e funcional seria uma maneira eficaz de conhecer melhor o protocolo.

Ainda pode demorar algum tempo para o *OpenFlow* tomar a frente no mercado, porém, com muita pesquisa e tempo, irá chegar o momento em que as redes vão passar a ser controladas de maneira inteligente, eficiente e à prova de falhas.

6 CONCLUSÃO

Dentre os objetivos descritos nesta monografia, conseguiu-se destacar o conceito de redes definidas por *software*, nas quais uma rede passa a ser controlada por uma estação de controle e deixa de depender do *software* presente nos *hardwares* de rede. Isso gera diversos pontos positivos como flexibilidade (maior controle sobre a rede) e maior desempenho e segurança.

Com os objetivos específicos deste trabalho, foi realizada a criação de uma rede virtual utilizando o *Mininet* e, utilizando controle previamente instalado (ptcp), foi possível simular uma rede *OpenFlow* com *switch* e *host* virtuais. Foi utilizado *Wireshark* para capturar os pacotes e demonstrar a comunicação entre *switch* e o controlador. Após ter sido feita essa análise, foi demonstrado um teste de velocidade entre uma rede que utiliza controlador e outra que não utiliza. Como demonstrado nos resultados e métodos, a rede que utiliza controlador se saiu melhor.

Ainda nos objetivos, foi proposto testar um controle. Para o teste foi escolhido o controle *Beacon*. Como a solução desse controle já estava disponível no próprio projeto, foi dispensada a implementação, porém foi feita uma análise e explicação do código. Em seguida, foi feito um teste de velocidade que chegou a atingir 2,6gbps.

Nos testes, foi observado que o controle *pctp* e o controle *Beacon* obtinham resultados quase que equivalentes, mas vale considerar o fato de que o controle *Beacon* estava executando na máquina host com *hardware* superior e o controle *ptcp* estava sendo executado localmente utilizando o *hardware* da máquina virtual.

O controle *Beacon* permite que a rede seja controlada por outra máquina remotamente. Então, foram realizados testes em uma máquina *core* 2 *duo* com 4gb de RAM e foram obtidos resultados na faixa de 1,8gbps de velocidade na rede, mesmo utilizando o *Wireshark*. Isso mostra que o protocolo *OpenFlow* não fica limitado a apenas uma estação de controle, que poderá ser facilmente transportado para outra máquina caso a estação principal falhe.

Por fim, pode-se concluir que, quando a rede passa a ser controlada por um controlador ela tem um melhor desempenho, isso ocorre pois o *switch* passa a ser responsável apenas por encaminhar pacotes, assim obtém-se um melhor aproveitamento do *hardware* do *Switch* levando ao aumento de velocidade.

7 REFERÊNCIAS

ANWER, M. B. E. A. Switchblade: a platform for rapid deployment of network protocols on programmable hardware. SIGCOMM ’10. Proceedings.. New York: [s.n.]. 2010. p. 183-194.

CHOWDHURY, M.; BOUTABA, R. N. V. State of the Art and Research Challenges. IEEE Communications Magazine, v. 47, n. 7, p. 20-26, 2009.

GREENE, K. T. Software-Defined Networking. MIT Technology Review. [S.l.]: [s.n.]. 2009.

GUDE, N. E. A. NOX: towards an operating system for networks. SIGCOMM Computer Communication Review. [S.l.]: [s.n.]. p. 105-110. 2008.

HAMILTON, J. N. The last bastion of mainframe computing. [S.l.]: [s.n.]. 2009.

JUNIPER NETWORKS. Network Operating System Evolution. [S.l.]: White paper, 2010.

MACAPUNA, C. A. B. OpenFlow e NOX: Propostas para Experimentação de Novas Tecnologias de rede. Trabalho de conclusão da disciplina Tópicos em Engenharia da Computação, Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas - Unicamp. Campinas, SP, Brasil. 2010.

MCKEOWN, N. E. A. OpenFlow: enabling innovation in campus networks. SIGCOMM Computer Communication Review. [S.l.]: [s.n.]. 2008. p. 69-74.

ROTHENBERG*, C. E. et al. OpenFlow e redes definidas por software: um novo paradigma de controle e inovação em redes de pacotes. CPqD Tecnologia, Campinas, v. 7, n. 1, p. 65-76, julho 2010.

INTERNET ENGINEERING TASK FORCE (IETF). Forwarding and Control Element Separation (ForCES), WG, 2011. Disponível em: <http://datatracker.ietf.org/wg/forces/charter/>. Acesso em: 15 mai. 2013

THE OPENFLOW SPECIFICATION. 2010. Disponível em: <http://www.openflowswitch.org/documents/openf low-wp-latest.pdf>. Acesso em: 21 mai. 2013.

DOCUMENTATION MININET.2013. Disponível em: <https://github.com/mininet/mininet/wiki/Documentation>. Acesso em: 15 jul.2013.

DOWNLOAD *BEACON*. 2010. Disponível em: <https://openflow.stanford.edu/display/*Beacon*/Releases> Acesso em: 18 jul. 2013.

OPENFLOW TUTORIAL.2010. Disponível em: <http://archive.openflow.org/wk/index.php/OpenFlow_Tutorial > Acesso em: 22 mai. 2013.