ISSN: 2175-1897

Árvore de Extensão Mínima no Processo de Conservação de Energia em Redes Definidas por Software

LIGIA RODRIGUES PRETE
Centro Paula Souza - Faculdade de Tecnologia de Jales – SP – Brasil
ligia.prete@fatec.sp.gov.br

CHRISTIANE MARIE SCHWEITZER
Universidade Estadual Paulista "Julio de Mesquita Filho"– SP – Brasil
chris@mat.feis.unesp.br

AILTON AKIRA SHINODA
Universidade Estadual Paulista "Julio de Mesquita Filho"– SP – Brasil
shinoda@dee.feis.unesp.br

ROGÉRIO LEÃO SANTOS DE OLIVEIRA
Centro Paula Souza - Faculdade de Tecnologia de Jales – SP – Brasil
rogerio.leao@fatec.sp.gov.br

Resumo – O consumo de energia no setor de Tecnologia da Informação e Comunicação tem crescido exponencialmente nos últimos anos, devido à quantidade crescente de equipamentos para armazenamento e processamento de dados. Rede Definida por Software e OpenFlow estão permitindo uma nova gama de aplicações e serviços para redes. Ao mesmo tempo, reduzir o impacto de energia nos centros de dados será um tema importante nos próximos anos. O trabalho apresenta um controlador OpenFlow que cria uma topologia na camada 2 e de acordo com as fases especificadas da aplicação, permite a redução de cinquenta por cento do consumo de energia elétrica na rede Openflow.

Palavras-chave: Redes Definidas por Software, OpenFlow, Conservação de Energia.

Abstract

Energy consumption in the Communication and Information Technology sector has grown exponentially in recent years due to the increasing amount of equipment for storage and data processing. Software Defined Networking and OpenFlow by are enabling a new range of applications and services for network. At the same time, reduce the impact energy in the data center will be an important theme in the coming years. The article presents an OpenFlow controller that creates a topology at Layer 2, and according to specified steps from the application, allows the reduction of fifty percent of power consumption in the OpenFlow network.

Keywords:

Software Defined Networking, OpenFlow, Energy Conservation.

ISSN: 2175-1897

1. Introdução

Nos dias atuais, empresas, governos e sociedade de uma maneira geral, incluíram mais um item em sua lista de preocupações, sendo as questões ambientais. A necessidade de adotar medidas ecologicamente corretas, torna-se necessária para ampliar o equilíbrio ao ecossistema do planeta (SILVA et al., 2010).

Rede Definida por Software (*Software Defined Networking* - SDN) é um paradigma emergente para comunicação de dados que está mudando completamente a configuração do plano de controle e transmissão de dados nas redes. A arquitetura SDN permite a criação de novos serviços e protocolos abertos que beneficiam os administradores, por meio da otimização nas redes, permitindo que aplicações se adaptem dinamicamente em suas infraestruturas, para suprir as próprias necessidades. Uma utilização interessante do paradigma SDN é a redução do consumo de energia em redes de larga escala.

O protocolo OpenFlow (MCKEOWN et. al, 2008) é um padrão aberto, sendo a tecnologia mais difundida da arquitetura SDN. Um elemento de rede habilitado para OpenFlow delega a decisão sobre como lidar com os pacotes de entrada para um elemento externo, denominado controlador. Os dispositivos de encaminhamento e o controlador remoto comunicam-se por meio de um canal seguro. Um *switch* OpenFlow é equipado com uma tabela de base de informação de encaminhamento, armazenando regras de correspondência para os pacotes de entrada (por exemplo, encaminhar para uma porta, descartar o pacote, ou modificar seu cabeçalho) e contadores. Se um pacote de entrada corresponde a uma regra de encaminhamento, a ação correspondente é tomada e os contadores são atualizados. Quando um pacote não corresponde a nenhuma regra, ele é enviado para o controlador, onde a lógica da aplicação formula uma nova regra que será implantada no dispositivo para encaminhar o tráfego de entrada que corresponde à mesma regra. OpenFlow está sendo adotado por um número crescente de usuários de rede de grande porte.

No ambiente de redes, o Protocolo de Árvore de Extensão (*Spanning Tree Protocol* - STP) (RFC 4318, 2005) é usado para resolver problemas de laços em redes comutadas, cuja topologia introduz anéis nas ligações, auxiliando na melhor performance da rede. O protocolo STP possibilita a inclusão de ligações redundantes entre os computadores, provendo caminhos alternativos no caso de falha de uma dessas ligações. Nesse contexto, ele serve para evitar a formação de laços entre os comutadores e permitir a ativação e desativação automática dos caminhos alternativos. Para isso, o algoritmo de Árvore de Extensão Mínima determina qual é o caminho mais eficiente (menor custo) entre cada segmento separado por *switches*.

Este estudo descreve o projeto e a implementação de um controlador OpenFlow que institui uma árvore de extensão entre os dispositivos de rede, com o objetivo de aumentar o seu aspecto ecológico. A abordagem visa reduzir o impacto ambiental e os custos das redes, desativando as conexões que apresentam laços entre dispositivos OpenFlow (OPENFLOW, 2011). Adota-se

ISSN: 2175-1897

uma estratégia para aumentar a facilidade de escolha, evitando os laços e permitindo a investigação sobre os métodos rápidos de transferências em caso de falhas.

O trabalho está organizado da seguinte forma: a Seção II apresenta o referencial teórico. Seção III descreve o método, as tecnologias e o experimento utilizado para a avaliação do protótipo. Seção IV analisa os resultados obtidos em uma rede de teste OpenFlow, a partir do controlador proposto. Finalmente, a Seção V oferece as considerações finais e descreve as possíveis evoluções de atividades de investigação.

2. Referencial Teórico

Otimização do consumo de energia elétrica nas redes de centros de dados é um tema que está recebendo ampla relevância. A necessidade de minimizar os custos operacionais de infraestruturas de Tecnologia da Informação é considerada importante, tal como a conectividade de rede, que, até o momento, era considerada intocável devido ao seu papel essencial.

Uma das áreas de aplicação para OpenFlow é a sua utilização em centro de dados. Existem diferentes projetos, a fim de melhorar a eficiência energética e reduzir o consumo de energia (JARSCHEL; PRIES, 2012).

O princípio da proposta é semelhante à usada por Heller (BRANDON et. al, 2010) e Joseph (CHABAREK et al., 2008), com relação à redução do consumo de energia, desligando as interfaces não utilizadas. As diferenças com a abordagem de Heller estão em evitar a exigência de topologias de árvore gorda com um conhecimento a priori da matriz de tráfego completa entre os *hosts*. A principal contestação com Joseph, que não se baseia em OpenFlow, é que considera os fluxos individuais em vez de tráfego agregado. Além disso, o controlador opera nas ligações entre os dispositivos, em vez de alterar toda a sua configuração.

A literatura relata um número limitado de contribuições que discutem a construção de laços livres de sobreposições com OpenFlow. No momento, não há soluções comuns para a criação de sobreposições livres de laços, tampouco contribuições para discutir a adoção de ligação para reduzir o impacto ambiental de uma rede OpenFlow.

3. Método

O objetivo do trabalho é implementar uma sobreposição entre os dispositivos OpenFlow na rede subjacente. A abordagem do módulo de Árvore de Extensão Mínima (*Minimum Spanning Tree* - MST) terá um benefício considerável se comparado com as soluções mais especializadas e estará disponível para um público maior de administradores de rede.

ISSN: 2175-1897

3.1 Controlador

Diferentes estruturas foram propostas como base para desenvolver novos controladores OpenFlow. Neste trabalho, o Floodlight (PROJECT FLOODLIGHT, 2015) foi usado para implementar o protótipo. Floodlight possui código aberto, escrito em Java, suporta múltiplos fluxos de execução, inclui módulos predefinidos para representar o estado dos elementos em uma rede OpenFlow e para implementar funcionalidades básicas de rede. Os módulos predefinidos modelam o estado dos dispositivos, verificando suas mudanças e propagando a notificação para outros módulos do controlador. Os módulos adicionais apresentam funcionalidades, como *switch* de aprendizagem e capacidades de roteamento elementares.

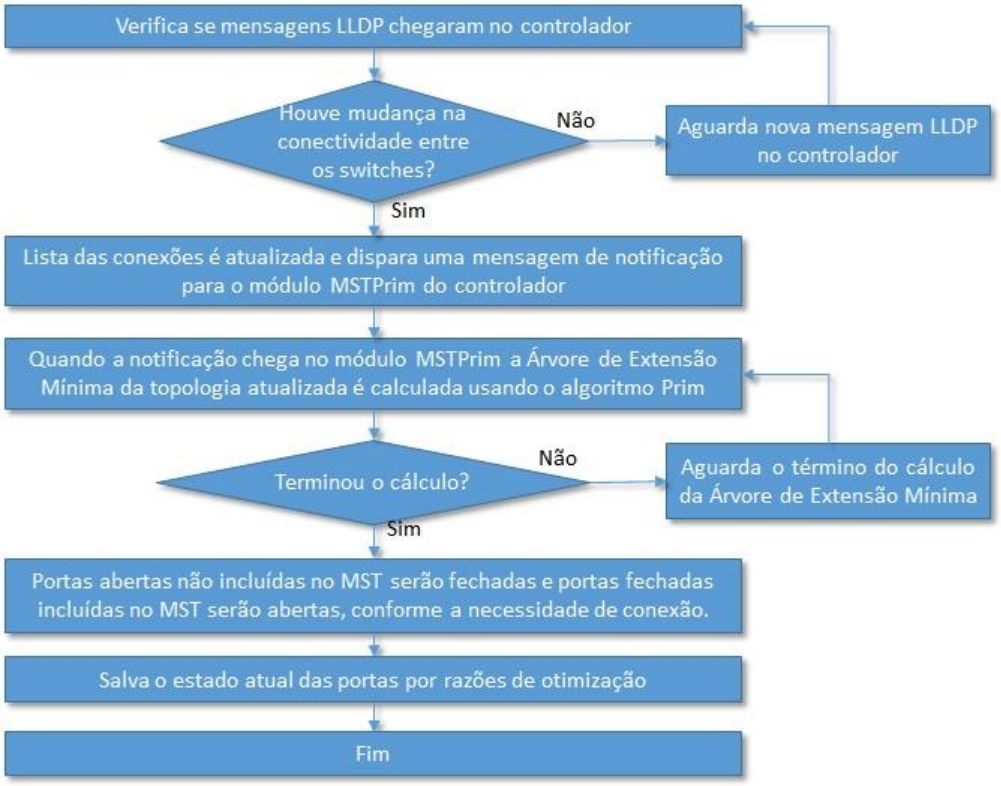
O módulo desenvolvido denominado MSTPrim foi inserido no controlador Floodlight e oferece uma API que pode ser usada pelas Aplicações REST (aplicações que trocam mensagens HTTP). Floodlight armazena o estado instantâneo da rede, utilizando o módulo *Topology*. A observação do Protocolo de Descoberta da Camada de Enlace (*Link Layer Discovery Protocol* - LLDP) é usado para capturar as mudanças na conectividade entre os *switches*. Pacotes LLDP são enviados do módulo *Topology* para os *switches* que retornam informações de conectividade. Quando a topologia tem mudança, a lista das conexões é atualizada e uma mensagem composta por *linkUpdate* (notificação) é enviada para os outros módulos do Floodlight, tanto em um modo síncrona e assíncrona (publicação e assinatura). A mensagem *linkUpdate* relata os endereços e as interfaces dos *switches* modificados.

Da mesma forma, o módulo *Device Manager* mantém a lista de como os dispositivos finais estão ligados aos *switches* OpenFlow. Cada vez que chega um pacote para o módulo, os cabeçalhos são analisados e os endereços da camada 2 são acoplados ao *switch* correspondente. O módulo do *switch* de aprendizagem (*Learning Switch*) fornece uma implementação básica da funcionalidade de comutação padrão. O papel fundamental do controlador é agregar todas as tabelas MAC em uma área de memória única, simplificando a operação de pesquisa que associa uma interface de dispositivo a um endereço MAC de *host* de destino.

Assim que uma mensagem LLDP chega ao controlador, é desencadeado o fluxograma mostrado na Figura 1. O módulo *Topology* captura as mensagens LLDP dos *switches* e notifica o módulo MSTPrim da mudança de estado na ligação ponto-a-ponto entre as duas interfaces do dispositivo. Em detalhe, o módulo *Topology* escuta o estado das portas e os pacotes de entrada. Um filtro é aplicado aos pacotes de entrada para detectar o tráfego LLDP. Quando uma alteração na conectividade entre os dispositivos é observada, o módulo *Topology* propaga uma notificação para os outros módulos do controlador.

ISSN: 2175-1897

Figura 1 - Fluxograma das ações desencadeadas por mudança de estado em uma ligação entre *switches*.



Fonte: Elaborado pelo autor.

O módulo MSTPrim ativa sua sequência por reagir as invocações das notificações e mensagens do estado das portas. Este módulo usa as informações no objeto *linkUpdate* (identificações de dispositivos e interfaces) para manter uma representação da topologia de rede atualizada, indicando quais ligações estão ativas.

Uma implementação eficiente do algoritmo Prim (SEDGEWICK; WAYNE, 2011) é utilizado para calcular o MST no gráfico atualizado. Em detalhe, o código utiliza filas de prioridade e estruturas para encontrar a união em reduzir o tempo de convergência para O(E logV), onde E é o número de ligações entre os *switches*.

Após o cálculo da Árvore de Extensão Mínima, um conjunto de mudanças é avaliado para minimizar o número de interações entre o controlador e os *switches*. O conjunto de mudanças mantém a lista de ligações que apenas mudaram na árvore e aquelas que vêm de invocações anteriores ao procedimento. Os itens no conjunto de mudanças são usados para enviar para os dispositivos um número mínimo de mensagens OpenFlow. Interfaces são desativadas de acordo com o conjunto de alterações, por meio do envio de mensagens para modificar o comportamento da porta física para desligado.

À medida que o controlador é acionado em cada mensagem LLDP, o módulo MSTPrim constrói uma Árvore de Extensão Mínima estável em O(E 2logV), usando mensagens O(E). Os pesos sobre os arcos do grafo da topologia podem ser ajustados de acordo com as necessidades da aplicação, por exemplo, informações

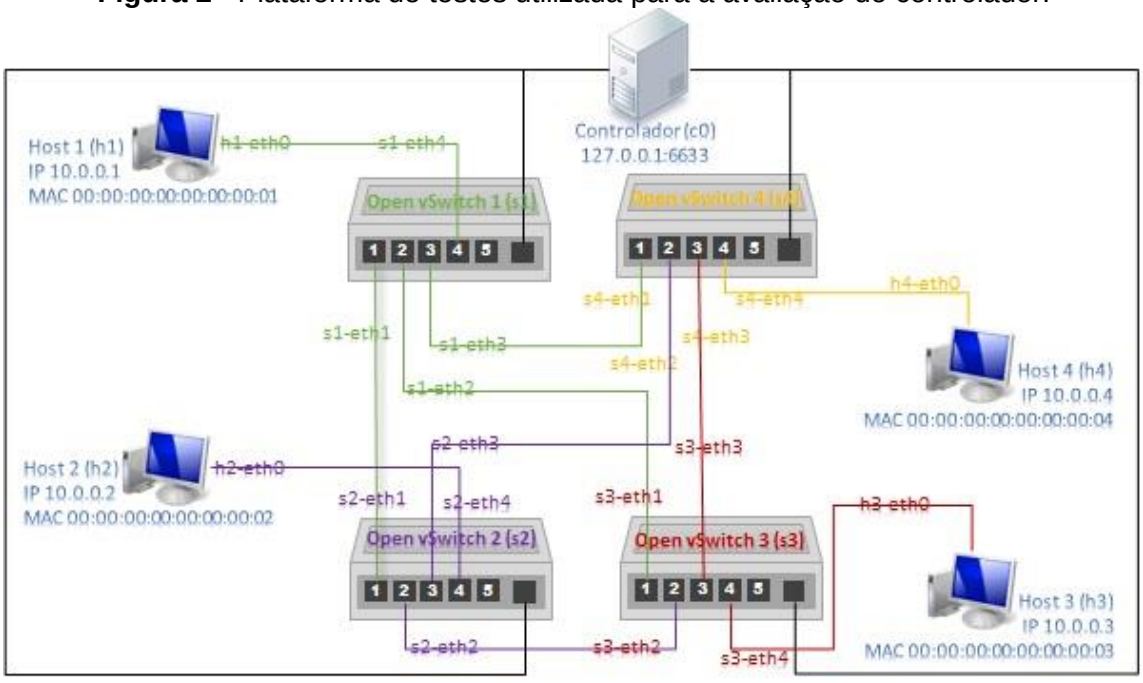
sobre a largura de banda disponível ou a latência média na ligação. Para simplificar, durante os testes foram consideradas as ligações que têm o menor peso unitário. Nesse caso, a fila de prioridade utilizada pelo algoritmo Prim considera o número MAC dos *switches* como critério de ordenação.

3.2 Plataforma de testes

Para avaliar a exatidão do controlador, foi usado um computador com Processador Intel® Core™ i5-3210M CPU @ 2.50 GHz 2.50 GHz, 6 GB de memória RAM e 300 GB de espaço livre no disco rígido. A infraestrutura virtual foi construída sobre uma plataforma de testes no *software* de virtualização VirtualBox. Este permite a execução de uma máquina virtual dentro de uma máquina física, é gratuito e está disponível para Windows, Mac e Linux. Para emulação da rede, foi empregado o MiniNet. Ele cria uma rede virtual OpenFlow (controlador, *switches*, *hosts* e ligações) em uma única máquina real ou virtual. A Figura 2 mostra a topologia emulada no MiniNet. Os componentes da rede são descritos a seguir:

- Os dispositivos de comutação foram criados usando quatro *switches* virtuais (s1, s2, s3 e s4) equipados com o software Open vSwitch (OPEN VSWITCH, 2015).
- Os hosts virtuais (cada um contendo um endereço de IP e MAC separado) foram configurados usando quatro máquinas virtuais (h1, h2, h3 e h4) rodando Linux. A conexão de cada host ao *switch* foi realizada com cabo ethernet virtual.
- A configuração da rede virtual é uma topologia totalmente em malha entre os *switches*.
- O controlador comunica com os dispositivos OpenFlow através de uma rede de gerenciamento separado (não mostrado na Figura 2).

Figura 2 - Plataforma de testes utilizada para a avaliação do controlador.



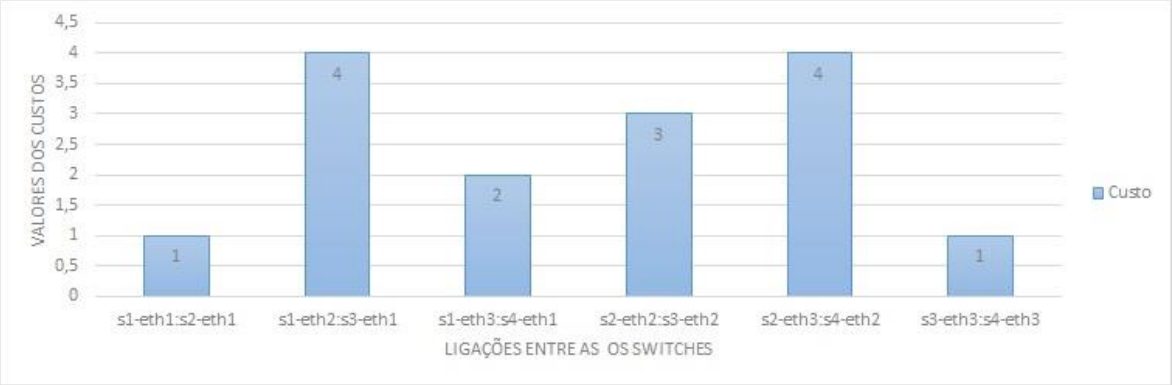
Fonte: Elaborado pelo autor.

Com este estudo, é possível evidenciar que o módulo MSTPrim identifica adequadamente a árvore de extensão que otimiza todas as conexões de *switch* a *switch*. Por meio da criação de um agente na qual detecta as condições da rede e desencadeia uma atualização nos custos das ligações, o módulo MSTPrim é capaz de adaptar-se em tempo real às configurações de mudanças na rede.

4. Resultados e Discussão

Foi desenvolvido um sistema de registro e realizaram-se testes para depurar o comportamento do controlador. O número de requisições ARP (*Address Resolution Protocol*) foi observado nos dispositivos para verificar a atividade do módulo MSTPrim. Além disso, a acessibilidade dos *hosts* é verificada para conferir a exatidão da árvore de extensão calculada. Para testar o software de implementação quatro fases têm sido executadas. A primeira apresenta os *switches* de origem/destino e suas respectivas portas ethernet utilizadas, além de mostrar os custos empregados entre todos os *switches* exibidos na Figura 3.

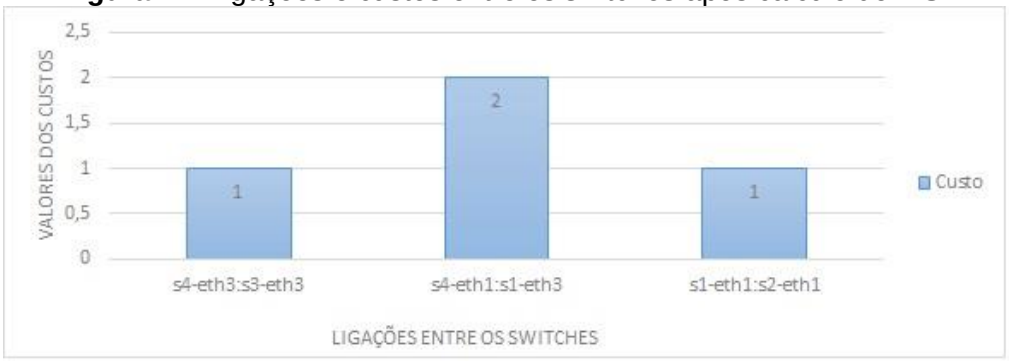
Figura 3 – Ligações e custos entre os *switches*



Fonte: Elaborado pelo autor.

Na segunda fase, executando o módulo MSTPrim, obtiveram-se as ligações que representam a topologia avaliada, mostrada na Figura 4. É possível verificar que somente as bordas com baixo custo foram mantidas abertas.

Figura 4 – Ligações e custos entre os *switches* após cálculo do MST



Fonte: Elaborado pelo autor.

O MST foi implantado nos *switches*, fechando as portas das ligações não usadas. Empregando o comando DPCTL[1], associado ao parâmetro SHOW[2], é possível identificar quais portas dos *switches* foram fechadas. No Quadro 1, por exemplo, é apresentado o estado das portas extraído do *switch* 4.

Quadro 1 – Estado das portas no *switch* 4

```
mininet> s4 dpctl show tcp:127.0.0.1:6637
features_reply (xid=0x4a7cacc5): ver:0x1, dpid:4 n_tables:255, n_buffers:256
features: capabilities:0xc7, actions:0xfff
1(s4-eth1): addr:a2:8f:c2:a8:6d:57, config: 0, state:0
    current:    10GB-FD COPPER
2(s4-eth2): addr:ba:d8:da:a1:26:5b, config: 0x1, state:0x1
    current:    10GB-FD COPPER
3(s4-eth3): addr:0e:1d:34:0a:18:b2, config: 0, state:0
    current:    10GB-FD COPPER
4(s4-eth4): addr:0e:1d:34:0a:18:b2, config: 0, state:0
    current:    10GB-FD COPPER
get_config_reply (xid=0xe67cf3b9): miss_send_len=0
```

Fonte: Gerado pelo MININET, 2015.

Observa-se no Quadro 1, que a porta eth2 do *switch* 4 tem o estado 0x1, o que significa que a porta foi desligada. A sobreposição da Árvore de Extensão Mínima desligou 6 portas, sendo, s1:eth2, s2:eth2, s2:eth3, s3:eth1, s3:eth2, s4:eth2, reduzindo o consumo de energia total das interfaces em 50%. Para a terceira fase, foram alterados os custos nas ligações, conforme a Figura 5.

Figura 5 – Novos custos nas ligações entre os *switches*

Fonte: Elaborada pelo autor

Enfim, a quarta fase apresenta o recálculo da Árvore de Extensão Mínima e obtiveram-se as ligações mostradas na Figura 6. Observa-se uma árvore diferente da anterior, devido à alteração dos custos.

Figura 6 – Novas ligações e novos custos entre os *switches* após o cálculo do MST

Fonte: Elaborado pelo autor.

[1] O comando DPCTL exibe a visibilidade e o controle da tabela de fluxo do switch.
[2] O parâmetro SHOW conecta ao switch e descarrega o estado de suas portas.

ISSN: 2175-1897

Do mesmo modo, usando o comando DPCTL e parâmetro SHOW é possível analisar quais portas dos *switches* foram fechadas. No Quadro 2, por exemplo, é apresentado novamente o estado das portas extraído do *switch* 4.

Quadro 2 – Estado das portas do *switch* 4

```
mininet> s4 dpctl show tcp:127.0.0.1:6637
features_reply (xid=0x461fa8c): ver:0x1, dpid:4 n_tables:255, n_buffers:256
features: capabilities:0xc7, actions:0xfff
1(s4-eth1): addr:a2:8f:c2:a8:6d:57, config: 0x1, state:0x1
    current:    10GB-FD COPPER
2(s4-eth2): addr:ba:d8:da:a1:26:5b, config: 0, state:0
    current:    10GB-FD COPPER
3(s4-eth3): addr:0e:1d:34:0a:18:b2, config: 0x1, state:0x1
    current:    10GB-FD COPPER
4(s4-eth4): addr:0e:1d:34:0a:18:b2, config: 0x1, state:0
    current:    10GB-FD COPPER
get config reply (xid=0x59b0cd47): miss send len=0
```

Fonte: Gerado pelo MININET, 2015.

A porta eth2, agora, está ativa, enquanto as portas eth1 e eth3 do *switch* 4 estão desligadas. A sobreposição da Árvore de Expansão Mínima desligou 6 portas, sendo, s1:eth2, s1:eth3, s3:eth1, s4:eth1, s4:eth3, reduzindo o consumo de energia total das interfaces do mesmo modo que o teste anterior em 50%.

As fases descritas mostram que a implementação proposta de uma variante do protocolo da árvore de extensão com recursos de economia de energia em um controlador OpenFlow se comporta corretamente. Ainda que os testes foram realizados ao longo de uma pequena rede virtual, feita de uma malha completa de *switches* de *software*, a validação é considerada positiva para a extensão de redes mais complexas, como WANs e centros de dados de árvores gordas.

5. Considerações finais

O paradigma de SDN, implementado em OpenFlow, permite projetar rapidamente e validar algoritmos inovadores para o controle da rede.

O trabalho explora a viabilidade de configurar uma malha de rede ethernet utilizando tecnologias SDN. A meta são centros de computação densos, típicos de serviços em nuvem, em que topologias ethernet livres de laços são importantes, bem como a eficiência energética. O protótipo proposto aborda o requisito básico de topologias livres de laços na camada 2, sendo pré-requisito para muitas configurações de rede experimentais e de produção. O módulo MSTPrim oferece a funcionalidade, evitando as desvantagens das soluções clássicas não OpenFlow, como STP, permitindo que os aplicativos especifiquem dinamicamente a métrica que o controlador usa para a construção da árvore de extensão.

A Árvore de Extensão Mínima permite a redução do consumo de energia, desligando as interfaces inativas. Validações iniciais mostram que o protótipo se comporta como esperado, resolvendo os problemas causados por laços entre dispositivos ethernet e reagindo corretamente em caso de mudanças na topologia.

Trabalhos futuros sobre Árvore de Extensão Mínima focará na solução para introduzir um cache com a lista de interfaces desativadas. Novos recursos serão

incluídos, entre eles, a definição de regras em interfaces para bloquear a transmissão de dados por um período limitado de tempo antes de ligá-las e a reconfiguração automática da topologia que mede de acordo com o tráfego atual. Os desempenhos do protótipo serão avaliados sobre topologias mais complexas para medir os tempos de reação e descobrir qualquer limitação do algoritmo.

Referências

BRANDON, Heller; SEETHARAMAN, Srini; MAHADEVAN, Priya; YIAKOUMIS, Yiannis; SHARMA, Puneet; BANERJEE, Sujata; MCKEOWN, Nick. ElasticTree: saving energy in data center networks, *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation (NSDI'10)*, San Jose, USA, p. 17-33, 28-30 Apr. 2010.

CHABAREK, Joseph; SOMMERS, Joel; BARFORD, Paul; ESTAN, Cristian; TSIANG, David; WRIGHT, Steve. Power Awareness in Network Design and Routing, *Proceedings of the 27th Conference on Computer Communications (IEEE INFOCOM)*, Phoenix, USA, p. 457-465, 13-18 Apr. 2008.

JARSCHEL, Michael; PRIES, Rastin. An OpenFlow-based energy-efficient data center approach, *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, Helsinki, Finland, p. 87-88, 13-17 Aug. 2012.

MCKEOWN, Nick; ANDERSON, Tom; BALAKRISHNAN, Hari; PARULKAR, Guru; PETERSON, Larry; REXFORD, Jennifer; SHENKER, Scott; TURNER, Jonathan. OpenFlow: Enabling innovation in campus networks, *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, Seattle, USA, p. 69-74, 17-22 Aug. 2008.

MININET. *An Instant Virtual Network on your Laptop*. 2015. Disponível em: <http://mininet.org/>. Acesso em: abr. 2015.

OPENFLOW. *Basic Spanning Tree*. 2011. Disponível em: <http://archive.openflow.org/wk/index.php/Basic_Spanning_Tree>. Acesso em: 23 maio 2015.

OPEN VSWITCH. Disponível em: <www.openvswitch.org>. Acesso em: 25 abr. 2015.

PROJECT FLOODLIGHT. *Open Source Software for Building Software-Defined Networks*. Disponível em: <http://www.projectfloodlight.org>. Acesso em: 11 jun. 2015.

RFC 4318. Network Working Group. *Definitions of Managed Objects for Bridges with Rapid Spanning Tree Protocol*. USA, 2005. Disponível em: <http://tools.ietf.org/html/rfc4318>. Acesso em: 10 jun. 2015.

SEDGEWICK, Robert; WAYNE, Kevin. *Algorithms*. 4th ed., Addison-Wesley Professional, 2011. 992 p.

SILVA, Manoel R. P.; ZANETI, Gislaine B.; ZAGO, Maria G. TI verde – princípios e práticas sustentáveis para aplicação em universidades. *Anais III Simpósio Brasileiro de Sistemas Elétricos*, Belém, Brasil, 2010.