Balanceamento de carga baseado em regras OpenFlow: analise de impacto durante a troca de regras

Clébio Dossa[1], Rodrigo da Rosa Righi[1], Vinicius Meyer[1]

[1]Programa de Pós-Graduação em Computação Aplicada – Universidade Do Vale do Rio dos Sinos (UNISINOS)

clebiodossa@gmail.com, rrrighi@unisinos.br, vinimeyer@hotmail.com

Abstract. *Actually, most services balance the load between distinct hosts forwarding connections with a load balance strategy in front. Usually, a dedicated server that may be a fault point and become expensive makes this load balance. The OpenFlow protocol definition allow us to use a new solution to address this issue. This work shows a load balance solutions between distinct hosts with the destination change of connections made by the network core.*

Resumo. *Atualmente muitos serviços distribuem a carga entre diversos hosts direcionando as conexões com alguma estratégia de balanceamento em frente. Este balanceamento geralmente é realizado por algum servidor dedicado que pode se tornar ponto único de falhas e ter um custo caro. A definição do protocolo OpenFlow permite uma solução alternativa e eficiente para este problema. Neste trabalho é apresentado uma solução para balanceamento de carga entre distintos hosts com a troca do destino do trafego realizada pelo núcleo da rede.*

1. Introdução

Desempenho e alta disponibilidade são itens críticos para *data centers atuais*. A infraestrutura de rede necessita maximizar a vasão, diminuir a latência e permitir uma elasticidade natural para atender as atuais demandas de aplicações. A comunicação entre a origem e o destino percorre múltiplos caminhos passando por distintos serviços e servidores. Cada um destes itens pode ser um ponto de falha, afetar a capacidade, a vasão ou simplesmente aumentar a latência. Desta forma, a simplificação e remoção destes pontos adicionais deve ser considerada na construção de um ambiente robusto e performático. Contudo, processos e equipamentos não podem ser rígidos e engessados.

Para atender a demanda necessária ocorre geralmente a aquisição de novos equipamentos, realização upgrade de hardware ou mesmo aumento da vazão. Neste cenário, mecanismos de balanceamento de carga realizam um importante trabalho distribuindo acessos concorrentes ou adequando aos limites da capacidade. Muitos métodos e fabricantes propõem soluções complexas, difíceis de serem implementadas [1] e testadas, rígidas e caras que necessitam de hardware potente para armazenar milhares de fluxos de comunicação. Além disso, o uso de equipamento dedicado para

balanceamento de carga adiciona latência ao processamento, podem ser um gargalo ou mesmo um ponto adicional de falha. Mesmo com tantas implementações disponíveis, a técnica mais eficaz de balanceamento de carga é um item incógnito para cada solução e depende de variáveis mutáveis para ser atingido em sua excelência.

Delegando a tarefa de balanceamento de carga ao núcleo da rede, onde equipamentos específicos já realizam a administração dos fluxos de conexão, se reduz a necessidade de adição de equipamentos dedicados à tarefa, também diminuindo a latência incrementada por estes equipamentos. Redes programáveis permitem a manutenção de regras de fluxos e fornecem estatísticas para tomada de decisão. A tecnologia do protocolo OpenFlow proporciona o acesso à um modelo que pode ser explorado para realização da tarefa de balanceamento de carga. Neste sentido, o OpenFlow permite a criação de técnicas distintas para o balanceamento de carga com grande flexibilidade, administrando cada fluxo da comunicação e proporcionando a habilidade de controlar cada host a qualquer momento de maneira centralizada[6].

Este trabalho realiza uma pesquisa no impacto do uso do protocolo OpenFlow para realização de balanceamento de carga. Esta é uma etapa de uma pesquisa mais ampla que tende à criação de uma solução que, com o uso do OpenFlow em conjunto do protocolo SNMP, proporciona a execução de balanceamento de carga adaptável, equilibrando a carga de uma forma eficaz. Com a visão de distintos pontos: fluxos de comunicação de todos os hosts do balanceamento através do OpenFlow e da capacidade de recursos de cada host através do SNMP, é possível ter uma visão estratégica da situação do ambiente. Existe uma grande diversidade de algoritmos de balanceamento de carga, entretanto, a análise de decisão com informações baseadas na rede não é comum de ser utilizada para este propósito [8]. Com isso, este trabalho e uma pesquisa necessária para o desenvolvimento de uma solução que, posteriormente, ira unir informações de trafego de rede com consumo de recursos dos servidores para tomada de decisão no balanceamento de carga.

A seguir uma breve introdução ao OpenFlow e a apresentação de trabalhos relacionados a esta pesquisa onde o balanceamento de carga com o uso do OpenFlow já foi explorado. Seguindo esta estrutura, a solução proposta está dividida na idealização do processo, a implementação, a metodologia de avaliação e os resultados desta etapa inicial, que tem como objetivo mapear o impacto da troca de trafego realizada pelo OpenFlow.

2. Tecnologia OpenFlow

A arquitetura OpenFlow pode ser dividida em alguns switches OpenFlow, um ou mais controladores, um canal dedicado e seguro de comunicação entre os switches e controladores e o protocolo OpenFlow para sinalizar e administrar os switches. O objetivo deste trabalho não é detalhar o funcionamento do OpenFlow, detalhes podem ser encontrados em [2].

Clássicos roteadores ou switches executam a tarefa de *data plane* e *control plane.* O plano de dados é onde as transmissões dos pacotes são realizadas e o plano de controle é onde as decisões de como este pacote será administrado são realizadas. O OpenFlow

separa estes dois planos e um protocolo define as trocas de mensagens entre eles criando um padrão de como o plano de dados solicita informações ao plano de controle e como o plano de controle informa as regras. O plano de controle define as decisões adicionando as regras de transmissão de pacotes no plano de dados, conforme figura 1.

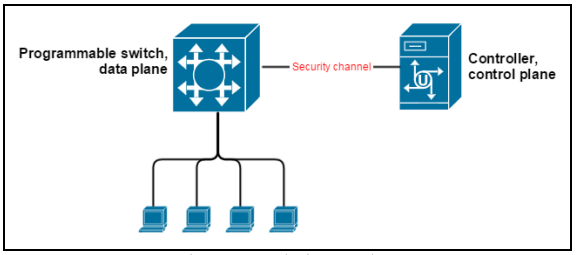

Figura 1. Topologia OpenFlow

3. Trabalhos relacionados

O uso do OpenFlow tem sido explorado em diversos aspectos. Muitos modelos e técnicas clássicas de administração e implementação de redes estão agora sendo implementadas com o uso do OpenFlow. O uso de balanceamento de carga diretamente no núcleo da rede não é uma atividade comum no modelo clássico de redes, entretanto o advento da tecnologia OpenFlow possibilita a absorção desta tarefa.

O trabalho desenvolvido em [5] demonstra a técnica de balanceamento de carga com OpenFlow onde o processo de decisão é definido por um algoritmo incorporado no plano de controle. Usando a estratégia de *Server-based load balancing* (SBLB) o controlador define o host com melhor capacidade para receber a requisição e o fluxo é alterado dinamicamente com troca de regras no plano de dados do switch OpenFlow. As avaliações neste trabalho demonstram uma grande flexibilidade e baixo custo de implementação e o uso desta técnica de balanceamento de carga aumenta o tempo de resposta de *Web Servers* e permite uma distribuição mais racional de recursos.

Em [6] uma avaliação é realizada usando diferentes técnicas de balanceamento de carga implementadas com o uso de OpenFlow: *Random choice*, *Time slice based choice* e *Weighted balancing*. A avaliação é baseada na melhor distribuição da carga em cada host usando ICMP *requests*. A técnica mais eficiente foi *Weighted balancing*, no entanto, a complexidade da implementação desta técnica é linear ao número de hosts que atendem o serviço pois, para cada fluxo recebido é realizada uma análise de performance em cada um dos hosts.

O uso do Openload, um software para gerar carga web e realizar medições de tempo de resposta e transações por segundo, foi tema da pesquisa em [7] onde a estratégia *Round robin* foi comparada com *Random choice* implementados em OpenFlow. Esta avaliação apontou *Round robin* com melhor desempenho, no entanto salientou a necessidade de tests em hardware real e não apensa testes em ambientes simulados.

4. Solução proposta

As técnicas clássicas para balanceamento de carga podem ser reproduzidas da mesma forma com o modelo proposto pelo protocolo OpenFlow inclusive, com melhor performance de acordo com a pesquisa demonstradas em [4] que indicam um melhor desempenho no modelo OpenFlow de switch comparado com modelos convencionais, provando ser uma melhor alternativa para software *Ethernet Switching* ou mesmo IP *Routing*.

Entretanto, a melhor solução para o problema, como implementar e o que controlar para atender o balanceamento são itens distintos. Os fluxos de conexões previamente estabelecidas não podem ser direcionados para distintos hosts, da mesma forma, a proporção de processamento por parte dos hosts precisa ser equilibrada sem sobrecarregar determinado host enquanto outros estão com boa capacidade de processamento.

O protocolo OpenFlow viabiliza a adição de regras no plano de dados que é alimentado com informações do destino que os fluxos entrantes devem tomar. Estas regras podem ser alteradas de acordo com a necessidade de processamento, vasão, latência ou mesmo o histórico dos fluxos. Basicamente, regras predefinidas de fluxos podem ser utilizadas para balanceamento de carga. Com o uso das distintas técnicas de balanceamento, este processo de alteração de fluxo pode ser dinâmico e automatizado através de algoritmos específicos.

A solução apresentada nesta pesquisa atende balanceamento de conexões para aplicações que não necessitam o uso de contexto (stateless). Tem-se em consideração que todo o host de destino tem a mesma capacidade de atender a solicitação entrante pois realiza o mesmo serviço. O processo abaixo visa analisar o impacto das trocas de regras em intervalos de tempo com o intuito de clarificar o impacto desta ação. Esta analise precisa ser considerada para desenvolvimento de um processo eficiente de

4.1 O Processo Idealizado

O balanceamento no contexto do OpenFlow é realizado alterando o destino do pacote para direcionar a conexão entrante para um host com o maior poder de processamento no momento da solicitação. Utilizando as técnicas de *layer 3* do OpenFlow, o pacote destinado ao serviço tem seu endereço IP de destino reescrito para o endereço de IP do host com maior capacidade de processamento e com menor quantidade de pacotes já recebidos. No momento em que o host retorna a solicitação, o IP de origem é alterado para o IP que foi inicialmente solicitado para não danificar a comunicação. Sendo duas alterações realizadas no pacote recebido: no momento em que ele entra e no momento em que sai.

A análise de desempenho realizada por [3] comprova a eficiência e vasão das operações em *layer 3* do OpenFlow superando entre 30% e 40% a performance de *switching layer 2* em implementações baseadas em Linux. Nestes testes também se observa uma ótima performance do OpenFlow para lidar com múltiplos fluxos e capacidade de gerencia-los combinado com a excelente vasão e baixa latência, itens que tem encorajado muito o uso desta tecnologia.

4.2 Desenvolvimento e Implementação

O experimento do funcionamento desta metodologia foi realizado com o uso de Openvswitch, que é um comutador virtual que suporta diversas camadas e um impressionante conjunto de recursos. Na figura 2 um exemplo de regra genérica que este comutador pode receber para alterar o destino de um pacote.

```
ovs-ofct add-flow <comutador> \
ip, tcp_flags=+syn, nw_dst=<IP origem>, \
actions=mod_nw_dst=<IP destino>, mod_dl_dst:<MAC destino>,output:<porta destino>
```

Figura 2. Regra para alteração do destino

O controle da *flag* de entrada é importante para não modificar o destino de conexões já estabelecidas. Opções adicionais podem ser utilizadas para controle de portas, protocolos e demais itens definidos pelo protocolo OpenFlow.

Da mesma forma que o destino foi alterado, uma regra de retorno se faz necessário para que o cliente que solicitou a conexão receba a informação de forma correta. A figura 3 contempla uma regra genérica para este objetivo.

```
ovs-ofct add-flow <comutador> \
ip, nw_src=<IP destino>,actions=mod_nw_src=<IP origem>,output:<porta origem>
```

Figura 3. Regra para alterar o pacote de resposta

A regra para alteração do pacote de resposta pode ser sempre estática, existindo uma para cada host onde o serviço é atendido.

Neste exemplo a conexão é sempre originada para o mesmo endereço IP. O cliente utiliza um ponto único de acesso e o comutador decide para quem será enviado a conexão.

4.3 Metodologia de Avaliação e Análise dos Resultados

Para realizar esta análise e provar o funcionamento desta técnica, uma rede virtual foi inicializada com o uso do emulador Mininet. Neste ambiente 5 switches e 16 hosts virtuais foram inicializados. A topologia do ambiente é apresentada na Figura 4.

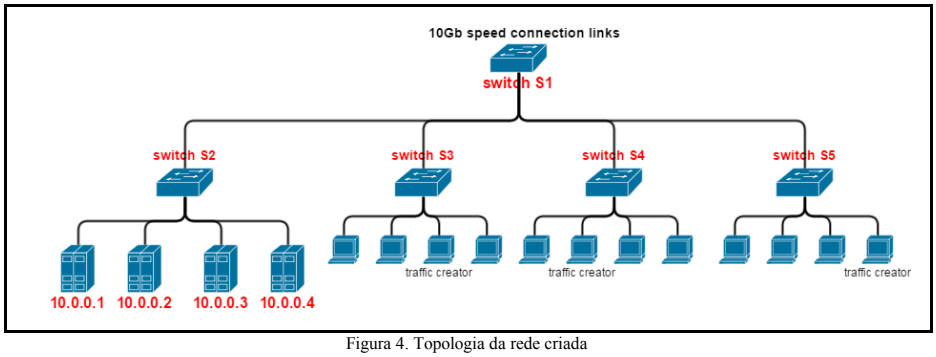

Figura 4. Topologia da rede criada

O switch S1 é responsável pelas conexões entre os outros 4. Os hosts existentes no switch S2 irão receber as conexões e o trafego será alternado entre todos os hosts em intervalos variados de 100 segundos. O script exemplificado na figura 5 é responsável pela alteração de tráfego entre os equipamentos. A figura 6 demonstra o fluxograma do script desenvolvido para esta análise.

```
#clean all rules
ovs-ofctl del-flows s2
#hosts that belongs to the load balance group
ovs-ofctl --strict add-flow s2 ip,priority=65535,nw_src=10.0.0.2,actions=mod_nw_src=10.0.0.1,output:5
ovs-ofctl --strict add-flow s2 ip,priority=65535,nw_src=10.0.0.3,actions=mod_nw_src=10.0.0.1,output:5
ovs-ofctl --strict add-flow s2 ip,priority=65535,nw_src=10.0.0.4,actions=mod_nw_src=10.0.0.1,output:5

while true
do
    #random choice to the destination host
    HOST=$((RANDOM%4+1))
    echo "Next destination $HOST \n";
    #delete old rule
    ovs-ofctl --strict del-flows s2 \
        ip,priority=65535,tcp_flags=+syn,nw_dst=10.0.0.1
    #add new rule
    ovs-ofctl --strict add-flow s2 \
        ip,priority=65535,tcp_flags=+syn,nw_dst=10.0.0.1, \
        actions=mod_nw_dst=10.0.0.$HOST,mod_dl_dst:00:00:00:00:00:0$HOST,output:$HOST

    #wait a random time between 0 and 10 seconds
    WAIT=$((RANDOM%100))
    echo "Waiting $WAIT \n\n"
    sleep $WAIT
done
```

Figura 5. Script para automação da troca de trafego

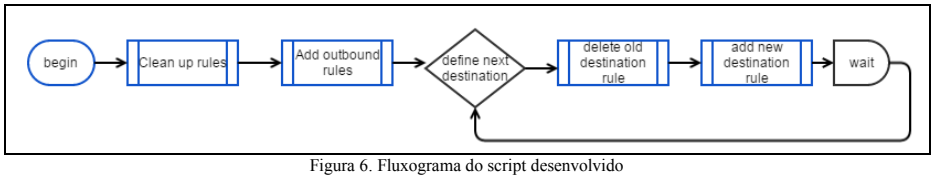

Figura 6. Fluxograma do script desenvolvido

O objetivo deste teste não é equilibrar a carga dos hosts de destino no switch S2, porém, validar se a troca de tráfego, com a técnica explicada no item anterior, pode causar alguma perda de pacote ou mesmo lentidão no processo durante o chaveamento entre

um e outro equipamento. Para esta validação, três equipamentos em pontos distintos desta topologia enviam requisições *ICMP request* para o equipamento de IP 10.0.0.1. Estas conexões foram balanceadas entre os equipamentos 10.0.0.1, 10.0.0.2, 10.0.0.3 e 10.0.0.4 de forma aleatória.

Foram realizados 133 trocas de destino e um total de 20400 pacotes enviados para estes 4 hosts. Nenhuma perda de pacote foi observada, apenas alguns desvios de tempo de resposta acontecem em cada atualização do comutador.

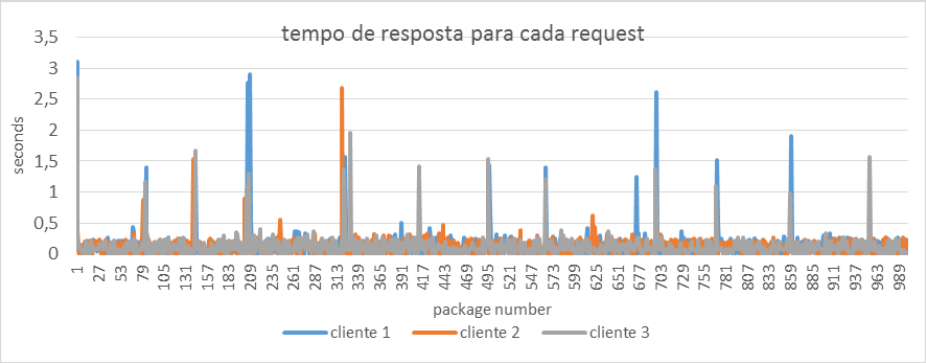

Figura 7. Incremento no tempo de resposta com a alteração de destino

Conforme observado na figura 7, há um incremento de tempo de resposta quando a troca de regras de fluxo acontece. Este tempo é devido à remoção da regra para que o fluxo seja direcionado há um host com melhor performance. Desta forma, a ação de troca de regras pode afetar as filas internas a serem processadas pelo equipamento.

5. Conclusão

Com os atrativos do auto desempenho da tecnologia OpenFlow demonstrados por trabalhos relacionados e a capacidade de realizar balanceamento de carga de forma nativa no núcleo da rede, removendo a necessidade de inclusão de servidores ou serviços adicionais para a realização desta tarefa, o OpenFlow se mostra uma excelente proposta para execução de balanceamento de carga simples onde a utilização de controle de contexto não se faz necessária. O processo de implementação desta tarefa é simples, com a inclusão e alteração de regras para modificação dos pacotes. Esta tarefa pode ser automatizada e atualizada de forma dinâmica de acordo com a necessidade do serviço prestado. Este processo permite elasticidade no quesito de adição, remoção e alteração de hosts para atender um determinado serviço.

Esta pesquisa demonstra que a atualização do plano de fluxos do OpenFlow para troca de regras não gera perda de pacotes ou cria um grande impacto no tempo de resposta, porém, há uma espera maior durante a troca de regras que deve ser considerada pois, os pacotes analisados tiveram uma pequena latência acrescida durante o tempo em que estiveram na fila do comutador aguardando a definição da regra de fluxo deste pacote. Desta forma, esta tarefa precisa ser veloz para minimizar o tempo em que os fluxos estão enfileirados aguardando a regra de decisão.

O uso do protocolo de monitoramento SNMP auxilia o processo de análise de carga de cada host. O modelo do protocolo OpenFlow proporciona uma visão clara e bem definida do processamento de fluxos onde estatísticas são controladas e acessadas de forma centralizada. Neste sentido, o host com melhor qualidade para processamento deve ser o responsável para processar a requisição. A melhor qualidade pode ser determinada considerando ambos itens monitorados onde a performance do hardware é um indicador de carga e o número de fluxos direcionados para cada host é um item *delta* de qualidade. Para não sobrecarregar o host com requisições de informação SNMP, este *delta* item pode ser utilizado para reduzir a capacidade de processamento do host em conjunto com a última informação de consumo de hardware recebida. A cada intervalo de análise um novo host é determinado e as regras de fluxo alteradas se necessário.

Com o uso desta medida de decisão de qualidade, mesmo hosts com diferentes capacidades de hardware podem ter o balanceamento equilibrado de uma forma eficaz. Como futuro desenvolvimento desta pesquisa um algoritmo que faz o uso de ambas as tecnologias será desenvolvido e a eficácia deste algoritmo analisada.

Referências

[1] Wang Peng, Lan Julong, Chen Shuqiao (2014) “OpenFlow based Flow Slice Load Balancing”, p. 72 -82, Communications, China (Volume:11 , Issue: 12)

[2] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S. e Turnet, J. (2008) “OpenFlow: enabling innovation in campus network”, p. 69-74, ACM SIGCOMM Computer Comnuication Review

[3] Bianco, A., Birke, R., Giraudo, L. e Palacin, M. (2010) “OpenFlow Switching: Data Plane Performance”, p. 1-5, Communications (ICC), 2010 IEEE International Conference

[4] C.C. Okezie, Okafor K.C, Udeze C.C (2013) “Open Flow Virtualization: a Declarative Infrastructure Optimization Scheme for High Performance Computing”, p. 232-244, Academic Research International, Vol.4

[5] Shang, Z., Chen, W., Ma, Q., Wu B (2013) "Design and implementation of server cluster dynamic load balancing based on OpenFlow", p. 691-697, Awareness Science and Technology and Ubi-Media Computing (iCAST-UMEDIA), 2013 International Joint Conference

[6] Marcon, D., Bays, L. (2012) “Flow Based Load Balancing: Optimizing Web Servers Resource Utilization”, p. 76-83, Journal of Applied Computing Research, vol. 1, n. 2

[7] Kaur, S., Singh, J., Kumar, K., Ghumman, N.S. (2015) “Round-robin based load balancing in Software Defined Networking”, p. 2136-2139, Computing for Sustainable Global Development (INDIACom)

[8] Belyaev, M., Gaivoronski, S. (2014) “Towards load balancing in SDN-networks during DDoS-attacks”. P. 1-6, Science and Technology Conference (Modern Networking Technologies) (MoNeTeC)