Estudo comparativo entre depuradores para SDN

Thales Nicolai Tavares[1], Anderson Monteiro da Rocha[1], Leonardo da Cruz Marcuzzo[1], Nilton Camargo Batista da Silva[1], Vinicius Fülber Garcia[1], Carlos Raniery Paula dos Santos[1]

[1]Departamento de Computação Aplicada PGCC – Universidade Federal de Santa Maria (UFSM)
Av. Roraima nº 1000 – 97.105-900 – Santa Maria – RS – Brasil

{tntavares,amonteiro,lmarcuzzo,nbatista,vfulber,csantos}@inf.ufsm.br

Abstract. *An implementation of Software Defined Networks allows the benefit of network technology professionals as an alternative to centralized network management. In addition, SDN emits a central programmable controller, enabling new and developed network applications. Instead of a control schedule, it also presents a possibility to introduce errors in the applications, as well as an unwanted behavior. To mitigate problem, there are tools called debuggers, as banks perform testicles for errors. This work proposes to perform a comparative study among SDNs debuggers, where they are presented as features, advantages and disadvantages.*

Resumo. *A implementação de Software Defined Networks permite benefícios aos profissionais da área de tecnologia de redes, como a alternativa de gerenciamento centralizado da rede. Além disso, SDN apresentam um controlador central programável, possibilitando que novas aplicações de rede sejam desenvolvidas. Ao proporcionar a programabilidade do controlador, também apresenta a possibilidade de introduzem erros nas aplicações, assim ocasionando um comportamento não desejado. Para mitigar problema, existem ferramentas chamadas depuradores, as quais realizam testes em busca de erros. Este trabalho tem o proposito de realizar um estudo comparativo entre depuradores SDNs, onde serão apresentadas as características, vantagens e desvantagens.*

1. Introdução

Com o crescimento da internet e sua utilização de forma comercial, a expansão das redes se deu de forma inevitável, porém a tecnologia de arquitetura baseada no modelo TCP/IP não acompanhou o crescimento no que se refere a sua inovação, além disso, o fato dos dispositivos serem controlados por softwares proprietários, engessou qualquer forma de desenvolvimento e personalização para a configuração das redes.

Consequentemente, surge as Redes Definidas por Software (SDN), que apresenta um novo conceito para a estrutura das redes que traz mais flexibilidade, rapidez e favorece novos serviços [Nadeau and Gray 2013]. Sendo assim, as Redes Definidas por Software representam uma nova maneira de olhar a forma como as redes são controladas, configuradas e operadas.

As SDN permitem maior flexibilidade, pois são controladas por softwares [Riggio et al. 2013]. Ao invés vez dos consoles de gerenciamento de redes e comandos

que exigem um grande esforço operacional, tornando complexa a administração em larga escala. Nesse novo conceito, pode-se começar a depurar redes assim como um software. Sendo assim, temos com objetiva apresentar um estudo comparativo entre depuradores para redes definidas por software e expor suas características.

O presente trabalho esta organizado de forma que a Seção 2 apresenta uma revisão sobre os temas envolvidos no trabalho, a Seção 3 apresenta os depuradores utilizados. A Seção 4 apresenta o ambiente os cenários envolvidos nos testes, na Seção 5 são apresentados os resultados e as considerações finais sobre o trabalho é apresentado na Seção 6.

2. Redes Definidas por Software

A rede definida pelo software é a maneira de abordar a rede de computadores através de abstrações de *software* em lugar de hardware especializado [Nadeau and Gray 2013]. Ao abstrair algumas das funcionalidades de baixo nível da rede em aplicações, permite que os administradores de rede gerenciar com mais facilmente as redes dinâmicas.

Nesse contexto, as rede definida por software separa a parte da infraestrutura de rede que determina onde a informação está sendo enviada, plano de controle, da parte onde os dados realmente se movem, plano de dados [Baldini et al. 2012]. Assim e permite a parte de tomada de decisão acontecer na aplicação.

2.1. OpenFlow

Afim de permitir realizar experimentos de novas propostas para redes de computadores em escalas menores, foi proposto o Comutador *OpenFlow* [Heller 2009], o qual possui a função de possibilitar seja executado experimentos em uma rede real. Nesse sentido a plataforma *OpenFlow* tem como objetivo criar um ambiente de rede de teste programável, unindo as qualidades da virtualização de redes com o conceito de Redes Definidas por Software [Guedes et al. 2012].

O protocolo *OpenFlow* apresenta grande vantagem em sua utilização, pois é possível realizar testar em infraestruturadas sem que interfira no tráfego real da rede de produção [McKeown et al. 2008], sem a necessidade que os fabricantes de comutadores exponham os projetos de software e hardware dos seus equipamentos. *OpenFlow* emprega o conceito de redes definidas por software, cujo o substrato físico é composto pela parte que cuida do tráfego de produção da rede e outra parte que cuida do tráfego experimental das novas propostas de rede. Assim, a plataforma *OpenFlow* procura oferecer uma opção controlável e programável, porém não pode-se descartar a possibilidade conter trechos de códigos com problemas [Canini et al. 2012].

3. Depuradores de rede

As redes de computadores possuem um elevado nível de dificuldade de depurar [Handigol et al. 2012]. Atualmente os gerentes de redes dispões de um conjunto de ferramentas como *tcpdump* para monitoramento dos dispositivos finais e *netflow* em *switches* e roteadores [Reitblatt et al. 2011]. Realizar a tarefa de depurar um rede é considerada árdua, pois as ferramentas buscam reconstruir o estado da rede de maneira ad-hoc, no entanto protocolos de camada 2 e 3 possuem mudanças de estados contínuos [Handigol et al. 2012]. Em Redes Definidas por Software ferramentas que possibilitam

realizar depuração de uma rede, da mesma maneira que realiza-se depuração em software [Handigol et al. 2012].

A ferramenta *OFRewind* é um depurador que possui o objetivo de auxiliar os gerentes de rede a localizar erros presentes em uma redes com *OpenFlow*, além de possibilita realizar a tarefa de análise da rede [Heller et al. 2013]. Ja a ferramenta NICE faz-se eficaz ao depurar programas OpenFlow, pois representa a verificação de um modelo e uma execução simbólica para encontrar erros em programas *OpenFlow* [Canini et al. 2012].

4. Caso de teste

- **OFRewind:** O cenário utilizado para realizar o teste para avaliação do OFRewind, será o *Anomalous Forwarding*, sendo baseado na documentação desenvolvidos pelos desenvolvedores. Para a elaboração dos testes, foram utilizados alguns *switches*, cada um com dois computadores conectados, conforme ilustrado na Figura 1 apresentada na Sessão 5.
- **NICE:** O cenário utilizado para realizar o teste para avaliação do NICE, será o *Pyswitch*, que é uma aplicação *OpenFlow* que tem como objetivo implementar o registro de endereços MAC Address, junto com o processo de *flooding* para destinos desconhecidos, muito comum em *Switches Ethernet*.

5. Resultados e Discussões

Com o proposito de atender os objetivos propostos neste trabalho, explorando os depuradores presentes para SDN, assim como suas características, faz-se necessário, para o melhor compreensão, estipular parâmetros de comparação entre essas ferramentas. Os parâmetros definidos são: instalação, usabilidade, documentação disponível, número de casos de teste disponíveis e eficiência.

5.1. OFRewind

Conforme os critérios definidos, os resultados do estudo e teste desta ferramenta são:

- **Instalação:** Não se aplica.
- **Usabilidade:** Embora seja uma ferramenta *Open source*, existem limitações. Por não conseguir retorno com o desenvolvedor, não foi possível obter o código fonte para compilar a ferramenta.
- **Quantidade de caso de teste:** Em sua documentação apresenta consigo três diferentes casos de teste, o qual possibilita o conhecimento das de suas funcionalidades.
- **Documentação disponível:** A ferramenta dispõe de uma documentação satisfatória, que com obtenção do código fonte, proporciona uma melhor elaboração de casos de teste para serem realizados nesta ferramenta.
- **Violações Encontradas:** Conforme documentação disponível, torna-se possível obter informações de casos de teste já realizado e documentados.

Para averiguar o desempenho de dispositivos com base de um novo fornecedor (fornecedor C), foi registrado o tráfego de um fluxo específico. Para afim de teste foi enviado um *ping* com 10 segundos de atraso entre um par de *hosts* conectados ao *switch* fornecedor

B. Logo foi usado o recurso de mapeamento de dispositivos e portas do *Ofreplay* para reproduzir o tráfego do host c7 para o c8 pela porta 8 e 42 pertencente ao *switch* fornecedor C na Figura 1.

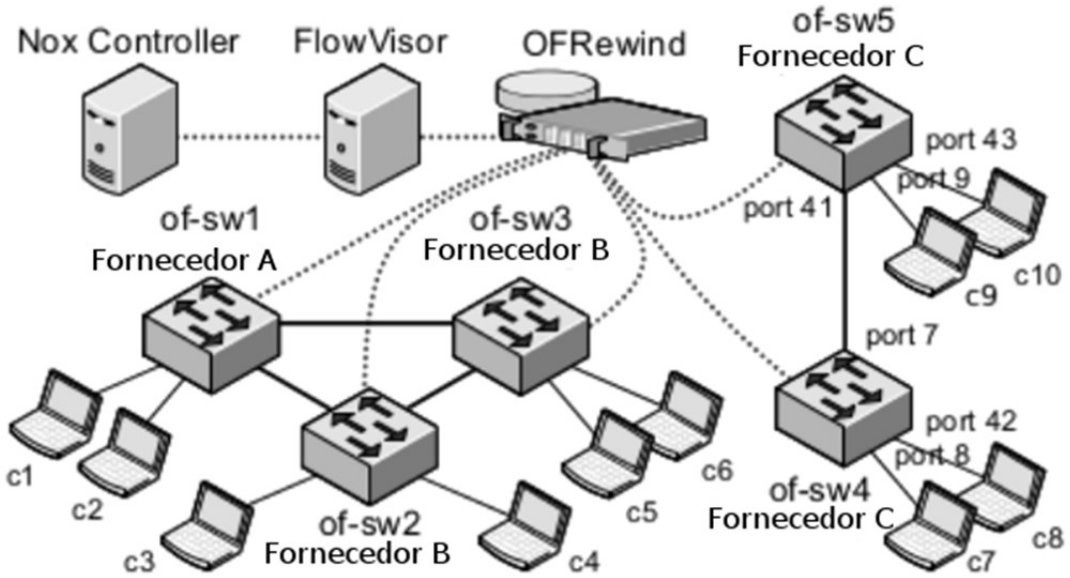

Figure 1. Topologia para testes do OFRewind

Logo depois da repetição, podemos percener uma limitação no switch fornecedor C. O fluxo do ping no perioldo de resolução do ARP, os mesmos transmitidos do host c7 são recebidos pelo host c10, mas não pelos host c8 c9. Na tabela 1 é apresentada a entrada da de fluxos criado no switch of-sw4. Conclui-se que a ação de Flooding não está devidamente aplicada pelo switch fornecedor C.

CONTADOR	AÇÃO
duration=181s	in_port=8
n_packets=0	dl_type=arp
n_bytes=3968	dl_src=00:a1:f9:32:b6:44
idle_timeout=60	dl_dst=ff:ff:ff:ff:ff:ff
hard_timeout=0	actions=FLOOD

Table 1. Tabela de entrada de fluxos *switch* of-sw4

5.2. NICE

Conforme os critérios definidos, os resultados do estudo e teste desta ferramenta são:

- **Instalação:** O processo de instalação é encontrado na página[1] dos desenvolvedores.Se faz essencial alguns *plugins* adicionais, como o Python com versão 2.6 ou superior.

[1]https://code.google.com/p/nice-of/wiki/DocRequirements

- **Usabilidade:** Embora o NICE não oferecer uma interface gráfica, a sua usabilidade é satisfatória. Por meio de apenas poucos comandos é possível realizar o teste da aplicação desejada. Como exemplo, para realizar o caso de teste *pyswitch*, é somente executar o NICE através da chamada de comando: "*./nice.py config/pyswitch.conf* ".
- **Quantidade de caso de teste:** Esta ferramenta disponibiliza somente três casos de testes, mas não impossibilita que seja adicionado novos testes pelo usuário.
- **Documentação disponível:** É disponibilizado em sua página[2] materiais sobre a ferramenta, o qual simplifica a pesquisas e o desenvolvimento, além de um manual completo da utilização deste depurador. Também é disponibilizados três artigos[3][4][5], em fóruns internacionais, sobre esta ferramenta.
- **Violações Encontradas:** Com a realização do teste padrão do NICE (pyswitch) e o resultado aprensentado na Figura 2, faz-se capaz de confirma a ocorrencia de tres falhas relatadas na documentação do NICE.

```
--- Results ---
Total states: 3102
Unique states: 1520
Revisited states: 1582
Maximum path length: 34
Invariant violations: 35
NoLoop              : 0 violations
internal check      : 0 violations
NoDrop              : 2 violations (first found after 13.17s, 1114 transitions)
StrictDirectRoute   : 28 violations (first found after 13.55s, 1155 transitions)
NoForgottenPackets  : 5 violations (first found after 13.17s, 1114 transitions)
ReturnContinueStop  : 0 violations
Symbolic engine quitting...
```

Figure 2. Resultados execução pyswitch.conf

5.3. Comparações

As comparações foram realizadas com base na utilização das ferramentas e com base na documentação disponibilizada Afim de que se tenha uma melhor compreensão, é apresentado em tabelas as comparações. Sendo a Tabela 2 apresenta os controladores suportados por cada depurador. Já as características analisadas dos depuradores obtidas através das pesquisas aqui realizadas, é apresentada na Tabela 3.

DEPURADOR	POX	NOX	TESTE DE SWITCH
OFRewind	não	sim	sim
NICE	sim	sim	não

Table 2. Compatibilidade depuradores

[2]https://code.google.com/archive/p/nice-of/
[3]http://infoscience.epfl.ch/record/170618
[4]http://infoscience.epfl.ch/record/169211
[5]http://infoscience.epfl.ch/record/167777

	EFICIÊNCIA	INSTALAÇÃO	USABILIDADE	DOCUMENTAÇÃO
OFRewind	bom	ruim	não se aplica	ótima
NICE	ótimo	bom	ótimo	ótimo

Table 3. Análise dos depuradores

6. Considerações Finais

Com base da pesquisa efetuada, foi possível constatar que Redes Definidas por Software apresenta uma ampla aceitação em meios acadêmicos e corporativos. O desenvolvimento do protocolo *OpenFlow* teve imensa relevância para a expansão das Redes de computadores. A proposta ofertada pelo protocolo *OpenFlow* concede diversos avanços em pesquisas abrangendo redes de computadores,a capacidade de segmentar o plano de dados e o plano de controle, é uma grande vantagem do uso de Redes Definidas por Software. A partir deste mecanismo se faz capaz de realizar teste de ambientes e configurações de redes sem a necessidade de que a rede fique fora de operação ou apresente irregularidade em seu funcionamento.

References

Baldini, G., Sturman, T., Biswas, A. R., Leschhorn, R., Godor, G., and Street, M. (2012). Security aspects in software defined radio and cognitive radio networks: A survey and a way ahead. *IEEE Communications Surveys & Tutorials*, 14(2):355–379.

Canini, M., Venzano, D., Peresini, P., Kostic, D., and Rexford, J. (2012). A nice way to test openflow applications. In *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, number EPFL-CONF-170618.

Guedes, D., Vieira, L., Vieira, M., Rodrigues, H., and Nunes, R. V. (2012). Redes definidas por software: uma abordagem sistêmica para o desenvolvimento de pesquisas em redes de computadores. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC 2012*, 30(4):160–210.

Handigol, N., Heller, B., Jeyakumar, V., Maziéres, D., and McKeown, N. (2012). Where is the debugger for my software-defined network? In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 55–60. ACM.

Heller, B. (2009). Openflow switch specification.

Heller, B., Scott, C., McKeown, N., Shenker, S., Wundsam, A., Zeng, H., Whitlock, S., Jeyakumar, V., Handigol, N., McCauley, J., et al. (2013). Leveraging sdn layering to systematically troubleshoot networks. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 37–42. 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. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74.

Nadeau, T. D. and Gray, K. (2013). *SDN: Software Defined Networks: An Authoritative Review of Network Programmability Technologies*. " O'Reilly Media, Inc.".

Reitblatt, M., Foster, N., Rexford, J., and Walker, D. (2011). Consistent updates for software-defined networks: Change you can believe in! In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, page 7. ACM.

Riggio, R., Rasheed, T., and Granelli, F. (2013). Empower: A testbed for network function virtualization research and experimentation. In *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*, pages 1–5. IEEE.