

Software Switch 1.3

An experimenter friendly OpenFlow switch

Eder Leão Fernandes¹

Orientador: Christian Rodolfo Esteve Rothenberg¹



UNICAMP

¹Faculdade de Engenharia Elétrica e de Computação
Universidade de Campinas

16 de Abril de 2015

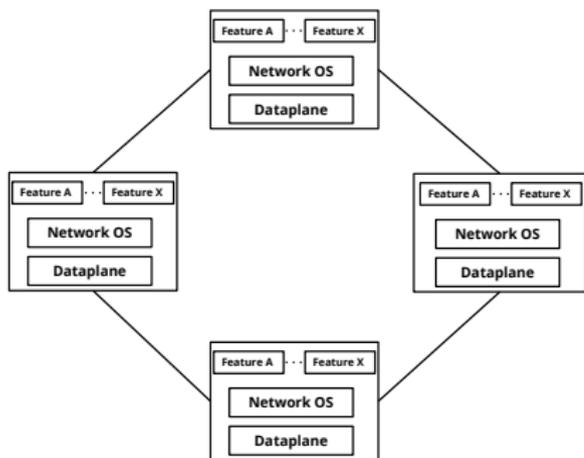
- 1** Introdução
- 2** Revisão Bibliográfica
- 3** Arquitetura
- 4** Desenvolvimento
- 5** Avaliação
- 6** Conclusão

- 1** Introdução
- 2 Revisão Bibliográfica
- 3 Arquitetura
- 4 Desenvolvimento
- 5 Avaliação
- 6 Conclusão

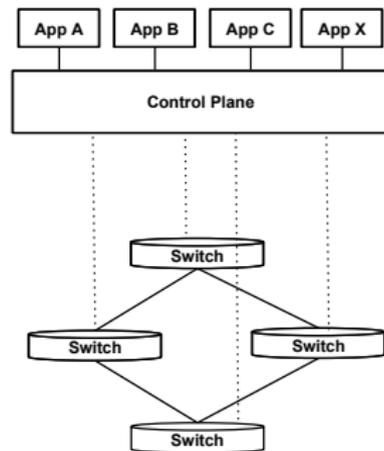
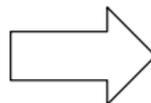
- Arquitetura de rede em que o **plano de controle** é separada do **plano de encaminhamento**
 - Visão global da rede
 - Programabilidade
 - Flexibilidade e automação do controle
- Comunicação entre o plano de controle e o plano de encaminhamento é realizada através de algum tipo de **protocolo** ou **API**.
 - O protocolo ***OpenFlow*** é um desses protocolos que habilitam SDN.

Introdução

Redes tradicionais vs SDN



Traditional Network



SDN Network

- O protocolo *OpenFlow* evolui de forma rápida nas mãos da *Open Network Foundation* (ONF)
- Controladores e *switches OpenFlow* por sua vez não conseguem acompanhar a velocidade das mudanças.
- Ferramentas de experimentação e validação são necessárias para manter o passo da evolução e permitir o desenvolvimento de aplicações no estado da arte
- O desenvolvimento de ferramentas atualizadas com as versões mais recentes é importante para antecipar o momento em que controladores e *switches* suportarem as especificações mais novas do *OpenFlow*

O objetivo do projeto é a implementação de um *switch* experimental *OpenFlow*, em *software* e com suporte à versão 1.3 do protocolo.

A partir do objetivo, os seguintes requisitos foram definidos:

- 1** Plenitude de funcionalidades do *OpenFlow* implementadas.
- 2** Código simples e de fácil extensão.
- 3** Integração direta em ambientes de experimentação e ferramentas de simulação.
- 4** Taxa de transferência máxima igual ou maior a 100 Mbps.

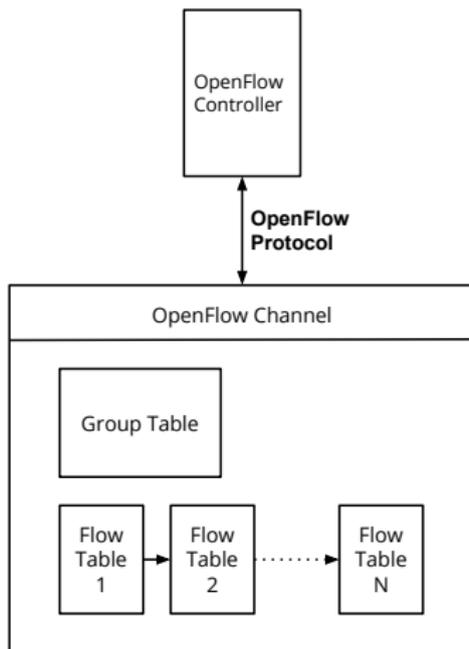
- 1 Introdução
- 2 Revisão Bibliográfica**
- 3 Arquitetura
- 4 Desenvolvimento
- 5 Avaliação
- 6 Conclusão

Para o desenvolvimento do *software switch*, além do protocolo *OpenFlow*, foi importante realizar o levantamento de ferramentas para o ambiente de testes, divididos em:

- *Software switches*
- Controladores
- Ferramentas de Teste e emulação

- Interface aberta padronizada pela *Open Network Foundation*
- Especificação define a interação do plano de controle com os *switches* que suportam o protocolo
- Define uma abstração de fluxos, os quais habilitam grande flexibilidade no encaminhamento de pacotes

Match	Priority	Instruction
all	0	apply actions -> output:controller
eth_dst:00:00:00:00:00:01	100	apply actions -> output:1

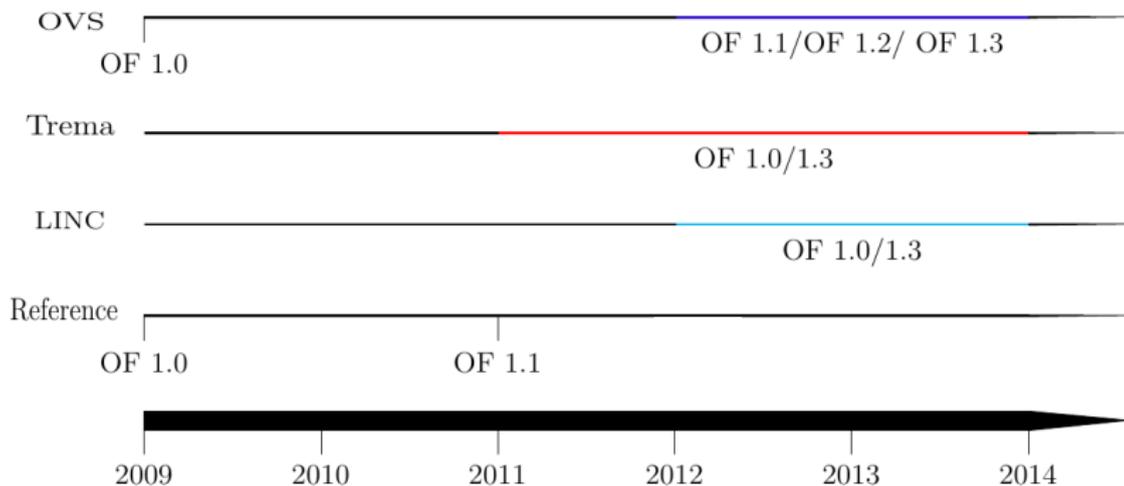


- Foram analisados os *software switches* de referência, o Open vSwitch, LINC e Trema
- No começo desse projeto estavam disponíveis apenas o OVS e o *reference switch*, enquanto LINC e Trema começaram a ser desenvolvidos durante o nosso trabalho

Switch	Linguagem	Ferramenta de emulação	Modo	OF-Config
Reference	C	Sim	userspace	Não
Open vSwitch	C	Sim	userspace/kernel	Não
LINC	Erlang	Não	userspace	Sim
Trema	C/Ruby	Não	userspace	Não

Revisão Bibliográfica

Linha do tempo do suporte ao OpenFlow nos software switches



Controladores

- NOX
- POX
- Floodlight
- Ryu

Teste e emulação

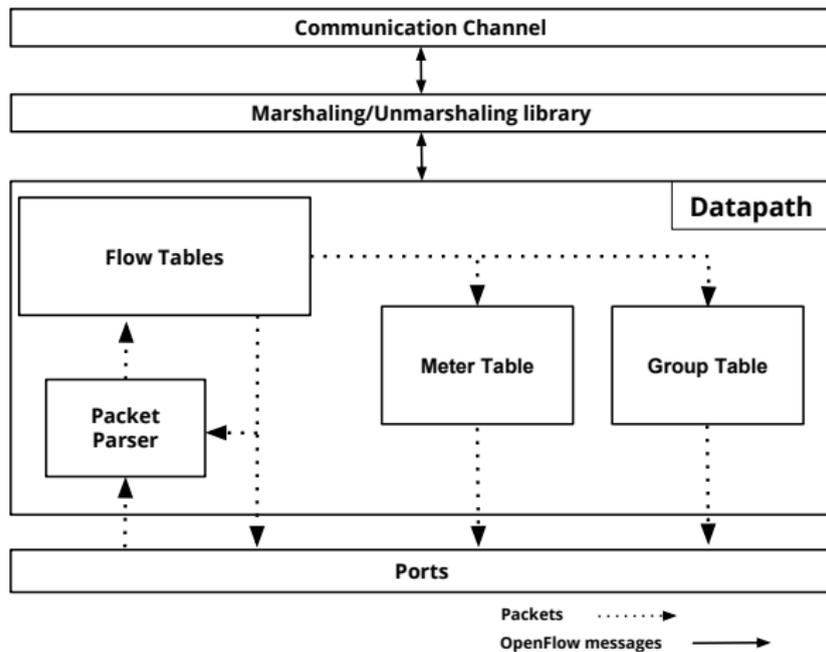
- OF-Test
- Ryu
- Mininet

- 1 Introdução
- 2 Revisão Bibliográfica
- 3 Arquitetura**
- 4 Desenvolvimento
- 5 Avaliação
- 6 Conclusão

Processo de definição da Arquitetura do *software switch* a partir do *fork* do *switch* de referência da versão 1.1 do *OpenFlow*

- 1 Processo simples de engenharia reversa para extrair os blocos funcionais do *switch* da versão 1.1
- 2 Identificação dos blocos faltantes a partir do *OpenFlow* da especificação do *OpenFlow* 1.3
- 3 Finalmente, o desenho da arquitetura foi construído seguindo uma abordagem *bottom-up*

Esse processo de extração foi importante pela falta de documentação e um desenho claro da relação entre os componentes do *switch* da versão 1.1



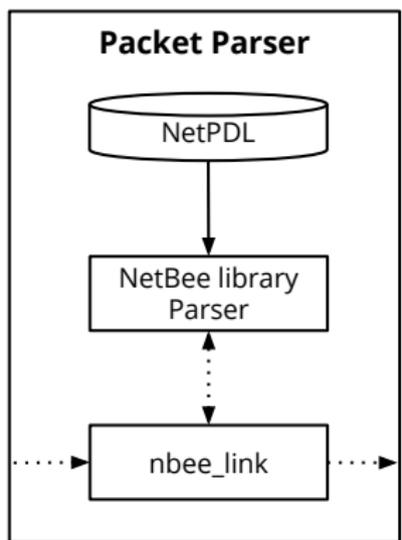
- 1 Introdução
- 2 Revisão Bibliográfica
- 3 Arquitetura
- 4 Desenvolvimento**
- 5 Avaliação
- 6 Conclusão

- Implementação do *OpenFlow* em software oferece grande flexibilidade
- Especificação descreve funcionalidades em alto nível
 - Liberdade para utilizar algoritmos e estruturas de dados
- Linguagens C e C++

- Biblioteca de representação de mensagens do protocolo
- Responsável pela conversão do formato interno para o formato de rede e vice-versa (Marshaling/unmarshaling *library* na figura da Arquitetura do *switch*)
- Biblioteca fácil de adicionar novas mensagens

O formato OXM substitui a antiga estrutura fixa das mensagens com campos de *match*

- Formato *Type-Lenght-Value* (TLV)
- Necessidade de pensar na melhor estrutura interna para suportar o formato
- Desafios na implementação do *parsing* de pacotes

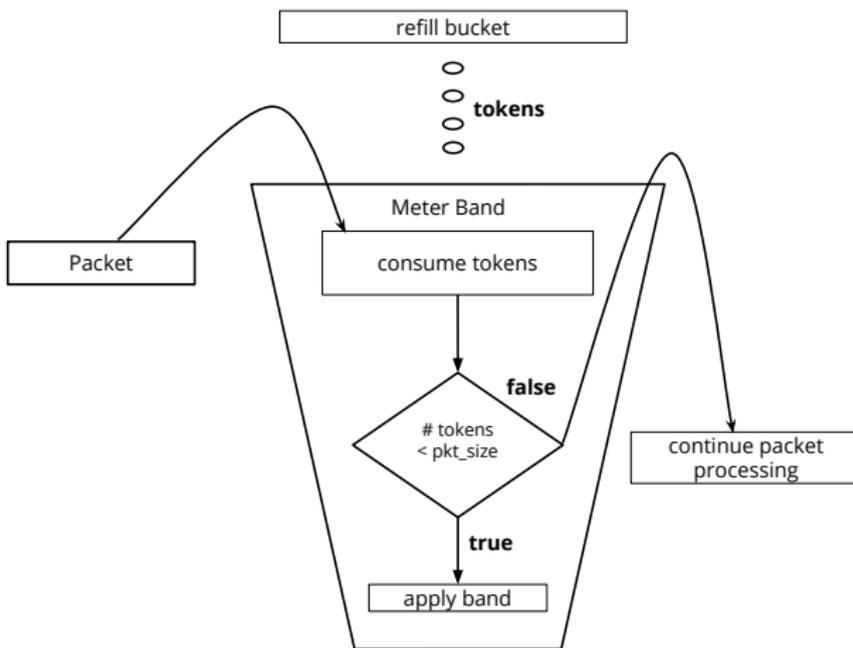


..... ► Packets

————► NetPDL packet description

Ao implementar a ação de alterar os campos do pacote, lidou-se com dois desafios:

- Consistência e validação do campo a ser modificado
- Recálculo dos valores de *Checksum*, existentes em alguns dos protocolos



Conexões auxiliares

Prova de conceito implementada, adicionando suporte à um canal adicional para mensagens do tipo *Packet In*

Papel do controlador

Além de permitir a troca do papel do controlador, o *switch* não permite que controladores escravos enviem mensagens que modifiquem o seu estado.

Filtragem de eventos

A implementação permite configurar os tipos de mensagem assíncronas que um controlador pode receber em função do seu papel no plano de controle.

- Licença BSD 3-clause
- GitHub
- Manutenção e testes de regressão

- 1 Introdução
- 2 Revisão Bibliográfica
- 3 Arquitetura
- 4 Desenvolvimento
- 5 Avaliação**
- 6 Conclusão

Os dois *frameworks* de testes descritos na revisão bibliográfica foram utilizados:

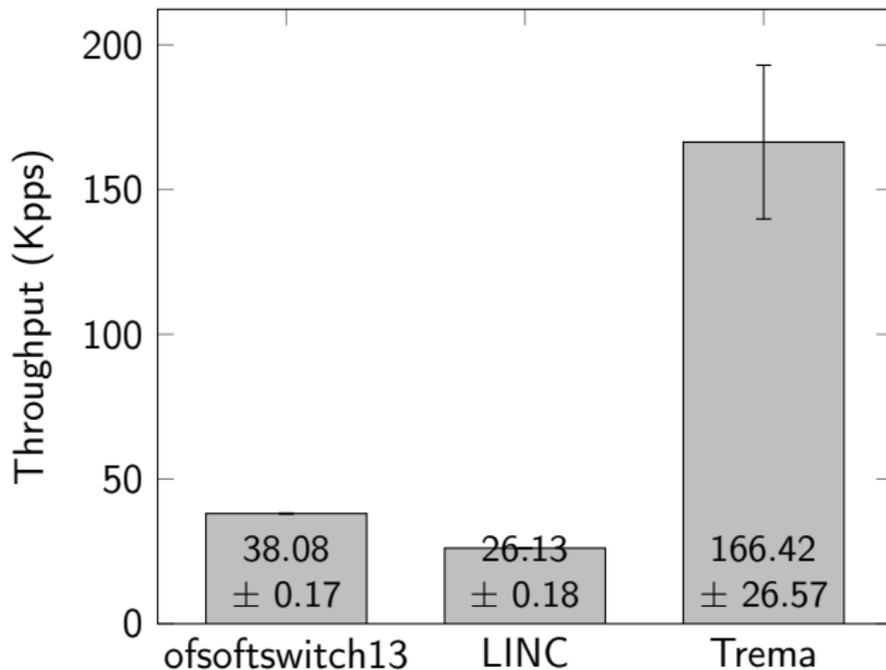
- **OFTest.** Validação das mensagens suportadas pelo *switch*.
 - Todos os testes básicos de mensagens foram bem sucedidos.
- **Ryu Test.** Resultados obtidos da certificação. Resultados validam funcionamento do *Datapath*.

Switch	Action	Set-Field	Match	Group	Meter	Total
ofsoftswitch13	89%	95%	99%	100%	83%	97%
Open vSwitch	60%	56%	74%	0%	0%	67%
LINC	42%	40%	60%	20%	0%	53%
Trema	89%	93%	99%	100%	94%	97%

- Objetivo dessa avaliação é verificar a máxima taxa de transferência do *software switch*
- Embora o objetivo não seja ser o *switch* com maior taxa de transferência, comparações com outros *softwares switches* são feitas para efeito de situar o nosso trabalho
- Comparações são interessantes para ajudar os usuários a decidir pela plataforma que atenda melhor os seus objetivos

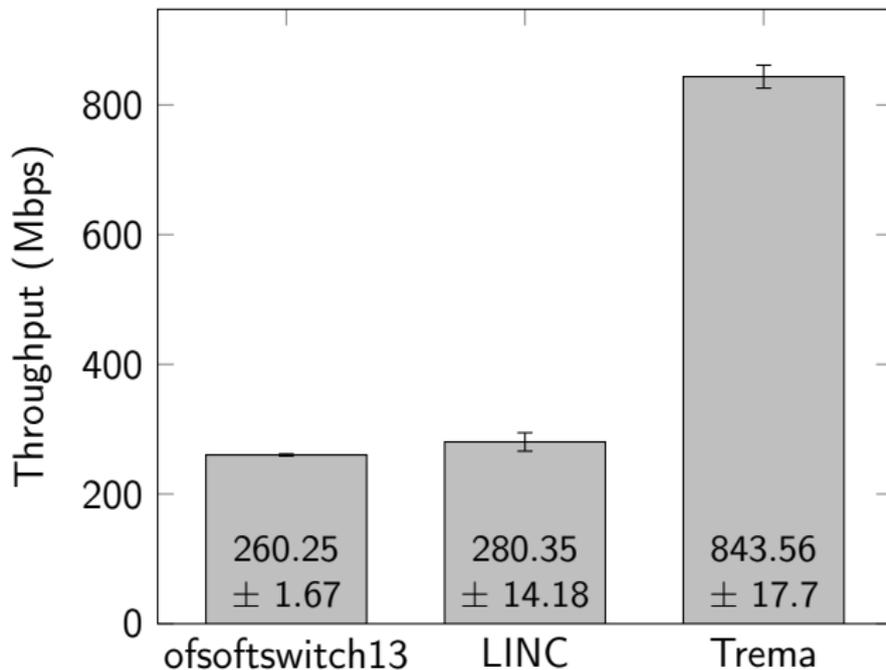
Avaliação

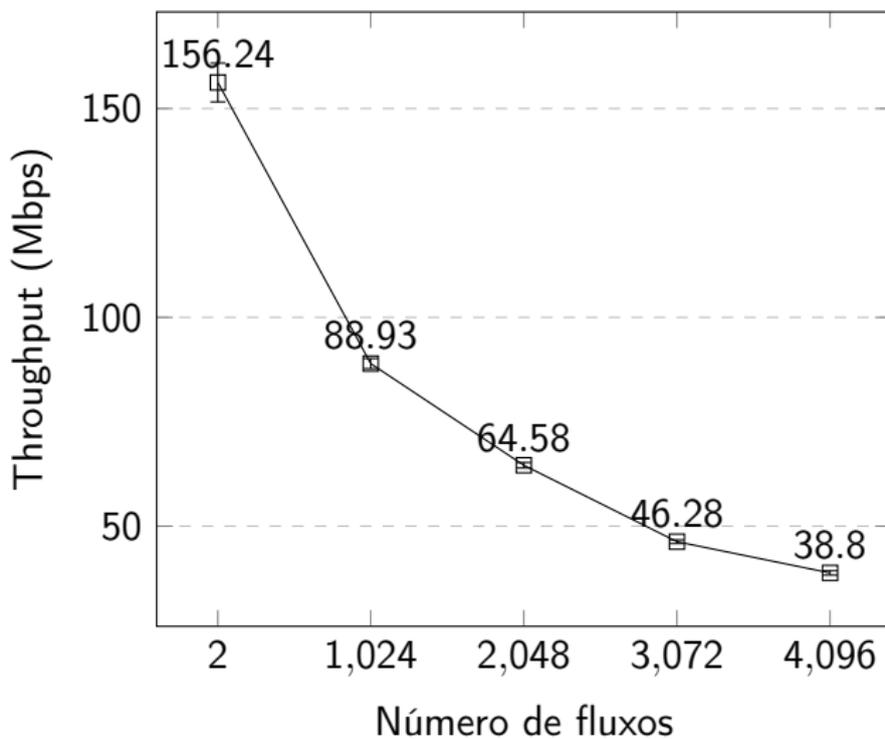
Máxima taxa de transferência para pacotes de 64 bytes

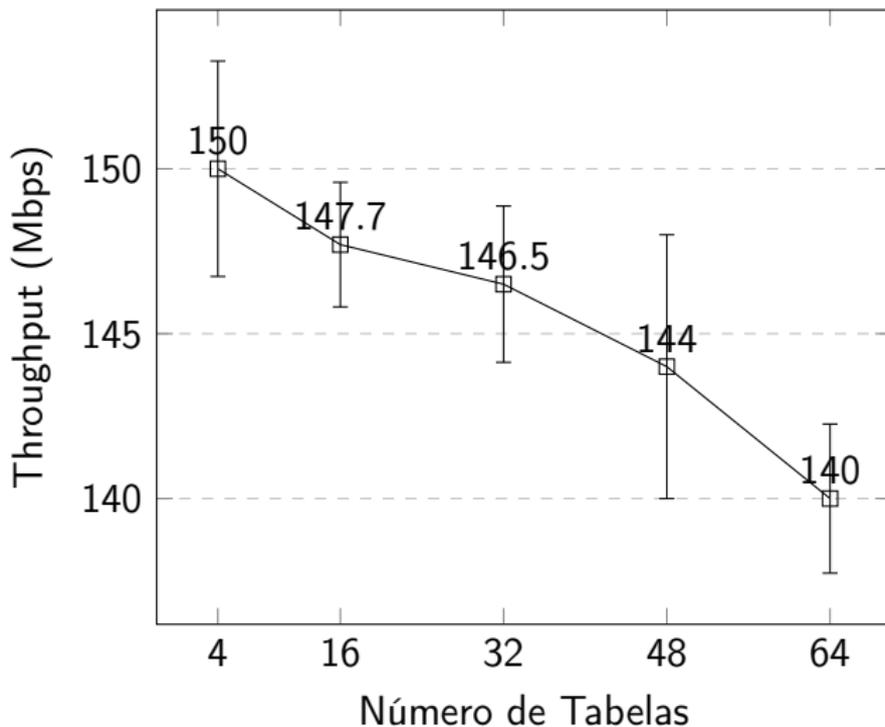


Avaliação

Máxima taxa de transferência para pacotes de 1500 bytes







Comparamos quanto de latência cada *software switch* adiciona em um simples teste de Ping.

Software Switch	Minímo (ms)	Média (ms)	Máximo (ms)	Desvio Padrão (ms)
ofsoftswitch13	0.30	1.07	1.82	0.31
LINC	303.90	554.77	821.48	253.03
Trema	0.12	0.40	0.48	0.04

- Código portátil com poucas modificações
- Porte para sistema OpenWrt em roteador wireless
- Novas oportunidades de experimentação

- 1 Introdução
- 2 Revisão Bibliográfica
- 3 Arquitetura
- 4 Desenvolvimento
- 5 Avaliação
- 6 Conclusão**

- Número de funcionalidades implementadas e funcionais próximo a 100% do total
- Comunidade *open source*
- Integração à instalação do Mininet
- Publicações

CPqD / ofsoftswitch13

Unwatch 45

★ Star 96

Fork 89

Contributors Traffic Commits Code frequency Punch card Network Members

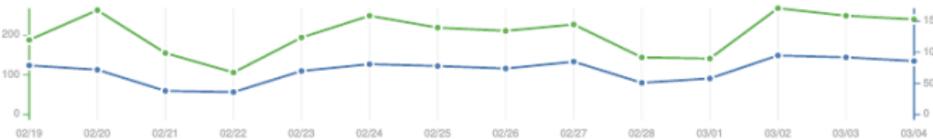
Git clones



336
Clones

79
Unique cloners

Visitors



3,125
Views

795
Unique visitors

- Base para prototipação de novas funcionalidades do *OpenFlow* pela ONF
- Academia
- Indústria

- Primeiro *switch OpenFlow* com suporte à versão 1.3
- Opção fácil e de baixo custo para testar funcionalidades mais avançadas do protocolo *OpenFlow*
- O *software switch* tornou-se um projeto *open source* reconhecido
- O projeto possibilitou o desenvolvimento de muitos trabalhos acadêmicos e na indústria
- Embora o Trema tenha maior desempenho, o nosso trabalho mostrou-se mais polivalente

“Houve mudança em relação ao TG1. Passou-se a utilizar o OFSoftswitch13 ao invés do Open vSwitch. Tal mudança ocorreu devida à simplificação substancial na implementação oferecida pelo OFSoftSwitch. O OFSoftSwitch13 oferece suporte nativo a operações na carga útil de pacotes e o monitoramento de protocolos da camada de aplicação. No Open vSwitch, em contraste, tais operações demandam diversas modificações no código do switch, nas bibliotecas de apoio utilizadas pelo mesmo e no kernel do Linux.”
(AppFlow: Suporte a Regras da Camada de Aplicação em Arquiteturas SDN OpenFlow)

"Because of the tight timelines in this project, if we found bugs in the switch, we wanted to quickly fix the bugs ourselves instead of depending on a third party to fix the bugs for us. The CPqD switch is a simpler switch than OVS and so its easier to fix bugs."
(CORD: Leaf-Spine Fabric with Segment Routing - Caso de uso do ONOS)

- Suporte às versões mais novas do *OpenFlow*
- Matching baseado em *hash*
- Novo mecanismo de análise de pacotes

Para mais informações sobre o projeto, código fonte e documentação, acesse:

<http://cpqd.github.io/ofsoftswitch13/>

- Eder Leão Fernandes, Christian Esteve Rothenberg. "OpenFlow 1.3 Software Switch". In Salão de Ferramentas XXXII Simpósio Brasileiro de Redes de Computadores - SBRC'2014, Florianópolis, 5 a 9 de Maio de 2014.
- E. L. Fernandes, C. Esteve Rothenberg and M. R. Salvador. "Software Defined Home Networking: Research Challenges and Innovation Opportunities."(invited paper), In International Workshop on Telecommunications (IWT'13), Santa Rita do Sapucaí, Brazil, 6-9 May 2013
- Rodrigo R. Denicol, Eder L. Fernandes, Christian E. Rothenberg, Zoltán Lajos Kis, "On IPv6 support in OpenFlow via Flexible Match Structures". OFELIA/CHANGE Summer School SummerSchool, Poster session, Berlin, Germany, November 7-11 November 2011.