Prospecção OpenFlow

Experimentos com o protocolo em plataformas de hardware

Marcel R. Faria

Janeiro/2016

Este documento é fruto de experimentos com o protocolo OpenFlow em plataformas de hardware comerciais. O objetivo foi ganhar experiência prática com o protocolo e entender as limitações deste nas suas atuais implementações. Como subprodutos foram gerados procedimentos de configuração de equipamentos Brocade, bem como instalação e operação do controlador ONOS.

Sumário

1. Introdução

Este documento é fruto de experimentos com o protocolo OpenFlow em plataformas de hardware comerciais. O objetivo foi ganhar experiência prática com o protocolo e entender as limitações deste nas suas atuais implementações. Como subprodutos foram gerados procedimentos de configuração de equipamentos Brocade, bem como instalação e operação do controlador ONOS. Os testes foram realizados no laboratório do projeto AmLight, utilizando-se ferramentas de software de código aberto.

O projeto AmLight, "Americas Lightpaths", é fruto de um consórcio colaborativo internacional entre redes acadêmicas, dentre as quais a RNP é membro. A AmLight provê uma arquitetura de conectividade de alta capacidade, entre a América Latina e os Estados Unidos, que suporta serviços de transporte de camada 2 e trânsito IP para as redes acadêmicas latino-americanas. Foi uma das primeiras redes do mundo a implementar SDN/OpenFlow[1] em produção para controle de seus equipamentos. Para isso necessitou validar em laboratório as funcionalidades dos equipamentos utilizados em sua rede e dos sistemas que seriam utilizados para controle e gerência da mesma.

O AmLight possui um laboratório próprio com um *subset* de seus dos equipamentos para propósitos de homologação de novas funcionalidades e serviços. Devido à necessidade de implementar OpenFlow, os equipamentos foram atualizados para suportar o protocolo.

2. Visão geral SDN/OpenFlow

Nos equipamentos de redes atuais, particularmente em switches ethernet e roteadores, há a separação das funções de plano de dados ("*data plane*"), que faz o encaminhamento dos dados, das de plano de controle ("*control plane*"), que toma as decisões sobre como estes serão encaminhados.

No plano de dados normalmente se encontram chips dedicados, cuja função é garantir o encaminhamento de pacotes da forma mais rápida e eficiente possível, com o mínimo de processamento. No plano de controle temos hardware mais lento, porém com maior capacidade de processamento, onde rodam os protocolos de camadas 2 e 3 e são mantidas as configurações e status das portas do equipamento. Ele também trata pacotes de exceção que não puderam ser processados no plano de dados.

Na arquitetura tradicional temos ambos os planos coexistindo em um mesmo equipamento, ou mesma "caixa", interligados por interfaces ou barramentos de alta velocidade. Os protocolos de comunicação entre os dois planos são proprietários, variando de fabricante para fabricante.

Software Defined Network, ou SDN, é uma arquitetura que propõem a separação do plano de controle do de dados, porém de forma a torná-lo uma entidade externa ao equipamento. Ou seja, no lugar de utilizarmos o plano de controle do fabricante, podemos fazer uso de nosso próprio.

Para que isso seja possível, é necessário formalizar a forma como os planos de controle e dados se comunicam, e também a interface que o plano de controle utilizará para especificar ou "programar" o comportamento do plano dados. Essa interface muitas vezes é referenciada como uma API (*Application Programming Interface*) ofertada pelo plano de dados ao plano de controle.

Na arquitetura SDN, o plano de controle é implementado pela figura do controlador, que é a parte do software que utiliza protocolos padronizados para se comunicar com um ou mais equipamentos de rede, que implementam o plano de dados. Ele é executado como uma aplicação convencional, podendo ser implementado em máquinas virtuais. O controlador interage com outras aplicações que, na verdade, fazem a orquestração da rede. A interface entre controlador e essas aplicações é referenciada como "*Northbound Interface*". Algumas implementações de controladores se autodenominam sistemas operacionais de rede por hospedarem aplicações e provisionarem estas funcionalidades, semelhantes à de um sistema operacional.

Em uma rede SDN pura, o plano de dados seria implementado por "*white-box*" switches, ou seja, equipamentos de hardware genérico, com um sistema operacional mínimo para permitir a conexão deste ao controlador. A interface entre os dois é chamada por alguns autores de "*Southbound Interface*". Atualmente

existem poucos fabricantes de "*white-box*" switches, sendo mais comum se encontrar equipamentos com funções hibridas, podendo operar tanto em modo convencional como em "*white-box*" ao mesmo tempo.

O esforço para padronização da arquitetura SDN tem sido liderado pela Open Networking Foundation (ONF)[2], organização sem fins lucrativos que tem se dedicado a elaborar padrões e difundir a tecnologia. Ela tem sido responsável pela padronização do protocolo OpenFlow, considerado o protocolo "*Southbound*" mais popular, que no momento de elaboração deste documento está em sua versão 1.5.1[3].

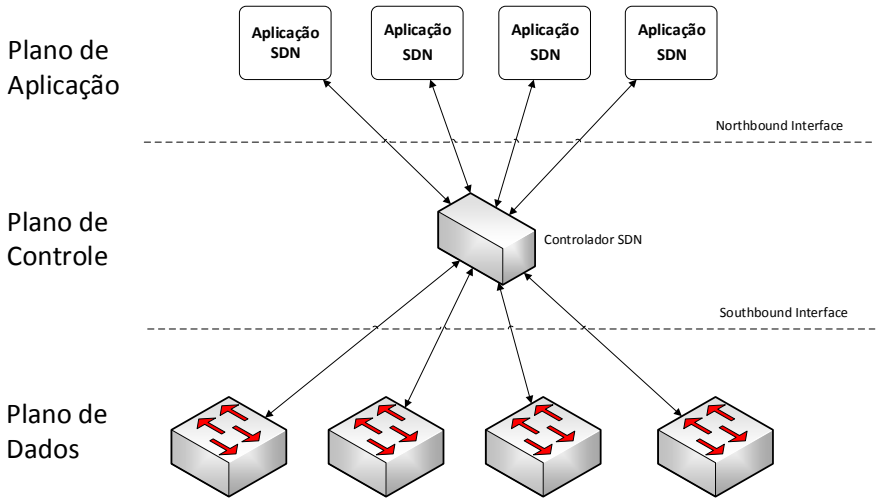


Figura 1: Arquitetura SDN segundo ONF.

A IETF e ITU-T também lançaram documentos descrevendo a arquitetura SDN[4 e 5] na visão destas duas organizações, que são similares ao proposto pela ONF.

Uma relação dos produtos e fabricantes que atualmente suportam o protocolo OpenFlow pode ser encontrada no site da ONF[6].

3. Prospecção

3.1. Descrição

A prospecção realizada se focou na arquitetura proposta pela ONF que tem, no protocolo OpenFlow, sua peça chave. Para poder trabalhar com OpenFlow é imprescindível que se tenha minimamente uma rede com equipamentos que suportem o protocolo, ou um plano de dados, e um controlador que permita "programar" o comportamento desses equipamentos, no caso, o plano de controle.

Atualmente muito da pesquisa realizada em OpenFlow utiliza soluções que virtualizam ou simulam uma rede de "*white-box*" switches. Como exemplo, temos Open vSwitch[7] - uma implementação completa de switch para ambientes virtuais, normalmente utilizado junto com o pacote de software Mininet[8], que permite criar redes virtuais com grande quantidade de nós.

Embora essas soluções se mostrem excelentes no aprendizado e na pesquisa para incremento do protocolo, ou mesmo a implementação deste em ambientes de nuvem, elas não seguem a realidade do suporte fornecido em muitas das implementações de hardware atualmente. O desenvolvimento de soluções em software é mais rápido e com custos muito inferiores.

Para que os fabricantes invistam no desenvolvimento de chips que suportem o protocolo OpenFlow de forma mais completa e efetiva, é necessário que haja padrões estáveis do protocolo para justificar o investimento

monetário e o tempo gasto no seu desenvolvimento. Da versão 1.0, lançada em dezembro de 2009, à 1.5.1 de março de 2015, tivemos cinco versões principais do padrão, e oito atualizações.

O objetivo da prospecção não foi o de se fazer uma avaliação extensiva sobre o suporte ao protocolo OpenFlow em plataformas de hardware comerciais, mas gerar experiência prática em o que seria necessário para implementar OpenFlow e gerenciar sua operação em plataformas de hardware comercial. Foram também gerados procedimentos para implantação do protocolo nos equipamentos disponíveis para testes.

Se contou com o apoio do projeto AmLight, que cedeu acesso a equipamentos de seu laboratório para experimentação. Os equipamentos eram switches Brocade, similares aos utilizados em alguns Pontos de Presença da rede Nacional de Ensino e Pesquisa.

Para os testes foram utilizadas ferramentas de software de código aberto atualmente disponíveis.

3.3. Topologia e pacotes de software utilizados

O laboratório disponibilizado pelo projeto AmLight consistia de uma máquina virtual hospedando os controladores, três switches Brocade, e mais quatro VM para testes de conectividade. Os modelos dos equipamentos Brocade eram: MLXe-4, MLX-4 e CES 2024F.

A máquina virtual do controlador rodava a distribuição Linux CentOS versão 6.7. As outras demais maquinas virtuais eram utilizadas apenas para realização de pings e traceroute. Abaixo temos um diagrama da topologia:

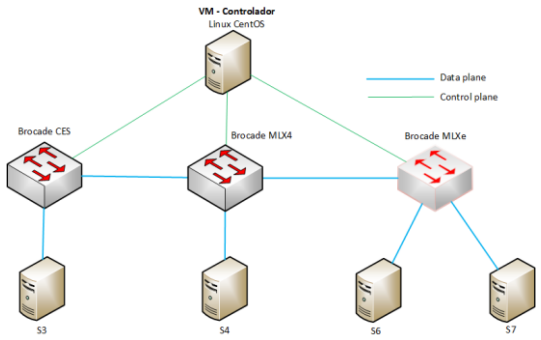


Figura 2: Laboratório AmLight.

Foram testados os seguintes controladores:

- ONOS (Open Network Operating System)[9] – versão 1.3.0;
- ODL (OpenDayLight)[10] – versão Lithium;
- Ryu Framework[11] – versão 3.25.

É importante ressaltar que os controladores testados não são aplicações SDN per si, ou seja sua função principal é apenas fazer a comunicação entre aplicações e plano de dados. Sem aplicações, não é possível orquestrar uma rede SDN. Todos eles, no entanto, já veem com algumas aplicações junto com as respectivas distribuições, normalmente com o intuito de servir de exemplo para desenvolvedores.

O pacote ONOS é relativamente recente, tendo seu primeiro *release* lançado em dezembro de 2014. Ele clama ser voltado para o mercado de provedores de Internet. É um esforço liderado pela Open Network Lab ou ON.LAB, organização sem fins lucrativos envolvendo pesquisadores das universidades Stanford e Berkeley, e empresas privadas, cujo segmento de atuação vai de provedores Internet, como AT&T e NTT, a fabricantes de

equipamentos de rede, como Cisco e Huawei. Recentemente foi anunciado que se tornaria um projeto colaborativo da Linux Foundation[12].

O OpenDayLight ou ODL, é um esforço mais antigo que se iniciou como um projeto de software aberto sendo custeado em conjunto por vários fabricantes de equipamentos de rede. Seu primeiro *release* público ocorreu em fevereiro de 2014. Atualmente também é listado como um projeto colaborativo da Linux Foundation.

Tanto o ONOS como o ODL são desenvolvidos em linguagem Java e utilizam como componente básico o pacote Apache Karaf[13], que funciona como um container onde rodam os pacotes do controlador. O Karaf também pode hospedar as aplicações que se comunicarão com o controlador servido de "*northbound interface*". No Karaf as aplicações podem ser carregadas e descarregadas de forma dinâmica.

O Ryu é um *framework* na linguagem Python sendo desenvolvido pela NTT Communications do Japão. Basicamente é um conjunto de bibliotecas e pequenos programas de exemplo para permitir o desenvolvimento de aplicações que utilizem OpenFlow. Atualmente o Ryu suporta até a versão 1.5 do protocolo OpenFlow. Ele foi útil nos experimentos pois forneceu uma forma de se ter um acesso de mais baixo nível nas trocas de mensagens entre controlador e equipamentos.

Todos os três controladores possuem algum tipo de interface REST, ou seja, é possível obter informações ou enviar comandos ao controlador utilizando-se requisições HTTP. Embora esse mecanismo possa ser considerado como uma "*northbound interface*", funcionando como um canal de comunicação entre controlador e aplicações SDN, o que se verificou foi que cada pacote de software implementa essa interface a sua maneira.

Para esta prospecção, o pacote de software escolhido para geração dos procedimentos de instalação e configuração da parte do controlador foi o ONOS por este fornecer, pelo menos na versão utilizada nos testes, uma interface web mais rica em funcionalidades na opinião do autor.

3.2. Experimentos realizados

O controlador Ryu foi utilizado para testar a conexão dos switches ao controlador. Foram também feitos testes com uma aplicação deste pacote que implementa uma interface Rest. O OpenDayLight foi instalado para testes com sua interface web, DLUX e também com sua API Rest.

A geração de fluxos foi realizada via interface web do controlador ONOS. A interface CLI do ONOS também foi utilizada, mas apenas para obtenção de status da topologia e das tabelas de fluxos.

Durante testes realizados, ficou clara a dependência que os equipamentos de rede têm do controlador. Uma vez configurado o protocolo e habilitadas as interfaces que irão suportá-lo, os únicos comandos disponíveis foram os de visualização de status e os para forçar a limpeza da tabela de fluxos.

A visualização dos fluxos instalados na tabela diretamente no equipamento se mostrou bastante prática para propósitos de *troubleshooting*, porém o autor pensa que pode tornar essa atividade complexa se o número de fluxos e nós de rede for muito grande. Novamente então necessitamos do controlador para entender o comportamento que foi "programado" para determinado fluxo.

3.3. Procedimentos gerados

Os procedimentos gerados na prospecção estão descritos em dois anexos deste documento:

- Anexo A – Configuração de OpenFlow em Switches Brocade;
- Anexo B - ONOS – Open Network Operation System.

4. Conclusões

O protocolo OpenFlow fornece funcionalidades que permitem acesso às funções de hardware de rede de uma forma que não é possível com os atuais protocolos abertos. No entanto, apesar das possibilidades ofertadas pela tecnologia, a implantação do protocolo parece estar ocorrendo de forma mais lenta, tanto em hardware, como em software (controladores ou aplicações SDN).

Apenas recentemente os fabricantes passaram suportarem a versão 1.3 do protocolo OpenFlow, e não tem se visto grande oferta de "*white-box*" switches. Com relação a software, os dois principais projetos de controladores, OpenDayLight e ONOS, ainda estão trabalhando no suporte às versões 1.3 e 1.4, enquanto a ONF já lançou a versão 1.5.1 do protocolo.

De fato, tem havido o surgimento de várias novas aplicações SDN, impulsionadas principalmente pelo meio acadêmico, porém, a impressão que se tem é que ainda estamos na fase de depuração dessas ideias, onde algumas serão abandonadas e outras ganharão suporte mais sólido da comunidade Internet.

As aplicações SDN são um fator particularmente crítico na adoção de redes definidas por software, pois são responsáveis pela camada de orquestração, podendo ser pensadas como a "inteligência" da rede. São elas que poderão justificar ao mercado a adoção do novo paradigma em termos de ganho monetário.

Ainda não está certo se as redes SDN irão no futuro substituir completamente o paradigma atual, mas parece claro que elas vieram para ficar. No entanto, a impressão que se tem após a realização da prospecção, é que a tecnologia ainda necessita de mais tempo para ganhar maturidade, antes de sua adoção em larga escala pelo mercado.

5. Agradecimentos

Esta prospecção só foi possível graças ao apoio do projeto AmLight, que cedeu acesso aos recursos de seu laboratório. Autor agradece em particular a Jerônimo Bezerra e Humberto Galiza, pelo apoio prestado durante a realização dos experimentos e sugestões na elaboração deste documento.

6. Referências

[1] "Benefits brought by the use of OpenFlow/SDN in the AmLight intercontinental research and education network" - Ibarra, J.; Bezerra, J.; Alvarez, H.; Cox, D.; Stanton, M.; Machado, I.; Grizendi, E.; Lopez. L. - IM 2015 - 14th IFIP/IEEE Symposium on Integrated Network and Service Management, Ottawa, Canada

[2] Open Networking Foundation (ONF) - https://www.opennetworking.org

[3] "OpenFlow® Switch Specification 1.5.1" - https://www.opennetworking.org

[4] RFC 7426 – "Software-Defined Networking (SDN): Layers and Architecture Terminology" - http://www.ietf.org/

[5] ITU-T Y.3300 – "Framework of software-defined networking" - http://www.itu.int

[6] "Products Listing - Open Networking Foundation" - https://www.opennetworking.org/products-listing

[7] Open vSwitch (OvS) - http://openvswitch.org/

[8] Mininet - http://mininet.org/

[9] ONOS (Open Network Operating System) - http://onosproject.org/

[10] OpenDaylight - https://www.opendaylight.org/

[11] Ryu SDN Framework - http://osrg.github.io/ryu/

[12] Linux Foundation – http://www.linuxfoundation.org/

[13] Karaf - http://karaf.apache.org/

Anexo A – Configuração de OpenFlow em Switches Brocade

1. Introdução

Este documento tem por finalidade descrever os procedimentos básicos para configuração de OpenFlow em switches Brocade e comandos para verificação de status.

O suporte a OpenFlow nos equipamentos deste fabricante começou na versão 5.6 do sistema operacional IronWare, com suporte a versão 1.0 do protocolo. Atualmente o IronWare está no release 5.9, com suporte a versão 1.3 do protocolo.

Neste documento não foi feito uma avaliação extensiva suporte do protocolo pelo fabricante. Portanto, independente da versão de IronWare que se deseja utilizar, recomenda-se que se avalie com o fabricante o suporte do equipamento, tanto no chassis quanto nos linecards, às features OpenFlow que se deseja utilizar. O equipamento utilizado na demonstração das configuração e captura da saída dos comandos foi um switch Brocade MLX-4 com IronWare versão 5.6.00f, utilizando-se versão 1.0 do OpenFlow.

1.1. Limitações e considerações

Pelo manual "Multi-Service IronWare - Software Defined Networking (SDN) - Configuration Guide"[1] a plataforma MLX- 4 tem as seguintes limitações:

- Número máximo de flows por porta: 4096
- Número máximo de flows suportados no switch: 64K
- Número máximo de VLANs protegidas: 2K

Abaixo algumas considerações sobre a atual implementação Brocade:

- O OpenFlow deve ser habilitado globalmente, antes de se habilitar interfaces para OpenFlow;
- Cada interface deve ser explicitamente habilitada ou desabilitada para uso de OpenFlow;
- Antes de desabilitar OpenFlow globalmente, cada interface deve ter OpenFlow desabilitado;
- Como ação default escolhida pela Brocade para portas habilitadas com OpenFlow, pacotes que não sofram match em nenhuma regra são descartados.

1.2. Modo de Operação Híbrido

Os equipamentos Brocade do laboratório, em sua atual implementação, suportam modo de operação híbrido, permitindo a habilitação de OpenFlow por porta.

Portas operando em modo híbrido (*hybrid port mode*) suportam tanto tráfego OpenFlow como tráfego normal roteado, isso é feito através do conceito de "Protected VLANs" e "Unprotected VLANs":

- "Protected VLANs" - tráfego não sujeito as regras OpenFlow;
- "Unprotected VLANs" - tráfego sujeito as regras OpenFlow.

2. Procedimentos de configuração

Esta seção descreve os procedimentos básicos para habilitar OpenFlow em um switch Brocade MLX-4, rodando IronWare R05.6.00.

2.1. Habilitado OpenFlow

Openflow dever ser habilitado globalmente no equipamento. A versão de OpenFlow a ser suportada depende da release de IroWare e dos módulos instalados no equipamento.

```
telnet@MLX4(config)#openflow enable ofv100
Warning: Please configure [system-max openflow-flow-entries #] to accept any flows
Warning: Please configure [system-max openflow-pvlan-entries #] to accept any Hybrid flows
Warning: Please configure [system-max openflow-unprotectedvlan-entries #] to accept
   Configured Unprotected VLANs for Hybrid ports
```

Logo após o OpenFlow ser configurado, são solicitados que outros quatro outros parâmetros de sistema “system-max“ sejam configurados, descritos em mais detalhes na próxima seção.

Para remover o comando acima, o OpenFlow deve ser primeiro desabilitado individualmente em todas as interfaces equipamento, que tenham posteriormente configuradas para o suportar o protocolo.

2.2. Configurando parâmetros de sistema

São necessários a configuração de quatro parâmetros de sistema para ativação de OpenFlow (não se aplica à série de switches NetIron CES):

- *openflow-flow-entries* - número de entradas;
- *openflow-pvlan-entries* - número de VLANs protegidas (não OpenFlow);
- *openflow-unprotectedvlan-entries* - número de VLANs Openflow;
- *np-openflow-flow-entries* - número de por NP por slot.

```
SSH@NetIron MLX-4 Router(config)#system-max openflow-flow-entries ?
  DECIMAL   Valid range 0 to 65536 (default: 0)

SSH@NetIron MLX-4 Router(config)#system-max openflow-pvlan-entries ?
  DECIMAL   Valid range 0 to 2048 (default: 0)

SSH@NetIron MLX-4 Router(config)#system-max openflow-unprotectedvlan-entries ?
  DECIMAL   Valid range 0 to 4096 (default: 0)

SSH@NetIron MLX-4 Router(config)#system-max np-openflow-flow-entries layer2or3 ?
  DECIMAL   Valid range 0 to 32768 (default: 0)
```

No exemplo abaixo, são configuradas 1000 entradas de flows, 1000 vlans protegidas, 1000 vlans OpenFlow e 1000 entradas para o NP (Network Processor) nos slots 1 e 2:

```
system-max openflow-flow-entries 1000

system-max openflow-pvlan-entries 1000

system-max openflow-unprotectedvlan-entries 1000

system-max np-openflow-flow-entries layer2or3 1000 slot 1 3
```

2.3. Configurando conexão com o controlador

A conexão com o controlador pode ser realizada de dois modos, com ou sem encriptação SSL habilitada:

- active – em que o equipamento busca conexão com o controlador;
- passive – aguarda o controlador se conectar.

A implementação da Brocade para a versão de software 5.6.0, comporta a configuração de até três controladores.

Normalmente a conexão com o controlador é iniciada pelo switch. No caso abaixo, está se configurando o endereço controlador, máquina X.X.X.45, com ssl desabilitado (sem canal seguro), e utilizando a porta TCP 6653 (porta alocada pela IANA):

```
SSH@NetIron MLX-4 Router(config)#openflow controller ip-address X.X.X.45 no-ssl port 6653
```

O switch, nesta implementação, após a configuração do comando acima, passa a tentar estabelecer conexão com o controlador em intervalos de aproximadamente de 5 em 5 segundos. No exemplo abaixo vemos o switch, ip X.X.X.35, tentar estabelecer comunicação com o controlador na máquina X.X.X.45. Como o controlador não está configurado, a máquina do controlador envia um reset da conexão TCP de volta para o switch:

```
# sudo tcpdump -nn -i ens160 port not 22 and ip host X.X.X.35
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens160, link-type EN10MB (Ethernet), capture size 65535 bytes
10:40:55.386529 IP X.X.X.35.7801 > X.X.X.45.6653: Flags [S], seq 414776148, win 65000, options
[mss 1460], length 0
10:40:55.386671 IP X.X.X.45.6653 > X.X.X.35.7801: Flags [R.], seq 0, ack 414776149, win 0,
length 0

10:41:00.386545 IP X.X.X.35.7801 > X.X.X.45.6653: Flags [S], seq 416017460, win 65000, options
[mss 1460], length 0
10:41:00.386683 IP X.X.X.45.6653 > X.X.X.35.7801: Flags [R.], seq 0, ack 1241313, win 0,
length 0

10:41:05.386532 IP X.X.X.35.7801 > X.X.X.45.6653: Flags [S], seq 417260092, win 65000, options
[mss 1460], length 0
10:41:05.386671 IP X.X.X.45.6653 > X.X.X.35.7801: Flags [R.], seq 0, ack 2483945, win 0,
length 0
```

Alguns controladores ainda utilizam a porta TCP 6633 para conexão com os equipamentos. Esta porta foi inicialmente utilizada nas primeiras implementações de OpenFlow, mas atualmente ela consta como sendo reservada na IANA.

2.4. Configurando ação padrão

Quando o pacote não faz *match* em nenhuma das regras de fluxo configuradas pelo controlador, a ação padrão da implementação Brocade é descartá-lo. Com isso a mensagem OpenFlow PACKET_IN nunca será gerada.

Essa mensagem é utilizada por algumas implementações de controladores, portanto dependo do controlador escolhido, o comando abaixo não é necessário.

O comando abaixo altera a ação padrão:

```
SSH@NetIron MLX-4 Router(config)#openflow default-behavior send-to-controller
```

A mensagem PACKET_IN então passará a ser gerada quando não houver nenhum *match*.

2.5. Habilitando OpenFlow nas portas

No exemplo abaixo temos uma porta habilitada para operar em OpenFlow em modo layer 2 (default), ou seja, fazer *match* apenas de campos do cabeçário ethernet:

```
!
interface ethernet 1/2
 openflow enable
 enable
!
```

Abaixo a porta e 1/1 está habilitada para operar em modo hibrido, sendo a VLAN 10 protegida (operando em modo convencional, não sujeita as regras para *match* de fluxos OpenFlow):

```
!
interface ethernet 1/1
 openflow protected-vlans 10
 openflow enable layer2 hybrid-mode
 enable
!
```

Na listagem abaixo temos uma porta habilitada em modo 2 e 3, ou seja, *match* de campos dos cabeçários ethernet e IP:

```
!
interface ethernet 1/3
 openflow enable layer23
 enable
!
```

O suporte a esse modo depende do módulo que comporta a porta. Alguns módulos só suportam o modo layer2.

2.6. Reiniciando switch

As configurações devem ser gravadas e o switch deve ser reiniciado para se tornarem ativas:

```
SSH@NetIron MLX-4 Router(config)#^Z
SSH@NetIron MLX-4 Router#write mem
Write startup-config done.
Copy STARTUP CONFIG FILE to standby MP, please wait.
Not Needed.
SSH@NetIron MLX-4 Router#reload
Checking for coherence...
Done.
Are you sure? (enter 'y' or 'n'): Connection to X.X.X.35 closed by remote host.
Connection to X.X.X.35 closed.
```

3.Verificando Status

Essa sessão lista alguns dos comandos que foram considerados mais úteis na opinião do autor para verificação de status e *troubleshhooting* de problemas.

O comando abaixo é uma das melhores opções para verificar as configurações OpenFlow. Ele fornece dados de versão, lista as portas estão habilitadas para o protocolo, o endereço do controlador e status da conexão com o mesmo. Alguns itens que merecem atenção foram realçados em vermelho:

```
SSH@NetIron MLX-4 Router#show openflow
Administrative Status:       Enabled
SSL Status:                  Enabled

Controller Type:             OFV 100
Number of Controllers:       1

Controller 2:
Connection Mode:       active, TCP,
Controller Address:     190.103.187.45
Connection Port:       6633
Connection Status:      OPENFLOW_ESTABLISHED


Match Capability:
L2 : Port, Source MAC, Destination MAC, Ether type, Vlan, Vlan PCP
L3 : Port, Vlan, Vlan PCP, Ethertype(IP,ARP,LLDP), Source IP, Destination IP, IP Protocol, IP
TOS, IP Src Port, IP Dst Port
L23: All

Normal Openflow Enabled Ports:      e1/1 e1/2 e3/1
Openflow Hybrid Interfaces:


Default action: send-to-controller
Maximum number of flows allowed: 1000
Active flow: 4

Maximum number of Protected Vlans allowed: 1000
Maximum number of Unprotected Vlans allowed: 1000
Total number of Unprotected Vlans: 0
SSH@NetIron MLX-4 Router#
```

O comando abaixo permite visualizar de forma rápida o status da conexão com o controlador:

```
SSH@NetIron MLX-4 Router#show openflow controller
Openflow controller information

-------------------------------------------------------------------------------------------
     Controller      Mode   TCP/SSL          IP-address      Port                Status
-------------------------------------------------------------------------------------------
              1    active       TCP      190.103.187.45      6633   OPENFLOW_ESTABLISHED
```

O identificador do datapath é uma informação útil para *troubleshooting*, pois é a forma como o switch é identificado no controlador:

```
SSH@NetIron MLX-4 Router#show openflow datapath-id

Openflow datapath id 0012f2f759000000
```

As interfaces com OpenFlow configurado podem ser obtidas:

```
SSH@NetIron MLX-4 Router#show openflow interface

Total number of Openflow interfaces: 6

Port  Link     Speed Tag MAC            OF-portid   Name           Mode
1/1   Up       10G   Yes 0012.f2f7.5900 1                          Hybrid-Layer2
1/2   Up       10G   Yes 0012.f2f7.5901 2                          Layer2
3/1   Up       1G    Yes 0012.f2f7.5960 97                         Layer2
3/2   Up       1G    Yes 0012.f2f7.5961 98                         Layer2
3/3   Up       1G    Yes 0012.f2f7.5962 99                         Layer2
3/4   Down     None  Yes 0012.f2f7.5963 100                        Layer2
```

Recursos utilizados:

```
SSH@NetIron MLX-4 Router#show openflow resources

Openflow Resources:

CAM Profile: default
Used - Number of HW entries consumed
Free - Number of Port based flows that can be successfully programmed

Slot: 1  Module: NI-MLX-10Gx4 4-port 10GbE Module
Openflow Layer2or3 Flows: MAX: 1000    Used: 0  Free: 1000

Slot: 3  Module: NI-MLX-1Gx20-SFP 20-port 1GbE-100FX Module
Openflow Layer2or3 Flows: MAX: 1000    Used: 0  Free: 1000
```

Um dos comandos mais importantes é o que lista a relação de todos os flows ativos no equipamento. Em um roteador, ele equivaleria a verificar a tabela de rotas do equipamento:

```
SSH@NetIron MLX-4 Router#show openflow flows
Total Number of data packets sent to controller:               13783
Total Number of data bytes sent to controller  :             1115940

Total Number of Flows: 2
   Total Number of Port based Flows: 2
   Total Number of L2 Generic Flows: 0
   Total Number of L3 Generic Flows: 0
   Total Number of L2+L3 Generic Flows: 0
   Total Number of L23 Generic Flows: 0

Total Number of Hardware entries for flows: 2
   Total Number of Hardware entries for Port flow: 2
   Total Number of Hardware entries for Generic flow: 0

Total Number of Openflow interfaces: 3
   Total Number of L2  interfaces: 3
   Total Number of L3  interfaces: 0
   Total Number of L23 interfaces: 0

Flow ID: 890 Priority: 100 Status: Active
   Rule:
      In Port:  e3/1
      Source Mac:  000f.1f69.97de
           Destination Mac:      0060.dd46.4762
      Source Mac Mask:   ffff.ffff.ffff
      Destination Mac Mask:    ffff.ffff.ffff
   Action: FORWARD
      Out Port:  e1/1
   Statistics:
      Total Pkts: 0
      Total Bytes: -NA-

Flow ID: 894 Priority: 100 Status: Active
   Rule:
      In Port:  e1/1
      Source Mac:  0060.dd46.4762
      Destination Mac:   000f.1f69.97de
      Source Mac Mask:   ffff.ffff.ffff
      Destination Mac Mask:    ffff.ffff.ffff
   Action: FORWARD
      Out Port:  e3/1
   Statistics:
      Total Pkts: 0
      Total Bytes: -NA-

SSH@NetIron MLX-4 Router#
```

Algumas vezes é necessário limpar a tabela de *flows* ou algum *flow* específico:

```
clear openflow flowid <num-flow>

clear openflow all
```

4. Referências

[1] “Multi-Service IronWare - Software Defined Networking (SDN) Configuration Guide – Supporting Multi-Service IronWare R05.6.00” – Brocade – 09/12/2013

Anexo B - ONOS – Open Network Operation System

1. Descrição

O controlador ONOS, ou Open Network Operating System, é um projeto de um controlador de código aberto, voltado, segundo o website[1] do projeto, para o segmento de provedores de Internet.

2. Instalação

O procedimento descrito neste documento utilizou o release 1.3.0 do ONOS, de 18 de setembro de 2015, também conhecida como “Drake”.

2.1. Pré-requisitos

Hardware

Abaixo os requisitos mínimos para instalação:

- Ubuntu 14.04 ou CentOS 6.7 64-bits
- 2 GB deRAM
- 2 ou mais núcleos de processamento

Software

São necessários 2 pacotes de software:

- Java 8 – Versão da Oracle
- ONOS 1.3.0

2.2. Instalando Oracle Java 8 no Linux

O procedimento abaixo foi feito na distribuição Ubuntu 14.04, mas é muito semelhante a que seria realizado no CentOS 6.7.

1. Fazer download do Java Java SE Development Kit 8 na Oracle e descompactar o pacote no diretório /usr/lib/jvm. No caso o pacote é o arquivo jdk-8u51-linux-x64.tar.gz, para linux 64 bits:

```
$ sudo tar xvzf jdk-8u51-linux-x64.tar.gz -C /usr/lib/jvm
$ sudo chown -R root:root /usr/lib/jvm/jdk1.8.0_51
```

2. Atualizar os links do sistema:

```
$ sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/jdk1.8.0 51/bin/javac 2

$ sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/jdk1.8.0 51/bin/java 2

$ sudo update-alternatives --config javac

There are 2 programs which provide 'javac'.

  Selection    Command
-----------------------------------------------
*+ 1           /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.85-2.6.1.2.el7_1.x86_64/bin/javac
   2           /usr/lib/jvm/jdk1.8.0_51/bin/javac
```

```
Enter to keep the current selection[+], or type selection number: 2

$ sudo update-alternatives --config java

There are 2 programs which provide 'java'.

  Selection    Command
-----------------------------------------------
*+ 1           /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.85-2.6.1.2.el7 1.x86 64/jre/bin/java
   2           /usr/lib/jvm/jdk1.8.0 51/bin/java

Enter to keep the current selection[+], or type selection number: 2
```

3. Checar instalação:

```
$ java -version
java version "1.8.0_51"
Java(TM) SE Runtime Environment (build 1.8.0_51-b16)
Java HotSpot(TM) 64-Bit Server VM (build 25.51-b03, mixed mode)
```

4. Setup de variáveis de ambiente:

```
$ sudo vi /etc/environment
JAVA HOME=”/usr/lib/jvm/jdk1.8.0 51”
JRE_HOME=”/usr/lib/jvm/jdk1.8.0_51/jre”
```

Deve-se deslogar e logar de novo para que a inclusão da variável JAVA_HOME seja efetivada.

2.3. Instalação ONOS

O procedimento abaixo foi feito para versão 1.3.0 do ONOS, binários já compilados, com base no guia do usuário[1].

1. Fazer download da última versão:

   https://wiki.onosproject.org/display/ONOS/Downloads

2. Descompactar o arquivo:

```
$ tar xvzf onos-1.3.0.tar.gz
```

Criar link simbólico para facilitar controle de versionamento:

```
$ ln -s onos-1.3.0 onos
$ ls -la onos
lrwxrwxrwx. 1 oess wheel 10 Oct 21 09:05 onos -> onos-1.3.0
```

3. Efetuar setup de variáveis de ambiente:

PATH – adicionar ao final o caminho para o diretório com os executáveis do ONOS.

JAVA_HOME – diretório raiz do Java

```
$ vi .bash_profile
PATH=$PATH:$HOME/.local/bin:$HOME/bin:$HOME/onos/bin

export PATH

# Inicializacao ONOS
JAVA_HOME=/usr/lib/jvm/jdk1.8.0_51
```

```
export JAVA_HOME
```

Deslogar e logar novamente para poder fazer uso das variáveis de ambiente.

5. Liberar firewall:

```
$ sudo iptables -I INPUT -p tcp -m tcp --dport 8181 -j ACCEPT
$ sudo iptables -I INPUT -p tcp -m tcp --dport 6633 -j ACCEPT
$ sudo iptables -I INPUT -p tcp -m tcp --dport 6653 -j ACCEPT
$ sudo iptables -I INPUT -m state --state NEW -m tcp -p tcp --dport 1099 -j ACCEPT
$ sudo iptables -I INPUT -m state --state NEW -m tcp -p tcp --dport 44444 -j ACCEPT
```

6. Houve algumas mensagens de erro relativas ao serviço RMI – aparentemente relacionados a instalação estar sendo feito em uma máquina virtual. Os problemas foram solucionados com a edição do arquivo "org.apache.karaf.management.cfg", alterando endereço 0.0.0.0 para 127.0.0.1:

```
$ cd onos-1.3.0/apache-karaf-3.0.3/etc/
$ vi org.apache.karaf.management.cfg

rmiRegistryHost = 127.0.0.1
rmiServerHost = 127.0.0.1
```

7. Iniciar o serviço:

```
$ cd onos/bin/
$ ./onos-service
Welcome to Open Network Operating System (ONOS)!

    ____  _  _______  ____
   / __ \/ |/ / __ \/ __/
  / / / /    / / / /\ \
  \____/_/|_/\____/___/


Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown ONOS.

onos>
```

O ONOS instala 4 fluxos por padrão em cada switch que se conecta a ele:

```
SSH@NetIron MLX-4 Router#show openflow flows
Total Number of data packets sent to controller:                3412
Total Number of data bytes sent to controller  :              300669

Total Number of Flows: 4
   Total Number of Port based Flows: 0
   Total Number of L2 Generic Flows: 0
   Total Number of L3 Generic Flows: 0
   Total Number of L2+L3 Generic Flows: 4
   Total Number of L23 Generic Flows: 0

Total Number of Hardware entries for flows: 16
   Total Number of Hardware entries for Port flow: 0
   Total Number of Hardware entries for Generic flow: 16

Total Number of Openflow interfaces: 4
   Total Number of L2  interfaces: 4
   Total Number of L3  interfaces: 0
   Total Number of L23 interfaces: 0

Flow ID: 378 Priority: 5 Status: Active
```

```
   Rule:
      In Port: generic
      Ether type:  0x00000800
        Action: FORWARD
      Out Port: send to controller

Flow ID: 379 Priority: 40000 Status: Active
   Rule:
      In Port: generic
      Ether type:  0x000088cc
   Action: FORWARD
      Out Port: send to controller

Flow ID: 380 Priority: 40000 Status: Active
   Rule:
      In Port: generic
      Ether type:  0x00000806
   Action: FORWARD
      Out Port: send to controller

Flow ID: 381 Priority: 5 Status: Active
   Rule:
      In Port: generic
      Ether type:  0x00000806
   Action: FORWARD
      Out Port: send to controller

Flow ID: 382 Priority: 5 Status: Active
   Rule:
      In Port: generic
      Ether type:  0x00000800
   Action: FORWARD
      Out Port: send to controller

Flow ID: 383 Priority: 40000 Status: Active
   Rule:
      In Port: generic
      Ether type:  0x000088cc
   Action: FORWARD
      Out Port: send to controller

Flow ID: 385 Priority: 5 Status: Active
   Rule:
      In Port: generic
      Ether type:  0x00000806
   Action: FORWARD
      Out Port: send to controller

SSH@NetIron MLX-4 Router#
```

8. Instalar suporte a OpenFlow:

```
onos> feature:install onos-openflow
```

Observações finais:

- O ONOS por padrão houve a porta TCP/6633, portanto os equipamentos de rede devem estar configurados com essa porta.

- Para o ONOS detectar os switches abaixo, com versões de IronWare 5.7 e 5.8:

```
SSH@NetIron CES 2024F-4X(config)#openflow hello-reply disable
```

3. Executando ONOS

1. Listando devices conectados no controlador. No caso temos 3 devices:

```
onos> devices
id=of:0012f2f759000000, available=true, role=MASTER, type=SWITCH, mfr=Brocade Communications,
Inc, hw=Multi-Service Ironware, sw=NI 5.6, serial=None, protocol=OF 10,
channelId=190.103.187.35:7802
id=of:748ef8a348800000, available=true, role=MASTER, type=SWITCH, mfr=Brocade Communications,
Inc, hw=Multi-Service Ironware, sw=NI 5.8, serial=None, protocol=OF_10,
channelId=190.103.187.34:7801
id=of:cc4e2446d9000000, available=true, role=MASTER, type=SWITCH, mfr=Brocade Communications,
Inc, hw=Multi-Service Ironware, sw=NI 5.7, serial=None, protocol=OF 10,
channelId=190.103.187.51:7801
```

2. O ONOS instala alguns flows nos switches para que direcionem frames para o controlador:

```
onos> flows
deviceId=of:00000004968bf976, flowRuleCount=3
   id=30000d499aebb, state=ADDED, bytes=0, packets=0, duration=91, priority=40000,
tableId=DEFAULT appId=org.onosproject.provider.host
      selector=[ETH TYPE{ethType=806}]
      treatment=DefaultTrafficTreatment{immediate=[OUTPUT{port=CONTROLLER}], deferred=[],
transition=None, cleared=false}
   id=40000d67d1601, state=ADDED, bytes=0, packets=3260, duration=91, priority=40000,
tableId=DEFAULT appId=org.onosproject.provider.lldp
      selector=[ETH TYPE{ethType=88cc}]
      treatment=DefaultTrafficTreatment{immediate=[OUTPUT{port=CONTROLLER}], deferred=[],
transition=None, cleared=false}
   id=40000d67ed0f7, state=ADDED, bytes=0, packets=0, duration=91, priority=40000,
tableId=DEFAULT appId=org.onosproject.provider.lldp
      selector=[ETH TYPE{ethType=8942}]
      treatment=DefaultTrafficTreatment{immediate=[OUTPUT{port=CONTROLLER}], deferred=[],
transition=None, cleared=false}
…
onos>
```

3. Visão geral da rede:

```
onos> onos:summary
node=127.0.0.1, version=1.1.0
nodes=1, devices=2, links=0, hosts=1, SCC(s)=2, flows=6, intents=0
onos>
```

4. Listado portas com OpenFlow habilitado por dispositivo:

```
onos> onos:ports
…
id=of:001bed9e4f000000, available=true, role=MASTER, type=SWITCH, mfr=Brocade Communications,
Inc, hw=Multi-Service Ironware, sw=NI 5.6, serial=None, protocol=OF 10,
channelId=10.200.0.50:7801
  port=1, state=enabled, type=fiber, speed=1000, portName=eth1/1
  port=2, state=enabled, type=fiber, speed=1000, portName=eth1/2
  port=4, state=enabled, type=fiber, speed=1000, portName=eth1/4
  port=20, state=enabled, type=fiber, speed=1000, portName=eth1/20
```

5. Obtendo help no ONOS:

```
onos> help <comando>

onos> help onos
```

6. Instalando e desinstalando uma aplicação:

```
onos> list

onos> feature:install <aplicação>

onos> feature:uninstall <aplicação>

onos> feature:repo-refresh

onos> feature:repo-list
```

7. Parando a execução de uma aplicação:

```
onos> stop <aplicação>
```

8. Reiniciando a execução de uma aplicação:

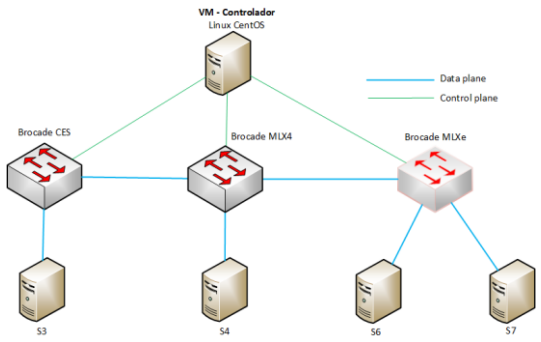
```
onos> start <aplicação>
```

9. Para obter help, executar “help onos:<comando>”:

- intents -i - lista os “intents”

4. Testes realizados

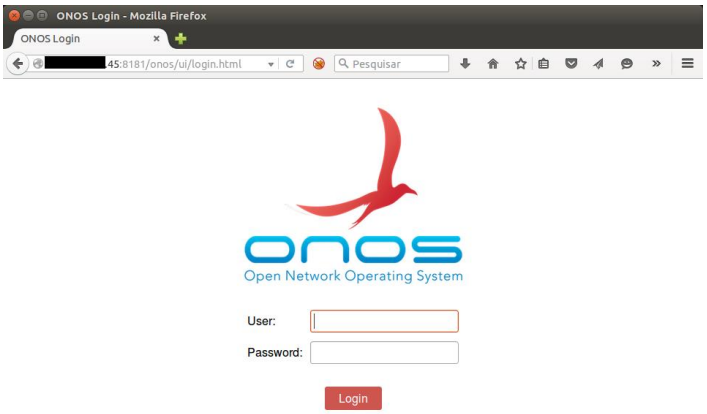
Esta seção descreve os testes realizados no laboratório do AmLight, com os equipamentos cedidos para prospecção. Optou-se pela utilização da interface gráfica do ONOS para realização dos testes.



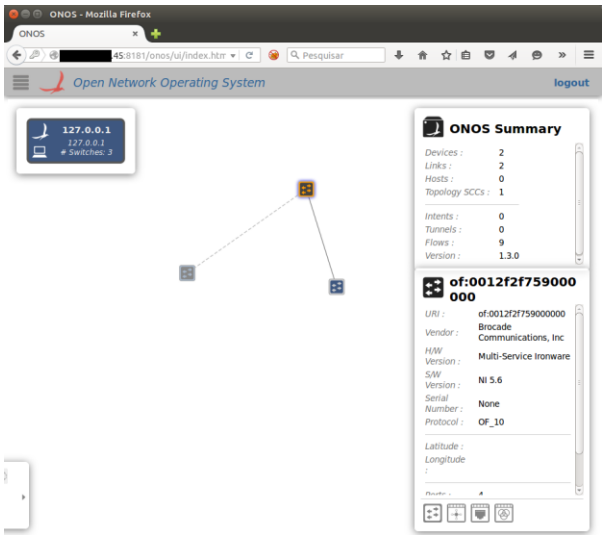
4.1. Interface Web

1. Efetuar login na interface web:

- http://X.X.X.X:8181/onos/ui/
- Usuário: onos / Senha: rocks

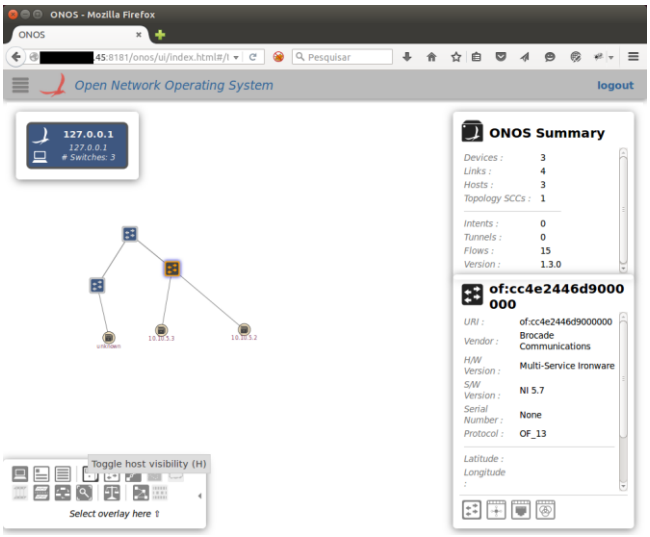


2. Após o login, o ONOS exibe o mapa da topologia detectada:

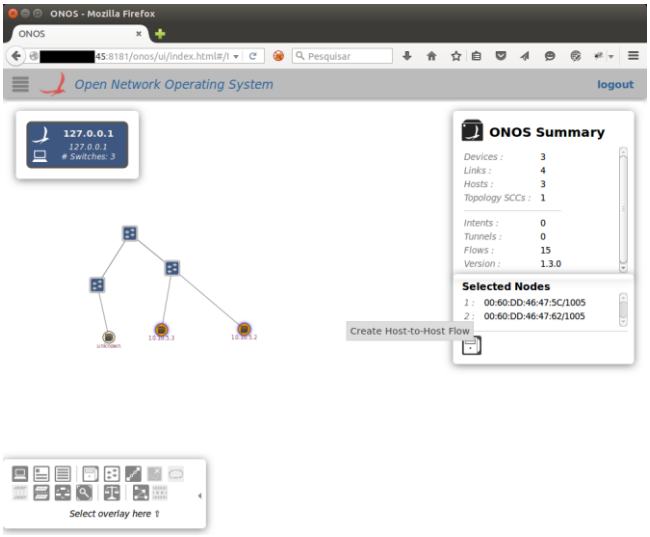


3. Executando ping das estações, permite que o ONOS detecte as estações conectas e respectivos ips – embora os pings não devam retornar resposta as estações:

```
onos> onos:hosts
id=00:0F:1F:69:97:DE/-1, mac=00:0F:1F:69:97:DE, location=of:0012f2f759000000/97, vlan=-1,
ip(s)=[10.10.20.4]
id=04:7D:7B:15:B6:B3/-1, mac=04:7D:7B:15:B6:B3, location=of:748ef8a348800000/1, vlan=-1,
ip(s)=[10.10.20.3]
onos>
```



4. Abaixo é criando uma conexão entre hosts 10.10.5.3 e 10.10.5.2:



Foi utilizado a funcionalidade “Intent” para se criar um caminho entre duas estações. A funcionalidade instalou as entradas para os MACs das estações na tabela de flows dos switches, porem a princípio o ping entre as duas estações não funcionou. Uma investigação mostrou que a causa era não haver aplicação carregada no ONOS para tratar da resolução ARP dos endereços IP das estações.

O ONOS também listou os flows no switch MLX rodando versão 1.0 de OpenFlow com estado “PENDING_ADD”. Uma verificação na tabela de flows do equipamento mostrou que os flows foram corretamente instalados. Especula-se que a causa possa estar na versão do pacote “onos-openflow-1.3.0”, que deve ser instalado para que o ONOS suporte OpenFlow. Não havia outra versão do pacote disponível na versão de ONOS instalada.

Os pings só funcionaram após ter-se adicionado manualmente as entradas ARP diretamente nas estações.

5. Referências

[1] “ONOS - A new carrier-grade SDN network operating system designed for high availability, performance, scale-out.” - http://onosproject.org/

[2] “User's Guide”, 18/03/2015, https://wiki.onosproject.org/display/ONOS/User%27s+Guide