MAC0352 - Redes de Computadores e Sistemas Distribuídos - 1s2021
EP3
Entrega até 8:00 de 26/7/2021
(INDIVIDUAL)

Prof. Daniel Macêdo Batista

1 Problema

Neste EP, você deverá implementar dois controladores SDN (*Software Defined Networking*) baseados no protocolo OpenFlow[1]. De forma resumida, o primeiro controlador vai permitir a criação de um *switch* de camada 2 (Camada de enlace) e o segundo controlador vai permitir a criação de um *firewall*. Os controladores serão escritos em Python.

Este EP é apresentado em um formato de tutorial e você **precisa** seguir os passos na ordem em que estão apresentados para que você seja capaz de entregar tudo que é solicitado. **Além de entregar os códigos dos dois controladores você precisará entregar um relatório com resultados de diversos experimentos que comprovarão o funcionamento esperado dos controladores**.

1.1 SDN

Redes Definidas por Software (SDN) são uma abordagem para redes de computadores na qual o administrador pode programar o comportamento da rede de forma dinâmica por meio de interfaces abertas. Isso é feito desacoplando o plano de dados (o envio propriamente dito dos pacotes) do plano de controle (a inteligência por trás da definição de para onde um pacote deve ir). Enquanto em uma rede tradicional um elemento como um *switch* ou um roteador precisa ser configurado individualmente com regras que definem as rotas e os filtros para os pacotes, em uma SDN é possível que esses *switches* e roteadores sejam controlados remotamente por um equipamento chamado de controlador. Com SDN há uma divisão bem clara entre os elementos das redes. Um controlador centraliza as regras do que fazer com cada pacote e envia essas regras para os *switches*[2], que por sua vez comutam de fato os pacotes de uma porta de entrada para uma porta de saída. Assim, se é necessário mudar as regras da rede por conta de um novo protocolo que passou a ser usado, isso só precisa ser feito mudando as regras no controlador, que as repassará para os *switches* e assim toda a rede já estará atualizada.

A comunicação entre o controlador e os *switches* pode usar diversos protocolos mas vamos considerar neste EP uma rede em que o protocolo utilizado é o OpenFlow. A ideia é que o controlador funcione em um computador convencional rodando GNU/Linux controlando um *switch* no qual serão conectados

[1] `https://en.wikipedia.org/wiki/OpenFlow`

[2] embora eles possam atuar como roteadores, vamos chamar esses elementos apenas de *switches* daqui em diante

diversos computadores. Para rodar o controlador, você pode usar o seu computador. Os demais computadores podem ser máquinas virtuais para que não seja necessário que você tenha que adquirir vários computadores para fazer o EP. Por fim, o ideal é que o *switch* seja de fato um switch dedicado que suporte o protocolo OpenFlow, mas para facilitar as coisas vamos virtualizar esse *switch* também usando uma implementação chamada Open vSwitch.

Com a virtualização, será construído um ambiente virtual que simula o cenário da Figura 1. Inicialmente o *switch* **s1** da figura agirá como um elemento "burro" que copiará todos os pacotes recebidos para todas as portas de saída. Aos poucos você deverá incluir "inteligência" nesse *switch* por meio do controlador **c0** para que ele chegue no cenário ideal em que os pacotes recebidos por uma porta são copiados apenas na porta na qual o destino dos pacotes está conectado. Esse será o primeiro código que você deverá entregar. Posteriormente, o *switch* **s1** deverá ser modificado pelo controlador **c0** para ser capaz de fazer filtragem de pacotes impedindo que alguns dados trafeguem na rede. Esse será o segundo código que você deverá entregar.

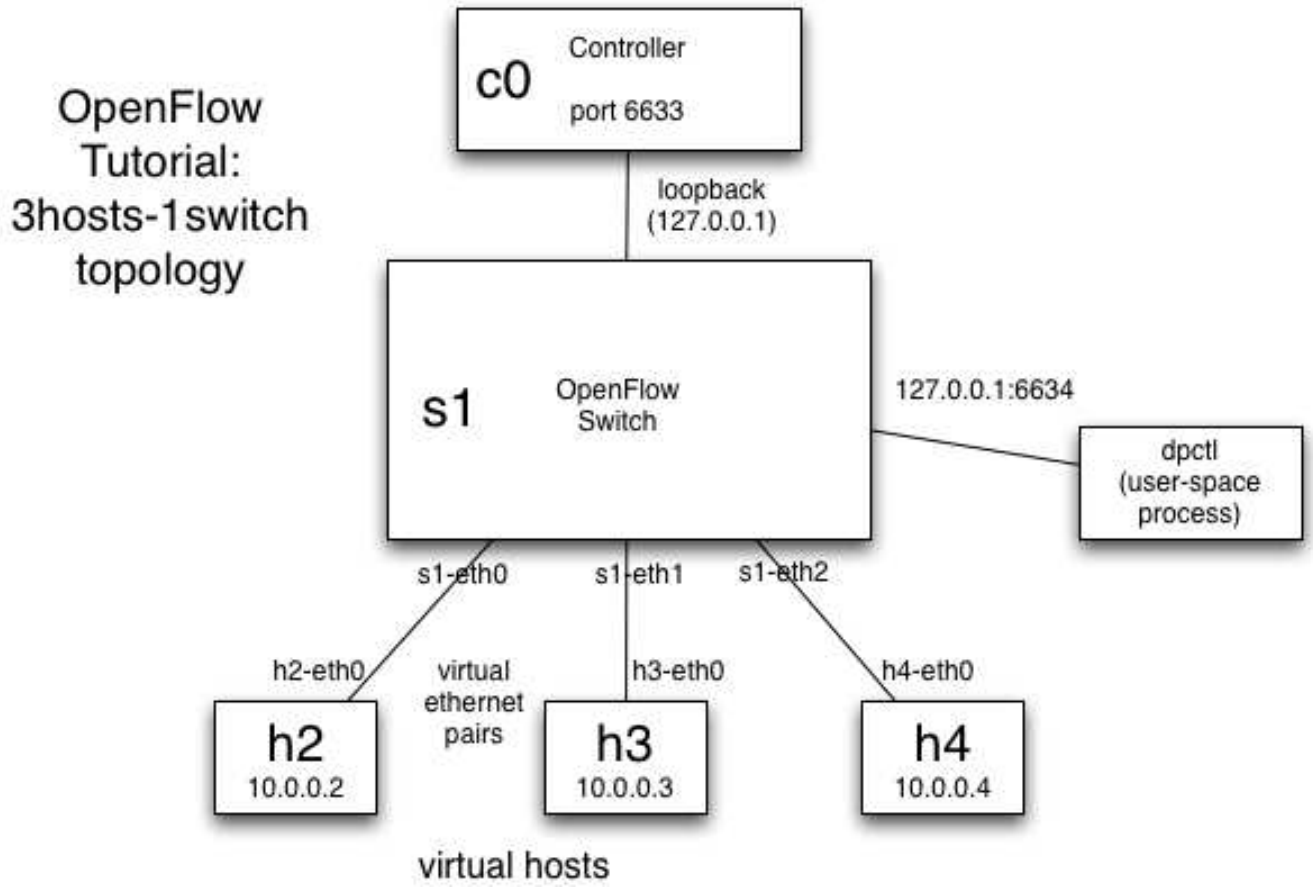

Figura 1: Topologia de 3 hosts e 1 switch simulada para o EP3 (`https://github.com/mininet/openflow-tutorial/wiki/Learn-Development-Tools`)

2 Requisitos

Como todo o EP será realizado em um ambiente virtual, diversos softwares precisarão ser instalados para que tudo funcione corretamente:

- Ubuntu: o SO sobre o qual todo o EP será realizado;
- Open vSwitch: o *switch* virtual no qual os 3 computadores virtuais serão conectados e que será controlado pelo controlador POX;
- POX: o controlador OpenFlow em Python;
- Mininet: um emulador de redes;
- Wireshark e tcpdump: *sniffers* de rede para capturar os pacotes e ver como está o funcionamento da rede;
- iperf e ping: ferramentas para comunicação entre os hosts virtuais da rede e que servirão para avaliar se o *switch* está funcionando como deveria.

Para sua sorte, já há uma Máquina Virtual (VM – *Virtual Machine*) com tudo isso instalado e funcionando corretamente. Os passos iniciais apresentados na próxima seção explicarão de onde baixar e como configurar essa VM.

3 Passo-a-passo do EP

As tarefas a serem realizadas neste EP são baseadas no tutorial *OpenFlow tutorial* disponibilizado no github do Mininet.

3.1 Passo 0

Acesse `https://github.com/mininet/openflow-tutorial/wiki` e leia a Seção *Overview* para ter uma noção do que será realizado no EP. De todos os passos apresentados na figura da Seção, **não** serão realizados os passos:

- *add multiple switch support*
- *modify switch to handle IP forwarding*
- *run switch on a real network*

Após a leitura sugerida espera-se que você tenha uma noção básica de como o OpenFlow funciona. Com essa noção básica, preencha a Seção 1 do Relatório. Também aproveite para já colocar seu nome e NUSP no cabeçalho do Relatório.

3.2 Passo 1

Acesse `https://github.com/mininet/openflow-tutorial/wiki/Installing-Required-Software`, faça o download da VM[3] que faça sentido para ser instalada em seu computador e instale o VirtualBox no seu computador para ser capaz de rodar a VM copiada.

[3] Virtual Machine Image (OVF format, 64-bit, Mininet 2.2.2) ou Virtual Machine Image (OVF format, 32-bit, Mininet 2.2.2)

3.3 Passo 2

Acesse `https://github.com/mininet/openflow-tutorial/wiki/Set-up-Virtual-Machine` e configure a máquina virtual copiada seguindo os passos do tutorial até o fim da Subseção *Command Prompt Notes*[4]. Após a leitura sugerida espera-se que você já seja capaz de acessar a VM via SSH a partir de um terminal na sua máquina real com o comando:

```
ssh -X mininet@<endereco_ip_da_maquina_virtual>
```

e usando a senha `mininet`. Uma vez estando com esse acesso habilitado, preencha a Seção 2 do Relatório[5].

3.4 Passo 3

Acesse `https://github.com/mininet/openflow-tutorial/wiki/Learn-Development-Tools`, e execute todas as etapas explicadas nesta página. Uma observação importante é que após executar o controlador com o comando `controller ptcp:6633`, você deve encerrar a execução do mininet, com o comando `quit` no prompt do mesmo, e re-executá-lo. Sem fazer isso o *switch* não se conectará no controlador. Preencha a Seção 3 do Relatório antes de executar o comando `sudo mn --topo single,3 --controller remote --switch user`. Preencha a Seção 4 do Relatório depois de executar o comando `sudo mn --topo single,3 --controller remote --switch user`.

Após a leitura sugerida espera-se que você já seja capaz de rodar softwares reais, como o iperf, o wireshark e o ping, sobre a rede emulada da Figura 1.

3.5 Passo 4

Acesse `https://github.com/mininet/openflow-tutorial/wiki/Create-a-Learning-Switch` e siga o que está explicado na Seção *Controller Choice: POX (Python)*. (Obs.: não instale a branch `betta` do POX, instale a estável). Prossiga até ler a Subseção *Benchmark Hub Controller w/iperf* inteira e preencha a Seção 5 do Relatório.

Continue a leitura da página deste passo e prossiga até ler a Subseção *of_tutorial.py Walkthrough* inteira. Neste ponto, você já deve estar com o seu novo controlador, o `switch.py` funcionando como um switch e não como um hub. Pule para a Subseção *Testing Your Controller*, leia apenas o primeiro parágrafo e preencha a Seção 6 do Relatório.

Continue a leitura da página deste passo e prossiga até o fim da Subseção *Testing Your Controller*. Neste ponto, você deve modificar o seu controlador `switch.py` para que ele seja capaz de adicionar uma regra no switch. Não apague as linhas que você teve que remover da versão anterior do switch, apenas comente elas. Preencha a Seção 7 do Relatório.

Guarde o `switch.py` pois você terá que submetê-lo dentro do .tar.gz do seu EP no e-Disciplinas.

[4]Obs: No tutorial, o nome do modo que você deve configurar a VM é *Host-only network*, mas em algumas versões do Virtualbox esse modo é chamado de *Host-only adapter*.

[5]Obs: Para preencher a Seção 2 do Relatório, você precisa de Internet. Se a VM não tem acesso à Internet, configure o modo da VM para NAT, preencha a Seção 2, e depois configure a VM para *Host-only adapter* de novo. Acesse a VM usando SSH e continue com os próximos passos.

3.6 Passo 5

Acesse `https://github.com/mininet/openflow-tutorial/wiki/Create-Firewall` e modifique o seu controlador, agora nomeado como `firewall.py` para que ele funcione como um *firewall*. Você está livre para definir como o seu firewall vai receber os parâmetros que devem ser:

- Endereço IP fonte
- Endereço IP destino
- Porta (Da camada de transporte)
- Protocolo (TCP ou UDP)

A regra padrão do *firewall* vai ser bloquear os pacotes que casem com os parâmetros passados para ele. Por exemplo, se você passar apenas o parâmetro de protocolo TCP, todos os pacotes TCP fluindo entre os 3 hosts da rede virtualizada devem ser descartados. Caso você especifique o Endereço IP fonte como sendo o IP do host 1 e o Endereço IP destino como sendo o IP do host 2, todos os pacotes entre os hosts 1 e 2 devem ser descartados. Explique como definir as regras do *firewall* no arquivo LEIAME que você tem que enviar dentro do .tar.gz do seu EP no e-Disciplinas. Preencha a Seção 8 do Relatório.

Guarde o `firewall.py` pois você terá que submetê-lo dentro do .tar.gz do seu EP no e-Disciplinas. Finalize seu Relatório preenchendo a Seção 9, informando a configuração do(s) computador(es) usado(s) e, caso você tenha usado alguma referência além dos links apresentados aqui neste enunciado, liste-as na Seção 10 do Relatório.

4 Relatório

O seu relatório deve seguir exatamente o modelo disponibilizado no e-Disciplinas em LaTeX no arquivo `mac0352-relatorio-ep3.tex`. Você precisa preencher o relatório com tudo que foi sendo pedido nos passos da Seção 3. **NÃO** adicione nenhuma nova seção ou subseção no relatório, **NÃO** remova nenhuma das seções ou subseções presentes e nem as perguntas do relatório. Se por algum acaso você não foi capaz de obter os resultados exigidos para o preenchimento de alguma seção ou subseção, deixe em branco.

5 Entrega

Você deverá entregar um arquivo .tar.gz contendo os seguintes itens:

- fonte do primeiro controlador nomeado como `switch.py`;
- fonte do segundo controlador nomeado como `firewall.py`;
- LEIAME explicando como passar as regras do `firewall.py`;
- .pdf do relatório.

O desempacotamento do arquivo .tar.gz deve produzir um diretório contendo os itens. O nome do diretório deve ser ep3-seu_nome. Por exemplo: ep3-joao_dos_santos.

A entrega do .tar.gz deve ser feita no e-Disciplinas.

O EP deve ser feito individualmente.

Obs.1: Serão descontados 2,0 pontos de EPs com arquivos que não estejam nomeados como solicitado ou que não criem o diretório com o nome correto após serem descompactados.

Obs.2: A depender da qualidade do conteúdo que for entregue, o EP pode ser considerado como não entregue, implicando em MF=0,0. Isso acontecerá por exemplo se for enviado um .tar.gz corrompido, um .pdf corrompido ou arquivos Python vazios.

Obs.3: O prazo de entrega expira às 8:00:00 do dia 26/7/2021.

6 Avaliação

70% da nota será dada pela implementação e 30% pelo relatório. Os critérios detalhados da correção serão disponibilizados apenas quando as notas forem liberadas.