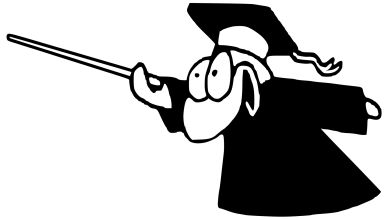
Programando um controlador

OIRC - Interconexão de redes de computadores
Prof. Dr. Ricardo José Pfitscher
ricardo.pfitscher@gmail.com

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

Objetivos de aprendizagem



- Compreender os impactos de loops nas redes
- Entender como implementar um controlador de propósito específico

Retomada

- Nas aulas anteriores nós trabalhamos com o mininet para construir topologias
- Inserimos um controlador externo nos testes automatizados
  - POX.l2_pairs
- O openflow gera diversos tipos de eventos nos controladores
  - PacketIn ➔ um pacote chegou e não foi possível fazer *match* com as regras atuais
  - ConnectionUp ➔ um novo switch está ativo na rede SDN
- O controlador deve tratar estes eventos para construir o mapa da rede e estabelecer as regras

Analisando um código POX

- Vamos observar e analisar o código do controlador l2_pairs
- Façamos alguns prints para verificar as propriedades dos objetos:

```
print "Evento:\n",event.__dict__
print "Pacote:\n",packet.__dict__
```

- Copie o código modificado para pox/ext:

```
pscp l2_pairs_mod.py mininet@IP:/home/mininet/pox/ext/
```

- Para executar o controlador modificado digite:

```
sudo ~/pox/pox.py l2_pairs_mod
```

- Para executar uma rede simples digite:

```
sudo mn --topo linear,4 --mac --switch ovsk --controller remote
```

- Faça alguns pings e verifique as saídas

Analisando um código POX

- Vamos observar e analisar o código do controlador l2_pairs
- O controlador só trata o evento packetIn
  - Ele cria uma tabela (dicionário) para mapear a tupla (conexão,mac) com a porta do switch
  - Depois ele verifica se já tem o MAC destino mapeado
    - Caso não esteja mapeado, vai fazer um *flood*
    - Caso esteja, vai instalar a regra no switch para as duas direções

Evento: connection up

- O Flood pode ser perigoso em algumas redes...
- Execute o exemplo disponível no **moodle** com o controlador modificado
  - topo_cycle.py

Evento: connection up

- O Flood pode ser perigoso em algumas redes...
- Execute o exemplo disponível no **moodle** com o controlador modificado
  - topo_cycle.py

Catástrofe, certo? A rede tem ciclo, assim, o *flood* vai fazer os pacotes ser encaminhados infinitamente...

Evento: connection up

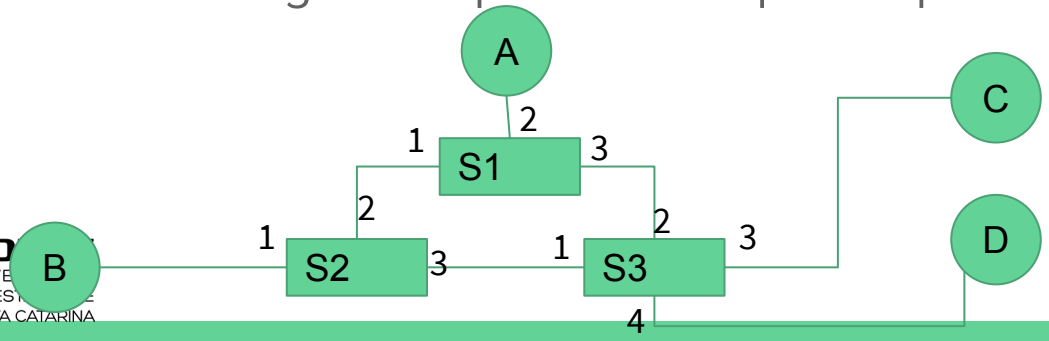
- O Flood pode ser perigoso em algumas redes...
- Execute o exemplo disponível no **Moodle** com o controlador modificado
  - topo_cycle.py

Evento: connection up

- ConnectionUp
  - Evento disparado quando um switch se conecta ao controlador
  - Você pode utilizar as informações enviadas pelo switch para gerenciar a instalação de regras na rede
  - exemplo: criar um dicionário de switches, criar uma matriz de adjacências
- Vamos utilizar o evento ConnectionUp para conhecer os nós e depois encaminhar de acordo com as políticas que queremos
  - Analise o controlador exemplo disponível no **moodle**
  - Execute esse controlador
  - Faça o pingall
  - Ping entre nós

Exercício

- Desenvolva um programa para gerar a topologia abaixo.
- Desenvolva um controlador openflow com as seguintes regras
  - Nós A, B, e C podem conversar entre si sem restrições
  - Nós D e C não podem se comunicar
    - Deve fazer um drop desses pacotes
  - O tráfego de D para B deve passar por S1
  - O tráfego de D para A deve passar por S2



Links úteis

- https://github.com/mininet/openflow-tutorial/wiki/Create-a-Learning-Switch
- http://www.brianlinkletter.com/using-the-pox-sdn-controller/
- https://openflow.stanford.edu/display/ONL/POX+Wiki
- http://flowgrammable.org/sdn/openflow/classifiers/