Universidade de Aveiro

Comunicações Móveis

Bruno Lemos 98221, Tiago Marques 98459, João Viegas 98372

6 de janeiro de 2023

Conteúdo

1 Introdução **1**

2 Tutoriais **2**
2.1 ACL . . . 2
2.2 Vlan . . . 3
2.3 Routing . . . 4
2.3.1 Routing between VLANs . . . 4
2.3.2 Static Routing . . . 4
2.4 Connection Tracking . . . 5
2.4.1 Stateful Firewall Rules . . . 5
2.4.2 Network Address Translation (NAT) . . . 6
2.5 Stacking . . . 7
2.5.1 Inter-VLAN routing . . . 7
2.5.2 Redundant stack links . . . 7
2.6 NFV Services Tutorial . . . 7
2.7 IPSEC - OVS . . . 7

3 A nossa rede **9**
3.1 Topologia . . . 10
3.2 Reproduzir a Rede . . . 11
3.2.1 Pre-Requisitos . . . 11
3.2.2 Execução . . . 11
3.3 Configurações da Rede . . . 12
3.3.1 *Stacking* . . . 12
3.3.2 Inter-VLAN Routing . . . 13
3.3.3 Teste . . . 13
3.3.4 Connection tracking - NAT . . . 14
3.3.5 Serviços NFV - DHCP . . . 16
3.3.6 Serviços NFV - DNS . . . 16

4 Grafana **19**
4.1 Integração . . . 19

5 SDN APP **21**

Lista de Figuras

2.1 Exemplos de ACLs usadas durante o tutorial 3
2.2 Representação gráfica de comunicação 4
2.3 Stateful Firewall Rules - Esquema simples 6

3.1 Topologia da nossa rede 10
3.2 Exemplo da configuração de um switch com stacking na nossa rede 12
3.3 Logs do Faucet sobre o estado das portas *stack* 12
3.4 Pacotes LLDP no wireshark 13
3.5 Atribuição dos endereços MAC e IP e criação do router 'virtual' 13
3.6 Realização de *pings* entre *hosts* de diferentes VLANs e pertencentes à mesma VLAN 13
3.7 ACL usada para implementar NAT 15
3.8 Comunicação iniciada pelo *host* h44 15
3.9 Comunicação iniciada pelo *host* h45 16
3.10 Atribuição dos endereços IP através de DHCP 16
3.11 Configuração da ACL para reencaminhar pacotes DNS para o nosso DNS/DHCP Server 17
3.12 Realização de uma *DNS Query* 18

4.1 Erro devolvido pelo grafana 20

5.1 Tentativa de conexação http com o controlador 22

Capítulo 1

Introdução

O presente relatório visa descrever o projeto de Comunicações Móveis relativo a Software Define Network. Software-Defined Networking (SDN) é uma arquitetura de rede que permite a separação do plano de controle do plano de encaminhamento de tráfego de rede. Isso permite que os administradores de rede configurem, gerenciem e monitorem a rede de forma mais eficiente e centralizada através do uso de controladores SDN.

No relatório em questão, é demonstrado como montar redes com controladores SDN para permitir uma maior flexibilidade e programabilidade na configuração das redes. Isso foi possível através da criação de regras de encaminhamento personalizadas e da implementação de mudanças de rede em tempo real.

Além disso, foi mostrado como a SDN pode levar a uma rede mais eficiente e fácil de gerenciar, permitindo uma configuração mais centralizada e rápida das redes.

Em resumo, o relatório apresenta a utilização da arquitetura SDN para montar redes com controladores e destacou os benefícios da SDN, incluindo a flexibilidade, programabilidade e eficiência na configuração e gestão de redes.

Capítulo 2

Tutoriais

Nesta parte do relatório, vamos apresentar os resultados de nossas experiências com o uso de redes definidas por software (SDN). Ao longo desta parte do trabalho, tivemos a oportunidade de realizar alguns tutoriais e seguir instruções passo a passo para configurar e gerenciar redes SDN.

Através destes tutoriais, aprendemos sobre os conceitos básicos de SDN e como essa tecnologia pode ser usada para simplificar o gestão de redes.

No final desta seção, vamos apresentar as nossas conclusões e recomendações sobre o uso de redes SDN com base na nossa experiência ao seguir os tutoriais. Esperamos que esta informação seja útil para aqueles que estão a considerar o uso de redes SDN nos seus projetos e que desejem aprender mais sobre como configurar e gerenciar essas redes.

2.1 ACL

Neste tutorial aprendemos como configurar listas de controle de acesso (ACLs) numa rede SDN usando o controlador Faucet. As ACLs são usadas para permitir ou negar o tráfego de rede com base em diferentes critérios, como endereços IP, protocolo, portas entre outros.

Com o tutorial também aprendemos a modificar os *fields* dos pacotes (macs, ips...) e fazer operações de vlans.

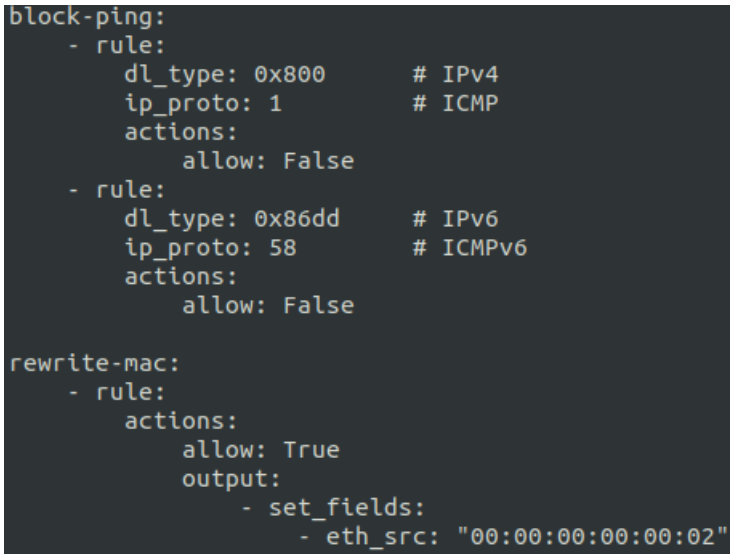


Figura 2.1: Exemplos de ACLs usadas durante o tutorial

A primeira ACL na imagem acima tem como intuito proibir o traffico ICMP e ICMPv6, ou seja, bloquear o trafico resultante de comandos como o ping, traceroute e outros.

A segunda ACL altera o endereço mac de origem para outro que no caso desta ACL seria "00:00:00:00:00:02", esta Access List pode ser especialmente util se quiser implementar algum tipo de privacidade na rede.

2.2 Vlan

As VLANs são usadas para criar múltiplas redes lógicas dentro de uma única rede física, permitindo que diferentes dispositivos ou grupos de dispositivos sejam isolados uns dos outros em camadas de rede virtuais. Conseguimos configurar VLANs básicas atribuindo dispositivos às VLANs usando tags VLAN. Também aprendemos como configurar regras de encaminhamento para permitir que os dispositivos nas VLANs se comuniquem uns com os outros e com dispositivos em outras VLANs.

No final conseguimos testar a configuração das VLANs usando o comando "ping"para enviar pacotes de teste entre dispositivos da mesma VLAN e verificar se eles são encaminhados corretamente. Para além disso, conseguimos também aplicar Access Lists (ACLs) em VLANs, invés de ser só nos portos dos switches.

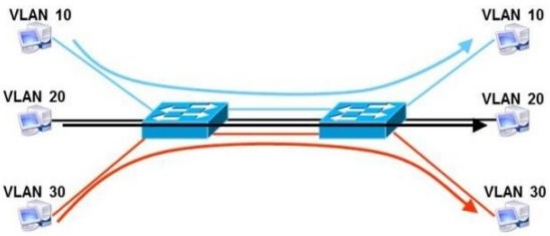


Figura 2.2: Representação gráfica de comunicação

2.3 Routing

Routing é o processo de encaminhamento de pacotes da rede de um dispositivo para outro através da rede.

No tutorial, é mostrado como configurar rotas básicas para permitir que os dispositivos da rede comuniquem uns com os outros e com dispositivos noutras redes. Também é explicado como realizar routing entre VLANs diferentes, routing estático e por fim como usar o protocolo BGP através de um serviço NFV conhecido como BRID.

2.3.1 Routing between VLANs

O faucet é usado como gateway para os nossos hosts e também usado para fazer *routeamento*. Para isso, precisamos atribuir um endereço IP e endereço MAC ao Faucet por cada VLAN que desejamos que tenha comunicação. Nos nossos *hosts* pertencentes a essas VLANs temos de lhes atribuir um *gateway* para o endereço IP daquela VLAN pertencente ao Faucet. Por fim criamos um router na configuração do Faucet e especificamos as VLANs que podem comunicar entre elas.

2.3.2 Static Routing

Introdução

O *Static Routing* é um método de configuração de rotas num sistema de *routing* que permite ao administrador de rede definir manualmente as rotas para os pacotes de dados que devem ser enviados para o seu destino final. Isso significa que, em vez de permitir que o sistema de *routing* escolha automaticamente a rota mais adequada para os pacotes de dados, o administrador de rede decide qual caminho os pacotes devem percorrer.

Um exemplo comum de utilização do*Static Routing* é quando um administrador de rede quer garantir que todos os pacotes de dados de um determinado host ou sub-rede sejam encaminhados através de uma determinada rota. Isso

pode ser útil, por exemplo, se o administrador de rede quiser garantir que todo o tráfego da Internet passe por um dispositivo de segurança, como um firewall, antes de chegar ao seu destino final.

Configuração

Para configurar o *Static Routing* é necessário seguir os passos seguintes:

1. Adicione uma rota em cada *host* para o gateway, que é o endereço IP virtual do faucet.
2. Abrir o arquivo de configuração do Faucet, que a maior parte das vezes é o arquivo faucet.yaml, e adicionar rotas estáticas nas VLAN/s que pretende que haja *routing*, através da especificação da rede destino e do *default gateway*.
3. Salvar o arquivo e reiniciar o Faucet para que as alterações entrem em vigor.

2.4 Connection Tracking

Connection Tracking é o processo de *tracking* o tráfego de rede através da rede e associá-lo a conexões lógicas entre dispositivos.

No tutorial configuramos o rastreamento de conexões básico para permitir que os dispositivos da rede comuniquem uns com os outros usando protocolos de camada de aplicação, como o TCP e o UDP. Também é explicado como usar regras de firewall para controlar o tráfego de rede baseado nas conexões rastreadas.

2.4.1 Stateful Firewall Rules

Introdução

Stateful Firewall Rules permitem que um firewall de rede mantenha o *tracking* do estado das ligações da rede que passam por ele e permita ou bloqueie o tráfego com base no estado da ligação. Estas regras são baseam-se em informações guardadas como endereços IP de origem e destino, porto e o protocolo a ser utilizado.

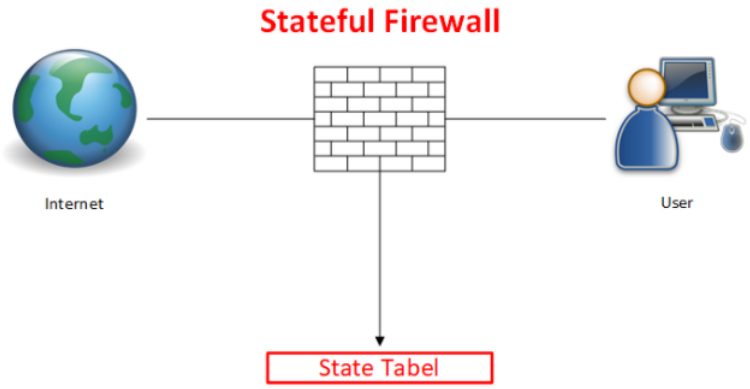


Figura 2.3: Stateful Firewall Rules - Esquema simples

Configuração

Para configurar *Stateful firewall Rules* é necessário seguir os seguintes passos:

1. Abrir o arquivo de configuração do Faucet. O arquivo de configuração geralmente é encontrado em "/etc/faucet/faucet.yaml".
2. Adicionar as regras de firewall
3. Salvar o arquivo e reiniciar o Faucet para que as alterações entrem em vigor.

2.4.2 Network Address Translation (NAT)

Introdução

Network Address Translation (NAT) é um processo que permite que um dispositivo altere o endereço IP e o número do porto de um pacote de rede enquanto é transmitido através da rede. O NAT é geralmente usado para permitir que dispositivos em uma rede privada acessem a Internet usando um único endereço IP público.

Configuração

Para configurar o NAT no Faucet é necessário seguir os passos seguintes:

1. Abra o arquivo de configuração do Faucet em um editor de texto. O arquivo de configuração geralmente é encontrado em /etc/faucet/faucet.yaml.
2. Adicionar as regras de NAT.
3. Salvar o arquivo e reiniciar o Faucet para que as alterações entrem em vigor.

2.5 Stacking

Stacking é o processo de usar várias ligações lógicas para aumentar a largura de banda e a confiabilidade de uma ligação da rede, ou seja, o controlador toma decisões de *routing* e *switching* no contexto de toda a rede e nao só a pensar num switch individualmente.

O Faucet para perceber a topologia dos links *stack* entre os switch usa o protocolo LLDP.

2.5.1 Inter-VLAN routing

Inter-VLAN routing with stacking é uma técnica usada para permitir que diferentes VLANs se comuniquem entre si através de uma *stack* de switches. Isso é útil quando é necessário dividir a rede Faucet em várias VLANs para segmentar o tráfego de rede ou aplicar políticas de segurança.

2.5.2 Redundant stack links

Redundant stack links são usados para conectar os switches Faucet em uma *stack* de maneira aumentar a disponibilidade da *stack*. Ao configurar links de *stack* de forma redundantes, é possível garantir que a *stack* continue funcionando mesmo se um link falhar.

2.6 NFV Services Tutorial

Com o tutorial "NFV Services"(Network Function Virtualization) configuramos serviços de NFV (Virtualização de Funções de Rede) numa rede SDN usando o controlador Faucet. A NFV é uma técnica que permite que funções da rede, como firewalls e balanceadores de carga, sejam implementadas como aplicações em execução em servidores virtuais em vez de equipamentos físicos dedicados.

Também configuramos serviços de NFV básicos com o Open vSwitch (OVS) para criar regras de encaminhamento que redirecionam o tráfego da rede para aplicativos de NFV em execução.

O tutorial também explica como usar *namespaces* para a configuração dos serviços de NFV e como integrar os serviços de NFV com outras funcionalidades do Faucet, como o rastreamento de conexões.

2.7 IPSEC - OVS

Neste tutorial aprendemos como configurar uma conexão de Segurança de Protocolo de Internet (IPSec) entre dois hosts usando o Open vSwitch (OVS), como configurar o OVS para suportar o IPSec e como usar a ferramenta ipsec para estabelecer uma conexão segura entre dois hosts.

Este tutorial inclui instruções detalhadas sobre como configurar um tunel IP-Sec usando certificados autoassinados, incluindo como configurar os protocolos de segurança e gerar as chaves que são usadas para proteger a conexão.

Capítulo 3

A nossa rede

Depois de realizados diversos tutoriais sobre temas relacionados com SDN's, incluindo VLANs (Virtual Local Area Networks), routing entre redes distintas, stacking de switches, de forma a usar os recursos da rede de maneira mais eficiente, e *tracking* de conexão. Neste relatório, após termos adquirido algum conhecimento destes conceitos e técnicas através dos tutoriais, montámos a nossa própria rede SDN. A seguir, apresentamos um resumo das etapas que seguimos para a criação da nossa rede.

3.1 Topologia

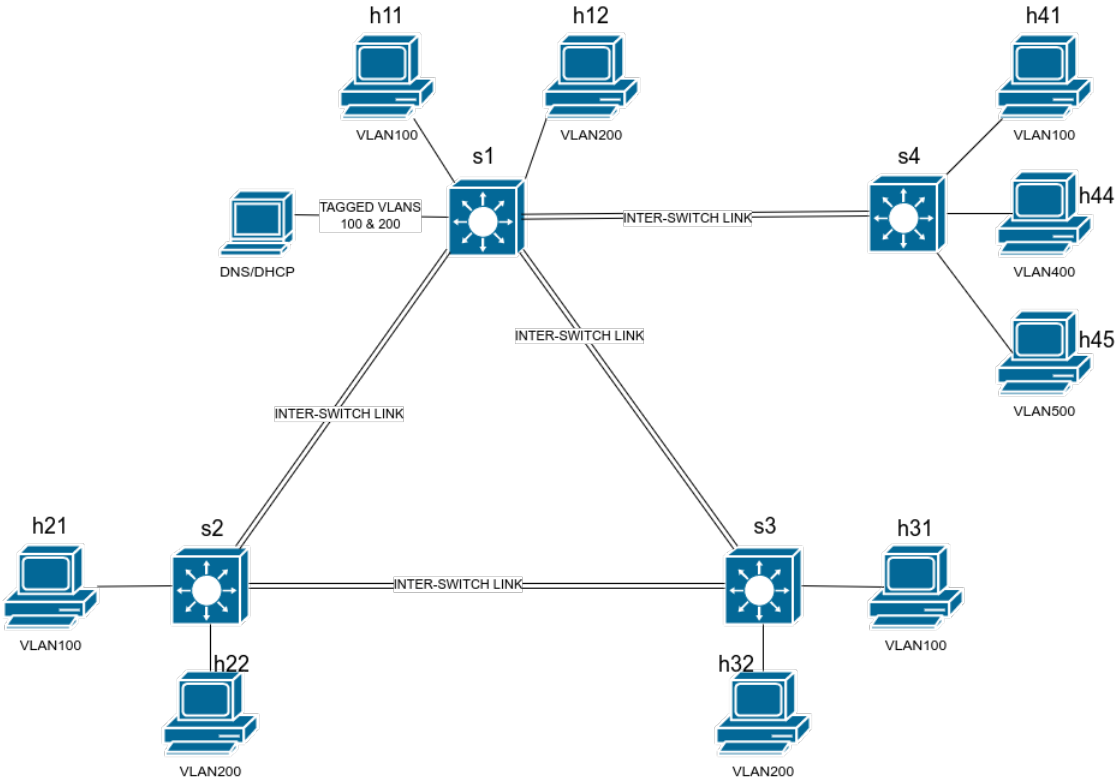


Figura 3.1: Topologia da nossa rede

Como podemos observar pela figura acima a nossa rede possui 4 VLANs (100, 200, 400, 500), as VLANs 100 e 200 serão usadas para testar routing entre VLANs e DNS, os endereços IP e Default Gateway dos hosts dessas VLANs serão configurados dinamicamente usando um serviço DHCP pertencente à nossa rede. De forma a hospedar os serviços DNS e DHCP na nossa rede teremos um host que irá servir servidor para esses serviços. Quanto as VLANs 400 e 500, estas têm como principal objetivo integrar o conceito de *connection tracking* fazendo o uso de uma ACL para implementar um mecanismo NAT.

Para o modo de operação do controlador foi escolhido o modo de 'stacking', devido a nossa rede possuir um loop entre os switches s1, s2 e s3, que permite que o controlador tomar decisões de 'switching' e 'routing' baseadas no contexto da rede permitindo assim um melhor uso dos recursos disponíveis e dando uma maior robustez à nossa rede.

3.2 Reproduzir a Rede

3.2.1 Pre-Requisitos

- Instalar Faucet container: Tutorial, secção Setting up Faucet
- Instalar Mininet: Hiperligação
- Pacote Conntrack: sudo apt-get install conntrack, usado para a implementação do mecanismo NAT
- Pacote Dnsmasq: sudo apt-get install dnsmasq, usado para implementar o serviço de DNS e DHCP

Nota: Não esquecer que passar as configurações do faucet para dentro da pasta inst no *faucet container*.

3.2.2 Execução

1. No diretório que contém os ficheiros do projeto executar o script de python faucet_mininet.py.

```
tiago@tiago-VirtualBox:~/Desktop/CM_SDN-master$ sudo python3 faucet_mininet.py
```

2. Na Mininet CLI, abrir o terminal do nosso DNS/DHCP server e configurá-lo usando o *bash script* 'conf_dns.sh'. Não esquecer de tornar o *script* executável.

3. De seguida na Mininet CLI, através do DHCP atribuimos os endereços IP e *default gateways* aos nossos *hosts* das VLANs 100 e 200.

```
mininet> source get_ip_dns
```

4. Por fim falta atribuir *default gateways* aos nossos *hosts* das VLANs 400 e 500 de forma a ser possível realizar *routing* entre as duas VLANs. *hosts* das VLANs 100 e 200.

```
mininet> source default_route
```

3.3 Configurações da Rede

3.3.1 *Stacking*

Configuração

Para ativar o *stacking* inicialmente é preciso definir um *root switch* atribuindo-lhe a menor prioridade, no caso da nossa rede é o switch s1. De seguida, é preciso definir os portos *stack* entre os routers que vão fazer parte da *stack*.

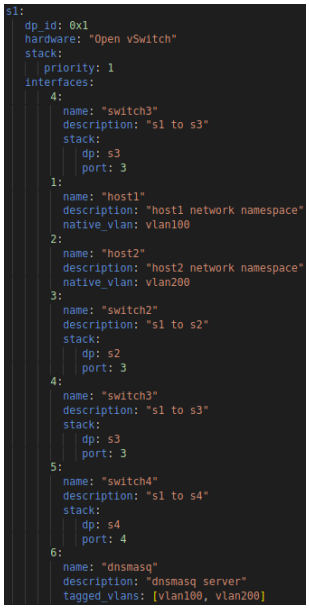


Figura 3.2: Exemplo da configuração de um switch com stacking na nossa rede

Ao carregar configuração o Faucet começará a enviar pacotes LLDP para conectar as portas *stack*. Podemos ver isso acontecer no log do Faucet quando os switches relatam o estado da porta *stack*. Também é possivel observar esses pacotes LLDP pelo Wireshark.

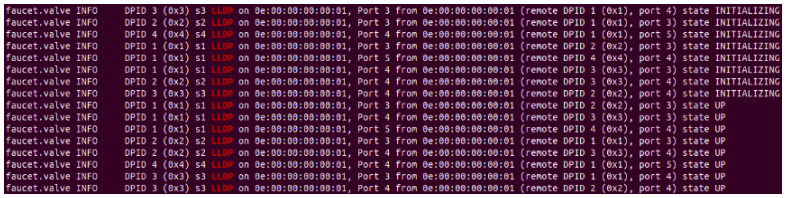


Figura 3.3: Logs do Faucet sobre o estado das portas *stack*

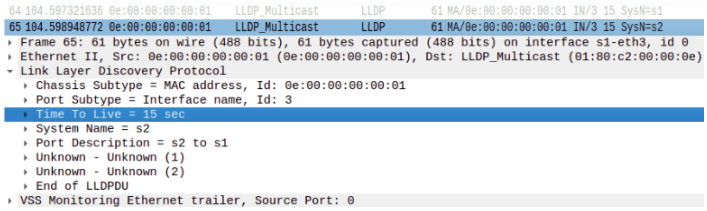


Figura 3.4: Pacotes LLDP no wireshark

3.3.2 Inter-VLAN Routing

Configuração

Para termos comunicação entre VLAN inicialmente é preciso usar o Faucet como *default gateway* para os nossos *hosts*, para isso iremos atribuir ao faucet um endereço IP e um endereço MAC por VLAN. Por fim criamos um router 'virtual' que nos permita estabelecer conexões entre as VLANs.

```
vlans:
    vlan100:
        vid: 100
        faucet_vips: ["10.0.1.254/24"]
        faucet_mac: "00:00:00:00:00:11"
        acls_in: [nfv-dns, allow-all]
    vlan200:
        vid: 200
        faucet_vips: ["10.0.2.254/24"]
        faucet_mac: "00:00:00:00:00:22"
        acls_in: [nfv-dns, allow-all]
    vlan400:
        vid: 400
        faucet_vips: ["10.0.4.254/24"]
        faucet_mac: "00:00:00:00:00:44"
    vlan500:
        vid: 500
        faucet_vips: ["10.0.5.254/24"]
        faucet_mac: "00:00:00:00:00:55"
routers:
    router-1:
        vlans: [vlan100, vlan200, vlan400, vlan500]
```

Figura 3.5: Atribuição dos endereços MAC e IP e criação do router 'virtual'

3.3.3 Teste

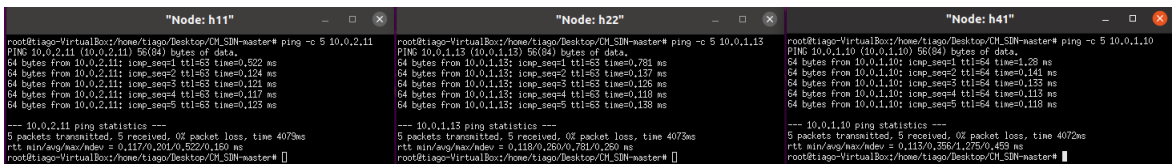


Figura 3.6: Realização de *pings* entre *hosts* de diferentes VLANs e pertencentes à mesma VLAN

Como podemos observar no imagem acima no primeiro terminal temos uma comunicação entre o *host* h11 e o *host* h22, portanto entre a VLAN100 e a VLAN200. No segundo terminal é realizada uma conexão do *host* h22 para o *host* h41, portanto uma comunicação inter-VLAN desta vez iniciada por um *host* pertencente à VLAN 200. No último terminal temos uma conexão entre *hosts* da mesma VLAN, neste caso do *host* h41 para o *host* 11.

3.3.4 Connection tracking - NAT

Configuração

A implementação do mecanismo NAT é feito através de uma ACL. Nesta ACL a primeira regra permite todo o tráfego ARP, que é usado para resolver os endereços MAC dos dispositivos na rede.

A segunda regra começa a rastrear todas as conexões IPv4 não rastreadas. Quando um pacote não rastreado é recebido, ele é reinjetado no pipeline com metadados de conexão adicionais e, em seguida, avaliado pelas Faucet Access Control Lists (ACLs) na tabela 0. O pacote original não rastreado é efetivamente descartado.

A terceira regra confirma uma nova conexão entre host44 e host45, fazendo *NATing* da conexão com o Faucet Virtual IP (VIP).

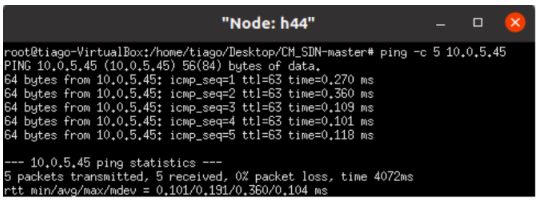
A quarta regra permite pacotes em qualquer direção de conexões existentes que foram iniciadas por *host* h44 para o *host* h45

E por fim a quinta regra bloqueia todo o tráfego IPv4 iniciado pelo *host* h45 para o *host* h44.

```
conntrack_fw:
    # Permit all ARP traffic such that hosts can resolve one another's MACs
    - rule:
        eth_type: 0x0806 #arp
        actions:
            allow: True
    # Begin tracking ALL untrackked IPv4 connections
    - rule:
        eth_type: 0x0800  # ipv4
        ct_state: 0/0x20 # match -trk (untracked)
        actions:
            # Re-inject tracked packet into the OP pipeline, containing additional connection
            # metadata, to default table 0. The tracked packet is again evaluated by Faucet ACLs
            # in table 0. The original, untracked packet is effectively dropped
            ct:
                zone: 10 # arbitrary conntrack zone ID to match agains later
                table: 0
    # Commit new ipv4 connection from host44 to host45
    - rule:
        eth_type: 0x0800 # ipv4
        ipv4_src: 10.0.4.44
        ipv4_dst: 10.0.5.45
        ct_state: 0x21/0x21 # match +new - packet to establish a new connection
        actions:
            # Commit the connection to the connection tracking module whicj will be stored
            # beyond the lifetime of packet in the pipeline
            ct:
                zone: 10 # the same conntrack zone ID as above
                flags: 1 # commit the new connection
                table: 1 # implicit allow the new connection packet via faucet table 1
                nat: #sNAT the connection to the faucet VIP
                    flags: 1
                    range_ipv4_min: 10.0.4.254
                    range_ipv4_max: 10.0.4.254
    # Allow packets in either direction from existing connections initiated by host44
    - rule:
        eth_type: 0x0800 # ipv4
        ct_zone: 10 # match packets associated with our conntrack zone ID
        ct_state: 0x22/0x22 # match +est - packets in an established connection
        actions:
            ct:
                zone: 10
                flags: 1
                table: 1
                nat:
                    flags: 1
    # Block all ipv4 traffic initiated by host45 to host44
    - rule:
        eth_type: 0x0800 # ipv4
        ipv4_src: 10.0.5.45
        ipv4_dst: 10.0.4.44
        actions:
            allow: False
```

Figura 3.7: ACL usada para implementar NAT

Testes



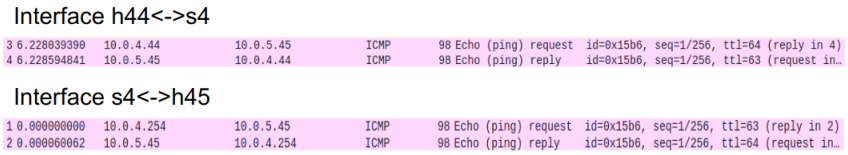


Figura 3.8: Comunicação iniciada pelo *host* h44

Neste teste foi realizado um ping entre o *host* h44 para o *host* h45, como podemos observar o trafego ao passar do switch s4 para o *host* h45 o endereço IP de origem

é alterado para o endereço IP do Faucet Virtual IP da VLAN 400.

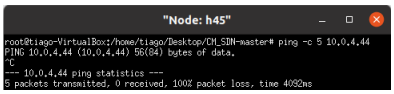


Figura 3.9: Comunicação iniciada pelo *host* h45

Neste segundo teste o conexão é iniciada pelo *host* 45 e como estabelecido na nossa ACL o trafego é bloqueado pelo que a nossa ACL está a funcionar corretamente.

3.3.5 Serviços NFV - DHCP

Configuração

Duas instâncias dnsmasq serão criadas dentro do *namespace* do nosso *host* DNS/DHCP. Uma instância irá servir *hosts* na VLAN 100 e a outra na VLAN 200. Forneceremos concessões de DHCP nos intervalos fornecidos, para além de atribuirmos um endereço IP também atribuimos a *gateway* para cada *host* apontar para o Faucet Virtual IP da sua VLAN correspondente.

```
mininet> source get_ip_dns
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/h11-eth1/00:00:00:00:00:02
Sending on   LPF/h11-eth1/00:00:00:00:00:02
Sending on   Socket/fallback
DHCPDISCOVER on h11-eth1 to 255.255.255.255 port 67 interval 3 (xid=0x49976261)
DHCPDISCOVER on h11-eth1 to 255.255.255.255 port 67 interval 6 (xid=0x49976261)
DHCPOFFER of 10.0.1.10 from 10.0.1.1
DHCPREQUEST for 10.0.1.10 on h11-eth1 to 255.255.255.255 port 67 (xid=0x61629749)
DHCPACK of 10.0.1.10 from 10.0.1.1 (xid=0x49976261)
bound to 10.0.1.10 -- renewal in 1699 seconds.
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/h12-eth1/00:00:00:00:00:03
Sending on   LPF/h12-eth1/00:00:00:00:00:03
Sending on   Socket/fallback
DHCPDISCOVER on h12-eth1 to 255.255.255.255 port 67 interval 3 (xid=0xfe6e506e)
DHCPOFFER of 10.0.2.10 from 10.0.2.1
DHCPREQUEST for 10.0.2.10 on h12-eth1 to 255.255.255.255 port 67 (xid=0x6e506efe)
DHCPACK of 10.0.2.10 from 10.0.2.1 (xid=0xfe6e506e)
bound to 10.0.2.10 -- renewal in 1557 seconds.
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
```

Figura 3.10: Atribuição dos endereços IP através de DHCP

3.3.6 Serviços NFV - DNS

Configuração

Nas duas instâncias dnsmasq criadas anteriormente no *namespace* do nosso *host* DNS/DHCP também foi configurado para resolver nomes de domínio, neste

caso para o nome "does.it.work", para os seus endereços IP correspondentes. De seguida vamos usar uma ACL de forma a reencaminhar pacotes DNS para o nosso *host* DNS/DHCP para permitir que nosso DNS/DHCP responda a consultas DNS para qualquer endereço IP pertencente as VLANs 100 e 200.

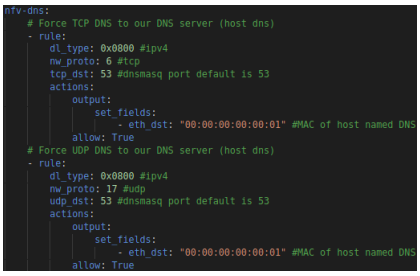


Figura 3.11: Configuração da ACL para reencaminhar pacotes DNS para o nosso DNS/DHCP Server

Nesta ACL a primeira regra força todo o tráfego DNS UDP para o servidor DNS combinando pacotes com IPv4, um protocolo UDP e uma porta de destino de 53 (a porta padrão para DNS) e reescrevendo o endereço MAC de destino do pacote para o endereço MAC do nosso *host* DNS/DHCP p que permite que o pacote seja reencaminhado.

A segunda regra força todo o tráfego DNS TCP para o servidor DNS combinando pacotes com IPv4, um protocolo TCP e uma porta de destino de 53 (a porta padrão para DNS) e reescrevendo o endereço MAC de destino do pacote para o endereço MAC do nosso *host* DNS/DHCP p que permite que o pacote seja reencaminhado.

O Conjunto destas regras garantem que todas as solicitações de DNS sejam enviadas ao servidor DNS especificado, o que faz com que o nosso *host* DNS/DHCP fique responsável por responder a todo o tráfego DNS vindo das VLAN 100 e 200.

Por fim é necessário configurar o *host* DNS/DHCP para poder lidar com os pacotes DNS recebidos com IPs pertencentes às VLANs 100 ou 200, isso pode ser feito adicionando algumas regras ao *iptables* que farão o NAT de todo o tráfego DNS para o endereço IP da interface VLAN correspondente.

Testes

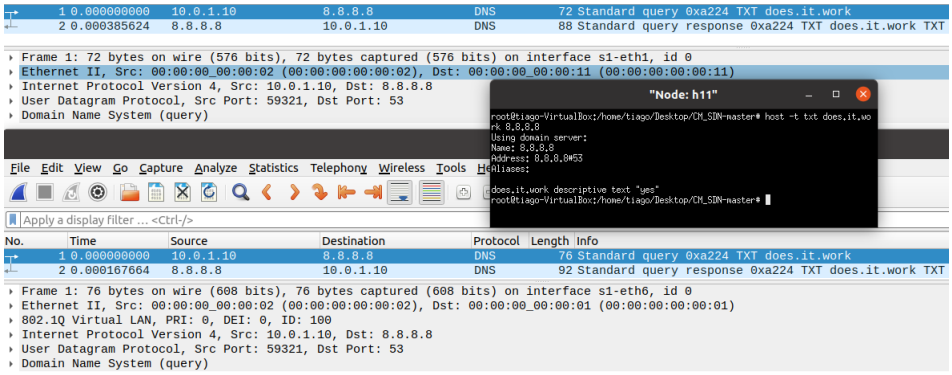


Figura 3.12: Realização de uma *DNS Query*

O *host* h11 faz uma *DNS Query* para o *domain name* "does.it.work"para o endereço IP 8.8.8.8 do suposto DNS server, endereço IP este que nem existe na rede. O pacote dns resultante dessa *query*, como podemos ver na imagem, ao chegar ao switch devido à nossa ACL é alterado, modificando o campo do endereço do MAC destino para o endereço MAC do nosso *host* DNS/DHCP "00:00:00:00:00:01"pelo que o pacote é reencaminhado para ele. Ainda como a ligação entre o switch s1 e o *host* DNS/DHCP é uma ligação *tagged* o switch acaba por adicionar uma VLAN tag com o ID 100 para que o pacote além de ser reencaminhado para o *host* DNS/DHCP será também reencaminhado para a interface responsável pela VLAN 100. Por fim, olhando para o terminal na imagem podemos verificar que a *DNS query* teve sucesso.

Capítulo 4

Grafana

Grafana é uma ferramenta de visualização de dados open-source que pode ser usada para exibir gráficos e dashboards com os dados de uma rede numa arquitetura de Software Defined Networking (SDN). Permite aos administradores de rede visualizar de forma gráfica os dados de tráfego de rede, erros, latências e outras métricas, o que pode ser útil para detetar problemas e otimizar o desempenho da rede.

O Grafana pode ser integrado com uma ampla variedade de fontes de dados, como sistemas de gestão de redes, bancos de dados de métricas de rede e aplicações de SDN. Fornece uma interface de usuário intuitiva que permite aos usuários criar e personalizar dashboards com os gráficos e métricas desejados. Além disso, o Grafana oferece recursos avançados de alertas, que permitem aos utilizadores definir regras para gerar alertas quando determinadas condições de rede são atendidas.

4.1 Integração

No âmbito da construção da nossa rede nós pretendiamos integrar o grafana para fazer o controlo de tráfego da nossa rede entre vlans, com o dhcp e com o DNS. Ápos algumas tentativas a tentar integrar este programa nas nossas redes, não conseguimos extrair quaisquer dados deste programa.

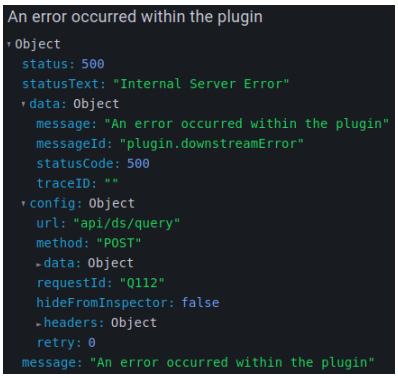


Figura 4.1: Erro devolvido pelo grafana

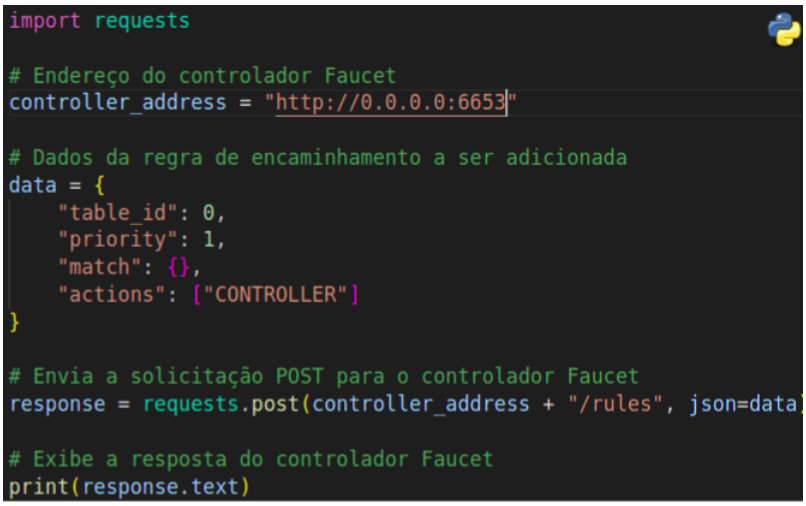
Capítulo 5

SDN APP

Por fim, para tentar tornar o nosso trabalho mais fácil de utilizar pensámos em criar uma sdn app. Esta app funcionaria como uma espécie de uma admin page, na qual o programador poderia simplesmente enviar algumas configurações para o controlador. Estas configurações poderm ser por exemplo:

- Adicionar novos hosts
- Novas configurações
- Editar algumas configurações que não estejam como o utilizador deseja
- Eliminar algumas configurações indesejadas

Infelizmente, depois de termos montado algumas redes com o controlador(Faucet) tentamos, do lado de fora do controlador, comunicar enviando pedidos http. Após várias tentativas de tentar enviar informação para a tabela de rules do controlador não conseguimos editar a tabela. Desta forma achamos que ficou algo por fazer apesar de não haver qualquer formma de resolver o problema de comunicação que tivemos.



```python
import requests

# Endereço do controlador Faucet
controller_address = "http://0.0.0.0:6653"

# Dados da regra de encaminhamento a ser adicionada
data = {
    "table_id": 0,
    "priority": 1,
    "match": {},
    "actions": ["CONTROLLER"]
}

# Envia a solicitação POST para o controlador Faucet
response = requests.post(controller_address + "/rules", json=data)

# Exibe a resposta do controlador Faucet
print(response.text)
```

Figura 5.1: Tentativa de conexação http com o controlador

Contribuições dos autores

Todos os autores participaram de forma igual na divisão, desenvolvimento e discussão deste trabalho pelo que a percentagem de contribuição para cada aluno fica:

- Bruno Lemos - 33.3 %
- Tiago Marques - 33.3 %
- João Viegas - 33.3 %