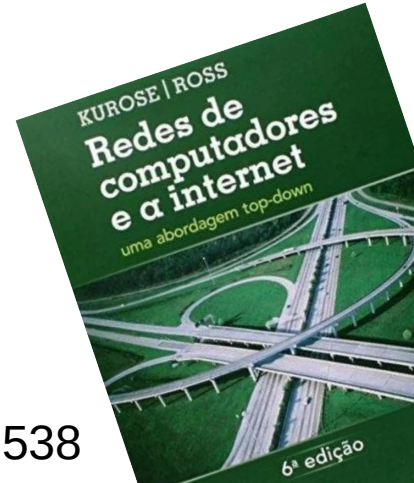
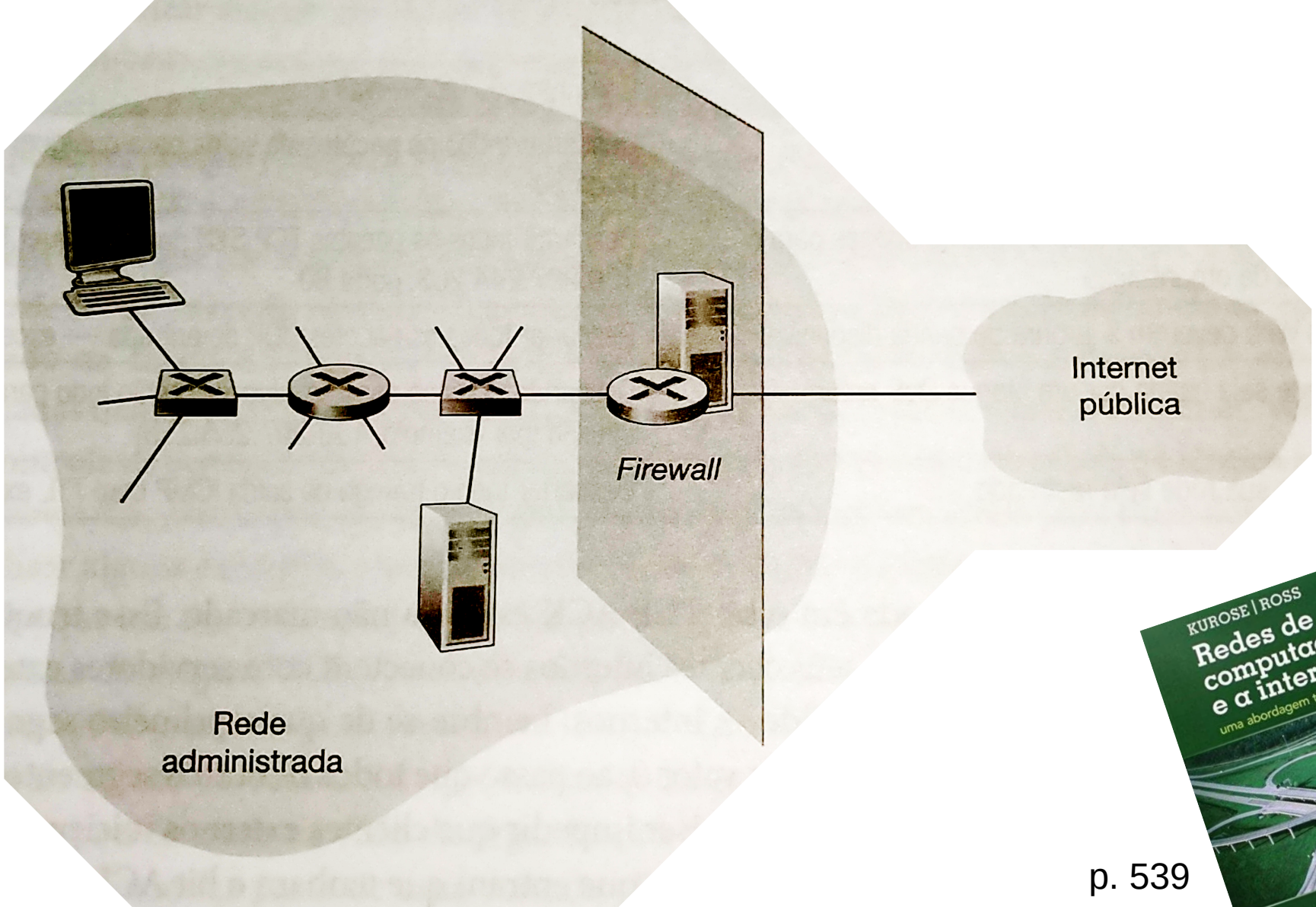
Firewall

com OpenFlow

“Um *firewall* é uma combinação de hardware e software que isola a rede interna de uma organização da Internet em geral, permitindo que alguns pacotes passem e bloqueando outros.”



p. 538


Internet
pública
Firewall
Rede
administrada

p. 539

KUROSE | ROSS
Redes de computadores e a internet
uma abordagem top-down
6ª edição

O que vem em `ryu.app.rest_firewall`?

- vLAN;
- Regras: `<MACs, MACd, P3, IPs/CIDR, IPd/CIDR, P4, Ps, Pd, Act>`
  - CRUD.
- Estatísticas para cada regra;

- Proativo: as regras devem ser adicionadas antes do fluxo de dados.

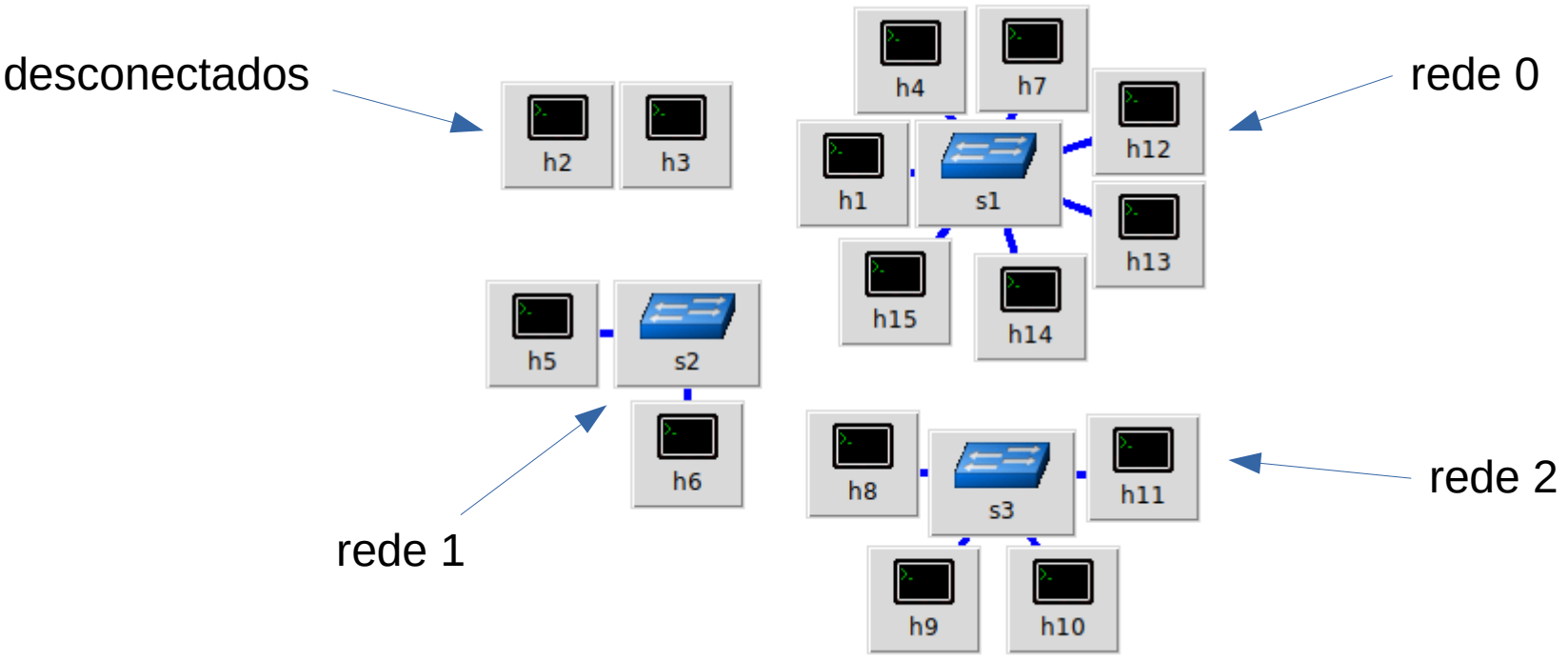
O que vem em `ryu.app.rest_firewall`?

- vLAN;
- Regras: `<MACs, MACd, P3, IPs/CIDR, IPd/CIDR, P4, Ps, Pd, Act>`
  - CRUD.
- Estatísticas para cada regra;

- Proativo: as regras devem ser adicionadas antes do fluxo de dados.

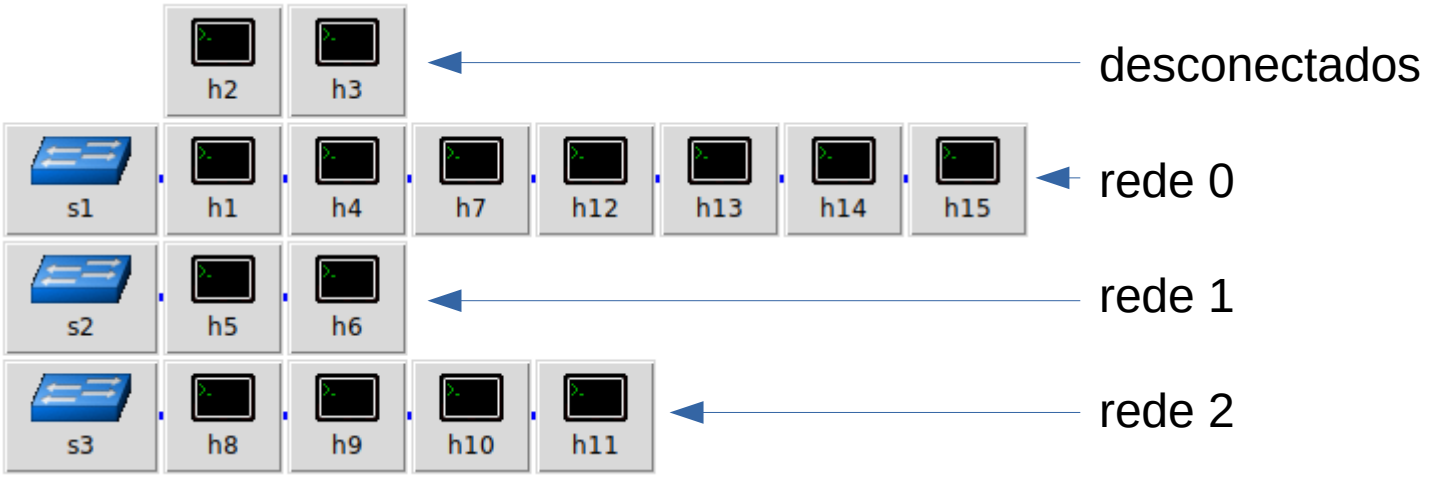
Mas o que é vLAN?

vLAN



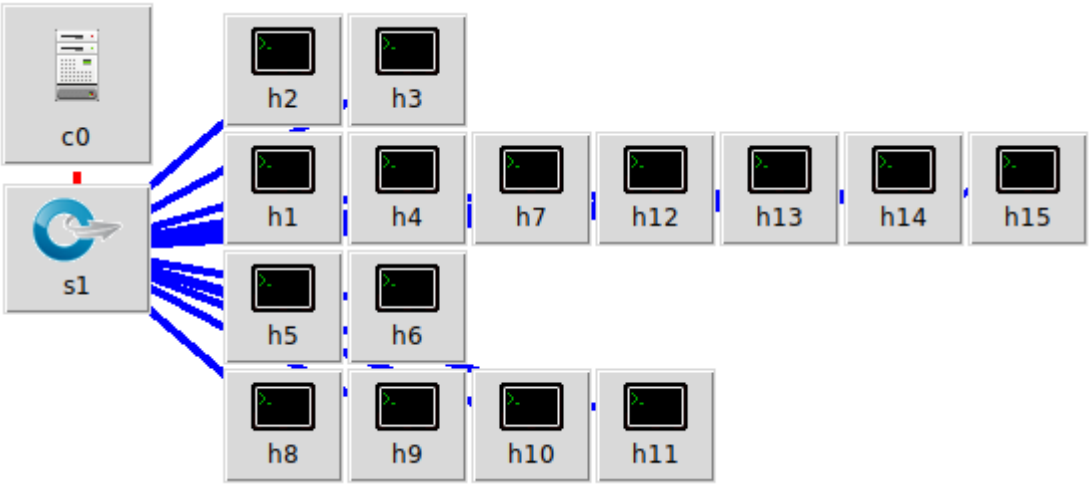
Como a rede se enxerga

vLAN



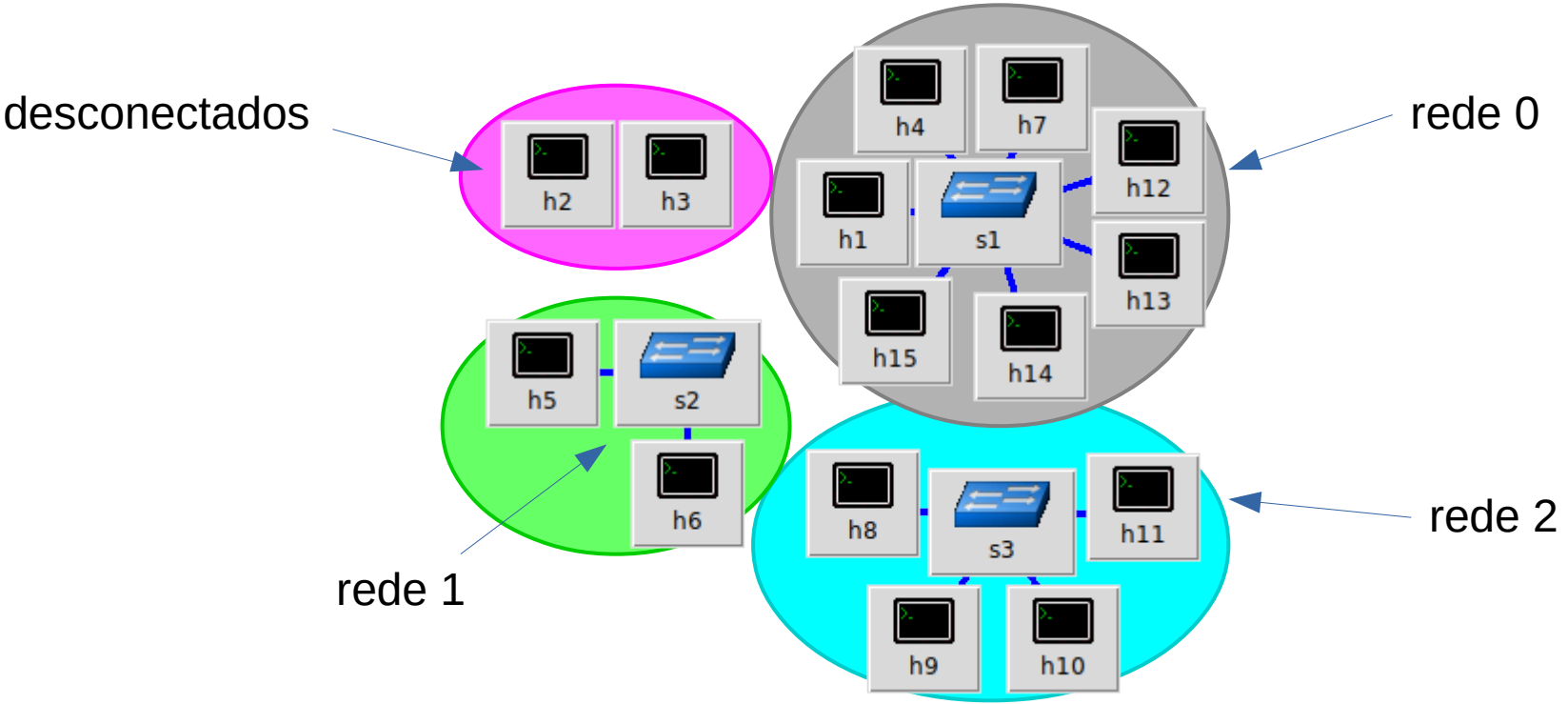
Como a rede se enxerga

vLAN



Como a rede é

vLAN



Como a rede se enxerga

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	-			✓			✓					✓	✓	✓	✓
2		-													
3			-												
4	✓			-			✓					✓	✓	✓	✓
5					-	✓									
6					✓	-									
7	✓			✓			-					✓	✓	✓	✓
8								-	✓	✓	✓				
9								✓	-	✓	✓				
10								✓	✓	-	✓				
11								✓	✓	✓	-				
12	✓			✓			✓					-	✓	✓	✓
13	✓			✓			✓					✓	-	✓	✓
14	✓			✓			✓					✓	✓	-	✓
15	✓			✓			✓					✓	✓	✓	-

O que vem em `ryu.app.rest_firewall`?

- vLAN;
- Regras: `<MACs, MACd, P3, IPs/CIDR, IPd/CIDR, P4, Ps, Pd, Act>`
  - CRUD.
- Estatísticas para cada regra;

- Proativo: as regras devem ser adicionadas antes do fluxo de dados.

O que vem em `ryu.app.rest_firewall`?

- vLAN;
- Regras: `<MACs, MACd, P3, IPs/CIDR, IPd/CIDR, P4, Ps, Pd, Act>`
  - CRUD.
- Estatísticas para cada regra;

- Proativo: as regras devem ser adicionadas antes do fluxo de dados.

O que vem em `ryu.app.rest_firewall`?

- vLAN;
- Regras: `<MACs, MACd, P3, IPs/CIDR, IPd/CIDR, P4, Ps, Pd, Act>`
  - CRUD.
- Estatísticas para cada regra;

- Proativo: as regras devem ser adicionadas antes do fluxo de dados.

E se eu quisesse um firewall reativo?

E se eu quisesse um firewall reativo?

O que é ser reativo?

Proativo × Reativo

- Adiciona regras na flow table **antes do fluxo existir**;
- Mudanças são feitas **diretamente** na flow table.

- Adiciona regras na flow table **à medida que os fluxos chegam**;
- Mudanças **refletem** na flow table como **consequência** de critérios externos ao controlador.

E se eu quisesse um firewall reativo?

Teria de implementá-lo

Implementando um firewall reativo

Começando de simple_switch_13.py

```
72
73      def actions_for_new_flow(self, event, message, datapath, ofproto, eth, parser, allow_action, disallow_action=[]):
74          self.load_rules()
75          default_action = allow_action if self.rules.get(
76              'default_action', True) else disallow_action
77          protocols = list(packet.Packet(message.data).protocols)
78          ipvx = [protocol for protocol in protocols if protocol.__class__.__name__ in [
79              'ipv4', 'ipv6']]
80          ipvx = ipvx[0] if len(ipvx) > 0 else None
81          tcudp = [protocol for protocol in protocols if protocol.__class__.__name__ in [
82              'tcp', 'udp']]
83          tcudp = tcudp[0] if len(tcudp) > 0 else None
84          vlanid_src = self.get_vlanid(
85              eth.src, ipvx.src if ipvx is not None else None)
86          vlanid_dst = self.get_vlanid(
87              eth.dst, ipvx.dst if ipvx is not None else None)
88          # devices banned from the network
89          if vlanid_src < 0 or vlanid_dst < 0:
90              return (disallow_action,)
91          # different vlans
92          if vlanid_src != vlanid_dst:
93              return (disallow_action,)
94          # special protocol on first flow
95          for rule in self.rules.get('special_protocols', list()):
96              match_action = allow_action if rule.get(
97                  'action', True) else disallow_action
98              if rule['protocol'].lower() in [protocol.__class__.__name__.lower() for protocol in protocols]:
99                  return (match_action, False)
100         # special tcp or udp port on first flow
101         if tcudp is not None:
102             for rule in self.rules.get('special_ports', list()):
103                 match_action = allow_action if rule.get(
104                     'action', True) else disallow_action
105             ports = [*([tcudp.src_port] if rule.get('src_matters') else []),
106                      *([tcudp.dst_port] if rule.get('dst_matters') else [])]
107             matchrange = list(
108                 range(rule.get('portrange_start', 0), rule.get('portrange_stop', 65535)+1))
109             for port in ports:
110                 if port in matchrange:
111                     return (match_action, False)
112         return (default_action,)
113
```

function(pacote, …)

Ação para o fluxo

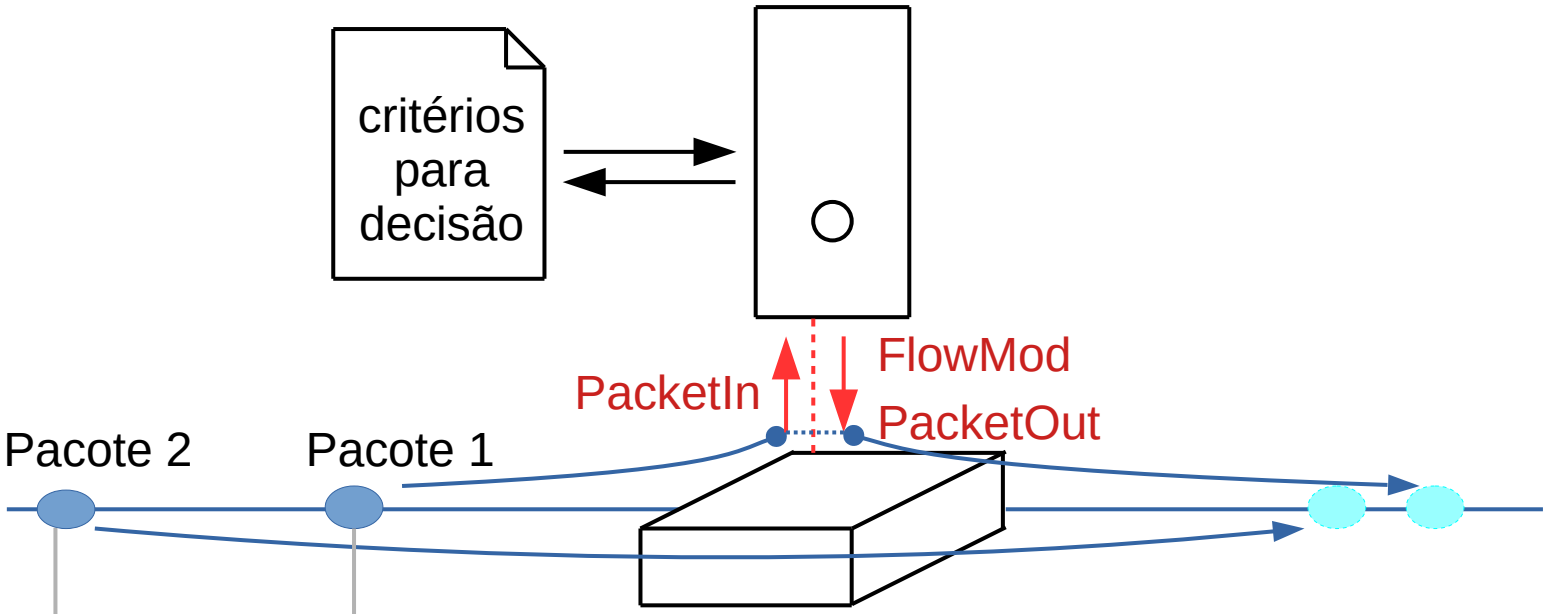
...e se tal ação deve ser salva na flow table

```
501
502 class SimpleSwitch13(app_manager.RyuApp):
503     OFP_VERSIONS = [ofproto_v1_3.OFP_VERSION]
504
505     def __init__(self, *args, **kwargs):
506         super(SimpleSwitch13, self).__init__(*args, **kwargs)
507         self.mac_to_port = {}
508         self.firewalls = dict()
509
510     @set_ev_cls(ofp_event.EventOFPSwitchFeatures, CONFIG_DISPATCHER)
511     def switch_features_handler(self, ev):
512         datapath = ev.msg.datapath
513         ofproto = datapath.ofproto
514         parser = datapath.ofproto_parser
515
```

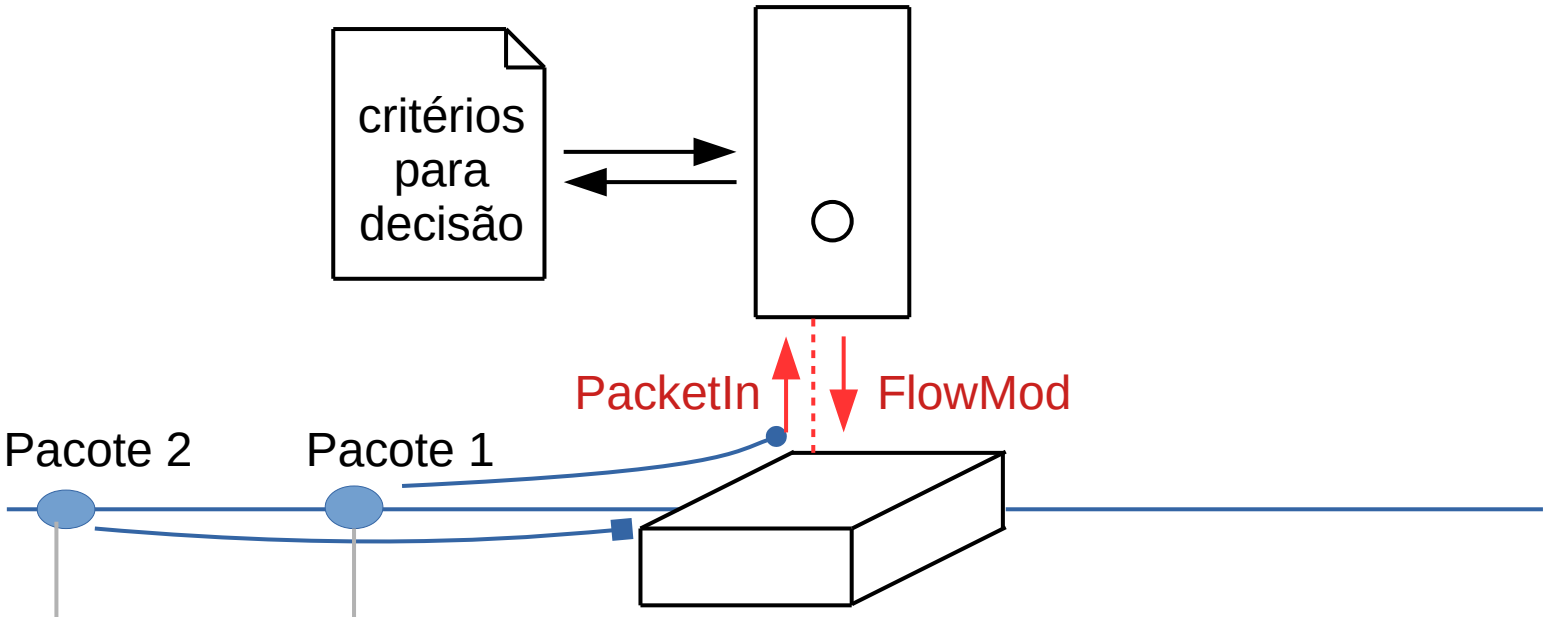
```
577
578         actions = [parser.OFPActionOutput(out_port)]
579
580         # run firewall on it
581         if datapath.id not in self.firewalls:
582             self.firewalls[datapath.id] = Firewall(datapath)
583         actions, save_in_flow_table, *_ = [
584             *list(self.firewalls[datapath.id].actions_for_new_flow(
585                 event=ev,
586                 message=msg,
587                 datapath=datapath,
588                 ofproto=ofproto,
589                 eth=eth,
590                 parser=parser,
591                 allow_action=actions
592             )),
593             True
594         ]
```

```
596         if save_in_flow_table:
597             # install a flow to avoid packet_in next time
598             if out_port != ofproto.OFPP_FLOOD:
599                 match = parser.OFPMatch(in_port=in_port, eth_dst=dst)
600                 # verify if we have a valid buffer_id, if yes avoid to send both
601                 # flow_mod & packet_out
602                 if msg.buffer_id != ofproto.OFP_NO_BUFFER:
603                     self.add_flow(datapath, 1, match, actions, msg.buffer_id)
604                     return
605                 else:
606                     self.add_flow(datapath, 1, match, actions)
```

E como ele decide?

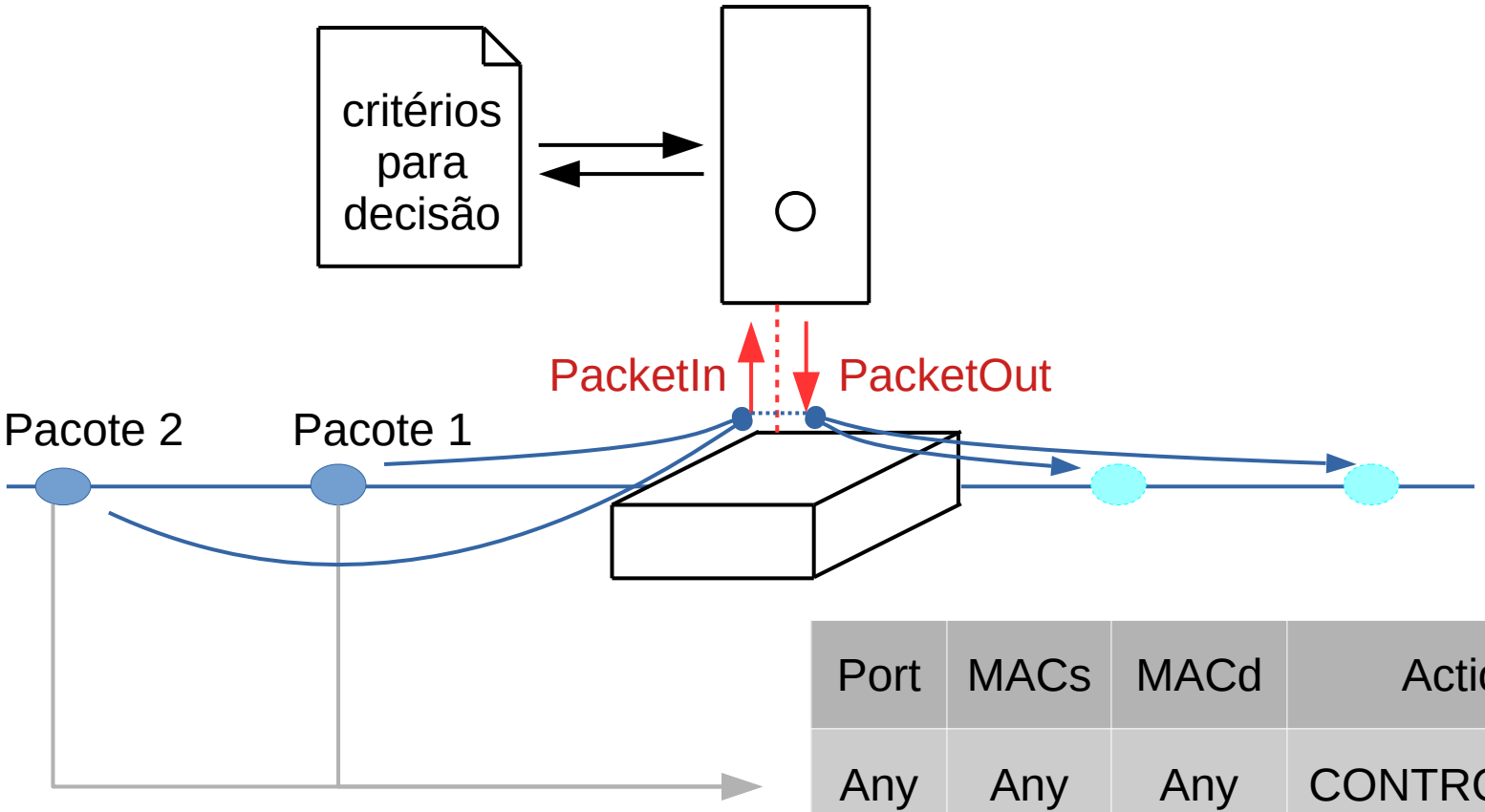


Port	MACs	MACd	Action
Any	Any	Any	CONTROLLER
1	::01	::02	PacketOut(2)



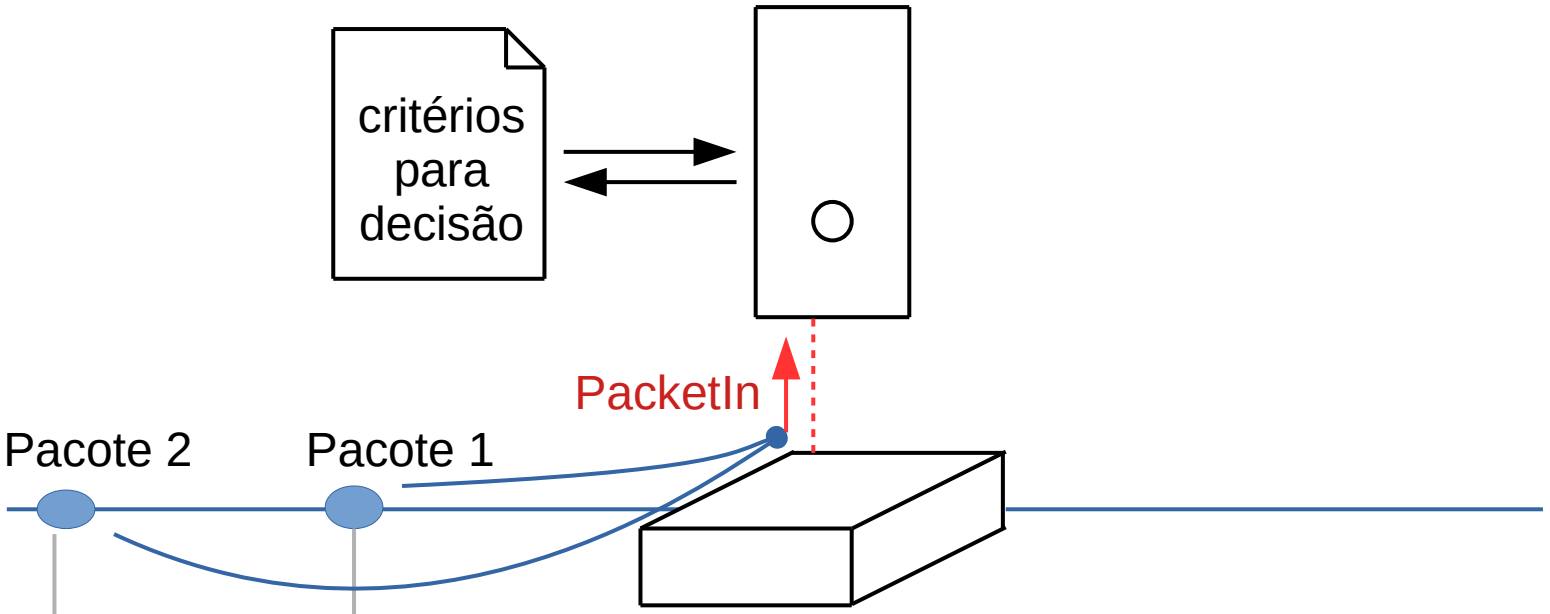
Port	MACs	MACd	Action
Any	Any	Any	CONTROLLER
1	::01	::02	DROP

Então ele *sempre* adiciona à flow table?



Port	MACs	MACd	Action
Any	Any	Any	CONTROLLER

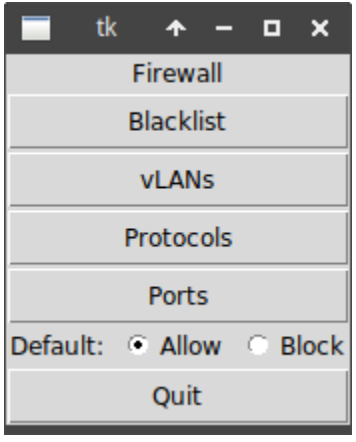
Problemas de performance!

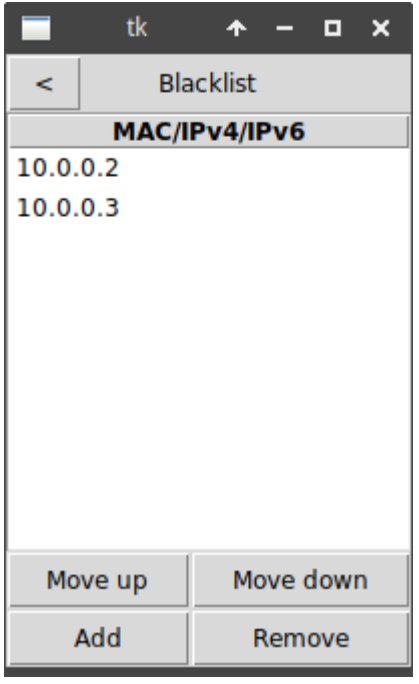


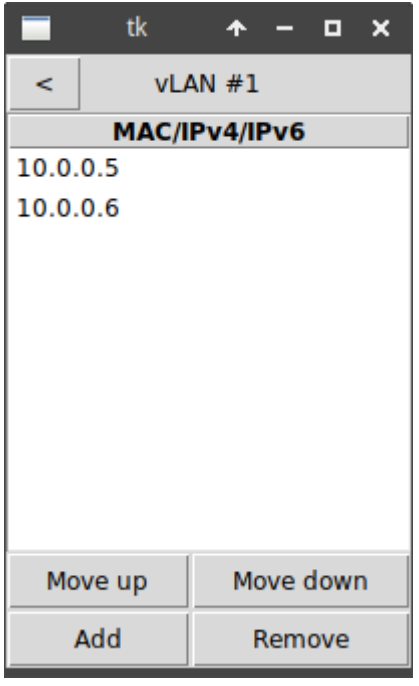
Port	MACs	MACd	Action
Any	Any	Any	CONTROLLER

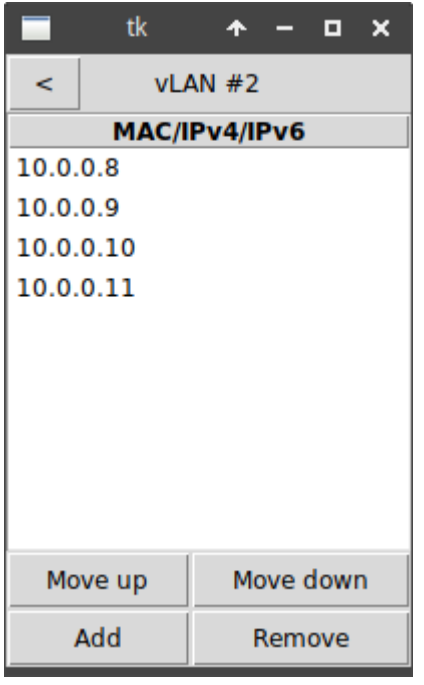
Problemas de performance!

Como os critérios foram adicionados

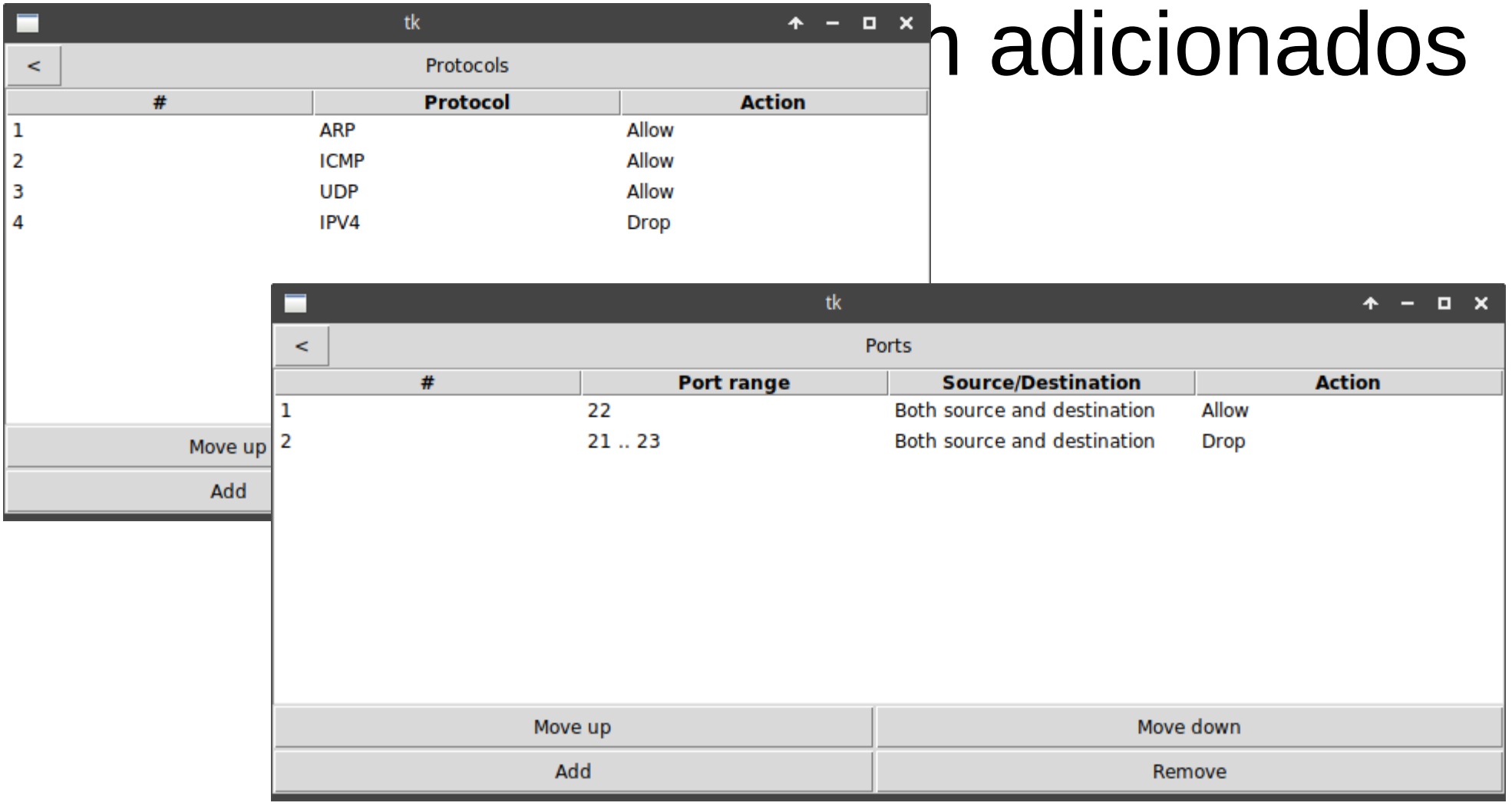








n adicionados



tk

< Blacklist

MAC/IPv4/IPv6
10.0.0.2
10.0.0.3

tk

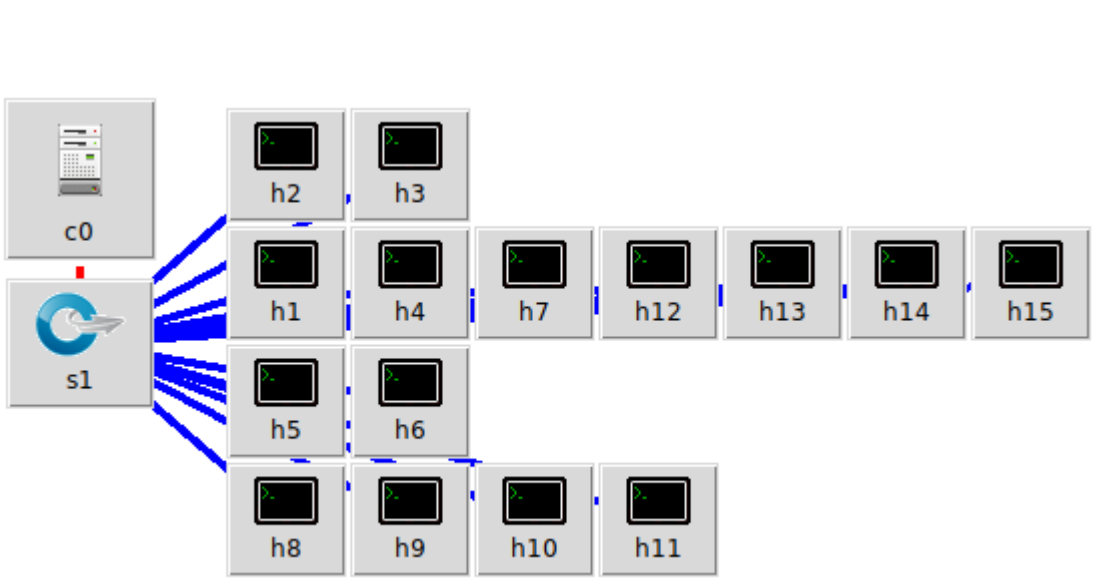
< vLAN #1

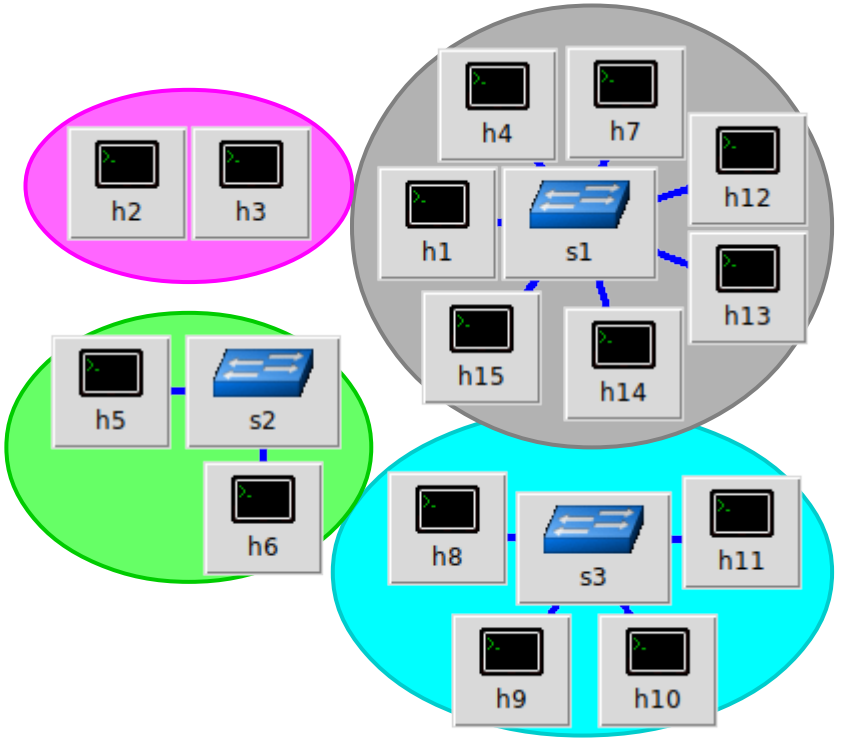
MAC/IPv4/IPv6
10.0.0.5
10.0.0.6

tk

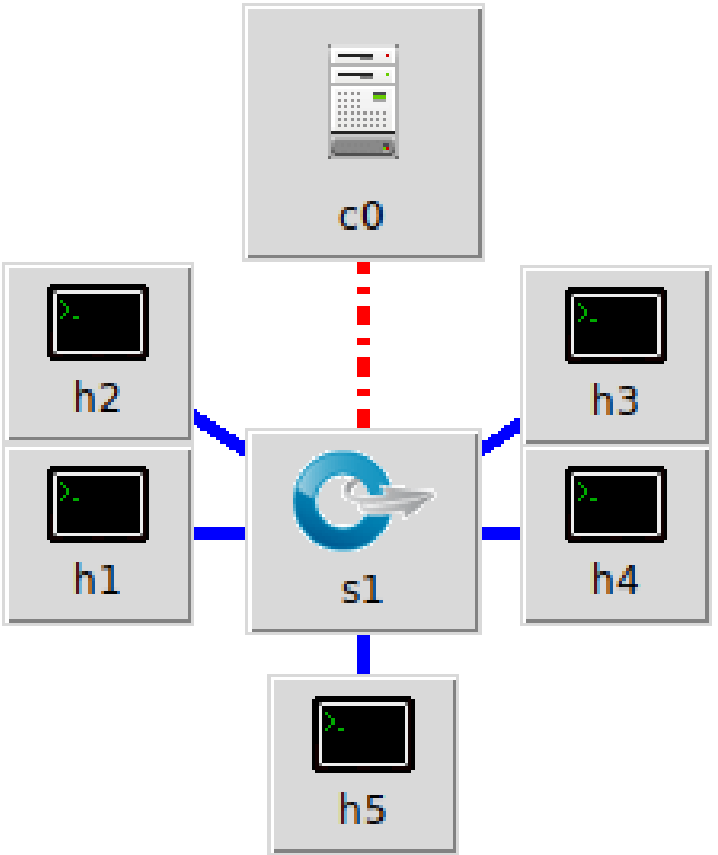
< vLAN #2

MAC/IPv4/IPv6
10.0.0.8
10.0.0.9
10.0.0.10
10.0.0.11



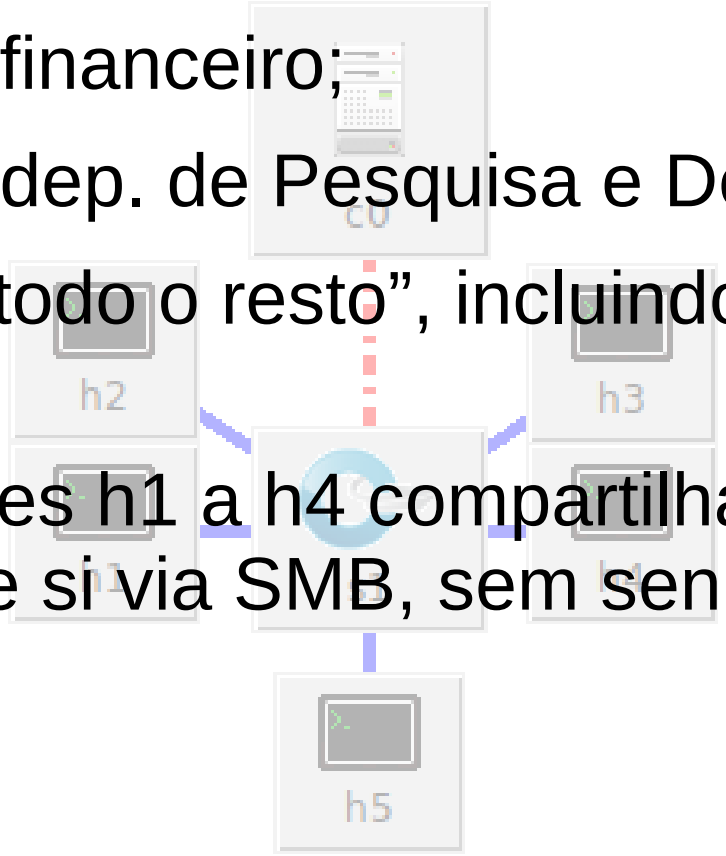


Exemplo



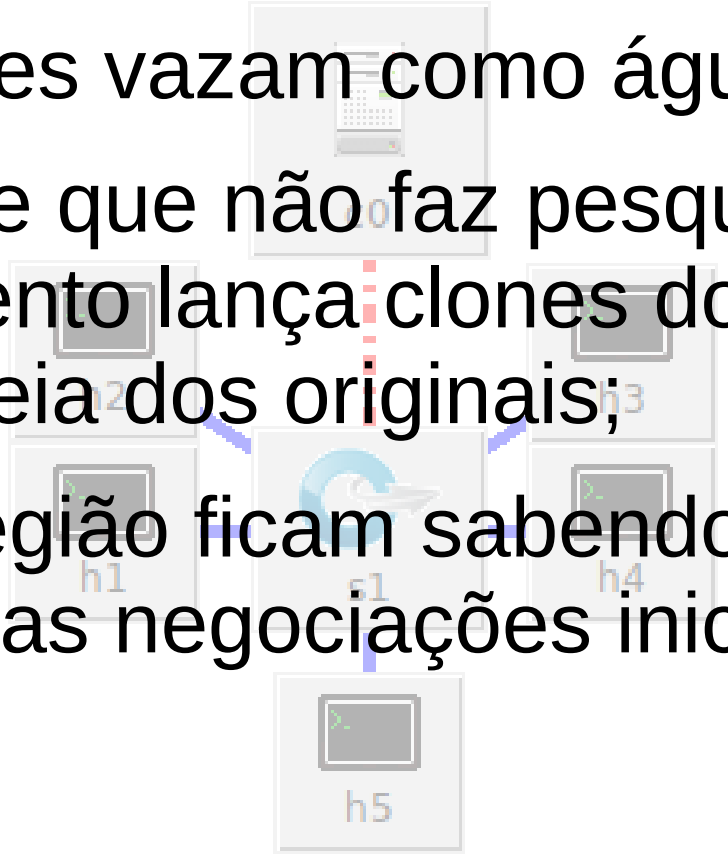
Exemplo

- h1 e h2 são do financeiro;
- h3 e h4 são do dep. de Pesquisa e Desenvolvimento;
- h5 representa “todo o resto”, incluindo o WiFi sem senha;
- Os computadores h1 a h4 compartilham documentos relevantes entre si via SMB, sem senha.



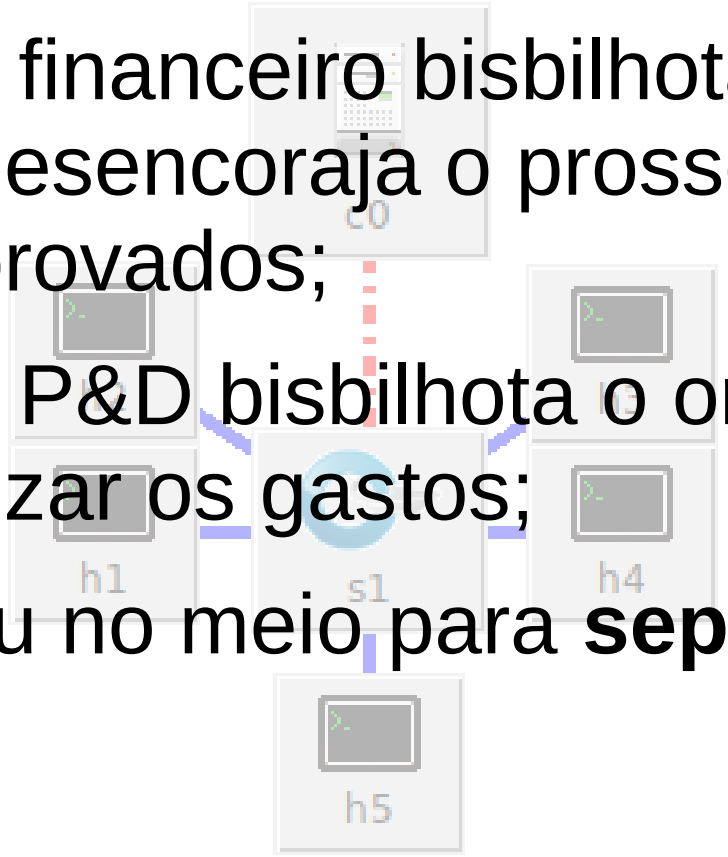
Exemplo

- As informações vazam como água;
- O concorrente que não faz pesquisa e desenvolvimento lança clones dos produtos antes da estreia dos originais;
- As lojas da região ficam sabendo do preço de custo antes das negociações iniciarem.



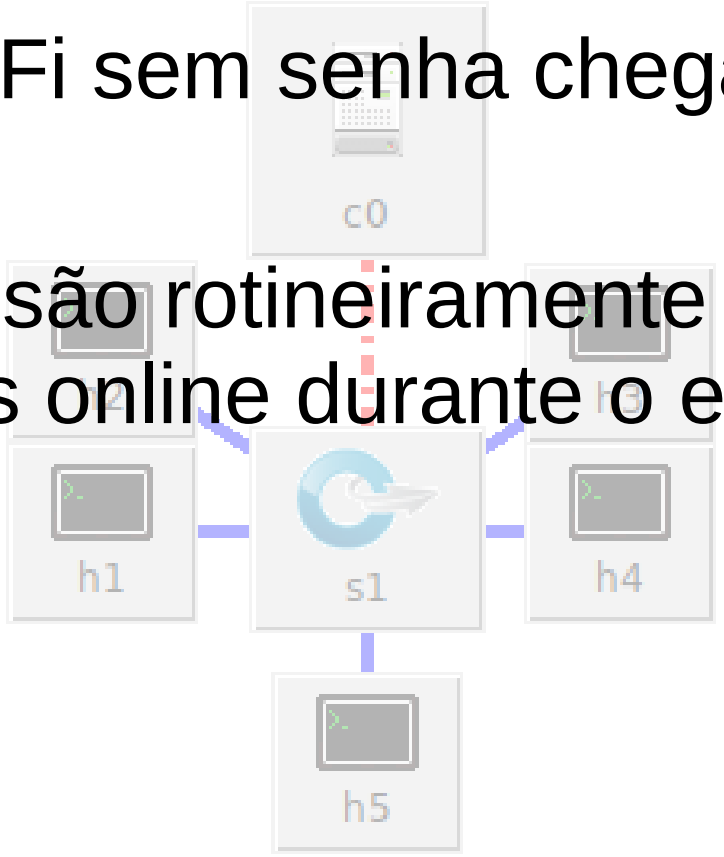
Exemplo

- O pessoal do financeiro bisbilhota o que o P&D vai gastar e desencoraja o prosseguimento de projetos já aprovados;
- O pessoal de P&D bisbilhota o orçamento para poder maximizar os gastos;
- O dono entrou no meio para **separá-los**.



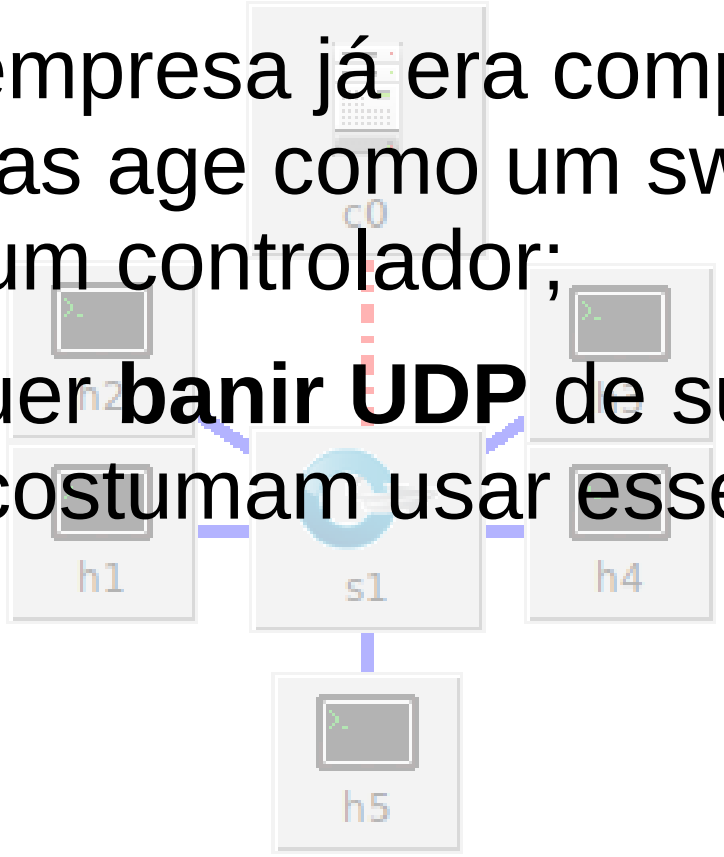
Exemplo

- O sinal do WiFi sem senha chega no meio da rua;
- Funcionários são rotineiramente flagrados jogando jogos online durante o expediente.



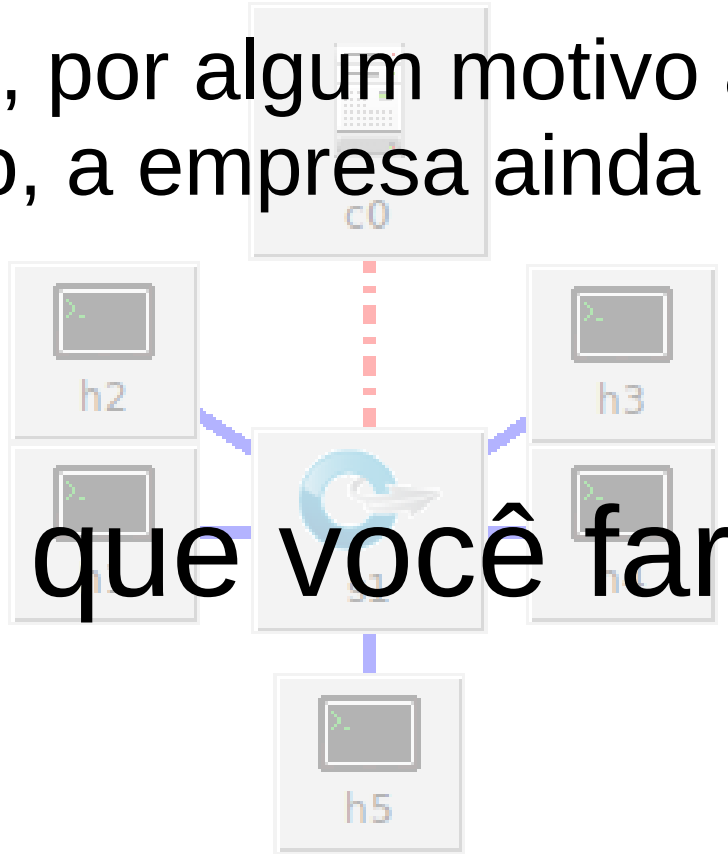
Exemplo

- O switch da empresa já era compatível com OpenFlow, mas age como um switch na ausência de um controlador;
- A empresa quer **banir UDP** de sua rede, já que jogos online costumam usar esse protocolo.



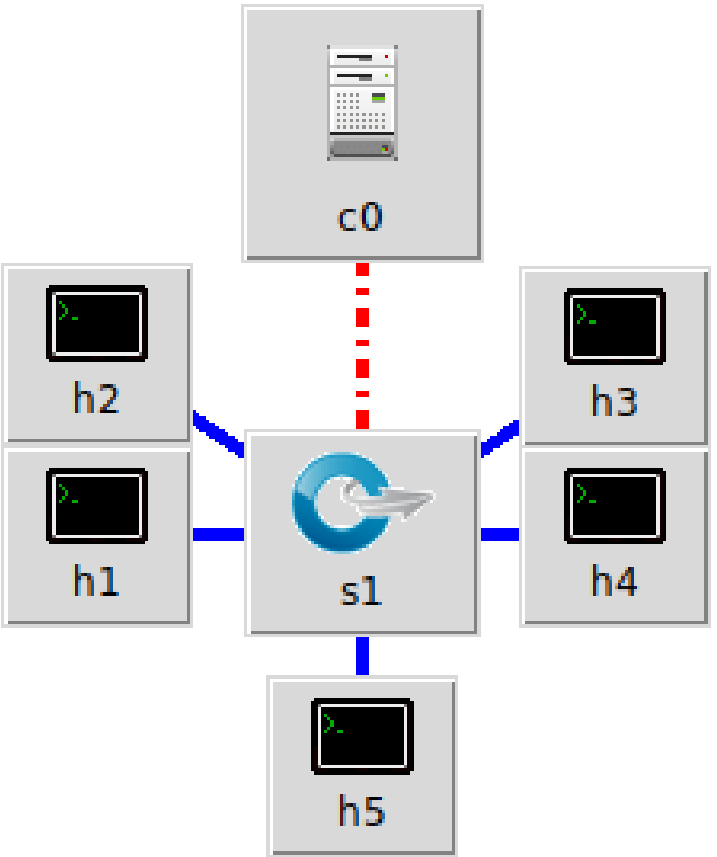
Exemplo

Supondo que, por algum motivo ainda desconhecido, a empresa ainda não fechou às portas…

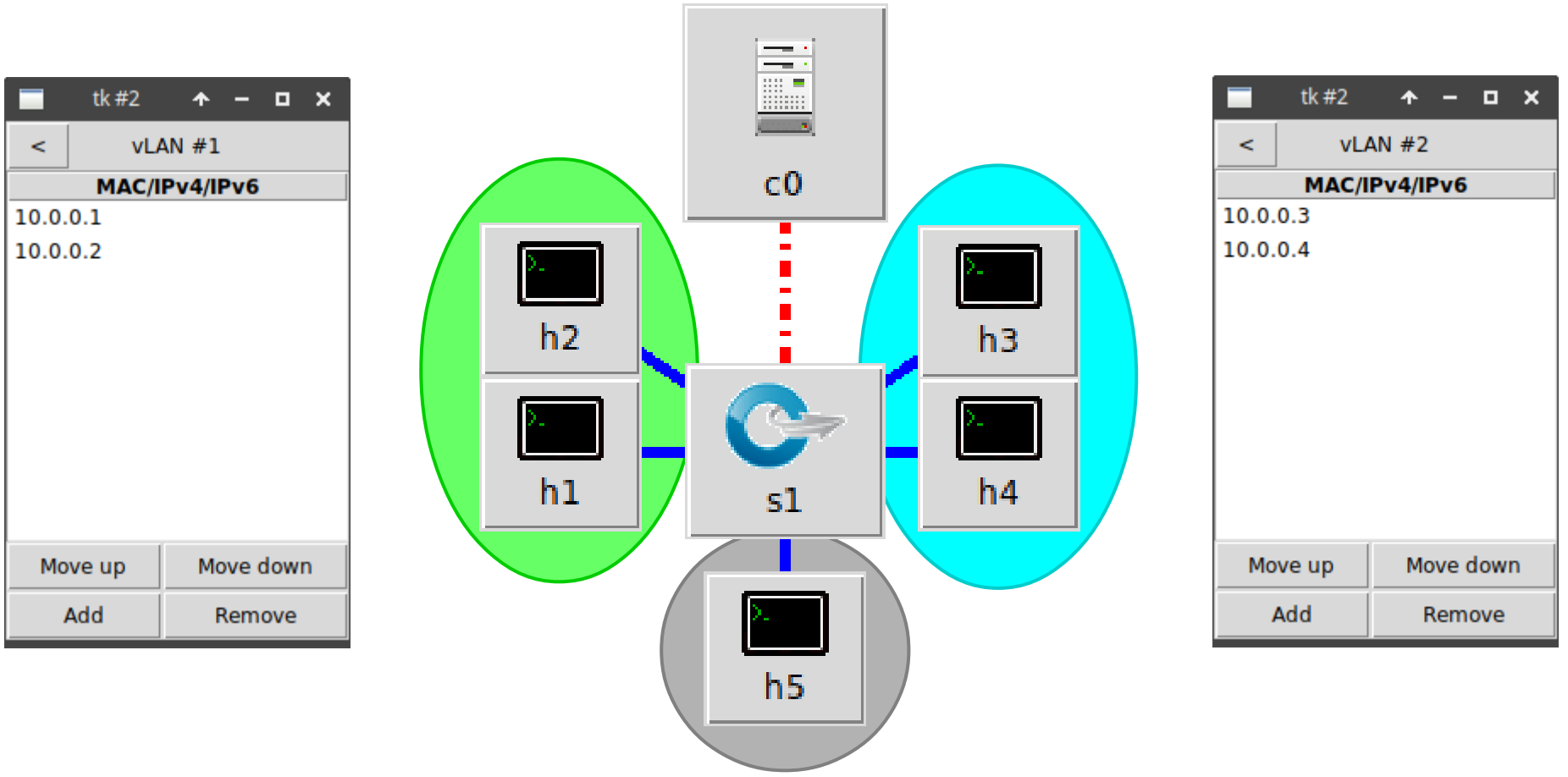


O que você faria?

Exemplo

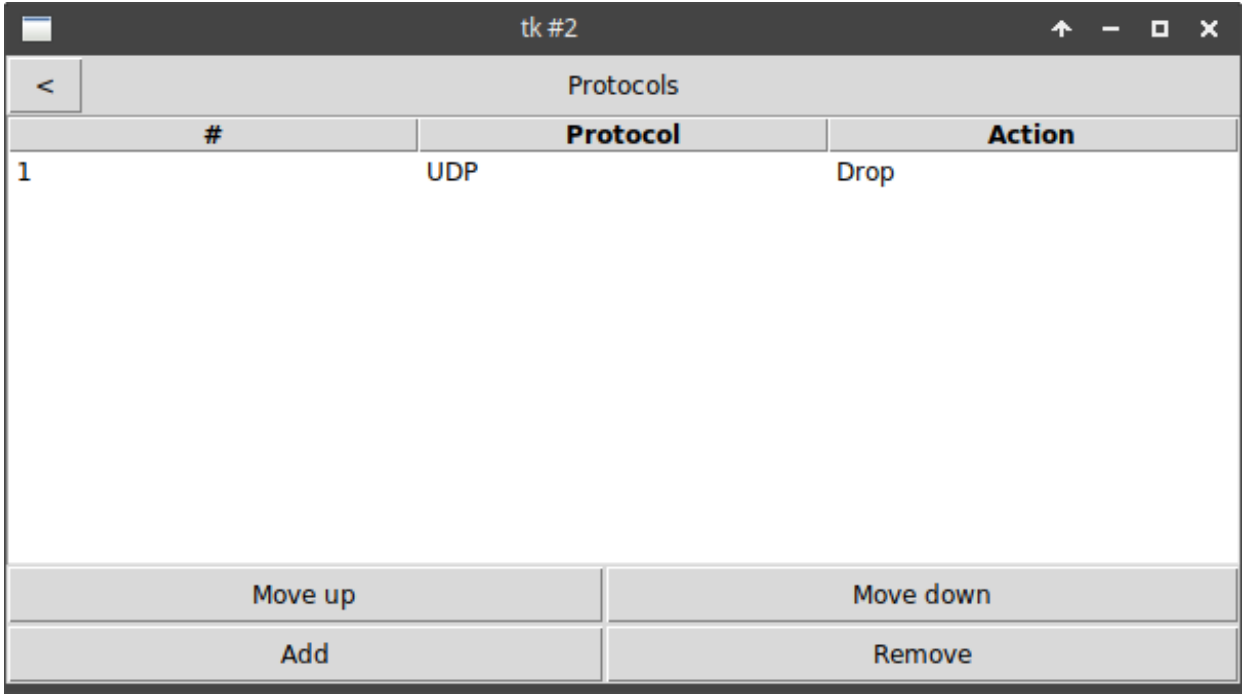


Exemplo


tk #2
<
vLAN #1
MAC/IPv4/IPv6
10.0.0.1
10.0.0.2
Move up
Move down
Add
Remove
c0
h2
h1
s1
h3
h4
h5
tk #2
<
vLAN #2
MAC/IPv4/IPv6
10.0.0.3
10.0.0.4
Move up
Move down
Add
Remove

	h1	h2	h3	h4	h5
h1	-	✓			
h2	✓	-			
h3			-	✓	
h4			✓	-	
h5					-

Exemplo


tk #2
<
Protocols
#
Protocol
Action
1
UDP
Drop
Move up
Move down
Add
Remove

Demonstração

Comandos

- sudo ryu-manager ../firewall.py
- sudo watch -n 0 ovs-ofctl dump-flows s1

- sudo ./topo.py
  - pingall
- sudo ./topo.py
  - h1 xterm &
    - iperf -u -s
  - h2 xterm &
    - iperf -u -c 10.0.0.1

- sudo ./topo.py
  - h1 xterm &
    - iperf -s
  - h2 xterm &
    - iperf -c 10.0.0.1

Dúvidas?