SDX: A Software Defined Internet Exchange

@SIGCOMM 2014

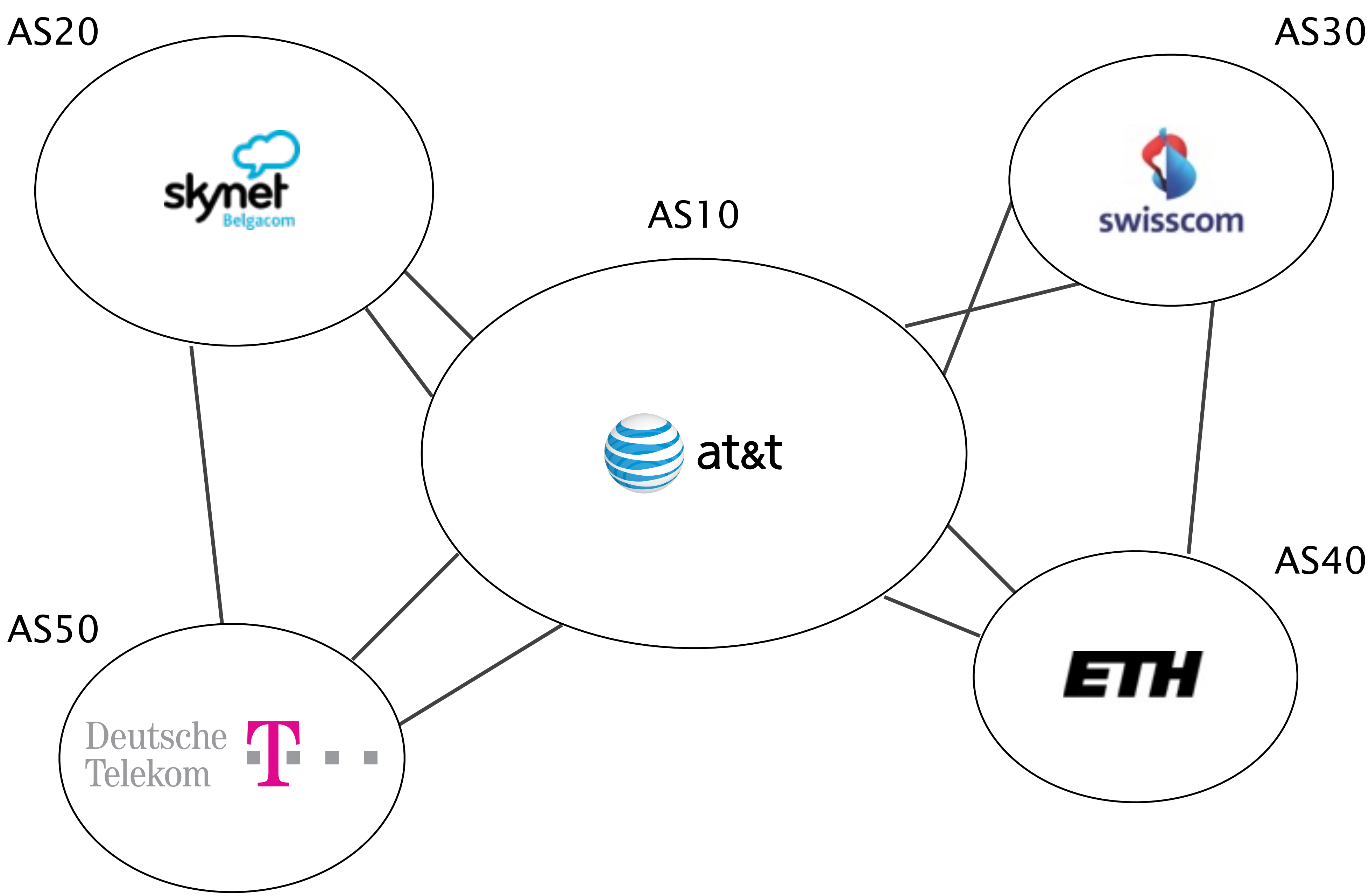
Laurent Vanbever

Princeton University

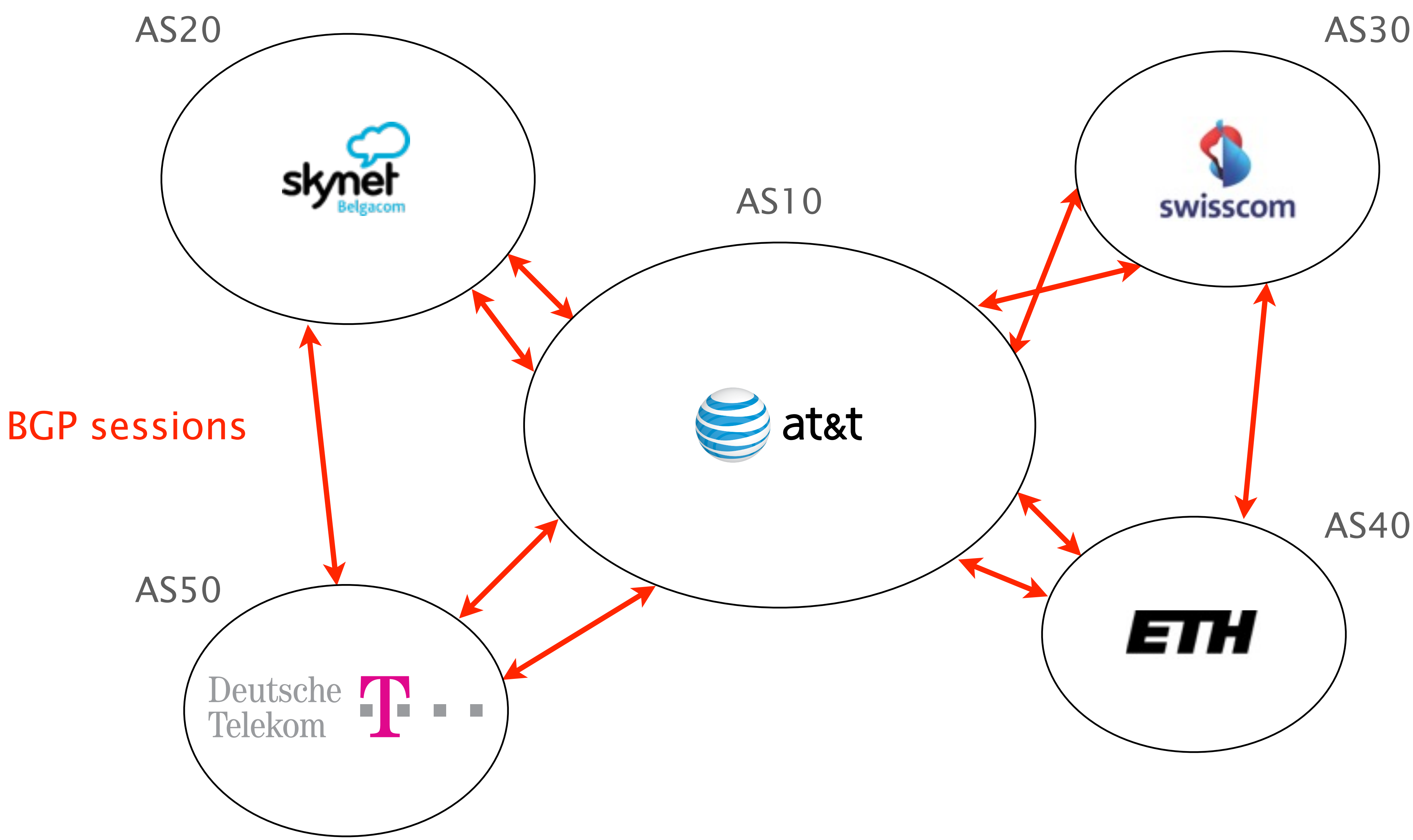
FGRE Workshop (Ghent, iMinds)

July, 8 2014

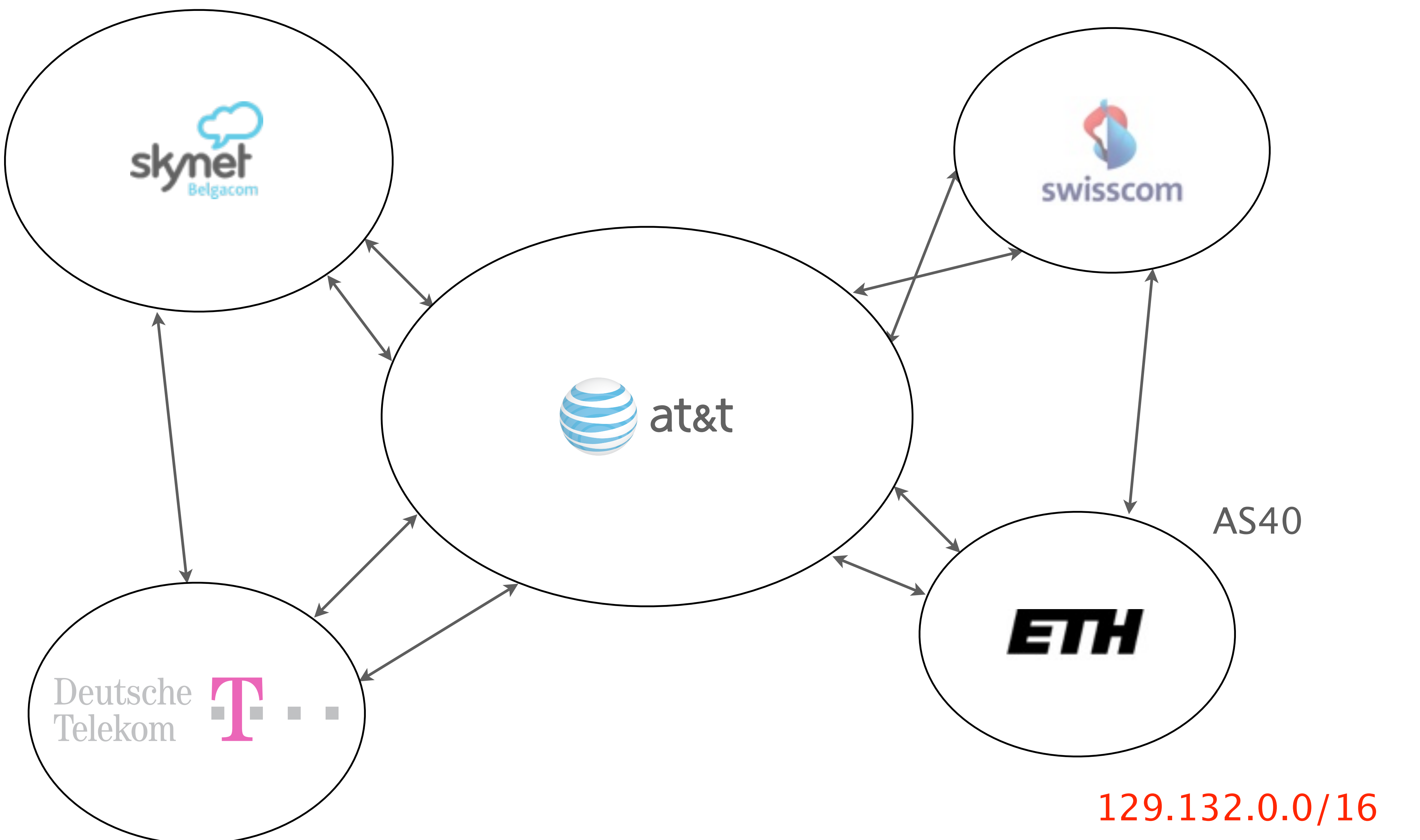
The Internet is a network of networks, referred to as Autonomous Systems (AS)



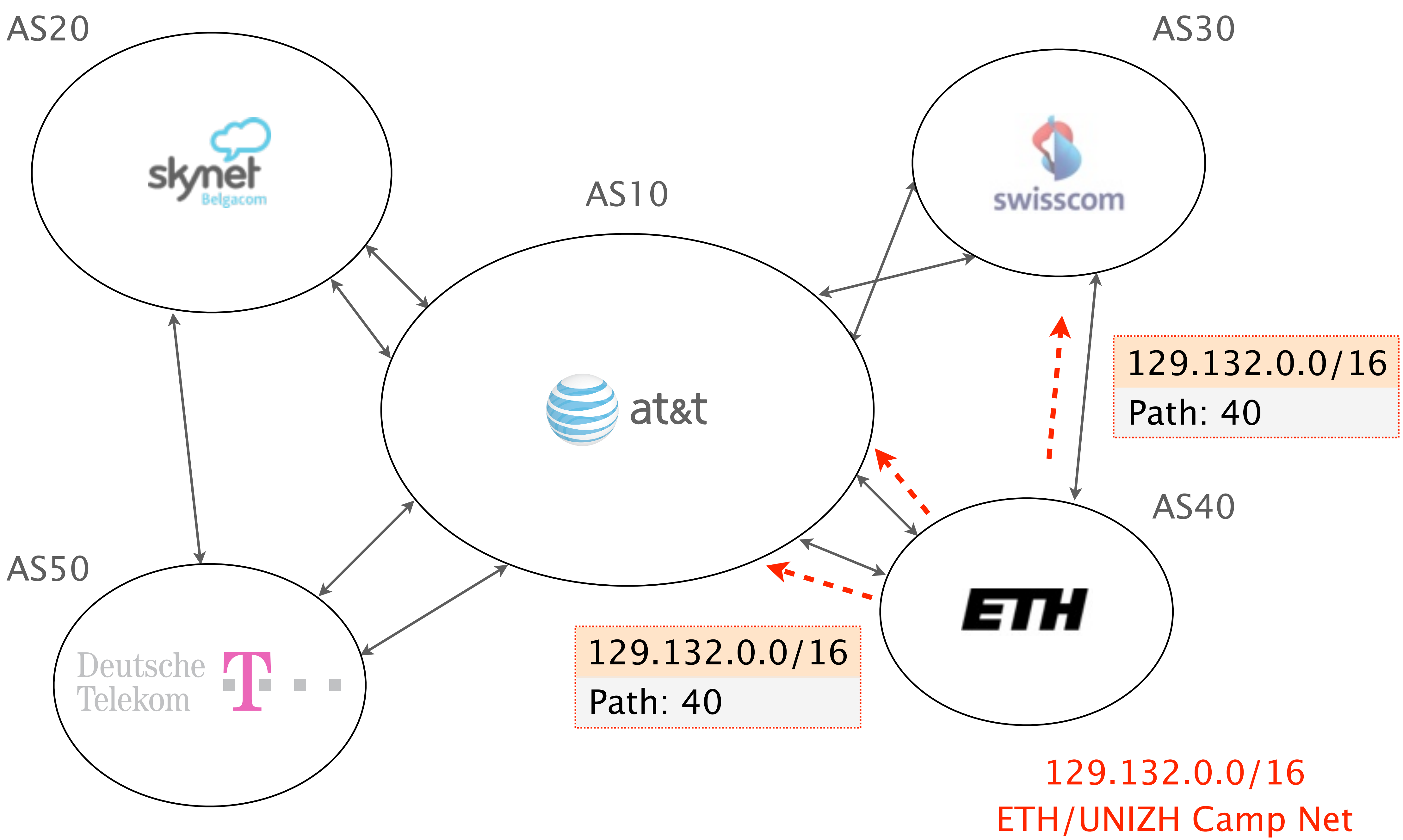
BGP is the routing protocol "glueing" the Internet together



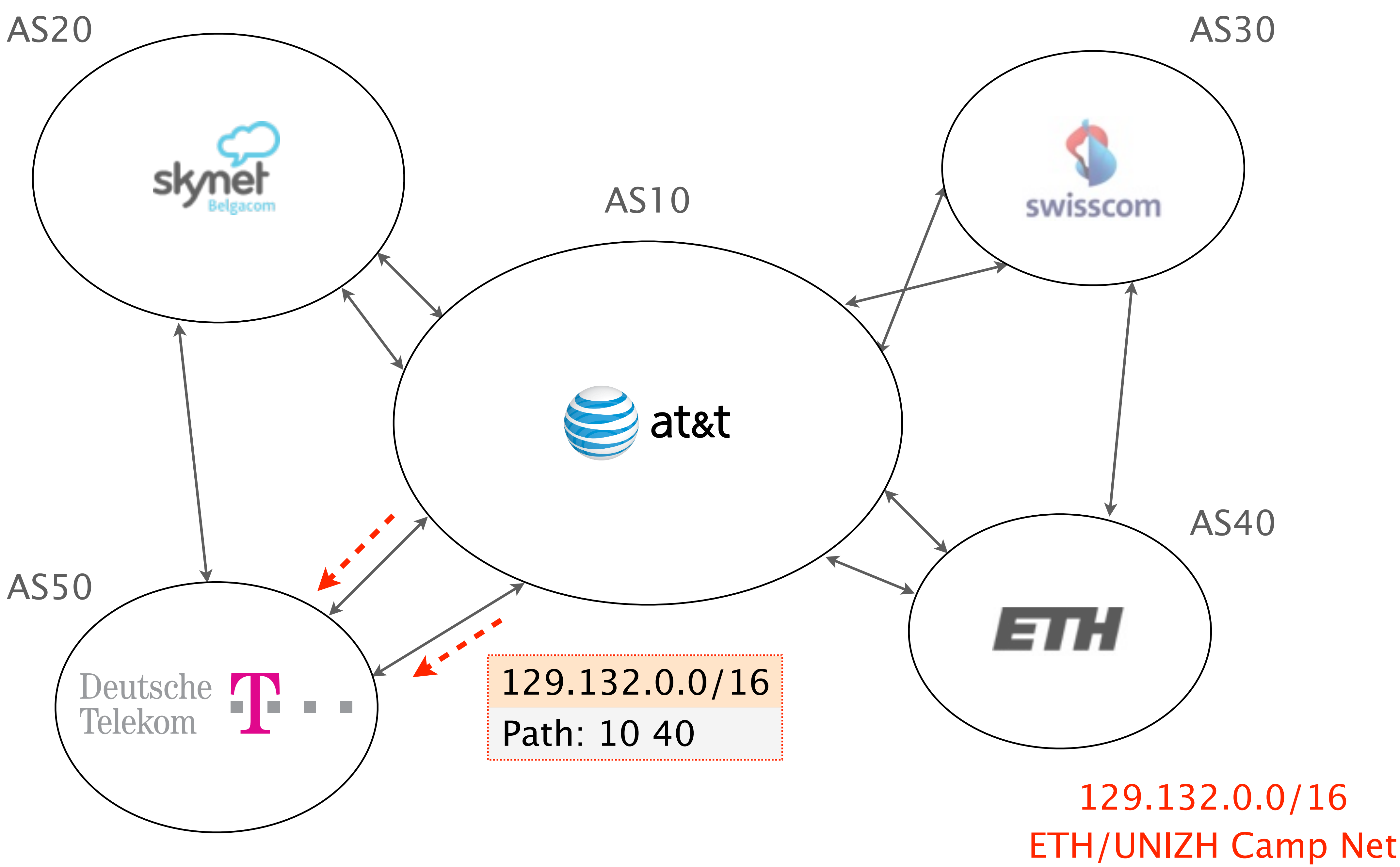
ASes exchange information about the IP prefixes they can reach



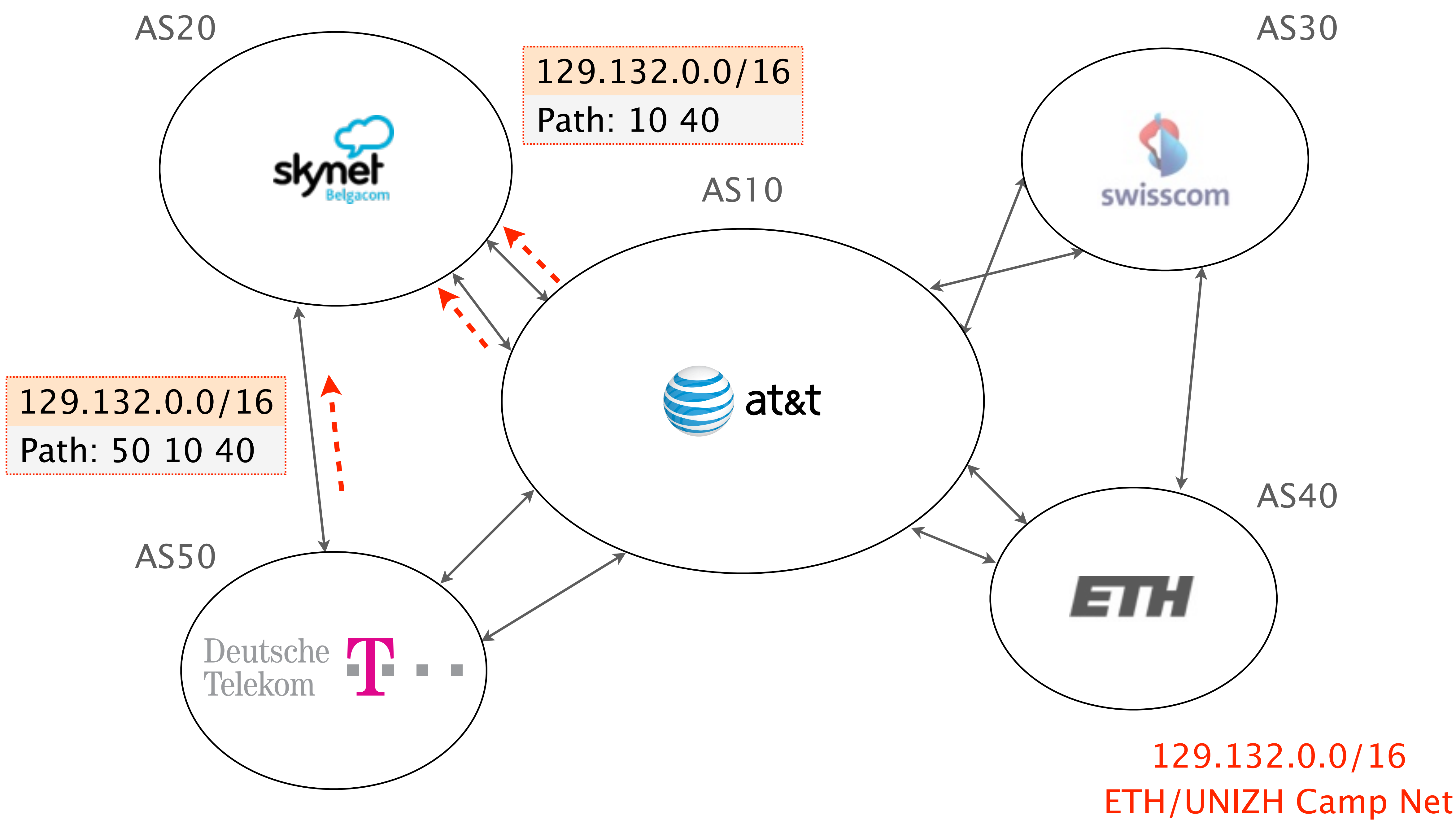
ASes exchange information about the IP prefixes they can reach



Reachability information is propagated hop-by-hop



Reachability information is propagated hop-by-hop



Life of a BGP router is made of three consecutive steps

while true:

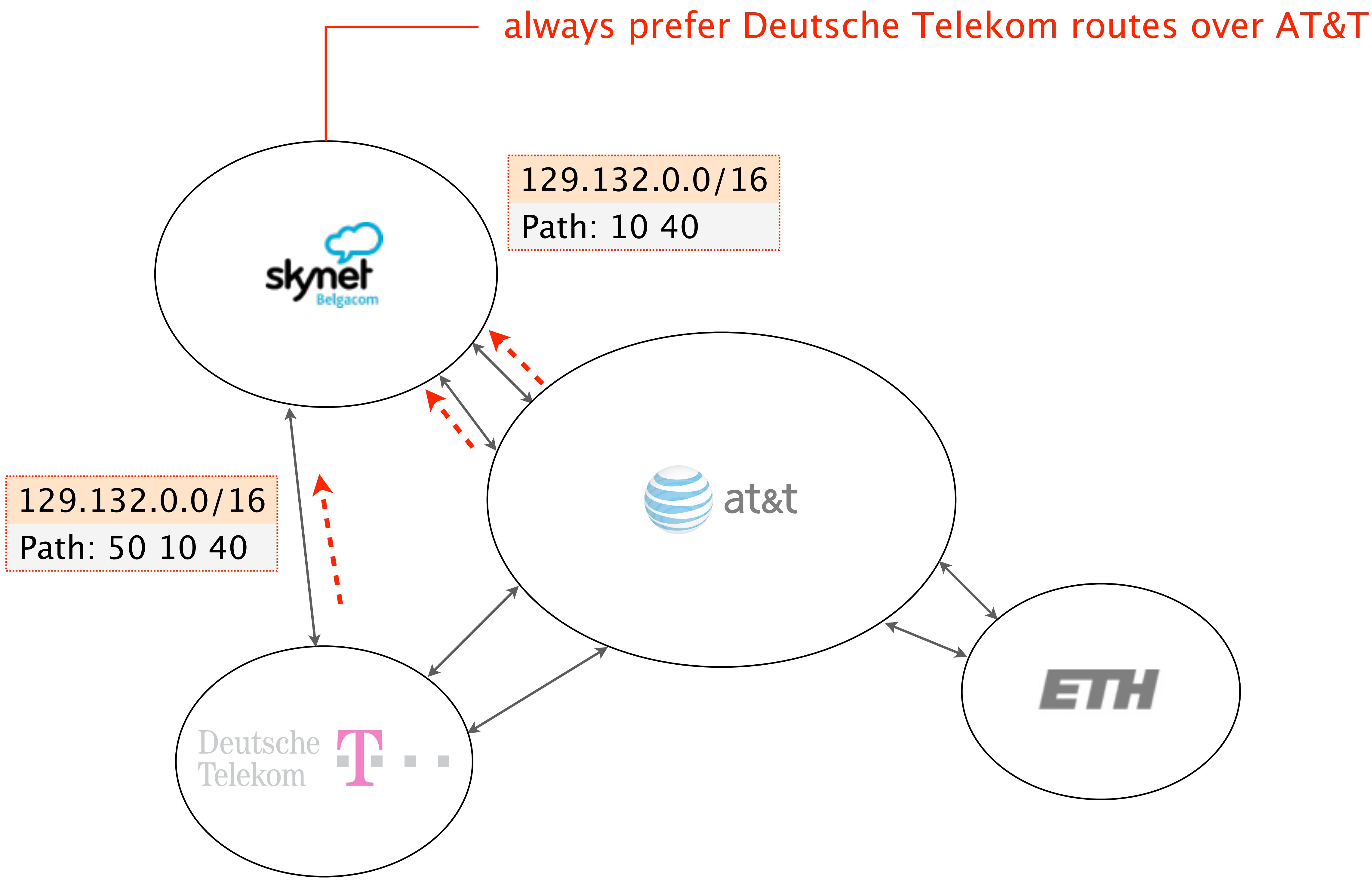
- receives routes from my neighbors
- select one best route for each prefix
- export the best route to my neighbors

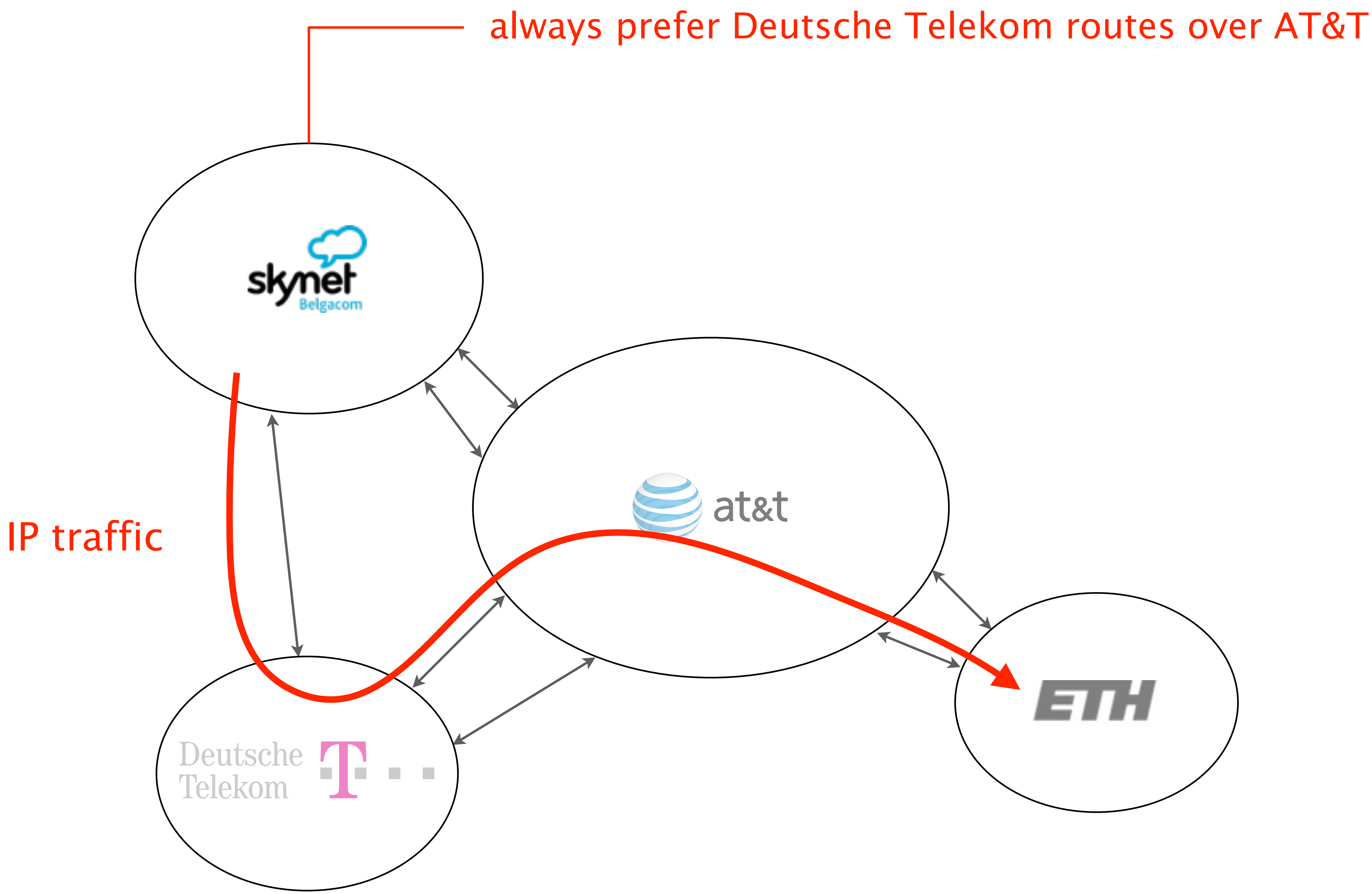
Each AS can apply local routing policies

Each AS is **free to**

- select and use any path

  preferably, the cheapest one


always prefer Deutsche Telekom routes over AT&T
129.132.0.0/16
Path: 10 40
129.132.0.0/16
Path: 50 10 40
skynet
Belgacom
at&t
Deutsche
Telekom
ETH


always prefer Deutsche Telekom routes over AT&T
skynet
Belgacom
IP traffic
at&t
ETH
Deutsche
Telekom
T

Each AS can apply local routing policies

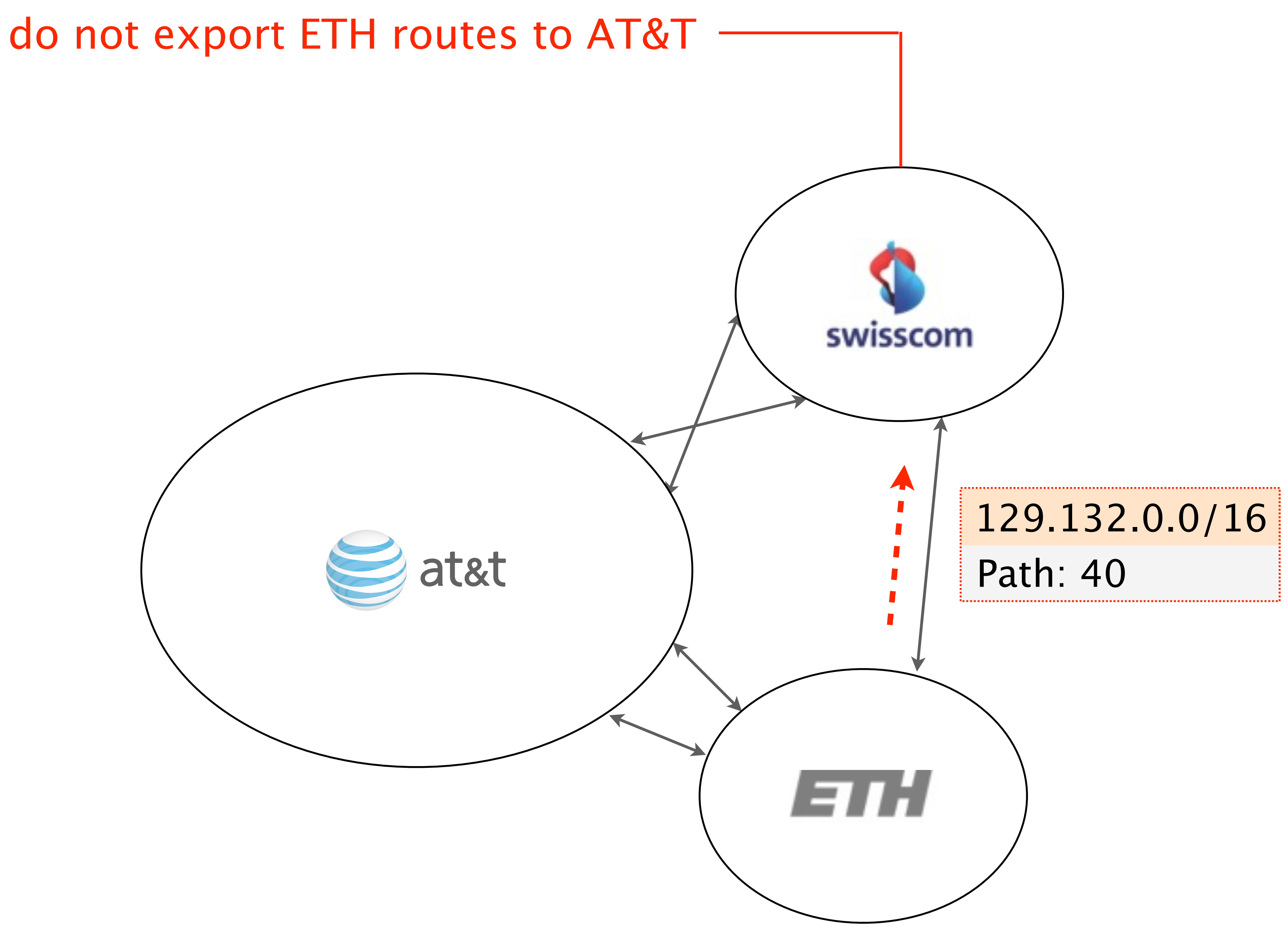
Each AS is free to

- select and use any path

  preferably, the cheapest one

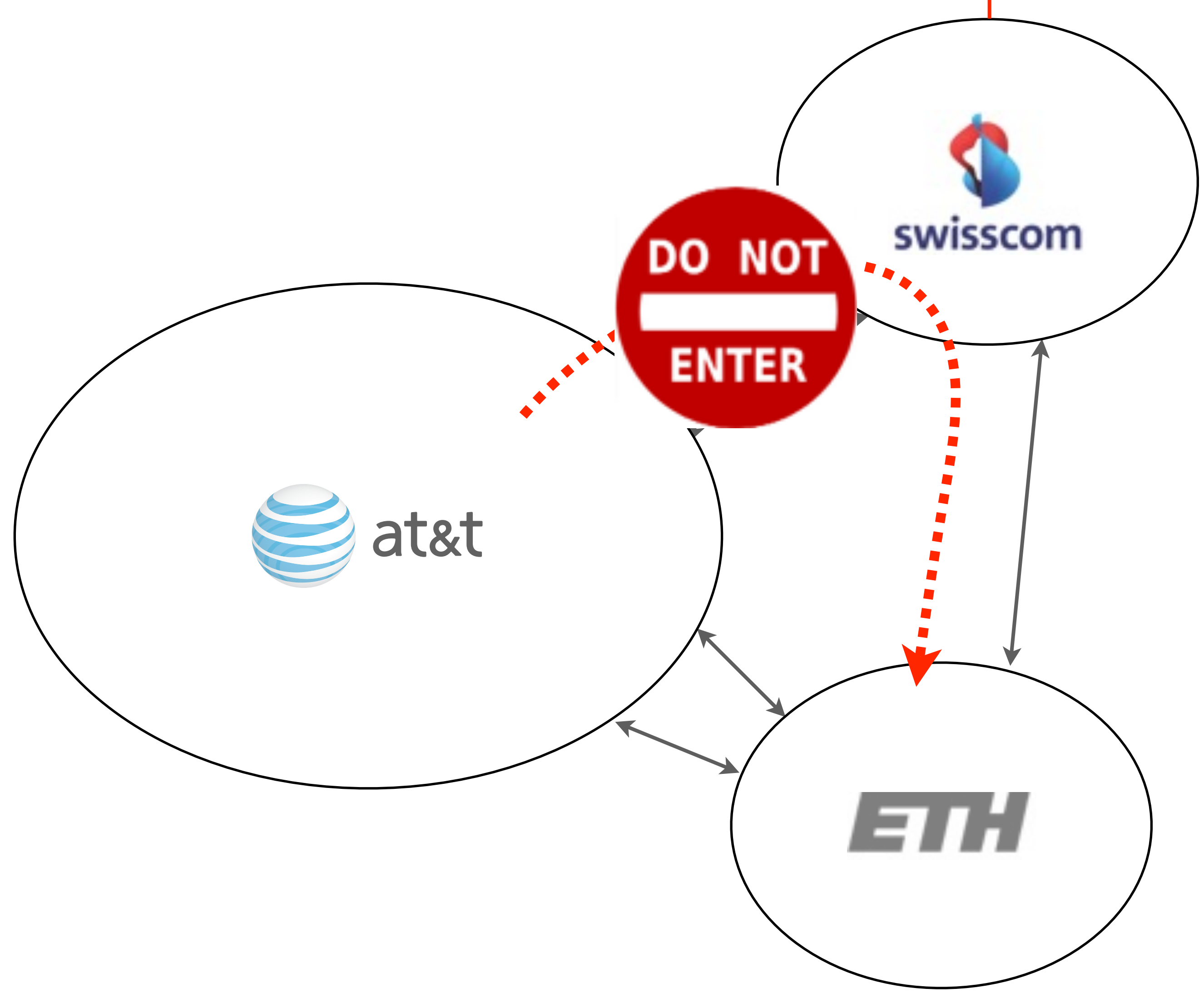
- decide which path to export (if any) to which neighbor

  preferably none, to minimize carried traffic



do not export ETH routes to AT&T
129.132.0.0/16
Path: 40

do not export ETH routes to AT&T



BGP is notoriously inflexible and difficult to manage

BGP is notoriously inflexible and difficult to manage

Fwd paradigm

Fwd control

Fwd influence

BGP is notoriously inflexible and difficult to manage

	BGP
Fwd paradigm	destination-based
Fwd control	indirect configuration
Fwd influence	local BGP session

SDN can enable fine-grained, flexible and direct expression of interdomain policies

	BGP	SDN
Fwd paradigm	destination-based	any
		source addr, ports, VLAN,…
Fwd control	indirect	direct
	configuration	open API (*e.g.*, OpenFlow)
Fwd influence	local	global
	BGP session	remote controller control

How do you deploy SDN in a network composed of 50,000 subnetworks?

How do you deploy SDN in a network composed of 50,000 subnetworks?

Well, you don't …

Instead, you aim at finding locations where deploying SDN can have the most impact

Instead, you aim at finding locations where deploying SDN can have the most impact

Deploy SDN in locations that

- connect a large number of networks
- carry a large amount of traffic
- are opened to innovation

Internet eXchange Points (IXP) meet all the criteria

Deploy SDN in locations that	AMS-IX
connect a large number of networks	670 networks
carry a large amount of traffic	2.9 Tb/s (peak)
are opened to innovation	BGP Route Server Mobile peering Open peering…

https://www.ams-ix.net

A single deployment can have a large impact

Deploy SDN in locations that

- connect a large number of networks
- carry a large amount of traffic
- are opened to innovation

AMS-IX

670 networks

2.9 Tb/s (peak)

BGP Route Server
Mobile peering
Open peering…

https://www.ams-ix.net

SDX = SDN + IXP

SDX = SDN + IXP

Augment the IXP data-plane with SDN capabilities

keeping default forwarding and routing behavior

Enable fine-grained inter domain policies

bringing new features while simplifying operations

SDX = SDN + IXP

Augment the IXP data-plane with SDN capabilities

keeping default forwarding and routing behavior

Enable fine-grained inter domain policies

bringing new features while simplifying operations

… with scalability and correctness in mind

supporting the load of a large IXP and resolving conflicts

SDX enables a wide range of novel applications

security	Prevent/block policy violation Prevent participants communication Upstream blocking of DoS attacks
forwarding optimization	Middlebox traffic steering Traffic offloading Inbound Traffic Engineering Fast convergence
peering	Application-specific peering
remote-control	Influence BGP path selection Wide-area load balancing

SDX: A Software Defined Internet Exchange

1 **Architecture**
programming model

2 **Scalability**
control- & data-plane

3 **Applications**
inter domain bonanza

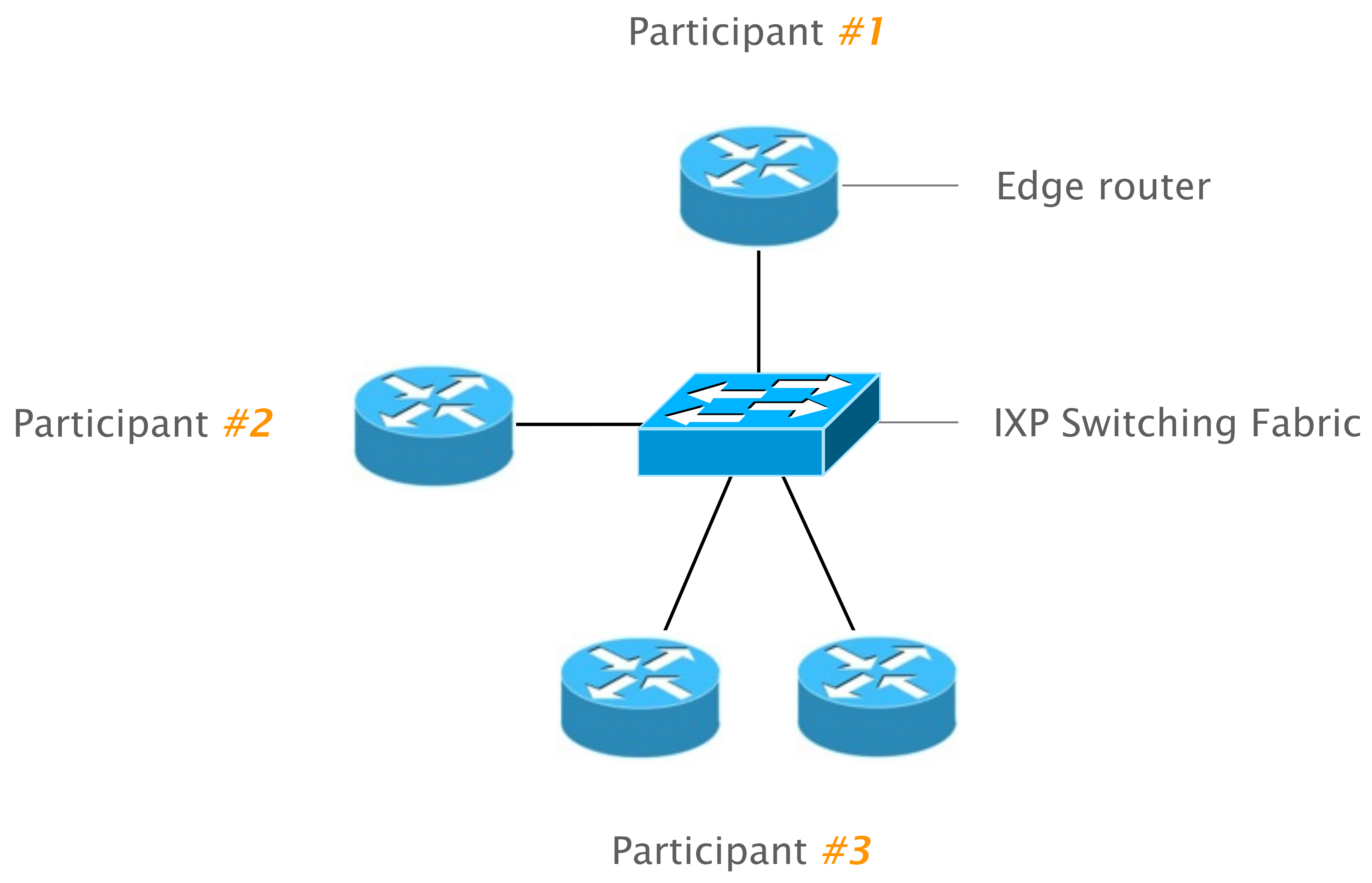
SDX: A Software Defined Internet Exchange

1 **Architecture**
programming model

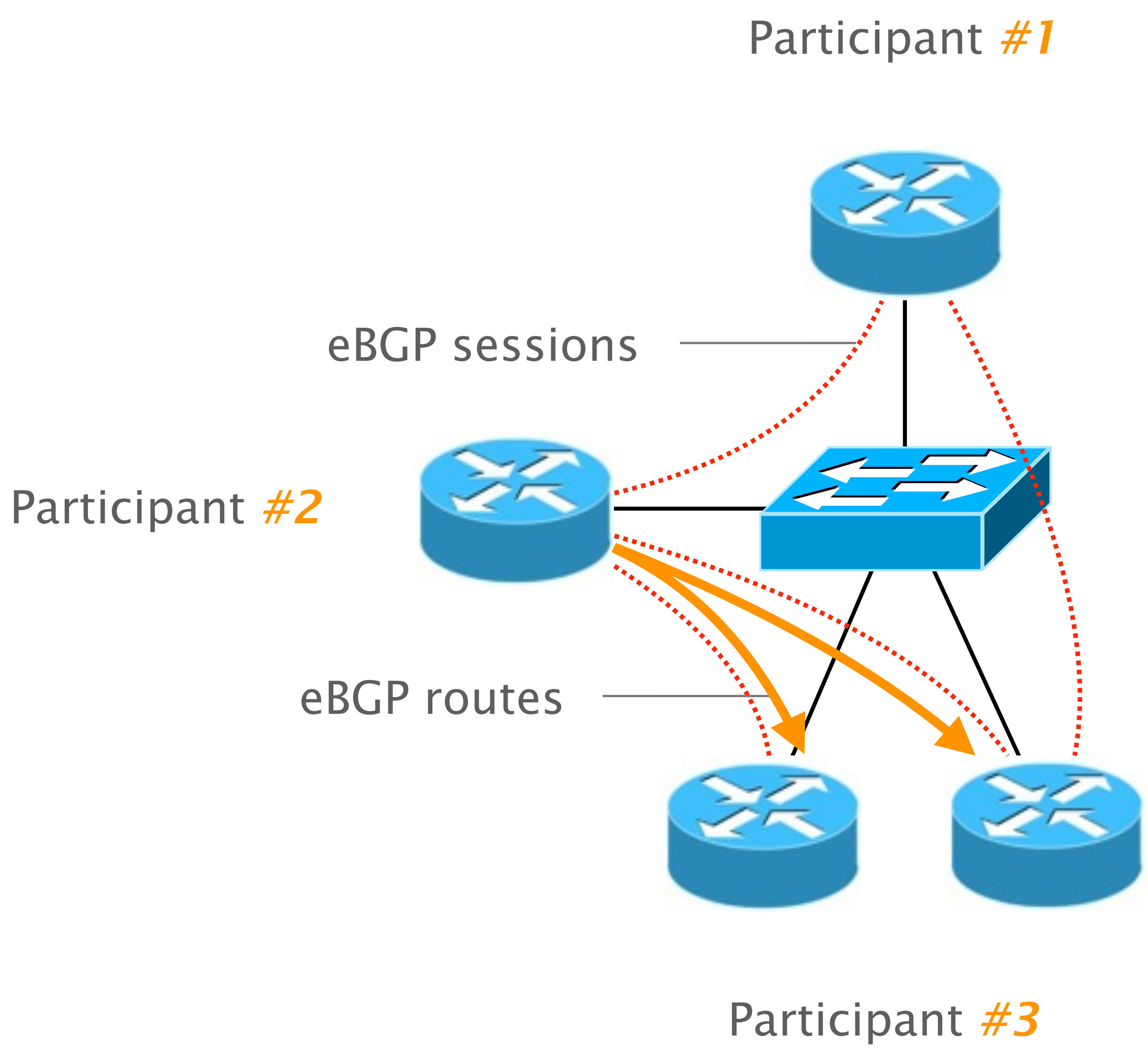
Scalability
control- & data-plane

Applications
inter domain bonanza

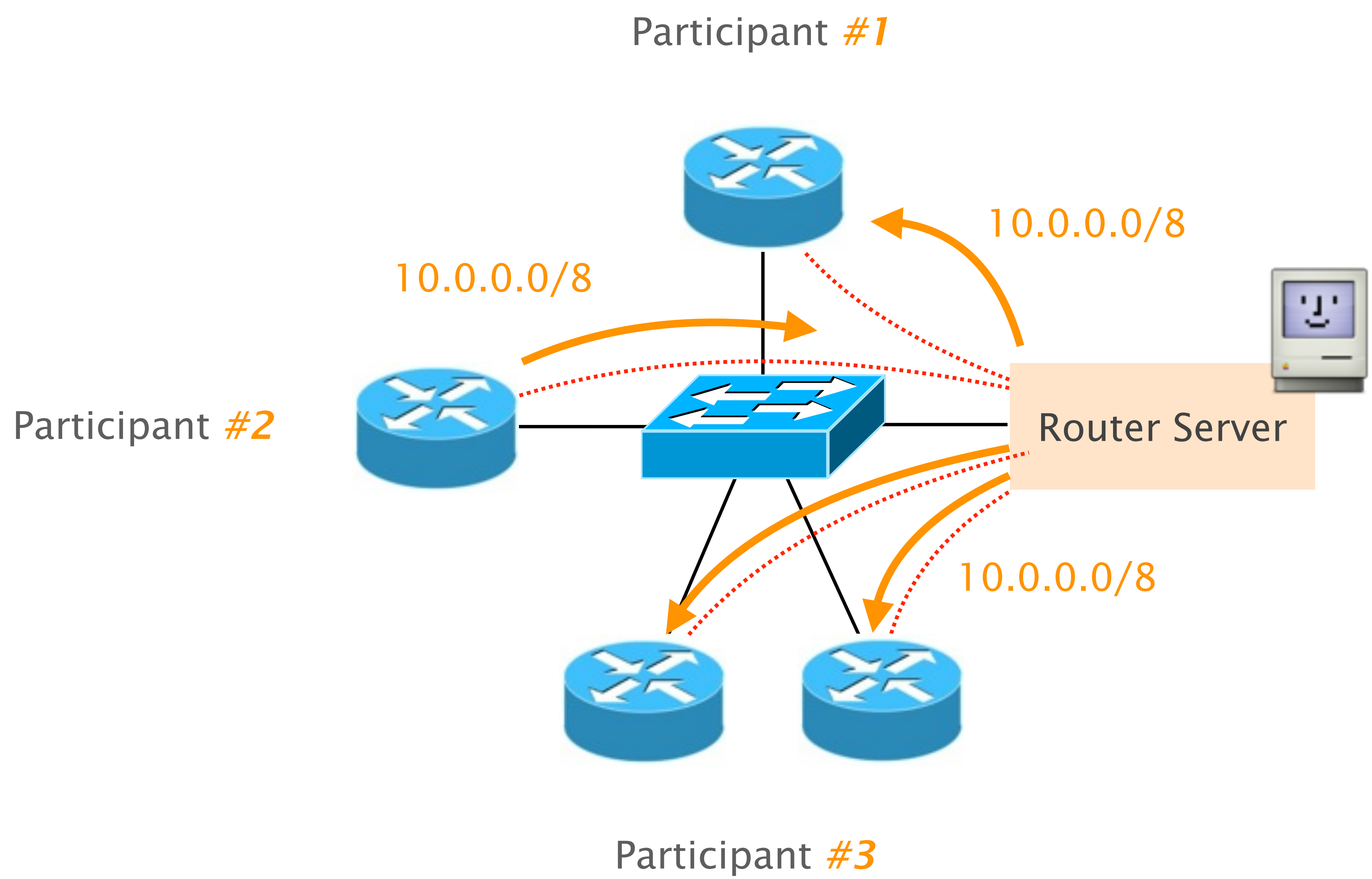
An IXP is a large layer-2 domain



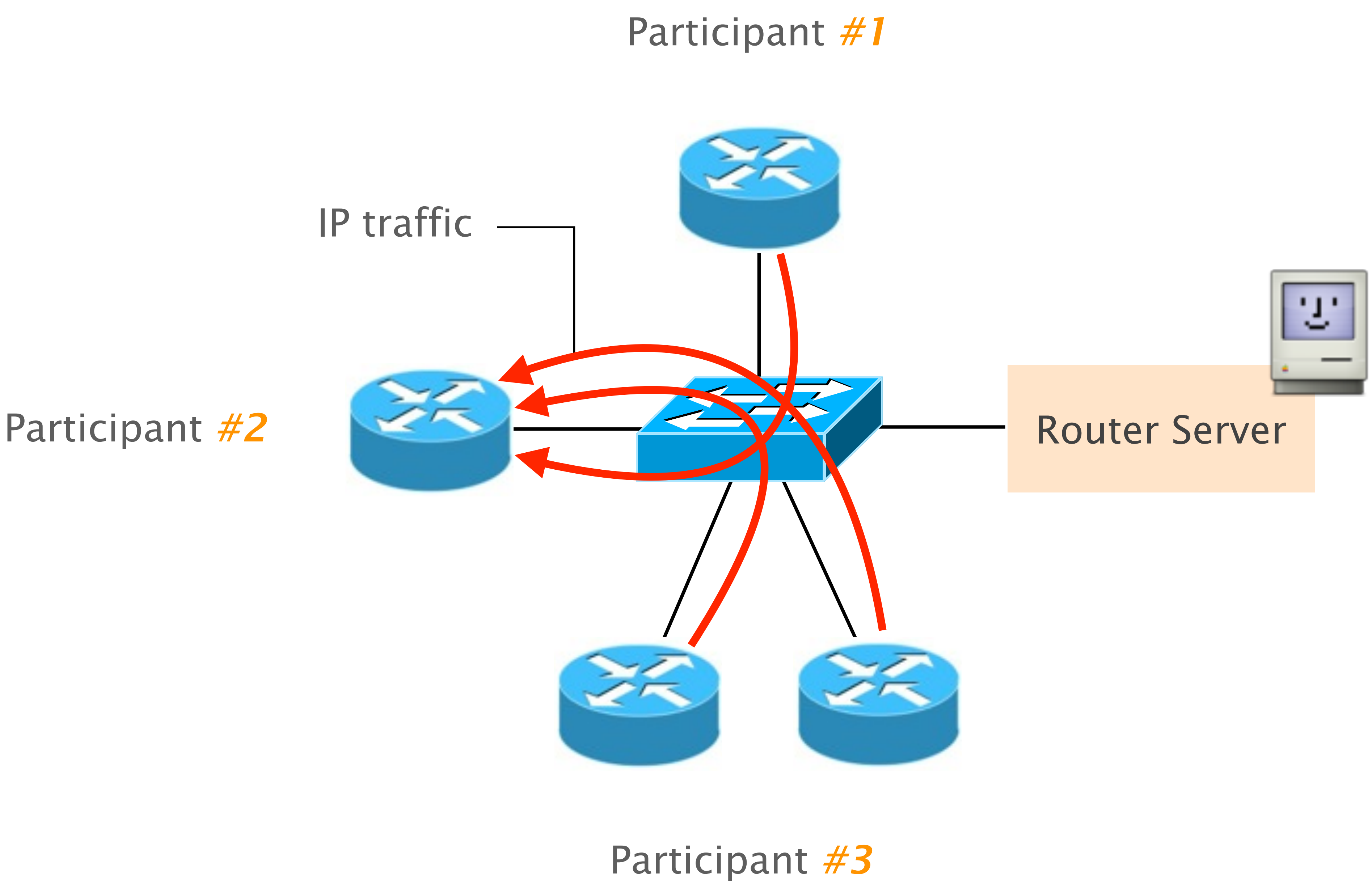
An IXP is a large layer-2 domain where participant routers exchange routes using BGP



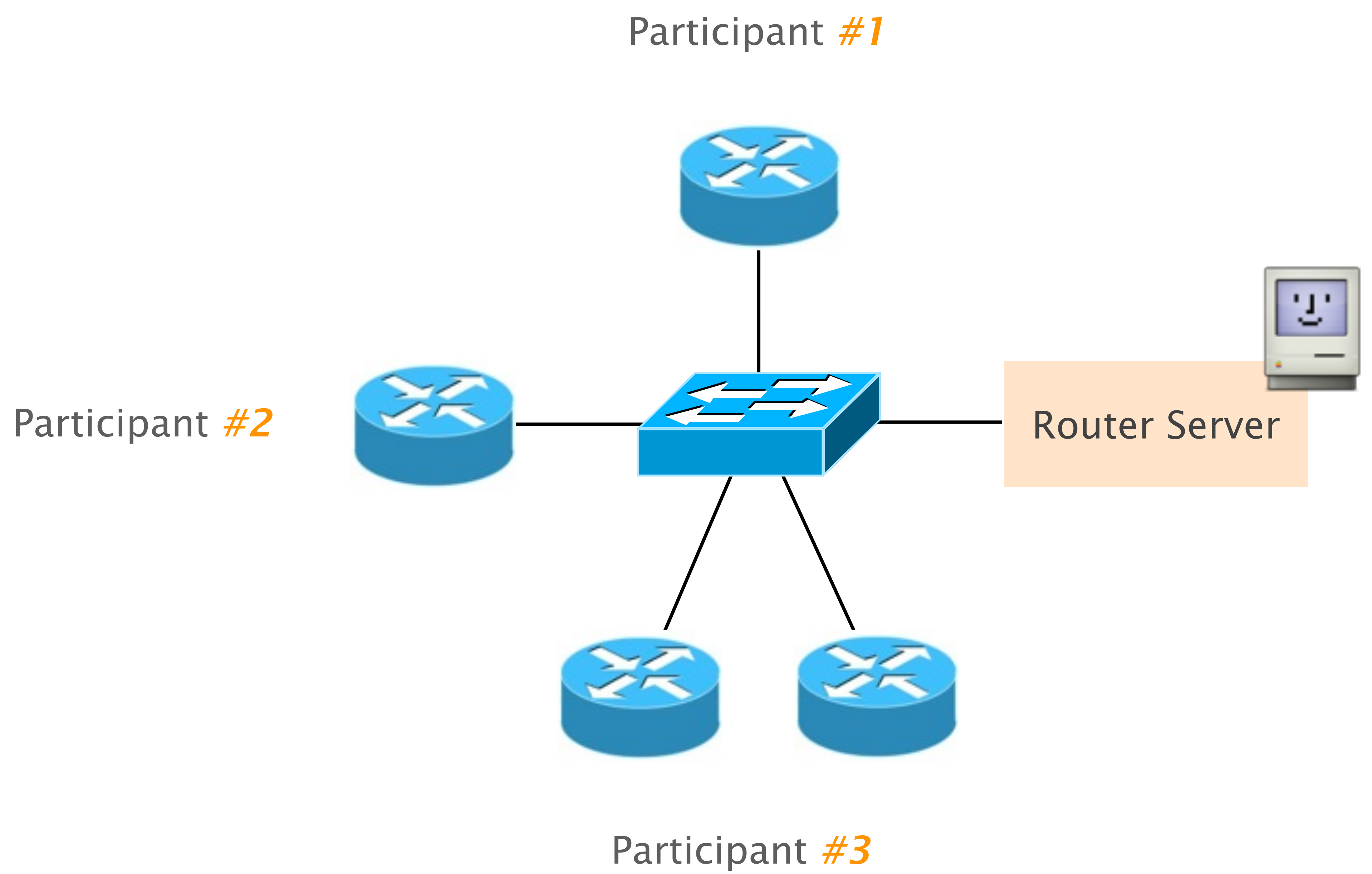
To alleviate the need of establishing eBGP sessions, IXP often provides a Route Server (route multiplexer)



IP traffic is exchanged directly between participants—IXP is forwarding transparent

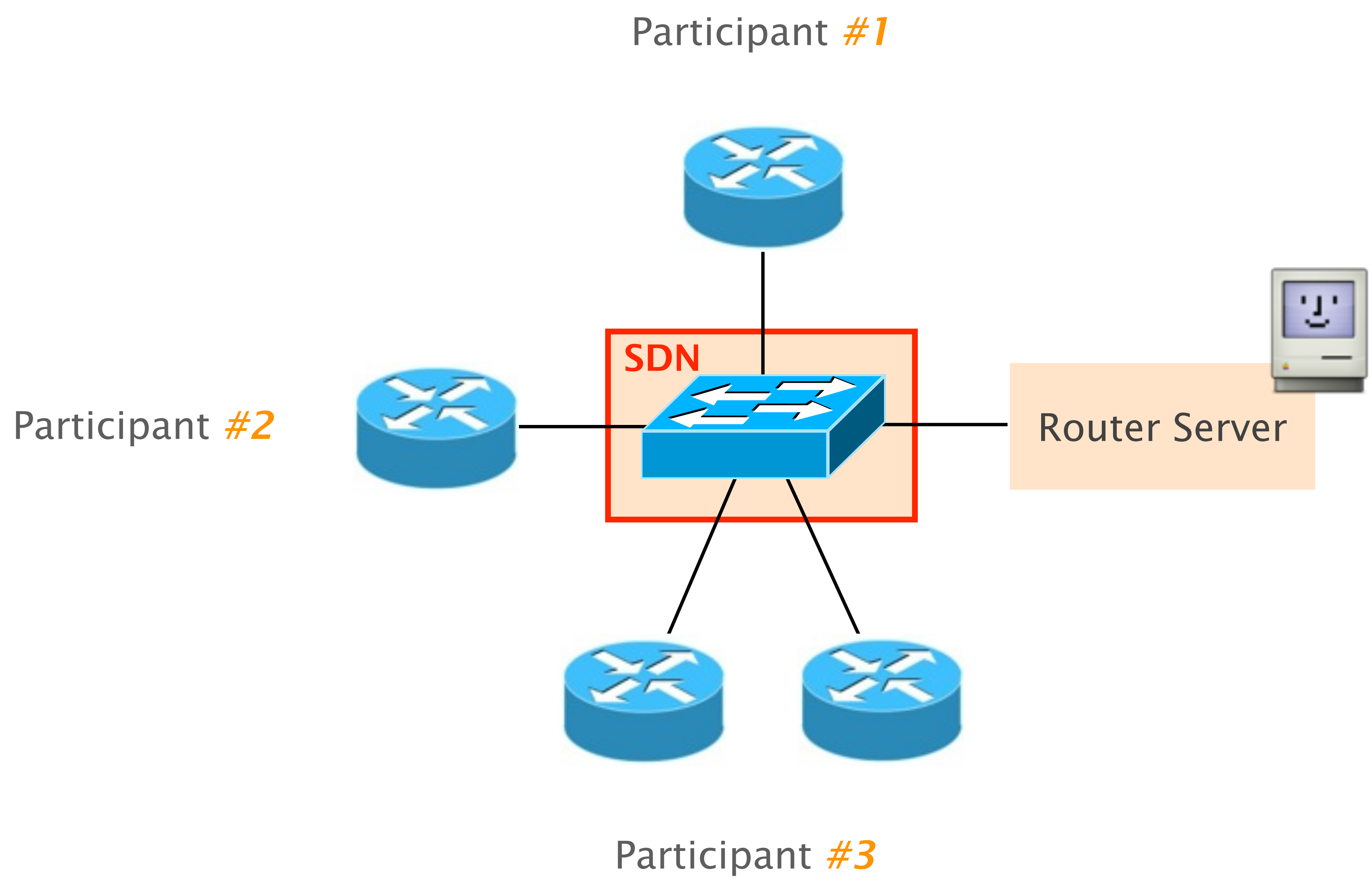


With respect to a traditional IXP, SDX...

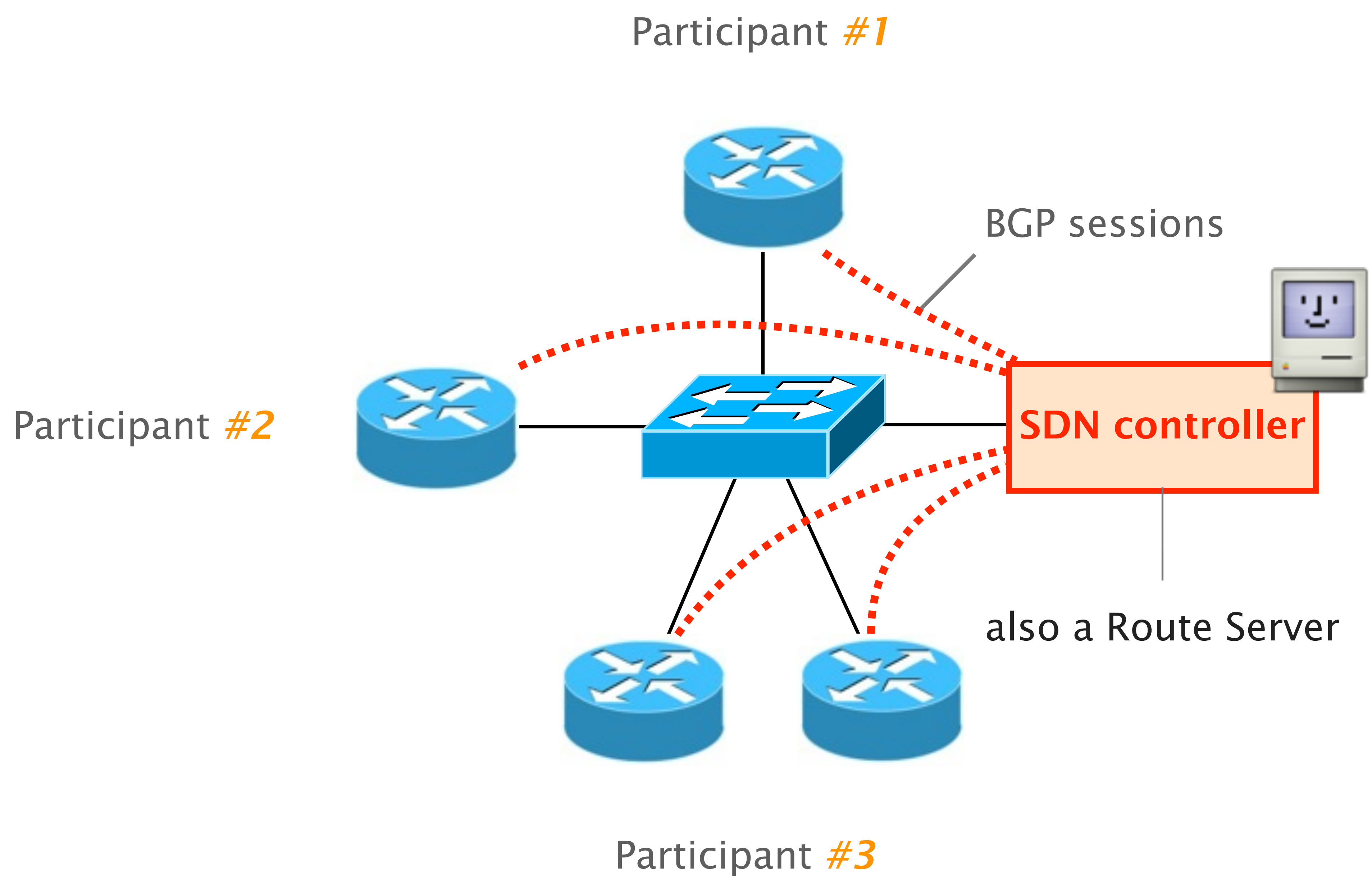


With respect to a traditional IXP, SDX's

data-plane relies on SDN-capable devices



With respect to a traditional IXP, SDX's control-plane relies on a SDN controller



SDX participants express their forwarding policies in a high-level language built on top of Pyretic (*)

(*) http://frenetic-lang.org/pyretic/

SDX policies are composed of
a *pattern* and some *actions*

```
match (     Pattern     ), then (  Actions  )
```

Pattern selects packets based on any header fields

Pattern

```
         eth_type
         vlan_id
         srcmac
match (  dstmac    , &&, ||  ), then (  Actions  )
         protocol
         dstip
         tos
         srcip
         srcport
         dstport
```

Pattern selects packets based on any header fields,

while actions forward or modify the selected packets

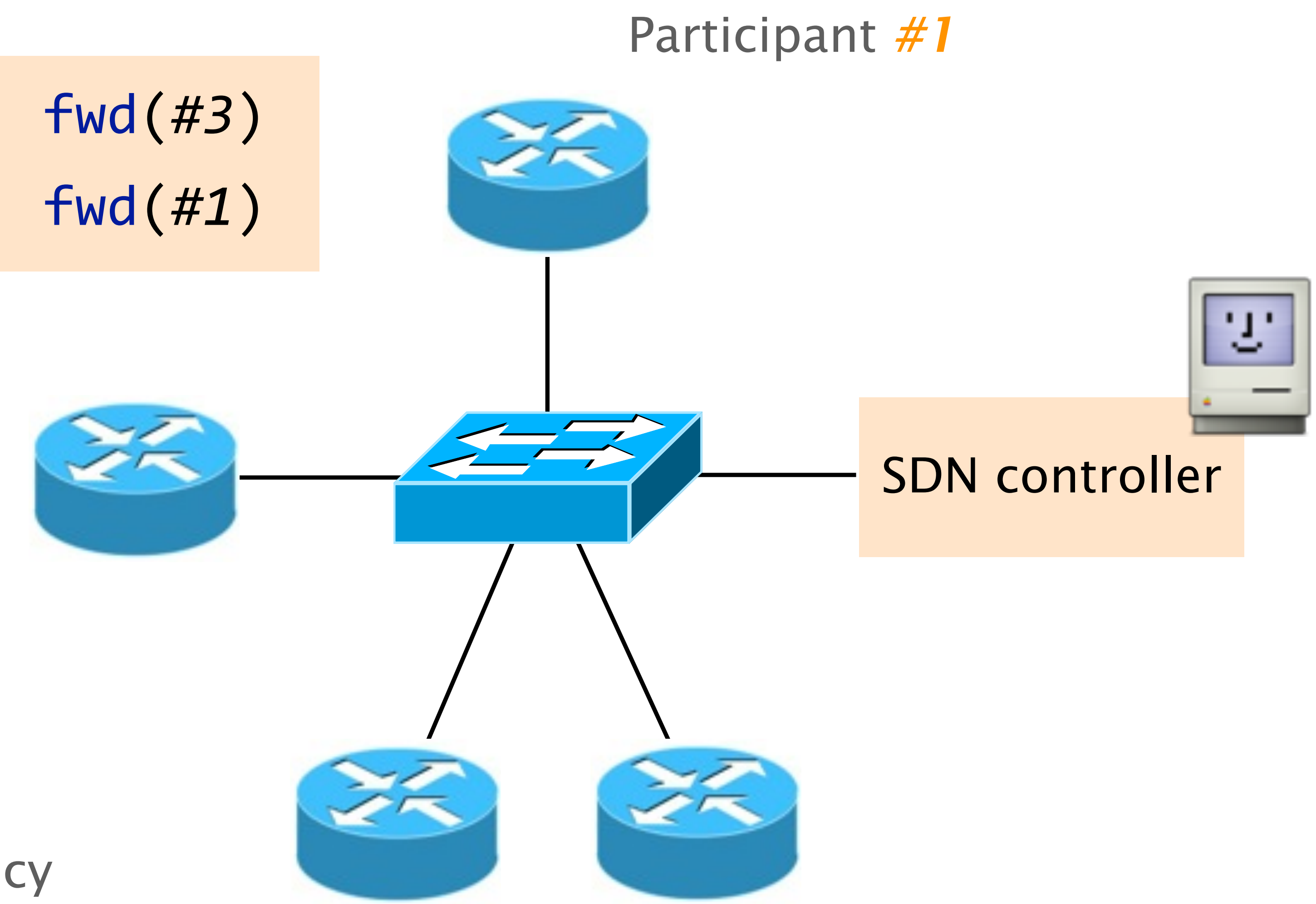
Actions

```
match (     Pattern     ), then ( drop
                                  forward )
                                  rewrite
```

Each participant writes policies independently and transmits them to the controller

Participant *#2* policy

```
match(dstport=80), fwd(#3)
match(dstport=22), fwd(#1)
```



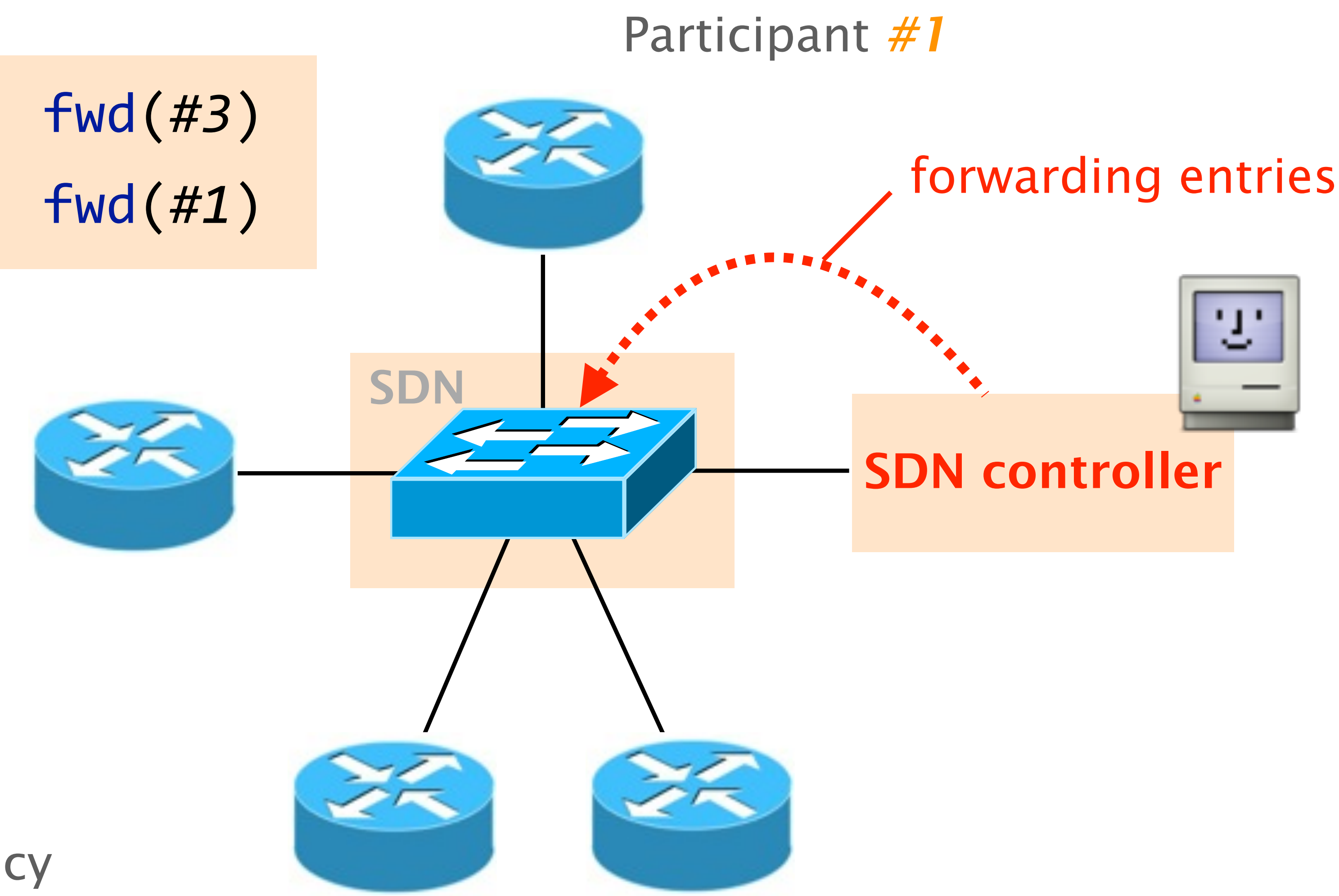
Participant *#3* policy

```
match(srcip=0*), fwd(left)
match(srcip=1*), fwd(right)
```

Given the participant policies,
the controller compiles them to SDN forwarding rules

Participant #2 policy

```
match(dstport=80), fwd(#3)
match(dstport=22), fwd(#1)
```



Participant #3 policy

```
match(srcip=0*), fwd(left)
match(srcip=1*), fwd(right)
```

Given the participant policies,
the controller compiles them to SDN forwarding rules

Ensuring isolation

Resolving policies conflict

Ensuring compatibility with BGP

Given the participant policies,
the controller compiles them to SDN forwarding rules

Ensuring isolation

Each participant controls
one virtual switch

connected to participants
it can communicate with

Resolving policies conflict

Ensuring compatibility with BGP

Given the participant policies,
the controller compiles them to SDN forwarding rules

Ensuring isolation

Resolving policies conflict

Participant policies are sequentially composed

in an order that respects business relationships

Ensuring compatibility with BGP

Given the participant policies,
the controller compiles them to SDN forwarding rules

Ensuring isolation

Resolving policies conflict

Ensuring compatibility with BGP

policies are augmented
with BGP information

guaranteed correctness
and reachability

SDX: A Software Defined Internet Exchange

Architecture

programming model

2 Scalability

control- & data-plane

Applications

inter domain bonanza

The SDX platform faces scalability challenges in both the data- and in the control-plane

data-plane

space

control-plane

time

data-plane

space

500,000 prefixes, 500+ participants, potentially *billions* of forwarding rules

control-plane

time

100s of policies that have to be updated dynamically according to BGP

To scale, the SDX platform leverages *domain-specific knowledge*

data-plane

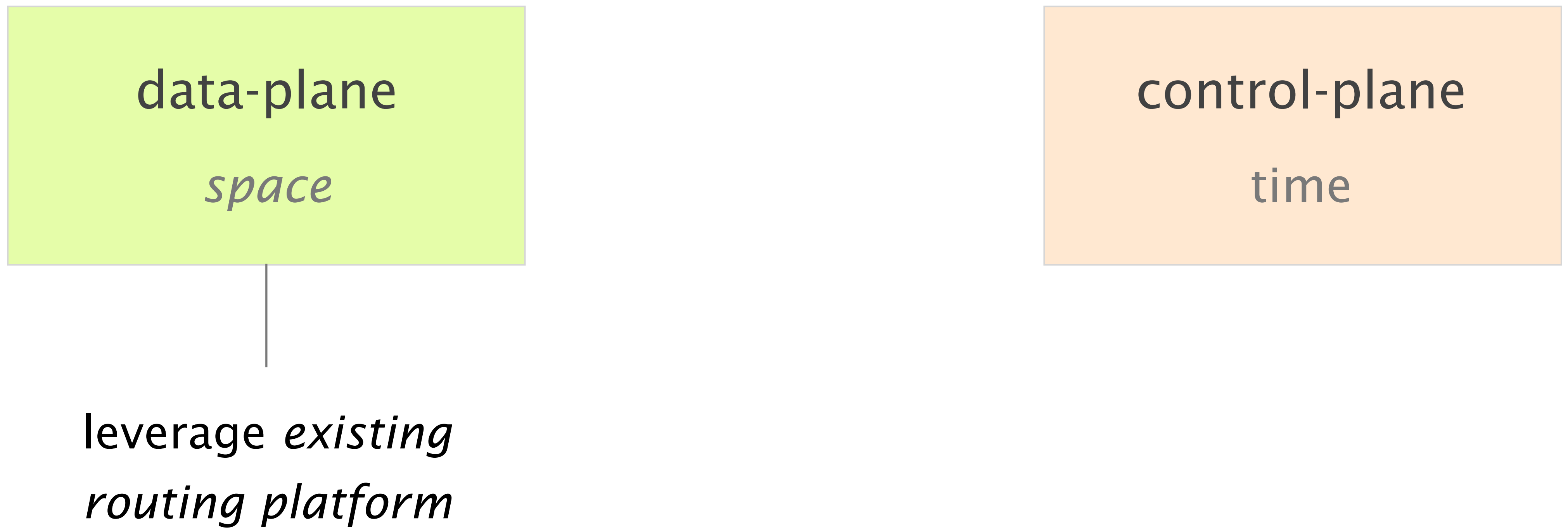
space

leverage *existing routing platform*

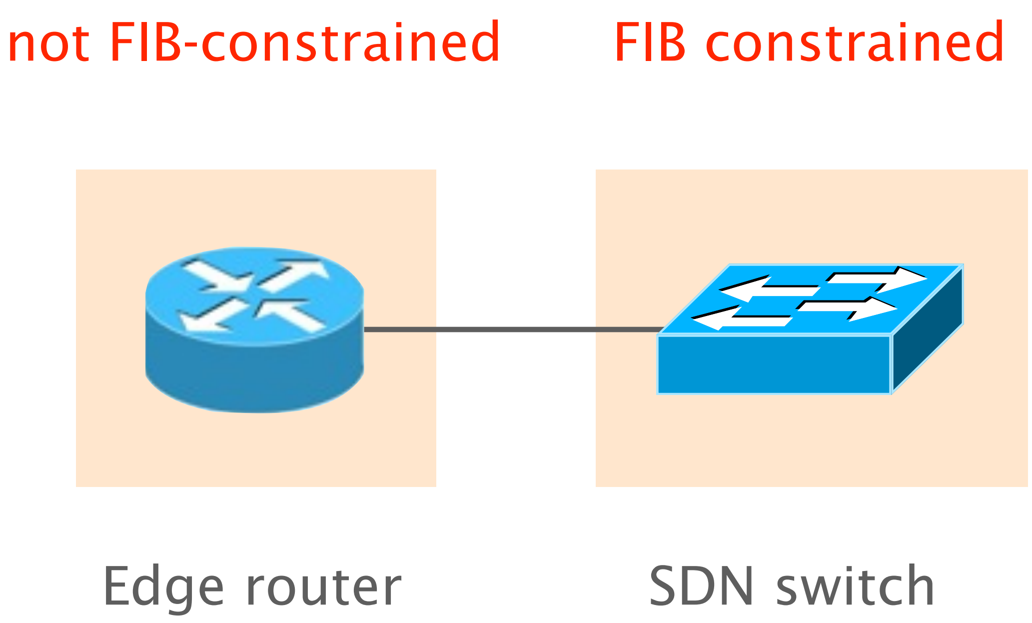
control-plane

time

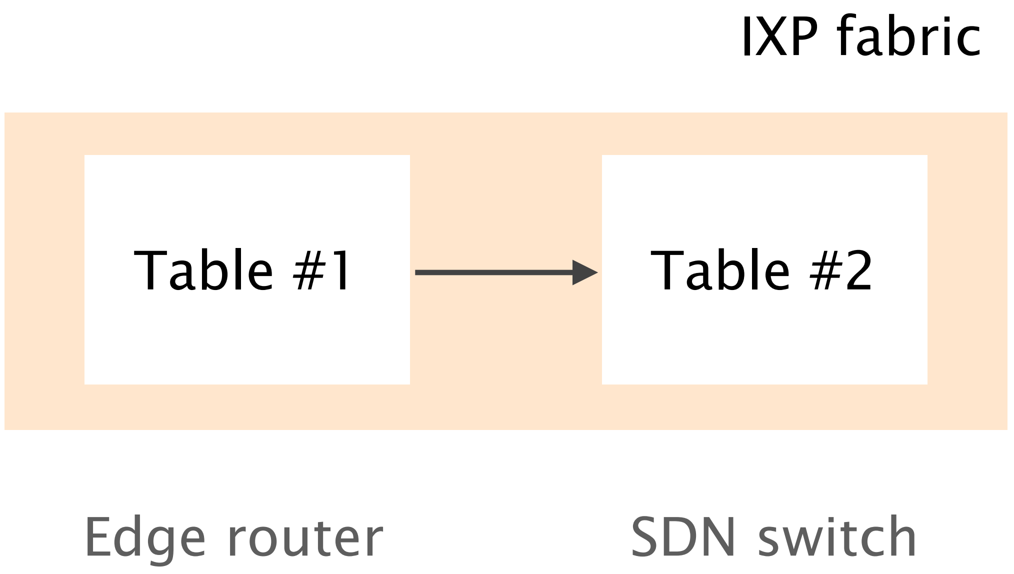
leverage *inherent policy structure*


data-plane
space
leverage existing routing platform
control-plane
time

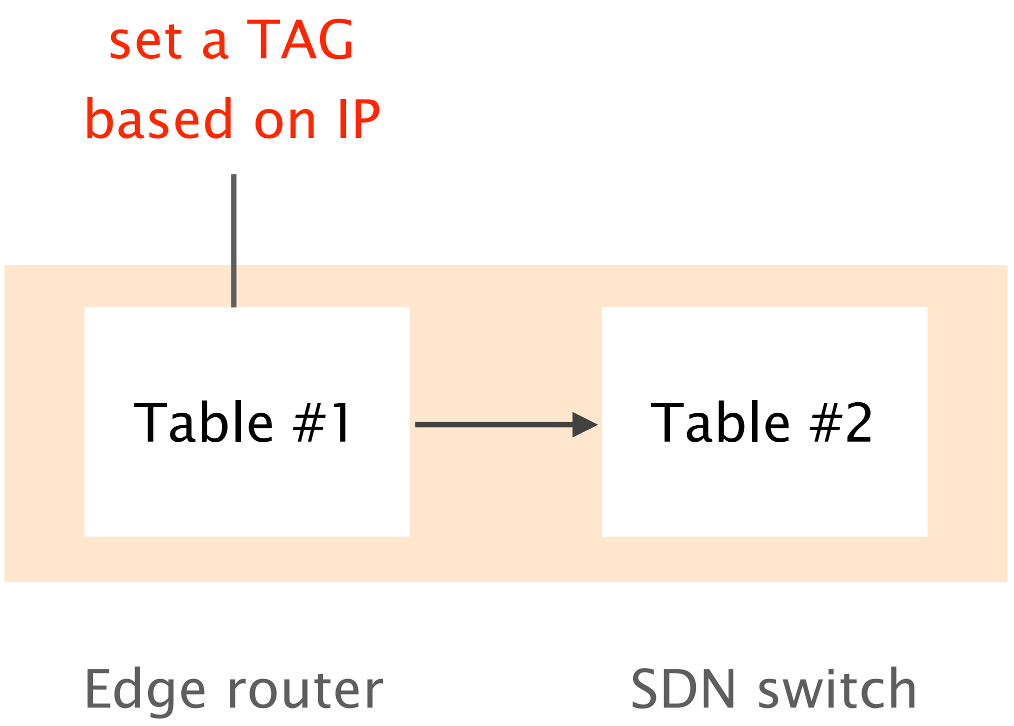
The edge routers, sitting next to the fabric,
are tailored to match on numerous IP prefixes



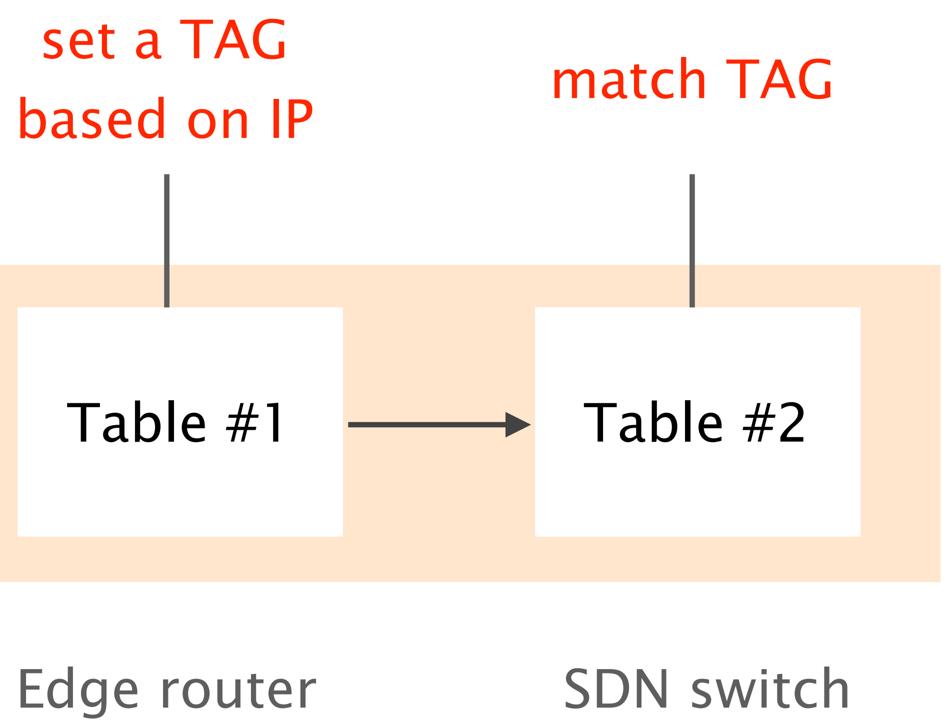
We consider routers FIB as the first stage of a multi-stage FIB



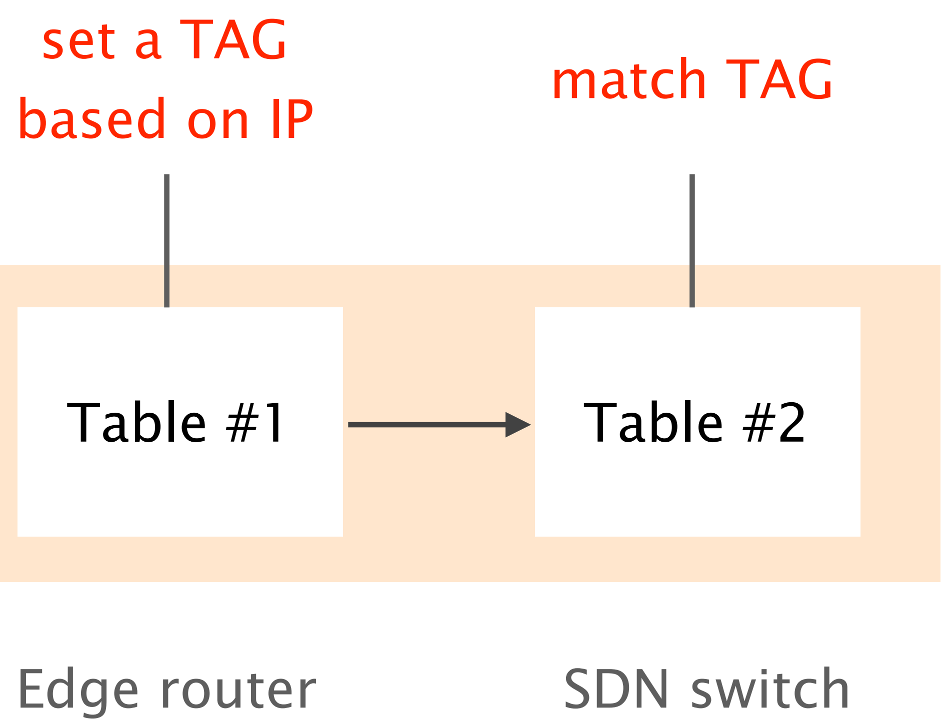
Routers FIB match on the destination prefix and set a tag accordingly



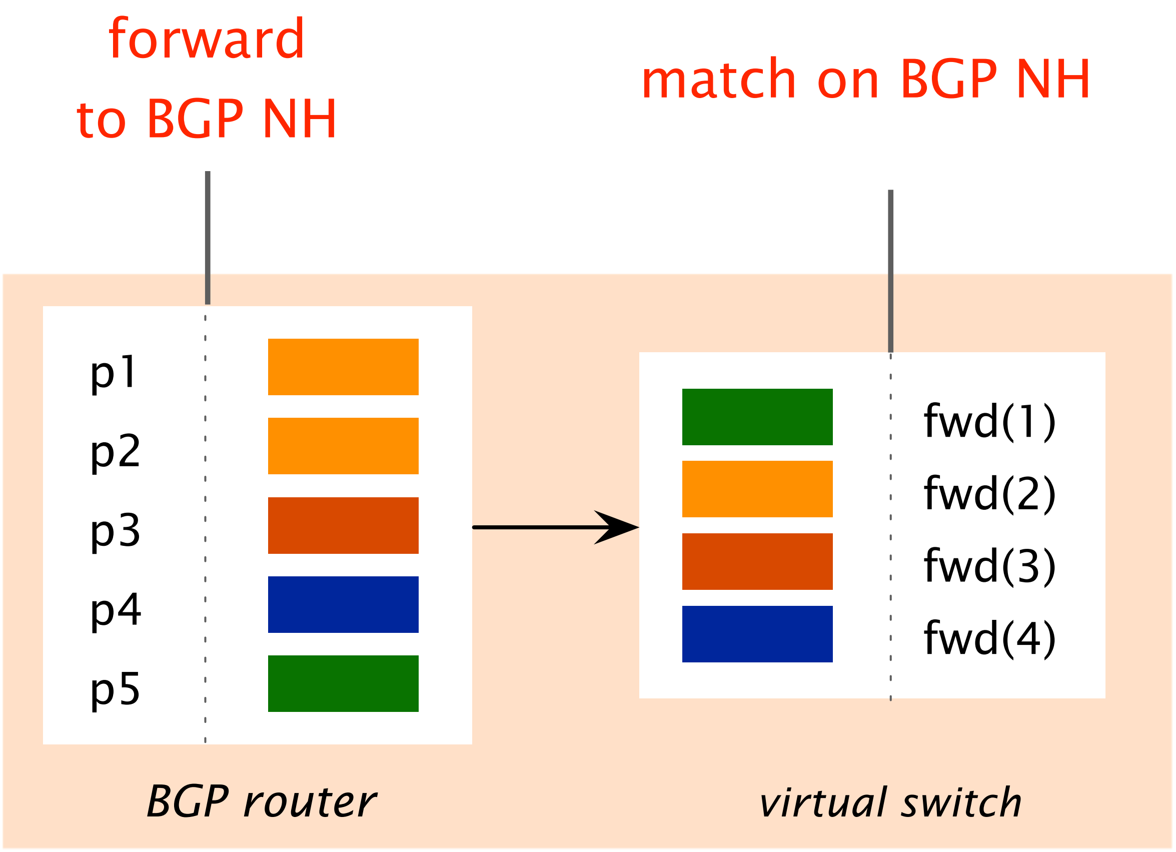
The SDN FIB matches on the tag, not on the IP prefixes



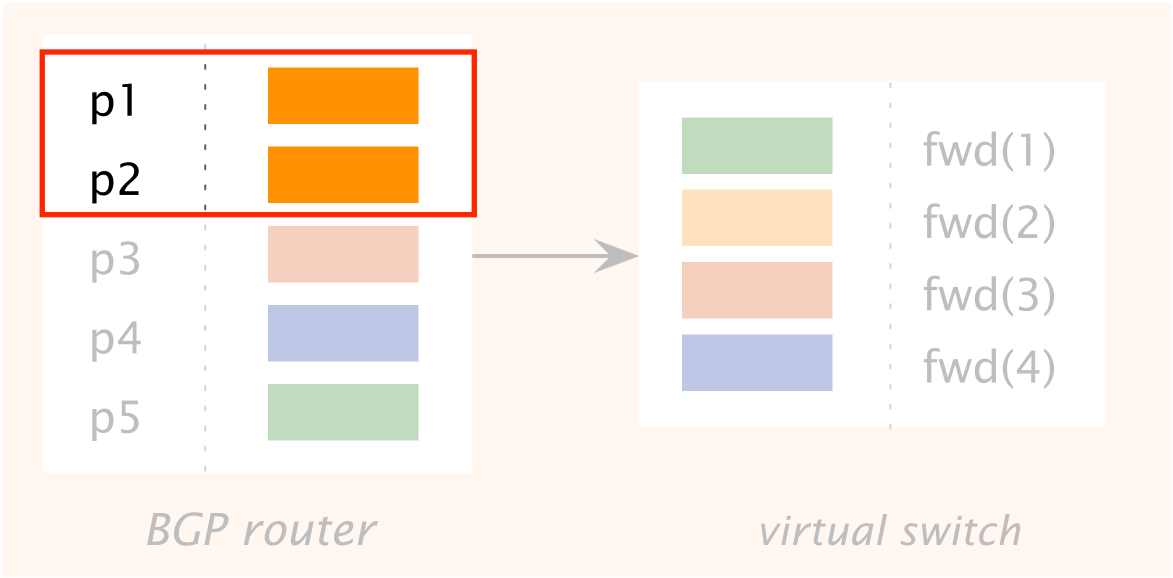
How do we provision tag entries in a router, and what are these tags?



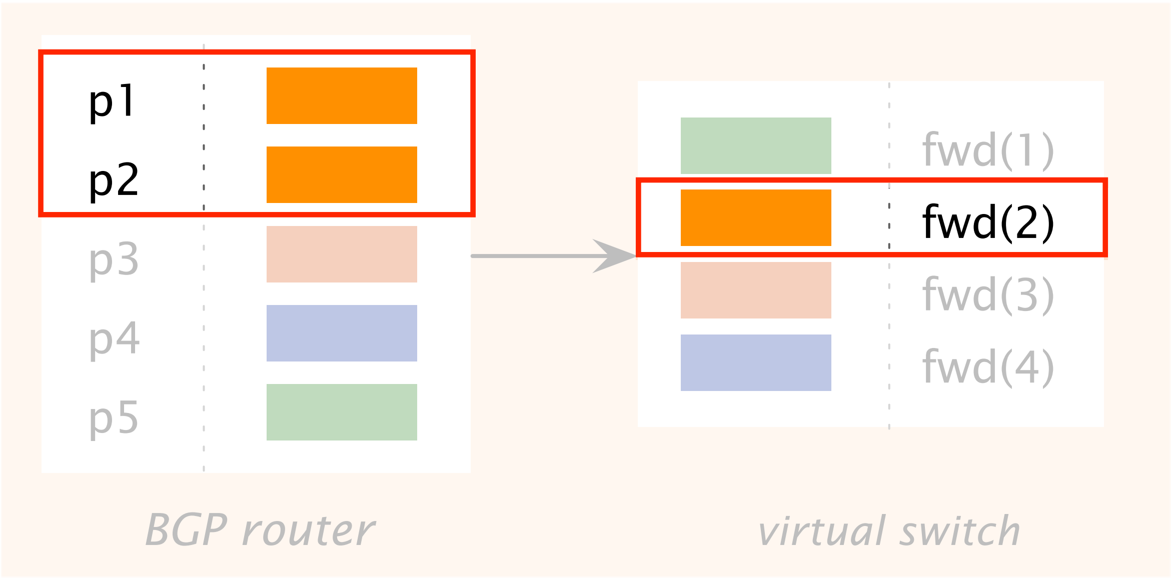
We use BGP as a provisioning interface and BGP next-hops as labels



All prefixes sharing the same forwarding behavior are grouped together using the same BGP next-hop

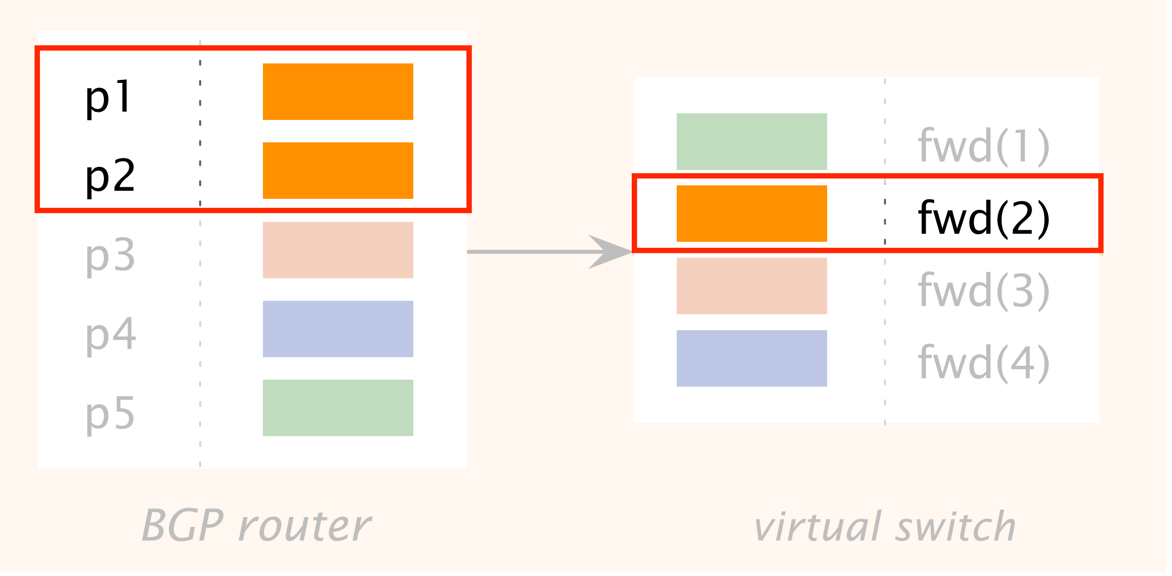


The SDX data-plane maintains one forwarding entry per prefix-group

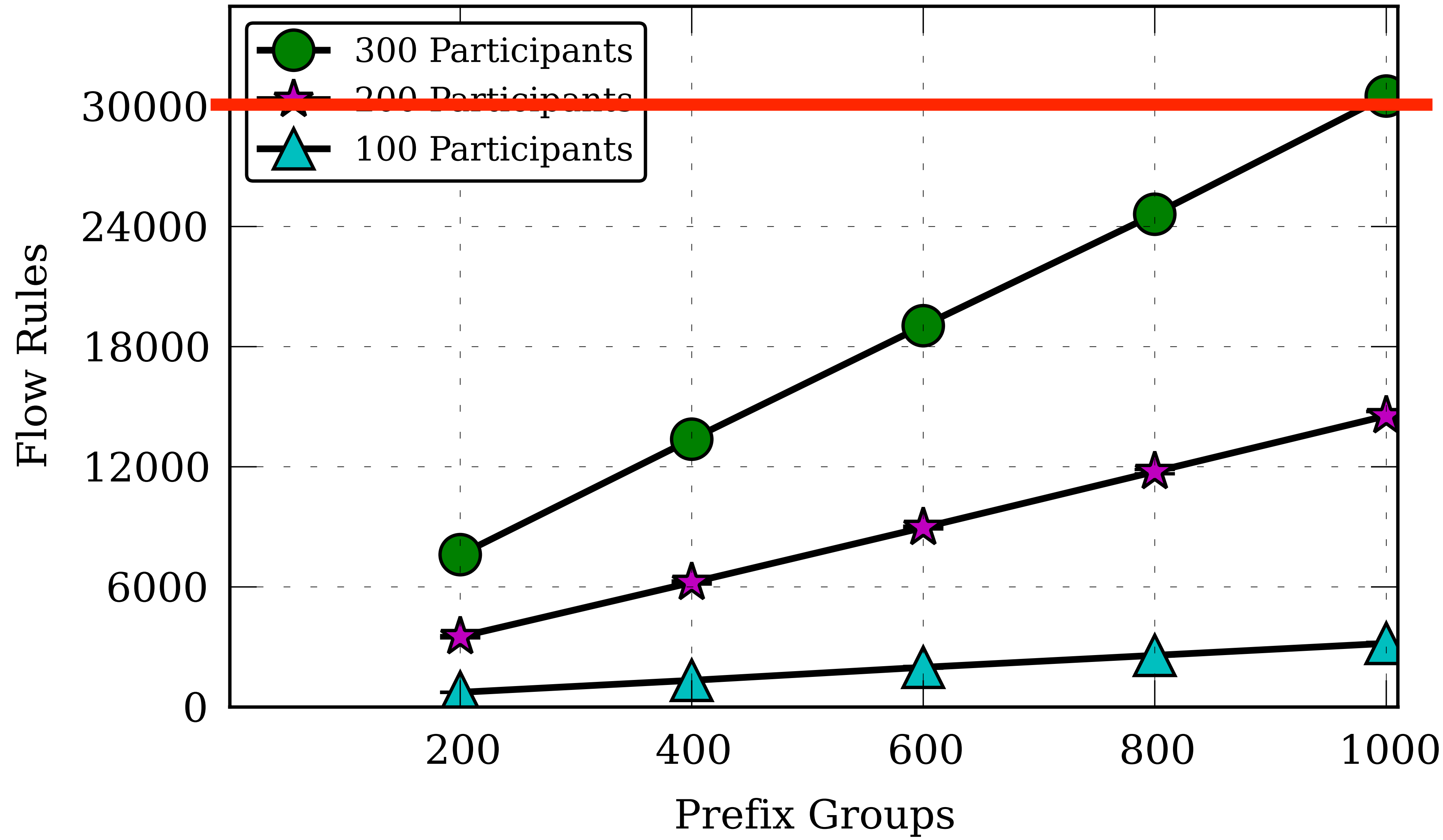


Data-plane utilization is reduced considerably as there are *way* more prefixes than prefixes groups

 # prefixes >> #prefixes groups



By leveraging BGP, the SDX can accommodate policies for hundreds of participants with less than 30k rules



data-plane

space

control-plane

time

leverage *inherent policy structure*

SDX policies exacerbate key characteristics that enable to speed-up compilation time considerably

Policies are often disjoint

Policy updates are local

Policy updates are bursty

SDX policies exacerbate key characteristics that enable to speed-up compilation time considerably

Policies are often disjoint

disjoint policy do not have to be composed together

significant gain as composing policies is time consuming

Policy updates are local

Policy updates are bursty

SDX policies exacerbate key characteristics that enable to speed-up compilation time considerably

Policies are often disjoint

Policy updates are local

Policy updates usually impact a few prefix-groups

75% of the updates affect no more than 3 prefixes

Policy updates are bursty

SDX policies exacerbate key characteristics that enable to speed-up compilation time considerably

Policies are often disjoint

Policy updates are local

Policy updates are bursty

policy changes are separated of large periode of inactivity

75% of the time, inter-arrival time between updates is at least 10s

The SDX controller adopts a two-staged compilation algorithm

Fast, but non-optimal algorithm upon updates

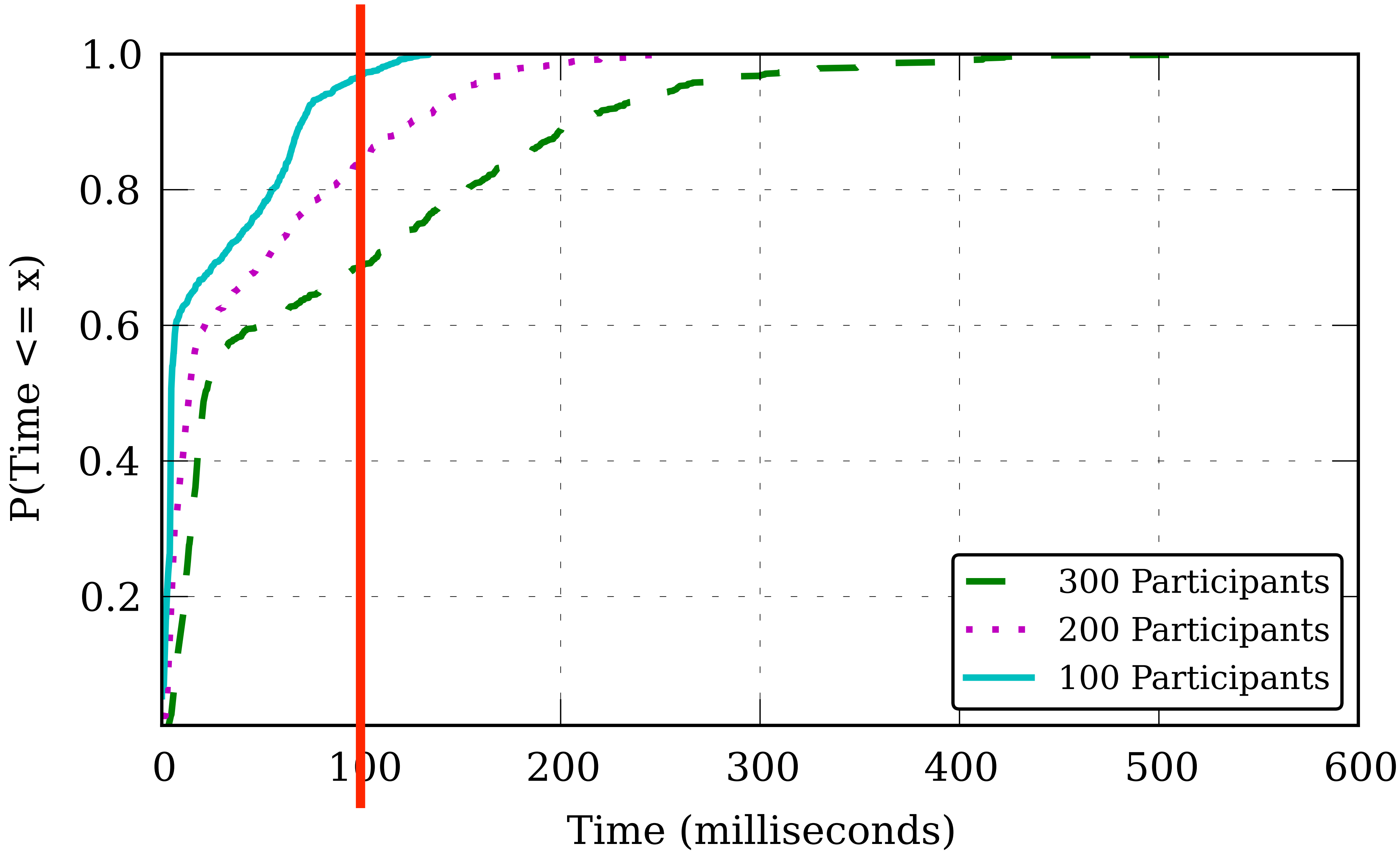
can create more rules than required

Slow, but optimal algorithm in background

recompute prefix groups

Time *vs* Space trade-off

In most cases, the SDX takes <100 ms
to recompute the global policy upon a BGP event



Novel Applications for a SDN-enabled Internet eXchange Point

Architecture

programming model

Scalability

control- & data-plane

3 Applications

inter domain bonanza

SDX enables a wide range of novel applications

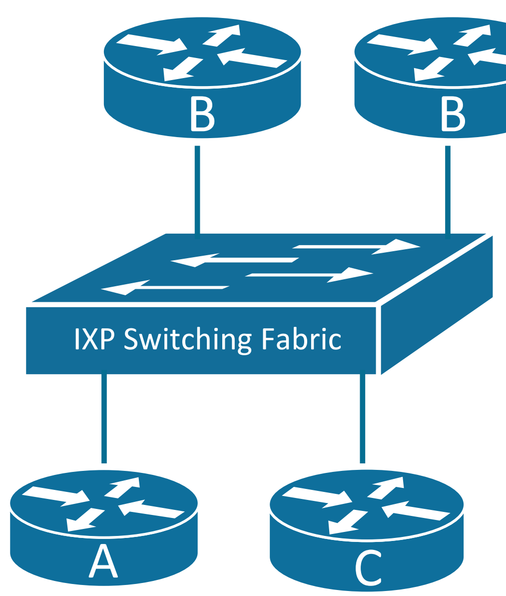
security	Prevent/block policy violation Prevent participants communication Upstream blocking of DoS attacks
forwarding optimization	Middlebox traffic steering Traffic offloading Inbound Traffic Engineering Fast convergence
peering	Application-specific peering
remote-control	Influence BGP path selection Wide-area load balancing

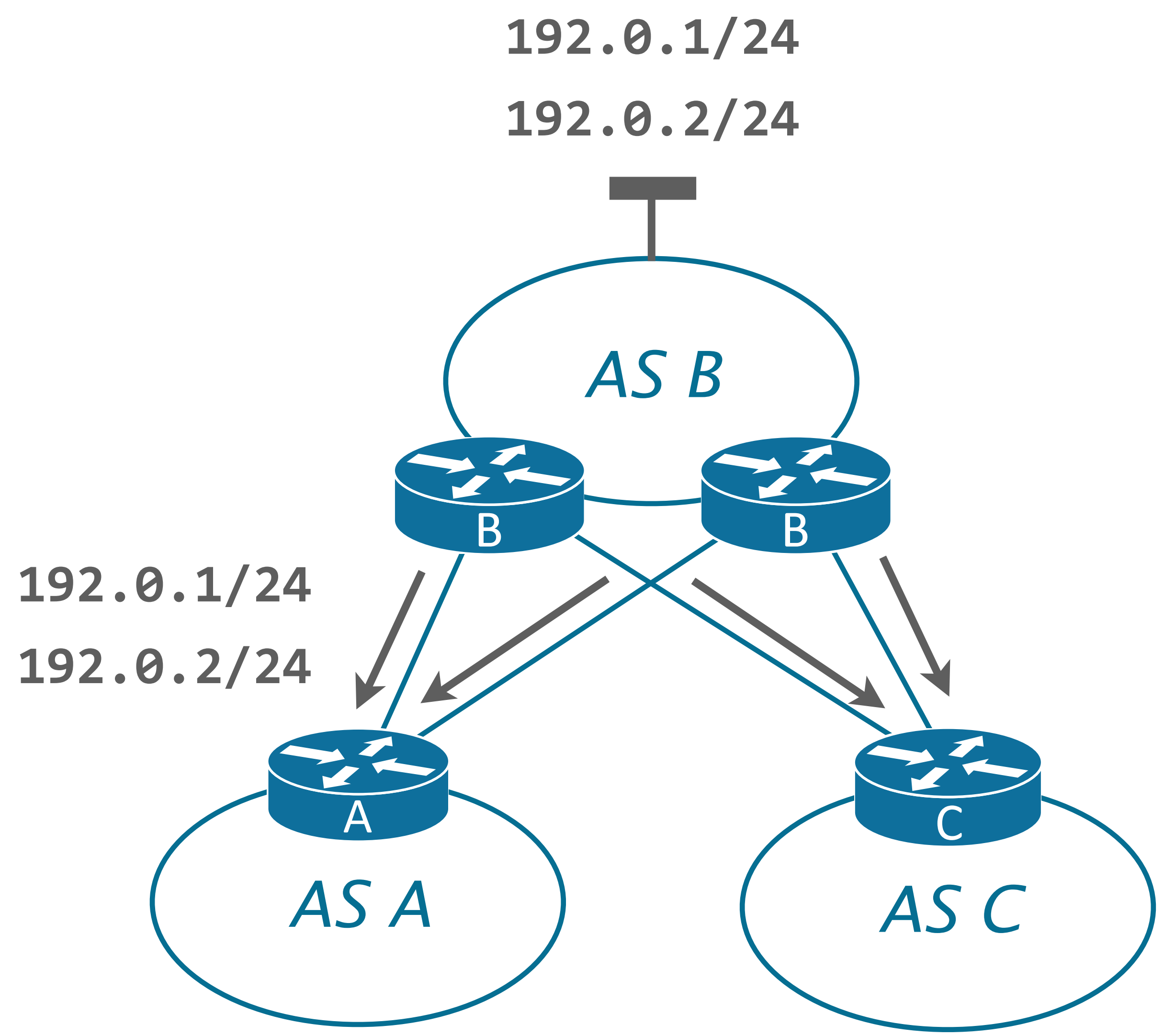
SDX enables a wide range of novel applications

security	Prevent/block policy violation Prevent participants communication Upstream blocking of DoS attacks
forwarding optimization	Middlebox traffic steering Traffic offloading **Inbound Traffic Engineering** Fast convergence
peering	Application-specific peering
remote-control	Influence BGP path selection Wide-area load balancing

SDX can improve inbound traffic engineering

Given an IXP Physical Topology and a BGP topology,

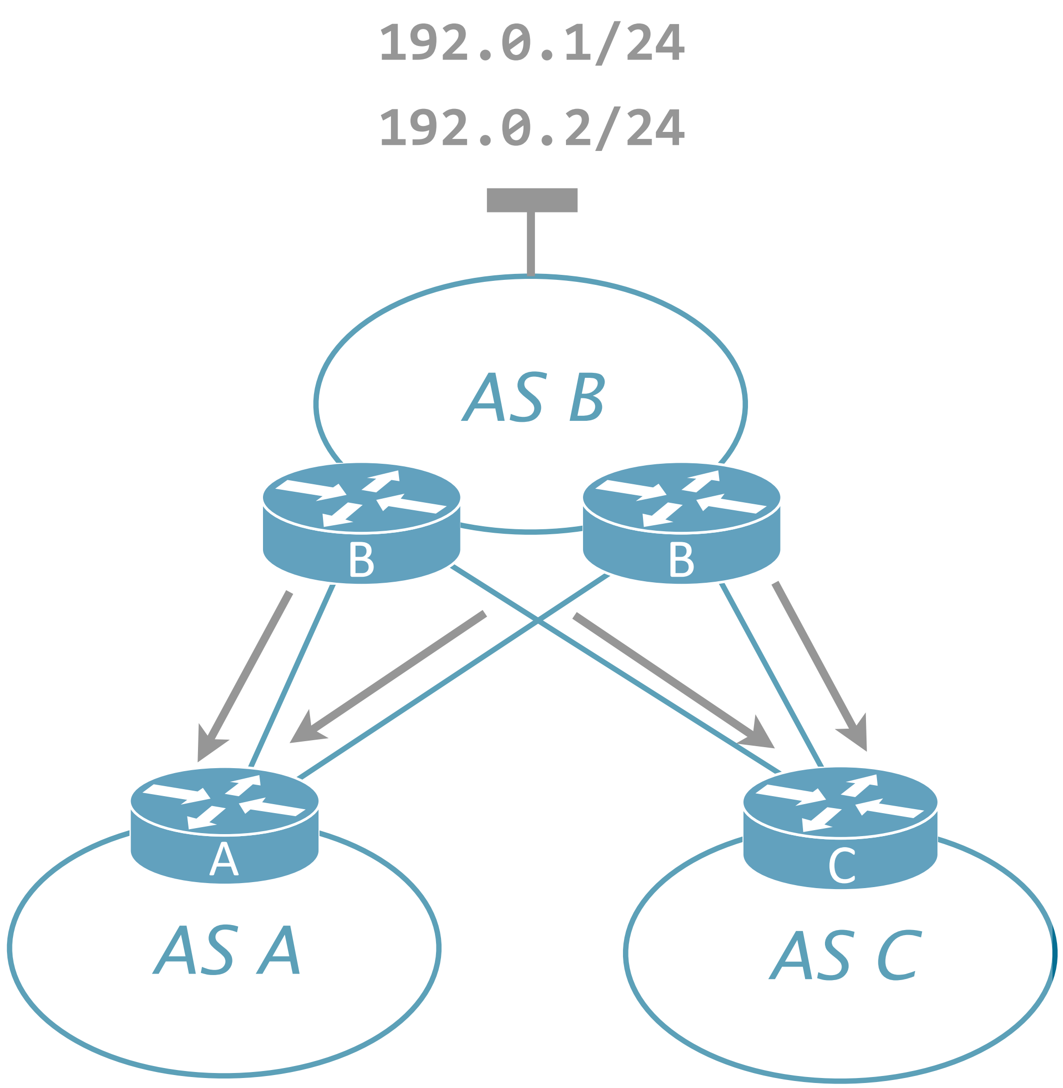




Given an IXP Physical Topology and a BGP topology, Implement B's inbound policies

B's inbound policies

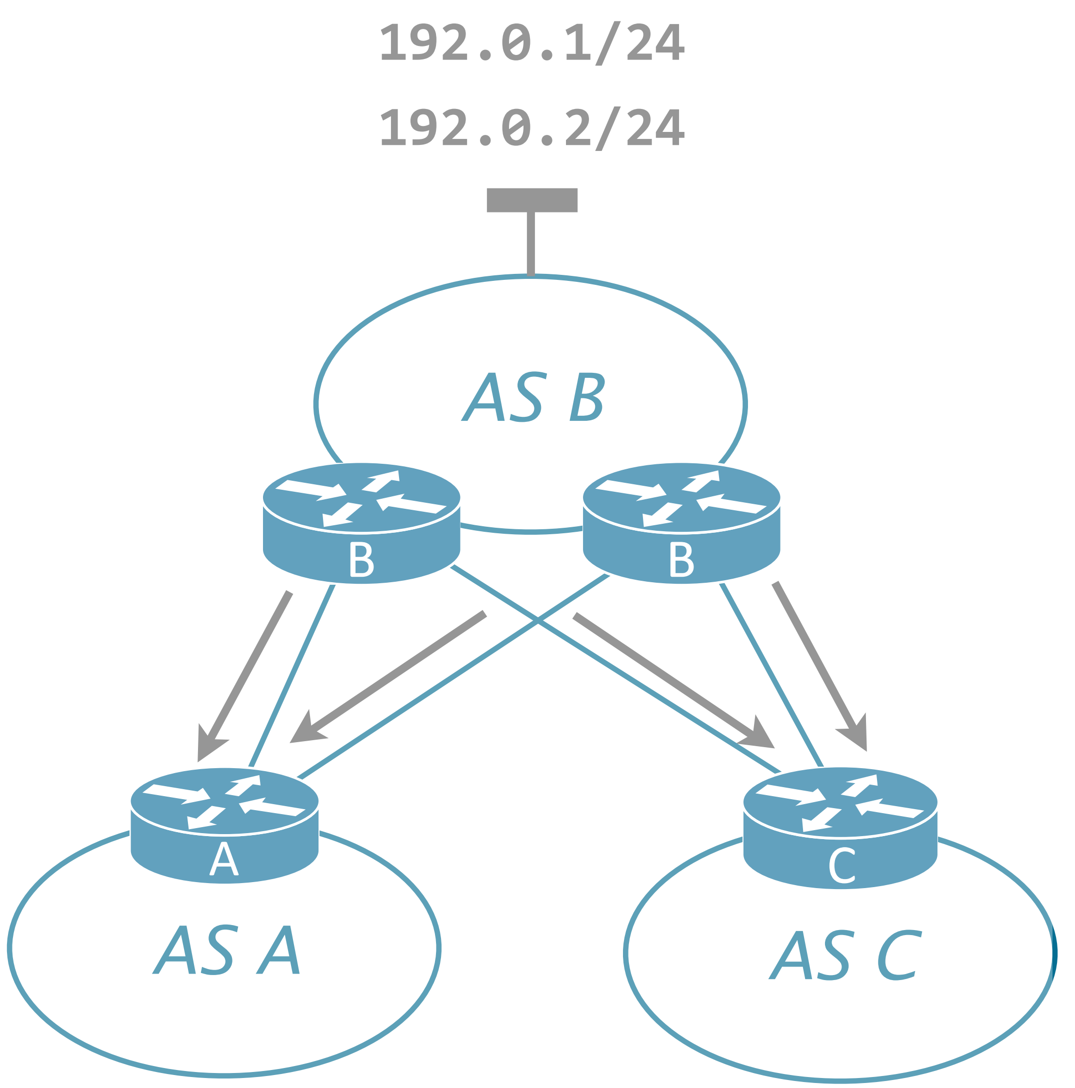
to	from	*receive on*
192.0.1/24	A	left
192.0.2/24	C	right
192.0.2/24	ATT_IP	right
192.0.1/24	*	right
192.0.2/24	*	left



How do you that with BGP?

B's inbound policies

to	from	*receive on*
192.0.1/24	A	left
192.0.2/24	C	right
192.0.2/24	ATT_IP	right
192.0.1/24	*	right
192.0.2/24	*	left



It is hard

BGP provides few knobs to influence remote decisions

Implementing such a policy is configuration-intensive

using AS-Path prepend, MED, community tagging, etc.

... and even impossible for some requirements

BGP policies **cannot** influence remote decisions based on source addresses

to	**from**	receive on
`192.0.2.0/24`	**`ATT_IP`**	right

In any case, the outcome is unpredictable

Implementing such a policy is configuration-intensive

using AS-Path prepend, MED, community tagging, etc.

There is *no guarantee* that remote parties will comply

one can only “influence” remote decisions

Networks engineers have no choice but to “try and see”

which makes it impossible to adapt to traffic pattern

With SDX, implement B's inbound policy is easy

SDX policies give any participant *direct* control on its forwarding paths

to	from	fwd
192.0.1/24	A	left
192.0.2/24	B	right
192.0.2/24	ATT_IP	right
192.0.1/24	*	right
192.0.2/24	*	left

B's SDX Policy

```
match(dstip=192.0.1/24, srcmac=A), fwd(L)
match(dstip=192.0.2/24, srcmac=B), fwd(R)
match(dstip=192.0.2/24, srcip=ATT), fwd(R)
match(dstip=192.0.1/24), fwd(R)
match(dstip=192.0.2/24), fwd(L)
```

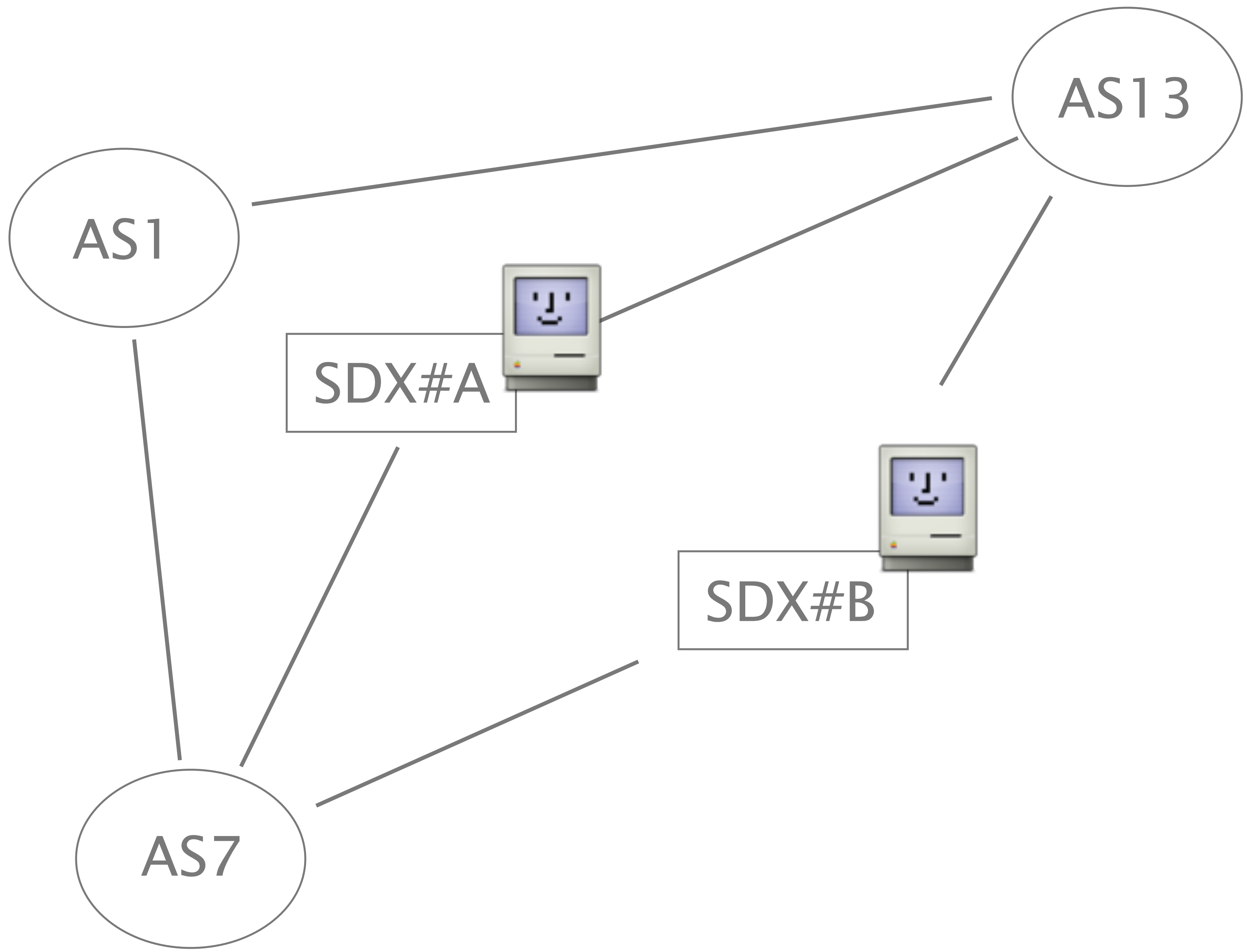
SDX enables a wide range of novel applications

security	Prevent/block policy violation Prevent participants communication Upstream blocking of DoS attacks
forwarding optimization	Middlebox traffic steering Traffic offloading Inbound Traffic Engineering Fast convergence
peering	Application-specific peering
remote-control	Influence BGP path selection Wide-area load balancing

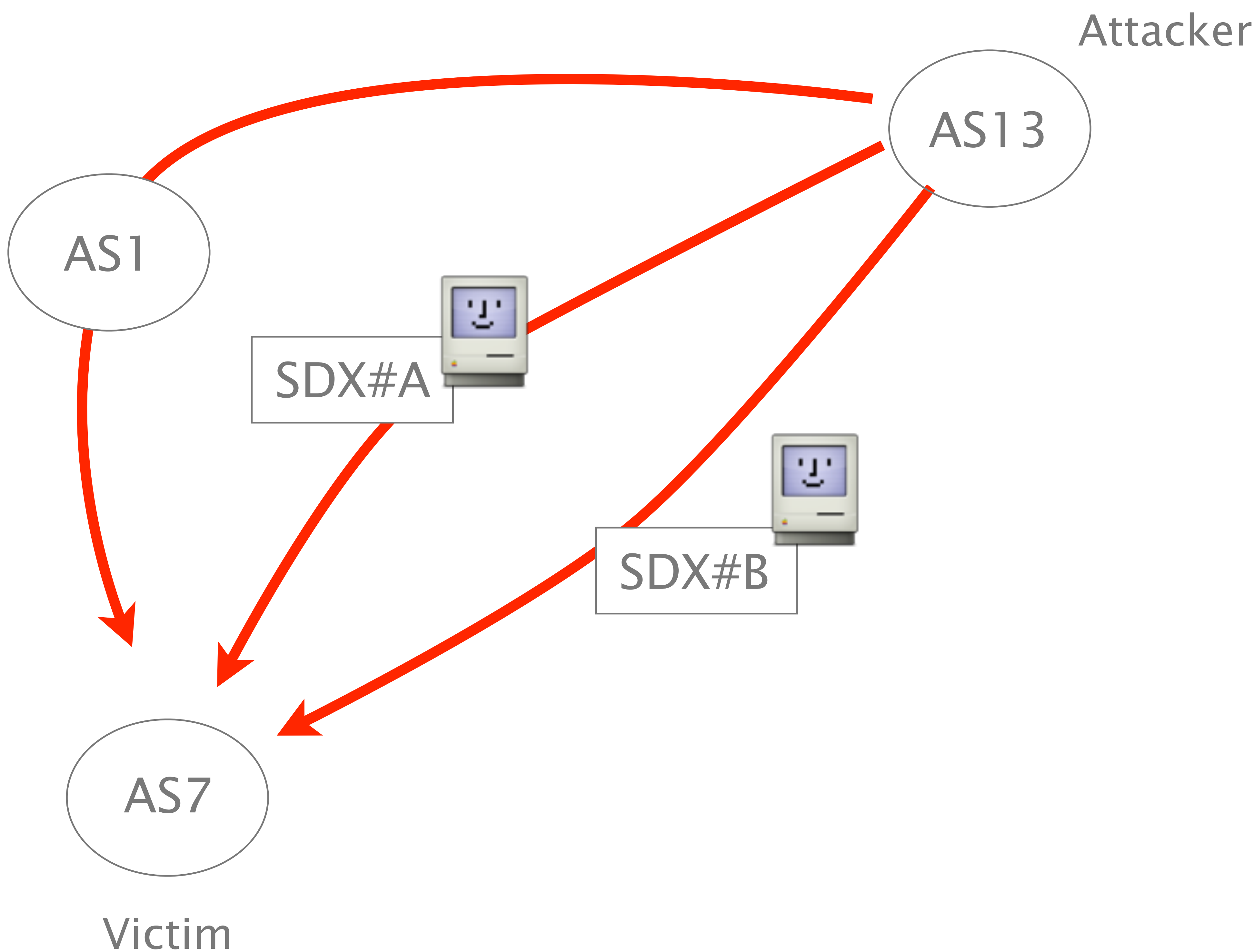
SDX enables a wide range of novel applications

security	Prevent/block policy violation Prevent participants communication **Upstream blocking of DoS attacks**
forwarding optimization	Middlebox traffic steering Traffic offloading Inbound Traffic Engineering Fast convergence
peering	Application-specific peering
remote-control	Influence BGP path selection Wide-area load balancing

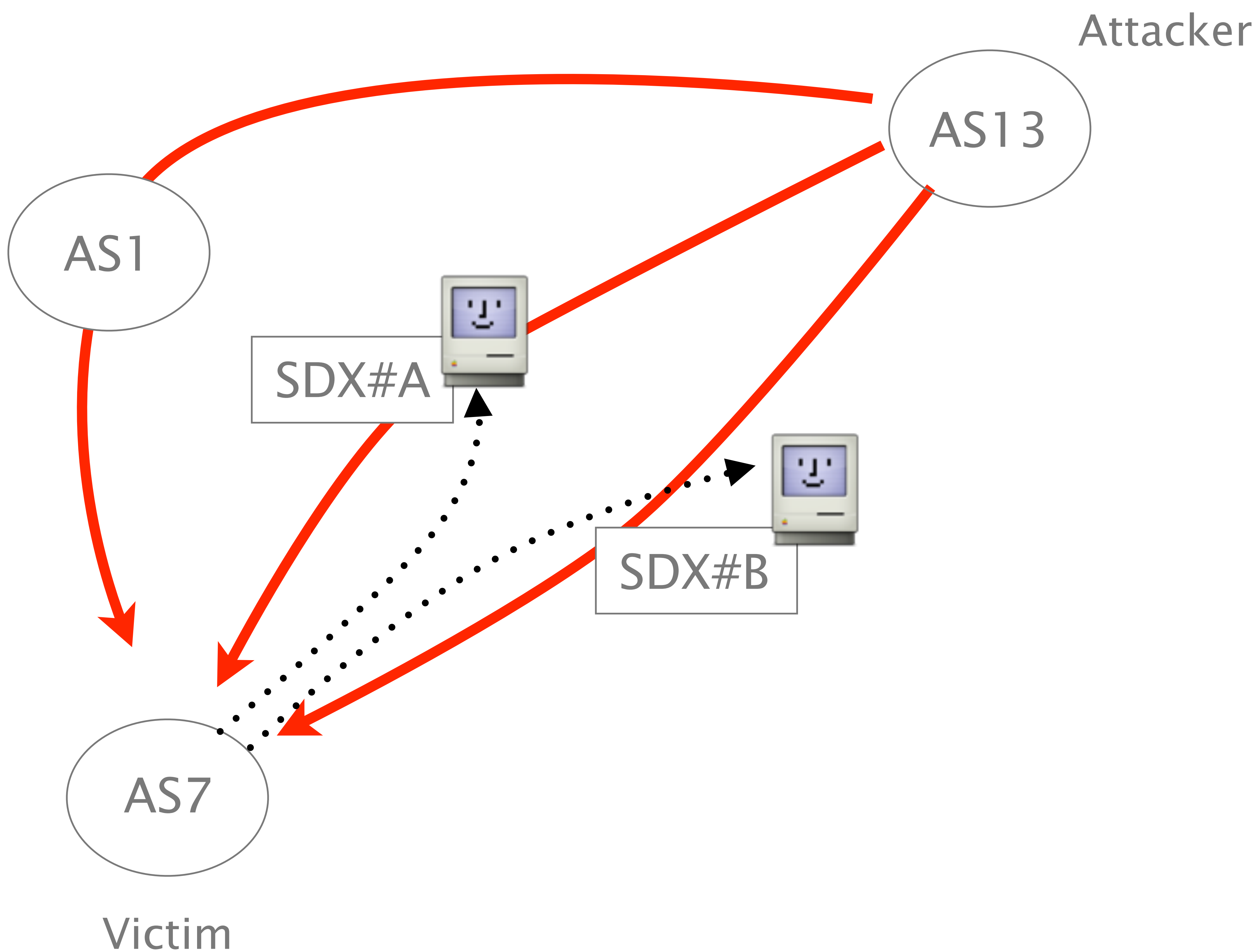
SDX can help in blocking DDoS attacks closer to the source



AS7 is victim of a DDoS attack originated from AS13



AS7 can remotely install *drop()* rule in the SDX platforms



```
match(srcip=Attacker/24, dstip=Victim/32) >> drop()
```

SDX: A Software Defined Internet Exchange

Architecture

programming model

Scalability

control- & data-plane

Applications

inter domain bonanza

Our SDX platform can serve as skeleton for a SDX ecosystem

We have running code (*)

with full BGP integration, check out our tutorial

We are in the process of having a first deployment

SNAP @ ColoATL, planned deployment with GENI

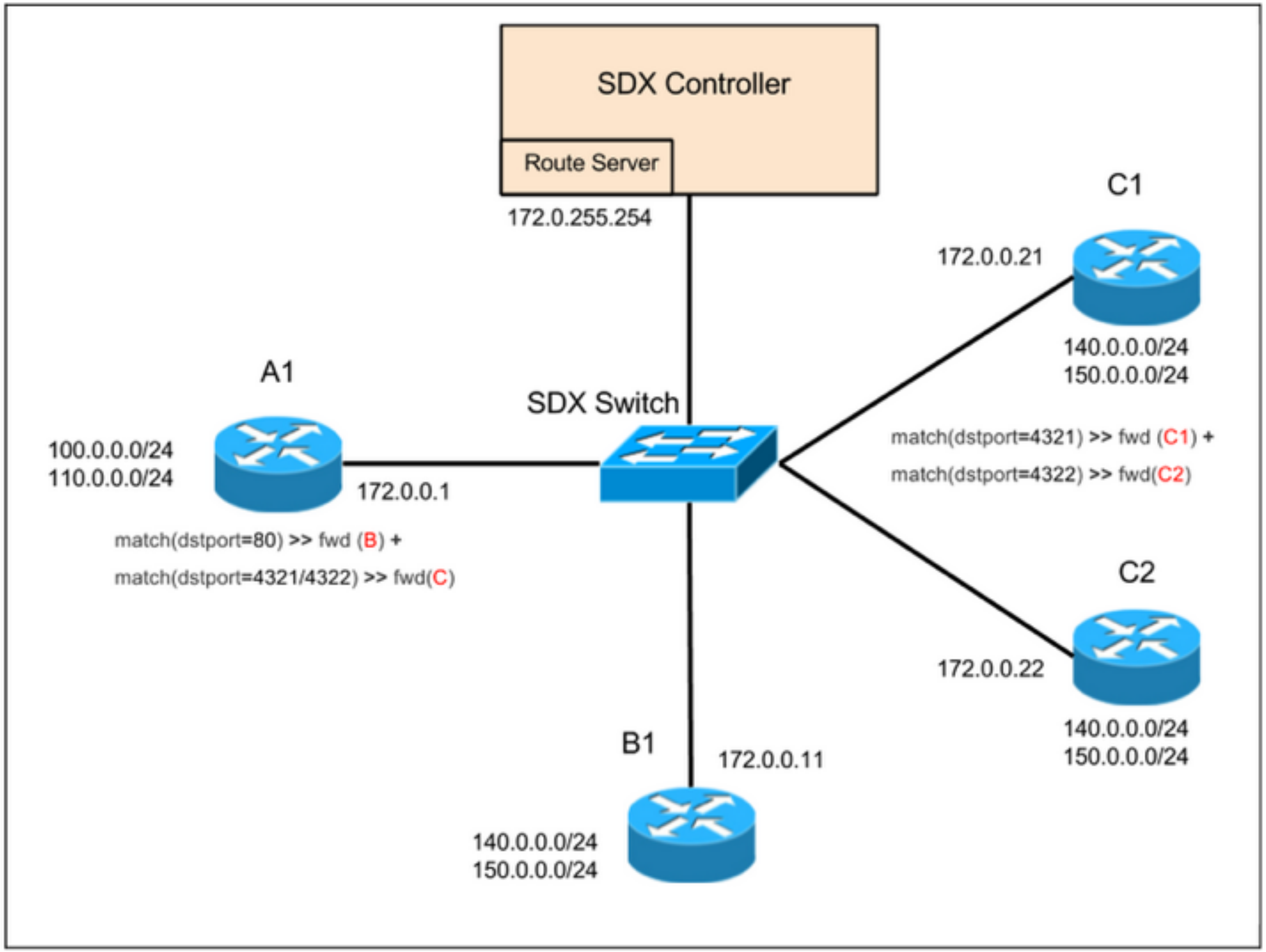
Many interested parties already

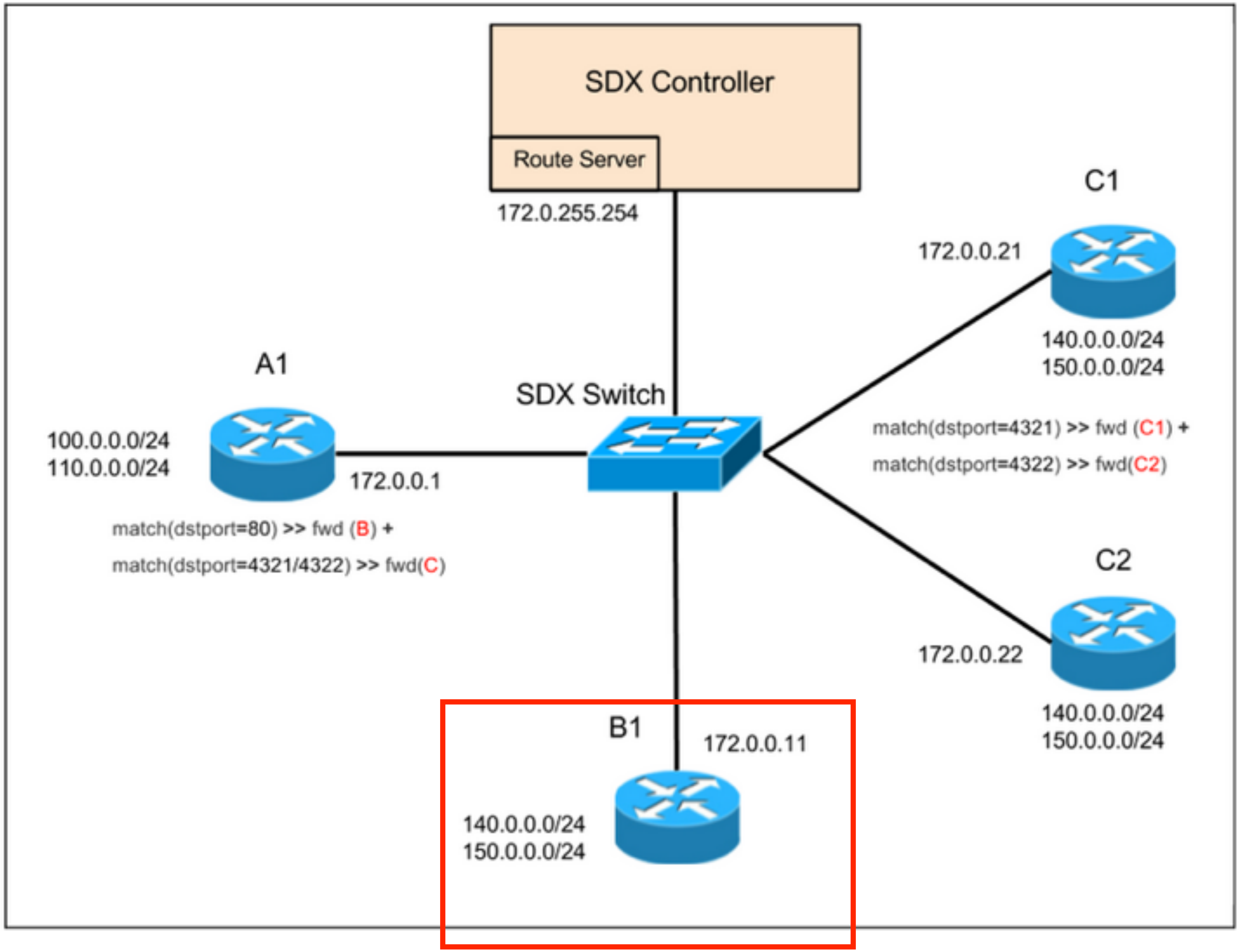
important potential for impact

(*) https://github.com/sdn-ixp/sdx/wiki

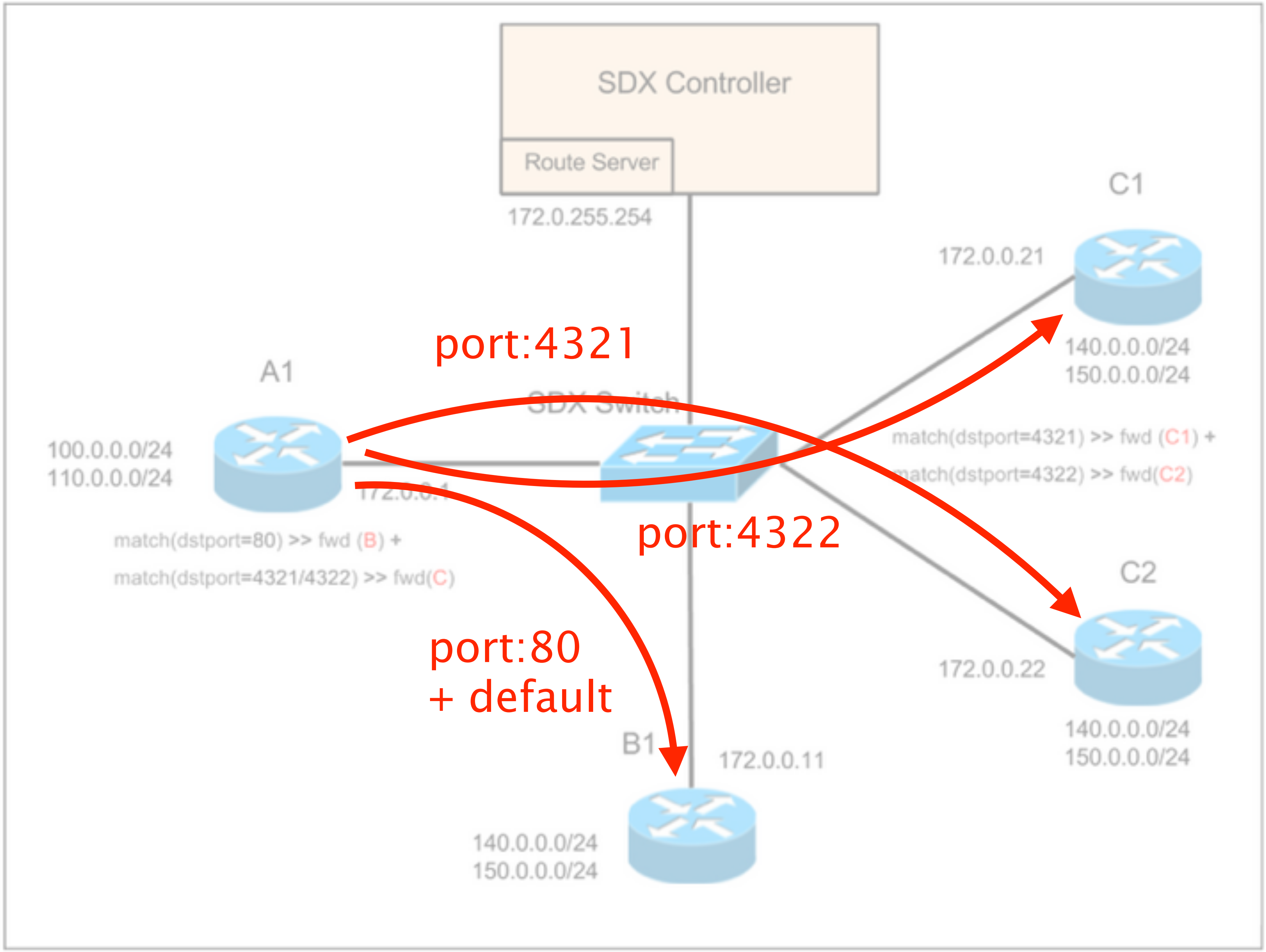
Demonstration

https://github.com/sdn-ixp/sdx/wiki


SDX Controller
Route Server
172.0.255.254
C1
172.0.0.21
140.0.0.0/24
150.0.0.0/24
A1
SDX Switch
100.0.0.0/24
110.0.0.0/24
172.0.0.1
match(dstport=4321) >> fwd (C1) +
match(dstport=4322) >> fwd(C2)
match(dstport=80) >> fwd (B) +
match(dstport=4321/4322) >> fwd(C)
C2
172.0.0.22
140.0.0.0/24
150.0.0.0/24
B1
172.0.0.11
140.0.0.0/24
150.0.0.0/24



BGP picked routes


SDX Controller
Route Server
172.0.255.254
C1
172.0.0.21
140.0.0.0/24
150.0.0.0/24
port:4321
A1
SDX Switch
100.0.0.0/24
110.0.0.0/24
172.0.0.1
match(dstport=4321) >> fwd (C1) +
match(dstport=4322) >> fwd(C2)
port:4322
match(dstport=80) >> fwd (B) +
match(dstport=4321/4322) >> fwd(C)
C2
port:80
+ default
172.0.0.22
140.0.0.0/24
150.0.0.0/24
B1
172.0.0.11
140.0.0.0/24
150.0.0.0/24

SDX: A Software Defined Internet Exchange

Laurent Vanbever

www.vanbever.eu

FGRE Workshop (Ghent, iMinds)

July, 8 2014