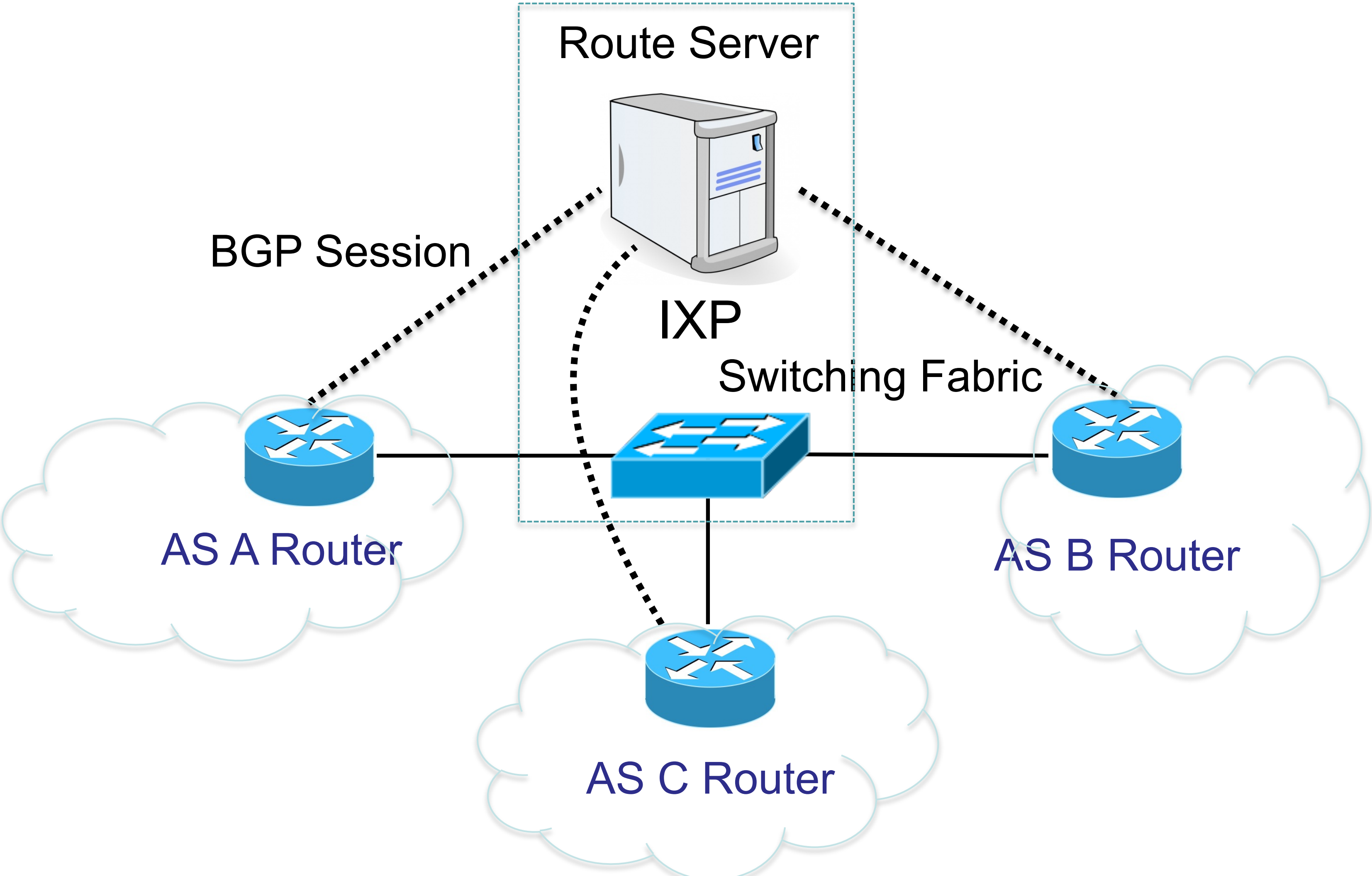
iSDX: An Industrial-Scale Software-Defined IXP

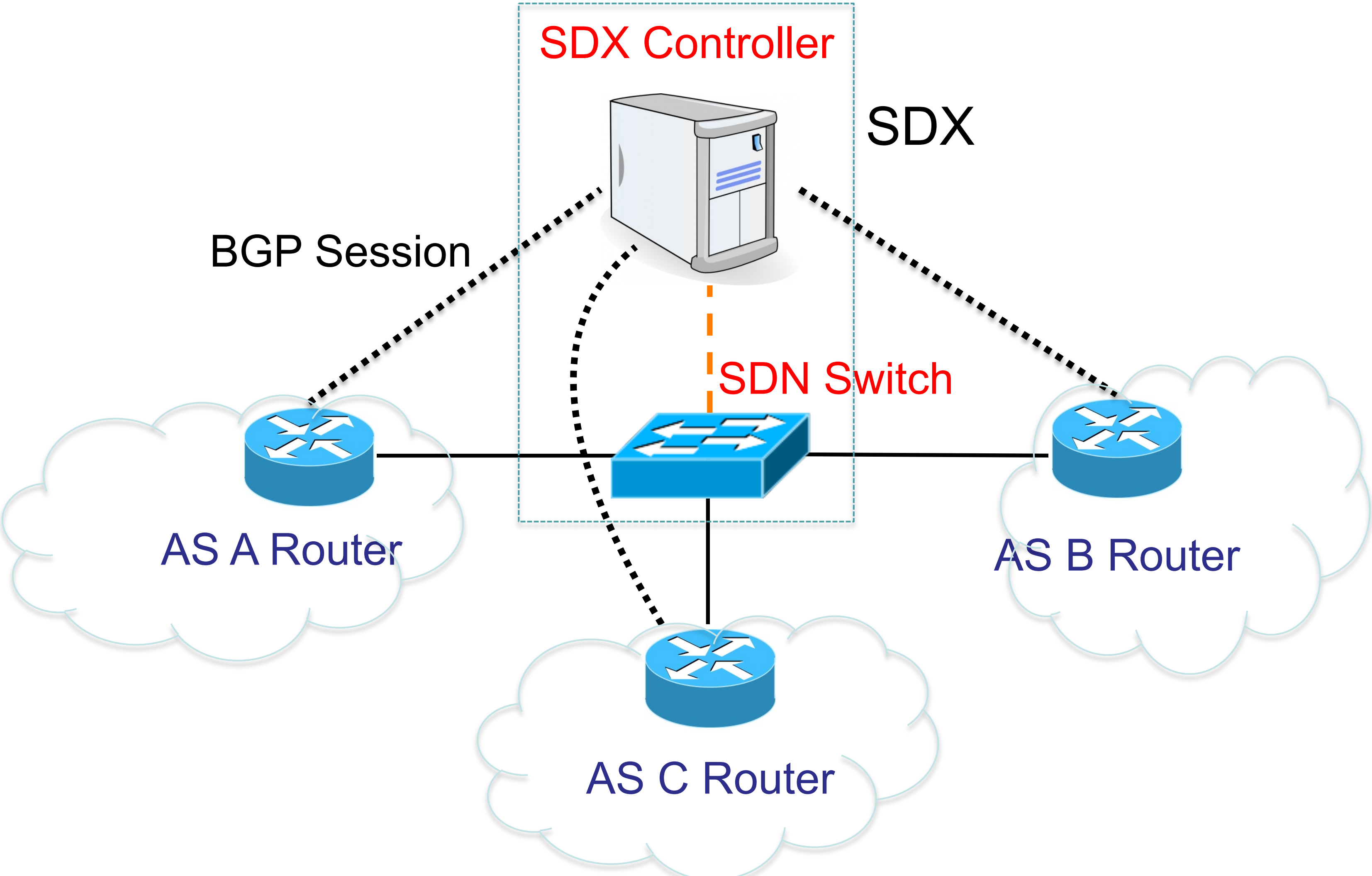
Nick Feamster
Princeton University
http://sdx.cs.princeton.edu/

Arpit Gupta, Robert MacDavid, Rüdiger Birkner,
Marco Canini, Jennifer Rexford, Laurent Vanbever

Internet Exchange Points (IXPs)



Software Defined IXPs (SDXs)



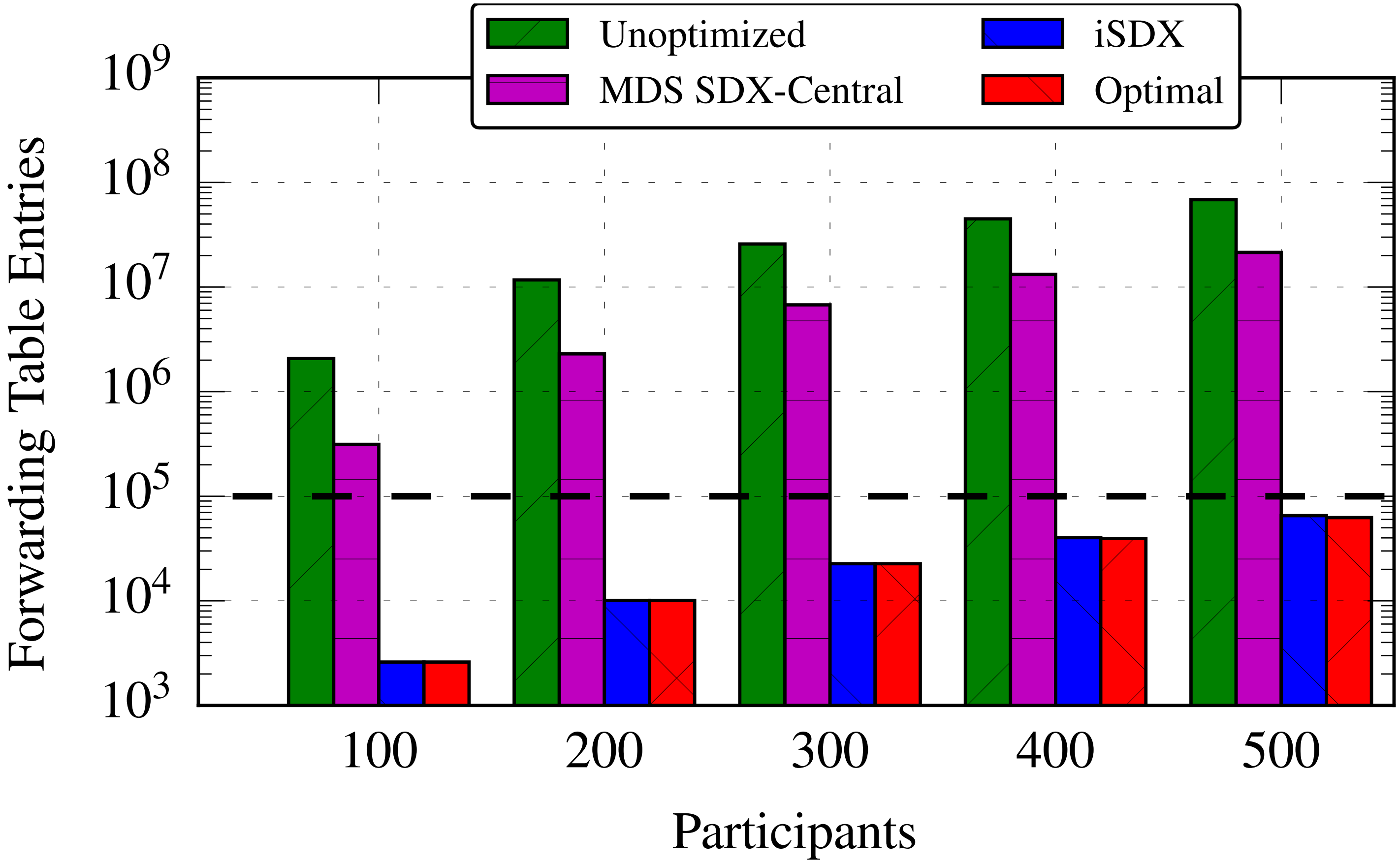
SDX Creates New Possibilities

- More flexible **business relationships**
  - Make peering decisions based on time of day, volume of traffic & nature of application

- More direct & flexible **traffic control**
  - Define fine-grained traffic engineering policies

- Better **security**
  - Prefer “more secure” routes
  - Automatically black hole attack traffic

Three Years of Research:
We Can Now Support Industry Scale



BGP routes and updates for large EU IXP in a commodity hardware switch.

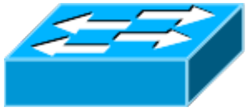
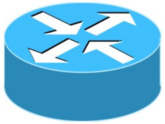
iSDX Evaluation: Summary

- **Data Plane State:**
  - Requires **65K < 100K** forwarding table entries

- **Data Plane Update Rate:**
  - Requires **0** < **2500** updates/second

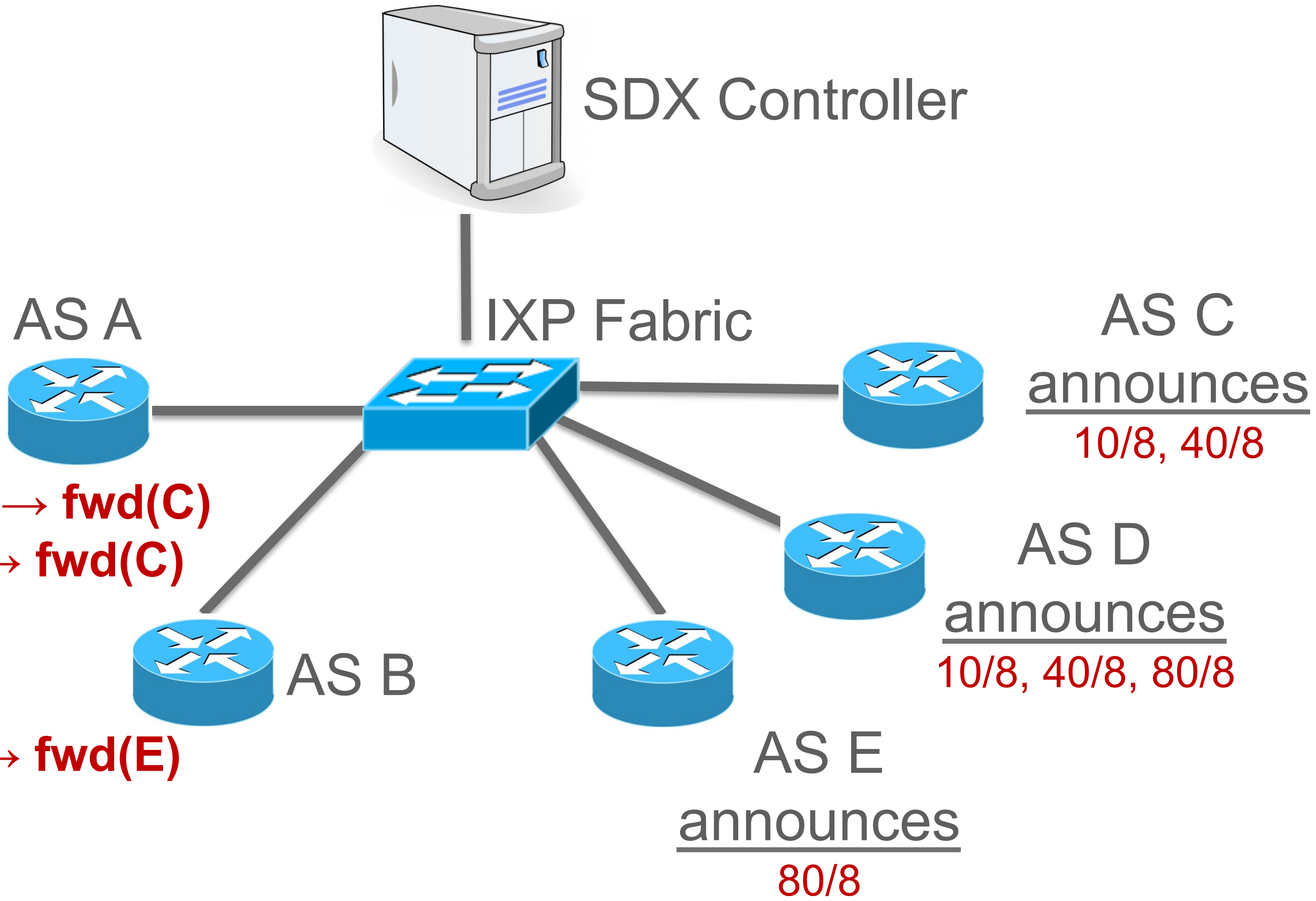
- **Other Goals:**
  - Processes BGP update bursts in real time **(50 ms)**
  - Requires only **360 BGP Next Hops** compared to 25K from previous solutions

Constraints (and Insight)

Devices	Operations	Data Plane Performance	
		State (# entries)	Update Rate (flow-mods/s)
	Match-Action on Multiple Headers	100K	2,500
	Matches on IP Prefixes only	~1M	N/A

Insight: Optimize the use of available resources on each device.

Simple Example



SDX Controller

IXP Fabric

AS A

AS B

AS C announces 10/8, 40/8

AS D announces 10/8, 40/8, 80/8

AS E announces 80/8

dPort = 443 → fwd(C)
dPort = 22 → fwd(C)

dPort = 80 → fwd(E)

Forwarding Table Entries at SDX

Number of forwarding table entries for
A & B’s Outbound SDN Policies

SDN Policies	# Forwarding Table Entries
dPort = 443 → fwd(C)	1
dPort = 22 → fwd(C)	1
dPort = 80 → fwd(E)	1

AS A

AS B

Number of Forwarding Entries

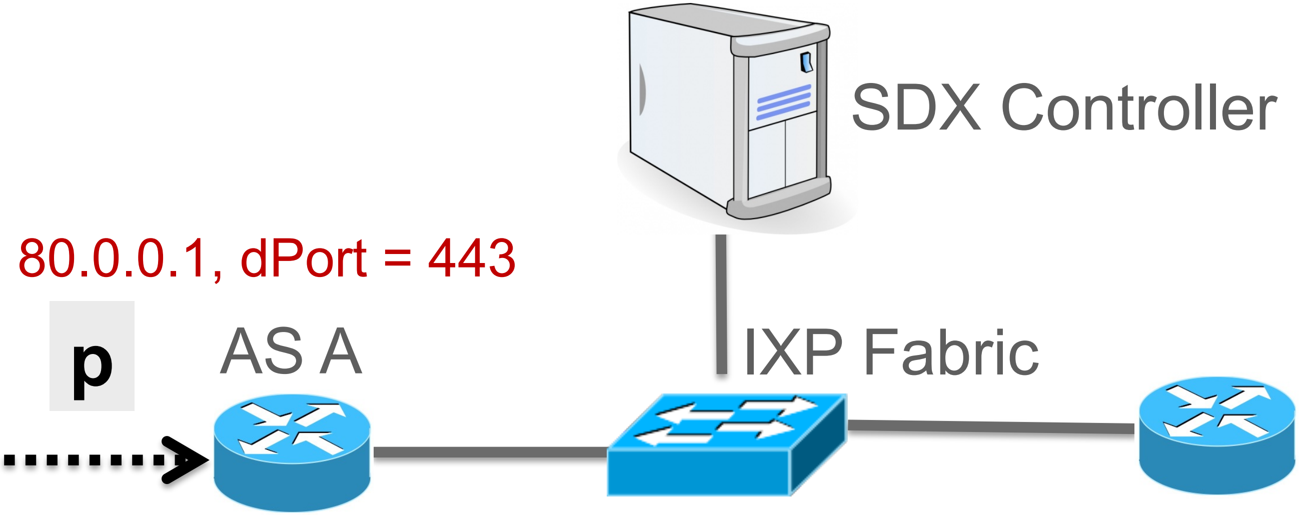
	Simple Example	Large IXP
Baseline	**3**	**62K**

- **Data from Large IXP:**
  - BGP RIBs & Updates from 511 participants
  - 96 million peering routes for 300K IP prefixes
  - 25K BGP updates for 2-hour duration

Scales, but is not congruent with BGP!

Congruence with BGP Policies

Problem: Need to ensure **p** is not forwarded to C.



SDX Controller

dIP = 80.0.0.1, dPort = 443

p

AS A

IXP Fabric

AS C

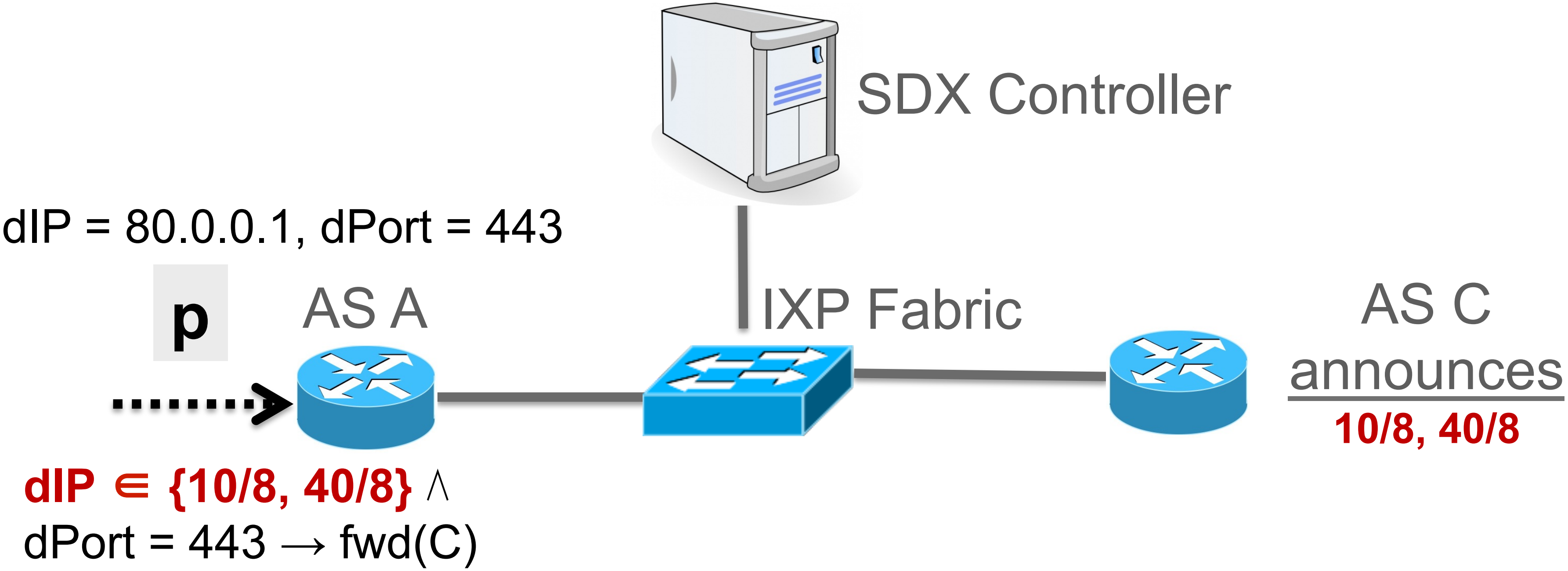
announces

10/8, 40/8

dPort = 443 → fwd(C)

Solution: SDN Policy Augmentation

Match on prefixes advertised by C.



SDX Controller

dIP = 80.0.0.1, dPort = 443

p

AS A

IXP Fabric

AS C

announces

10/8, 40/8

dIP ∈ {10/8, 40/8} ∧
dPort = 443 → fwd(C)

Data Plane State Explosion!

SDN Policies	# Forwarding Table Entries			
	10/8	40/8	80/8	
dPort = 443 → fwd(C)	1	1	0	4
dPort = 22 → fwd(C)	1	1	0	
dPort = 443 → fwd(D)	1	1	1	3

SDN policy augmentation increases forwarding table entries.

Number of Forwarding Entries

	Simple Example	**Large IXP**
Baseline	3	62K
Policy Augmentation	**7**	**68M**

Cannot support
forwarding table entries and update rate.

Three Insights (and Optimizations)

- Many prefix, policy combinations have exactly the same forwarding decision
  - **Optimization:** Forwarding equivalence
- Per-participant forwarding decisions have even more commonality
  - **Optimization:** Independent forwarding equivalence
- Advertisements can be encoded as FEC entries
  - **Optimization:** Reachability encoding

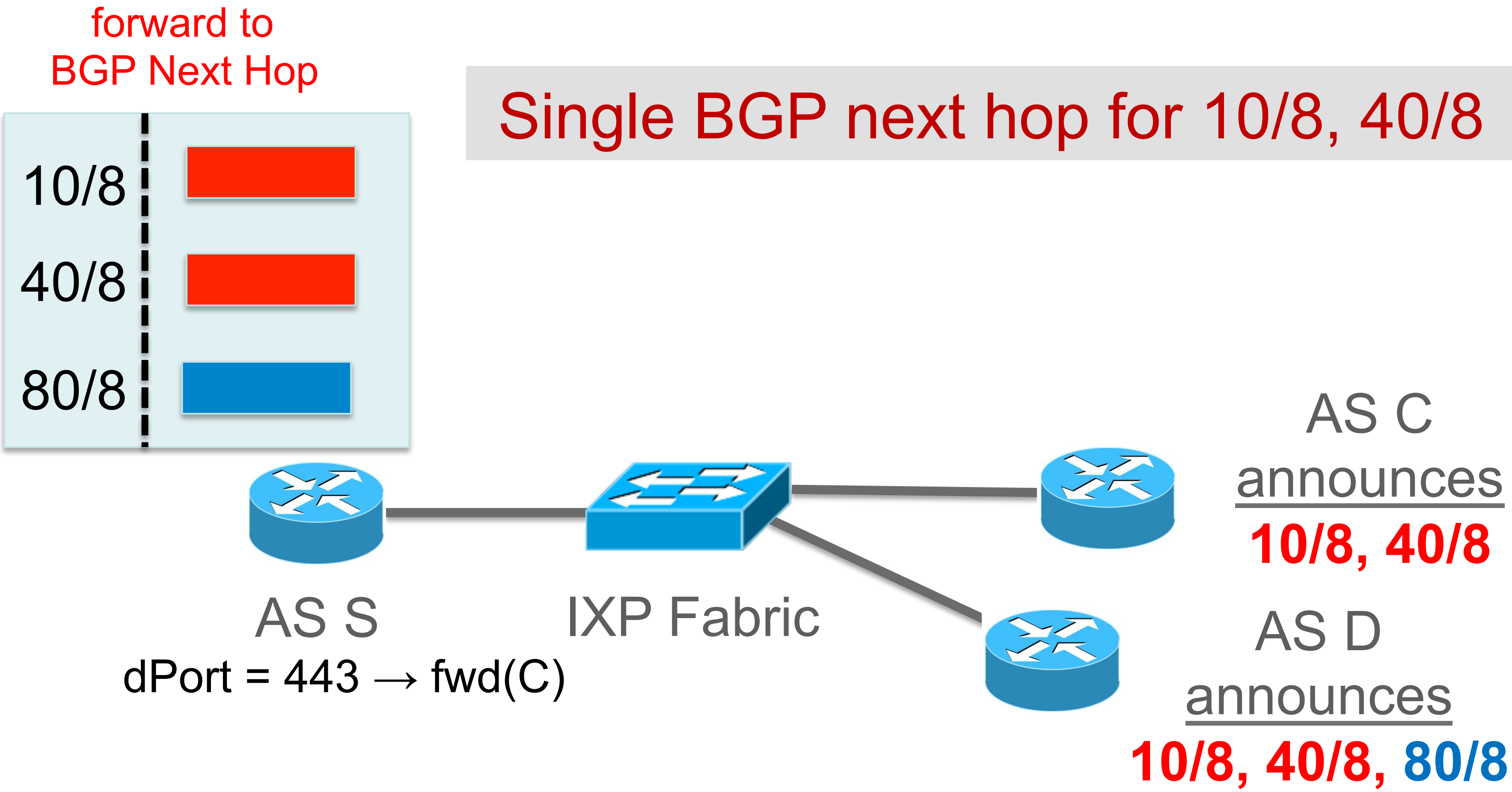
Forwarding Equivalence Classes

SDN Policies	# Forwarding Table Entries		
	10/8	40/8	80/8
dPort = 443 → fwd(C)	1	1	0
dPort = 22 → fwd(C)	1	1	0
dPort = 443 → fwd(D)	1	1	1

10/8, 40/8 exhibit similar forwarding behavior.

Applying Forwarding Equivalence

Single BGP next hop for 10/8, 40/8



forward to
BGP Next Hop

10/8 🟥

40/8 🟥

80/8 🟦

AS S

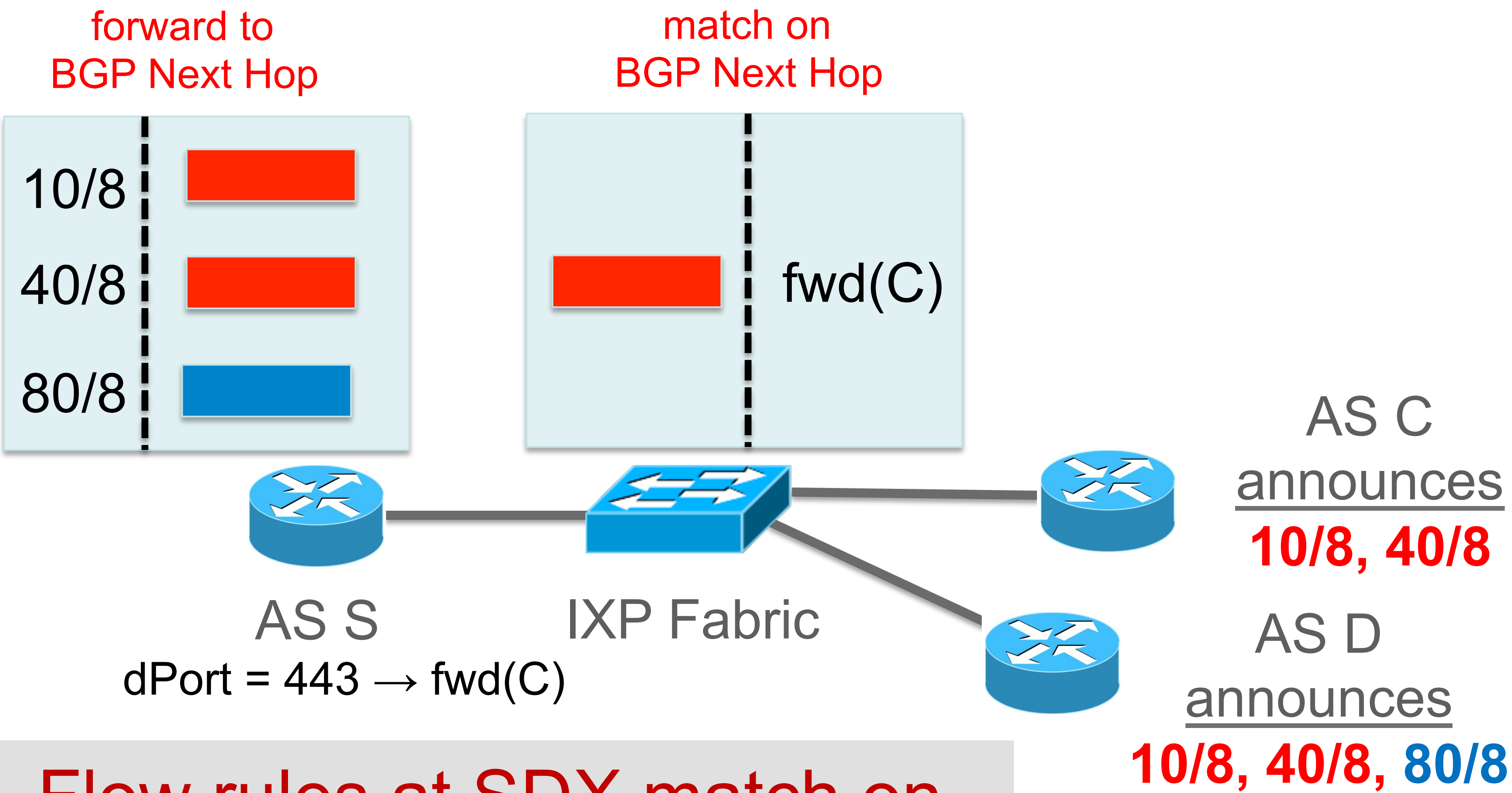
dPort = 443 → fwd(C)

IXP Fabric

AS C
announces
10/8, 40/8

AS D
announces
10/8, 40/8, 80/8

Applying Forwarding Equivalence



forward to
BGP Next Hop

10/8	🟥
40/8	🟥
80/8	🟦

match on
BGP Next Hop

🟥	fwd(C)

AS S
dPort = 443 → fwd(C)

IXP Fabric

AS C
announces
10/8, 40/8

AS D
announces
10/8, 40/8, 80/8

Flow rules at SDX match on
BGP next hops.

Number of Forwarding Entries

	Simple Example	**Large IXP**
Baseline	3	62K
Policy Augmentation	7	68M
***FEC Computation**	**4**	**21M**

[*SIGCOMM'14]

Still not possible to support forwarding table entries and update rate.

Three Insights (and Optimizations)

- Many prefix, policy combinations have exactly the same forwarding decision
  - **Optimization:** Forwarding equivalence
- Per-participant forwarding decisions have even more commonality
  - **Optimization:** Independent forwarding equivalence
- Advertisements can be encoded as FEC entries
  - **Optimization:** Reachability encoding

What If Each Participant Computes FEC Independently?

SDN Policies	# Forwarding Table Entries	
	{10/8, 40/8}	80/8
dPort = 443 → fwd(C)	1	0
dPort = 22 → fwd(C)	1	0
dPort = 443 → fwd(D)	1	1

Independent FEC computation is more efficient.

Independent FEC Computation

- Large number of SDX participants
  - Many different policies on groups of prefixes
  - Leads to a large number of small FECs of prefixes

- Compute FECs independently
  - Separate computation per participant
  - Leads to small number of large FECs, and less frequent recomputation
  - Enables “scale out” of the FEC computation

Independent FEC Computation

Idea: Each participant independently computes FECs.

SDN Policies	# Forwarding Table Entries	
	{10/8, 40/8}	80/8
dPort = 443 → fwd(C)	1	0
dPort = 22 → fwd(C)	1	0

2

dPort = 443 → fwd(D)	1

1

Number of Entries

	Simple Example	**Large IXP**
Baseline	3	62K
Policy Augmentation	7	68M
FEC Computation	4	21M
Independent FEC Computation	**3**	**763K**

Still not possible to support forwarding table entries and update rate.

Three Insights (and Optimizations)

- Many prefix, policy combinations have exactly the same forwarding decision
  - **Optimization:** Forwarding equivalence

- Per-participant forwarding decisions have even more commonality
  - **Optimization:** Independent forwarding equivalence

- Advertisements can be encoded as FEC entries
  - **Optimization:** Reachability encoding

BGP & SDN Coupling

Incoming BGP Update:

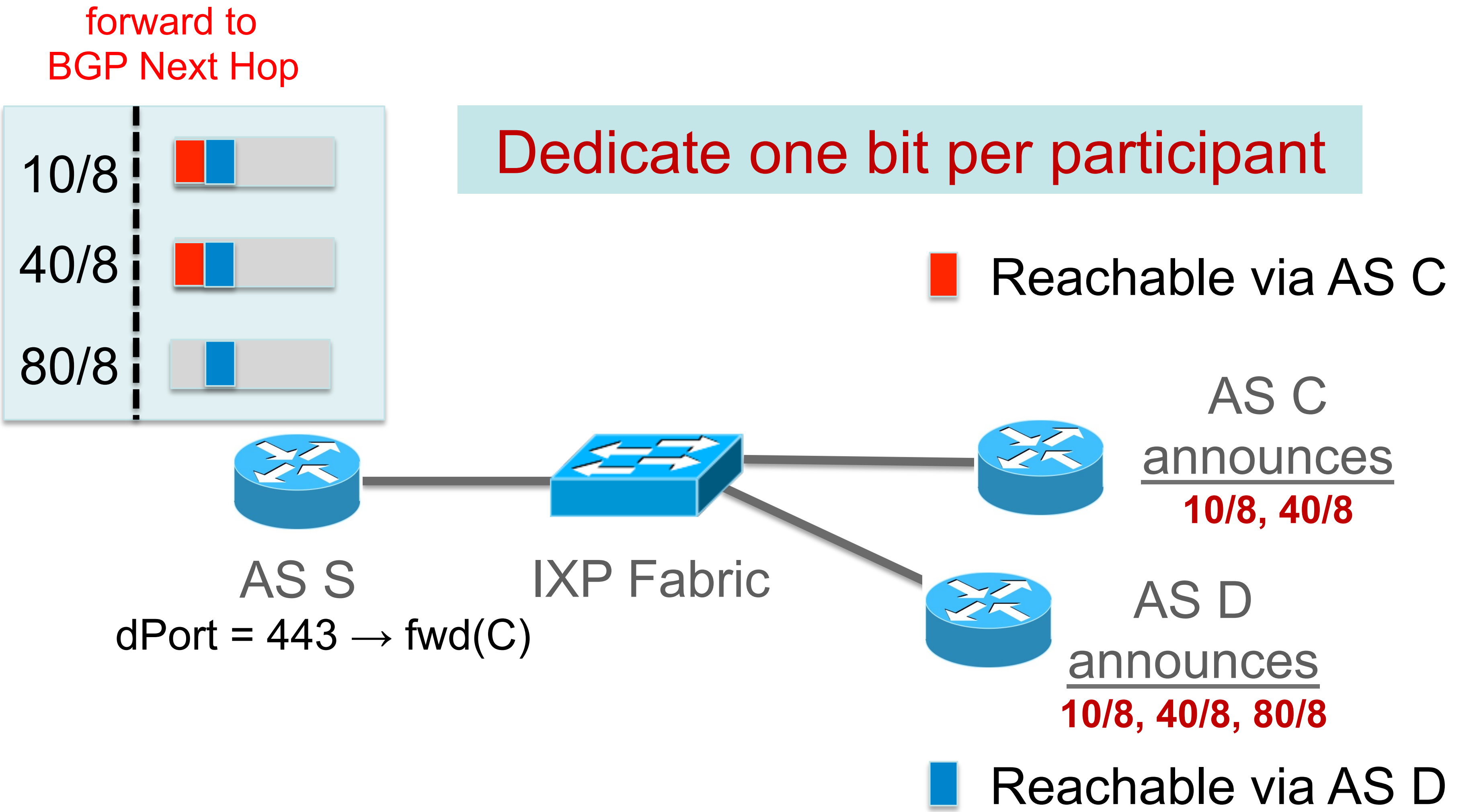
{AS D withdraws route for prefix 10/8}

SDN Policies	# Forwarding Table Entries		
	10/8	40/8	80/8
dPort = 443 → fwd(C)	1	1	0
dPort = 22 → fwd(C)	1	1	0
dPort = 443 → fwd(D)	**1 → 0**	1	1

Decoupling BGP from SDN

- Apply advances in commodity hardware switches
  - Support for Bitmask Matching (OpenFlow 1.3)
- Extend BGP “next hop” encoding
  - So far: encode FECs (single field)
  - Idea: encode **reachability encoding**
- Changing only the BGP announcements
  - No need to update the SDX data plane!

Reachability Encoding



Dedicate one bit per participant

forward to
BGP Next Hop

10/8 🟥🟦
40/8 🟥🟦
80/8 ⬜🟦

🟥 Reachable via AS C

AS C
announces
10/8, 40/8

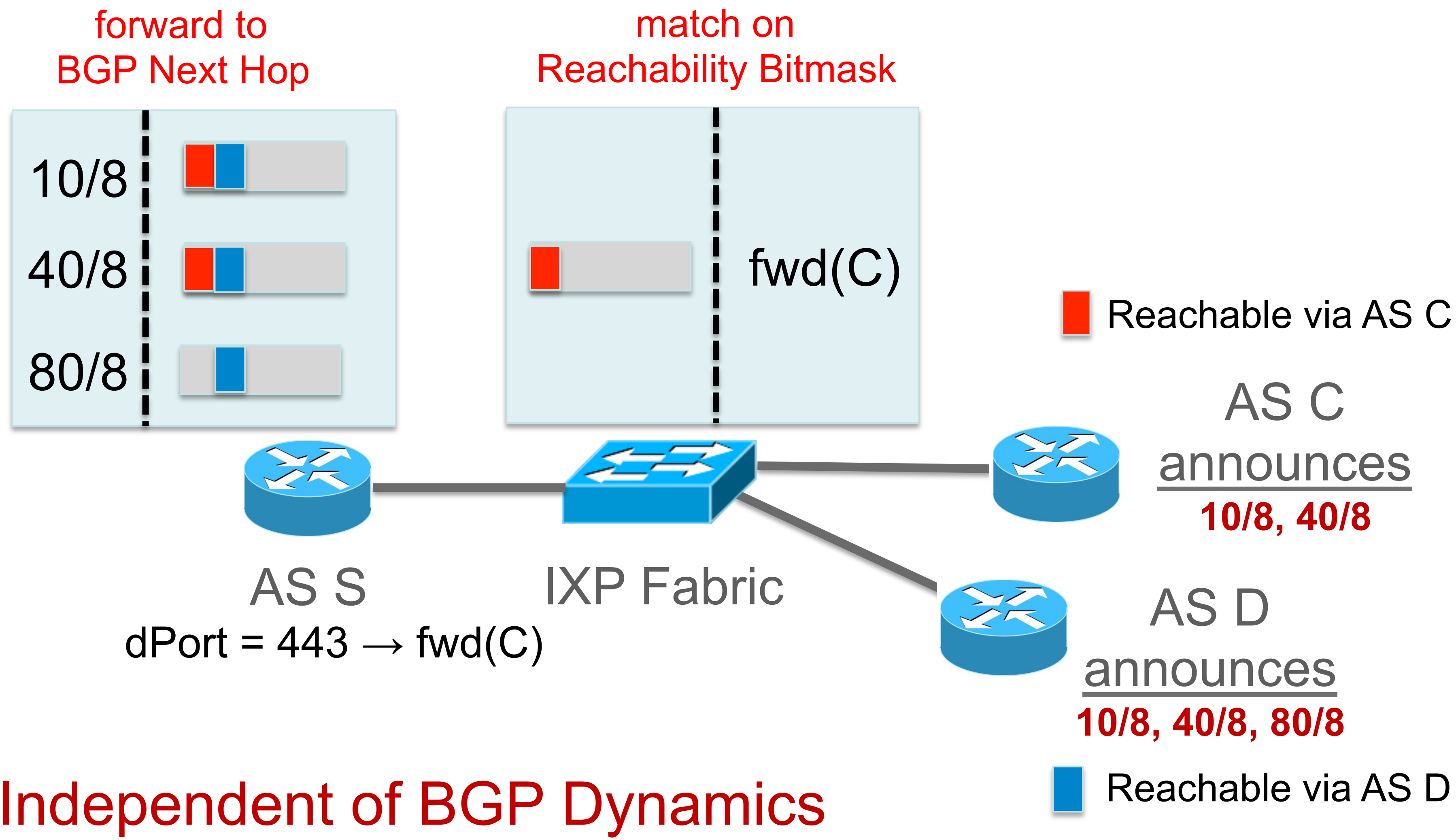
AS S
dPort = 443 → fwd(C)

IXP Fabric

AS D
announces
10/8, 40/8, 80/8

🟦 Reachable via AS D

Reachability Encoding



forward to
BGP Next Hop

10/8 🟥🟦
40/8 🟥🟦
80/8 🟦

match on
Reachability Bitmask

🟥 fwd(C)

🟥 Reachable via AS C

AS C
announces
10/8, 40/8

AS S
dPort = 443 → fwd(C)

IXP Fabric

AS D
announces
10/8, 40/8, 80/8

🟦 Reachable via AS D

Independent of BGP Dynamics

Reachability Encoding

SDN Policies	# Forwarding Table Entries
	C
dPort = 443 → fwd(C)	1
dPort = 22 → fwd(C)	1

2

dPort = 443 → fwd(D)	1

1

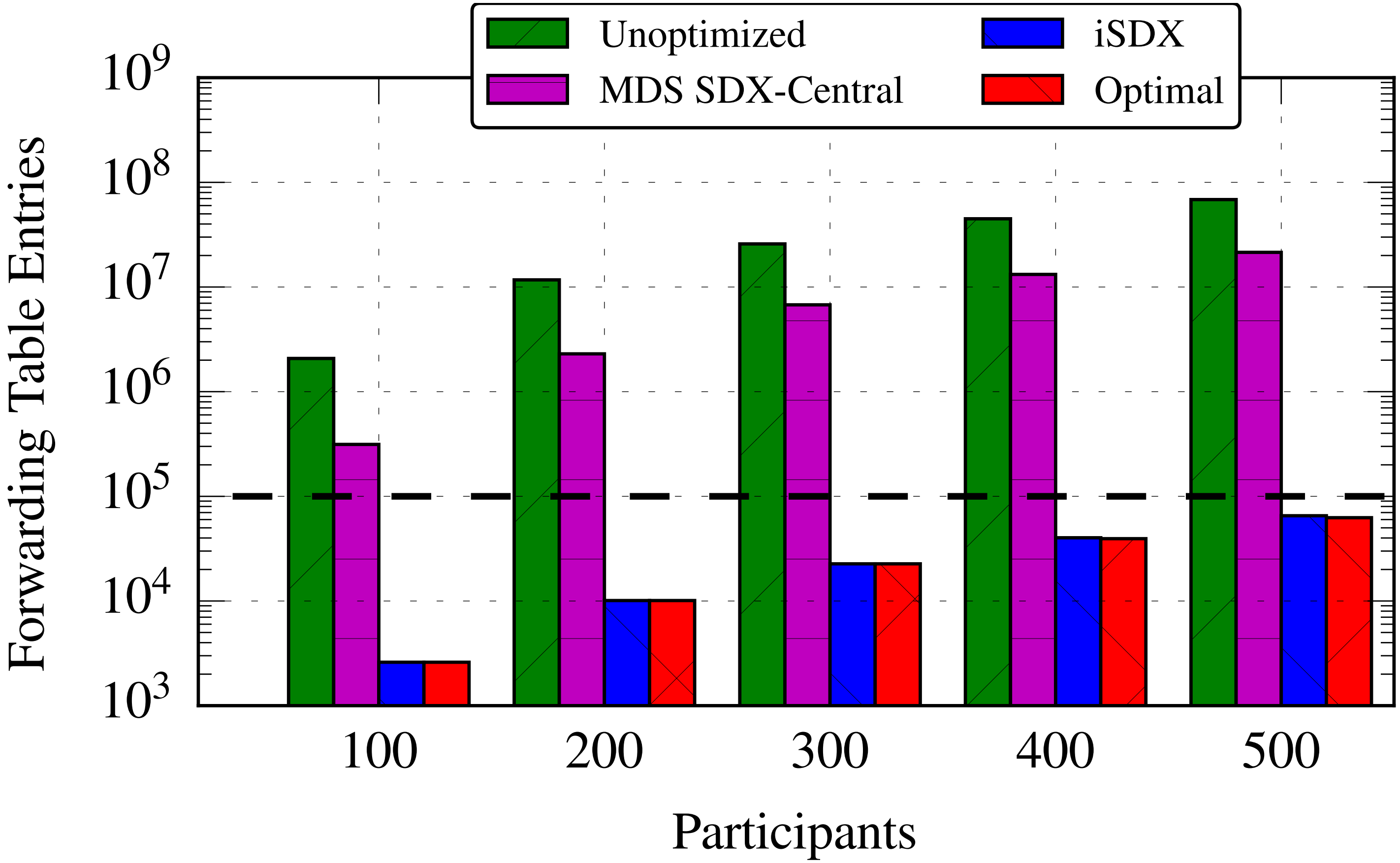
Reduces Data Plane State

Number of Forwarding Entries

	Simple Example	**Large IXP**
Baseline	3	62K
Policy Augmentation	7	68M
FEC Computation	4	21M
Independent FEC Computation	3	763K
Reachability Encoding	**3**	**65K**

We can now run SDX over commodity hardware switches.

We Can Do This at Industry-Scale!



BGP routes and updates for large EU IXP in a commodity hardware switch.

iSDX Evaluation: Summary

- **Data Plane State:**
  - Requires **65K < 100K** forwarding table entries

- **Data Plane Update Rate:**
  - Requires **0** < **2500** updates/second

- **Other Goals:**
  - Processes BGP update bursts in real time **(50 ms)**
  - Requires only **360 BGP Next Hops** compared to 25K from previous solutions

You Can Run iSDX Today

http://sdx.cs.princeton.edu

- Running code
  - Vagrant & Docker based setup
  - Instructions to run with **Hardware Switches**

- Ongoing efforts
  - Hosted by **Open Networking Foundation**
  - Deployment
    - Inter-agency exchange
    - IXPs in Europe & Asia