

SDN Usecases

ECE/CS598HPN

Radhika Mittal

B4: Experience with a Globally- Deployed Software Defined WAN

Google

SIGCOMM'13

B4: Google's Software-Defined WAN

- Google operates two separate backbones:
 - B2: carries Internet facing traffic
 - Growing at a rate faster than the Internet
 - B4: carries inter-datacenter traffic
 - More traffic than B2
 - Growing faster than B2

B4: Google's Software-Defined WAN

- Google operates two separate backbones:
 - B2: carries Internet facing traffic
 - Growing at a rate faster than the Internet
 - **B4: carries inter-datacenter traffic**
 - More traffic than B2
 - Growing faster than B2

B4: Google's Software-Defined WAN

Among the first and largest SDN/OpenFlow deployment.



Why SDN/OpenFlow?

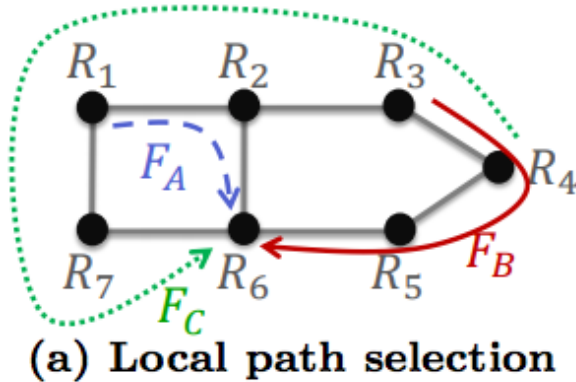
- Opportunity to reason about global state
 - Simplified coordination and orchestration.
- Exploit raw speed of commodity servers.
 - Latest generation servers are much faster than embedded switch processors.
- Decouple software and hardware evolution.
 - Control plane software can evolve more quickly.
 - Data plane hardware can evolve slower based on programmability and performance.

What did B4 use SDN for?

- Centralized routing.
 - Basic functionality.
 - Allowed Google to develop and stress test the SDN architecture.
- Centralized traffic engineering.
 - Allocating routes (and bandwidth) to groups of flows.
 - Also allows prioritizing some flows over others.
 - Enables running the WAN at higher utilization.

Traffic Engineering

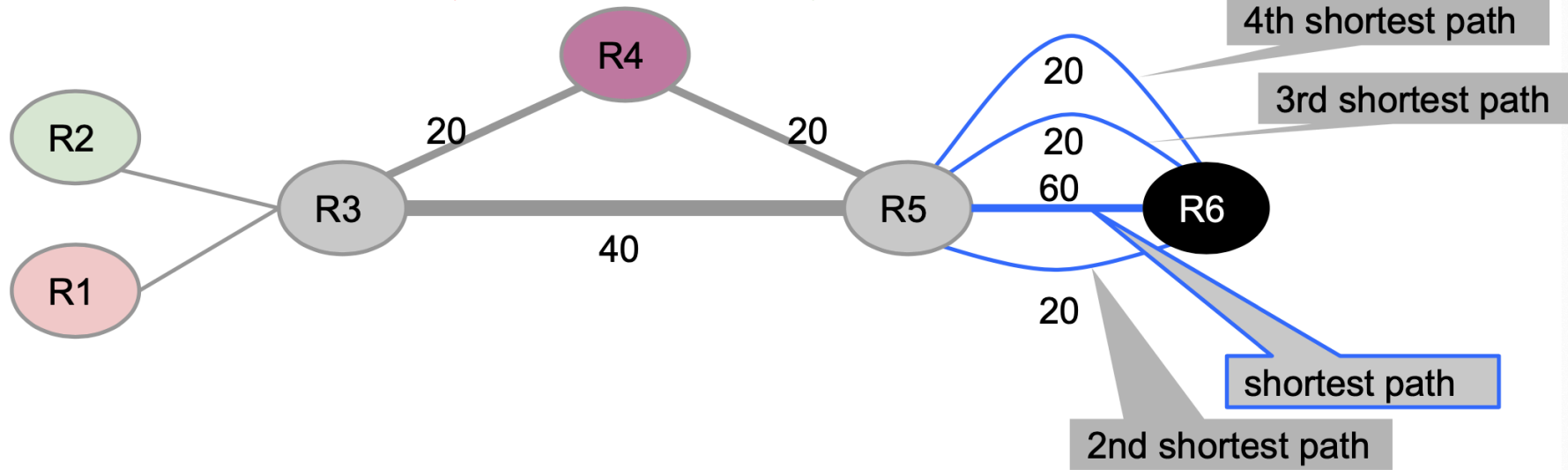
- Traditionally accomplished via MPLS tunnels.
 - Tunnels defines routes and priority.
 - Ingress routers locally and greedily map flows to tunnels.



- Centralized TE using SDNs allows closer to optimal routes.

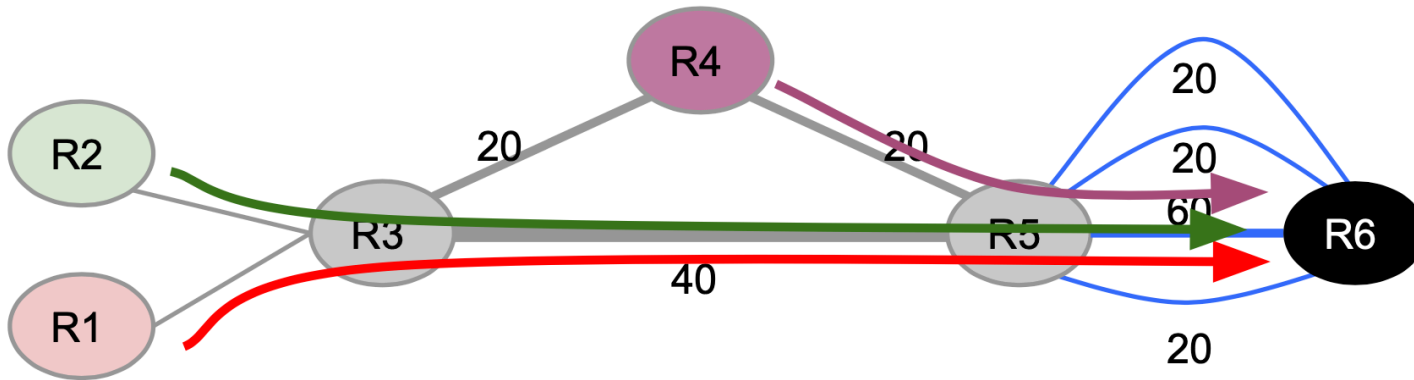
Traffic Engineering: another example

Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20



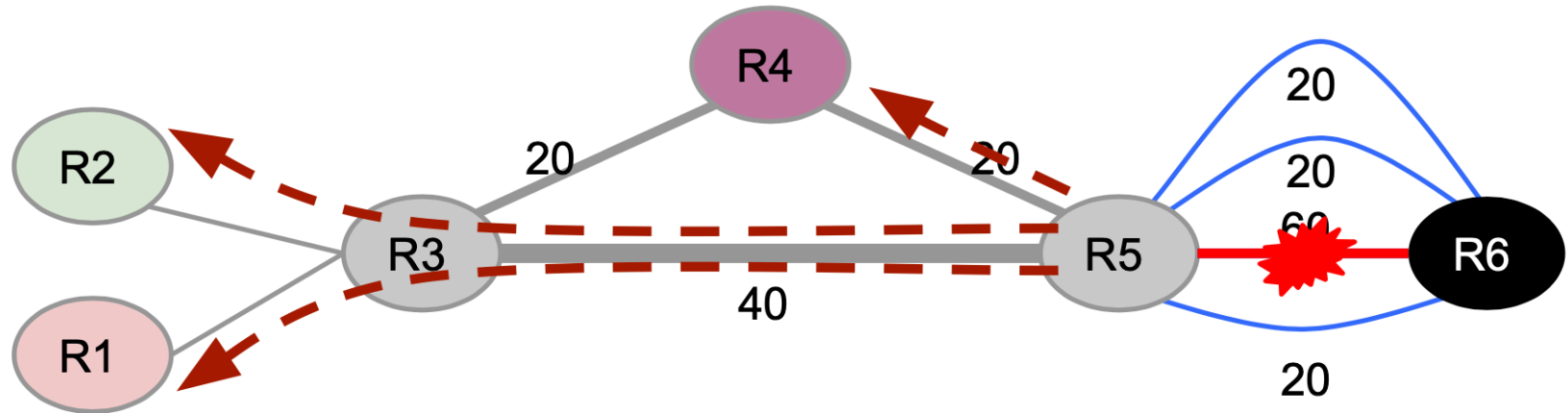
Traffic Engineering: another example

Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20



Traffic Engineering: another example

Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20

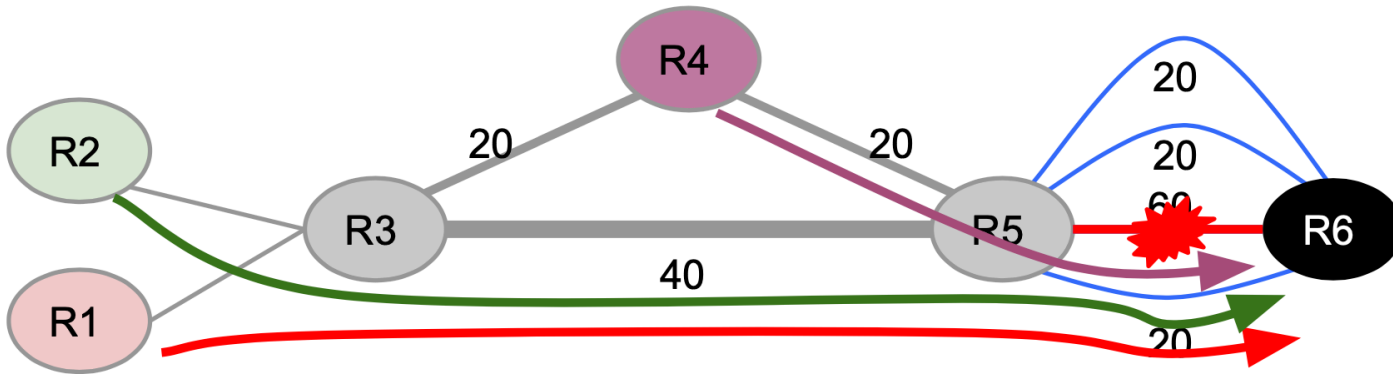


R5-R6 link fails

- R1, R2, R4 *autonomously* find next best path

Traffic Engineering: another example

Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20



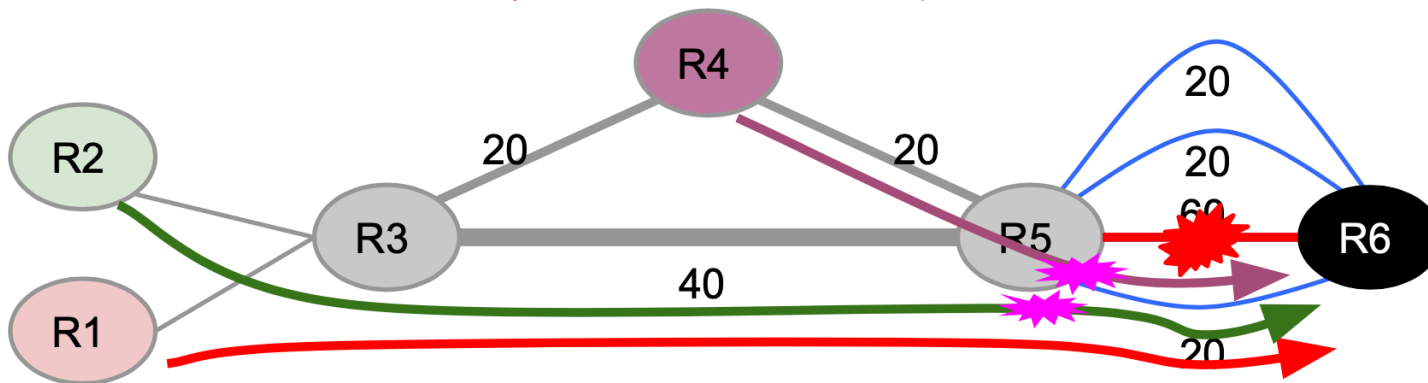
R5-R6 link fails

- R1, R2, R4 *autonomously* try for next best path
- R1, R2, R4 push **20** altogether

No Traffic Engineering

Traffic Engineering: another example

Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20



R5-R6 link fails

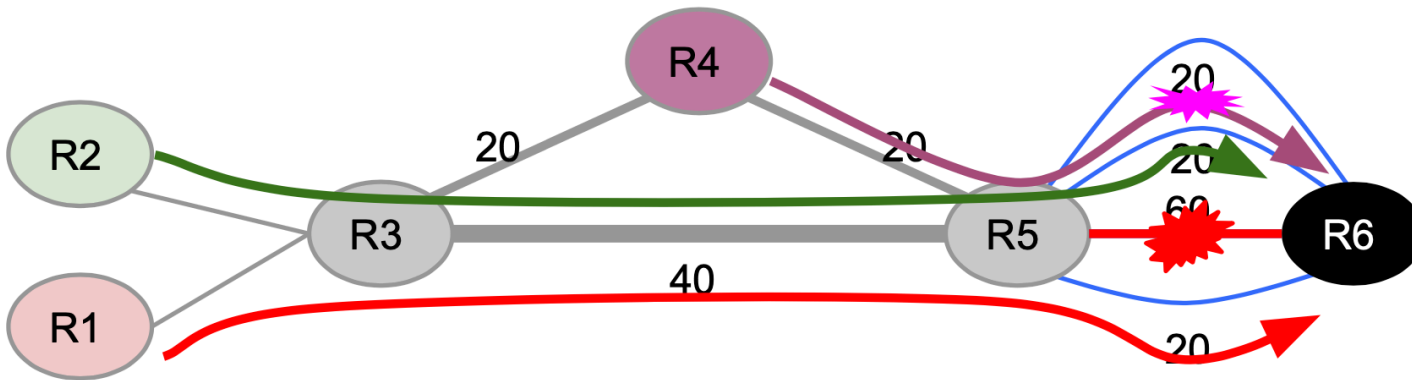
- R1, R2, R4 *autonomously* try for next best path
- R1 wins, R2, R4 retry for next best path

Distributed Traffic Engineering Protocols

e.g. MPLS + RSVP

Traffic Engineering: another example

Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20



R5-R6 link fails

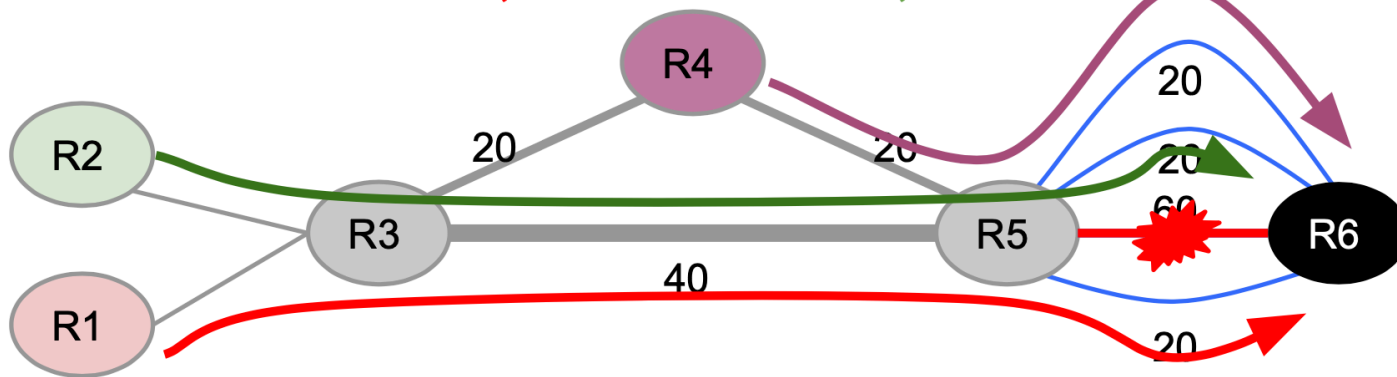
- R1, R2, R4 *autonomously* try for next best path
- R1 wins, R2, R4 retry for next best path
- R2 wins this round, R4 retries again

Distributed Traffic Engineering Protocols

e.g. MPLS + RSVP

Traffic Engineering: another example

Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20



R5-R6 link fails

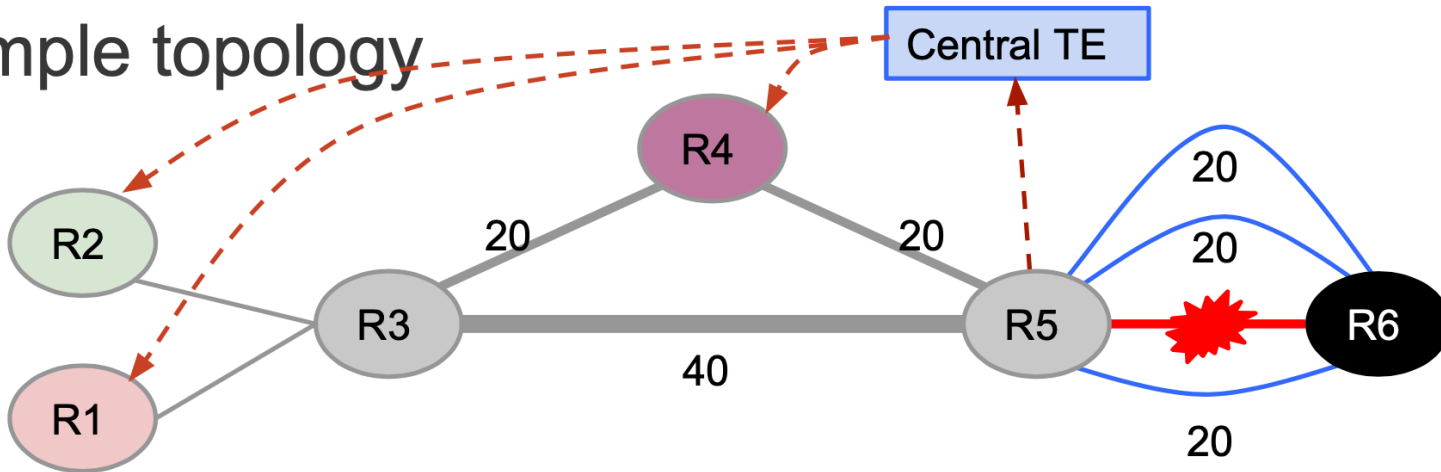
- R1, R2, R4 *autonomously* try for next best path
- R1 wins, R2, R4 retry for next best path
- R2 wins this round, R4 retries again
- R4 finally gets third best path!

Distributed Traffic Engineering Protocols

e.g. MPLS + RSVP

Traffic Engineering: another example

Simple topology



Flows:

- R1->R6: 20; R2->R6: 20; R4->R6: 20

R5-R6 fails

- R5 informs TE, which programs routers in one shot

Centralized Traffic Engineering Protocols

B4 SDN architecture

protocol

silicon

protocol

silicon

protocol

silicon

protocol

silicon

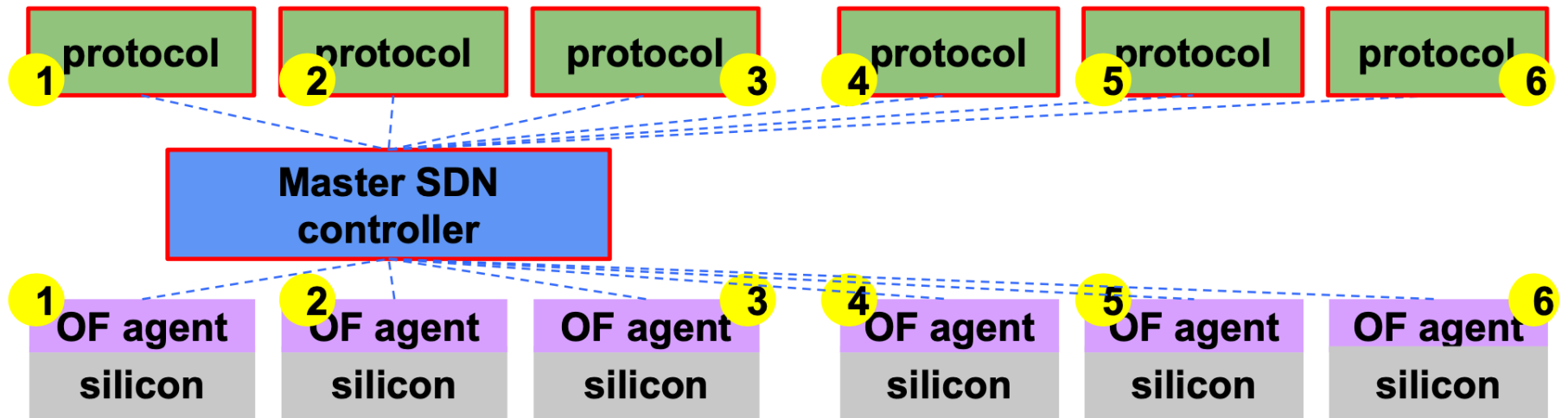
protocol

silicon

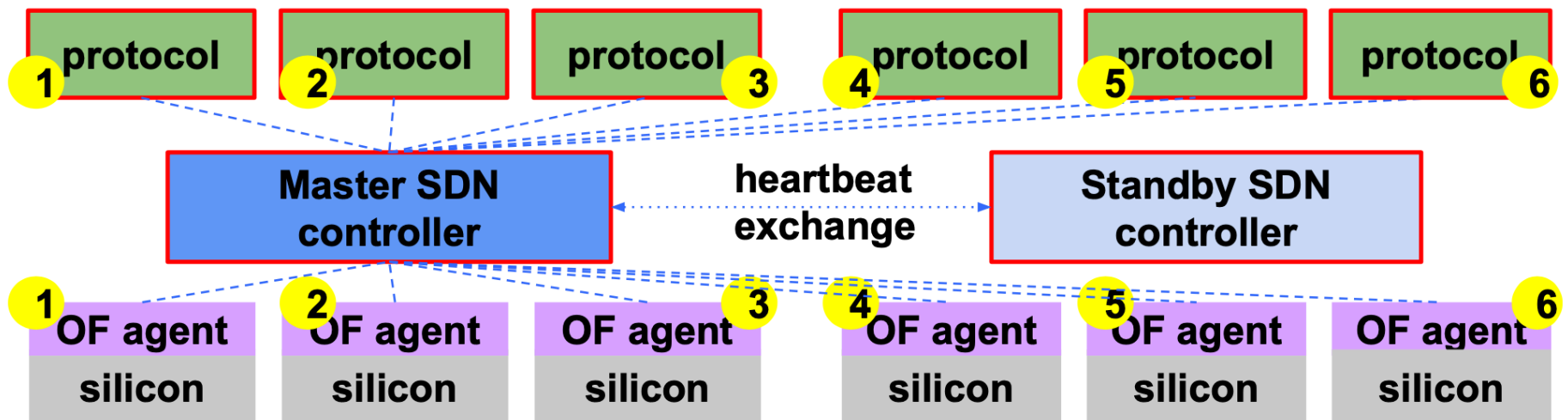
protocol

silicon

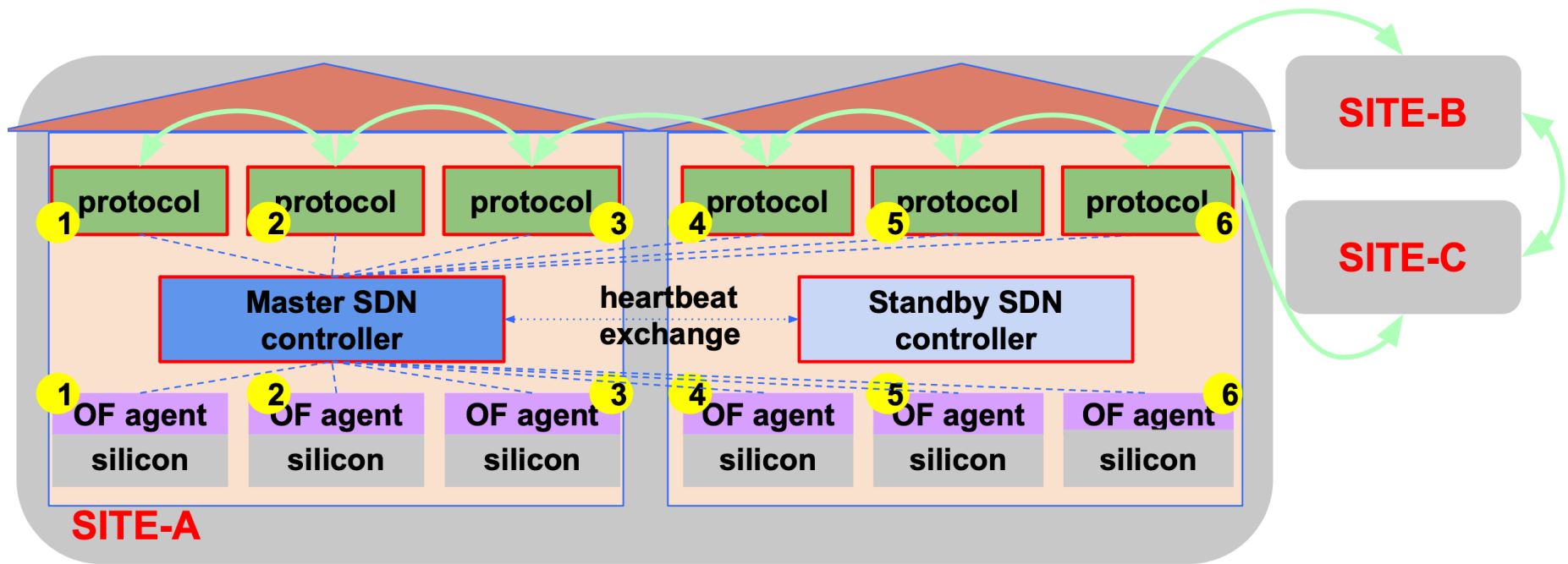
B4 SDN architecture



B4 SDN architecture

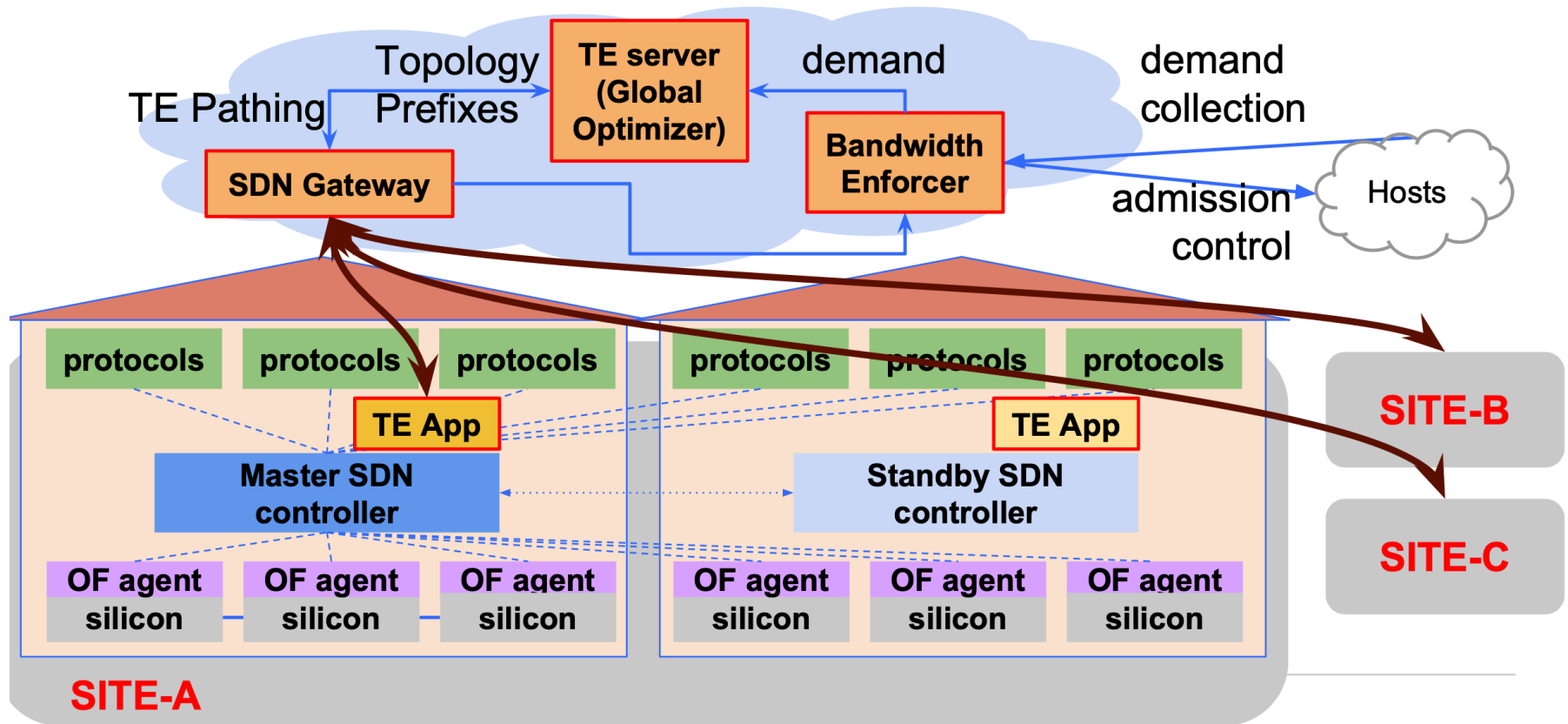


B4 SDN architecture



Unit of management is a site = fabric

B4 SDN architecture



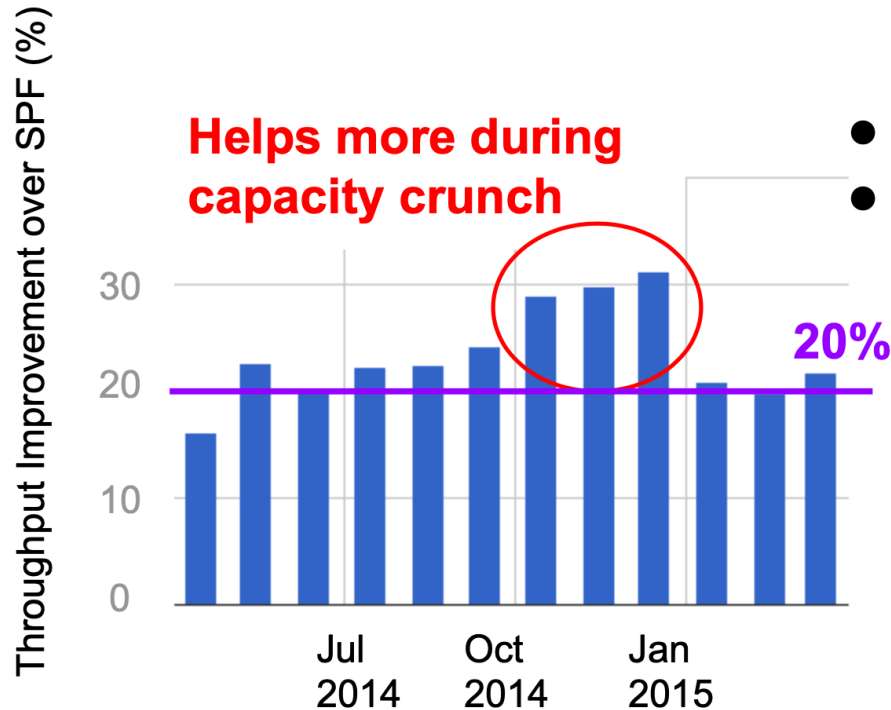
B4 Traffic Engineering

- Objective: max-min fairness
 - A: 10Gbps, B: 5Gbps, total link capacity = 12Gbps
 - $B = 5\text{Gbps}$
 - $A = 7\text{Gbps}$
 - A: 10Gbps, B: 5Gbps, C: 2Gbps, link capacity = 12Gbps
 - $C = 2\text{Gbps}$
 - $B = 5\text{Gbps}$
 - $A = 5\text{Gbps}$
 - Same demands, $W(A) = 2, W(B) = 1, W(C) = 1$, link capacity = 12Gbps
 - $C = 2\text{Gbps}$
 - $B = 3.33\text{Gbps}$
 - $A = 6.67\text{Gbps}$
- Grouping flows with different demands together leads to piecewise linear fairness functions.
- Greedy (water-filling) heuristic to allocate across multiple paths.
- *Bandwidth Enforcer, SIGCOMM'15 has more details on TE algorithms*

Other notable aspects

- Routing decoupled from TE.
 - Cautious about too many changes at once.
- Custom switch hardware.
 - No existing hardware supported their needs.
- Dedicated out-of-band control-plane network.
- Hot controller stand-bys
 - Controller instance failure has no impact on packet loss.

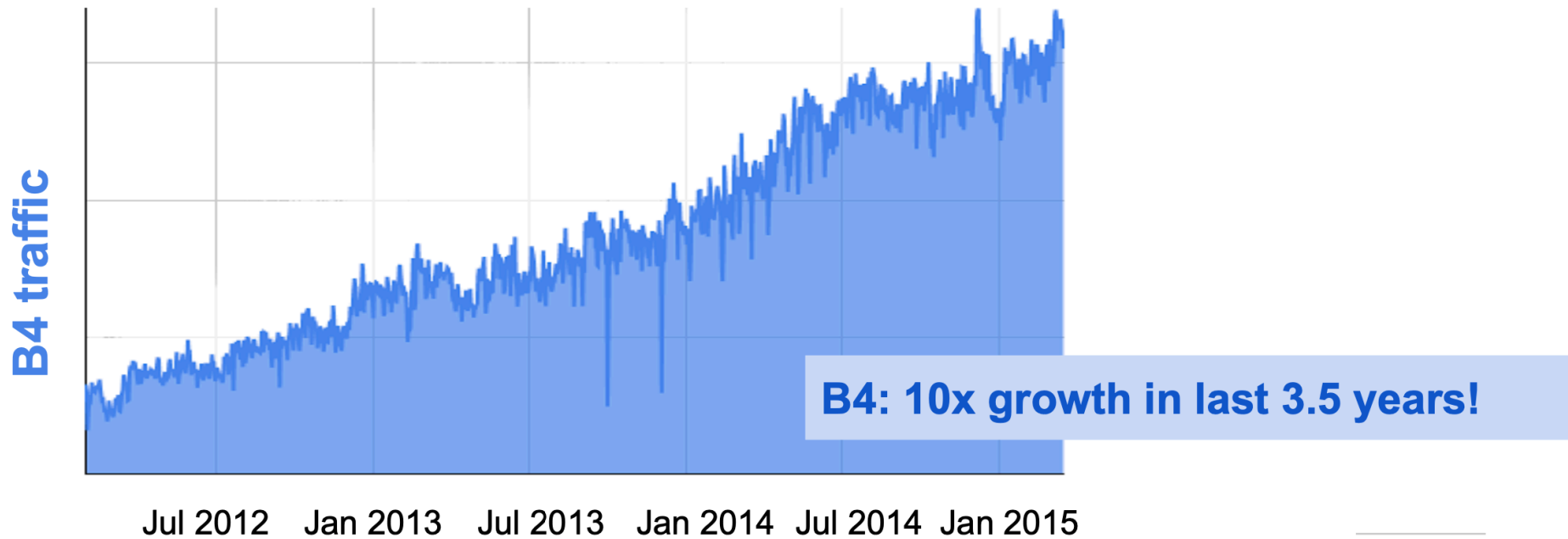
Benefit of Centralized TE



- ~20% increase in throughput over SPF
- Larger benefits during capacity crunch

Lowens the requirement for bandwidth provisioning

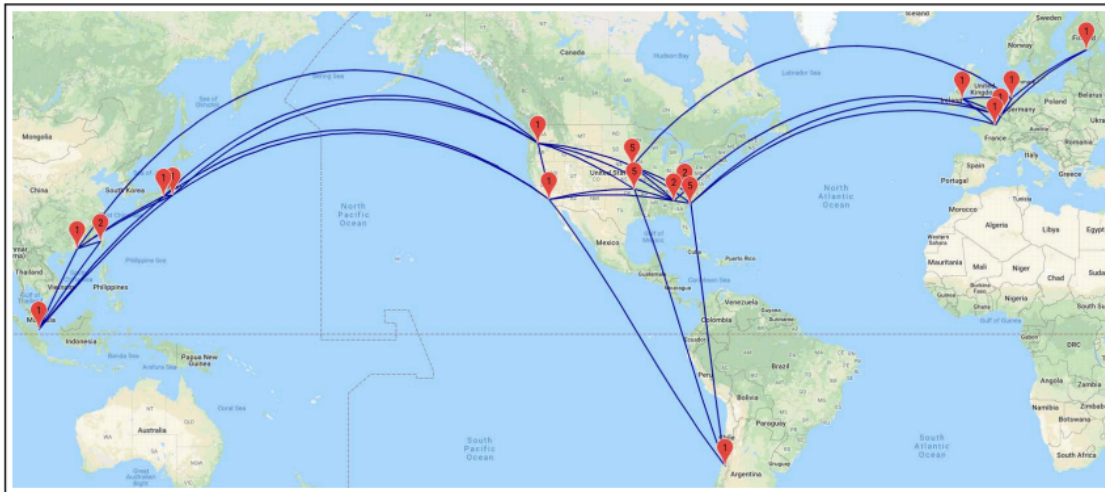
Benefit of Centralized TE



Other thoughts?

B4 and After: SIGCOMM'18

- Growth in traffic: more sites, larger sites, more paths.
 - Flat topology scales poorly:
 - Hierarchical topology at each site.
 - Hierarchical traffic engineering.
 - Customized hierarchical two-phase rule matching in switches.
 - Integrated routing and TE (BGP/ISIS used as fallback).

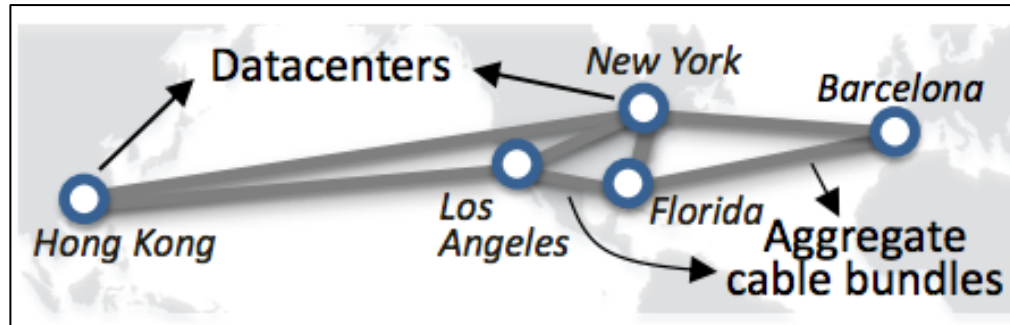


33 sites, 2018

(43 sites, 2026)

Another software-defined WAN

- SWAN (WAN connecting Microsoft's datacenter)
 - Goal: increase WAN link utilization.
 - Centralized and global traffic engineering.

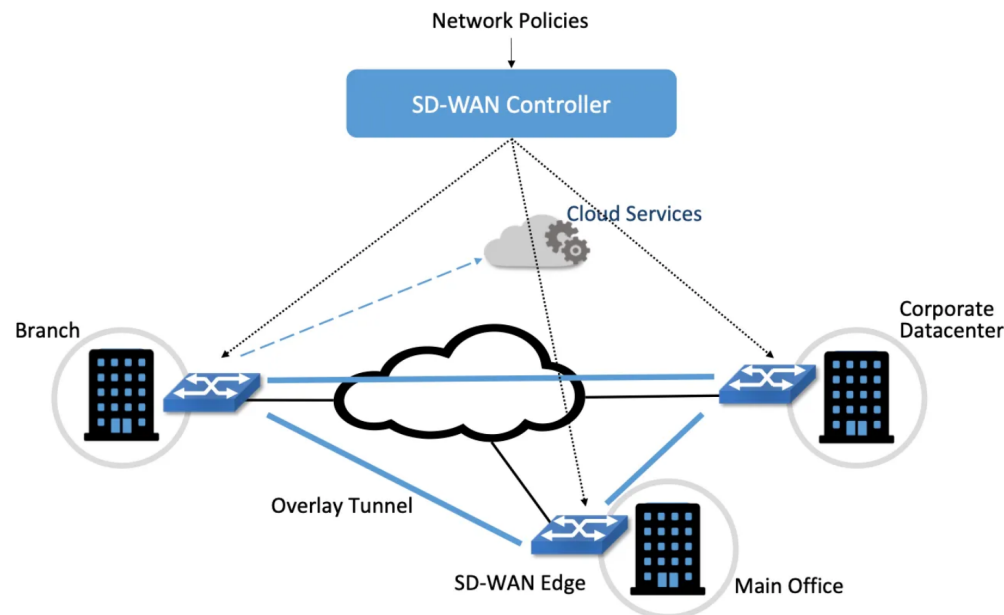


SDNs in ISPs

- My understanding of status quo:
 - Centralized TE computations.
 - Enforcement via MPLS (or segment routing).

SDNs in ISPs

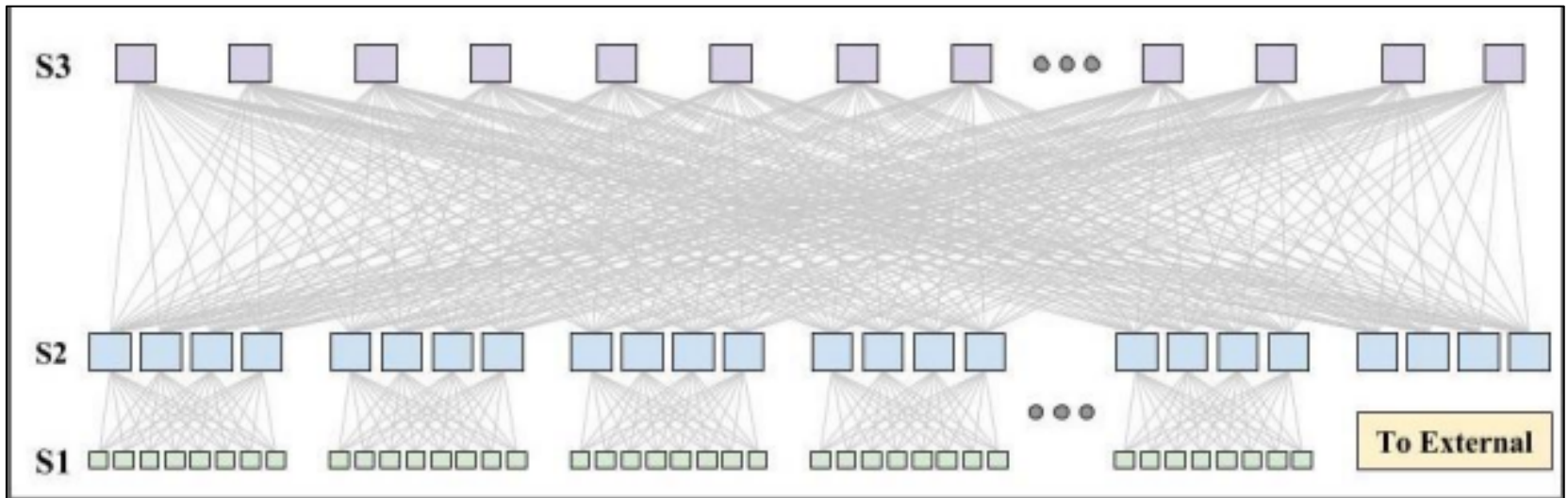
- SD-WAN
 - ISPs provide this as a service to enterprises.
 - Centrally managed overlay network between different sites of an enterprise and the cloud.



Other SDN usecases at Google

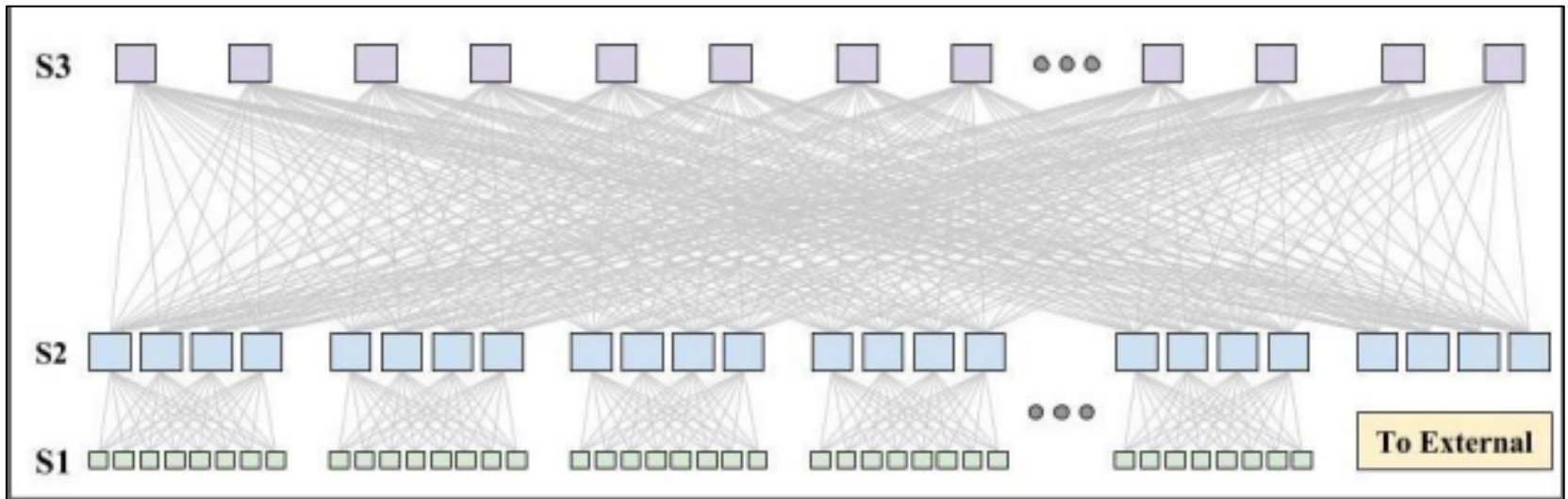
Datacenter routing

- Thousands of switches distributed across clusters.
- High communication overhead for distributed routing.



Datacenter routing

- Jupiter (Google's Datacenter), SIGCOMM'15
 - Centralized configuration for baseline static topology.
 - Centralized dissemination of link state.
 - Each switch reacts locally to changes.



Datacenter routing

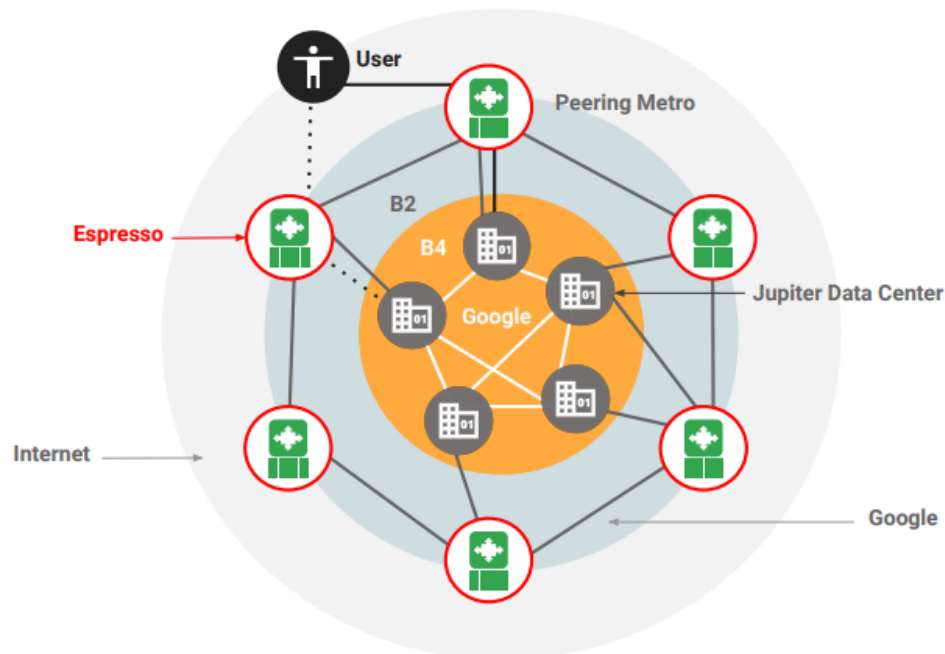
- Jupiter (Google's Datacenter)
 - Use of SDN was key to enabling evolution in Jupiter's topology
 - Jupiter evolving: transforming google's datacenter network via optical circuit switches and software-defined networking
 - SIGCOMM'22

Policy enforcement at user-facing edge

- Internet edge routers implement rich set of features:
 - Access control, firewall, BGP routing policies.
- Policies require global, cross-layer optimizations.
 - Might also require switch upgrades, that affect availability.

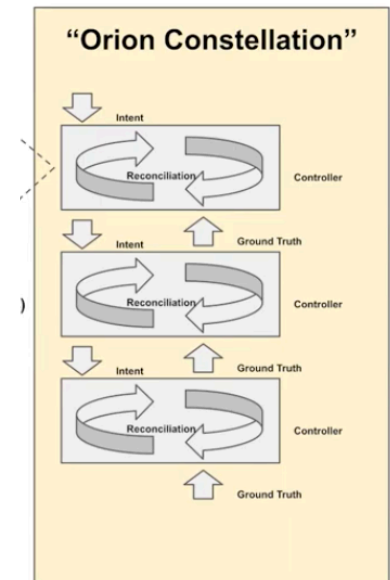
Policy enforcement at user-facing edge

- Espresso (SIGCOMM'17):
 - Global software control plane to compute policies.
 - Local control plane to translate policy to forwarding rules.



Google's own control plane

- Orion: Google's Software-Defined Networking Control Plane (NSDI'21): used with Jupiter and B4.
 - modular micro-service based controller
 - multiple layers
 - Inter-block routing -> intra-block routing -> per-node flow programming.
 - intent flows down, ground truth flows up
 - pub-sub system



SDN in Stratosphere

- Loon's aerospace mesh network (SIGCOMM'22)

