SDN Usecases

ECE/CS598HPN

Radhika Mittal

Logistics

- No class this Friday, Sept 15th.
- Warm-up assignment due next Wednesday (Sept 20th).
    - *Have you found a grading partner?*
- Would you like your opinions to be anonymous or is name calling ok?

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

Slide content from Subhasree Mandal

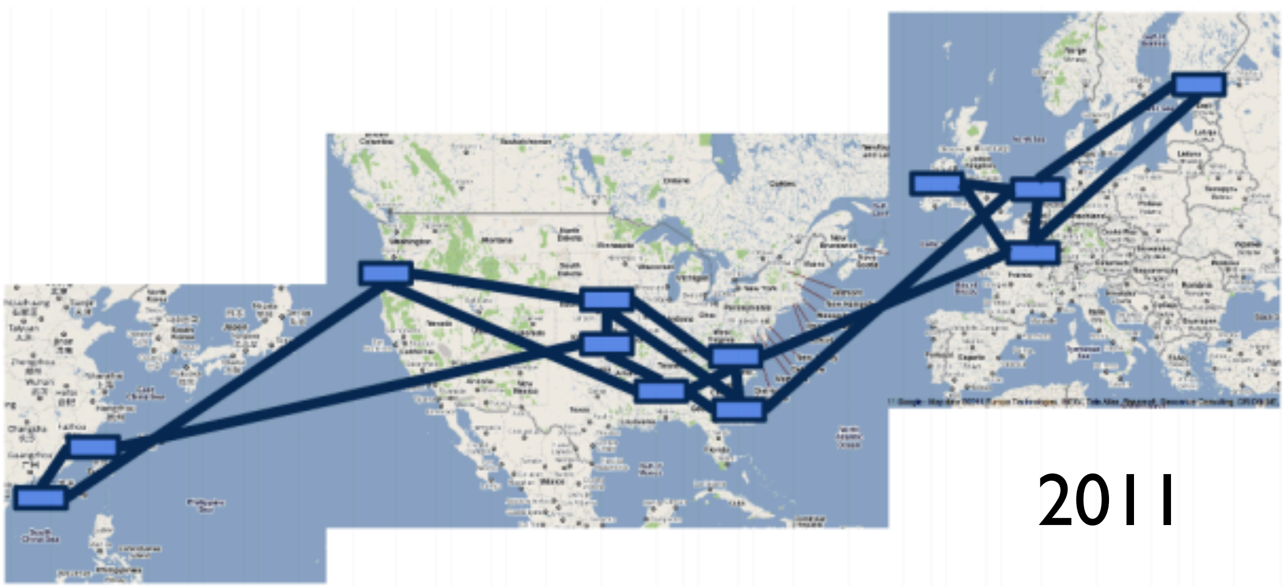
B4: Google's Software-Defined WAN

- 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**

Slide content from Subhasree Mandal

B4: Google's Software-Defined WAN

Among the first and largest SDN/OpenFlow deployment.



2011

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.

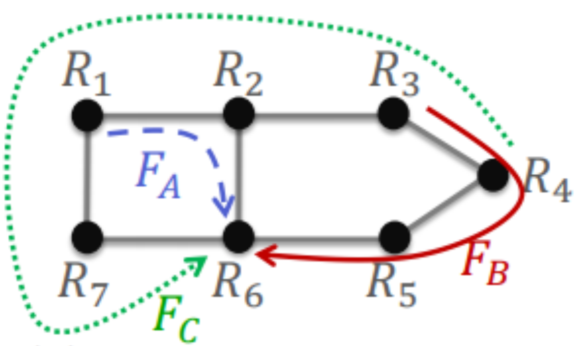
What did B4 use SDN for?

- 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.



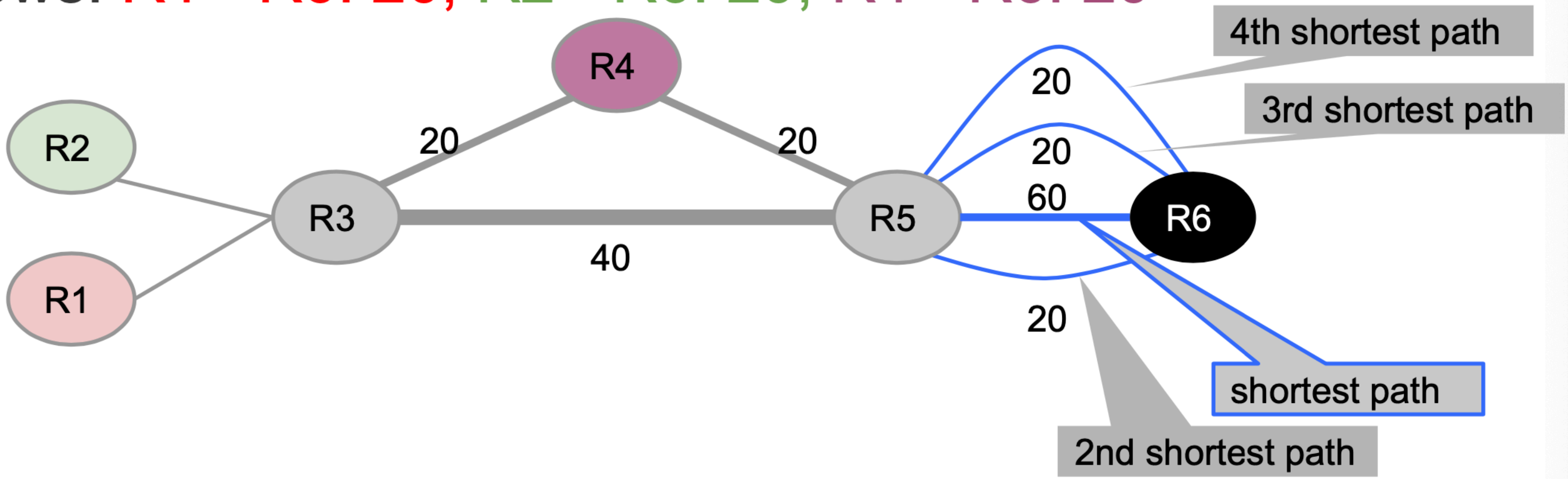
(a) Local path selection

- Centralized TE using SDNs allows closer to optimal routes.

Example from Microsoft's SWAN, SIGCOMM'13

Traffic Engineering: another example

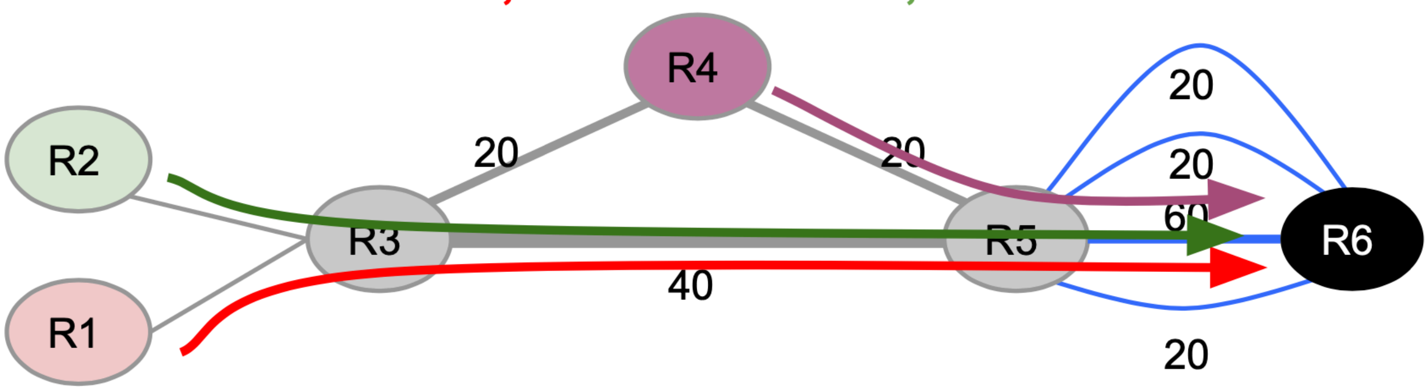
Flows: R1->R6: 20; R2->R6: 20; R4->R6: 20



Slide content from Subhasree Mandal

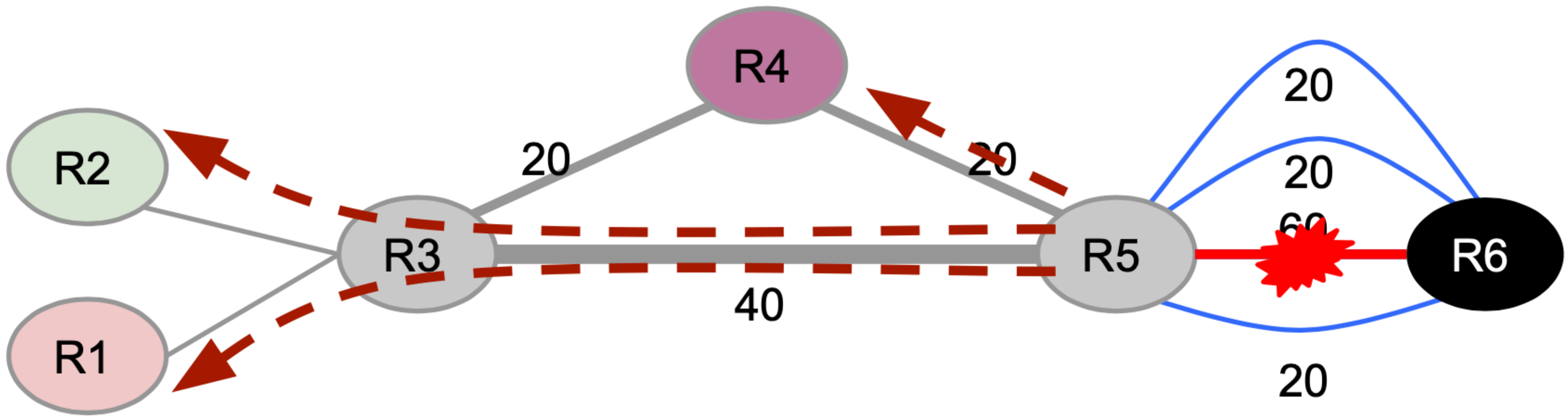
Traffic Engineering: another example

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



Traffic Engineering: another example

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

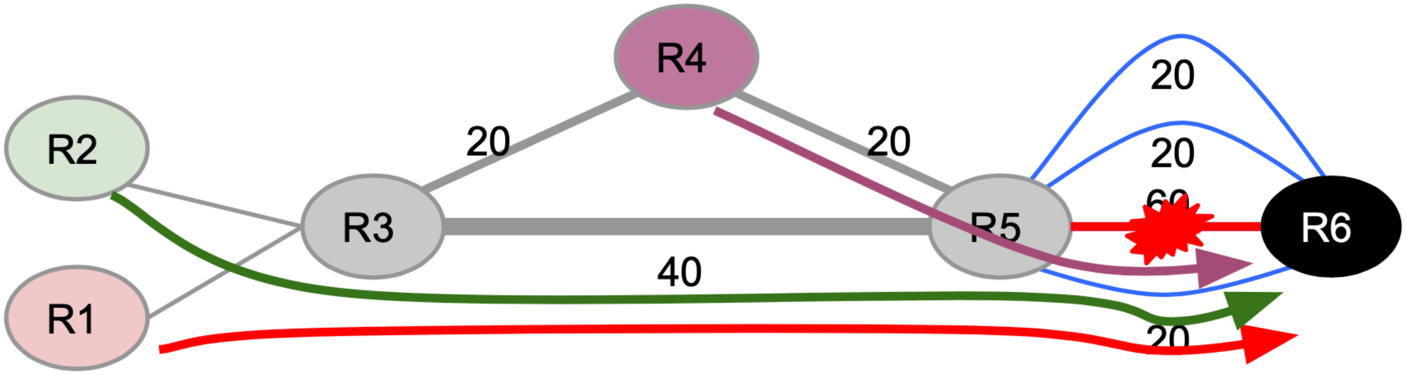


R5-R6 link fails

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

Traffic Engineering: another example

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



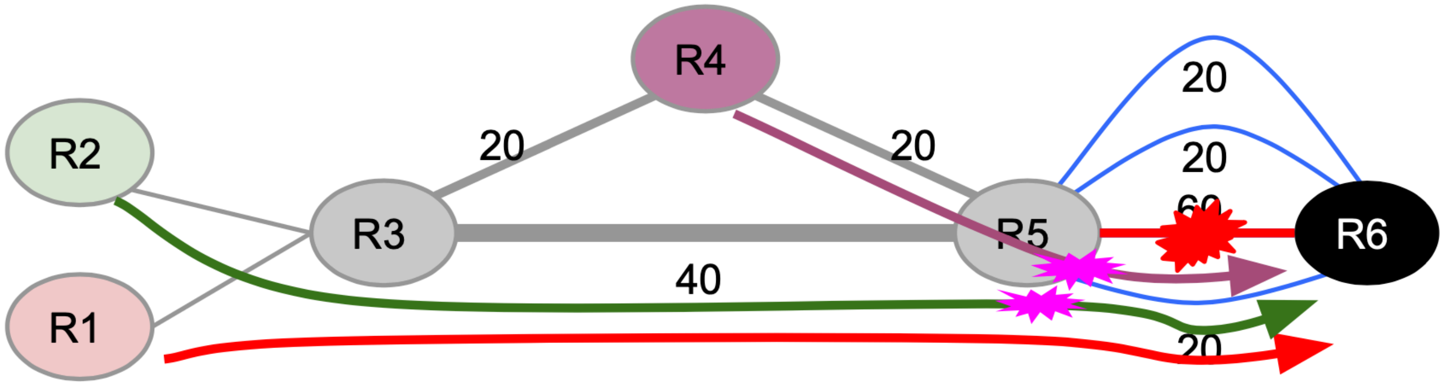
No Traffic Engineering

R5-R6 link fails

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

Traffic Engineering: another example

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



Distributed Traffic Engineering Protocols

e.g. MPLS + RSVP

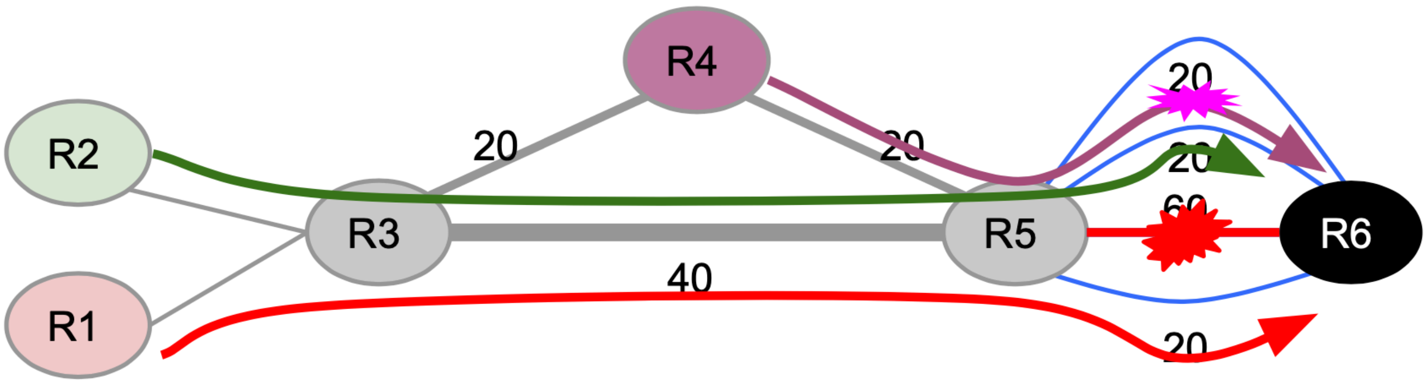
R5-R6 link fails

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

Slide content from Subhasree Mandal

Traffic Engineering: another example

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



Distributed Traffic Engineering Protocols

e.g. MPLS + RSVP

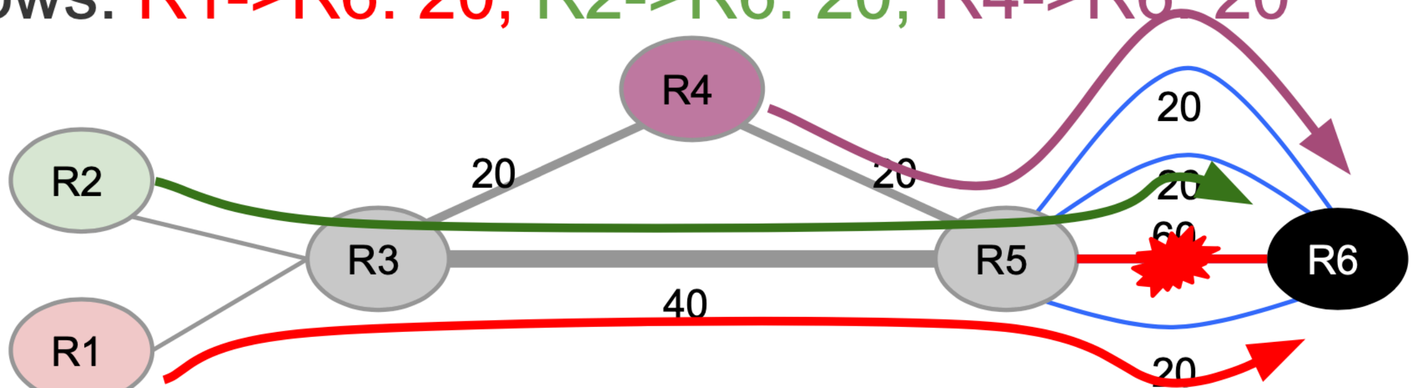
R5-R6 link fails

- 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

Slide content from Subhasree Mandal

Traffic Engineering: another example

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



Distributed Traffic Engineering Protocols

e.g. MPLS + RSVP

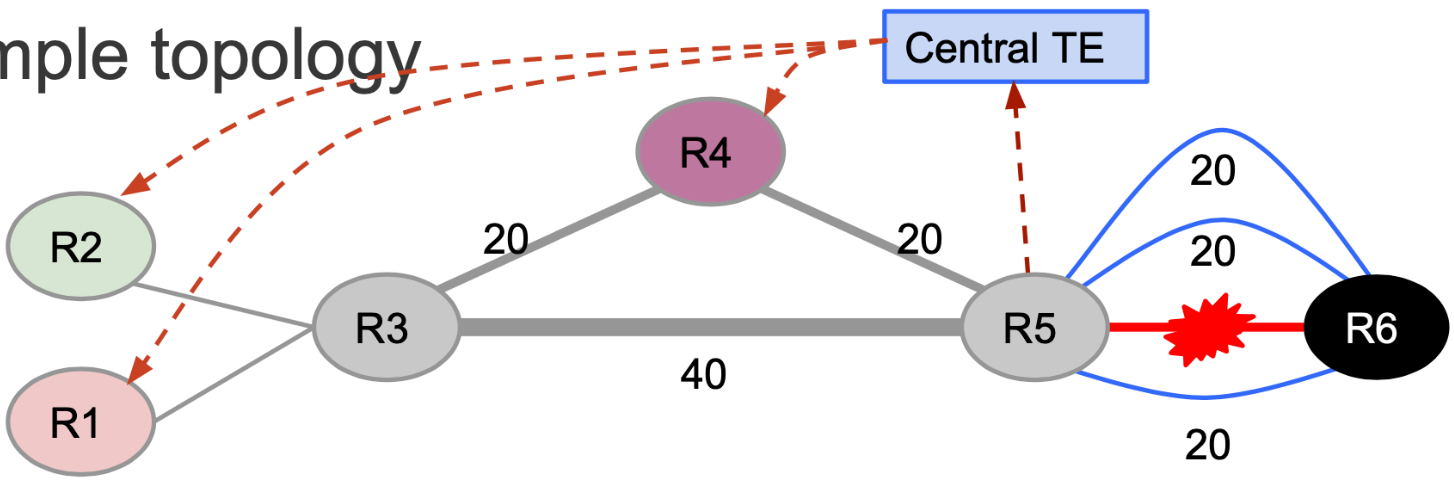
R5-R6 link fails

- 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!

Slide content from Subhasree Mandal

Traffic Engineering: another example

Simple topology



Centralized Traffic Engineering Protocols

Flows:

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

R5-R6 fails

- R5 informs TE, which programs routers in one shot

Slide content from Subhasree Mandal

Limitation of OpenFlow faced by B4

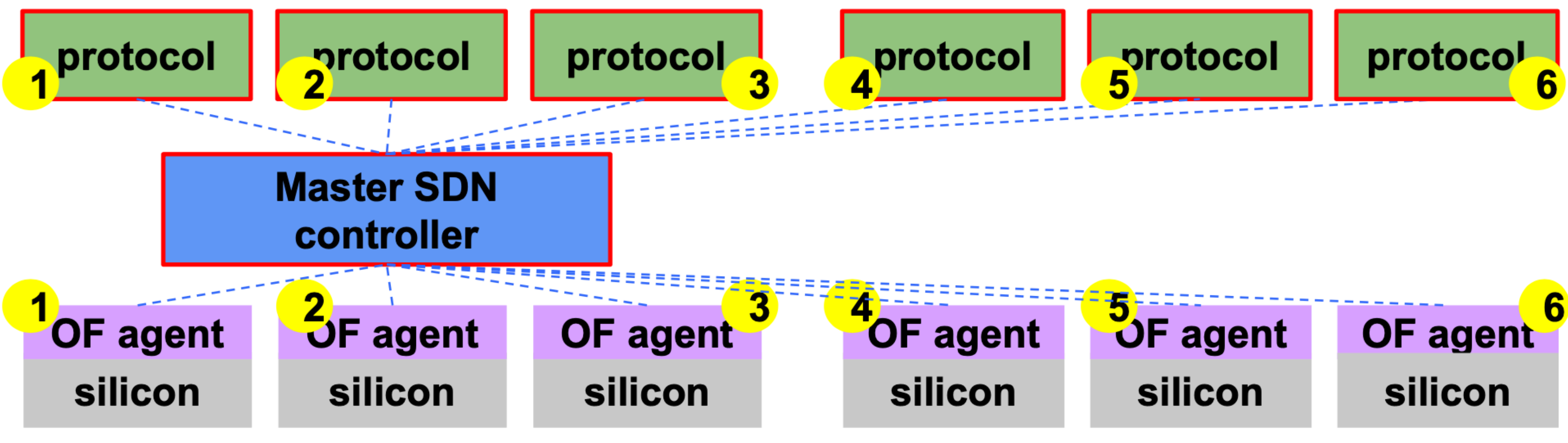
- Needs somewhat fancier switch behavior.
  - TE enforced using IP-in-IP tunnels.
  - Switches should understand how to parse headers for tunneling.
    - Encapsulate with tunnel IP at source ingress.
    - Decapsulate tunnel IP at destination egress.

- Developed their own switches that supported a slightly extended version of OpenFlow.

B4 SDN architecture

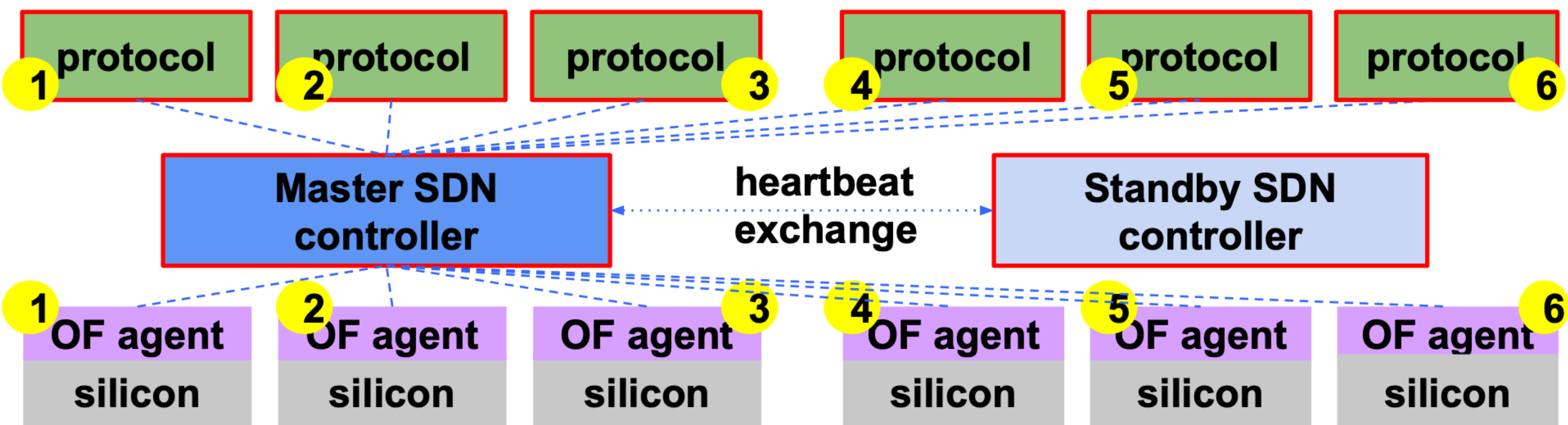
Slide content from Subhasree Mandal

B4 SDN architecture



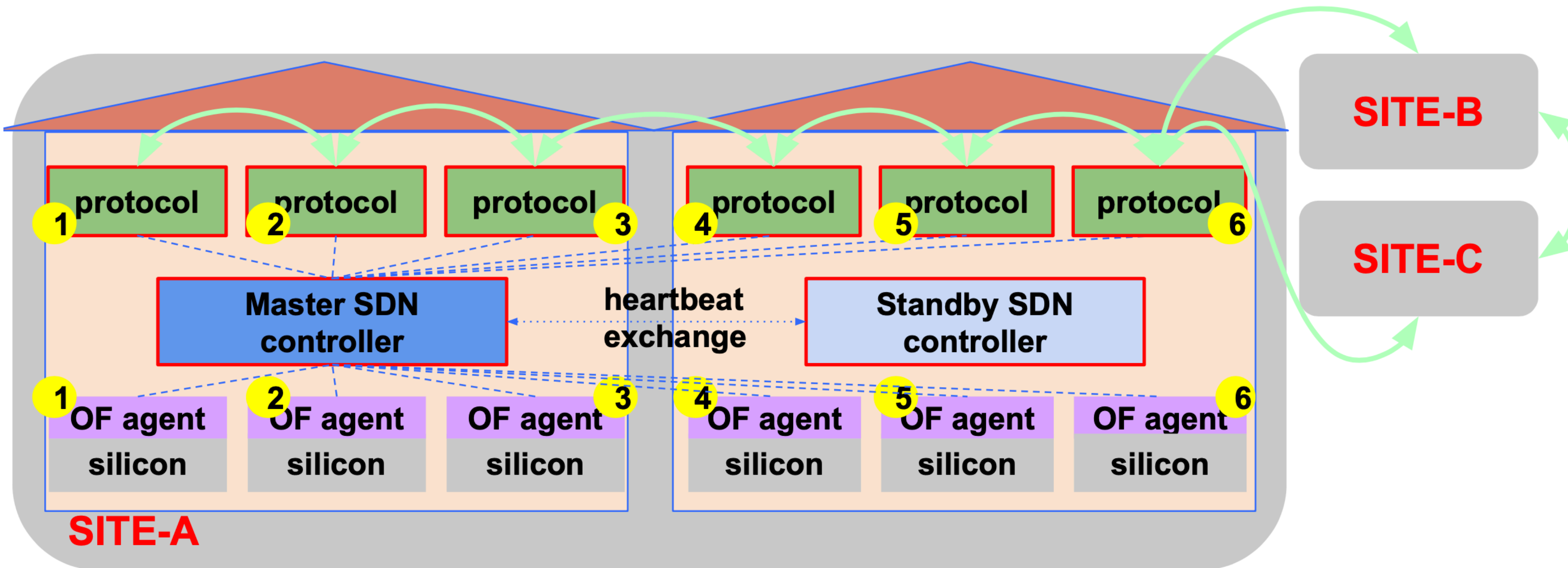
Slide content from Subhasree Mandal

B4 SDN architecture



Slide content from Subhasree Mandal

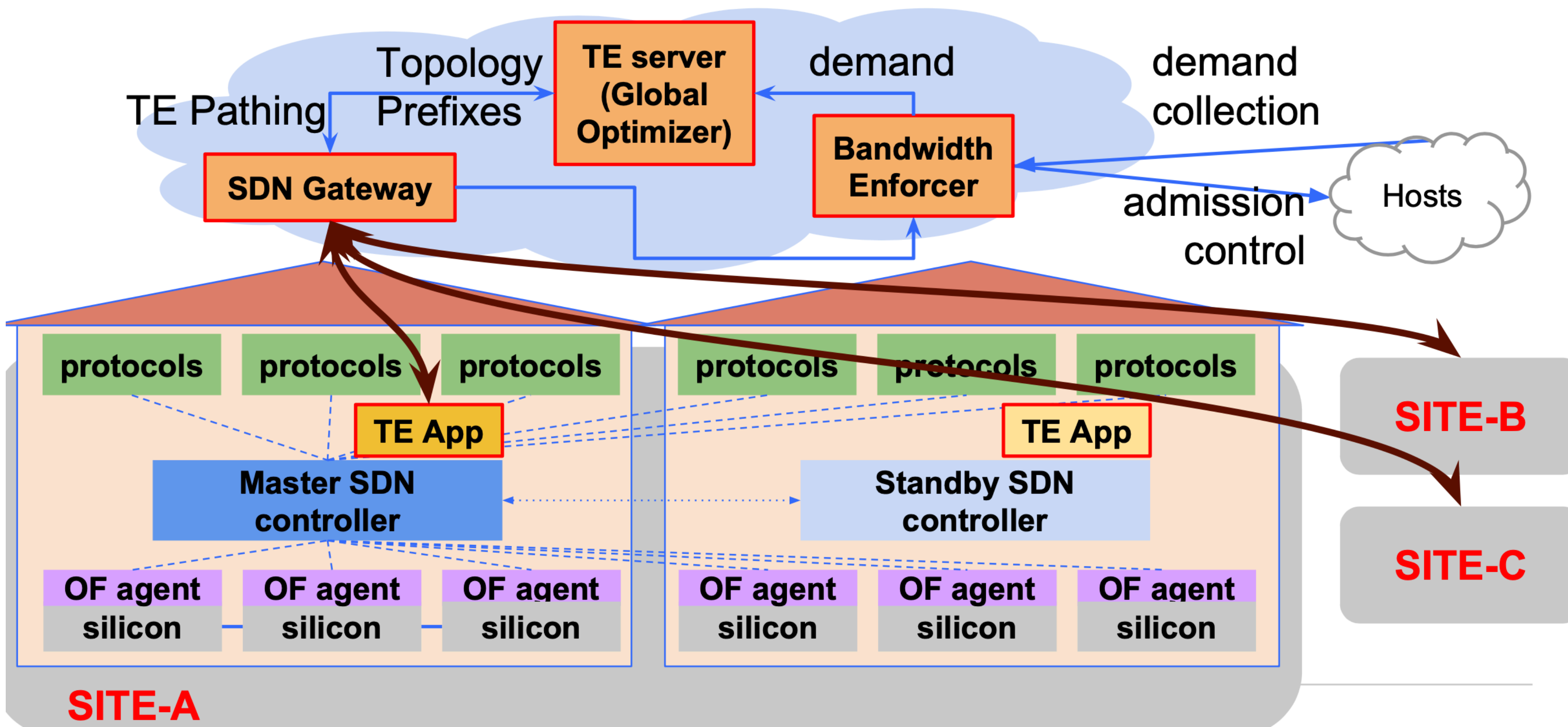
B4 SDN architecture



Unit of management is a site = fabric

Slide content from Subhasree Mandal

B4 SDN architecture

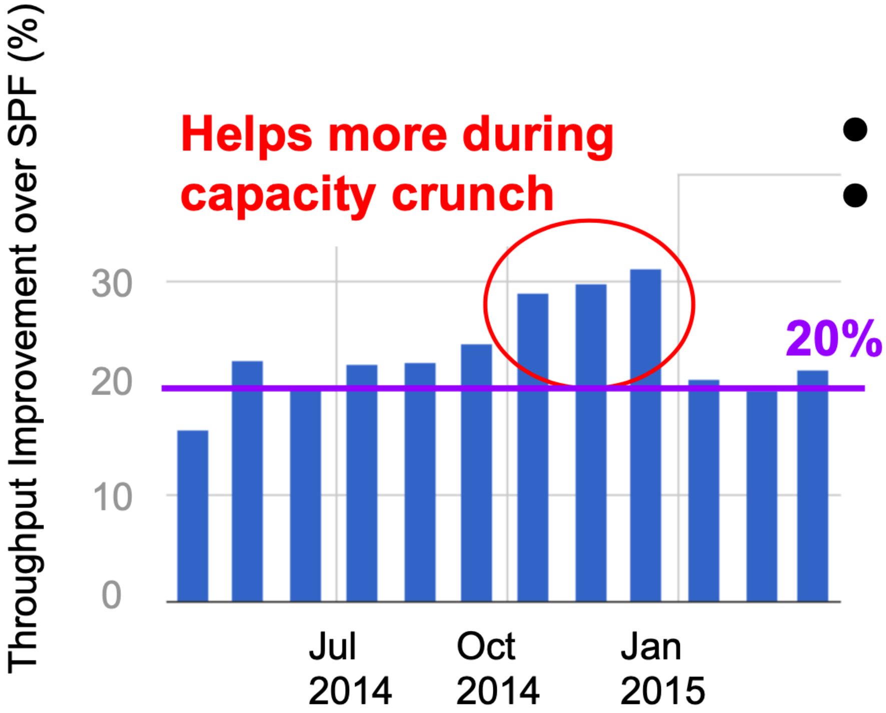


Slide content from Subhasree Mandal

B4 Traffic Engineering

- Objective: max-min fairness
  - A: 10Gbps, B: 5Gbps, total link capacity = 12Gbps
    - B = 5Gbps
    - A = 7 Gbps
  - A: 10Gbps, B: 5Gbps, C: 2Gbps, link capacity = 12Gbps
    - C = 2Gbps
    - B = 5Gbps
    - A = 5Gbps
  - Same demands, W(A) = 2, W(B) = 1, W(C) = 1, link capacity = 12Gbps
    - C = 2Gbps
    - B = 3.33Gbps
    - A = 6.67Gbps
- Greedy (water-filling) heuristic to do this across multiple paths.
- *Bandwidth Enforcer, SIGCOMM'15 has more details on TE algorithms*

Benefit of Centralized TE

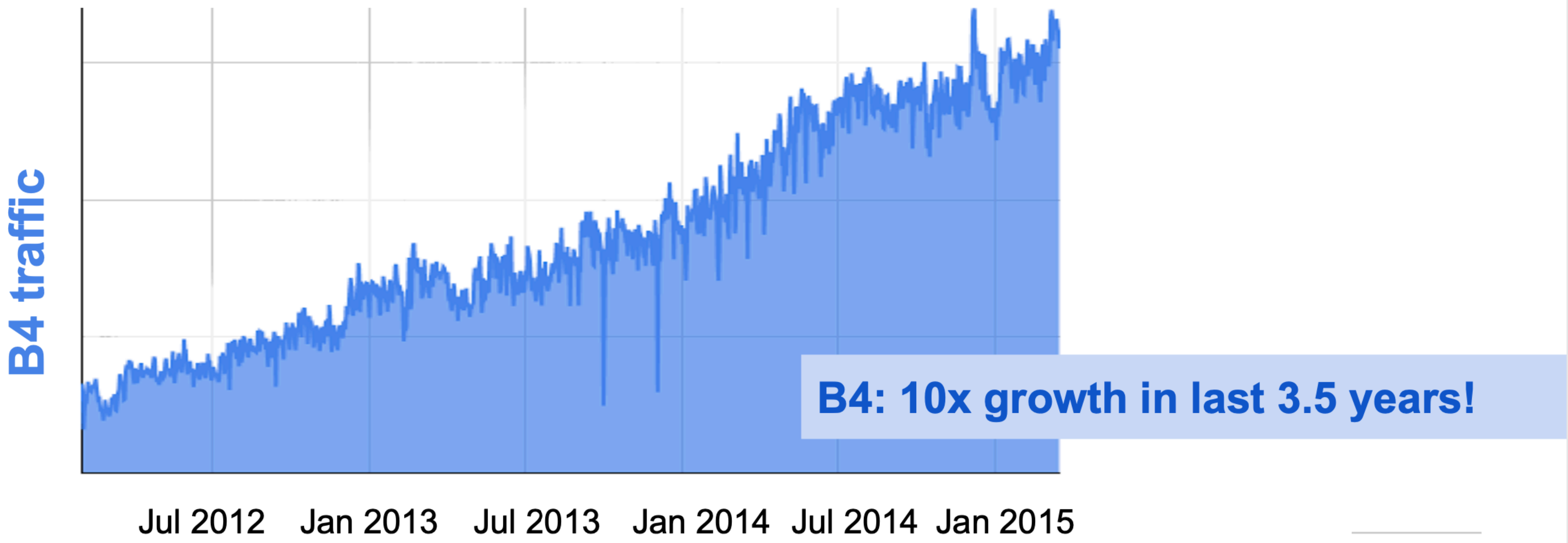


- ~20% increase in throughput over SPF
- Larger benefits during capacity crunch

Lowers the requirement for bandwidth provisioning

Slide content from Subhasree Mandal

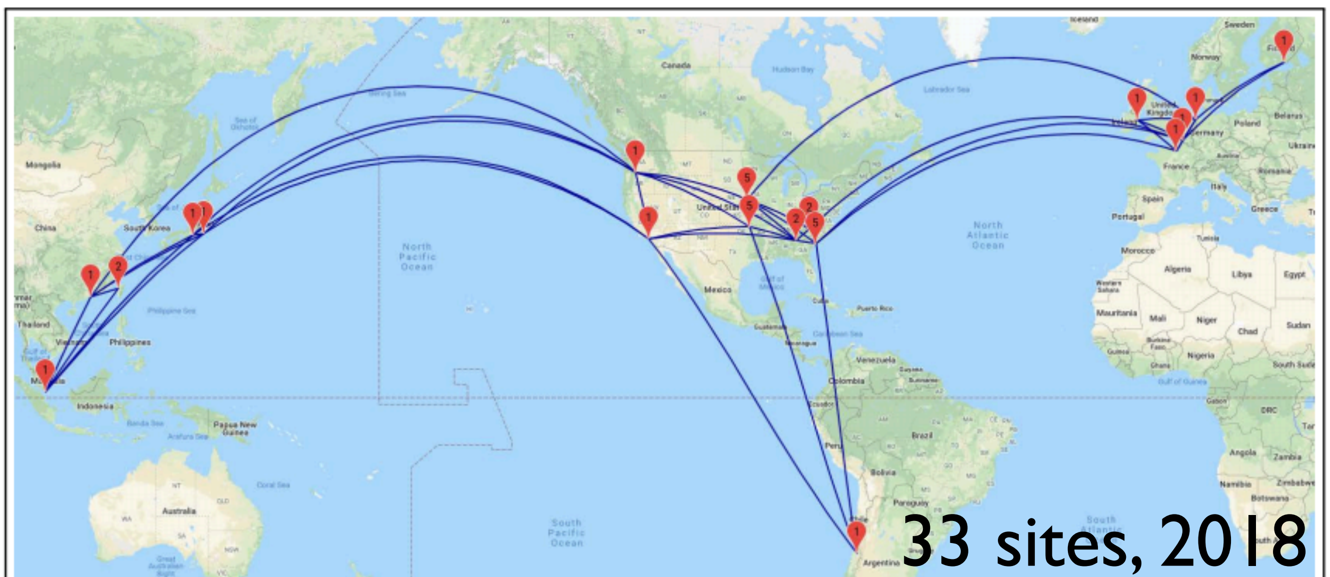
Benefit of Centralized TE



Slide content from Subhasree Mandal

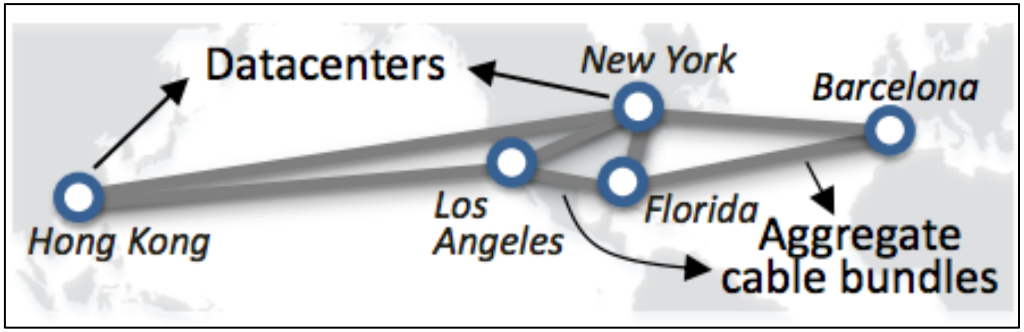
B4 and After: SIGCOMM'18

- Growth in traffic: more sites, larger sites, more paths.
  - Flat topology scales poorly:
    - Hierarchical topology at each site.
  - Hierarchical traffic engineering.



Another software-defined WAN

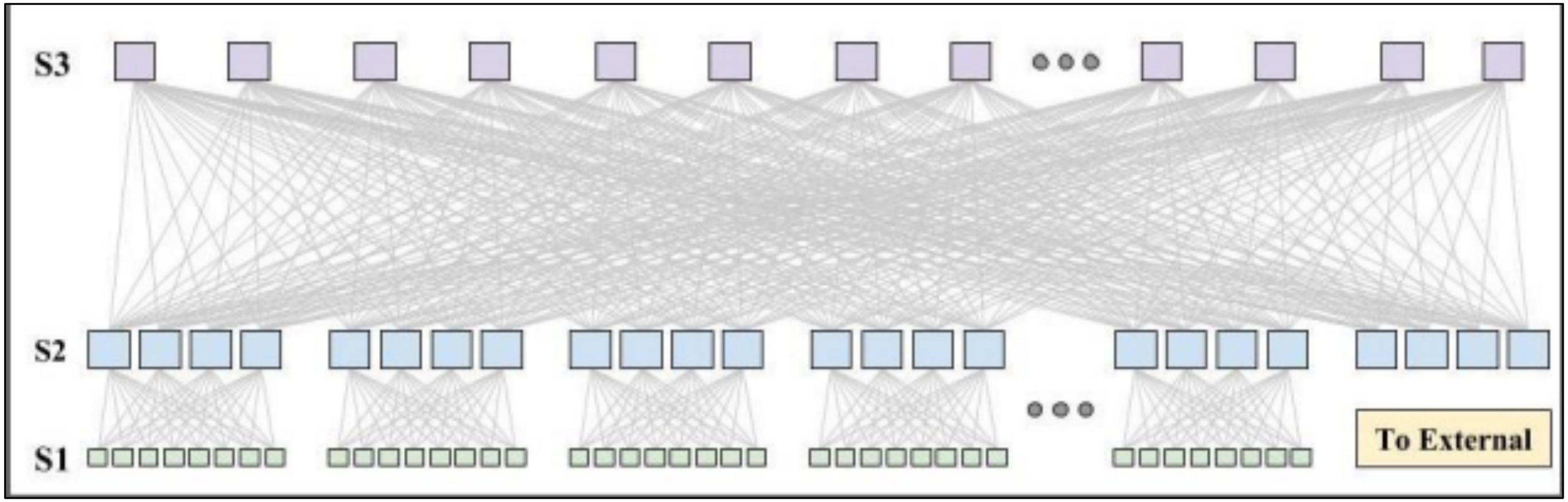
- SWAN (WAN connecting Microsoft's datacenter)
  - Goal: increase WAN link utilization.
    - Centralized and global traffic engineering.



Other SDN usecases at Google

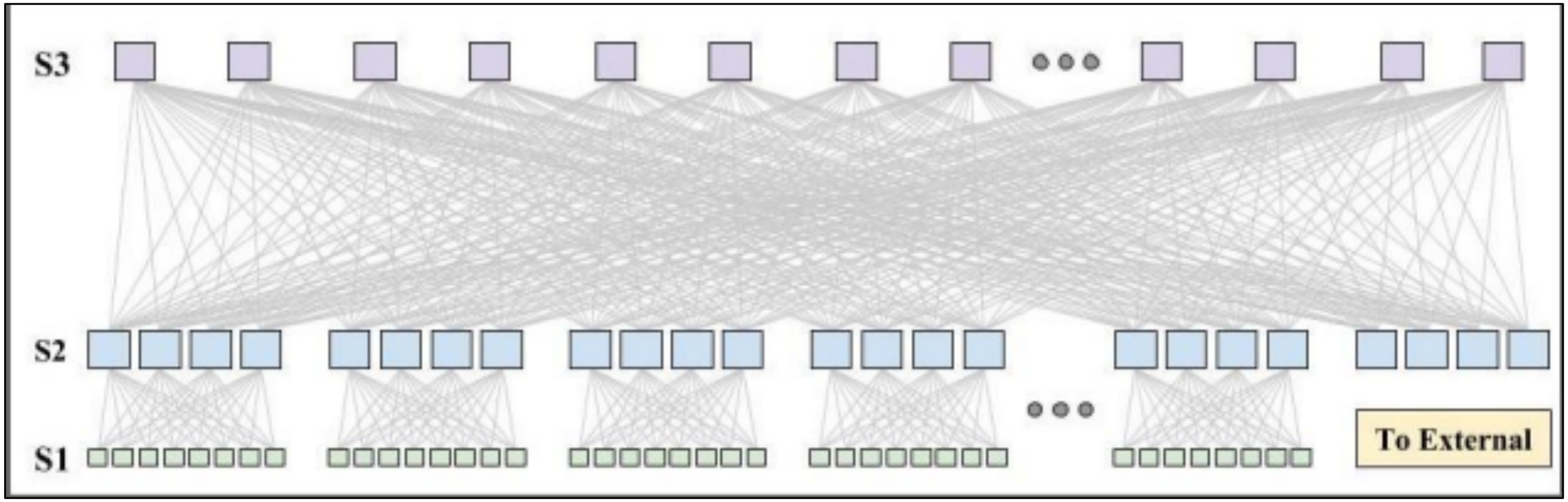
Datacenter routing

- Few 100-1000 switches distributed across clusters.
- High communication overhead for distributed routing.
- Symmetric topology: multipath equal cost forwarding.



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.

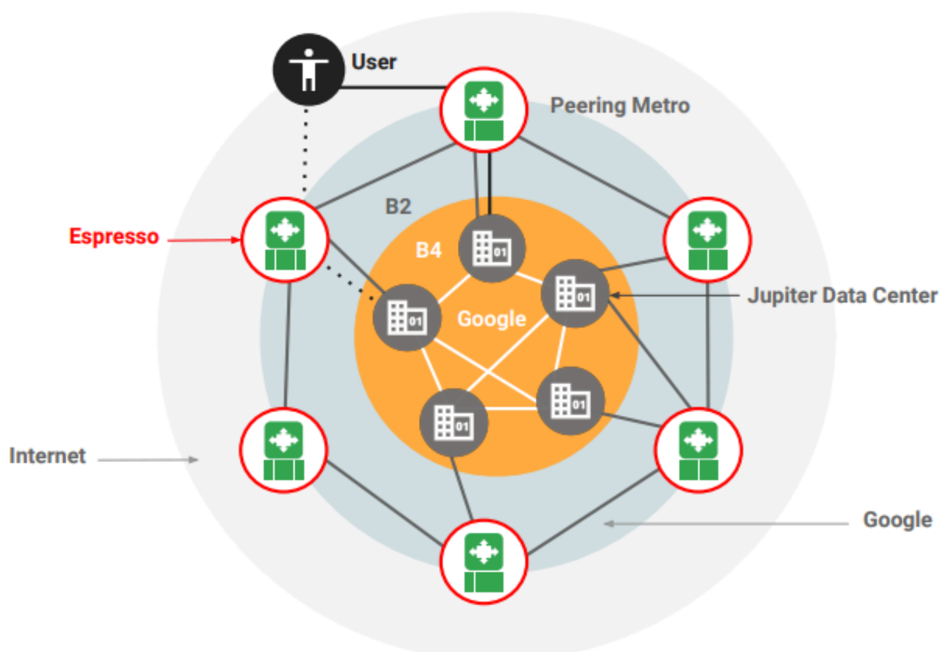


Datacenter routing

- 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

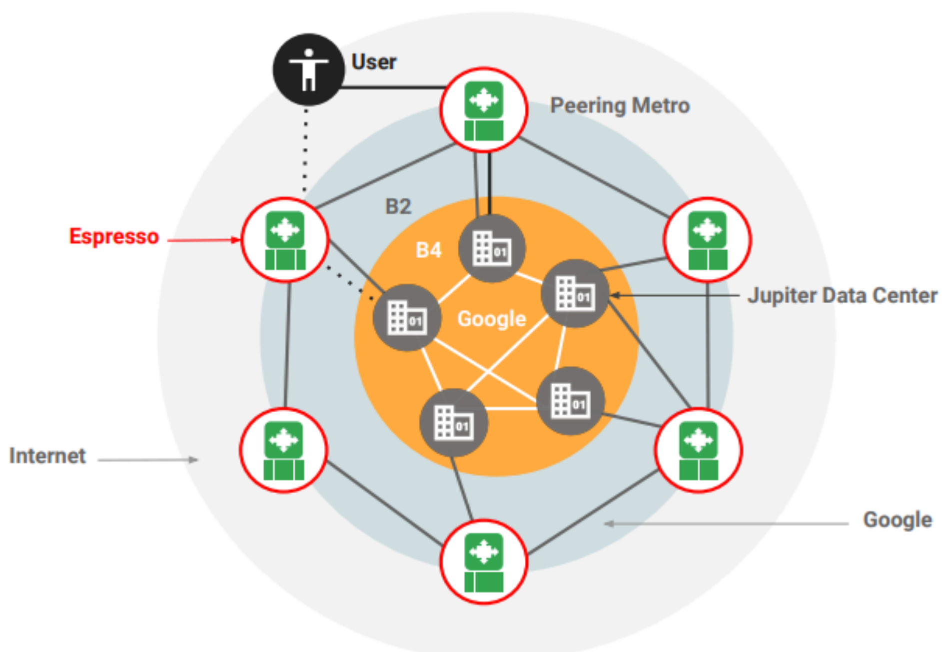
Policy enforcement at user-facing edge

- 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.



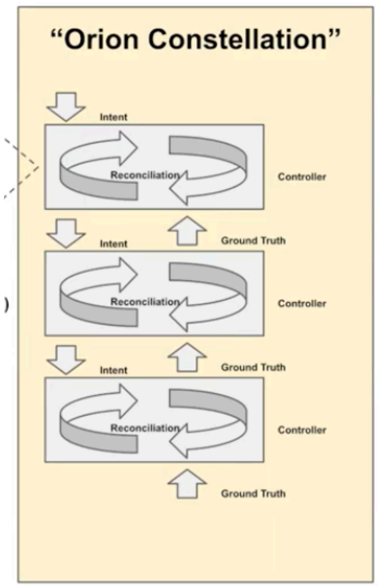
Policy enforcement at user-facing edge

- 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)