Jupiter rising: A decade of Clos topologies and centralized control in Google’s datacenter networks

Phil Sykes
Google NetOps (Dublin)
philsykes@google.com

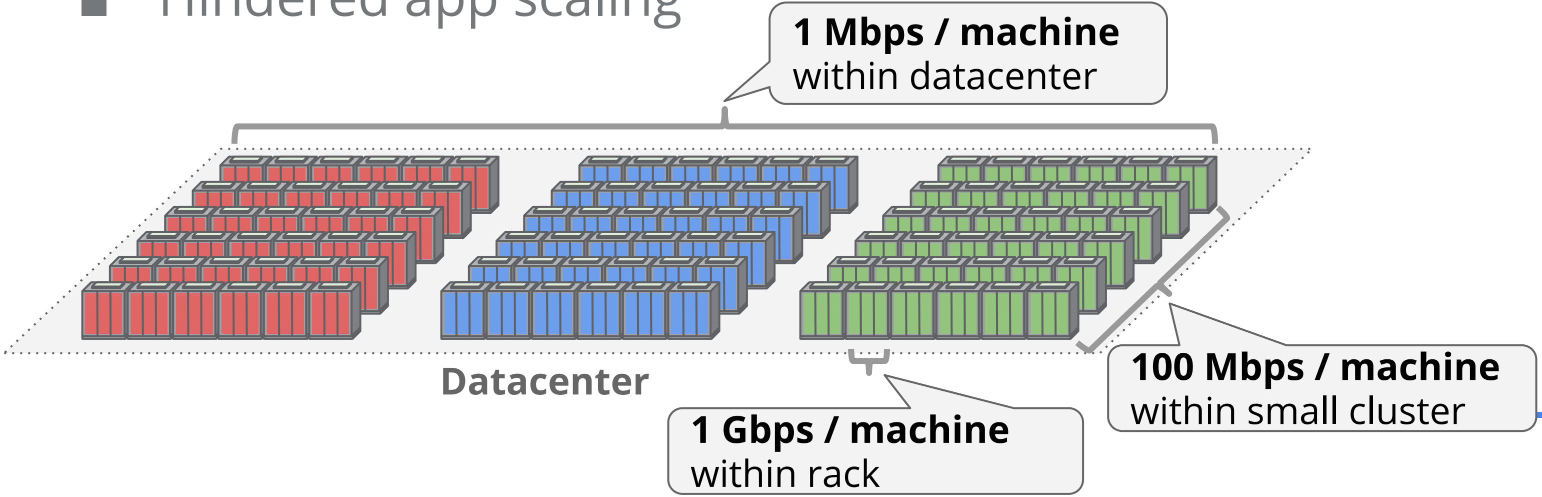
Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, and Amin Vahdat

On behalf of several teams in Google:

Platforms Networking Hardware and Software Development, Platforms SQA, Mechanical Engineering, Cluster Engineering, NetOps, Global Infrastructure Group (GIG), and SRE.

Grand challenge for datacenter networks

- Tens of thousands of servers interconnected in clusters
- *Islands of bandwidth* a key bottleneck for Google a decade ago
    - Engineers struggled to optimize for b/w locality
    - Stranded compute/memory resources
    - Hindered app scaling

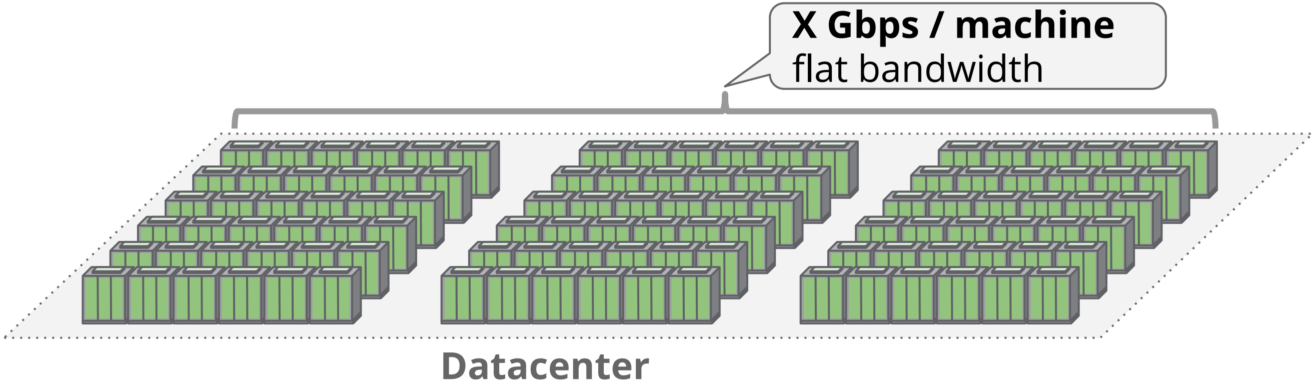

1 Mbps / machine
within datacenter

100 Mbps / machine
within small cluster

1 Gbps / machine
within rack

Grand challenge for datacenter networks

- **Challenge: Flat b/w profile across all servers**
  - Simplify job scheduling (remove locality)
  - Save significant resources via better bin-packing
  - Allow application scaling



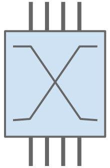
Motivation

- **Traditional network architectures**
  - Cost prohibitive
  - Could not keep up with our bandwidth demands
  - Operational complexity of “box-centric” deployment

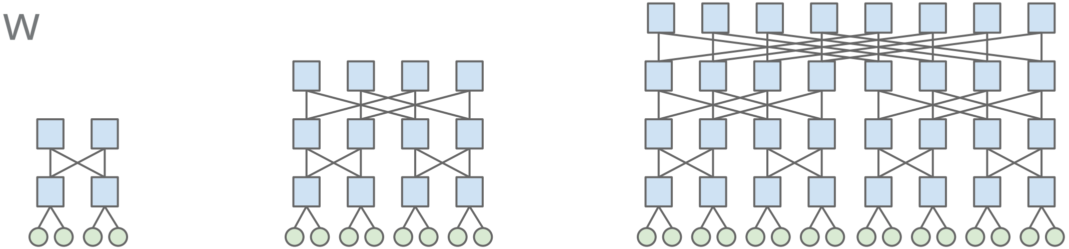
- **Opportunity: A datacenter is a single administrative domain**
  - One organization designs, deploys, controls, operates the n/w
  - ...And often also the servers

Three pillars that guided us

Merchant silicon: General purpose, commodity priced, off the shelf switching components



Clos topologies: Accommodate low radix switch chips to scale nearly arbitrarily by adding stages



Centralized control / management

SDN: The early days

- Control options
    - Protocols: OSPF, ISIS, BGP, etc; *Box-centric* config/management
    - Build our own

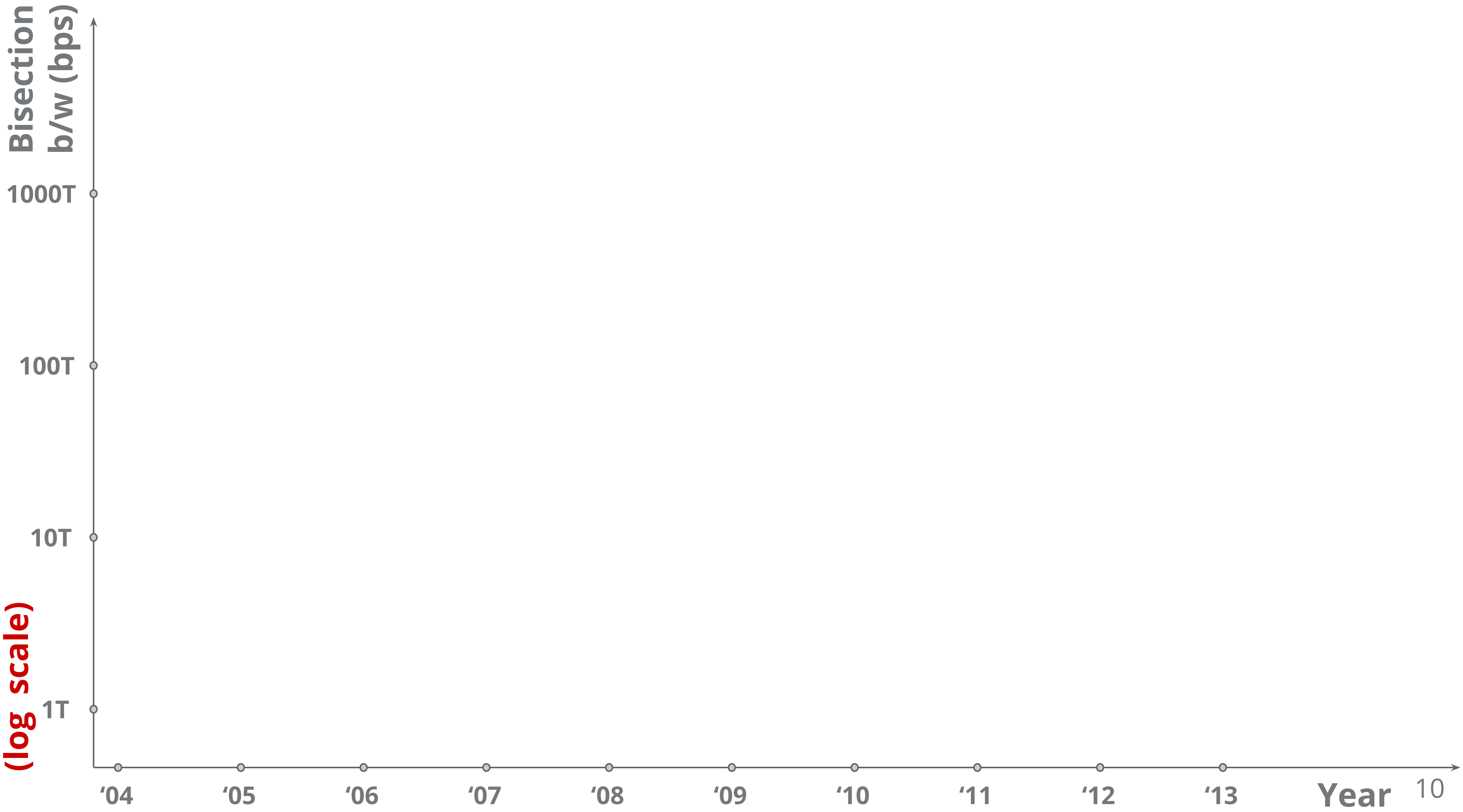
- Reasons we chose to build our own central control/management:
    - Limited support for **multipath forwarding**
    - No **robust** open source stacks
    - **Broadcast** protocol scalability a concern at scale
    - Network **manageability** painful with individual switch configs

Challenges faced in building our own solution

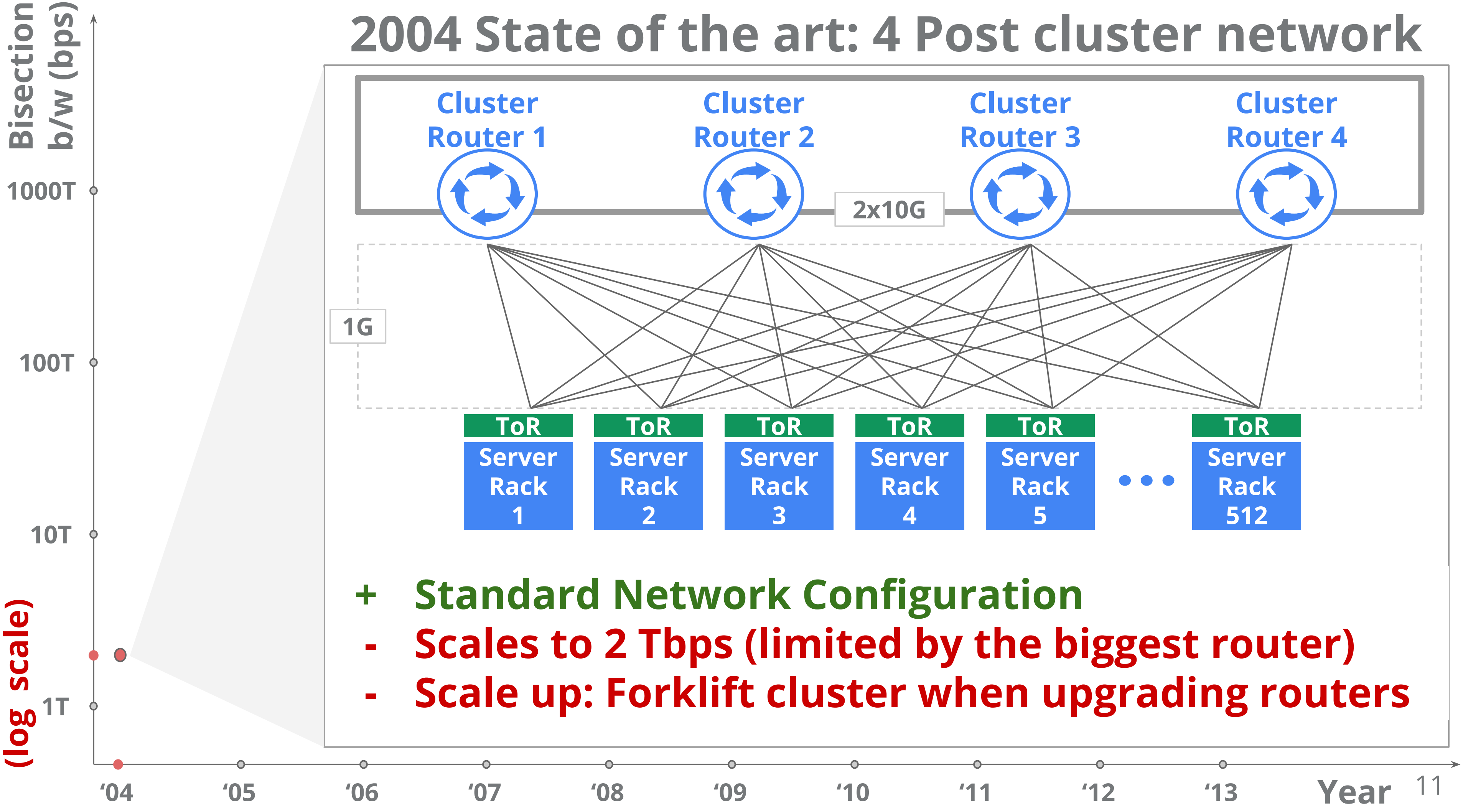
- **Topology and deployment**
  - Introducing our network to production
  - Unmanageably high number of cables/fiber
  - Cluster-external burst b/w demand
- **Control and management**
  - Operating at huge scale
  - Routing scalability / routing with massive multipath
  - Interop with external vendor gear
- **Performance and reliability**
  - Small on-chip buffers
  - High availability from cheap/less reliable components

Outline

- Motivation
- **Network evolution**
- Centralized control / management
- Experience


Bisection b/w (bps)
1000T
100T
10T
1T
(log scale)
'04
'05
'06
'07
'08
'09
'10
'11
'12
'13
Year

2004 State of the art: 4 Post cluster network

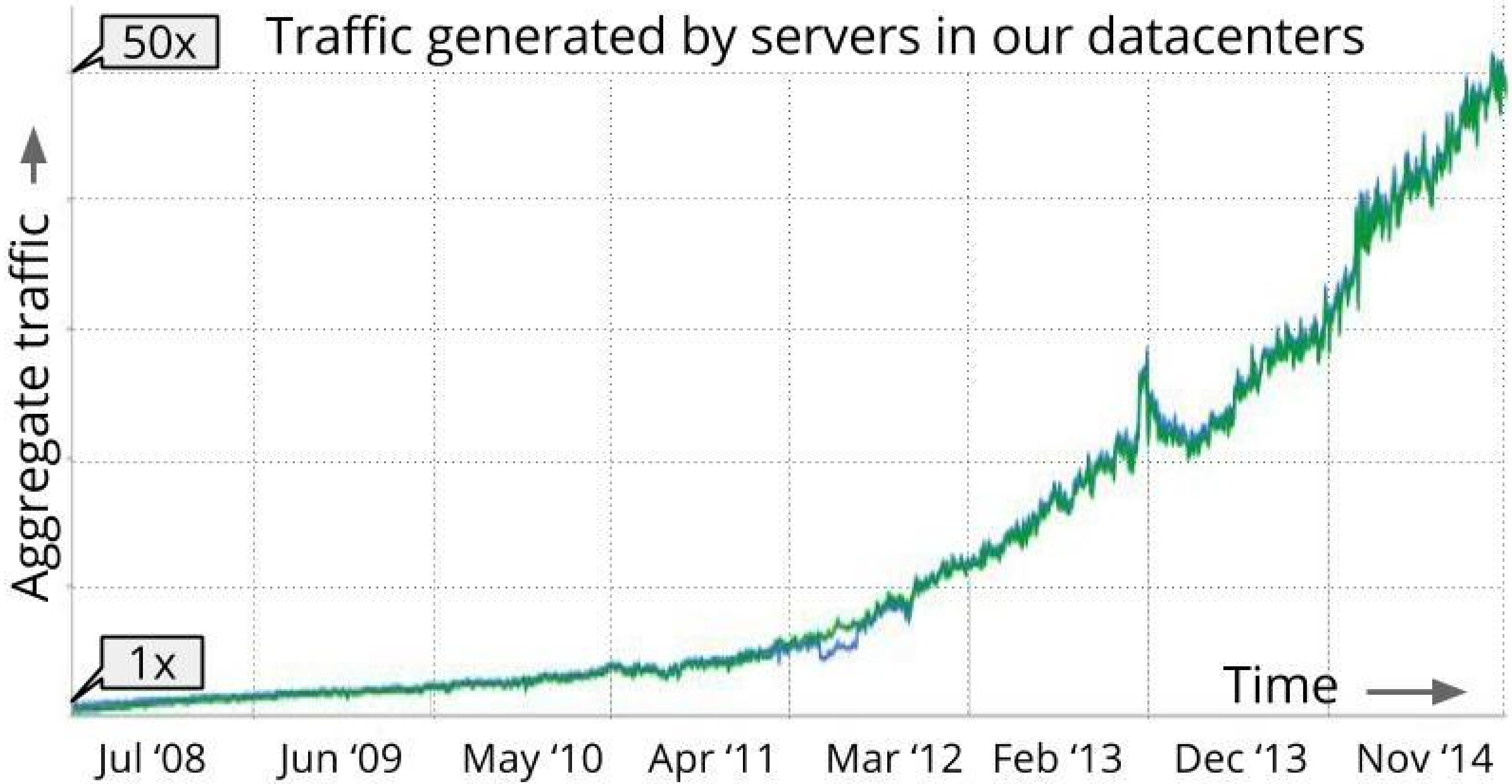


+ Standard Network Configuration

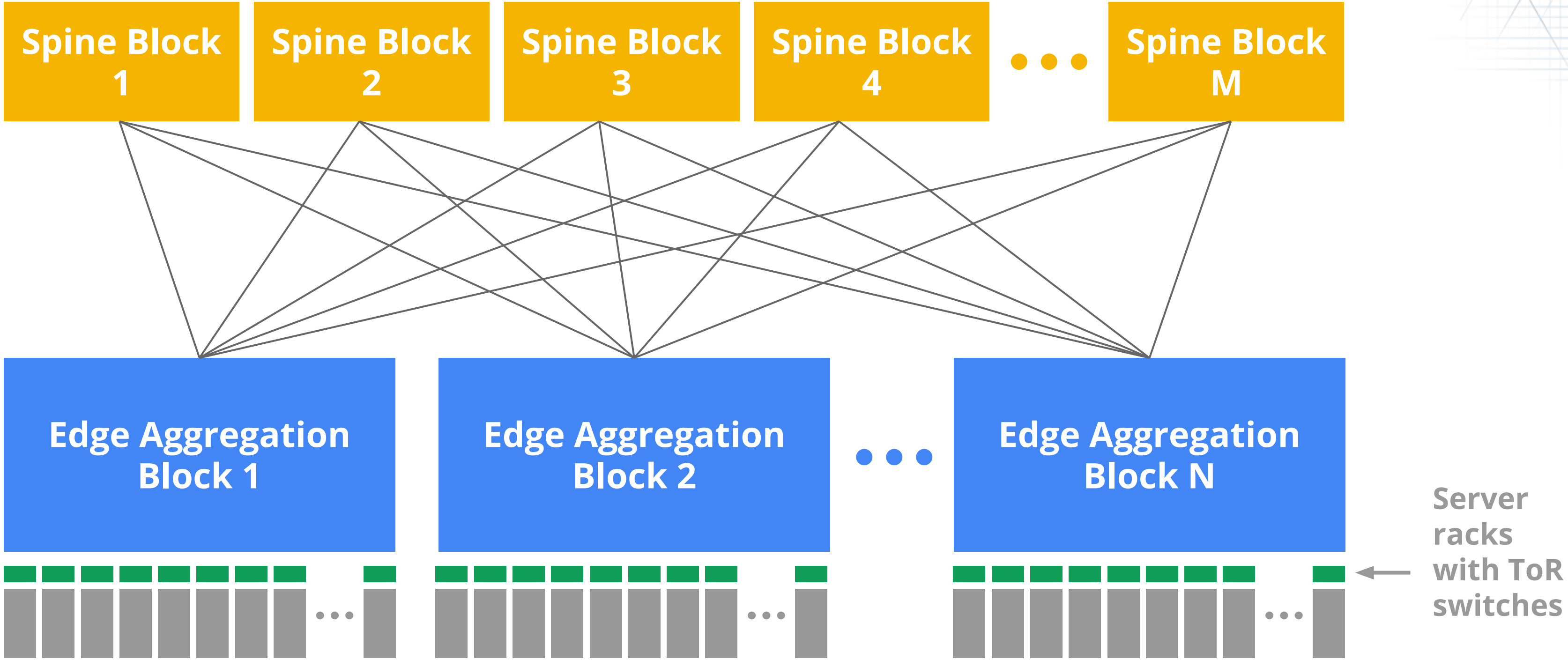
- Scales to 2 Tbps (limited by the biggest router)

- Scale up: Forklift cluster when upgrading routers

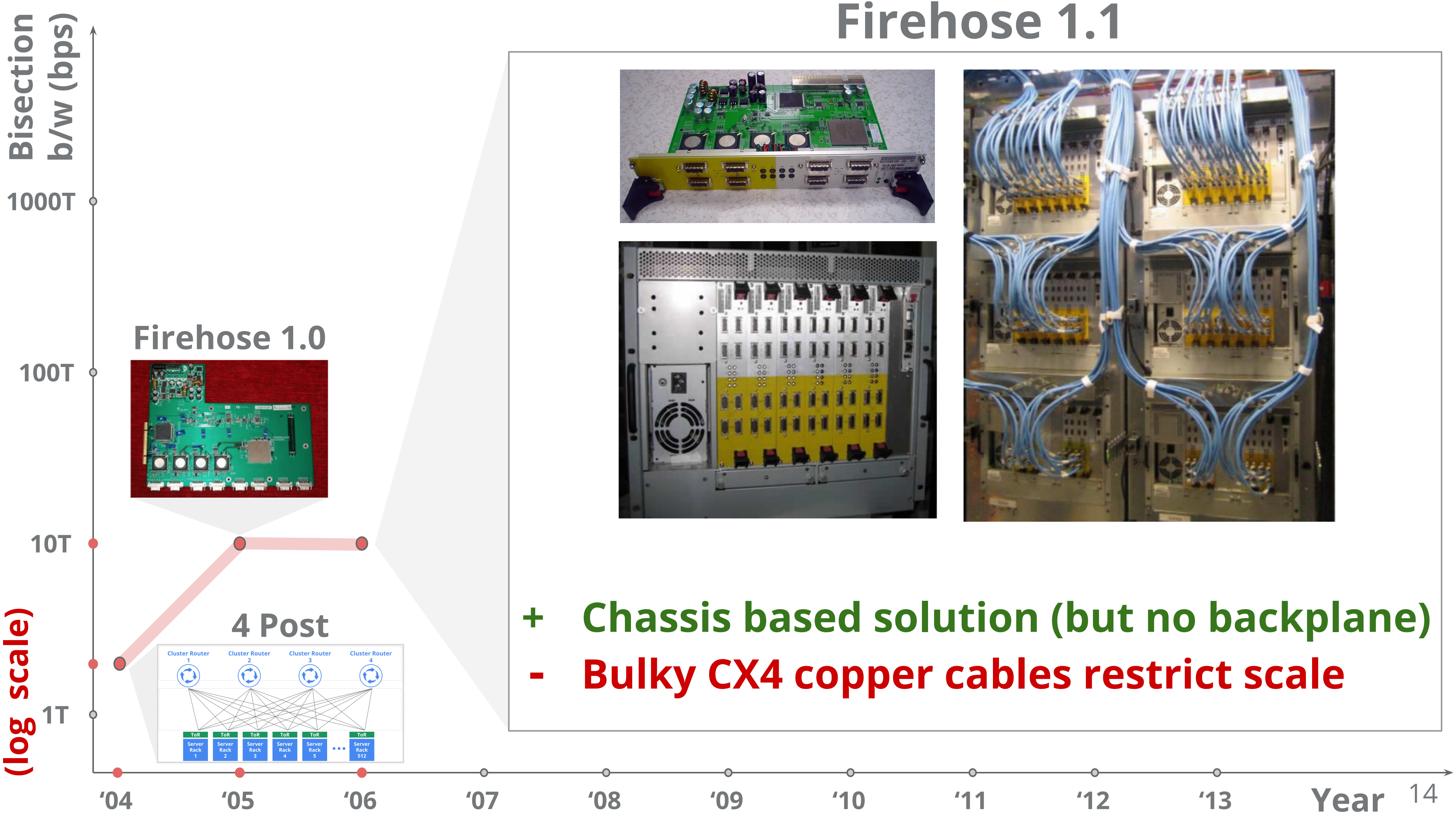
DCN bandwidth growth demanded much more



Five generations of Clos for Google scale



Firehose 1.1



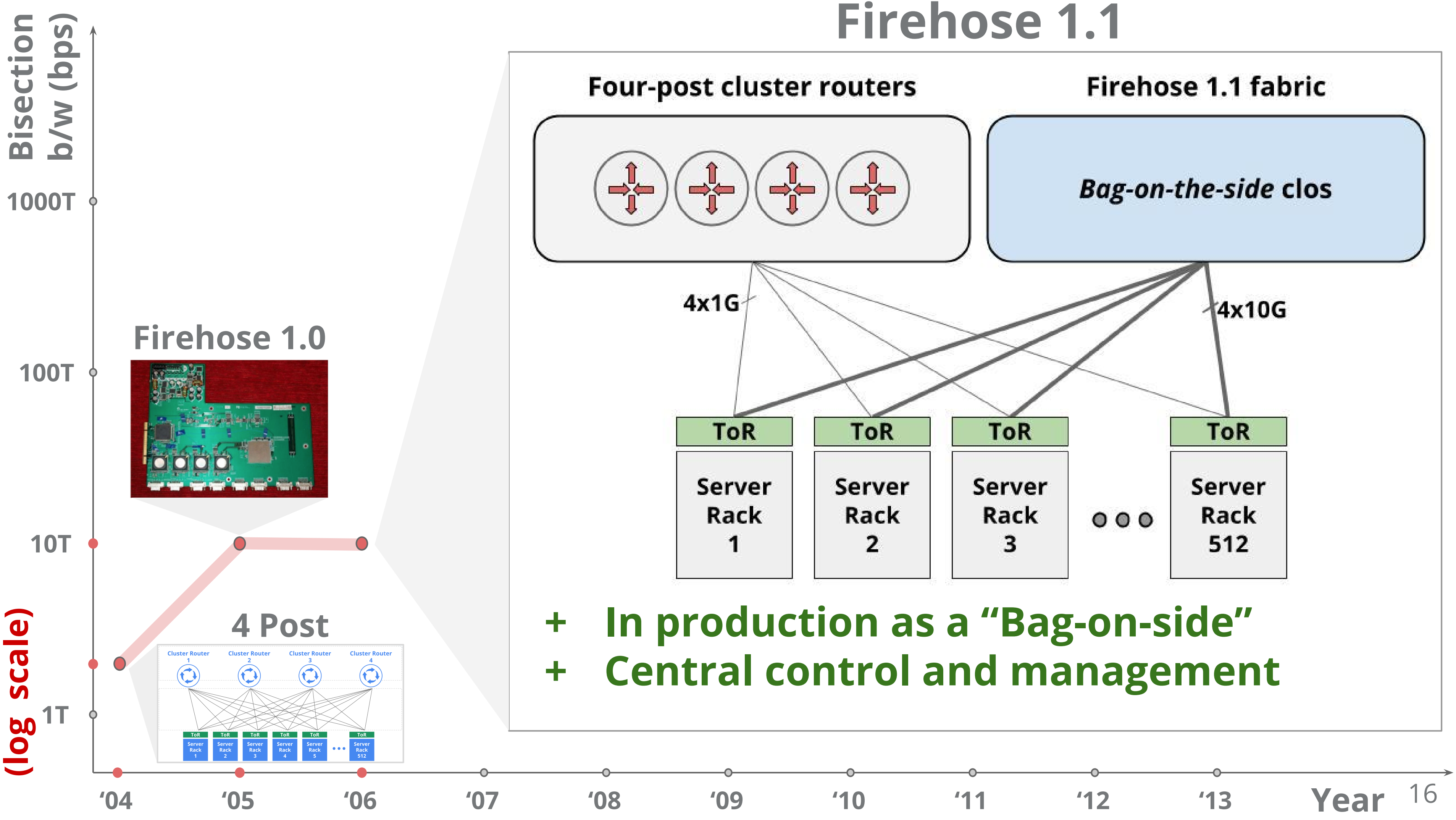
+ Chassis based solution (but no backplane)

- Bulky CX4 copper cables restrict scale

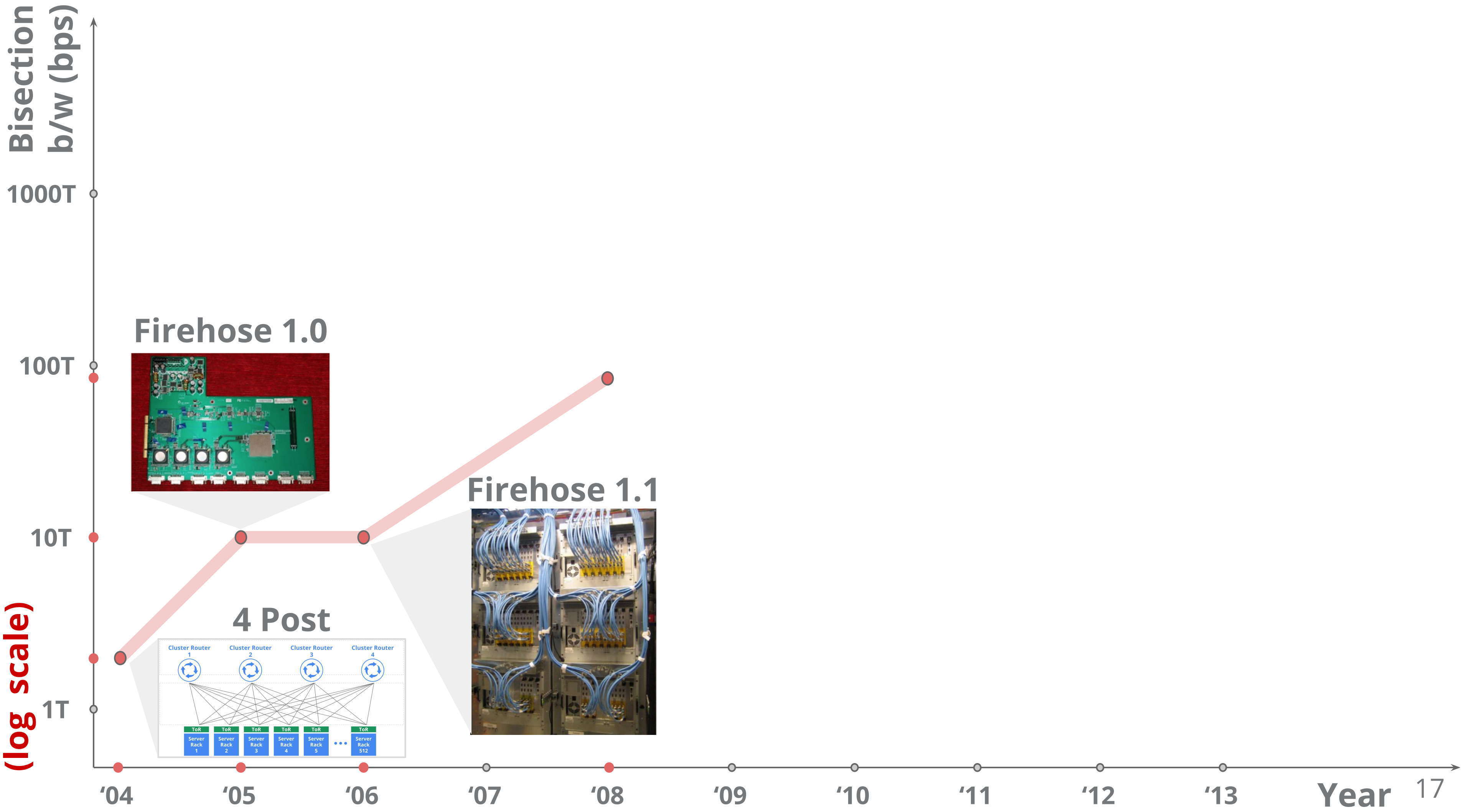
Challenges faced in building our own solution

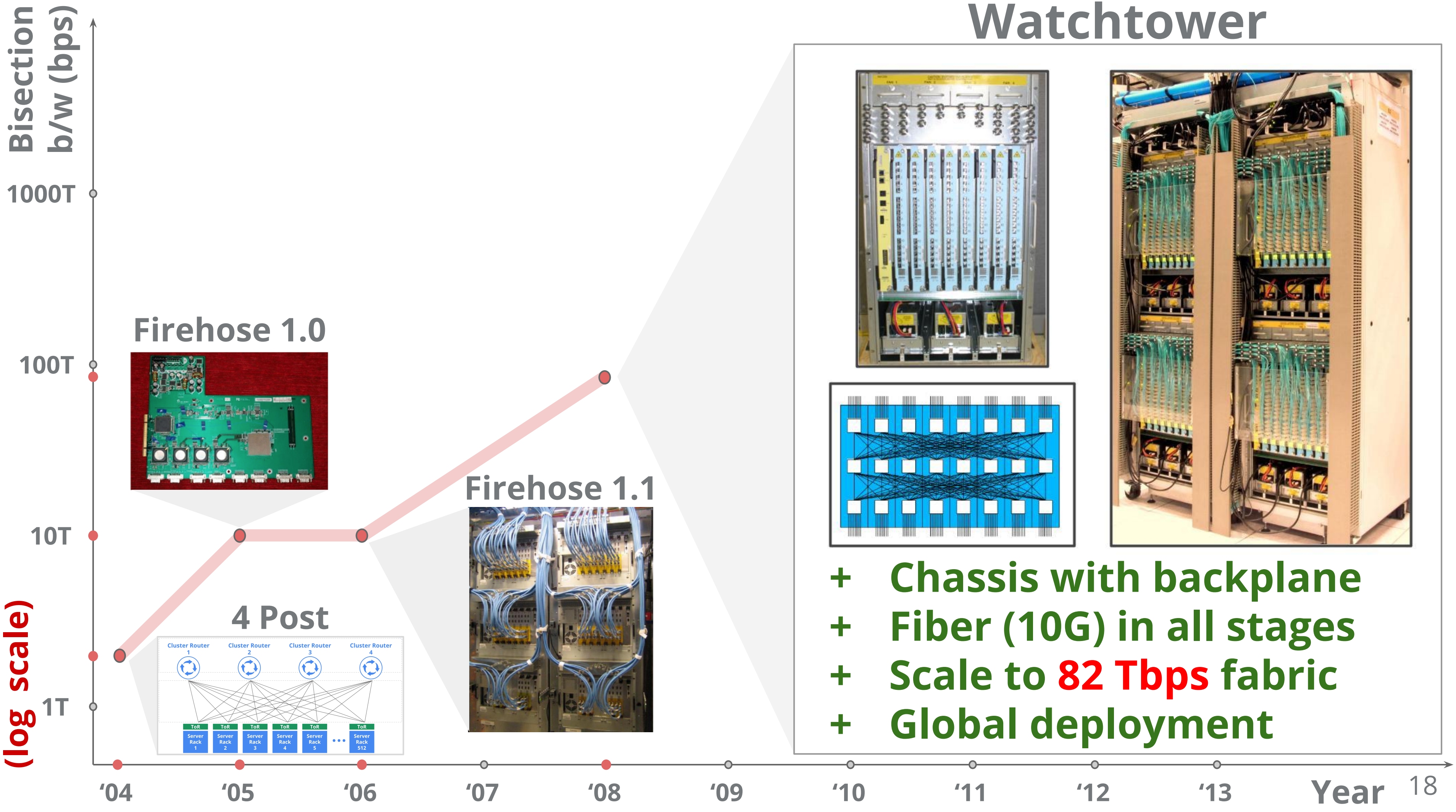
- **Topology and deployment**
  - **Introducing our network to production**
  - Unmanageably high number of cables/fiber
  - Cluster-external burst b/w demand
- **Control and management**
  - Operating at huge scale
  - Routing scalability / routing with massive multipath
  - Interop with external vendor gear
- **Performance and reliability**
  - Small on-chip buffers
  - High availability from cheap/less reliable components

Firehose 1.1



- + In production as a "Bag-on-side"
- + Central control and management


Bisection b/w (bps)
(log scale)
1000T
100T
10T
1T
Firehose 1.0
Firehose 1.1
4 Post
Cluster Router 1
Cluster Router 2
Cluster Router 3
Cluster Router 4
ToR
Server Rack 1
Server Rack 2
Server Rack 3
Server Rack 4
Server Rack 5
Server Rack 512
'04
'05
'06
'07
'08
'09
'10
'11
'12
'13
Year



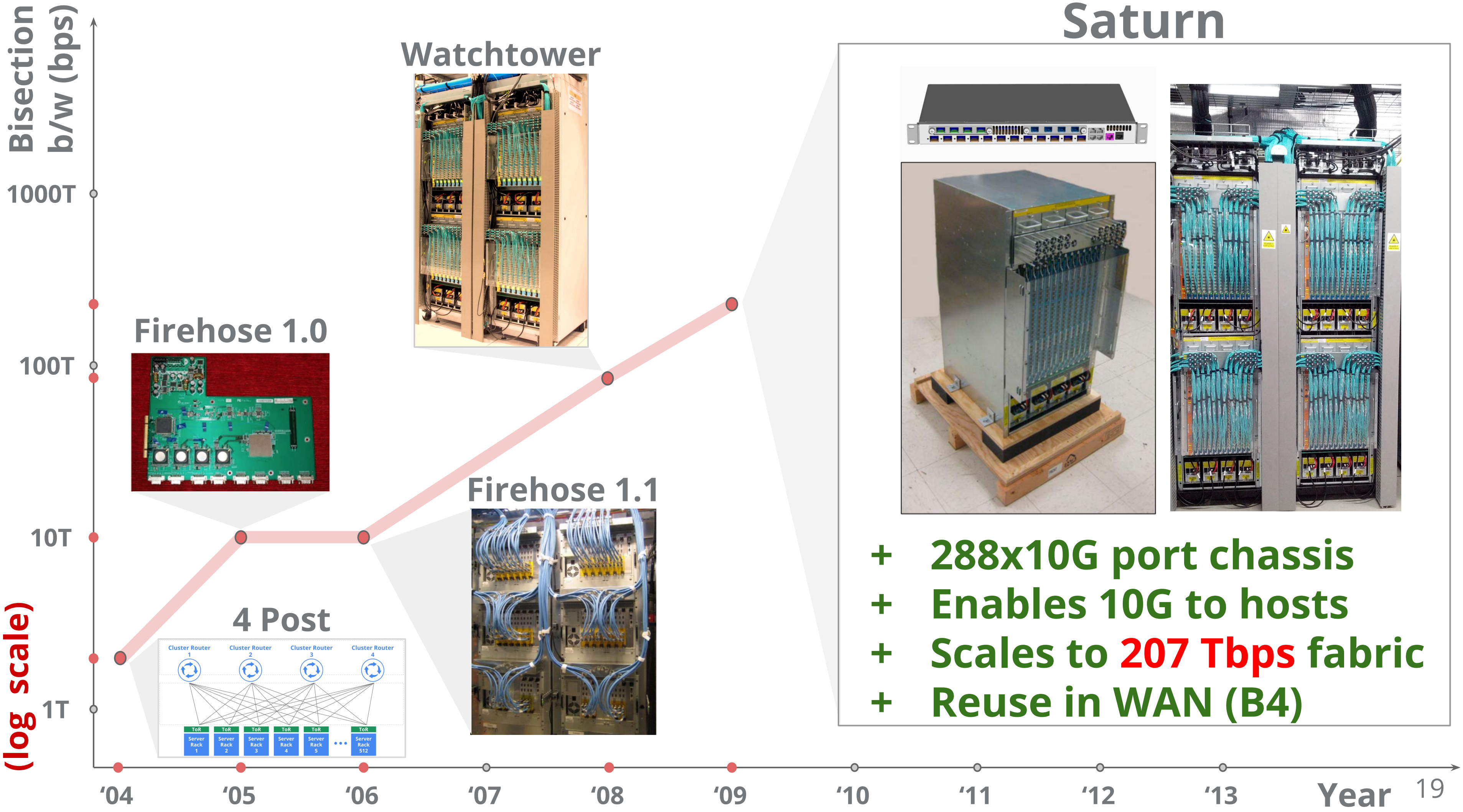
Firehose 1.0

4 Post

Firehose 1.1

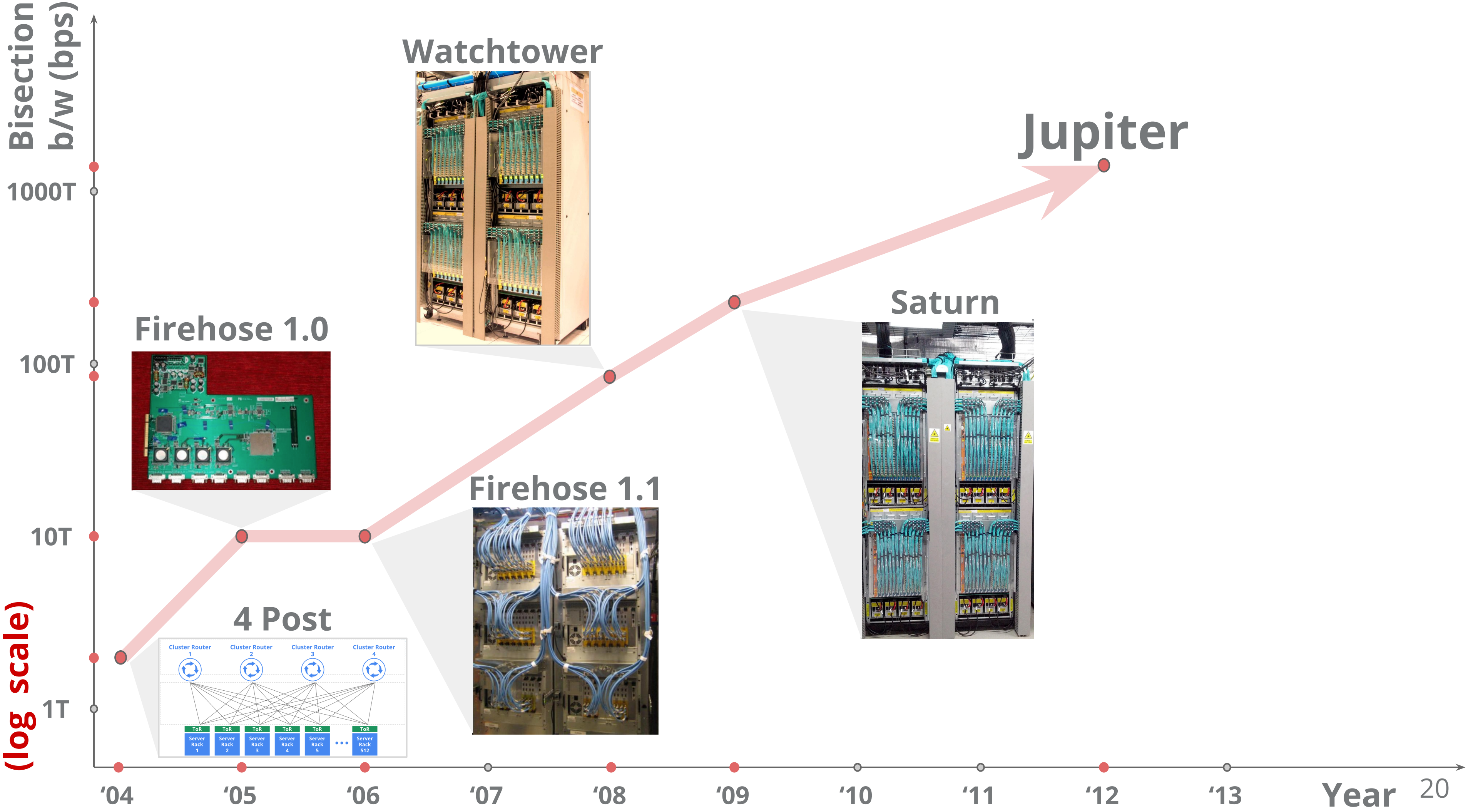
Watchtower

- + Chassis with backplane
- + Fiber (10G) in all stages
- + Scale to 82 Tbps fabric
- + Global deployment

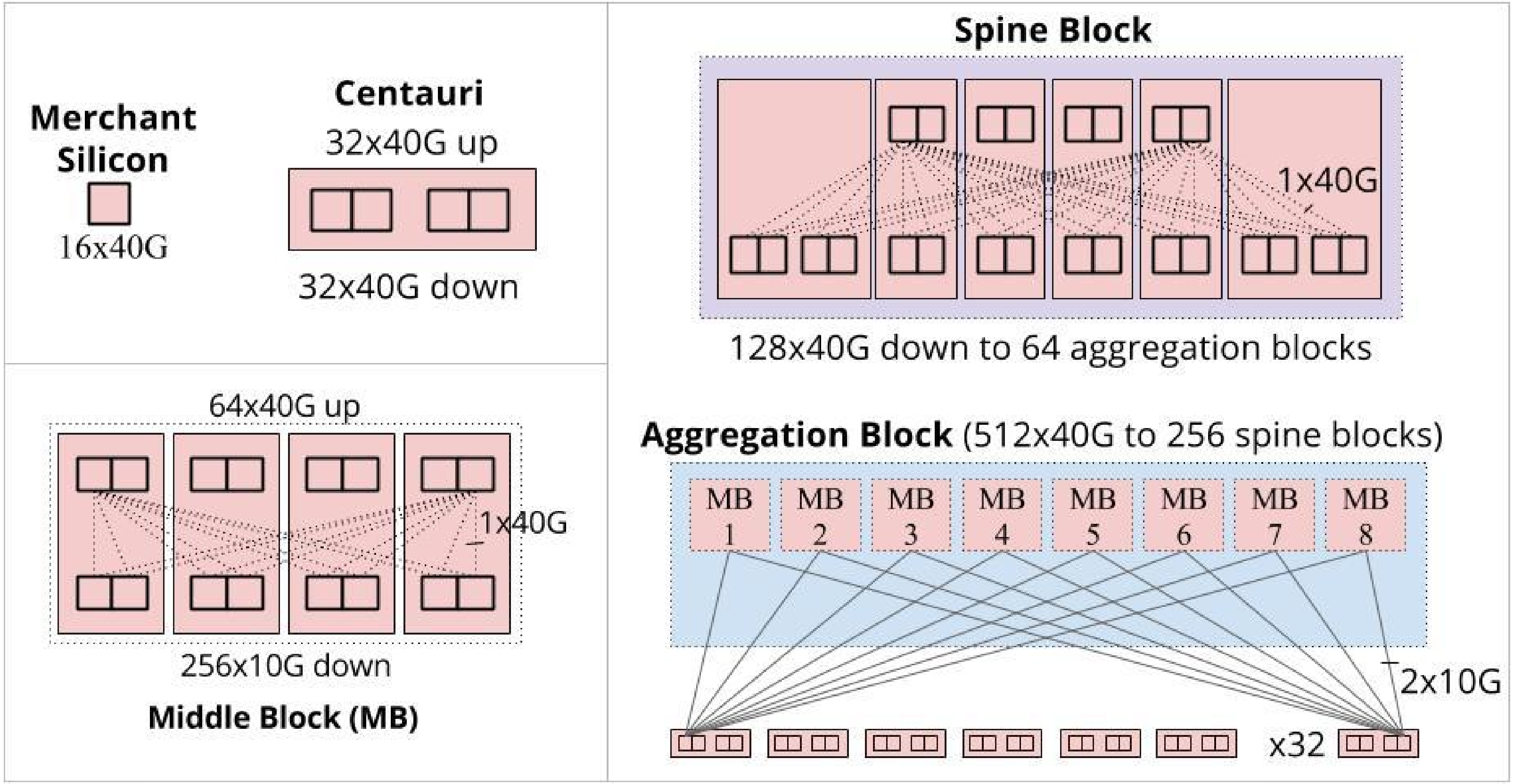


Saturn

- + 288x10G port chassis
- + Enables 10G to hosts
- + Scales to 207 Tbps fabric
- + Reuse in WAN (B4)


Bisection b/w (bps)
(log scale)
1000T
100T
10T
1T
Watchtower
Jupiter
Saturn
Firehose 1.0
Firehose 1.1
4 Post
Cluster Router 1
Cluster Router 2
Cluster Router 3
Cluster Router 4
ToR
Server Rack 1
Server Rack 2
Server Rack 3
Server Rack 4
Server Rack 5
...
Server Rack 512
'04
'05
'06
'07
'08
'09
'10
'11
'12
'13
Year

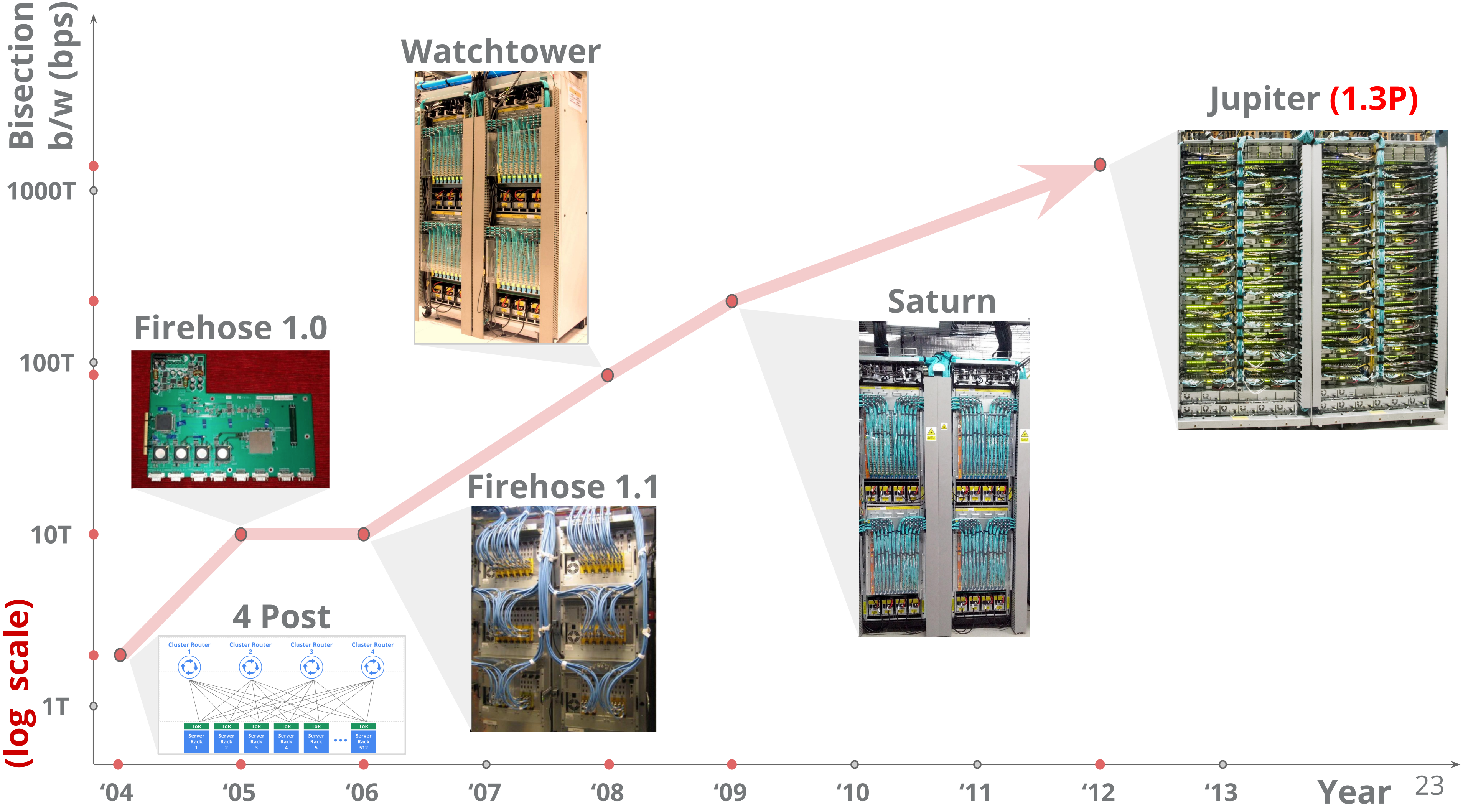
Jupiter topology



+ Scales out building wide 1.3 Pbps

Jupiter racks

- + Enables 40G to hosts
- + External control servers
- + OpenFlow


Bisection b/w (bps)
(log scale)
1000T
100T
10T
1T
'04
'05
'06
'07
'08
'09
'10
'11
'12
'13
Year
Watchtower
Jupiter (1.3P)
Firehose 1.0
Saturn
Firehose 1.1
4 Post
Cluster Router 1
Cluster Router 2
Cluster Router 3
Cluster Router 4
ToR
Server Rack 1
Server Rack 2
Server Rack 3
Server Rack 4
Server Rack 5
...
Server Rack 512

Challenges faced in building our own solution

- **Topology and deployment**
  - Introducing our network to production
  - Unmanageably high number of cables/fiber
  - Cluster-external burst b/w demand
- **Control and management**
  - **Operating at huge scale**
  - **Routing scalability / routing with massive multipath**
  - Interop with external vendor gear
- **Performance and reliability**
  - Small on-chip buffers
  - High availability from cheap/less reliable components

Network control and config

New conventional wisdom from engineering systems at scale

- **Logically centralized control** plane beats full decentralization
- **Centralized configuration and management** dramatically simplifies system aspects
- **Scale out** >> Scale up

Challenges faced in building our own solution

- **Topology and deployment**
  - Introducing our network to production
  - Unmanageably high number of cables/fiber
  - Cluster-external burst b/w demand
- **Control and management**
  - Operating at huge scale
  - Routing scalability / routing with massive multipath
  - Interop with external vendor gear
- **Performance and reliability**
  - **Small on-chip buffers** → **Tune switches (eg ECN) and Hosts (DCTCP)**
  - High availability from cheap/less reliable components

Challenges faced in building our own solution

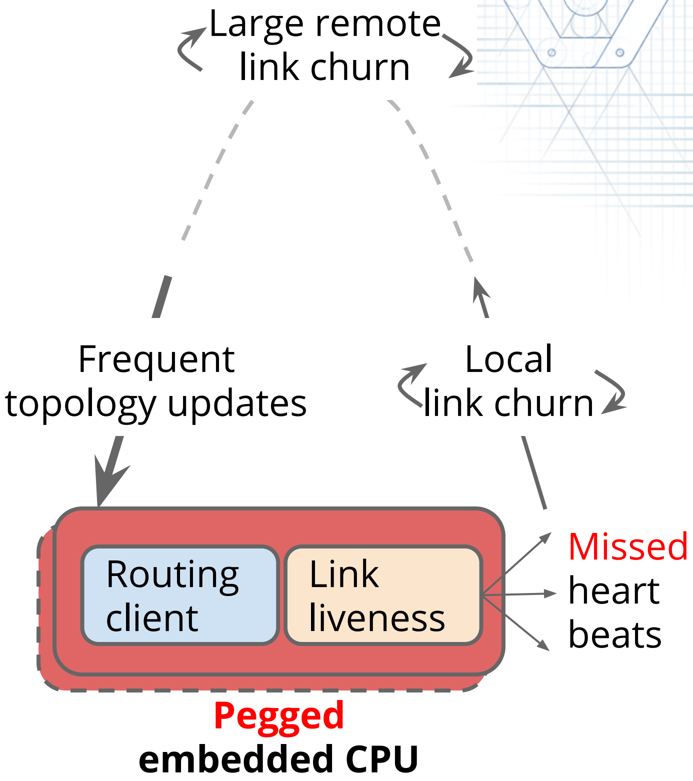
- **Topology and deployment**
  - Introducing our network to production
  - Unmanageably high number of cables/fiber
  - Cluster-external burst b/w demand
- **Control and management**
  - Operating at huge scale
  - Routing scalability / routing with massive multipath
  - Interop with external vendor gear
- **Performance and reliability**
  - Small on-chip buffers
  - **High availability from cheap/less reliable components**

↳ **Redundancy; diversity; implement only what was needed**

Experience: Outages

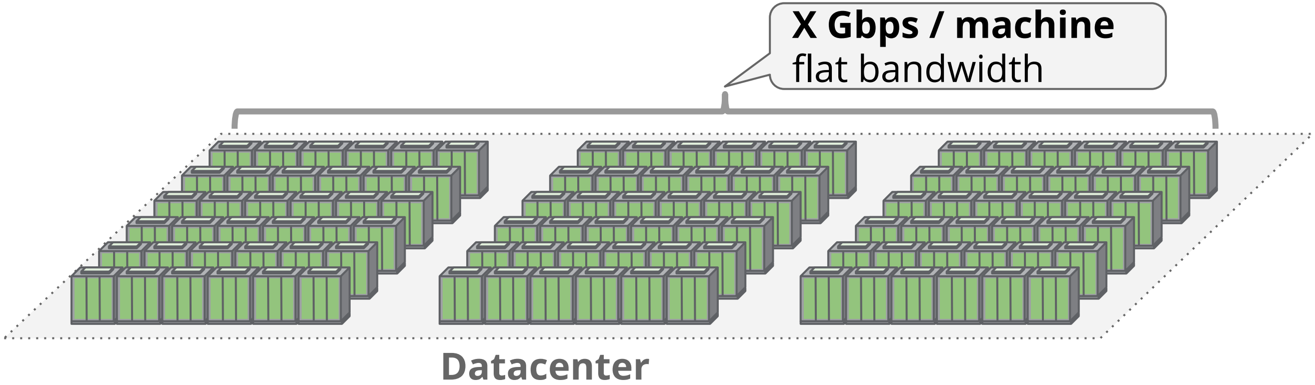
Three broad categories of outages:

- **Control software failures at scale**
  - Cluster-wide reboot did not converge
    - Liveness protocol contended for cpu with routing process
  - Cannot test at scale in a hardware lab
    - Developed virtualized testbeds
- **Aging hardware exposes corner cases**
- **Component misconfigurations**



Grand challenge for datacenter networks

- **Challenge: Flat b/w profile across all servers**
  - Simplify job scheduling (remove locality)
  - Save significant resources (better bin-packing)
  - Allow application scaling
- Scaled datacenter networks to Petabit scale in under a decade
- Bonus: reused solution in campus aggregation and WAN


X Gbps / machine
flat bandwidth
Datacenter