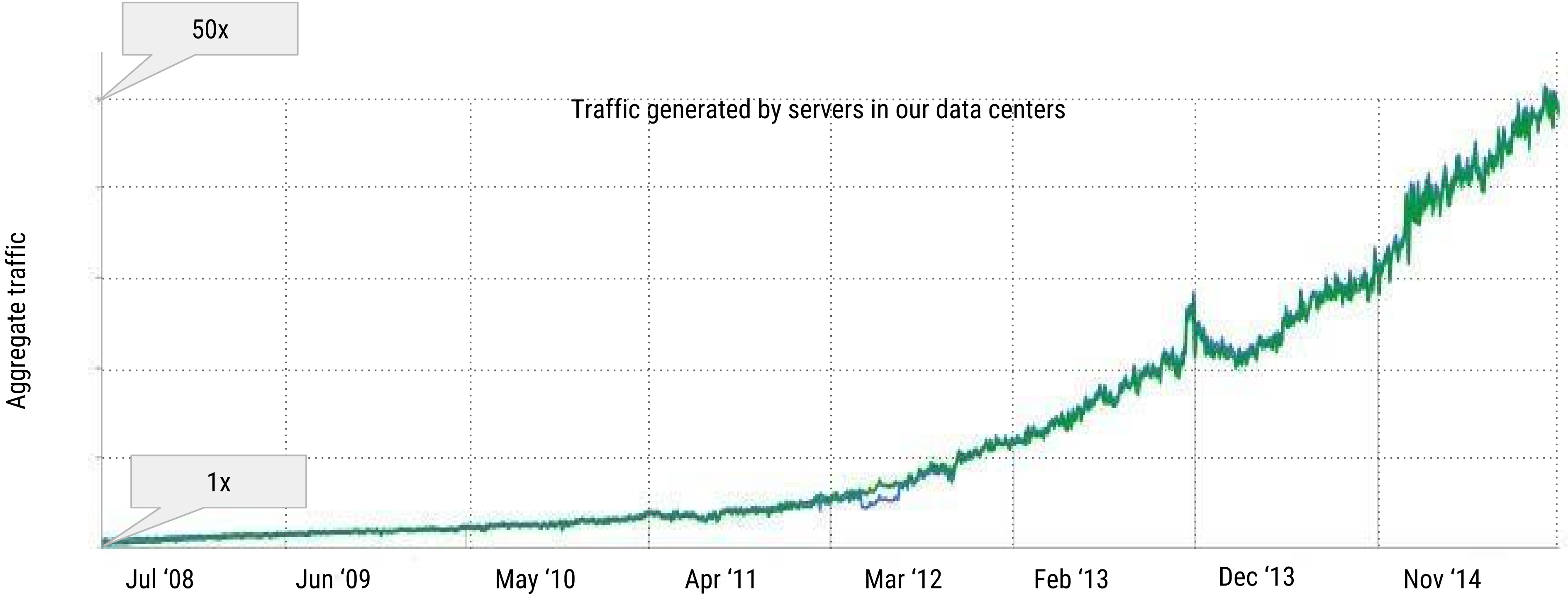
Sensors and Data Analytics in Large Data Center Networks

Hossein Lotfi
Google Networking

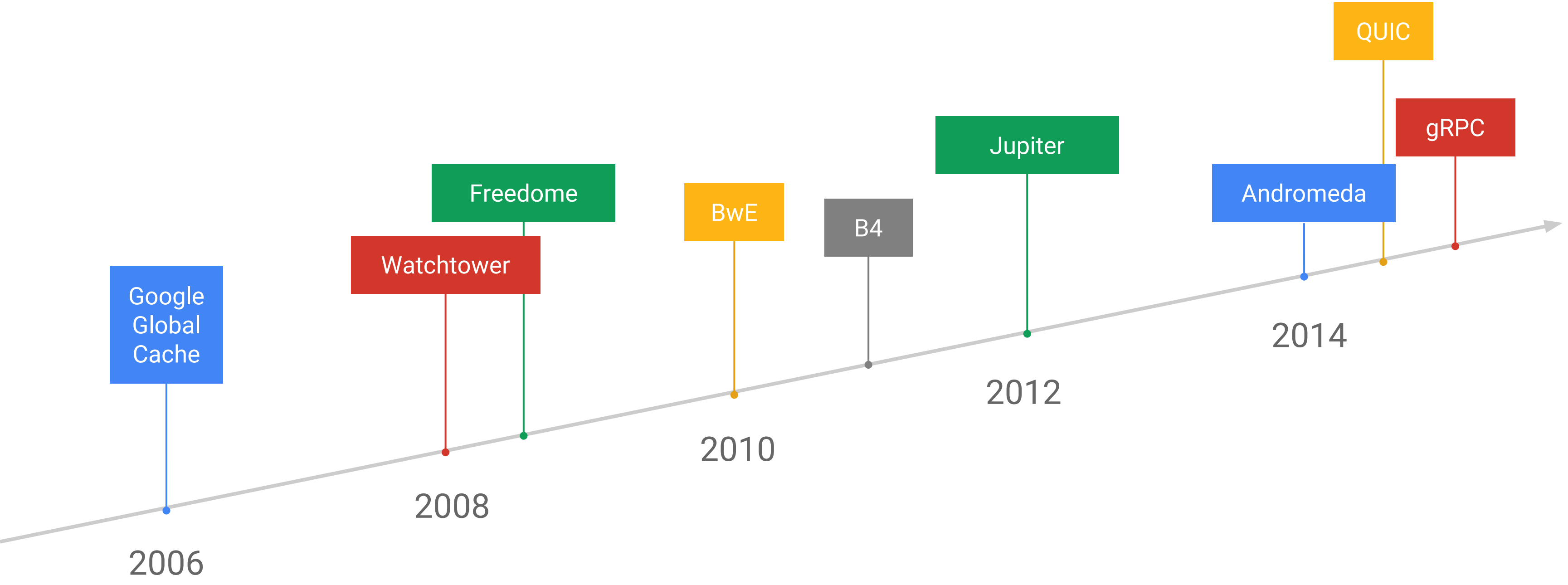
A quick overview of SDN evolution in Google data center networks

DCN Bandwidth Growth

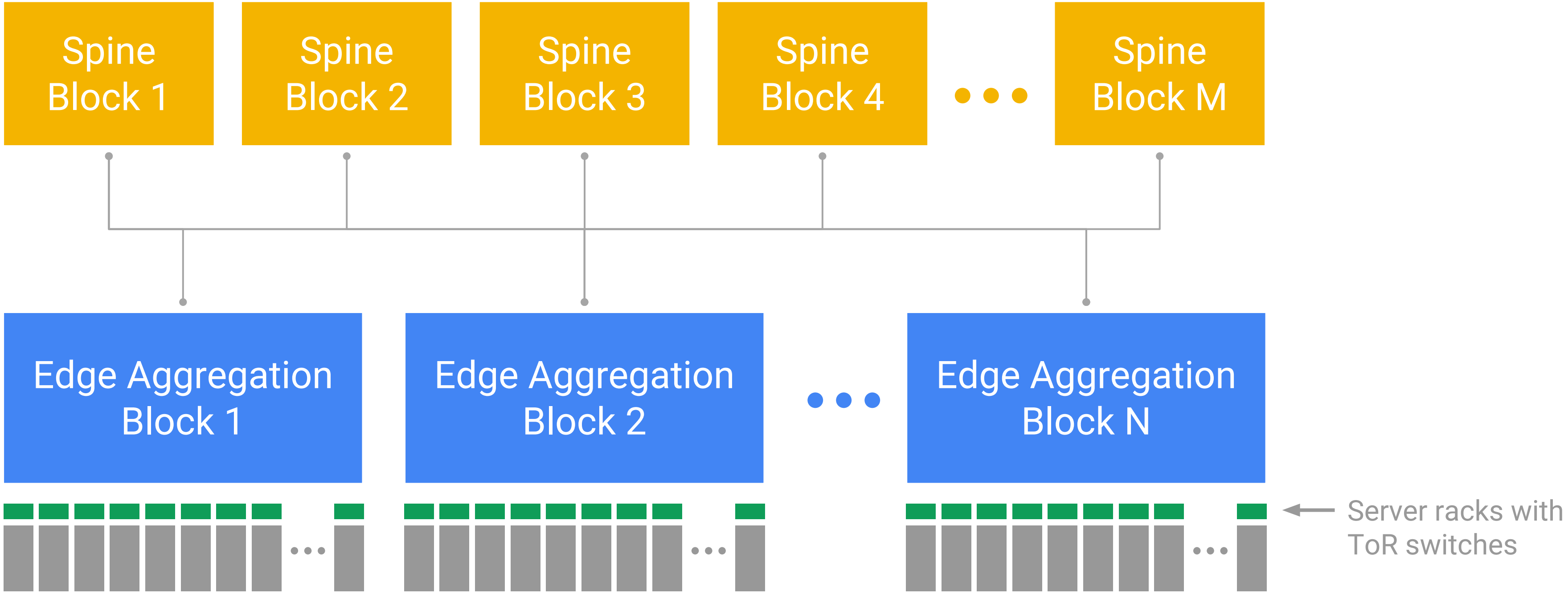


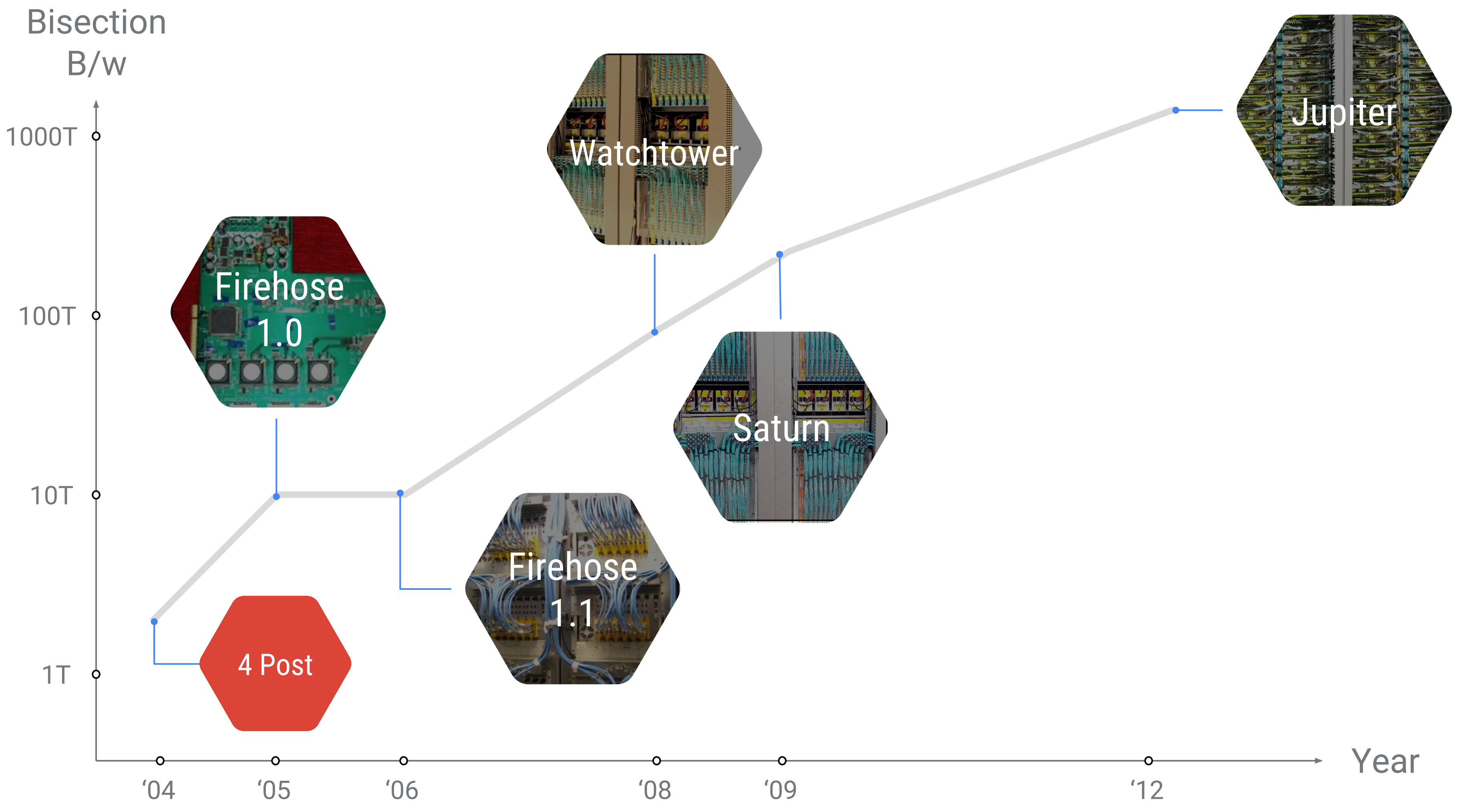
Google Networking Innovations

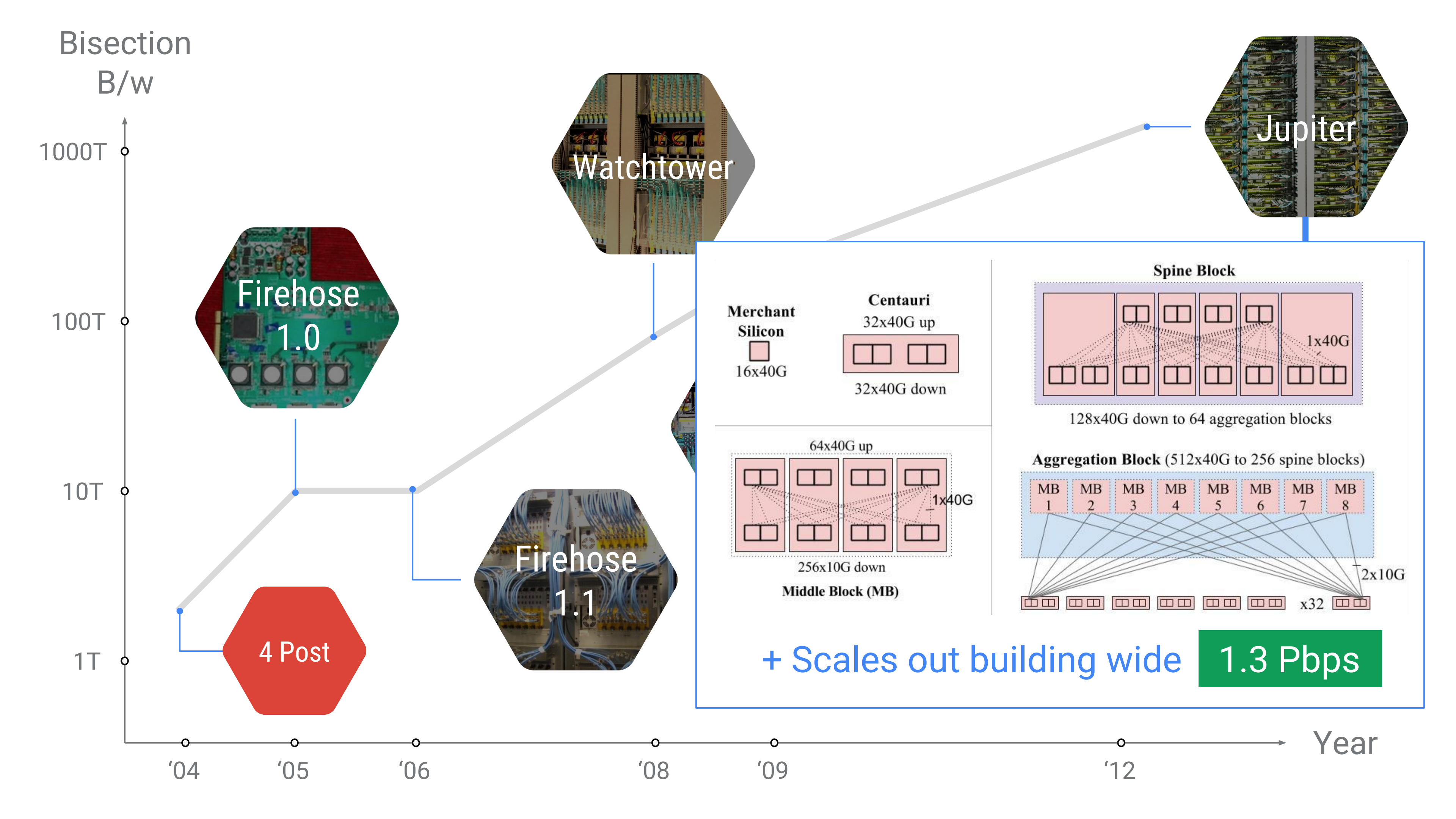
Our distributed computing infrastructure required networks that did not exist

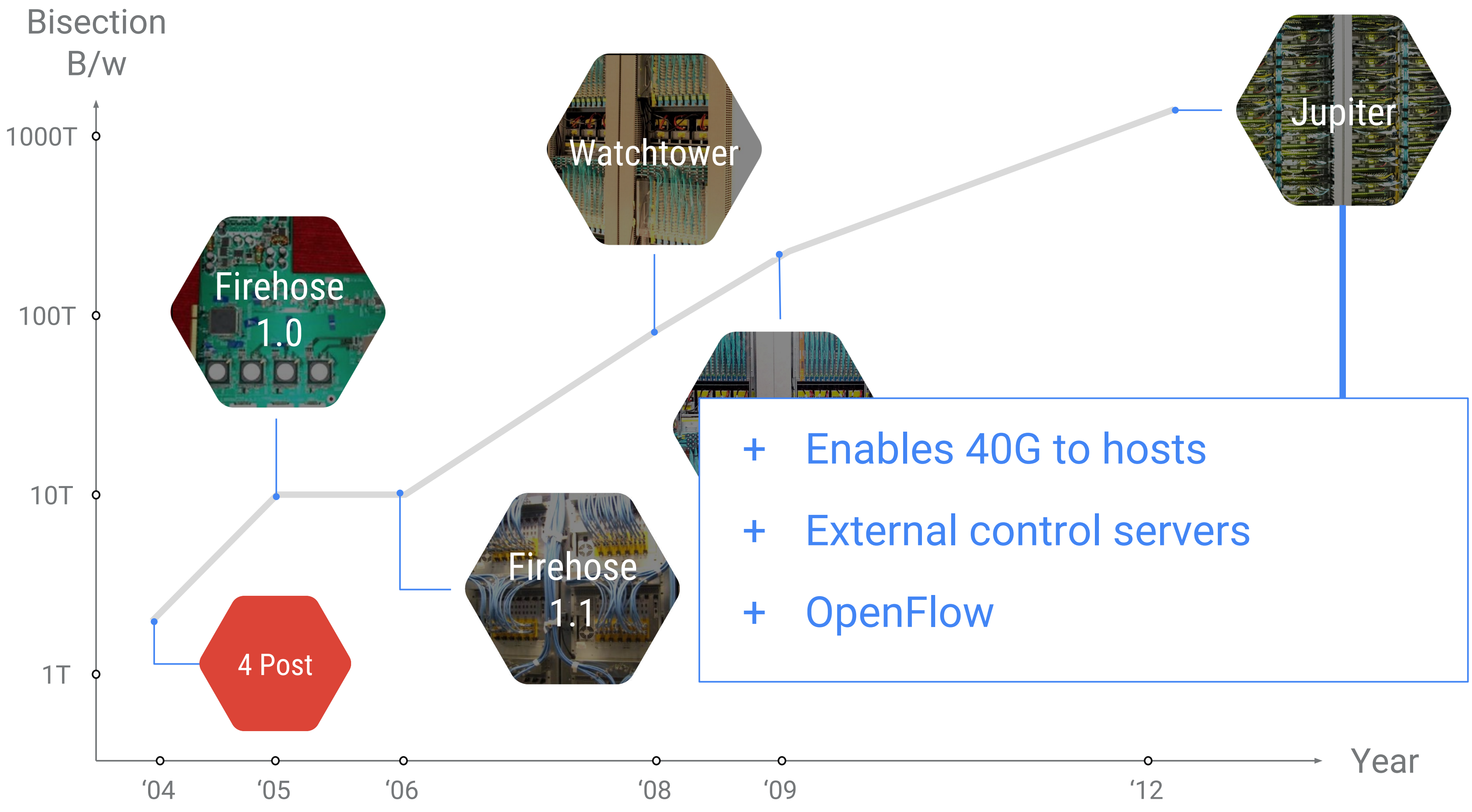


Five Generations of Networks for Google scale




Bisection B/w
1000T
100T
10T
1T
Watchtower
Jupiter
Firehose 1.0
Saturn
Firehose 1.1
4 Post
'04
'05
'06
'08
'09
'12
Year


Bisection B/w
1000T
100T
10T
1T
Watchtower
Jupiter
Firehose 1.0
Firehose 1.1
4 Post
Merchant Silicon
16x40G
Centauri
32x40G up
32x40G down
Spine Block
1x40G
128x40G down to 64 aggregation blocks
64x40G up
1x40G
256x10G down
Middle Block (MB)
Aggregation Block (512x40G to 256 spine blocks)
MB 1
MB 2
MB 3
MB 4
MB 5
MB 6
MB 7
MB 8
2x10G
x32
+ Scales out building wide
1.3 Pbps
'04
'05
'06
'08
'09
'12
Year



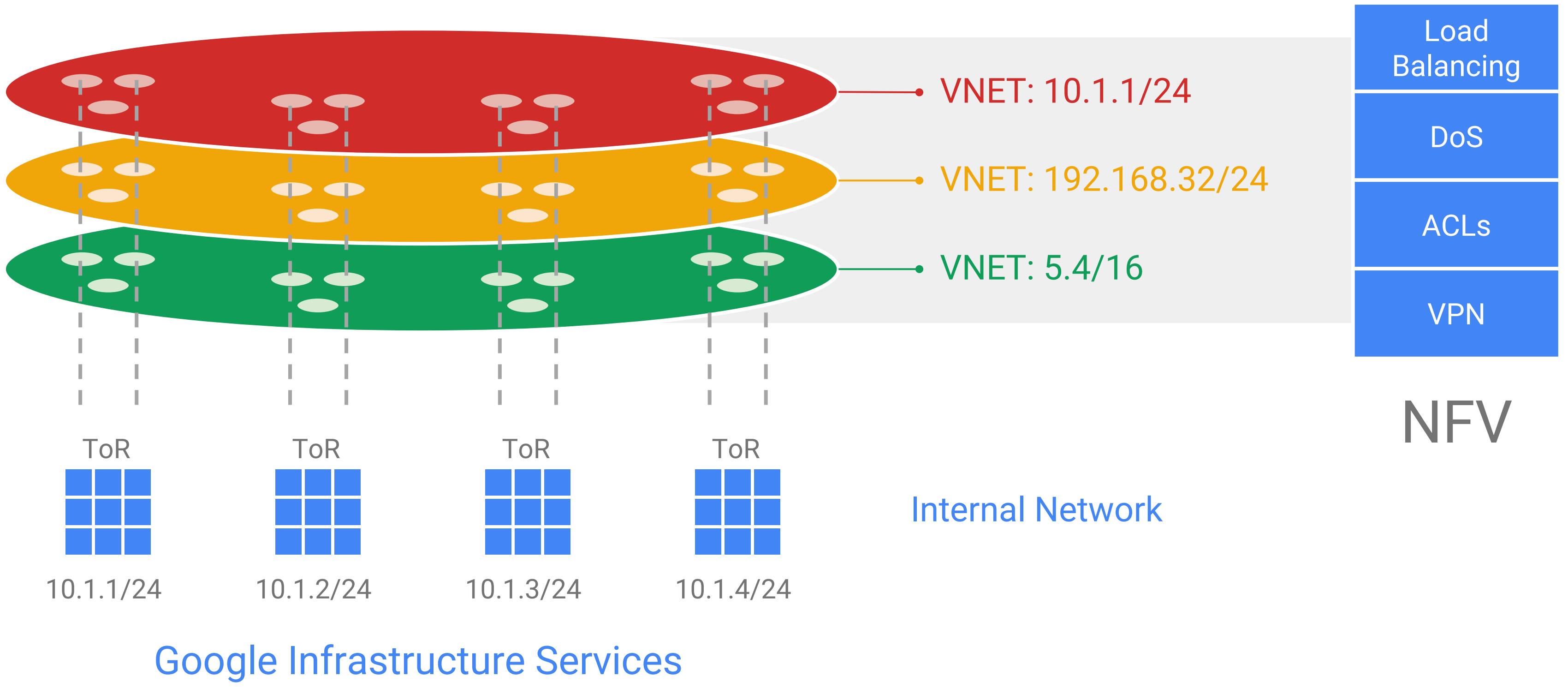
- Enables 40G to hosts
- External control servers
- OpenFlow

Characteristics of Data Center Networks

- Commodity Hardware
- Little buffering
- Tiny round trip times
- Massive multi-path
- Latency, tail latency as important as bandwidth
- Homogeneity, protocol modification much easier
- Common infrastructure across Google apps and Google Cloud Platform

B4: Google's Software Defined WAN

Andromeda Network Virtualization

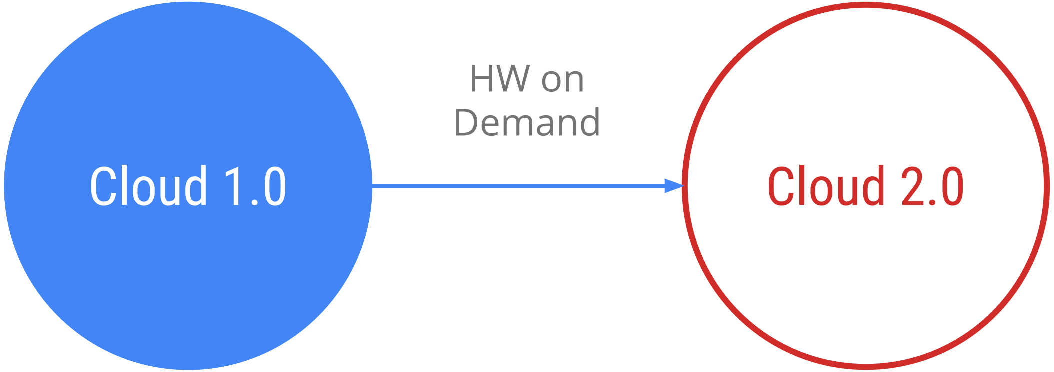


Waves of Cloud Computing

Last Decade

Virtualization delivers capex savings to enterprise DCs

Now



Public cloud frees enterprise from private HW infrastructure

Scheduling, load balancing primitives, “big data” query processing

The Third Wave of Cloud Computing

Serverless compute, actionable intelligence, and machine learning

Not data placement, load balancing, OS configuration and patching

Why Balance Matters @ Building Scale

An unbalanced data center means:

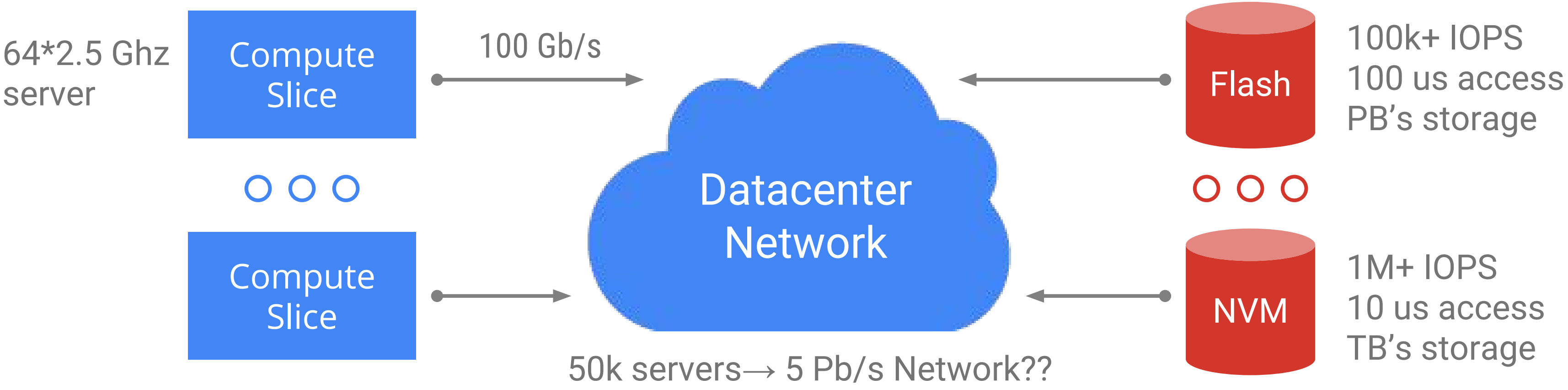
- Some resource is scarce...*limiting your value*
- Other resources are idle...*increasing your cost*

Substantial resource stranding [Eurosys 2015] if we cannot schedule at scale

Amdahl’s lesser known law:

1Mbit/sec of IO for every 1 Mhz of computation in parallel computing

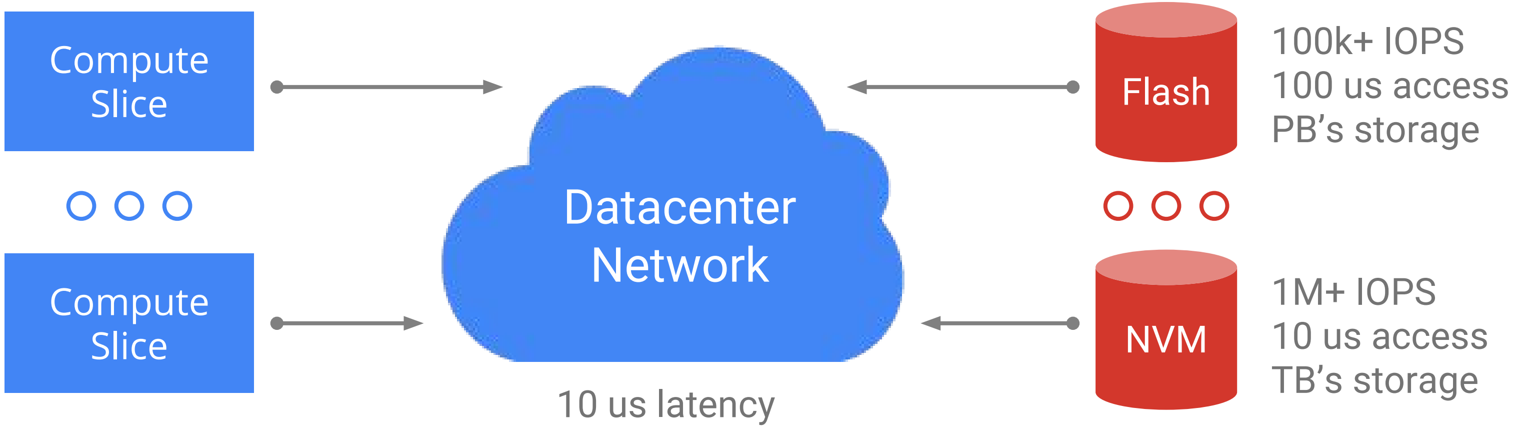
Bandwidth @ Building Scale



Based on Amdahl's observation, we might need a 5 Pb/s network

- Even with 10:1 oversub → 500Tb/s datacenter network
- Every building needs more bisection than the Internet

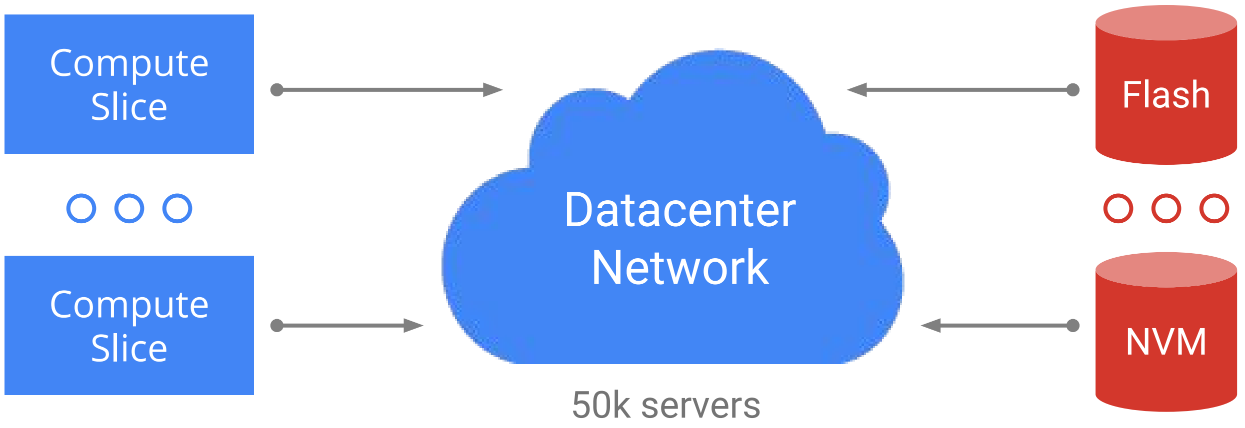
Latency @ Building Scale



To exploit future NVM, we need ~10 usec latency

- Even for Flash, we need 100 usec latency
- Or, expensive servers sit idle while they wait for IO

Availability @ Building Scale



Cannot take down a XX MW building for maintenance

- New servers always added; older ones decommissioned... with zero service impact
- Network evolves from 1G → 10G → 40G → 100G → ???

Making the Network Disappear!

Software Defined Networking enables the network to disappear, driving the next wave of computing

So, why do we need Telemetry & Analytics in DC Fabrics?

5+ # of fabric **architectures** in production

10+ kinds of **switches** as part of fabrics in production

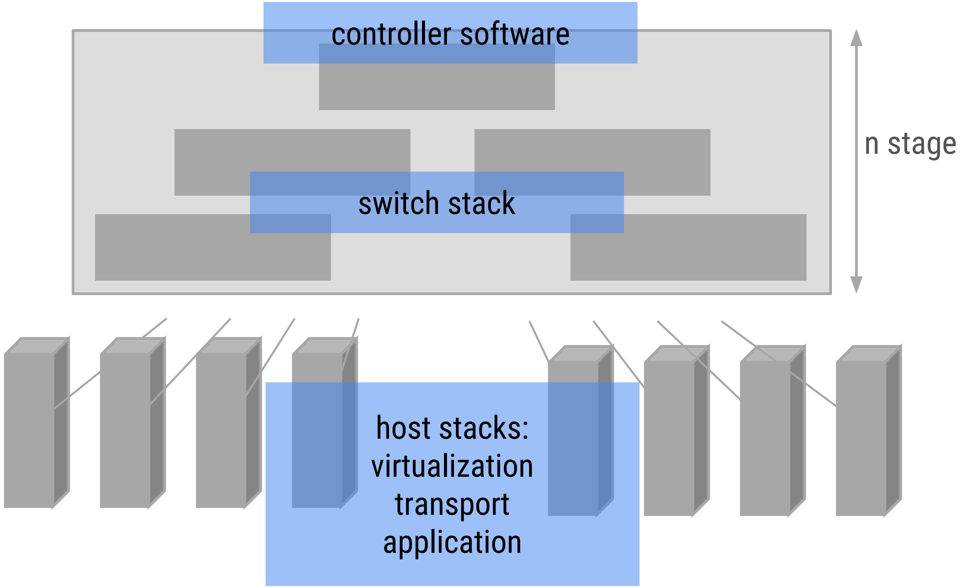
20+ # **consumers** per production fabric

Need Sensors, Software and Systems that help

- perform network design and modeling
- perform topology, configuration and routing verification
- perform smart analytics for root cause isolation

Data Center Fabrics are Complex

1. State is distributed across various elements inside and out of the fabric
2. Complex interaction between the states
3. Multiple uncoordinated writers of state under various control loops
4. Large: Impossible to humanly observe state and react to ambiguity or faults



Challenges are similar for SDN-centric or traditional protocol-based networks

Life of a typical Data Center Fabric

BUILD (init topology)

- Design the network topology
- Model the intended topology
- Populate (deploy, wire-up) DC floor

CONNECT (routing, reachability)

- Design connectivity policies
- Create the intended configuration
- Push configuration to devices

OPERATE (Apps & SLA)

- Define application SLAs
- Measure SLAs and traffic characteristics
- Feed stats to TE, enforcers, PCR schedulers

Safety, Correctness and Visibility in a DC Fabric

BUILD (init topology)

- Design & model the network topology
- Populate (deploy, wire-up) DC floor
- **Verify deployed topology against intent**

CONNECT (routing, reachability)

- Create connectivity policies & config
- Push configuration to devices
- **Verify routing consistency**

OPERATE (Apps & SLA)

- Define application SLAs
- **Measure SLAs and traffic characteristics**
- Feed stats to TE, enforcers, PCR schedulers

Compounded by **Scale**

BUILD
& topology

CONNECTIVITY
& routing

SLA
& app visibility

0.25Million+
links **per fabric****

10,000+
switches **per fabric****

10 Million +
routing rules **per fabric****

30,000 burst
updates within **1 min****

3.5 Billion
searches **per day**

300 hours
of video uploaded **every minute**

**Typical numbers seen in large data center fabrics

Systems to Enable Safety, Correctness & Visibility

BUILD
& topology

CONNECTIVITY
& routing

SLA
& app visibility

01.

Topology Verification

To continually verify that what is deployed is what was intended

02.

Route Consistency

To verify routing state consistency between controllers and data plane

03.

Traffic Characteristics **

To verify host-granular reachability and measure traffic characteristics

**The analytics system described here focuses primarily on the host-level reachability and packet-loss characterization of app2app communication

01. Topology Verification at Scale

How do we verify that what has been deployed and wired up matches intended topology

in a 10,000+ node / 250,000+ links fabric

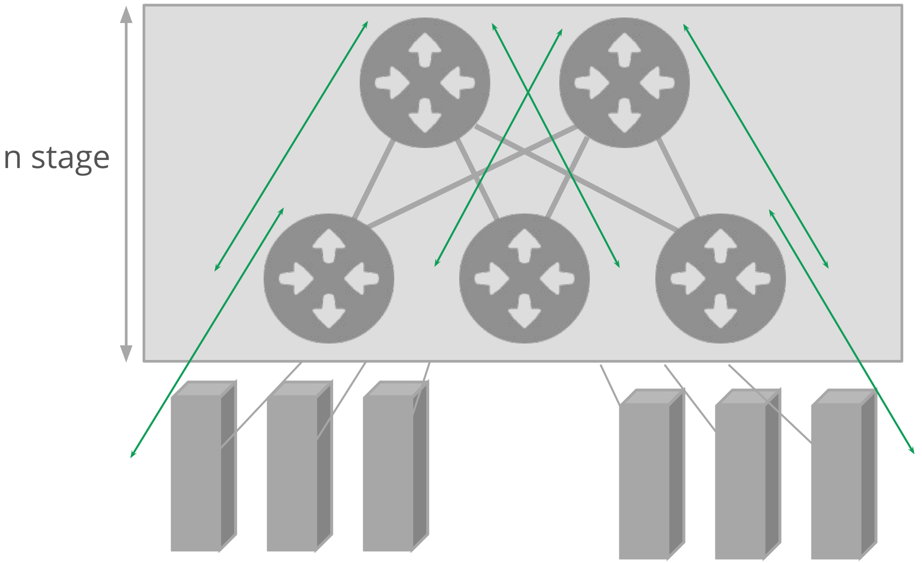
01. Topology Verification at Scale

1. Read in the **intended model** of the fabric
   Generate a topology to **verify** against

2. Generate **probe** traffic from Hosts
   This is not switched like production traffic

3. Don't rely on just destination-based routing
   Source Route: Ensures targeted **full coverage**

4. Analytics App: Takes all this data generated and **localize connectivity** problems



Simultaneous Detection of Topological faults within a minute of occurrence

02. Routing Consistency Verification at Scale

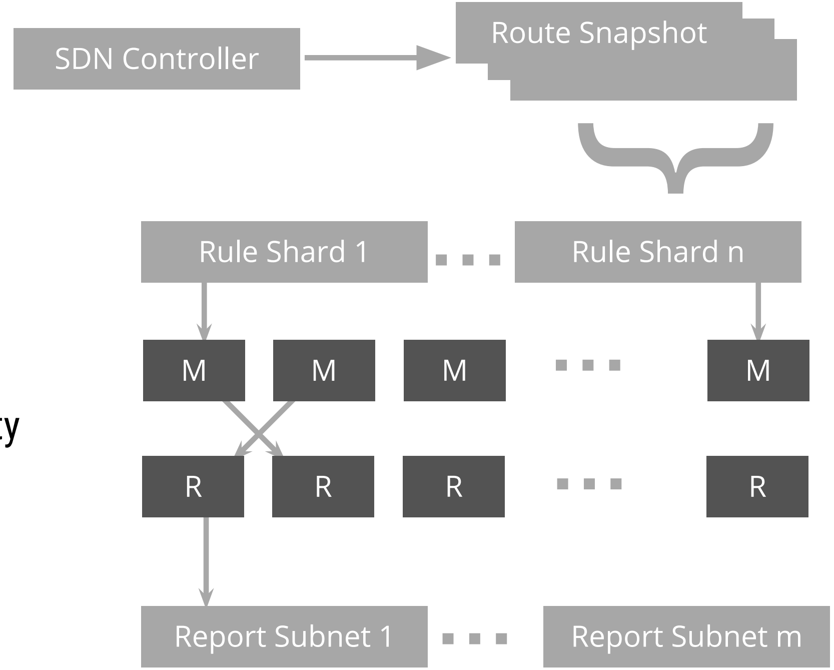
How do we verify consistency between configured policy, routing state and forwarding state

in a fabric with 10M+ rules and detect & isolate loops & black holes quickly

Google

02. Libra: Routing Consistency Verification at Scale

1. Generate **snapshot** of the routing state by recording route change events
2. **Map:** Create a network slice by destination subnet by picking rules relevant to that subnet against a shard of the full set
3. **Reduce**: Construct a directed forwarding graph and verify properties such as loop freedom and reachability
4. At every subsequent routing update, only analyze **incremental** updates



Detection of Loops & Holes within 1ms of occurrence in a 10K Node Network

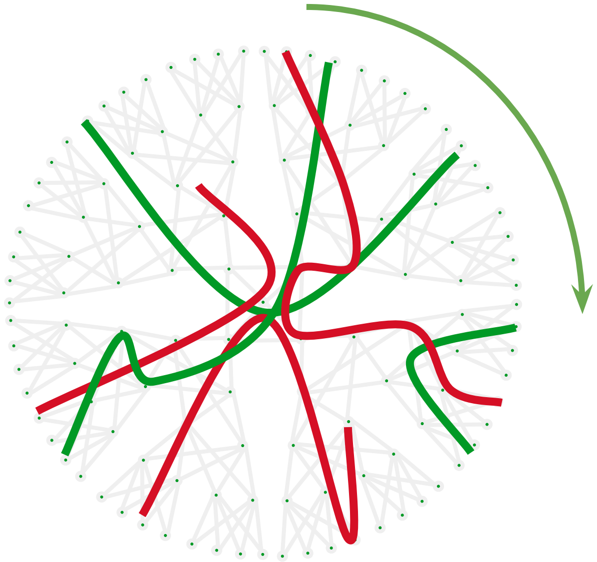
03. App-level SLA measurements at Scale

How do we measure host-level reachability and app-level traffic characteristics comprehensively

across all host-pairs and all traffic classes with varying SLA & TE needs

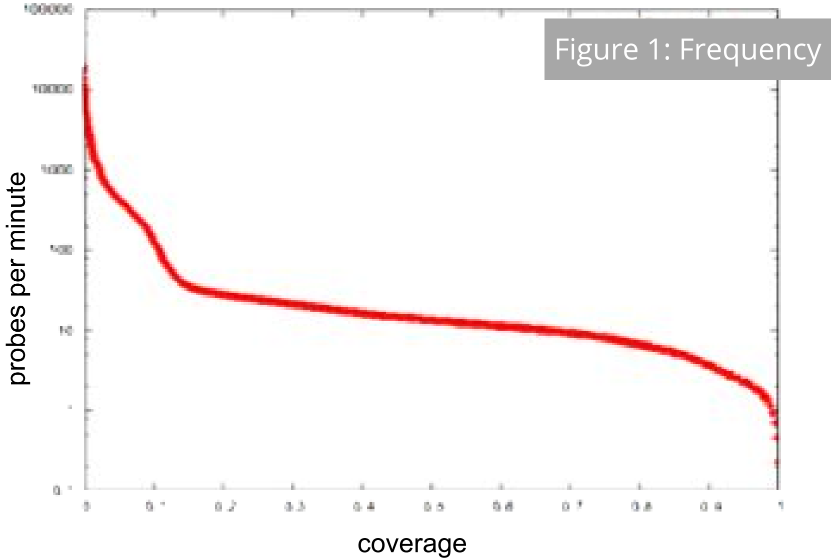
03. App-level SLA measurements at Scale

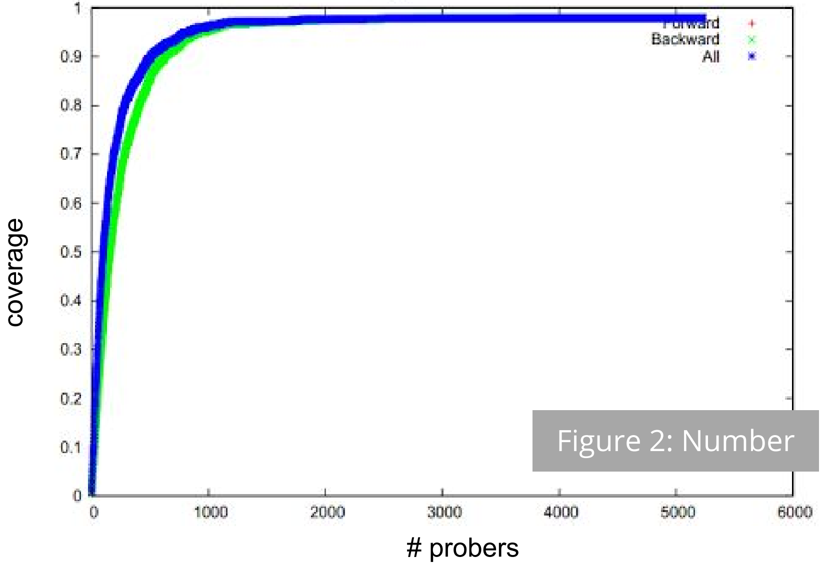
1. **Randomly** pick a subset of hosts and generate probes to and from the hosts.
2. Exercise src2dest and dest2src paths through entire **host-fabric-host SW stack**
3. Structured rotation of probes across all hosts & traffic queues for **full coverage**
4. **Correlate** probe loss & latency in fwd & rev directions across a number of probe sets to **localize** issues



Google

03. App-level SLA measurements at Scale - Results





1 — **Speed and Overhead**
Balance #probes & detection time. [O(secs)]

2 — **Scale and Efficiency**
Relatively few probes give significant coverage

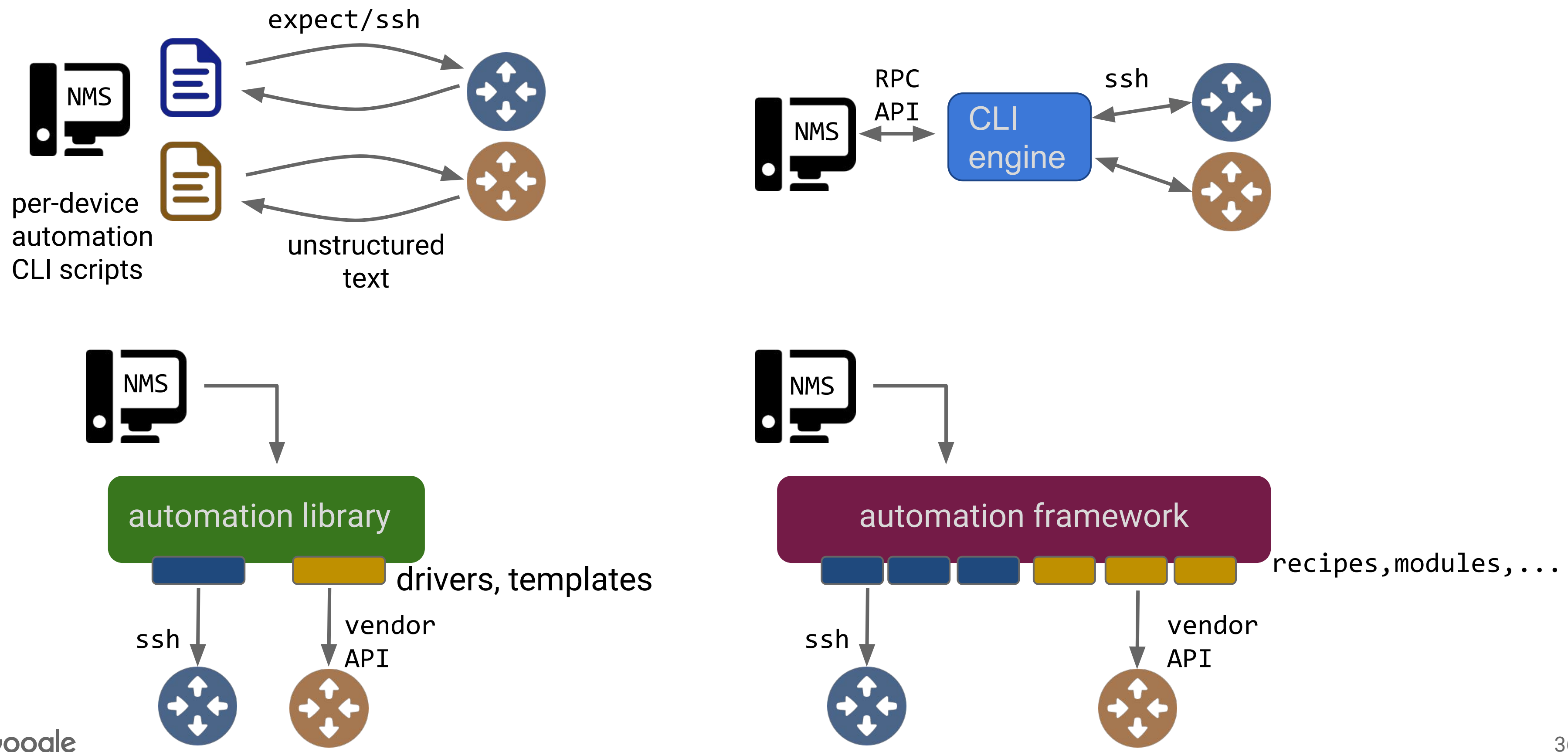
Detection of reachability problems within minutes of occurrence

Modeling and Transporting Telemetry Data

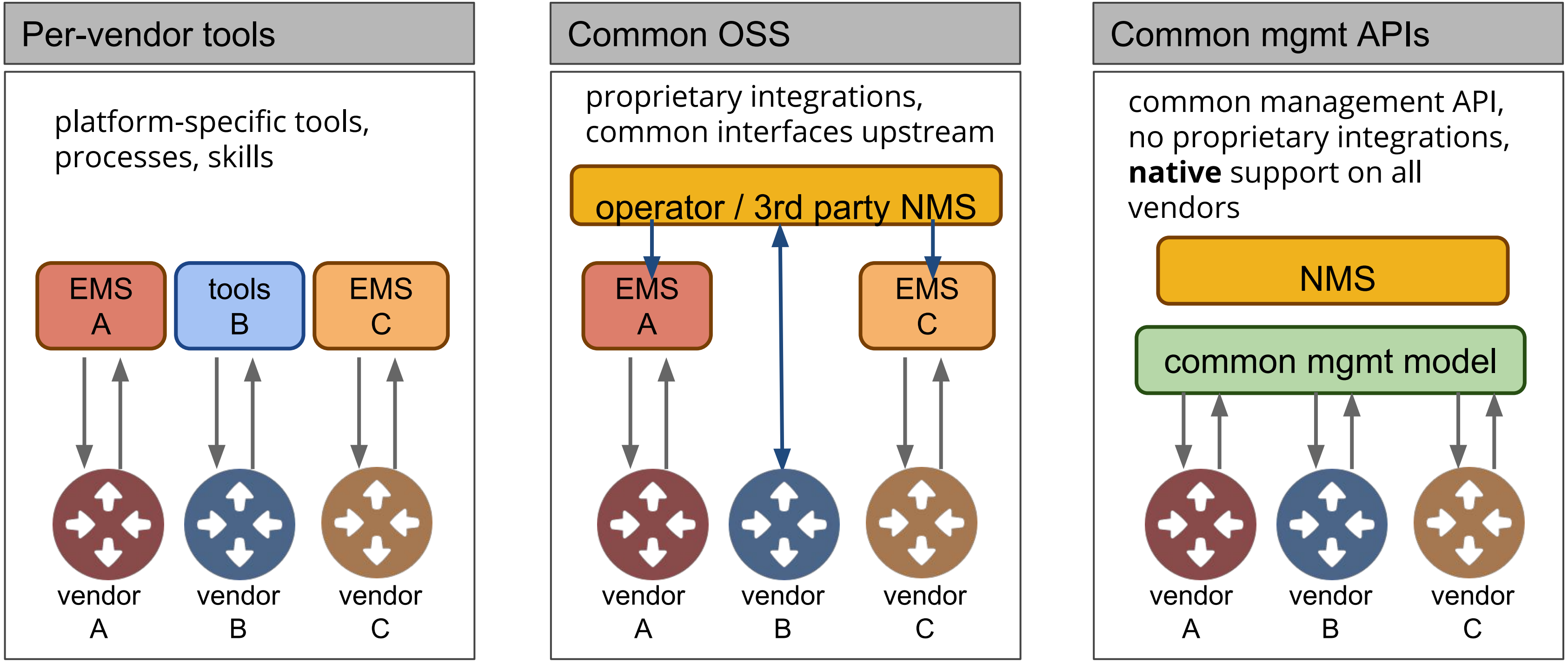
Network equipment

- **We've discussed a lot of external software telemetry sources...what about data from network devices themselves?**
- **Today, SNMP is often the de facto network telemetry protocol. Time to upgrade!**
    - **legacy implementations** -- designed for limited processing and bandwidth
    - **expensive discoverability** -- re-walk MIBs to discover new elements
    - **no capability advertisement** -- test OIDs to determine support
    - **rigid structure** -- limited extensibility to add new data
    - **proprietary data** -- require vendor-specific mappings and multiple requests to reassemble data
    - **protocol stagnation** -- no absorption of current data modeling and transmission techniques

Network automation has come a long way ...



Toward a vendor-neutral, model-driven world

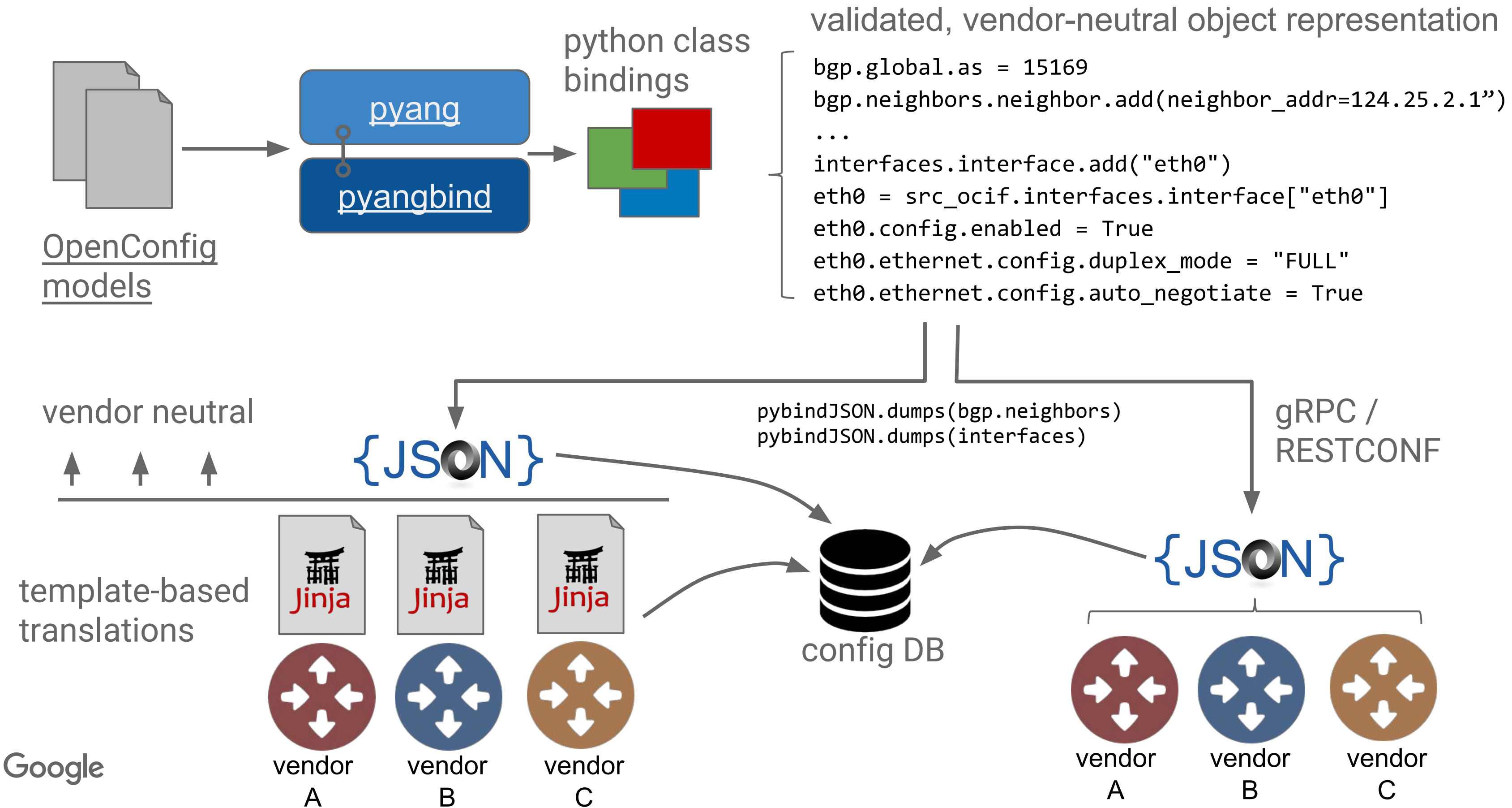


OpenConfig : user-defined models

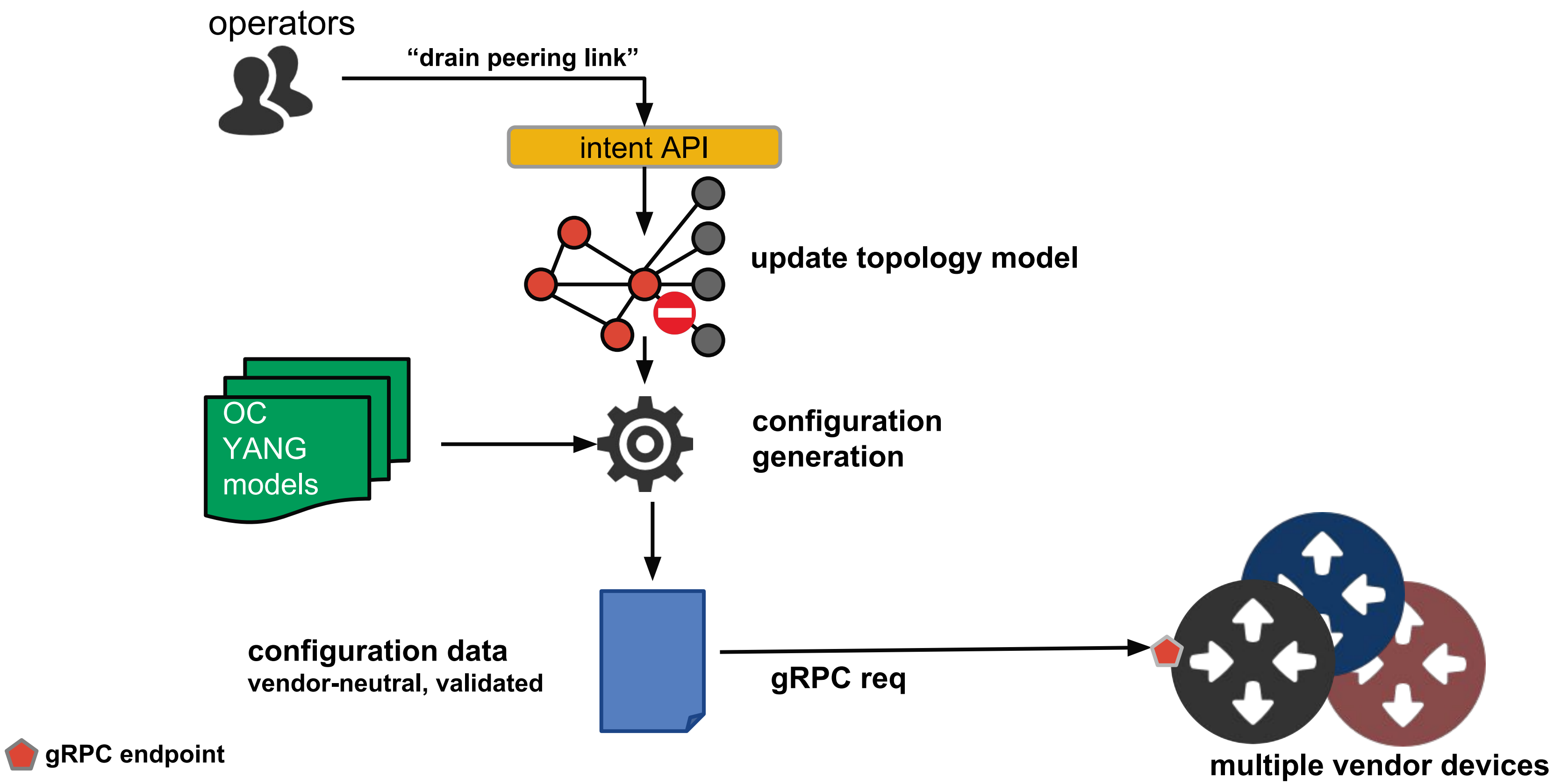
- informal industry collaboration among network operators
- data models for configuration and operational state, code written in YANG
- organizational model: informal, structured like an open source project
- development priorities driven by operator requirements
- engagements with major equipment vendors to drive native implementations
- engagement with standards (IETF) and OSS (ODL, ONOS, goBGP, Quagga)

TeraStream

OSS stack for model-based programmatic configuration



Example Configuration pipeline

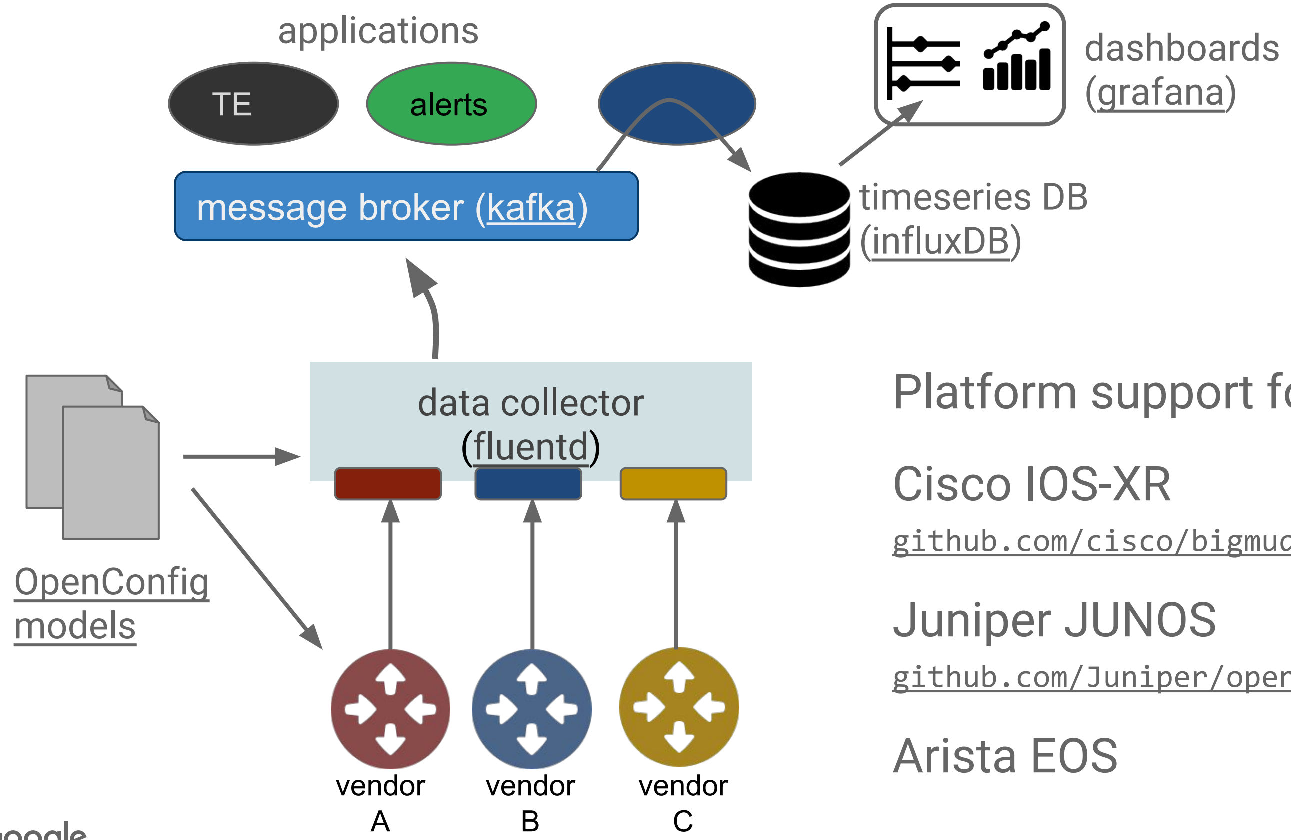


Next generation network telemetry: OPENCONFIG

www.openconfig.net

- network elements stream data to collectors (push model)
- data populated based on vendor-neutral models whenever possible
- utilize a publish/subscribe API to select desired data
- scale for next 10 years of density growth with high data freshness
  - other protocols distribute load to hardware, so should telemetry
- utilize modern transport mechanisms with active development communities
  - e.g., gRPC (HTTP/2), Thrift, protobuf over UDP

OSS stack for model-based streaming telemetry



Platform support for streaming telemetry:

Cisco IOS-XR

github.com/cisco/bigmuddy-network-telemetry-stacks

Juniper JUNOS

github.com/Juniper/open-nti

Arista EOS

That IS a lot of data...

Now that your network infrastructure is richly instrumented...how do you extract this information?

We use a RPC framework optimized for encrypted, streaming, and multiplexed connections.

...and so can you.

GRPC

http://grpc.io

Key Take-Aways

- Think outside the **BOX**
- Sensors and **DATA ANALYTICS** are key for building data center networks
- **HUMANS** are (almost) useless,
- **OpenConfig,** vendor-neutral model-driven config and telemetry
- **gRPC,** a transport mechanism for telemetry data

There are only two ways you can see Jupiter

[Google Data Center 360° Tour](https://youtu.be/zDAYZU4A3w0)

https://youtu.be/zDAYZU4A3w0

References

1. *"Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network,"* **SIGCOMM 2015**.
2. "Network Detective: Finding network blackholes", **Israel Networking Day 2014.**
3. "Libra: Divide and Conquer to Verify Forwarding Tables in Huge Networks", **NSDI 2014.**
4. *"B4: Experience With a Globally-Deployed Software Defined WAN,"* **SIGCOMM 2013**.
5. *"Bandwidth Enforcer: Flexible, Hierarchical Bandwidth Allocation for WAN Distributed Computing,"* **SIGCOMM 2015**.

THANK YOU

Hossein Lotfi

HosseinL@google.com