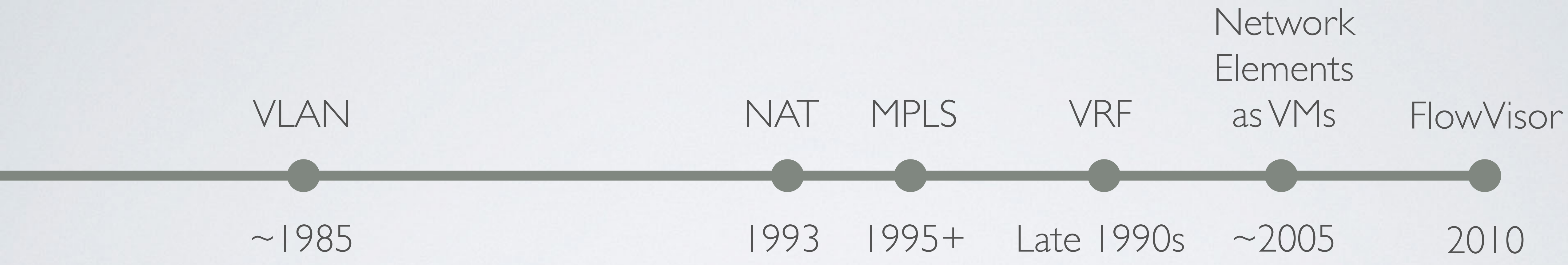
NETWORK VIRTUALIZATION IN MULTI-TENANT DATACENTERS

Teemu Koponen

with

Keith Amidon, Peter Balland, Martín Casado, Anupam Chanda, Bryan Fulton, Igor Ganichev, Jesse Gross, Natasha Gude, Paul Ingram, Ethan Jackson, Andrew Lambeth, Romain Lenglet, Shih-Hao Li, Amar Padmanabhan, Justin Pettit, Ben Pfaff, Rajiv Ramanathan, Scott Shenker, Alan Shieh, Jeremy Stribling, Pankaj Thakkar, Dan Wendlandt, Alexander Yip, and Ronghua Zhang

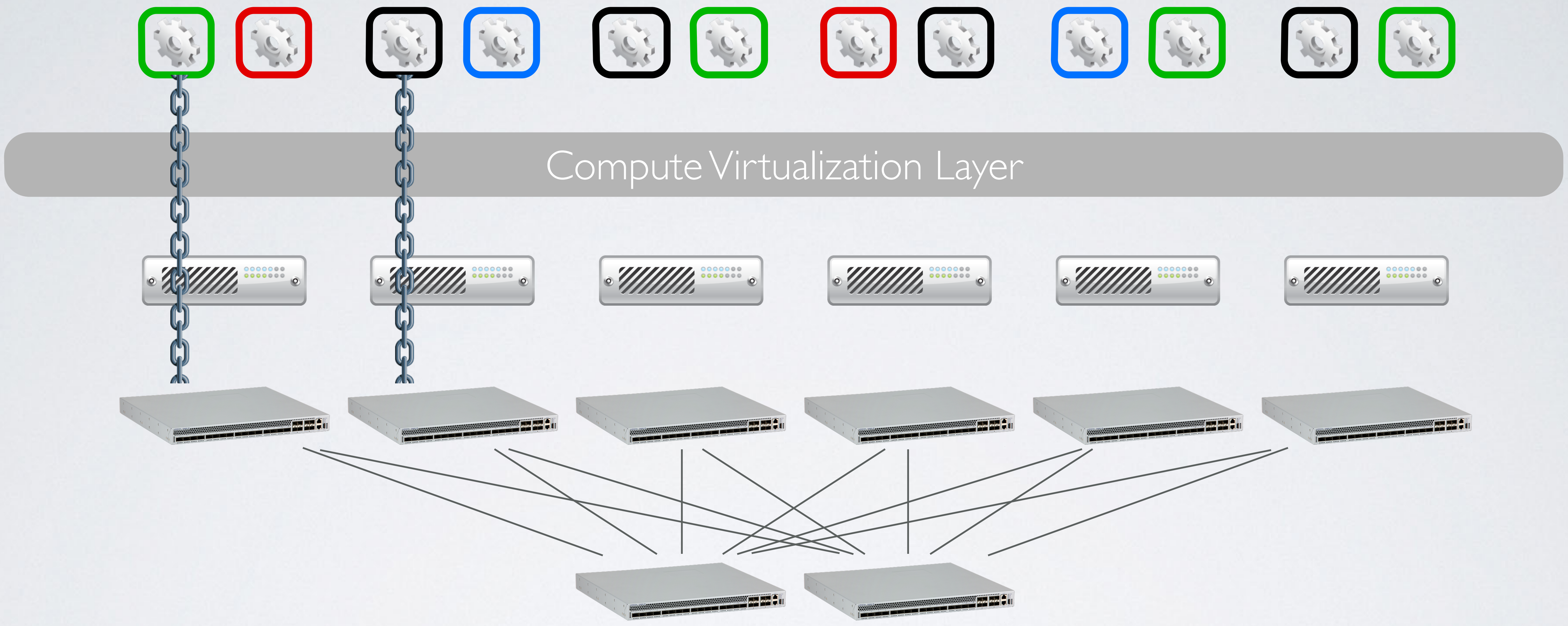
NETWORK VIRTUALIZATION?



VLAN	NAT	MPLS	VRF	Elements as VMs	FlowVisor
Subnet	IP address space	Path	L3 FIB	Elements	ASIC

Plenty of primitives but **no** network virtualization per se.

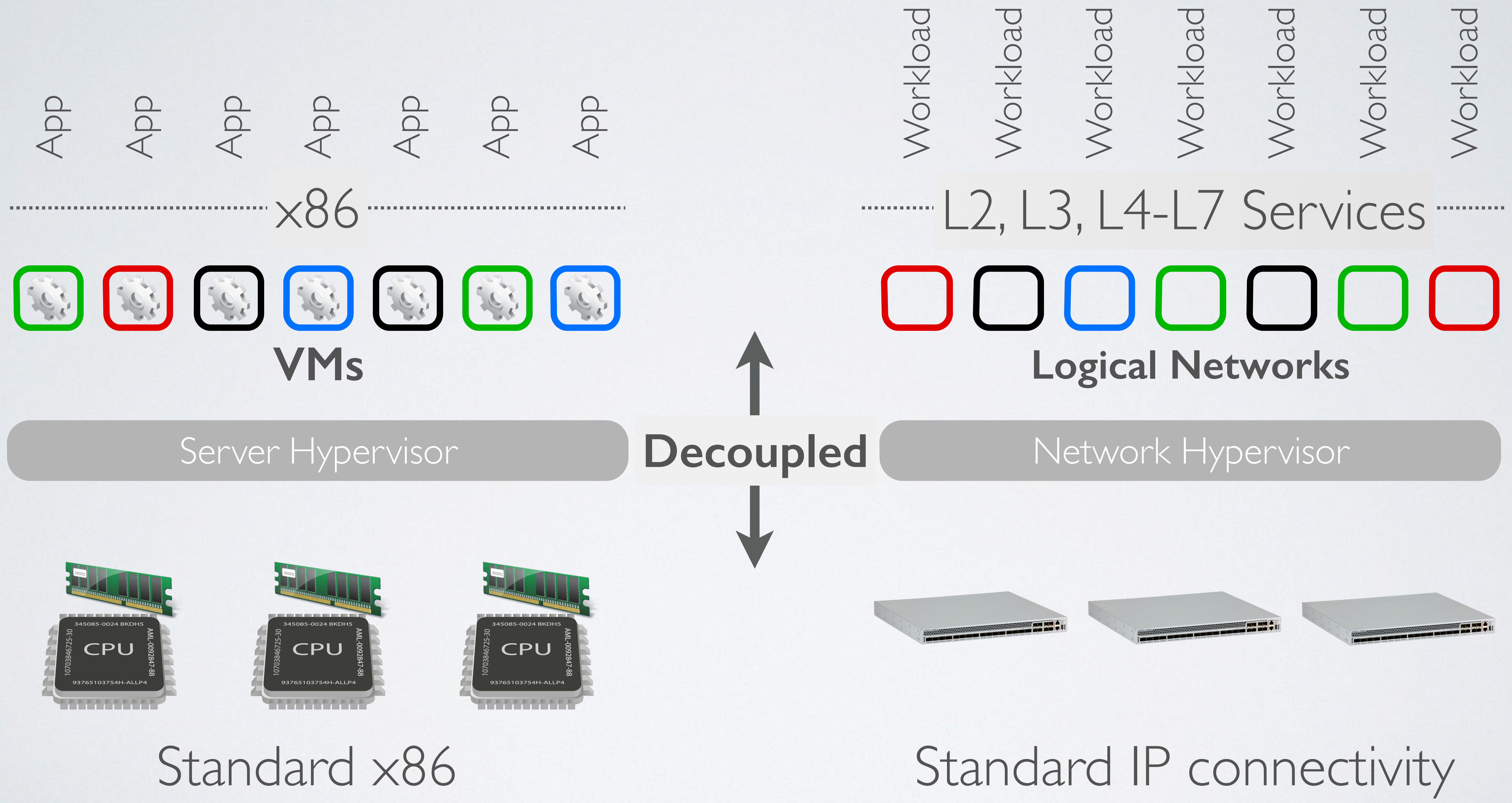
MULTI-TENANT DATACENTERS



Result with the aforementioned primitives:

- Slow provisioning
- Limited VM placement
- Mobility is limited
- Hardware dependent
- Operationally intensive
- …

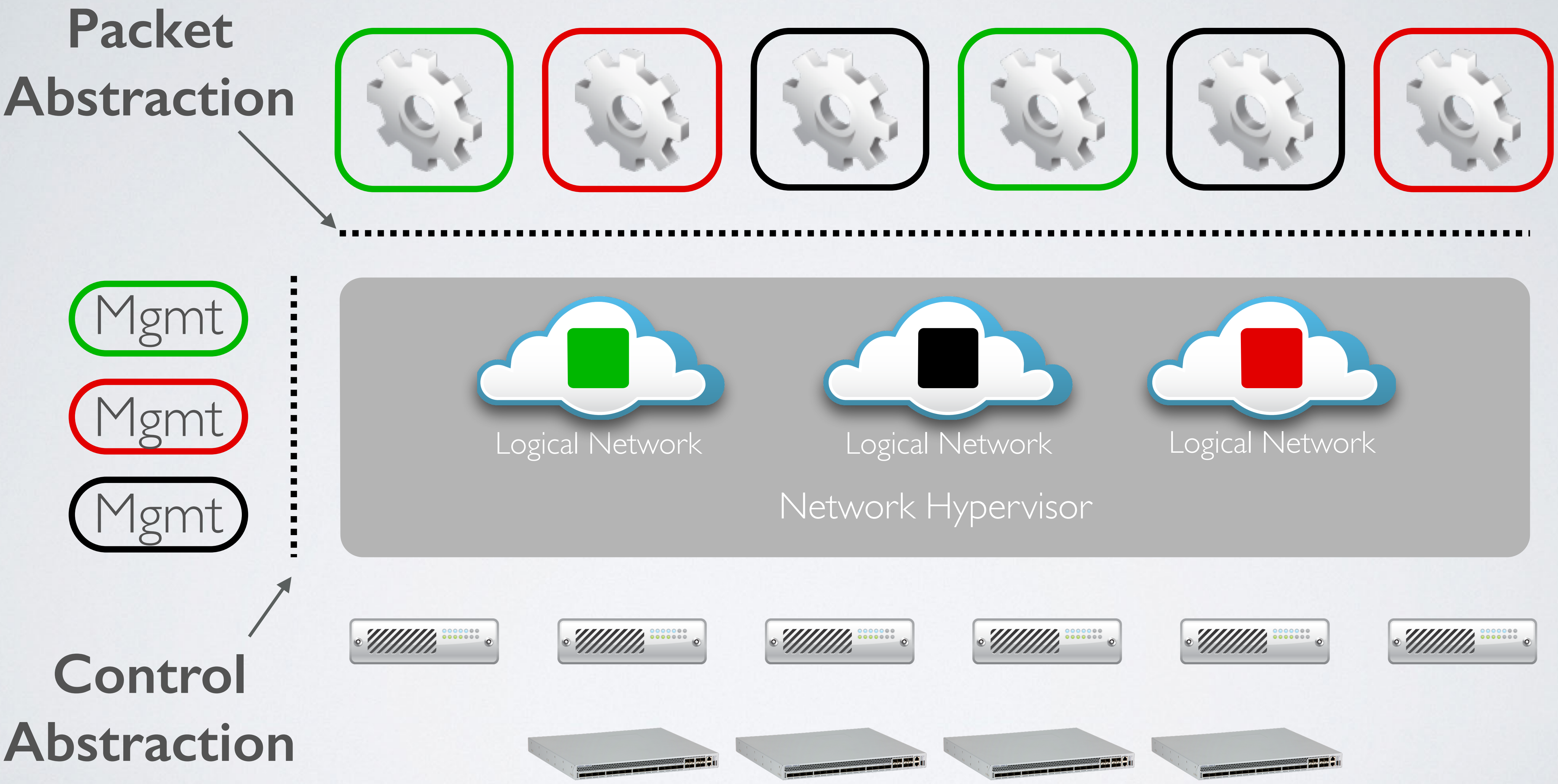
NETWORK HYPERVISOR



AGENDA

- Overall design of NVP network hypervisor.
- Design challenges.
- Hard lessons learnt.
- What's next in network virtualization?

WHAT IS A NETWORK HYPERVISOR?



Packet Abstraction + Control Abstraction = Network Hypervisor

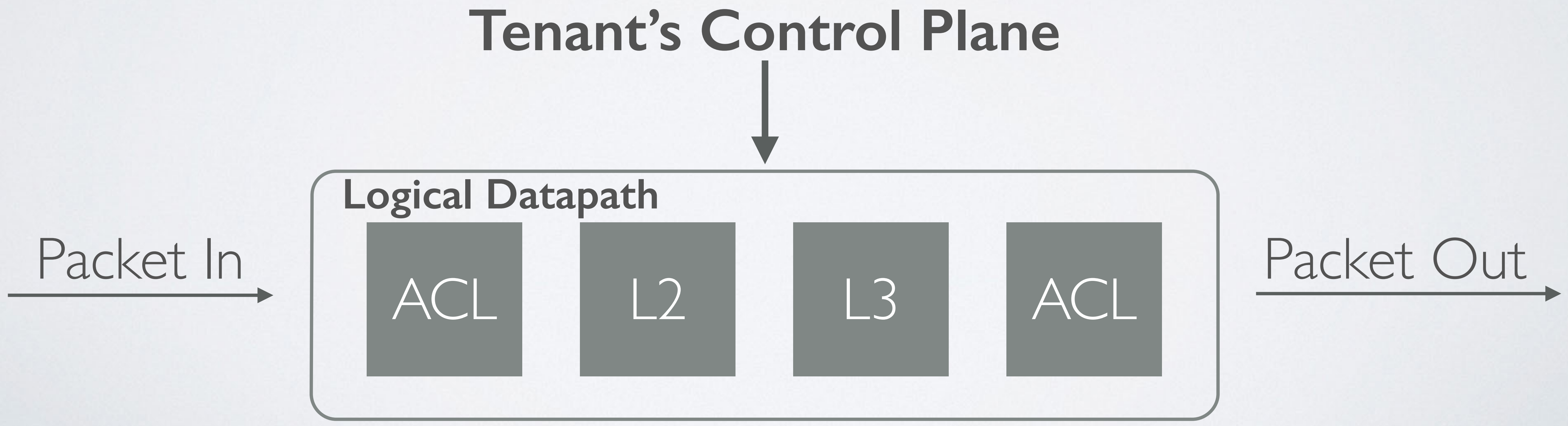
WHAT ARE THE ABSTRACTIONS?

Packet abstraction

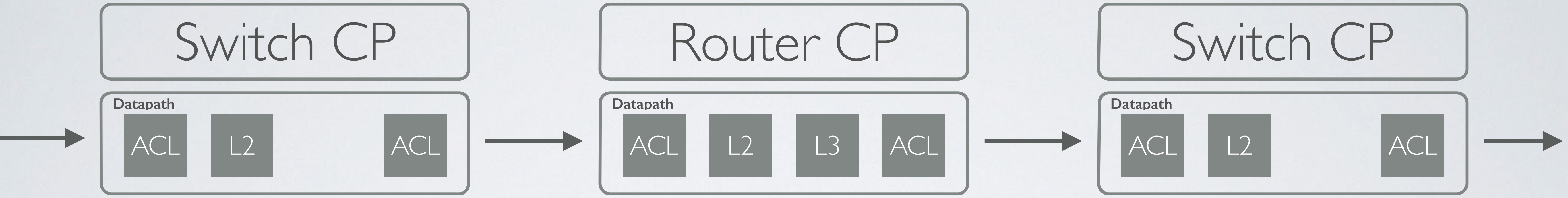
- Compliance with standard TCP/IP stack is a necessity:
    - L2, L3 semantics (unicast, ARP, …)

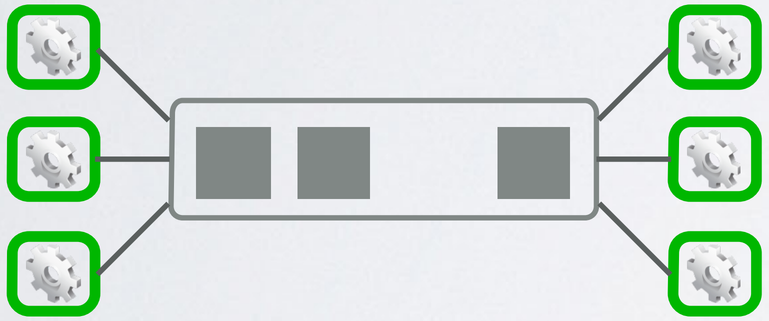
Control abstraction

- Networking has no single high level control interface.
- There's a low-level one though!

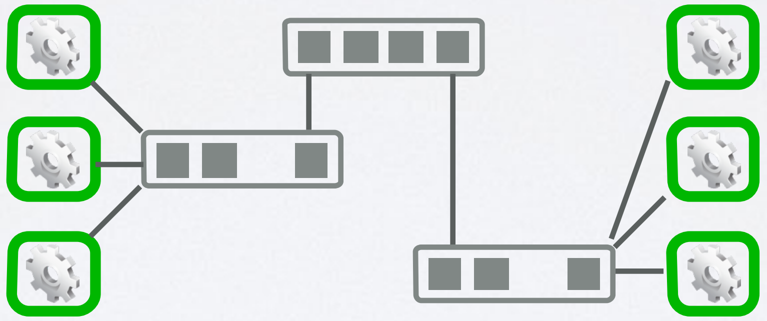


GENERALITY OF DATAPATH

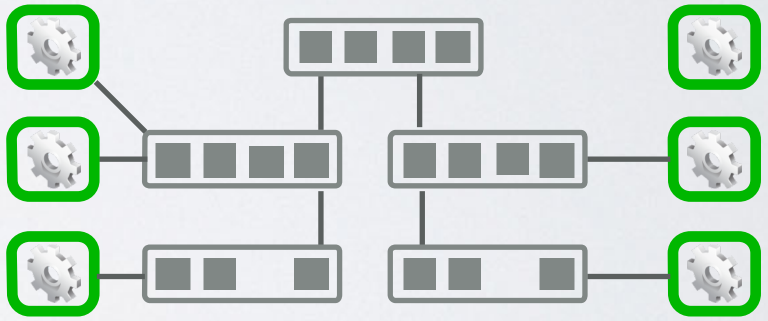




One logical switch



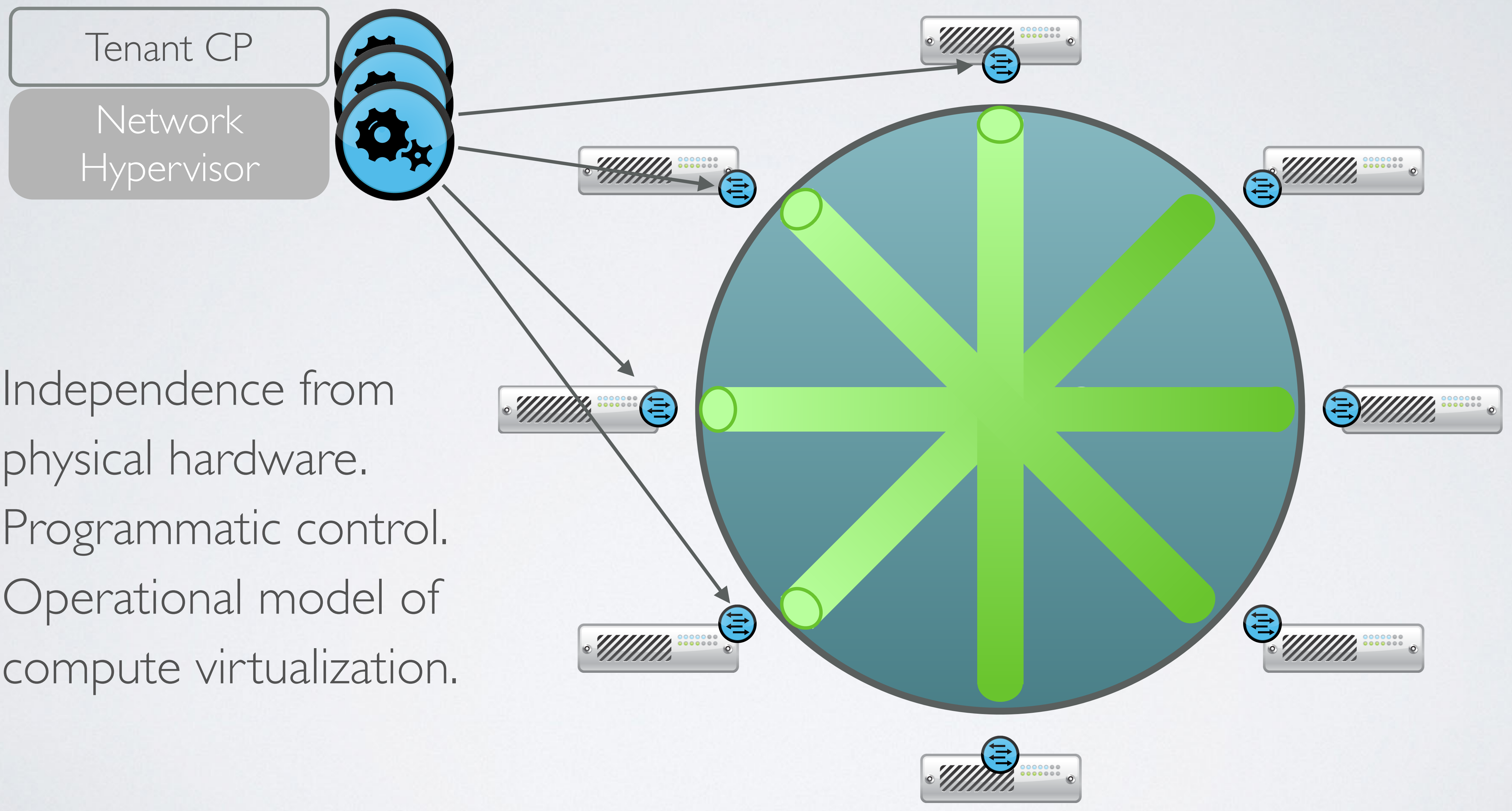
2-tier logical topology



Arbitrary logical topology

Faithful reproduction of physical network service model.

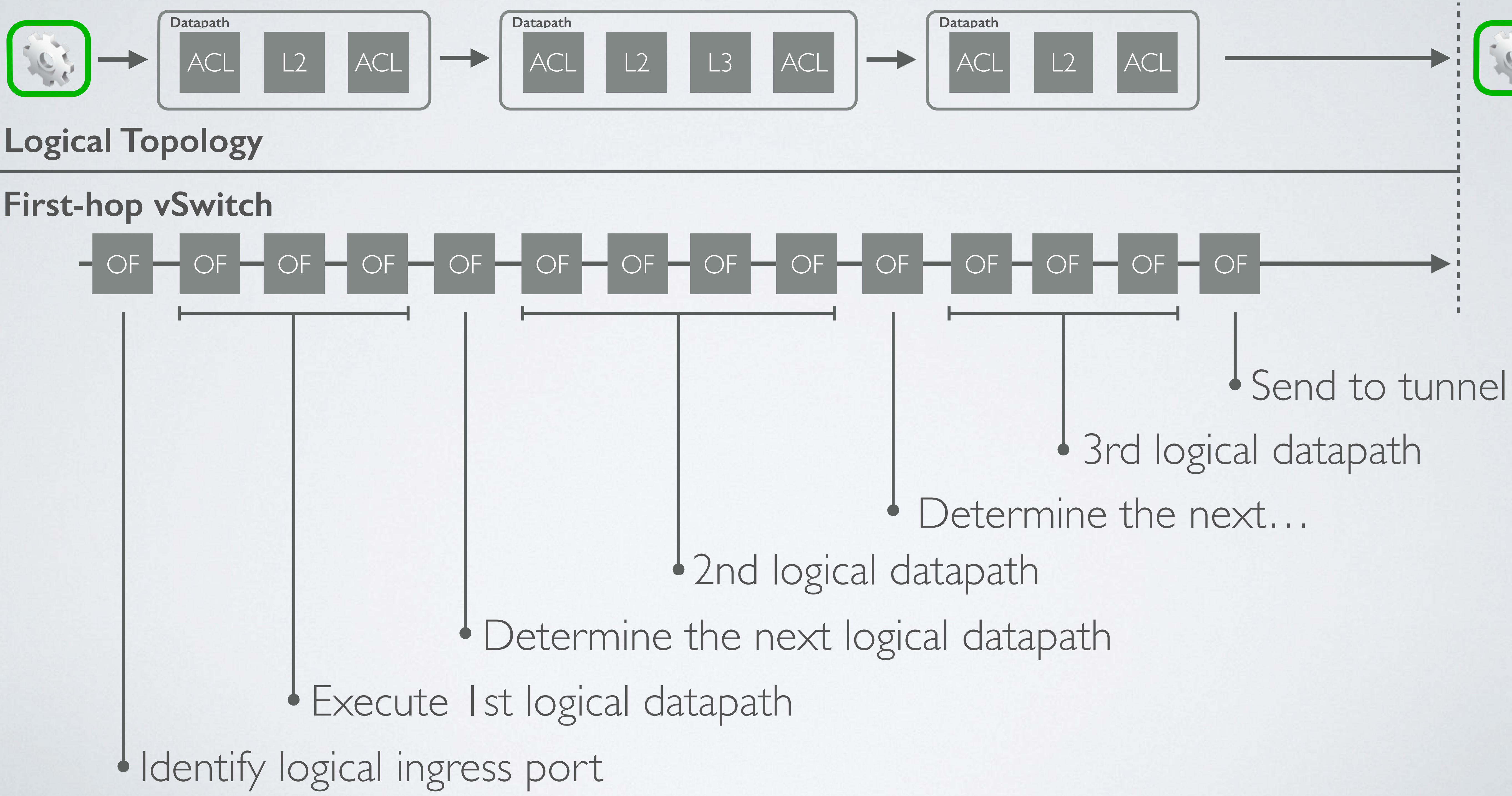
WHERE TO IMPLEMENT?



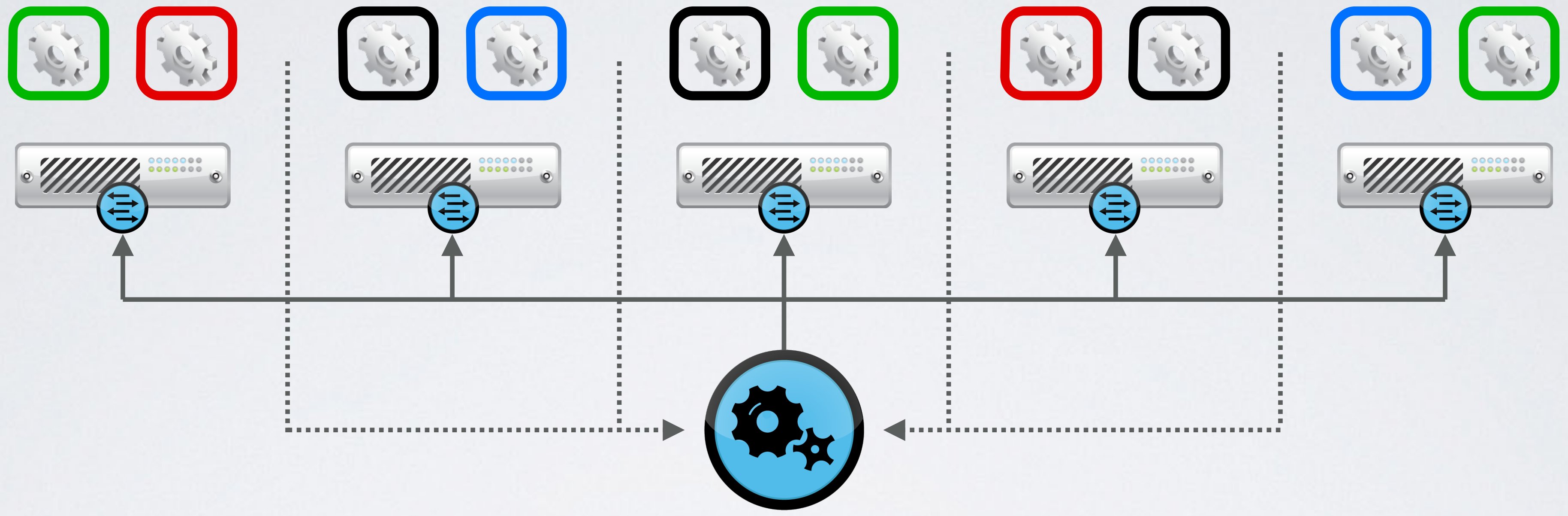
- Independence from physical hardware.
- Programmatic control.
- Operational model of compute virtualization.

No extra x86 hops: just the source and destination hypervisor!

INSIDE THE VIRTUAL SWITCH



COMPUTATIONAL CHALLENGE



1. Controllers learn the location of VMs.

2. Controllers proactively compute & push all forwarding state required to connect each VM.

$\longrightarrow$ Forwarding State = **F**(configuration, VM locations)

Repeat above as logical configuration or physical configuration (VM placement) changes.

Challenge: How to compute $O(N^2)$ volume of low-level OpenFlow and OVSDB state, when inputs change all the time.

STATE COMPUTATION

Forwarding State = F(configuration, VM locations)

1. How to Scale Computation

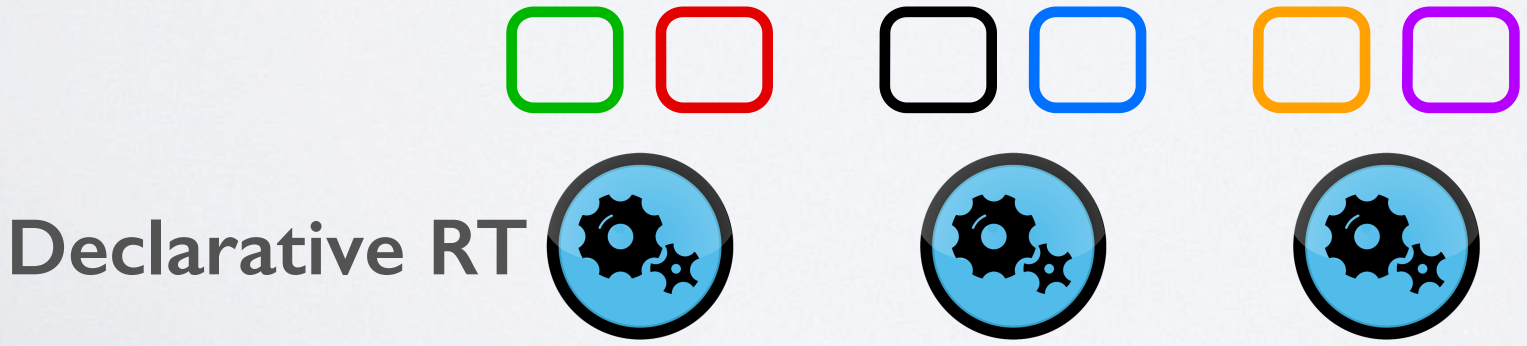
- **Incremental** computation and pushing for quick updates.

2. How to Guarantee Correctness

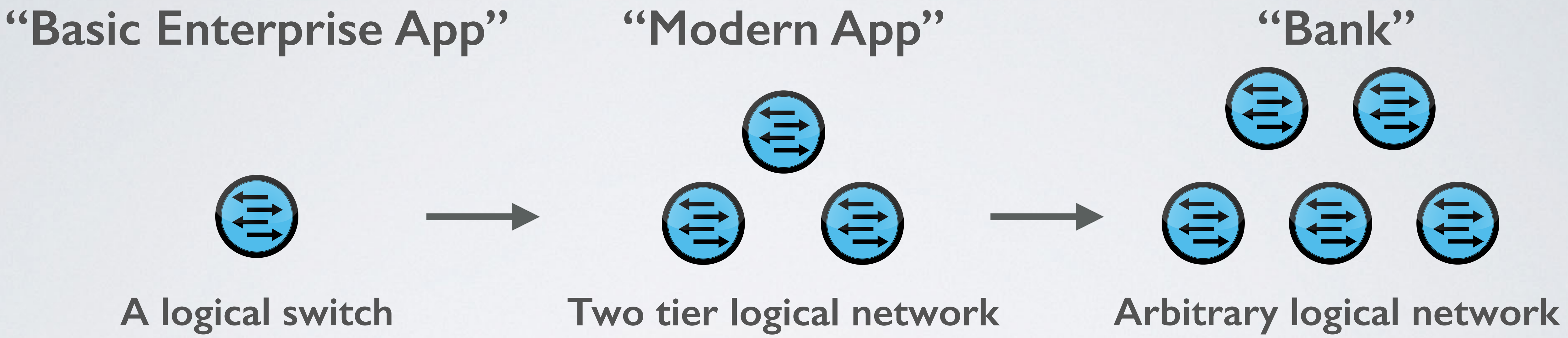
- Avoid all handwritten finite state machines, **machine generated** instead.

$\longrightarrow$ Datalog based declarative language to program F.

$\longrightarrow$ Shard the computation across controller cluster.

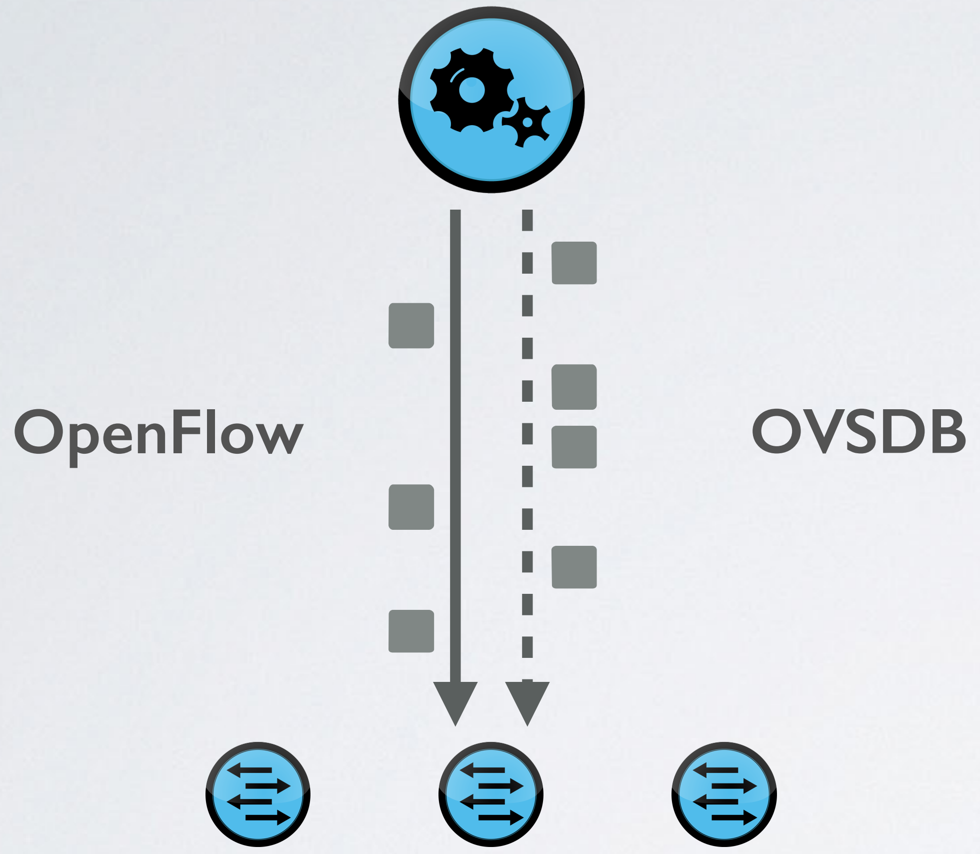


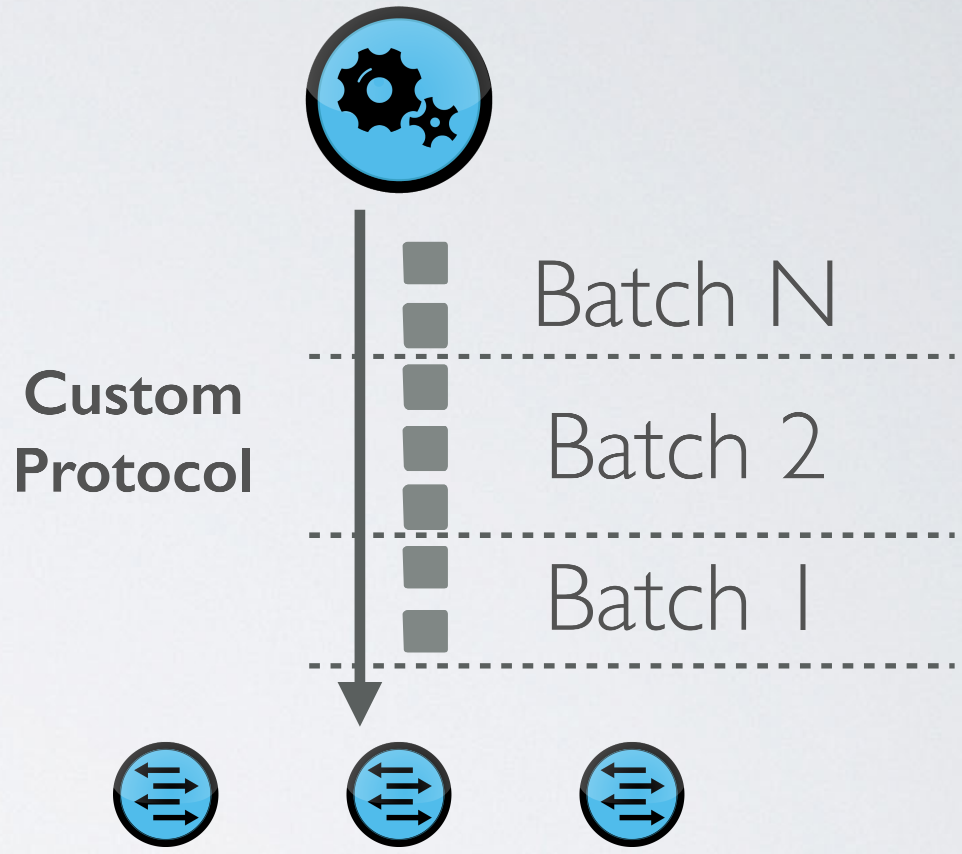
LESSONS LEARNT: ABSTRACTIONS



- Assumptions about logical network structure often embedded into the workload.
- A single L2 domain sufficient for initial, simple workloads.
- To support more complex workloads without changing them, more complex logical topologies become a necessity.

LESSONS: FAILURE ISOLATION





Two Channels, No Atomic Updates

- Proactive pushing of all state not enough to decouple controllers from data plane.
- Connection may die while pushing updates.

↓

Data plane may operate over incomplete state!

One Channel, Atomic Updates

- Atomically applied, batched updates.
- Connection failure does not result in incomplete state.

↓

At most old state.

LESSONS: SCALING

OPENFLOW IS EXPENSIVE

Too primitive

- Simple operations take several flow entries.
  - For example, tunnel failover, encapsulation header ops.
  - Lots of redundancy.

Too tightly coupled

- Each switch requires some flow customization; can't just blindly replicate flows.
- To compute flow entries, may have to wait for responses from the OVS configuration database.

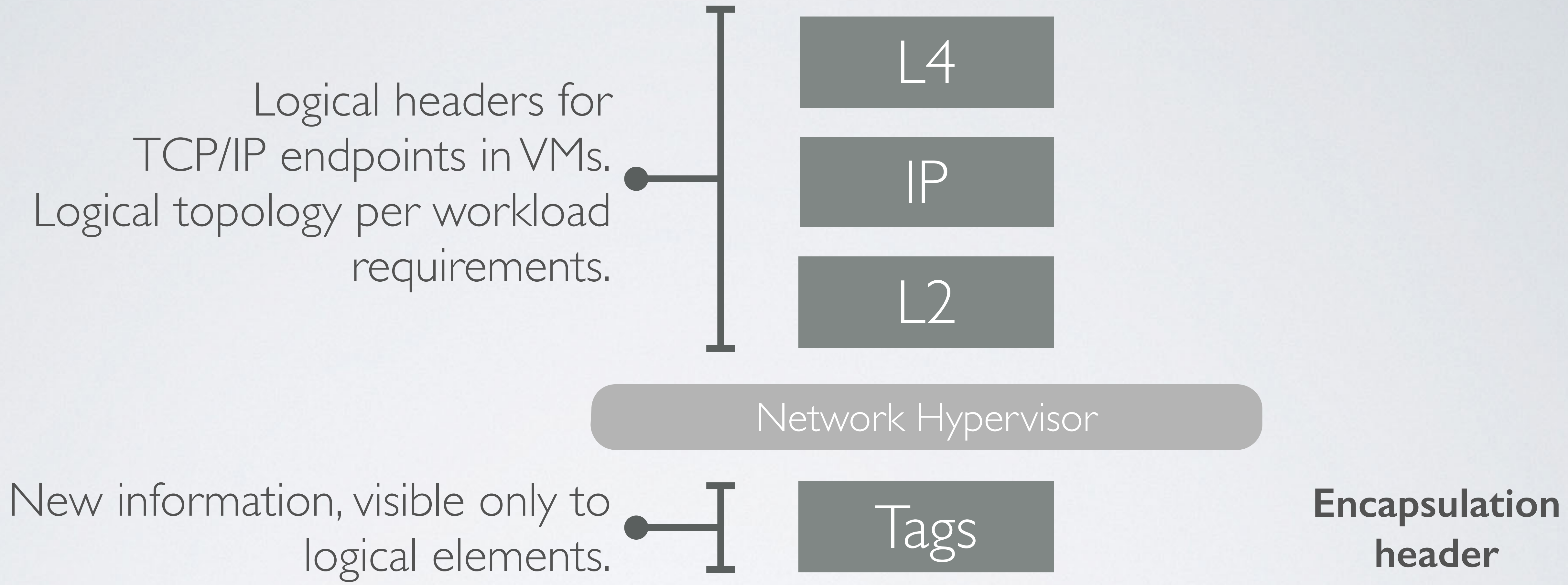
↓

Replace a OF & OVSDB with a network virtualization specific protocol.

↓

OpenFlow becomes a protocol internal to the hypervisor.

CONCLUSION: WHAT'S NEXT


Logical headers for TCP/IP endpoints in VMs. Logical topology per workload requirements.
L4
IP
L2
Network Hypervisor
New information, visible only to logical elements.
Tags
Encapsulation header

Without Network Virtualization

- Workload may run on a topology where addresses provide little information.
- For instance, firewall rules defined over exact /32 addresses!

With Network Virtualization

- New "out-of-band" header fields without breaking legacy TCP/IP stacks.
- **Huge** implications to enforcing security policies: groups, users in packet…

THANK YOU! QUESTIONS?