

# Software Defined Networking

*C-MUG*

January 2013

# A Brave New World

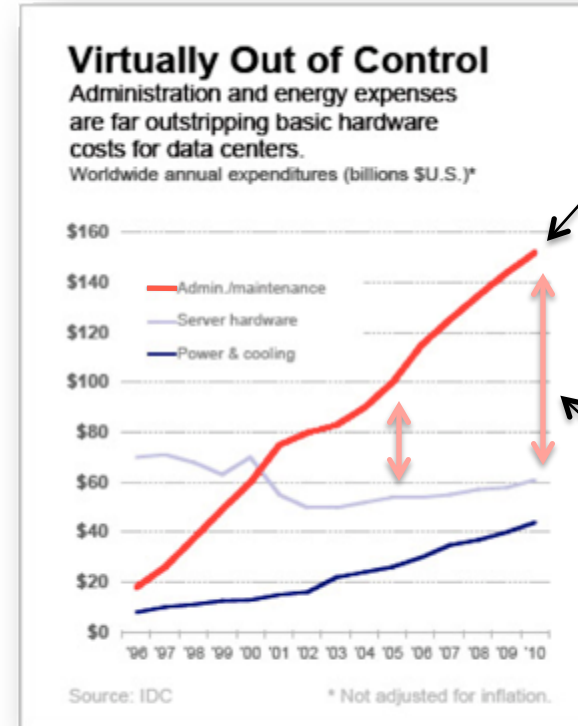
## Software Defined Networking:

- What problems are we (the industry) trying to solve?
- Understanding the SDN approach
- What does a real world SDN solution look like?

*What does SDN mean to me, the network professional...?*

# Networking's Dirty Secret

- The networking industry is still driven by speeds-and-feeds, but OPEX remains unsolved
- Moore's Law for hardware
  - Brought down costs
  - Delivered server virtualization
- No Moore's Law for OPEX
  - Manageability is just as important as performance



No Moore's law  
for OPEX

What will happen  
when servers go  
from 1G to 10G?  
How big does this  
gap become?

# The SDN Approach

*How do we solve problems the industry has ignored?*

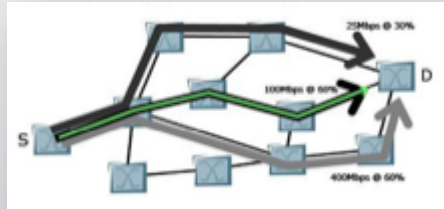
## Policy Provisioning

Edge policy statically-defined in edge device configuration



## Bandwidth Engineering

Difficult to engineer bandwidth per-flow (requires MPLS/TE)



## No Central Control

No single-point of administration and control



# The Networking Feature Gap



## Granular Path Control

- Create and manage multiple paths (at scale)
- Engineer flows to specific forwarding paths
- Most protocols still based on Shortest-Path-First (SPF)



## Network-Wide Awareness and Correlation

- Protocols are distributed with distributed configuration
- Awareness of endpoint location and mobility
- Dynamic policy follows logical endpoints



## Distributed Protocols / Static Configuration

- Single point of configuration for policy and behavior
- Programmatic Interface (Northbound) for Orchestration

# SDN in Perspective

*How do we break away from 25 years of network legacy?*

- **Can we design networks to behave as a single entity?**
  - ... not a collection of distributed systems
- **Can we provide a centralized point for policy?**
  - Dynamically allocate bandwidth based on application needs
  - Single point of policy administration and control
- **How can we remove manual tasks from running the network?**
  - Let the network automatically correlate policies to endpoints
  - Dynamically program path and policy control
  - Allow external systems to ask the network for services (without opening a trouble-ticket)



# Reinventing (DIY) Networking



- In 2005 Google was the first to build their own network hardware
- First it was rumored, then:
  - “Mystery Google Switch Shows up in Small-Town Iowa”
  - Google Planet8541 Pluto Edge Switch
- *They were followed by Amazon some time later...*



*Why? Because they **could not buy** a network that **did what they needed**.*

# Optics versus packets

- Rethinking the design of traditional data center backbones
- Can **optical** technology **improve scaling**?
- Google was the first with DIY, they are the first with optical in the DC

## Helios: A Hybrid Electrical/Optical Switch Architecture for Modular Data Centers

Nathan Farrington, George Porter, Sivasankar Radhakrishnan, Hamid Hajabdolali Bazzaz, Vikram Subramanya, Yeshaiahu Fairman, George Papan, and Amin Vahdat

University of California, San Diego

### Abstract

The basic building block of ever larger data centers has shifted from a rack to a modular container with hundreds or even thousands of servers. Delivering scalable bandwidth among such containers is a challenge. A number of recent efforts promise full bisection bandwidth between all servers, though with significant cost, complexity, and power consumption. We present Helios, a hybrid electrical/optical switch architecture that can deliver significant reductions in the number of switching elements, cabling, cost, and power consumption relative to recently proposed data center network architectures. We explore architectural trade offs and challenges associated with realizing these benefits through the evolution of a fully functional Helios prototype.

### Categories and Subject Descriptors

C.2.1 (Computer-Communication Networks): Network Architecture and Design—circuit-switching networks, packet-switching networks, network topology

### General Terms

Design, Experimentation, Measurement, Performance

### Keywords

Data Center Networks, Optical Networks

### 1. INTRODUCTION

The past few years have seen the emergence of the modular data center [12], a self-contained shipping container complete with servers, network, and cooling, which many refer to as a pod. Organizations like Google and Microsoft have begun constructing large data centers out of pods, and many traditional server vendors now offer products in this space [8, 13, 15, 16]. Pods are now the focus of systems [27] and networking research [10]. Each pod typically holds between 250 and 1,000 servers. At these scales, it is possible to construct a non-blocking switch fabric to interconnect all

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.  
SIGCOMM '10, August 30–September 3, 2010, New Delhi, India.  
Copyright 2010 ACM 978-1-4503-0200-2/10/08 ... \$10.00.



Figure 1: Helios is a 3-level multi-rooted tree of pod switches and core switches. The core consists of both traditional electrical packet switches and MEEMS-based optical circuit switches. Superlinks are defined in §3.1.

of the servers within an individual pod. However, interconnecting hundreds to thousands of such pods to form a larger data center remains a significant challenge.

Supporting efficient inter-pod communication is important because a key requirement in data centers is flexibility in placement of computation and services. For example, a cloud computing service such as EC2 may wish to place the multiple virtual machines comprising a customer's service on the physical machines with the most capacity irrespective of their location in the data center. Unfortunately, if these physical machines happen to be spread across multiple pods, network bottlenecks may result in unacceptable performance. Similarly, a large-scale Internet search engine may run on thousands of servers spread across multiple pods with significant inter-pod bandwidth requirements. Finally, there may be periodic bandwidth requirements for virtual machine backup or hotspots between partitioned computation and storage (e.g., between EC2 and S3).

In general, as services and access patterns evolve, different subsets of nodes within the data center may become tightly coupled, and require significant bandwidth to prevent bottlenecks. One way to achieve good performance is to statically provision bandwidth between sets of nodes with known communication locality. Unfortunately, given current data





# OpenFlow in Perspective

## SDN



- SDN is much more than OpenFlow
  - *Adding OpenFlow capabilities doesn't make your network SDN-compliant any more than adding jet fuel to your car makes it fly.*
- OpenFlow is an important toolkit for low-level control of forwarding
  - Assembly language of network programmability / SDN
  - Low-level control of forwarding ASICs
  - It's about controlling *your own* forwarding behavior
- SDN is a method for solving a problem,  
*not a product or a protocol*

***Let's look at problems people are trying to solve:***

# Google's OpenFlow SDN

Networks Today Are Very  
Exciting

Google



## Networking Reality for the Masses

Google

- I am on the edge of my seat when:
  - A link fails
  - I perform a switch software upgrade
- I pray to my favorite deity when:
  - Multiple applications try to share the same network
  - I add a new customer to an existing network
- I don't even bother trying to:
  - Expand my network fabric
  - Set a security/isolation policy
  - Migrate a virtual machine
  - Enforce performance isolation
  - Use traffic engineering
  - Introduce a product from a new vendor

Google Confidential and Proprietary

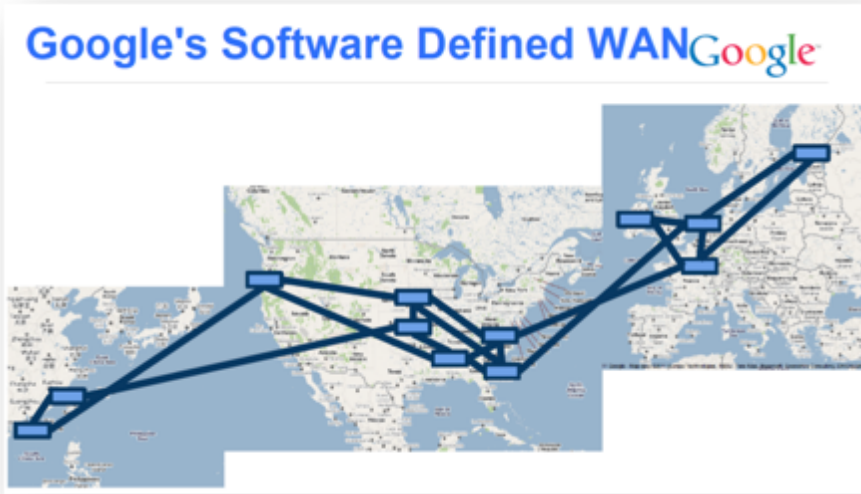
**1Q 2010:** GOOG begins OpenFlow Project

**1Q 2012:** All Inter-DC traffic on SDN-Controlled WAN

### ■ Benefits (Per Google)

- Unified view of the network fabric
- High utilization **centralized traffic engineering** provides a global view of the supply and demand of network resources.
- Faster failure handling whether it be link, node or otherwise are handled much faster...systems converge more rapidly to target optimum and the behavior is predictable.
- With SDN, better and more rigorous testing is done ahead of rollout accelerating deployment....hitless upgrades
- High fidelity test environment, the entire backbone is emulated in software.
- **Compute capability of network devices is no longer a limiting factor as control and management resides on external servers/ controllers. Large-scale computation, path optimization in our case, is done using the latest generation of servers**

# GOOG's OpenFlow SDN in Perspective



## *Focus: the bandwidth engineering problem*

- Centralized control of paths / bandwidth
- Requires low-level control of forwarding behavior (OF)
- Natively integrate network into the Application Stack

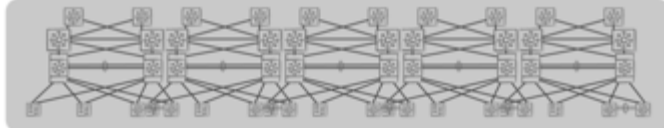
Network vendors unable to provide effective bandwidth engineering:

- Google is ready, willing and able to operationalize home-grown network solution
- Leverage “big-data” solution for bandwidth optimization problem (Google’s core-competence)
- Offline path computation via controller, hardware forwarding programmed via OpenFlow
- Integration with RPC manager allows for call-admission control (CAC)

# Overlay Solution – Virtualize

## The network as a barrier to the cloud

Device-by-device management      No easy scale-out model  
Distributed state      No consistency across physical & virtual hosting  
VLANs don't scale      **Poor Programmatic Control**  
Difficult to meet SLAs      VM bound to single pod  
Hard to spread workloads across failure zones  
Vendor specific configuration tools      **Limited Multi-Tenancy**  
L3-L7 choke points      No customers control on addressing      Accountability  
**Inflexible Workload Placement**      Reconfiguration required on migration

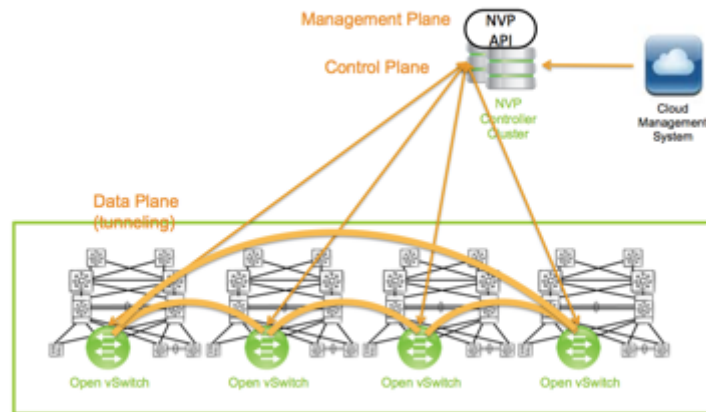


Physical Network



6

## Network Virtualization Platform (NVP)



16

### *Focus: the edge policy problem*

- Move control over edge port policy from the network to the virtual switch in the server
- Central Control of policy, coupled with Data Center Orchestration Systems

# SDN Reality Check:

If you think SDN and OpenFlow are about making networking cheaper  
... *you've missed the point.*



*"Reality is people spend a lot of money on networking gear, once it is installed it works. Don't touch it, it may break it. Once people have adopted a certain network topology, they are highly unlikely to change it. **Unless something really better comes along.**"*

~7:21 mark @ NFD April 2012

SDN is about delivering NEW CAPABILITIES

– in a word, it's about: **Innovation**



# Data Center: A Brave New World



# Rethinking Network Fundamentals



*The datacenter is the first place with **truly different** network requirements*

**Legacy designs are no longer an option**

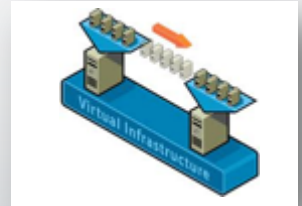
## Cross-sectional Bandwidth

- Maximize bandwidth / Zero latency
- Physical location matters in network



## Endpoint Mobility

- vMotion
- Dynamic Resource Scheduling



## Centralized Network Orchestration

- Single point of control, **API-driven**
- The network is orchestrated as another Data Center resource





# The performance problem...

## “To Infinity and Beyond!”

- Buzz Lightyear (Toy Story, 1995)

### Data Center Network Requirements:

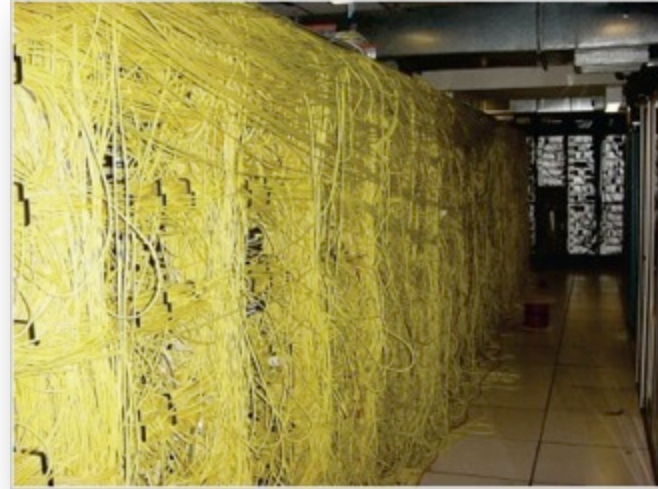
- Infinite any-to-any bandwidth
- Infinitely-low latency

# Where's my Terabit Ethernet Switch?

*Faster network connections would mean faster development, as new internal or customer-facing applications were quickly run through their paces and refined. Ideally, each server could send a full 1G bps to any other server in the data center, but that would **require the 64 Terabit Ethernet pipes Facebook can't buy today** -- or 6,400 of today's 10-Gigabit Ethernet connections, which isn't really feasible, he said. A fabric of that size would require 160 of the largest switches available, Lee said. Just imagine trying to manage these," Lee said. "There are a lot of things that are missing from this data center fabric. All of these things are essential to running an Ethernet fabric, yet **none of them exist at this scale.**" Lee said his job is to piece together the elements that are available.*

**- Donn Lee, Facebook**

"Facebook Sees Need for Terabit Ethernet", PCWorld, 2/2/2010



8,000 Node Fat Tree Design

- **10 tons of cable**
- 55,296 cat-6 cables
- 1,128 separate cable bundles

# Scale: Chasing the Dream

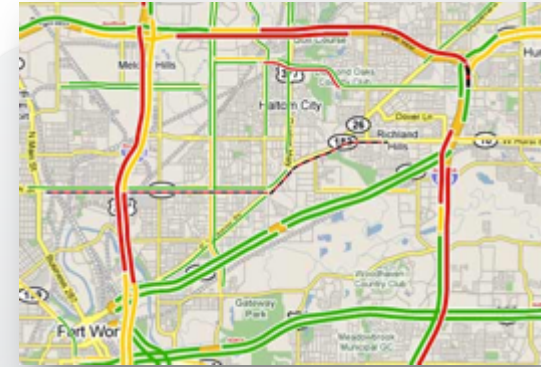
*Scale-up reliability **gets expensive faster than reliable...**  
asymptotically approaches “unaffordable” but  
never gets to “good enough”.*

**- James Hamilton, Amazon**

# Bandwidth is Inflexible

*“The key observation is that the available bisectional bandwidth is **inflexible**. It cannot be allocated to the points in the data center that would benefit most from it, since it is fixed to a pre-defined topology. While supporting arbitrary all-to-all communication may not be required, supporting bursty inter-pod communication requires a non-blocking topology using traditional techniques. This results in a dilemma: unfortunately, network designers must pay for the expense and complexity of a **non-blocking topology** despite the face that vast, though dynamically changing, portions of the topology **will sit idle** if the network designer wishes to prevent localized bottlenecks.”*

**- Amin Vahdat, Google**



*“(...) existing techniques perform sub-optimally due to the **failure to use multipath diversity** (...)”*

**The Case for Fine-Grained Traffic Engineering in Data Centers**

– Microsoft Research (2010)

<http://research.microsoft.com/apps/pubs/?id=136776>

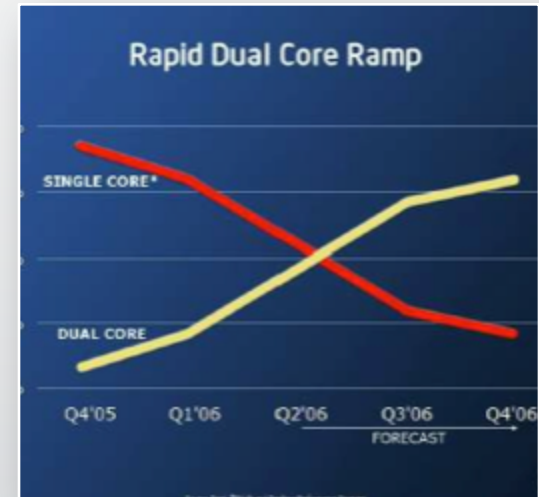
# Scaling Bandwidth: The Fat Tree



- Overcome protocol limitations
  - Leverage equal-cost forwarding
    - L3 Overlay (VXLAN, NV-GRE, STT)
    - TRILL / FabricPath or MPLS
- “We’re gonna need a bigger tree”
  - Fat tree: core requirements scale geometrically
  - Affordable Core uplinks not much faster than edge ports (40G vs 10G)
  - Always a limited diameter for non-blocking performance (“pod-size” will be limited)

# Scaling Trees Needs Dense Aggregation

*This starts to look a lot like the Single Core Processor of networking...*

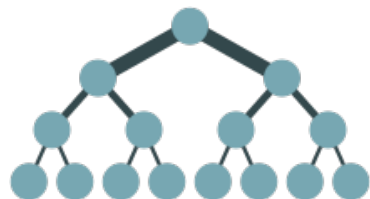


# Running Out of Scalability:

## *The Point of Diminishing Return in CPUs...*

In early 2003, Intel showed the design of Tejas and a plan to release it sometime in 2004, but put it off to 2005 later. Intel, however, announced it canceled the development on May 7, 2004. Analysts attribute the delay and eventual cancellation to the heat problems due to the extreme power consumption of the core, as that was the case in development of Prescott and its mediocre performance increase over [Northwood](#). This cancellation reflected Intel's intention to focus on dual-core chips for the [Itanium](#) platform. With respect to

# Single Core Aggregation Example



- Example: 100,000 2x10 GbE Servers
  - At Over-Subscription Ratio of 1:1
    - Need 2,000,000 GbE equivalent = **50,000 x 40 GbE**
    - for CLOS topology: need additional ~100,000 ports
  - Largest 40 GbE agg switch today is 72 ports
    - We will see 96 ports coming soon in 2U, at 1-2 kW
  - 100k servers = 1500 switches
    - **1.5-3.0 MW** – just for interconnection overhead

***And we have not done a thing about Amin's inflexible bisectional bandwidth problem...***



# Don't build bigger, build **smarter**...



- Computation. Algorithms. Math. **This is SDN**
- Derive network topology and orchestration directly from application or tenant configuration.

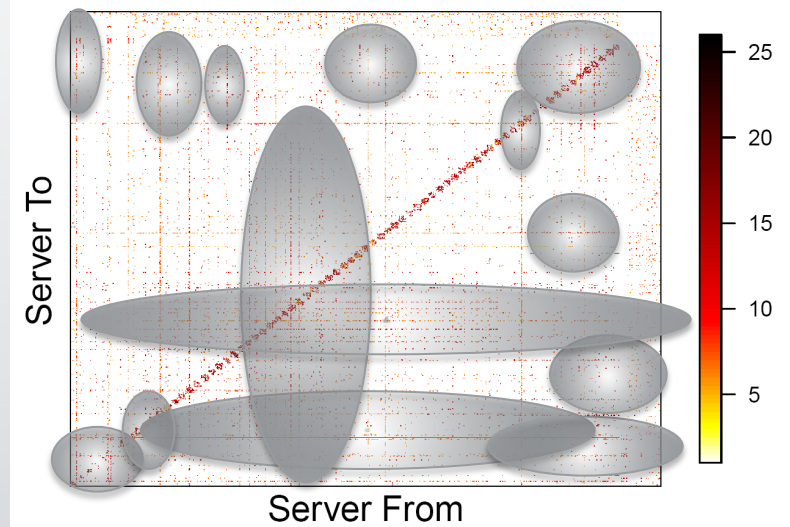
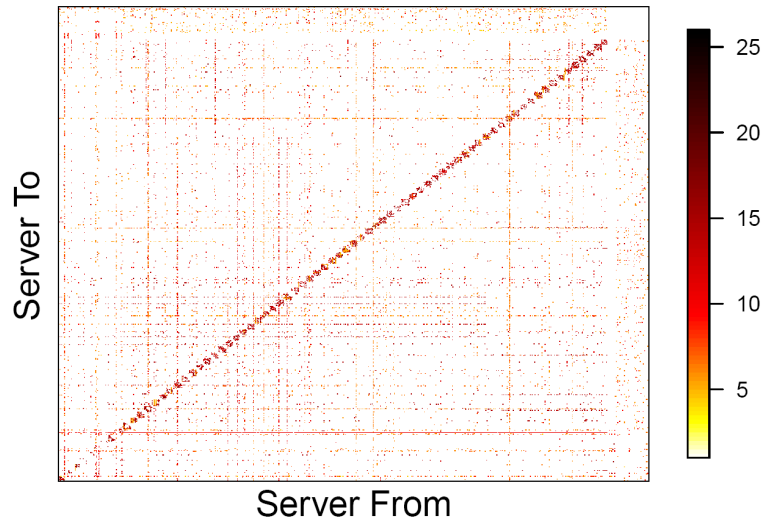


- Deliver high-bandwidth, optical interconnect
- Allocate bandwidth **on-demand**, where it's needed
- Maintain bandwidth/loss/latency SLAs



- Centralized, Controller-based solution
- API-driven, Programmable by design
- Application **Affinity** intelligently delivers bandwidth

# Do you see the Affinities?



*Social networking for servers...*

# SDN Value: Correlation






## Automated, Bandwidth-on-Demand

- Old school: policies defined manually at edge ports
  - Ex: CoS Classification via ACL
- Plexxi SDN: **Affinity-driven network policy**
  - Flexible, Optical Infrastructure allows bandwidth allocation
  - Policy dynamically follows endpoints



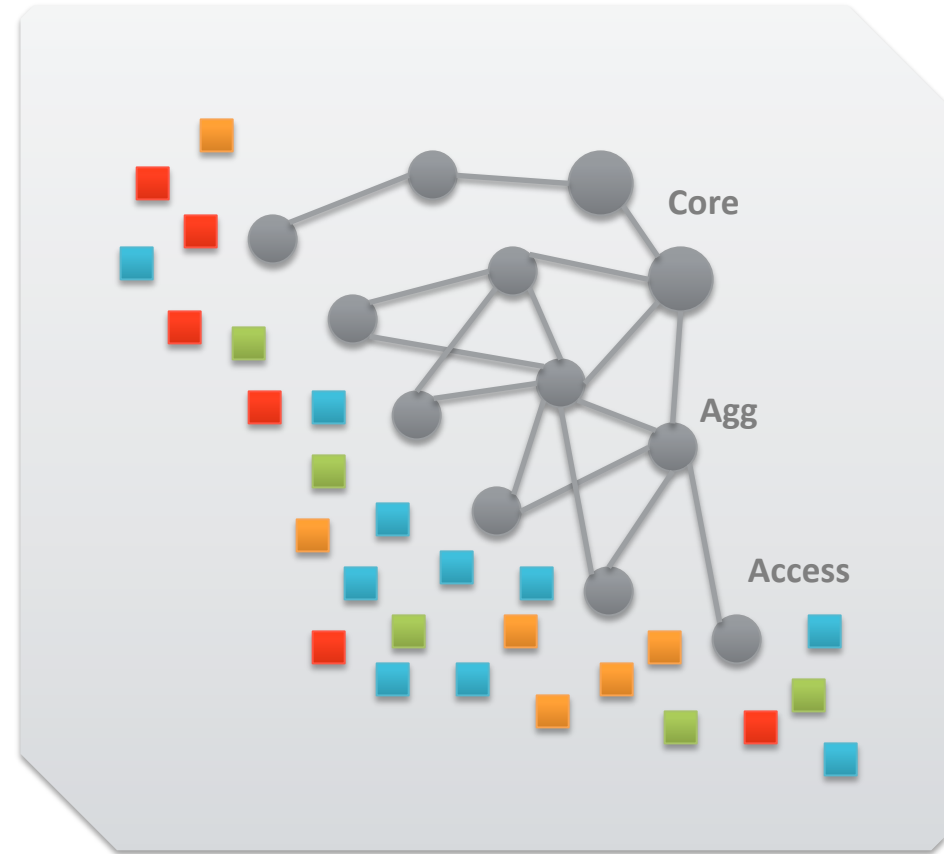
1. Define Affinity Policies in Controller
2. Gather node data from external sources
  - E.g. vCenter / OpenStack / OSS or Provisioning Systems
3. Automatically correlate node IDs to network location
4. Dynamically program policy rules to network location

# Plexxi Elements

- **Open APIs for expressing application and/or tenant affinities.**  
 Applies equally to virtualized and/or clustered and/or multi-tenant.
- **Underlying physical architecture is efficient, flexible and fluid.**  
 A truly “orchestrate-able” infrastructure.
- **A robust controller implementation that leverages modern computing platforms and computation.**  
 Predictable, deterministic, and more reliable than legacy layered protocols.

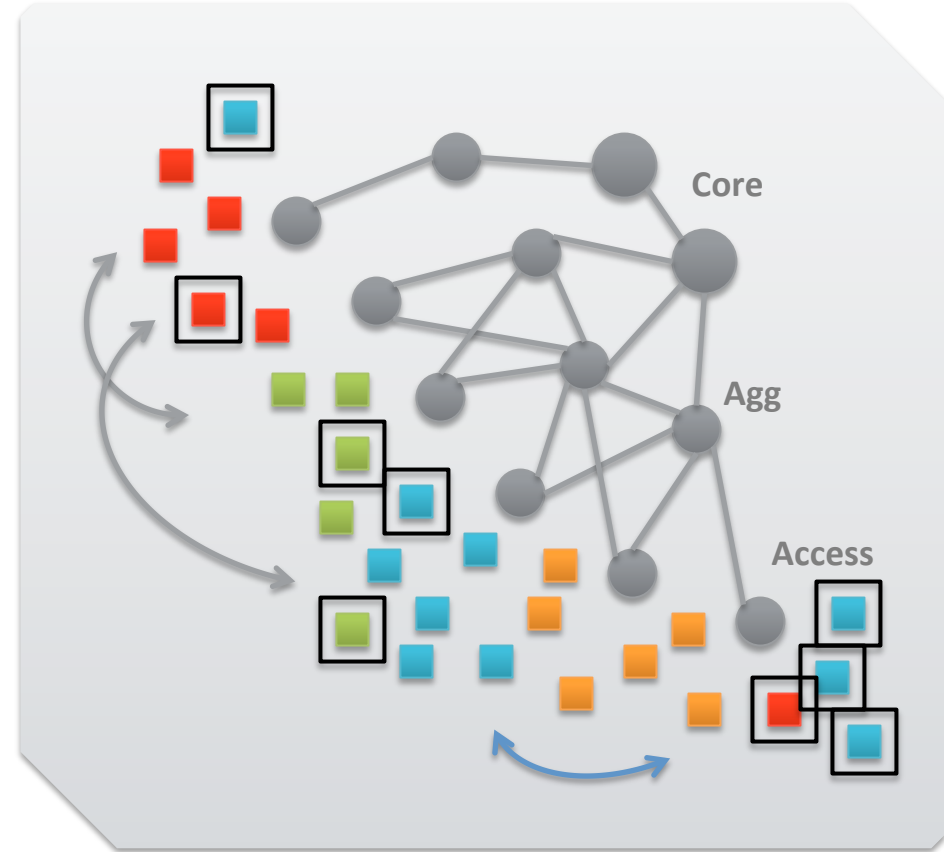
# Affinity Fitting

- You want to organize the network topology based on the application workloads.
- But, in a legacy network, you are constrained by where the wires go and the operation of legacy protocols...



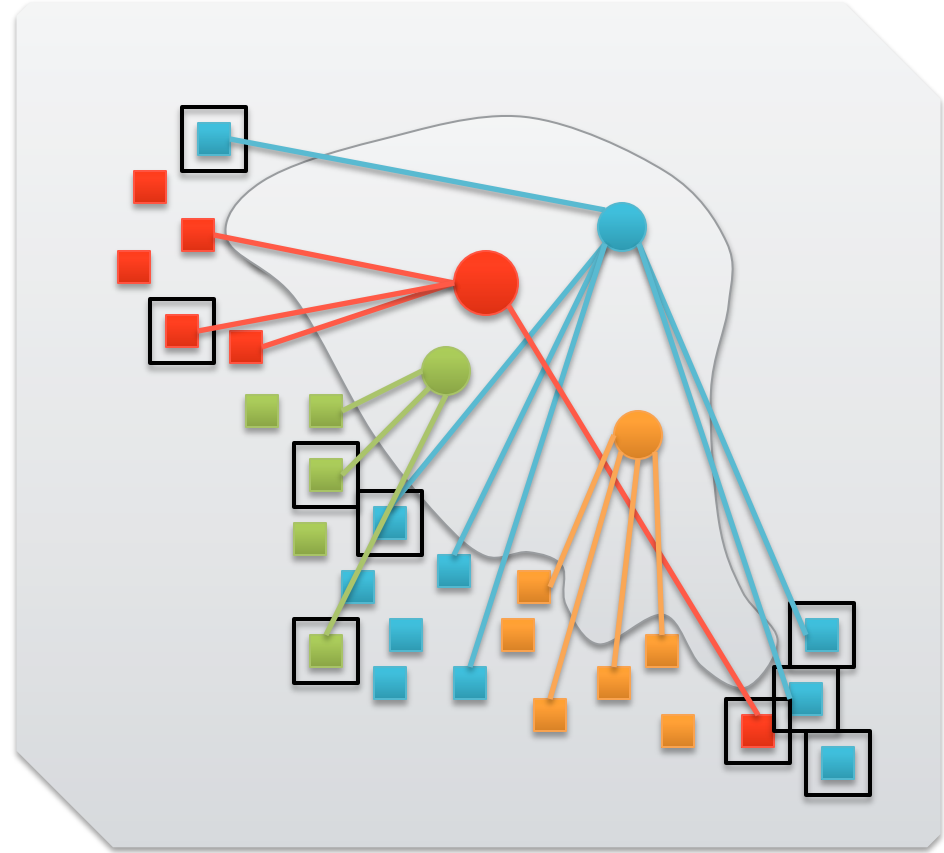
# Orchestration

- You can make progress via VM mobility
- But, there are difficult limitations, and, in many cases, application resources are shared or fixed in place
  - Database servers
  - Gateways
  - Clusters
  - ...



# What if...

- There was a way to build a single-tier, *MULTICORE* network?
- This changes the economics of hierarchical, switched networks...



# The Evolution of Computer Networking

*We have reached an inflection point...*

Shared  
Media

Switched  
Hierarchies

What's  
Next

1<sup>st</sup> Epoch

2<sup>nd</sup> Epoch

3<sup>rd</sup> Epoch

1980

1996

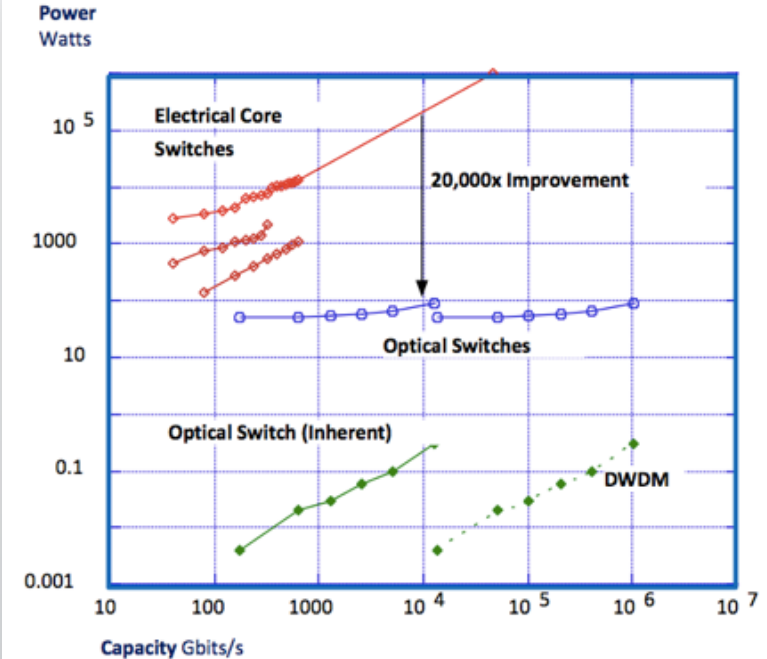
2012



# The Future Future...



*As we look toward building exabit-scale networks...*



Source: Dr. John Bowers, UCSB, Calient Technologies

# What Does SDN Mean to me?

- The need for your expertise is not going away
  - Expertise is still needed to solve hard problems
- Day-to-Day Networking may get less painful
  - we can finally shed some of the old legacy that makes networking complicated / hard to manage
  - The network can now integrate more tightly with applications and Data Center orchestration systems
  - You will be able to work on interesting problems – rather than mundane problems
- You don't need to become a programmer
  - But learning basics like Python and REST is useful

