Datacenter Networking

Major Components of a Datacenter

• Computing hardware (equipment racks)
• Power supply and distribution hardware
• Cooling hardware and cooling fluid distribution hardware
• Network infrastructure
• IT Personnel and office equipment
Growth Trends in Datacenters

• Load on network & servers continues to rapidly grow
  – Rapid growth: a rough estimate of annual growth rate:
    enterprise datacenters: ~35%, Internet datacenters: 50% - 100%
  – Information access anywhere, anytime, from many devices
    • Desktops, laptops, PDAs & smart phones, sensor networks, proliferation of broadband

• Mainstream servers moving towards higher speed links
  – 1-GbE to 10-GbE in 2008-2009
  – 10-GbE to 40-GbE in 2010-2012

• High-speed datacenter-MAN/WAN connectivity
  – High-speed datacenter syncing for disaster recovery
Datacenter Networking

- A large part of the total cost of the DC hardware
  - Large routers and high-bandwidth switches are very expensive
- Relatively unreliable – many components may fail.
- Many major operators and companies design their own datacenter networking to save money and improve reliability/scalability/performance.
  - The topology is often known
  - The number of nodes is limited
  - The protocols used in the DC are known
- Security is simpler inside the data center, but challenging at the border
- We can distribute applications to servers to distribute load and minimize hot spots
**Networking components (examples)**

**Highly scalable DC Border Routers**
- 3.2 Tbps capacity in a single chassis
- 10 Million routes, 1 Million in hardware
- 2,000 BGP peers
- 2K L3 VPNs, 16K L2 VPNs
- High port density for GE and 10GE application connectivity
- Security

**High Performance & High Density Switches & Routers**
- Scaling to 512 10GbE ports per chassis
- No need for proprietary protocols to scale
Common data center topology

- **Core**
  - Layer-3 router

- **Aggregation**
  - Layer-2/3 switch

- **Access**
  - Layer-2 switch

- Servers

- Internet
Data center network design goals

- High network bandwidth, low latency
- Reduce the need for large switches in the core
- Simplify the software, push complexity to the edge of the network
- Improve reliability
- Reduce capital and operating cost
Datacenter Networking

Avoid this... and simplify this...
Can we avoid using high-end switches?

- Expensive high-end switches to scale up
- Single point of failure and bandwidth bottleneck
  - Experiences from real systems

**DCell**
Interconnect

DCell Ideas

• #1: Use mini-switches to scale out
• #2: Leverage servers to be part of the routing infrastructure
  – Servers have multiple ports and need to forward packets
• #3: Use recursion to scale and build complete graph to increase capacity
Data Center Networking

One approach: switched network with a hypercube interconnect

- Leaf switch: 40 1Gbps ports + 2 10 Gbps ports.
  - One switch per rack.
  - Not replicated (if a switch fails, lose one rack of capacity)
- Core switch: 10 10Gbps ports
  - Form a hypercube
- Hypercube – high-dimensional rectangle
Hypercube properties

- Minimum hop count
- Even load distribution for all-all communication.
- Can route around switch/link failures.
- Simple routing:
  - Outport = f(Dest xor NodeNum)
  - No routing tables
Interconnect

A 16-node (dimension 4) hypercube
64-switch Hypercube

Core switch: 10Gbps port x 10

How many servers can be connected in this system?

81920 servers with 1Gbps bandwidth

Leaf switch: 1Gbps port x 40 + 10Gbps port x 2.
Inside Project Blackbox, racks of up to 38 servers apiece generate tremendous heat. A panel of fans in front of each rack forces warm exhaust air through a heat exchanger, which cools the air for the next rack (detail), and so on in a continuous loop.

**DESIGN SPECS**
- Dimensions: 8 x 8 x 20 feet
- Weight: 20,000 pounds
- Cooling water supply: 60 gallons per minute
- Computing capacity: 7 terabytes
- Data storage: 2 petabytes
Appendix
Typical layer 2 & Layer 3 in existing systems

- **Layer 2**
  - One spanning tree for entire network
    - Prevents looping
    - Ignores alternate paths

- **Layer 3**
  - Shortest path routing between source and destination
  - Best-effort delivery
Interconnect

Problem With common DC topology

• Single point of failure

• Over subscription of links higher up in the topology
  – Trade off between cost and provisioning

• Layer 3 will only use one of the existing equal cost paths

• Packet re-ordering occurs if layer 3 blindly takes advantage of path diversity
Fat-tree based Solution

Connect hosts together using a fat-tree topology

- Infrastructure consists of cheap devices
  - Each port supports same speed as the end host
- All devices can transmit at line speed if packets are distributed along existing paths
- A k-ary fat-tree is composed of switches with k ports
  - How many switches? ... $5k^2/4$
  - How many connected hosts? ... $k^3/4$
Use the same type of switches in the core, aggregation, and edge, with each switch having $k$ ports.
Interconnect

Fat-tree Modified

• Enforce special addressing scheme in DC
  – Allows host attached to same switch to route only through switch
  – Allows inter-pod traffic to stay within pod
  – unused.PodNumber.switchnumber.Endhost

• Use two level look-ups to distribute traffic and maintain packet ordering.
Interconnect

2-Level look-ups

• First level is prefix lookup
  – Used to route down the topology to endhost
• Second level is a suffix lookup
  – Used to route up towards core
  – Diffuses and spreads out traffic
  – Maintains packet ordering by using the same ports for the same endhost
Comparison of several schemes

- Hypercube: high-degree interconnect for large net, difficult to scale incrementally
- Butterfly and fat-tree: cannot scale as fast as DCell
- De Bruijn: cannot incrementally expand
- DCell: low bandwidth between two clusters (sub-DCells)