eecs589 ADVANCED COMPUTER NETWORKS

Roy, A., et al., "Inside the Social Network's (Datacenter) Network" *Proc. of ACM SIGCOMM '15*, 45(4):123-137, Oct. 2015

Microsoft Datacenter Traffic

Previous Microsoft studies found datacenter traffic to be:

- 50-80% rack local
- frequently concentrated and bursty
- bimodal in packet sizes (ACK/MTU)
- on/off
- mostly in small flows, <5 concurrent large flows

Facebook Datacenter Traffic

Characteristics of Facebook datacenter traffic:

- neither rack local nor all pairs
- demand is wide-spread, uniform, and stable due to load balancing
- small packets, continuous arrivals, not on/off
- many concurrent flows due to connection pooling
- rapidly changing, internally bursty heavy hitters, reducing the efficacy of traffic engineering
- only Hadoop's MapReduce-style traffic agrees with Microsoft's characterization

Implications

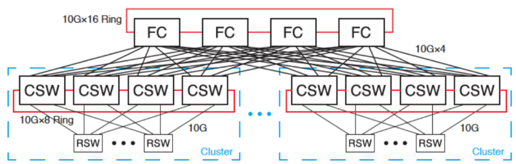
Datacenter network designs assume:

- worst-case, all-pair traffic matrix, with equal frequency and intensity ⇒ maximize bisection bandwidth
- hot-spots, due to oversubscription, to be alleviated with bypass, secondary connectivities (wireless, optical)
  - which requires traffic demand to be predictable and stable to be feasible
- stylized traffic allows for specialized switch design (buffer sizing, port count, etc.)

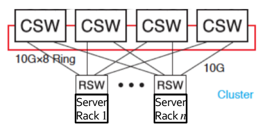
Datacenter Topology

Similar to Google's first gen network:

- multiple sites connected by a backbone
- each site contains one or more buildings (datacenters)
- each datacenter contains multiple clusters



- each cluster employs a 3-tier, 4-post topology
- 10-Gbps servers



Cluster

Unit of deployment

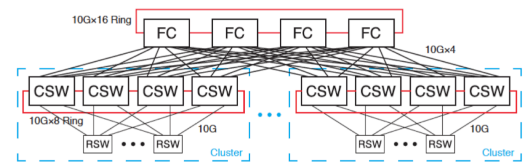
May be of a single function, e.g., cache cluster

Or multi-function: front-end cluster comprising web/front-end servers, load balancers, and cache servers

Inter-cluster, intra-datacenter connected by FC switches

Similar to Google,

- inter-datacenter, intra-site connected by aggregation switch
- inter-site connected by datacenter router



Server

Each server has precisely one role:

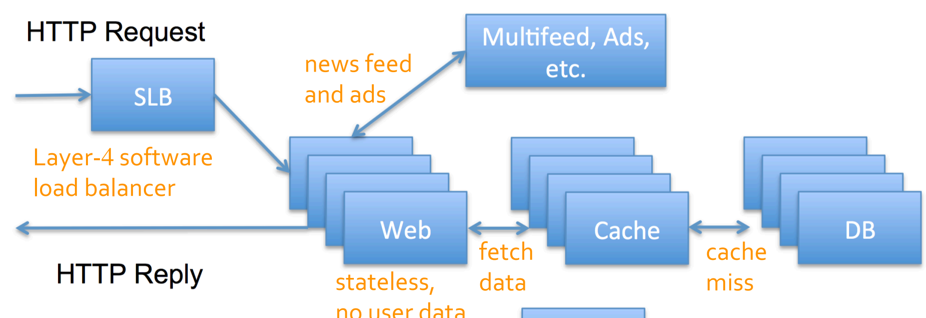
- web/front-end server
- mysql (db) server
- cache leader
- cache follower
- multifeed server to assemble news feed and serve ads
- Hadoop server for offline analysis and data mining

A small number of servers can be dynamically repurposed

No virtual machines (same as Microsoft)

Each rack contains only servers of the same role

Services



src\dst	Web	Cache	MF	SLB	Hadoop	Rest
Web	-	63.1	15.2	5.6	-	16.1
Cache-l	-	86.6	5.9	-	-	7.5
Cache-f	88.7	5.8	-	-	-	5.5
Hadoop	-	-	-	-	99.8	0.2

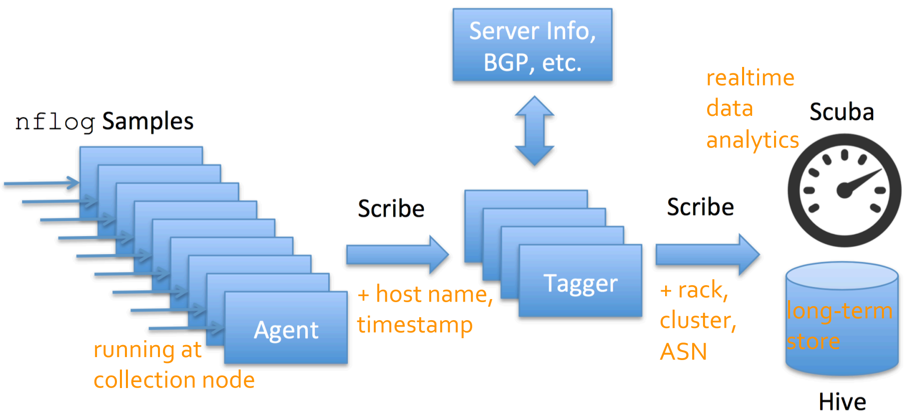
outbound traffic intensity (%)

offline analysis and data mining, not involved with serving end-user requests

Data Collection

Cannot collect every packet, instead use:

1. Fbflow: sample packet headers (1:30K sampling rate) across entire global network



Data Collection

Cannot collect every packet, instead use:

2. Port mirroring: collect all packet headers of a single machine or rack for a few minutes
   - by mirroring a ToR port to a collection host on the same rack
     - placement opportunistic, depending on space availability
   - a kernel module sitting atop the Ethernet driver extracts headers and spools it to remote storage
   - no loss
   - deployed at 5 different (type of) racks to monitor:
     - a rack of web servers
     - a Hadoop node
     - a cache leader node
     - a cache follower node
     - a multifeed node

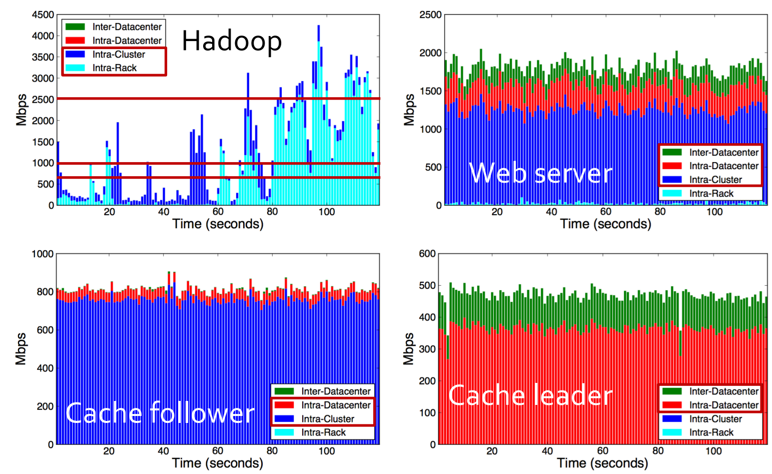
Traffic Characterization

Characterize traffic across 3 different types of cluster: Hadoop, Web/front-end, and cache clusters

Utilization:

- server to ToR links: < 1%, even in heaviest utilized Hadoop cluster, it's < 5%
- ToR to CSW links: median: 10-20%, with the busiest 5% reaching 23-46%
- CSW to FC links: higher

Locality

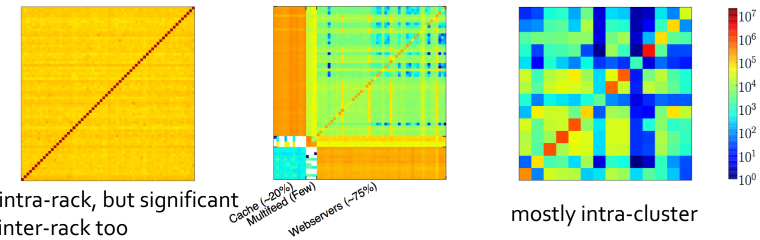


Relative proportions of the locality are stable despite diurnal traffic pattern

Implications of Locality

Traffic matrix:

heavy inter-rack between web servers and caches



heavy intra-rack, but significant (80%) inter-rack too

mostly intra-cluster

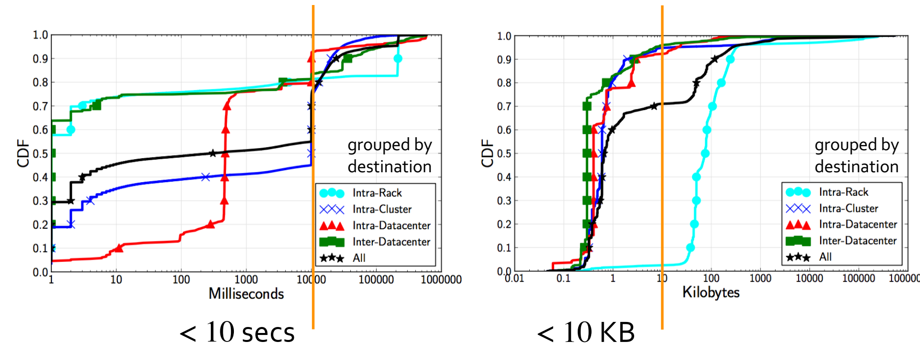
(a) Rack-to-rack, Hadoop cluster (b) Rack-to-rack, Frontend cluster (c) Cluster-to-cluster

Homogenous topology will lead to over-/under-provisioning in different parts of the datacenter

Stability of traffic patterns means no need for rapid reconfigurability

Outbound Flow Characteristics

Most Hadoop flows are short and small, but varies across servers

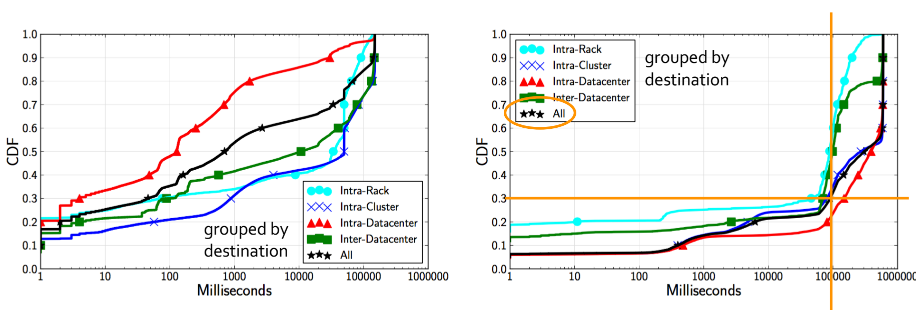


< 10 secs

< 10 KB

Outbound Flow Characteristics

Non-Hadoop flows are more uniform across servers due to load balancing and last longer due to connection pooling, but traffic per flow is bursty [surely on/off?]



(a) Web servers (b) Cache follower

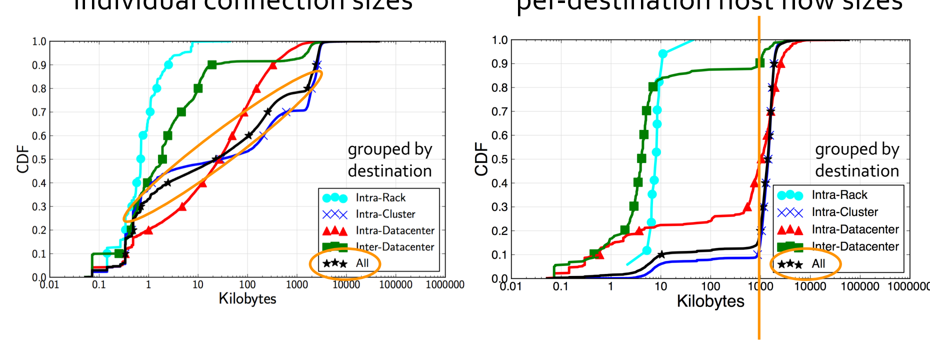
only 30% lasts < 100 secs

Outbound Flow Characteristics

Cache flow sizes reflect load balancing over time

individual connection sizes

per-destination host flow sizes

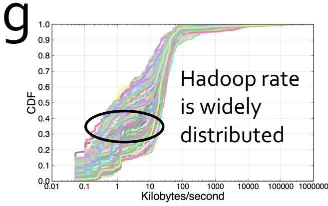


widely distributed

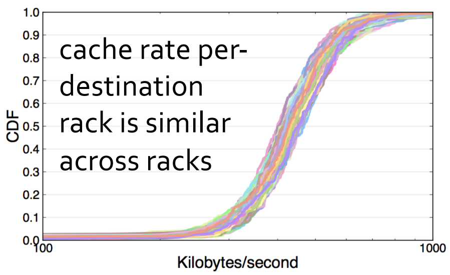
tightly distributed around 1 MB

Impact of Load Balancing

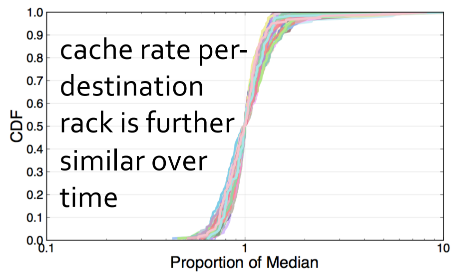
Load balancing smooths out traffic, reducing effectiveness of traffic engineering



(a) Hadoop (rate, each series is 1 second)



(b) Cache (rate, each series is 1 second)



(c) Cache (stability, each series is a rack)

Impact of Load Balancing

Load is monitored ⇒ large increases in load would be actively mitigated

Hot objects are temporarily cached at the web servers

Persistently popular objects are replicated across caches

Top-50 most popular objects are evenly spread across all caches

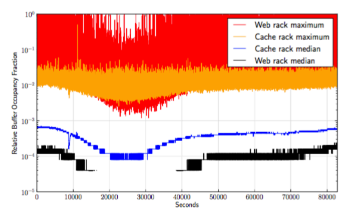
No heavy hitters (set of flows responsible for 50% of traffic volume) due to load balancing and caching

Switch Design

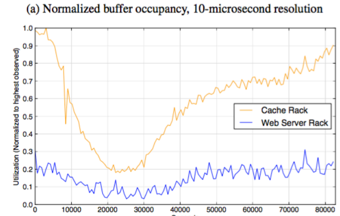
Low traffic volume (in bytes), but high packet rate: even at 10% utilization, median packet size of 175 bytes means 85% of link packet forwarding capacity [no Nagle?]

Packet arrivals from a single source host are not ON/OFF, but arrivals for a single destination host are ON/OFF

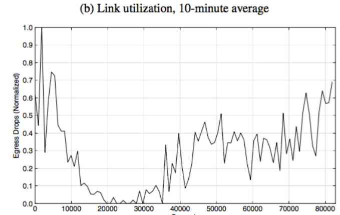
Buffers overflow, especially for web servers [LRD traffic after all?]



(a) Normalized buffer occupancy, 10-microsecond resolution



(b) Link utilization, 10-minute average



(c) Web rack egress drops, 15-minute average

Discussion

Traffic observed reflects the design and implementation of a single service, is it the best design and implementation?

Traffic characteristics change as:

- service changes, e.g., more videos
- implementation or design changes, e.g., is having a cache cluster the best design?
  - or would it be better to spread cache servers across clusters?

Can all datacenter traffic be so regularized?

If so, are remaining datacenter hard problems (research issues) above the network layer?