Inside the Social Network’s *Datacenter* Network

by Facebook

Peter Hu, zh369,

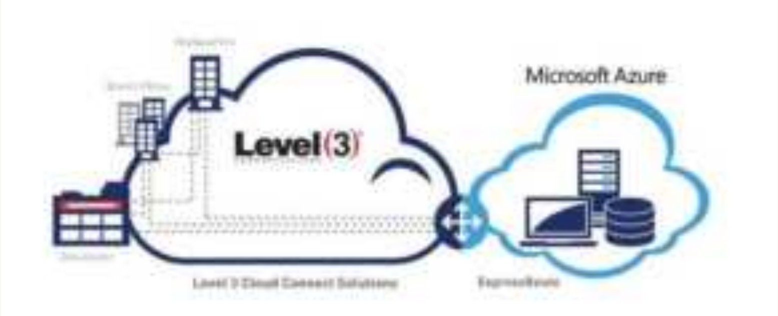
Magdalene College.

23 Oct 2025.

Presented at Network Architectures.

Contents

Contents



<u>Traditional</u> vs. <u>Social Network's</u> *Datacenter Network*

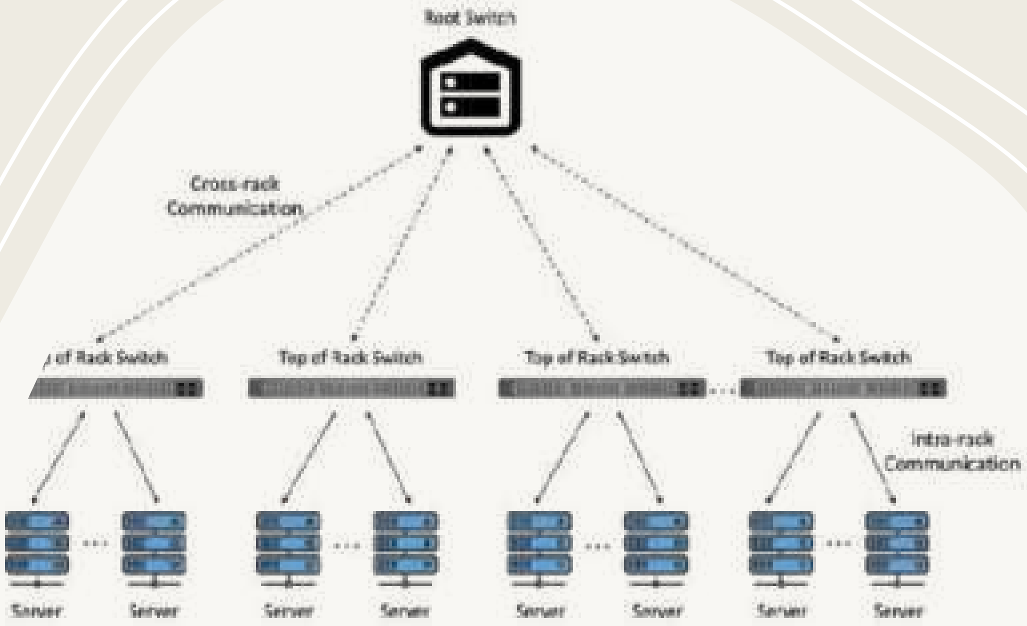
app, website, database, etc.

1. Traffic locality and Utilization

- **<u>Traditional</u> datacenters:**
  - **rack-local** — servers within a single rack communicate heavily
  - e.g., web front-end ↔ app ↔ DB tiers.
- predictable utilization patterns.

<u>Social network</u> datacenters:

- **less rack-local and more cross-cluster or cross-fabric** — social interactions, feeds, and graph traversals mean data must move between many services
- Result: **Lower per-link utilization** but **wider spread of demand** — traffic touches more racks and paths.



2. Demand distribution and Dynamics

Traditional datacenters: stable and structured workloads — e.g., enterprise apps, batch processing, or well-partitioned databases.

- Demand is frequently **concentrated and bursty.**
- Hotspots and bursts are limited and **easier to predict**.

Social network datacenters:

wide-spread,

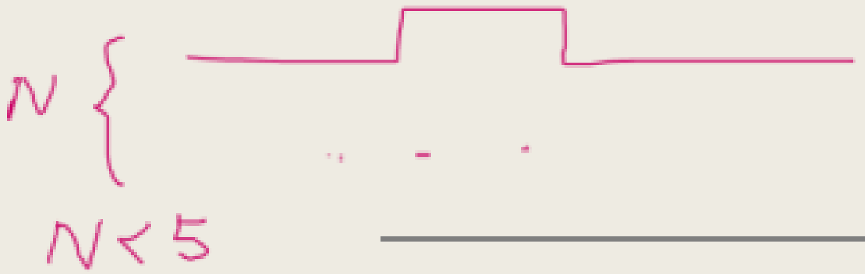
uniform, and stable

Workloads are **highly dynamic and bursty**, driven by unpredictable user activity (viral posts, live events).

The “heavy hitters” (popular posts, trending topics) change rapidly.

This leads to **short-term traffic surges** that require adaptive traffic engineering.

3. Flow Characteristics



<u>Traditional</u> datacenters:

- Often dominated with **on/off behavior**

  e.g., backups, replication, or batch analytics).
- <5 concurrent large transfers at once.

<u>Social network</u> datacenters:

- **Millions of *small, short-lived*** flows

  e.g., requests for likes, comments, thumbnails, and metadata.

- **Continuous** arrival of small packets
  - creates **high concurrency** and **microbursts** that
    - challenge queue management and flow scheduling.

Previously Publications	**New Findings**	**Potential Impacts**
most traffic is rack local	Traffic is **neither** rack local **nor** all-to-all; low utilization (§4)	non-uniform fabrics
Demand is frequently concentrated and bursty	Demand is **wide-spread, uniform, and stable**, with rapidly changing, internally bursty heavy hitters (§5)	Traffic engineering
Bimodal ACK/MTU packet size, on/off behavior; <5 concurrent large flows	**Small** packets (outside of Hadoop), **continuous** arrivals; **many concurrent** flows (§6)	SDN controllers; Circuit/hybrid switching.

Contents

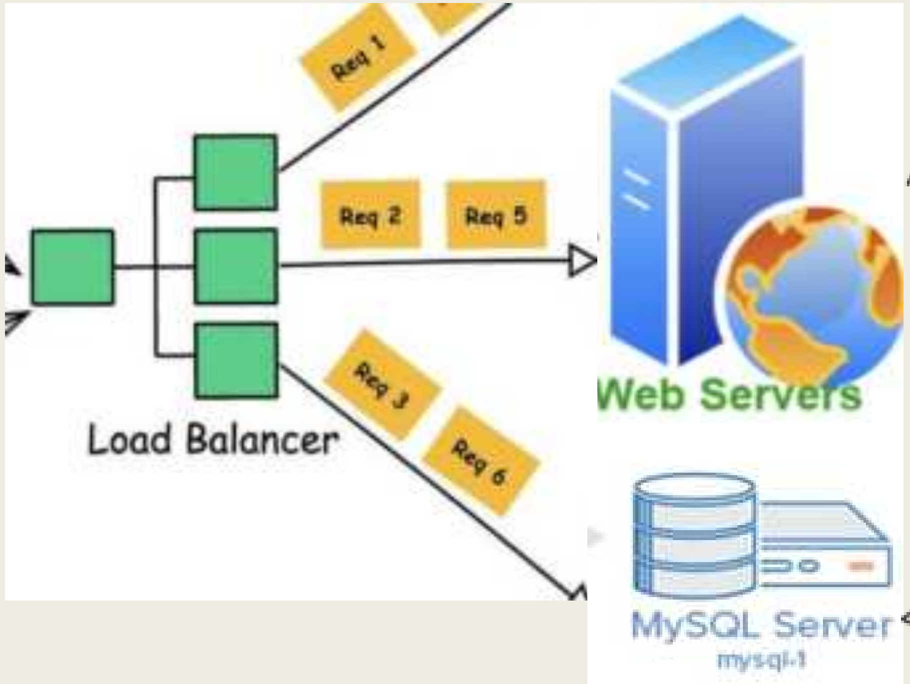
Each Facebook cluster, either has

Cache Servers
Cache Servers
Cache Servers

(A) homogeneous machines

---

hadoop
offline data mining
map-reduce



Cache Servers

Note that each Facebook machine typically has precisely <u>one role</u>

to ease provisioning and management.

(B) heterogeneous machines

FC
“FatCat”
aggregation
switch

CSW
(Cluster SWitch)

Higher-
level

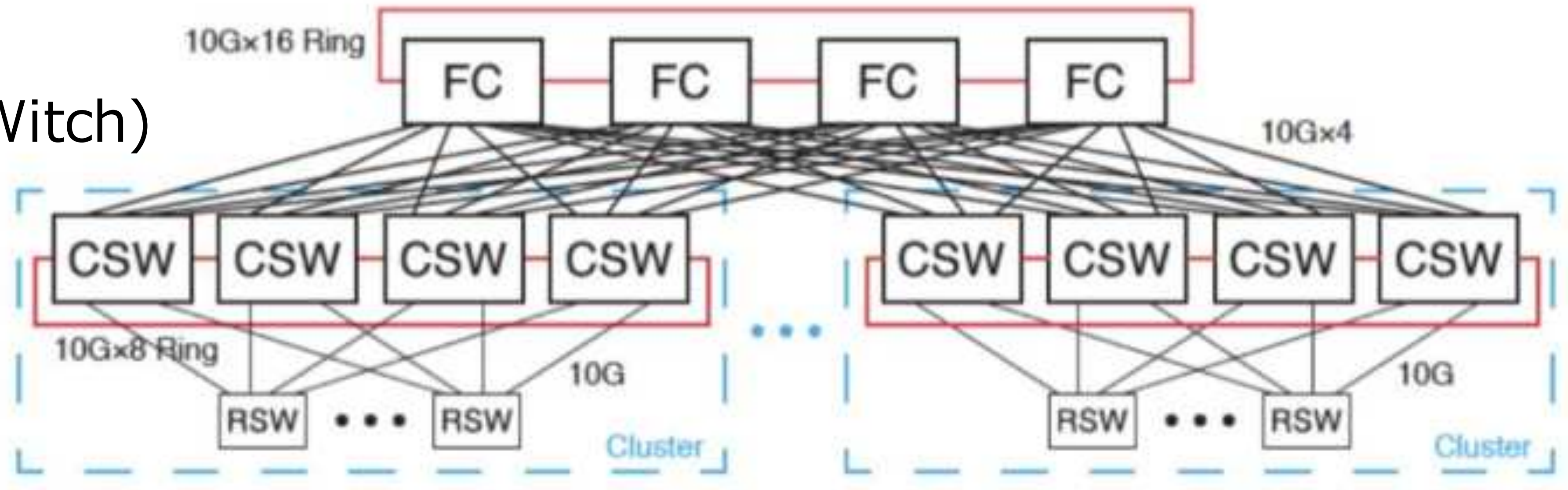


Figure 1: Facebook’s 4-post cluster design [19]

RSW
(Rack SWitch)

44 * 10G downlinks
(4 or 8) * 10G uplinks

Facebook network topology

4-point cluster

Pros

Additional redundancy in 4-post **reduces outages greatly**.

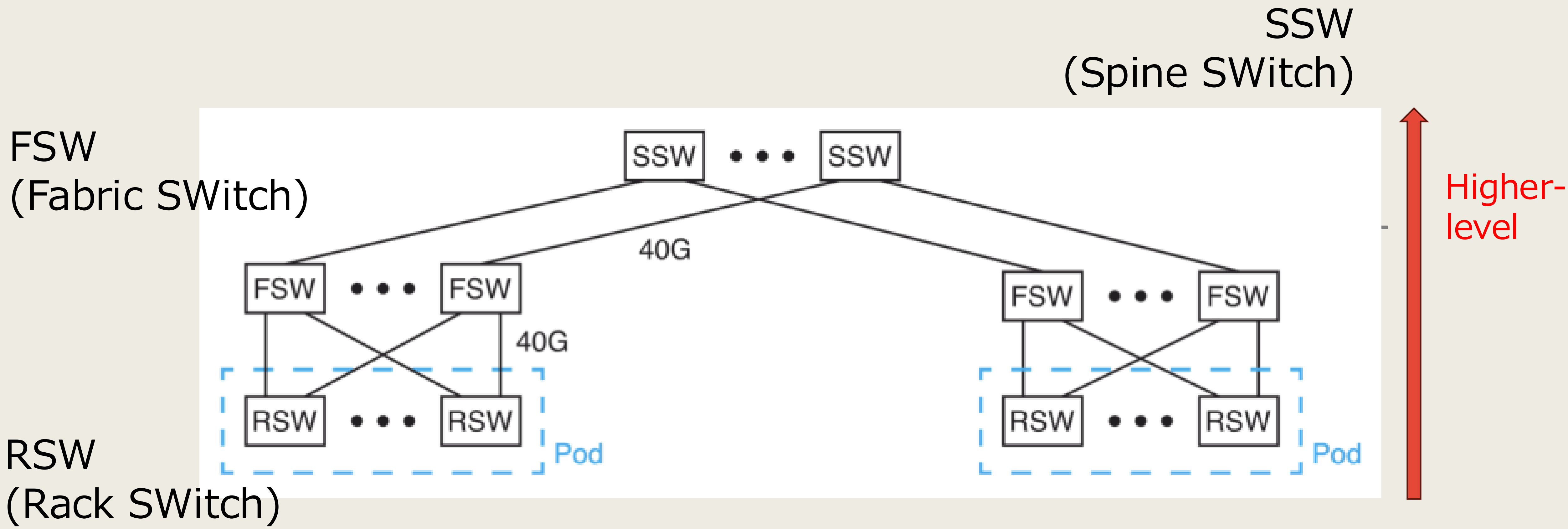
FatCat offers short distance for traffic **between** clusters.

Cons

Uses very **large**, modular CSW and FC switches.

- Failure in one CSW/FC reduces intra-cluster capacity to **75%**.

- Cluster size is limited by the size of the CSW.



advantage: an FSW **failure** reduces intra-pod capacity to **97.9%;** likewise for SSW
disadvantage**: daunting cabling complexity**

Alternative network topology

5-stage Folded Clos

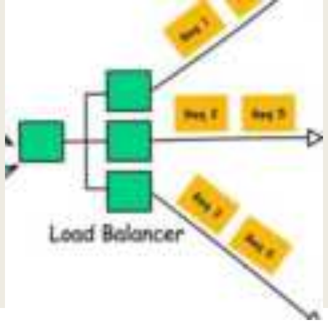
Contents

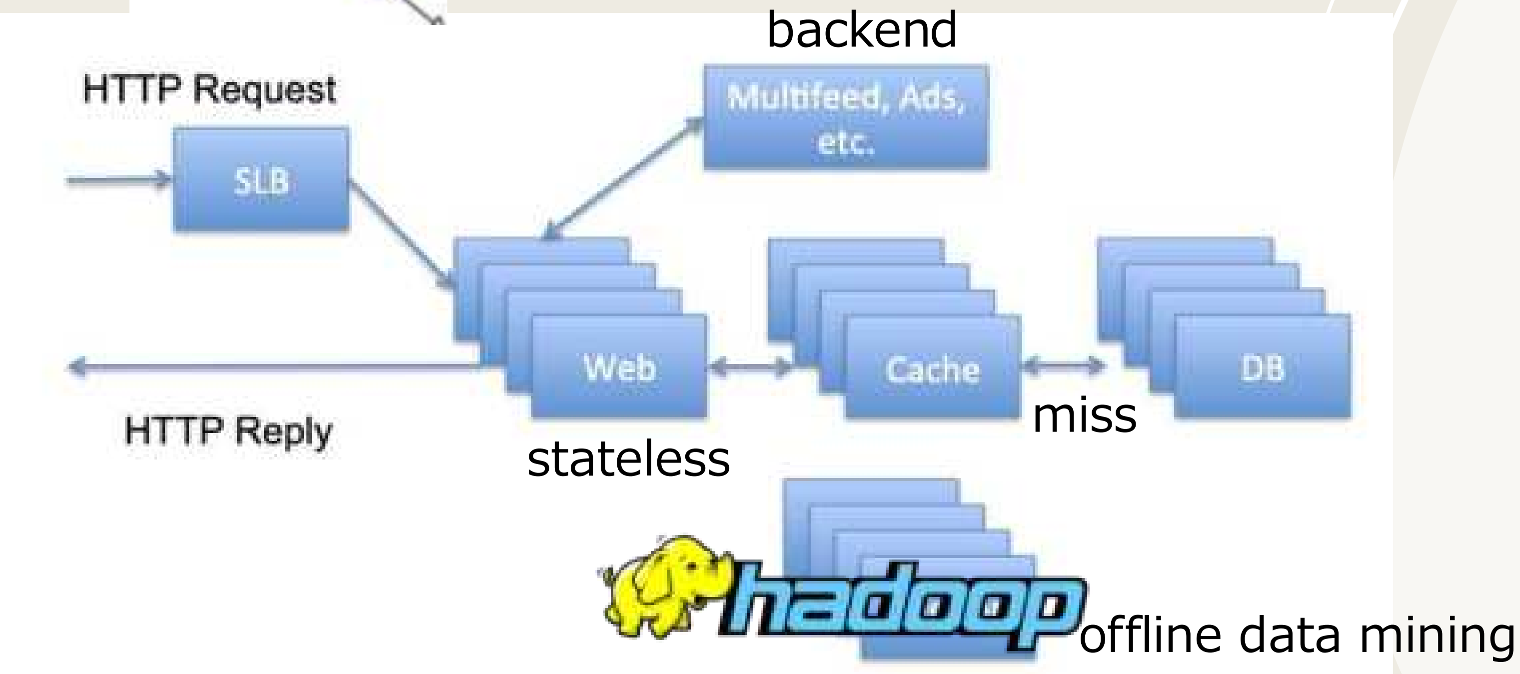
- Traditional vs. Social network's Datacenter network.
- Facebook cluster, network topology.
- **Data collection methods.**
- Experiments, results and implications.
- Main features, concerns.

Services provided

HTTP REQUEST

http://facebook.com/





Type	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 percentage matrix

Data collection sources

collection from HUGE real-world data

Fbflow: **constantly samples** packet headers across Facebook's entire global network.

sampling rate is 1: 30, 000; per-minute granularity.

port mirroring, focuses on a **single** machine (or rack) at a time, allowing us to collect **complete** packet-header traces for a brief period of time at particular locations within a single datacenter.

Fbflow: **constantly samples** packet headers across Facebook’s entire global network.
sampling rate is 1: 30, 000; per-minute granularity.

headers: src, dst IP addresses, port No., and protocol
metadata: machine name and capture time

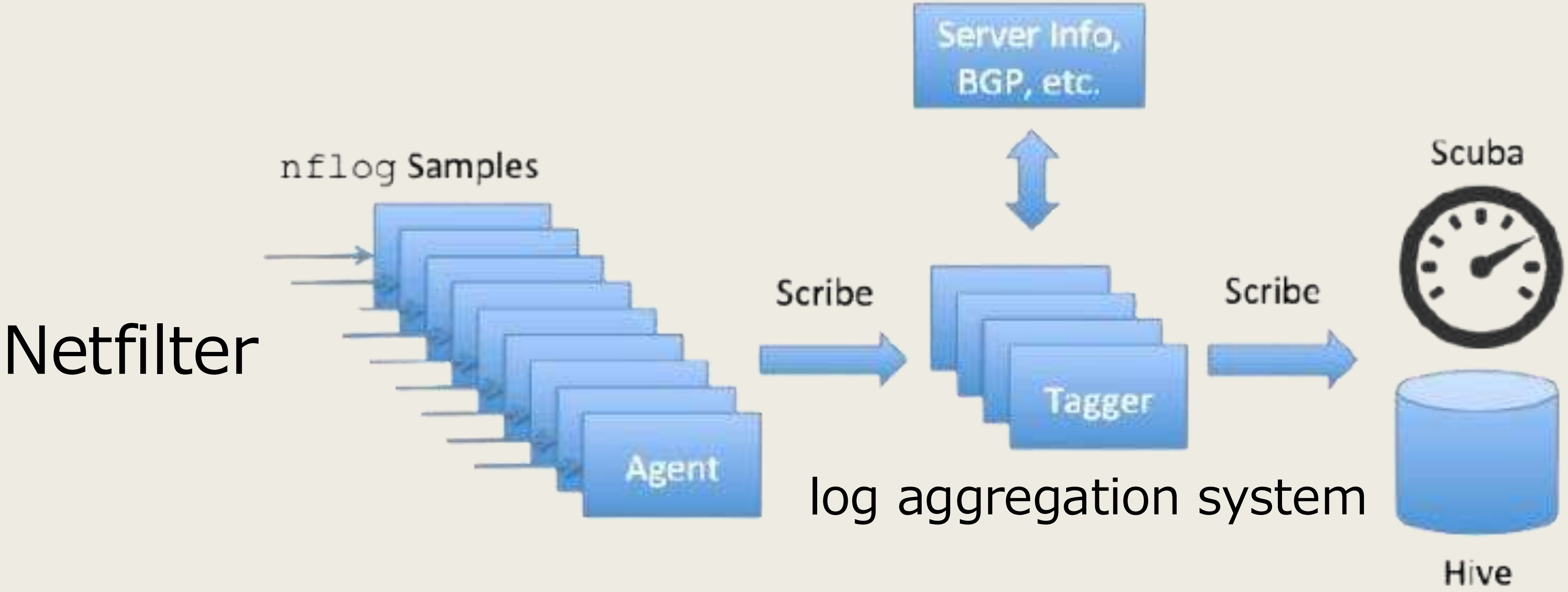


Figure 3: Fbflow architecture

long-term analysis

Contents

1. Utilization and Traffic locality

Host link speed	10 Gbps Ethernet across all hosts
Average access link utilization (host ↔ RSW)	< 1% (1-minute average)
Utilization pattern	Follows diurnal and day-of-week trends (≈ 2× variation, not order-of-magnitude)
Most loaded links (1-minute scale)	99% of links < 10% utilization
Cluster-level variation	**Hadoop** clusters ≈ 5× heavier than Frontend
RSW ↔ CSW link utilization (median)	10–20%
RSW ↔ CSW busiest 5% links utilization	23–46%
Comparison to prior datacenters	Higher utilization due to 1→10 Gbps edge upgrades but only 10→40 Gbps aggregation upgrades
Cluster variance at aggregation	Heaviest clusters ≈ 3× higher than lightest
CSW ↔ FC link utilization	Higher overall, but inter-cluster differences smaller (uplinks provisioned per demand)

1. Traffic locality and Utilization

Hadoop:
diverse

Web:
stable,

within-cluster

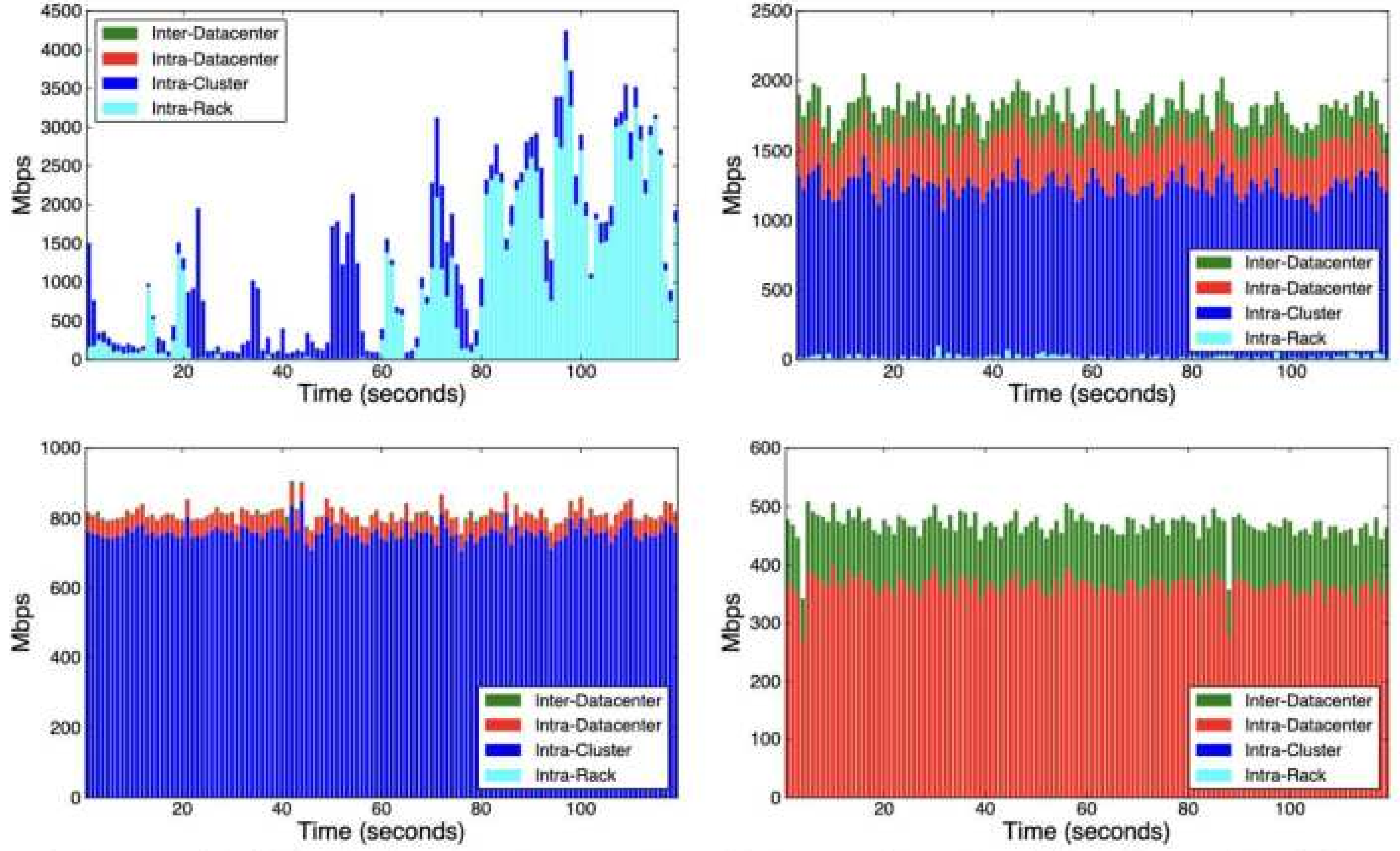


Figure 4: Per-second traffic locality by system type over a two-minute span: Hadoop (top left), Web server (top right), cache follower (bottom left) and leader (bottom right) (Note the differing y axes)

Cache follower:

↔ Web

Cache leader:

coherency,
backing

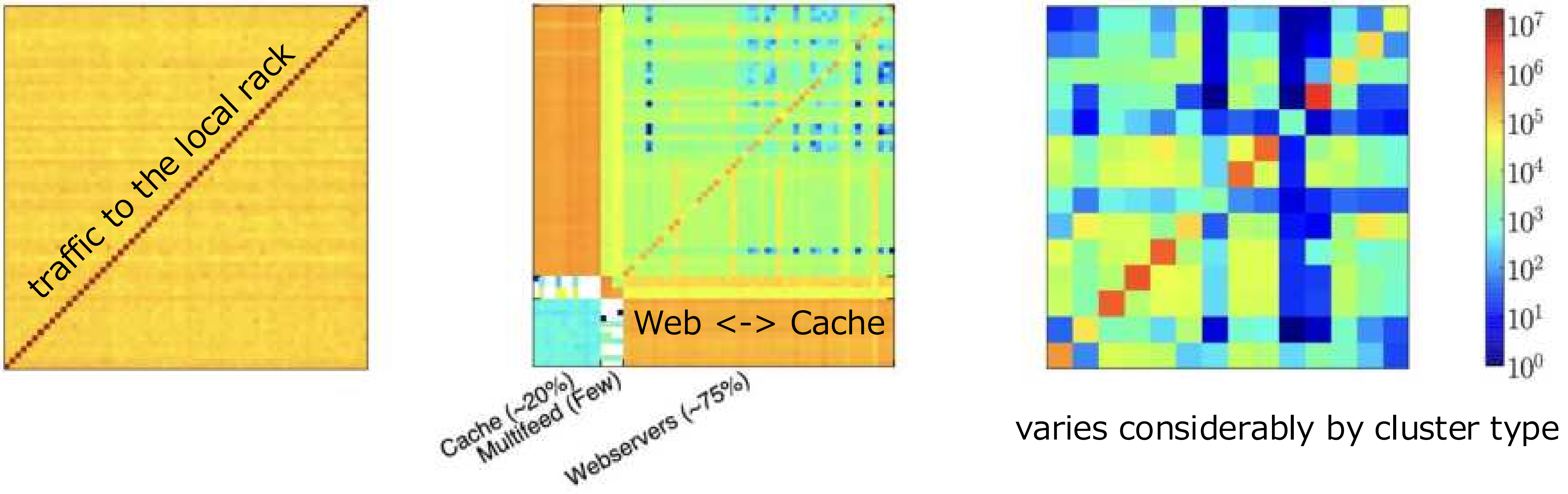
Frontend

Locality	All	Hadoop	FE	Svc.	Cache	DB
Rack	12.9	13.3	2.7	12.1	0.2	0
Cluster	57.5	80.9	81.3	56.3	13.0	30.7
DC	11.9	3.3	7.3	15.7	40.7	34.5
Inter-DC	17.7	2.5	8.6	15.9	16.1	34.8
Percentage		23.7	21.5	18.0	10.2	5.2

Table 3: Different clusters have different localities; last row shows each cluster's contribution to total network traffic

all Facebook's machines during a 24-hour period in January 2015
traffic patterns remain stable day-over-day

1. Traffic locality and Utilization



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

Figure 5: Traffic demand by source (x axis) and destination (y axis). The graphs are each normalized to the lowest demand in that graph type (i.e., the Hadoop and Frontend clusters are normalized to the same value, while the cluster-to-cluster graph is normalized independently).

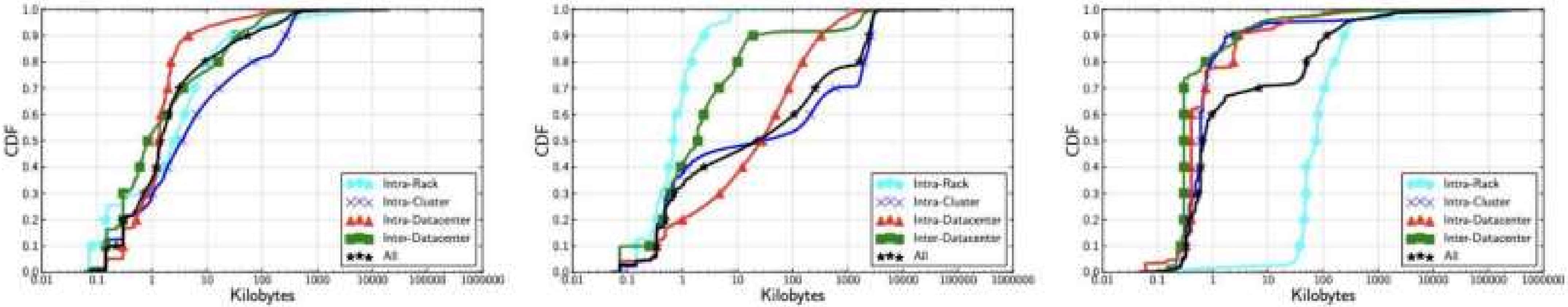
1. Traffic locality and Utilization

Implications:

- **non-uniform** fabric technologies that can deliver higher bandwidth to certain locations than others.

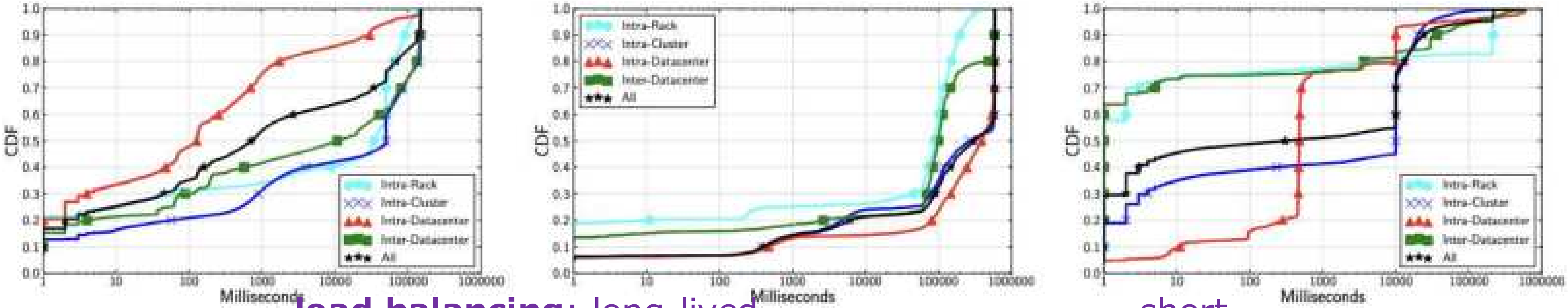
- the lack of significant levels of **intra-rack** locality

2. Traffic engineering: flow



(a) Web servers low throughput (b) Cache follower (c) Hadoop

Figure 6: Flow size distribution, broken down by location of destination



load balancing: long-lived short

(a) Web servers (b) Cache follower (c) Hadoop

Figure 7: Flow duration distribution, broken down by location of destination

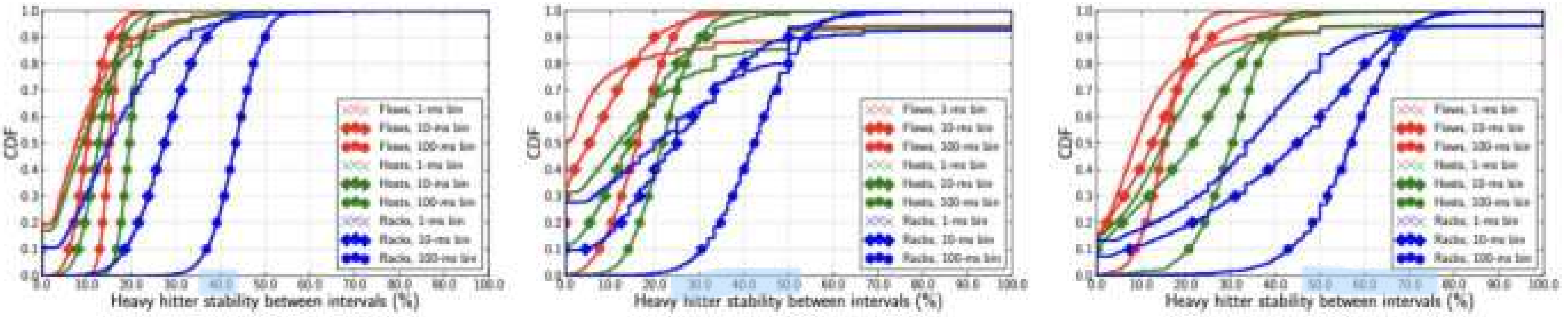
2. Traffic engineering: Heavy hitters

the minimum set of flows/hosts that is responsible for <u>50%</u> of the observed traffic volume over a fixed period.

Intuitively, they signify an **imbalance** that can be acted upon—if they are persistent for enough time, and large enough compared other flows that treating them differently **makes a difference**.

Type		Number			Size (Mbps)		
		p10	p50	p90	p10	p50	p90
Web	f	1	4	15	1.6	3.2	47.3
	h	1	4	14	1.6	3.3	48.1
	r	1	3	9	1.7	4.6	48.9
Cache (f)	f	8	19	35	5.1	9.0	22.5
	h	8	19	33	8.4	9.7	23.6
	r	7	15	23	8.4	14.5	31.0
Cache (l)	f	1	16	48	2.6	3.3	408
	h	1	8	25	3.2	8.1	414
	r	1	7	17	5	12.6	427
Hadoop	f	1	2	3	4.6	12.7	1392
	h	1	2	3	4.6	12.7	1392
	r	1	2	3	4.6	12.7	1392

Table 4: Number and size of heavy hitters in 1-ms intervals for each of flow(f), host(h), and rack(r) levels of aggregation



(a) Cache follower (b) Cache leader (c) Web servers

fraction of the heavy hitters that **remain** in subsequent time

Figure 10: Heavy-hitter stability as a function of aggregation for 1/10/100-ms time windows

2. Traffic engineering

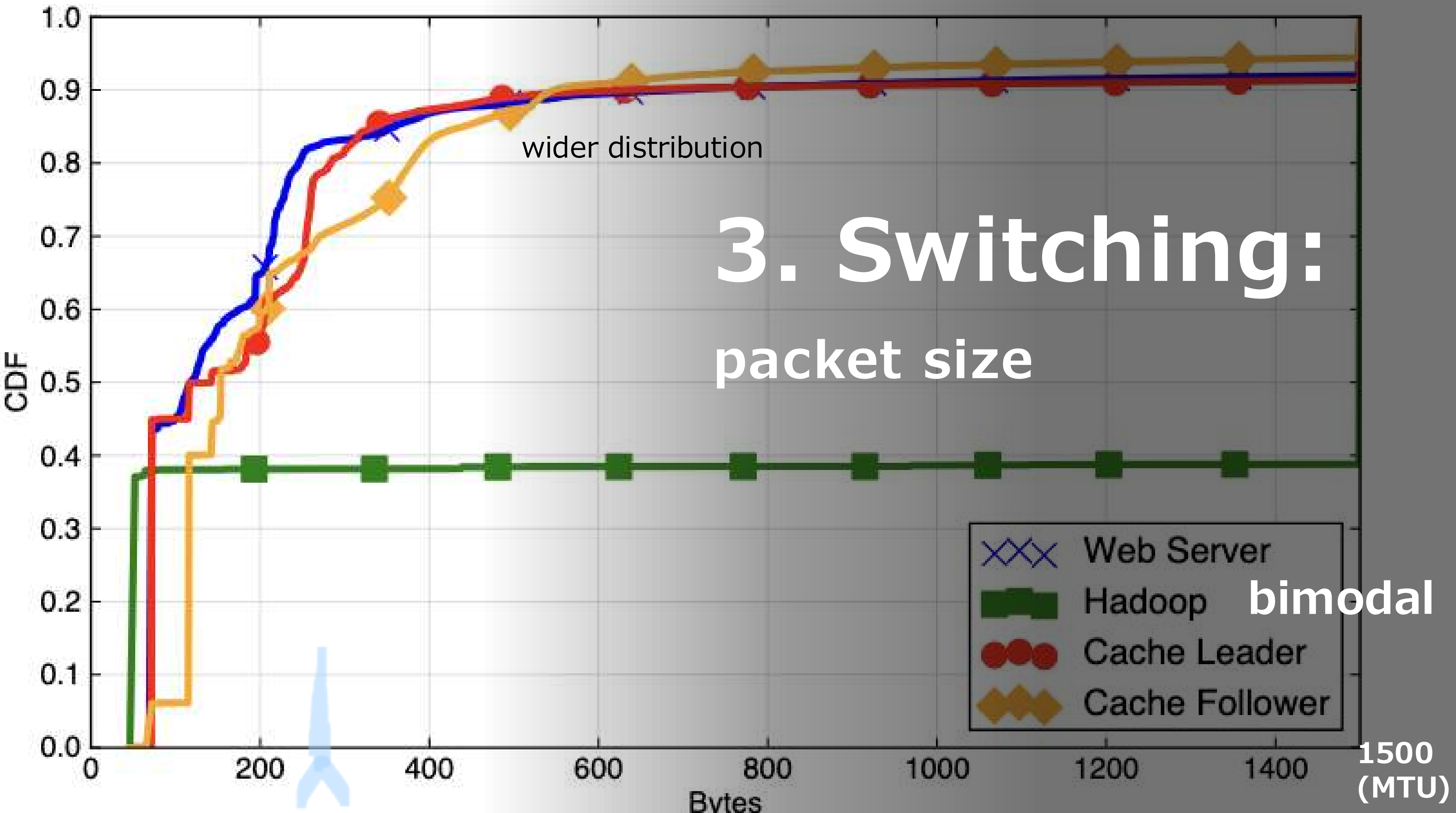
Implications:

- services use application-level load balancing to great effect, however, leaving **limited** room for in-network approaches.

- identifying **heavy hitters** and then treating them specially,
  - *via provisioning a circuit, moving to a lightly loaded path, alternate buffering strategies, etc*.
- yet it’s very challenging to identify them all.

3. Switching:

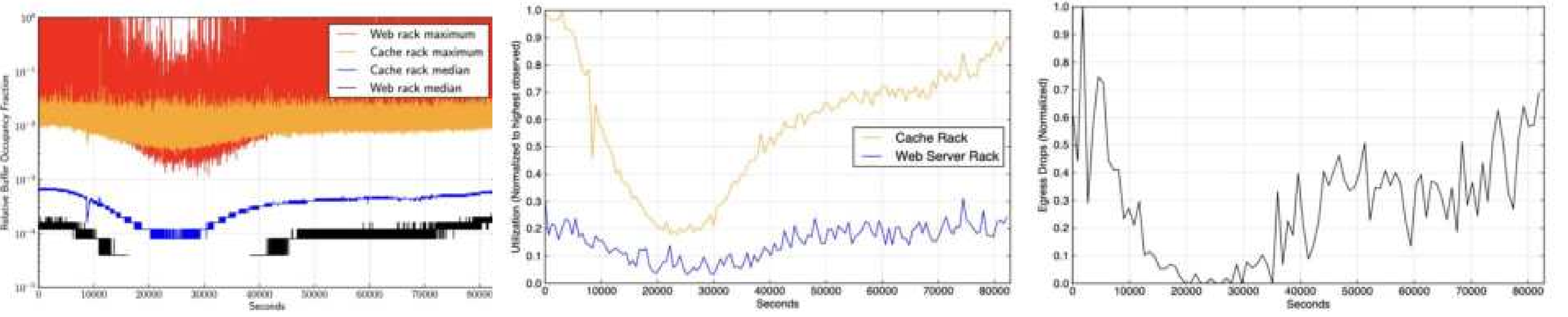
packet size



3. Switching

Arrival pattern: a lack of on/off traffic,
+ higher flow intensity, and bursty individual flows

=> increase in **buffer** utilization and overruns.



(a) Normalized buffer occupancy, 10-microsecond resolution (b) Link utilization, 10-minute average (c) Web rack egress drops, 15-minute average

1 %

More problems for future traffic

3. Switching

Implications:

- combine **circuit switching** for high throughput with packet switching (flexible, fine-grained).

- **SDN controllers** to manage routing, flow scheduling, and load balancing.
  - dynamically control flow placement to improve utilization and reduce congestion.

Contents

- Traditional vs. Social network's Datacenter network.
- Facebook cluster, network topology.
- Data collection methods.
- Experiments, results and implications.
- **Main features, concerns.**

Features, Concerns

Limited by

1. **datacenter** network,
2. **social network** applications,
3. Network **topology** (4 point),
4. data collection **sampling** frequency, analysis **accuracy**, and many more.

Thank you!

Peter Hu, zh369,

Magdalene College.