Cloud Computing: traffic, topology, network virtualization

Guillaume Urvoy-Keller

December 12, 2017

Source documents

- Robert Birke, Mathias Björkqvist, Cyriel Minkenberg, Martin Schmatz, Lydia Y. Chen: When Virtual Meets Physical at the Edge: A Field Study on Datacenters' Virtual Traffic. SIGMETRICS 2015: 403-415
- Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, Alex C. Snoeren: Inside the Social Network's (Datacenter) Network. SIGCOMM 2015: 123-137
- Ashkan Aghdai, Fan Zhang, Nadun Dasanayake, Kang Xi, H. Jonathan Chao: Traffic measurement and analysis in an organic enterprise data center. HPSR 2013: 49-55

Source documents

- Keqiang He, Alexis Fisher, Liang Wang, Aaron Gember, Aditya Akella, Thomas Ristenpart: Next stop, the cloud: understanding modern web service deployment in EC2 and azure. Internet Measurement Conference 2013: 177-190
- Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, Amin Vahdat: Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. SIGCOMM 2015: 183-197

Outline

1. Bird's eye view
2. IBM
3. Facebook
4. EDC
5. EC2
6. Google

What are we talking about?

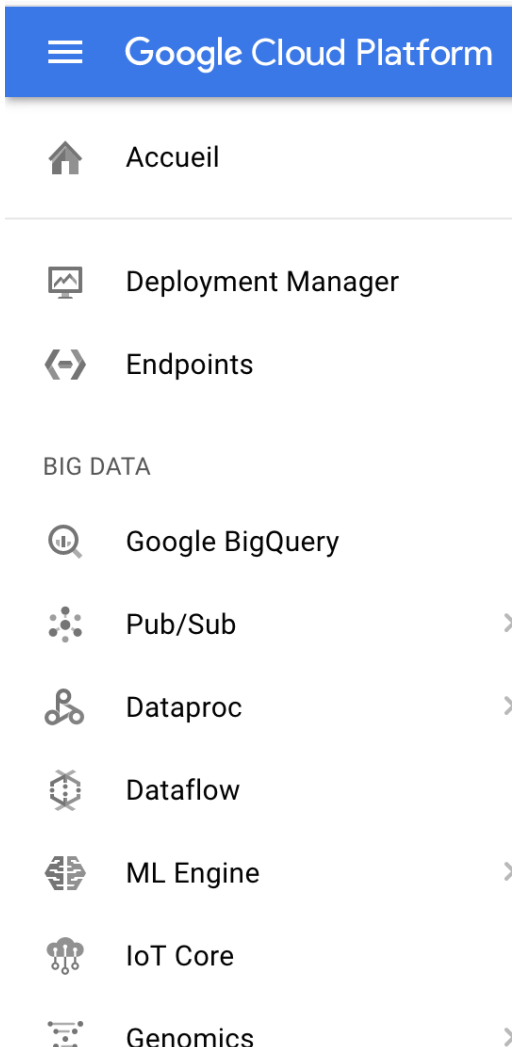
Key questions

- Which application is run?
- Is it directly on hardware or is there virtual machines?
- Is the infrastructure shared or not?

Which applications?

- Legacy applications hosted in the cloud
  - A company relocating part of its servers in the cloud
  - Relieved from the burden of hardware maintenance
- Big data applications
  - Can be run directly on hardware with possibly lightweight virtualization
    Typical examples: Google Web Search Engine, clusters running Spark (batch processing of static data) or Storm (streaming (processing of live data, a.k.a unbounded streams))
  - A company using a big data service (storage + computing software) from a cloud provider

Big Data Service offered by Google



Longitudinal study of IBM infrastrure

- Known as the original virtualization company
- Invented virtualization back in the 1960s to set up partitions within the mainframe
- February 1990, IBM released the RS/6000 (a.k.a. POWER processor) based servers.
  Combined with mainframe $\Rightarrow$ mission-critical virtualization.
- PowerVM hypervisors to manage the global system
- **Live migration was introduced with POWER6 in May 2007.**

Longitudinal study of IBM infrastrure

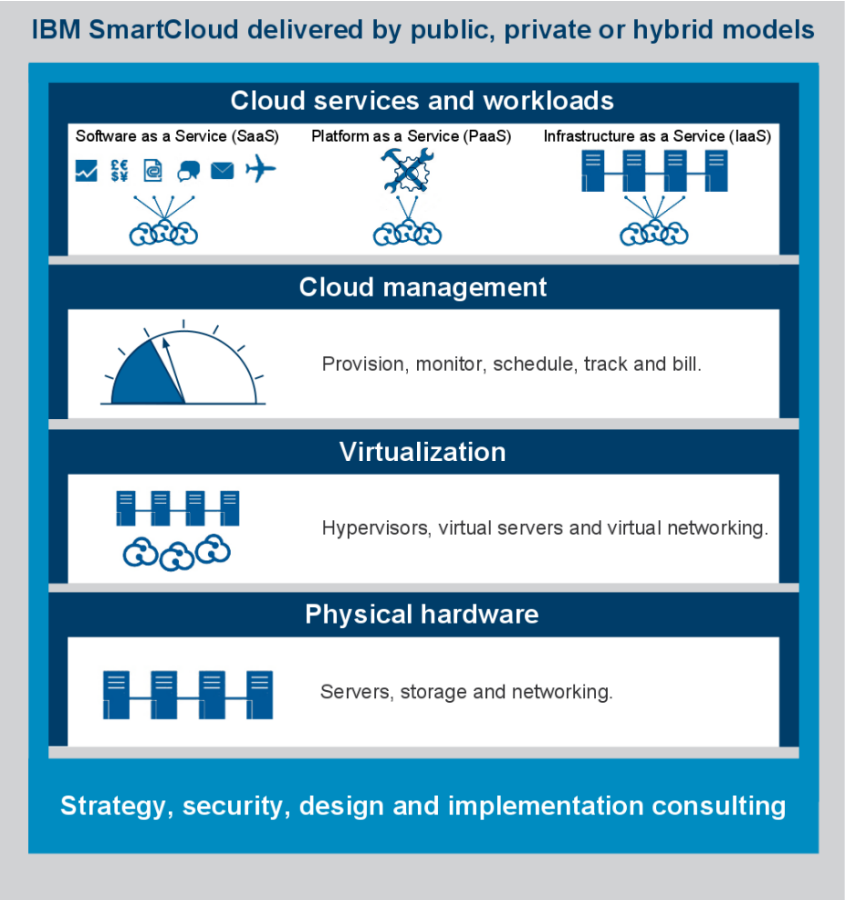


Figure: By Sydney Powell - Sydney Powell, FAL,

Longitudinal study of IBM infrastrure

IBM offers:

- Public cloud service, i.e. shared or dedicated infra maintained by IBM
- Private cloud service

What is private cloud?

A private cloud allows companies to customize their environments according to their own unique needs and based on their own security requirements. Private cloud leverages the benefits of public cloud, including rapid deployment, scalability, ease of use and elasticity—but can also offer additional capabilities such as greater control, increased performance, predictable cost, tighter security and flexible management options.

IBM Research Paper

When Virtual Meets Physical at the Edge: A Field Study on Datacenters' Virtual Traffic

Robert Birke
IBM Research Zurich Lab
Saümerstrasse 4
Rüschlikon, Switzerland
bir@zurich.ibm.com

Mathias Björkqvist
IBM Research Zurich Lab
Saümerstrasse 4
Rüschlikon, Switzerland
mbj@zurich.ibm.com

Cyriel Minkenberg
IBM Research Zurich Lab
Saümerstrasse 4
Rüschlikon, Switzerland
sil@zurich.ibm.com

Martin Schmatz
IBM Research Zurich Lab
Saümerstrasse 4
Rüschlikon, Switzerland
mrt@zurich.ibm.com

Lydia Y. Chen
IBM Research Zurich Lab
Saümerstrasse 4
Rüschlikon, Switzerland
yic@zurich.ibm.com

Dataset

- 90 0000 Virtual Machines
- 8000+ (Physical) Hosts (aka box) with :
  - Virtualization solution like VMware and IBM techno
  - Able to switch on/off and migrate VMs for consolidation or maintenance
  - Geographical coordinates available for boxes (and their VMs)
- 300+ corporates
- April 2013-April 2014

VM Consolidation

Table 1: Statistics of VM consolidation levels.

Percentiles	25^{th}	50^{th}	75^{th}	90^{th}	99^{th}
VMs per box [#]	4	8	16	35	60

On average:

- Average at 11 VMs per box
- A box has on average 15 CPUs and 60 GB of RAM
- A VM has 2 vCPU
  One vCPU = in general half a core with hyperthreading
  (microprocessor level technique to share between programs)
- ... and 4.8 GB of RAM

VM migration

Summary over one moth (April 2013)

- 56% of boxes experienced a VM migration
- 31% of VMs
- Migration per box is twiced the one of VMs because a migration of VM gives two migrations (in for the first box and out for the second box)

Definition of migration:

- if it takes less than 15 min, it is called live
- otherwise cold (stop VM, migrate VM and restart)

Limitation

- Throughput values at the hour time scale $\Rightarrow$ miss some short time load spikes

Virtual Traffic Demand

- Each box (and also VMs) has multiple NICs
- A NIC is inactive if less than 10pps over an entire day
  - A threshold of 0 is inefficient as there always at least some broadcast traffic like ARP

Virtual Traffic Demand

Table 2: Network resource and traffic of boxes and VMs: no. of NICs, no. of inactive NICs, and daily traffic volume [GB].

	Boxes			VMs		
	Mean	50^{th}	99^{th}	Mean	50^{th}	99^{th}
NICs [#]	8.5	7	22	2.3	1	19
Inact. NICs [#]	4.2	4	16	1.0	0	14
Volume [GB]	259	70	2700	18	2	305

- Presence of long tail
- Volume per host is $x14$ the one of VM (compatible with 11 VM per host)

Volume increase over one year - from 2013 to 2014

Table 3: Yearly traffic volume growth [%].

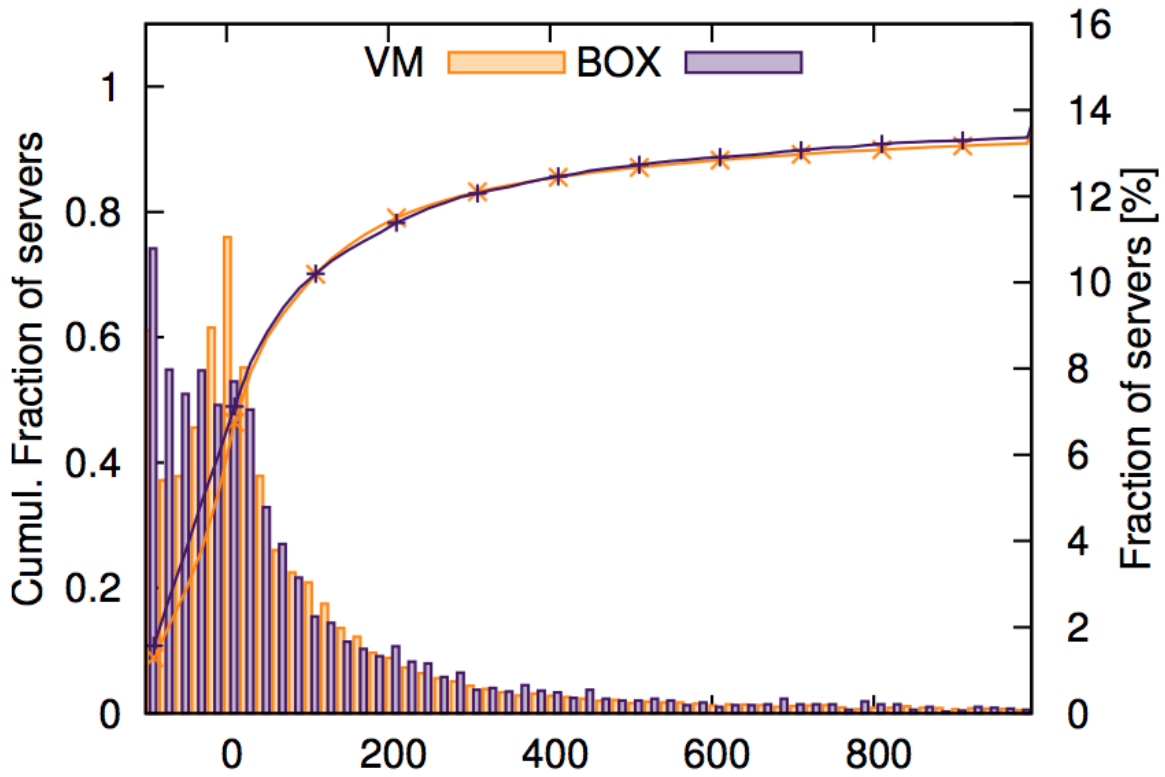
Boxes			VMs		
Mean	Median	99^{th}	Mean	Median	99^{th}
7674	22	20081	24425	27	76174

- A heavy tail
- Average over one month

Volume increase over one year - from 2013 to 2014

CDF and PDF

Q: how would you do this computation ?



Migration

- On-line vs offline
  - def : less or more than 15 min in their dataset
  - Technological challenge is high : in the first case, you drop the memory in a file while in the second, you transfer memory blocks until change rate is low in origin VM
- Far vs Near $\sim$ same datacenter or different one

Table 4: Distribution and migration volume of four classes of migration: combinations of {far, near} $\times$ {on-line, off-line}.

	On-line		Off-line	
	Near	Far	Near	Far
Distribution [%]	50.75	2.83	44.71	1.72
Migration volume [MB]	724	454	575	458

Migration

- They observed a (not fully convincing) trend that there is a 2x factor between memory size and what is sent on the network during migration.

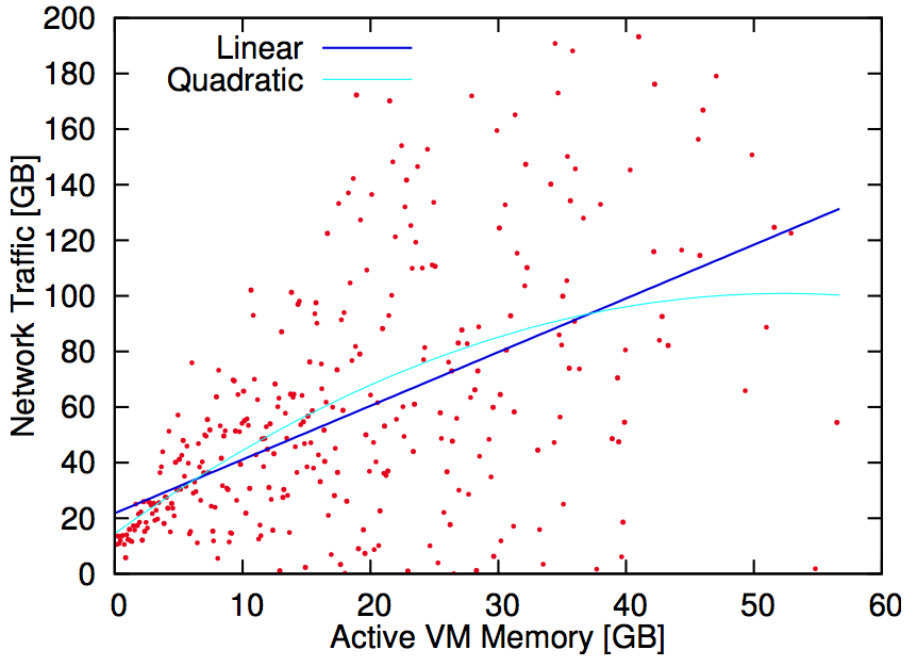


Figure 5: Estimating network overhead per migrating volume

Migration

Post-copy procedure :

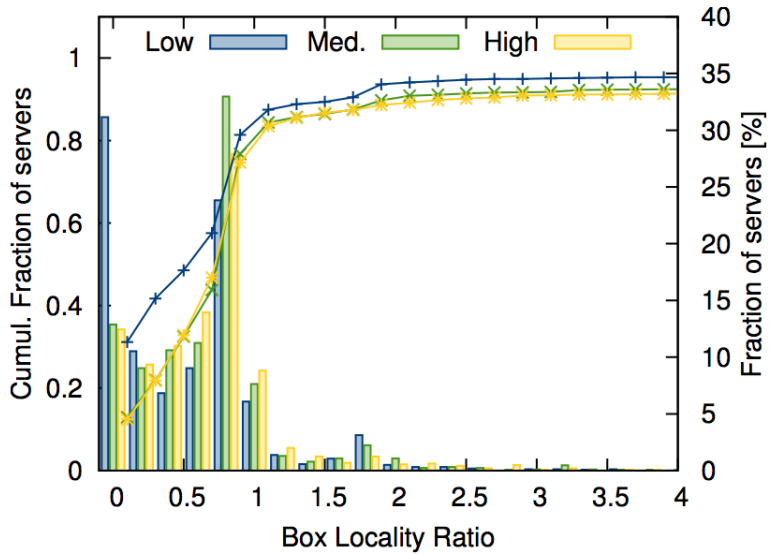
1. Make an initial copy of memory of the VM in the new box
2. Every x milliseconds, re-copy the dirty pages, ie. pages that were changed since last pool
3. Stop when rate of dirty pages is low enough and stop VM in initial box

Traffic Locality

- Def: Ratio of traffic generated by VMs of a box to traffic generated by the box (to the outside world)
- Ratio=1 : traffic is purely to outside VMs
- Ratio $\leq$ 1 when for instance there is migration of VM
- Ratio $\geq$ 1: more local traffic

Traffic locality

- Consolidation levels : Low: 1-5 VMs, Medium: 6-15 VMs and High $>$ 15 VMs
- Median at 0.8 $\Rightarrow$ 12% of traffic due to hypervisor
- Little local traffic in general $\Rightarrow$ room for traffic aware VM scheduler



Facebook

Inside the Social Network's (Datacenter) Network

Arjun Roy, Hongyi Zeng†, Jasmeet Bagga†, George Porter, and Alex C. Snoeren

Department of Computer Science and Engineering
University of California, San Diego

†Facebook, Inc.

Preliminary remarks made by authors

- " These traces are dominated by traffic gen- erated as part of a major Web search service, which, while certainly significant, may differ from the demands of other major cloud services. " :-)
- Data center topology often try to maximize bisectional bandwidth
  - Bissectional bandwidth: maximum amount of bandwidth in the data center measured by bisecting the graph of the data center at any given point – *Tony Li, Internet construction crew, emeritus*
  - Assumes a worst case scenario with all to all traffic
  - ... maybe too expensive when there is traffic locality

Summary of findings

Finding	Previously published data	Potential impacts
Traffic is neither rack local nor all-to-all; low utilization (§4)	50–80% of traffic is rack local [12, 17]	Datacenter fabrics [4, 36, 21]
Demand is wide-spread, uniform, and stable, with rapidly changing, internally bursty heavy hitters (§5)	Demand is frequently concentrated and bursty [12, 13, 14]	Traffic engineering [5, 14, 25, 39]
Small packets (outside of Hadoop), continuous arrivals; many concurrent flows (§6)	Bimodal ACK/MTU packet size, on/off behavior [12]; <5 concurrent large flows [8]	SDN controllers [1, 22, 28, 32, 34]; Circuit/hybrid switching [7, 20, 30, 39]

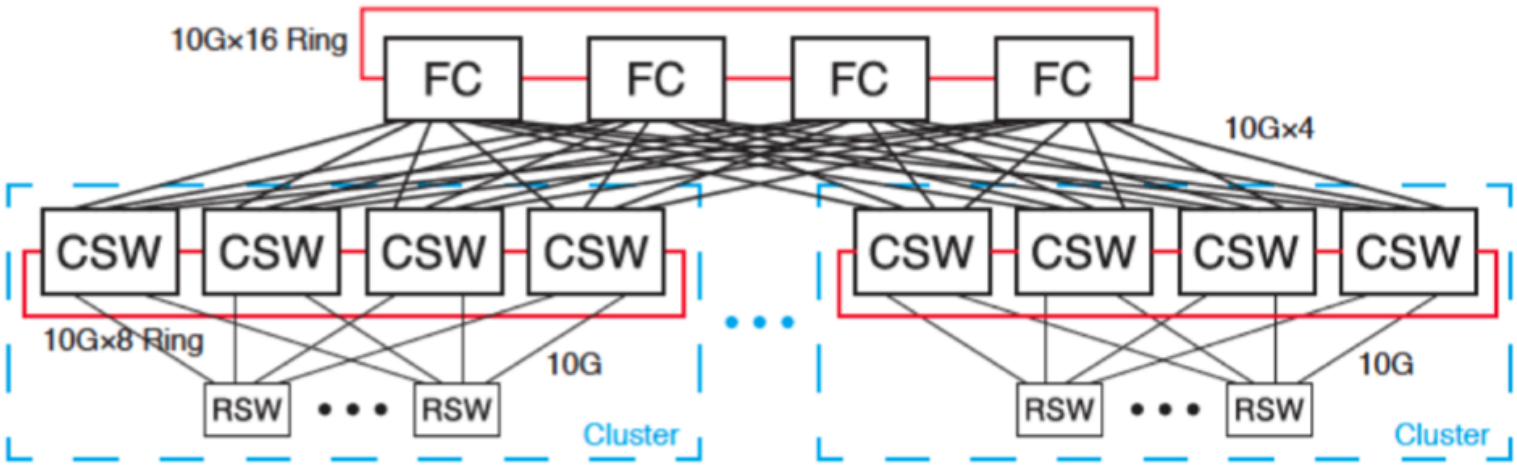
Table 1: Each of our major findings differs from previously published characterizations of datacenter traffic. Many systems incorporate one or more of the previously published features as design assumptions.

FB Datacenters

- A site consists of several datacenter buildings (one building=one DC)
- Each DC hosts several clusters
- Private backbone network to interconnect sites

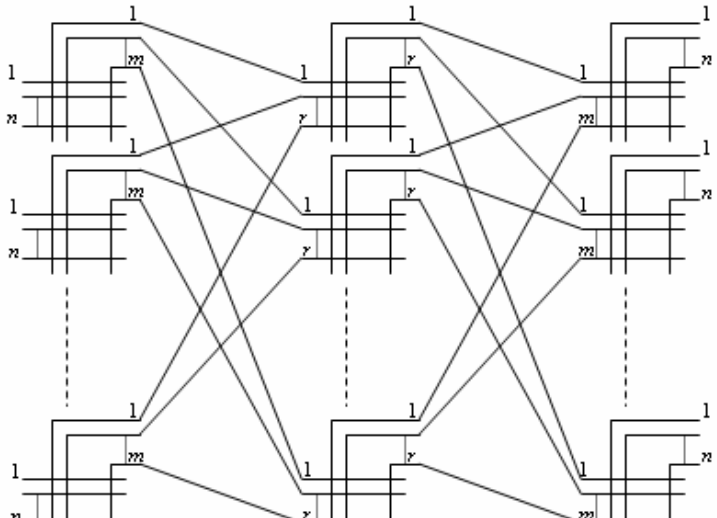
DC topology

- RSW: Rack Switch, CSW: Cluster Switch, FC: Fat Cat
- Clos topology



Clos Topology

- Initially proposed by Charles Clos in 1952 to overcome the limits of single crossbars (electro-mechanical telephone switches).
- Initially with three stages and r, m and n crossbars respecitvely of different sizes : n$\times$m, r$\times$ r and m$\times$n
- Property: if m $\geq$ 2n$-$1, the Clos network is strict-sense nonblocking: an unused input on an ingress switch can always be connected to an unused output on an egress switch, re-arranging existing calls. (wikipedia)



Clos Network

- Initially proposed for telecommunication networks then deprecated by technological advances
- Come-back in the 1990s to build Ethernet internal architecture
- Come back in the 2000s to connect switches in data center and benefit from muli-pathing

Clos, Leaf-Spine

- Leaf-Spine is a folded Clos
- Desired properties:
  - Same latency between any two servers : good for “East-West" traffic
  - Easily extensible architecture: simply add a spine
  - Resilience to failure

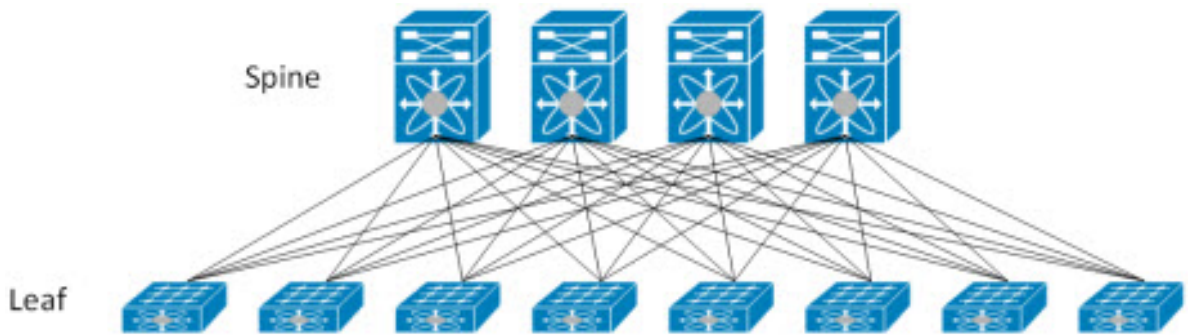


Figure: https://blog.westmonroepartners.com/a-beginners-guide-to-understanding-the-leaf-spine-network-topology/

How to benefit from multiple paths

- Traditional layer 2 technology relies on Spanning Tree that prevents multipaths
- Alternatives:
  - Use routing with Equal-Cost Multipathing (ECMP)
  - Use layer 2 technology (you preserve VLANs ie a flat Ethernet network) using Transparent Interconnection of Lots of Links (TRILL) or Software Defined Networking (SDN)
  - Ref on Trill by Radia Perlman (Spanning Tree mother) : `https://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-53/143-trill.html`

Tat tree topology

- proposed by Charles E. Leiserson in 1985
- for any switch, the number of links going down to its siblings is equal to the number of links going up to its parent in the upper level.

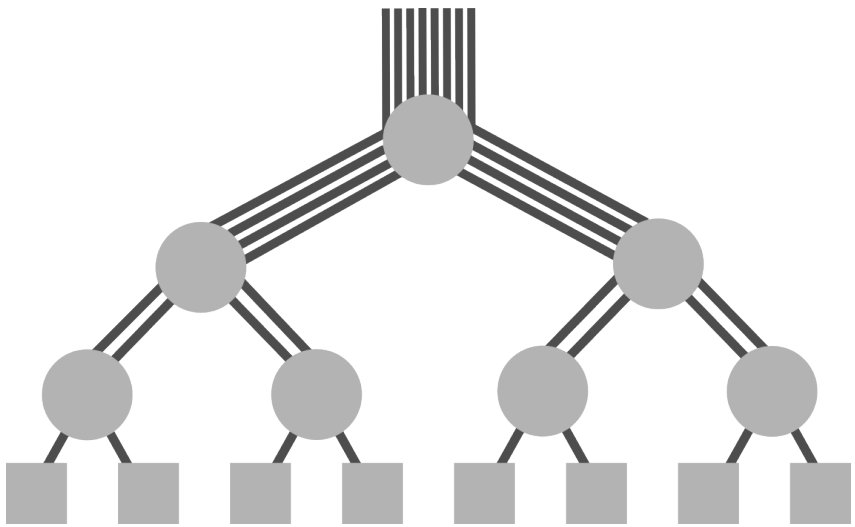


Figure: http://clusterdesign.org/fat-trees/

Fat tree topology

- A special case of Clos network
- Organization into pods (of Ethernet switches)

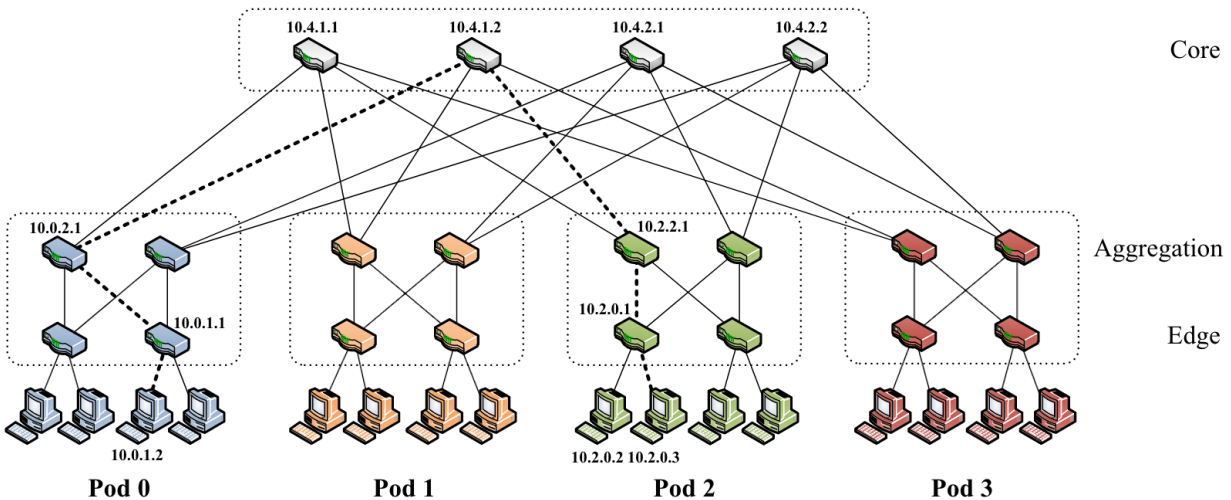
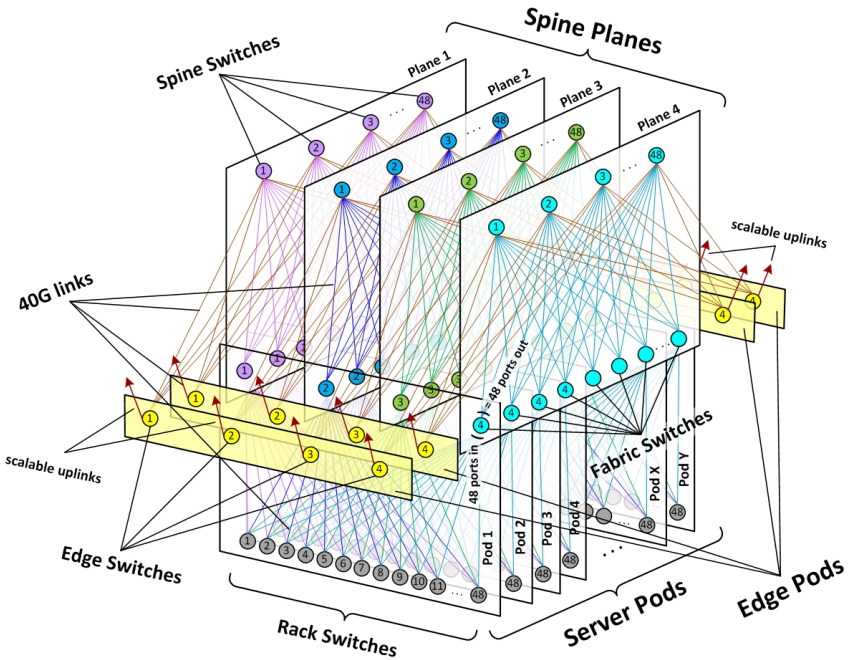


Figure: `http://ccr.sigcomm.org/online/files/p63-alfares.pdf`

Let's get back to Facebook

- Face huge increase of inter cluster traffic while users (you!) to server traffic increases at a modest rate
- Need a more modular architecture
- Looks like this (see `https://code.facebook.com/posts/360346274145943`)



Services

- SLB: Layer 4 load balancer
- Cache-l: leader and Cache-f: follower
- Hadoop tasks in background, not involved in servicing users (different from Google We Services)

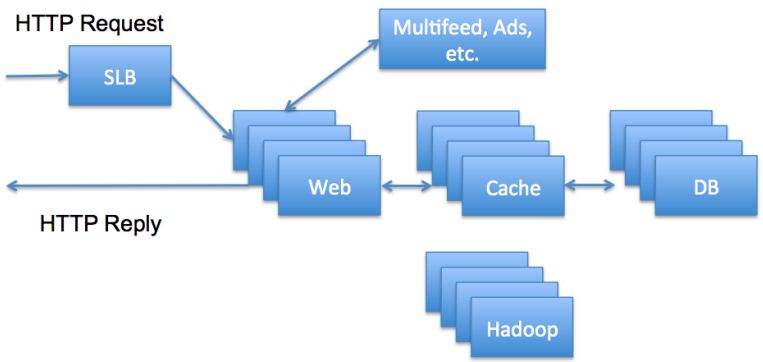


Figure 2: How an HTTP request is served

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

Measurement set-up

- Fbflow: Sampling of packets at 1:30000 using nflog feature of Netfilter (Linux Kernel packet manipulation module)
  - Sampled packet sent to Tagger servers for appending additional info: rack number, server number, etc.
  - Then to Hive : data warehouse infrastructure built on top of Hadoop
- Packet mirroring at RSW (Top of the Rack Switch).
  - Limited to a few minutes of trace capture

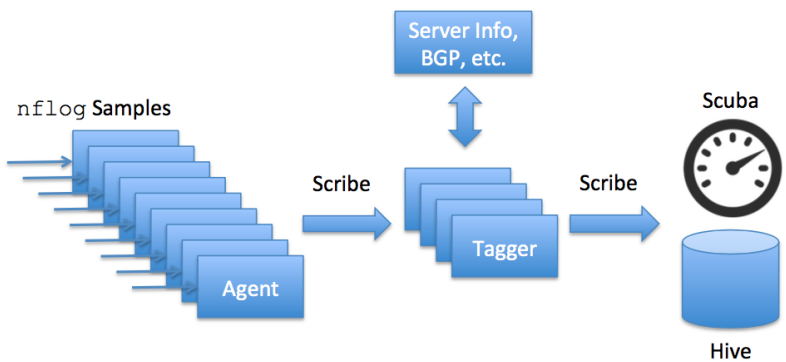


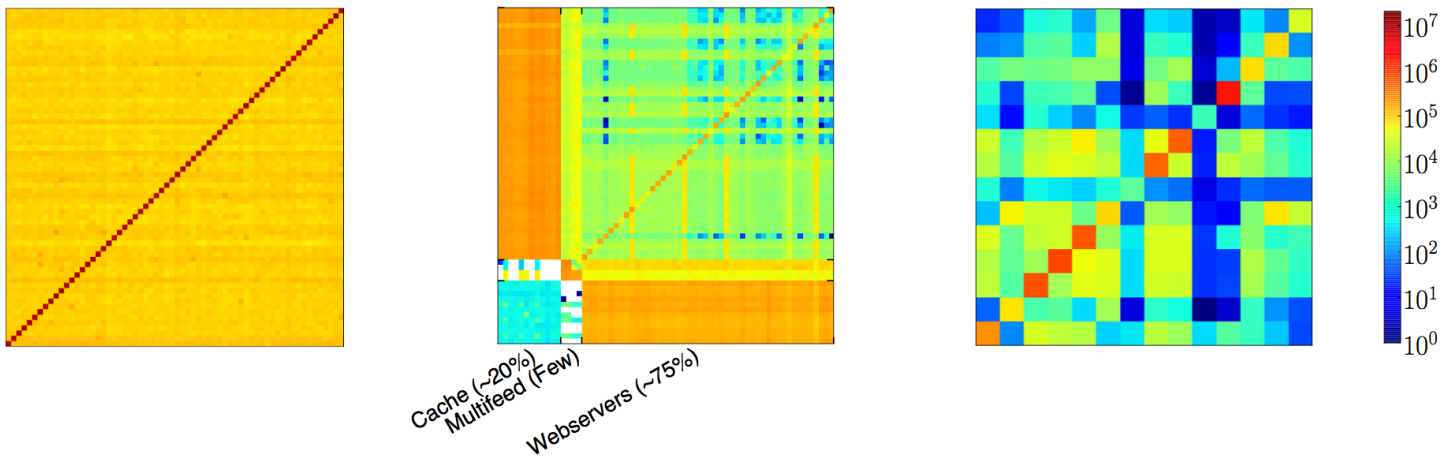
Figure 3: Fbflow architecture

Links Utilization

- As of paper writing time (2014), FB transited to 10 Gb/s for RSW and 10 to 40 Gb/s for CSW
- Consequently, quite low utilizations:
  - RSW: average 1% over 1 min interval and 99% of links
  - CSW: median between 10–20% across clusters. Busiest 5% of the links seeing 23–46% utilization.
- Diurnal traffic pattern with x2 load variation
  - In typical companies, this figure is much more larger (x10 at least)

Traffic locality and stability

Reminder: a datacenter consists of clusters that consist of racks



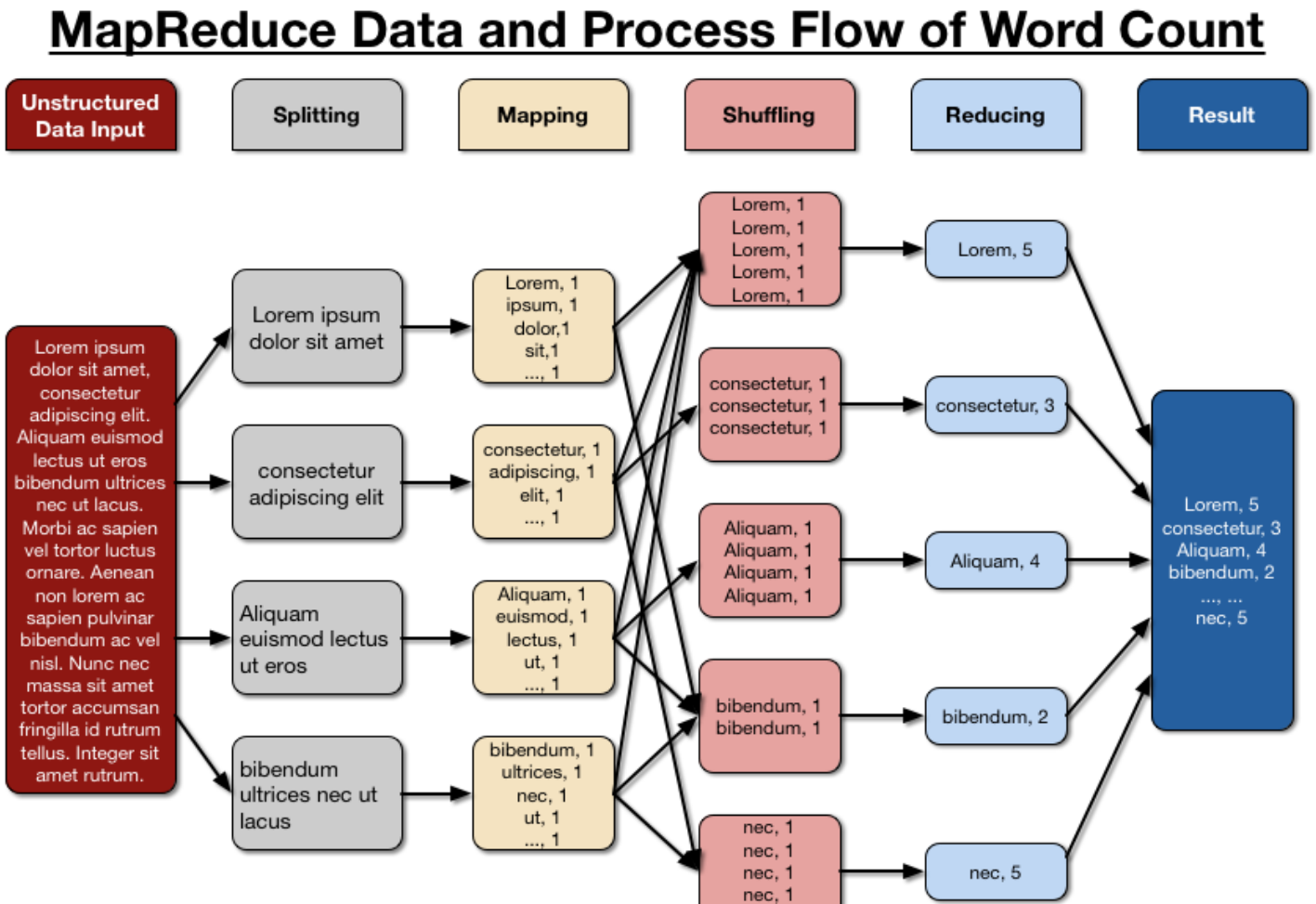
(a) Rack-to-rack, Hadoop cluster (b) Rack-to-rack, Frontend cluster (c) Cluster-to-cluster

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).

Traffic locality and stability

- Hadoop is less stable
  - directly related to the phases – see next slide
  - but significantly local (rack level)
- Web and f-cache quite local
- leader cache maintains “coherency" (user level?) and work at datacenter

Hadoop 101


MapReduce Data and Process Flow of Word Count
Unstructured Data Input
Splitting
Mapping
Shuffling
Reducing
Result
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam euismod lectus ut eros bibendum ultrices nec ut lacus. Morbi ac sapien vel tortor luctus ornare. Aenean non lorem ac sapien pulvinar bibendum ac vel nisl. Nunc nec massa sit amet tortor accumsan fringilla id rutrum tellus. Integer sit amet rutrum.
Lorem ipsum dolor sit amet
consectetur adipiscing elit
Aliquam euismod lectus ut eros
bibendum ultrices nec ut lacus
Lorem, 1 ipsum, 1 dolor,1 sit,1 ..., 1
consectetur, 1 adipiscing, 1 elit, 1 ..., 1
Aliquam, 1 euismod, 1 lectus, 1 ut, 1 ..., 1
bibendum, 1 ultrices, 1 nec, 1 ut, 1 ..., 1
Lorem, 1 Lorem, 1 Lorem, 1 Lorem, 1 Lorem, 1
consectetur, 1 consectetur, 1 consectetur, 1
Aliquam, 1 Aliquam, 1 Aliquam, 1 Aliquam, 1
bibendum, 1 bibendum, 1
nec, 1 nec, 1 nec, 1 nec, 1
Lorem, 5
consectetur, 3
Aliquam, 4
bibendum, 2
nec, 5
Lorem, 5 consectetur, 3 Aliquam, 4 bibendum, 2 ..., ... nec, 5

Traffic Matrix

Previous result obtained for a single cluster of each type

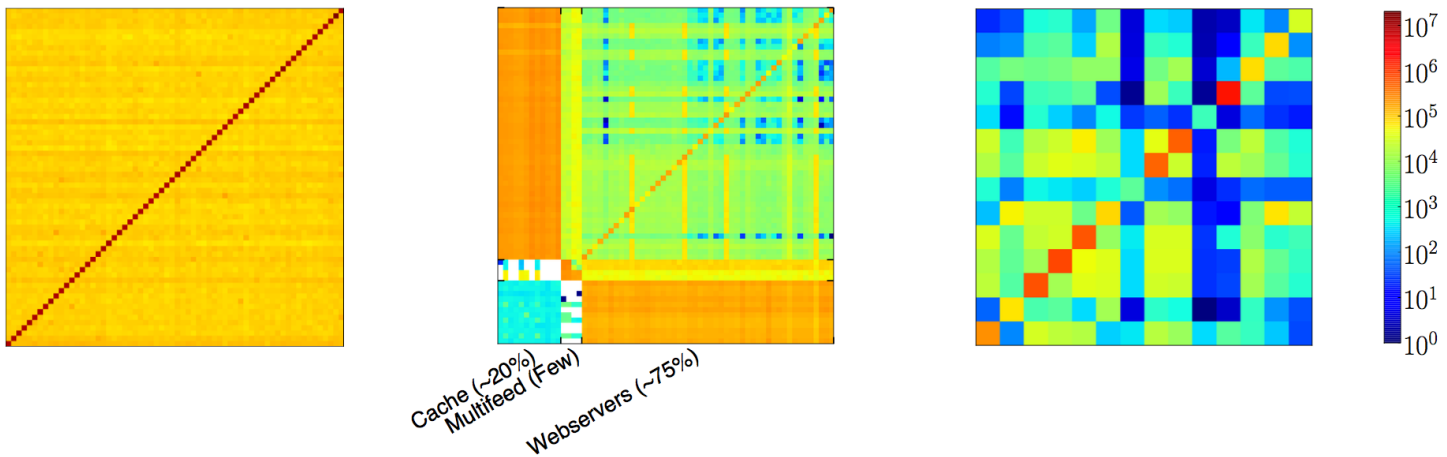
Q: is it representative of all FB $\Rightarrow$ Use of sampling of 64 clusters over 24 hours

Hadoop is the more local

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

Traffic Matrix



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

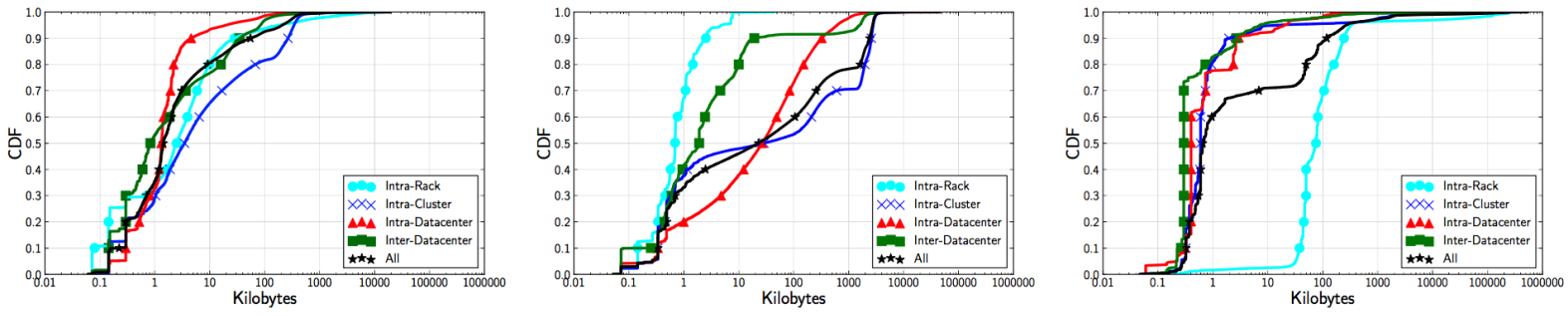
Figure 5: Traffic demand by source (x axis) and destination (y axis). The graphs are each normalized to the lowest demand in that graph type (i.e., the Hadoop and Frontend clusters are normalized to the same value, while the cluster-to-cluster graph is normalized independently).

Traffic Matrix

- Hadoop : strong diagonal $\Rightarrow$ traffic stay local
- Web servers talk to Web caches
- Cluster to cluster : still some amount of locality (diagonal)
- Locality cannot be increased in the case of FB as objects have to be gathered as a function of social graph $\Rightarrow$ difficult to map on objects placement!

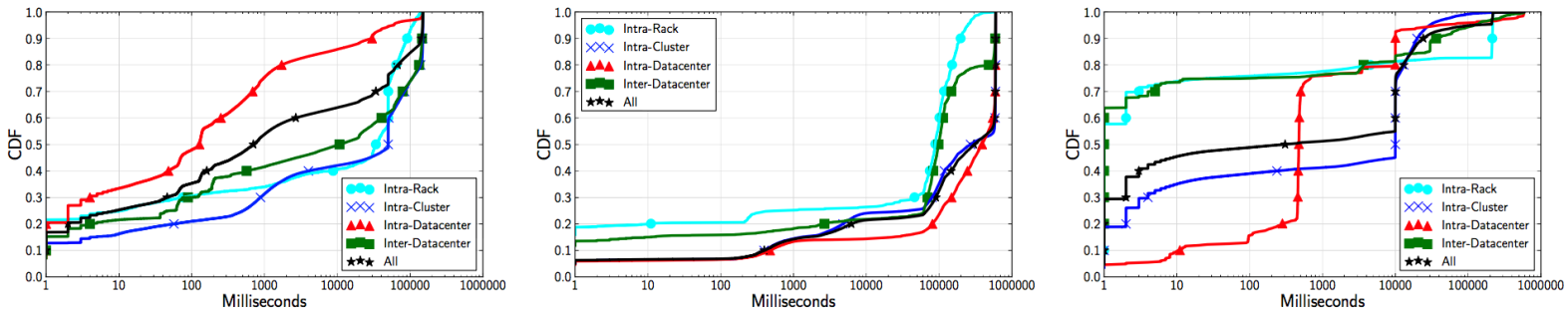
Flow size and duration

Use flow mirroring: 10 minute captures



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

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



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

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

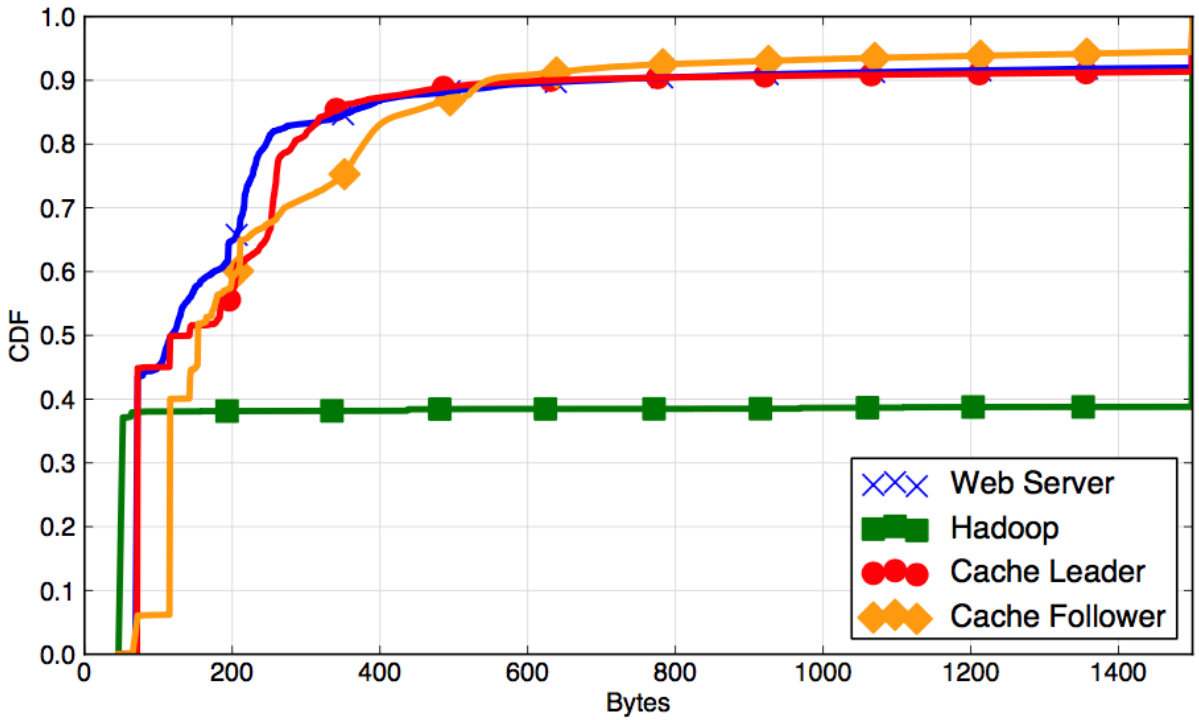
Flow size and duration

- Some services, e.g., cache, designed with long lived connections
- Hadoop: a lot of small flows (but beware of the Hadoop phase you look at....)

Packet size

Very much application dependent:

- Hadoop: TCP acks or full MSS packets
- Web : small packets



FB: implications for traffic engineering

- Traffic engineering: taking advantage of traffic characteristics to improve service
  - e.g., isolate heavy hitters flows (flows that consume significant fraction of bandwidth over time) to give them a different treatment in switches or route them differently
- Study looked for heavy hitters but
  - prevalence and stability difficult to leverage
  - Likely conclusion: use of load balancing + caching is effective a preventing heavy hitters **in the case of FB** (would maybe not hold everywhere)

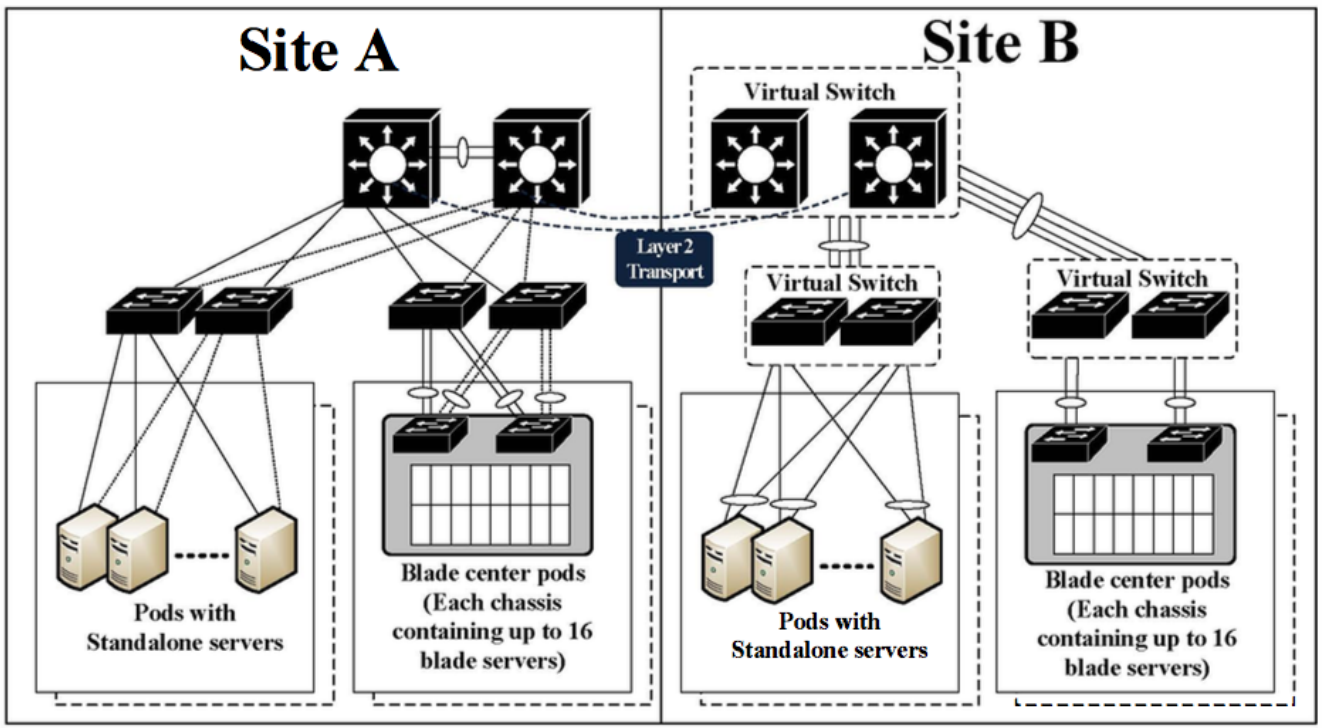
Entreprise Data Center vs. Cloud Data Center

2013 IEEE 14th International Conference on High Performance Switching and Routing

Traffic Measurement and Analysis in an Organic Enterprise Data Center

Ashkan Aghdai, Fan Zhang, Nadun Dasanayake, Kang Xi, H. Jonathan Chao

Measurement of a CISCO based Data Center



EDC Measurement

- Site A: historical one with spanning tree - cannot use of all links
- Site B: uses CISCO FabricPath support
- Several 10s of thousands of users
- Heavy use of virtualization: 55% of virtualized servers.
- Mix of applications: email, le sharing, web services, data base, CRM (customer relationship management), payroll, and purchasing.
- Measurement methods: mirroring at routers and SNMP data (low level of granularity, e.g. nb of packets at interface)

EDC Measurement: key observations

Sparse matrix at different time scales $\Rightarrow$ servers tend to communicate with a few other servers and some servers (e.g. authentication) are popular.

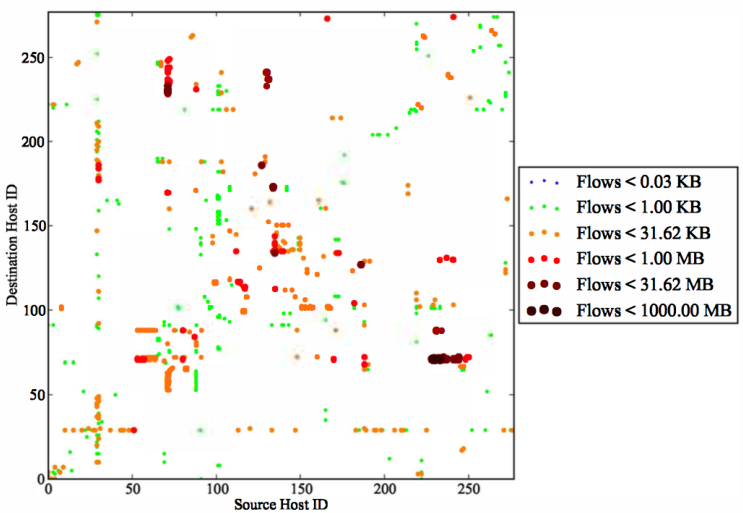


Fig. 2. Traffic matrix on CA2 over a 10-second period.

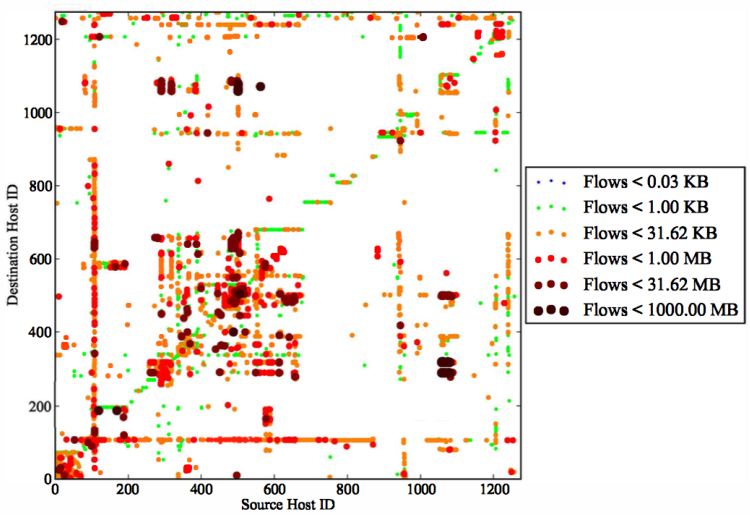


Fig. 3. Traffic matrix on CA1 over a period of 30 minutes.

Traffic locality

Further confirmed by CDF

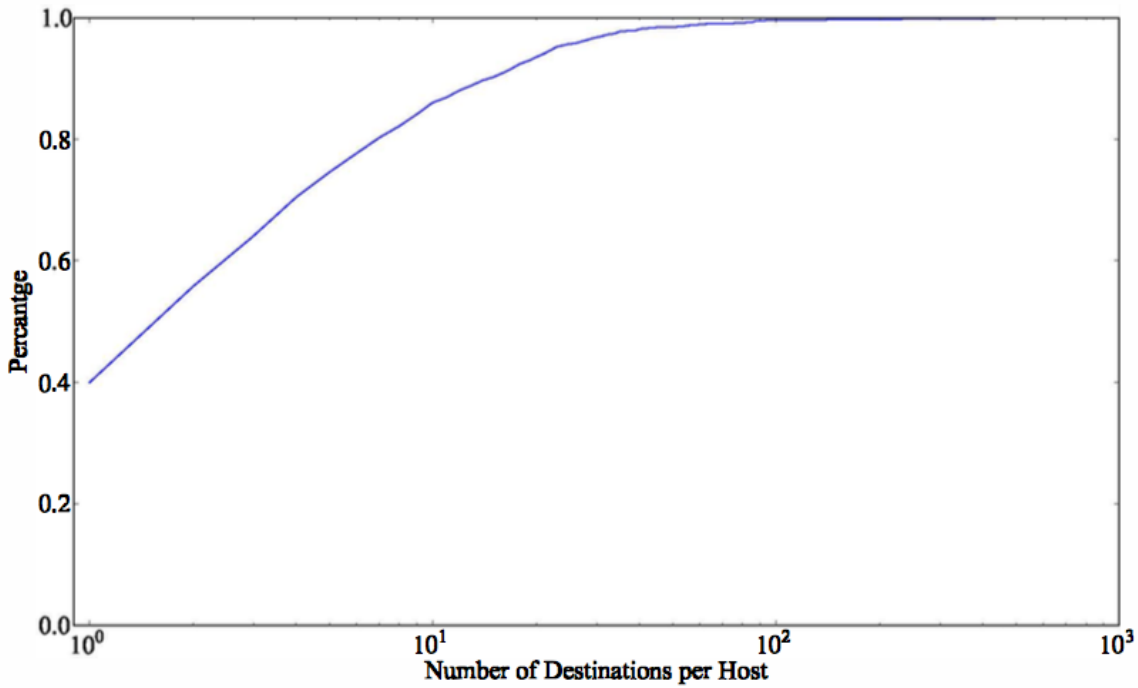


Fig. 4. CDF of number of destinations per host, 30-minute period, CA1.

Flow size and duration

CA (core) and AS (edge server)

Q: can we compare with FB values?

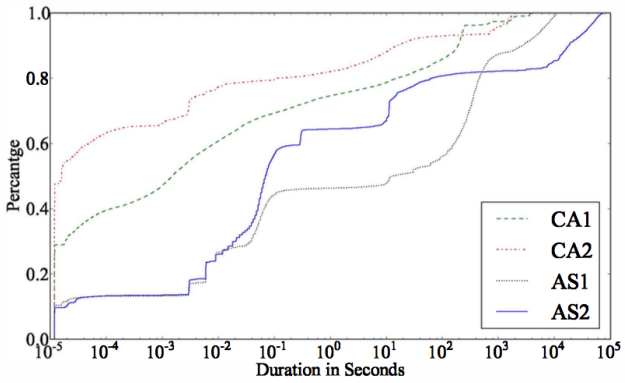


Fig. 5. CDF of flow duration on selected links.

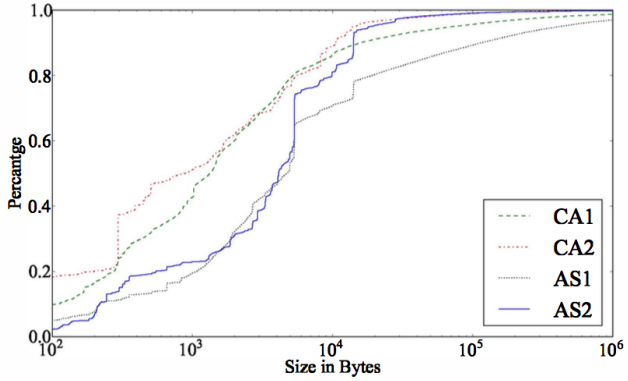


Fig. 6. CDF of flow size on selected links.

Typical deployments in EC2

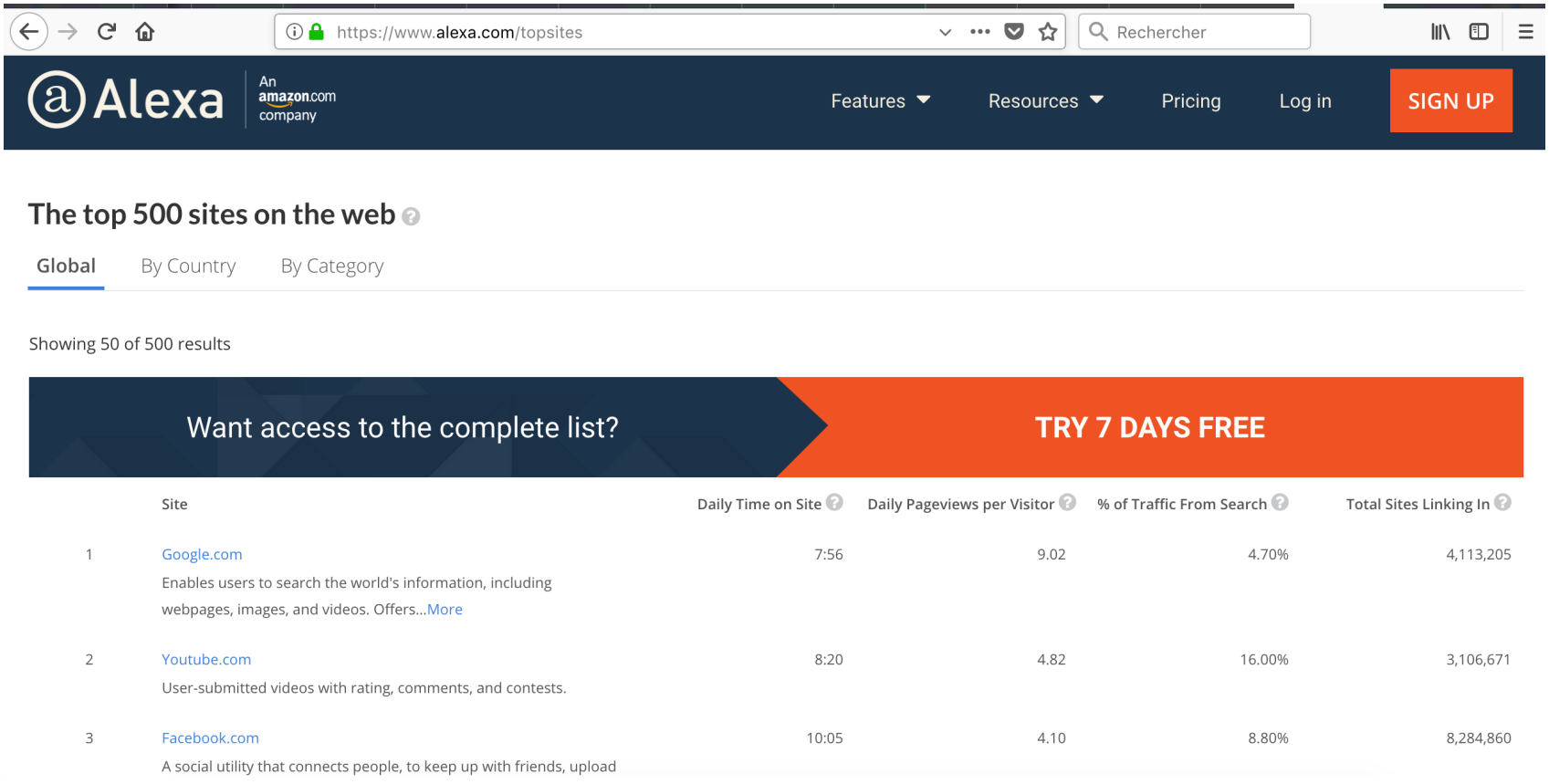
Next Stop, the Cloud: Understanding Modern Web Service Deployment in EC2 and Azure

Keqiang He, Alexis Fisher, Liang Wang, Aaron Gember, Aditya Akella, Thomas Ristenpart
University of Wisconsin – Madison
{keqhe,afisher,liangw,agember,akella,rist}@cs.wisc.edu

Measurement methodology

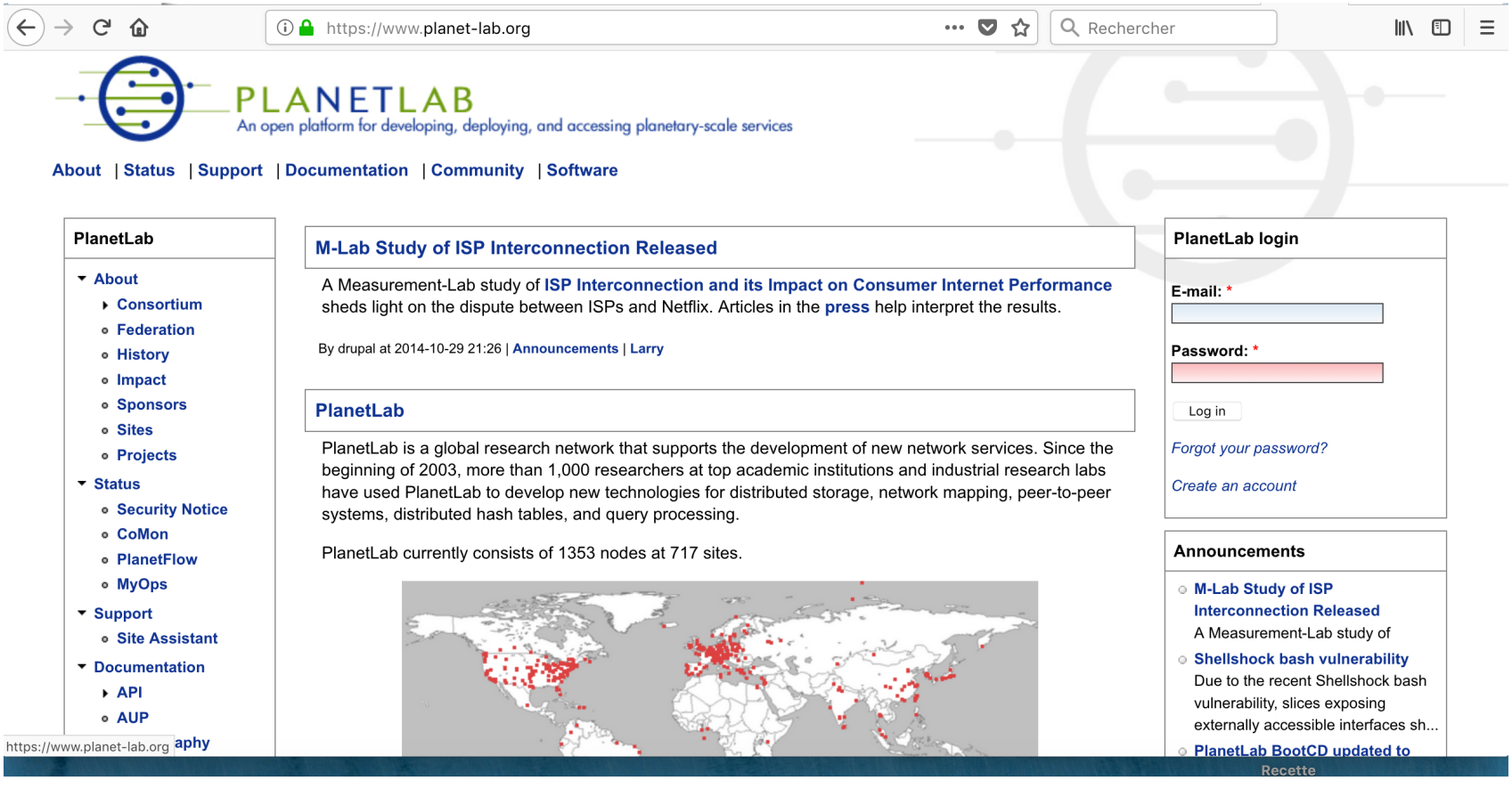
- DNS requests to find fingerprint of cloud providers for the 1 million first Alexa sites
  - A domain might be partly hosted in the cloud, e.g. www.domain.com, cdn.domain.com or news.domain.com
  - They generated a list of possible subdomains + queried the DNS to check if Azure/AWS IP addresses
  - 80K domains and 713K sub-domains
- Packet capture from a large university, University of Wisconsin-Madison - 1week in 2012
  - Unsurprisingly, most traffic is HTTP(s) + minority of FTP, IRC, etc.
- PlanetLab measurements

Alexa


https://www.alexa.com/topsites
Rechercher
Alexa
An amazon.com company
Features
Resources
Pricing
Log in
SIGN UP
The top 500 sites on the web
Global
By Country
By Category
Showing 50 of 500 results
Want access to the complete list?
TRY 7 DAYS FREE
Site
Daily Time on Site
Daily Pageviews per Visitor
% of Traffic From Search
Total Sites Linking In
1
Google.com
Enables users to search the world's information, including webpages, images, and videos. Offers...More
7:56
9.02
4.70%
4,113,205
2
Youtube.com
User-submitted videos with rating, comments, and contests.
8:20
4.82
16.00%
3,106,671
3
Facebook.com
A social utility that connects people, to keep up with friends, upload
10:05
4.10
8.80%
8,284,860

PlaneLab

Distributed measurement platform. Gives you full access to a so-called slice ($\sim$ VM)



Cloud providers service

- Amazon Web Services, Azure, Rackspace, OVH... allow to rent VMs
- ..in different regions..
- .. and different reliability zones (AWS2), i.e. separate power infrasturcture.....
- ... and many more services:
  - load balancers e.g., Amazon Elastic Load Balancer and Azure Traffic Manager
  - PaaS e.g., Amazon Elastic Beanstalk, Heroku, and Azure Cloud Services
  - Content-distribution networks, e.g., Amazon CloudFront and Azure Media Services
  - DNS hosting e.g., Amazon route53
  - etc.

What do tenants use?

Typical deployment for a web server

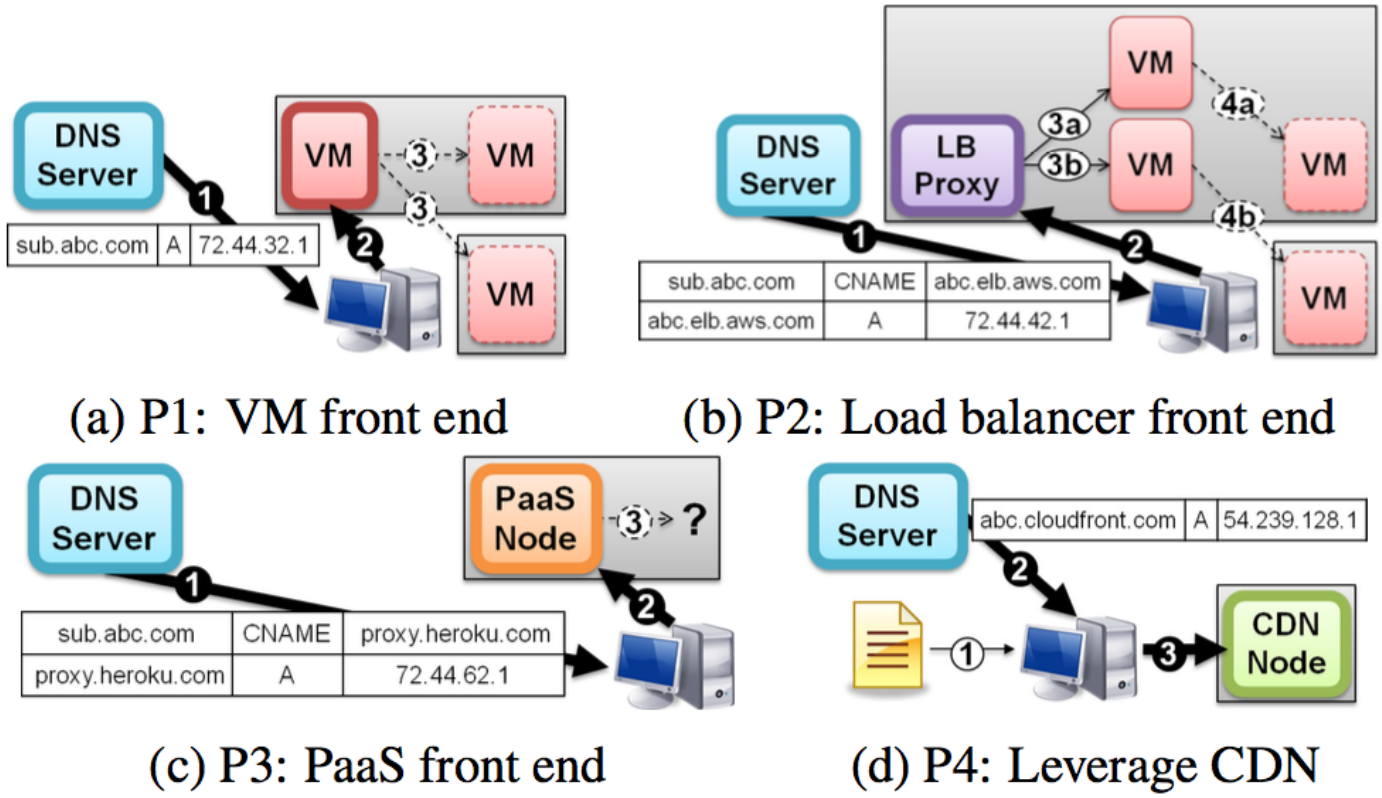


Figure 1: Deployment patterns for web services.

Result for Alexa list

4% of domains

Provider	# Domains (%)	# Subdomains (%)
EC2 only	3,277 (8.1%)	685,725 (96.1%)
EC2 + Other	34,721 (86.1%)	21,628 (3.0%)
Azure only	184 (0.5%)	6,328 (0.9%)
Azure + Other	1,858 (4.6%)	225 (<0.01%)
EC2 + Azure	293 (0.7%)	4 (<0.01%)
Total	40,333 (100.0%)	713,910 (100.0%)
EC2 total	38,291 (94.9%)	707,357 (99.1%)
Azure total	2,335 (5.8%)	6,557 (0.9%)

Table 3: Breakdown of domains and subdomains based on their use of EC2, Azure, and/or other hosting services.

Result for Alexa list - Top 10 domains

Rank	Domain	Total # Subdom	# EC2 Subdom
9	amazon.com	68	2
13	linkedin.com	142	3
29	163.com	181	4
35	pinterest.com	24	18
36	fc2.com	89	14
38	conduit.com	40	1
42	ask.com	97	1
47	apple.com	73	1
48	imdb.com	26	2
51	hao123.com	45	1

Table 4: Top 10 (by Alexa rank) EC2-using domains, their total number of subdomains, and the number of EC2-using subdomains.

Results from UW Madison trace

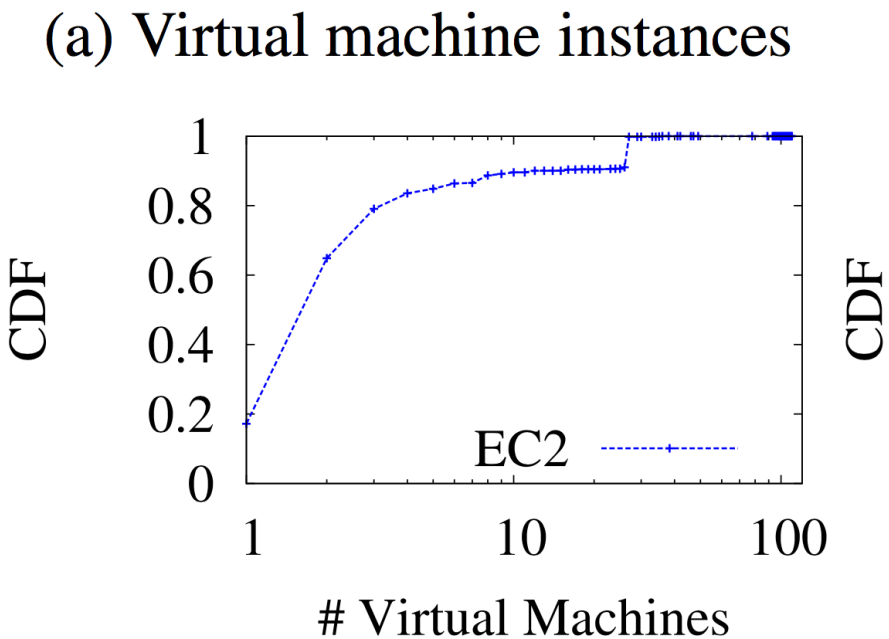
Top 10 domains represent the majority of cloud hosted domains

EC2			Azure		
Domain	**Rank**	**Traffic (%)**	**Domain**	**Rank**	**Traffic (%)**
dropbox.com (d)	119	595.0 (68.21)	atdmt.com	11,128	27.0 (3.10)
netflix.com (d)	92	14.8 (1.70)	msn.com	18	20.9 (2.39)
truste.com (d)	15,458	9.2 (1.06)	microsoft.com	31	19.7 (2.26)
channel3000.com	29,394	6.4 (0.74)	msecnd.net	4,747	13.5 (1.55)
pinterest.com (d)	35	5.1 (0.59)	s-msn.com	25,363	12.5 (1.43)
adsafeprotected.com (d)	371,837	4.7 (0.53)	live.com	7	11.8 (1.35)
zynga.com	799	3.9 (0.44)	virtualearth.net	147,025	9.2 (1.06)
sharefile.com	20,533	3.6 (0.42)	dreamspark.com	35,223	7.1 (0.81)
zoolz.com	272,006	3.2 (0.36)	hotmail.com	2,346	6.3 (0.72)
echoenabled.com (d)	-	2.7 (0.31)	mesh.com	-	4.5 (0.52)
vimeo.com	137	2.3 (0.26)	wonderwall.com	-	3.2 (0.36)
foursquare.com	615	2.2 (0.25)	msads.net	-	2.5 (0.29)
sourcefire.com	359,387	1.9 (0.22)	aspnetcdn.com	111,859	2.3 (0.26)
instagram.com (d)	75	1.5 (0.17)	windowsphone.com	1,597	2.0 (0.23)
copperegg.com	122,779	1.5 (0.17)	windowsphone-int.com	-	2.0 (0.23)

Table 5: Domains with highest HTTP(S) traffic volumes (in GB) in the packet capture. Percentages are relative to the total HTTP(S) traffic across both clouds in the capture. Domains marked with (d) appeared on DeepField's Top 15 [28].

VM front end in EC2

- Name directly resolves to an AWS IP address
- 72% of EC2-based hosting
- In general, tenants seek for resilience (i.e. more than one VM)



PaaS front end in EC2

- PaaS are frenquently built atop IaaS, e.g. Elastic Beanstalk (EBS) and Heroku built on top of EC2
- Detection by searching CNAME with 'elasticbeanstalk' or any of 'heroku.com', 'herokuapp', 'herokucom', and 'herokussl' + AWS IP address
- 8% of EC2-hosted clients
- Heroku is more popular (97% of cases) then EBS
- Heroku:
  - 58,141 subdomains that use Heroku are associated with just 94 unique IPs.
  - $\frac{1}{3}$ of subdomains using Heroku share CNAME 'proxy.heroku.com'

Elastic Load Balancer based EC2 deployments

- A load balancer managed by AWS (you don't see it as a machine on which you can log in)
- Detection by CNAME that ends with elb.amazonaws.com
- 4% of domains
- AWS is multiplexing several domains/sub-domains per ELB instances (distinct IP addresses obtained while resolving CNAMEs)

Front end in Azure

- VMs and PaaS environments both encompassed in logical "Cloud Services" (CS) $\Rightarrow$ cannot be distinguished
- Detection of CS with CNAME with "cloudapp.net"
- 17% of Azure-deployment resolve to a single IP address, 82% to a CNAME
- In case of CNAME, 70% are CS
- 2% use Azure Traffic Manager to balancer load over regions

CDN and DNS in Azure and EC2

- CDN
  - Amazon Cloud Front with different IP range than EC2
  - Azure CDN with same cloud range
  - $\sim$ 6000 domains in AWS and 54 in Azure
- DNS
  - Vast majority of DNS were hosted outside AWS and Azure
  - Maybe because it was a legacy and well performing service...

Summary on deployment by clients

Cloud	Feature	# Domains	# Subdomains	# Inst.
EC2	VM	19.9K (52.5%)	505.6K (71.5%)	28.3K
	ELB	9.9K (25.9%)	27.1K (3.8%)	15.7K
	BeanStalk (w/ ELB)	188 (0.5%)	280 (< 0.01%)	455
	Heroku (w/ ELB	622 (1.6%)	1.9K (0.3%)	2.4K
	Heroku (no ELB)	1.3K (3.5%)	58.1K (8.2%)	94
Azure	CS	863 (37.0%)	4.5K (68.3%)	790
	TM	52 (2.2%)	100 (1.5%)	78

Table 7: Summary of cloud feature usage.

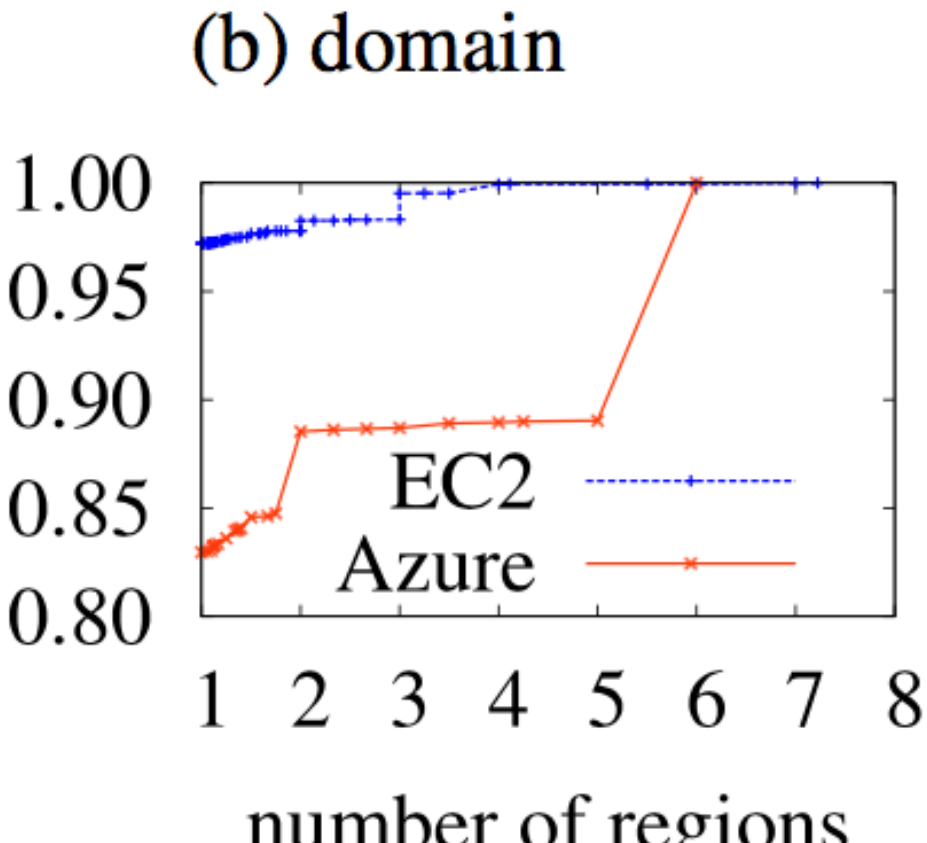
Regions

Popularity is not homogeneous (a function of birth date of DC probably)

Region	Location	# Dom	# Subdom
ec2.us-east-1	Virginia, USA	25,722	521,681
ec2.eu-west-1	Ireland	6,834	116,366
ec2.us-west-1	N. California, USA	3,950	40,548
ec2.us-west-2	Oregon, USA	1,548	15,635
ec2.ap-southeast-1	Singapore	1,800	20,871
ec2.ap-northeast-1	Tokyo, Japan	2,223	16,965
ec2.sa-east-1	São Paulo, Brazil	625	14,866
ec2.ap-southeast-2	Sydney, Australia	313	554
az.us-east	Virginia, USA	268	862
az.us-west	California, USA	161	558
az.us-north	Illinois, USA	590	2,071
az.us-south	Texas, USA	1,072	1,395
az.eu-west	Ireland	564	1,035
az.eu-north	Netherlands	573	1,205
az.ap-southeast	Singapore	379	632
az.ap-east	Hong Kong	333	502

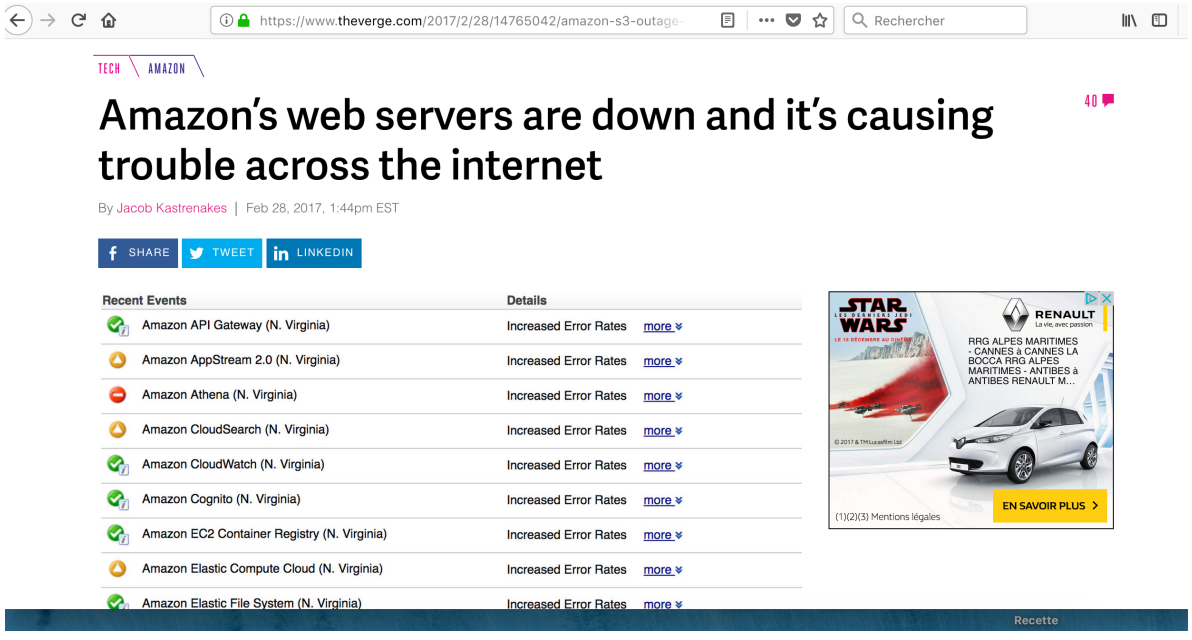
Regions

Most domains hosted in a single region



Regions

Maybe it is still the case in 2017...



Wide area performance

- They measured latency and throughput (HTTP GET of a 2MB object) for 3 consecutive days to three regions from different locations (Planetlab)
- Performance vary significantly between regions

Wide area performance

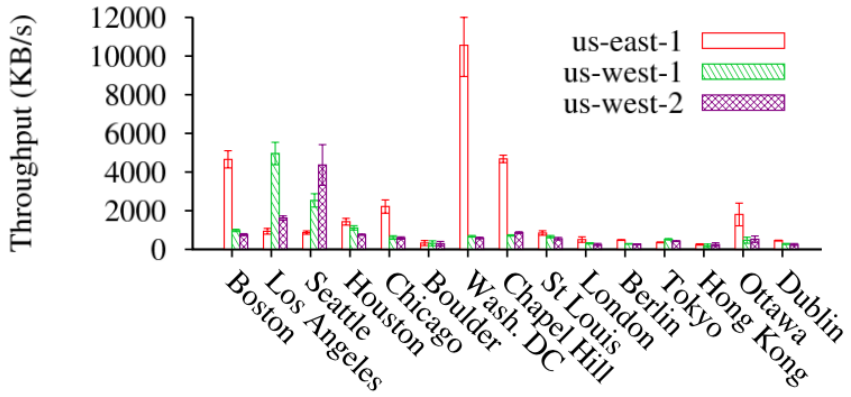


Figure 9: Average throughput between representative clients and EC2 regions in the US.

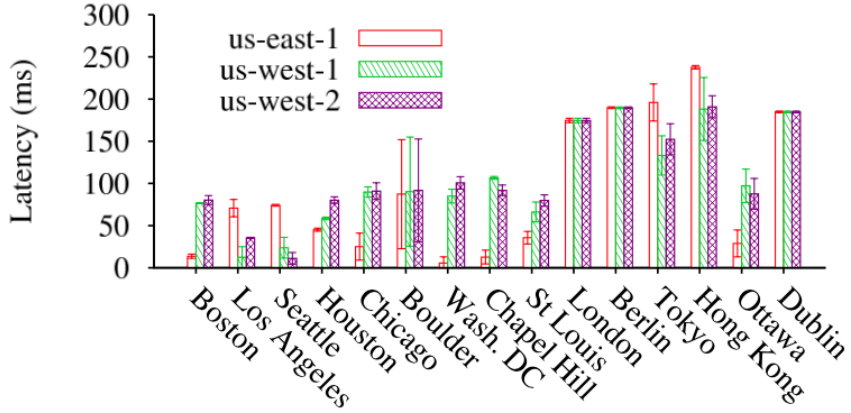


Figure 10: Average latency between representative clients and EC2

Wide area performance

Using multiple regions with algo that direct client to the best one pays off...but the best region for a client vary over time

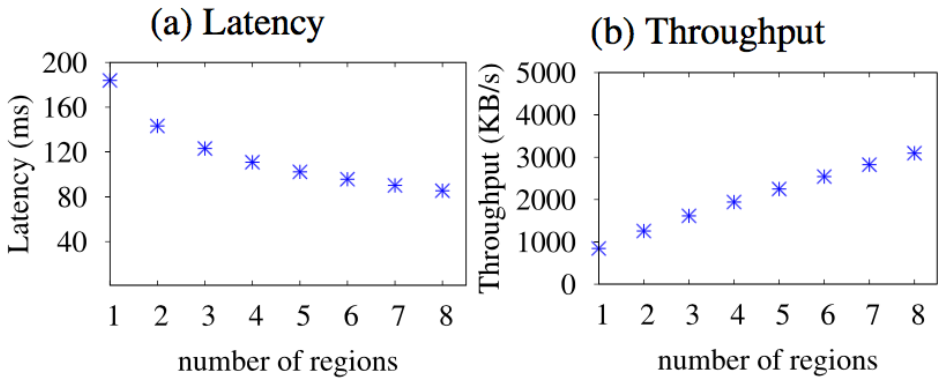


Figure 12: Average latency/throughput across all clients using an optimal k-region deployment.

Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network

Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, and Amin Vahdat
Google, Inc.
jupiter-sigcomm@google.com

Preliminary justification: traffic increase

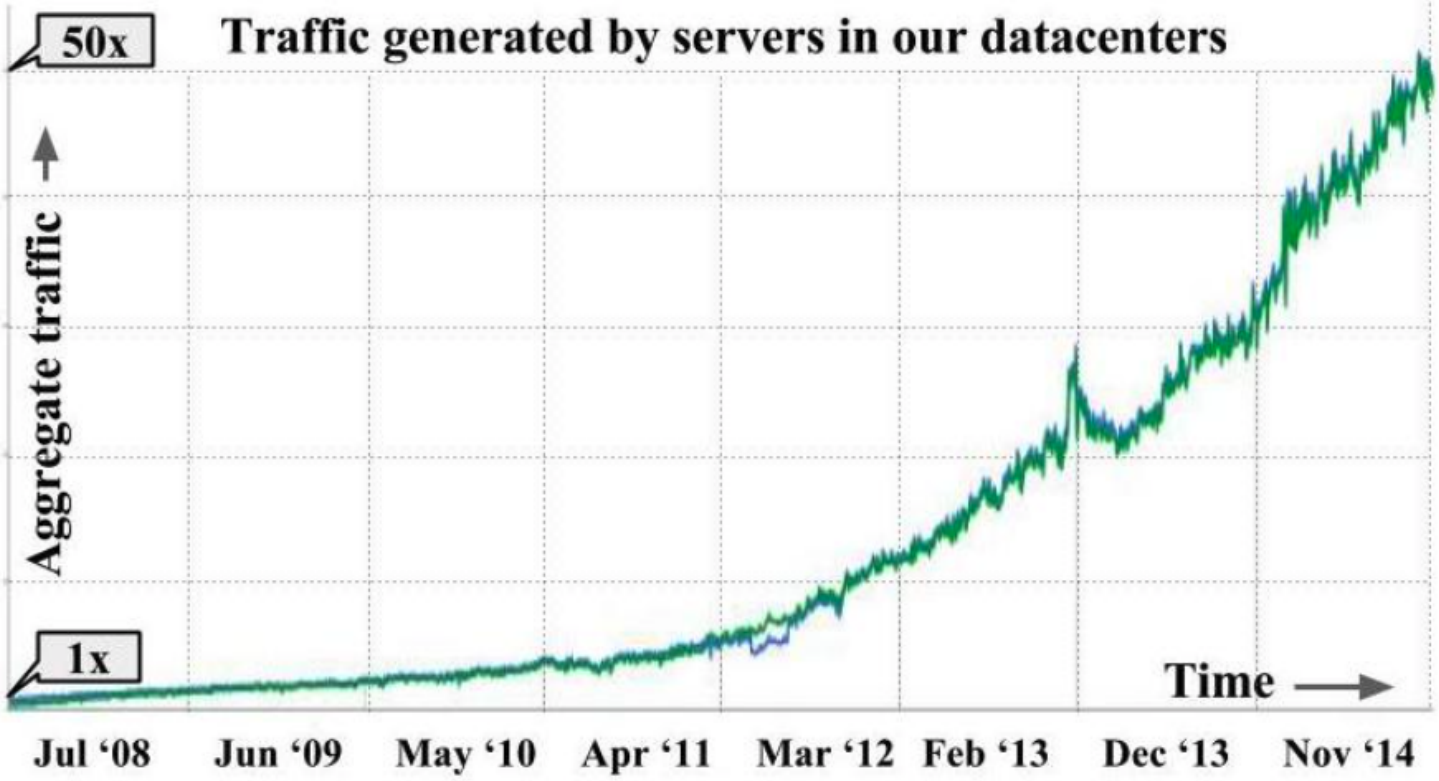


Figure 1: Aggregate server traffic in our datacenter fleet.

Their approach

- Clos topology:
  - Can scale easily by adding stages
  - Path diversity and redundancy
  - Managing the cables is complex
- Merchant silicon
  - Using ISP switches turn out to be too costly and useless functionalities (high reliability)
  - Their approach: built their own switches directly from merchant silicon (off-the-shelf chip components)
- Centralized control protocols (SDN before SDN...)

Initial deployment: 2004

- Highest density Ethernet switches available, 512 ports of 1GE, to build the spine of the network (CRs or cluster routers)
- With up to 40 servers per ToR, up to 20k servers per cluster
- Limitation to 100 Mbps per server

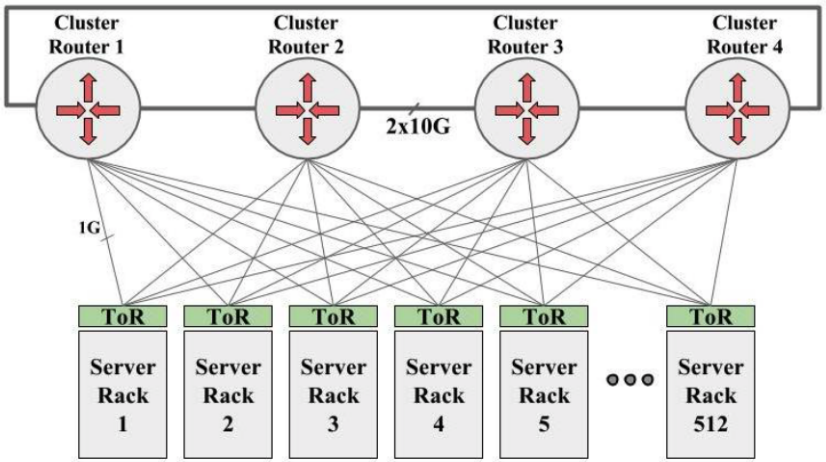


Figure 2: A traditional 2Tbps four-post cluster (2004). Top of Rack (ToR) switches serving 40 1G-connected servers were connected via 1G links to four 512 1G port Cluster Routers (CRs) connected with 10G sidelinks.

What they want

- Large clusters because:
  - They process large amount of data (Web index) that they spread over the cluster (using HDFS - Hadoop File System)
  - The larger the cluster the smaller the number of replica per blocks of data as less correlated failures
  - The larger the cluster, the easier it is schedule large jobs
- As clusters are large + data is spread $\Rightarrow$ need for high bisectional bandwidth.. to reach data

Traffic breakdown + traffic spreading

Job Category	B/w (%)
Storage	49.3
Search Serving	26.2
Mail	7.4
Ad Stats	3.8
Rest of traffic	13.3

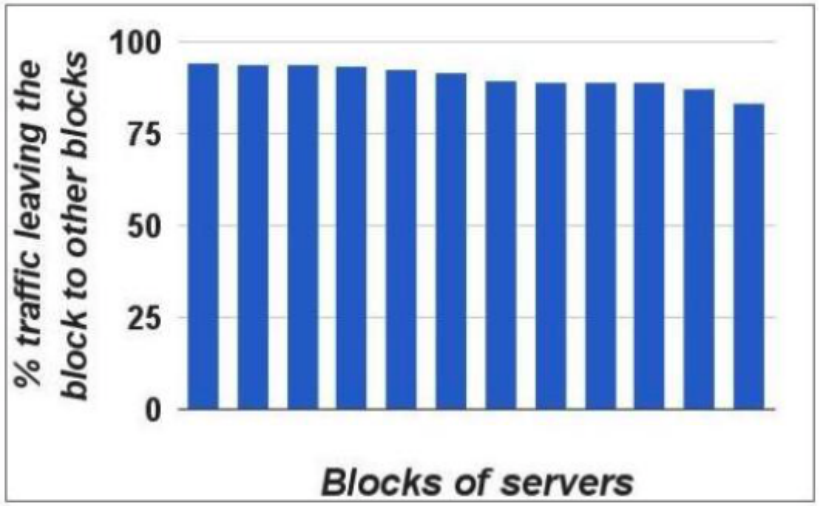


Figure 3: Mix of jobs in an example cluster with 12 blocks of servers (left). Fraction of traffic in each block destined for remote blocks (right).

Their position on alternative topology

HyperX [1], Dcell [17], BCube [16] and Jellyfish [22] deliver more e cient bandwidth for uniform random communication patterns. However, to date, we have found that the benefits of these topologies do not make up for the cabling, management, and routing challenges and complexity.

Topology Generation

Datacenter Generation	First Deployed	Merchant Silicon	ToR Config	Aggregation Block Config	Spine Block Config	Fabric Speed	Host Speed	Bisection BW
Four-Post CRs	2004	vendor	48x1G	-	-	10G	1G	2T
Firehose 1.0	2005	8x10G 4x10G (ToR)	2x10G up 24x1G down	2x32x10G (B)	32x10G (NB)	10G	1G	10T
Firehose 1.1	2006	8x10G	4x10G up 48x1G down	64x10G (B)	32x10G (NB)	10G	1G	10T
Watchtower	2008	16x10G	4x10G up 48x1G down	4x128x10G (NB)	128x10G (NB)	10G	nx1G	82T
Saturn	2009	24x10G	24x10G	4x288x10G (NB)	288x10G (NB)	10G	nx10G	207T
Jupiter	2012	16x40G	16x40G	8x128x40G (B)	128x40G (NB)	10/40G	nx10G/ nx40G	1.3P

Table 2: Multiple generations of datacenter networks. (B) indicates blocking, (NB) indicates Nonblocking.

Software Control

- Initial question: deploy traditional decentralized routing protocols such as OSPF/IS-IS/BGP to manage the fabrics ?
- Challenges:
  - Scale! OSPF area difficult to configure at this scale. BGP sounded difficult also
  - Lack of support for (equal) multipath
- Their approach:
  - A centralized controller that collects and redistribute link-state info at cluster level
  - ... using out-of-band control network

Overall, we treated the datacenter network as a single fabric with tens of thousands of ports rather than a collection of hundreds of autonomous switches that had to dynamically discover information about the fabric.

Intra cluster routing

- All switches configured with baseline or intended topology
- Neighbor discovery protocol to check link status
  - Thousands of cables invariably leads to multiple cabling errors.
  - + misbehaving cables (error rate)
  - Info sent to Controller than redistribute it in a compressed manner (fit in 64 KB)
- Each switch computes routes locally - **Firepath** routing protocol
- Addressing is simple: each rack is an IP subnet and they favor subnetting to enable address aggregation $\Rightarrow$ no VM and no migration of addresses (KISS!)

Recovery/reconvergence time

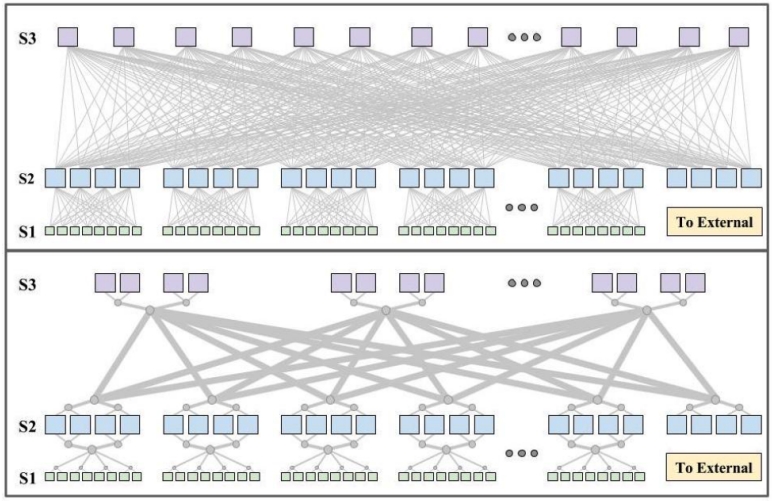


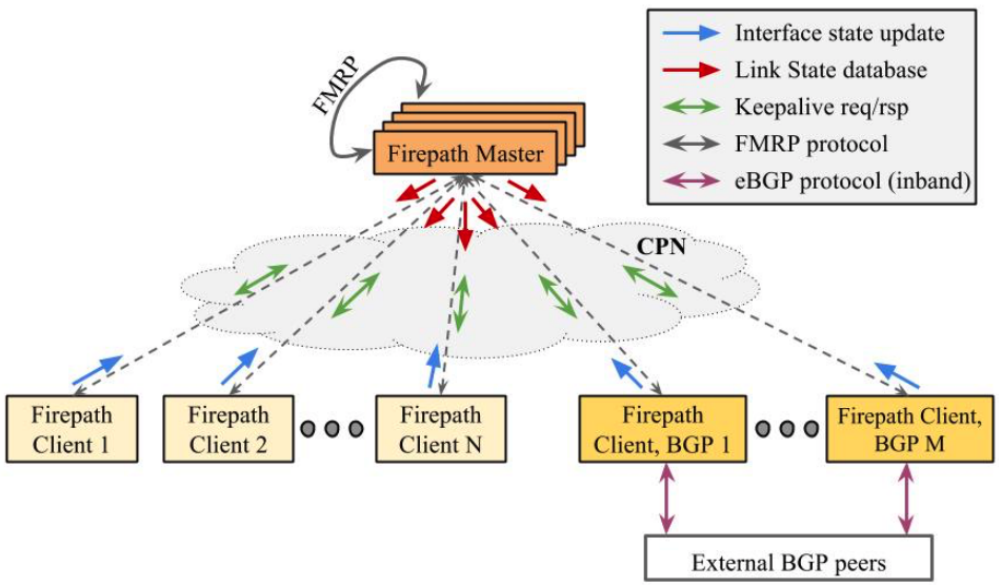
Figure 10: Reducing deployment complexity by bundling cables. Stages 1, 2 and 3 in the fabric are labeled S1, S2 and S3, respectively.

Failure Event	Recovery time (ms)	Convergence type
S2-S3 link	125	Local; affected S2, S3 chassis route around failed link
S3 chassis	325	Local; adjacent S2s route around failed S3
ToR-S2 link	4000	Non-local; all S3s avoid one S2 for impacted ToR
S2 chassis	325	Local; adjacent S3s, ToRs route around failed S2

Table 4: Summary of convergence times in a Saturn cluster.

Routing outside the cluster

- Dedicated to right end side cell in previous topology figure
- Use of BGP
- Synchronization between BGP and Firepath



Observed failures

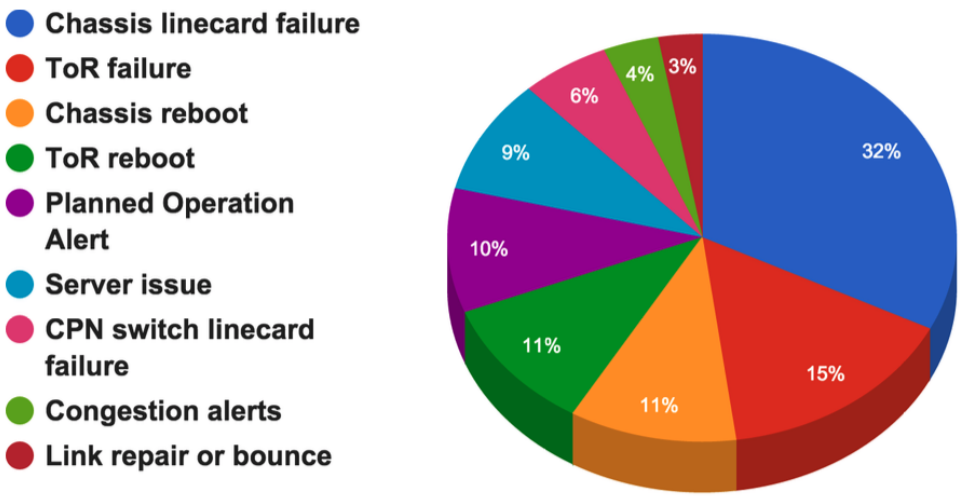
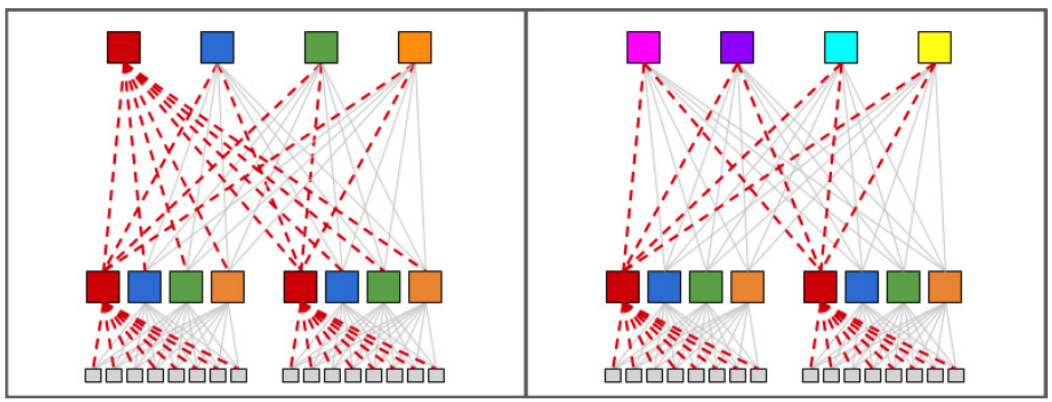


Figure 19: Alerts in Firehose/Watchtower fabrics over 9 months in '08-'09.

"The high incidence of chassis linecard failures was due to memory errors on a particular version of merchant silicon and is not reflective of a trend in linecard failure rates."

Switch firm upgrades

- They developed strategies to upgrade sets of switches simultaneously
- First, red links are put in down mode
- Set of links chosen to limit cluster degradation to 25%
- Then updates are down
- Switches treated like servers (and Google masters servers!) for image management



Congestion

- They observed congestion (1%) even losses when utilization approaching 25%
- Several factors:
  - Burstiness of flows at short time scale typically, incast or outcast (flows arriving or leaving the same server)
  - Commodity switches had too small buffer for TCP to work appropriately
  - Oversubscription, esp ToRs' uplinks
  - Imperfect flow hashing $\Rightarrow$ imbalanced load

Congestion

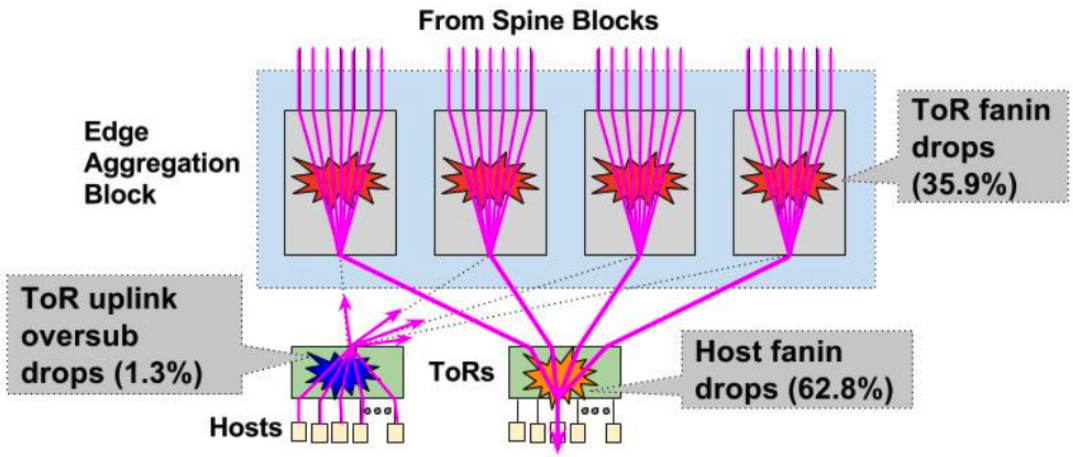


Figure 21: Congestion hotspots in a Saturn fabric.

Congestion

Mitigation solutions:

- Use of QoS to drop low priority traffic at switches
- Tuning of TCP receiver window to cap congestion window as $Win = \min(Adv_w, Cong_w)$
- Improvement of ECMP
- Use o ECN (Explicit Congestion Notification) of TCP

Net result: decrease loss rate by a factor of 10