Initial “Killer apps”

Cloud virtualization

• Create separate virtual networks for tenants
• Allow flexible placement and movement of VMs

WAN traffic engineering

• Drive utilization to near 100% when possible
• Protect critical traffic from congestion

Key characteristics of the above

• Special-purpose deployments with less diverse hardware
• Existing solutions aren’t just annoying, they don’t work!
How large online services work
How large online services work

The Internet
Why multiple data centers?

- Data availability
- Load balancing
- Latency
- Local data laws
- Hybrid public-private operation
Inter-data center traffic is significant

IEEE INFOCOM, 2011
A First Look at Inter-Data Center Traffic Characteristics via Yahoo! Datasets

Yingying Chen¹, Sourabh Jain¹, Vijay Kumar Adhikari¹, Zhi-Li Zhang¹, and Kuai Xu²

¹University of Minnesota-Twin Cities ²Arizona State University
Inter-data center traffic is significant

IEEE INFOCOM, 2011

A First Look at Inter-Data Center Traffic Characteristics via Yahoo! Datasets

Yingying Chen¹, Sourabh Jain¹, Vijay Kumar Adhikari¹, Zhi-Li Zhang¹, and Kuai Xu²

¹University of Minnesota-Twin Cities
²Arizona State University

In this study we consider five major Yahoo! data centers (DAX, DCP, PAO, HK, and UK). DAX, DCP, and PAO are located in the US, and provide services like email, messenger, and video to clients in the US. The UK data center provides services like email, messenger, and video to clients in Europe. The HK data center provides services like email, messenger, and video to clients in Asia. The British Telecommunications PLC (BT) is the Tier-1 Internet Service Provider (ISP) for these data centers.

As mentioned earlier, there have been a few recent studies [1], [2] regarding traffic characteristics within a single site. However, little is known about the inter-data center (D2D) traffic dynamics among multiple data centers within a single ISP. This is the first attempt at analyzing and characterizing inter-data center traffic.

In this section, we provide an overview of the datasets and Yahoo! data centers and their connectivity. We also describe the network segments, which are located at Dallas (DAX), Washington DC (DCP), Atlanta (PAO), and Hong Kong (HK).

Characterization of multiple services: In this section, we present the characteristics of various services that are provided by Yahoo! to its clients. These services include web, email, messenger, and video. We provide a brief overview of the services and their characteristics.

As shown in Table 1, there are two categories of services: i) services that are mostly served using a dedicated set of IP addresses, and ii) services that are served using a pool of IP addresses. In this study, we consider the services that are served using a dedicated set of IP addresses.

It is interesting to note that though D2C traffic is primarily driven by client-server interactions, the flow direction may change due to network and data center changes. This is evident from Table 1, which shows that in addition to D2C traffic, there is also a significant amount of D2D traffic.

We observe that D2C traffic is strongly correlated with D2D traffic. This is evident from the significant overlap in IP addresses between the two categories of services. On the other hand, client-server traffic shows no significant trends over the day; on the other hand, client-server traffic shows no significant trends over the day.

Contributions: The primary contribution of this paper is that we develop novel heuristics and methodologies for characterizing inter-data center traffic. We also propose a novel methodology for extracting the background D2D traffic from the aggregate D2D traffic using the NetFlow datasets collected at a Tier-1 ISP.

To facilitate this, we collect anonymized NetFlow datasets from different data centers. The NetFlow data is collected at a Tier-1 ISP, with the emphasis on inferring the background D2D traffic.

Furthermore, we show that several D2C services are strongly correlated with each other. This is evident from the significant overlap in IP addresses between the two categories of services. This is further evident from the significant overlap in IP addresses between the two categories of services.

Finally, we include an index of terms used in this paper. This index includes terms like IP addresses, NetFlow, and NetFlow datasets.
“Back office” web traffic: server-to-server rather than directly communicating with user
% of web traffic that is “back office” in 4 ISP, IXP data sets
“B4 has been in deployment for three years, now carries more traffic than Google’s public facing WAN, and has a higher growth rate.”
Traditional WAN approach: MPLS
Traditional WAN approach: MPLS

Link-state protocol (OSPF / IS-IS)
Traditional WAN approach: MPLS

- Link-state protocol (OSPF / IS-IS)
- Also flood available bandwidth info
- Fulfill tunnel provisioning requests
Traditional WAN approach: MPLS

- Link-state protocol (OSPF / IS-IS)
- Also flood available bandwidth info
- Fulfill tunnel provisioning requests
- Update network state, flood info
Traditional WAN approach: MPLS

- Link-state protocol (OSPF / IS-IS)
- Also flood available bandwidth info
- Fulfill tunnel provisioning requests
- Update network state, flood info
Problem 1:  inefficiency

Achieving High Utilization with Software-Driven WAN

Chi-Yao Hong (UIUC)  Srikanth Kandula  Ratul Mahajan  Ming Zhang
Vijay Gill  Mohan Nanduri  Roger Wattenhofer (ETH)
Microsoft

Time
Utilization
Problem 2: in flexible sharing

2x the bandwidth!
B4 key design decisions

[Jain et al., SIGCOMM 2013]

Separate hardware from software

B4 routers custom-built from merchant silicon

Drive links to 100% utilization

Centralized traffic engineering
Google’s B4

“B4: Experience with a Globally-Deployed Software Defined WAN”
Jain et al., ACM SIGCOMM 2013

circa 2011
Google’s B4

“B4: Experience with a Globally-Deployed Software Defined WAN”
Jain et al., ACM SIGCOMM 2013
Google’s B4: view at one site

Data center network

Cluster border routers

WAN routers

iBGP / IS-IS to other sites

eBGP

iBGP / IS-IS to other sites

Quagga

OpenFlow Controller

TE server

From “Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google’s Datacenter Network”, Singh et al., ACM SIGCOMM’15

Figure 8: Firehose 1.1 deployed as a bag-on-the-side Clos fabric.

Figure 9: A 128x10G port Watchtower chassis (top left). The internal non-blocking topology over eight linecards (bottom left). Four chassis housed in two racks cabled with fiber (right).

3.3 Watchtower: Global Deployment

Our deployment experience with Firehose 1.1 was largely positive. We showed that services could enjoy substantially more bandwidth than with traditional architectures, all with lower cost per unit bandwidth. Firehose 1.1 went into production with a handful of clusters and remained operational until recently. The main drawback to Firehose 1.1 was the deployment challenges with the external copper cabling.

We used these experiences to design Watchtower, our third-generation cluster fabric. The key idea was to leverage the next-generation merchant silicon switch chips, 16x10G, to build a traditional switch chassis with a backplane. Figure 9 shows the half rack Watchtower chassis along with its internal topology and cabling. Watchtower consists of eight line cards, each with three switch chips. Two chips on each line card have half their ports externally facing, for a total of 16x10GE SFP+ ports. All three chips also connect to a backplane for port to port connectivity. Watchtower deployment, as seen in Figure 9 was substantially easier than the earlier Firehose deployments. The larger bandwidth density of the switching silicon also allowed us to build larger fabrics with more bandwidth to individual servers, a necessity as servers were employing an ever-increasing number of cores.

Fiber bundling further reduced the cabling complexity of Watchtower clusters. Figure 10 shows a Watchtower fabric deployment without any cable bundling. Individual fibers of varying length need to be pulled from each chassis location, leading to significant deployment overhead. The bottom figure shows how bundling can substantially reduce complexity. We deploy two chassis in each rack and co-locate two racks. We can then pull cable bundles to the midpoint of the co-located racks, where each bundle is split to each rack and then further to each chassis.

Finally, manufacturing fiber in bundles is more cost effective than individual strands. Cable bundling helped reduce fiber cost (capex + opex) by nearly 40% and expedited bringup of Watchtower fabric by multiple weeks. Table 3 summarizes the bundling and cost savings.

Table 3: Benefits of cable bundling in Watchtower.

<table>
<thead>
<tr>
<th>Bundle Type</th>
<th># Bundles</th>
<th>Normalized Cost of Fiber/m with Bundling (capex + opex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2-S3 bundles (16-way)</td>
<td>512</td>
<td>55%</td>
</tr>
<tr>
<td>S2-ToR bundles (8-way)</td>
<td>960</td>
<td>60%</td>
</tr>
<tr>
<td>Total cable bundles</td>
<td>1472</td>
<td>57%</td>
</tr>
</tbody>
</table>

188

From “Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google’s Datacenter Network”, Singh et al., ACM SIGCOMM’15
Google’s B4: Traffic engineering
Vs. Semi-Distributed TE

What aspects of B4 would have been difficult with MPLS-based TE such as TeXCP?

What aspects of B4 are similar to TeXCP?
1. How does B4 scale?
   - Subsecond centralized scheduling of more traffic than Google’s public WAN serves!

2. What does B4 assume about network’s traffic?
   - In what environments would these assumptions be violated?
   - In what other environments would they be valid?
How does B4 scale?
How does B4 scale?

Hierarchy

- Not a simple controller-to-switch design!
How does B4 scale?

Hierarchy

- Not a simple controller-to-switch design!
How does B4 scale?

Hierarchy

- Not a simple controller-to-switch design!

Aggregation

- Node = site (data center)
- Link = 100s of links
- Flow group = \{src, dst, QoS\} tuple

Figure 1: B4 worldwide deployment (2011).

Figure 2: B4 architecture overview.
How does B4 scale?

Hierarchy

- Not a simple controller-to-switch design!

Aggregation

- Node = site (data center)
- Link = 100s of links
- Flow group = \{src, dst, QoS\} tuple

Algorithms

- Greedy heuristic approximation algorithm
Design makes what assumption about traffic to approach 100% utilization on some links?

- High priority traffic is in the minority
- Elastic traffic is the majority (backups, offline data analytics, ...)

![Graph showing HIPRI/LOPRI packets and dropped fractions over time]
What assumptions about traffic?

Design makes what assumption about traffic to approach 100% utilization on some links?

- High priority traffic is in the minority
- Elastic traffic is the majority (backups, offline data analytics, ...)

When would that assumption be violated?

- Google’s user-facing wide area network