Advanced Computer Networking (ACN)

IN2097 – WiSe 2023–2024

Prof. Dr.-Ing. Georg Carle

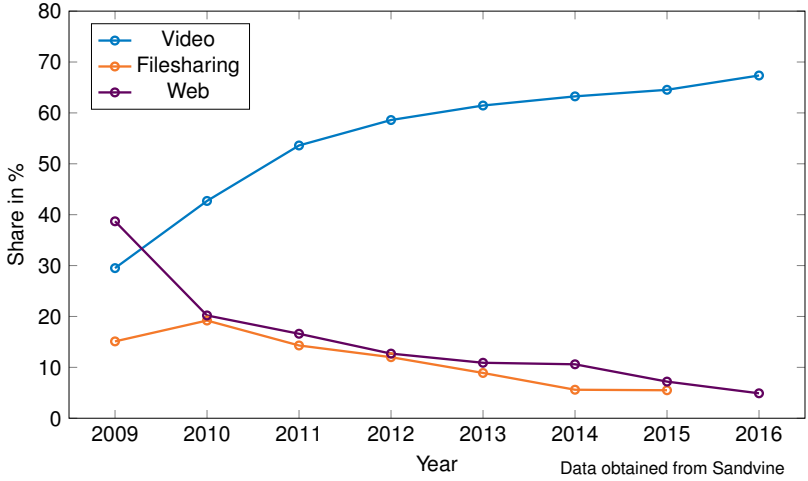
Sebastian Gallenmüller, Max Helm, Benedikt Jaeger,
Marcel Kempf, Patrick Sattler, Johannes Zirngibl

Chair of Network Architectures and Services
School of Computation, Information, and Technology
Technical University of Munich

Content Delivery Networks (CDN)

Content Delivery Networks (CDN)

Motivation - Internet Development - North America



Data sources: [1]–[6]

Introduction
Motivation

Problems

- Websites often need to handle a lot of traffic
- Video on demand is mainstream and replacing TV
- Videos need a lot of bandwidth
- Connecting to a server far away introduces latency
- Latency is the foe of user satisfaction
- Distributed Denial of Service (DDoS) attacks are common (e.g., Mirai botnet)

Introduction
Goal

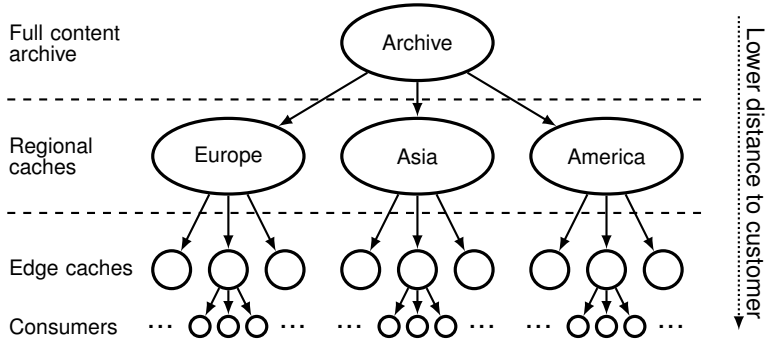
What do we want to achieve?

- Distribute the load over a lot of content servers
- Have servers physically as near as possible at the customer site
- Content servers at the “end” of the CDN serving static content (e.g., videos) are called “edge-caches”
- Edge-caches may be placed in customer ISP networks (Netflix does that)
- Distributing load can mitigate DDoS
- Caching can reduce the amount of traffic between ASes

Benefits

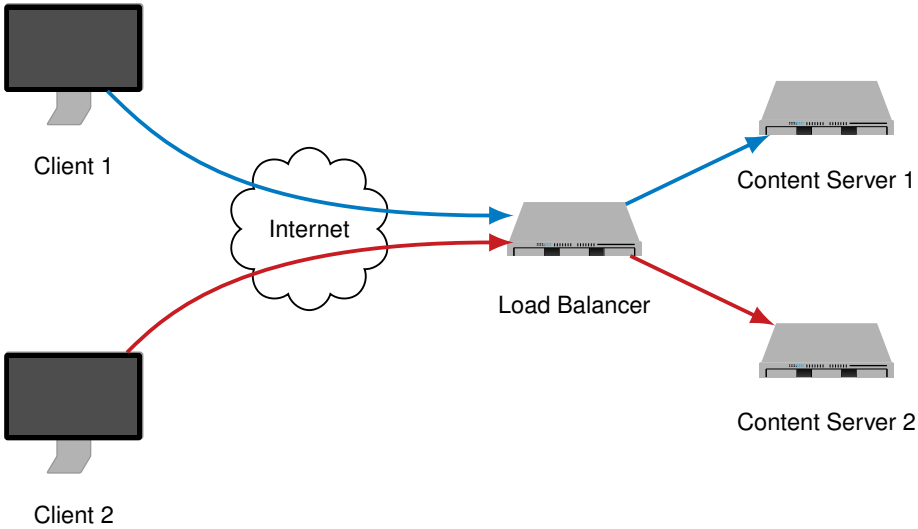
- For the consumer:
  - Lower latency & higher reliability
- For ISP of the consumer:
  - Lower traffic, reduced costs & better service quality
- For content providers:
  - Lower traffic, reduced costs & better service quality

Cache hierarchy



- Archive holds all the data (e.g. the whole video library)
  - Archive may, as well, be distributed itself
- Regional caches get their content from the archive
- Edge caches (located at ISPs / IXPs) get their content from their regional cache
- If a cache does not hold the requested content, it asks its parent
- Real deployments may have more levels / be more sophisticated

Typical architecture



Hashing
Recall: 5-tuple Hashing

Problem

- Data-plane devices forward packets
- Packets usually belong to a flow (think: TCP session)
- There may be multiple next hops for the destination of a packet
- If different next hops are used within one flow, packets may be reordered
- Reordering messes with TCP Congestion-Control

Solution

- Compute so called 5-tuple hash of the following fields:
  1. Source IP address
  2. Destination IP address
  3. Source port
  4. Destination port
  5. Type of L4-protocol
- Hash is the same for all packets within one flow
- Choose next hop by means of the hash
- Result: All packets of one flow use the same path from source to destination

Modulo hashing

Problem setting

- Need to map one client to one of N servers

Naive solution (also called modulo hashing)

- Let h be the 5-tuple hash
- Redirect the client to server: $h \mod N$

Modulo hashing

Problem setting

- Need to map one client to one of N servers

Naive solution (also called modulo hashing)

- Let h be the 5-tuple hash
- Redirect the client to server: $h \mod N$

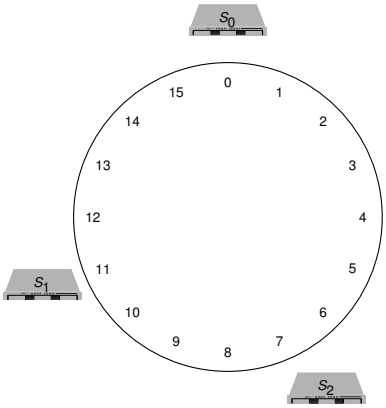
Problem

- What happens if N changes?
- Every client is hashed to a new location

Consistent Hashing

Principle of Consistent Hashing [7]

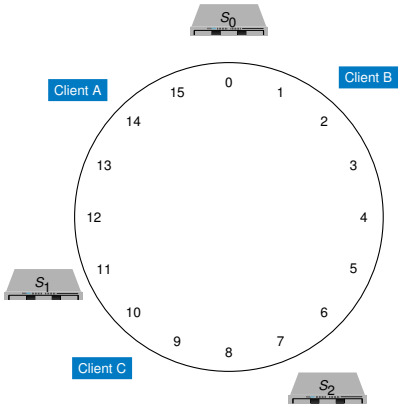
- Map each client to a point on the edge of a circle (e.g. [0 ... 15])



Consistent Hashing

Principle of Consistent Hashing [7]

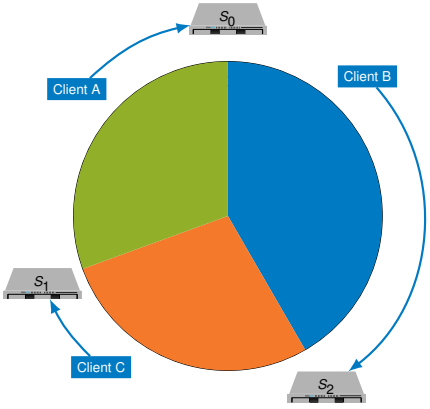
- Map each client to a point on the edge of a circle (e.g. [0 ... 15])
- Clients walk around the circle in order to find which server to use



Consistent Hashing

Principle of Consistent Hashing [7]

- Map each client to a point on the edge of a circle (e.g. [0 ... 15])
- Clients walk around the circle in order to find which server to use



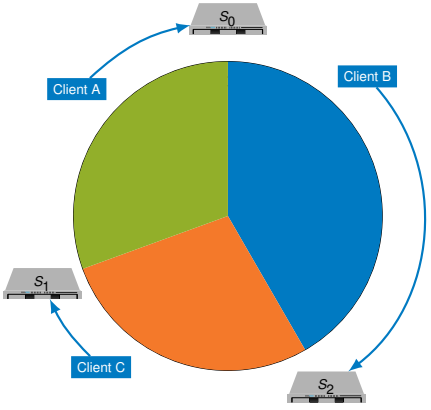
Consistent Hashing

Principle of Consistent Hashing [7]

- Map each client to a point on the edge of a circle (e.g. [0 ... 15])
- Clients walk around the circle in order to find which server to use

What happens if the number of servers *N* changes?

- Points are removed or added on the circle
- Clients are remapped according to the available positions on the circle



Consistent Hashing

Principle of Consistent Hashing [7]

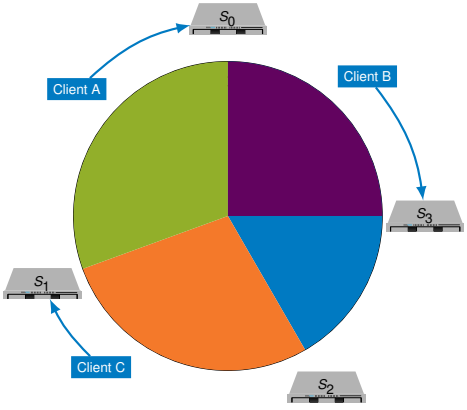
- Map each client to a point on the edge of a circle (e.g. [0 ... 15])
- Clients walk around the circle in order to find which server to use

What happens if the number of servers *N* changes?

- Points are removed or added on the circle
- Clients are remapped according to the available positions on the circle

Advantage

- Only $\frac{K}{N}$ keys need to be remapped on average with K the number of clients



Consistent Hashing

Principle of Consistent Hashing [7]

- Map each client to a point on the edge of a circle (e.g. [0 ... 15])
- Clients walk around the circle in order to find which server to use

What happens if the number of servers *N* changes?

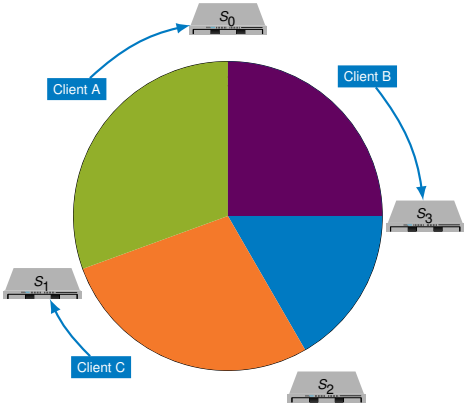
- Points are removed or added on the circle
- Clients are remapped according to the available positions on the circle

Advantage

- Only $\frac{K}{N}$ keys need to be remapped on average with K the number of clients

Improvement

- Map servers not to a single but multiple positions in the circle $\rightarrow$ more even distribution of clients to server mapping

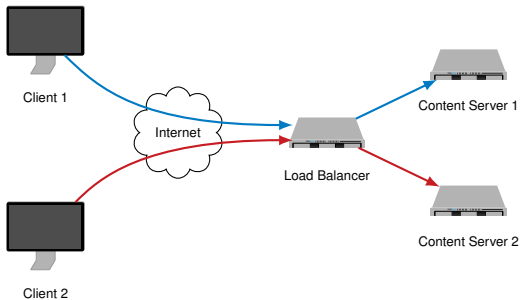


Content Delivery Networks (CDN)

TCP/HTTP-based Load Balancing
TCP Load Balancers

Idea

- Most high-volume traffic is transferred over TCP
- Have one front-end server and multiple back-end servers
- Back end servers may perform heavy tasks (e.g., PHP scripts, database accesses)
- Front end servers only forward packets to the back end
- Sessions are equally distributed between back-end servers



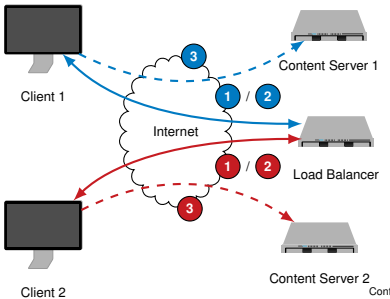
TCP/HTTP-based Load Balancing
HTTP Load Balancers

Problem

- Most user-facing content (e.g., Netflix) is sent via HTTP(S)
- “Stupid” TCP load balancers are rather inefficient - need for central front end, which is under load
- Way better: Direct the user to one server out of a pool

Solution

- Have one redirection server (front end)
- Front end knows a pool of content servers
- Front end always answers with a “302 Temporarily Moved”
- Browser will automatically connect to the content server mentioned in the “moved” message



TCP/HTTP-based Load Balancing
Redirection example

```
Munich ~$ nc google.com 80
GET /
HTTP/1.0 302 Found
Cache-Control: private
Content-Type: text/html; charset=UTF-8
Referrer-Policy: no-referrer
Location: http://www.google.de/?gfe_rd=cr&ei=0JltWZ_4C4SV8QeC0qCoCQ
Content-Length: 258
Date: Tue, 18 Jul 2017 05:17:04 GMT

<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>302 Moved</TITLE></HEAD><BODY>
<H1>302 Moved</H1>
The document has moved
<A HREF="http://www.google.de/?gfe_rd=cr&amp;ei=0JltWZ_4C4SV8QeC0qCoCQ">here</A>.
</BODY></HTML>

Munich ~$ dig +short google.com
172.217.22.238

Munich ~$ dig +short www.google.de
172.217.22.99
```

TCP/HTTP-based Load Balancing
HTTPS Deployment

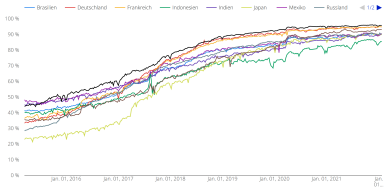


Figure 1: Percentage of HTTPS requests in Chrome[1]

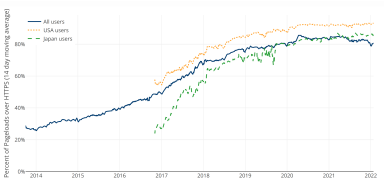


Figure 2: Percentage of HTTPS requests in Firefox[2]

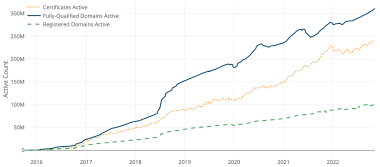


Figure 3: Active Let's Encrypt certificates and FQDNs[3]

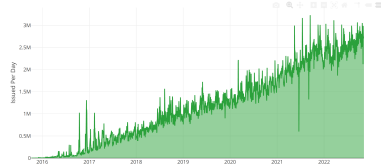


Figure 4: Certificates issued by Let's Encrypt per day[4]

1 https://transparencyreport.google.com/https

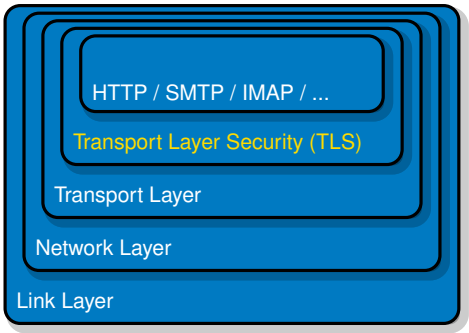
2 https://letsencrypt.org/stats/

3 https://transparencyreport.google.com/https

4 https://letsencrypt.org/stats/

TCP/HTTP-based Load Balancing
HTTPS Problem

Recall: Layering of TLS:



Why is encryption potentially problematic?

- Front end may want to select a server based on URL, cookies, . . .
  - NGINX supports selecting a consistent content server based on a special cookie
- Traffic contains confidential content (e.g., passwords, cookies, . . .)
- Front end server needs to have the TLS-Certificate
- What happens after the connection is decrypted?

TCP/HTTP-based Load Balancing
HTTPS Problem

Encryption is great, but inconvenient

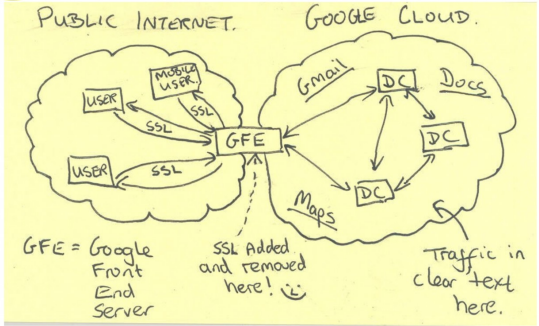
- USE ENCRYPTION :)
- CDN deployments may be very large - multiple racks or rooms
- Where should you decrypt user traffic?

Options

- Use an SSL/TLS front-end - special high-performance hardware
  - Snooping inside the network yields clear text
  - Very cost efficient
  - One central configuration
- Perform decryption/encryption on the content server
  - May burn a lot of CPU cycles
  - Decryption as late as possible
  - Easily deployed out-of-the-box

TOP SECRET//SI//NOFORN

Current Efforts - Google



TOP SECRET//SI//NOFORN

[5] B. Gellman and A. Soltani, "Nsa infiltrates links to yahoo, google data centers worldwide, snowden documents say," Wahington Post: https://www.washingtonpost.com/world/national-security/nsa-infiltrates-links-to-yahoo-google-data-centers-worldwide-snowden-documents-say/2013/10/30/e51d661e-4166-11e3-8b74-d89d714ca4dd_story.html, 2013

Content Delivery Networks (CDN)

DNS-based Load Balancing
Recall: DNS [9]

What is DNS?

- Most distributed database on this planet
- “Domain Name System”
- Resolves domain names (acn.net.in.tum.de) into IP addresses
- Quite flexible
- Nameserver: Has an up-to-date copy of the zone file (knows mapping: name $\longrightarrow$ IP)
- Resolver: Queries nameservers on behalf of the client

DNS-based Load Balancing
DNS based CDNs

Idea

- TCP/HTTP Load balancers still need some kind of traffic waste / useless RTT
- Why not directly connect to the correct content server?
- Servers are usually found via DNS
- Therefore DNS is the obvious choice

Two different possibilities

- Lots of entries
- Geo-based

DNS-based Load Balancing
Many DNS entries (Naive approach)

Idea

- DNS may return multiple A/AAAA entries
- Client chooses one randomly
- In average, each server should get equal amount traffic
- Client/DNS does the load balancing

Example

```
Munich ~$ dig +short netflix.com
52.50.245.135
54.171.208.83
54.171.226.127
52.51.95.138
52.50.133.131
52.48.44.196
52.49.219.75
52.48.236.77
```

DNS-based Load Balancing
Geo-DNS

Existing Problems

- Naive DNS load balancing may lead to endpoints far away

Solution

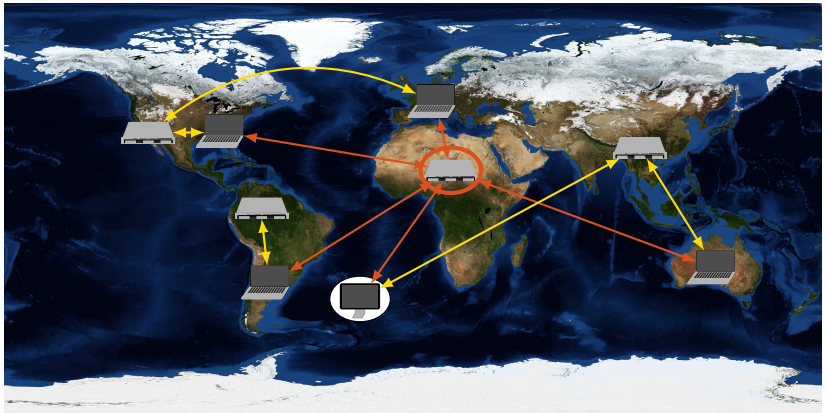
- Nameserver has IP address of resolver
- Resolver may give the /24 prefix of the client
- Clients often use ISP resolvers (often same city as clients)
- Nameserver knows the rough geographical location of the resolver/client
- Provide the resolver/client with the nearest content server
- May still be mislead (use resolver in another country, which doesn't give the /24 prefix ...)

Example: youtube.com [6]

City	IPv4 address	IPv6 address
Germany, Falkenstein	216.58.213.206	2a00:1450:4005:803::200e
USA, North Carolina	74.125.138.190, ...	2a00:1450:4011:805::1001
USA, New Jersey	172.217.3.110	2607:f8b0:4006:818::200e
United Kingdom, London	74.125.206.190, ...	2a00:1450:400c:c04::5b

6 Data obtained via `https://check-host.net/check-dns?host=youtube.com`

DNS-based Load Balancing
Geo-DNS Example



World image created by NASA, https://visibleearth.nasa.gov/view.php?id=73909

Nameserver

Content Server

User

Ship
Satellite internet via China

DNS

HTTP(S)

DNS-based Load Balancing
Combination of both

Lots of different entries for different locations

```
Munich ~$ dig +short netflix.com
52.208.128.101
52.18.15.9
52.19.40.147
52.209.210.113
52.17.219.77
52.208.135.54
52.19.20.249
52.208.236.195

Frankfurt ~$ dig +short netflix.com
54.76.83.27
54.77.81.254
54.77.186.213
54.77.210.48
176.34.151.201
52.17.227.174
54.76.226.73
54.72.216.241
```

DNS-based Load Balancing
DDoS Protection HTTPS Problems

DDoS Protection scheme

- Set DNS up to point to the protection provider's CDN (e.g. Cloudflare)
- All traffic passes through the CDN
- CDN caches static content (e.g., videos)
- Forwards dynamic content requests (e.g., PHP-websites) to the "real" webserver

Problem

- CDN needs to terminate the TLS connection
- CDN has complete access to the connection, including secret cookies, passwords, . . .

How is the connection between the CDN and the real webserver protected?

- Maybe not at all
- Maybe by TLS
- Maybe by TLS without certificate validation
- Maybe by IPsec

Content Delivery Networks (CDN)

Anycast-based Load Balancing
Anycast

Problems with DNS

- Still may be misleading
- DNS was never intended to work that way

Solution

- Packets are routed through BGP
- Manipulating routes via BGP
- Assign lots of content servers the same IP
- Announce the IP prefix through lots of different sites / peerings

Result

- BGP takes care of finding the best available content server
- BGP routers may have multiple routes to the same prefix
- If router doesn't use 5-tuple hashing, the same flow might reach two different content servers
- Good thing: Everybody uses 5-tuple hashing :)
- No problem at all for UDP ⟶ Used for DNS

Content Delivery Networks (CDN)

CDN in practice
Design of a cache

Goal

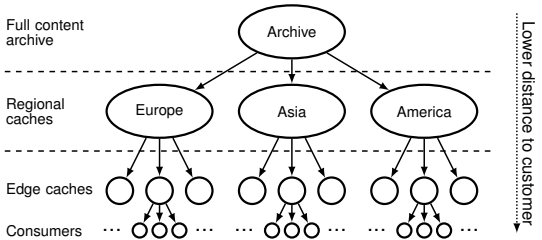
- Decrease latency of requests from clients
- Increase the hit-cache ratio

Architecture design

- Which objects need to be cached?
- Where to cache?
- Size of the cache?
- Caching strategy?
- How to update the cache?

Metrics

- Cache-hit / cache-miss



On a typical CDN server cluster serving web traffic over two days, 74% of the roughly 400 million objects in cache were accessed only once and 90% were accessed less than four times.

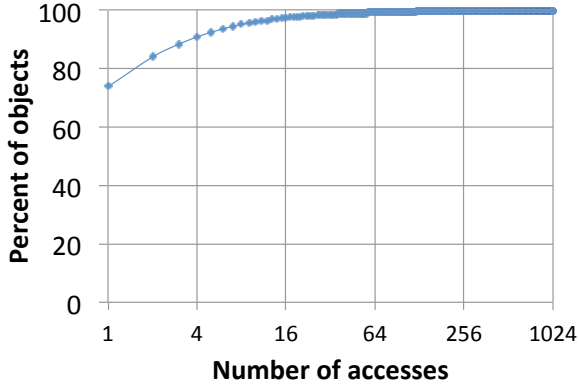


Figure 5: Access per file (source: [10])

CDN in practice
Cache filtering

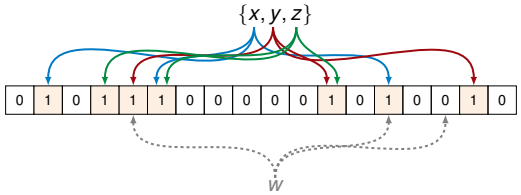
Problem

- Many objects are only accessed once
- Uses disk space without benefits
- More accessed objects could be evicted from cache (LRU)

Solution

- Use cache filtering
- Bloom filter (space efficient)
- Use filter to decide which objects to cache
- Example policy: only cache objects already seen once

CDN in practice
Bloom Filter as Cache Filter



Bloom Filter

- Stochastic data structure
- Map object hashes into table with binary entries
- Use multiple hash functions to decrease false positives

Challenges

- Speed: Use single hash and partition it into multiple hashes
- Size: Balance between false positives, number of hash functions, number of stored objects
  - Example: 100 million objects, 0.1% false positives, 10 hash functions → 175 MB
- Complexity: Take other metrics into consideration

Impact on performance

Byte hit rates increased when cache filtering was turned on between March 14th and April 24th because not caching objects that are accessed only once leaves more disk space to store more popular objects

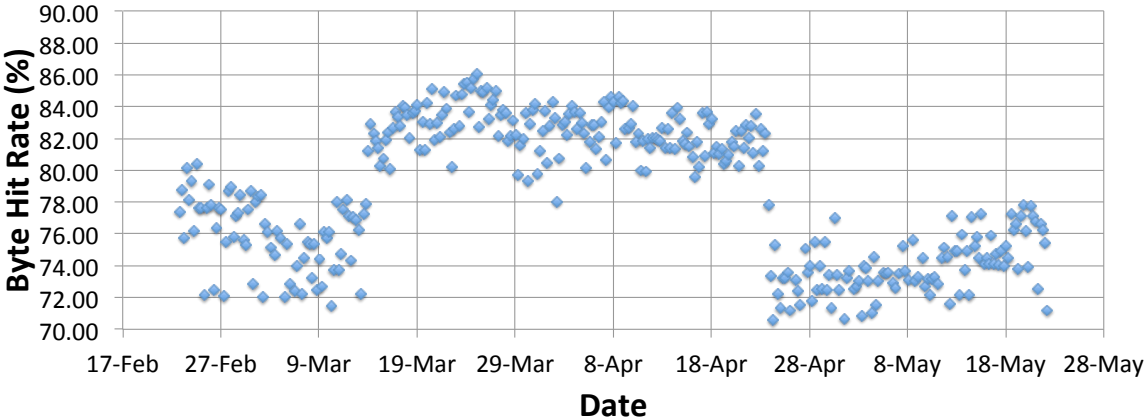


Figure 6: Byte hit rates (source: [10])

Turning on cache filtering decreases the rate of disk writes by nearly one half because objects accessed only once are not written to disk.

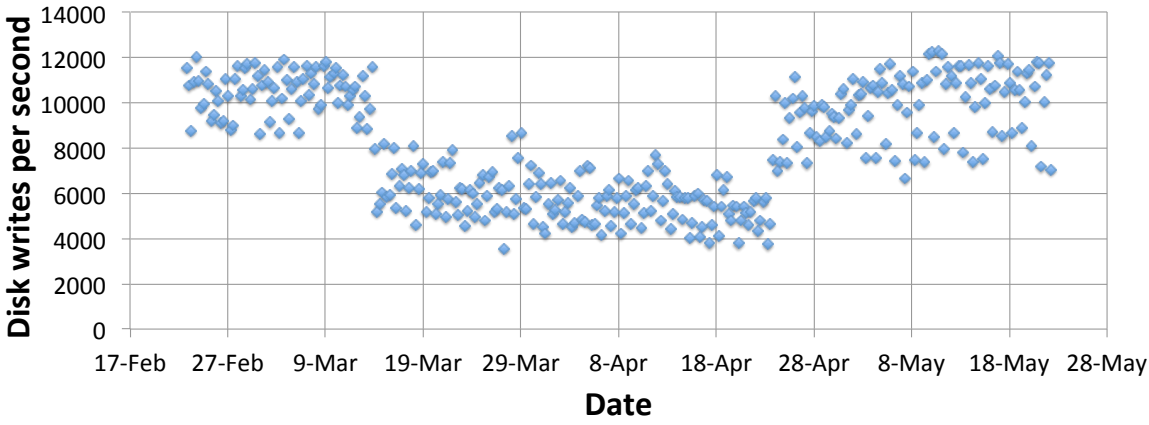


Figure 7: Disk writes (source: [10])

Content Delivery Networks (CDN)

Content Delivery Networks (CDN)

[1] Sandvine, Global Internet Phenomena Report, fall 2011, 2011.

[2] Sandvine, Global Internet Phenomena Report, fall 2012, 2012.

[3] Sandvine, Global Internet Phenomena Report, fall 2013, 2013.

[4] Sandvine, Global Internet Phenomena Report, fall 2014, 2014.

[5] Sandvine, Global Internet Phenomena Report, latin america & north america, 2015.

[6] Sandvine, Global Internet Phenomena Report, latin america & north america, 2016.

[7] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web," in Proceedings of the 29th Annual ACM Symposium on Theory of Computing, ser. STOC '97, ACM, 1997, pp. 654–663. DOI: 10.1145/258533.258660.

[8] B. Gellman and A. Soltani, "Nsa infiltrates links to yahoo, google data centers worldwide, snowden documents say," Wahington Post: https://www.washingtonpost.com/world/national-security/nsa-infiltrates-links-to-yahoo-google-data-centers-worldwide-snowden-documents-say/2013/10/30/e51d661e-4166-11e3-8b74-d89d714ca4dd_story.html, 2013.

[9] P. Moackapetris, Domain Names – Implementation and Specification, https://tools.ietf.org/html/rfc1035, 1987.

[10] B. M. Maggs and R. K. Sitaraman, "Algorithmic Nuggets in Content Delivery," SIGCOMM Comput. Commun. Rev., vol. 45, no. 3, pp. 52–66, Jul. 2015, ISSN: 0146-4833. DOI: 10.1145/2805789.2805800.