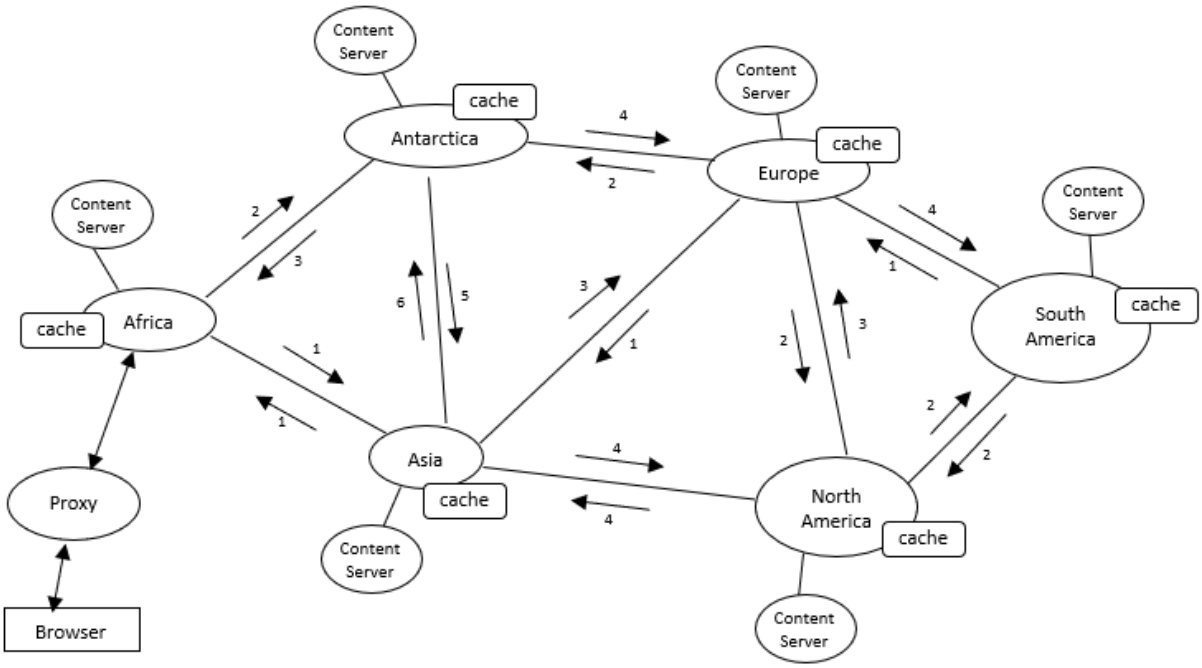
Content Delivery Network

Introduction:

Content Delivery Network is a geographically distributed group of servers which work together to provide fast delivery of internet content. A CDN does not host content but it does help cache content at network edge which improves website performance by means of increasing content availability, improving the website load times and reducing the band with costs. Thus, by implementing different cache schemes, providing the internet content closer to the end user.

Design:



Fig(a): Topology of CDN servers with proxy, delay information, content-servers, cache storage.

Designed CDN, is a network of six geographically interconnected regions namely africa, antarctica, asia, europe, northamerica and southamerica to deliver content relatively with high performance to end user. Each CDN server is connected to its neighbor CDN servers and will have delay information to send packets to its neighbors which will be provided in configuration file. Also, Each CDN node has its own content server connected to it which serves content and Each CDN node has cache storage to keep the contents for a period of time. A proxy server will be connected to one of the CDN servers and will take request directly from the client and forwards it.

Each CDN node will find the Round-Trip Time to its neighbor CDN servers and will use it to find the shortest paths with least latency to all the CDN servers using Distance Vector Routing algorithm. The response forwarding in the CDN network is done along the same path as the request forwarding. For the cache design, there will be two types of cache schemes,

1. '**type1**': Caches the response along each node as it is forwarded to the origin of request. (for instance, if the request to Europe content server is raised, it caches the content at Europe, Asia, and Africa CDNs in a single request.)
2. '**type2'**: Caches the response just on the first hop that does not have the content on the cache yet when the response is forwarded. (for instance, if the request to Europe content server is raised, it first caches at Europe CDN, then at Asia CDN in second request, then at Africa CDN in third request).

Implementation:

Measuring Delays: Each CDN node will have its configuration file in which it will have its geographical region information that identifies its location and IP address and port details to its content server, and also links details to its neighbour CDN servers. When the CDN starts running it will start measuring the Round-Trip Time to its neighbours by sending a GET request 'ping'. Both the CDNs will wait for a delay time to its neighbours mentioned in the config file while sending and receiving the request.

Distance Vector Routing Algorithm: After all CDN servers finish finding the RTT, they will start running the Distance Vector Routing Protocol to find the shortest path to all the CDN servers by Bellman Ford's principle $\mathbf{d_x(y) = min \{c(x,v) + d_v(y)\}}$**.** Where $d_x(y)$ = cost of least-cost path from X to Y. Each CDN will maintain a routing table and shares it with its neighbors and when each CDN node finds the least RTT to another node, routing table gets updated. After a node updates its routing table, it immediately shares the updated routing table with its neighbor CDN servers. When every node has updated routing table, it will have the shortest path to all the other CDN servers.

Proxy and Content Servers: Content servers will be connected to its CDN servers and serve the requested files without any delay. Proxy server is connected to one of the CDN server which is mentioned when it is getting started and will also have type of cache that it wants to use.

Caching: Once, every servers starts ready to serve the requests, Client will request a hostname (e. g. , http://europe/fileD.html)which indicates the region where the HTTP server is located. The browser connects to a proxy which forwards the request to the CDN server it is connected to. The CDN server then decides if it should fetch this file directly from the Content Server or through the CDN network, which is decided by type of cache scheme that is requested to use.

When the CDN server gets the request, it checks region info from hostname and if it is the designated region to serve the content, it will check the following conditions…

1. If the requested content is in cache memory, it writes the response back.
2. If the requested content is not in cache memory, it will forward the request to content server and gets the file if it exists there and caches the content for a 100 second of time based on type of cache.

If the CDN server is not the designated region to serve the content, it will check following conditions…

1. If the requested content is in cache memory, it writes the response back.
2. If the requested content is not in cache memory, it forwards the request to next CDN server which is in shortest path (gets this info from routing table) and caches the response for a 100 second of time based on type of cache.

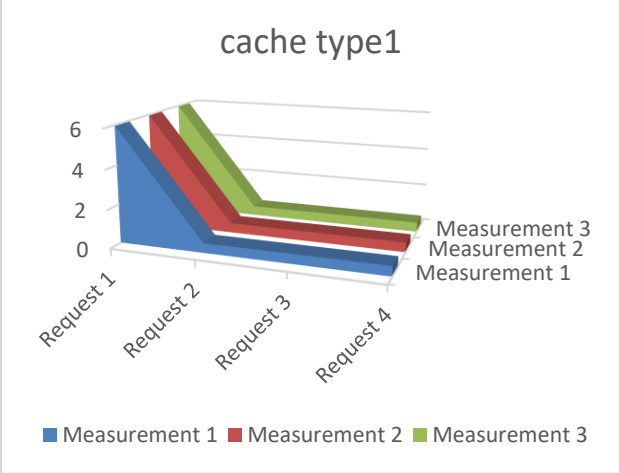
Logging: Whenever a CDN server gets a request it logs request info such as where the request is coming from, request headers and to where the next request heading to. And while getting a response it logs info about response headers and content cache information at CDN.

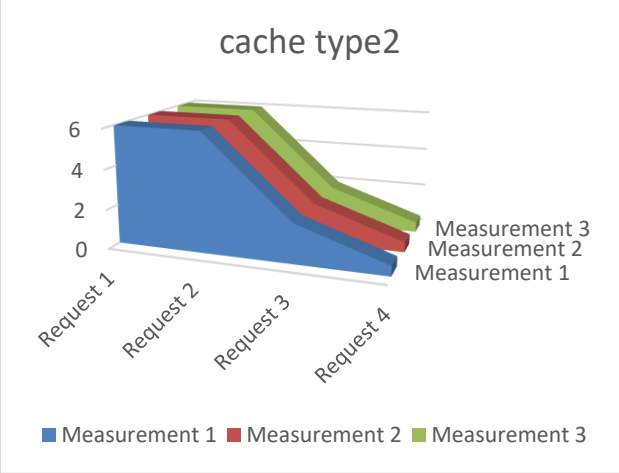
New Features: Few small features added are…

1. While caching the content, if two different region CDN server have same file named contents, file name conflicts arise even though two files are different. To resolve this issue, I have maintained individual cache storage for each region.
2. While serving the different types of files, maintaining mimetype for Content-type implemented dynamically without hardcoding.

Measurements: For Instance request from the client is **http://europe/fileD.html**, the shortest path would be… Browser---> Proxy ---> Africa ---> Asia ---> Europe ---> Content Server.

Type1 cache	Request 1(seconds)	Request 2(seconds)	Request 3 (seconds)	Request 4 (seconds)
Measurement 1	6	0.5	0.5	0.5
Measurement 2	6	0.5	0.5	0.5
Measurement 3	6	0.5	0.5	0.5
Type2 cache	Request 1(seconds)	Request 2(seconds)	Request 3 (seconds)	Request 4 (seconds)
Measurement 1	6	6	2	0.5
Measurement 2	6	6	2	0.5
Measurement 3	6	6	2	0.5





Conclusion:

From the measurements, What I observed is that both the type1 and type2 cache schemes are good in their own way. As for the type1 cache scheme, it will cache the response at all the CDN node in the response path for first request itself. The cache will last for 100 seconds of time, after that content in cache gets erased at a time at all the CDN nodes. It works good only for the short period frequent requests. As for the type2 cache scheme, it will cache the response for single hop at a time for a request in response path. Even the cache timeout is 100 seconds, it seems like it lasts for more than the time out at subsequent hops. So, I believe it works good for requests that are not frequent for the content.