بسمه تعالی

دانشگاه علم و صنعت ایران

School of Computer Engineering

Advanced Computer Networks

Application Layer, Video Streaming, and CDN

Part 4

Seyed Hamed Rastegar

Fall 1401

Streaming multimedia: DASH

*D*ynamic, *A*daptive *S*treaming over *H*TTP

- The demand for video streaming through the web has been increasing day by day.
  - ➢ Thus It is important to have common video streaming which supports all the formats and plays in all the devices like Apple devices, Android devices, etc.
- DASH (a.k.a MPEG-DASH) is one which aims to address the interoperability between various servers and devices.
- DASH is a streaming standard by MPEG that enables streaming of media content over the Internet delivered from conventional HTTP web servers.
- It provides interoperability between various servers and devices.

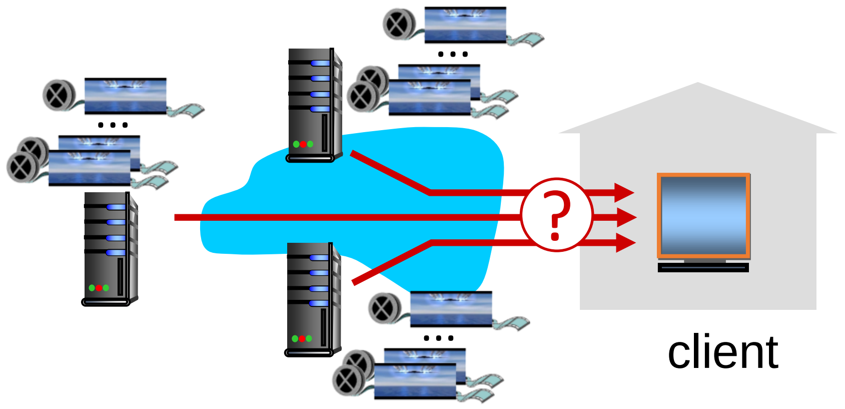
Streaming multimedia: DASH

- It is the first international standardized HTTP-based adaptive bit-rate streaming protocol which supports a broad range of devices like:
  - mobile phones, tablets, PCs, TVs, laptops, set-top boxes, game consoles and so on.
- DASH is an adaptive bitrate streaming technology where the multimedia content is captured and stored on an HTTP server and is delivered client players using HTTP.
- In DASH standard, the media content exists on the server in two parts.
  - **Media Presentation Description (MPD):** MPD describes a **manifest** of the available content, its various alternatives, their URL addresses, and other characteristics.
  - **Segments**: It contains the actual multimedia bit streams in the form of chunks, in single or multiple files.

Streaming multimedia: DASH

server:

- divides video file into multiple chunks
- each chunk encoded at multiple different rates
- different rate encodings stored in different files
- files replicated in various CDN nodes
- *manifest file:* provides URLs for different chunks

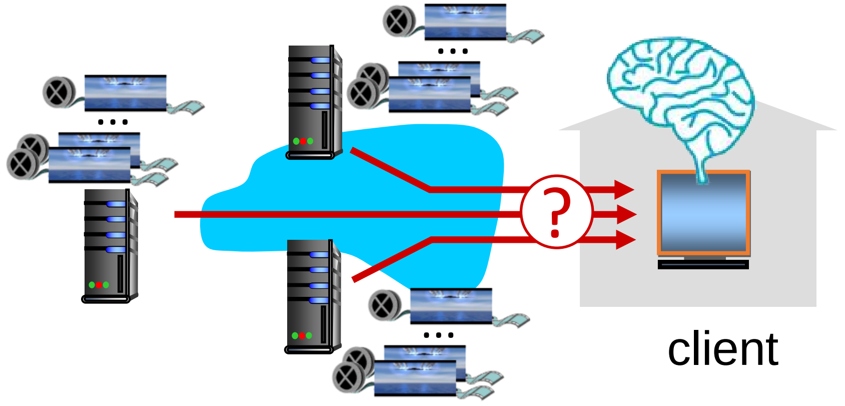


client:

- periodically estimates server-to-client bandwidth
- consulting manifest, requests one chunk at a time
  - chooses maximum coding rate sustainable given current bandwidth
  - can choose different coding rates at different points in time (depending on available bandwidth at time), and from different servers

Streaming multimedia: DASH

- *"intelligence"* at client: client determines
  - *when* to request chunk (so that buffer starvation, or overflow does not occur)
  - *what encoding rate* to request (higher quality when more bandwidth available)
  - *where* to request chunk (can request from URL server that is "close" to client or has high available bandwidth)



Streaming video = encoding + DASH + playout buffering

Streaming multimedia: DASH

- **Some Remarks**:

❖ While downloading chunks, the client also measures the received bandwidth and runs a **rate determination algorithm** to select the chunk to request next.

❖ Naturally, if the client has a lot of video buffered and if the measured receive bandwidth is high, it will choose a chunk from a high-rate version.

❖ And naturally if the client has little video buffered and the measured received bandwidth is low, it will choose a chunk from a low-rate version.

❖ DASH therefore allows the client to freely switch among different quality levels.
  - o <u>Challenge</u>: noticeable visual quality degradation
  - ➢ <u>Solution</u>: using multiple intermediate versions for smooth transition

Streaming multimedia: DASH

- **Some Remarks**:

❖ By dynamically monitoring the **available bandwidth** and **client buffer level**, and **adjusting the transmission rate** with version switching, DASH can often achieve continuous playout at the best possible quality level without frame freezing or skipping.

❖ Furthermore, since the client (rather than the server) maintains the intelligence to determine which chunk to send next, the scheme also improves **server-side scalability**.

❖ We note that for many implementations, the server not only stores many versions of the video **but also** separately stores many versions of the audio. Each audio version has its own quality level and bit rate and has its own URL.

- ➢ In these implementations, the client dynamically selects both video and audio chunks, and locally synchronizes audio and video playout.

Content distribution networks (CDNs)

challenge: how to stream content (selected from millions of videos) to hundreds of thousands of *simultaneous* users?

- *option 1:* single, large "mega-server"
- Problems:
  - long (and possibly congested) path to distant clients
    - for client far from the data center, server-to-client packets will cross many communication links and likely pass through many ISPs.
    - If one of these links provides a throughput that is less than the video consumption rate, the end-to-end throughput will also be below the consumption rate, resulting in annoying freezing delays for the user.
    - The problem got worse by increasing number of links.
  - single point of failure, point of network congestion
  - Content provider wastes bandwidth and also should pay the ISPs for the same popular video each time.
  - and also an important challenge: this solution *doesn't scale*

Content distribution networks (CDNs)

challenge: how to stream content (selected from millions of videos) to hundreds of thousands of *simultaneous* users?

- *option 2:* store/serve multiple copies of videos at multiple geographically distributed sites *(CDN)*

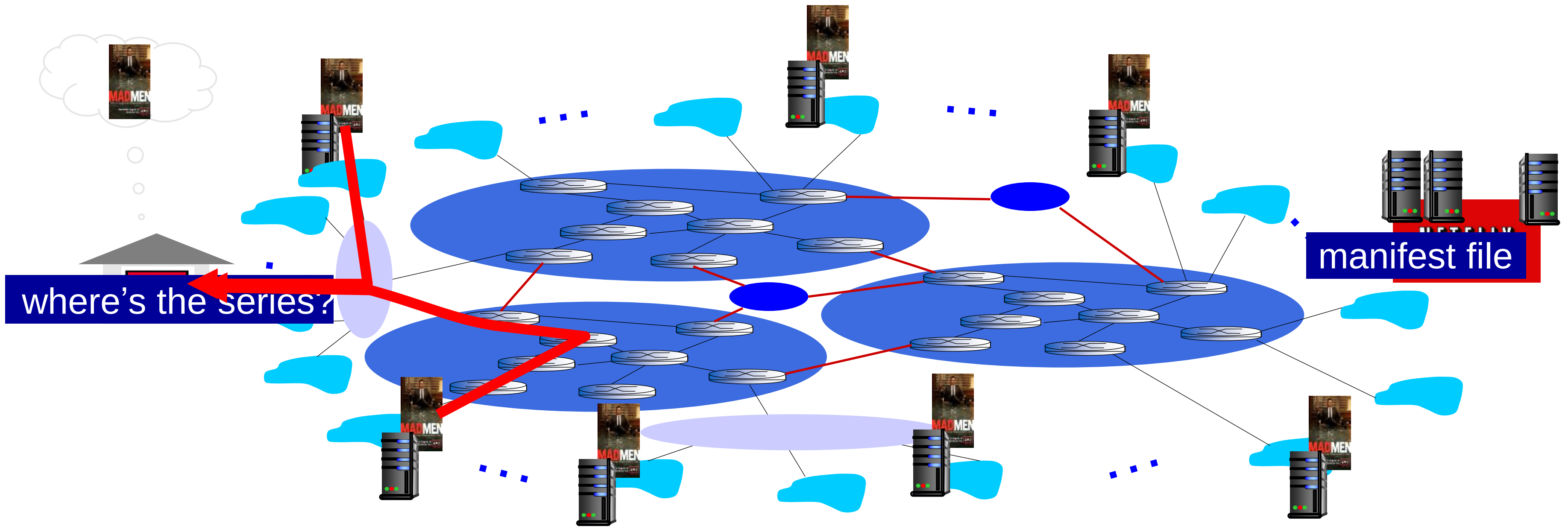
- The CDNs can be categorized as:
  - *private CDN*, that is, owned by the content provider itself; for example, Google's CDN distributes YouTube videos and other types of content
  - *third-party CDN,* that distributes content on behalf of multiple content providers; Akamai, Limelight and Level-3 all operate third-party CDNs.

Content distribution networks (CDNs)

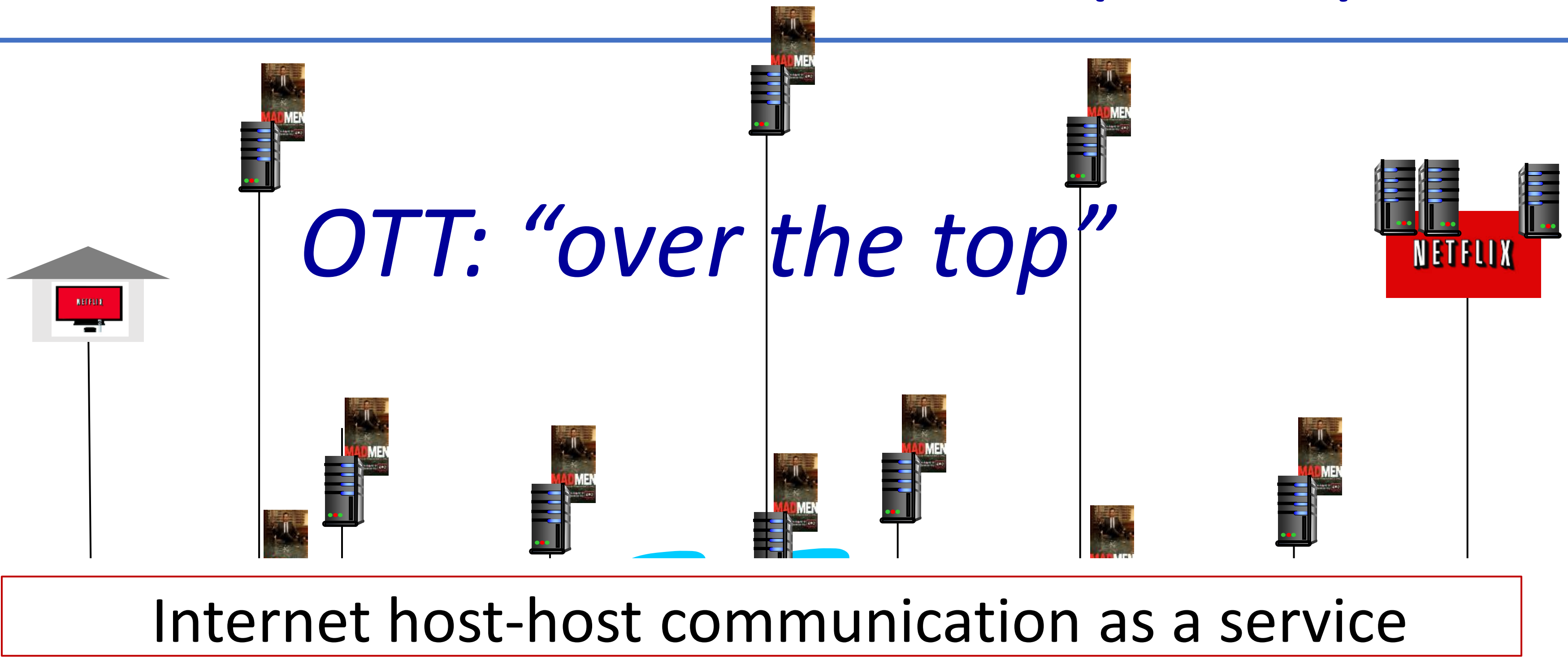
- CDNs typically adopt one of two different server placement philosophies
  - *enter deep:* push CDN servers deep into many access networks
    - close to users
    - Akamai: 240,000 servers deployed in > 120 countries (2015)
    - Because of this highly distributed design, the task of maintaining and managing the clusters becomes challenging.
  - *bring home:* smaller number (10's) of larger clusters in POPs near access nets
    - used by Limelight
    - *Instead of getting inside the access ISPs, these CDNs typically place their clusters in IXPs.*
    - *Compared with the enter-deep design philosophy, the bring-home design typically results in lower maintenance and management overhead, possibly at the expense of higher delay and lower throughput to end users.*

Content distribution networks (CDNs)

- CDN: stores copies of content (e.g. a series) at CDN nodes
- subscriber requests content, service provider returns manifest
  - using manifest, client retrieves content at highest supportable rate
  - may choose different rate or copy if network path congested



Content distribution networks (CDNs)



OTT challenges: coping with a congested Internet from the "edge"

- what content to place in which CDN node?
- from which CDN node to retrieve content? At which rate?

Content distribution networks (CDNs)

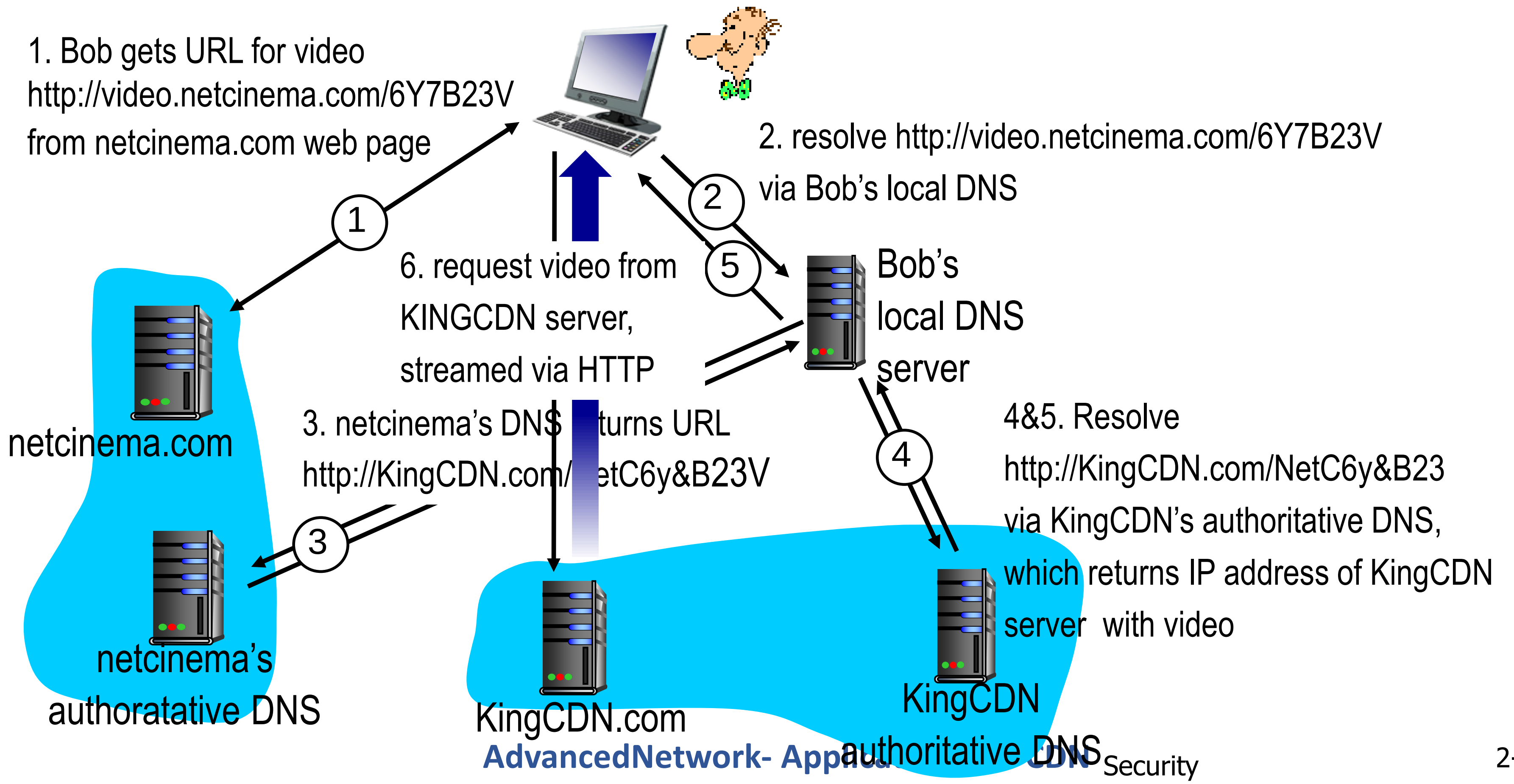
Some Remarks:

- Once its clusters are in place, the CDN replicates content across its clusters.
- The CDN may not want to place a copy of every video in each cluster, since some videos are rarely viewed or are only popular in some countries.
- In fact, many CDNs do not **push** videos to their clusters but instead use a simple **pull strategy**:
  - If a client requests a video from a cluster that is not storing the video, then the cluster retrieves the video (from a central repository or from another cluster) and stores a copy locally while streaming the video to the client at the same time.
- Similar to Web caching, when a cluster's storage becomes full, it removes videos that are not frequently requested

CDN Operation: a closer look

Bob (client) requests video http://video.netcinema.com/6Y7B23V

- video stored in CDN at http://KingCDN.com/NetC6y&B23V


1. Bob gets URL for video http://video.netcinema.com/6Y7B23V from netcinema.com web page
2. resolve http://video.netcinema.com/6Y7B23V via Bob's local DNS
3. netcinema's DNS returns URL http://KingCDN.com/NetC6y&B23V
4&5. Resolve http://KingCDN.com/NetC6y&B23 via KingCDN's authoritative DNS, which returns IP address of KingCDN server with video
6. request video from KINGCDN server, streamed via HTTP
1
2
3
4
5
Bob's local DNS server
netcinema.com
netcinema's authoratative DNS
KingCDN.com
KingCDN authoritative DNS

Content distribution networks (CDNs)

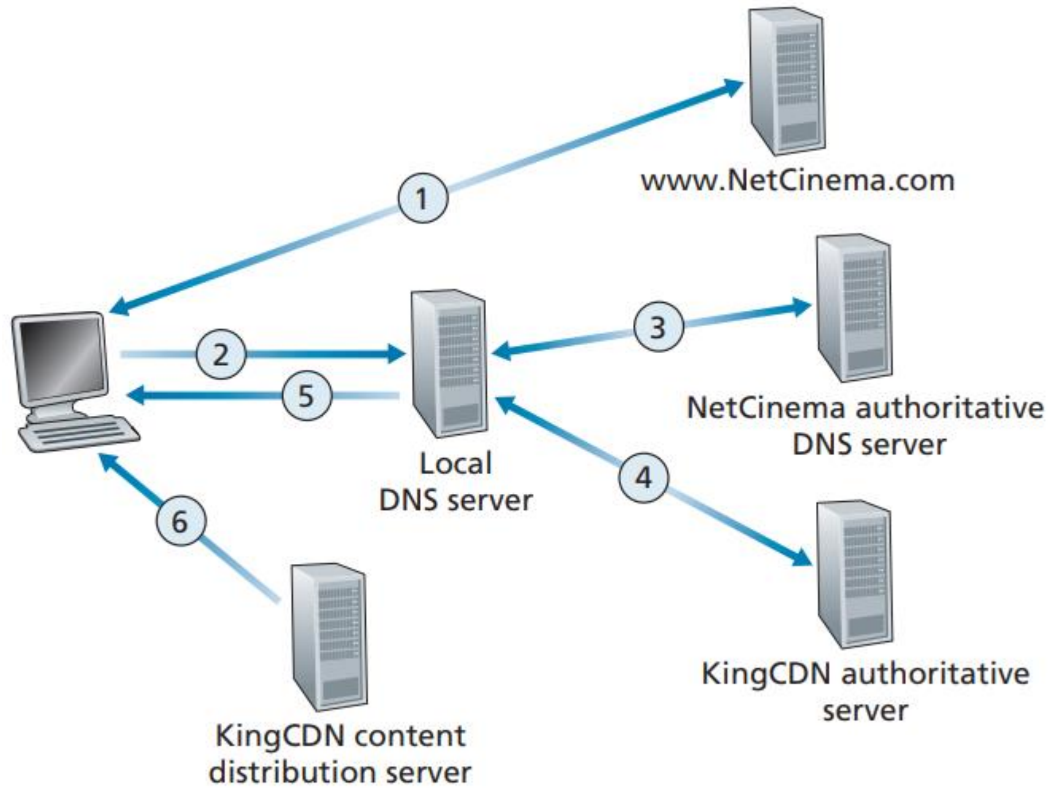
CDN Operation: (In more details)

- Example: Watching a video from NetCinema.com

Step 1: The user visits the Web page at NetCinema.

Step2: When the user clicks on a movie link (e.g. http://video.netcinema.com/6Y7B23V), the user's host sends a DNS query for video.netcinema.com.

Step 3: The user's Local DNS Server (LDNS) relays the DNS query to an authoritative DNS server for NetCinema, which observes the string "video" in the hostname video.netcinema.com. To "hand over" the DNS query to KingCDN, instead of returning an IP address, the NetCinema authoritative DNS server returns to the LDNS a hostname in the KingCDN's domain, for example, a1105.kingcdn.com.


1
www.NetCinema.com
2
5
3
Local DNS server
NetCinema authoritative DNS server
4
6
KingCDN authoritative server
KingCDN content distribution server

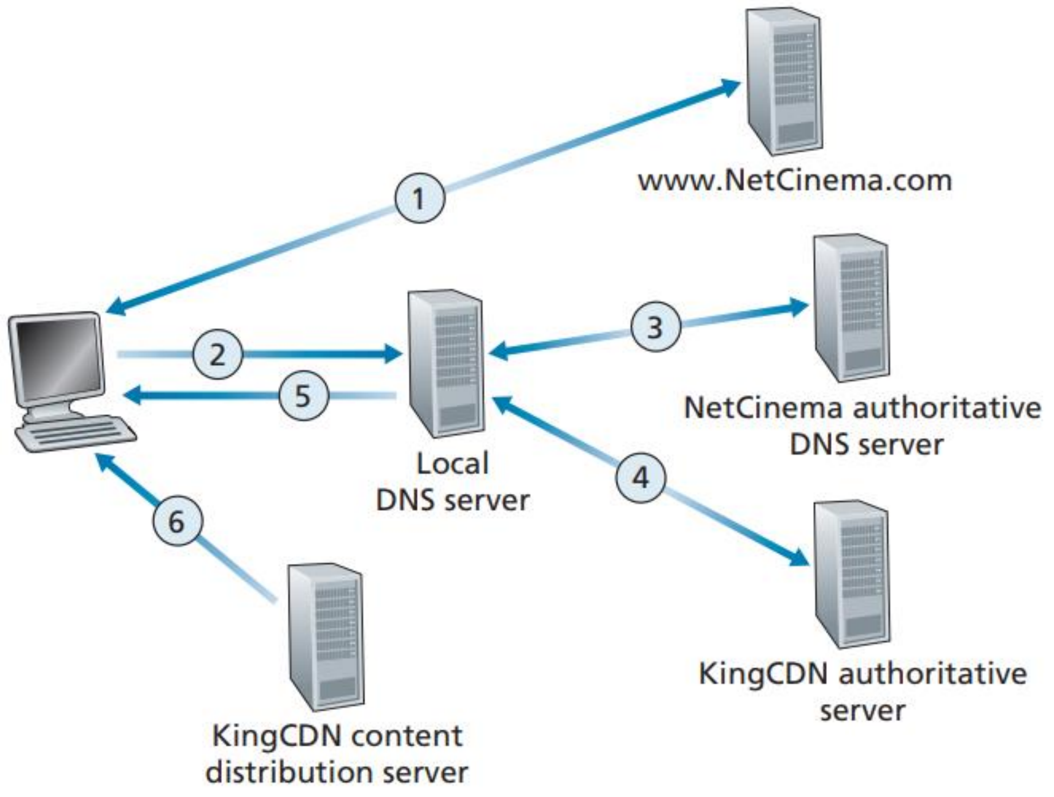
Content distribution networks (CDNs)

CDN Operation:

- Example (Cont'd)

Step 4: From this point on, the DNS query enters into KingCDN's private DNS infrastructure. The user's LDNS then sends a second query, now for a1105.kingcdn. com, and KingCDN's DNS system eventually returns the IP addresses of a KingCDN content server to the LDNS. It is thus here, within the KingCDN's DNS system, that the CDN server from which the client will receive its content is specified.

Step 5: The LDNS forwards the IP address of the content-serving CDN node to the user's host



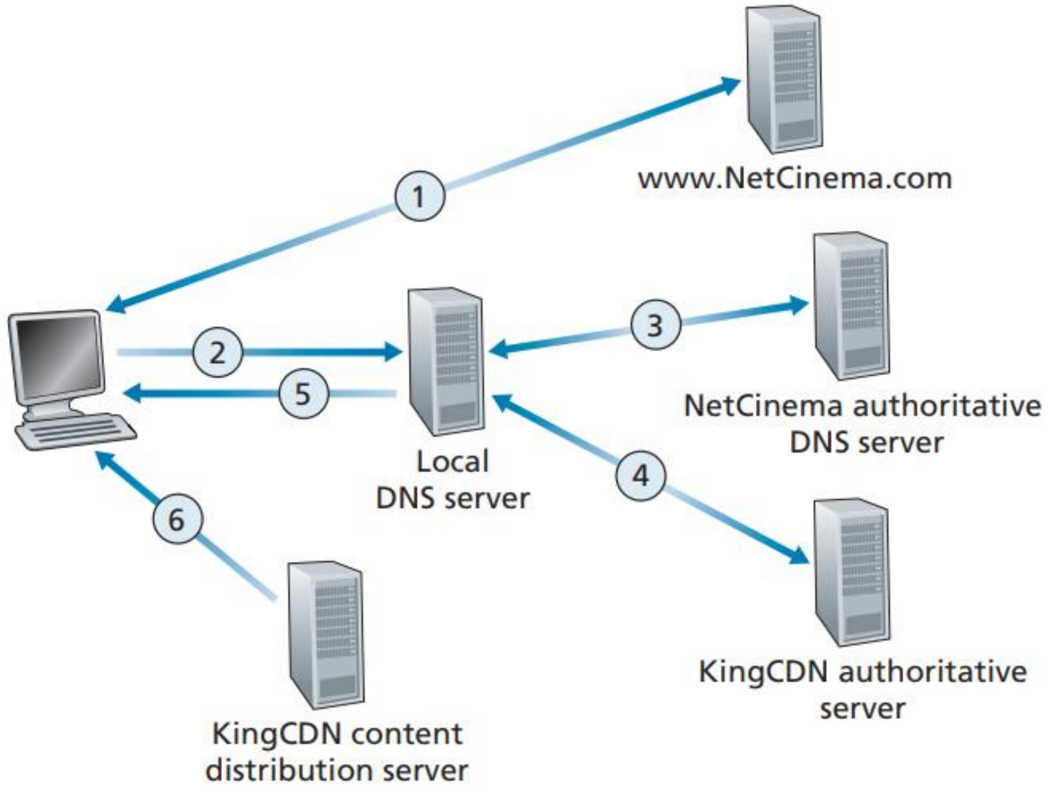
www.NetCinema.com
1
2
3
4
5
6
Local DNS server
NetCinema authoritative DNS server
KingCDN authoritative server
KingCDN content distribution server

Content distribution networks (CDNs)

CDN Operation:

- Example (Cont'd)

<u>**Step 6**</u>: Once the client receives the IP address for a KingCDN content server, it establishes a direct TCP connection with the server at that IP address and issues an HTTP GET request for the video. If DASH is used, the server will first send to the client a manifest file with a list of URLs, one for each version of the video, and the client will dynamically select chunks from the different versions


1
www.NetCinema.com
2
5
3
NetCinema authoritative DNS server
Local DNS server
4
6
KingCDN authoritative server
KingCDN content distribution server

Content distribution networks (CDNs)

Cluster Selection Strategies:

- At the core of any CDN deployment is a *cluster selection strategy*, that is, a mechanism for dynamically directing clients to a server cluster or a data center within the CDN.
- As we just saw, the CDN learns the IP address of the client's LDNS server via the client's DNS lookup.
- After learning this IP address, the CDN needs to select an appropriate cluster based on this IP address.
- CDNs generally employ proprietary cluster selection strategies.
  - ➢ We now briefly survey a few approaches, each of which has its own advantages and disadvantages.

Content distribution networks (CDNs)

Cluster Selection Strategies:

- One simple strategy is to assign the client to the cluster that is **geographically closest**.
- Using commercial geo-location databases (such as Quova and MaxMind), each LDNS IP address is mapped to a geographic location.
- When a DNS request is received from a particular LDNS, the CDN chooses the geographically closest cluster, that is, the cluster that is the fewest kilometers from the LDNS “as the bird flies.”

Content distribution networks (CDNs)

Cluster Selection Strategies: Pros and Cons of geographically closest method

- Such a solution can work reasonably well for a large fraction of the clients.
- However, for some clients, the solution may perform poorly, since the geographically closest cluster may not be the closest cluster in terms of the length or number of hops of the network path.
- Furthermore, a problem inherent with all DNS based approaches is that some end-users are configured to use remotely located LDNSs, in which case the LDNS location may be far from the client's location.
- Moreover, this simple strategy ignores the variation in delay and available bandwidth over time of Internet paths, always assigning the same cluster to a particular client.

Content distribution networks (CDNs)

Cluster Selection Strategies

- In order to determine the best cluster for a client based on the current traffic conditions, CDNs can instead perform **periodic real-time measurements** of delay and loss performance between their clusters and clients.
- For instance, a CDN can have each of its clusters periodically send probes (for example, ping messages or DNS queries) to all of the LDNSs around the world.
- One drawback of this approach is that many LDNSs are configured to not respond to such probes.

Content distribution networks (CDNs)

Case Studies: Google's Network Infrastructure

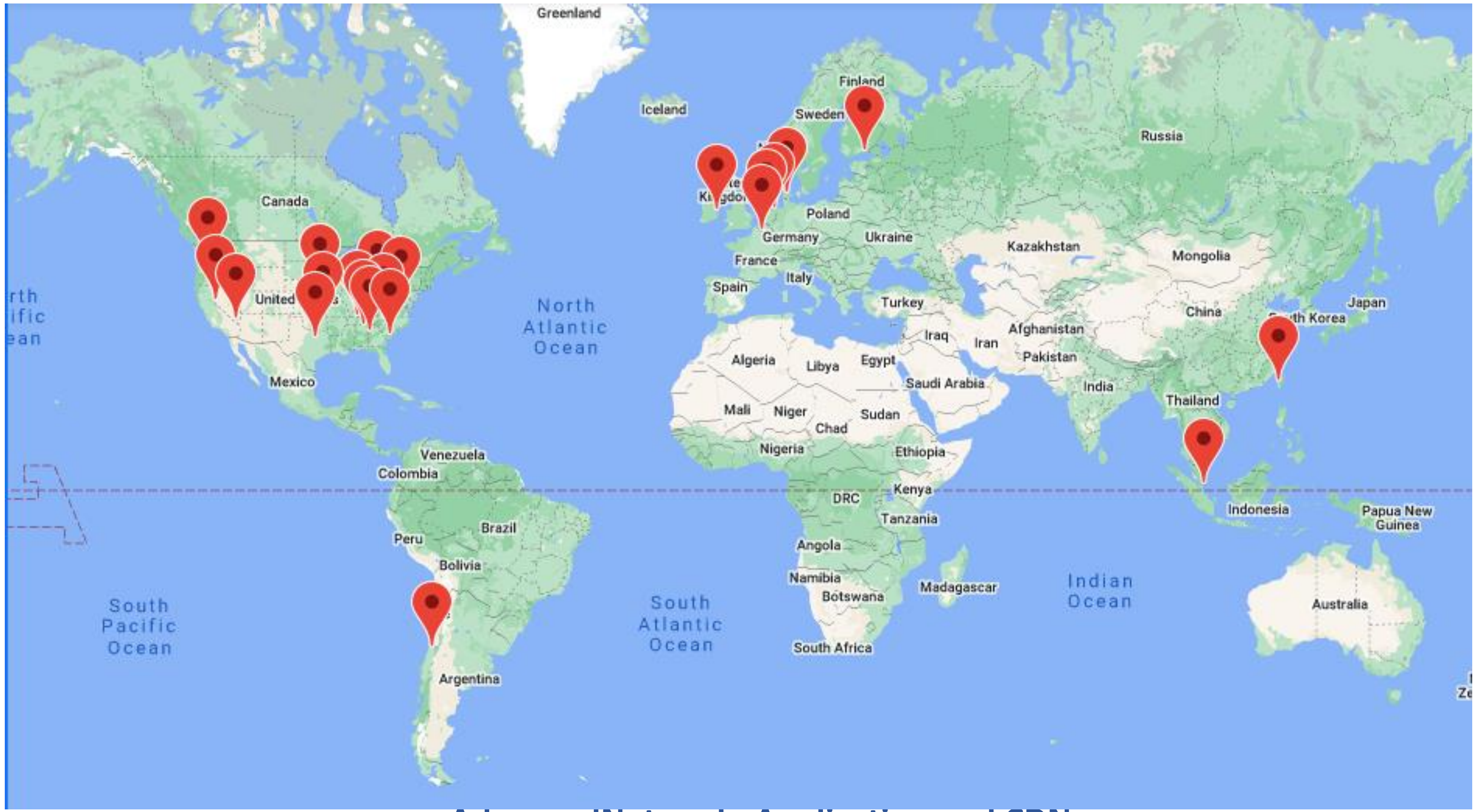
- To support its vast array of services—including search, Gmail, calendar, YouTube video, maps, documents, and social networks—Google has deployed an extensive private network and CDN infrastructure.

- Google's CDN infrastructure has three tiers of server clusters:

  - Nineteen "mega data centers" in North America, Europe, and Asia, with each data center having on the order of 100,000 servers. These mega data centers are responsible for serving dynamic (and often personalized) content, including search results and Gmail messages.

Content distribution networks (CDNs)

Case Studies: Google's Network Infrastructure



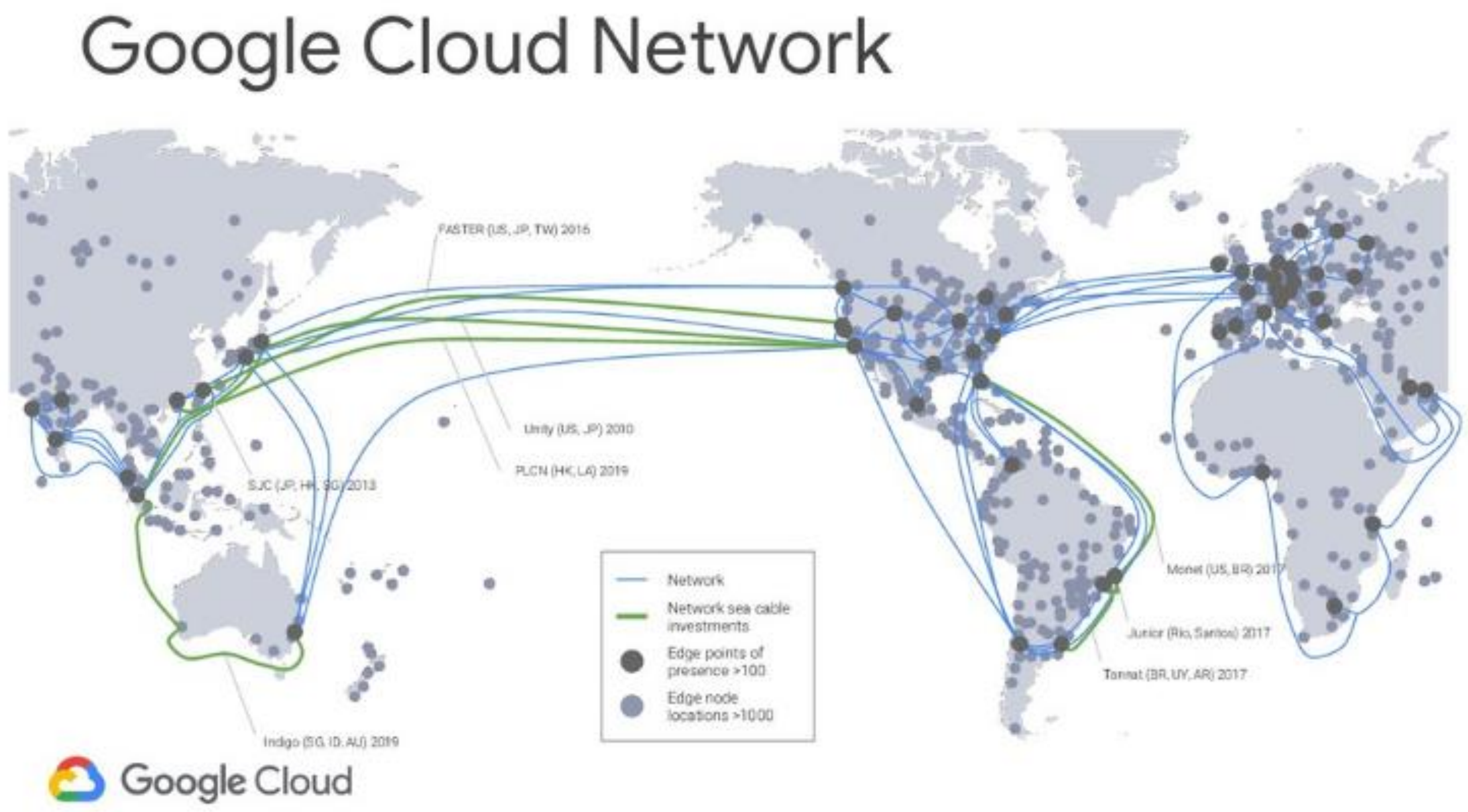
Content distribution networks (CDNs)

Case Studies: Google's Network Infrastructure

- Google's CDN infrastructure has three tiers of server clusters: (Cont'd)
  - With about 90 clusters in IXPs scattered throughout the world, with each cluster consisting of hundreds of servers. These clusters are responsible for serving static content, including YouTube videos.
  - Many hundreds of "enter-deep" clusters located within an access ISP. Here a cluster typically consists of tens of servers within a single rack. These enter-deep serve static content, including the static portions of Web pages that embody search results.

Content distribution networks (CDNs)

Case Studies: Google's Network Infrastructure



Content distribution networks (CDNs)

Case Studies: Google's Network Infrastructure

- All of these data centers and cluster locations are networked together with Google's own private network.

- When a user makes a **search query**,
  - often the query is **first** sent over the local ISP to a **nearby enter-deep cache**, from where **the static content** is retrieved;
  - while providing the static content to the client, the nearby cache also forwards the query over **Google's private network** to one of the mega data centers, from where the **personalized search** results are retrieved.

- For a YouTube video, the **video itself** may come from one of the **bring-home caches**, whereas **portions of the Web page** surrounding the video may come from the nearby **enter-deep cache**, and the **advertisements** surrounding the video come from **the data centers**.

Content distribution networks (CDNs)

Case Studies: Netflix

- As of 2020, Netflix is the leading service provider for online movies and TV series in North America.

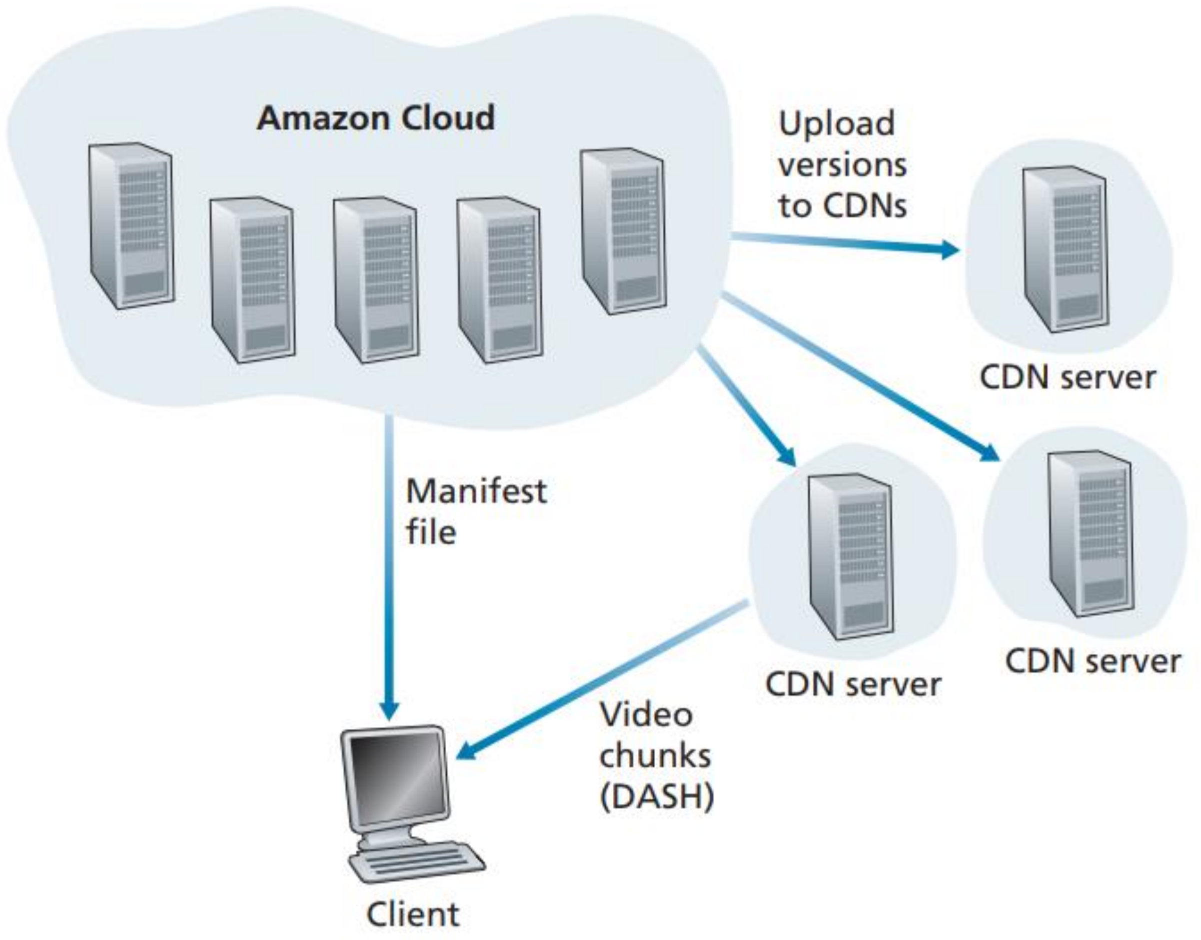
- Netflix video distribution has two major components:
  - the Amazon cloud
  - its own private CDN infrastructure.

- Netflix has a Web site that handles numerous functions, including user registration and login, billing, movie catalogue for browsing and searching, and a movie recommendation system.

Content distribution networks (CDNs)

Case Studies: Netflix

- As shown in the Figure, this Web site (and its associated backend databases) run entirely on Amazon servers in the Amazon cloud.



Content distribution networks (CDNs)

Case Studies: Netflix

- Additionally, the Amazon cloud handles the following critical functions:
  - **Content ingestion**. Before Netflix can distribute a movie to its customers, it must first ingest and process the movie. Netflix receives studio master versions of movies and uploads them to hosts in the Amazon cloud.
  - **Content processing**. The machines in the Amazon cloud create many different formats for each movie, suitable for a diverse array of client video players running on desktop computers, smartphones, and game consoles connected to televisions. A different version is created for each of these formats and at multiple bit rates, allowing for adaptive streaming over HTTP using DASH.
  - **Uploading versions to its CDN**. Once all of the versions of a movie have been created, the hosts in the Amazon cloud upload the versions to its CDN.

Content distribution networks (CDNs)

Case Studies: Netflix

- When Netflix first rolled out its video streaming service in 2007, it employed three third-party CDN companies to distribute its video content.
- Netflix has since created its own private CDN, from which it now streams all of its videos.
- To create its own CDN, Netflix has installed server racks both in IXPs and within residential ISPs themselves.
- Netflix currently has server racks in over 200 IXP locations. There are also hundreds of ISP locations housing Netflix racks.

Content distribution networks (CDNs)

Case Studies: Netflix

- Each server in the rack has several 10 Gbps Ethernet ports and over 100 terabytes of storage.
- The number of servers in a rack varies: IXP installations often have tens of servers and contain the entire Netflix streaming video library, including multiple versions of the videos to support DASH.
- Netflix does not use pull-caching to populate its CDN servers in the IXPs and ISPs. Instead, Netflix distributes by pushing the videos to its CDN servers during off-peak hours.
- For those locations that cannot hold the entire library, Netflix pushes only the most popular videos, which are determined on a day-to-day basis.

Content distribution networks (CDNs)

Case Studies: Netflix

- Let’s take a closer look at the interaction between the client and the various servers that are involved in movie delivery.
- As indicated earlier, the Web pages for browsing the Netflix video library are served from servers in the Amazon cloud.
- When a user selects a movie to play, the Netflix software, running in the Amazon cloud, first determines which of its CDN servers have copies of the movie.
- Among the servers that have the movie, the software then determines the “best” server for that client request.
- If the client is using a residential ISP that has a Netflix CDN server rack installed in that ISP, and this rack has a copy of the requested movie, then a server in this rack is typically selected. If not, a server at a nearby IXP is typically selected.

Content distribution networks (CDNs)

Case Studies: Netflix

- Once Netflix determines the CDN server that is to deliver the content, it sends the client the IP address of the specific server as well as a manifest file, which has the URLs for the different versions of the requested movie.

- The client and that CDN server then directly interact using a proprietary version of DASH.

- Netflix uses chunks that are approximately four-seconds long.

- While the chunks are being downloaded, the client measures the received throughput and runs a rate-determination algorithm to determine the quality of the next chunk to request.

Content distribution networks (CDNs)

Case Studies: Netflix (Overview)

- Netflix embodies many of the key principles discussed earlier in this section, including adaptive streaming and CDN distribution.
- However, because Netflix uses its own private CDN, which distributes only video (and not Web pages), Netflix has been able to simplify and tailor its CDN design.
- In particular, Netflix does not need to employ DNS redirect, to connect a particular client to a CDN server; instead, the Netflix software (running in the Amazon cloud) directly tells the client to use a particular CDN server.
- Furthermore, the Netflix CDN uses push caching rather than pull caching: content is pushed into the servers at scheduled times at off-peak hours, rather than dynamically during cache misses.

Content distribution networks (CDNs)

Case Studies: YouTube

- With hundreds of hours of video uploaded to YouTube every minute and several billion video views per day, YouTube is indisputably the world’s largest video sharing site.
- YouTube began its service in April 2005 and was acquired by Google in November 2006. Although the Google/YouTube design and protocols are proprietary, through several independent measurement efforts a basic understanding about how YouTube operates has been gained.
- As with Netflix, YouTube makes extensive use of CDN technology to distribute its videos.
- Similar to Netflix, Google uses its own private CDN to distribute YouTube videos, and has installed server clusters in many hundreds of different IXP and ISP locations.

Content distribution networks (CDNs)

Case Studies: YouTube

- From these locations and directly from its huge data centers, Google distributes YouTube videos.
- Unlike Netflix, however, Google uses pull caching, and DNS redirect.
- Most of the time, Google's cluster-selection strategy directs the client to the cluster for which the RTT between client and cluster is the lowest; however, in order to balance the load across clusters, sometimes the client is directed (via DNS) to a more distant cluster.

Content distribution networks (CDNs)

Case Studies: YouTube

- YouTube employs HTTP streaming, often making a small number of different versions available for a video, each with a different bit rate and corresponding quality level.
- YouTube does not employ adaptive streaming (such as DASH), but instead requires the user to manually select a version.
- Not only are YouTube videos streamed from server to client over HTTP, but YouTube uploaders also upload their videos from client to server over HTTP.
- YouTube processes each video it receives, converting it to a YouTube video format and creating multiple versions at different bit rates. This processing takes place entirely within Google data centers.