Content Distribution Networks

1. Vivesecting youtube: An active measurement study
   INFOCOM 2012
2. Unreeling netflix: understanding and improving multi-CND movie delivery
   INFOCOM 2012

Domain Name System (DNS): Part I

- Problem statement:
  - Average brain can easily remember 10 digits
  - On average, IP addresses have 10.28 digits
  - We need an easier way to remember IP addresses

- Solution:
  - Use alphanumeric names to refer to hosts
  - Add a distributed, hierarchical protocol (called DNS) to map between alphanumeric host names and IP addresses
  - We call this Address Resolution

Domain Name Space

- Root DNS servers
- Top level domain servers
- Country Domains
- Generic Domains

DNS Root name servers

- Responsible for root zone
- About 12 root name servers worldwide
- Local name servers are configured to contact well-known root servers
Domain Name Service

- The domain name service consists of
- Domain name space
- Name servers
  - In each zone, there is a primary name server and one or more secondary name servers
- Name servers contain two kinds of address mappings:
  - Authoritative mappings: For hosts within the zone
  - Cached mappings: For previously requested mappings to hosts not in the zone
- Resolvers
  - Programs that extract information from name servers in response to client requests

Domain Name Hierarchy

DNS Protocol

- When client wants to know an IP address for a host name
  - Client sends a DNS query to the local name server in its zone
  - If name server contains the mapping, it returns the IP address to the client
  - Otherwise, the name server forwards the request to the root name server
  - The request works its way down the tree toward the host until it reaches a name server with the correct mapping

DNS Protocol Example

Scenario: recursive processing
remus.rutgers.edu tries to resolve an IP address for venus.cs.yale.edu using a recursive query
ns-lcsr.rutgers.edu obtains the IP address for cs.yale.edu
DNS Protocol

Another Example

Scenario: Iterative query
remus.rutgers.edu tries to resolve an IP address for venus.cs.yale.edu using an iterative query.
Remus sends a request to cs.yale.edu to get the IP address.

DNS Caching

- Going to the root server and then down the tree every time we need to resolve an address is inefficient.
- Introduce address caching at name servers.
  - Store host-to-IP-address mappings from recently requested host names at name server.
  - When the same address is requested later, use the cached version at the local name server instead of recursively querying other name servers again.

DNS records

- RR format: (class, name, value, type, ttl)
- Type=A Name is hostname, value is IP address
- Type=NS name is domain, value is authoritative name server for this domain
- Type=CNAME name is an alias name
- Type=MX value is hostname for mail server
DNS records

- All information stored in resource records (RR): a four tuple
  - `<name, value, type, TTL>`
    - If type = A then name = hostname, value = IP address
      `<rags.rutgers.edu,168.14.2.4,A>`
    - If type = NS then name = domain, value = name of DNS server
      `<ns.lcsr.rutgers.edu,128.64.13.2,NS>`
    - If type = CNAME then name = generic host name, value is the actual hostname
      `<cs.rutgers.edu,aramis.rutgers.edu,CNAME>`
    - If type = MX, then value is the mail server for the host name in the type field
      `<cs.rutgers.edu,dragon.rutgers.edu,MX>`

DNS Message

In DNS, all communications use a single format called a message. The top level format of message is divided into 5 sections (some of which are empty in certain cases) RR (Resource record)

<table>
<thead>
<tr>
<th>RR (Resource record)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td></td>
</tr>
<tr>
<td>Question</td>
<td>the question for the name server</td>
</tr>
<tr>
<td>Answer</td>
<td>RRs answering the question</td>
</tr>
<tr>
<td>Authority</td>
<td>RRs pointing toward an authority</td>
</tr>
<tr>
<td>Additional</td>
<td>RRs holding additional information</td>
</tr>
</tbody>
</table>

The answer section contains RRs that answer the question; the authority section contains RRs that point toward an authoritative name server; the additional records section contains RRs which relate to the query, but are not strictly answers for the question.

DNS Message format

- Clients communicate with DNS servers using either TCP or UDP on port 53

<table>
<thead>
<tr>
<th>Transaction Identification</th>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Questions</td>
<td>Number of Answer RRs</td>
</tr>
<tr>
<td>Number of Authoritative RRs</td>
<td>Number of Additional RRs</td>
</tr>
</tbody>
</table>

DNS Message Fields

- **Transaction Identification**: Random number used to match client queries with name server responses
- **Flags**:

<table>
<thead>
<tr>
<th>QR</th>
<th>Opcode</th>
<th>AA</th>
<th>TC</th>
<th>RD</th>
<th>RA</th>
<th>RCODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Query</td>
<td>1</td>
<td>Standard query</td>
<td>1</td>
<td>Inverse query</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>Response</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Authoritative answer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Translated DNS packet</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Recursion desired</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Recursion available</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Return code</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>No error</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Name error</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

0           1       2       3       4       5       6       7
0 15 16 31
DNS Message Fields (cont’d)

- **Number of Questions**: Number of DNS queries in the packet
- **Number of Answer RRs**: Number of DNS responses in the packet
- **Number of Authoritative RRs**: Number of authoritative DNS responses in the packet
- **Number of Additional RRs**: Number of other DNS responses in the packet (usually contains other DNS servers in domain)
- **Questions & Answers**: Variable length fields to store DNS queries and DNS server responses

Content distribution networks: Part II

- CDNs leverage strategically placed distributed caches for load balancing and web request redirection systems. Results in
- Scales the server requirements
- Efficient delivery of Web content.
- Multiple replicas of content being hosted.
- Request from client routed to “good” replica.
  - **Network proximity** – Typically derived through network routing tables, such as BGP.
  - **Geographical proximity** – Used to redirect all users within a certain region to a POP.
  - **Response time** – The redirector could maintain knowledge of response times to the POPs, thus always redirecting to the one with the lowest latency.
  - **User type** – e.g., paying customer gets access to better service than non-paying. The user information could be based upon a cookie retrieved from the client system, or an authentication process.

CDN terms

- **Origin server**: Server that holds the authoritative copy of the content
- **CDN server**: A replica server owned by the CDN provider
- **CDN name server**: A DNS like name server used for redirection
- **Client**

Types of CDN

- **DNS redirection**
  - Full site Content delivery
    - All requests are redirected by DNS to CDN server
  - Partial site content delivery
    - URL are modified to be resolved by CDN name server
- **URL rewriting**
- **Hybrid scheme**
- **CDN used by ISPs to improve end-user latency**
- **Caches used to reduce bandwidth required**
Example CDN: Akamai

- Akamai deploys content servers around the world.
- Use information about network conditions to direct requests to appropriate server.
- Operate on behalf of the information provider (CNN, abcnews etc).

Others include limelight, level 3, Amazon cloudFront

Operation

1. GET <url>
2. Document+embedded Object url’s point to Akamai network
3. Request for embedded Objects.
4. URL for object is rewritten as Hostpart: Akamaicontentserver/Originserver
   www.cnn.com/obama.gif --> a44.g.akamai.net/www.cnn.com/obama.gif

Akamaizing web page

- Content providers need to modify content by including akamai redirectors
- Akamai resource locators.
- Mapping of original URL into name within Akamai network.
  - Target server(s).
  - Content provider id.
  - Original URL.
  - Used to get object from origin server (first time, and?).
- Akamaizing Web site:
  - Transform URLs into ARLs.
  - Automated using "launcher" tool.
  - Can be integrated into publishing environment.

Using DNS...

- CDNs can use DNS to map origin server name to CDN server IP address.
- 2-level hierarchy: high-level (HL) and low-level (LL) DNS Web servers.
Using DNS

- Redirect.aff.akamai.net
- Root name server
- List of akamai HLDNS servers
- HLDNS
  - TTL=Days
- Local name server
- List of LLDNS
  - TTL=30s-1min
- Close-by server
  - TTL=30min

CDN benefits

- Highly scalable
- High speed connections to content servers
- Fault tolerant
- Clever use of DNS hack for redirection

Part III: Case Studies

- References
- How do you “tube”? ACM SIGMETRICS 2011
- Vivisecting youtube, INFOCOM 2012
- Slides from conference presentations

Video Delivery Basics

1. HTTP GET request for video URL
2. HTTP reply containing html to construct the web page
3. HTTP GET request for FLV stream
4. HTTP reply and a link to stream the FLV stream
server selection
- File server mapping done in two ways
- Dynamic DNS resolution
  - DNS returns different IP addresses for a given DNS name
  - http redirect
    - Use http status code 3xx [With new URL]
    - Web browser does a get from the new site
- http redirect
- DNS returns different IP addresses for a given DNS name
- Video ID namespace
  - Each video identifies by 11 character string
  - \{a-Z,0-9, , -\} giving \(64^{11}\) video ID space
  - E.g., UIREuv2UcAN
  - Video names are uniformly distributed over the ID space
  - Each Video ID mapped to 192 names/URLs

video ID namespace
- Each video identifies by 11 character string
- \{a-Z,0-9, , -\} giving \(64^{11}\) video ID space
- E.g., UIREuv2UcAN
- Video names are uniformly distributed over the ID space
- Each Video ID mapped to 192 names/URLs

DNS namespace
- Each video identifies by 11 character string
- \{a-Z,0-9, , -\} giving \(64^{11}\) video ID space
- E.g., UIREuv2UcAN
- Video names are uniformly distributed over the ID space
- Each Video ID mapped to 192 names/URLs

Cache hierarchy namespace
- Five "anycast" namespaces
  - lscache: "visible" in the URLs
    - each ns representing fixed number of "logical" servers
    - logical servers mapped to physical servers via DNS
    - Pick the "best" IP (geographical proximity)
- 2 "unicast" namespaces
  - rhost: google locations
  - rhostisp: ISP locations
    - mapped to a single server
  - Video id mapped to a fixed hostname
Cache hierarchy

Three levels of caches
- Video id can return domain name of any level cache
- Mostly Primary cache
- Sometimes secondary, tertiary
- Cache miss, http redirect to tertiary cache
- Tertiary cache (5 of them) have all of youtube content
  - Full replication only at this level

Physical Cache Hierarchy & Locations

~ 50 cache locations
- ~40 primary locations
  - including ~10 non-Google ISP locations
- 8 secondary locations
- 5 tertiary locations

Geo-locations using
- city codes in unicast hostnames, e.g., r1.sjc01g01.c.youtube.com
- low latency from PLnodes (< 3ms)
- clustering of IP addresses using latency matrix

Static Mappings between and among Video Id Space and Namespaces

- Video id’s are mapped to anycast namespaces via a fixed mapping
  - video id space “divided” evenly among hostnames in each namespace
- Mappings among namespaces are also fixed
  - 1-to-1 between primary and secondary (192 hostnames each)
  - 3-to-1 between primary or secondary to tertiary (64 names)

- Advantages:
  - URLs for videos can be generated w/o info about physical video servers
  - thus web servers can operate “independently” of video servers

HTTP Redirections

Cache misses vs Load Balancing

- Redirection probability for “hot” vs “cold” videos
- Redirections via unicast names are used mostly among servers within same location:
  - likely for load balancing
Cache Misses and Load Balancing

- Cache misses mostly handled via “background fetch”
  - 3 ms vs 30+ ms
- Sometimes via HTTP redirections
  - also used for load-balancing
  - redirections follows cache hierarchy
    - cache misses in an ISP primary cache locations re-directed to a Google primary cache location
    - up to 9 re-directions (tracked via a counter in URL)

YouTube Video Delivery Dynamics: Summary

- Locality-aware DNS resolution
- Handling load balancing & hotspots
  - DNS change
  - Dynamic HTTP redirection
    - local vs. higher cache tier
- Handling cache misses
  - Background fetch
  - Dynamic HTTP redirection

YouTube summary
- Infrastructure determines service location
- Files fully replicated in datacenters
- Primary and secondary caches used to service requests
- Video file Namespace hashed to primary and secondary caches
- Dynamic load balancing
- DNS anycast
- HTTP Redirect
- Geographic proximity
- Server load
- Network load

What Makes Netflix Interesting?
- Commercial, feature-length movies and TV shows
  - and not free; subscription-based
- Nonetheless, Netflix is huge!
  - 25 million subscribers and ~20,000 titles (and growing)
  - consumes 30% of peak-time downstream bandwidth in North America
- A prime example of cloud-sourced architecture
  - Maintains only a small “in-house” facility for key functions
    - e.g., subscriber management (account creation, payment, …)
  - Majority of functions are sourced to Amazon cloud (EC2/S3)
    - user authentication, video search, video storage, …
  - DNS service is sourced to UltraDNS
  - Leverate multiple CDNs for video delivery
    - Akamai, Level 3 and Limelight
- Can serve as a possible blue-print for future system design
  - (nearly) “infrastructure-less” content delivery – from Netflix’s POVs
  - minimize capex/opex of infrastructure, but may lose some “control” in terms of system performance, …
What are the Key Questions?

- What is overall Netflix architecture?
  - what components is out/cloud-sourced?
  - namespace management & interaction with CDNs
- How does it perform rate adaption & utilize multiple CDNs?
  - esp., CDN selection strategy
- Does Netflix offer best user experience based upon the bandwidth constraints? If not, what can be done?

Challenges in conducting measurement & inference:

Unlike YouTube, Netflix is not a free, “open” system:
- only subscribers can access videos
- most communications are encrypted; authentication, DRM, …

Measurement Infrastructure

And Basic Measurement Methodology

- Create multiple user/subscriber accounts
- 13 “client” computers located in different cities in US
  - 5 different ISPs: residential/corporate networks
  - non-proxy mode and proxy mode (via PlanetLab)
  - play movies and capture network traces (via tcpdump)
  - e.g., to discover the communication entities and resources involved
  - "tamper data" plugin to read manifest files
  - conduct experiments to study system behavior & dynamics
    e.g., how Netflix selects CDNs & adapts to changes in network bandwidth
- Also conducted measurements/experiments via mobile devices (iPhone/iPad, Nexus 1 Android Phone) over WiFi

Netflix Architecture

- Netflix has its own “data center” for certain crucial operations (e.g., user registration, billing, …)
- Most web-based user-video interaction, computation/storage operations are cloud-sourced to Amazon AWS
- Video delivery is out/cloud-sourced to 3 CDNs
- Users need to use MS Silverlight player for video streaming

Netflix Architecture: Video Streaming

- Manifest file comes from Netflix’s control server
- Periodic updates are sent to the control server
- Frequent downloads in the beginning, becomes periodic after buffer is filled
Netflix Videos and Video Chunks

- Netflix uses a numeric ID to identify each movie
  - IDs are variable length (6-8 digits): 213530, 1001192, 70221086
  - Video IDs do not seem to be evenly distributed in the ID space
  - These video IDs are not used in playback operations

- Each movie is encoded in multiple quality levels, each is identified by a numeric ID (9 digits)
  - Various numeric IDs associated with the same movie appear to have no obvious relations

Netflix Videos and Video Chunks

- Videos are divided in “chunks” (of roughly 4 secs), specified in the URL path:
  - Limelight:
    http://netflix-094.vo.llnwd.net/s/stor3/384/534975384.ismv/range/0-57689?p=5&h=1311456547&k=2cac0bf4e2c522e57700ec1f9d4ace
  - Akamai:
    http://netflix094.as.nflximg.com.edgesuite.net/sa53/384/534975384.ismv/range/0-57689?token=1311456547_411862e41a33dc93ee71e2e3b3fd8534
  - Level3:
    http://nflx.i.ad483241.x.lcdn.nflximg.com/384/534975384.ismv/range/0-57689?etime=20110723212907&movieHash=094&encoded=06847414df0656e697cbd
- Netflix uses a version of (MPEG-)DASH for video streaming

DASH: Dynamic Adaptive Streaming over HTTP

- Dynamic Adaptive Streaming over http
- Used by Netflix and other video streaming services
- Client centric approach to video delivery
- Client performs rate adaptation
- Server is standard http server
- Provides content in multiple formats, multiple encodings
- Allows use of http CDNs

DASH: Dynamic adaptive streaming over HTTP

- Not really a protocol; it provides formats to enable efficient and high-quality delivery of streaming services over the Internet
  - Enables HTTP-CDNs; reuse of existing technology (codec, DRM, …)
  - Move “intelligence” to client: device capability, bandwidth adaptation, …
- In particular, it specifies Media Presentation Description (MPD)
DASH Data Model and Manifest Files

- **DASH MPD:**
  - Media Presentation
  - Period 0, start=0s
  - Period 1, start=100s
  - Period 2, start=200s
  - Representation 1: 500kbit/s, width 640, height 480, duration=10s
  - Representation 2: 100kbit/s

- **Segment Index:**
  - MPD only; MPD+segment; segment only
  - Segment Indexing:
    - MPD only; MPD+segment; segment only

**Media Description**

- Media Presentation Descriptor
  - MDP requested over http
  - Describes all segments
  - Timing information, byte ranges

**Dynamic changing of streams**

- Media has several periods
- Each period has several Adaptation Sets: Audio, video, close caption
- Several Representations per Adaptation
- Several segments per Representation

**DASH Data model**

- Describes all segments
- Timing information, byte ranges
Netflix Videos and CDN Namespaces

Netflix video streaming is handled directly by CDNs

- How are Netflix videos mapped to CDN namespaces & servers?
  
  **Limelight:**
  

  **Akamai:**
  
  http://netflix094.as.nflximg.com.edgesuite.net/sa53/384/534975384.ismv/range/0-57689?token=1311465471_1411862e41a3s9533ed9a8e69d4ace

  **Level3:**
  
  http://nflx.i.ad483241.x.lcdn.nflximg.com/384/534975384.ismv/range/0-57689?etime=20110723212907&movieHash=094&encoded=06b47414d869e697cdd

  

Netflix Manifest Files

- A manifest file contains metadata
- Netflix manifest files contain a lot of information
  - Available bitrates for audio, video and trickplay
  - MPD and URLs pointing to CDNs
  - CDNs and their “rankings”
  
  ```xml
  <nccp:cdn>
    <nccp:name>level3</nccp:name>
    <nccp:cdnid>6</nccp:cdnid>
    <nccp:rank>1</nccp:rank>
    <nccp:weight>140</nccp:weight>
  </nccp:cdn>
  <nccp:cdn>
    <nccp:name>limelight</nccp:name>
    <nccp:cdnid>4</nccp:cdnid>
    <nccp:rank>2</nccp:rank>
    <nccp:weight>120</nccp:weight>
  </nccp:cdn>
  <nccp:cdn>
    <nccp:name>akamai</nccp:name>
    <nccp:cdnid>9</nccp:cdnid>
    <nccp:rank>3</nccp:rank>
    <nccp:weight>100</nccp:weight>
  </nccp:cdn>
  ```
Manifest File Analysis & CDN Ranking

We collected a large number of manifest files.

- Our analysis reveals the following:
  - different CDN rankings for different user accounts
  - does not change over time, geographical location or computer

- Why does Netflix use a static CDN ranking for each user?
  - Is it based upon performance?
  - Is there a single best CDN?
  - Does CDN performance remain unchanged over time?
  - Does this strategy give users the best possible experience?

Netflix: Adapting to Bandwidth Changes

- Two possible approaches
  - Increase/decrease quality level using DASH
  - Switch CDNs

- Experiments
  - Play a movie and systematically throttle available bandwidth
  - Observe server addresses and video quality

- Bandwidth throttling using the "dummynet" tool
  - Throttling done on the client side by limiting how fast it can download from any given CDN server
  - First throttle the most preferred CDN server, keep throttling other servers as they get selected

Adapting to Bandwidth Changes

- Lower quality levels in response to lower bandwidth
- Switch CDN only when minimum quality level cannot be supported
- Netflix seems to use multiple CDNs only for failover purposes!

Will “Intelligent” CDN Selection Help?

- How much can we improve if we have an oracle telling us the optimal CDN to use at each instance?

- Oracle-based “upper bound” yields significant improvement over “average” CDN
- Best CDN (based initial measurements) close to the “upper bound”
Selecting “Best” CDN via Initial Bandwidth Measurement Incurs Little Overhead:

Just 2 measurements are enough!

“Intelligent” CDN Selection …

We can improve user experiences by performing more “intelligent” CDN selection.

- One strategy: selecting “best” CDN using initial measurements!
- What about using multiple CDNs simultaneously?

Average improvement is 54% and 70% for residential hosts and PlanetLab nodes over the best CDN!

Netflix Study Summary

- Netflix employs an interesting cloud-sourced architecture
  - Amazon AWS cloud + 3 CDNs for video streaming
- Netflix video streaming utilizes DASH
  - Enables it to leverage CDNs
  - Performs adaptive streaming for feature-length movies
  - Allows DRM management (handled by Netflix + MS Silverlight)
- Load-balancing, cache misses or other video delivery dynamics are handled internally by each CDN
- Netflix uses a static ranking of CDNs (per user)
  - Multiple CDNs are used mostly for the fail-over purpose
  - How the static ranking is determined is still a mystery (to us!)
- More “intelligent” CDN selection is possible to further improve user-perceived quality (and make them happier!)
  - As higher resolution/3D movies come into picture, these improvements will have a considerable impact