CDNs and Overlay Networks

Introduction

1. Video applications
   - Among the most popular applications today since video related traffic accounts for the majority of the Internet traffic today
   - Focus on the architectures and mechanisms we have in place to support the delivery of video and content in general
     - Content Delivery Networks
       * Challenges
       * Techniques to work around challenges
   - DNS: Important application on its own but also plays a role in content distribution

Introduction to Content Distribution Networks

1. Classic Approach
   - Put content on a single, publicly accessible web server
   - Simple design: Even at scale, having a single, massive data center to service all requests for one Internet video company is a simple design
   - Drawbacks
     - Global distribution: No matter where a single data center is placed, there's potentially vast geographic distance between users and the data center
       * Server-to-client packets traverse many communication links
     - Viral clips: Spike in demand, but also many requests for the exact same data
       * Wasteful to keep sending the same data over the same link
     - Single point of failure: Natural disaster or power outage can totally disrupt the distribution of content
2. Content Distribution Networks
   - Traditional solution is insufficient
   - CDNs are networks of multiple, geographically distributed servers and/or data centers with copies of content that direct users to a server or server cluster that can best serve the user's request
   - Address all the drawbacks of the traditional approach, but encounter new challenges

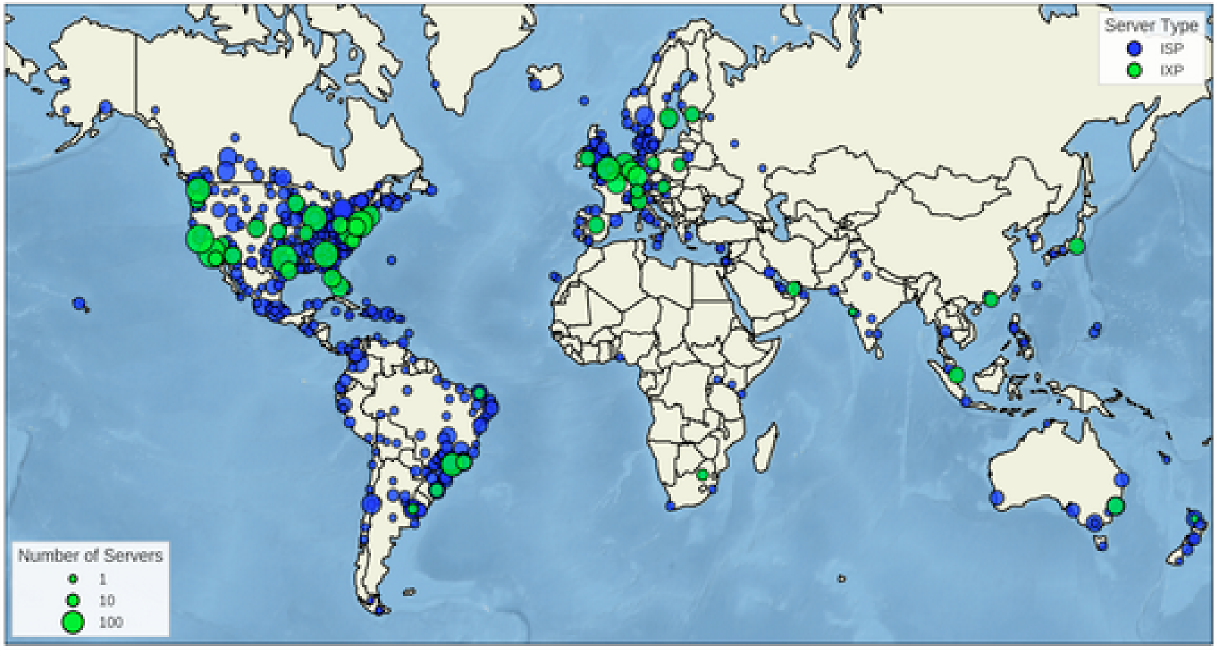
Content Delivery Challenges

1. Six Major Challenges of Internet Applications
   - Peering point congestion: Business and financial motivation to upgrade the "first mile" (web hosts) and "last mile" (end users) but not the "middle mile"
     - Expensive peering points between networks with no revenue
     - Become bottlenecks, causing packet loss and increased latency
   - Inefficient routing protocols: BGP was scaled well, but was not designed for modern demands
     - Only uses AS hop count and doesn't take other factors into account (congestion, latency, etc.)
     - Well-documented vulnerabilities to malicious actions
     - Not efficient for the modern Internet
   - Unreliable networks: Outages occur often
     - Accidents: Misconfigured routers, power outages, etc.
     - Malicious: DDoS attacks or BGP hijacking
     - Natural disasters
   - Inefficient communication protocols: Like BGP, TCP was not designed for the demands of the modern Internet
     - Lots of overhead
     - Distance between server and end user becomes a bottleneck because each packet requires an ack
     - TCP enhancements are slow to be implemented

- Scalability: Internet applications need to be able to respond to current demand by changing resource usage
  - Video going viral, Black Friday shopping
  - Scaling infrastructure is expensive and takes time, but difficult to forecast
- Application limitations and slow rate of change adoption: Even if better protocols are developed, adoption can be slow.
  - Old browsers don't support newer protocols, so even if the server side is upgraded, there's not benefit unless the end users also upgrade their client software

CDNs and the Internet Ecosystem

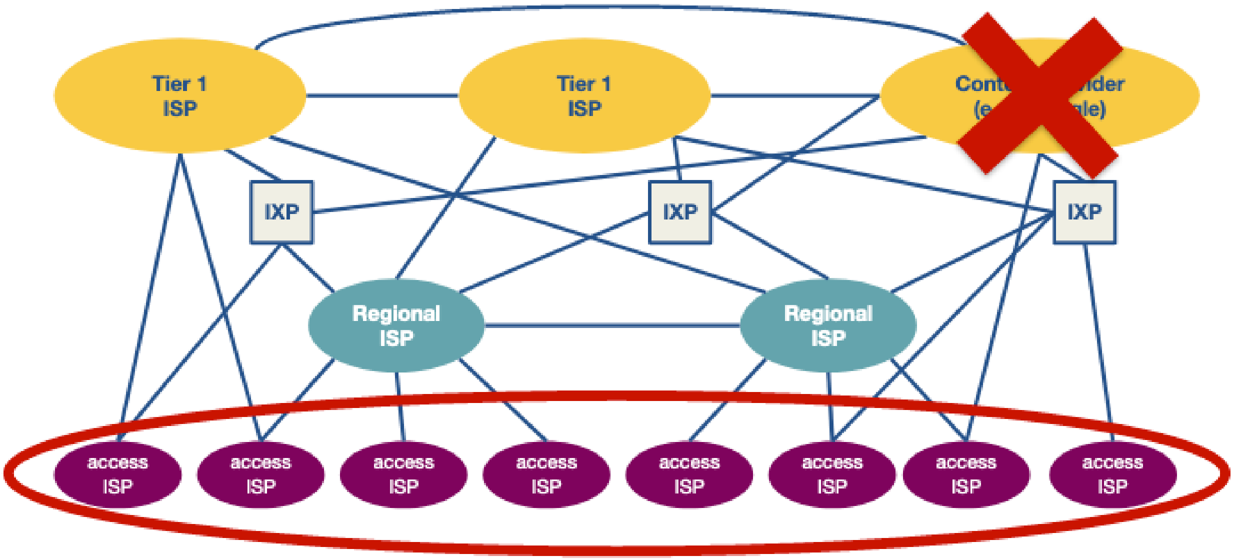
1. Evolution of Internet ecosystem
   - Internet wasn't designed for large scale content delivery, but has evolved
     - Increased demand for online content, especially video
     - Demand has spurred the growth of CDNs
   - Topological flattening: Traditional hierarchical topology has transitioned to be more flat
     - Traditionally, tier 1 ISPs formed the backbone of the Internet
     - IXPs offer interconnections between networks
       * Offer new services
       * Lower network operation costs for ISPs and interconnection costs
   - These shifts mean that we are seeing more traffic being generated and exchanged locally, instead of traversing the complete hierarchy
     - Major players (Google, Facebook) have shifted the focus from tier-1 ISPs to the edge and end users
2. Netflix Open Connect Infrastructure
   - Servers are strategically located around the world to be able to locally serve the end users, bypassing the tier-1 networks for content distribution
   - Challenges of CDNs
     - Cost
     - Real estate
     - Physical devices
     - Power
     - Must be well-connected to the Internet
     - Maintenance and upgrades
   - Private CDNs: Owned by content provider (Google)
   - Public CDNs: Distribute content on behalf of multiple content providers (Akamai and Limelight)



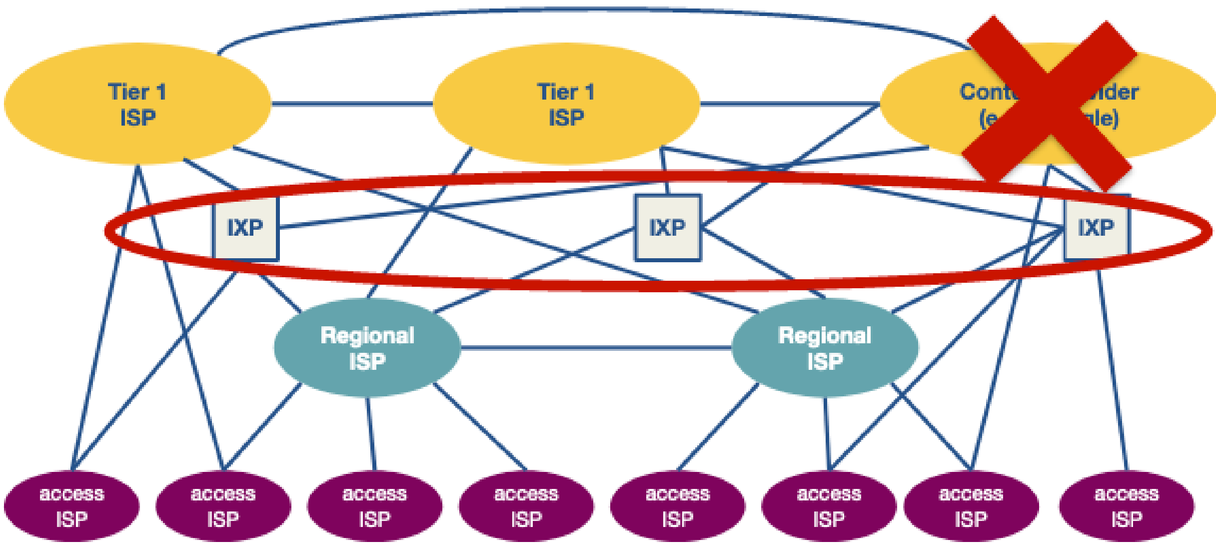
Netflix Server Deployment

CDNs Server Placement Approaches

1. Server Placement Tradeoffs
    - Lots of server clusters that replicate content
    - Where should clusters be placed?
        - Lots of small clusters to get as close as possible to the users
        - Deploying fewer but larger clusters to critical areas
    - Small clusters: "Enter Deep"
        - Place clusters "deep" into the access networks around the world
            * Akamai has clusters in over 1700 locations
        - Make distance between user and closest server cluster as small as possible to reduce delay and increase available throughput
        - Downside: Difficult to manage and maintain
    - Large clusters: "Bring Home"
        - Place fewer larger server clusters at key points, typically in IXPs, "bringing the ISPs home"
        - Fewer clusters to manage and maintain, but users experience higher delay and lower throughput
    - CDNs also employ a hybrid approach
        - Google has 16 "mega data centers", ~50 clusters of hundreds of servers at IXPs, and many hundreds of clusters of tens of server at access ISPs



Enter Deep



Bring Home

How a CDN Operates

1. CDN Operation
    - CDN needs to decide which server cluster should service the request
        - Based on location of user, load on the servers, current traffic, etc.
        - DNS plays a large role in this process
2. Example
    - ExampleMovies pays ExampleCDN to distribute their content
        - User visits examplemovies.com and navigates to Star Wars page

    - Users clicks the link http://video.examplemovies.com/R2D2C3PO37 and user's host ssends a DNS query for the domain video.examplemovies.com
    - DNS query goes to the user's local DNS server (LDNS)
        * DNS server issues an iterative DNS query for "video" to the authoritative DNS server for examplemovies.com, which sends back a hostname like a1130.examplecdn.com
    - User's LDNS sends a query to ExampleCDN's name server, which returns an IP address of an appropriate content server to the user's LDNS
    - User's LDNS returns the ExampleCDN Ip address
    - User's client directly connects via TCP to the IP address provided by the user's LDNS and sends an HTTP GET request for the video
  - By intercepting the requests with DNS, CDNs have the opportunity to choose where to direct users, based on location and/or current conditions

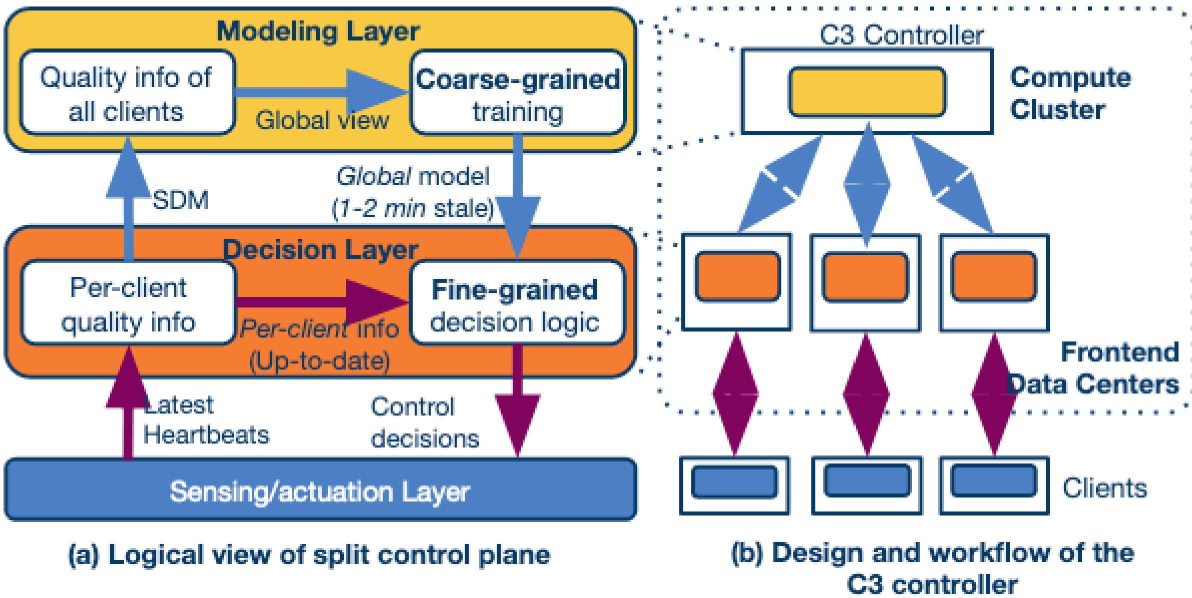
CDN Server Selection

1. Server Selection
    - Picking the right cluster/server is important because it impacts the end-user performance
        - Picking a server that's too far away or overwhelmed can cause the playback to freeze
    - Two main steps
        - Map client to a cluster
        - Select a server from the cluster

Cluster Selection Strategies

1. Cluster Selection
    - Pick the geographically closest cluster "as the crow flies"
        - CDN name server doesn't interact with the user, but the user's LDNS
            * Some customers use a remote LDNS
        - Geographically closest cluster might not provide the best end-to-end network performance
            * Routing inefficiencies lead to higher RTT
            * Congestion along path
        - Relying on a static cluster selection strategy can be inefficient since underlying network conditions are dynamic
    - What metric should be used for cluster selection?
        - Network-layer metrics, such as delay, available bandwidth, or both
        - Application layer metrics, such as re-buffering ratio, average bitrate, or page load time
    - How do we obtain real-time measurements?
        - Active measurements: LDNS could probe multiple clusters, monitor RTT, and use the "closest" server
            * Would create lots of traffic
        - Passive measurements: Name server system in the CDN could keep track of performance metrics based on current traffic conditions
            * Can be kept at an aggregate level
    - Passive measurement limitations
        - Requires a centralized controller with a real-time view of the network conditions between all client-cluster pairs
2. Distributed System for Performance Metrics
    - Coarse-grained global layer operates at larger time scales (minutes)
        - Gloabl view of client quality measurements
        - Builds a data-driven prediction model of video quality
    - Fine-grained per-client decision layer that operates the the millisecond time scale
        - Makes actual decisions upon a client request
        - Based on latest (but possibly stale) pre-computed global model and up-to-date per-client state
    - Challenge

– Needs to have data for different subnet-cluster pairs
– Some clients deliberately need to be routed to sub-optimal clusters



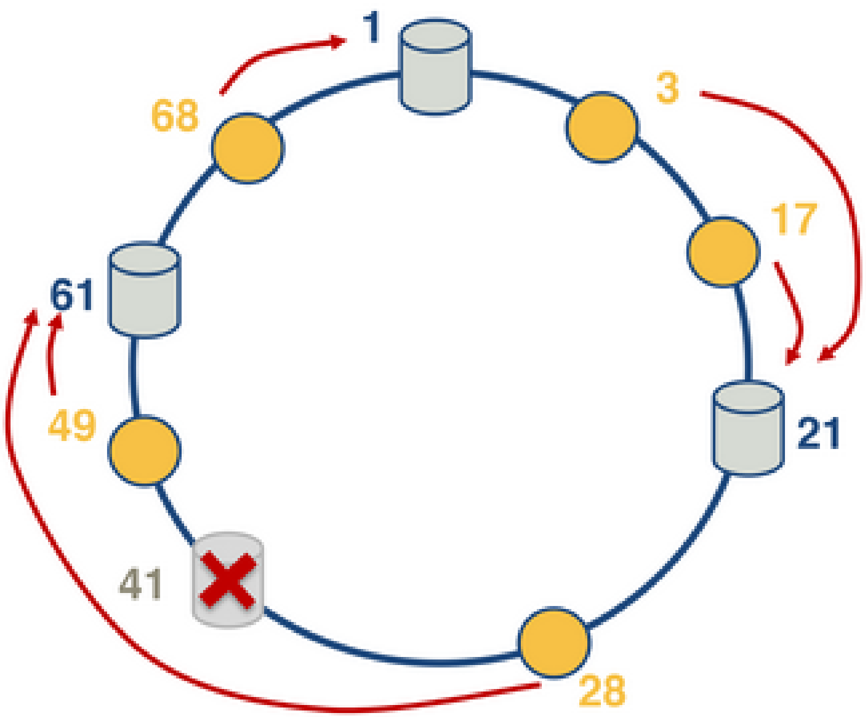
Distributed System that uses a Two-layered System

Policy for Server Selection

1. Server Selection
   - Could pick a random server, but it's better to do some load balancing
     - Even this is insufficient in a CDN as not all servers have all the content
     - Data is fetched as needed (lazily)
   - Requests should be routed to the server with the data in cache
     - Keep a hash of the content and always map requests for the same content to the same server based on this hash
   - Challenges
     - Cluster is dynamic, servers can fail. Mapping requires updating
       * Ideally, only redistribute the hashs on the failed server, not recompute the entire table

Consistent Hashing

1. Distributed Hash Table
   - Consistent hashing: Tends to balance load by assigning roughly the same number of content IDs
     - Requires relatively little movement of these content IDs when nodes join and leave the system
   - Basic idea: Map the servers and content to the same space
     - When a server leaves, we don't have to recalculate anything
   - Proven to be optimal: Least number of keys need to be remapped to maintain load balance on average
   - Part of a distributed lookup protocol called Chord



Consistent Hashing

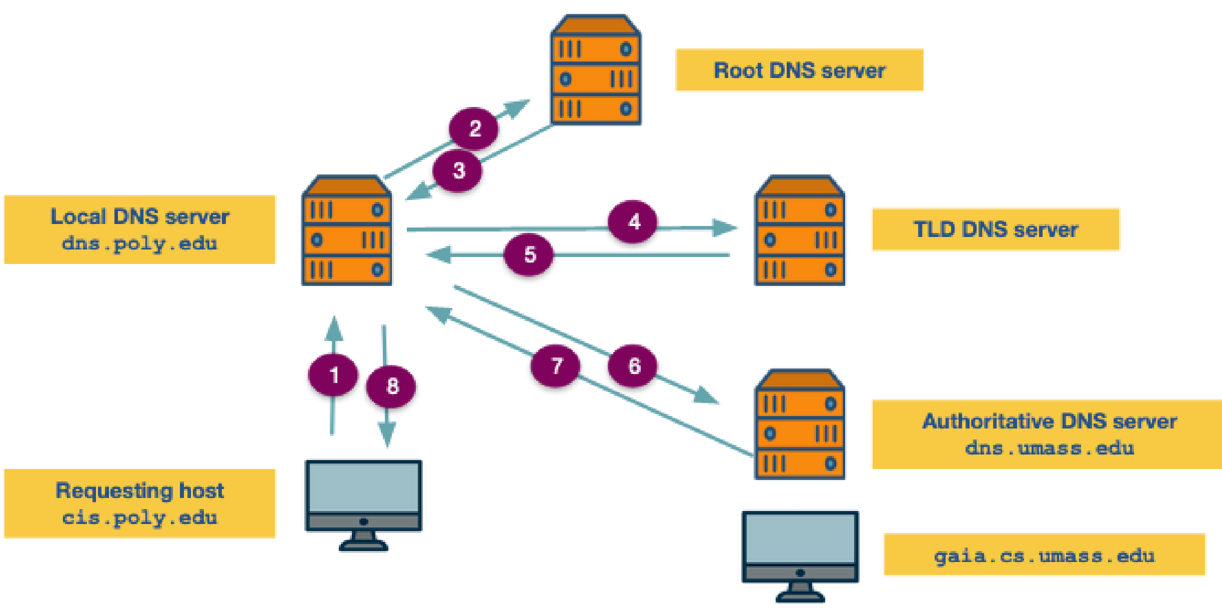
Network Protocols Used for Cluster/Server Selections

1. What are the protocols used for selecting a CDN server?
   - DNS
   - HTTP redirection
   - IP Anycast

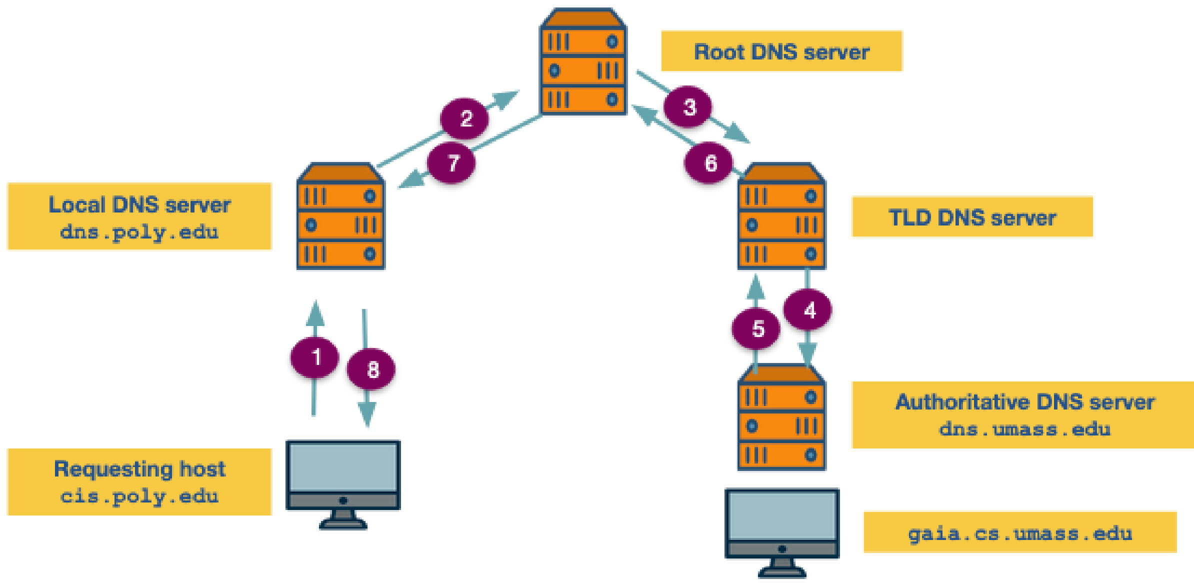
Server Selection Strategies: The DNS Protocol

1. Why do we need DNS?
   - DNS is an application layer protocol that allows hosts to query this database and provide the translation of hostnames to IP addresses
   - Maps hostnames to IP addresses
     - Hostnames are easier to remember
     - IP addresses are easier for routers to process
2. How does DNS work?
   - User host runs the client side of the DNS application
   - Browser extracts the hostname and passes it to client side of the DNS application
   - DNS client sends a query containing the hostname of DNS
   - DNS client eventually receives a reply which included IP address for the hostname
   - As soon as the host receives the IP addresses, it can initiate a TCP connection to the HTTP server located at that port at that IP
3. Other Services Offered by DNS
   - Mail server/host aliasing: Get canonical hostname from alias hostname
   - Load distribution: DNS server responds with the entire set of addresses but rotates the address ordering with each reply (load balancing)

4. DNS Hierarchy
   - Simple model: Single DNS server
     - Single point of failure
     - Very difficult for a single server to handle all the volumes of the querying traffic
     - Central database cannot be close to all querying clients
     - Maintaining the database would be a huge undertaking
   - Handling a request
     - Client contacts the root server, which returns the IP address of a top level domain server
     - Top level domain server returns a referral to the authoritative server for the domain
     - Authoritative server returns the domain-to-IP mapping to reach the domain
   - Root DNS server
     - 13 total servers, each of which is a network of replicated servers
   - Top level domain (TLD) servers
     - Responsible for .com, .org, .edu, etc.
   - Authoritative servers
     - Organization's authoritative DNS server keeps the DNS records that need to be publicly accessible
   - Local DNS servers
     - Each ISP has one or more local DNS servers
     - Hosts that connect to an ISP are provided with the IP addresses of one or more local DNS servers
5. How DNS Works: Recursive and Iterative DNS Queries
   - Process of obtaining the mapping of a hostname to an IP address is known as name-address resolution
   - Iterative query: Querying host is referred to a different DNS server in the chain, until it can fully resolve the request
   - Recursive query: The querying host, and each DNS server in the chain, query the next server and delegates the query to it
     - First query from the requesting host to the local DNS server is recursive, and the remaining queries are iterative



Iterative DNS Queries



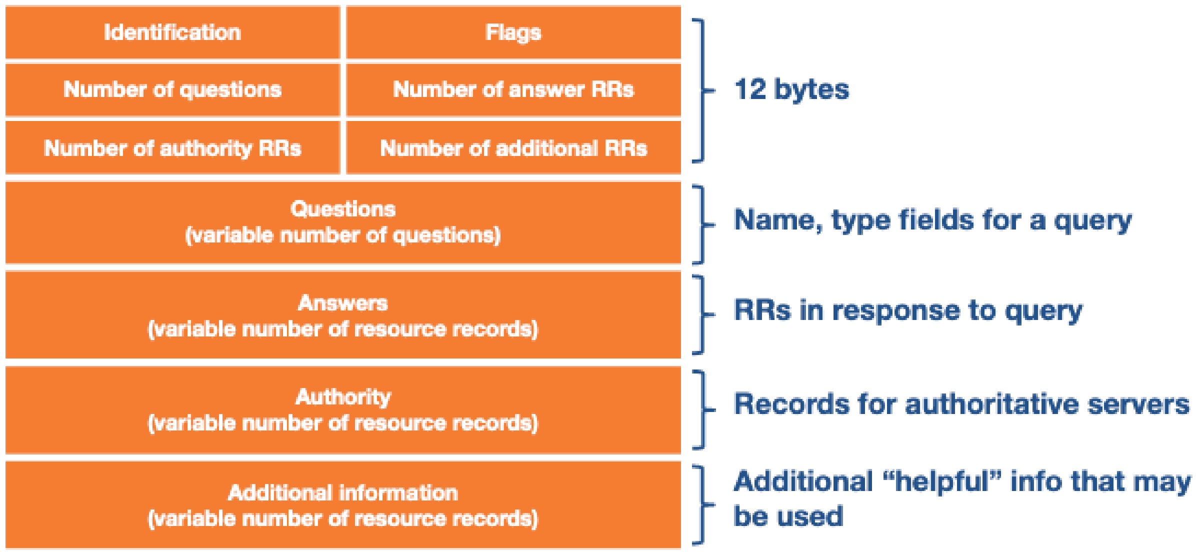
Recursive DNS Queries

More on DNS: Making Responses Faster with Caching

1. DNS Caching
   - Improve performance by caching responses
   - In both iterative and recursive queries, after a server receives the DNS reply of mapping from any host to IP address, it stores this information in the cache memory before sending it to the client

More on DNS: Resource Records and Messages

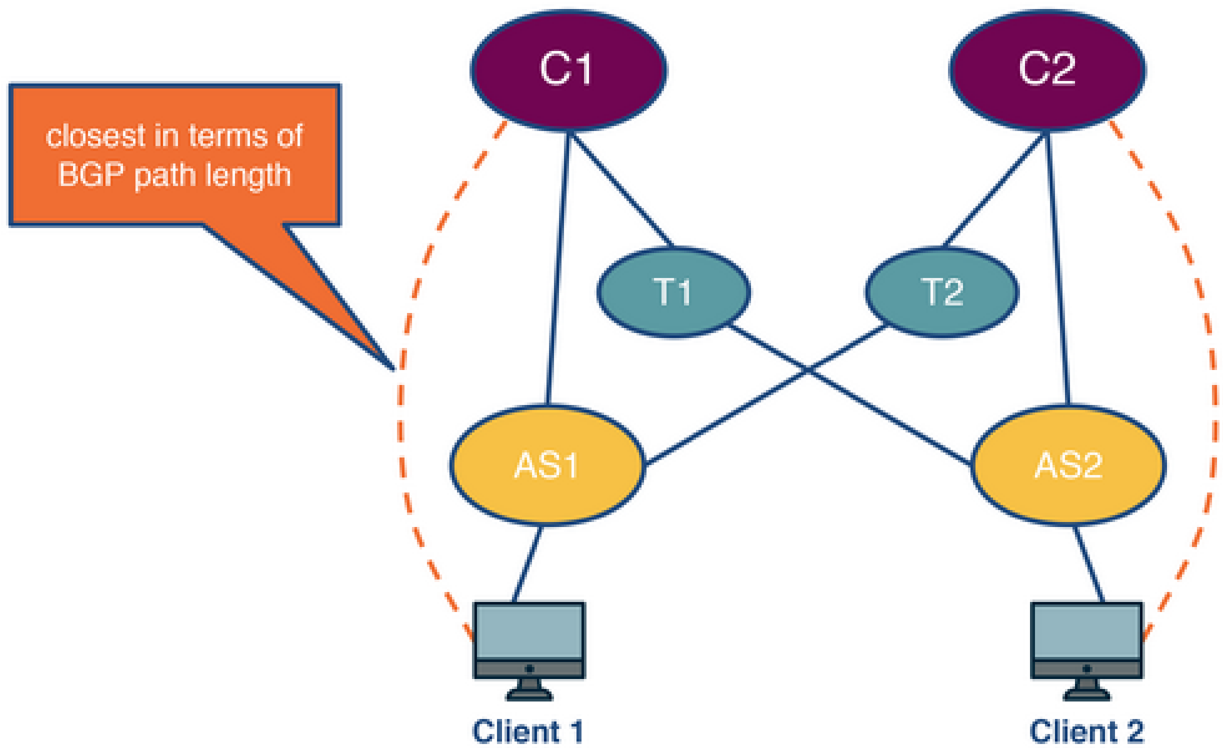
1. DNS Records
   - DNS servers store mappings between hostnames and IP addresses as resource records (RRs)
     - Contained inside DNS reply messages
     - Four fields: Name, value, type, TTL
   - Name/value pairs
     - Type=A: Name is domain name, value is IP address
     - Type=NS: Name is domain name, value is appropriate authoritative DNS server
     - Type=CNAME: Name is alias hostname, value is canonical name
     - Type=MX: Name is alias hostname of a mail server, value is the canonical name of the email server
2. DNS Message Format
   - ID: Identifier for the query, allows client to match queries with responses
   - Flags section: Query or response, recursive or not
   - Question section: Information about the query (hostname, type)
   - Answer section: Resource records for the hostname that was originally queried
   - Authority section: Resource records for more authoritative servers
   - Additional section: Other helpful records



DNS Message Format

Server Selection Strategies: IP Anycast

1. IP Anycast
   - IP anycast: Route a client to the "closest" server, as determined by BGP
     - Achieved by assigning the same IP address to multiple servers belonging to different clusters
     - Multiple BGP routes for the same IP address corresponding to different cluster locations will propagate to the public Internet
     - When a BGP router receives multiple route advertisements for this IP address, it treats the as multiple paths
       * Shortest path is stored and used for routing packets
   - Example:
     - Client from AS1 is routed to C1 because it's only 1 AS hop away
     - Client from AS2 is routed to C2 because it's only 1 AS hop away
   - Generally expect better performance, but doesn't take things like congestion into account, so not typically used in CDNs
     - Doesn't adapt to the dynamic nature of the Internet
     - Still used in routing to the DNS server



IP Anycast

Server Selection Strategies: HTTP Redirection

1. HTTP Redirection
    - Protocol works at the HTTP layer in the network stack
    - When a client sends a GET request to a server, it can redirect the client to another server by sending an HTTP response with a code 3xx and the name of the new server
        - Client should now fetch content from this new server
        - Incurs an additional HTTP request, which can correspond to one or more RTTs, for the client to fetch the content
    - Uses
        - Load balancing: Spontaneous video demands
            * Server can send HTTP redirects to some of the clients
            * Doesn't require any central coordination
            * YouTube uses this kind of mechanism for load balancing