Università degli Studi di Bergamo

DIPARTIMENTO DI INGEGNERIA E SCIENZE APPLICATE

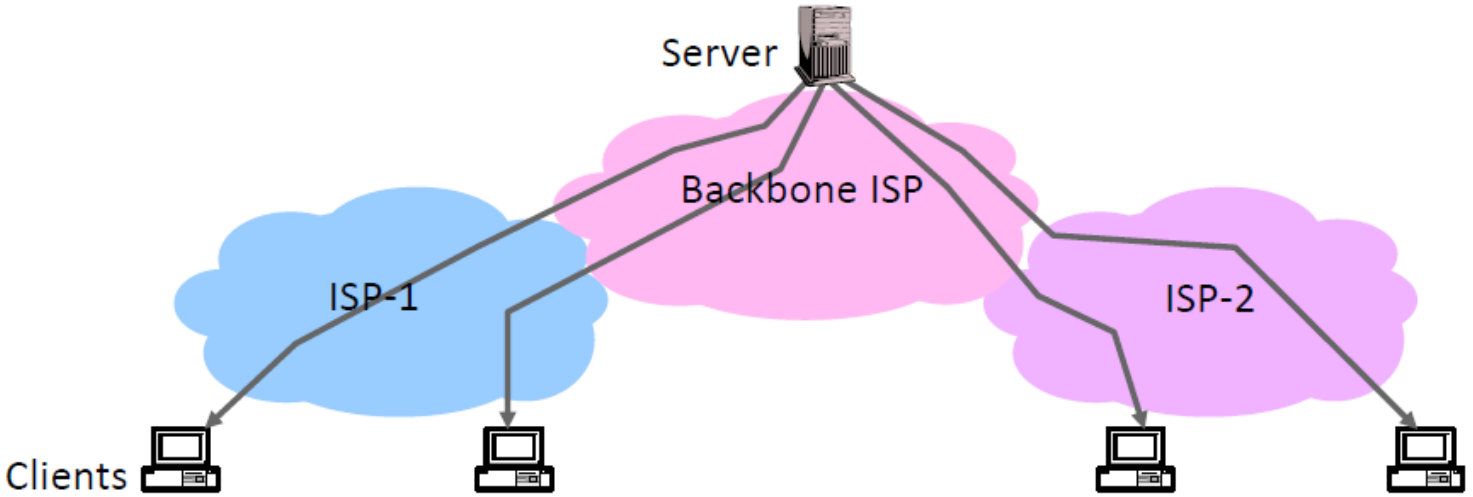
RETI INTERNET MULTIMEDIALI

Content Delivery Networks

Il documento è adattato da materiale cortesemente messo a disposizione dal Prof. Vittorio Trecordi

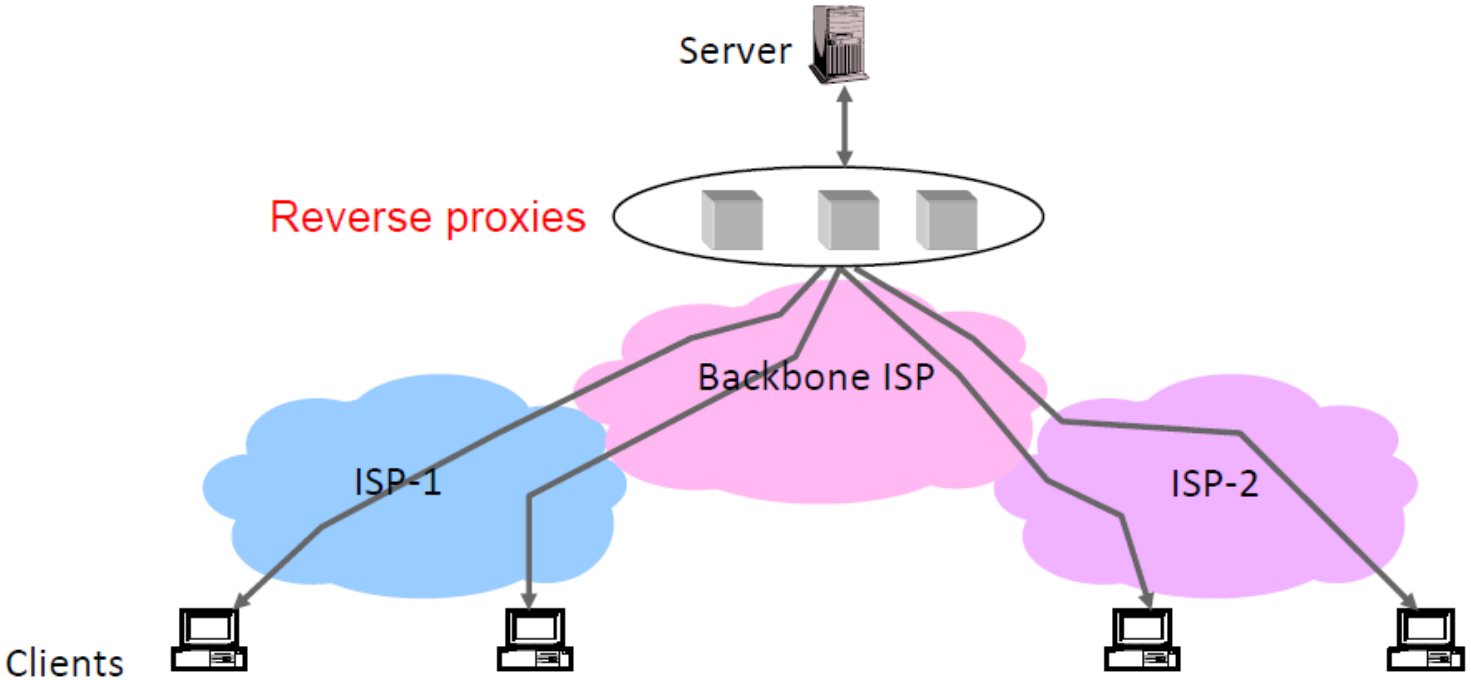
Nuove esigenze e nuove strategie

- La diffusione di contenuti con l'architettura Web ha avuto uno sviluppo notevole grazie al suo paradigma semplice e di grande efficacia
  - Per mezzo dell'architettura Web è possibile trasferire contenuti di tipo eterogeneo: testo, audio, video, immagini, ecc.
- Storicamente i contenuti via Web sono stati resi disponibili per mezzo di *server centralizzati* (origin server) nella rete di backbone
  - Molti client che accedono alla medesima informazione generano un carico su tale server e sulla rete che può essere evitato o ridimensionato
- Nuove strategie per la diffusione efficiente di contenuti via Web sono state definite per
  - Fronteggiare l'aumento del traffico in rete
  - Assicurare latenze accettabili
  - Occupare le risorse in modo efficiente



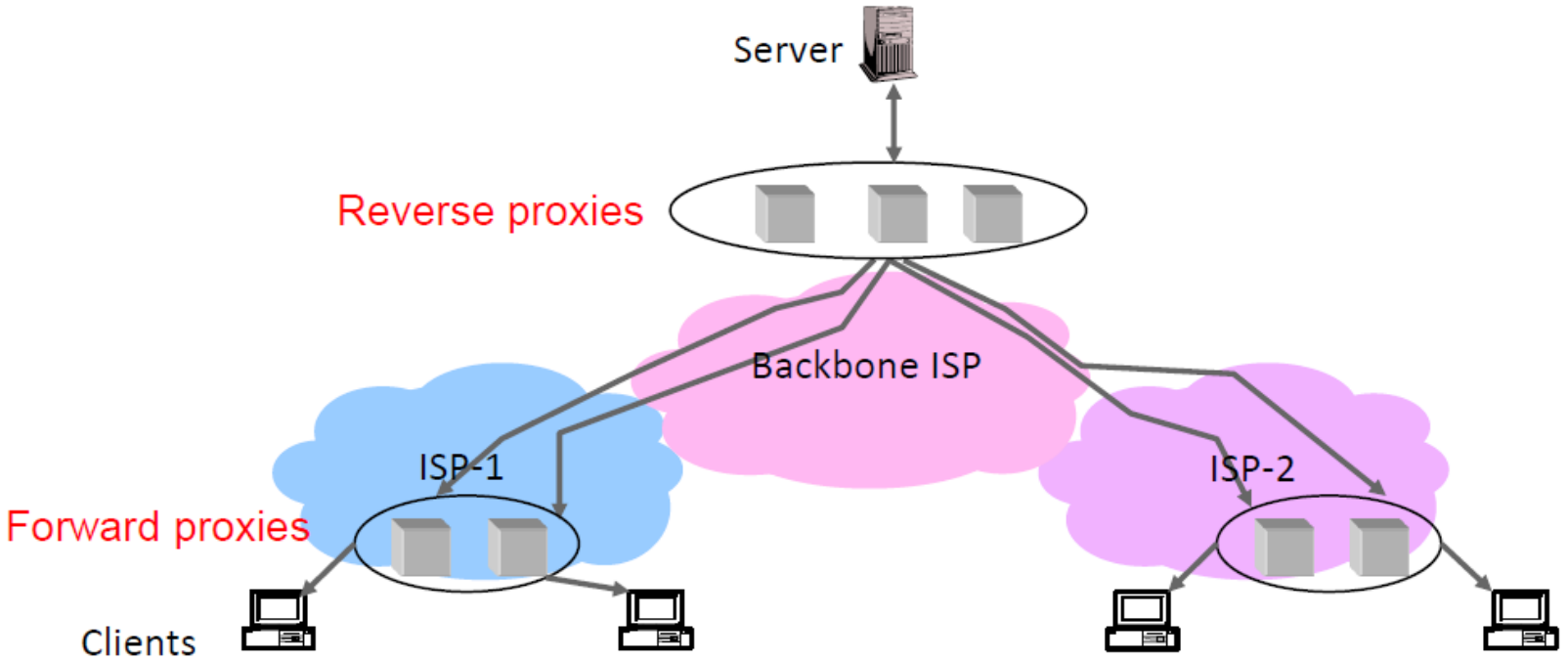
Impiego di Proxy

- L'introduzione di sistemi che replicano i contenuti (sistemi di *caching*) e si frappongono tra client e origin server (detti *Proxy* o *Proxy Applicativi*) è una pratica che tende a migliorare il sistema di distribuzione dei contenuti
- Una possibilità è usare dei *Reverse proxy* utilizzati come front-end del server dai fornitori di contenuti per ridurre il carico del server per la diffusione di contenuti statici



Impiego di Proxy

- I *Forward proxy* sono sistemi intermedi con funzioni di replica dei contenuti posizionati in prossimità dei client e dispiegati nella rete aziendale o dall'ISP nella sua rete
- Questa posizione consente di ridurre la latenza di accesso e contenere il carico di rete per contenuti popolari
- Chiameremo sia i Forward che i Reverse proxy con il termine generico di *cache*



Replica dei contenuti

- La replica dei contenuti è efficace se
  - I contenuti sono consistenti e validi
  - I contenuti sono richiesti da una pluralità di utenti (*località spaziale e temporale*)
- Due diverse modalità di caching
  - Il contenuto può essere replicato dalle cache opportunisticamente quando viene richiesto da un utente (*sistema pull*)
  - Le repliche dei contenuti possono essere sistematicamente programmate (*sistema push*)
    - La copia può essere generata attraverso un sistema esplicito di diffusione che propaga copie dei contenuti che verosimilmente beneficeranno del caching (contenuti popolari)
- Due diverse tipologie di accesso alle cache
  - Nel *Transparent Caching* le cache sono trasparenti rispetto ai client (intercettano il traffico in transito)
  - Nel *Caching Esplicito* i client sono configurati per puntare ad un sistema intermedio che fa caching

Principio del caching

- La capacità di storage delle cache che possono essere posizionate vicino agli utenti è usualmente scarsa
  - Si mira a replicare vicino agli utenti solo i contenuti acceduti più comunemente
  - E' importante definire una strategia per determinare i contenuti usati comunemente (contenuti popolari)
- Gli accessi sono spesso correlati tra loro e si dice che possiedono *località*
  - *Località temporale*: l'informazione che viene consultata in un certo istante sarà consultata nuovamente in tempi brevi
  - *Località spaziale*: l'informazione consultata da un punto sarà verosimilmente consultata da punti adiacenti

Consistenza

- *Consistenza*: assicura che le repliche sono coerenti e allineate
- Diversi gradi di consistenza
  - *Forte*: si evita la consegna di repliche non consistenti
  - *Debole*: si consegnano repliche non consistenti con bassa probabilità
- Esistono tre meccanismi per controllare le repliche di contenuti nelle cache
  - *Invalidation*: dipende *dall'expected expiry time*, definito dall'origin server. Una replica viene invalidata dopo la scadenza di tale tempo
  - *Freshness*: garantisce che una replica in cache possa essere considerata «fresca», ovvero non obsoleta
  - *Validation*: permette di controllare se una replica nella cache è ancora valida, anche dopo la scadenza dell'expected expiry time

Caching cooperativo

- Quando la cache non contiene un contenuto specifico (*cache miss*), anziché chiederlo all'origin server può richiederlo ad altre cache
- Due tipologie di caching cooperativo
  - *Flat*: tutte le cache si trovano allo stesso livello
  - *Gerarchico*: le cache sono strutturate in livelli gerarchici e la diffusione dei contenuti è organizzata ad albero

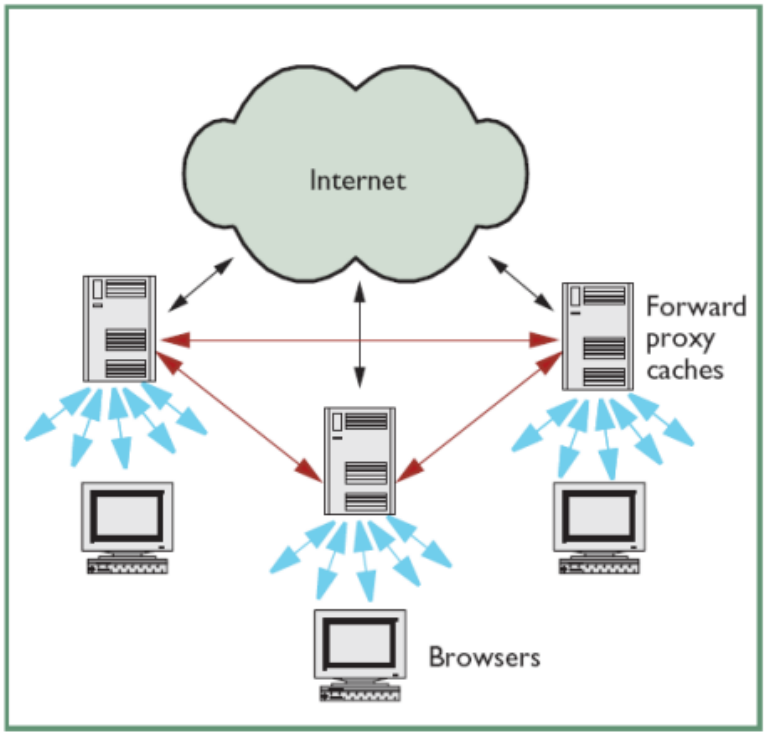


Figure 5. Cooperative caching. Caches communicate with peers before making requests over the Web.

From B. Davison: "A Web caching primer", IEEE Internet Computing, 5(4):38-45, Jul.-Aug. 2001

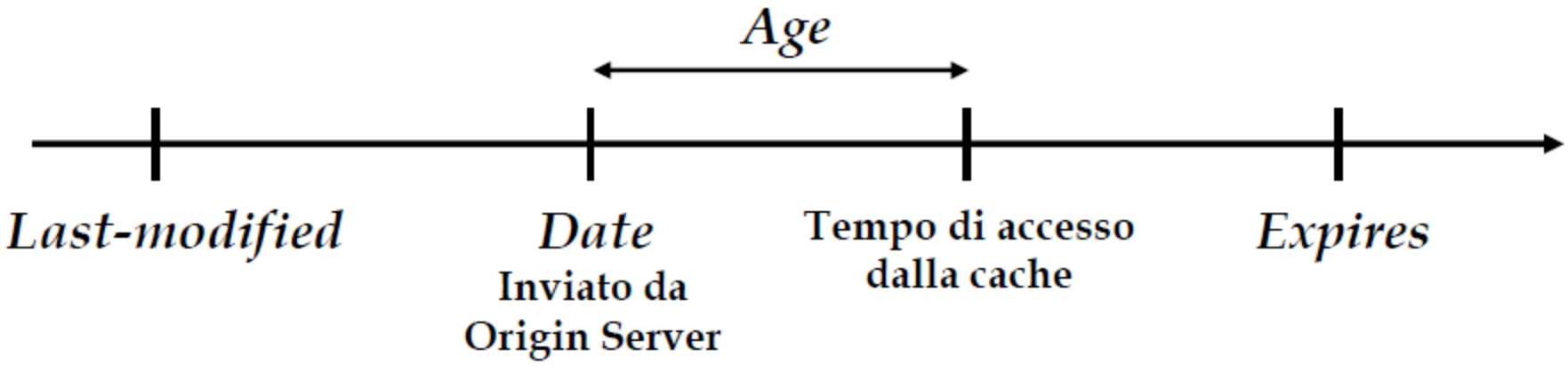
Direttive HTML e HTTP

- Le direttive HTML e HTTP consentono a client e server di impostare le modalità di caching di un contenuto
- Direttive HTML
  - Es. <*meta http-equiv*="pragma" *content="no-cache*">
  - Facile da controllare per gli autori delle pagine web
  - Limitato a oggetti HTML
- Direttive HTTP
  - Es. *Cache-control*:, *Expires*:, *If-Modified-Since:*
  - Campi dell'intestazione HTTP
  - Più facili da gestire da parte delle cache
  - Sono più diffuse

Direttive HTTP imperative

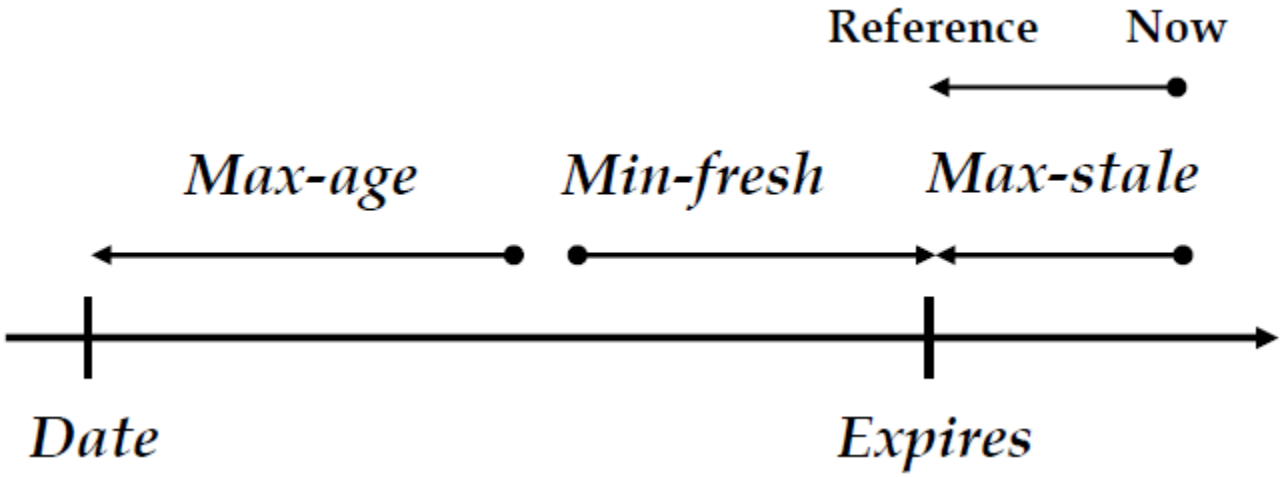
- Hanno la precedenza su altri controlli delle cache
- Nelle richieste e nelle risposte
  - *cache-control: no-store*: impedisce il caching degli oggetti
  - *cache-control: no-transform*: impedisce alla cache di operare trasformazioni dei dati
- Solamente nelle richieste
  - *cache-control: only-if-cached*: richiede l'uso esclusivo di un contenuto presente in cache
    - Se il contenuto non è presente in cache, la cache risponde con un *gateway timeout*

Direttive sul ciclo di vita di un contenuto



- *Last-modified*: istante in cui il contenuto è stato modificato dall'origin server
  - L'origin server e le cache devono riferirsi a un comune clock di riferimento
  - Non differenzia tra cambiamenti minori e maggiori
- *Date*: Indica l'istante in cui l'oggetto è stato inviato dall'origin server alla cache
- *Expires*: predizione del server di quando le copie devono essere sostituite
- *Age*: tempo passato dall'oggetto nella cache
  - *Date + Age < Expires*

Direttive di tempistica



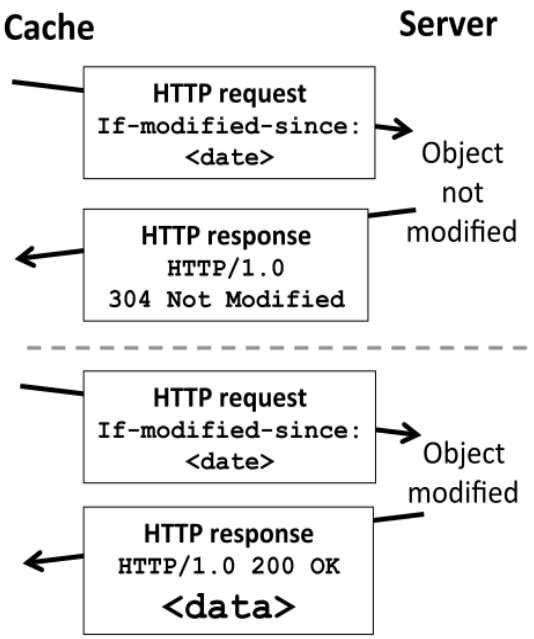
- I client possono richiedere contenuti che hanno una certa esigenza di tempistica
  - *Max-age*: il client può non gradire informazioni con una certa obsolescenza (riferito a *Date*)
  - *Min-fresh*: il client si assicura che un contenuto sia sufficientemente distante dallo stato di *Expiry*
  - *Max-stale*: Il client potrebbe accettare un contenuto un po' obsoleto (di default le cache non restituiscono contenuti obsoleti)

Expiry predetta dal server

- L'origin server limita il tempo di vita dei contenuti che invia alle cache per mezzo dell'*Expires*
- L'expiry time è però una predizione del server e può essere sbagliata, per questo è necessario ricorre alla *validazione* una volta raggiunto l'expiry time
- Nota: se si ha garanzia che l'origin server non modifica un oggetto prima del suo expiry time (Expires), il client può essere sicuro della consistenza forte

Validazione da parte del client

- Il client cerca una replica valida
- La *validazione* è la verifica effettuata per capire se, una volta che l'expiry time è scaduto, una copia è ancora utilizzabile
- Se l'expiry time è stato raggiunto per il contenuto richiesto
  1. La cache invia una GET request con
     - *if-modified-since*: data della copia in cache
     - *if-none-match*: Etag della copia in cache
       - » Etag: stringa che distingue diverse versioni di un contenuto
  2. Il server risponde con uno tra
     - Il contenuto (se modificato)
     - Response code *HTTP/1.0 304 Not Modified*



Esempio di validazione

GET /img/ietf.png HTTP/1.1
Host: irtf.org
Referer: http://irtf.org/
If-Modified-Since: Fri, 06 May 2011 10:01:43 GMT
If-None-Match: "aa0b06-754-4a29893aa8fc0"

HTTP GET Request

HTTP/1.1 304 Not Modified
Date: Tue, 24 May 2011 05:21:29 GMT
ETag: "aa0b06-754-4a29893aa8fc0“
Expires: Tue, 31 May 2011 05:21:29 GMT

HTTP Response

Eterogeneità dei contenuti

- Esistono tipologie eterogenee di contenuti
  - *Contenuti statici*: relativamente stabili nel tempo (es., HTML, immagini, archivi)
  - *Contenuti volatili*: modificati di frequente, periodicamente o da eventi in corso (news, eventi sportivi, titoli di borsa)
  - *Contenuti dinamici*: creati dinamicamente sulla base della richiesta del client (motori di ricerca, e-commerce)
- I *contenuti multimediali*, ovvero quei contenuti audio/video aventi dimensioni tipicamente maggiori rispetto ad altri tipi di contenuti (video clip, mp3, animazioni Flash), sono solitamente *statici*

Contenuti trattabili in cache

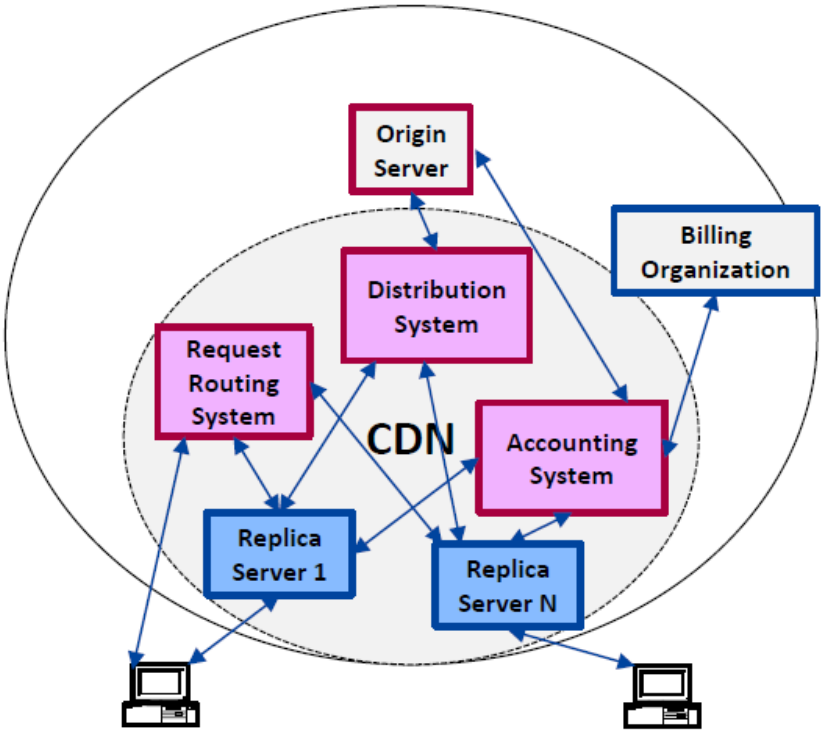
- Una porzione significativa dei contenuti (>50%) sono «uncacheable»
- Le principali fonti di uncacheability sono
  - Volatilità: cambiano di frequente nel tempo e sarebbe dispendioso allineare le copie in cache
  - Dinamicità: le cache non sono solitamente disegnate per generare dinamicamente contenuti
  - SSL: i dati cifrati non sono trattabili in cache
  - Advertising/Analytics: il proprietario ad esempio di un banner pubblicitario vuole misurare il numero di click
- Tuttavia la maggior parte dei contenuti volatili o dinamici ha una dimensione modesta, mentre i contenuti statici e multimediali sono voluminosi

Cosa è una Content Delivery Network (CDN)

- Una CDN è un'infrastruttura creata per distribuire efficacemente i contenuti dei server Web più popolari agli utenti di Internet
- Le CDN si basano sulla distribuzione programmata e intelligente di repliche dei contenuti del server principale del *Content Provider* (origin server) ad una molteplicità di server disposti sulla rete da un *CDN Provider*
  - Il servizio di CDN è offerto dai CDN Provider ai Content Provider che hanno contenuti popolari richiedenti una diffusione vasta e capillare
- Il servizio CDN punta a migliorare le prestazioni
  - Riduzione della latenza di accesso a un contenuto
  - Riduzione della banda occupata in rete

Componenti dell'architettura CDN

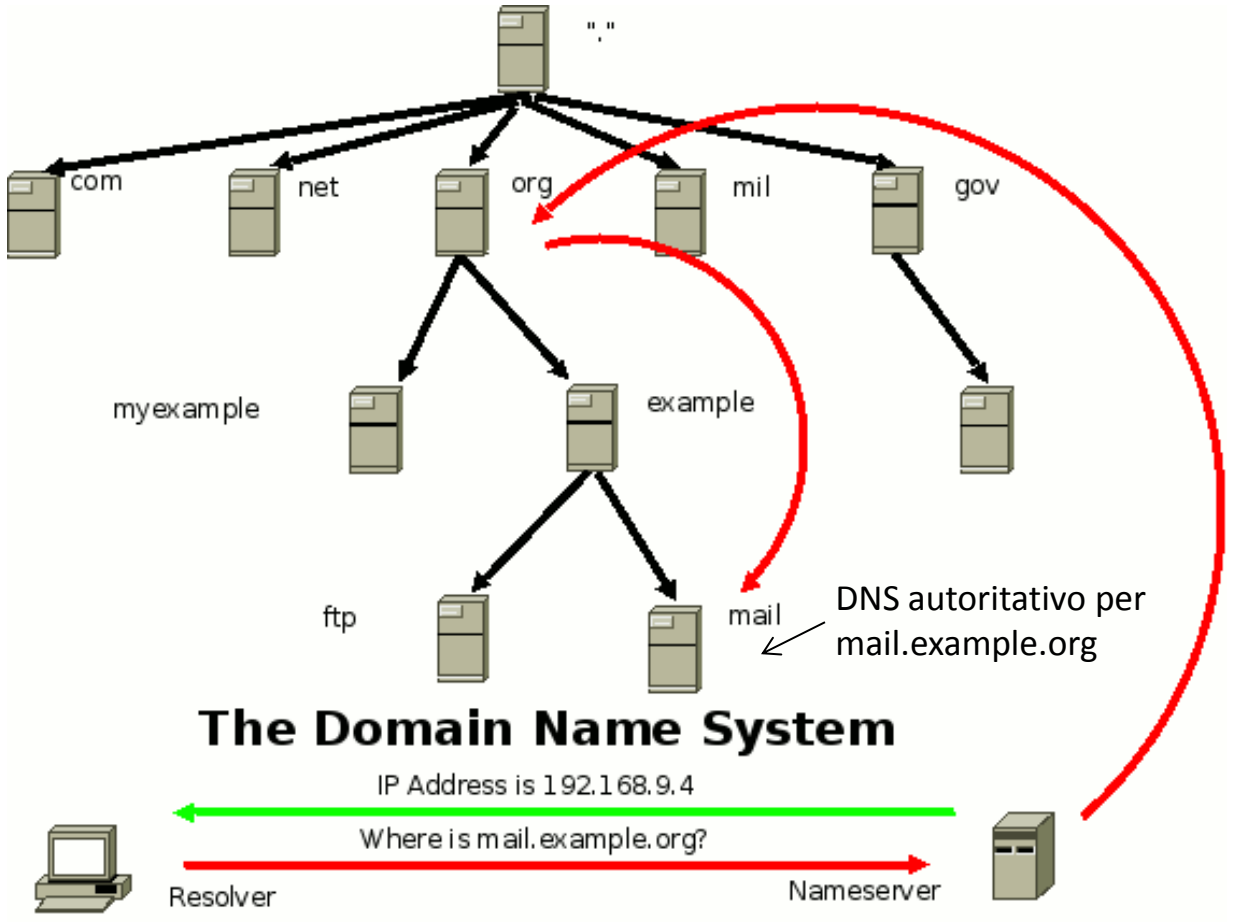
- Componenti di *content delivery*
  - Origin server e insieme di replica server (cache)
- Componente di *distribuzione* del contenuto
  - Replica il contenuto dell'origin server nei Replica server e mantiene la consistenza
- Componente di *request routing*
  - Indirizza le richieste degli utenti verso un server (origin o replica)
  - Interagisce con il componente di distribuzione per mantenere una copia aggiornata del contenuto
- Componente di *accounting*
  - Mantiene i log degli accessi degli utenti
  - Effettua analisi del traffico e permette al Content Provider di effettuare la tariffazione



Componente di request routing

- Come si indirizza la richiesta di un client ad uno specifico cache server della CDN?
- Si usa il sistema DNS (Domain Name Server)
- Vengono utilizzati due meccanismi
  - *DNS redirection*
  - *URL rewriting*

Domain Name Server (DNS)



DNS redirection

- Il DNS autoritativo del sito Web può
  - Delegare la risoluzione dell'hostname in un indirizzo IP a un name server controllato dalla CDN
  - Effettuare direttamente la risoluzione di un indirizzo ad un cache server della CDN (se la CDN può gestire direttamente il DNS autoritativo)
- In entrambi i casi
  - La scelta di un determinato cache server della CDN è effettuata con la traduzione da hostname a indirizzo IP
  - La scelta è effettuata dal sistema DNS interno alla CDN

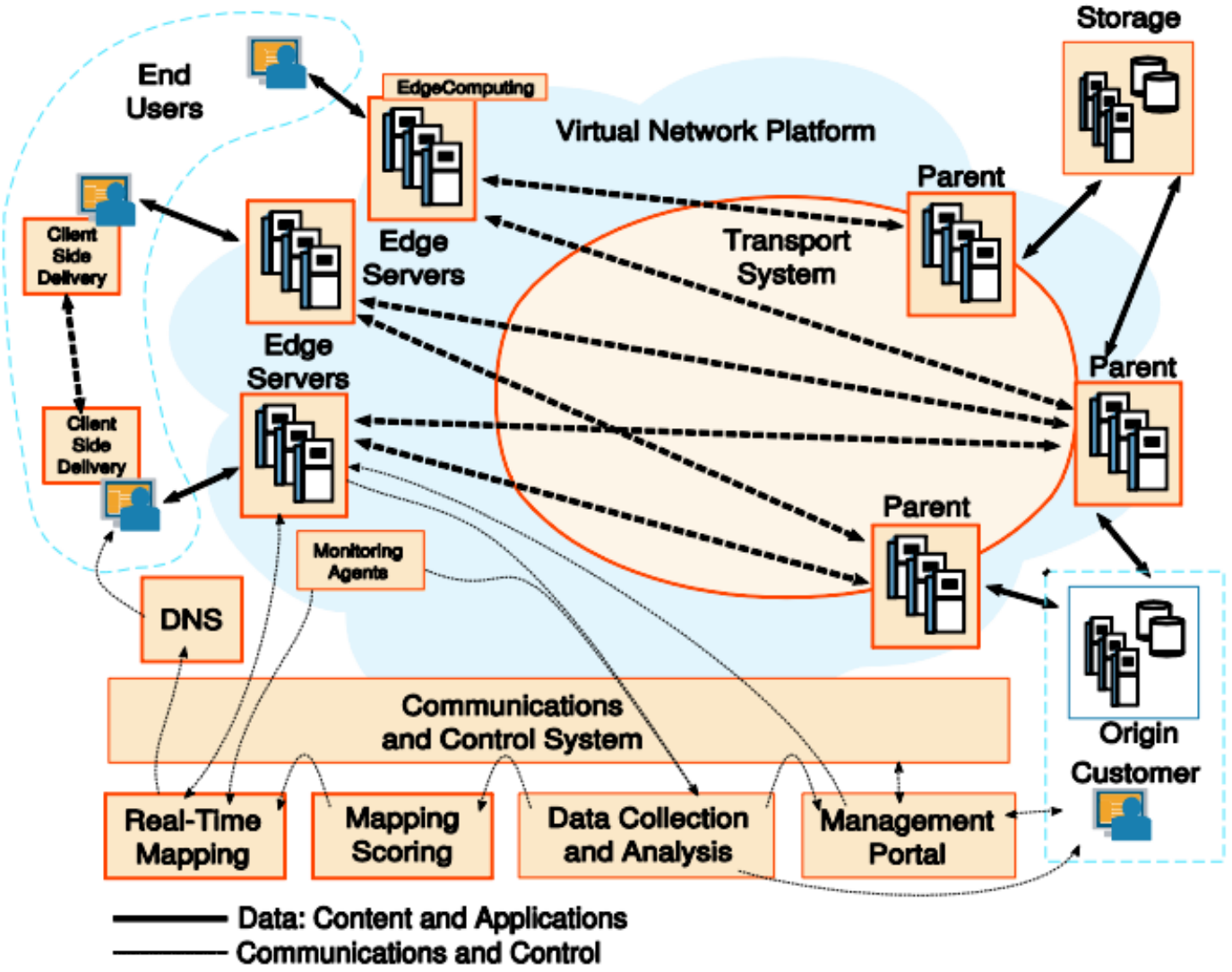
URL rewriting

- Il Content Provider riscrive le URL presenti nella pagina HTML in modo da far apparire che gli embedded object sono localizzati su un cache server
- Così ci sarà bisogno di una risoluzione specifica dell'hostname dei vari embedded object, il cui name server autoritativo è sotto il controllo della CDN
  - Nella risoluzione del nuovo hostname, i DNS della CDN indirizzeranno le richieste dei client per gli embedded object verso un cache server della CDN
- E' possibile usare più di un cache server per gli embedded object di una pagina

Akamai

- Akamai è una compagnia statunitense che si occupa di servizi Internet
  - E’ nata nel 1998 in Massachussets
  - Tra le altre cose, è il più importante CDN Provider del mondo
- Possiede la CDN più estesa al mondo
  - Numerosissimi Content Provider utilizzano la CDN Akamai per la diffusione dei propri contenuti
  - Tra le varie compagnie che hanno una partnership commerciale con Akamai si possono citare Apple, Facebook e Twitter

Akamai Architettura

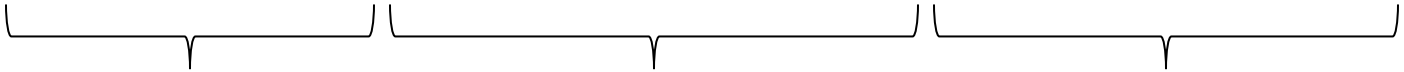


Akamai
Routing indiretto e URL rewriting

- Un sito Web che vuole avere una parte dei propri contenuti distribuiti da Akamai deve rinominare le URL ad esse relative con un prefisso specifico
- La risoluzione dell'hostname in un indirizzo IP di un cache server di Akamai è eseguita dal DNS di Akamai
- Il cache server prescelto è «vicino» al client e non deve essere sovraccarico
  - Akamai effettua misurazioni e test per ottenere una mappa della rete Internet
  - In questo modo il DNS può stabilire il cache server ottimo da cui reperire uno specifico contenuto da parte di uno specifico utente

Akamai
ARL: Akamai Resource Locator

http://a620.g.akamai.net/7/620/16/259fdbf4ed29de/www.cnn.com/i/22.gif



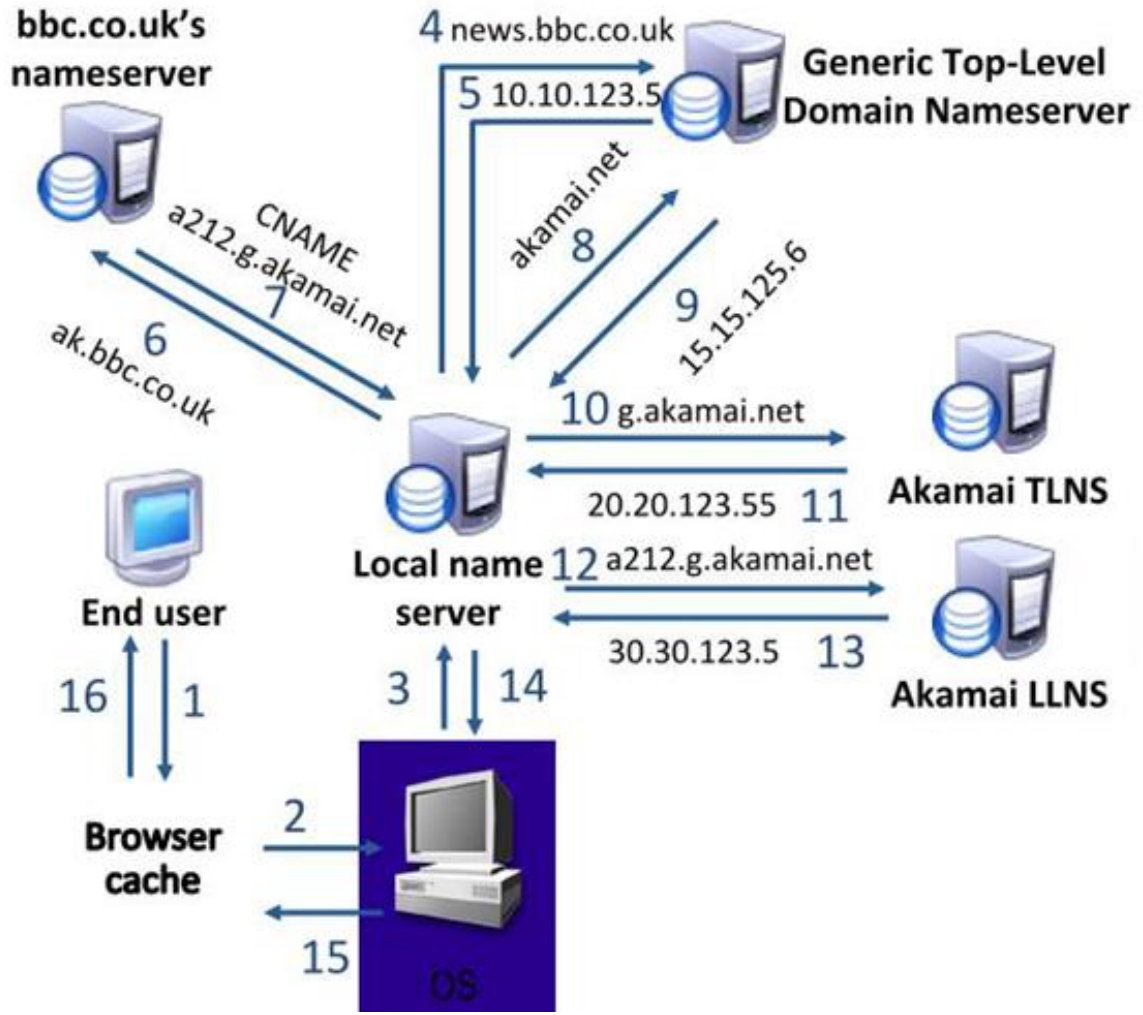
Host Part | Akamai Control Part | Content URL

- Il content provider seleziona il contenuto che sarà ospitato da Akamai
- Akamai fornisce uno strumento che trasforma l'URL (www.cnn.com/i/22.gif) nella ARL riportata sopra
  - L'host part della ARL viene inviata come risposta dal DNS autoritativo del contenuto al Nameserver del client
- In questo modo, il client accede ai cache server di Akamai (a620.g.akamai.net/) e non all'origin server
- Se il server Akamai non ha il contenuto in cache esso viene richiesto all'origin server

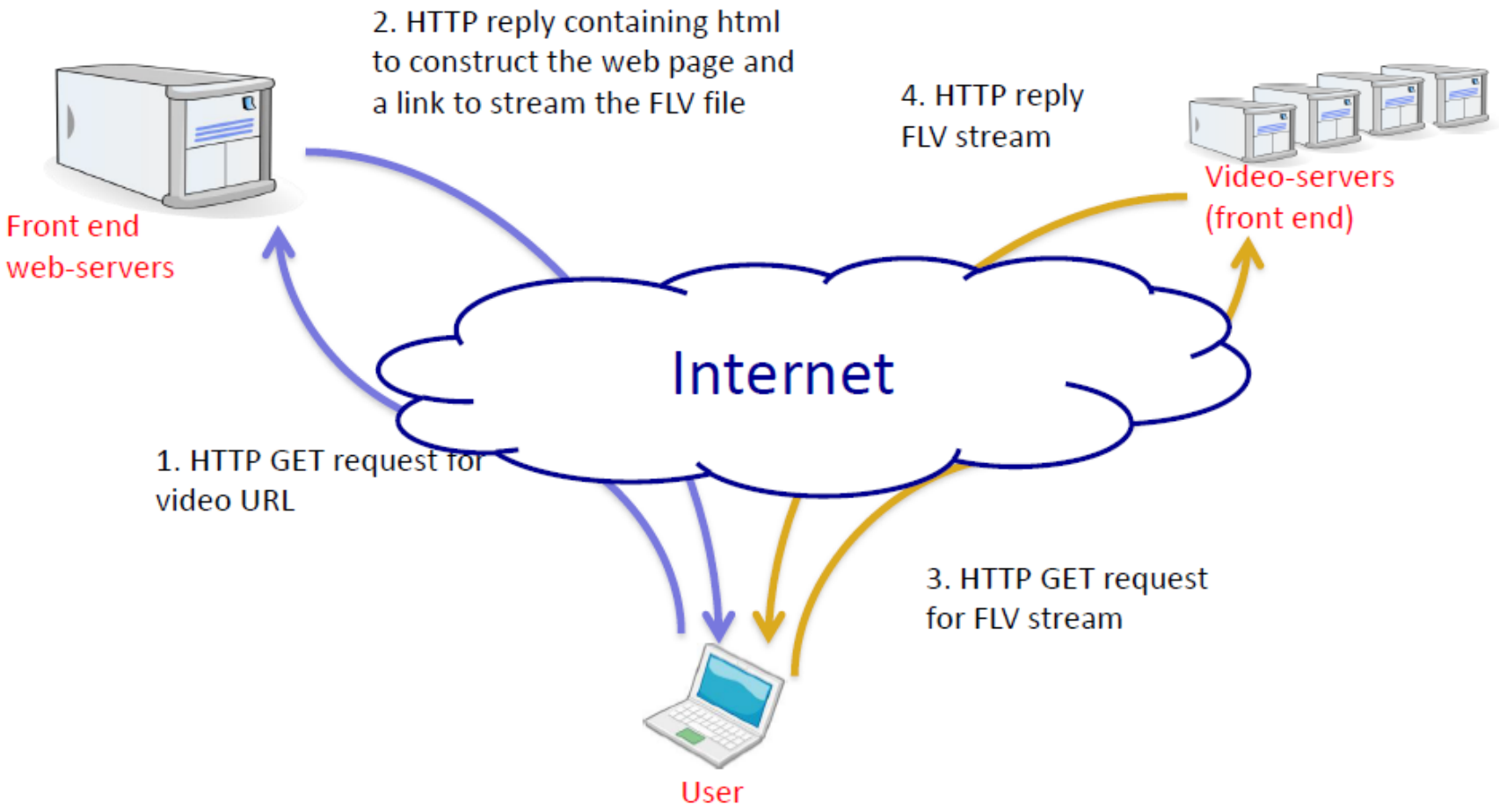
Akamai
Ridirezione DNS a due livelli

- Akamai prevede due livelli di ridirezione DNS
  1. Akamai Top-Level Name Server (TLNS)
     - Localizzati: 4 in US, 3 in EU e 1 in Asia
     - I TLNS rispondono con un LLNS, preso da un insieme di 8 LLNS vicini all'utente
  2. Akamai Low-Level Name Server (LLNS)
     - Puntano verso gli Akamai Edge Server che consegnano i contenuti
     - Effettuano bilanciamento del traffico

Akamai
Ridirezione DNS a due livelli



Architettura Youtube
Modalità base

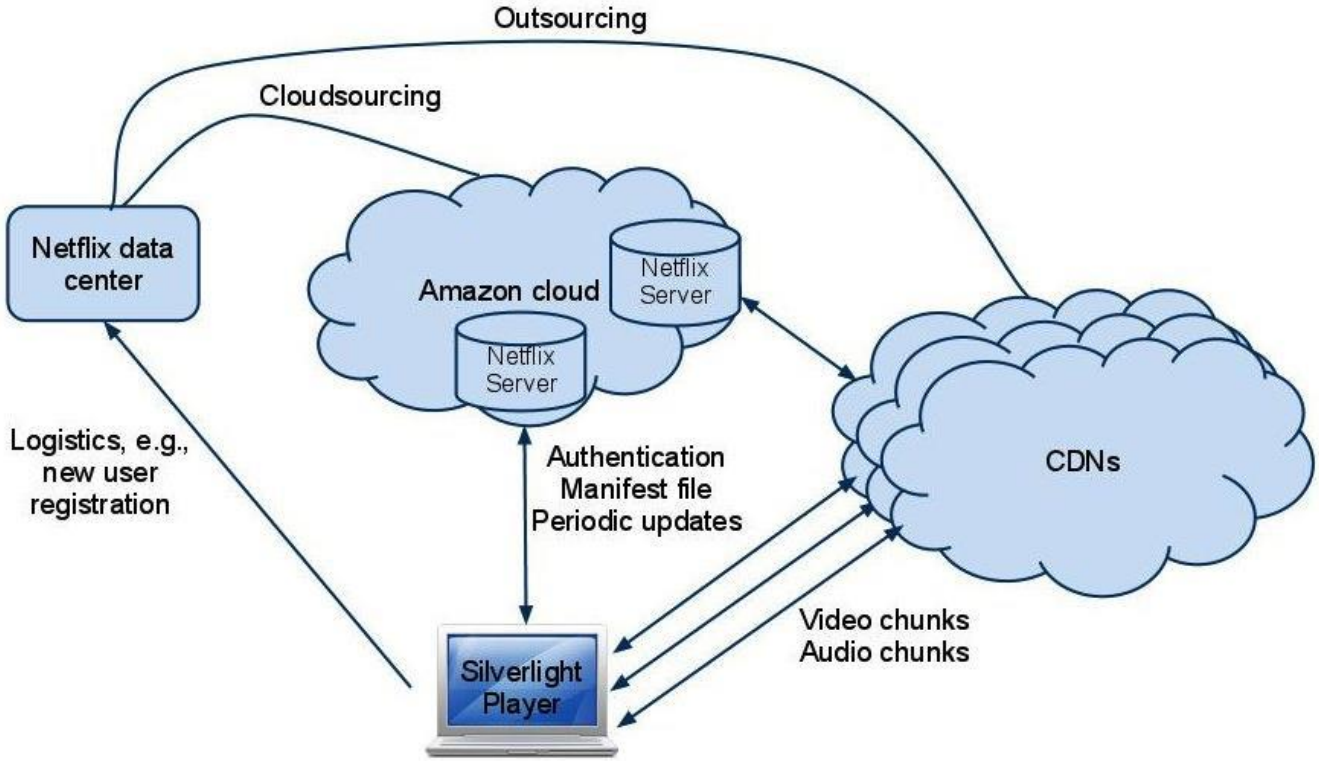


Architettura Youtube
Nuova modalità

- Struttura di cache a tre livelli: primaria, secondaria e terziaria
- Bilanciamento del carico dei server sulla base dell'informazione di localizzazione
- Circa 50 cache
  - 40 primarie (circa 10 presso ISPs)
  - 8 secondarie
  - 5 terziarie

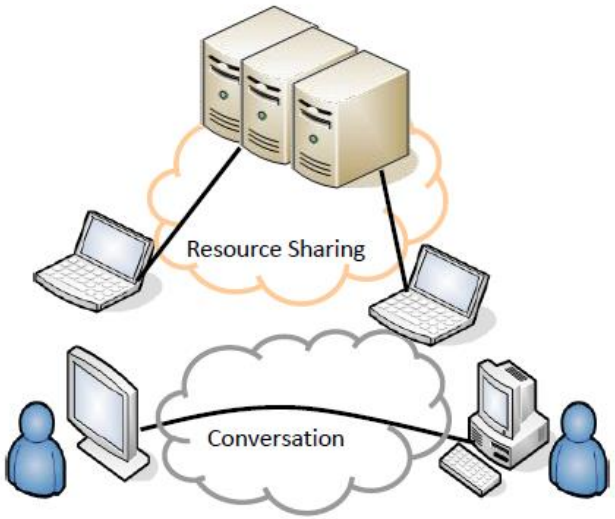
Architettura Netflix

- Netflix ha un proprio Datacenter per le funzioni di gestione (es. registrazione e tariffazione), per il resto si basa su Cloud AWS (Amazon Web Service) di Amazon per i video streaming server
- Utilizza tre CDN per la consegna di contenuti video (a codifica scalabile)



Un Nuovo paradigma Information Centric Networking

- Progressivamente si assiste ad un cambio di paradigma dell'uso preminente della rete Internet
  - Da sistema di condivisione di risorse e di conversazione tra host a sistema di distribuzione di contenuti in varie forme e con volumi crescenti
- La consapevolezza di queste tendenze ha sollecitato lo studio di un nuovo paradigma per Internet che sposta il fuoco dell'attenzione sul trattamento dei contenuti: **Information Centric Networking**
- E' un paradigma alternativo a IP, basato sui contenuti piuttosto che sugli indirizzi



20 exabytes/month added in 2011