# Web caching e sistemi per Web content delivery

Valeria Cardellini
Università di Roma Tor Vergata

## Tipi di caching

- Caching lato client (*content consumer*)
  - I browser memorizzano su disco le risorse per accessi futuri
- Caching lato proxy (*terze parti*)
  - I proxy server memorizzano le risorse più richieste
  - Proxy server gestiti da istituzioni o ISP
- Caching lato server (*content provider*)
  - Le cache lato server riducono il carico sulla piattaforma server
- Distribuzione dei contenuti e servizi (*terze parti*)
  - Gli edge server rappresentano dei nodi in una rete di livello applicativo per la distribuzione e consegna dei contenuti e servizi Web
  - Edge server gestiti da società specializzate



## Politiche di rimpiazzamento della cache

- Quando una risorsa viene richiesta dal proxy ad un server (origin server o altro proxy server) dovrebbe essere memorizzata in cache, per sfruttare il principio di località
  - Cache del proxy su disco
- Se la cache è piena, occorre rimpiazzare una o più risorse presenti in cache per fare spazio alla nuova risorsa
  - Occorre definire una politica di rimpiazzamento della cache (**cache replacement**)



## Politiche di rimpiazzamento della cache (2)

- Differenza fondamentale rispetto alle politiche di rimpiazzamento studiate per la memoria cache o per la memoria virtuale in un calcolatore
- *Nel calcolatore*: caching di *parti* di risorse, parti aventi dimensioni fisse (blocchi o pagine)
- *Nel Web*: caching di *intere* risorse, caratterizzate da dimensioni largamente variabili (distribuzioni heavy-tailed)
  - Distribuzione heavy-tailed: la coda della distribuzione è più pesante della coda della distribuzione esponenziale
  - Grandissimi valori sono possibili con probabilità non nulla
    - Ad es., dimensioni delle risorse Web da $10^3$ a $10^7$ byte
  - Una variabile casuale X con $F(x)=P[X \leq x]$ è distribuita heavy-tailed (con indice di tail $\alpha$) se:

    $1-F(x) = Prob[X>x] \sim c\, x^{-\alpha}$ , $0 < \alpha < 2$

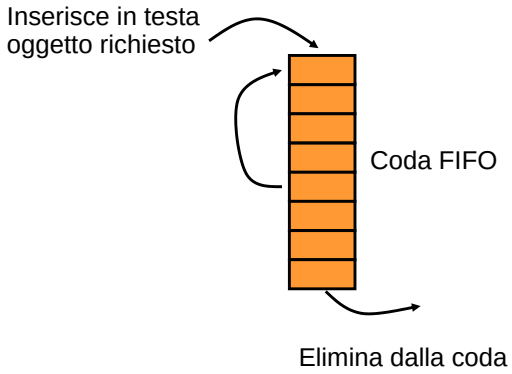
# Politiche di rimpiazzamento della cache (3)

- Principali classi di politiche di rimpiazzamento della cache in ambito Web
  - Derivate dal caching tradizionale: **LRU e LFU**
  - Basate su una chiave primaria (tipicamente dimensione dell'oggetto)
    - Rimpiazzamento dell'oggetto più grande all'interno della cache
    - Varianti di LRU applicate ad un sottoinsieme di oggetti (ad es. LRU-threshold)
  - Basate su più chiavi
  - Basate sul costo
    - **Greedy-Dual-Size** e sue varianti



# Least Recently Used (LRU)

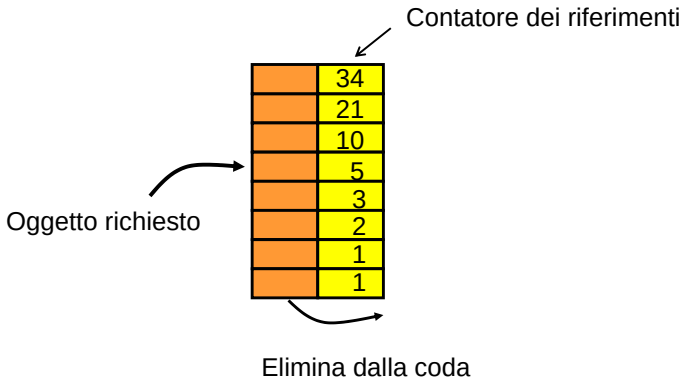
- Rimpiazza l'oggetto usato meno recentemente






# Least Frequently Used (LFU)

- Rimpiazza l'oggetto usato meno frequentemente






# Greedy-Dual-Size

- Associa un costo ad ogni oggetto
- Rimpiazza l'oggetto con il rapporto minimo tra costo e dimensione
  - Riferimento: P. Cao. S. Irani, "Cost-aware WWW proxy caching algorithms", Proc. of USENIX Symp. on Internet Technologies and Systems, 1997.
- Variante Greedy-Dual*
  - Sfrutta il fatto che oggetti appartenenti alla stessa pagina hanno maggiore probabilità di essere richiesti insieme
  - Se un qualunque oggetto nella pagina genera un cache hit, tutte gli oggetti della pagina sono mantenuti in cache
  - Riferimento: S. Jin, A. Bestavros, "GreedyDual* Web caching algorithm: Exploiting the two sources of temporal locality in Web request", Computer Communications, 2001.

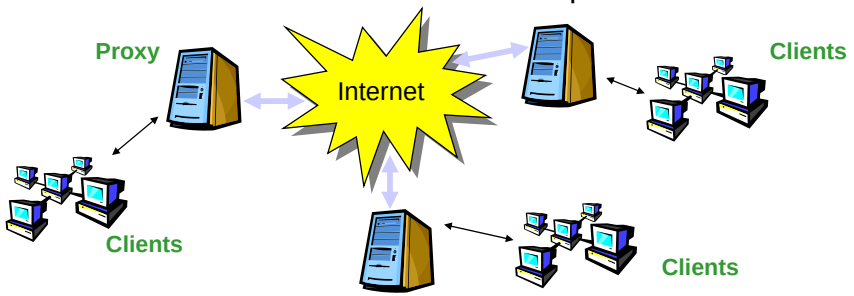
# Metriche di prestazione

- Per misurare l'efficacia del caching (sia delle politiche di rimpiazzamento della cache sia del caching cooperativo che non) si usano le seguenti metriche:
  - Tasso di hit
    - Percentuale di richieste servite dalla cache
  - Tasso di byte hit
    - Percentuale di byte serviti dalla cache
  - Banda risparmiata
    - Quantifica la diminuzione nel numero di byte richiesti agli origin server
  - Riduzione della latenza



# Web caching cooperativo

- Diverse cache possono cooperare per …
  - Aumentare il cache hit rate
  - Migliorare le prestazioni
    - Distribuire meglio il carico tra più server
    - Aumentare la scalabilità
  - Aumentare l'affidabilità
- Problemi di overhead dovuti alla cooperazione



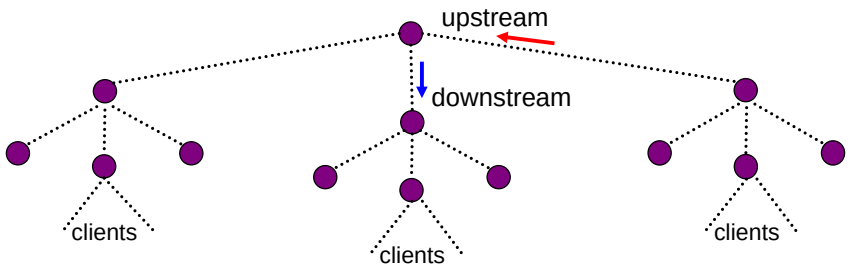



# Modalità di cooperazione

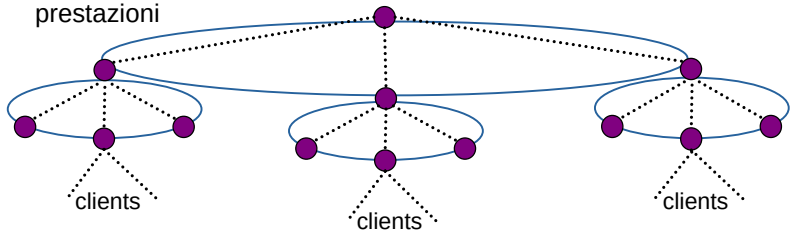
- A chi chiedere informazioni?
  - Modalità **gerarchica**: relazione ***genitori-figli*** verticale
    - Se un cache server figlio ha un cache miss chiede al cache server genitore
  - Modalità **flat**: relazione ***paritetica*** orizzontale
    - Se un cache server ha un cache miss chiede a tutti o a un sotto-insieme di cache fratelli (*peer*)
- Quando chiedere informazioni?
  - Schema **query-based**: query al momento del cache miss
  - Schema **informed-based**: scambio periodico di informazioni tra i cache server indipendentemente dalla richiesta



# Modalità gerarchica

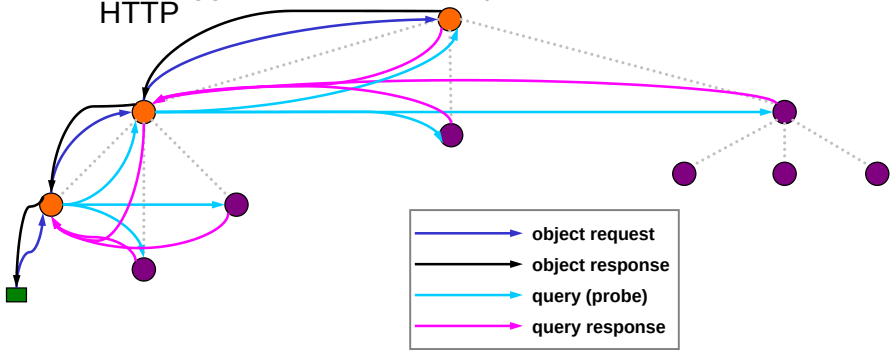



- Cooperazione tra pari livello (sibling) per aumentare le prestazioni



# Internet Cache Protocol (ICP)

- Schema di tipo **query-based** basato su probe di tipo multicast tra sibling
- Si contatta il proxy che risponde per primo con un hit per ottenere la risorsa oppure si attende di ricevere tutte le risposte di miss (o la scadenza di un timeout)
- Messaggi ICP su UDP, recupero della risorsa su HTTP





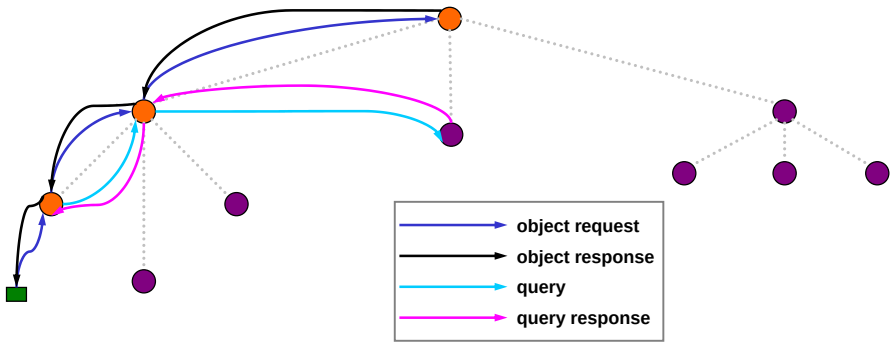

# Cache Digests

- Schema di tipo **informed-based** basato sullo scambio tra i cache server di un *digest* del contenuto delle varie cache
- Ogni cache server replica la directory (*summary*) di ciascuno dei suoi peer
- Un peer viene contattato solo se il summary locale indica un hit per la risorsa cercata
- Summary basato su *filtri di Bloom* per ridurre l'overhead di memorizzazione
  - Ogni summary è di circa tre ordini di grandezza più piccola della cache corrispondente
  - Usando i filtri di Bloom possono esserci falsi hit (circa 1%) ma non falsi miss



# Gerarchia con Summary e ICP

• L'uso di summary ad ogni livello della gerarchia permette di ridurre il numero di query tra sibling risultanti in miss fino al 95%

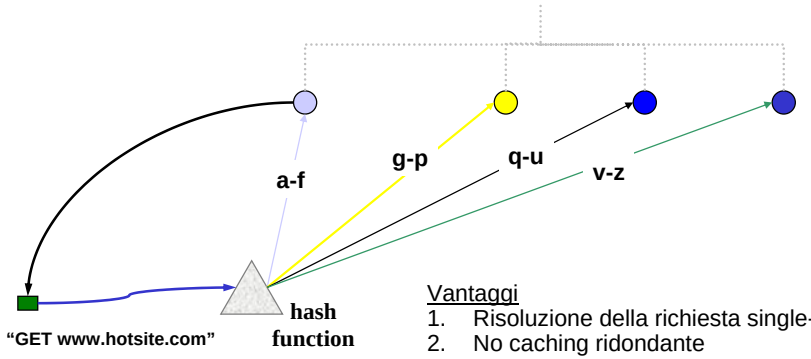





# Scambio di informazioni

- Schema **query-based**
  - Minor rischio di informazioni incongruenti (*false hit* e *false miss*)
  - Maggiore latenza: occorre aspettare la risposta
  - Minore overhead di comunicazione (si scambiano informazioni solo quando serve)
- Schema **informed-based**
  - Maggior rischio di informazioni incongruenti
  - Minima latenza (si sa già dove andare)
  - Maggiore overhead di comunicazione (si scambiano informazioni periodicamente)
- Quindi: tradeoff tra overhead e incongruenza

# Cache Array Routing Protocol (CARP)




Vantaggi
1. Risoluzione della richiesta single-hop
2. No caching ridondante
3. Consente implementazioni client-side
4. Riconfigurazione possibile

Svantaggi
1. Schema prettamente locale
2. Difficoltà di bilanciamento
3. Miss penalty



# Squid: caratteristiche

- Squid è il server proxy più usato
  http://www.squid-cache.org/
  - Progettato per sistemi Unix-like
  - Free e open-source
  - Efficienza
  - Stabilità ed affidabilità (processo di sviluppo open)
    - Duane Wessels è il responsabile
- Squid supporta
  - Protocolli applicativi HTTP, FTP, Gopher, WAIS …
  - Tunneling tra client e server dei protocolli SSL, HTTPS, TLS
  - architetture distribuite e gerarchiche di proxy cooperativi
  - Protocolli di cooperazione ICP (default), CARP, Cache Digests
  - Meccanismi sofisticati per il controllo degli accessi
  - Caching trasparente
  - Funzionalità di HTTP server accelerator
  - Caching delle risoluzioni DNS



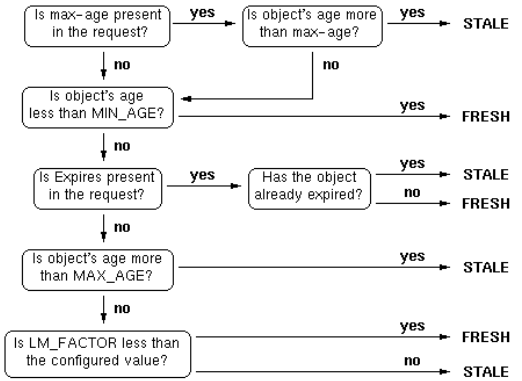
# Squid: controllo degli accessi

- Per decidere quali client sono abilitati ad usarlo come proxy, Squid utilizza una Access Control List (ACL)
  - Lista di indirizzi IP (a livello di dominio) per filtrare le richieste in ingresso, ad esempio:
    ```
    acl myclients src 172.16.5.0/24
    http_access allow myclients
    ```
- Squid può anche essere configurato per disabilitare l'accesso a determinati server Web
    ```
    acl dumb dstdomain notbright.com nitwit.com
    http_access deny dumb
    ```
- Validazione delle risorse in cache attuata da Squid
  - Versioni iniziali: utilizzavano un meccanismo basato sul time to live (TTL) degli oggetti di tipo *esplicito* o *implicito* (*fisso* o *adattativo*)
  - Versioni recenti: utilizzano un modello basato sul tasso di refresh



# Squid: gestione della cache

- Refreshment della cache
- LRU come politica di default per il rimpiazzamento della cache

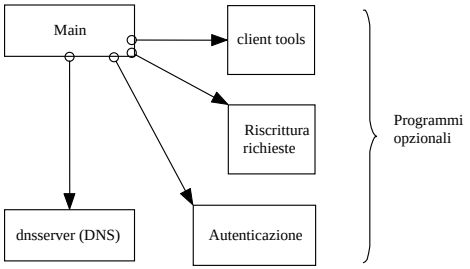



age = now – object date
lm_age = object date – last modified time
lm_factor = age / lm_age

MIN_AGE, MAX_AGE, LM_FACTOR are configuration variables and depend on the object's URL as well

Last-Modified: optional header containing the last modification date of an object
Expires: optional expiry time specified by HTTP servers
max-age: optional directive which specifies the maximum allowed age of an object

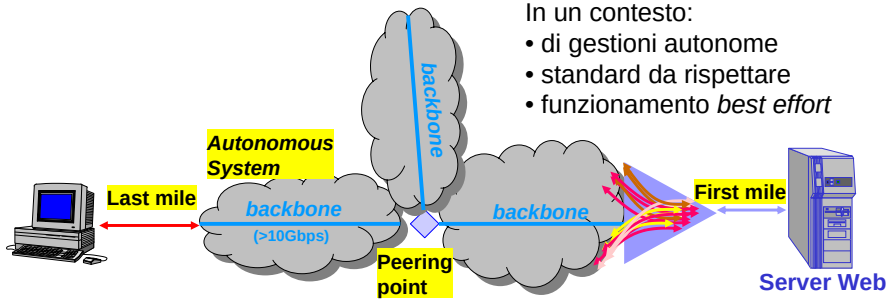
# Squid: architettura




- Proxy a singolo processo
- I/O non bloccante
  - Uso di *callback function* (in C con puntatori a funzioni)
  - Per la risoluzione dei nomi in indirizzi IP non usa gethostbyname() ma un processo esterno (dnsserver)



# Colli di bottiglia del delivery




In un contesto:
- di gestioni autonome
- standard da rispettare
- funzionamento *best effort*

- First mile:
  - linea di collegamento tra il server Web ed il resto di Internet
- Peering point
  - punto di connessione tra diversi sistemi autonomi
- Backbone
- Last mile
  - collegamento del client al proprio ISP



# Content delivery

- **Content delivery**: strategia per distribuire il contenuto mediante rete IP su una base di pay-per-delivery
- Content Delivery Network o Content Distribuiton Network (CDN)
- Il content delivery ha molte probabilità di dar luogo a profitti

- Il problema del network delivery
  - Più di 9000 Autonomous System (AS)
  - Nessuna rete gestisce più del 5% del traffico
  - La grande maggioranza degli AS gestisce meno dell'1% del traffico
- Soluzione al problema del network delivery
  - Distribuire i server in molti (quasi tutti) gli AS



# Content Delivery Network

- Content Delivery Network (o Content Distribution Network):
  - formata da un'infrastruttura di server (detti *content server*, *delivery server* o *edge server*) che lavorano insieme
  - gli edge server della CDN sono distribuiti su una vasta area geografica per permettere la fornitura dei contenuti e servizi Web da locazioni più vicine all'utente
  - il content provider delega il servizio del contenuto del proprio sito Web (fornito dall'*origin server*) ad una compagnia che gestisce una CDN (*content outsourcing*)
  - gli edge server della CDN forniscono soltanto il contenuto dei siti Web gestiti dalla CDN

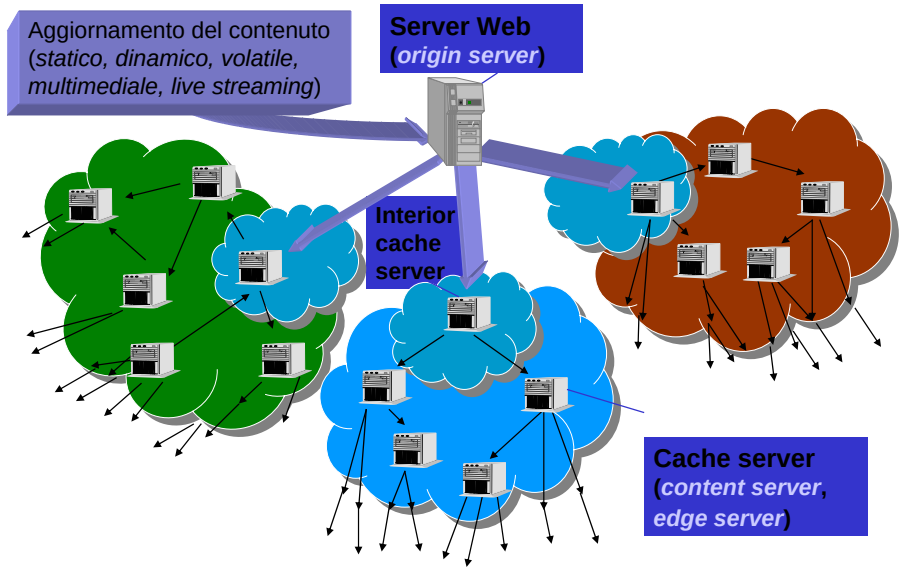
# Web caching e CDN

- Web caching e CDN hanno in comune:
  - architettura costituita da un'infrastruttura di cache server che operano in modo cooperativo
  - i cache server sono distribuiti su una vasta area geografica per permettere la distribuzione del contenuto Web da locazioni più vicine all'utente
- Per il resto differiscono sostanzialmente:

| | |
|---|---|
| • Il Web caching opera in modo indipendente dai siti Web<br>• Costo per ISP (risparmio di banda), non per i content provider<br>• Caching generalizzato: viene memorizzano il contenuto di tutti i siti Web | • Il content provider delega la distribuzione del contenuto alla CDN<br>• Costo per i content provider, non per ISP<br>• Caching selettivo: viene memorizzato solo il contenuto dei siti Web gestiti dalla CDN |



# Architettura di una CDN






# Perché usare una CDN?

- Lo scopo è analogo al Web caching:
  - avvicinare il contenuto del sito Web all'utente per ridurre il tempo di latenza ed il carico sull'origin server
- I Web cluster possono essere lontani dagli utenti
- La congestione di Internet può far fallire le migliori architetture di server Web
- I picchi di traffico (*flash crowd*) possono provocare un crash dei server Web
- I sistemi Web distribuiti geograficamente richiedono che il content provider possegga l'infrastruttura di servizio



# Vantaggi di CDN rispetto a Web caching

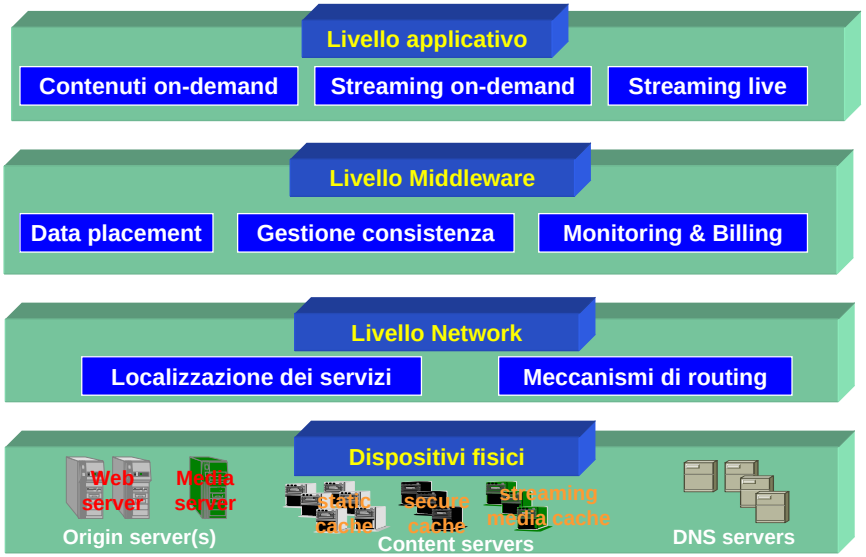
| Web caching | CDN |
|---|---|
| • cache dei server proxy riempite in seguito alle richieste degli utenti | • contenuto memorizzato in anticipo (*prefetching*) |
| • overhead sul sito Web derivante dall'offloading ripetuto per ogni proxy server che accede al sito | • overhead sul sito Web derivante dall'unico offloading |
| • il content provider non ha il controllo del contenuto nella cache dei proxy server (*elevati problemi di consistenza*) | • il content provider ha il controllo diretto del contenuto degli edge server |
| • il content provider perde molte informazioni sulla distribuzione del proprio contenuto | • una CDN può fornire al content provider statistiche accurate sui pattern di accesso al sito Web |

# Contenuti serviti da CDN

- Una CDN è generalmente utilizzata per fornire i seguenti servizi di delivery:
  - Distribuzione contenuti non streaming
    - Contenuto statico delle pagine Web, in particolare immagini
    - Contenuto dinamico
      - Per lo più contenuto "volatile" (contenuto statico che cambia frequentemente)
    - Contenuto con autenticazione
      - Autenticazione prevista dal protocollo HTTP
    - Contenuto sicuri
      - Protocollo HTTPS
  - Distribuzione contenuti streaming on-demand
    - Il contenuto è digitalizzato e memorizzato come media file su media server. Esempio: video-on-demand, clip musicali
  - Distribuzione contenuti streaming live
    - Il contenuto è distribuito quasi istantaneamente come media file. Esempio: eventi sportivi e musicali



# Componenti dell'architettura di una CDN






# Gestione della consistenza

- Come garantire che le copie delle risorse sugli edge server siano consistenti con la risorsa sull'origin server?
- Approcci possibili per gestire la consistenza guidati dall'origin server:
  - Aggiornamento periodico
  - Propagazione dell'aggiornamento
  - Aggiornamento on-demand
  - Invalidazione



# Meccanismi di routing per CDN

## come indirizzare la richiesta al server "migliore"

- Una questione fondamentale riguarda il meccanismo di routing con cui ridirigere una richiesta di un client per un oggetto servito dalla CDN verso un edge server
- Principali tecniche adottate:
  - uso del Domain Name System: ***DNS redirection***
  - riscrittura URL con hostname simbolici: ***URL rewriting***
  - riscrittura URL con indirizzo IP: ***URL-IP rewriting***
- Altre possibilità
  - Redirezione mediante protocollo HTTP/RTSP
  - Anycast, IP tunneling
- Le società CDN usano nomi altisonanti (es. *Global Traffic Management system*), ma in realtà i principali dispositivi per il routing sono costituiti da **DNS autoritativi** arricchiti con funzionalità di controllo sullo stato della rete e degli edge server

# DNS redirection

- Il DNS autoritativo del sito Web delega la risoluzione dell'hostname nell'URL in un indirizzo IP ad un DNS controllato dalla CDN
  - Nel mapping da hostname a indirizzo IP si può effettuare la scelta di un determinato content server
- Il valore del Time-To-Live (TTL) assegnato dal DNS autoritativo è prossimo a 0, in modo tale che la CDN possa modificare spesso il mapping tra hostname e indirizzo IP per:
  - facilitare il bilanciamento del carico tra gli edge server
  - limitare il caching delle risoluzioni indirizzi



# DNS redirection (2)

- Esistono due tipi di CDN che adottano la tecnica del routing mediante DNS:
  - *full-content delivery* (o *first hit at CDN*)
    - L'origin server è nascosto a tutti, eccetto che alla CDN
    - Il DNS autoritativo dell'origin server è modificato in modo da delegare le richieste di risoluzione di indirizzo al DNS server autoritativo della CDN (*DNS outsourcing*)
    - Tutte le richieste indirizzate all'origin server arrivano agli edge server
  - *partial-content delivery* (o *first hit at origin*)
    - L'origin server modifica nella pagina HTML le URL degli oggetti incorporati nella pagina (soprattutto immagini), in modo tale che siano risolti dal name server autoritativo della CDN (ad esempio, www.foo.com/bar.gif viene modificato in cdn-foo.net/www.foo.com/bar.gif)

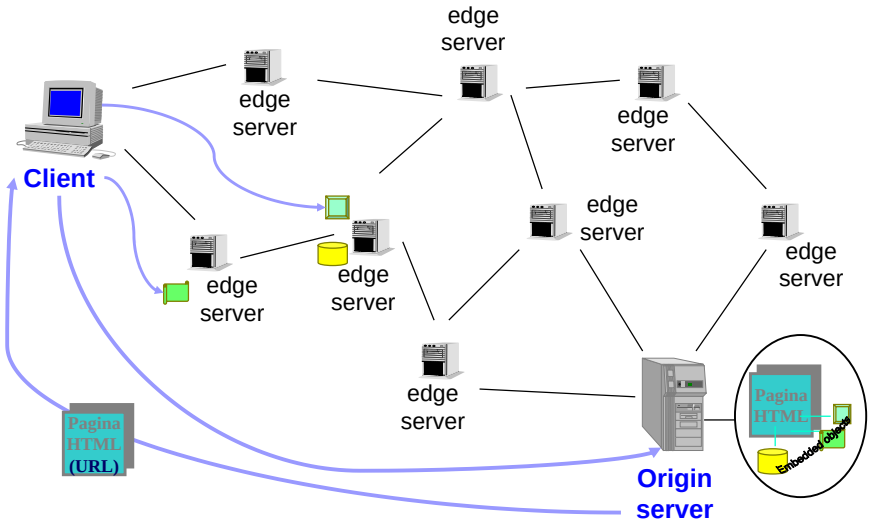


# URL rewriting

- L'origin server riscrive dinamicamente le URL presenti nella pagina HTML base in modo da ridirigere le richieste dei client per gli *embedded object* verso un determinato edge server della CDN
- Due alternative di rewriting
  - URL riscritta con indirizzo IP: indirizzamento diretto dell'edge server
    - Soluzione meno usata
  - URL riscritta con hostname: indirizzamento indiretto dell'edge server
    - Nella fase di risoluzione del nuovo hostname, il DNS server della CDN selezionerà l'edge server opportuno



# Routing mediante URL rewriting



# Diffusione delle CDN

- Alcune società di CND:
  - Akamai http://www.akamai.com
  - Mirror Image http://www.mirror-image.com
  - EdgeStream http://www.edgestream.com
- Diffusione molto rapida dell'uso di CDN:
  - Nel Novembre 1999, soltanto 1-2% dei 700 siti Web più popolari utilizzavano CDN per servire il proprio contenuto
  - Nel Dicembre 2000, il 17% dei 1030 siti Web più popolari utilizzavano CDN per servire il proprio contenuto



# Partial content delivery in Akamai

- L'origin server che vuole delegare il servizio di parte delle proprie risorse all'infrastruttura di edge server gestita da Akamai deve rinominare le URL ad esse relative usando un prefisso specifico
  - Il prefisso include un hostname, ad esempio a799.g.akamai.net
- Risoluzione dell'hostname nell'indirizzo IP di un edge server di Akamai gestita dai server DNS di Akamai
- L'edge server prescelto è "vicino" al name server locale del client



# Partial content delivery in Akamai (2)

- Esempio:
  - Richiesta per la pagina Web http://www.xyz.com/ che contiene la risorsa image.gif
  - L'URL della risorsa http://www.xyz.com/image.gif è riscritta come http://a799.g.akamai.net/3/799/388/9fd0a26b9d5686/www.xyz.com/image.gif
  - Il prefisso a799.g.akamai.net individua un edge server di Akamai che possiede la risorsa
  - Tramite il DNS, il client risolve a799.g.akamai.net in un indirizzo IP, ad esempio 193.206.138.7 (distante solo 5 hop dal client, anziché più di 17 come l'origin server)

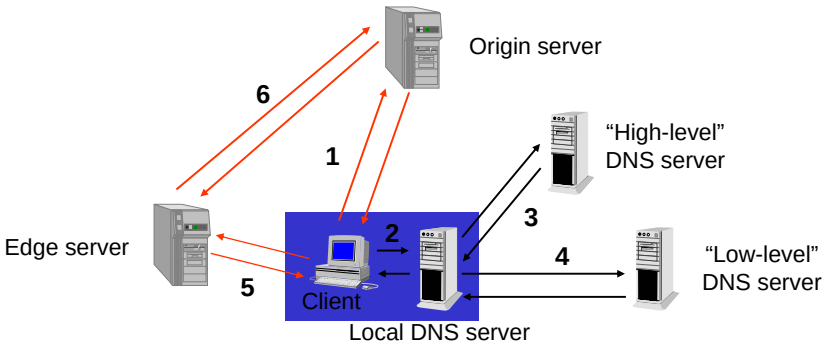


# Partial content delivery in Akamai (3)

- Esempio (*continua*):
  - Se l'edge server 193.206.138.7 possiede la risorsa image.gif, la invia al client
  - Altrimenti, richiede la risorsa tramite un protocollo interno ad un altro edge server di Akamai oppure all'origin server www.xyz.com e la memorizza nella propria cache
  - Un altro client appartenente ad un'altra rete che richiede la stessa pagina avrà un link uguale per l'immagine http://a799.g.akamai.net/3/799/388/9fd0a26b9d5686/www.xyz.com/image.gif ma l'indirizzo IP corrispondente potrebbe essere diverso

# DNS redirection in Akamai

- Akamai utilizza un doppio livello di name server per la risoluzione dei nomi in indirizzi IP degli edge server
  - Esempio (dopo la prima risoluzione del DNS)






# Un esempio: il sito di Torino 2006

- Il sito di Torino 2006 (http://www.torino2006.org) è stato gestito da Akamai con un approccio di tipo full-content delivery
  - Al termine dell'evento il servizio di CDN è stato dismesso
- Come scoprirlo? Usando il comando `dig`

```
claudius:~$ dig www.torino2006.org
; <<>> DiG 9.2.5 <<>> www.torino2006.org
…
;; QUESTION SECTION:
;www.torino2006.org.          IN      A

;; ANSWER SECTION:
www.torino2006.org.     77677  IN    CNAME  torino2006.org.edgesuite.net.
torino2006.org.edgesuite.net. 12877 IN  CNAME  a835.g.akamai.net.
a835.g.akamai.net.      20     IN      A      80.239.170.222
a835.g.akamai.net.      20     IN      A      80.239.170.217
```

TTL pari a 20 sec.

Web cluster



# Un esempio: il sito di Torino 2006 (2)

```
;; AUTHORITY SECTION:
g.akamai.net.      1069   IN    NS    n6g.akamai.net.
g.akamai.net.      1069   IN    NS    n7g.akamai.net.
g.akamai.net.      1069   IN    NS    n8g.akamai.net.
g.akamai.net.      1069   IN    NS    n0g.akamai.net.
g.akamai.net.      1069   IN    NS    n1g.akamai.net.
g.akamai.net.      1069   IN    NS    n2g.akamai.net.
g.akamai.net.      1069   IN    NS    n3g.akamai.net.
g.akamai.net.      1069   IN    NS    n4g.akamai.net.
g.akamai.net.      1069   IN    NS    n5g.akamai.net.

;; ADDITIONAL SECTION:
n0g.akamai.net.    795    IN    A     195.43.191.23
n1g.akamai.net.    2594   IN    A     195.43.191.25
n2g.akamai.net.    2581   IN    A     195.43.191.30
n3g.akamai.net.    795    IN    A     195.43.191.31
n4g.akamai.net.    795    IN    A     195.43.191.23
n5g.akamai.net.    2594   IN    A     195.43.191.25
n6g.akamai.net.    795    IN    A     213.156.44.148
n7g.akamai.net.    2594   IN    A     63.210.47.50
n8g.akamai.net.    795    IN    A     213.156.44.148
```

I name server di Akamai



# Un esempio: il sito di Torino 2006 (3)

- Dove erano localizzati i Web cluster di indirizzo 80.239.170.222 e 80.239.170.217?
- Usando un servizio di traceroute, si scopriva che tutti e due i cluster erano localizzati in Francia
- Usando un altro Internet Service Provider (oppure risolvendo il nome del sito in un diverso momento) era possibile ottenere un risultato diverso per la risoluzione del nome www.torino2006.org in un indirizzo IP

# Content peering

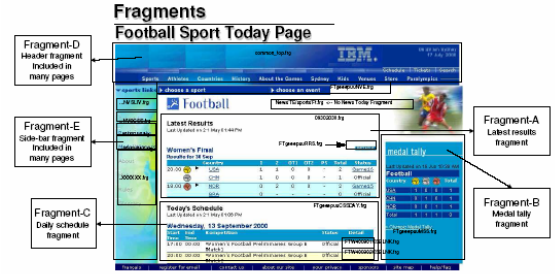
- Cooperazione ed interoperabilità tra più CDN allo scopo di migliorare la scalabilità, la tolleranza ai guasti e le prestazioni
- Le soluzioni proposte per il content peering adottano il routing tra CDN effettuato a livello del Domain Name System
  - Il server DNS (*potenziato*) deve essere in grado di fornire l'indirizzo di una CDN che serve una regione geografica
  - L'intera CDN agisce come un content server, recuperando il contenuto e tariffando il costo dell'operazione alla CDN richiedente



# Caching di contenuti Web dinamici

- Numerose proposte per risolvere il problema, tra cui
  - Active Cache: applet Java eseguiti sui cache server
  - Soluzioni integrate con tecnologie di database, es. Oracle 10g
  - **Frammentazione della pagina**
- Problema: mantenere la consistenza delle copie
- Soluzione basata su frammentazione della pagina

I frammenti di cui è composta la pagina possono essere statici o dinamici, avere diverse caratteristiche di personalizzazione e di durata




Fonte: A. Iyengar, L. Ramaswamy, B. Schroeder, "Techniques for Efficiently Serving and Caching Dynamic Web Content", Book chapter in *Web Content Delivery*, Springer, 2005.



# Tecnologia ESI

- Tecnologia standard Edge Side Includes (ESI) per la frammentazione della pagina
  - Riferimento: http://www.esi.org/
  - Obiettivo: separare le funzionalità di consegna del contenuto e generazione del contenuto per incrementare la scalabilità e ridurre i costi
- Linguaggio di markup basato su XML per descrivere componenti di una pagina Web in termini di frammenti che sono cacheable e non-cacheable
  - Tag XML per specificare frammenti ESI in una pagina Web
- Per ciascun frammento ESI si può specificare un diverso requisito per il caching



# Tecnologia ESI (2)


Il TTL permette di specificare fino a quando è valida la copia del frammento in cache