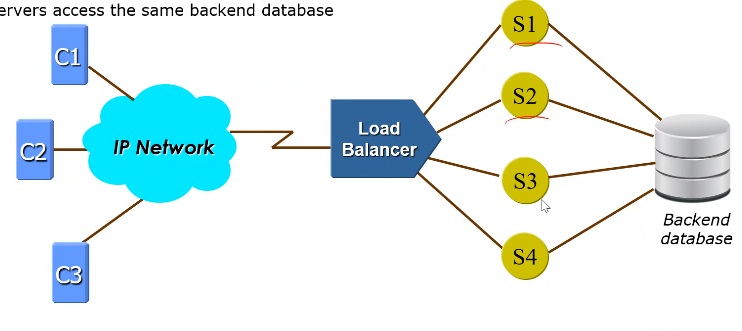
**CDN: Content delivery Network**

----------------------------------------------------------------------------------------------------------------------------------------------

NOTA: cosa i player (Amazon, google, netflix) possono fare per migliorare la **scalabilità**?

Loro hanno bisogno di fornire servizi a diversi utenti (da tutto il mondo, larga scala), possibilmente personalizzati.

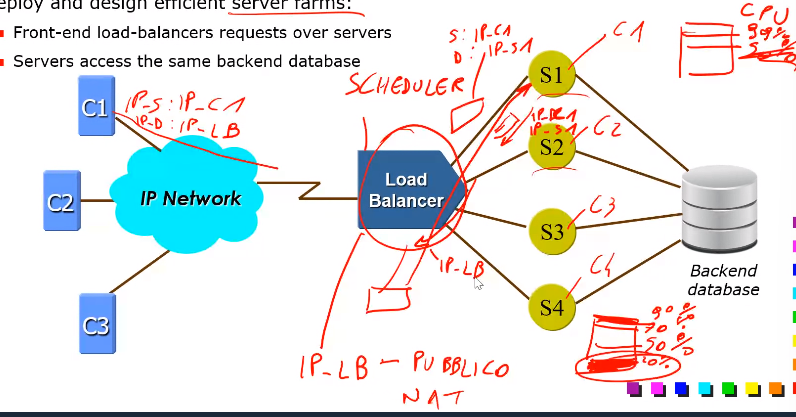
I Player hanno tanti server 🡪 **ServerFarm**

Davanti a tutti i server è presente un **LoadBalancer** che serve per bilanciare il carico tra i diversi server. Dunque, *il LoadBalancer riceve le richieste (pacchetto http) e scegli e dove inoltrare questo pacchetto per* ***massimizzare le performance del sistema o per bilanciare il carico o altri motivi.*** Inoltra quindi le richieste su server più scarichi. *L’OBBIETTIVO NON E’ SEMPRE BILANCIARE*

ALTRE PROSPETTIVE: si potrebbe essere interessati ad altri obbiettivi:

* Consumare meno risorse
* **Consumare meno energie** 🡪 si è visto che dal punto di vista di consumo energetico non cambia nulla se un dispositivo sta lavorando al 50 o al 90%, cambia se va oltre al 90% (0-20 basso consumo ; 20-90 consumo medio, >90 elevato consumo) 🡪 dunque dal punto di visto energetico, se mi serve 100, non mi conviene avere due al 50 ma uno ad 85 e uno a 15.

**Dal punto di vista di networking:**

* **caso 1:** Nel dns potrei avere la entry che punta ad IP del loadBalancer, all’interno il **loadBalancer** capisce che non è lui il destinatario ma deve inoltrare la richiesta ad uno dei server (che ha indirizzo privato quindi **fa da NAT** 🡪 riceve un pacchetto con indirizzo destinazione il suo e lo inoltra ad un server mettendo l’ip di quel server). Server fornisce risposta con ip\_dst loadBalancer e ip\_sorgente il proprio.  
  *I vari server avranno IP privati, il loadBalancer IPpubblico.*
* **Caso2**: i vari nodi possono avere tutti lo stesso IP, il dns (es: di google) fornisce all’utente quell’IP per raggiungere il server, il **load balancer**(che è il punto d’ingresso ai server**) intercetta i pacchetti** e li inoltra ad uno qualunque di quei serve con quell’id. Questo metodo obbliga ad avere i server completamente isolati tra di loro.

**LOAD BALANCER fa da collo di bottiglia (inoltre potrebbe andar bene per gestire richieste per un singolo datacenter ma non per datacenter distribuiti ad esempio)**

*NOTA: carico = CPU, banda, disco, ram NOTA: scheduler: ricerca del server*

potrebbero non esserci load balancer, potrei avere più load balancer per uno stesso datacenter

***CACHE****: in una rete aggiungo un dispositivo di web cache che salva una copia locale della pagina del server a cui un client ha fatto accesso in modo tale che, se un altro client vorrà goggle.com, potrà fornire la pagina direttamente senza accedere al server. 🡪 posso diminuire latenza. (chiamato servizio di proxy)*

*Progettato per mantenere il traffico locale. Con https, ogni pacchetto è cifrato e il contenuto a livello http non è visibile dai proxy 🡪 non funziona il servizio di proxy. Stesso problema se si usano VPN.*

Riepilogo su cosa servono:

* **Limitare traffico a locale** 🡪 operatore ha meno costi (magari evita di passare da alcuni A.S. che dovrebbe pagare in quanto ci sarebbe traffico su link di transito di un altro A.S.)
* Fornire **migliore QoExpirence** 🡪 minimizzare la latenza e migliorare throughput
* Posso avere **del contenuto personalizzato e localizzato**: Se siamo in Italia posso avere delle webCache italiane dove la mia pagina ha pubblicità in italiano.

**HTTP CACHING**: il campo che si salva in cache deve avere una certa validità temporale

1. Data di scadenza

1.1 Vita rimanente

Se arriva una richiesta per una pagina con **cache non scaduta** 🡪 accedo solo alla cache.

Richiesta arriva per pagina in **cache scaduta**: (chiede se ci sono state modifiche da data di scadenza)

* **No** 🡪 Spedisce e aggiorna data di scadenza
* **Si** 🡪 Aggiorna il contenuto e poi lo spedisce

*A livello storico: Web proxies, transparent proxies, SSL (https 🡪 non fa funzionare cache)*

Nota: Google.com vorrebbe il più possibile gestire le richieste di contenuto mentre gli operatori di rete preferiscono evitare per motivi di costi, avendo cache consumano di meno.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Es: streaming video difficile fare in cache

**CDN: Content Delivery Network**

Si basano su una network di cache 🡪 riempio il mondo di cache in modo tale che l’utente accede alla cache più vicina a sé. (Server originale negli stati uniti e tante repliche/cache nel mondo). Avvicinando il router all’utente **migliora QoS, QoE, migliore Availability, migliore load balancing, uso migliore di banda.**

**FOCUS SUL CONTENUTO**: queste repliche vengono fatte dal ContentProvider(google) e non dai provider (TIM, come è con le normali cache).

Per alcuni player è difficile fare questo e perciò esistono anche dei **CDNprovider** che svolgono questo compito al posto loro (es: laStampa si occupa di scrivere notizie e non vogliono implementare CDN, allora si affidano ad un CDN provider).

**Immagine che contiene testo, cartone animato, diagramma, disegno

Descrizione generata automaticamenteCome funzionano:**

* Sorgente emette streaming video
* Arriva al server
* CDN permette di avere una rete di server
* **Contenuto** quindi è **collezionato** al **centrale** che **propaga il contenuto ai vari server dalla CDN** dai quali l’utente vede la partita

Immagine che contiene schizzo, disegno, clipart, Line art

Descrizione generata automaticamente

*Le cache venivano fatte dai Provider per ridurre banda consumata.***La scelta di dove mettere i server per le repliche e quanti, viene fatta dai ContentProvider.**

C*ache reactive: utente deve fare la richiesta prima che questo sia in cache.*  
**SDN proactive:** contenuto propagato prima in cache, **senza aspettare richiesta**.

Cache proxies 🡪 x utente  
**CDN: content provider e client**

**Immagine che contiene testo, schermata, diagramma, linea

Descrizione generata automaticamenteCome funziona?**

* Server principale in cui ho le notizie
* CDN distribution node che distribuisce
* Vari CDN sparsi per il mondo vicini all’utente

Nel caso in cui content provider aggiorna contenuto:

* Aggiornamento server principale che propaga aggiornamento a distribution node che propaga a tutti i server in giro

**Devo modficare tutte le url, invece di puntare al server grosso principale, deve puntare alla cdn vicina.**

SERIE DI DOMANDE:

* **Come** replicare il contenuto
* **Dove** replicare il contenuto
* Come trovare/Accedere al contenuto replicato
* Quale replica associare all’utente?
* Come fornire accesso diretto ai server replica?

**QUALE SERVER? Decisione che spetta al content provider**

* Server **meno carico** 🡪 per bilanciare
* Miglior **performance** 🡪 per migliorare performance
* **Qualsiasi** sia vivo 🡪 fornisce fault tollerance
* **Geograficamente** 🡪 più vicino

**Come direzionare i client verso un particolare server?**

* Uso **DNS**:
  + Dns fornisce direttamente l’ip di una certa replica
  + Ottima per routing time 🡪 **vado direttamente sul server che mi interessa**
  + Nel dns non posso fare decisioni molto precise (**no fine-grained**)
  + Non posso differenziare molto il contenuto dato che guardo solo il nome richiesto(non guardo tutta url)
  + Le richieste al dns, molte volte non le fa il client stesso ma il dns resolver
  + Non riesco a fare decisioni basate sul carico attuale dei server
* **Uso http redirect**:
  + Noi facciamo richiesta http verso il server
  + In http si fa get
  + Server risponde con redirect
    - Se non la ha, dice di contattare un altro server
  + **Posso fare decisioni molto specifiche 🡪 fine-graned**
    - Utente1 lo mando dal server A
    - Utente2 lo mando dal server B
  + Ho aggiunto del carico 🡪 traffico verso il server originale
* **Anycast** 🡪 traffico ad uno qualsiasi (in quanto sono tutte valide)
  + Trasparente al client
  + Ci pensa infrastrutture di rete
  + Trasparente al controllo 🡪 non posso controllarlo 🡪 **decisione fatta da router scegliendo uno qualunque**

# DNS basate su CDN

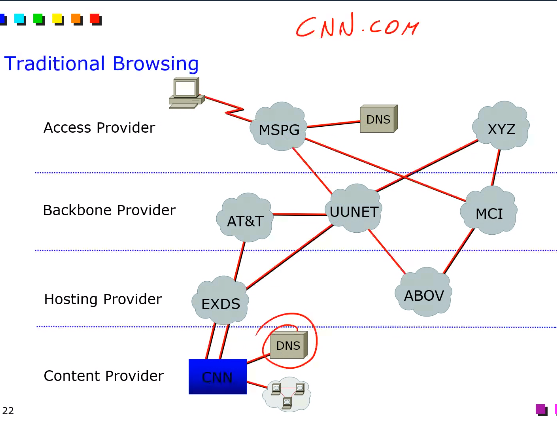
Richiesto per cnn.com, aggiungendo IPsorgente(IP-SA) (ip che molto spesso è l’ip del dns dell’operatore).

Nomi dell’host usati per ridirigere il traffico alla miglior replica 🡪 scelta dopo che avviene la traduzione da nome ad indirizzo IP.

Dns server svolgono il ruolo di **Content Routers**:

* **Replica routing table**: molte repliche a livello 3 (nome+ ip) (non vedono il contenuto della richiesta) (es: RTT, Server Load, Layer 3 metrics, response time)
* Metric misure difficili
* Fine graned ????????? forse ha detto difficili

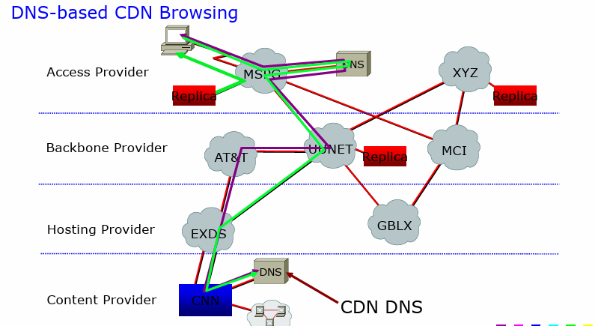
**MODO TRADIZIONALE**

 Immagine che contiene testo, diagramma, schermata, linea

Descrizione generata automaticamente Immagine che contiene testo, diagramma, schermata, linea

Descrizione generata automaticamente

**LOADING TIME**: tempo tra quando invio la prima richiesta e ricevo l’ultima risposta

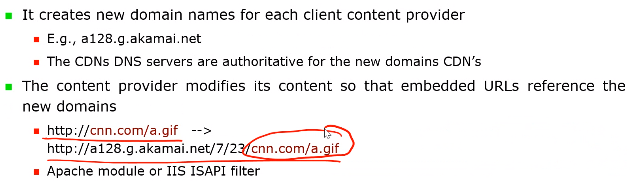
****

**MOTO CON CDN BASATE SU DNS**

* **DNS** **mi risponde dicendo che vicino a me c’è una  
   replica** e di prendere da lì ciò che cerco senza accedere  
  alla serverFarm.
* Molto spesso: pagina web html dal server principale,  
  il vario contenuto attraversa le repliche attraverso  
  akamalization (nome della cosa che sto aggiungendo + altro coso lungo che serve a far capire ad akamali che quel nome è associato ad una specifica replica)

Dunque **akamali aggiunge prefisso** per identificare la replica

Limitazioni: chiedo per host name e non per url, non riesco a splittare richieste grandi, difficile usare stesso hostName per contenuto statico e dinamico



usando più nomi per ognli client, posso suddividere il contneuto in puù gruppi

Immagine che contiene testo, Carattere, schermata, algebra

Descrizione generata automaticamenteCome funziona: se contenuto già in cache, faccio richiesta a quello di più basso livello che mi fornisce già il contenuto

Immagine che contiene testo, diagramma, schermata, linea

Descrizione generata automaticamenteContenuto non cached:

Immagine che contiene testo, diagramma, schermata, linea

Descrizione generata automaticamenteContenuto cached

**URL BASED CDNs**

URL usate per ridirigere il traffico al miglior Content server  
Per far ciò serve che la connessione termini con TCP 🡪 non ottimo perché complessa, costosa, ritardi

**Concludendo**: CDN vengono usate sia per traffico web che per streaming multimediale

* Le repliche avranno contenuti leggermente diversi (ad esempio, pubblicità locale)
* Il peering dei contenuti è ancora un problema irrisolto

-slide 30