Caching și replicare în sisteme distribuite

Student,

Mihaela Trașcă, M2 IISC

Cuprins

1. Introducere 5
2 Sisteme de replicare și de caching 7
2.1 Diagrama sistemului 7
2.2 Caching de proxy 7
2.3 Caching web ierarhic 10
2.4 Caching in sisteme distribuite 12
2.5 Replicarea folosind multicast 14
2.6 Diferenta intre caching si replicare 14
3 Replicarea 17
3.1 Relatia dintre client si replicare 17
3.2 Relatia între două replici 17
4. Caching 19
4.1 Comunicatia dintre client si cache 19
4.2 Comunicatia intre sistemele de caching 20
4.3 Caching de proxy inversat 21
5. Elemente de comunicatie in retea 23
5.1 Protocolul WCCP de web cache 23
5.2 Protocolul de control Transparent Proxy Agent Control 23
Bibliografie 24

1. Introducere

Fenomenul Internet a apărut în urmă cu câțiva ani. Într-un interval scurt de timp Internetul a devenit o sursă de informații și, numărul de persoane adăugate la World Wide Web a crescut exponențial. A devenit un punct central al comerțului electronic, servicii de informare și educație. Acest lucru a condus la creșterea oportunităților de afaceri pentru toți cei implicați în oferirea serviciilor de Internet. Toată lumea cu o miză în afacerea de construire a internetului de mâine - furnizorii de conținut, furnizorii de servicii de rețea, companii de telecomunicații, producătorii de hardware, dezvoltatorii de software lucrează sub presiunea de creștere constantă. Această presiune conduce la o latență a crescută pe Web. Latența poate fi din mai multe motive. Serverele pot fi suprasolicitate din cauza traficului crescut mai ales atunci când există o creștere bruscă în cererea pentru un anumit tip de resursă. Rețelele se pot aglomera datorită creșterii traficului și acest lucru este foarte frecvente în legăturile transoceanice care de multe ori costă milioane de dolari pe lună.

Într-un efort de a reduce această latență indusă, industria comunicațiilor a venit cu mai multe propuneri. Un mod comun de a face acest lucru este cel de a actualiza resursa încărcată: un server mai rapid, un comutator mare, reengineering rețea. Totuși, această abordare nu este întotdeauna din punct de vedere economic fezabilă și, mai important, de asemenea, nu ia în considerare numeroasele părți implicate într-o singură tranzacție Web. În plus față de utilizator și furnizorul de informații, mai mulți furnizorii de servicii Internet și puncte de schimb participă la schimbul de informații. O tranzacție este la fel de bună ca cea mai slabă verigă a rețelei.

Memorarea în cache s-a dovedit un instrument util în reducerea finală de latență a unui utilizator în Web.

- Cache-ul diminuează nevoia de banda de rețea, de obicei, cu 35% sau mai mult, reducând traficul de la browser la serverele de conținut.
- Memoria cache poate îmbunătăți Quality of Service (QoS) pentru utilizatorii de browser prin furnizarea a unor conținuturi cu o lărgime de bandă mai mare și reducerea latenței.
- Memoria cache adună informații mai bune de gestionare a rețelei și permite managementul mai inteligent rețelei.

Memoria cache este o locație de stocare temporară a informațiilor copiate. Există peste un miliard de pagini (sau obiecte) pe internet. Mai mulți utilizatori accesează aceleași obiecte. Un exemplu ar fi sigla Yahoo.com care apare în aproape toate paginile Yahoo. Imaginea trebuie să fie livrată la browser de fiecare dată când browserul accesează oricare dintre paginile Yahoo și aceste pagini sunt solicitate de mai multe ori în fiecare zi, de către diferiți utilizatori.

Un cache Web este un sistem informatic dedicat care va monitoriza cererile de obiecte și stochează obiectele, cum le preia de pe server. Cu privire la cererile ulterioare memoria cache va livra obiecte din depozitarea acestora, mai degrabă decât transmisia cererii către serverul de origine. Fiecare obiect web se schimbă în timp și, prin urmare, are o durată de viață utilă sau

"freshness". Când durata de viață a unui obiect expiră, este responsabilitatea cache-ului Web să obțină o nouă versiune a obiectului. Cu cât numărul de cereri pentru același obiect este mai mare cu atât mai eficient va fi cache-ul Web în reducerea traficului din amonte și va contribui, de asemenea, la reducerea încărcării serverelor, care rezultă în mai puțină latență.

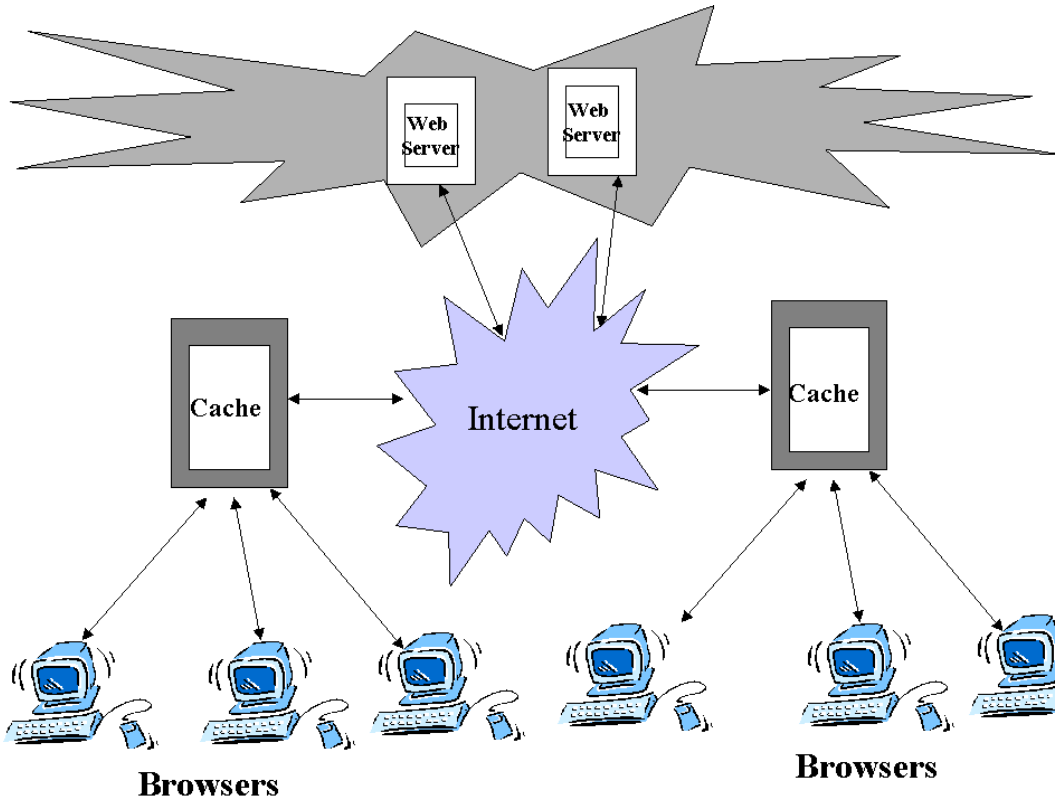

Fig. 1 Arhitectura de bază

Arhitectura de bază a unui sistem web caching este prezentată în fig.1. Sistemele de caching sunt dislocate la granița de rețea, de obicei, la punctele de intrare într-o rețea. Aceste sisteme de caching vor monitoriza traficul care iese prin ele și îndeplinesc funcțiile de cache așa cum este explicat mai sus.

Replicarea este o tehnică similară cache-ul, dar este, în general, considerată a fi mai activă. Procesul de replicare copiază conținutul cache și împinge conținutul de la unul sau mai multe servere de cache din întreaga rețea. Replicarea este obligată să difuzeze obiecte printre servere pentru a menține prospețimea conținutului pe mai multe servere, ceea ce duce la reducerea traficului în rețea. În mod obișnuit același conținut este împins în mai multe mașini ceea ce face mai eficientă utilizarea multicast. Replicarea este critică în operațiunile globale, în cazul în care costul de trafic internațional este mare și trebuie să fie găsite modalități pentru a reflecta date, fără a utiliza prea multă lățime de bandă.

2 Sisteme de replicare și de caching

2.1 Diagrama sistemului

Figura 2 descrie componentele care alcătuiesc un sistem de caching Web și replicare, cu comunicarea între componente.

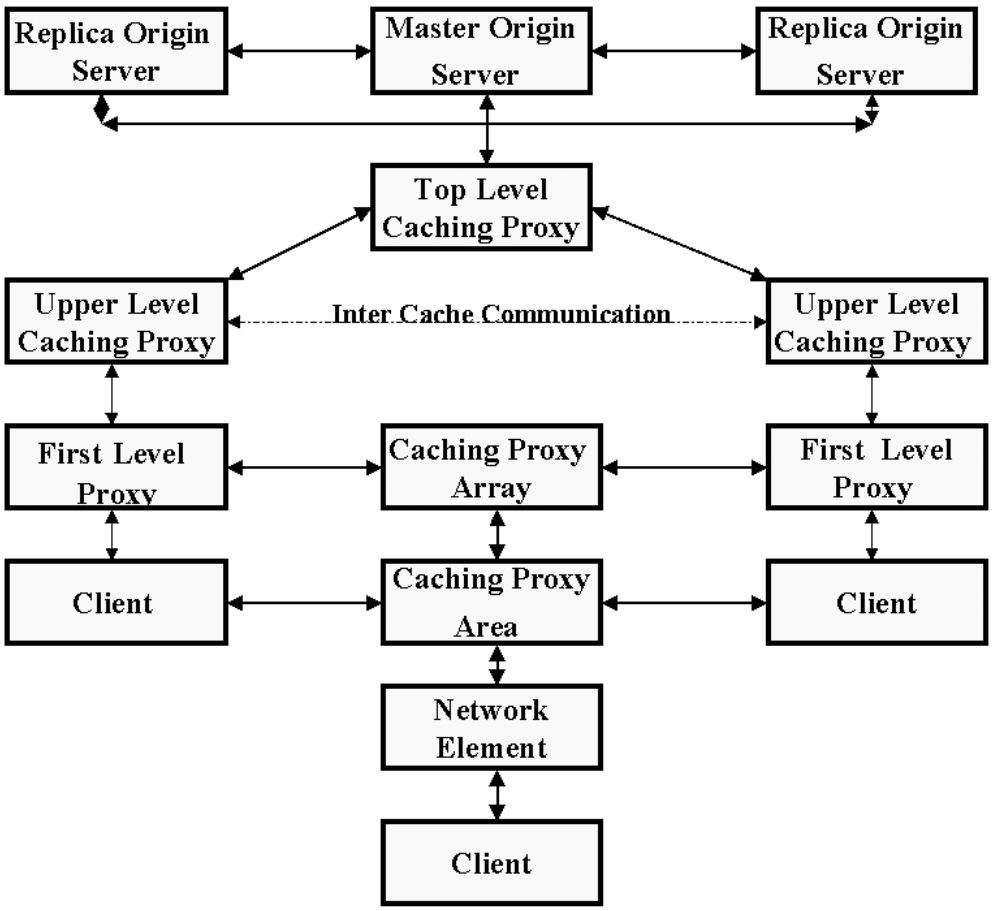

Fig. 2 Diagrama sistemului de replicare și de caching

2.2 Caching de proxy

Serverele proxy sunt, de obicei, utilizate în rețele la punctele de intrare în rețea și, în multe cazuri, pentru a preveni ca traficul nedorit să intre în rețea și pentru a monitoriza traficul

care iese din rețea. Datorită locației sale, este logic să implementeze funcționalitatea cache pe servere proxy. Prin urmare, de aici și denumirea de cache de proxy.

Cache-ul de proxy ajută rețelele de întreprinderi și furnizorii de servicii de Internet să realizeze beneficiile cache-ul prin partajarea unui obiect din memoria cache între mai multe obiecte. În cazul în care utilizatorul rețelei solicită un obiect, cererea este mai întâi trimisă la proxy. În cazul în care proxyul are obiectul cerut atunci proxyul trimite obiectul clientului solicitant. Dacă obiectul solicitat nu este la proxy, atunci serverul va genera o nouă cerere și pe seama clientului, obiectul este preluat de pe server, se păstrează la nivel local și se trimite o copie înapoi la clientul solicitant. Când un client diferit solicită obiectul, serverul poate satisface cererea local. Avantajul este că, dacă un utilizator solicită un anumit obiect, atunci acesta este disponibil pentru restul utilizatorilor rețelei până în momentul în care "prospețimea" obiectului expiră.

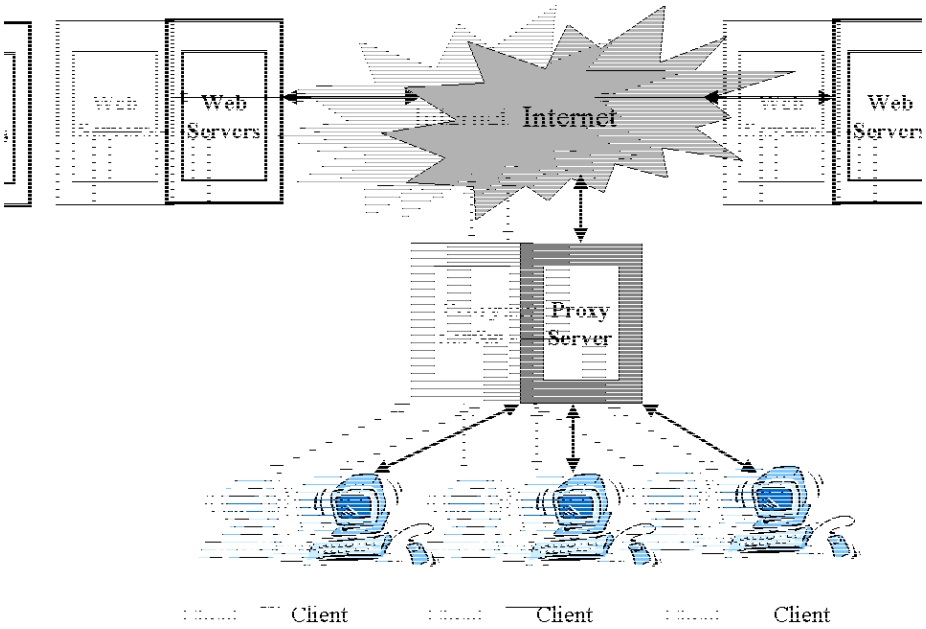

Fig. 3 Cache-ul de proxy

Caching-ul de proxy tradițional necesită configurarea manuală a browser-ului, astfel încât browser-ul sau orice alt utilizator cunoaște adresa proxy. Ar fi mai bine dacă am putea intercepta traficul web fără a mai configura fiecare browser. Această transparență înseamnă că utilizatorul nu trebuie să fie conștient de proxy.

Un cache Web este declarat a fi transparent în cazul în care clienții pot accesa memoria cache fără a fi nevoie pentru a configura browser-ul, fie folosind un URL de configurare automată sau manuală a proxyului. Cache-ul transparent se potrivește perfect în infrastructura

rețelei curente. Ele funcționează mai mult ca firewall-uri transparente, decât un set de servere proxy discrete. Este logic pentru a încorpora cache-ul transparent în rețea din cauza cantității reduse de date implicate. Totuși, odată inclus în rețea cache-ul transparent trebuie să fie la fel de scalabil și rezistent la defect ca restul rețelei.

O cerere HTTP poate fi redirecționată într-o memorie cache de proxy în moduri diferite. Cel mai popular mod este de a utiliza un comutator L4. Comutatorul L4 va prinde pachetele IP cu un port de destinație 80 și transmite pachetele la un cache de proxy. Cererea HTTP necesită ca o conexiune TCP să fie stabilită și primului pachet să fie un pachet SYN. Comutatorul L4 va transmite acest pachet la cache-ul de proxy. Memoria cache de proxy va acționa ca server și va transmite un pachet SYN / ACK corespunzător adresei IP a serverului de origine în loc de a lui. Clientul va crede că pachetele vin de la serverul de origine și funcționarea memoriei cache de proxy va continua așa cum este descris în paragraful de mai sus.

În cazul caching-ului transparent alt cache proxy care se află între memoria cache proxy curentă și serverul de origine poate intercepta cereri din memoria cache-ului de proxy curent. Prin urmare, atunci când clientul primește în cele din urmă obiectul solicitat, clientul nu știe dacă acesta este serverul de origine sau cache-ul de proxy sau un alt cache de proxy intermediar care a răspuns la cerere, deoarece header-ul IP din răspuns conține întotdeauna adresa IP a serverului de origine. Memoria cache proxy poate sau nu să stea în calea directă dintre serverul de origine și client. Dacă nu, atunci, în caz de eșec de cache, cu condiția ca și comutatorul să monitorizeze cache-ul, switch-ul poate redirecționa traficul direct la server. De asemenea, este posibil ca switch-urile să facă echilibrarea sarcinii, prin distribuirea traficului între serverele web. Această metodă de caching transparent, necesită ca un protocol să ruleze între comutatoare și cache sau între cache-uri.

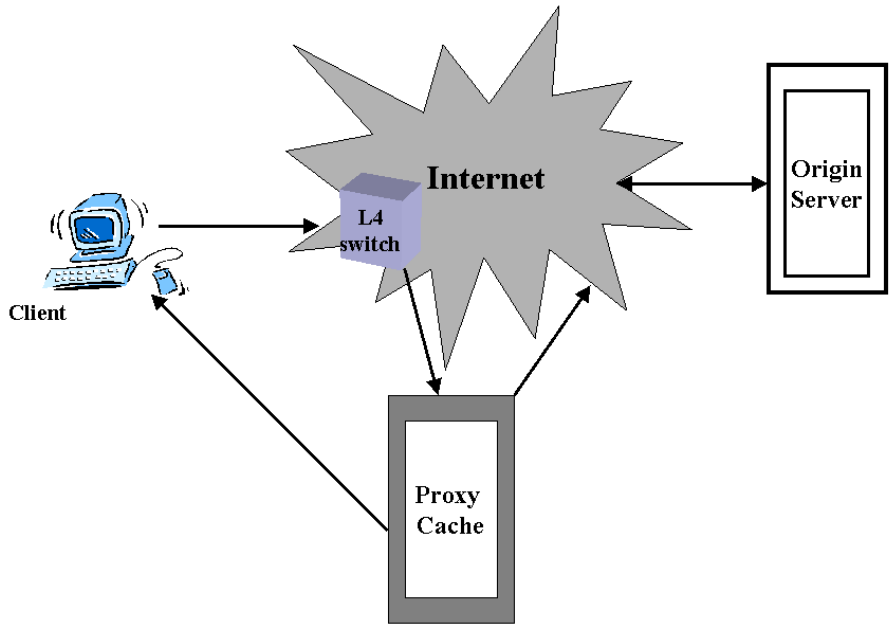

Fig. 4 Caching Transparent

2.3 Caching web ierarhic

Caching-ul de proxy subliniază punctul în care schimbul este util. Dar fiecare cache are o dimensiune finită și există o limită a numărului de obiecte care pot fi memorate în cache. Un singur cache va reduce cantitatea de trafic generat de clienți. În mod similar, un grup de cache Web poate beneficia de distribuirea unui anumit cache Web. Prin urmare, caching-ul a fost implementat ierarhic așa cum este prezentat în figura 5. Într-un mesh de cache sau o memorie cache ierarhică, un cache va stabili relații cu alte cache-uri din apropiere. Există două tipuri de relații: mamă și frate. Un cache-mamă este, în esență, un nivel mai sus într-o ierarhie cache. Un cache-frate este la același nivel. Termenii "vecini" și "la egal la egal" sunt folosite pentru a se referi fie la părinții sau frații, care sunt la un hop depărtare.

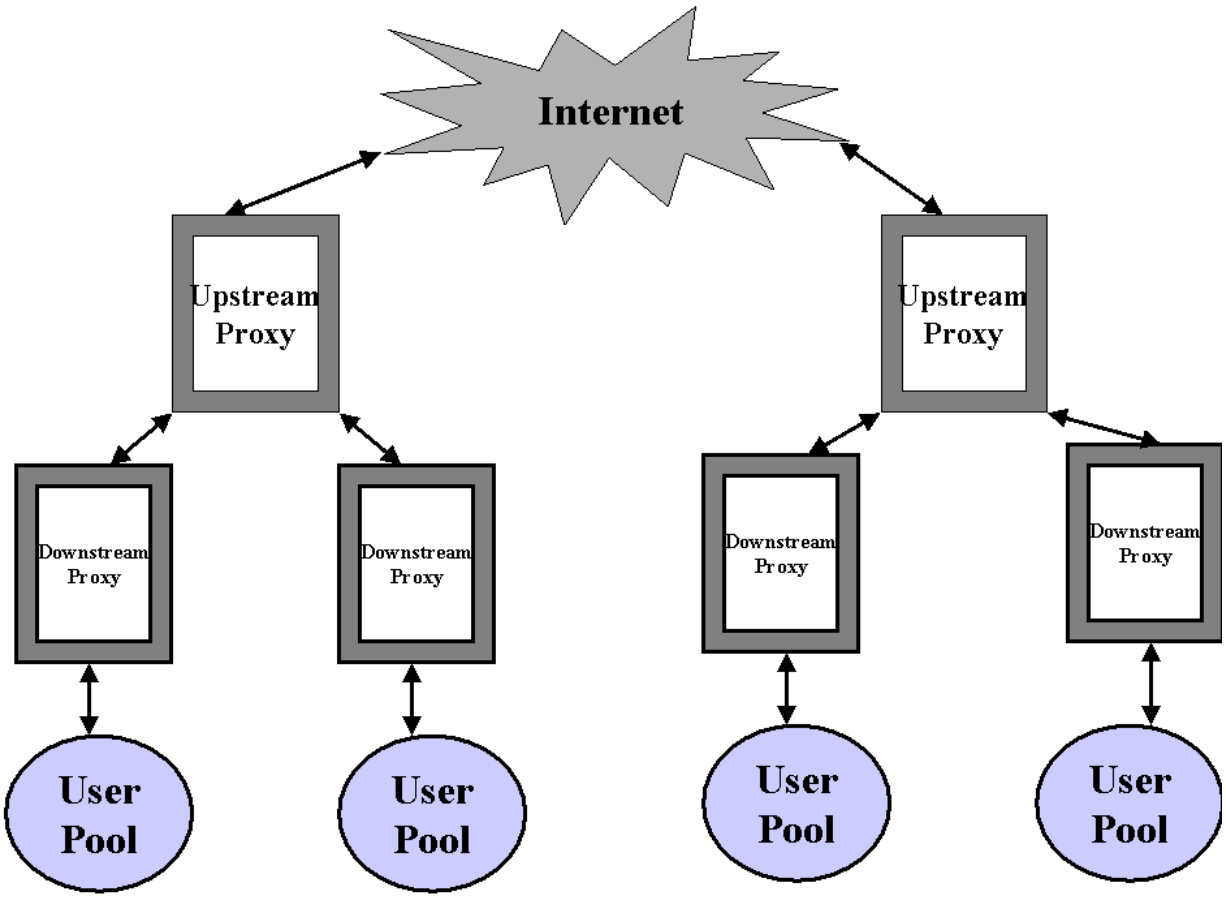

Fig. 5 Caching ierarhic

Fluxul general al documentelor este în sus în ierarhia cache. Când un cache nu deține un obiect, acesta poate solicita, prin intermediul ICP, dacă vreunul dintre vecinii săi ICP are acel obiect. În cazul în care oricare dintre vecinii săi are obiectul ["neighbour hit"], atunci îl preia de la acel vecin. Dacă niciunul dintre vecini nu are obiectul ["neighbour miss"], cache-ul trebuie să înainteze cererea fie la un părinte sau direct la serverul de origine. Părintele va interoga recursiv până în momentul în care obiectul este găsit. Diferența esențială dintre mamă și frate este că un "neighbour hit" poate fi descărcat de pe ambii, iar un " neighbour miss " poate fi preluat numai de la un părinte. Cu alte cuvinte, într-o relație frate, o memorie cache poate cere de la frate numai obiecte pe care fratele le are în cache deja în timp ce același cache poate cere unui părinte pentru a prelua orice obiect, indiferent dacă este sau nu în cache.

Squid permite ierarhii complexe de cache. De exemplu, se poate preciza dacă un anumit vecin poate folosi doar o anumită clasă de cereri, precum URL-urile de pe un domeniu DNS specificat. În plus, este posibil să se trateze un vecin ca frate pentru unele cache-uri și ca părinte pentru alții.

Modelul ierarhic de cache descris aici include o serie de caracteristici pentru a preveni cache-ul de nivel superior de a deveni puncte de sufocare. O modalitate este de a restricționa părinții pe domenii. O altă modalitate este de a se asigura că cache-ul cere înainte la vecinii săi. O clasă mare de cereri web sunt exemple de neatins: interogări de baze de date, autentificarea, datele sesiunii criptate. Cache-ul de nivel inferior ar trebui să recunoască astfel de solicitări și trebuie asigurat că acestea nu sunt trecute prin cache-mamă.

2.4 Caching în sisteme distribuite

Implementările de cache ierarhic tradiționale au arătat următoarele probleme:

- cererea poate să se deplaseze mai multe hopuri într-o ierarhie pentru a ajunge la date și datele pot traversa mai multe hopuri înapoi pentru a ajunge la clienți.
- În al doilea rând, ratările de cache vor fi întârziate în mod semnificativ prin faptul că se traversează ierarhia.
- Partajarea datelor între cache.
- Partajarea cache de nivel superior poate fi prea departe de client și timpul pentru obiect de a ajunge la client este inacceptabil.

Pentru a depăși aceste neajunsuri introducem caching în sisteme distribuite care permite distribuirea proxy-urilor de cache din punct de vedere geografic pe distanțe mari și încearcă să depășească unele dintre dezavantajele cache-ului ierarhic tradițional. Memoriile cache sunt organizate în clustere cache cu o ierarhie definită între ele așa cum se arată în figura 6. Acest lucru se realizează prin utilizarea optimizărilor care să permită definirea de relații complexe între cache.

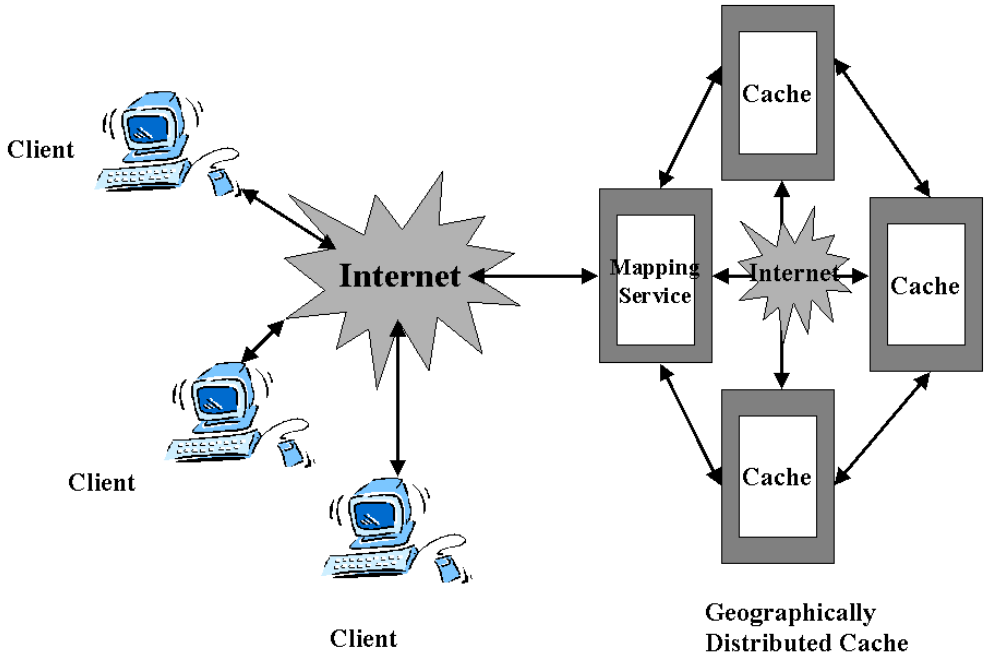

Fig. 6 Caching în sisteme distribuite

Caching-ul distribuit are mai multe avantaje:

- Distribuția de încărcare pe server, cu proxy-uri din aval.
- Îmbunătățirea performanței clientului prin aducerea cache-ului mai aproape de client.
- Îmbunătățirea ratei de cache datorită creșterii în spațiu a cache.

Există mai multe strategii care pot fi utilizate pentru proiectarea serverelor de cache distribuite.

Prima strategie are 3 caracteristici importante:

- Separarea datelor și căile de metadate și menținerea unei ierarhii de căi de metadate care urmăresc locațiile în care au fost stocate de copii ale datelor.
- Menținerea indiciilor de localizare, astfel încât cache-ul poate localiza exemplare din jur fără prea multe date
- Folosind cache direct la transferurile de date cache pentru a evita stocarea și întârzieri de redirecționare.

Utilizarea de indicii de localizare permite memoriei cache localizarea unei memorii cache vecine cu o copie a datelor și inițierea unui cache pentru a transfera date fără a trece prin ierarhia completă. Cache-ul în sistem va detecta ratări la nivel local, folosind indicii și trece direct la

server. O ierarhie meta-distribuție este folosită pentru a distribui în mod eficient indicii și stochează datele la baza ierarhiei.

A doua strategie utilizează distribuirea de cache în care datele sunt partajate în apropierea clienților prin realizarea unei estimări corecte a ceea ce clienții pot accesa în viitorul apropiat. Astfel, înlătură penalizările clienților, care sunt întotdeauna experimentate atunci când clientul accesează prima dată obiectului.

2.5 Replicarea folosind multicast

Replicarea folosește multicast pentru a împinge date pe serverele oglindă de origine. Replicarea datelor și distribuirea în funcție de necesități se numește oglindire. Replicarea pură diferă de cache în sensul in care sistemele de caching "trag datele" de la serverul de origine, în timp ce sistemele de replicare au tendința de a face "push" datelor pentru a menține copii în oglindă ale aceloraşi, date la diferite locații în rețea. Cel mai bun efort (“best effort”) utilizează multicastul UDP/IP, în timp ce multicastul de încredere are nevoie de mașini suplimentare de protocol, pentru a recupera pierderile de pachete. Multicastul “best efford” pur și simplu profită de IP Multicast în disemnarea de obiecte. Cu toate acestea obiectele sunt stocate în mașini la distanță numai dacă obiectul este încă intact. Codul de control poate fi folosit pentru a verifica dacă obiectele livrate sunt intacte. În cazul în care obiectele sunt intacte acestea pot fi salvate în timp ce obiectele care nu intacte sunt eliminate. Multicastul “best effort” deși este un protocol ușor poate eșua atunci când rata de eroare este mare, în cazul în care un anumit pachet este aruncat poate fi necesar să se renunțe la întregul obiect. Multicastul de încredere pe de altă parte utilizează UDP / IP multicast pentru cel mai bun efort de difuzare, dar folosește protocol bazat pe ACK / NACK pentru a detecta pierderile de pachete și le retransmite. LSAMexplained a utilizat mai târziu conceptul de canale multicast definite care permit multicast doar la un set de clienți în locul tuturor clienților.

2.6 Diferența între caching și replicare

Sistemele de caching:

- reduc latența rețelei prin aducerea conținutului mai aproape de consumator.
- sunt, în esență, reactive în timp ce un obiect de date este stocat în cache numai atunci când clientul solicită acest lucru.
- îndeplinesc obiectivele de reducere a traficului prin obtinerea de conținut la cerere.
- au probleme de consecvență, datorită naturii lor reactive
- pot avea probleme de fiabilitate, deoarece acestea sunt în mod normal plasate în punctele de intrare în rețea și un eșec cache poate aduce, uneori, întreaga rețea în jos.

Sisteme de replicare:

- ştiu exact când un obiect se schimbă şi împing obiectul imediat.
- asigura prospeţime conţinutului din cauza naturii lor reactivă.
- au o toleranţă foarte mare la defecte ca urmare a replicarii de date, ceea ce asigură că, dacă un server de web pică cererile pot fi redirecţionate către un alt server de origine.
- consumă mai mult spaţiu pe disc.
- trebuie algoritmi eficienţi pentru echilibrarea sarcinii.
- poate creşte traficul în reţea, dacă multicastul nu este utilizat în mod raţional.

3 Replicarea

3.1 Relatia dintre client si replicare

Vom descrie cooperarea și comunicarea între clienți (atât agenți de utilizator și cache proxy) și serverele de replicare. Acestea sunt folosite pentru a descoperi un server web optim. Redirectarea URL este un mecanism simplu și utilizat în mod obișnuit pentru a conecta clienții web cu servere de replici, este de a utiliza protocolul HTTP cod de răspuns 307 Redirecționare temporară. Replica conectată inițial poate decide să se conecteze la servicii sau decide să redirecționeze clientul la serverul de replicare corect. Se bazează pe securitatea HTTP în întregime.

Redirectarea DNS. DNS oferă un client mai sofisticat la politica de replicare. Serverele DNS implementează ordinea de adrese bazate pe calitatea politicilor de servicii. Atunci când un client Web rezolvă numele de server web, DNS poate fi utilizat pentru a comanda adresele IP ale serverelor web începând cu replica cea mai optimă și se termină cu replica puțin optimă. DNS cu toate acestea are un control limitat asupra cererii de a ajunge la replica potrivită. Mai multe servere intermediare au tendința de a face cache URL-ului pentru traducerea IP. Mai mult decât atât, unele browsere stochează în memoria cache de obicei unele translatări de adrese. Alegerea replicii cea mai optimă se schimbă în timp. Prin urmare, sistemul DNS specifică o perioadă de valabilitate de cache-ului rezultatul rezoluției de nume logic. DNS prevede, de asemenea capacități de echilibrare a sarcinii prin distribuirea cererilor către replica cea mai optimă.

3.2 Relația între două replici

Replicarea în oglindă pe loturi presupune că serverul care replică, trebuie să fie actualizat și va iniția comunicarea cu serverul de origine master. Comunicarea se stabilește la intervale pe baza tranzacțiilor din coada de așteptare, care sunt amânate pentru prelucrare. Politica de planificare poate fi bazată pe o serie de factori, dar trebuie să apară într-o perioadă de timp. Odată ce comunicarea este stabilită seturi de date sunt copiate la serverul de replicare. Se bazează pe securitatea protocoalelor care sunt folosite pentru a iniția comunicația datelor și de transfer. FTP și RDIST sunt protocoalele cele mai utilizate.

Replicarea la cerere. În acest model replica dobândește conținutul după cum este necesar, la cerere. Aceasta este urmată în mod normal, prin proxy invers, în cazul în care un client poate solicita un document și atunci când nu se poate găsi, inițiază comunicarea cu serverul de origine.

Replicarea sincronizată presupune că serverul de origine, de replicare va coopera cu ajutorul unor strategii de sincronizare pe baza protocoalelor de replicare, de specialitate pentru a menține seturile replicate coerente. Strategiile de sincronizare variază de la bine coerent în cazul

în care are loc la fiecare câteva minute până la slab coerente, în care are loc sincronizarea la fiecare câteva ore. În mod normal, sincronizarea urmează un model de coerență conceput pentru a reduce lățimea de bandă de rețea necesară pentru a reproduce conținutul.

4. Caching

4.1 Comunicația dintre client și cache

Clientul de memorie cache se ocupă de modurile în care clienții pot găsi cache-ul. Configurarea manuală a proxy-ului, deși acest lucru nu poate fi exact numit ca un protocol, este important să se includă acest lucru. După cum sugerează și numele fiecare utilizator va configura manual clienții săi web și necesită ca utilizatorul să aibă cunoștințe de protocoale proxy și politicile locale. Potențialul de a merge greșit este mare deoarece fiecare utilizator va stabili în mod individual preferințe.

Autoconfigurarea de proxy [PAC] O pagină Javascript pe un server web împarte informații cu privire la modul în care se pot găsi proxy-uri. Clienții trebuie să pointeze la URL-ul acestei pagini. Prin urmare, nici o configurare manuală nu va fi necesară. Acest script va locui la o anumită adresă URL centrală de unde poate fi accesat de toți clienții. Totuși, utilizatorii încă mai trebuie să configureze această adresă URL în browser-ul lor sau client Web. Un avantaj clar al PAC este că administratorii pot actualiza configurația proxy fără intervenția utilizatorului. Prin urmare, ea permite punerea în aplicare a unei politici comune în întreaga organizație.

Web Proxy Auto Discovery [WPAD]. Implementările clienților Web se confruntă cu o gamă amețitoare de protocoale de descoperire a resurselor la niveluri diferite de punere în aplicare și implementare. Această complexitate împiedică desfășurarea unui " Web Proxy Auto Discovery ". WPAD nu definește un nou standard. În schimb WPAD folosește mecanisme de descoperire a resurselor pre-existente în Internet pentru a efectua descoperirea proxy. Astfel, protocolul de WPAD se reduce la furnizarea clientului web un mecanism pentru a descoperi PAC URL. WPAD nu precizează care proxy-uri vor fi utilizate efectiv. Scriptul PAC va decide acest lucru.

Protocolul WPAD specifică următoarele:

- modul de utilizare a fiecărui mecanism pentru scopul specific de descoperire automată a unui proxy web
- ordinea în care trebuie efectuate mecanismele
- setul minim de mecanisme care trebuie să fie încercat de WPAD

HTTP este utilizat pentru a comunica cu un proxy web. Un proxy web poate sprijini, de asemenea alte protocoale. HTTP 1.1 adăugat suport pentru directivele cache. Aceste directive cache furnizează informații cu privire la capacitatea de caching a obiectelor web, timp de valabilitate, precum și alte informații pentru memoria cache de utilizat. Problema de confidențialitate merge în ambele sensuri cu proxy-uri. Proxy-urile pot fi utilizate pentru a anonimiza utilizatorii la serverele web și pot fi utilizate pentru a urmări fiecare mișcare facută pe web a utilizatorului. Redirecționarea utilizatorilor printr-un server proxy oferă acces la toate

informațiile despre utilizarea și conținutul documentelor. Acest lucru poate fi un risc de securitate, sau poate fi util să existe un punct de control pentru comunicare, ca în firewall-uri. Controlul accesului, bazat pe adrese IP nu sunt indicate în antetele HTTP, și astfel cache-ul nu are nici o modalitate de a ști dacă un obiect web are accesul restricționat până în momentul în care antetele suplimentare sunt furnizate de către serverul de origine.

4.2 Comunicația între sistemele de caching

Internet Cache Protocol [ICP] este probabil este cel mai răspândit protocol de cache din lume și a fost folosit pentru prima dată în SQUID. ICP este folosit de cache pentru a interoga alte servere cache despre obiecte web, pentru a vedea dacă un obiect web este prezent în alte cache-uri. Pentru a face acest lucru folosește UDP. Din moment ce UDP este de încredere prin design, o estimare a disponibilității rețelei poate fi calculată prin pierderea ICP. Această măsurare a pierderii rudimentară împreună cu timpi de dus-intors sunt utile în cache pentru echilibrarea încărcării link-urilor. ICP oferă indicii despre localizarea obiectelor web în cache-urile vecine Deși este posibil să se mențină ierarhii cache fără a utiliza ICP, lipsa de ICP sau ceva similar interzice existența unor relații de meta-comunicare între frați, adică mecanisme pentru a interoga cache cu privire la un anumit document. ICP nu transmite informații despre antetele HTTP asociate cu un obiect web. Antetele HTTP pot include controlul accesului și directivele cache. Din moment ce cache-ul cere obiecte, și apoi descărca obiectele folosind HTTP, pot să apară puncte cache false. Deoarece ICP are încredere în validitatea unei adrese a unui pachet IP, este susceptibil de spoofing IP. Securitatea este o problemă cu ICP peste UDP, din cauza naturii sale de conexiune.

HyperText Caching Protocol [HTCP] este un protocol experimental, care este folosit pentru a descoperi cache-ul HTTP și a datelor din memoria cache, gestionarea seturi de cache HTTP și activitate de monitorizare a cache-ului. Acesta poate fi folosit pentru a monitoriza cache-ul la distanță adăugări și eliminări și sugestii despre trimiterea obiectelor web, cum ar fi locațiile obiectelor care se pot insera in cache. HTPC include antete HTTP în timp ce ICP nu. Antetele au informații vitale care pot fi utilizate de către cache proxy. Dacă nu se utilizează în protocol autentificarea este posibil pentru o terță parte neautorizată să modifica un cache folosind protocolul TCP. Protocolul HTCP folosesc autentificarea secretă MD5.

Large Scale Active Middleware [LSAM] Proxy Cache oferă un sistem de cache distribuit care folosește multicast pentru transmisia automată a paginilor web populare. LPC folosește cache proxy care sunt implementate ca o pompă și o ierarhie distribuită filtru așa cum se arată în fig 7. Aceste componente pot urmări în mod automat popularitatea unor grupuri de pagini web și, de asemenea, gestionează în mod automat transmisia de la server către client. Sistemul face cache la pagini ăn punctele de agregare ale rețelei, oferind avantajele unei singure memorii cache centrala, un proxy partajat. Mecanismul monitorizează serverul de grupuri de afinitate. Aceasta va crea și va distruge canale multicast pentru diferite grupuri de afinitate mult sau mai puțin populare. Cache-ul de filtrare este mai aproape de client și are o memorie cache partiționată,

pentru a permite mai multe canale. Cereri individuale declanșează reacțiile multicast, atunci când paginile sunt membri ai unor grupuri de afinitate activi. O cerere este verificat la proxy-uri intermediare și transmisă prin intermediul său la server, la fel ca în ierarhiile proxy convenționale. Răspunsul este multicast la filtre din grup si unicast de la proxyul final inapoi la client. Cererile ulterioare sunt gestionate la nivel local de la filtrele de lângă clienți.

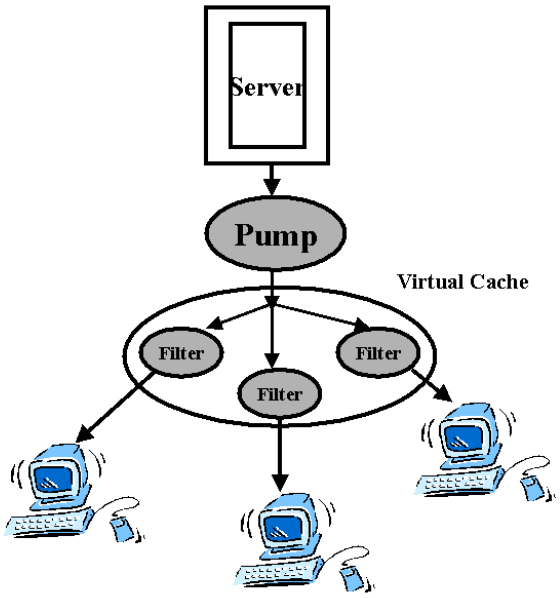

Fig. 7 Relațiile de filtrare și pompare

LSAM folosește un mecanism puternic de tranzacții securitate. Acesta utilizează servicii de semnătură și criptare digitale, după caz, la obiecte MIME.

4.3 Caching de proxy inversat

Motoarele de cache sunt frecvent utilizate de clienti din apropiere pentru a asigura un timp de răspuns mai rapid de rețea și utilizarea lățimii de bandă WAN minimă. Astfel, cache-ul stocheaza conținutul cel mai des accesat de clienți. In plus, memoria cache poate fi, de asemenea, utilizată, în fața fermelor de servere web pentru a crește capacitatea serverelor. Această configurație este numită cache invers, deoarece motoarele cache stochează doar conținutul de la serverele care sunt în față. Această caracteristică este deosebit de importantă atunci când motoarele de cache sunt ferme de servere, în care un anumit conținut este mai popular decât alte tipuri de conținut. Folosirea proxyului invers permite administratorilor să prevină un număr mic de cereri URL care au impact asupra performanței globale ale serverelor. Un alt avantaj al acestui fapt este că acesta nu este necesar să se identifice URL-uri de la care cererea este mare, indicate sau administrate independent de cea mai mare parte a URL-uri pe server manual. Fig 9 prezintă o relație many-to-many de caching de proxy inversată. Nu este o relație dedicată între

server și client. Este mai degrabă o relație many-to-many. Proxyul invers permite echilibrarea sarcinii, în plus față de capacitatea de a scala în mod dinamic.

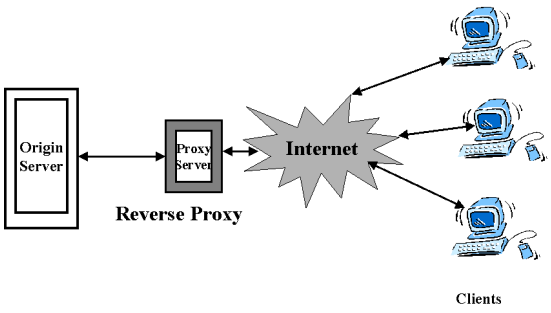

Fig. 8 Caching invers de proxy

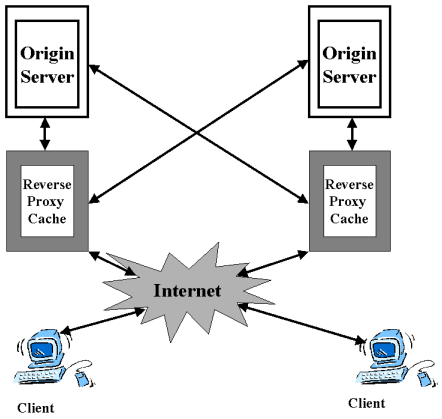

Fig. 9 Caching invers many-to-many

5. Elemente de comunicare în rețea

5.1 Protocolul WCCP de web cache

WCCP este un protocol proprietar CISCO, care rulează pe seria lor de produse Cache Engine 500. Protocolul WCCP este utilizat pentru:

- asociera unui singur router cu unul sau mai multe servere cache în scopul redirecționării transparentă a traficului HTTP.
- permite unui cache web să dicteze modul în care routerul distribuie transparent - redirecționarea traficul între cache-uri web asociate.

Un cache se poate adăuga dinamic la ferma de servere de el insusi. IP-ul sursă al cache-ului este folosit pentru a identifica în mod unic server. WCCP este foarte scalabil. Se folosește un hash de 256-bucket pentru redirecționarea care este stocată pe un router. Orice nou motor cache, care se adaugă este alocat în acest tabel hash. Ori de câte ori vine un nou pachet IP, adresa IP este folosită pentru a indexa în tabelul hash. Obiectele indexate indică la care web cache ar trebui să fie redirecționat pachetul. Motoarele cache pot fi inserate într-un cluster de operare complet. Periodic routerul distribuie o listă de proxy-uri pe care le consideră utilizabil. Aceste informații sunt furnizate pentru cache-urile web desemnate. WCCP permite dinamic "vindecarea", în cazul în care un anumit grup cache eșuează. Router-ul, de asemenea, se asigură că traficul HTTP de la membri ai fermei cache la serverul origine nu este redirecționat.

5.2 Protocolul de control Transparent Proxy Agent Control

TPACT se desfășoară între un element de rețea și, funcționând ca un element de reorientare și în afara căilor de politici transparente proxy caching. Protocolul este similar cu WCCP. Acesta permite unul sau mai multe proxy-uri de cache să se înregistreze cu un singur element de rețea, pentru a primi trafic web redirecționat. Toate proxiurile caching participante, spre deosebire de WCCP funcționează ca un cvorum care dictează distribuția web a traficului.

Bibliografie

1. http://www.cse.wustl.edu/~jain/cis788-99/ftp/web_caching/index.html#BM1_1_Overview_of_Web, accesat pe 8 februarie 2016

2. http://software.ucv.ro/~cbadica/dadr/cap1.pdf, accesat pe 8 februarie 2016

3. http://www.iem.ase.ro/docs_Romanian/Fascicule/2011-2012/Fascicula_5-Arhitectura_de_trading_orientata_pe_servicii.pdf, accesat pe 8 februarie 2016

4. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.36.6546, accesat pe 8 februarie 2016

5. http://www.distributed-systems.net/papers/2007.ic.pdf, accesat pe 8 februarie 2016