Multipath TCP

Părăușanu Dragoș
Master IISC

Cuprins

1. Ce aduce nou MPTCP 3
2. TCP vs MPTCP 4
2.1 Conceptul MPTCP 4
2.2 Conexiunea MPTCP 5
3. Arhitectura MPTCP 6
3.1 Inițializarea conexiunii 7
3.2 Inițializarea unui subflow nou 8
4. Controlul congestiei în MPTCP 9
4.2 Controlul congestiei cuplat pentru MPTCP 12
4.3 Adaptarea link-urilor balansată pentru controlul congestiei 13
Bibliografie 14

1. Ce aduce nou MPTCP

Odată cu evoluția Internetului, cererea de resurse pentru Internet sunt în creștere, dar de multe ori aceste resurse (în special, de lățime de bandă) nu pot fi pe deplin utilizate datorită constrângerilor de protocol atât pe sisteme cât și în cadrul rețelei. Dacă s-ar fi folosit aceste resurse în același timp, experiența utilizatorului final ar putea fi cu mult îmbunătățită. Astfel de îmbunătățiri ar reduce, de asemenea, cheltuielile necesare cu infrastructura care altfel ar fi necesare pentru a crea un echivalent de îmbunătățire în experiența utilizatorului.

Transportul prin multiple căi ține să realizeze anumite obective ale punerii în comun a resurselor prin utilizarea mai multor căi despărțite. Principalele avantaje ale tranportului prin multiple căi sunt:

- mărirea rezistenței conexiunii prin oferirea mai multor căi, protejând utilziatorii finali;
- mărirea eficienței utilizării resurselor, și de asemenea mărirea capacității rețelei disponibile utilizatorilor finali.

MPTCP este o versiune modificată a TCP care implementează transportul informației prin multiple căi și atinge aceste obiective prin punerea în comun a mai multor resurse pentru o cnexiune de transport de date, fiind transparentă aplicației.

Putem privii MPTCP ca un set de extensii pentru protoclul TCP care permit transferul informației prin Internet folosind multiple căi simultan. Acest protocol este concentrat pe utilizarea multiplelor căi utilizator-utlizator unde unu sau ambii utilizatori au mai multe intrări și mai multe adrese.

Pentru a simplifica design-ul, putem presupune că prezența multiplelor adrese la o gazdă este suficientă pentru a indica existența căilor multiple. Aceste căi nu trebuie să fie distincte ele pot împărții între ele multe rute. Și în această situație, utilizarea căilor multiple este benefică deoarece cresc șansele ca o conexiune să nu pice din cauza unor noduri intermediare. Există și algoritmi de control al congestiei care asigură funcționarea corectă si nu una haotică.

Există trei aspecte care țin de compatibilitatea cu TCP:

- Contrangeri externe, protocolul trebuie să funcțționeze cu majoritatea vastă de echipamente deja existente, cum ar fi NAT, firewall-uri și proxy-uri. Mai mult, protocolul nu trebuie să asume că un segment va ajunge pe cablu la destinație nemodificat.
- Contrangeri de aplicații, protocolul trebuie să fie utilziabil fără a aduce modificări la aplicațiile existente care folostesc API-ul TCP. Mai mult, trebuie asigurat același model al serviciului ca pentru TCP.
- Compatibilitatea cu TCP, protocolul va trebui să fie capabil să funcționeze ca un TCP standard fără intervenția utilizatorului atunci când se folosesc host-uri care nu suportă MPTCP.

2. TCP vs MPTCP

MPTCP funcționează la nivelul transport și își propune să fie transparent pentru ambele straturi superioare și inferioare. Este un set de caracteristici suplimentare pe partea de sus a TCP. Figura 1 ilustrează această stratificare. MPTCP este proiectat pentru a fi ușor de utilizat de către aplicații mai vechi cu nici o modificare.

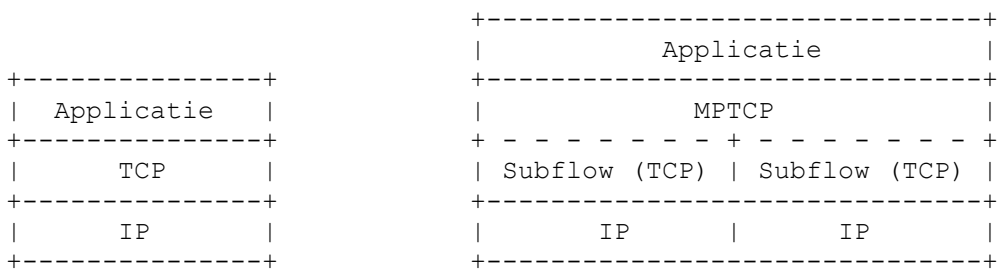

Figura 1: Compararea standardului TCP cu MPTCP

2.1 Conceptul MPTCP

Această secțiune oferă un rezumat la nivel înalt de funcționare normală a MPTCP, și este ilustrat de scenariul prezentat în Figura 2.

La o cerere de non-MPTCP-conștient, MPTCP se va comporta la fel ca TCP. API-uri extinse ar putea oferi control suplimentar la aplicații-MPTCP. O cerere începe prin deschiderea unui soclu TCP în mod normal. Semnalizarea și operarea MPTCP sunt manipulate de către implementarea MPTCP. O conexiune MPTCP începe în mod similar cu o conexiune obișnuită TCP. Acest lucru este ilustrat în figura 2, unde o conexiune MPTCP este stabilit între adresele A1 și B1 pe Gazde A și B, respectiv. Dacă sunt disponibile căi suplimentare, sesiuni TCP suplimentare (denumite MPTCP "subflows") sunt create pe aceste căi, și sunt combinate cu sesiunea existentă, care continuă să apară ca o singură conectarea la aplicațiile de la ambele capete. Crearea sesiunilor suplimentare TCP este ilustrată între Adresa A2 pe Host A și adresa B1 pe Host B. MPTCP identifică mai multe căi prin prezența adreselor multiple la gazde. Combinații ale acestor adrese multiple echivalează cu căile suplimentare. În exemplul, de alte aplicații potențiale căi care ar putea fi înființate sunt A1 <-> B2 și A2 <-> B2. Deși această sesiune suplimentară este prezentată ca fiind inițiată de A2, ar putea la fel de bine să fie inițiată de B1.

Descoperirea si configurarea subflow-urilor suplimentare vor fi realizate printr-o metodă de gestionare a calăilor. MPTCP adaugă numere de secventa la nivel de conexiune pentru a permite reasamblarea segmente care sosesc pe subflowuri multiple cu diferite întârzieri de rețea. Subflowurile sunt terminate ca și conexiunile TCP regulate, cu un four-way handshake FIN. Conexiunea MPTCP este terminată printr-o conexiune FIN.

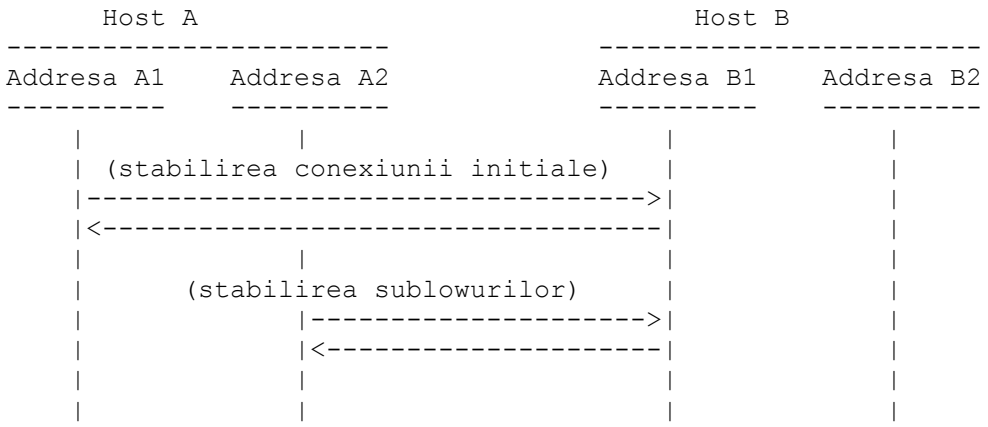

Figura 2: Examplu MPTCP - scenariu

2.2 Conexiunea MPTCP

O conexiune TCP Multipath furnizează un flux de octeți bidirecțional între două gazde care comunica prin TCP și, prin urmare, nu are nevoie de nici o modificare la aplicație. Cu toate acestea, Multipath TCP permite gazdelor de a utiliza căi diferite, cu diferite adrese IP pentru a face schimb de pachete care fac parte din conexiunea MPTCP. O conexiune TCP Multipath apare ca o conexiune normală TCP la o aplicație. Cu toate acestea, în nivelul rețea, fiecare subflow MPTCP arata ca un flux TCP regulat ale cărui segmente transporta un nou tip de opțiune TCP. Multipath TCP gestionează crearea, eliminarea și utilizarea acestor subflowuri pentru a trimite date. Numărul de subflowuri care sunt gestionate în cadrul unei conexiuni Multipath TCP nu este fix și poate fluctua în timpul duratei de viață a conexiunii Multipath TCP. Toate operațiunile MPTCP sunt semnalate cu o opțiune de TCP - un singur tip numeric sau MPTCP, cu "sub-tipuri" pentru fiecare mesaj MPTCP.

În arhitectura tradițională a Internet, dispozitivele de rețea funcționează la nivelul rețea și nivelurile inferioare, cu nivelurile de deasupra rețelei instanțiate numai pe gazdă. În timp ce această arhitectură, din figura 3, a fost inițial adoptată în mare măsură, această stratificare nu mai reflectă "adevărul" în Internet odată cu apartiția nodurilor intermediare. Echipamentele intermediare interven în mod obișnuit la nivelul transport; uneori chiar duc la încheierea completă a conexiuni, lăsând astfel nivelul de aplicație ca primul nivel real end-to-end, așa cum se arată în figura 3.

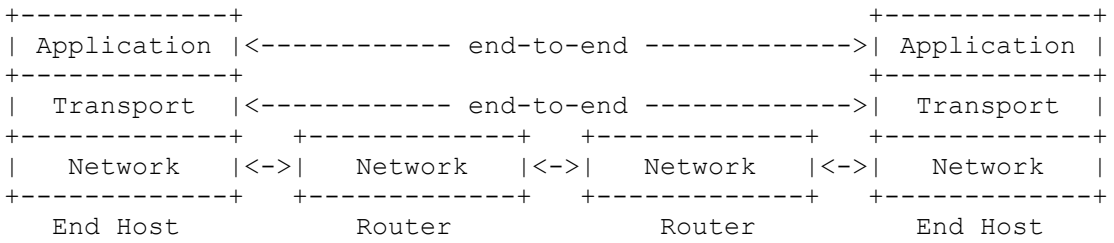

Figura 3: Arhitectura tradițională a Internet

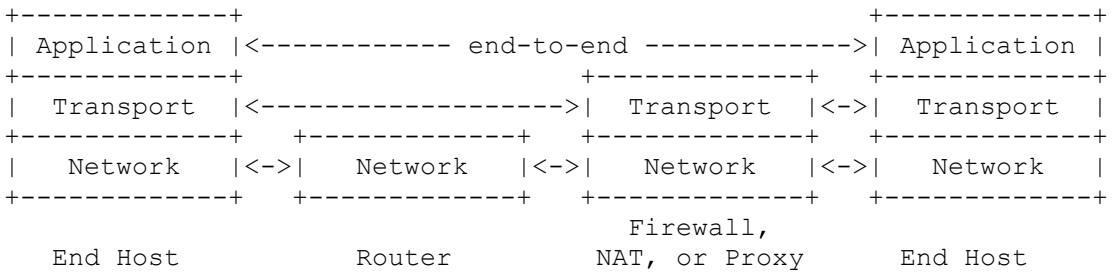

Figure 4: Internetul în realitate

Modificările pentru a sprijini Multipath TCP rămân la nivelul transport, deși sunt necesare unele cunoștințe de nivelul de rețea de bază. Multipath TCP ar trebui să lucreze cu IPv4 și IPv6 alternativ, într-o conexiune care poate opera pe ambele rețele IPv4 și IPv6.

3. Arhitectura MPTCP

Toate operațiunile MPTCP sunt semnalate cu ajutorul câmpurilor de antet TCP opționale. Un singur număr de opțiune TCP ("Kind") a fost atribuit de IANA pentru MPTCP, și apoi mesajele individuale vor fi determinate printr-un "subtip", ale cărui valori sunt de asemenea stocate într-un registru IANA.

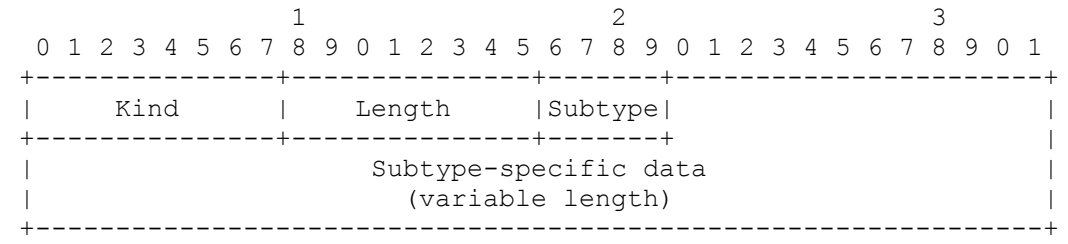

Figure 5: Formatul opțiunii MPTCP

Aceste opțiuni MPTCP asociate cu inițierea subflow-urilor sunt folosite în pachetele cu setul de fanioane SYN setat. In plus, exista o opțiune MPTCP pentru semnalizarea metadatelor, pentru a se asiguracu segmentatele de date pot fi recombinate, pentru livrarea către aplicație.

Opțiunile rămase, cu toate acestea, sunt semnale care nu trebuie să fie pe un pachet specific, cum ar fi cele pentru semnalizarea de adrese adiționale. În timp ce o implementare poate dori să trimită opțiuni MPTCP cât mai curând posibil, nu este posibil să se combine toate opțiunile dorite (atât cele pentru MPTCP și pentru TCP, cum ar fi SACK) intr-un singur pachet. Prin urmare, o aplicație poate alege să trimiteță ACK-uri duplicat cu informații suplimentare de semnalizare. Acest lucru schimba semantica unui ACK duplicat; acestea sunt de obicei trimise doar ca un semnal al unui segment pierdut în TCP. Prin urmare, o punere în aplicare a MPTCP care a primit un ACK duplicat care conține o opțiune MPTCP nu trebuie trata ca un semnal de congestie. In plus, o implementare MPTCP nu ar trebui să trimită mai mult de două ACK-uri duplicate odată cu scopul de a trimite opțiuni MPTCP, pentru a se asigura ca echipamentele intermediare nu interpretează greșit acest lucru ca pe un semn de congestie.

3.1 Inițializarea conexiunii

Inițierea conexiunii începe cu un SYN, SYN / ACK, ACK schimbate pe o singură cale. Fiecare pachet conține opțiunea MPTCP multipath (MP CAPABLE) (Figura 6). Această opțiune declară că expeditorul sau este capabil să execute Multipath TCP și dorește să facă acest lucru pe această conexiune. Schimbul MP_CAPABLE în această specificație (v1) este diferită de cea specificată în v0 [5]. În cazul în care o gazdă acceptă mai multe versiuni ale MPTCP, expeditorul opțiunii MP_CAPABLE trebuie să semnalizeze cel mai mare numărul de versiune suportă. Inițiatorul conexiunii pasiv, la primirea acestei opțiuni, va semnala numărul de versiune pe care dorește să îl utilizeze, care trebuie să fie egal sau mai mic decât numărul de versiune indicat în opțiunea MP_CAPABLE inițială. Având în vedere că schimbul SYN este diferit între V1 și V0 schimbul nu poate fi coborât imediat, și, prin urmare în cazul în care capătul a solicitat o versiune mai mică, atunci inițiatorul ar trebui să răspundă cu un ACK, fără nici o opțiune MP_CAPABLE, pentru a utiliza TCP regulat. În cazul în care inițiatorul susține versiunea cerută, pe conexiunile viitoare gazdă-țintă, inițiatorul poate memora preferința de versiune. Alternativ, inițiatorul poate închide conexiunea cu un RST TCP și imediat o poate restabili cu versiunea solicitată de MPTCP. Opțiunea MP_CAPABLE este de lungime variabila, cu diferite câmpuri incluse în funcție de ce tip de pachete este utilizat. Opțiunea CAPABIL MP completă este prezentată în figura 6.

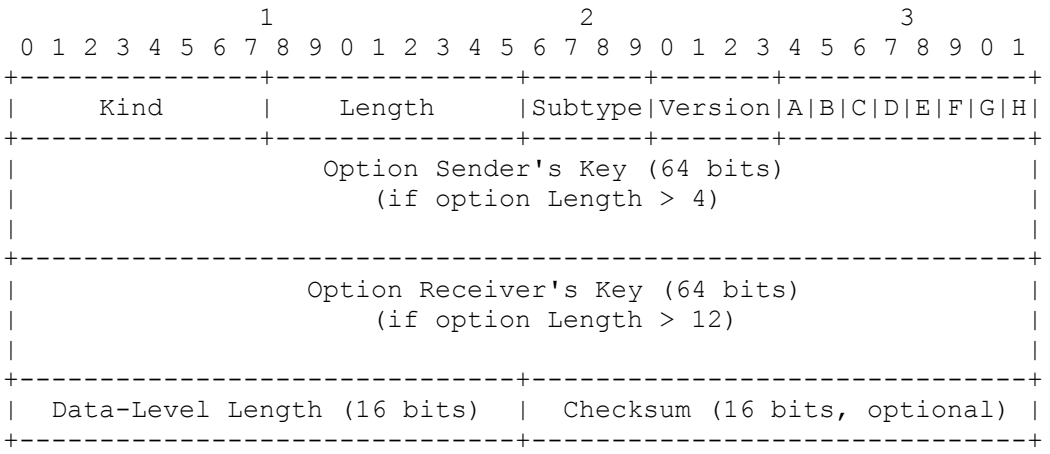

Figure 6: Opțiunea Multipath Capable (MP_CAPABLE)

Opțiunea MP_CAPABLE este inclusă în SYN, SYN / ACK, și pachetele ACK care încep primul subflow de conexiune MPTCP, precum și în primul pachet care transporta date, în cazul în care inițiatorul dorește să trimită mai întâi. Datele incluse de fiecare opțiune sunt după cum urmează, unde A = inițiator și B = ascultător.

- SYN (A-> B): numai primii patru octeti (Lungime = 4).
- SYN / ACK (B-> A): cheia lui B pentru această conexiune (Lungime = 12).
- ACK (fără date) (A-> B): cheia lui A, urmată de cheia lui B (Lungime = 20).
- ACK (cu prima de date) (A-> B): cheia lui A, urmată de cheia lui B, urmat de lungime, și de verificare opțională (Lungime = 22 sau de 24).

3.2 Inițializarea unui subflow nou

După o conexiune MPTCP a început cu schimbul MP_CAPABLE, în continuare pot fi adăugate subflowuri la conexiune. Gazdele au cunoștințe despre propriile adrese și poat deveni conștiente de adrese celeilalte gazdei prin schimburi de semnalizare. Folosind aceaste adrese, o gazdă poate inița un nou subflow peste o pereche de adrese neutilizată în prezent. Este permis pentru o gazdă într-o conexiune sa inițieze crearea unui nou subflow. Un nou subflow este pornit ca un schimb normal de TCP SYN / ACK. Opțiunea MPTCP MP_JOIN este utilizată pentru a identifica conexiunea ca să fie alăturată noului subflow. Acesta utilizează detele care au fost schimbate schimbul inițial MP_CAPABLE. În primul MP_JOIN din pachetul SYN, ilustrat în figura 6, inițiatorul trimite un token, număr aleator, și un ID al adresei.

La primirea unui SYN cu o opțiunea MP_JOIN care conține un token valid pentru o conexiune MPTCP existentă, beneficiarul ar trebui să răspundă cu un SYN / ACK conținând, de asemenea, o opțiune MP_JOIN care conține un număr aleator și un (cei mai semnificativi 64 de biți) mesaj HMAC. Această versiune a opțiunii este prezentată în figura 7. Dacă jetonul este

necunoscut, sau gazda dorește să refuze stabilirea subflow-ului (de exemplu, din cauza unei limite privind numărul de subflow-uri va permise), receptorul va trimite înapoi un semnal de reset (RST), analog cu un port necunoscut in TCP, care conține o opțiune MP_TCPRST, cu un cod corespunzător. Deși calculul unui HMAC necesită operații criptografice, se crede că jetonul pe 32 de biți în MP_JOIN SYN oferă o protecție suficientă împotriva atacurilor exahustive.

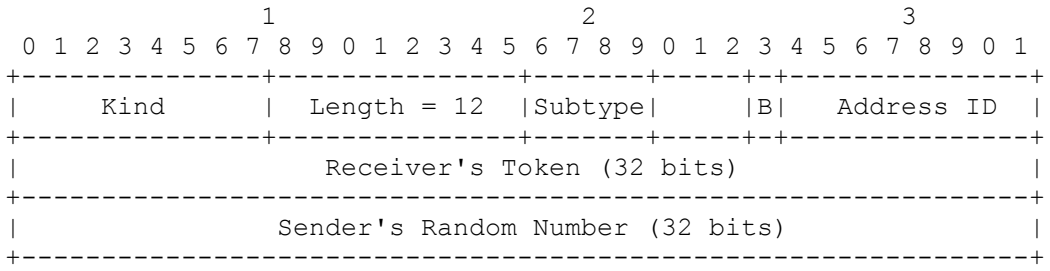

Figura 6: opțiunea Join Connection (MP_JOIN)(pentru SYN inițial)

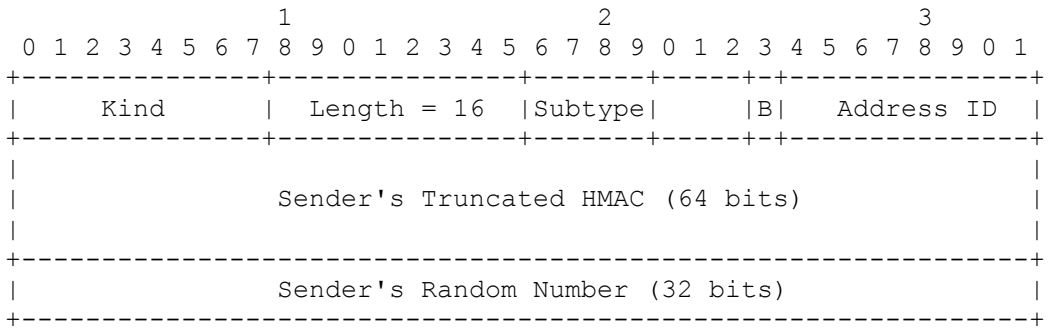

Figure 7: opțiunea Join Connection (MP_JOIN) Option

4. Controlul congestiei în MPTCP

Corectitudinea bottleneck-urilor este doar o cerință pe care controlul congestiei multipath ar trebui să o îndeplinească. Următoarele trei țeluri capturează proprietățile dorite ale unui algoritm de control al congestiei:

- Obiectivul 1 (Îmbunătățirea Throughput) Un flux pe mai multe căi trebuie să funcționeze cel puțin la fel de bine ca un singur flux care folosește cea mai bună dintre căi.
- Obiectivul 2 (a nu dăuna) Un flux pe mai multe căi nu ar trebui să folosească o capacitatea mai mare de la oricare dintre resursele partajate de diferite căi decât dacă ar fi fost un singur flux, folosind doar una dintre aceste căi.

- Obiectivul 3 (Balanța congestie) Un flux pe mai multe căi ar trebui să mute cat mai mult trafic posibil de pe căile sale cele mai aglomerate.

Controlerul de congestie urmărește să stabilească fluxuri multiple de lățime de bandă totală să fie aceeași cu cea a unui flux TCP care ar ajunge pe cea mai bună cale a multipath. Pentru a estima lățimea de bandă a unui flux TCP, fluxul de multipath estimează rate ale pierderii și timpi de dus-întors (RTT) și calculează rata țintă. Apoi, se reglează agresivitatea generală (parametrul alfa) pentru a atinge rata dorită.

4.1 Controlul congestiei bazat pe întârziere – algoritmul Weighted Vegas

wVegas este un algoritm de control al congestiei bazat pe întârziere și utilizează, de asemenea, comportamentul TCP nemodificat în cazul pierderii. WVegas este o alternativă pentru LIA, actualul algoritm de control a congestiei în MPTCP. Algoritmul se aplică în principal în faza de evitare a congestiei și, în faza de început lent, adaugă, de asemenea, o șansă de a intra în faza de evitare a congestiei mai devreme, ceea ce este similar cu punerea în aplicare a TCP-Vegas. Scăderea fazei de evitare a congestiei, retransmite rapid și algoritmi de recuperare rapidă sunt la fel ca TCP.

Următoarele operații de wVegas trebuie să fie efectuate la capătul fiecărei runde de transmisie.

Pentru un subflow r, diferența dintre rata de transmisie așteptată și rata de transmisie reală se calculează după cum urmează:

$$diff_r = \left(\frac{cwnd_r}{base_{rtt_r}} - \frac{cwmd_r}{rtt_r} \right) * base_{rtt_r}$$

În cazul în care subflow se află în faza de început lent, iar $diff_r$ este mai mare decât gamma, acesta trebuie să intre în faza de evitare a congestiei. Această operațiune este moștenită de la TCP-Vegas, poate fi realizat prin stabilirea *ssthresh_r* la ($cwmd_r$ - 1). Pe de altă parte, în cazul în care $diff_r$ nu este mai mare decât gamma în faza de început lent, wVegas trebuie să acționeze în același mod ca și TCP.

În faza de evitare a congestiei, în cazul în care $diff_r$ nu este mai mic decât $\propto_r$, $rate_r$trebuie să fie actualizat. Parametrii $weight_r$ și $\propto_r$, trebuie să fie optimizați înainte de a ajusta fereastra și îmbunătățirea preciziei $base_{rtt_r}$. Parametrii $rate_r$, $weight_r$ și $\propto_r$ trebuie să fie calculați după cum urmează:

$$rate_r = \frac{cwnd_r}{rtt_r}$$

$$weight_r = \frac{rate_r}{\sum rate_r}$$

$$\propto_r = weight_r * total_{\propto}$$

$\sum rate_r$ este rata totală a tuturor subflowurilor care aparțin la un singur flux MPTCP. După potrivirea parametrului $\propto_r$, daca este necesară ajustarea, $cwnd_r$ trebuie să fie modificat după cum urmează:

- dacă $diff_r$ este mai mare decât $\propto_r$ atunci $cwnd_r = cwnd_r - 1$
- dacă $diff_r$ este mai mic decât $\propto_r$ atunci $cwnd_r = cwnd_r + 1$

Ultima carcina cu care wVegas are de a face este de a încerca să se golească cozile legăturilor. Această operațiune este de a îmbunătăți precizia $base_{rtt_r}$. Ideea este de a micșora fereastra de congestie odată ce s-a detectat întârzierea de așteptare este mai mare decât un anumit prag, astfel încât legătura de strangulare poate goli pachetele salvate. Și, astfel, toate fluxurile implicate au o șansă de a obține întârzierea de propagare mai precisă.

Mai întâi, wVegas trebuie să calculeze întârzierea curentă de așteptare, după cum urmează:

$$queue_{delay_r} = rtt_r - base_rtt_r$$

Dacă întârzierea de așteptare curentă este mai mică decât $queue_{delay_r}$ salvat, $queue_{delay_r}$ trebuie să fie înlocuit cu cel actual. Și dacă întârzierea de așteptare curent este de două ori mai mare decât $queue_{delay_r}$, următoarele operații trebuie efectuate:

$$cwnd_r = cwnd_r * 0{,}5 * \frac{base_rtt_r}{rtt}$$

Principiul Egalitatății congestiei arată că o deplasare corectă și eficientă a traficului presupune ca orice efort de pentru a egaliza gradul de aglomerare al fluxurilor trebuie făcut pe toate căile sale disponibile. Acest principiu a fost demonstrat în [VEGAS]. Instantiind algoritmul iterativ de aproximare, ponderată în Vegas (wVegas), un algoritm bazat amanari pentru controlul congestiei multiple, a fost dezvoltat, care folosește pachete de așteptare ca semnale de congestie, obținându-se astfel load balancing cu granulație fină. Simulările arată că, în comparație cu algoritmi pe bază de pierderi, wVegas este mai sensibil la schimbările de congestie a rețelei și realizează mai deplasara traficului în timp util și de convergență mai rapidă. În plus, așa cum el

ocupă mai puține tampoane de legătură, wVegas cauzează rar pierderi de pachete și arată mai bine corectitudinea intra-protocol.

4.2 Controlul congestiei cuplat pentru MPTCP

De multe ori obiectivele sunt conectate prin mai multe căi, dar comunicațiile sunt, de obicei limitate la o singură cale pentru fiecare conexiune. Utilizarea resurselor în cadrul rețelei ar fi mai eficientă dacă ar fi posibil ca aceste căi multiple să fie utilizate în același timp. Multipath TCP este o propunere de a obține un transport multipath în TCP. Noi algoritmi de control a congestiei sunt necesari pentru protocoale de transport pe căi multiple, cum ar fi multipath TCP, cum algoritmii pentru o singură cale au o serie de probleme, în contextul multipath. Una dintre problemele importante este faptul că execuția algoritmilor existenți, cum ar fi TCP standard, independent pe fiecare cale ar da căi multiple de curgere mai mult decât partea sa justă de la un link strangulare traversată de mai mult de unul dintre subflows sale. Mai mult, este de dorit ca o sursă cu mai multe căi disponibile să transfera mai mult trafic folosind cea mai puțin aglomerată din căi, obținând o proprietate numită "pooling resource" în cazul în care un pachet de link-uri se comportă efectiv ca o legătură comună cu o capacitate mai mare. Acest lucru ar spori eficiența generală a rețelei și, de asemenea, capacitatea sa de a rezista la eșec.

Algoritmul prezent se aplică numai la faza de creștere a stării de evitare a congestiei precizând modul în care fereastra este mărită, la primirea unui ACK. Startul lent, retransmitea rapidă, și algoritmi de recuperare rapidă, precum și scăderea multiplicativă sunt metode de evitare a congestiei aceleași ca și în standard TCP.

Să presupunem că $cwnd_i$ este fereastra de congestie pe subflow i iar $cwnd_{total}$ va fi suma dintre ferestrele de congestionare a tuturor subflow-urilor conexiunii. Notăm p_i, rtt_i și mss_i rata de pierdere, timp tur-retur și dimensiunea maximă a segmentului pe subflow i.

Algoritmul propus trebuie ca pentru fiecare ACK recepționat pe subflowul i să mărească $cwnd_i$ cu:

$$\min\left(\frac{\propto * octeți_{ack} * mss_i}{cwnd_{total}}, \frac{octeți_{ack} * mss_i}{cwnd_i}\right)$$

Formula de majorare este de minimul între creșterea calculată pentru subflow multipath (primul argument la min), și creșterea TCP care s-ar obține în același scenariu (al doilea argument). În acest fel, ne asigurăm că orice subflow pe mai multe căi nu poate fi mai agresiv decât un flux TCP în aceleași circumstanțe.

Parametrul $\propto$ este un parametru al algoritmului care descrie agresivitatea fluxului multipath. Pentru a îndeplini obiectivul 1 (îmbunătăți randamentul), valoarea alfa este aleasă astfel încât debitul total al fluxului de multipath este egal cu debitul unui flux TCP care s-ar obține pe calea cea mai bună.

Pentru a calcula $cwnd_{total}$, este o greșeală ușor de făcut să adunăm $cwnd_i$ în toate subflowurile: atunci când un flux este în retransmite rapidă, $cwnd$ este de obicei mărit și nu mai reprezintă dimensiunea ferestrei de congestie reală. Comportamentul corect este de a utiliza valoarea *ssthresh* (prag început lent) pentru fluxuri în retransmite rapidă la calculul $cwnd_{total}$. Pentru a răspunde la conexiunile care sunt limitate de aplicație, calculul ar trebui să ia în considerare minimul dintre *flight_size_i* și $cwnd_i$, și *flight_size_i* și *ssthresh_i*.

Formula pentru calculul parametrului $\propto$ este:

$$\propto = cwnd_{total} * \frac{MAX(\frac{cwnd_i}{rtt_i^2})}{\left(SUM\left(\frac{cwnd_i}{rtt_i^2}\right)\right)^2}$$

4.3 Adaptarea link-urilor balansată pentru controlul congestiei

"Balanced linked adaptation" este un algoritm de control al congestiei pe baza de fereastra pentru MPTCP. Principalul obiectiv de proiectare a Balia este acela de a compromite în mod sistematic proprietăți diferite, cum ar fi ușurința TCP și capacitatea de reacție prin dezvoltarea de înțelegere structurală a algoritmilor MPTCP într-un nou cadru de proiectare. De exemplu, este larg suspectat că există un compromis între utilizarea și capacitatea de reacție în care se dovedește că în acest cadru acest compromis este într-adevăr inevitabil. Prin parametrizarea diferitelor proprietăți structurale, Balia generalizează algoritmi existenți și în mod explicit echilibrează compromisul. De asemenea, vom demonstra matematic că Balia are un punct de echilibru unic, și că este asimptotic stabil. Prin urmare, Balia poate oferi performanțe echilibrate din punct de vedere ușurința și capacitatea de reacție.

Fiecare sursă s are un set de trasee r. Ca un caz special, setul poate fi un singleton caz in care Balia sereduce la TCP Reno. Fiecare cale r menține un set alcătuit din w_r (fereastră congestie) și măsurile cu durata calatoriei RTT r. Adaptarea ferestrei de Balia este următoarea:

- pentru fiecare ACK pe calea r, se mărește w_r cu:

$$\frac{x_r}{rtt_r * \left(SUM(x_k)\right)^2} * \frac{1 + \propto_r}{2} * \frac{4 + \propto_r}{5}$$

- pentru fiecare pachet pierdut pe calea r, fereastra w_r se mi;orează cu:

$$\frac{w_r}{2} * \min\{\propto_r, 1.5\}$$

unde $x_r = \frac{w_r}{rtt_r}$ și $\propto_r = \frac{\max\{x_k\}}{x_r}$

Bibliografie

[1] A. Walid, Bell Labs, Samsung Electronics, Caltech; Balanced Linked Adaptation Congestion Control Algorithm for MPTCP, Internet-Draft

[2] C. Raiciu, M. Handly, Coupled Congestion Control for Multipath Transport Protocols, Internet Engineering Task Force (IETF), ISSN: 2070-1721

[3] C. Raiciu, A. Ford, M. Handley, TCP Extensions for Multipath Operation with Multiple Addresses, Internet Engineering Task Force Internet-Draft

[4] C. Raiciu, A. Ford, M. Handley, S. Barre, Architectural Guidelines for Multipath TCP Development, Internet Engineering Task Force (IETF), ISSN: 2070-1721