Một Thiết Kế Multipath TCP Dựa Trên Độ Trễ

Trần Thị Bảo Yến
Công ty NETSOFT
Viễn Thông TP. Hồ Chí Minh
TP. Hồ Chí Minh, Việt Nam
Email:ttbyen.hcm@vnpt.vn

Lê Tuấn Anh, Trần Công Hùng, Huỳnh Trọng Thưa
Khoa Công nghệ Thông tin
Học viện Công nghệ Bưu chính Viễn Thông
TP. Hồ Chí Minh, Việt Nam
Email: {letuanh,conghung,htthua}@ptithcm.edu.vn

Tómtắt—Trong thời đại phát triển của công nghệ hiện nay, cùng với sự xuất hiện của nhiều ứng dụng đòi hỏi tính năng truyền dữ liệu trong mạng tốc độ cao và/ hay độ trễ lớn – Bandwidth Delay Product (BDP) lớn, thì nhu cầu sử dụng các thiết bị di động thông minh được trang bị nhiều card mạng cũng tăng cao. Dẫn đến sự ra đời của các giao thức multi-path TCP trong việc hỗ trợ truyền dẫn dữ liệu thông qua các thiết bị thông minh.Một số giao thức multi-path TCP gần đây như MPTCP, weighted-Vegas đã được đề xuất để ứng dụng cho các thiết bị đa card mạng trong việc truyền dữ liệu giữa hai điểm đầu cuối. Các thuật toán này phần nào cải thiện được hiệu suất, tính cạnh tranh và khả năng cân bằng trên đường truyền, tuy nhiên vẫn phải đương đầu với các vấn đề thường thấy khi hoạt động trong mạng tốc độ có BDP lớn, do cả hai thuật toán trên đều được thiết kế cho mạng tốc độ thấp và/ hay độ trễ nhỏ. Đó là lý do chúng ta cần một giao thức multi-path TCP khắc phục được những khuyết điểm trên.Chúng tôi đề xuất MPFAST – một giải thuật được mở rộng từ FAST TCP, sử dụng kỹ thuật phát hiện tắc nghẽn delay-based và đặc biệt hiệu quả trong mạng có BDP lớn.

Từ khóa—queueing delay; FAST TCP; BDP lớn; Multipath TCP.

I. GIỚI THIỆU

Thuật toán điều khiển tắc nghẽn trong giao thức TCP Reno được phát triển vào năm 1988 và đã được cải tiến nhiều lần. Thuật toán này, về cơ bản, hoạt động khá tốt và góp phần ngăn chặn tắc nghẽn bằng độ lớn, tốc độ, tải trọng và kết nối. Tuy nhiên, với mạng có BDP lớn, TCP Reno thường xuyên trở thành một nút thắt cổ chai.

Thiết kế của HSTCP và STCP phần nào khắc phục được các khuyết điểm của TCP Reno và đặc biệt dành riêng cho mạng tốc độ cao và/ hay độ trễ lớn. Tuy nhiên, cả hai thuật toán đều ảnh hưởng đến tính công bằng trên đường truyền, do cố gắng chiếm băng thông của các giao thức TCP khác. Song song đó, cả hai đều dựa vào giải pháp loss-based, không những không hiệu quả về mặt sử dụng đường truyền mà còn có khả năng khiến thông tin về tắc nghẽn có thể không chính xác.

Do đó, đề tài này sẽ dựa trên giải pháp delay-based được sử dụng trong giao thức FAST TCP. Điều khiển tắc nghẽn sử dụng delay-based khắc phục được nhược điểm của loss-based, đặc biệt trong mạng có BDP lớn. Sử dụng queuing delay làm thước đo tắc nghẽn, đem lại hai ưu điểm quan trọng.

Thứ nhất, queuing delay cung cấp thông tin ước lượng chính xác hơn loss-based do khả năng mất gói trên mạng có độ trễ lớn ít xảy ra và thông tin tắc nghẽn dựa vào mất gói trong phương pháp loss-based thiếu chính xác hơn dựa vào độ trễ trong phương pháp delay-based. Việc đo lường dựa vào một gói tin bị mất chỉ cung cấp một bit thông tin, trong khi đó, việc đo lường dựa vào độ trễ cung cấp đa bit thông tin. Thông tin này làm cho việc cài đặt hệ thống về trạng thái ổn định mà vẫn đảm bảo tính công bằng và sử dụng hiệu quả trở nên dễ dàng hơn.

Thứ hai, tính linh hoạt của queuing delay có thể thích ứng với dung lượng hệ thống. Điều này duy trì tính ổn định khi hệ thống mở rộng quy mô.

Song song đó, sự phát triển của các giao thức điều khiển tắc nghẽn đa đường (Multiple paths TCP) đang được chú ý và đánh giá cao. Minh chứng chính là sự lần lượt ra đời của các giao thức như MPTCP [2], weighted-Vegas (wVegas) [18], MPCubic [1]. Tuy nhiên các giao thức trên, ít nhiều còn tồn đọng nhiều khuyết điểm khi thực thi trong mạng có BDP lớn, chi tiết sẽ được trình bày ở chương sau. Do đó, trong đề tài này, chúng tôi đề xuất Multi-path FAST TCP (MPFAST), một thuật toán mở rộng của FAST TCP đã đề cập phía trên cho truyền dữ liệu đa đường. Thuật toán này hứa hẹn hiệu quả trong mạng với BDP lớn. Hơn nữa, do dựa trên thuật toán FAST TCP, MPFAST có khả năng ngăn chặn việc giảm hiệu suất trong mạng không dây nơi khả năng mất gói xảy ra ngẫu nhiên do lỗi sóng truyền cao hơn do lỗi tắc nghẽn mạng.

II. CÁC NGHIÊN CỨU LIÊN QUAN

Để truyền dữ liệu hiệu quả, không nghẽn mạng trong khi vẫn duy trì được tính công bằng, tính sẵn sàng và ổn định, nhiều giao thức đa đường đã được nghiên cứu và ứng dụng. Tuy nhiên, một số giao thức không đáp ứng được hiệu quả khi sử dụng trong mạng tốc độ cao và/ hay độ trễ lớn. Các giao thức này chủ yếu dựa trên hai kỹ thuật chính: loss-based và delay-based. Chi tiết về hai kỹ thuật này sẽ được trình bày chi tiết ở chương sau. Dưới đây là một số giao thức multipath tiêu biểu.

ISBN: 978-604-67-0349-5

Sau khi nghiên cứu hoạt động của MPTCP [2] với thuật toán Linked Increase Algorithm (LIA) cùng nhiều mô phỏng khác nhau, R. Khalili và N. Gast cùng các đồng nghiệp đã kết luận hai vấn đề vẫn còn tồn tại trong giao thức MPTCP.Thứ nhất, nâng cấp từ TCP chuẩn thành MPTCP có thể làm giảm băng thông của các giao thức TCP khác trên cùng đường truyền mà không mang lại lợi ích cho các TCP đã được nâng cấp. Thứ hai, MPTCP thường tranh giành băng thông với các giao thức khác. Tuy đã được cải thiện với thuật toán Opportunistic Linked Increase Algorithm (OLIA) [7] nhưng ít hiệu quả khi sử dụng trong mạng có BDP lớn.

Một nghiên cứu gần đây, wVegas [18] đã giải quyết vấn đề trong điều khiển tắc nghẽn trong mạng đa đường áp dụng phương pháp delay-based làm dấu hiệu phát hiện nghẽn mạng. Tuy nhiên, wVegas chỉ thích hợp cho các ứng dụng trong mạng tốc độ thấp và/ hay độ trễ nhỏ. Theo [9], wVegas hoàn toàn không hiệu quả trong mạng có BDP lớn.

MPCubic [1], giao thức dựa trên phương pháp loss-based, được phát triển từ Cubic TCP, dành riêng cho mạng BDP lớn. MPCubic không chỉ thể hiện tính công bằng mà còn sử dụng cơ chế fast recovery hiệu quả. Tuy nhiên, do dựa trên loss-based, MPCubic không được đánh giá cao trong trường hợp phát hiện tắc nghẽn sớm.

III. THIẾT KẾ MULTIPATH FAST TCP

A. *Thuật toán FAST TCP*

Giao thức FAST TCP ra đời đã mang lại những thay đổi tích cực cho các ứng dụng thực thi trên mạng có BDP lớn. Như đã nói ở các chương trước, phương pháp delay-based hiệu quả hơn loss-based trong việc dự đoán tắc nghẽn. Sau đây, đề tài sẽ tóm tắt lại thuật toán của giao thức FAST TCP:

Xem xét một nguồn i của thuật toán FAST TCP điều chỉnh kích thước cửa sổ w_i(t) bằng công thức sau:

$$w_i(t+1) = \gamma\left(\frac{d_i w_i(t)}{d_i + q_i(t)} + \alpha_i\big(w_i(t), q_i(t)\big)\right) + (1-\gamma)w_i(t) \quad (1)$$

với $\gamma \in (0,1]$, d_i và q_i tương ứng là thời gian truyền và độ trễ. Ta có, $T_i(t) := d_i + q_i(t)$ là RTT được tính bởi nguồn i, và $x_i(t) := w_i(t)/T_i(t)$ là thông lượng của nguồn i tại thời điểm t (đơn vị tính pkts/sec). Hằng số $\alpha_i(w_i, q_i)$ – thể hiện số packet của nguồn i tồn tại tại routers trên đường truyền – được tính như sau:

$$\alpha_i(w_i, q_i) = \begin{cases} a_i w_i & , nếu\ q_i = 0 \\ \alpha_i & , TH\ khác \end{cases} \quad (2)$$

Trong trường hợp $\alpha_i(w_i, q_i) = \alpha_i$ (hằng số), công thức (1) được viết lại như sau:

$$w_i(t+1) = w_i(t) + \gamma(\alpha_i - x_i(t)q_i(t)), \quad (3)$$

Với thuật toán điều khiển tắc nghẽn đơn đường FAST TCP, giá trị $x_i(t)$ tại thời điểm cân bằng được tính từ (3) là:

$$\hat{x}_i = \frac{\alpha_i}{\hat{q}_i}. \quad (4)$$

Tương tự, $\hat{q}_i$ là giá trị của $q_i(t)$ tại thời điểm cân bằng.

B. *Thuật toán MPFAST*

Bây giờ, đề tài sẽ trình bày giải thuật tắc nghẽn đa đường được mở rộng từ giao thức FAST TCP, gọi là MPFAST. Thiết kế của MPFAST dựa trên ý tưởng từ FAST TCP được thực hiện như sau:

Gọi r là subflow của MPFAST dùng để truyền tải dữ liệu trên đường r. Thuật toán điều khiển tắc nghẽn MPFAST của nguồn s tương ứng với kích thước cửa sổ điều khiển tắc nghẽn $w_{s,r}$ trên đường truyền r được tính như sau:

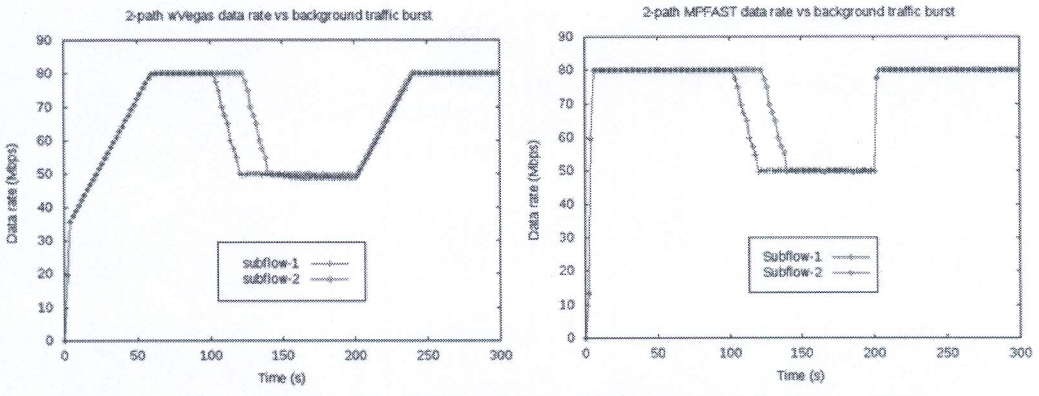
$$w_{s,r}(t+1) = \gamma\left(\frac{d_{s,r} w_{s,r}(t)}{d_{s,r} + q_{s,r}(t)} + \theta_{s,r}(t)\alpha_s\right) + (1-\gamma)w_{s,r}(t), \quad (5)$$

với $\theta_{s,r}(t) \in [0,1]$ (lưu ý rằng $\sum_r \theta_{s,r} = 1$) là ký hiệu trọng số của nguồn s trên đường truyền r. Chúng ta định nghĩa N là số subflow của kết nối MPFAST. Tương tự, chúng ta có giá trị của subflow r của nguồn s được tính từ (4):

$$\hat{x}_{s,r} = \frac{\hat{\theta}_{s,r}\alpha_s}{\hat{q}_{s,r}}. \quad (6)$$

Giá trị tại thời điểm cân bằng của tổng thông lượng gửi từ nguồn s, $y_s(t)$ được định nghĩa như sau:

ISBN: 978-604-67-0349-5



Hình 2. Tính hiệu quả của giao thức multipath trong mạng có BDP lớn. Khi có các luồng background traffic: (a) wVegas phản hồi chậm; (b) MPFAST phản hồi nhanh hơn.

$$\hat{y}_s := \sum_{r=1}^{N} \hat{x}_{s,r} = \alpha_s \sum_{r=1}^{N} \frac{\hat{\theta}_{s,r}}{\hat{q}_{s,r}}. \quad (7)$$

Để xác định $\hat{\theta}_{s,r}$, chúng ta sẽ xem như tất cả đường truyền có cùng độ trễ, q_s. Chúng ta có (8) từ công thức (7) như sau:

$$\hat{y}_s = \alpha_s \sum_{r=1}^{N} \frac{\hat{\theta}_{s,r}}{\hat{q}_{s,r}} = \frac{\alpha_s}{\hat{q}_s} \sum_{r=1}^{N} \hat{\theta}_{s,r} = \frac{\alpha_s}{\hat{q}_s} \quad với \sum_{r=1}^{N} \hat{\theta}_{s,r} = 1, \quad (8)$$

Với $\hat{\theta}_{s,r}, \hat{q}_{s,r}$ và $\hat{y}_s$ lần lượt là giá trị tại thời điểm cân bằng của $\theta_{s,r}(t), q_{s,r}(t)$ và $y_s(t)$. Từ đó, chúng ta có (6) trở thành:

$$\hat{x}_{s,r} = \frac{\hat{\theta}_{s,r}\alpha_s}{\hat{q}_s}, \quad (9)$$

Lấy (9) chia (8), chúng ta có:

$$\hat{\theta}_{s,r} = \frac{\hat{x}_{s,r}}{\hat{y}_s}. \quad (10)$$

Phương trình (6) và (10) cho thấy mỗi subflow thuộc đường r có số lượng gói tin ứ đọng tại các router tỉ lệ thuận với tích $\hat{\theta}_{s,r}\alpha_s$. Ở tiêu chí 2, giả sử 2 luồng MPFAST cạnh tranh với 1 luồng FAST TCP tại 1 link thắt cổ chai, các luồng này có cùng độ trễ (queueing delay). Bởi vì α_i (thuộc luồng FAST TCP) và α_s (luồng MPFAST) giống hệt nhau, tốc độ truyền của 2 luồng MPFAST được chia bởi tham số $\hat{\theta}_{s,r}$. Trong trường hợp này, $\hat{\theta}_{s,r}$ xấp xỉ 1/2, do đó, tổng băng thông của 2 luồng MPFAST bằng với băng thông luồng FAST TCP.

Các subflow trên đường r điều chỉnh kích thước cửa sổ tắc nghẽn bởi đoạn mã giả dưới đây:

Chúng ta đều biết RTT có thể tăng hoặc giảm làm ảnh hưởng đến sự ổn định của thuật toán. Vì vậy, chúng ta cần 1 giá trị RTT trung bình (`average_RTT`) có thể thích nghi với sự điều chỉnh RTT mà không phản ứng quá nhạy cảm với các thay đổi mạng. Để tính được giá trị này, chúng ta sử dụng công thức tính `smoothed RTT` như sau:

`New_RTT = (`α `* Old_RTT) + ((1 -` α`)* Newest_RTT)`

Với $\alpha \in [0, 1]$, giá trị α càng gần 1 sẽ cung cấp độ mịn cho tốc độ và tránh được những thay đổi đột ngột do sự tăng giảm thất thường của RTT. Ngược lại, giá trị α càng gần 0 giúp RTT phản ứng nhanh hơn với sự thay đổi của mạng. Tham số `baseRTT` chính là giá trị RTT không bao giờ thời gian gói tin chờ ở hàng đợi.

IV. MÔ PHỎNG & ĐÁNH GIÁ

A. Kịch bản mô phỏng

Trong chương này, đề tài sẽ tập trung thực hiện mô phỏng và đánh giá kết quả mô phỏng, từ đó đưa ra kết luận về giải thuật MPFAST dựa trên ba tiêu chí của thiết kế điều khiển tắc nghẽn đa đường, bao gồm:

1) Tăng hiệu quả sử dụng tài nguyên mạng: khả năng khai thác băng thông đường truyền khi hoạt động ở mạng có BDP lớn. Mô phỏng ở hình 1.a, đánh giá hiệu quả của giao thức MPFAST trên đường truyền có sự xuất hiện đột ngột của các luồng giao thức khác.

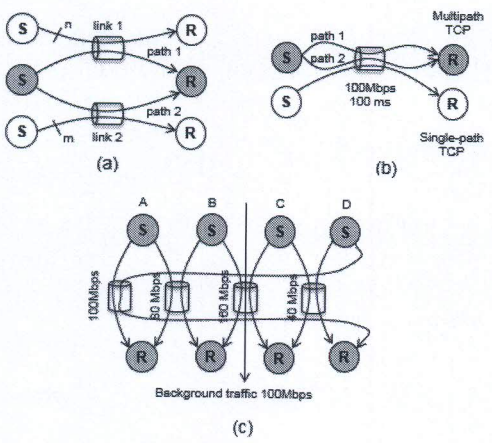
2) Chia sẻ công bằng: tính công bằng khi cạnh tranh với các luồng TCP khác trên đường truyền với RTT khác nhau. Mô phỏng ở hình 1.b, giúp khai thác tính công bằng của giao thức MPFAST trên đường truyền khi cạnh tranh với giao thức FAST TCP.

3) Cân bằng tắc nghẽn: điều khiển luồng từ nơi đang tắc nghẽn đến nơi ít hoặc còn không tắc nghẽn, cân bằng băng thông giữa các subflow và không ảnh hưởng đến các giao thức khác trên cùng đường truyền. Mô phỏng ở hình 1.c để đánh giá khả năng cân bằng của giao thức MPFAST.

Mô phỏng được thực hiện với công cụ NS2, các mô hình đều có chung lựa chọn SACK (Selective

ISBN: 978-604-67-0349-5

Acknowledgment), kích thước gói tin 1448 bytes, kích thước bộ đệm router là BDP, và dữ liệu lấy mẫu mỗi 2s.



Hình 1. Kịch bản mô phỏng

B. Kết quả mô phỏng:

Sau khi thực thi các kịch bản mô phỏng trên với công cụ NS2, đề tài tiến hành đánh giá tính hiệu quả của giao thức MPFAST dựa trên các tiêu chí đã đề cập trước đó: (1) tính hiệu quả (2) tính công bằng và (3) khả năng cân bằng tắc nghẽn.

1) Tính hiệu quả:

Để tăng tính khách quan, đề tài thực hiện cùng 1 mô phỏng với 2 thuật toán khác nhau: wVegas và MPFAST để so sánh tính hiệu quả khi thực thi trong môi trường mạng có cùng cấu hình.

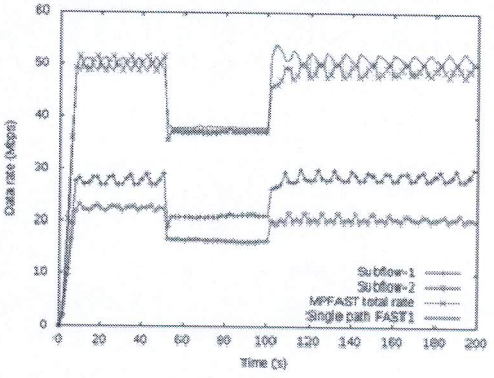
Đề tài sẽ thực hiện kịch bản mô phỏng như hình 1.a cho cả haigiao thức wVegas và MPFAST. Các luồng dữ liệu được truyền qua hai link trên có cùng cấu hình, với dung lượng là 80Mbps và thời gian truyền là 50ms. Kèm theo đó, 6 luồng background traffic được sử dụng trên link 1 ở giây thứ 100, trên link 2 sẽ được bắt đầu ở giây thứ 120 với 8 luồng, tất cả các luồng ở hai link đều kết thúc vào giây thứ 200. Các luồng background traffic lần lượt xuất hiện sau luồng trước nó mỗi 3s.

Hình 2 cho thấy hình ảnh so sánh rõ nét nhất về tính hiệu quả trong việc khai thác đường truyền. Ta thấy khả năng khai thác đường truyền của giao thức MPFAST ở hình 2.b khá tốt. Trong khi giao thức wVegas (hình 2.a) khá chật vật để đạt đến tốc độ mong đợi (trong trường hợp này là 80Mbps), thì MPFAST chỉ cần một thời gian rất ngắn để làm được điều này. Sự xuất hiện của background traffic trong khoảng thời gian từ giây 100 đến 120 gây ảnh hưởng đến cả 2 giao thức. Tuy nhiên, MPFAST được đánh giá tối ưu hơn wVegas khi điều khiển luồng dữ liệu khá cân bằng. MPFAST điều chỉnh tốc độ xuống thấp hơn (khoảng1/2 của tốc độ ban đầu) và có sự ổn định, cũng như cân bằng cần thiết, và sau khi background traffic kết thúc, nó nhanh chóng lấy lại tốc độ ban đầu. Trong khi đó, wVegas rớt tốc độ gần như mất kiểm soát tới mức rất thấp (dưới1/2 tốc độ ban đầu), và cũng mất thời gian dài hơn sau khi tắt background traffic để đạt đến con số 80Mbps.

2) Tính công bằng

Đề tài thực hiện mô phỏng ở hình 1.b, sử dụng đường truyền này có dung lượng 100Mbps cùng thời gian truyền 100ms. Mô phỏng này đánh giá tính công bằng của giao thức MPFAST với FAST TCP khi cùng tồn tại trên đường truyền.

Kết quả mô phỏng qua hình 3 cho thấy MPFAST công bằng chia sẻ băng thông mạng với giao thức FAST TCP. Băng thông FAST TCP gần như bằng tổng 2 subflow MPFAST, hay nói cách khác băng thông mỗi subflow chỉ bằng 1/2 băng thông FAST TCP. Điều đó có nghĩa MPFAST chia băng thông cho mỗi subflow sao cho tổng băng thông bằng với băng thông của giao thức đang cạnh tranh với nó nhằm đạt tính công bằng trên mạng. Ngay cả khi có sự xuất hiện của các luồng background traffic, tính công bằng này vẫn được đảm bảo. Kết quả trên cho thấy MPFAST thỏa được tiêu chí thứ 2 của giao thức điều khiển tắc nghẽn đa đường.



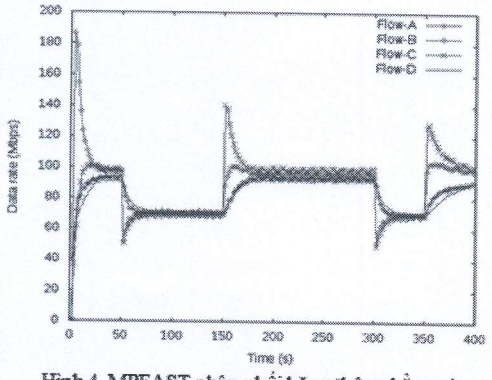
Hình 3. MPFAST thể hiện tính công bằng với FAST TCP trên cùng đường truyền.

3) Khả năng cân bằng tắc nghẽn

Đề tài thực hiện mô phỏng với kịch bản như hình 1.c. Kịch bản mô phỏng này sử dụng các luồng background traffic để phân tích khả năng cân bằng tải trong môi trường mạng không ổn định. Đề tài dùng 4 link với thời gian truyền như nhau – 50ms – tuy nhiên, dung lượng khác nhau, lần lượt là: 100Mbps, 80Mbps, 160Mbps và 40Mbps. 10 luồng background traffic với dung lượng 10Mbps mỗi luồng được đưa vào đường truyền ở giây 100 và đồng loạt kết thúc ở giây 200. Kết quả mô phỏng đạt được như ở hình 4. Trong khoảng thời gian có sự xuất hiện của background traffic, các luồng dữ liệu có sự suy giảm, khoảng 70Mbps, giữ tốc độ đó đến khi background traffic tắt và lấy lại tốc độ ban đầu. Hai khoảng còn lại trên đường truyền, trước và sau sự có mặt các luồng background traffic này, tốc độ đường

ISBN: 978-604-67-0349-5

truyền luôn ở trạng thái ổn định và như mức mong đợi, khoảng 95Mbps. Vài giây đầu trước khi đạt được tốc độ này, băng thông có hơi dao động, nhưng sau đó vẫn nhanh chóng ổn định. Mức ổn định 95Mbps được tính bằng các lấy tổng băng thông các luồng chia cho số luồng. Qua mô phỏng trên, MPFAST chứng minh nó cân bằng tải tốt trên môi trường mạng, có khả năng chuyển luồng dữ liệu từ link đang tắc nghẽn đến link ít tắc nghẽn hơn và duy trì trạng thái ổn định trên cả đường truyền.



Hình 4. MPFAST phân phối băng thông bằng nhau cho các luồng con ngay cả khi mạng có thay đổi.

V. KẾT LUẬN

Bắt nguồn từ ý tưởng thiết kế giao thức điều khiển tắc nghẽn đa đường cho mạng tốc độ cao và/ hay độ trễ lớn, qua những tìm hiểu về điều khiển tắc nghẽn trong TCP và Multipath TCP, cùng như các nghiên cứu tiên phong, chúng tôi đề xuất thuật toán MPFAST, dựa trên thuật toán của giao thức FAST TCP, dùng kỹ thuật delay-based và các cơ chế điều khiển tắc nghẽn phù hợp. Thông qua các mô hình mô phỏng và việc phân tích kết quả có được, MPFAST thể hiện là một giao thức điều khiển tắc nghẽn đa đường thỏa mãn đủ các tiêu chí thiết kế đã đưa ra, từ việc cân bằng tải hiệu quả đến tính chất công bằng và khai thác tốt tài nguyên hệ thống mạng có BDP lớn.

TÀI LIỆU THAM KHẢO

[1] Le Tuan Anh, Rim Haw, Choong Seon Hong and Sungwon Lee (2012), "*A Multipath Cubic TCP Congestion Control with Multipath Fast Recovery over High Bandwidth-Delay Product Networks*", IEICE Transactions on Communications, Vol.E95-B, No.7, pp.2232-2244.

[2] A. Ford, C. Raiciu, M. Handley, S. Barre, J. Iyengar (2011), "*Architecture guidelines for Multipath TCP Development*", IETF RFC 6182.

[3] A.Ford, C.Raiciu, M.Handley, U.College London, O.Bonaventure, U.Catholique de Louvain, (2013), "TCP Extensions for Multipath Operation with Multiple Addresses", Internet Engineering Task Force, RFC 6824.

[4] Cheng Jin, David X. Wei Steven, H. Low (2004), "*FAST TCP: Motivation, Architecture, Algorithms, Performance*", IEEE InfoCOM.

[5] Janey Hoe (August 1996). Improving the startup behavior of a congestion control scheme for tcp, *In ACM Sigcomm'96*, http://www.acm.org/sigcomm/sigcomm96/program.html.

[6] Jim Martin, Arne Nilsson, and Injong Rhee (June 2003). Delay-based congestion avoidance for TCP. IEEE/ *ACM Trans. On Networking, 11(3):356-369.*

[7] Khalili, R.; Gast, N.; Popovic, M.; Le Boudec, J.-Y. (2013), "*MPTCP Is Not Pareto-Optimal: Performance Issues and a Possible Solution*," Networking, IEEE/ACM Transactions on , vol.21, no.5, pp.1651,1665.

[8] M. Allman, V. Paxson, and W. Stevens (1999). *TCP congestion control*. RFC 2581, http://www.faqs.org/

[9] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow (October 1996). *TCP Selective Acknowledgment Options*. RFC 2018, ftp://ftp.isi.edu/in-notes/rfc2018.txt.

[10] NS-2 network simulator. [Online]. Available: http://www.isi.edu/nsnam/ns/

[11] Ren Wang (March 2004), "*TCP Startup Performance in Large Bandwidth Delay Networks*", Computer Science Department, University of California, Los Angeles.

[12] S. Floyd (2003), "*High-speed TCP for Large Congestion Windows*", RFC 3649.

[13] S. Floyd and T. Henderson (April 1999). *The New Reno modification to TCP's Fast Recovery algorithm*. RFC 2282, http://www.faqs.org/rfcs/rfc2282.html.

[14] Tom Kelly (2003), "*Scalable TCP: Improving Performance in Highspeed Wide Area Networks*", First International Workshop on Protocols for Fast Long Distance Networks, Geneva.

[15] V. Jacobson. Congestion avoidance and control (August 1988). *Proceedings of SIGCOMM'88, ACM.* An updated version is available via ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z.

[16] V. Jacobson, R. Braden, and D. Borman (May 1992). *TCP extension for high performance*. RFC 1323, ftp://ftp.isi.edu/in-notes/rfc1323.txt.

[17] W. Stevens (January 1997). *TCP Slow Start, Congestion Avoidance, Fast Restransmit, and Fast Recovery algorithms*. RFC 2001, http://www.faqs.org/rfcs/frc2001.html.

[18] Y. Cao, M. Xu and X. Fu (2012), "*Delay-based congestion control for Multipath TCP*", IEEE ICNP.

[19] Y.Li (September 2002), *Implementing High Speed TCP*. http://www.hep.ucl.ac.uk/~ytl/tcpip/hstcp/index.html.

[20] Z. Wang and J. Crowcroft (April 1992). Eliminating preodic packet losses in the 4.3-Tahoe BSD TCP Congestion control algorithm. *ACM Computer Communications Review*.

ISBN: 978-604-67-0349-5