MÔ HÌNH HÓA PHÂN CHIA TỶ LỆ POD UPF TRONG MẠNG LÕI 5G DỰA TRÊN HPA KUBERNETES

Đặng Thanh Chương, Hoa Lý Cương

Khoa Công nghệ Thông tin, Trường Đại học Khoa học, Đại học Huế

Email: dtchuong@hueuni.edu.vn, cuonghl@hueuni.edu.vn

Ngày nhận bài: 24/01/2024; ngày hoàn thành phản biện: 8/02/2024; ngày duyệt đăng: 5/3/2024

TÓM TẮT

Trong mạng lõi 5G và tương lai là 6G, chức năng mặt phẳng người dùng UPF (User Plane Function) chịu trách nhiệm vận chuyển dữ liệu từ và đến các thuê bao trong các phiên PDU (Protocol Data Unit). UPF thường được triển khai trong phần mềm và được đóng gói vào một máy ảo hoặc vùng chứa (container) mà có thể được khởi tạo dưới dạng từng phiên bản UPF với yêu cầu tài nguyên cụ thể. Để tiết kiệm mức tiêu thụ tài nguyên cần thiết, số lượng phiên bản UPF được khởi tạo phải phụ thuộc vào số lượng phiên PDU mà khách hàng yêu cầu. Điều này được kiểm soát bởi thuật toán phân chia tỉ lệ (thay đổi quy mô). Bài báo trình bày về mô hình hóa bài toán phân chia tài nguyên trong mạng 5G, sử dụng mô phỏng Pod UPF trong Kubernetes tích hợp Open5GS bằng HPA (Horizontal Pod Autoscaler). Thông qua kết quả mô phỏng, ưu điểm của mô hình này được phân tích và đánh giá.

Từ khóa: UPF, Pod, HPA, Kubernetes (K8S), 5G.

1. MỞ ĐẦU

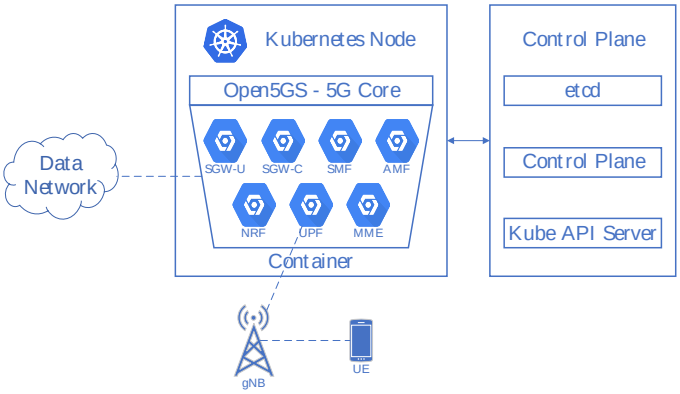
Các mạng thế hệ thứ năm (5G) và các mạng thế hệ sau 5G (chẳng hạn, thế hệ thứ sáu - 6G) sẽ cung cấp dịch vụ cho khách hàng trong các ngành dọc, như (truyền thông trong giao thông, IoT, phẫu thuật từ xa, [1]–[8]. Việc chuyển đổi các thành phần mạng và các hàm chức năng mạng từ phần cứng chuyên dụng sang các vùng chứa dựa trên phần mềm đã bắt đầu được triển khai [8]. Tổ chức 3GPP (Third Generation Partnership Project) đã đưa ra khung lõi 5G với nhiều thành phần chức năng mạng dựa trên kiến trúc hướng dịch vụ SBA (Service Based Architecture). Các kiến trúc 5G này được triển khai trên phần mềm và được thực thi bên trong máy ảo (Virtual Machine) hoặc vùng chứa (container) trong môi trường đám mây [9]. Trong tương lai, mạng 6G có thể duy trì các chức năng mặt phẳng người dùng của lõi 5G [8]. Điều đáng nhấn mạnh là các máy ảo và vùng chứa lưu trữ các chức năng mạng (được gọi là phiên bản) được thực thi

trong các khung điều phối đám mây quản lý và chỉ định tài nguyên đám mây cho các phiên bản (instances).

Trong mặt phẳng điều khiển, các phiên bản chức năng mạng phải được sắp xếp (khởi chạy và kết thúc) để đáp ứng với sự biến động của nhu cầu lưu lượng từ khách hàng. Ví dụ: khách hàng bắt đầu yêu cầu phiên PDU trước khi truyền dữ liệu; lưu lượng truy cập của các yêu cầu như vậy có thể phụ thuộc vào các khoảng thời gian trong ngày. Trong thời gian lưu lượng truy cập cao nhất, nhiều phiên bản sẽ được khởi chạy và sử dụng nhiều hơn so với thời gian thông thường. Nghĩa là, các nhà khai thác (operator) cần áp dụng các thuật toán thích hợp để kiểm soát việc sử dụng tài nguyên của các chức năng mạng.

Kubernetes là một nền tảng nguồn mở để điều phối các container, nó dùng để quản lý các ứng dụng được đóng gói dựa vào các hạ tầng đám mây. Một Pod là đơn vị tính toán nhỏ nhất có thể triển khai cho một ứng dụng nào đó và nó cũng có thể chứa nhiều container. Các máy (bao gồm máy vật lý và máy ảo) trên đám mây có thể được xem như là một nút với các nguồn tài nguyên nào đó (CPU, bộ nhớ, ổ đĩa,…). Bộ điều khiển Kubernetes sẽ cấu hình các Pod với yêu cầu cho trước về các nút và sẽ chạy các container trong các Pod này.

Trong mạng 5G, các phần tử lõi của mạng được đóng gói thành các container được tổ chức trong mỗi Pod. Tài nguyên của mỗi Pod sẽ được quản lý bởi Kubernetes. Để thiết lập UPF, một lựa chọn tự nhiên là sắp xếp các UPF đến các Pod, ở đó mỗi Pod sẽ chạy một container riêng rẽ (Hình 1).



Hình 1. Mô hình bài toán 5G trong Kubernetes [15].

Trong 5G, UPF là chức năng mặt phẳng người dùng, được sử dụng để truyền dữ liệu. Cụ thể, UPF chịu trách nhiệm duy trì bảng định tuyến và thiết lập các giao thức GTP (General packet radio service Tunneling Protocol) để chuyển tiếp các gói tin. Trước khi truyền, UPF giải mã tiêu đề của gói và tìm kiếm các quy tắc định tuyến (routing rules) trong tập dữ liệu của nó. Theo 3GPP-R15, chức năng quản lý phiên SMF (Session Management Function) có thể kiểm soát/điều khiển nhiều UPF. Tuy nhiên, đối với các

nhà khai thác, làm thế nào để thiết lập một số lượng lớn UPF với chi phí thấp nhất đồng thời đáp ứng yêu cầu của người dùng luôn là một thách thức [1].

Mạng 5G sẽ có thể phục vụ nhiều dịch vụ với đa dạng yêu cầu khác nhau. Do đó, các phiên PDU có thể có nhiều yêu cầu khác nhau. Để hoạt động được, các điều phối mạng có thể áp dụng một cách tiếp cận nào đó (thủ công) để giới hạn số loại phiên PDU. Đối với mỗi loại phiên PDU, một loại UPF sẽ được tạo ra để xác định yêu cầu về tài nguyên hệ thống như bộ nhớ, CPU,….

Bài báo này xem xét việc áp dụng chức năng HPA (Horizontal Pod Autoscaling) trong Kubernetes nhằm đánh giá bài toán quản lý tài nguyên theo hướng phân chia tỉ lệ tự động (autoscaling) đối với các phiên bản UPF trong lõi 5G. Giả thiết rằng các phiên bản UPF được kiểm soát bởi các vùng chứa trong Kubernetes.

Các đóng góp của bài báo gồm:

- Mô hình hóa mạng lõi 5G bằng Open5GS tích hợp Kubernetes, trong đó các thành phần của Open5GS được đóng gói bằng thành các Pod để quản lý.
- Xây dựng bài toán phân chia tỉ lệ (mở rộng quy mô) đối với các phiên bản Pod UPF trong Kubernetes với giải thuật HPA.

Các phần còn lại của bài báo được bố cục như sau: Phần II khảo sát các tài liệu liên quan về phân chia tỉ lệ tự động (autoscaling) trong đám mây để quản lý tài nguyên trong lõi 5G. Phần III sẽ mô tả bài toán phân chia tỉ lệ (mở rộng quy mô) đối với các phiên bản UPF trong Kubernetes. Phần IV trình bày kết quả phân tích dựa trên mô hình đề xuất. Cuối cùng là phần kết luận và hướng phát triển, được trình bày chi tiết trong phần V.

2. CÁC CÔNG TRÌNH NGHIÊN CỨU LIÊN QUAN

Đã có một vài công trình nghiên cứu về hiệu năng của Kubernetes. Nhóm tác giả Nguyen và cộng sự [7] đã so sánh các giải pháp độ đo (metrics-server solution) và nêu bật ảnh hưởng của các tham số cấu hình khác nhau trong cả hai chỉ số tài nguyên. Một số công trình đã đặc biệt tập trung nghiên cứu về mạng lõi 5G. Járó và cộng sự [8] đã thảo luận về sự phát triển và ảo hóa của các dịch vụ mạng. Họ đã xem xét tính khả dụng, kích thước và hoạt động của máy chủ ứng dụng viễn thông TAP (Telecommunication Application Server). Tang và cộng sự [9] đã đề xuất một phương pháp dự báo lưu lượng truy cập dựa trên hồi quy tuyến tính (linear regression-based traffic forecasting method) cho các chức năng mạng ảo VNF (Virtual Network Function). Họ cũng thiết kế các thuật toán để triển khai các chuỗi chức năng dịch vụ của VNF theo lưu lượng dự đoán (predicted traffic). Alawe và cộng sự [10] với công cụ OpenAirInterface đã sử dụng các kỹ thuật học sâu để dự đoán lưu lượng mạng 5G và phân chia tỉ lệ (mở rộng quy mô)

các phiên bản chức năng quản lý truy cập AMF (Access and Mobility Management Function).

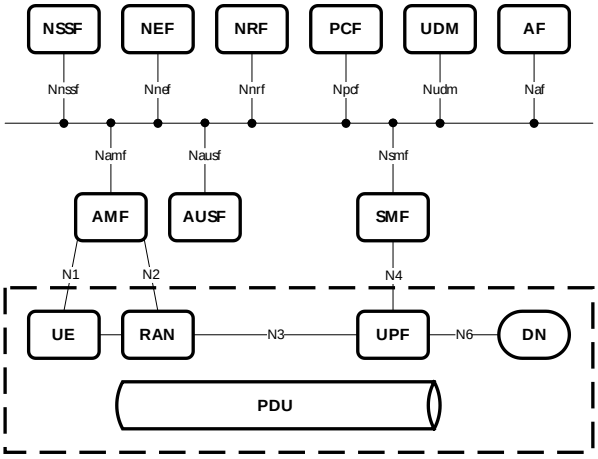
Subramanya và cộng sự [11] đã sử dụng mạng Perceptron đa lớp (multilayer perceptrons) để dự đoán số lượng UPF cần thiết trong hệ thống. Kumar và cộng sự [12] đã đưa ra một phương pháp để tăng quy mô và nhân rộng (scaling up and scaling out) các phiên bản UPF và triển khai mạng lõi 5G trên đám mây AWS. Rotter và Do [4] đã trình bày mô hình phân tích hàng đợi để phân chia tỉ lệ của các phiên bản 5G UPF dựa trên các thuật toán ngưỡng. Mô hình hàng đợi của họ cung cấp đánh giá nhanh về các thuật toán chia tỷ lệ dựa trên hai ngưỡng.

Từ việc xem xét các tài liệu liên quan, có thể nhận thấy rằng các thuật toán phân chia tỷ lệ được báo cáo trong hầu hết các công trình cho đến nay liên quan đến các vấn đề như kiểm soát số lượng VM, VNF, phiên bản container bằng cách sử dụng một số ngưỡng nhưng chưa đề cập đến việc quản lý cấp phát UPF là một chức năng quan trọng trong mạng 5G để truyền kết nối dữ liệu giữa người dùng với mạng dữ liệu DN (Data Network). Do đó, mục tiêu chính trong bài báo này đề cập đến mô hình hóa bài toán phân chia tỷ lệ UPF Pod với kiến trúc mô phỏng theo mạng lõi 5G của Open5GS (chưa được thực hiện mô phỏng ở các công trình nêu trên) dựa trên nền tảng mã nguồn mở Kubernetes và thuật toán HPA để thực hiện phân chia tỷ lệ sẽ được giới thiệu cụ thể ở các phần sau.

3. MÔ HÌNH HÓA BÀI TOÁN PHÂN CHIA TỈ LỆ CÁC PHIÊN BẢN UPF 5G DỰA TRÊN HPA TRONG KHUNG K8S

3.1. Các phiên bản 5G UPF

Kết nối giữa thiết bị người dùng UE (User Equipment) và mạng dữ liệu DN trong 5G yêu cầu thiết lập phiên PDU. Trong kết nối này, trước tiên, UE kết nối trực tiếp với trạm cơ sở gNB (gNodeB) trong mạng truy cập vô tuyến RAN (Radio Access Network) và thông qua mạng truyền tải đến mạng lõi 5G, cung cấp điểm cuối cho mạng dữ liệu bên ngoài (Hình 2).

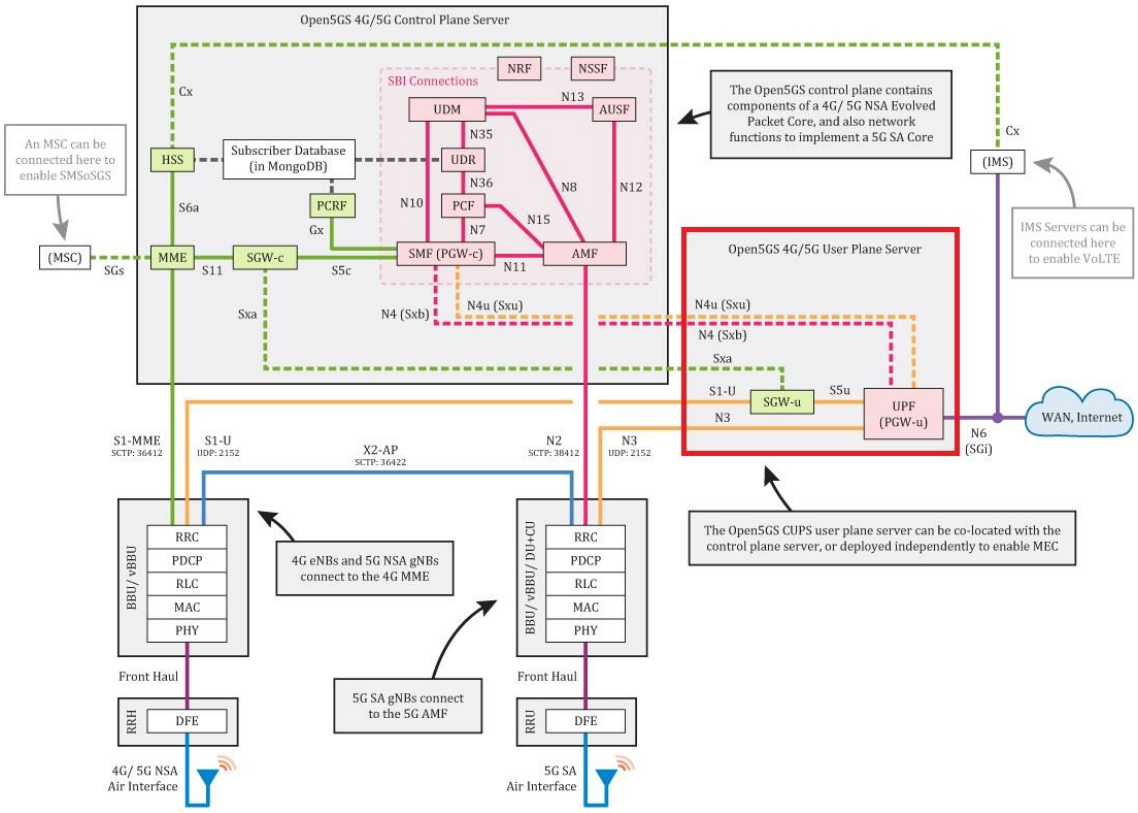


Hình 2. Kiến trúc mạng 5G theo 3GPP [13].

Mạng truyền tải có thể là kết nối không dây, có dây hoặc mạng truyền dẫn quang và lõi 5G bao gồm một tập hợp các hàm chức năng mạng khác nhau triển khai kiến trúc dựa trên dịch vụ SBA (Service-Based Architecture) [1]. Các hàm chức năng mạng bao gồm chức năng quản lý truy cập và di động AMF, thực hiện xác thực các UE và kiểm soát quyền truy cập của UE vào cơ sở hạ tầng; chức năng quản lý phiên SMF, giúp thiết lập/đóng các phiên PDU và theo dõi trạng thái của phiên PDU; và chức năng mặt phẳng người dùng UPF [1]-[3]. Trong khi AMF và SMF là một phần của mặt phẳng điều khiển, thì UPF chịu trách nhiệm về chức năng của mặt phẳng người dùng. UPF phục vụ như là một điểm neo (anchor) phiên PDU và cung cấp điểm kết nối cho mạng truy cập tới lõi 5G. Ngoài ra, UPF cũng xử lý việc kiểm tra, định tuyến và chuyển tiếp các gói và nó cũng có thể xử lý QoS, áp dụng các quy tắc lưu lượng cụ thể, … Việc phân tách mặt phẳng điều khiển và người dùng CUPS (Control and User Plane Separation) đảm bảo rằng các thành phần riêng lẻ có thể mở rộng quy mô một cách độc lập và cũng cho phép xử lý dữ liệu được đặt gần biên mạng hơn.

Mô hình hóa kiến trúc mạng lõi 5G có thể được mô hình bằng bộ mô phỏng Open5GS, như được trình bày trong Hình 3, trong đó nó được chia thành 4 khối riêng biệt với các chức năng khác nhau:

- Khối mặt phẳng điều khiển (Open5GS 4G/5G Control Plane Server) chứa các thành phần liên quan đến gói tin của mạng 4G/5G, đồng thời cũng chứa các chức năng mạng để triển khai lõi 5G.
- Khối mặt phẳng người dùng (Open5GS 4G/5G User Plane Server) đảm nhận vai trò quản lý lưu lượng dữ liệu truyền tải giữa mặt phẳng điều khiển với các trạm cơ sở 4G/5G, đây cũng là vùng triển khai các phiên UPF.
- Các trạm cơ sở eNB trong mạng 4G (BBU/vBBU) và trạm cơ sở gNB trong mạng 5G (BBU/vBBU/DU+CU) có chức năng kết nối mạng với các chức năng và dịch vụ di động.



Hình 3. Kiến trúc mô phỏng Open5GS [14].

3.2. HPA trong K8S

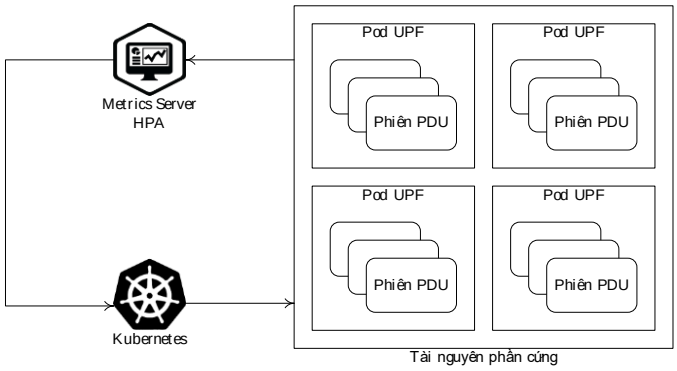
HPA là thuật toán phân chia tỉ lệ tự động của Kubernetes. Nó sử dụng mức hiệu suất CPU trung bình, được ký hiệu là $\bar{\rho}$, như một sự quan sát (observation) để tính toán số lượng các Pod cần thiết, được ký hiệu là $d_{\text{desired}}(t)$ tại thời điểm t. Có hai tham số có thể định cấu hình: hiệu suất sử dụng CPU ρ_{target} và dung sai (tolerance) ν. Phương trình được sử dụng bởi HPA là [6]:

$$d_{\text{desired}}(t) = \left\lceil d_{\text{on}}(t) \frac{\bar{\rho}(t)}{\rho_{\text{target}}} \right\rceil \tag{1}$$

Trong đó $\lceil x \rceil$ là ký hiệu hàm phần nguyên lớn hơn hay bằng x và $d_{\text{on}}(t)$ là số lượng các Pod tại thời điểm t. HPA sẽ kiểm tra $d_{\text{desired}}(t)/d_{\text{on}}(t) \in [1-\nu, 1+\nu]$. Nếu không, HPA sẽ đưa ra hành động chia tỉ lệ để đưa số lượng bản sao đến gần giá trị mong muốn hơn. Quy trình được mô tả ở trên được thực hiện định kỳ với khoảng thời gian ΔT. Khoảng thời gian này có thể được đặt thông qua cấu hình Kubernetes [8-9].

Ứng dụng của HPA tích hợp trong Kubernetes cần các giá trị thích hợp của ρ_{target} và ν. Người điều hành hệ thống có thể trải qua một quá trình thử nghiệm khó khăn và

sai sót để tìm ra cấu hình có thể tối thiểu số lượng Pod trong khi vẫn duy trì mức QoS. Trong Kubernetes, HPA đóng vai trò tự động cập nhật tài nguyên khối lượng công việc (workload resource) (chẳng hạn như Deployment), với mục đích tự động phân chia tỉ lệ (thay đổi quy mô) khối lượng công việc để phù hợp với nhu cầu (Hình 4).

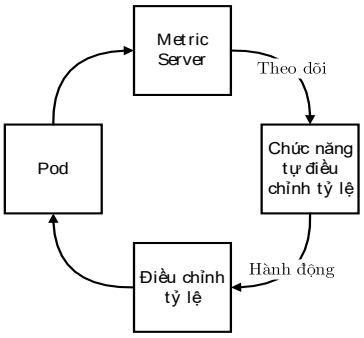


Hình 4. Cách thức hoạt động của HPA trong Kubernetes [6].

Chia tỷ lệ theo chiều ngang (Horizontal scaling) có nghĩa là phản ứng đối với trường hợp khi tải tăng lên là triển khai nhiều Pod hơn. Điều này khác với chia tỷ lệ theo chiều dọc, đối với Kubernetes có nghĩa là chỉ định nhiều tài nguyên hơn (ví dụ: bộ nhớ hoặc CPU) cho các Pod đang chạy cho khối lượng công việc. Nếu tải giảm và số lượng Pod cao hơn mức tối thiểu đã định cấu hình, thì HorizontalPodAutoscaler sẽ thu nhỏ quy mô lại bằng cách giảm tài nguyên khối lượng công việc (Deployment, StatefulSet hoặc tài nguyên tương tự khác).

HPA được triển khai dưới dạng tài nguyên API Kubernetes và bộ điều khiển. Tài nguyên xác định hành vi của bộ điều khiển. Bộ điều khiển tự động thay đổi quy mô nhóm theo chiều ngang, chạy trong mặt phẳng điều khiển Kubernetes, điều chỉnh định kỳ quy mô mong muốn của mục tiêu (ví dụ: triển khai) để khớp với các số liệu được quan sát như mức sử dụng CPU trung bình, mức sử dụng bộ nhớ trung bình hoặc bất kỳ số liệu tùy chỉnh nào khác mà người dùng chỉ định.

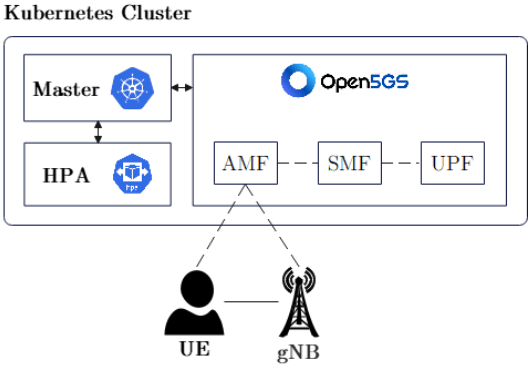
Hình 5 thể hiện mối quan hệ giữa chức năng tự điều chỉnh tỷ lệ với các thành phần khác. Một Metric Server sẽ giám sát việc sử dụng tài nguyên của các Pod và cung cấp chức năng tự điều chỉnh tỷ lệ và thống kê các số liệu thông qua các API Metric. Chức năng tự điều chỉnh tỷ lệ sẽ tính toán số lượng cần thiết số lượng bản sao của Pod và có thể quyết định xem có thực hiện điều chỉnh hay không. Việc điều chỉnh này có thể thực hiện thông qua giao diện điều khiển.



Hình 5. Mô hình mô tả chức năng tự điều chỉnh phân chia tỷ lệ Pod trong Kubernetes.

3.3. Mô phỏng và phân tích kết quả

Open5GS ở đây được tích hợp vào Kubernetes với các Pod chính là các thành phần chủ yếu trong kiến trúc mạng 5G của 3GPP. Ở đây chúng tôi chỉ thực hiện phân chia tỷ lệ HPA đối với Pod UPF với chức năng HPA được thực hiện trong mặt phẳng điều khiển.



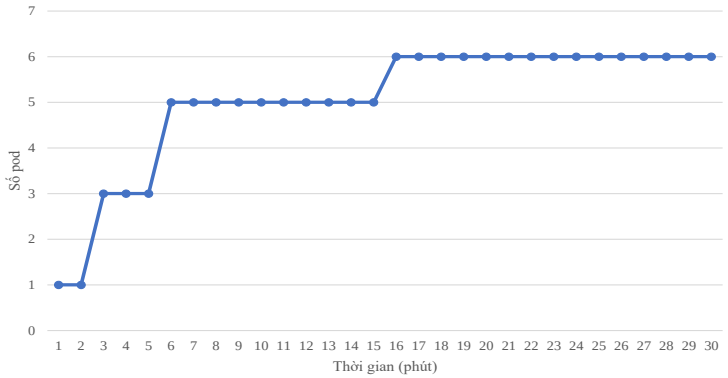
Hình 6. Mô hình bài toán 5G trong Kubernetes với Kubernetes [15].

Chúng tôi mô phỏng với các tham số như sau:

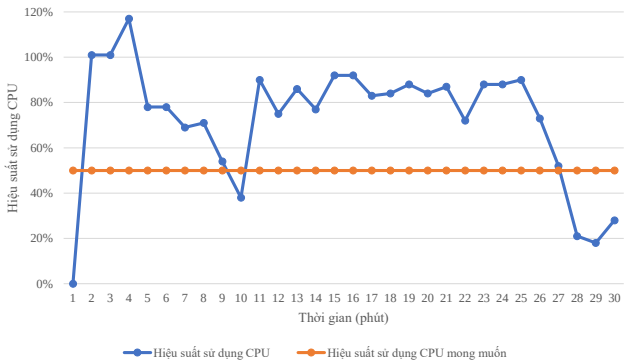
Bảng 1. Bảng các giá trị tham số mô phỏng.

Tham số	Giá trị
Số lượng nút master	1
Số lượng nút worker	1
Số lượng Pod tối thiểu (D_{min})	1
Số lượng Pod tối đa có thể (D_{max})	10
Khoảng thời gian giữa hai yêu cầu đến (khoảng thời gian giữa hai phiên PDU đến) liên tiếp	0,01

Với kết quả mô phỏng thực tế theo những tham số mô phỏng như trên, bình quân mỗi Pod UPF sẽ xử lý được 1000 yêu cầu trong 1 phút, sau đó hệ thống tăng lên 2 Pod UPF thành 3 Pod UPF do khi này yêu cầu đến nhiều dẫn đến hiệu suất sử dụng CPU tăng cao (117% và cao hơn mức mong muốn là 50%) (Hình 7, Hình 8), tiếp theo đó hệ thống cấp phát thêm 2 Pod UPF (do khi này hiệu suất sử dụng CPU tuy giảm nhưng vẫn gần gấp đôi). Thời gian sau đó do khi này hiệu suất sử dụng CPU đã ổn định và khi tăng lên 92% (điểm cực đại tính từ khi phút thứ 6 trở đi) hệ thống chỉ bổ sung thêm 1 CPU. Khi này hiệu suất sử dụng CPU đã ổn định nên không thực hiện cấp phát thêm Pod UPF. Chúng ta có thể thấy rằng lúc đầu HPA sẽ bổ sung mỗi lần thêm 2 Pod UPF như khi hệ thống đã gần đạt được trạng thái ổn định hệ thống sẽ chỉ bổ sung 1 Pod UPF để đảm bảo rằng số lượng Pod UPF không vượt quá số lượng Pod UPF tối đa ở đây là 10.



Hình 7. Số lượng Pod UPF khi thực hiện HPA.



Hình 8. Hiệu suất sử dụng CPU khi thực hiện HPA.

Trong khi đó hiệu suất sử dụng CPU có sự biến thiên lên xuống. Hiệu suất sử dụng CPU cao nhất là 117% thì hệ thống sẽ thực hiện HPA để đáp ứng yêu cầu của hệ thống (Hình 8). Chú ý rằng ở Hình 8 khi hiệu suất CPU giữ nguyên tức là khi đó hệ thống sẽ thực hiện HPA để tăng Pod UPF. Điều này được giải thích là do khi hiệu suất CPU đạt quá hiệu suất CPU mong muốn, hệ thống sẽ tự động bổ sung thêm Pod UPF để thực hiện HPA cân bằng tải hiệu suất trong hệ thống khi số lượng yêu cầu đến ngày

càng tăng. Dựa vào công thức (1), chúng tôi đã tính được số lượng Pod UPF mong muốn ($d_{desired}$) theo lý thuyết ở Bảng 2. Trong đó, ở phút thứ 3 sẽ có sự cách biệt lớn từ lý thuyết ($d_{desired} = 7$) so với thực tế ($d_{on} = 3$) ($d_{desired}/d_{on} = 2,3$) và ở phút thứ 28 với $d_{desired} = 2$ so với thực tế $d_{on} = 6$ ($d_{desired}/d_{on} = 0,3$). Điều này có thể giải thích do lúc đầu hệ thống chưa cấp phát Pod UPF nhiều và khi hệ thống đã dần ổn định thì sẽ giải phóng những Pod UPF, tuy nhiên ở gần cuối khi này số lượng HPA theo yêu cầu đã giảm nhưng hệ thống chưa giảm do đó đây là vấn đề cần tối ưu trong tương lai.

Bảng 2. Bảng các giá trị $d_{desired}$ và $d_{desired}/d_{on}$.

Phút	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$d_{desired}$	0	2	6	7	5	8	7	7	5	4	9	8	9	8	9
$d_{desired}/d_{on}$	0,0	2,0	2,0	2,3	1,7	1,6	1,4	1,4	1,0	0,8	1,8	1,6	1,8	1,6	1,8
Phút	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
$d_{desired}$	11	10	10	11	10	10	9	11	11	11	9	6	3	2	3
$d_{desired}/d_{on}$	1,8	1,7	1,7	1,8	1,7	1,7	1,5	1,8	1,8	1,8	1,5	1,0	0,5	0,3	0,5

4. KẾT LUẬN

Trong bài báo này chúng tôi đã giới thiệu mô hình và mô phỏng 5G trong Kubernetes kết hợp với HPA để thực hiện phân chia tự động Pod UPF theo thực tế, với Pod UPF là cầu nối quan trọng để đảm bảo kết nối giữa các trạm cơ sở gNB với mạng dữ liệu DN. Kết quả mô phỏng đã chỉ ra rằng hệ thống chúng tôi xây dựng là đúng và phù hợp với thực tế. Tuy nhiên vấn đề tối ưu hiệu suất sử dụng CPU chưa được chúng tôi tính tới mà sẽ nghiên cứu trong thời gian tới. Trong hướng tiếp theo chúng tôi cũng sẽ xem xét mở rộng mô hình này bằng cách đưa vào các ngưỡng cấp phát và thu hồi UPF (theo lưu lượng UE đến), đồng thời sẽ áp dụng học tăng cường nhằm cải tiến HPA tự động phân chia tỷ lệ Pod UPF.

TÀI LIỆU THAM KHẢO

[1]. 5GPP, "View on 5G Architecture, "White paper 22.891,5GPPP Architecture Working Group, 07 2018. Version 14.2.0.

[2]. 3GPP, "5G; Study on scenarios and requirements for next generation access technologies," Technical Specification (TS) 38.913, 3rd Generation Partnership Project (3GPP), 07 2020. Version 16.0.0 Release 16.

[3]. 3GPP, "Technical Specification Group Services and System As- pects;System architecture for the 5G System (5GS);Stage 2," Technical Specification (TS) 23.501, 3rd Generation Partnership Project (3GPP), 12 2020. Version 16.7.0.

[4]. C. Rotter and T. V. Do, "A queueing model for threshold-based scaling of UPF instances in 5G core," IEEE Access, vol. 9, pp. 81443–81453, 2021.

[5]. Kubernetes Documentation. Accessed: Aug. 6, 2023. [Online]. Available: https://kubernetes.io/docs/home/

[6]. Nguyen, Hai T., Do, TV, Rotter, "Scaling UPF Instances in 5G/6G Core with Deep Reinforcement Learning", IEEE ACCESS 9 pp. 165892-165906. , 15 p. (2021).

[7]. T.-T. Nguyen, Y.-J. Yeom, T. Kim, D.-H. Park, and S. Kim, ''Horizontal pod autoscaling in Kubernetes for elastic container orchestration,'' Sensors, vol. 20, no. 16, p. 4621, Aug. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/16/4621.

[8]. G. Jaro, A. Hilt, L. Nagy, M. A. Tundik, and J. Varga, ''Evolution towards telco-cloud: Reflections on dimensioning, availability and operability: (Invited paper),'' in Proc. 42nd Int. Conf. Telecommun. Signal Process. (TSP), Jul. 2019, pp. 1–8.

[9]. H. Tang, D. Zhou, and D. Chen, ''Dynamic network function instance scaling based on traffic forecasting and VNF placement in operator data centers,'' IEEE Trans. Parallel Distrib. Syst., vol. 30, no. 3, pp. 530–543, 2019.

[10]. I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, ''Improving traffic forecasting for 5G core network scalability: A machine learning approach,'' IEEE/ACM Trans. Netw., vol. 32, no. 6, pp. 42–49, Nov./Dec. 2018.

[11]. T. Subramanya, D. Harutyunyan, and R. Riggio, ''Machine learning driven service function chain placement and scaling in MEC-enabled 5G networks,'' Comput. Netw., vol. 166, Jan. 2020, Art. no. 106980. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619310254.

[12]. D. Kumar, S. Chakrabarti, A. S. Rajan, and J. Huang, ''Scaling telecom core network functions in public cloud infrastructure,'' in Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci. (CloudCom), Dec. 2020, pp. 9–16.

[13]. https://www.3gpp.org/technologies/5g-system-overview

[14]. https://open5gs.org/open5gs/

[15]. https://aws.amazon.com/vi/blogs/industries/5g-core-implementation-on-amazon-elastic-kubernetes-service-anywhere-on-bare-metal-2/

MODELING OF SCALING UPF POD IN 5G CORE NETWORK BASED ON HPA KUBERNETES

Dang Thanh Chuong, Hoa Ly Cuong

University of Sciences, Hue University

Email: dtchuong@hueuni.edu.vn, cuonghl@hueuni.edu.vn

ABSTRACT

The UPF (User Plane Function) is responsible for transporting data to subscribers in PDU (Protocol Data Unit) sessions in the 5G and future 6G networks. UPF is typically implemented in software and packaged into a virtual machine or container that can be instantiated as individual UPF instances with specific resource requirements. The number of created UPF instances should be determined by the number of PDU sessions requested by the client to save the required resource consumption. This is controlled by a scaling algorithm. The article presents a model of resource division problem in 5G networks, using the simulation of Pod UPF in Kubernetes integrated with Open5GS via Horizontal Pod Autoscaler (HPA). Through simulation results, the advantages of this model are analyzed and evaluated.

Keywords:. UPF, Pod, HPA, Kubernetes (K8S), 5G.

Đặng Thanh Chương sinh ngày 23/3/1975 tại TP Vinh. Ông tốt nghiệp cử nhân ngành Vật lý năm 1997 và thạc sĩ chuyên ngành Tin học tại Trường ĐHKH Huế vào năm 2004. Ông nhận học vị tiến sĩ chuyên ngành Toán học năm 2014 tại Viện CNTT - Viện hàn lâm khoa học và công nghệ Việt Nam. Từ năm 1997 đến nay, ông công tác tại khoa CNTT, Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Lý thuyết hàng đợi, Mạng 5G, Mạng truyền dẫn quang.

Hoa Lý Cương sinh năm 1990 tại An Giang. Ông tốt nghiệp cử nhân ngành Công nghệ Thông tin năm 2012 tại Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia thành phố Hồ Chí Minh và thạc sĩ chuyên ngành Khoa học Máy tính tại Trường Đại học Khoa học, Đai học Huế vào năm 2017. Hiện tại đang là nghiên cứu sinh ngành Khoa học Máy tính tại Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Lý thuyết hàng đợi, Đánh giá hiệu năng mạng.