ASSESSMENT OF TRANSMISSION DELAY OF REAL-TIME VIDEO STREAMING FROM A TELEMETRY SURVEILLANCE SYSTEM

Nguyen Toan Thang, Dinh Xuan Lam*, Vo Chi Dong, Do Manh Dung, Do Thi Phuong
TNU - University of Information and Communication Technology

ARTICLE INFO

Received: 17/4/2023
Revised: 24/5/2023
Published: 24/5/2023

KEYWORDS

Internet of Things
Raspberry Pi
Surveillance system
Video streaming
Video Delay

ABSTRACT

The advent of the industrial revolution 4.0 and the explosive development of advanced technologies such as IoT (Internet of Things) have solved most of the essential needs of a constantly changing digital society. One of those is the environmental video surveillance system. This paper presents such a system including compact devices such as Raspberry Pi. This system is assembled and programmed to be able to transmit images or videos from the environment area via the Internet to the user's end devices. In order to transmit video from the surveillance system to the user's end devices, the most popular video transmission protocols are tested in practice, such as HLS, DASH, RTMP, and RTSP. As a result, a difference in video transmission latency between these protocols through different experimental scenarios was found. In particular, the HTTP-based protocol group such as HLS and DASH has higher transmission latency than RTMP and RTSP and much higher than the latency of video encoding at IoT devices. The main reason comes from the different streaming video and video encoding techniques between the above protocols.

ĐÁNH GIÁ ĐỘ TRỄ TRUYỀN TẢI VIDEO THỜI GIAN THỰC TỪ HỆ THỐNG QUAN TRẮC HÌNH ẢNH TỪ XA

Nguyễn Toàn Thắng, Đinh Xuân Lâm*, Võ Chí Đông, Đỗ Mạnh Dũng, Đỗ Thị Phượng
Trường Đại học Công nghệ thông tin và Truyền thông – ĐH Thái Nguyên

THÔNG TIN BÀI BÁO

Ngày nhận bài: 17/4/2023
Ngày hoàn thiện: 24/5/2023
Ngày đăng: 24/5/2023

TỪ KHÓA

Internet vạn vật
Raspberry Pi
Trạm quan trắc
Video streaming
Độ trễ video

TÓM TẮT

Sự ra đời của cuộc cách mạng công nghiệp 4.0 cùng với sự phát triển bùng nổ của các công nghệ tiến tiến như IoT (Internet of Things) đã giải quyết phần lớn các nhu cầu thiết yếu của một xã hội số đang không ngừng thay đổi. Một trong số đó là các hệ thống quan trắc môi trường bằng hình ảnh từ xa. Bài báo này trình bày một hệ thống quan trắc như vậy bao gồm các thiết bị nhỏ gọn như Raspberry Pi. Hệ thống này được lắp ghép và lập trình để có thể truyền được hình ảnh hoặc video từ môi trường thông qua mạng Internet đến các thiết bị đầu cuối của người dùng. Để truyền được video từ trạm quan trắc đến thiết bị đầu cuối, các giao thức truyền tải video phổ biến lần lượt được thử nghiệm trong thực tế như HLS, DASH, RTMP và RTSP. Kết quả cho thấy có sự khác biệt về độ trễ truyền tải video giữa các giao thức này thông qua các kịch bản thí nghiệm khác nhau. Cụ thể, nhóm giao thức dựa trên HTTP như HLS và DASH có độ trễ truyền tải cao hơn nhóm giao thức RTMP và RTSP và cao hơn hẳn độ trễ mã hóa video tại thiết bị IoT. Nguyên nhân chính đến từ sự khác nhau về kỹ thuật truyền video thời gian thực và công nghệ mã hóa video giữa các giao thức trên.

DOI: https://doi.org/10.34238/tnu-jst.7770

* Corresponding author. *Email: dxlam@ictu.edu.vn*

1. Giới thiệu

Sự phát triển bùng nổ của các dịch vụ mạng Internet và xu hướng phát triển mạnh mẽ của các nền tảng công nghệ 4.0 như công nghệ Internet vạn vật (Internet of Things - IoT) đã mang lại cho người dùng những phương pháp mới trong việc theo dõi sự thay đổi của môi trường hay cây trồng, vật nuôi trong lĩnh vực nông nghiệp [1], [2]. Một hệ thống IoT ứng dụng trong giám sát từ xa được triển khai ngoài môi trường có thể bao gồm máy tính nhỏ, hệ thống cảm biến, cụm camera với hệ thống cơ điện tử điều hướng, mô đun kết nối Internet sử dụng mạng 3G hoặc LTE, ắc qui kèm hệ thống sạc tự động từ pin năng lượng mặt trời [3]. Các thiết bị này được kết nối và điều khiển bởi một máy tính nhỏ ví dụ như Raspberry Pi [4]. Một hệ thống như vậy được gọi là trạm quan trắc. Toàn bộ dữ liệu hình ảnh và video từ trạm quan trắc này sẽ được truyền đến một máy chủ trung gian trên điện toán đám mây, dữ liệu sau đó được xử lý và truyền tới thiết bị đầu cuối của người dùng qua mạng Internet.

Trên thực tế, camera của một hệ thống trạm quan trắc có khả năng cung cấp các loại phân giải video khác nhau từ SD, HD cho đến Full HD, thậm chí có thể là 2K hoặc 4K. Để truyền các dữ liệu video này tới thiết bị của người dùng đầu cuối, hạ tầng của trạm sử dụng các giao thức khác nhau như RTMP, RTSP, HLS hoặc DASH [5], [6]. Đây là các giao thức phổ biến trong video streaming đã được thế giới nghiên cứu nhiều, tuy nhiên trong một hệ thống trạm quan trắc từ xa, với các yêu cầu đặc trưng như điều khiển trong thời gian thực hoặc truyền hình ảnh rõ nét để quan sát đối tượng thì chưa có nhiều nhà nghiên cứu tìm hiểu, đặc biệt là độ trễ truyền tải video trong thời gian thực. Với hạ tầng mạng truyền thống thiết kế cho trạm chủ yếu là không dây, thì việc đánh giá độ trễ truyền tải video của những giao thức này cũng như chất lượng video nhận được ở người dùng đầu cuối là cần thiết để từ đó các nhà nghiên cứu, nhà cung cấp dịch vụ có thể cải thiện khả năng truyền video cho các hệ thống quan trắc ngoài trời.

Đối tượng nghiên cứu chính của bài báo là một hệ thống trạm quan trắc có trang bị camera phân giải cao có khả năng truyền dữ liệu qua mạng Internet thông qua một máy chủ trung gian sử dụng các giao thức video streaming. Toàn bộ hệ thống sử dụng giao thức RTMP để truyền video từ trạm đến máy chủ, video từ máy chủ được truyền tới thiết bị đầu cuối của người dùng bằng một trong các giao thức RTMP, RTSP, HLS hoặc DASH. Nghiên cứu tập trung đánh giá và so sánh độ trễ truyền tải video của các giao thức này dựa trên quan sát và phân tích thực tế. Kết quả của nghiên cứu sẽ chỉ ra ưu nhược điểm của các giao thức truyền video phổ biến hiện nay. Từ đó, các nhà cung cấp dịch vụ mạng và nhà sản xuất hệ thống trạm quan trắc có thể lựa chọn giao thức phù hợp hoặc cải tiến để tăng chất lượng truyền video. Bài báo cũng trình bày chi tiết phương pháp thí nghiệm, đánh giá độ trễ truyền tải video của các giao thức truyền video trực tuyến, thời gian thực, là tài liệu tham khảo cho sinh viên, học viên và các nhà nghiên cứu cùng lĩnh vực.

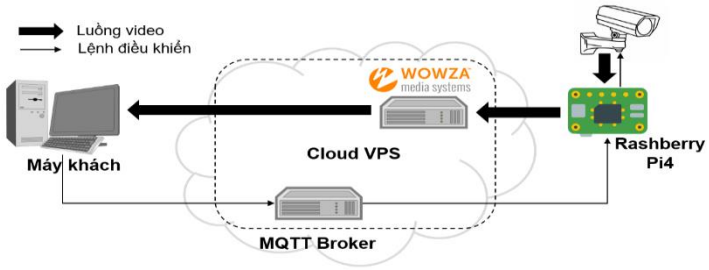
Hệ thống quan trắc dựa trên IoT đã và đang được nhiều nhà khoa học trên thế giới nghiên cứu, Prathibha và đồng tác giả trong [7] giới thiệu một hệ thống giám sát trong nông nghiệp hoàn chỉnh về mặt thiết kế vi mạch và các thiết bị. Một hệ thống tương tự như vậy và có kèm thêm thiết kế ứng dụng trên điện thoại di động cũng được đề xuất trong [8] bởi Suma và đồng tác giả. Tuy nhiên, trong các bài báo này, các tác giả chưa đánh giá hiệu năng của hệ thống đã thiết kế, mà chủ yếu là giới thiệu. Trong nghiên cứu [3], Abas và đồng tác giả đã giới thiệu chi tiết một hệ thống quan trắc hoàn chỉnh bao gồm cả thiết kế phần cứng, hệ thống mạng và nguyên lý hoạt động. Các tác giả đã đánh giá hiệu năng của hệ thống thông qua một số tham số QoS như tốc độ truyền file trung bình, độ trễ và tỷ lệ gói tin bị lỗi. Tuy nhiên, các tác giả đánh giá hiệu năng của hệ thống dựa trên kiến trúc mạng mesh gồm nhiều trạm quan trắc khác nhau giao tiếp qua kênh truyền không dây. Nghiên cứu của chúng tôi tập trung vào đánh giá ưu nhược điểm của các giao thức truyền phát video đầu cuối. Các nghiên cứu của Lyndon trong [5] có tính tương đồng với chúng tôi khi tác giả có so sánh hiệu năng của các giao thức RTMP, RTSP và HLS. Tuy nhiên, nghiên cứu này chỉ tập trung đánh giá hoạt động của các giao thức trên thiết bị di động dựa trên

mô hình chấm điểm, trong khi nghiên cứu của chúng tôi so sánh hiệu năng của các giao thức dựa trên độ trễ truyền tải video đầu cuối.

2. Phương pháp nghiên cứu

2.1. Kiến trúc tổng quát hệ thống quan trắc từ xa bằng hình ảnh

Hình 1 mô hình hóa kết nối mạng của các thiết bị thuộc hệ thống quan trắc, bao gồm các thiết bị đầu cuối, không gian điện toán đám mây và kết nối mạng giữa các thành phần. Trong thí nghiệm, *Máy khách* là một máy tính cá nhân, chạy hệ điều hành Windows 11 với CPU Intel Core i5, bộ nhớ RAM 16Gb, ổ cứng SSD. Cấu hình trên đảm bảo quá trình giải mã và render video không gây ra nhiều độ trễ.



Hình 1. *Mô hình kết nối các thiết bị trong hệ thống quan trắc*

Hình 2. *Mạch Raspberry Pi4*

Phía cụm camera quan trắc bên phải Hình 1 bao gồm một máy tính Raspberry Pi4, cấu hình bao gồm CPU tốc độ 1.5GHz quad-core 64-bit ARM Cortex-A72 CPU, bộ nhớ RAM 4GB of LPDDR4 SDRAM và Full throughput Gigabit Ethernet. Raspberry Pi Camera 12 megapixel có thể thay đổi ống kính sử dụng cảm biến Sony IMX477 cho độ phân giải của ảnh chụp lên tới 4K, video quay Full HD. Cả máy tính khách và cụm Raspberry Pi4 này được kết nối Internet băng thông rộng lên tới 80 Mbps thông qua mạng LAN của phòng thí nghiệm. Video truyền từ Pi camera được cấu hình với độ phân giải 1920x1080 ở tốc độ framerate 60fps.

Để có thể điều khiển cụm Raspberry Pi4 và camera qua mạng Internet, máy khách phát lệnh lên MQTT Broker thông qua một máy chủ riêng ảo (Virtual Private Server - VPS) có cấu hình gồm 2 core CPU 3.5GHz, bộ nhớ RAM 4Gb và không gian lưu trữ SSD. VPS chạy hệ điều hành Windows Server và cài hai ứng dụng MQTT Broker và Wowza Media Server sẽ được mô tả chi tiết trong phần sau.

2.1.1. Raspberry Pi4

Thiết bị chính của trạm bao gồm mạch Raspberry Pi4, cảm biến camera kèm ống kính được lắp trực tiếp vào mạch. Raspberry Pi4 là một loại máy tính nhỏ gọn được thiết kế để đáp ứng nhu cầu của các nhà phát triển và người dùng cá nhân. Được phát triển bởi Raspberry Pi Foundation, Raspberry Pi4 có thể được sử dụng để xây dựng các dự án đa dạng, từ các thiết bị IoT đơn giản đến các hệ thống máy tính phức tạp. Máy tính Raspberry Pi4 trong Hình 2 được trang bị các bộ vi xử lý ARM, bộ nhớ RAM, kết nối Internet và các cổng kết nối khác để kết nối với các thiết bị ngoại vi.

2.1.2. Raspberry Pi Camera

Raspberry Pi Camera 12 megapixel có thể thay đổi ống kính và có thể được kết nối với các phiên bản Raspberry Pi thông qua cổng DSI và cổng CSI. Máy ảnh được điều khiển bằng phần mềm có sẵn trên Raspberry Pi OS và hỗ trợ nhiều chế độ chụp ảnh khác nhau, bao gồm chụp ảnh đơn, chụp ảnh liên tục và chụp ảnh timelapse. Máy ảnh Raspberry Pi có khả năng chụp ảnh và quay video với độ phân giải cao lên đến 12 megapixel và có thể ghi lại video 4K với tốc độ 30 khung hình/giây hoặc video 1080p với tốc độ 60 khung hình/giây.

Hình 3. *Raspberry Pi Camera*

2.1.3. MQTT Server Broker

Giao thức truyền thông điệp MQTT theo mô hình publish/subscribe, được sử dụng cho các thiết bị IoT với băng thông thấp, độ tin cậy cao và khả năng được sử dụng trong mạng lưới không ổn định. Nó dựa trên một Broker khá ít tác vụ xử lý và được thiết kế có tính mở, đơn giản và dễ cài đặt. Chính vì có thiết kế đơn giản, kích thước gói tin nhỏ chỉ 2 Bytes cho trường Header và phần còn lại là dung lượng cho lệnh điều khiển nên hoạt động của MQTT gần như không ảnh hưởng tới độ trễ truyền tải video đầu cuối kể cả trong hệ thống mạng băng thông thấp. Như vậy, lựa chọn MQTT sẽ làm tăng hiệu quả trong các thí nghiệm đo độ trễ truyền tải video.

2.1.4. Wowza Streaming Server

Wowza Streaming Server là một phần mềm phân phối video trực tuyến, thời gian thực được phát triển bởi Wowza Media Systems. Nó cho phép người dùng truyền phát video và âm thanh trực tuyến trên Internet thông qua các giao thức như RTMP, RTSP, HLS và MPEG-DASH. Wowza Streaming Server được sử dụng rộng rãi cho các ứng dụng trực tuyến như phát sóng truyền hình trực tuyến, video trực tiếp, hội nghị trực tuyến, trò chơi trực tuyến và các ứng dụng khác liên quan đến video trực tuyến. Nó hỗ trợ nhiều định dạng video và âm thanh, cho phép người dùng tự do tùy chỉnh và thiết lập các cấu hình phù hợp với nhu cầu của họ.

2.2. Phương pháp nghiên cứu và thí nghiệm

2.2.1. Phương pháp nghiên cứu

Trong bài báo này, phương pháp nghiên cứu chủ yếu dựa trên thí nghiệm thực tế, xây dựng hệ thống trạm quan trắc từ các thiết bị có sẵn với phần mềm được lập trình trước. Dựa trên quan sát thực tế và đánh giá sơ bộ các yếu tố có thể ảnh hưởng tới chất lượng truyền phát video đầu cuối trong một số tình huống như xem video streaming thời gian thực, truyền video ở các độ phân giải khác nhau. Độ trễ truyền tải của các giao thức được thể hiện trực tiếp trên màn hình của thiết bị đầu cuối. Bài báo tập trung vào hai nội dung chính là phương pháp xây dựng mô hình thực nghiệm và phương pháp đánh giá hiệu năng của các giao thức truyền video RTMP, RTSP, HLS và DASH. Trong đó, hiệu năng của các giao thức trên được đánh giá chủ yếu dựa trên độ trễ của video nhận được tại máy khách ở các độ phân giải SD, HD và Full HD trong các mô hình truyền dẫn khác nhau, cụ thể:

- Truyền video trực tiếp từ Raspberry Pi lên màn hình
- Truyền video qua mạng Internet như mô tả ở Hình 1

Việc đánh giá hiệu năng của từng giao thức ở các mô hình truyền dẫn và độ phân giải video khác nhau sẽ cung cấp cho các nhà nghiên cứu một cái nhìn tổng quan về công nghệ quan trắc hình ảnh dựa trên IoT, từ đó các nhà nghiên cứu và các nhà sản xuất cũng như người dùng quan tâm có thể lưu ý lựa chọn công nghệ phù hợp cho sản phẩm của mình trên thực tế.

2.2.2. Phương pháp thí nghiệm

a) Mô tả hoạt động của hệ thống

Trạm quan trắc có thể được vận hành theo nhiều cách khác nhau, như khởi tạo và điều khiển từ phần mềm cài trên máy trạm hoặc thiết bị di động, cũng có thể điều khiển thông qua giao diện web. Thí nghiệm trong bài báo này không chú trọng về thiết kế hoàn chỉnh của phần mềm quản lý, mà chỉ tập trung đánh giá độ trễ của video ở các độ phân giải khác nhau, sử dụng các giao thức khác nhau. Vì vậy, các thí nghiệm được thực hiện thông qua các chương trình được thiết kế đơn giản, dựa trên dòng lệnh để khởi tạo toàn bộ hệ thống cũng như ghi lại các bản nhật ký chứa số liệu để phân tích về sau. Quá trình video được khởi tạo và truyền tới thiết bị đầu cuối của người dùng như mô tả trong Hình 1 theo các bước như sau:

(1) Máy khách phát lệnh "bắt đầu video stream" tới MQTT broker trên VPS thông qua mạng Internet;

(2) MQTT broker chuyển tiếp yêu cầu tới Raspberry Pi4;

(3) Raspberry Pi4 khởi động chương trình Libcamera. Chương trình này đọc dữ liệu từ camera và đồng thời phát stream lên Wowza Media Server theo giao thức RTMP push.

(4) Khi Raspberry Pi4 đẩy luồng video lên Wowza Media Server, nó cung cấp thông tin về tên ứng dụng trên Wowza Media Server sẽ tiếp nhận stream, và tên stream. Tên stream sẽ được Wowza Media Server sử dụng để tạo ra đường link phát tương ứng cho mỗi giao thức mà ứng dụng đó hỗ trợ.

(5) Wowza Media Server nhận luồng video đẩy lên bằng giao thức RTMP từ Raspberry Pi4 và xử lý nó để tạo ra các định dạng video, âm thanh và metadata khác phù hợp để phân phối video trực tuyến.

(6) Wowza Media Server tạo ra luồng video phát tương ứng với các giao thức được hỗ trợ (và kích hoạt) của ứng dụng (bao gồm RTMP, RTSP, HLS, DASH) cùng với đường link tương ứng mà máy khách có thể sử dụng để phát stream.

(7) Máy khách sử dụng đường link đã được tạo để phát theo giao thức được lựa chọn.

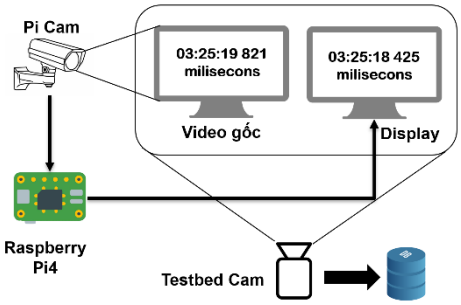
b) Xây dựng các kịch bản thí nghiệm

Trên thực tế, việc so sánh chất lượng truyền video trực tuyến, thời gian thực là một bài toán khó, đặc biệt với sự cải tiến không ngừng của các giao thức truyền và hạ tầng mạng ngày càng mạnh mẽ. Để giảm độ trễ tối đa, các nhà phát triển hạ tầng mạng, dịch vụ và giao thức luôn cố gắng cắt giảm tối đa các dữ liệu hoặc thuật toán dư thừa để đảm bảo người dùng luôn xem được video với chất lượng tốt nhất và độ trễ nhỏ nhất để đảm bảo giá trị chất lượng trải nghiệm (Quality of Experience) cao nhất. Tuy nhiên, do thiết kế cơ bản của mạng máy tính, dữ liệu video vẫn phải được đóng gói trong các gói tin và truyền đi như các ứng dụng khác, và vì vậy nó vẫn bị ảnh hưởng ít nhiều bởi các tham số chất lượng dịch vụ mạng (Quality of Service), đặc biệt là độ trễ. Trong live streaming, độ trễ của hình ảnh là yếu tố ảnh hưởng trực tiếp tới chất lượng trải nghiệm của người dùng. Vậy việc đánh giá đỗ trễ của video là rất cần thiết và cũng là mục tiêu chính của bài báo này.

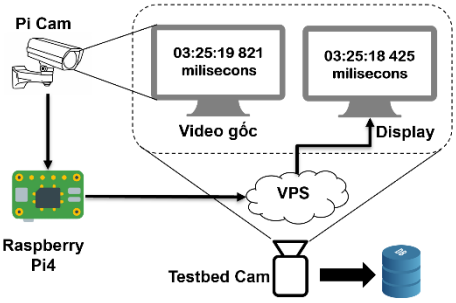
Các kịch bản được trình bày trong phần này được thực hiện để so sánh độ trễ truyền video trực tuyến (live streaming) ở các mô hình truyền dẫn khác nhau và video nguồn cũng được thay đổi các độ phân giải khác nhau. Sự chênh lệch về độ trễ của video cho thấy sự khác biệt, ưu nhược điểm của các giao thức được đánh giá. Để đánh giá độ trễ của video có thể thực hiện bằng nhiều phương pháp khác nhau như VideoLat [9], vDelay [10], and AvCloak [11], tuy nhiên, các phương pháp này áp dụng cho các trường hợp truyền phát và xem video khác nhau. Trong bài báo, video được truyền phát trực tuyến, thời gian thực từ một hệ thống IoT, là dạng video quan trắc môi trường. Để phù hợp với bài toán thực tế, các kịch bản thí nghiệm đánh giá độ trễ của video được thực hiện dựa trên yếu tố quan trắc trực tiếp từ hình ảnh và có tính toán độ trễ dựa trên thời gian thực. Cụ thể, các thí nghiệm được chia thành 2 kịch bản chính, tại mỗi kịch bản, độ phân giải của video nguồn được lần lượt thay đổi từ SD, HD sang Full HD. Điểm chung của các kịch bản là đều có cụm thiết bị quan trắc bao gồm mạch Raspberry Pi4, Camera gắn kèm và 2 màn hình hiển thị. Trong đó, màn hình thứ nhất là một bộ máy tính cá nhân bình thường, chạy một đoạn mã nguồn cho phép hiển thị thông tin thời gian hiện tại tính đến milli giây, Pi Camera sẽ quay trực tiếp mà hình này để truyền hình ảnh đến mạch Raspberry Pi4, như vậy, video nguồn sẽ là cảnh quay trực tiếp màn hình thứ nhất, nội dung chính là thời gian hiện tại tính đến milli giây. Video nguồn này được quay trong thời gian cố định ở tất cả các kịch bản là 30 giây. Màn hình thứ hai là màn hình của máy khách, hiển thị video được phát từ Pi Camera. Trong các kịch bản đều có một Testbed Cam độc lập để chụp cả hai màn hình này theo mỗi 2 giây. Testbed Cam là một thiết bị Webcam được kết nối với một máy tính và được điều khiển chụp tự động thông qua một script được viết bằng ngôn ngữ Python, sử dụng thư viện OpenCV.

Các hình ảnh được chụp từ Testbed Cam sau đó được lưu trữ vào cơ sở dữ liệu để phân tích về sau. Sự khác nhau của các kịch bản nằm ở mô hình truyền dữ liệu. Cụ thể, Hình 4 mô tả cấu hình thí nghiệm cho kịch bản số 1, trong đó màn hình hiển thị video được kết nối trực tiếp tới mạch Raspberry Pi4 thông qua cổng miniHDMI tích hợp trên mạch.

Trong trường hợp này, đỗ trễ của video không phụ thuộc vào hệ thống mạng vì không có gói tin nào được truyền đi. Mạch Raspberry Pi4 được điều khiển trực tiếp từ lệnh trên cửa sổ console của hệ điều hành Raspbian, hoặc được điều khiển thông qua giao thức SSH từ một máy tính khác. Tuy nhiên, các lệnh điều khiển Raspberry Pi4 không ảnh hưởng gì tới quá trình thiết bị này truyền video lên màn hình. Để ghi nhận lại độ trễ của video, Pi Cam quay màn hình video nguồn có nội dung là thời gian hiện tại tính đến milli giây và phát lên màn hình thứ hai. Tại thời điểm Testbed Cam chụp lại, sự chênh lệch giữa 2 hình ảnh thời gian này chính là độ trễ của toàn hệ thống. Tuy nhiên, ở đây không có sự xuất hiện của mạng, nên độ trễ này chủ yếu là độ trễ mã hóa và giải mã video, render video tại mạch Raspberry Pi.



Hình 4. *Kịch bản 1 - Xem video trực tiếp từ hệ thống*



Hình 5. *Kịch bản 2 - Video được truyền qua mạng Internet*

Hình 5 mô tả cấu hình của kịch bản thí nghiệm thứ 2, trong kịch bản này, cụm Raspberry Pi và Pi Cam được giữ nguyên cùng với hai màn hình như các kịch bản khác. Mô hình kết nối của kịch bản này hoàn dựa trên mạng Internet như mô tả ở Hình 1. Trong đó VPS được tích hợp sẵn các phần mềm Wowza Media Server và MQTT broker đóng vai trò là máy chủ trung gian nhận lệnh điều khiển và phân phối video từ Raspberry Pi tới các người dùng khác nhau có kết nối Internet.

Trong kịch bản này, Pi Cam vẫn quay màn hình có chứa nội dung là thời gian hiện tại được tính đến milli giây và phát trực tuyến, thời gian thực lên Wowza Media Server. Máy khách sử dụng một ứng dụng nhỏ được viết bằng ngôn ngữ C# để có thể xem video phát trực tiếp từ hệ thống quan trắc. Từ cấu hình của kịch bản, ta có thể dễ dàng nhận ra độ trễ của video sẽ phần lớn đến từ độ trễ của mạng Internet, thời gian xử lý gói tin đầu cuối, và đặc biệt là thời gian Wowza Media Server thực hiện các tác vụ xử lý video như transcode, transmux, nén, giảm độ phân giải video để đáp ứng yêu cầu của máy khách. Ngoài ra, độ trễ của video cũng bị ảnh hưởng trực tiếp từ độ phân giải của các video nguồn SH, HD và Full HD. Phần tiếp theo của bài báo sẽ trình bày chi tiết kết quả thí nghiệm từ các kịch bản được trình bày như trên.

3. Kết quả nghiên cứu

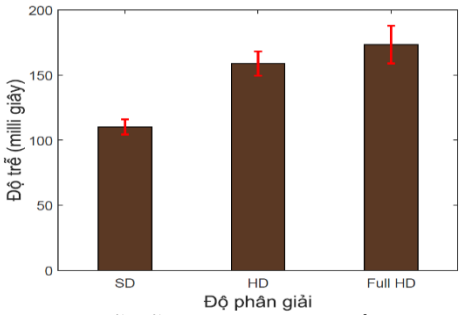
Phần này trình bày các kết quả nghiên cứu chính của bài báo, trong đó các kịch bản thí nghiệm như mô tả ở trên được thực hiện ít nhất 30 lần cho mỗi giao thức ở mỗi độ phân giải khác nhau để đảm bảo ý nghĩa về mặt thống kê. Độ trễ của video là độ trễ trung bình được tính trực tiếp từ epoch time (hay được gọi là thời gian Unix, là hệ thống mô tả một điểm trong thời gian. Thời gian Unix được định nghĩa bằng số giây kể từ 00:00:00 theo giờ Phối hợp Quốc tế ngày 1 tháng 1 năm 1970, trừ đi giây nhuận) hiện lên màn hình tại mỗi lần lấy mẫu, độ trễ của video là hiệu của hai con số hiện lên màn hình nguồn và màn hình của *Máy khách* như mô tả trong các Hình 4, Hình 5, giá trị của hiệu này được tính đến milli giây. Quá trình lấy mẫu được thực hiện

tại các thời điểm khác nhau trong ngày và trong vòng 1 tuần để đảm bảo tính chất ngẫu nhiên của mạng máy tính. Tất cả các thiết bị trong thí nghiệm đều được kết nối với nguồn điện dân dụng và hệ thống mạng của phòng thí nghiệm để đảm bảo hiệu năng hoạt động cao nhất.

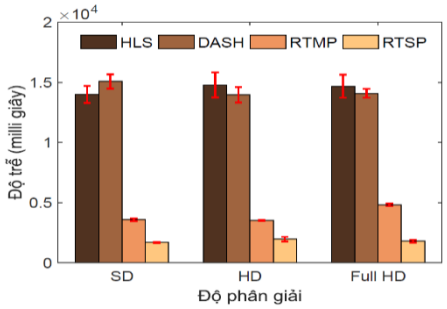
3.1. Kịch bản 1 – Xem video trực tiếp từ hệ thống

Hình 6 biểu thị độ trễ trung bình của video trong kịch bản thí nghiệm đầu tiên, video được truyền trực tiếp từ mạch Raspberry Pi4 lên màn hình. Trong biểu đồ, trục *x* là các biểu đồ cột biểu thị độ trễ trung bình của video ở các độ phân giải SD, HD và Full HD, trục *y* thể hiện độ trễ trung bình của video tính bằng milli giây với khoảng tin cậy 95%, và được thể hiện bằng các error bar trên đỉnh mỗi cột.

Biểu đồ cho thấy độ trễ trung bình của video có xu hướng tăng lên khi truyền video ở các độ phân giải lớn dần. Video có độ phân giải Full HD khi truyền trực tiếp lên màn hình có độ trễ trung bình lớn nhất là 173 milli giây, trong khi giá trị này thấp hơn ở các độ phân giải HD và SD lần lượt là 158 và 110 milli giây. Hiện tượng này cũng dễ hiểu, vì trong kịch bản thí nghiệm này, video được truyền trực tiếp từ mạch Raspberry Pi4 lên màn hình, như vậy không hề có độ trễ của mạng hay các máy chủ trung gian. Độ trễ ở đây về cơ bản chỉ là trễ do mã hóa và giải mã video ở các thiết bị đầu cuối. Độ phân giải video lớn hơn đồng nghĩa với việc kích thước của chúng lớn hơn dẫn tới trễ mã hóa cũng cao hơn. Kết quả thí nghiệm này cho thấy thời gian xử lý video của mạch Raspberry Pi4 là hoàn toàn chấp nhận được, với độ trễ rất nhỏ chỉ 1/5 giây, và kết quả này cũng có thể được xem là giá trị tham chiếu cho các kết quả thí nghiệm khi truyền video thời gian thực qua mạng Internet.



Hình 6. *Đồ trễ của video khi truyền trực tiếp*



Hình 7. *Đồ trễ của video khi truyền qua Internet*

3.2. Kịch bản 2 – Video được truyền tải qua mạng Internet

Hình 7 cho thấy độ trễ trung bình của video được truyền qua mạng Internet theo kịch bản 3 ở các độ phân giải khác nhau và với các giao thức khác nhau. Trong biểu đồ, trục *x* là các nhóm biểu đồ cột biểu thị độ trễ của video ở các độ phân giải SD, HD và Full HD, trục *y* thể hiện độ trễ trung bình của video tính bằng milli giây. Các cột trong biểu đồ với các màu khác nhau từ đậm đến nhạt thể hiện độ trễ trung bình của các giao thức lần lượt là HLS, DASH, RTMP và RTSP với khoảng tin cậy 95%, và được thể hiện bằng các error bar trên đỉnh mỗi cột.

Biểu đồ cho thấy không có nhiều sự khác biệt khi video được truyền ở các độ phân giải khác nhau là SD, HD và Full HD. Hiện tượng này cũng khá dễ hiểu khi mạng Internet ở phòng thí nghiệm có tốc độ tải trung bình khoảng 80 Mbps, với tốc độ như thế này thì gần như không ảnh hưởng nhiều đến việc truyền video ở độ phân giải nào. Trên thực tế, kích thước các gói tin của video ở độ phân giải Full HD cũng không lớn hơn quá nhiều so với các độ phân giải khác và hoàn toàn không thể bị tắc nghẽn ở tốc độ Internet kể trên. Tuy nhiên, biểu đồ vẫn cho thấy độ trễ truyền tải video có xu hướng tăng chậm khi sử dụng giao thức RTMP để truyền.

Sự khác biệt lớn nhất về độ trễ truyền tải video được thể hiện qua việc so sánh các giao thức khác nhau. Biểu đồ cho thấy, độ trễ truyền video trực tuyến, thời gian thực sử dụng nhóm giao thức dựa trên HTTP là HLS và DASH là cao nhất, tiếp theo là giao thức RTMP và RTSP. Độ trễ trung bình khi truyền bằng HLS và DASH vào khoảng 15 giây, trong khi RTMP và RTSP lần

lượt chỉ khoảng 3 và 2 giây. Các kết quả này hoàn toàn tương đồng với nghiên cứu của Wang và đồng tác giả trong [12] cùng kiến trúc thí nghiệm sử dụng Wowza Media Server. Các tác giả ghi nhận trong thí nghiệm thực tế độ trễ của HLS khoảng 12 giây và RTMP khoảng 2 giây.

Giao thức RTMP và RTSP được thiết kế để có độ trễ thấp hơn so với giao thức HLS và DASH vì một số lý do sau, thứ nhất, RTMP và RTSP được thiết kế để truyền phát trực tiếp dữ liệu video và âm thanh từ máy chủ đến máy khách. Điều này cho phép dữ liệu được truyền phát một cách nhanh chóng và hiệu quả hơn so với các giao thức HLS và DASH, nơi dữ liệu phải được tải về máy chủ, lưu trữ và chia nhỏ thành các phân đoạn trước khi được truyền phát, do đó, tạo ra độ trễ cao hơn. Thứ hai, RTMP và RTSP không sử dụng mã hóa bổ sung trong quá trình truyền phát, điều này giúp giảm độ trễ so với HLS và DASH nơi dữ liệu phải được mã hóa và giải mã bổ sung trong quá trình truyền phát. Cuối cùng, dữ liệu video khi truyền bằng các giao thức HLS và DASH từ trạm quan trắc lên máy chủ phải thông qua quá trình mã hóa base64, là phương thức chuyển đổi dạng mã hóa 2 chiều từ *binary* sang *string* để có thể gửi đi được trong mạng Internet một cách dễ dàng. Các *binary* lúc này sẽ được thể hiện bằng các ký tự mã ASCII. Quá trình mã hóa này cũng tạo ra độ trễ đáng kể góp thêm vào tổng độ trễ truyền tải.

Tuy nhiên, giao thức RTMP của Adobe Systems là giao thức độc quyền, yêu cầu một máy chủ phân phối và một kết nối gần như không ngừng giữa máy khách và máy chủ. Yêu cầu một máy chủ phân phối có thể làm tăng giá thành thực hiện và các gói tin RTMP có thể sẽ bị các tường lửa chặn. Hơn nữa, nhóm giao thức RTMP và RTSP yêu cầu phải có phần mềm xem video riêng hoặc phải có Flash tích hợp vào trình duyệt, điều này là không khả thi khi các trình duyệt hiện đại đều không cho phép tích hợp các ứng dụng dựa trên Flash.

Biểu đồ Hình 7 cũng cho thấy giao thức RTMP có độ trễ cao hơn so với RTSP là bởi vì giao thức RTSP được sử dụng chủ yếu cho IPTV và dùng giao thức UDP ở lớp chuyển vận (Transport Layer – TCP/IP) để truyền dữ liệu, và hiệu quả truyền tải tương đối cao khi môi trường mạng ổn định. Trong khi giao thức RTMP chủ yếu được sử dụng để truyền âm thanh và video trên Internet. Nó sử dụng giao thức TCP ở lớp chuyển vận. Do môi trường Internet đôi khi kém ổn định nên việc sử dụng RTMP đảm bảo chất lượng truyền của video, tuy nhiên độ trễ truyền của nó tương đối cao và hiệu quả truyền dẫn thấp hơn nếu so với RTSP.

Dựa vào kết quả đo đạc và phân tích như trên, người dùng có thể tùy tình huống thực tế khi vận hành một trạm quan trắc để quyết định xem nên dùng giao thức nào để truyền video tới các thiết bị đầu cuối. Nếu là một trạm quan trắc có điều khiển hướng quay video, cần phải giảm tối đa độ trễ thì nhóm giao thức RTMP và RTSP là một lựa chọn phù hợp. Tuy nhiên, ở các tình huống sử dụng hoặc mục đích khác thì nhóm giao thức HLS và DASH dựa trên HTTP sẽ thân thiện với các tường lửa hơn và có thể tận dụng được lợi thế của các cơ chế đệm HTTP trên web. Khả năng này còn có thể giảm cả băng thông sử dụng liên quan tới việc chuyển phát video và cả việc nhiều dữ liệu có thể được cung cấp từ các bộ đệm web thay vì từ các máy chủ gốc, và cải thiện chất lượng dịch vụ, do dữ liệu đệm thường gần với người xem hơn nên có thể lấy dễ dàng hơn.

4. Kết luận

Trong bài báo này, tác giả đã giới thiệu một hệ thống quan trắc từ xa, thời gian thực đơn giản bao gồm các thiết bị có sẵn như Raspberry Pi4, Pi Camera và một hệ thống kết nối qua VPS sử dụng giao thức điều khiển MQTT và một phần mềm phân phối video Wowza Streaming Server. Các thành phần này cấu thành một trạm quan trắc cơ bản sử dụng cho các mục đích như giám sát môi trường, quan trắc vùng trồng nông nghiệp và nhiều ứng dụng khác trên thực tế có thể sử dụng. Bài báo cũng trình bày phương pháp thí nghiệm và đánh giá độ trễ truyền tải video trung bình của các giao thức truyền video khác nhau là RTMP, RTSP, HLS và DASH.

Kết quả thí nghiệm cho thấy, thứ nhất độ trễ của quá trình mã hóa, render video tương đối thấp, trung bình chỉ dưới 200 milli giây, điều này đảm bảo video được tạo ra từ máy tính Raspberry Pi4 đảm bảo chất lượng, có độ phân giải cao và độ trễ thấp. Thứ hai, thông qua việc thí nghiệm truyền video bằng các giao thức khác nhau, tác giả nhận thấy nhóm giao thức dựa trên

HTTP là HLS và DASH có độ trễ trung bình tổng thể lớn, nguyên nhân chủ yếu do quá trình mã hóa, tái cấu trúc video thành các độ phân giải khác nhau để phục vụ công nghệ truyền video thích ứng. Trong khi nhóm giao thức RTMP và RTSP có độ trễ trung bình tổng thể nhỏ hơn nhiều, chỉ khoảng 2 đến 3 giây. Nhóm giao thức này thích hợp trong trường hợp người dùng cần điều khiển trạm quan trắc thời gian thực để xoay camera, phóng to hoặc thu nhỏ. Tuy nhiên, nhóm giao thức RTMP và RTSP yêu cầu máy chủ riêng, phần mềm xem video riêng hoặc phải có Flash tích hợp vào trình duyệt, và đặc biệt nhóm giao thức này nhạy cảm với tường lửa trong các hệ thống mạng đầu cuối. Trong trường hợp này, nhóm giao thức dựa trên HTTP tuy độ trễ cao nhưng lại tối ưu hơn về băng thông, sử dụng bộ đệm của trình duyệt web và không bị chặn bởi tường lửa. Nhóm giao thức này phù hợp với các trường hợp sử dụng không cần thời gian thực, như giám sát vùng trồng tự động, thu thập ảnh, video cho các quá trình phân tích về sau.

Hướng nghiên cứu tiếp theo của bài báo tập trung cải tiến nhóm giao thức dựa trên HTTP để giảm độ trễ và tối ưu quá trình truyền video sử dụng hệ thống đã thiết kế như trên.

Lời cảm ơn

Nghiên cứu này được hỗ trợ bởi đề tài nghiên cứu khoa học cấp cơ sở Trường Đại học Công nghệ thông tin và Truyền thông – Đại học Thái Nguyên (Mã số: T2023-07-03).

TÀI LIỆU THAM KHẢO/ REFERENCES

[1] J. Boobalan, V. Jacintha, J. Nagarajan, K. Thangayogesh, and S. Tamilarasu, "An IOT Based Agriculture Monitoring System," *IEEE International Conference On Communication And Signal Processing (ICCSP)*, April, 2018, pp. 0594-0598.

[2] T. Baranwal, and P. K. Pateriya, "Development of IoT Based Smart Security And Monitoring Devices for Agriculture," *IEEE 6th International Conference-Cloud System and Big Data Engineering (Confluence)*, Jan. 2016, pp. 597-602.

[3] K. Abas, K. Obraczka, and L. Miller, "Solar-powered, Wireless Smart Camera Network: An Iot Solution For Outdoor Video Monitoring," *The International Journal for the Computer and Telecommunications Industry*, vol. 118, pp. 217-233, 2018.

[4] C. W. Zhao, J. Jegatheesan, and S. C. Loon. "Exploring IoT Application Using Raspberry Pi." *International Journal of Computer Networks and Applications*, vol. 2, no. 1, pp. 27-34, 2015.

[5] L. Nuñez and R. M. Toasa, "Performance Evaluation of RTMP, RTSP and HLS Protocols for IPTV in Mobile Networks," *IEEE 15th Iberian Conference on Information Systems and Technologies (CISTI)*, June 2020, pp. 1-5.

[6] A. Kaur and S. Singh, "A Survey of Streaming Protocols for Video Transmission," *Proceedings of the International Conference on Data Science, Machine Learning and Artificial Intelligence*, Aug. 2021, pp. 186-191.

[7] S. R. Prathibha, A. Hongal, and M. P. Jyothi, "Iot Based Monitoring System in Smart Agriculture," *IEEE International Conference On Recent Advances In Electronics And Communication Technology (ICRAECT)*, March 2017, pp. 81-84.

[8] N. Suma, S. R. Samson, S. Saranya, G. Shanmugapriya, and R. Subhashri, "IOT Based Smart Agriculture Monitoring System," *International Journal on Recent and Innovation Trends in computing and communication*, vol. 5, no. 2, pp. 177-181, 2017.

[9] J. Jansen, "VideoLat: An Extensible Tool for Multimedia Delay Measurements," *Proceedings of the 22nd ACM International Conference on Multimedia*, Orlando, FL, USA, Nov. 2014, pp. 683–686.

[10] O. Boyaci, A. Forte, S. A. Baset, and H. Schulzrinne, "vDelay: A tool to measure capture-to-display latency and frame rate," *Proceedings of the 11th IEEE International Symposium on Multimedia*, San Diego, CA, USA, 14–16 December, 2009, pp. 194–200.

[11] A. Kryczka, A. Arefin, and K. Nahrstedt, "AvCloak: A tool for black box latency measurements in video conferencing applications," *Proceedings of the 2013 IEEE International Symposium on Multimedia (ISM)*, Anaheim, CA, USA, 9–11 December, 2013, pp. 271–278.

[12] B. Wang, X. Zhang, G. Wang, H. Zheng, and B. Y. Zhao, "Anatomy of a personalized livestreaming system," *Proceedings of the 2016 Internet Measurement Conference*, New York, NY, USA, November, 2016, pp. 485-498.