



Samsung Innovation Campus

| Vạn vật kết nối – IoT

Together for Tomorrow!
Enabling People

Education for Future Generations

Chương 6.

Truyền thông Mạng Và Lập trình Mạng

Vạn vật kết nối – IoT

Mô tả Chương



Mục tiêu của các chương

- ✓ Tìm hiểu về đặc điểm cần có của cơ sở hạ tầng và các thiết bị phần cứng sử dụng kết nối mạng.
- ✓ Tìm hiểu về cấu trúc liên kết mạng và cấu hình mạng theo khoảng cách.
- ✓ Tìm hiểu khái niệm về lập trình socket và hiểu lập trình mạng bằng Raspberry Pi.
- ✓ Có thể lưu trữ dữ liệu đã thu thập bằng cách sử dụng cơ sở dữ liệu và tìm kiếm dữ liệu đã thu thập được.
- ✓ Trong thực tế, có thể sử dụng Raspberry Pi làm máy chủ cơ sở dữ liệu.



Nội dung của các chương

- ✓ Bài 1. Tìm hiểu về Mạng
- ✓ Bài 2. Các giao tiếp
- ✓ Bài 3. Lập trình Socket
- ✓ Bài 4 Hướng dẫn truyền thông dữ liệu qua socket với Raspberry Pi
- ✓ Bài 5. Sử dụng Raspberry Pi làm máy chủ cơ sở dữ liệu

Bài 1.

Tìm hiểu về Mạng

| 1.1. Tổng quan về Mạng

- | 1.2. Thiết bị Mạng
- | 1.3. Mạng chuyển mạch
- | 1.4. Phương thức truyền dữ liệu

| 1.5. Phân loại mạng theo kích thước

- | 1.6. Các cấu trúc liên kết mạng
- | 1.7. Sử dụng Mạng

Tổng quan về Mạng?

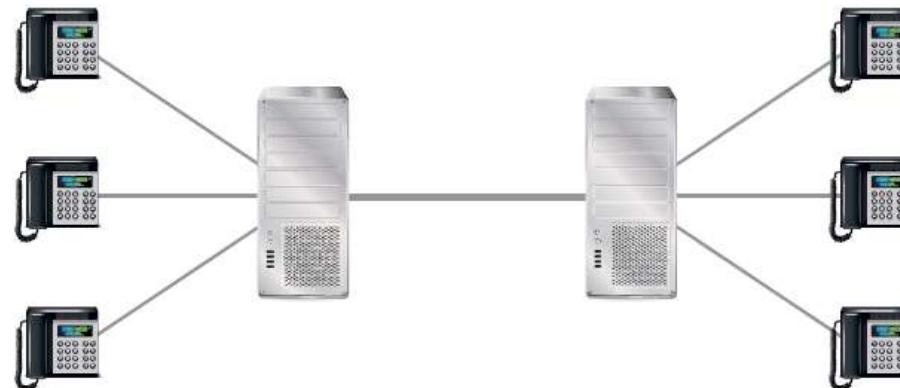
| Một hệ thống các máy tính được kết nối với nhau bằng phương tiện truyền dẫn nhằm trao đổi thông tin.

- ▶ Mạng lưới máy tính
 - Nơi gửi, nơi nhận, phương tiện truyền dẫn
- ▶ phương tiện truyền dẫn
 - Dây nối, không dây
- ▶ giao tiếp
 - Một quy trình hoặc quy tắc xác định đối tượng và cách thức giao tiếp dữ liệu khi truyền dữ liệu giữa hai thực thể giao tiếp với nhau.
 - Các giao tiếp truyền thông cơ bản
 - TCP, IP, UDP, HTTP

Mạng Điện thoại

| Mạng viễn thông đầu tiên dựa trên mã Morse do Samuel Morse phát minh vào năm 1837, và mạng điện thoại do Alexander Bell phát minh vào năm 1876.

- ▶ Về cơ bản, mạng điện thoại truyền các tín hiệu điện dạng tương tự (analog)
 - Nếu khoảng cách truyền xa, tín hiệu sẽ yếu đi và sẽ cần đến một bộ khuếch đại.
 - Các mạng máy tính đời đầu cũng sử dụng các mạng điện thoại để truyền thông dữ liệu.
- ▶ Ban đầu, mạng điện thoại chỉ có thể kết nối một - một đầu cuối với nhau.
 - Khi số lượng điện thoại tăng lên, kết nối một - một trở nên đắt đỏ và không hiệu quả bởi số lượng đường dẫn tăng lên.

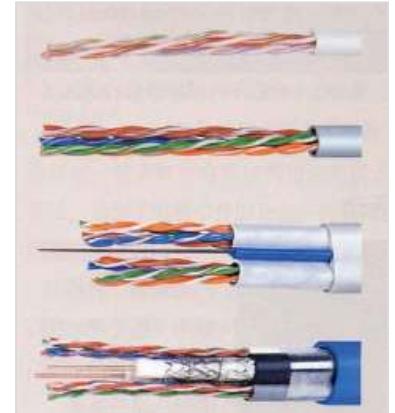


Lịch sử Mạng

- I Lịch sử xây dựng mạng sử dụng máy tính trong một khoảng thời gian không quá dài, vào những năm 1960, một phương pháp thô sơ để kết nối nhiều thiết bị đầu cuối vào một máy tính đã được thử nghiệm.
 - ▶ ARPANET (Mạng lưới cơ quan với các đề án nghiên cứu tiên tiến)
 - Năm 1969, Bộ Quốc phòng Hoa Kỳ đã tạo ra mạng ARPANET để chia sẻ và truyền tải tài nguyên của các dự án trên khắp nước Mỹ.
 - ▶ SNA (Kiến trúc Mạng Hệ thống)
 - Năm 1972, IBM
 - ▶ Ethernet
 - Năm 1974, Xerox
 - ▶ NSFNET
 - Năm 1986, Quỹ Khoa học Quốc gia (NFS) đã kết nối mạng của họ, NFSNET, với mạng ARPANET.
 - ▶ TCP/IP
 - Năm 1982, giao tiếp TCP/IP được tạo và được sử dụng làm giao tiếp mạng Internet.
 - WWW
 - Năm 1992, WWW được phát triển và khiến mạng Internet lan rộng nhanh chóng.

Phương tiện Truyền dẫn

- | Mạng nào cũng cần một phương tiện truyền dẫn để kết nối nơi gửi và nơi nhận.
- | Mỗi phương tiện truyền dẫn đều có một tính chất đặc trưng như băng thông, độ trễ khi truyền dẫn, v.v.
 - ▶ **Băng thông**
 - Dải tần số giữa tần số tối đa và tần số tối thiểu của tín hiệu được truyền qua phương tiện dữ liệu
 - Băng thông rộng hơn giúp truyền nhiều dữ liệu hơn trên một đơn vị thời gian
 - ▶ **Cáp xoắn đôi**
 - Phương tiện truyền dẫn được cấu tạo bởi hai sợi đồng xoắn đôi với nhau và được bọc lớp vỏ nhựa bên ngoài
 - ▶ **Cáp đồng trục**
 - Phần lõi dây dẫn điện ở giữa được bọc một lớp cách điện và một lớp dẫn điện (sợi kim loại, giấy bạc) khác phía ngoài lớp cách điện.
 - Băng thông rộng hơn và việc tốc độ truyền dữ liệu nhanh hơn nhờ băng thông cao hơn cáp xoắn đôi, nhưng đắt hơn.



Cáp xoắn đôi



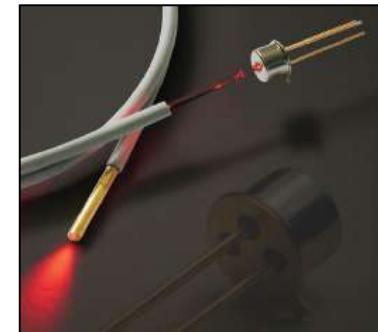
Cáp đồng trục

Các loại phương tiện truyền dẫn

| Sợi quang truyền ánh sáng thông qua các sợi thủy tinh mỏng hơn sợi tóc, cáp quang không truyền dữ liệu bằng tín hiệu điện như ở các loại cáp đồng (cáp xoắn kép).

▶ **Sợi quang**

- Tia sáng được truyền đi nhanh hơn nhiều so với tín hiệu điện
- Băng thông rất cao và ít bị ảnh hưởng bởi các sóng điện từ (nhiều điện tử)
- Độ bảo mật tín hiệu cao hơn nhiều so với các phương tiện truyền dẫn khác vì không bị ảnh hưởng từ các thiết bị liên lạc (hoạt động dựa trên tín hiệu điện)



Optical fiber



Satellite

Bài 1.

Tìm hiểu về Mạng

- | 1.1. Tổng quan về Mạng
- | **1.2. Thiết bị Mạng**
- | 1.3. Mạng chuyển mạch
- | 1.4. Phương thức truyền dữ liệu
- | 1.5. Phân loại theo kích thước
- | 1.6. Các cấu trúc liên kết mạng
- | 1.7. Sử dụng Mạng

Thiết bị truy cập mạng: Modem

| Modem là một thiết bị giúp chuyển tín hiệu số (digital) thành tín hiệu tương tự (analog) và ngược lại.

▶ Modem

- Khi mới ra mắt, nó được sử dụng để kết nối, truyền thông giao tiếp giữa các máy tính với nhau thông qua mạng điện thoại có sẵn.
- Vì mạng điện thoại chỉ truyền được các tín hiệu analog như giọng nói, nên để các máy tính giao tiếp với nhau qua mạng điện thoại thì cần một bộ chuyển đổi để chuyển tín hiệu analog sang tín hiệu digital và ngược lại.

Modem có chức năng như:

- Bộ điều chế
 - Modem chuyển dữ liệu digital của máy tính sang tín hiệu analog.
- Bộ giải điều chế
 - Có chức năng phục hồi tín hiệu analog về tín hiệu digital



Card giao tiếp mạng và thiết bị trung tâm

| Card giao tiếp mạng (NIC) thường được gọi là card LAN và card Ethernet giúp kết nối máy tính với các mạng LAN (mạng máy tính cục bộ).

- ▶ Card giao tiếp mạng (NIC)

- Không giống các modem, kết nối qua đường dây điện thoại và truyền tín hiệu analog, card mạng trực tiếp truyền tín hiệu digital.

- ▶ Thiết bị kết nối Hub

- Truyền dữ liệu từ máy tính này sang máy tính khác

- Thiết bị kết nối trung tâm

- Truyền dữ liệu đến từ máy tính này sang máy tính khác trong một mạng kết nối đơn giản.

- Vì các máy tính sử dụng một băng thông nên khi tăng số lượng máy tính sẽ làm giảm tốc độ truyền dữ liệu trong mạng.

- Thiết bị chuyển mạch trung tâm (switch)

- Ngoài chức năng chuyển tiếp dữ liệu, thiết bị này còn có khả năng chuyển đổi giao tiếp để kết nối giữa các mạng với nhau.

- Hoạt động của mạng sẽ hiệu quả hơn vì băng thông không bị chia ra như cách thức hoạt động của thiết bị hub mà được sử dụng cho một kết nối điểm - điểm.

→ Đắt hơn

Bộ lặp và Cầu nối

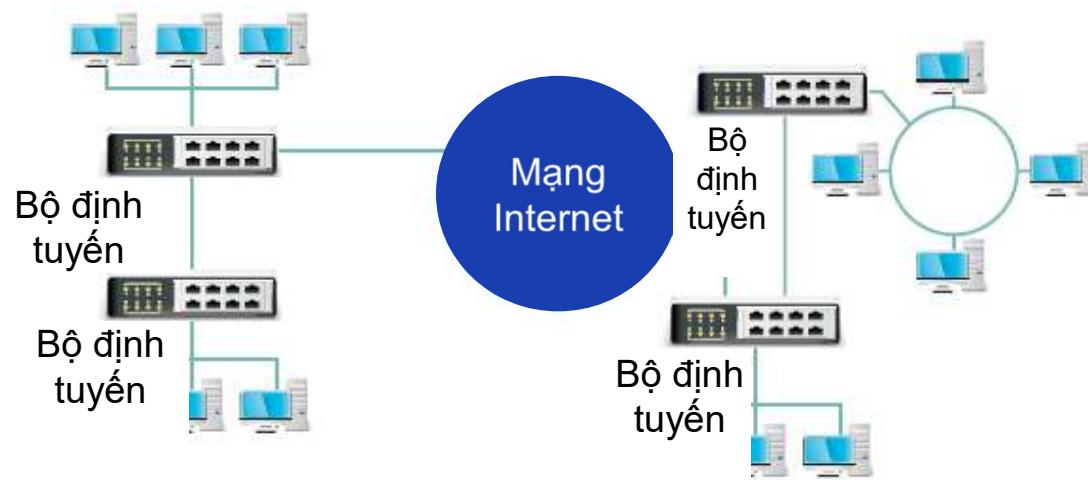
Bộ lặp là một thiết bị có chức năng mở rộng khoảng cách truyền dữ liệu của một mạng, và cầu nối là một thiết bị giúp kết nối hai hoặc nhiều mạng LAN với nhau để tạo thành một mạng thống nhất.

- ▶ Bộ lặp (Repeater)
 - Bộ lặp khuếch đại và tái tạo các tín hiệu yếu trong quá trình truyền tín hiệu qua nó
- ▶ Cầu nối (Bridge)
 - Cầu nối thu nhận và xác định địa chỉ trong một khung (frame) được truyền trong mạng, chấp nhận frame nếu địa chỉ trong khung tương ứng với địa chỉ của mạng LAN đang kết nối với bộ cầu nối, và gửi khung này đến mạng LAN có địa chỉ tương ứng trong khung dữ liệu.
 - Giảm lưu lượng trên toàn mạng.

Bộ định tuyến

| Bộ định tuyến là một thiết bị kết nối các mạng như mạng LAN, MAN, và WAN với nhau.

- ▶ Bộ định tuyến định tuyến (xác định đường đi) các gói dữ liệu theo địa chỉ logic chứa trong gói tin (vd: địa chỉ IP)
- ▶ Bộ định tuyến có trách nhiệm kết nối giữa các mạng với nhau
- ▶ Tính toán định tuyến
 - ▶ Bộ định tuyến tìm kiếm các địa chỉ Internet đích trong gói dữ liệu được truyền đến bộ định tuyến để xác định lộ trình tiếp theo cho gói dữ liệu.
- ▶ Gói dữ liệu được chuyển tiếp qua hai mạng hay nhiều mạng (Mô hình OSI 3 lớp)

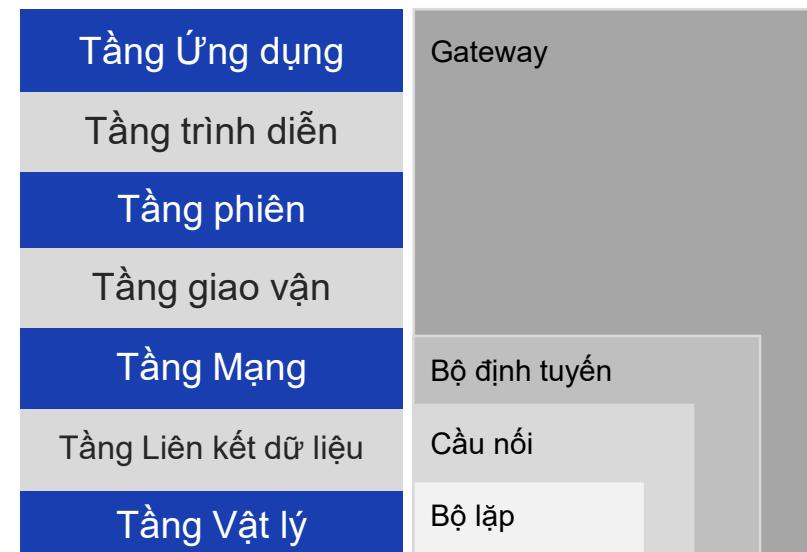


Gateway

| Gateway là một điểm kết nối của một mạng, có vai trò như một cổng vào hoặc cổng ra với một mạng khác.

▶ Gateway

- Có vai trò như một bộ chuyển đổi giao tiếp
 - Kết nối hai mạng có giao tiếp khác nhau
- Ngày nay, gateway và bộ định tuyến được sử dụng kết hợp với nhau.



Mô hình OSI và các thiết bị đi kèm

Bài 1.

Tìm hiểu về Mạng

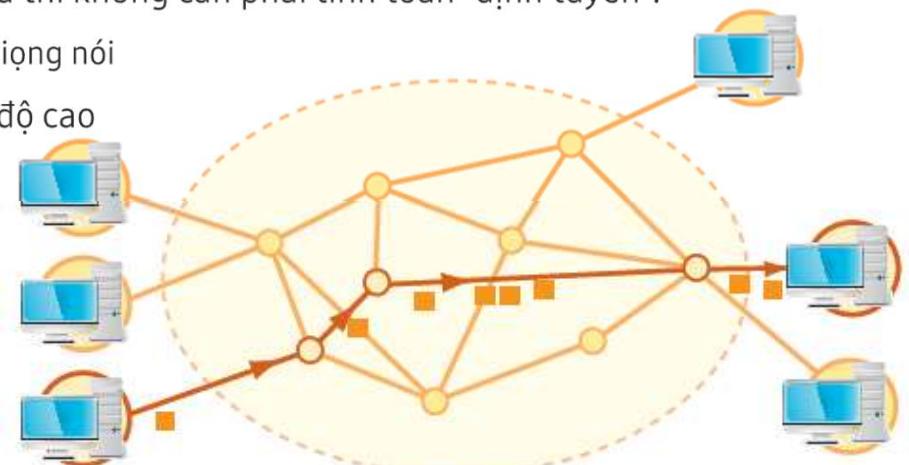
- | 1.1. Tổng quan về Mạng
- | 1.3. Thiết bị Mạng
- | **1.2. Mạng chuyển mạch**
- | 1.4. Phương thức truyền dữ liệu
- | 1.5. Phân loại mạng theo kích thước
- | 1.6. Các cấu trúc liên kết mạng
- | 1.7. Sử dụng Mạng

Mạng chuyển mạch

| Trong mạng chuyển mạch, sau khi xác định bên gửi và bên nhận, một tuyến liên kết thích hợp trong số các đường truyền của mạng sẽ được thiết lập.

- ▶ Chuyển mạch kênh (Circuit Switching)

- Mạch: Tập hợp các đường truyền (các kênh giao tiếp)
- Sau khi thiết lập đường truyền, các máy tính khác sẽ không thể sử dụng được đường truyền này dù không có dữ liệu nào được truyền. Đường truyền này chỉ có thể sử dụng trong một liên kết mới khi nó được giải phóng.
➔ Nhược điểm của chuyển mạch kênh là khả năng tối ưu hóa đường truyền.
- Sau khi thiết lập đường truyền và bắt đầu truyền dữ liệu thì không cần phải tính toán “định tuyến”.
 - Phù hợp với truyền dữ liệu theo thời gian thực, ví dụ như giọng nói
 - Chỉ được sử dụng để truyền khối lượng dữ liệu lớn ở tốc độ cao
- Ví dụ điển hình như mạng điện thoại

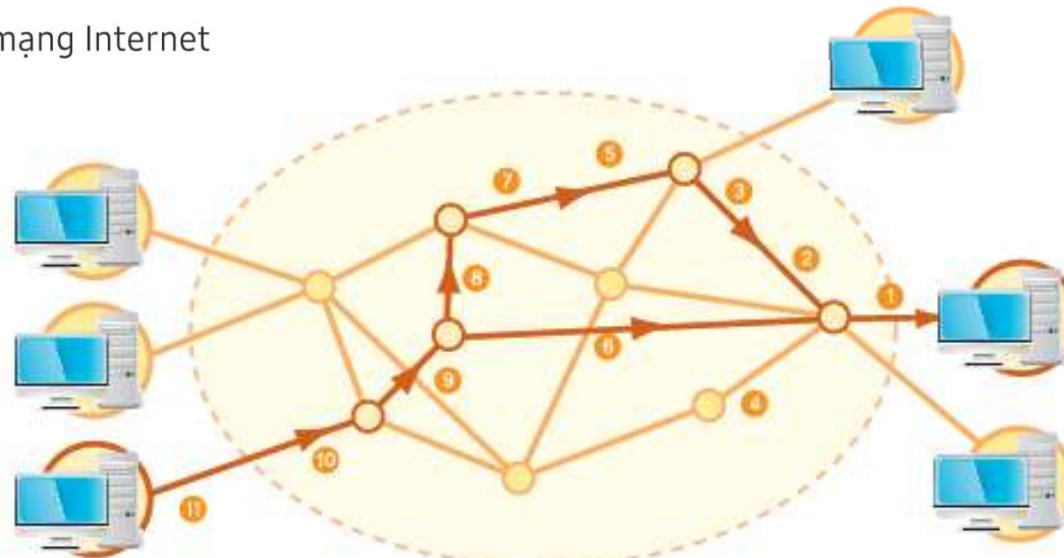


Chuyển mạch gói

| Không thiết lập trước một tuyến liên kết cố định, truyền các gói dữ liệu độc lập với nhau.

▶ Chuyển mạch gói

- Mỗi gói dữ liệu có thể được truyền trên một liên kết khác nhau, tùy theo tình trạng lưu lượng của mạng.
- Không thiết lập một tuyến liên kết cố định giữa phía phát và phía thu
 - Sử dụng đường truyền hiệu quả hơn vì nhiều gói dữ liệu có thể đến các đích khác nhau cùng dùng chung một tuyến liên kết
- Ví dụ điển hình là mạng Internet



Bài 1.

Tìm hiểu về Mạng

- | 1.1. Tổng quan về Mạng
- | 1.2. Thiết bị Mạng
- | 1.3. Mạng chuyển mạch
- | **1.4. Phương thức truyền dữ liệu**
- | 1.5. Phân loại mạng theo kích thước
- | 1.6. Các cấu trúc liên kết mạng
- | 1.7. Sử dụng Mạng

Truyền đơn công, bán song công, song công toàn phần

| Dựa theo hướng và tính đồng thời của luồng dữ liệu, phương thức giao tiếp của mạng được phân chia như sau.

▶ **đơn công**

- Kiểu giao tiếp trong đó dữ liệu được truyền theo một hướng xác định.
- Trong đó, một thiết bị chỉ có chức năng truyền thông tin và thiết bị còn lại chỉ có chức năng nhận thông tin.

▶ **bán song công**

- Hai thiết bị đều có thể giao tiếp theo hai hướng, nhưng không đồng thời.
- Sẽ xảy ra xung đột nếu cả hai thiết bị cùng truyền dữ liệu đồng thời.
 - Để tránh xung đột, trước khi truyền cần kiểm tra xem có thể truyền dữ liệu được không.

▶ **song công toàn phần**

- Hai thiết bị có thể đồng thời truyền dữ liệu theo cả hai hướng
 - Nhận và gửi dữ liệu đồng thời.

Truyền dữ liệu tương tự (analog) và số (digital)

| Dữ liệu tương tự là dữ liệu có giá trị thay đổi liên tục theo thời gian, còn dữ liệu số có các giá trị rời rạc, là bộ số nguyên của giá trị cơ bản (thường là 2).

▶ Truyền dữ liệu tương tự

- Dữ liệu tương tự là thông tin có độ lớn thay đổi liên tục theo thời gian
 - Âm thanh, áp suất, nhiệt độ
- Các tín hiệu tương tự sẽ yếu đi khi khoảng cách truyền dữ liệu tăng lên
 - Để tín hiệu mạnh trở lại, cần phải sử dụng bộ khuếch đại ở các vị trí nhất định

▶ Truyền dữ liệu số

- Truyền các tín hiệu số tương ứng với bít giá trị 0 hoặc 1
- Sử dụng bộ lặp để giải quyết những vấn đề do khoảng cách (suy hao, nhiễu)

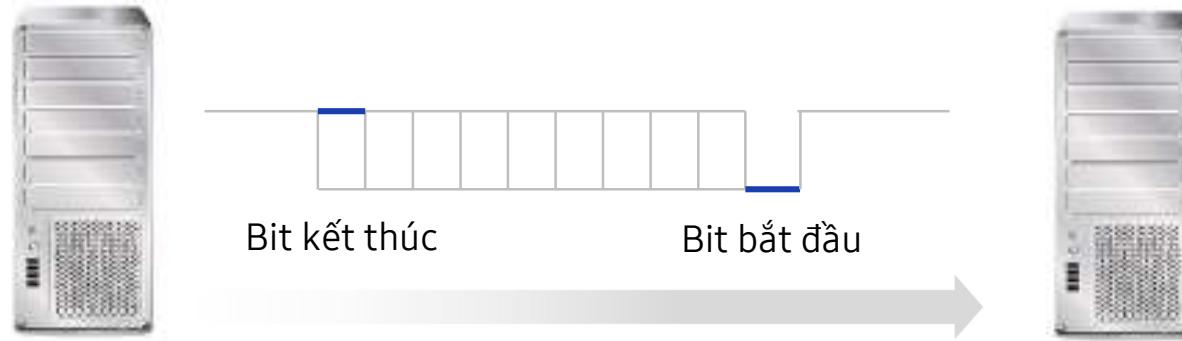
Truyền tín hiệu nối tiếp và song song

- | Truyền tín hiệu nối tiếp là phương thức truyền các bit theo thứ tự và có tính liên tục qua một đường truyền giao tiếp.
- | Truyền dữ liệu song song là phương thức truyền nhiều bit đồng thời thông qua nhiều đường truyền
 - ▶ Truyền dữ liệu nối tiếp
 - Tốc độ truyền chậm vì phải truyền từng bit một tại một thời điểm, tuy nhiên chi phí cho đường truyền rất thấp
 - Chủ yếu được sử dụng trong truyền tải thông tin khoảng cách xa, sử dụng trong các chuẩn giao tiếp như RS-232C và RS-423
 - ▶ Truyền dữ liệu song song
 - So với truyền nối tiếp, truyền song song cho phép truyền dữ liệu nhanh hơn nhưng lại cần chi phí cao hơn để xây dựng đường truyền giao tiếp.
 - Ít được sử dụng để kết nối giữa các thiết bị đầu cuối và thường được dùng để kết nối ở tầm gần như kết nối máy tính với thiết bị ngoại vi (như máy in).

Truyền dữ liệu đồng bộ và bất đồng bộ

| Truyền dữ liệu bất đồng bộ là phương pháp truyền dữ liệu theo từng ký tự, khi dữ liệu được gửi từ bên truyền đến bên nhận không đòi hỏi dùng chung nhịp thời gian (xung nhịp).

- ▶ Truyền dữ liệu bất đồng bộ
 - Trong phương pháp truyền dữ liệu bất đồng bộ, tín hiệu dữ liệu được phân biệt bằng bit bắt đầu (bit start), các bit dữ liệu và bit kết thúc (bit stop)
 - Bên truyền và bên nhận sử dụng các xung đồng hồ độc lập nhưng chúng được “đồng bộ” thông qua các bit bắt đầu và kết thúc để nhận diện dữ liệu.
 - Truyền dữ liệu bất đồng bộ có chi phí truyền tải thấp vì cấu trúc của các thiết bị truy cập đơn giản.
 - Dữ liệu càng dài sẽ càng làm tăng nguy cơ xuất hiện các lỗi do cộng dồn khi có sự sai lệch về xung đồng hồ (xung nhịp) giữa các phía.



| Truyền dữ liệu đồng bộ là phương pháp truyền trong đó thông tin xung nhịp được khôi phục ở phía thu để sử dụng trong quá trình thu nhận dữ liệu, tức là 2 phía có chung nguồn xung nhịp.

▶ Truyền dữ liệu đồng bộ

- Để cải thiện hiệu suất truyền dữ liệu, bên truyền và bên nhận truyền dữ liệu theo một định dạng dữ liệu nhất định mà hai bên đã thống nhất.
 - Bên gửi sẽ chèn và truyền các thông tin điều khiển trước và sau khi truyền khối dữ liệu.
- Trong truyền dữ liệu đồng bộ, việc khôi phục lỗi (sửa lỗi) khi xảy ra bằng cách truyền lại dữ liệu.



Bài 1.

Tìm hiểu về Mạng

- | 1.1. Tổng quan về Mạng
- | 1.2. Thiết bị Mạng
- | 1.3. Mạng chuyển mạch
- | 1.4. Phương thức truyền dữ liệu
- | 1.5. Phân loại mạng dựa trên kích thước
- | 1.6. Các cấu trúc liên kết mạng
- | 1.7. Sử dụng Mạng

Mạng LAN

| Mạng LAN là một mạng kết nối giữa các thiết bị trong phạm vi tương đối gần.

▶ Mạng LAN (mạng cục bộ)

- Một mạng giúp kết nối giữa các máy tính, máy in và các thiết bị mạng khác trong phạm vi gần
 - Triển khai trên một khu vực do một tổ chức quản lý
 - Ví dụ: một mạng được cài đặt trong khuôn viên trường, tòa nhà công ty, nhà máy, v.v.
- Ban đầu, một chuẩn mạng LAN máy tính, hay còn gọi là mạng Ethernet có tốc độ truyền dữ liệu từ 10 đến 100 Mbps
- Gần đây, một số chuẩn mạng LAN tốc độ cao đã được triển khai như mạng ethernet Gigabit, ATM, FDDI và mạng LAN không dây.



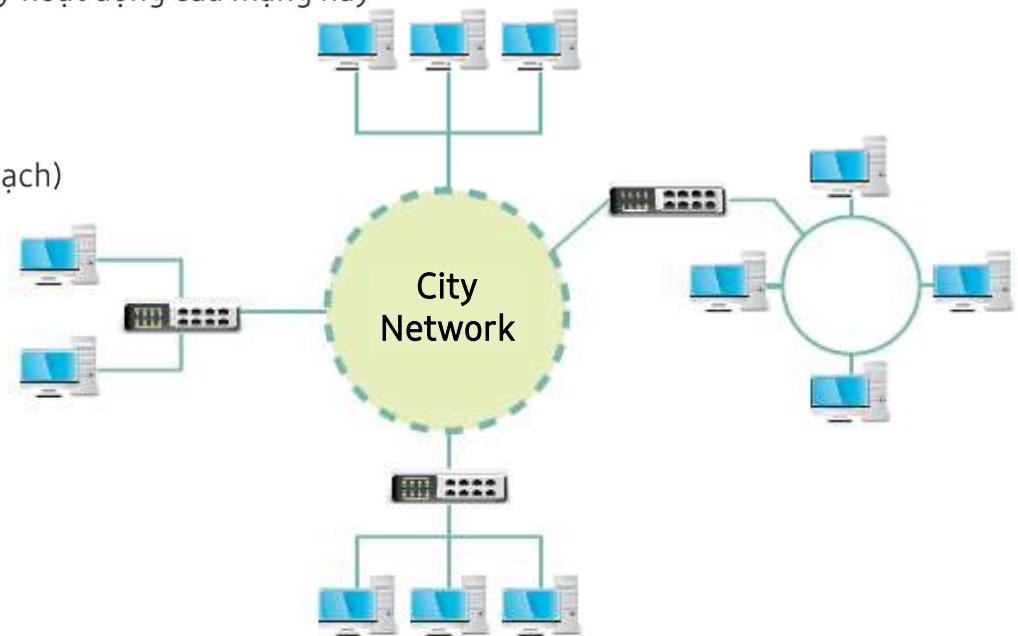
I Mạng LAN có chất lượng giao tiếp tương đối tốt và tốc độ nhanh

- ▶ Tính năng
 - Truyền dữ liệu hiếm khi bị chậm. Sử dụng và quản lý mạch giao tiếp chất lượng tốt
 - Chất lượng giao tiếp tốt. Ít xảy ra lỗi khi truyền. Tốc độ truyền nhanh
 - Các thiết bị như máy tính và máy in có thể được kết nối và sử dụng dễ dàng. Dễ mở rộng.
- ▶ Chuẩn mạng LAN
 - Ethernet
 - Phát triển bởi Xerox vào năm 1976
 - tốc độ 10 Mbps, thuật toán CSMA/CD
 - Mạng Ethernet tốc độ cao
 - 100Mbps, 100BASE-T
 - Mạng Ethernet Gigabit
 - 1 Gbps
 - FDDI (Giao diện dữ liệu phân tán sợi quang)
 - Mạng này thường dùng sợi quang như một phương tiện truyền dữ liệu để triển khai mạng LAN tốc độ cao
 - Mạng này thường được dùng ở các mạng xương sống (mạng đường trực) cần băng thông rộng và tốc độ truyền tải nhanh

Mạng MAN

Mạng MAN (Mạng đô thị) là một mạng rộng hơn mạng LAN một chút

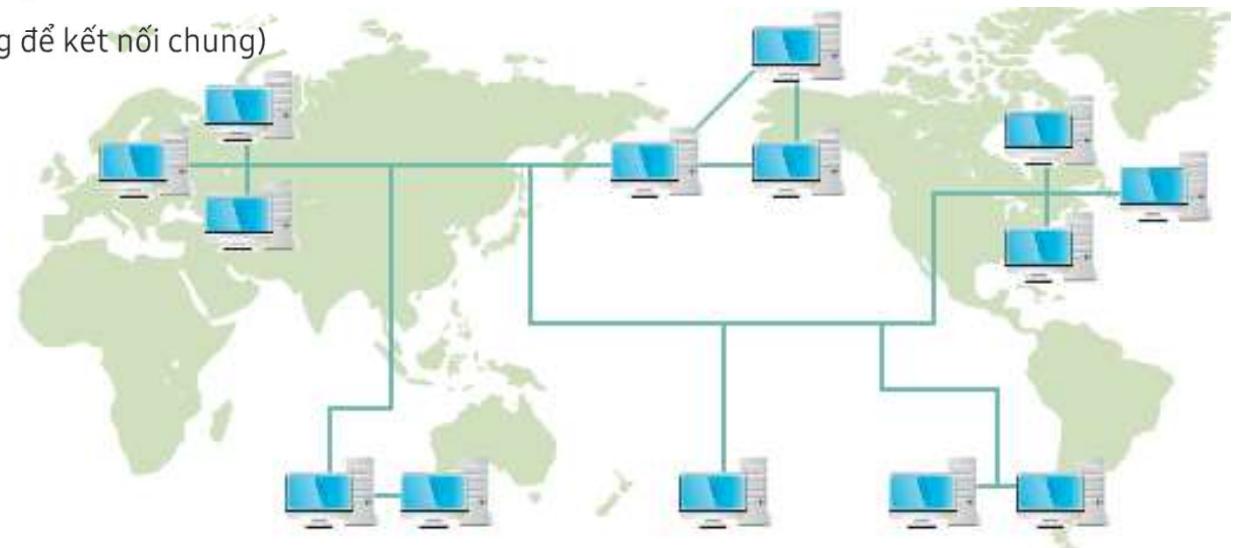
- ▶ MAN (Mạng đô thị)
 - Một mạng giúp kết nối trong phạm vi một ngôi làng hoặc một thành phố, một mạng mở rộng hoặc kết nối giữa các mạng LAN với nhau.
 - Các nhà mạng thường cung cấp và quản lý hoạt động của mạng này
- ▶ Mạng điện thoại là một ví dụ của mạng MAN
 - Cung cấp dịch vụ SMDS
(dịch vụ số liệu tốc độ megabit được chuyển mạch)



Mạng WAN

WAN là một mạng kết nối các quốc gia với nhau

- ▶ WAN (Mạng diện rộng)
 - Một mạng có phạm vi rất rộng
 - Ví dụ về mạng WAN là mạng Internet kết nối toàn thế giới
 - Nói chung, mạng WAN có phạm vi kết nối rộng hơn mạng LAN
 - Cần rất nhiều tiền để xây dựng mạng này.
 - Tốc độ chậm hơn mạng LAN (do sử dụng để kết nối chung)



Bài 1.

Tìm hiểu về Mạng

- | 1.1. Tổng quan về Mạng
- | 1.2. Thiết bị Mạng
- | 1.3. Chuyển đổi Mạng
- | 1.4. Phương thức truyền dữ liệu
- | 1.5. Phân loại mạng dựa trên kích cỡ
- | **1.6. Cấu trúc liên kết mạng**
- | 1.7. Sử dụng Mạng

Cấu trúc liên kết Mạng

| Cấu trúc liên kết mạng là sự sắp xếp vật lý hoặc logic của các nút và đường dẫn kết nối tới mạng.

- ▶ Cấu trúc liên kết mạng
 - Nút (nút mạng)
 - Một thiết bị giao tiếp có địa chỉ xác định, được kết nối với một mạng máy tính.
 - được gọi là một máy trạm (host).
 - Mỗi máy tính, bộ định tuyến, máy in, v.v., là một nút.
- ▶ Các dạng
 - Dạng tuyến (dạng bus)
 - Dạng sao
 - Dạng vòng
 - Dạng cây
 - Dạng lưới

Dạng Bus

| Trong dạng tuyến, tất cả nút mạng đều được kết nối thông qua một đường dây hay đường truyền chung, gọi là một tuyến (bus).

- ▶ Dạng tuyến

- Cấu trúc này khá đơn giản và chi phí cho dây cáp rất nhỏ.
- Nếu một nút truyền dữ liệu khi một nút khác cũng đang truyền dữ liệu thì sẽ xảy ra xung đột.
 - Nếu xảy ra xung đột, thì sau đó sẽ phải truyền lại dữ liệu. Nếu có nhiều nút hoặc lưu lượng dữ liệu cao, xung đột sẽ xảy ra thường xuyên và hiệu suất của mạng sẽ giảm.



Dạng sao

- ▶ Các nút mạng kết nối đến một thiết bị trung tâm bằng liên kết điểm - điểm
- ▶ Khả năng xử lý và độ tin cậy của mạng con phụ thuộc vào thiết bị trung tâm này
 - ▶ Các thiết bị trung tâm thông minh có thể kiểm soát lượng giao tiếp trong mạng hoặc ngăn xung đột
- ▶ Cấu trúc tập trung giúp phát hiện lỗi đơn giản, bảo trì dễ dàng và kiểm soát đường truyền đơn giản hơn

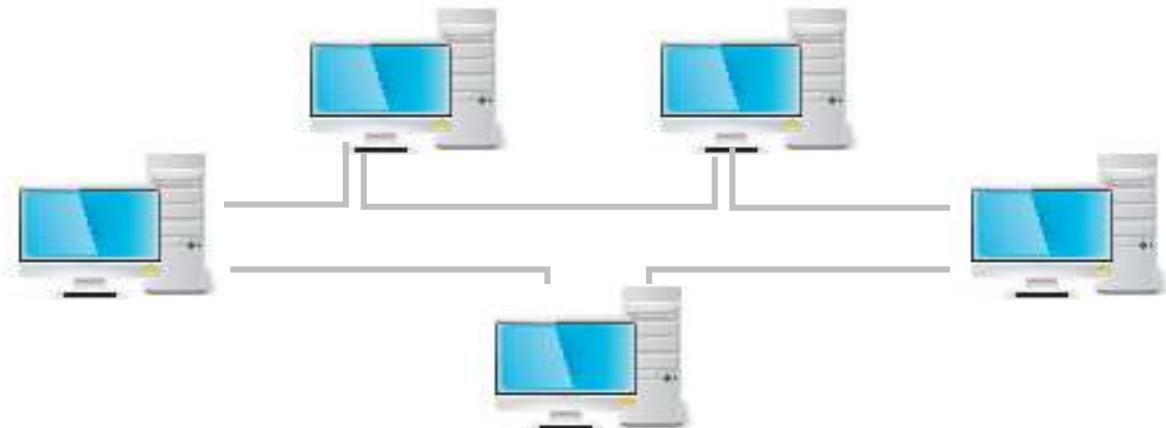


Dạng vòng

| Trong liên kết vòng, các nút của mạng được kết nối với nhau trong một vòng tròn.

▶ Dạng vòng

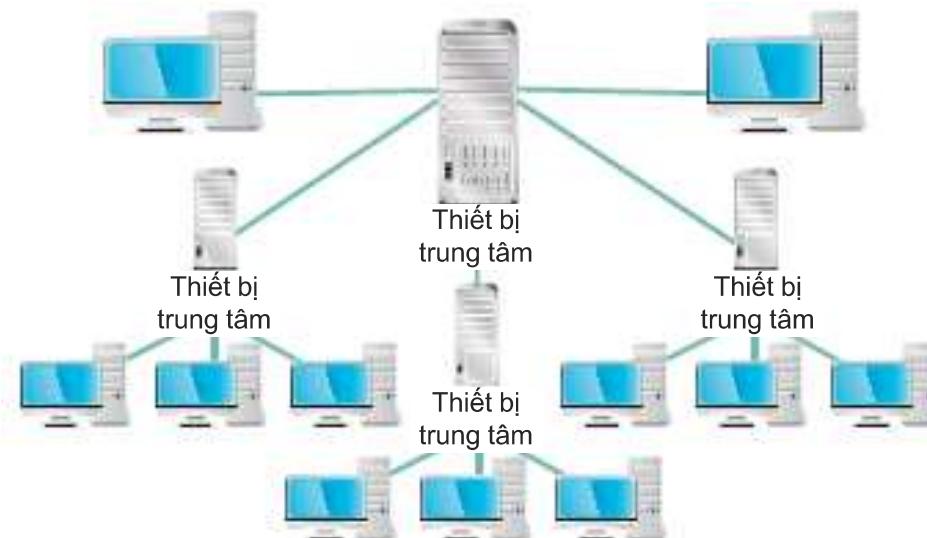
- Các dữ liệu được truyền từ một nút sẽ được truyền liên tục theo vòng tròn.
- Các nút, ngoài các nút nhận, luôn phát tín hiệu và truyền dữ liệu đến nút tiếp theo.
 - Quá trình phát lại có thể làm giảm lỗi trong quá trình truyền dữ liệu.
- Khi một lỗi xảy ra, việc tìm máy chủ bị lỗi sẽ rất dễ dàng.
- Việc số lượng các nút tăng lên không có ảnh hưởng lớn đến hiệu suất của mạng.
- Kiểu
 - Vòng đơn
 - Vòng đôi



Dạng cây

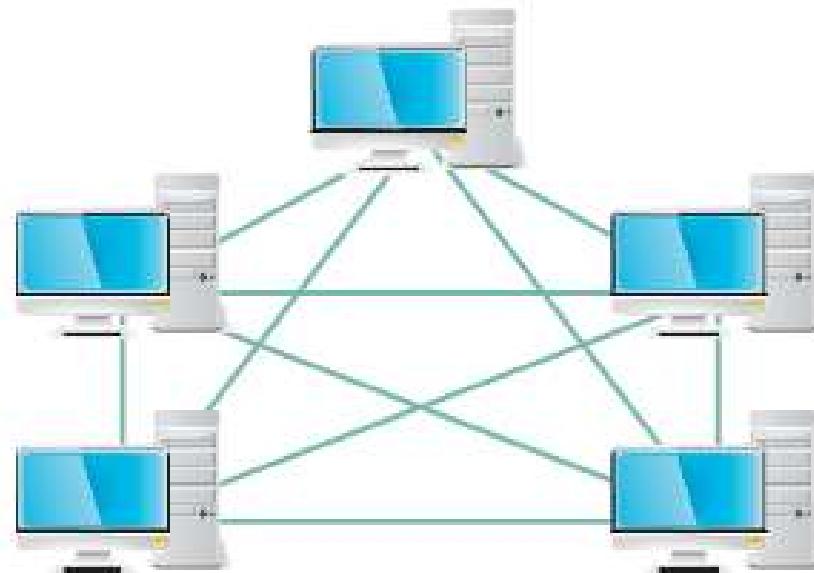
| Liên kết cây là một cấu trúc mà ở đó các nút được kết nối theo hình cây, còn kết nối dạng lưới là cấu trúc mà ở đó các nút đều được kết nối điểm - điểm trực tiếp với nhau mà không cần một nút điều khiển trung tâm.

- ▶ Dạng cây
 - Nút trên cùng của cấu trúc cây là nơi thiết bị trung tâm được đặt và kiểm soát các nút khác ở nhánh dưới.
 - Vì việc kiểm soát đơn giản nên kiểu mạng này rất dễ quản lý và mở rộng
 - Lưu lượng tập trung có thể gây tắc nghẽn, nếu điểm trung tâm bị lỗi, toàn bộ mạng sẽ bị lỗi theo



Dạng lưới

- | Hiệu quả mạng tốt vì không cần chuyển tiếp thiết bị nào trong quá trình truyền dữ liệu.
 - ▶ thậm chí nếu đường truyền bị lỗi, dữ liệu có thể được truyền đi thông qua các con đường khác.
- | Được sử dụng trong các mạng có chi phí cao nhưng có độ tin cậy tốt vì mạng này khá phức tạp và cần nhiều đường truyền



Bài 1.

Tìm hiểu về Mạng

- | 1.1. Tổng quan về Mạng
- | 1.2. Thiết bị Mạng
- | 1.3. Chuyển đổi Mạng
- | 1.4. Phương thức truyền dữ liệu của
Mạng

- | 1.5. Phân loại mạng dựa trên kích cỡ
- | 1.6. Cấu trúc liên kết mạng
- | **1.7. Sử dụng Mạng**

Địa chỉ IP

| Tất cả máy tính và thiết bị mạng kết nối mạng Internet phải có một địa chỉ IP riêng

▶ Địa chỉ IP

- Địa chỉ IP duy nhất

- Là địa chỉ IP của một máy tính mà không có máy tính nối mạng Internet nào khác trên thế giới có cùng địa chỉ IP đó.

- Tất cả bốn chữ số $8 * 4 = 32$ bit

- Cấu trúc

- Trong bốn chữ số, hai hoặc ba chữ số đầu tiên là địa chỉ của mạng.

- Số còn lại là địa chỉ của máy tính trong mạng đó.

- Chẳng hạn với địa chỉ 147.46.X.X, thì địa chỉ của mạng là 147.46.0.0 và mạng này có thể chứa khoảng 216 máy tính.

- Nếu địa chỉ của một công ty là 203.246.245.X, thì địa chỉ mạng của công ty đó sẽ là 203.246.245.0 và mạng này có thể chứa khoảng 28 máy tính.

Mạng con

| Mạng con là một đơn vị mạng nhỏ trong một mạng lớn

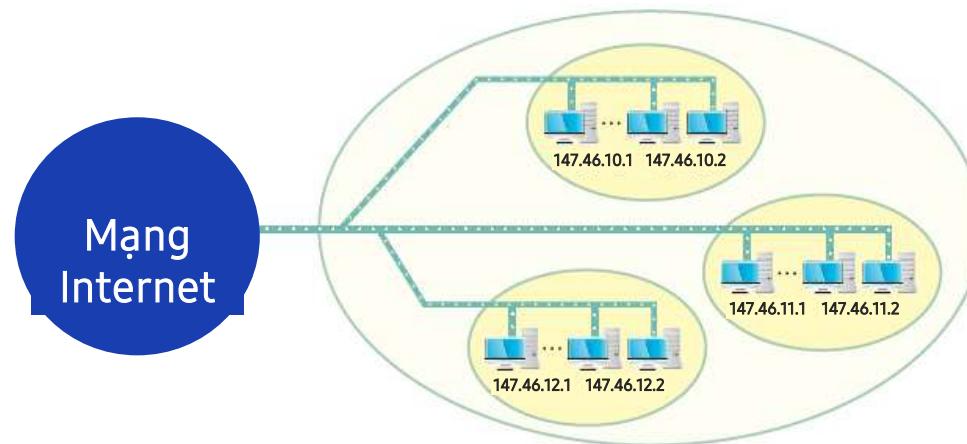
▶ Mạng con

- Mạng có tên 147.46.X.X

- 147.46.10.X, 147.46.11.X... Giống như địa chỉ ở bên trái, nó có thể được chia thành nhiều mạng nhỏ hơn, gọi là mạng con.

▶ Mặt nạ mạng con

- Nếu đặt mặt nạ mạng con là 255.255.255.0 (giá trị nhị phân tương ứng là 11111111 11111111 11111111 00000000), khi một gói dữ liệu đi vào mạng, nó sẽ đi vào mạng con có địa chỉ được xác định theo các bit 1 trong mặt nạ mạng.



Gateway mặc định và máy chủ DNS

| Các cài đặt Gateway mặc định và máy chủ DNS là các thông tin quan trọng khi cấu hình IP

▶ Gateway mặc định

- Một điểm kết nối mà các gói dữ liệu cần đi qua khi một gói dữ liệu từ một máy tính bên trong mạng đi tới một mạng bên ngoài hoặc một gói dữ liệu từ mạng bên ngoài được gửi vào mạng riêng hoặc mạng con bên trong.
 - Tại văn phòng và trường học, địa chỉ gateway sẽ dựa trên vị trí hiện tại của máy tính.

▶ Máy chủ DNS

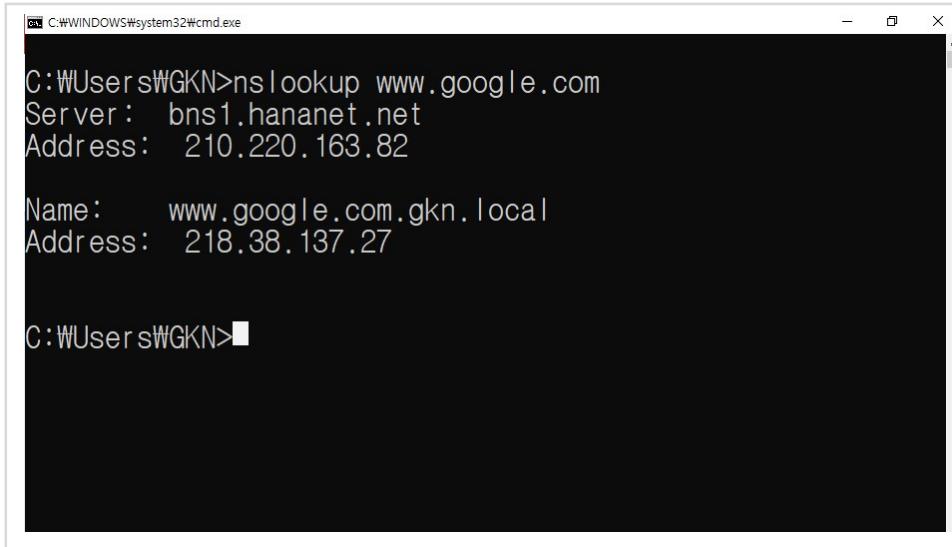
- Máy chủ này chuyển đổi tên miền thành các địa chỉ IP.
 - Máy chủ có chức năng chuyển tên miền ví dụ như www.google.com thành các địa chỉ IP ví dụ như 172.217.24.68
- Tên miền
 - Địa chỉ IP sử dụng các tên miền bằng các ký tự chữ cái, ví dụ như www.naver.com.
 - Lý do là vì các địa chỉ IP dạng số khá khó nhớ và khó sử dụng.

Kiểm tra trạng thái mạng

- | Khi gõ “cmd” trong [Start]-[Run], một cửa sổ sẽ hiện ra để bạn có thể kiểm tra trạng thái hoặc cài đặt với các lệnh ipconfig, nslookup, tracert, v.v.
 - ▶ ipconfig
 - ▶ Liệt kê trạng thái kết nối của mạng hiện tại
 - ▶ Địa chỉ vật lý, địa chỉ IP, máy chủ DHCP, v.v.

| Chuyển tên miền thành địa chỉ IP

- ▶ nslookup
 - Chuyển tên miền bằng chữ cái sang địa chỉ IP



```
C:\Users\GKN>nslookup www.google.com
Server: bns1.hananet.net
Address: 210.220.163.82

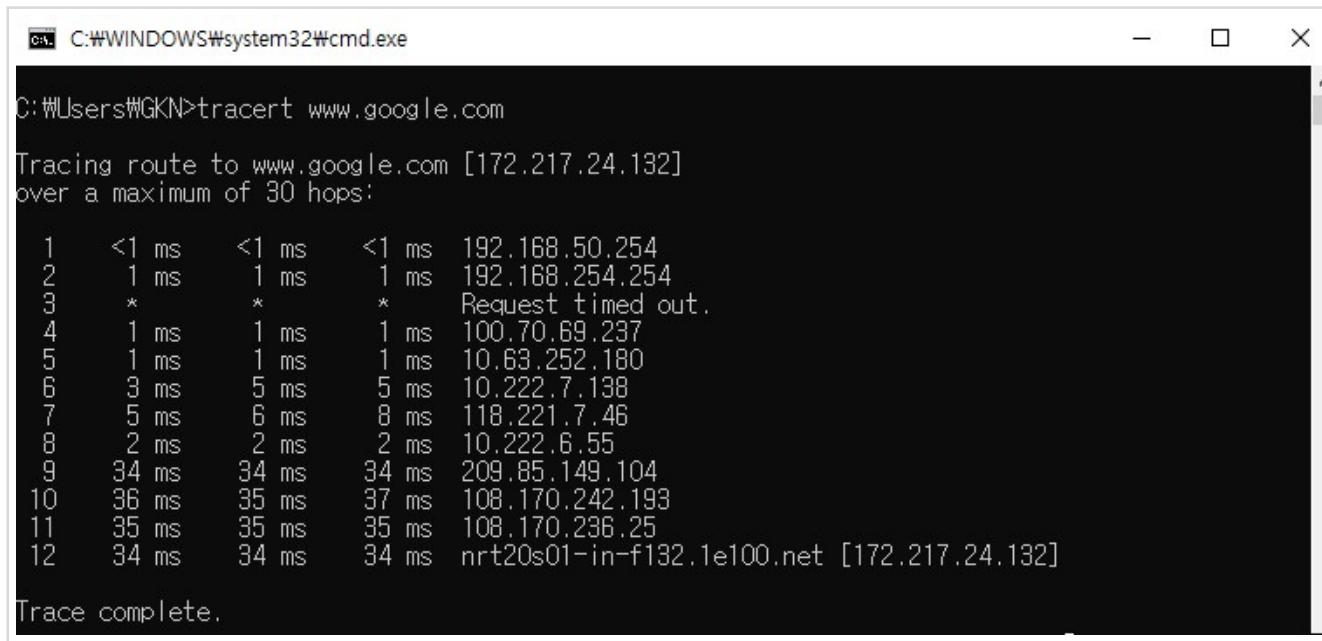
Name: www.google.com.gkn.local
Address: 218.38.137.27

C:\Users\GKN>
```

I Lệnh hiển thị đường dẫn mạng

▶ tracert

- Định tuyến đến một điểm đích cho trước (www.google.com)
- Không tác dụng với các đường dẫn thứ 3, 9 và 10, v.v.
- Ngày nay, vì lý do bảo mật, các dấu vết này thường bị chặn trên mạng của bạn, do đó, không phải lúc nào cũng có thể hiển thị thông tin của định tuyến được.



The screenshot shows a Windows Command Prompt window titled 'cmd.exe' with the path 'C:\WINDOWS\system32\cmd.exe'. The command entered is 'C:\Users\GKN>tracert www.google.com'. The output displays the tracing route to www.google.com over 12 hops, with the last hop being 'nrt20s01-in-f132.1e100.net [172.217.24.132]'. The first three hops are local (1-3), followed by several intermediate routers (4-11), and finally the destination (12).

```
C:\Users\GKN>tracert www.google.com

Tracing route to www.google.com [172.217.24.132]
over a maximum of 30 hops:

 1  <1 ms    <1 ms    <1 ms  192.168.50.254
 2  1 ms     1 ms     1 ms  192.168.254.254
 3  *         *         *      Request timed out.
 4  1 ms     1 ms     1 ms  100.70.69.237
 5  1 ms     1 ms     1 ms  10.63.252.180
 6  3 ms     5 ms     5 ms  10.222.7.138
 7  5 ms     6 ms     8 ms  118.221.7.46
 8  2 ms     2 ms     2 ms  10.222.6.55
 9  34 ms    34 ms    34 ms  209.85.149.104
10  36 ms    35 ms    37 ms  108.170.242.193
11  35 ms    35 ms    35 ms  108.170.236.25
12  34 ms    34 ms    34 ms  nrt20s01-in-f132.1e100.net [172.217.24.132]

Trace complete.
```

Bài 2.

giao tiếp truyền thông

- | 2.1. giao tiếp là gì?
- | 2.2. Mô hình OSI
- | 2.3. giao tiếp truyền thông TCP/IP
- | 2.4. HTTP
- | 2.5. MQTT
- | 2.6. CoAP
- | 2.7. WebSocket

giao tiếp

- | Vì có nhiều kiểu nền tảng và hệ thống trong một mạng, nên các nền tảng và hệ thống này sử dụng giao tiếp quy ước chung để giao tiếp với nhau.
- | Các giao tiếp mạng cơ bản:
 - ▶ giao tiếp điều khiển lớp vận chuyển (TCP), giao tiếp Internet (IP), giao tiếp biên dịch địa chỉ mạng (ARP), giao tiếp cấu hình máy chủ (DCHP)
 - ▶ Một tập hợp logic của các giao tiếp hoạt động như một chồng giao tiếp.
- | Cách tốt nhất để hiểu về giao tiếp:
 - ▶ Hãy nghĩ nó giống như các quy định trong giao tiếp bằng ngôn ngữ của con người.
 - Ex** (động từ, biểu thức kiểm toán, phương pháp định tuyến, phương thức kết nối ban đầu, v.v.,)
- | Tùy theo chức năng, các giao tiếp có thể dễ hiểu hoặc khó hiểu.

| Hầu hết các giao tiếp đều xét đến các vấn đề sau:

- ▶ Khởi tạo kết nối
 - Máy khách hay máy chủ khởi tạo kết nối, cần trao đổi thông tin gì trước khi giao tiếp?
- ▶ Tư vấn về các đặc tính kết nối
 - Làm thế nào để mã hóa giao tiếp của giao tiếp, và làm thế nào để truyền các khóa mã hóa bảo mật giữa các máy tính khi giao tiếp với nhau?
- ▶ Định dạng dữ liệu
 - Làm thế nào để sắp xếp dữ liệu trong gói tin, và kiểu sắp xếp nào được sử dụng trong xử lý dữ liệu ở phía thu nhận?
- ▶ Phát hiện lỗi và sửa lỗi
 - Chuyện gì sẽ xảy ra nếu một gói tin mất quá nhiều thời gian để đến được đích? Máy khách sẽ khôi phục như thế nào nếu không thiết lập được kết nối giữa các máy tính trong một khoảng thời gian ngắn?
- ▶ Hủy bỏ kết nối
 - Làm thế nào để một máy tính thông báo cho máy tính khác rằng kết nối đã bị hủy, và thông tin nào phải được gửi cuối cùng để hủy kết nối đúng cách?

Bài 2.

giao tiếp truyền thông

- | 2.1. giao tiếp là gì
- | 2.2. Mô hình OSI
- | 2.3. giao tiếp truyền thông TCP/IP
- | 2.4. HTTP
- | 2.5. MQTT
- | 2.6. CoAP
- | 2.7. WebSocket

Mô hình OSI?

| Mô hình tiêu chuẩn quốc tế do Tổ chức Tiêu chuẩn Quốc tế (ISO) tạo ra

- ▶ Mô hình kết nối các hệ thống mở (OSI)
 - Năm 1978, Tổ chức Tiêu chuẩn Quốc tế (ISO) đã thiết lập mô hình tiêu chuẩn quốc tế OSI để cho phép hai hệ thống giao tiếp mà không cần quan tâm đến các cấu trúc bên trong.
 - Đây là một mô hình giúp tìm hiểu và thiết kế một cấu trúc mạng có thể tương tác an toàn.
- ▶ 7 tầng
 - Tầng vật lý (tầng 1), Tầng liên kết dữ liệu (tầng 2), Tầng mạng (tầng 3), Tầng vận chuyển (tầng 4), Tầng phiên (tầng 5), Tầng trình diễn (tầng 6), Tầng ứng dụng (tầng 7)
 - Mỗi tầng này độc lập với nhau.
 - Thay đổi trong một tầng/lớp không ảnh hưởng tới các tầng/lớp khác.
 - Các thiết bị mạng có thể giao tiếp bình thường nếu chúng đáp ứng được một số tầng trong bảy tầng này theo chức năng và chuẩn hóa chúng.

Tầng 7	Ứng dụng
Tầng 6	Trình diễn
Tầng 5	Phiên
Tầng 4	Vận chuyển
Tầng 3	Mạng
Tầng 2	Liên kết dữ liệu
Tầng 1	Vật lý

I Mô hình OSI 7 Tầng

- ▶ Từ tầng ứng dụng đến tầng vật lý
 - Ở phía phát, dữ liệu được truyền từ tầng ứng dụng xuống tầng vật lý, còn ở hệ thống phía thu nhận, dữ liệu sẽ được chuyển tuần tự từ tầng vật lý lên đến tầng ứng dụng.
- ▶ Tính độc lập của các Dịch vụ giao tiếp
 - Nếu một tầng giao tiếp cung cấp một dịch vụ nào đó, các tầng khác sẽ không cung cấp dịch vụ đó.
- ▶ Giao tiếp giữa hai tầng liền kề nhau
 - Mỗi tầng giao tiếp với tầng ngay bên trên và ngay bên dưới nó. (Ví dụ: Truyền sau khi truyền dữ liệu từ Tầng 2, Tầng 1 và Tầng 3)



I Mô hình OSI 7 Tầng

- ▶ Các giao tiếp này dựa trên mô hình tham chiếu OSI đạt chuẩn công nghiệp và được phân loại theo chức năng của chúng.
- ▶ Tầng ứng dụng (tầng 7)
 - Tầng ứng dụng ở trên cùng mô hình OSI cung cấp cho người dùng quyền tiếp cận đến các tài nguyên mạng.
 - Cung cấp tất cả các giao diện cơ bản cho môi trường mạng.
 - Thường là tầng duy nhất mà người dùng có thể nhìn thấy.
- ▶ Tầng trình diễn (6 tầng)
 - Chuyển đổi dữ liệu từ tầng ứng dụng sang một định dạng có thể đọc được.
 - Mã hóa và giải mã các dữ liệu được gửi hoặc nhận từ tầng ứng dụng.
 - Tầng này sử dụng cơ chế mã hóa và giải mã để đảm bảo dữ liệu được an toàn.



I Mô hình OSI 7 Tầng

▶ Tầng phiên (tầng 5)

- Quản lý các phiên làm việc hoặc đối thoại giữa hai máy tính.
- Kết nối, quản lý và ngắt tất cả các thiết bị giao tiếp.
- Có khả năng ngăn chặn tình trạng mất kết nối đột ngột và ngắt kết nối đúng cách giữa các máy tính.
- Tầng này có nhiệm vụ kiểm tra xem các tác vụ giao tiếp là đơn công hay song công.

▶ Tầng giao vận (tầng 4)

- Gửi dữ liệu tin cậy xuống tầng dưới.
- Bao gồm định dạng, điều khiển lưu lượng, phân đoạn, hợp đoạn và quản lý lỗi.
- Tầng này đảm bảo không có lỗi trong kết nối điểm - điểm.
- Công việc truyền dữ liệu an toàn và tin cậy rất phức tạp nhưng cũng rất quan trọng.
- Là tầng nơi tường lửa và các máy chủ proxy hoạt động



I Mô hình OSI 7 Tầng

▶ Tầng Mạng (tầng 3)

- Một trong những tầng OSI phức tạp nhất, chịu trách nhiệm định tuyến giữa các mạng vật lý.
- Tầng này chịu trách nhiệm quản lý các địa chỉ logic, phân đoạn gói dữ liệu, nhận diện giao tiếp, và phát hiện lỗi tại các máy chủ tính kết nối mạng.

▶ Tầng Liên kết dữ liệu (tầng 2)

- Chịu trách nhiệm truyền dữ liệu giữa các mạng vật lý.
- Cung cấp một lược đồ định địa chỉ và kiểm tra lỗi
- Các thiết bị như bộ cầu nối (bridge) và bộ chuyển mạch (switch) hoạt động ở tầng này

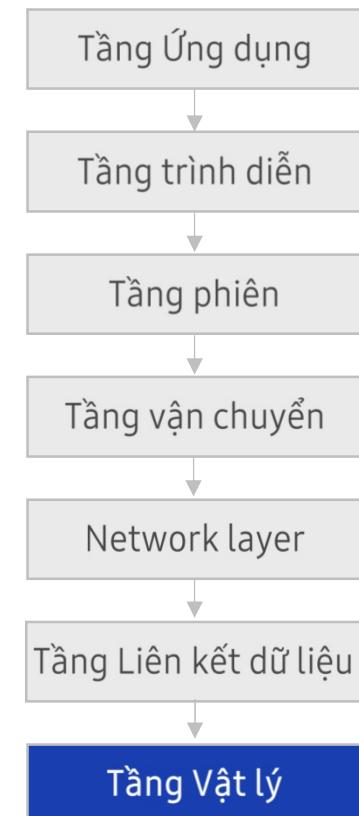


Mô hình OSI 7 Tầng

Tầng Vật lý (tầng 1)

- Phương tiện vật lý được sử dụng để truyền dữ liệu vào mạng.
- Xác định các đặc tính vật lý và điện tử cho tất cả phần cứng, bao gồm điện áp, bộ chia mạng, bộ chuyển đổi mạng, bộ lặp, và các thông số kỹ thuật của dây cáp.
- Thiết lập và ngắt kết nối, cung cấp tài nguyên giao tiếp chung, chuyển đổi tín hiệu tương tự sang tín hiệu số và ngược lại.

Tầng	Các giao tiếp
Ứng dụng	HTTP, SMTP, FTP, Telnet
Trình diễn	ASCII, MPEG, JPEG, MIDI
Phiên	NetBIOS, SAP, SDP, NWLink
Giao vận	TCP, UDP, SPX
Mạng	IP, IPX
Liên kết dữ liệu	Ethernet, Token Ring, FDDI, AppleTalk



So sánh giữa kết nối hữu tuyến và vô tuyến (1/2)

Yếu tố	Kết nối hữu tuyến	Kết nối vô tuyến
Thiết lập đường truyền	<ul style="list-style-type: none"> Việc thiết lập đường kết nối cần thực hiện, có thể gặp những vấn đề khó khăn trong thiết lập 	<ul style="list-style-type: none"> Không cần thiết lập.
Cự ly kết nối	<ul style="list-style-type: none"> Kết nối chặng dài. 	<ul style="list-style-type: none"> Giới hạn
Tần suất lỗi hệ thống	Luôn ổn định	<ul style="list-style-type: none"> Có thể xuất hiện những điểm sự cố
Chi phí bảo trì	<ul style="list-style-type: none"> Chi phí bảo trì thấp. 	<ul style="list-style-type: none"> Chi phí bảo trì cao.
Tần suất lỗi giao tiếp	<ul style="list-style-type: none"> Ít lỗi. 	<ul style="list-style-type: none"> Nhiều lỗi
Ảnh hưởng của nhiễu	<ul style="list-style-type: none"> Các yếu tố nhiễu tác động ít hơn. 	<ul style="list-style-type: none"> Vấn đề nhiễu ảnh hưởng nhiều hơn và thường xuyên hơn.

Comparing wire communication and wireless communication (2/2)

Yếu tố	Kết nối hữu tuyến	Kết nối vô tuyến
Ảnh hưởng của môi trường	<ul style="list-style-type: none"> Chống chịu được trước sự thay đổi của môi trường. 	<ul style="list-style-type: none"> Chịu ảnh hưởng của các biến động môi trường như: mưa, gió, tuyến,....
Yêu cầu cấp nguồn	<ul style="list-style-type: none"> Không yêu cầu. Đường truyền dữ liệu và đường cấp nguồn có thể được sử dụng trên cùng một cáp. 	<ul style="list-style-type: none"> Yêu cầu cấp nguồn.
Thay pin nguồn	<ul style="list-style-type: none"> Không cần 	<ul style="list-style-type: none"> Thay thế thủ công và thực hiện định kỳ
Chi phí cho thiết bị truyền thông	<ul style="list-style-type: none"> Chi phí thấp 	<ul style="list-style-type: none"> Nó có thể là chi phí bổ sung.
Lắp đặt Antenna	<ul style="list-style-type: none"> Không có 	<ul style="list-style-type: none"> Một số vấn đề có thể xảy ra tại vị trí lắp đặt anten.
Cấu hình mạng 1:N	<ul style="list-style-type: none"> Multi-Drop is automatically built into the modem. 	<ul style="list-style-type: none"> 1:N configuration is possible, often with problems in the field.

Bài 2.

giao tiếp truyền thông

- | 2.1. giao tiếp là gì
- | 2.2. Mô hình OSI
- | **2.3. giao tiếp truyền thông TCP/IP**
- | 2.4. HTTP
- | 2.5. MQTT
- | 2.6. CoAP
- | 2.7. WebSocket

giao tiếp truyền thông TCP/IP

- | giao tiếp điều khiển lớp vận chuyển/giao tiếp Internet
- | Được phát triển bởi DARPA
- | Không có tiêu chuẩn giao tiếp chính thức nào
- | Hệ thống phân cấp tương tác của các mô-đun cung cấp chức năng cụ thể
- | Xác định 5 tầng cơ bản
 - ▶ Tầng ứng dụng
 - ▶ Tầng vận chuyển (máy trạm– máy trạm)
 - ▶ Tầng mạng (Internet)
 - ▶ Tầng truy cập mạng
 - ▶ Tầng vật lý

Tầng Vật lý

- | Giao diện vật lý giữa DTE (ví dụ máy tính hoặc thiết bị đầu cuối) và một phương tiện truyền dẫn
- | Xác định các yếu tố:
 - ▶ Đặc điểm của phương tiện
 - ▶ Bản chất của tín hiệu
 - ▶ Tốc độ truyền dữ liệu

Tầng Truy cập (truy nhập) mạng

- | Trao đổi dữ liệu giữa các hệ thống trong một mạng chung
- | Sử dụng địa chỉ của máy chủ và máy đích
- | Có thể ưu tiên truyền dữ liệu
- | Các phần mềm ở tầng này phụ thuộc vào mạng
 - ▶ Ví dụ: mạng X.25 hay mạng Ethernet
- | Tính chất độc lập, có nghĩa là phần mềm xử lý không cần quan tâm đến một mạng cụ thể (kết nối vật lý mà mạng sử dụng)

Tầng Internet

- | Internet là sự kết nối của hai hoặc nhiều mạng
- | Tầng Internet xử lý các tác vụ giống như tầng truy cập mạng nhưng là giữa các mạng chứ không phải là giữa các nút trong mạng
- | Sử dụng IP để đánh địa chỉ và định tuyến trên toàn mạng
- | Được triển khai trong các máy trạm và các bộ định tuyến

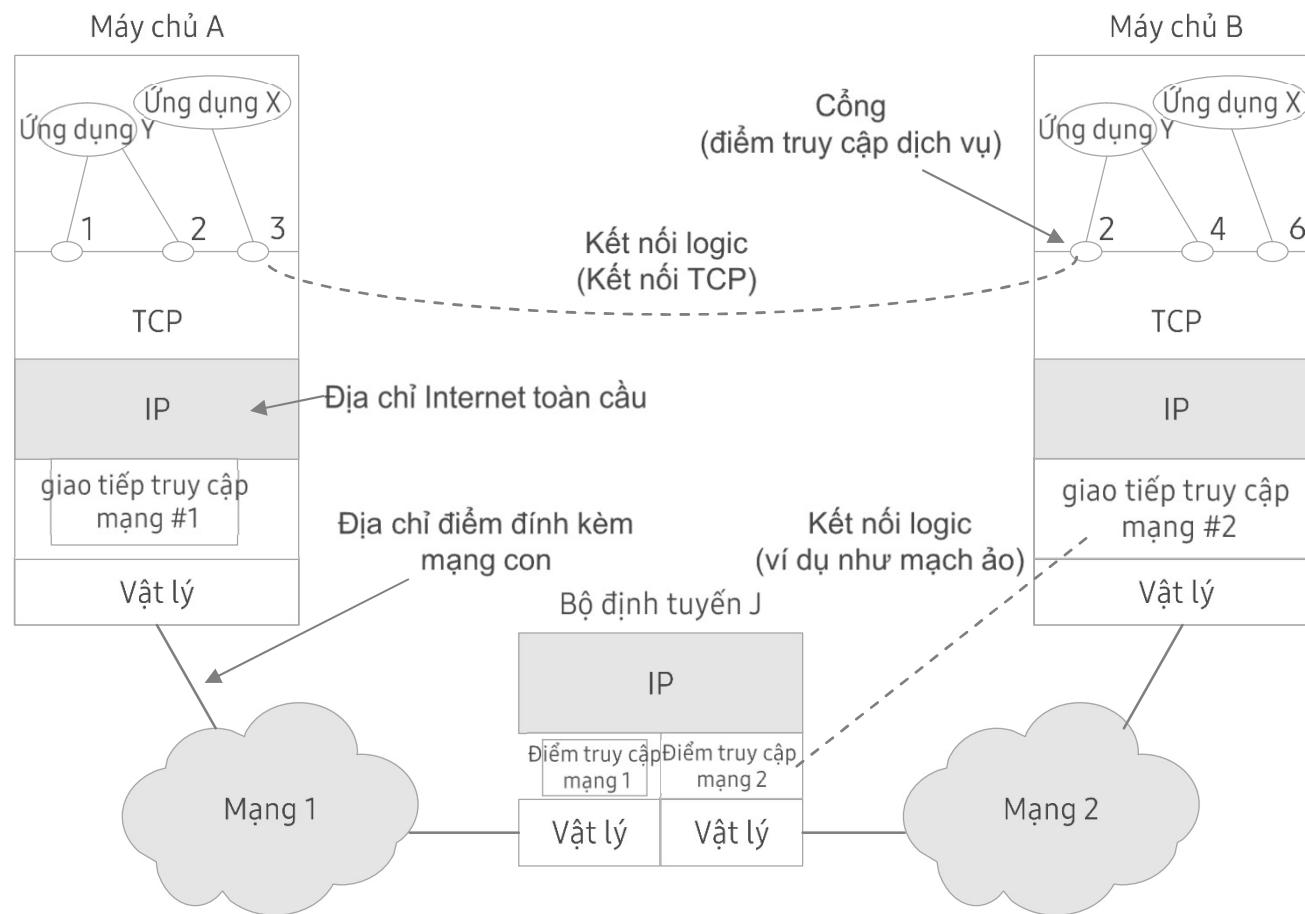
Tầng giao vận

- | Còn được gọi là tầng máy chủ - máy chủ
- | Trao đổi dữ liệu một cách tin cậy giữa các ứng dụng
- | Sử dụng các giao tiếp TCP để truyền dữ liệu

Tầng ứng dụng

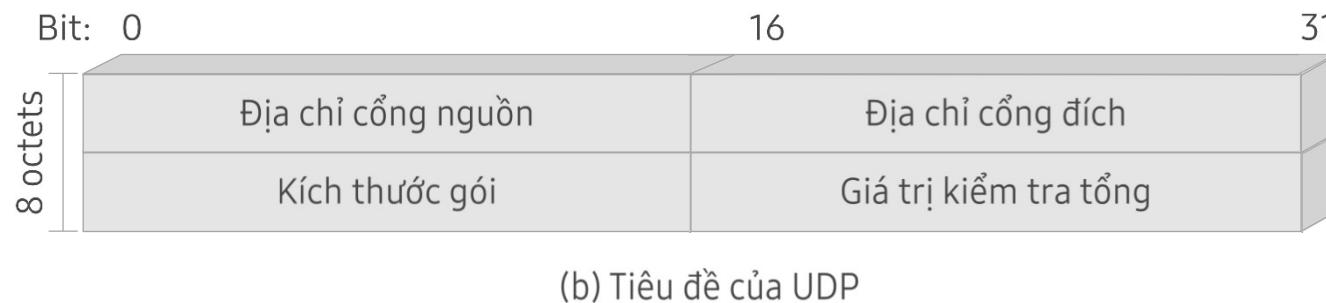
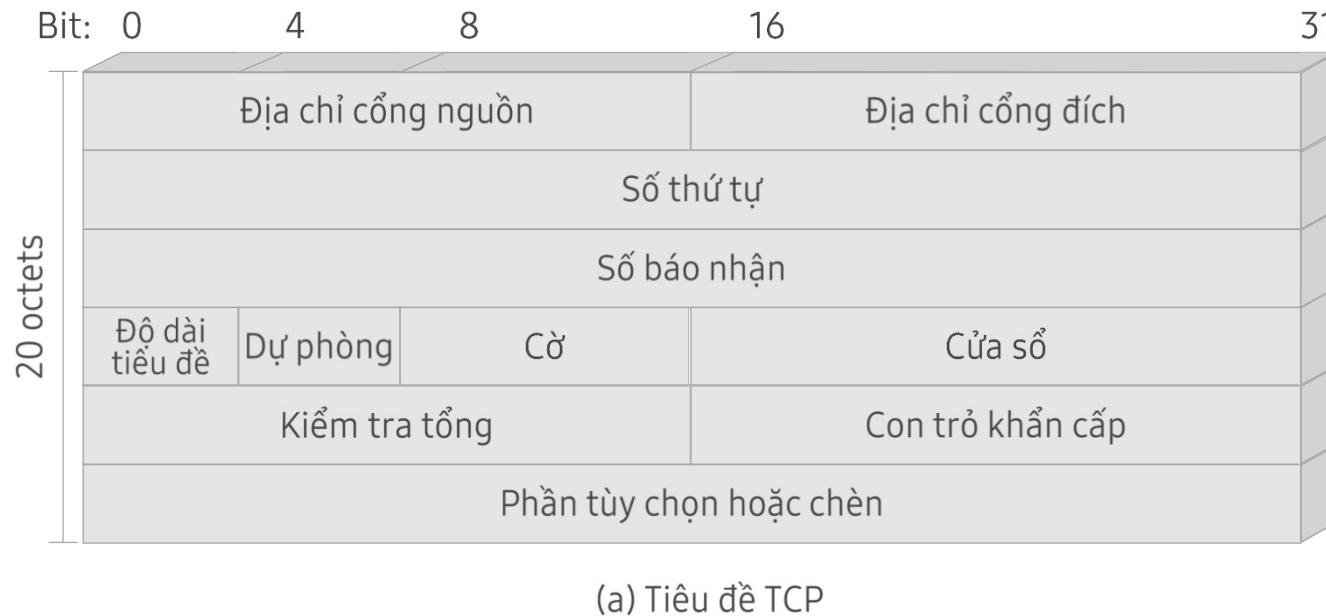
- | Logic cần thiết để hỗ trợ nhiều ứng dụng
- | Các mô-đun độc lập hỗ trợ từng loại ứng dụng
 - ▶ Ví dụ: truyền tệp tin

Hoạt động của TCP/IP



TCP & UDP

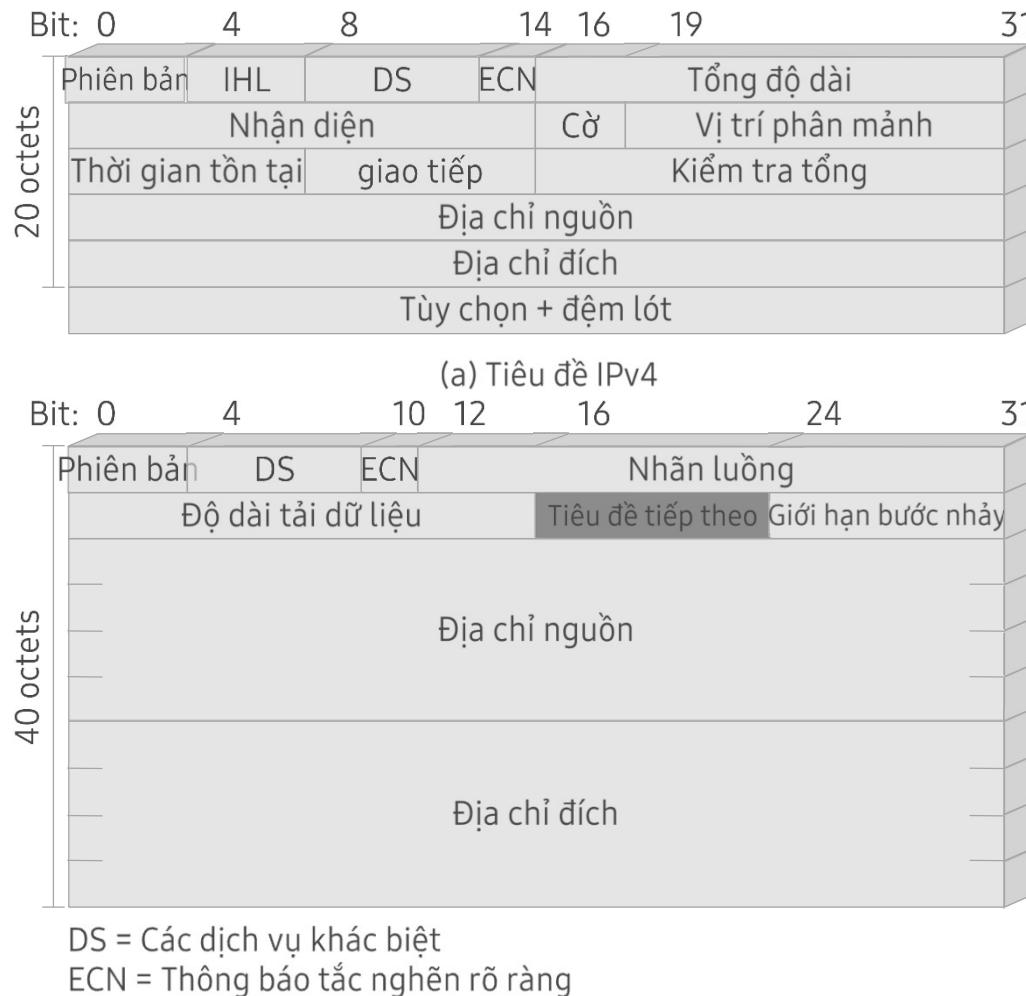
- | Hầu hết các ứng dụng TCP/IP đều sử dụng giao tiếp TCP cho tầng vận chuyển
- | giao tiếp TCP cung cấp kết nối (liên kết logic) giữa hai đối tượng, áp dụng cơ chế điều khiển lỗi, điều khiển luồng.
- | UDP (giao tiếp gói tin người dùng) không duy trì kết nối, do đó không đảm bảo việc truyền dữ liệu, không đảm bảo trình tự hoặc chống trùng lặp dữ liệu.



IP và IPv6

- | IP sử dụng cho các địa chỉ nguồn và đích, với độ dài là 32 bit
- | IPv6 (tiêu chuẩn 1996) hỗ trợ các địa chỉ 128 bit
- | Quá trình truyền dữ liệu đến IPv6 sẽ rất chậm

Tiêu đề IP



Ứng dụng TCP/IP

| SMTP (giao tiếp truyền tải thư tín đơn giản)

- ▶ Thiết bị e-mail cơ bản, truyền dữ liệu giữa các máy chủ

| FTP (giao tiếp truyền tệp tin)

- ▶ Gửi các tệp tin từ một hệ thống đến một hệ thống khác theo yêu cầu của người dùng

| SSH (Shell bảo mật)

- ▶ Hỗ trợ khả năng đăng nhập từ xa có bảo mật, cho phép người dùng đăng nhập từ xa vào máy tính

Kết nối mạng Internet

- | Các mạng được kết nối với nhau thường sử dụng giao tiếp TCP/IP
- | Đối với người dùng, đó là một mạng lớn
- | Mạng Internet toàn cầu là ví dụ tiêu biểu nhất, nhưng mạng nội bộ và mạng ngoại bộ cũng là các ví dụ của kết nối mạng

Bộ định tuyến

- | Thiết bị được sử dụng để kết nối các mạng độc lập với nhau
- | Có nhiều chức năng quan trọng:
 - ▶ Cung cấp liên kết giữa các mạng
 - ▶ Cung cấp cơ chế định tuyến và truyền dữ liệu giữa các quá trình trên các hệ thống ở các mạng khác nhau
 - ▶ Cung cấp các chức năng mà không yêu cầu chỉnh sửa cấu trúc mạng

So sánh giữa TCP/IP và OSI



Đánh địa chỉ

| Mỗi liên hệ giữa Địa chỉ và tầng trong giao tiếp TCP/IP



| Địa chỉ vật lý

- ▶ Địa chỉ của liên kết
- ▶ Địa chỉ của nút được xác định trong mạng WAN hoặc mạng LAN
- ▶ Địa chỉ của NIC (bộ điều khiển giao tiếp mạng – card mạng) Ethernet, 6 byte (48 bit).
- ▶ Unicast, multicast, broadcast

07:01:02:01:2C:4B

địa chỉ vật lý gồm 6 byte (12 chữ số thập lục phân)

| Địa chỉ logic (hoặc địa chỉ IP)

- ▶ Định danh các máy tính (máy trạm) đang kết nối mạng Internet: địa chỉ 32 bit
- ▶ Địa chỉ Unicast (địa chỉ một người dùng), Địa chỉ Multicast (nhiều người nhận), broadcast (tất cả hệ thống trong mạng riêng)
- ▶ Được chuyển đổi sang địa chỉ IPv6 có độ dài địa chỉ gấp bốn lần.

| Địa chỉ cổng

- ▶ Địa chỉ để phân biệt các quy trình gửi và nhận dữ liệu trong giao tiếp máy tính - máy tính
- ▶ Địa chỉ được gán với mỗi ứng dụng truyền thông dữ liệu qua mạng
- ▶ Có độ dài 16 bit (từ 0 đến 65535)

Bài 2.

giao tiếp truyền thông

- | 2.1. giao tiếp là gì
- | 2.2. Mô hình OSI
- | 2.3. giao tiếp truyền thông TCP/IP
- | 2.4. **HTTP**
- | 2.5. MQTT
- | 2.6. CoAP
- | 2.7. WebSocket

Khái niệm HTTP

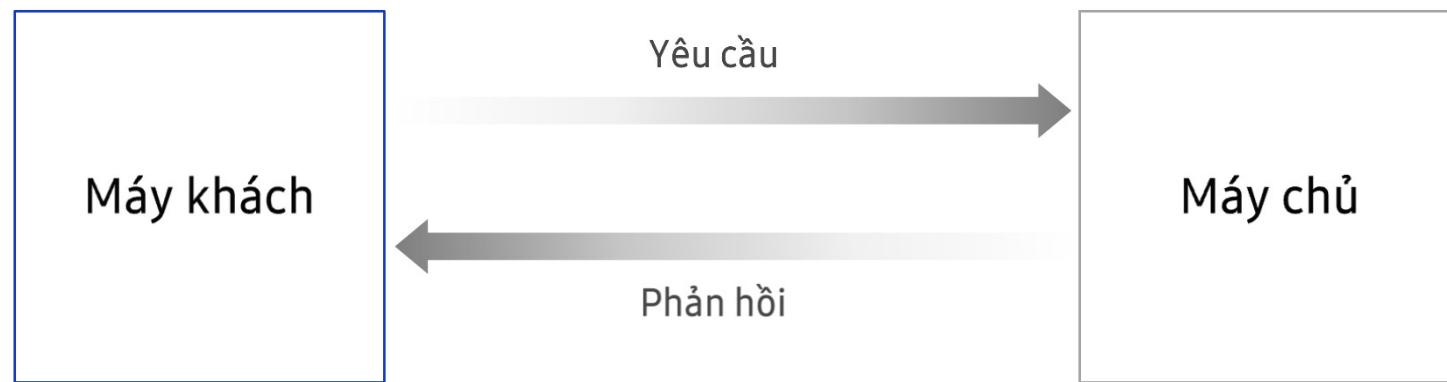
- | giao tiếp truyền tải siêu văn bản (HTTP) là một giao tiếp sử dụng mô hình máy chủ/máy khách để gửi và nhận dữ liệu qua mạng Internet.
- | giao tiếp truyền thông dựa trên cơ chế yêu cầu/phản hồi giữa trình duyệt web và máy chủ web

Tính năng và lợi ích của HTTP

- | Sau khi kết nối đến máy chủ trong một giao tiếp không kết nối, khi nhận được phản hồi cho yêu cầu, kết nối giữa máy khách và máy chủ sẽ được giải phóng (hủy).
 - ▶ ưu điểm: hiệu quả của dịch vụ cao, không cần bảo dưỡng nhiều.
 - ▶ nhược điểm: Không xác định được trạng thái trước đó tại máy khách sau khi hủy kết nối nên không thể lưu các thông tin về máy khách tại máy chủ.
- | Cách để giải quyết vấn đề "không thể xác định trạng thái" của máy khách
 - ▶ Cookie: xử lý vấn đề này bằng cách lưu các thông tin trạng thái cho máy khách

Môi trường hoạt động của HTTP

- | Máy khách: Một máy tính có phần mềm kết nối với máy chủ thông qua một đường dẫn (URL) kèm theo dữ liệu yêu cầu (chrome, firefox, ie, v.v.).
- | Máy chủ: Một máy tính có phần mềm có chức năng đáp ứng các yêu cầu từ máy khách, diễn giải và phản hồi các yêu cầu đó (Apache, nginx, IIS, lighttpd)



Phương pháp HTTP

- | GET: Yêu cầu thông tin
- | POST: chèn/gửi thông tin
- | PUT: cập nhật thông tin
- | DELETE: xóa thông tin
- | HEAD: chỉ yêu cầu các thông tin tiêu đề
- | OPTIONS: Yêu cầu phương thức mà máy chủ web hỗ trợ
- | TRACE: Trả về yêu cầu từ máy khách

Mã phản hồi HTTP

| 1XX phản hồi có điều kiện

| 2XX thành công

- 200: Thành công, 201: Đã tạo, 202: Đã cho phép, 204: Không có nội dung
- 205: đặt lại nội dung, 206: thành công một phần, 207: đa trạng thái

| 3XX điều hướng lại

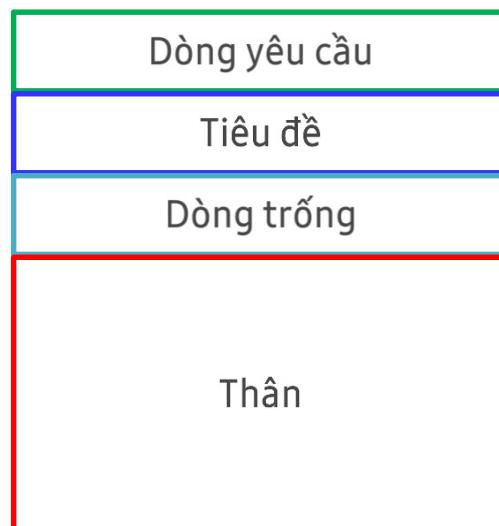
- 300: nhiều mục chọn, 301: chuyển động cố định, 302: chuyển động tạm thời
- 303: tầm nhìn ở các vị trí khác, 304: không được chỉnh sửa, 305: sử dụng proxy

| 4XX Lỗi yêu cầu

- 400: Yêu cầu sai, 401: Lỗi ủy quyền, 403: Bị cấm 404: Không tìm thấy
- 405: Phương thức không được phép 406: Không được phép, 408: Hết thời gian yêu cầu
- 410 không tồn tại, 411: yêu cầu khai báo độ dài

Định dạng dữ liệu yêu cầu và phản hồi HTTP

Bản tin yêu cầu



Dòng yêu cầu

[Phương thức] [URL] [Phiên bản][CR][LF]

Bản tin phản hồi



Dòng trạng thái

[Phiên bản] [SC] [SP][CR][LF]

(SC: Mã trạng thái), SP: Cụm từ trạng thái

Bài 2.

giao tiếp truyền thông

- | 2.1. giao tiếp là gì
- | 2.2. Mô hình OSI
- | 2.3. giao tiếp truyền thông TCP/IP
- | 2.4. HTTP
- | **2.5. MQTT**
- | 2.6. CoAP
- | 2.7. WebSocket

Khái niệm MQTT

- | Message Queue Telemetry Transport (MQTT) là một giao tiếp gửi thông điệp được phát triển bởi IBM vào năm 1999 để gửi thông điệp tin cậy (giám sát từ xa) cho các thiết bị nhỏ như dụng cụ đo và cảm biến trong các môi trường mạng có độ trễ cao và thường bị mất tín hiệu.
- | MQTT là giao tiếp hoạt động theo mô hình Publish / Subscribe cấp thấp, được sử dụng trong mô hình giao tiếp máy - máy và IoT.

Ưu nhược điểm của MQTT

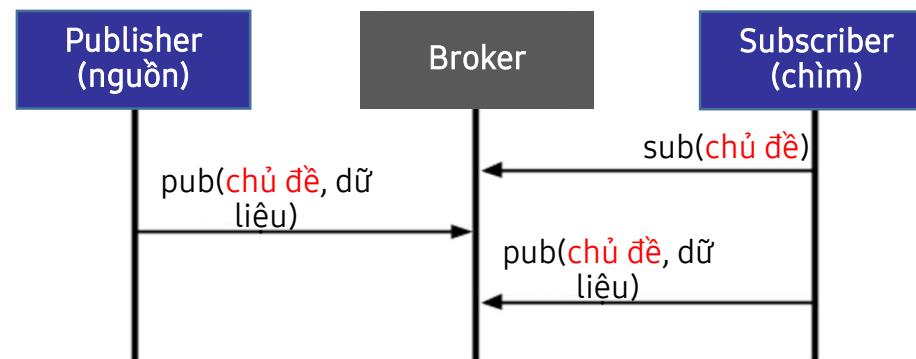
- | Định dạng thông điệp đơn giản, yêu cầu băng thông mạng hẹp và mức tiêu thụ năng lượng thấp.
- | Được thiết kế cho các môi trường mạng hoạt động với mức tiêu thụ năng lượng thấp và có độ trễ , khả năng lỗi/mất dữ liệu cao khi truyền tải, phù hợp với các thiết bị nhỏ và thiết bị thu thập thông tin cảm biến.
- | giao tiếp này phù hợp để sử dụng trên các thiết bị nhỏ nhưng lại không hỗ trợ cụm thiết bị, do đó rất khó để có thể xây dựng một hệ thống thông điệp lớn.

Các tính năng của MQTT

| Publish/Subscribe

- ▶ Về nguyên lý, giao tiếp MQTT thực hiện thủ tục “xuất bản” (publish) một thông điệp và đăng ký (subscribe) theo dõi trên một kênh thông tin được quản lý trên máy chủ MQTT (Broker).
- ▶ Cả Publisher (Bên xuất bản) và Subscriber (Bên đăng ký theo dõi) đều là khách hàng của Broker (Máy chủ). Publisher kết nối đến Broker để thiết lập kênh thông tin (topic), còn Subscriber kết nối đến để đăng ký theo dõi một kênh thông tin nào đó.
- ▶ Nhiều Publisher và Subscriber có thể kết nối đến một Broker để xuất bản hoặc đăng ký theo dõi các kênh tại máy chủ. Nhiều máy khách có thể đăng ký theo dõi cùng một chủ đề.

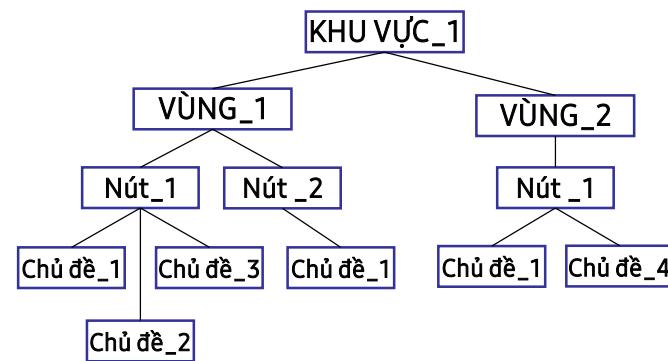
| Môi trường hoạt động theo kiểu xuất bản (publish)/đăng ký theo dõi (subscribe)



- ▶ Các Broker – máy chủ MQTT đóng vai trò trung gian giữa Publisher và Subscriber. Vì nhiều Subscriber có thể đăng ký một kênh thông tin nên nó cũng rất hữu ích để xây dựng phương thức giao tiếp 1:N.

I Chủ đề - kênh thông tin (topic)

- ▶ Publisher và Subscriber xuất bản/đăng ký theo dõi thông điệp trên từng kênh. Đây được gọi là một “chủ đề” (topic) trong MQTT.
- ▶ Các chủ đề có cấu trúc phân cấp, được phân tách bằng các dấu gạch chéo (/) và hỗ trợ các ký tự đại diện để chỉ rõ nhiều chủ đề khi thực hiện các thủ tục xuất bản/đăng ký theo dõi các thông điệp.
- ▶ Trong MQTT, các chủ đề được xây dựng bằng cách sử dụng dấu '/'. Nếu có ý định cấu hình các lớp như trong hình dưới, sẽ rất dễ dàng để quản lý dữ liệu, ví dụ như cảm biến IoT.



- ▶ Kiểu đại diện
 - + (Cấp độ đơn): Xử lý ngẫu nhiên một chủ đề
 - # (Đa cấp độ): Liên quan đến nhiều chủ đề con, chỉ được đặt ở phía dưới cùng của chuỗi chủ đề
- ▶ Ví dụ về việc sử dụng đại diện
 - sensors / NODE_NAME / + / CPU
 - Sensors / NODE_NAME / #

I QoS (Chất lượng dịch vụ)

- ▶ MQTT có ba cấp độ chất lượng dịch vụ (QoS)
 - Cấp độ 0: Chỉ 1 thông điệp gửi, có thể mất
 - Cấp độ 1: Thông điệp được gửi ít nhất một lần, khả năng thành công cao gấp đôi
 - Cấp độ 2: Thông điệp được gửi đúng một lần, đảm bảo chất lượng nhưng đòi hỏi hệ thống phải có hiệu suất cao

※ tham khảo:

Cấp 0: Một tin nhắn chỉ được gửi một lần. Sau khi chuyển tiếp, phía nhận sẽ không kiểm tra dữ liệu nhận được.

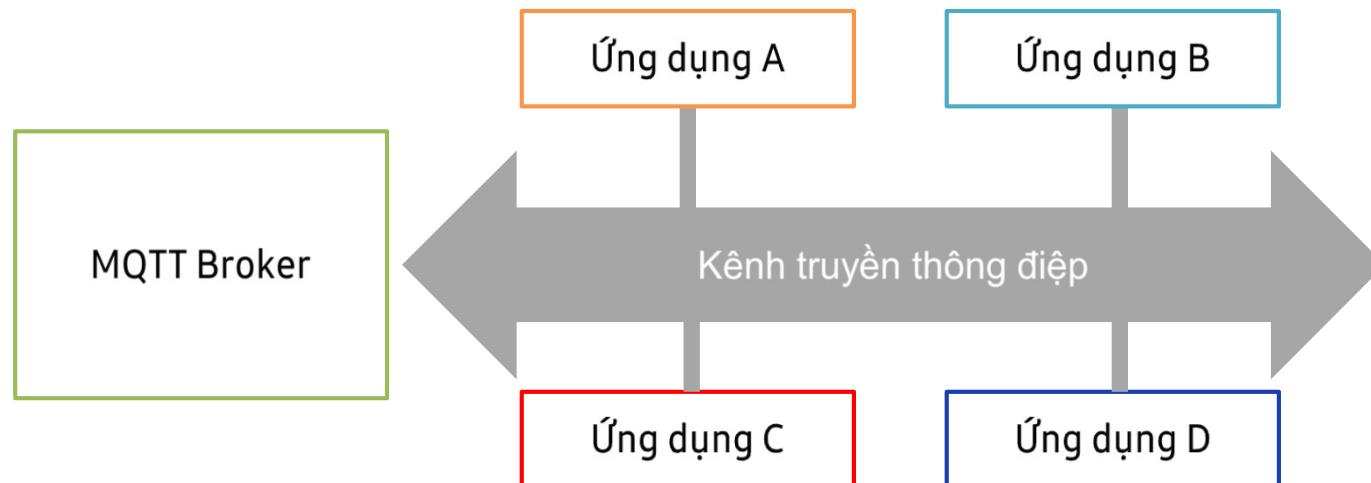
Cấp 1: Một tin nhắn được gửi nhiều hơn một lần. Quá trình “bắt tay” (handshaking) được theo dõi, nhưng khả năng nhận trùng lặp vẫn có thể xảy ra vì việc theo dõi không chặt chẽ.

Cấp 2: Một tin nhắn chỉ được gửi một lần và tất cả các quy trình handshaking đều được kiểm tra.

- ▶ Mức QoS càng cao, chất lượng truyền thông càng tốt, nhưng có thể làm giảm hiệu suất. Do đó, QoS của MQTT được quyết định tùy theo tính chất của dự án.

I Kênh truyền thông điệp/bản tin

- MQTT có hệ thống kênh truyền thông điệp. Khi máy chủ broker MQTT tạo ra một kênh truyền thông điệp và gửi một thông điệp từ publisher đến máy chủ (broker), các ứng dụng đích kèm kênh truyền đó sẽ đọc thông điệp.
- Các thông điệp thuộc các chủ đề khác nhau có thể di chuyển trên kênh truyền thông điệp, tạo ra một kênh truyền thông điệp có tên xác định để phân biệt giữa các thông điệp/bản tin.



Định dạng bản tin MQTT

- Có thể tùy ý lựa chọn định dạng bản tin MQTT, trừ phần tiêu đề luôn có độ dài cố định là 2 byte

	0	1	2	3	4	5	6	7
Byte1	Kiểu thông điệp			Cờ DUP	Cấp độ QoS		RETAIN	
Byte2			Độ dài còn lại					

| Kiểu thông điệp

Kiểu	Đánh số	Mô tả
Dùng sau	0	Dùng sau
CONNECT	1	Máy khách yêu cầu kết nối đến máy chủ
CONNACK	2	Báo nhận kết nối
PUBLISH	3	Xuất bản thông điệp
PUBACK	4	Báo nhận xuất bản
PUBREC	5	Xuất bản đã nhận (xác nhận đã gửi phần 1)
PUBREL	6	Xuất bản đã nhận (xác nhận đã gửi phần 2)
PUBCOMP	7	Xuất bản hoàn thành (xác nhận đã gửi phần 3)
SUBSCRIBE	8	Máy khách yêu cầu đăng ký theo dõi
SUBACK	9	Báo nhận đăng ký theo dõi
UNSUBSCRIBE	10	Báo nhận hủy đăng ký theo dõi
UNSUBACK	11	Báo nhận hủy đăng ký theo dõi
PINGREQ	12	Yêu cầu PING
PINGRESP	13	Phản hồi PING
DISCONNECT	14	Máy khách ngắt kết nối
Dùng sau	15	Dùng sau

| Cờ DUP & cấp độ QoS và RETAIN

- ▶ RETAIN chỉ được sử dụng trong các thông điệp của thủ tục publish
- ▶ (Dù các thông điệp được lưu trong hàng chờ)

Vị trí Bit	Tên	Mô tả
3	DUP	Phân phối trùng lặp
2-1	QoS	Chất lượng dịch vụ
0	RETAIN	Cờ RETAIN

I Cấp độ QoS

Giá trị QoS	Bit 2	Bit 1	Mô tả
0	0	0	Gửi và Quên (Tối đa một lần)
1	0	1	Xác nhận phân phối (Ít nhất một lần)
2	1	0	Gửi và Quên (Chính xác một lần)
3	1	1	Gửi và Quên (Tối đa một lần)
4	1	1	Dùng sau

| Độ dài còn lại

- ▶ Được sử dụng để đo kích thước toàn bộ thông điệp, bao gồm tiêu đề thay đổi. Nó sử dụng từ 1 đến 4 byte tùy theo kích thước dữ liệu.

| Tiêu đề thay đổi

- ▶ Một số lệnh MQTT có chứa một phần tử tiêu đề thay đổi. Các tiêu đề thay đổi được đặt giữa các tiêu đề cố định và tải dữ liệu

Thử nghiệm với MQTT

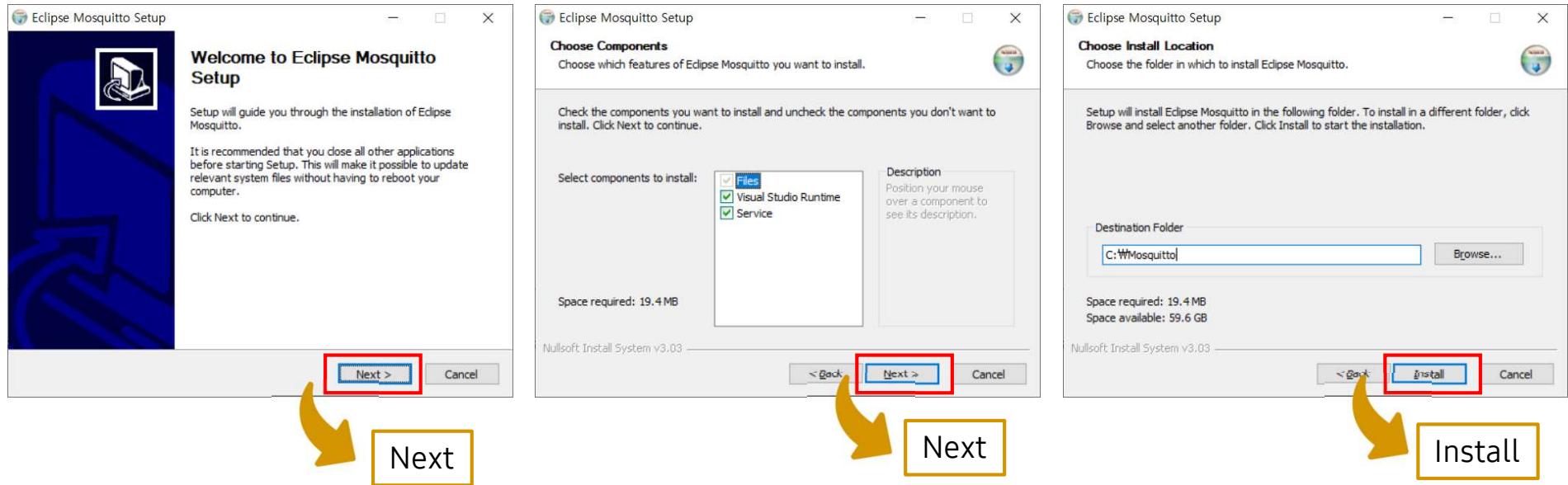
| Triển khai một chương trình trò chuyện đơn giản, để hiểu rõ hơn về giao tiếp MQTT

| Môi trường và các công cụ thực hiện:

- ▶ Windows 10 64bit
- ▶ Ruby 2.3.3
- ▶ Mosquitto 1.4.7
- ▶ (không bắt buộc) Raspberry pi

Thiết lập môi trường làm việc với MQTT

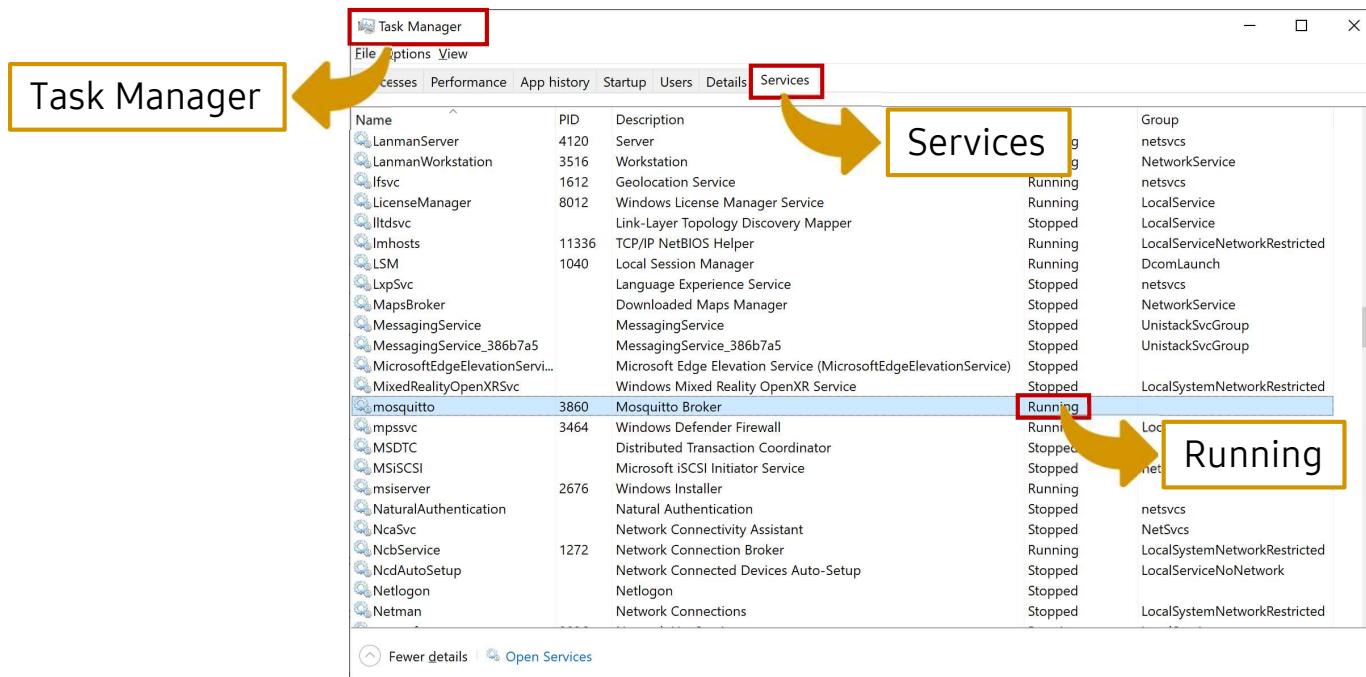
- | Truy cập trang web Mosquitto (<https://mosquitto.org/download/>) để tải xuống phiên bản Windows 'mosquitto-2.0.12-install-windows-x64.exe' và cài đặt trong C:\mosquitto.



Chạy máy chủ MQTT Broker Server (mosquitto)

I Chạy máy chủ MQTT - Mosquitto

- Sau khi cài đặt hoàn tất, Mosquitto được đăng ký như một dịch vụ chạy trong Windows. Bạn có thể kiểm tra xem dịch vụ Mosquitto có đang chạy thông qua trình quản lý tác vụ của Windows hay không. Nhấn tổ hợp phím Crtl + Shift + Esc để mở trình quản lý tác vụ của Window, và chuyển tới trang thông tin dịch vụ (services) trong cửa sổ tác vụ.

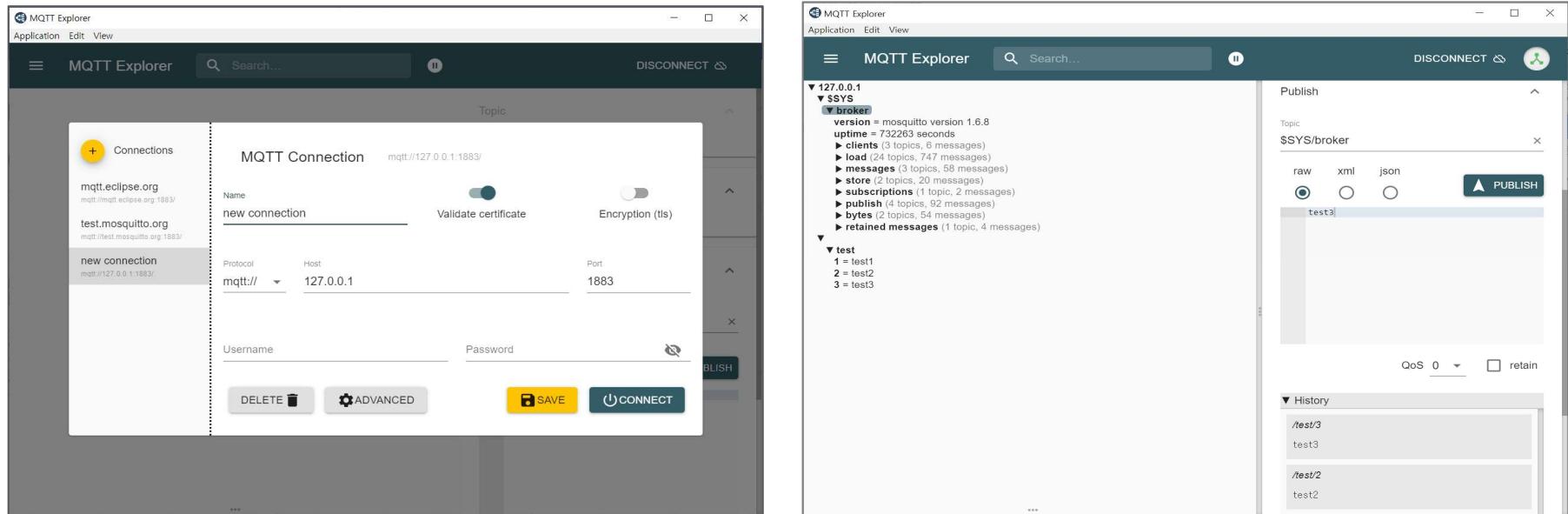


| Kiểm tra máy chủ Mosquitto

- Truy nhập trang web: <https://mqtt-explorer.com/> để tải về chương trình “MQTT Explorer”, chương trình có chức năng kết nối và truyền thông dữ liệu qua giao tiếp MQTT với giao diện GUI.

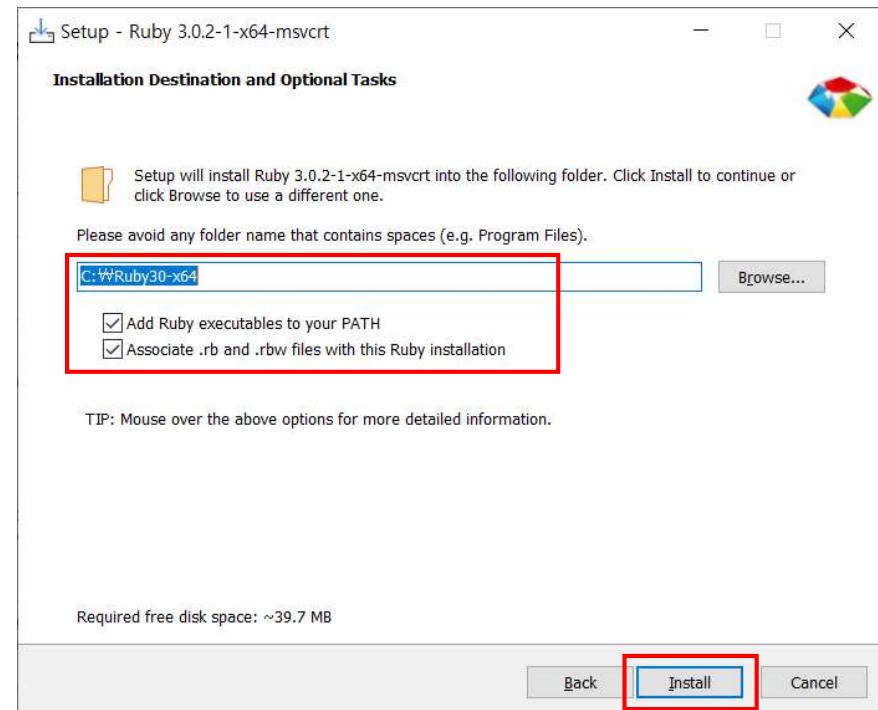
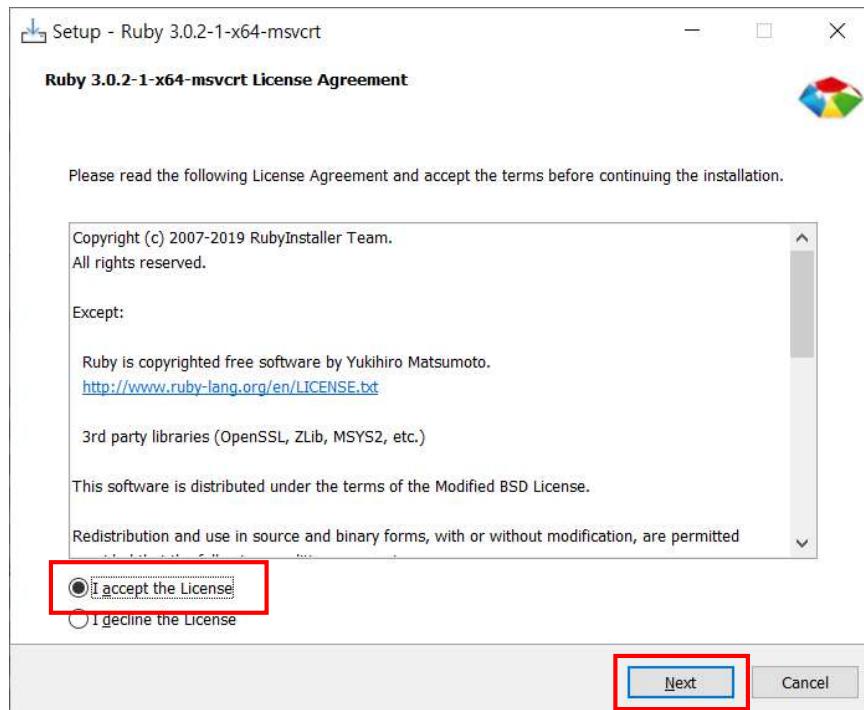
(link download: <https://github.com/thomasnordquist/MQTT-Explorer/releases/tag/0.0.0-0.4.0-beta1>)

- Thiết lập một kết nối, như mô tả trong hình bên trái. Thiết lập địa chỉ IP của Host (máy chủ MQTT) là 127.0.0.1 (hoặc localhost, địa chỉ cục bộ) và địa chỉ cổng là 1883. Ta có thể thấy rằng tin nhắn được tạo và nhận như trong hình (bên phải).

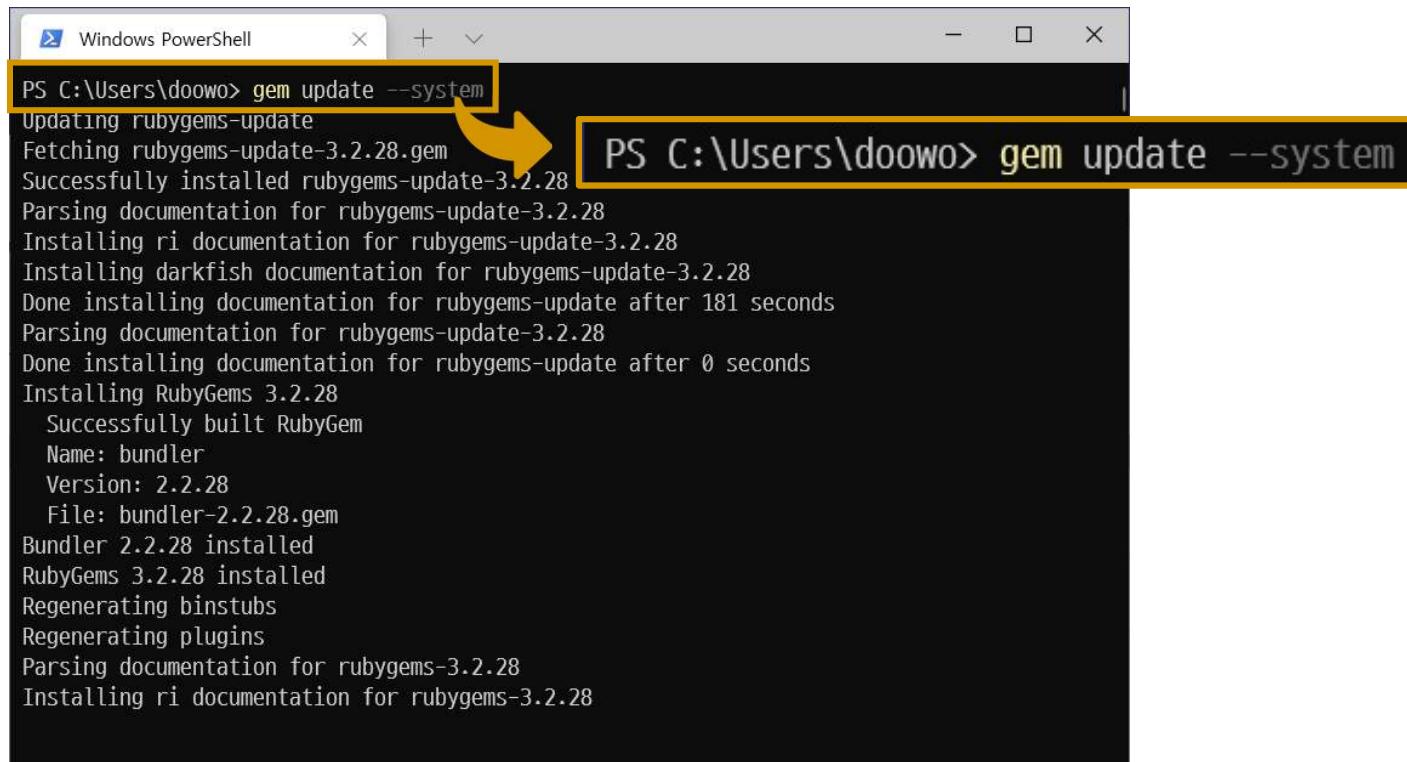


Cài đặt phần mềm Ruby để xây dựng chương trình phía máy trạm

- | Sử dụng phần mềm/môi trường phát triển Ruby để xây dựng chương trình phía máy trạm trong mô hình kết nối, truyền thông với máy chủ qua MQTT.
- | Cài đặt ruby bằng cách chạy file cài đặt ‘rubyinstaller-3.0.2-1-x64.EXE’.
 - ▶ Truy nhập trang web <https://rubyinstaller.org/downloads/> để tải về file cài đặt Ruby.

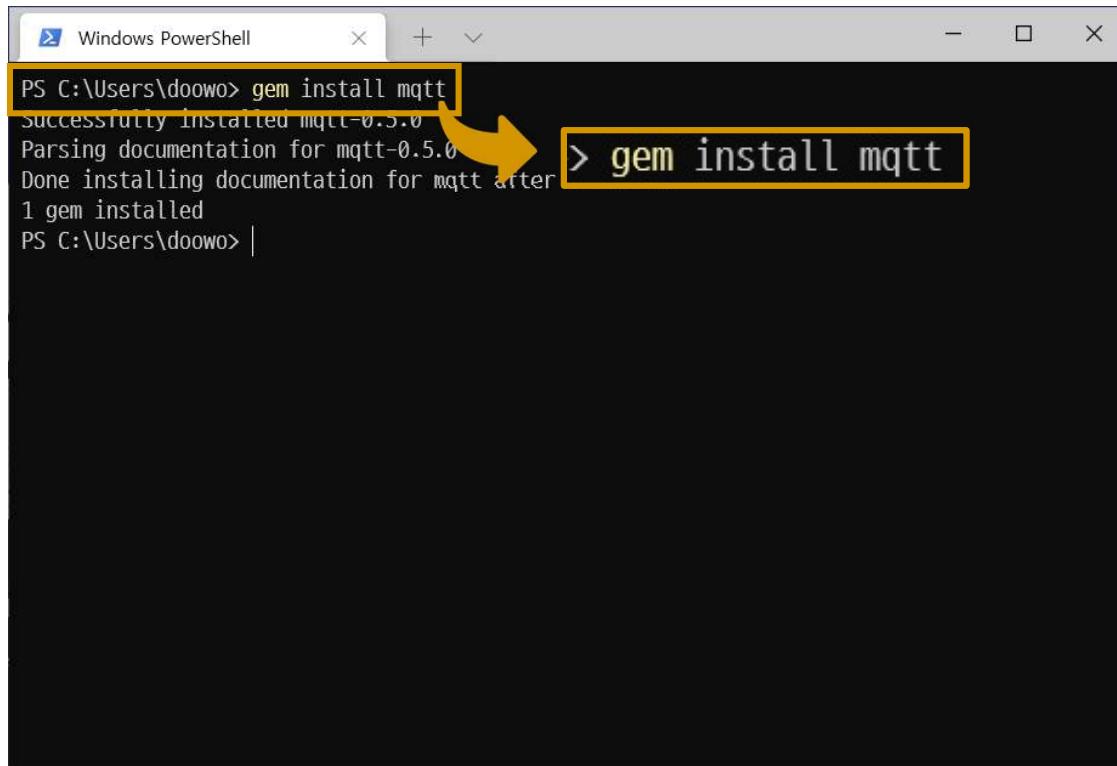


- | RubyGems là một công cụ quản lý gói (thư viện) cho Ruby.
- | Về cơ bản, Ruby gem hỗ trợ và thực hiện giao tiếp MQTT.
- | Truy nhập trang web <https://rubygems.org/pages/download> để tải về công cụ RubyGems.
 - ▶ <https://rubygems.org/>



```
Windows PowerShell
PS C:\Users\doowo> gem update --system
Updating rubygems-update
Fetching rubygems-update-3.2.28.gem
Successfully installed rubygems-update-3.2.28
Parsing documentation for rubygems-update-3.2.28
Installing ri documentation for rubygems-update-3.2.28
Installing darkfish documentation for rubygems-update-3.2.28
Done installing documentation for rubygems-update after 181 seconds
Parsing documentation for rubygems-update-3.2.28
Done installing documentation for rubygems-update after 0 seconds
Installing RubyGems 3.2.28
  Successfully built RubyGem
    Name: bundler
    Version: 2.2.28
    File: bundler-2.2.28.gem
Bundler 2.2.28 installed
RubyGems 3.2.28 installed
Regenerating binstubs
Regenerating plugins
Parsing documentation for rubygems-3.2.28
Installing ri documentation for rubygems-3.2.28
```

- | Sau khi đã cài đặt Ruby Gem, chạy PowerShell (Start/Command Prompt) trên Windows, sau đó cài đặt thư viện MQTT theo hình minh họa với các lệnh dưới đây.
- ▶ <https://rubygems.org/>



```
Windows PowerShell
PS C:\Users\doowo> gem install mqtt
Successfully installed mqtt-0.5.0
Parsing documentation for mqtt-0.5.0
Done installing documentation for mqtt after
1 gem installed
PS C:\Users\doowo>
```

A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "gem install mqtt" is entered at the prompt. The output shows the gem was successfully installed, documentation was parsed, and the process completed. A yellow arrow points from the text "gem install mqtt" back to the command line where it was typed.

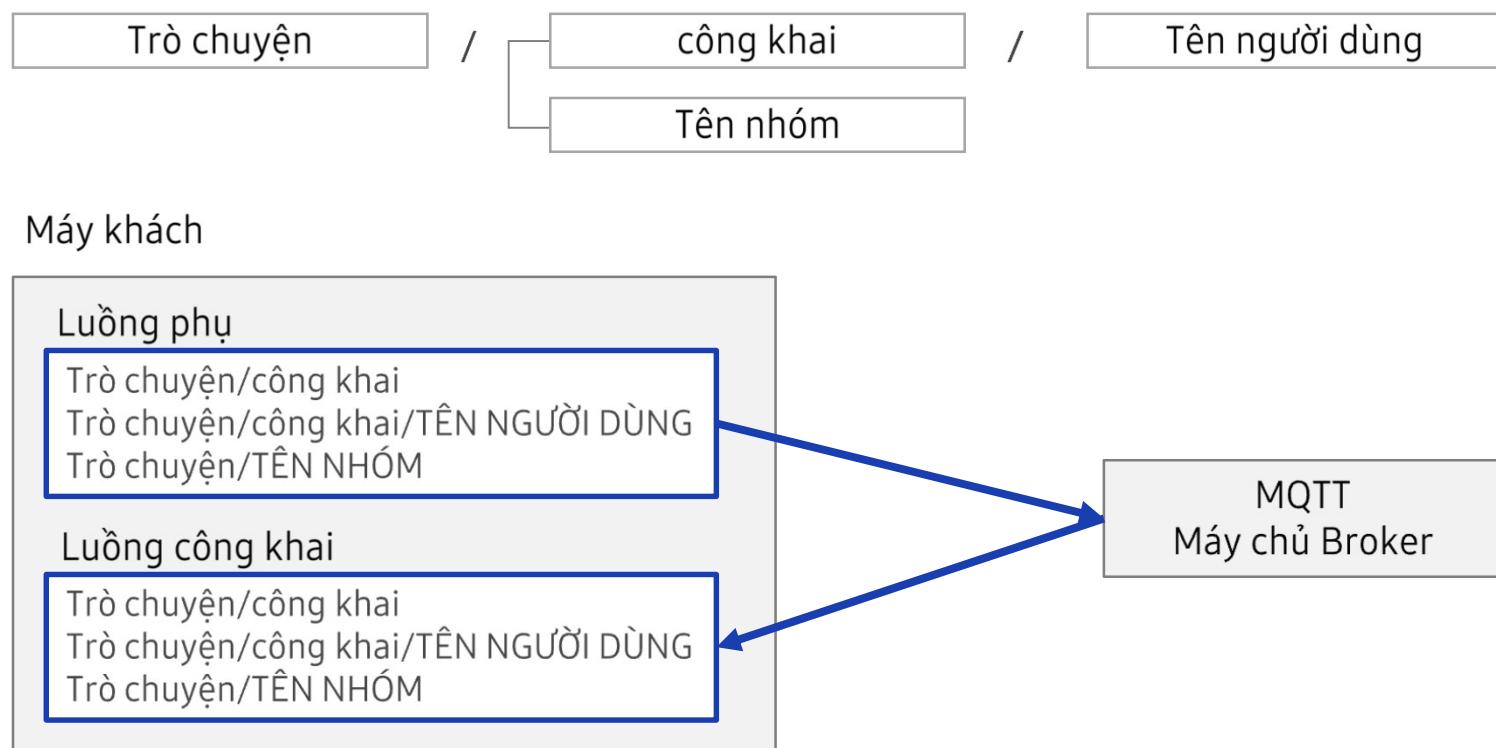
Chạy Chương trình trò chuyện MQTT (Ruby)

| Chạy Command Prompt với Ruby và sử dụng lệnh dưới đây

- ▶ cd C:\
- ▶ ruby filename.rb Argument value

Tạo Chương trình trò chuyện MQTT bằng Ruby

- | Sơ đồ chủ đề của chương trình trò chuyện MQTT và sơ đồ cấu trúc hệ thống.



Sử dụng thư viện API của MQTT bằng Ruby

I Khai báo thư viện lập trình (API)

- ▶ require 'rubygems'
require 'mqtt' # Load the mqtt library from the line of code

I Tạo đối tượng máy khách MQTT.

- ▶ mqtt = MQTT::Client.new ('Địa chỉ IP của broker MQTT')

I Kết nối bằng đối tượng máy khách MQTT

- ▶ mqtt.connect('test.mosquitto.org') do |client|
perform operations
end

| Lệnh gửi bản tin: Nhập chủ đề và thông điệp để gửi tới máy chủ

- ▶ client.publish topic, payload

| Lệnh đăng ký theo dõi kênh thông tin: Nhập chủ đề để đăng ký theo dõi, là tham số của hàm subscribe()

- ▶ client.subscribe('topic1')
- ▶ client.subscribe('topic1', 'topic2')
- ▶ client.subscribe('foo/#')

| Nhận thông điệp: Nhận về thông tin các kênh đăng ký theo dõi và các thông điệp/bản tin trên các kênh này.

Mã lệnh chương trình phía máy trạm MQTT

```
1 require 'rubygems'
2 require 'mqtt'
3 require 'readline'
4
5 # nickname
6 nickname=ARGV[0]
7
8 # localhost test
9 mqtt = MQTT::Client.new('localhost')
10
11 mqtt.connect do |client|
12   client.subscribe('chat/public')
13   client.subscribe("chat/private/#{nickname}")
14   enter_msg = "#{nickname} enter room!!"
15   client.publish 'chat/public', enter_msg
16
17 # Publish Thread
18 Thread.new do
19   while message = Readline.readline("", true)
20     case message
21     when /^priv\s*(\w*)\s*(.*)/    # individual message
22       client.publish "chat/private/#{$1}", "<#{nickname}> : #{$2}"
23     when /^quit(.*)/             # Exit
24       client.publish 'chat/public', "#{nickname} has quit (#{$1})"
25       exit 1
26     else                      # public message
27       client.publish 'chat/public', "#{nickname} : #{message}"
28     end
29   end
30 end
31
32 loop do
33   topic, message = client.get
34   print message, "\n"
35 end
36 end
```

- | Mã nguồn chương trình trò chuyện MQTT
- | Tin nhắn công khai: tin nhắn
- | Nói chuyện riêng: \ w usr_name message
- | Thoát: \ q message
- | Chạy phương thức:
ruby Chat_Example.rb username

Kết quả chạy Chương trình trò chuyện MQTT bằng Ruby

Mở hai cửa sổ như hình bên dưới và thay đổi tên người tham gia. Bạn có thể thấy rằng cuộc trò chuyện đang diễn ra thông qua MQTT Broker.

The image shows two side-by-side Windows PowerShell windows. Both windows have a yellow box around the command line and its output. A yellow arrow points from the left window's output to the right window's input field, indicating the flow of messages between the two participants.

Left Window (Daniel's Session):

```
PS C:\Mosquitto> ruby Chat_example.rb Daniel
Daniel enter room!!
Thomas enter room!!
Thomas : Hi ~~~
I'm Daniel.
Daniel : I'm Daniel.
```

Right Window (Thomas's Session):

```
PS C:\Mosquitto> ruby Chat_example.rb Thomas
Thomas enter room!!
Hi ~~~
Thomas : Hi ~~~
Daniel : I'm Daniel.
```

Bottom Line:

```
PS C:\Mosquitto> ruby Chat_example.rb Thomas
```

Raspberry pi: Cấu hình chương trình trò chuyện MQTT

- | Chạy máy chủ mosquitto trên Windows
- | Thay đổi mã nguồn mẫu của chương trình trò chuyện MQTT
 - ▶ Trước khi thay đổi: mqtt = MQTT::Client.new('localhost')
 - ▶ Sau khi thay đổi: mqtt = MQTT::Client.new('mosquitto server ip')

```
# Username  
usr_name=ARGV[0]  
  
# connect to mqtt server in localhost  
#mqtt = MQTT::Client.new('localhost')  
mqtt = MQTT::Client.new('210.107.192.185')
```

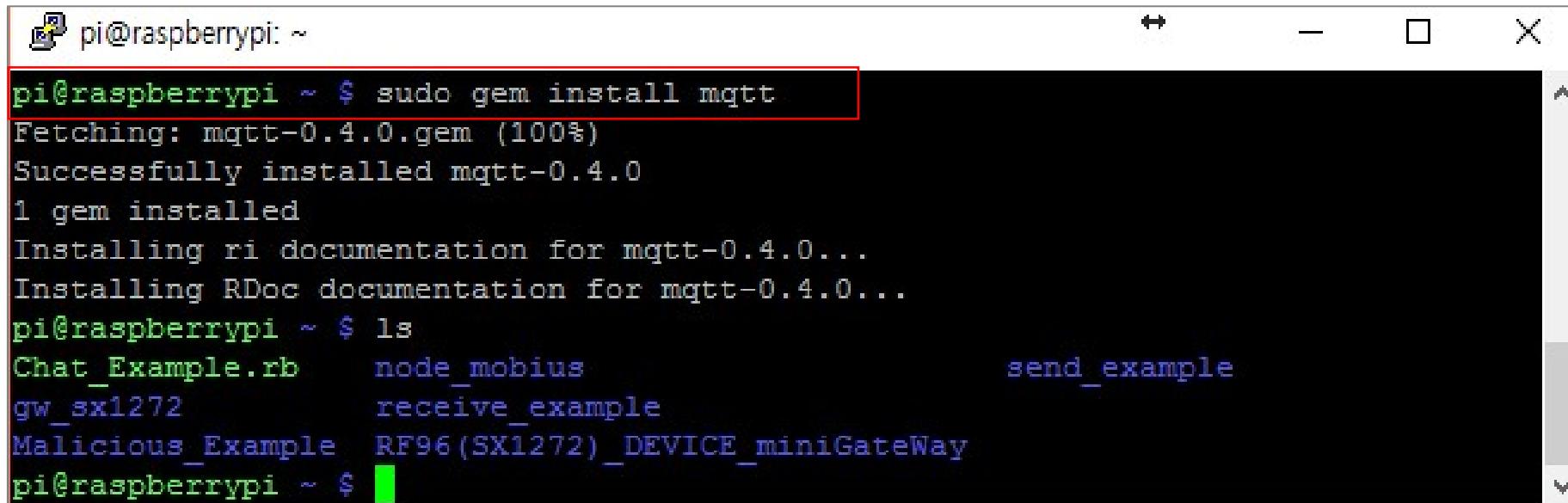
Lưu các tệp tin nguồn của Ruby bằng samba trong thư mục gốc của Pi

Network > 192.168.0.7 > pi	
name	Date Modified
gw_sx1272	2016-09-26
Malicious_Example	2016-11-24
node_mobius	2016-12-14
receive_example	2017-01-08
RF96(SX1272)_DEVICE_miniGateWay	2016-11-24
send_example	2016-11-24
Chat_Example	2017-01-18

| Kết nối Raspberry pi bằng putty hoặc màn hình VNC và sử dụng lệnh dưới đây

sudo gem install mqtt

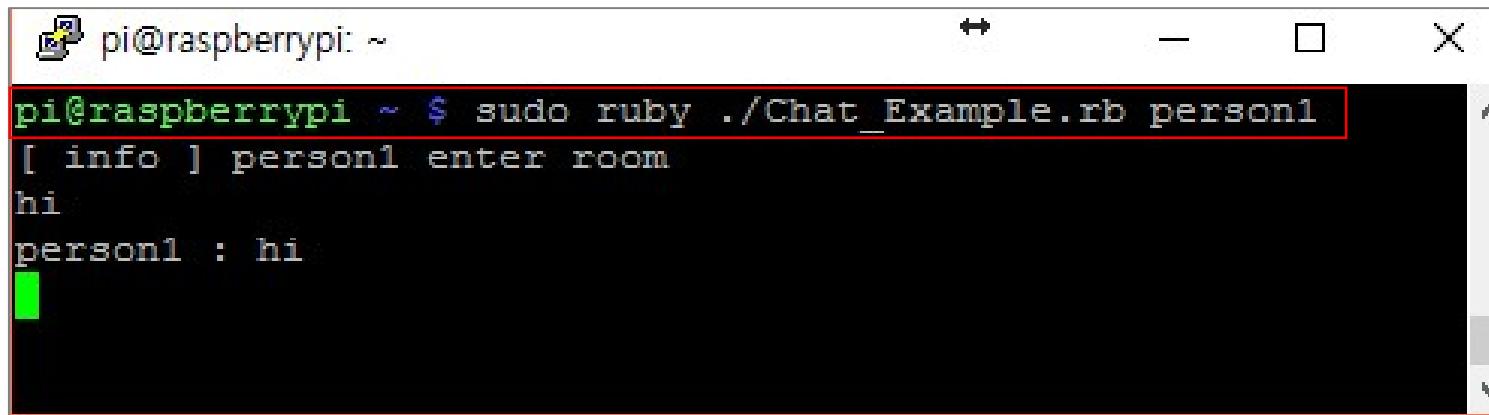
- ▶ Cài đặt thư viện MQTT từ gói cài đặt Rubygems



```
pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo gem install mqtt
Fetching: mqtt-0.4.0.gem (100%)
Successfully installed mqtt-0.4.0
1 gem installed
Installing ri documentation for mqtt-0.4.0...
Installing RDoc documentation for mqtt-0.4.0...
pi@raspberrypi ~ $ ls
Chat_Example.rb      node_mobius          send_example
gw_sx1272           receive_example
Malicious_Example   RF96(SX1272)_DEVICE_miniGateWay
pi@raspberrypi ~ $
```

| Khởi động chương trình trò chuyện MQTT bằng lệnh dưới đây

- ▶ sudo ruby ./filename username



A screenshot of a terminal window titled "pi@raspberrypi: ~". The window shows a chat session between two users:

```
pi@raspberrypi ~ $ sudo ruby ./Chat_Example.rb person1
[ info ] person1 enter room
hi
person1 : hi
```

The command `sudo ruby ./Chat_Example.rb person1` is highlighted with a red box.

Bài 2.

giao tiếp truyền thông

- | 2.1. giao tiếp là gì
- | 2.2. Mô hình OSI
- | 2.3. giao tiếp truyền thông TCP/IP
- | 2.4. HTTP
- | 2.5. MQTT
- | **2.6. CoAP**
- | 2.7. WebSocket

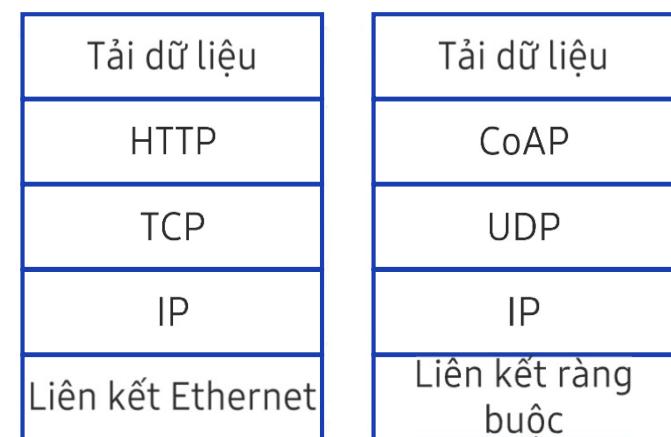
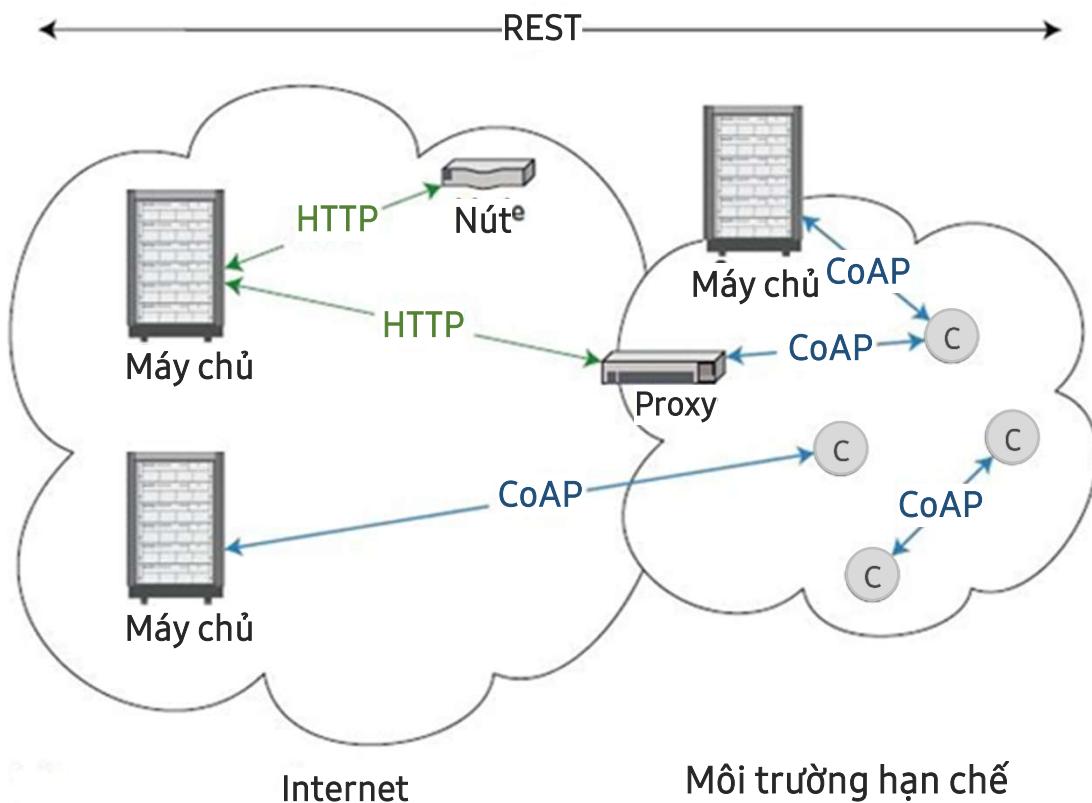
Khái niệm CoAP

- | giao tiếp ứng dụng ràng buộc (CoAP) là một giao tiếp phổ biến, hỗ trợ mô hình kết nối RESTful giữa máy chủ web và các thiết bị cảm biến hoạt động trên các phần cứng có cấu hình thấp, trong môi trường mạng với độ trễ và tỉ lệ mất dữ liệu cao.
- | Đây là một ch่อง giao tiếp gồm tầng vật lý phát triển dựa trên tiêu chuẩn IEEE 802.15.4 và tầng mạng sử dụng IPv6.

Tính năng của CoAP

- | Phương pháp tiếp cận sử dụng RESTful cho phép chuyển đổi và tích hợp dễ dàng trong chương trình đang sử dụng giao tiếp HTTP.
- | Khả năng thông điệp bị phân mảnh thấp
- | Hỗ trợ Unicast là Multicast trong môi trường UDP.
- | CoAP định nghĩa bốn kiểu thông điệp: có thể xác nhận, không thể xác nhận, báo nhận (ACK) và đặt lại (reset).
- | CoAP là một giao tiếp thay thế cho HTTP chứ không phải là phiên bản giản lược của HTTP.

Môi trường hoạt động của CoAP



Xử lý các thông điệp/bản tin CoAP

| Bản tin Confirmable (bản tin có thể xác nhận - CON)

- ▶ Được sử dụng để biết được một thông điệp hoặc phản hồi cho một thông điệp đã gửi thành công hay chưa.
- ▶ Nhận một thông điệp ACK/RST như một thông điệp phản hồi.

| Bản tin báo nhận (Acknowledgment - ACK)

- ▶ Được phân chia thành bản tin báo nhận (ACK) và bản tin định danh (MID) cho bản tin CON.
- ▶ Các bản tin ACK/RST là các bản tin phản hồi từ phía thu.

| Bản tin Reset (bản tin đặt lại - RST)

- ▶ Khi nhận được bản tin CON từ máy khách nhưng thông báo lỗi xử lý.

Ex ▶ Ví dụ: Khởi động lại nút

| Bản tin không thể xác nhận (Non-Confirmable - NON)

- ▶ Kiểu thông điệp không yêu cầu gửi phản hồi báo nhận (ACK)

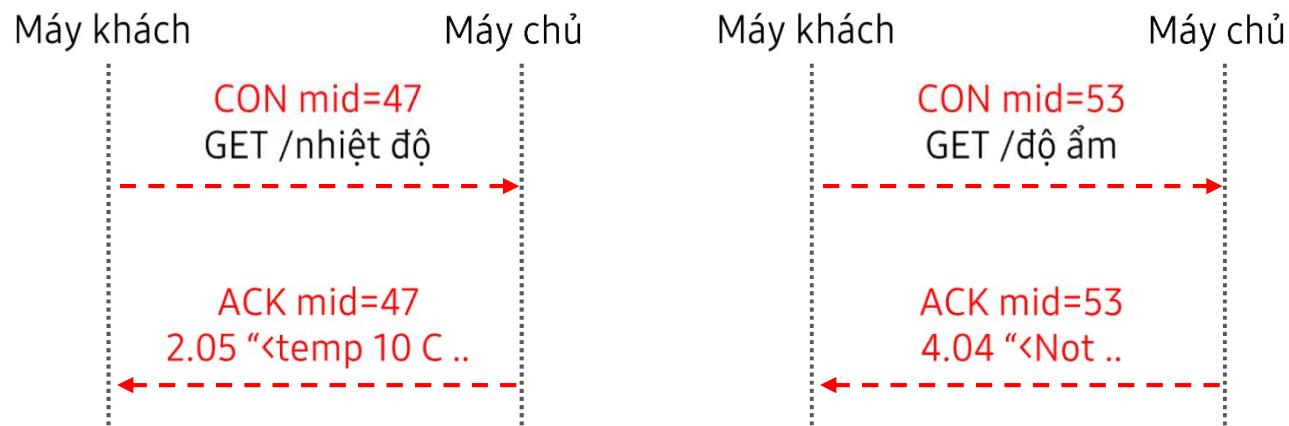
Ex Đọc giá trị lặp đi lặp lại từ cảm biến

Các phương thức trong CoAP

- | GET: nhận thông tin từ một tài nguyên được định danh qua đường dẫn URI gửi kèm theo yêu cầu
- | POST: Yêu cầu cập nhật hoặc tạo tài nguyên trên máy chủ theo đường dẫn URI trong một yêu cầu
- | PUT: Yêu cầu cập nhật mã định dạng tài nguyên với phần nội dung thông điệp theo URI trong một yêu cầu
- | DELETE: Yêu cầu xóa một mã định dạng tài nguyên tại đường dẫn URI trong yêu cầu

Trao đổi thông điệp CoAP

- | Giao tiếp qua CoAP tương tự như mô hình khách/chủ
- | Hỗ trợ URI và Content-type giống như HTTP
- | Trao đổi thông điệp cũng tương tự như giao tiếp HTTP, không cần báo nhận (ACK)
 - ▶ Máy khách khởi tạo/gửi yêu cầu
 - ▶ Máy chủ gửi phản hồi đến một tài nguyên thông qua mã định dạng tài nguyên - URI
 - ▶ Máy chủ cũng gửi kèm theo một mã phản hồi trong bản tin phản hồi



<https://tools.ietf.org/html/rfc7252>

Mô hình giao dịch CoAP

| UDP (giao tiếp gói dữ liệu người dùng)

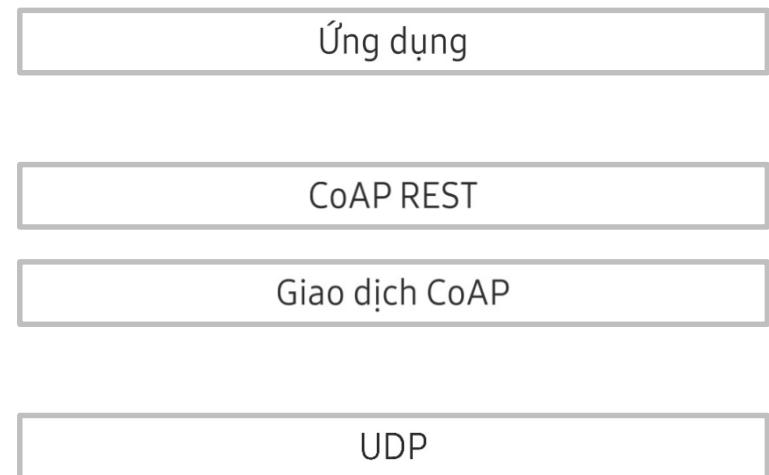
- ▶ CoAP sử dụng giao tiếp UDP để truyền thông điệp

| Giao dịch

- ▶ Một bản tin/thông điệp được gửi tới một điểm cuối
- ▶ CON, NON, ACK, RST

| REST

- ▶ Điều phối giao dịch (Transaction piggyback)
- ▶ Mã phản hồi và các tùy chọn (URI, content-type)



Định dạng thông điệp CoAP

Ver: Phiên bản,

T: Kiểu giao dịch (CON, NON, ACK...)

OC: Số lựa chọn

Code: Phương pháp hoặc mã phản hồi

Message ID: Một ID duy nhất được thiết lập bởi phía phát

Mã phản hồi thông điệp CoAP

	coap-03		coap-06	
	Code	Description	Code	Description
Request	1	GET	1	GET
	2	POST	2	POST
	3	PUT	3	PUT
	4	DELETE	4	DELETE
	40	100 Continue	-	-
	80	200 OK	69	2.05 Content
	81	201 Created	65	2.01 Created
	-	-	66	2.02 Deleted
	124	304 Not Modified	67	2.03 Valid
	-	-	68	2.04 Changed
	160	400 Bad Request	128	4.00 Bad Request
	-	-	129	4.01 Unauthorized
	-	-	130	4.02 Bad Option
	-	-	131	4.03 Forbidden
	164	404 Not Found	132	4.04 Not Found
	165	405 Method Not Allowed	133	4.05 Method Not Allowed
	-	-	141	4.13 Request Entity Too Large
	175	415 Unsupported Media Type	143	4.15 Unsupported Media Type
	200	500 Internal Server Error	160	5.00 Internal Server Error
	-	-	161	5.01 Not Implemented
	202	502 Bad Gateway	162	5.02 Bad Gateway
	203	503 Service Unavailable	163	5.03 Service Unavailable
	204	504 Gateway Timeout	164	5.04 Gateway Timeout
	-	-	165	5.05 Proxying Not Supported
	240	Token Option required	-	-
	241	Uri-Authority Option required	-	-
	242	Critical Option not supported	-	-

Source: Thomas Pötsch, ComNets, Master Thesis, 2011

Cấu trúc mã phản hồi
8 bit

```
0 1 2 3 4 5 6 7
+-+-+-----+
|class| detail |
+-+-+-----+
```

3 cấp:

- 2 – Thành công
- 4 – Lỗi máy khách
- 5 – Lỗi máy chủ

```
0 1 2 3 4 5 6 7
+-+-+-----+
|class| detail |
+-+-+-----+
|1 0 0|0 0 1 0 0|
+-+-+-----+
```

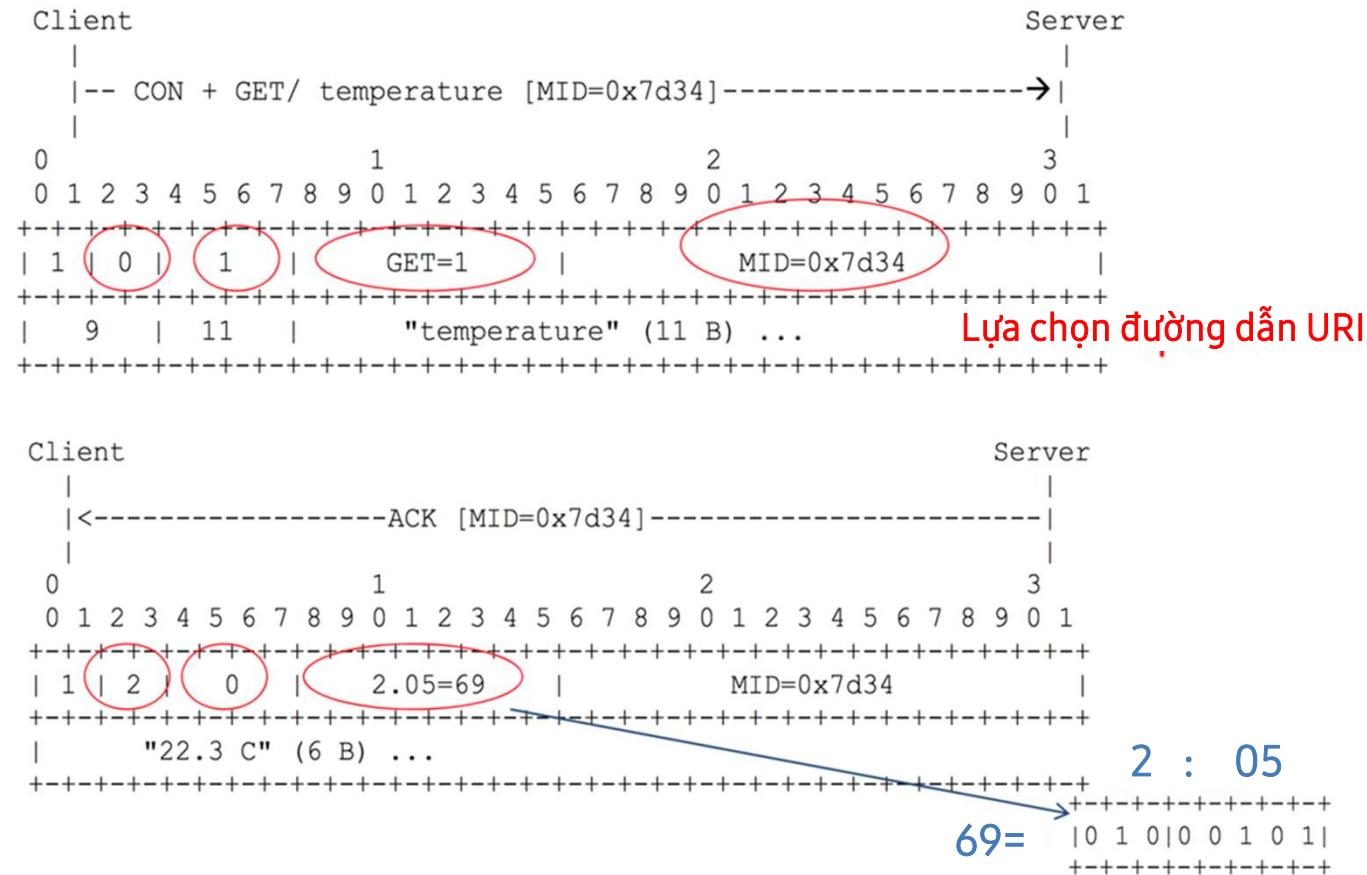
4 : 04

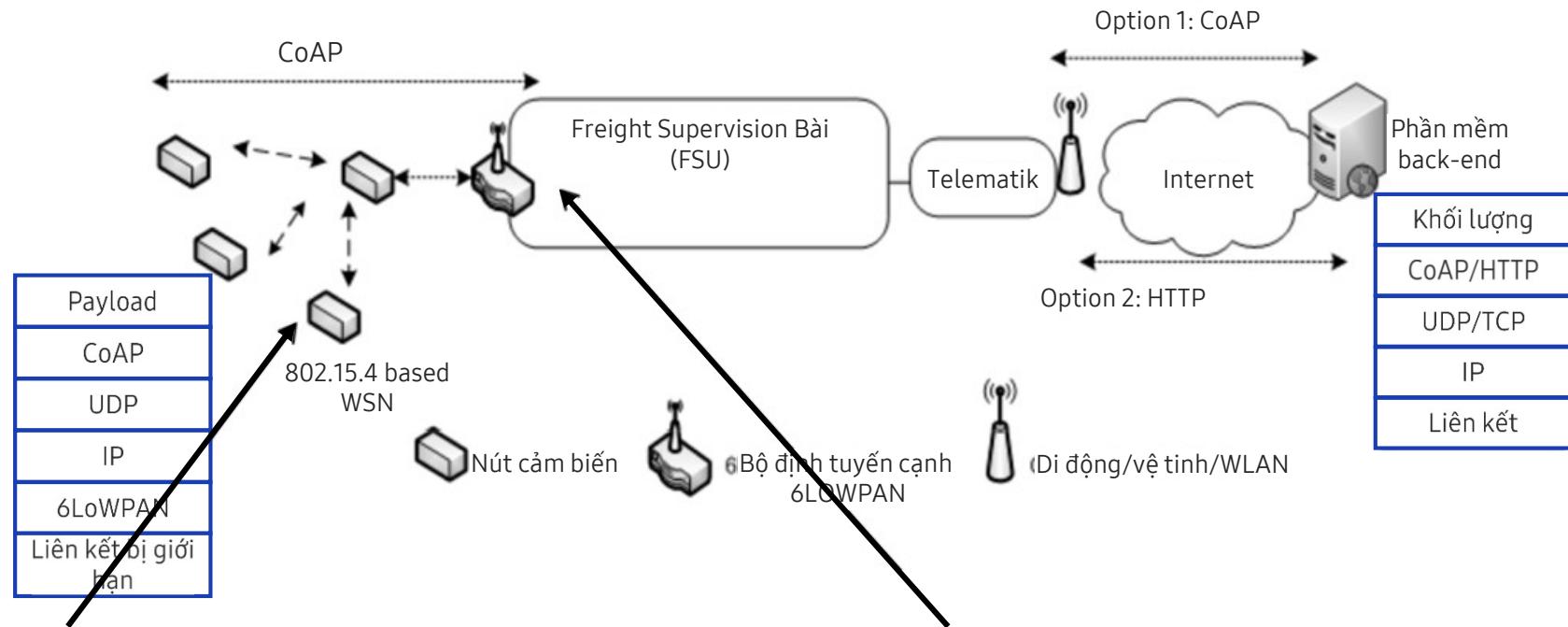
E.g., “Not Found” được viết là
4.04 – chỉ giá trị thập phân
132.

Các tùy chọn CoAP

- | Content-Type: Chỉ ra định dạng của tải dữ liệu ứng dụng
- | Max-Age: Tuổi thọ tối đa để một thông điệp được lưu trong bộ nhớ đệm
- | Proxy-Uri: Xác định một absolute Uri cho một máy chủ proxy
- | Token: Xác định một absolute Uri cho một máy chủ proxy
- | Uri-Host: Chỉ ra máy chủ Internet của một tài nguyên (chỉ được bổ sung nếu không ghi địa chỉ IP đích của yêu cầu)
- | Uri-Path: Chỉ ra tài nguyên của máy chủ
- | Uri-Port: Chỉ ra cổng của máy chủ (chỉ bổ sung nếu khác với đích của yêu cầu cổng UDP)
- | Uri-Query: Chỉ ra các tùy chọn khác cho yêu cầu
- | v.v.

Ví dụ về thông điệp và hoạt động của CoAP





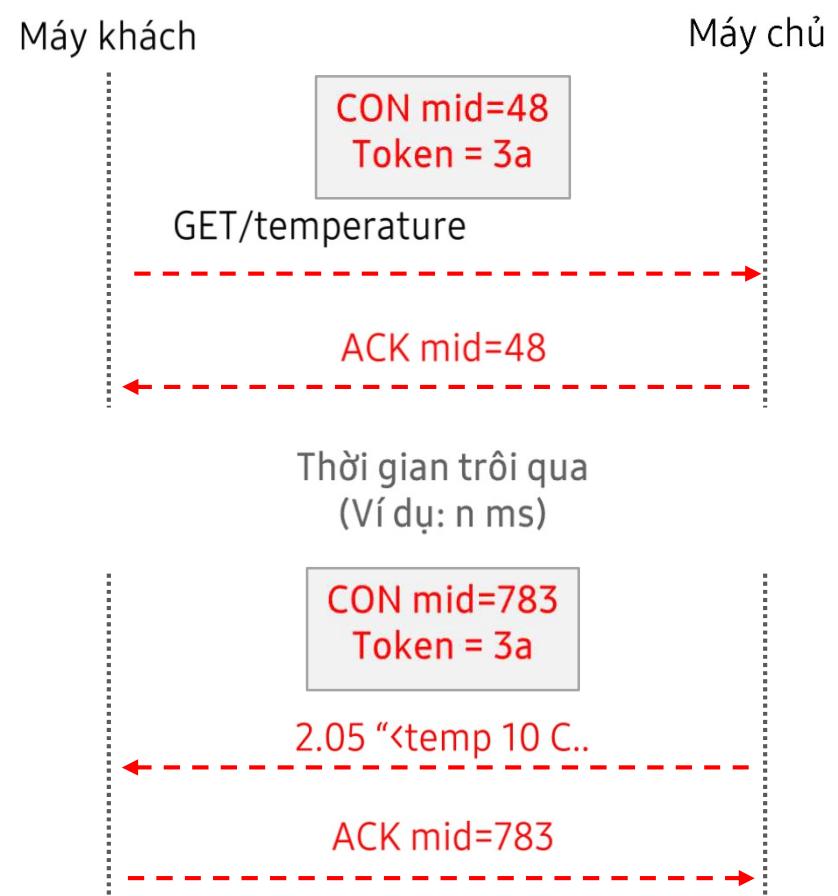
Resource	GET	PUT	Comments
/st	X		Nhiệt độ
/sh	X		Độ ẩm
/sv	X		Điện áp
/r	X		Nhiệt độ, độ ẩm Và điện áp
/l	X	X	LEDs
/ck	(X)	X	Khóa mã hóa AES

Resource	GET	PUT	Comments
/ni	X		Thông báo nút Tích hợp vào 6LoWPAN/RPL network
/warntemplow		X	Cảnh báo dưới đây Nhiệt độ thấp
/warntemphi		X	Cảnh báo trên đây Nhiệt độ cao

More details: <http://www.intelligentcontainer.com/>

CoAP và HTTP

- | Hỗ trợ truyền dữ liệu đơn hướng và đa hướng đáng tin cậy qua giao tiếp UDP
 - ▶ Truyền lại, kiểm soát tắc nghẽn đơn giản
- | Trao đổi thông điệp bất đồng bộ
- | Hai tầng con giữa tầng ứng dụng và giao tiếp UDP
- | Máy chủ cung cấp một danh sách các liên kết thông qua URI tài nguyên phổ biến.
(Khám phá tài nguyên)



| MQTT

- ▶ Kết nối với máy chủ broker trung tâm bằng các thủ tục xuất bản/đăng ký (M: N)
- ▶ Máy chủ broker nhận thông điệp và chuyển tiếp thông điệp đó cho subscriber
- ▶ Hỗ trợ kết nối liên tục, truyền dữ liệu theo thời gian thực
- ▶ Sử dụng giao tiếp lớp dưới là TCP
- ▶ Không có chức năng khám phá dịch vụ, cần hỗ trợ bởi các giao tiếp khác như DNS - SD, SSDP

| CoAP

- ▶ Kết nối giữa một máy chủ và một máy khác tham gia (1: 1)
- ▶ Không phù hợp cho giao tiếp dữ liệu dựa trên sự kiện
- ▶ Phù hợp cho các thông tin trạng thái trong một môi trường phân tán
- ▶ Chỉ sử dụng giao tiếp UDP
- ▶ Tích hợp tính năng khám phá dịch vụ

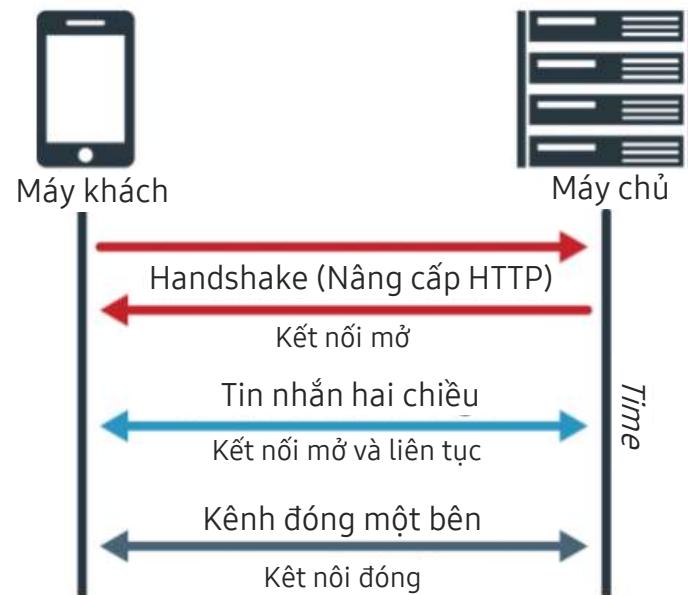
Bài 2.

giao tiếp truyền thông

- | 2.1. giao tiếp là gì
- | 2.2. Mô hình OSI
- | 2.3. giao tiếp truyền thông TCP/IP
- | 2.4. HTTP
- | 2.5. MQTT
- | 2.6. CoAP
- | **2.7. WebSocket**

Khái niệm WebSocket

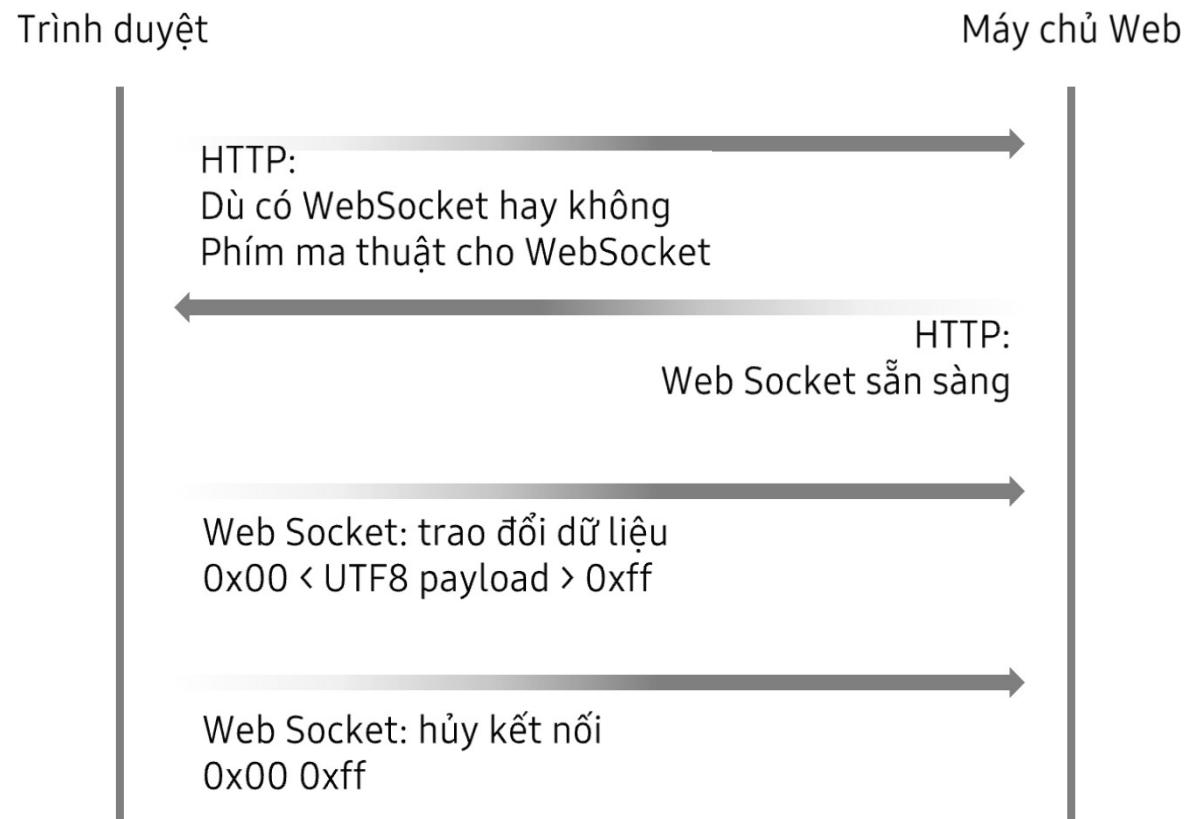
- | Là giao tiếp được phát triển để giải quyết các thiếu sót của giao tiếp HTTP
- | HTTP không duy trì kết nối. Ví dụ, khi có mươi yêu cầu được gửi đến, ở phía máy chủ để tạo ra 10 kết nối sẽ có 10 kết nối khác bị hủy bỏ. Trong tất cả các yêu cầu, phần tiêu đề luôn được đính kèm
➔ giao tiếp này không có tính tương tác theo thời gian thực.
- | Web Socket hỗ trợ giao tiếp hai chiều thời gian thực và chỉ chấp nhận các luồng thông điệp có định dạng UTF8.



Các tính năng của Web Socket

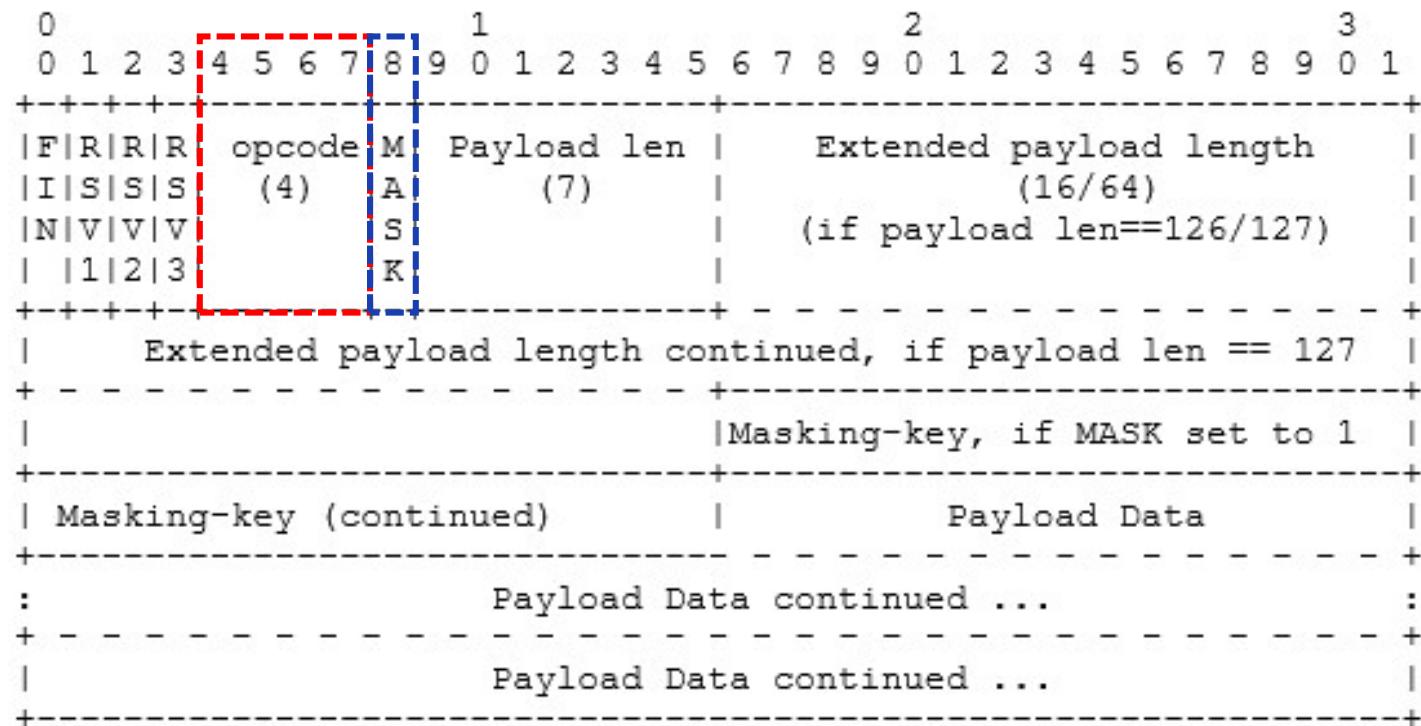
- | Giao tiếp web hai chiều tốc độ cao theo thời gian thực
- | Dễ dàng chứa được nhiều người dùng cùng một lúc
- | Tạo khung dữ liệu bằng HTML5, hỗ trợ giao tiếp bằng văn bản
- | Thay thế cho Comet, một phương pháp đẩy dữ liệu qua HTTP để xử lý dữ liệu theo thời gian thực

Quy trình WebSocket

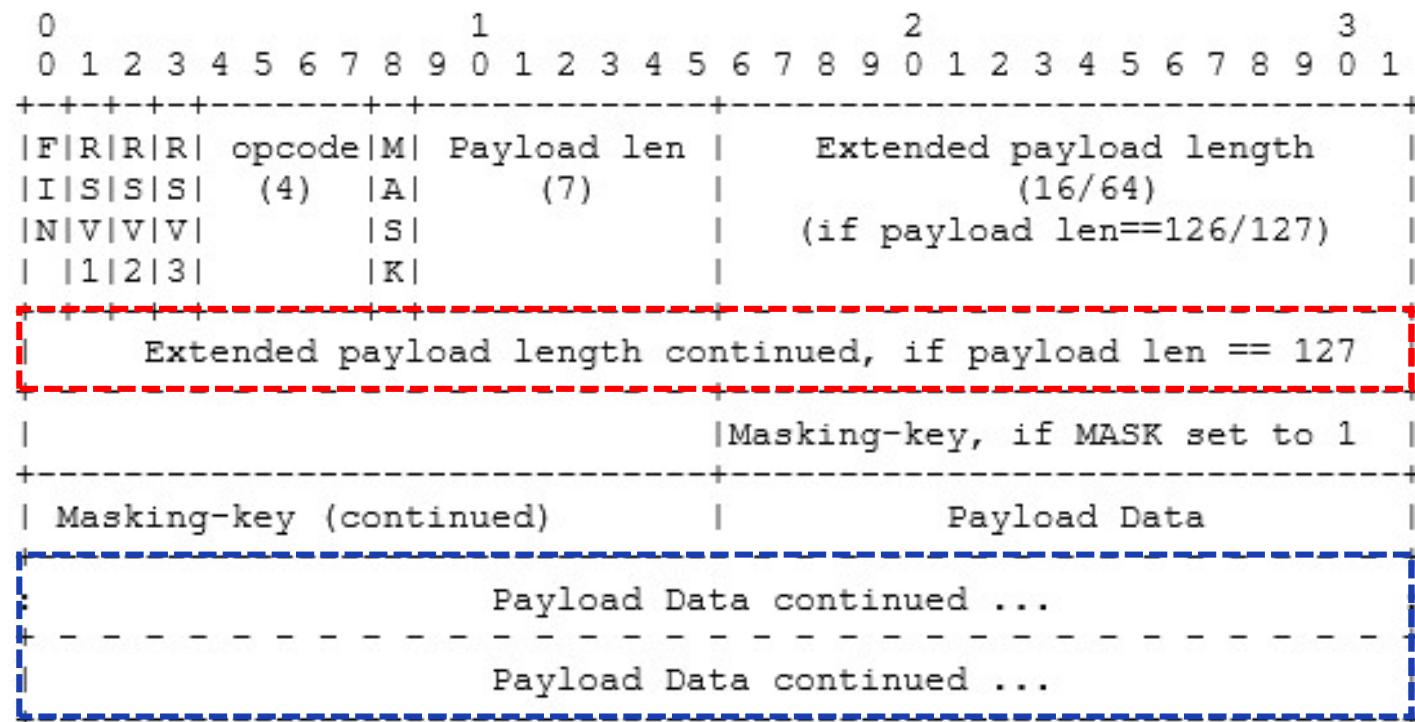


Định dạng thông điệp WebSocket

- | Mã thực thi: Trạng thái của frame hiện tại
 - | Mặt nạ: Nếu giá trị của trường tương ứng là 1, trường tải dữ liệu sẽ được XOR thành trường khóa mặt nạ.



- | Độ dài/kích thước phần tải trọng: Độ dài của dữ liệu trong phần tải trọng, dữ liệu truyền thông
- | Dữ liệu tải trọng: Dữ liệu truyền thông



Bài 3.

Lập trình Socket

| 3.1. Socket

- | 3.2. Lập trình TCP/IP Socket
- | 3.3. Lập trình UDP Socket
- | 3.4. Lập trình đa hướng

Socket là gì?

- | Socket là một điểm cuối của liên kết dữ liệu hai chiều giữa hai chương trình chạy trên các nút (máy tính) trong một mạng. Một socket ở phía máy chủ lắng nghe trên một cổng, với địa chỉ IP cụ thể, trong khi một socket khác tại phía máy khách kết nối với máy chủ để truyền thông, giao tiếp.
- | Cách thức xử lý dữ liệu truyền thông của Socket, được xác định qua hai thuộc tính:
 - ▶ Dạng thức địa chỉ, xác định giao tiếp lớp mạng được sử dụng; và
 - ▶ Loại socket xác định giao tiếp lớp vận chuyển được sử dụng.

Các loại Socket

| Tùy thuộc vào giao tiếp tầng vận chuyển được sử dụng, socket có thể là SOCK_DGRAM hoặc SOCK_STREAM.

- ▶ **SOCK_DGRAM** dùng để truyền tải gói theo **hướng thông báo**: Các socket này thường được liên kết với giao tiếp gói dữ liệu người dùng (UDP) để thực hiện việc gửi các bản tin riêng lẻ, có mức tin cậy kém. Socket datagram thường được sử dụng trong trường hợp thứ tự của các bản tin không quan trọng, chẳng hạn trường hợp cần gửi một bản tin tới nhiều máy khách.
- ▶ **SOCK_STREAM** dành cho **truyền tải theo luồng**, thường được sử dụng với liên kết sử dụng giao tiếp TCP để truyền dữ liệu: trong đó TCP cung cấp khả năng truyền dữ liệu đáng tin cậy và có trật tự giữa hai máy trạm kết nối, kết hợp với việc xử lý và kiểm soát lỗi, phù hợp với các ứng dụng liên quan đến việc truyền một lượng lớn dữ liệu, yêu cầu chính xác.

Mô-đun Socket

| Để sử dụng socket trong Python, cần khai báo module/gói thư viện socket.

Import socket

- ▶ Cách thức cơ bản để sử dụng module socket là thông qua hàm `sockets()`, hàm trả về một đối tượng socket với các phương thức (hàm) để thực hiện các lệnh đối với socket hệ thống. Trong Python, Socket hỗ trợ một số họ địa chỉ tùy theo giao tiếp lớp mạng, gồm:
 - `AF_INET`: đây là loại địa chỉ phổ biến nhất, sử dụng IPv4 để đánh địa chỉ mạng. Hầu hết các mạng internet hiện nay vẫn đang sử dụng IPv4
 - `AF_INET6`: đây là thế hệ tiếp theo của giao tiếp internet sử dụng IPv6 và cung cấp một số tính năng không có trong IPv4.
 - `AF_UNIX`: đây là họ địa chỉ cho Unix Domain Sockets (UDS), một giao tiếp liên lạc giữa các quá trình có sẵn trên các hệ thống tuân thủ POSIX. Việc triển khai này cho phép truyền dữ liệu giữa các quy trình trên một hệ điều hành mà không cần thông qua mạng.

Các dịch vụ liên quan đến mạng

- | Các chức năng và phương thức của socket cung cấp quyền truy cập vào các tác vụ liên quan đến mạng
 - ▶ Hàm `gethostname()`: để lấy về tên của máy chủ hiện tại

```
pi@raspberrypi:~ $ python
Python 2.7.16 (default, Oct 10 2019, 22:02:15)
[GCC 8.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import socket
>>> print(socket.gethostname())
raspberrypi
>>> █
```

- ▶ Hàm `gethostbyname()`: để chuyển đổi tên của máy chủ thành địa chỉ IP của nó bằng cách tham khảo cấu hình DNS của hệ điều hành

```
>>> print(socket.gethostbyname("google.com"))
142.250.199.110
>>> █
```

| Các chức năng và phương thức của socket cung cấp quyền truy cập vào các tác vụ liên quan đến mạng

- ▶ Hàm `gethostname_ex()`: để lấy về các thông tin về tên, địa chỉ của máy chủ

```
>>> name, aliases, addresses = [socket.gethostname_ex("google.com")]
>>> print("Name : ", name)
('Name : ', 'google.com')
>>> print("Aliases : ", aliases)
('Aliases : ', [])
>>> print("Addresses : ", addresses)
('Addresses : ', ['172.217.31.174'])
>>> █
```

- ▶ Hàm `gethostbyaddr()`: để thực hiện tra cứu ngược tên miền từ địa chỉ IP của máy chủ

```
>>> [socket.gethostbyaddr("172.217.31.174")]
('nrt12s22-in-f14.1e100.net', [], ['172.217.31.174'])
>>> █
```

| Các chức năng và phương thức của socket cung cấp quyền truy cập vào các tác vụ liên quan đến mạng

- ▶ Hàm getfqdn(): để biết thêm thông tin đặt tên về máy chủ

```
>>> socket.getfqdn("www.google.com")
'nrt12s28-in-t4.1e100.net'
>>> socket.getfqdn()
'raspberrypi'
>>>
```

- ▶ Có thể truy cập nhiều chức năng liên quan đến mạng, tham khảo qua trang thông tin:
<https://docs.python.org/3/library/socket.html>

Bài 3.

Lập trình Socket

- | 3.1. Socket
- | **3.2. Lập trình TCP/IP Socket**
- | 3.3. Lập trình UDP Socket
- | 3.4. Lập trình đa hướng

Giao tiếp máy khách-máy chủ TCP/IP

- | Socket có thể được cấu hình để hoạt động như một máy chủ hoặc máy khách, để có thể giao tiếp hai chiều thông qua TCP, chương trình sử dụng socket kiểu **SOCK_STREAM**.
- | Chúng tôi sẽ triển khai một **ứng dụng echo** đơn giản, có chức năng nhận tất cả dữ liệu đến và gửi chúng trở lại người gửi. Trong chương trình sẽ triển khai cả socket máy khách và máy chủ. Ngoài ra, chương trình sẽ sử dụng địa chỉ loopback cục bộ, có giá trị là 127.0.0.1 hoặc localhost, cho các kết nối mạng của chương trình.

Chương trình Echo tại máy chủ

| Trong chương trình trên máy chủ, cần thực hiện trình tự các phương thức `socket()`, `bind()`, `listen()` và `accept()`.

- ▶ `socket()`: hàm tạo một đối tượng socket mới với họ địa chỉ và loại socket.
- ▶ `bind()`: hàm liên kết đối tượng socket của chương trình với một địa chỉ cụ thể bao gồm đíc chỉ máy chủ và địa chỉ cổng.
- ▶ `listen()`: hàm cho phép máy chủ bắt đầu chờ các yêu cầu kết nối từ máy khách, hàm trả về số lượng kết nối không được chấp nhận trước đó.
- ▶ `accept()`: hàm chấp nhận các kết nối đến và trả về một bộ tham số (`conn`, `địa chỉ`) trong đó `conn` là một socket mới có thể được sử dụng để gửi và nhận dữ liệu từ kết nối và `địa chỉ` là của đối tượng socket ở đầu kia trên kết nối.
- ▶ `close()`: hàm giải phóng/đóng socket, không thực hiện kết nối nữa.

| ~/rasp_ex/socket/echo_server.py (1)

- ▶ Dòng lệnh 4: Khởi tạo một đối tượng socket, thông qua hàm khởi tạo `sock()`, với các tham số cho hàm gồm: họ địa chỉ (`socket.AF_INET`), loại socket (`socket.SOCK_STREAM`).
- ▶ Dòng lệnh 7-9: Liên kết đối tượng socket với địa chỉ theo định dạng ('192.168.0.3', 999). Sử dụng phương thức `bind()` để liên kết socket tại địa chỉ 192.168.0.3 trên cổng 999.
(giả thiết 192.168.0.3 là địa chỉ IP của Raspberry Pi.)
- ▶ Dòng lệnh 12: Thực hiện lắng nghe, chờ các yêu cầu kết nối đến máy chủ, với số lượng kết nối đến là 1.

```
1 import socket
2
3 # Create a TCP/IP socket
4 sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5
6 # Bind the socket to the port
7 server_address = ('192.168.0.3', 999)
8 print('Starting up on {} port {}'.format(*server_address))
9 sock.bind(server_address)
10
11 # Listen for incoming connections
12 sock.listen(1)
```

| ~/rasp_ex/socket/echo_server.py (2)

- ▶ Gọi hàm `recv(16)` của đối tượng `connection` để lấy về dữ liệu nhận về trên kết nối với độ dài tối đa là 16 (byte).
- ▶ Nếu có dữ liệu được nhận về, chương trình sẽ thực hiện gửi phản hồi cho phía gửi bằng cách gọi phương thức `sendall(data)` của đối tượng **kết nối**, nếu không, chương trình sẽ xuất ra thông điệp thông báo không có dữ liệu được nhận.
- ▶ Cuối đoạn mã lệnh, khi quá trình giao tiếp với máy khách hoàn tất (tất cả các đoạn tin nhắn đã được truyền đi), chương trình gọi hàm `close()` của đối tượng `connection`.
- ▶ Trong đoạn mã lệnh, có thể sử dụng cấu trúc `try/finally` để đảm bảo rằng hàm `close()` được gọi trong cả trường hợp xảy ra lỗi khi gửi dữ liệu.

```

13
14     while True:
15         # Wait for a connection
16         print('waiting for a connection')
17         connection, client_address = sock.accept()
18     try:
19         print('connection from', client_address)
20
21         # Receive the data in small chunks and retransmit it
22         while True:
23             data = connection.recv(16) # Line 23
24             print('received {!r}'.format(data))
25             if data:
26                 print('sending data back to the client')
27                 connection.sendall(data) # Line 27
28             else:
29                 print('no data from', client_address)
30                 break
31
32     finally:
33         # Clean up the connection
34         print("Closing current connection")
35         connection.close() # Line 35

```

Máy khách Echo

I echo_client.py (1)

- ▶ Khác với chương trình trên máy chủ, chương trình trên máy khách chỉ cần thực hiện chuỗi các hàm socket() và connect().
- ▶ Khởi tạo một đối tượng socket tương tự chương trình trên máy chủ.
- ▶ Tiếp theo, kết nối socket với máy chủ đang thực hiện lắng nghe tại địa chỉ ('192.168.0.3', 999), sử dụng phương connect() với tham số chuyển vào hàm là địa chỉ của máy chủ.

(Giả thiết địa chỉ IP của Raspberry Pi trong trường hợp này là 192.168.0.3)

```
1 import socket  
2  
3 # Create a TCP/IP socket  
4 sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
5  
6 # Connect the socket to the port where the server is listening  
7 server_address = ('192.168.0.3', 999)  
8 print('connecting to {} port {}'.format(*server_address))  
9 sock.connect(server_address)
```

Máy khách Echo

I echo_client.py (2)

- Trong cấu trúc `try/finally` của chương trình, thiết lập giá trị cho tin nhắn là một chuỗi byte, và sử dụng phương thức `sendall()` của đối tượng `sock` để gửi tin nhắn này đi, biến tin nhắn làm đối số của hàm này.
- Sau đó, thiết lập các biến `amount_received` với giá trị ban đầu là 0 và biến `amount_expected` mang giá trị là độ dài của tin nhắn được gửi đi, để theo dõi việc nhận lại bản tin này.

```
11  try:  
12      # Send data  
13      message = b'It is very long message but will only be transmitted in chunks of 16 at a time'  
14      print('sending {!r}'.format(message))  
15      sock.sendall(message)  
16  
17      # Look for the response  
18      amount_received = 0  
19      amount_expected = len(message)  
20  
21      while amount_received < amount_expected:  
22          data = sock.recv(16)  
23          amount_received += len(data)  
24          print('received {!r}'.format(data))  
25  
26  finally:  
27      print('closing socket')  
28      sock.close()
```

Máy khách Echo

I echo_client.py (3)

- ▶ Gọi hàm `recv(16)` của đối tượng `sock` để nhận dữ liệu được gửi phản hồi lại từ máy chủ, với độ dài dữ liệu đọc về có kích thước tối đa là 16 byte. Quá trình đọc dữ liệu sẽ tiếp tục được thực hiện trong vòng lặp cho đến khi số byte dữ liệu nhận về bằng (hoặc lớn hơn) kích thước bản tin đã được gửi đi.
- ▶ Cuối cùng, sử dụng hàm `close()` để đóng kết nối trên đối tượng `sock`.

```
11  try:  
12      # Send data  
13      message = b'It is very long message but will only be transmitted in chunks of 16 at a time'  
14      print('sending {!r}'.format(message))  
15      sock.sendall(message)  
16  
17      # Look for the response  
18      amount_received = 0  
19      amount_expected = len(message)  
20  
21      while amount_received < amount_expected:  
22          data = sock.recv(16)  
23          amount_received += len(data)  
24          print('received {!r}'.format(data))  
25  
26  finally:  
27      print('closing socket')  
28      sock.close()
```

Chạy chương trình máy chủ

Mở chương trình Terminal trên Raspberry Pi, nhập lệnh để chạy chương trình

- ▶ sudo python3 ~/rasp_ex/socket/echo_server.py
- ▶ Khi chương trình trên máy chủ được chạy, nó sẽ đợi máy khách kết nối

```
> sudo python3 ~/rasp_ex/socket/echo_server_udp.py
```

```
pi@raspberrypi:~ $ sudo python3 rasp_ex/socket/echo_server.py
Starting up on 192.168.0.3 port 999
waiting for a connection
```

I Chạy chương trình máy khách trên Windows (1)

- ▶ Bật cmd prompt hoặc sử dụng PyCharm để chạy chương trình echo_client.py

(Trong ví dụ này, echo_client.py được thực thi bằng PyCharm trong Windows.)

```
C:\Users\dongh\anaconda3\python.exe C:/Users/dongh/Desktop/echo_client.py
connecting to 192.168.0.3 port 999
sending b'It is very long message but will only be transmitted in chunks of 16 at a time'
received b'It is very long '
received b'message but will'
received b' only be transmi'
received b'tted in chunks o'
received b'f 16 at a time'
closing socket

Process finished with exit code 0
```

I Windows (2)

- ▶ Kiểm tra kết nối và giao tiếp giữa chương trình trên máy chủ Raspberry Pi và chương trình máy trạm trên Windows.
- ▶ Máy chủ Raspberry Pi nhận được thông báo khi giao tiếp socket TCP/IP với chương trình chạy trên máy khách Windows.

```
pi@raspberrypi:~ $ sudo python3 rasp_ex/socket/echo_server.py
Starting up on 192.168.0.3 port 999
waiting for a connection
connection from ('192.168.0.11', 56622)
received b'It is very long '
sending data back to the client
received b'message but will'
sending data back to the client
received b' only be transmi'
sending data back to the client
received b'tted in chunks o'
sending data back to the client
received b'f 16 at a time'
sending data back to the client
received b''
no data from ('192.168.0.11', 56622)
closing current connection
waiting for a connection
```

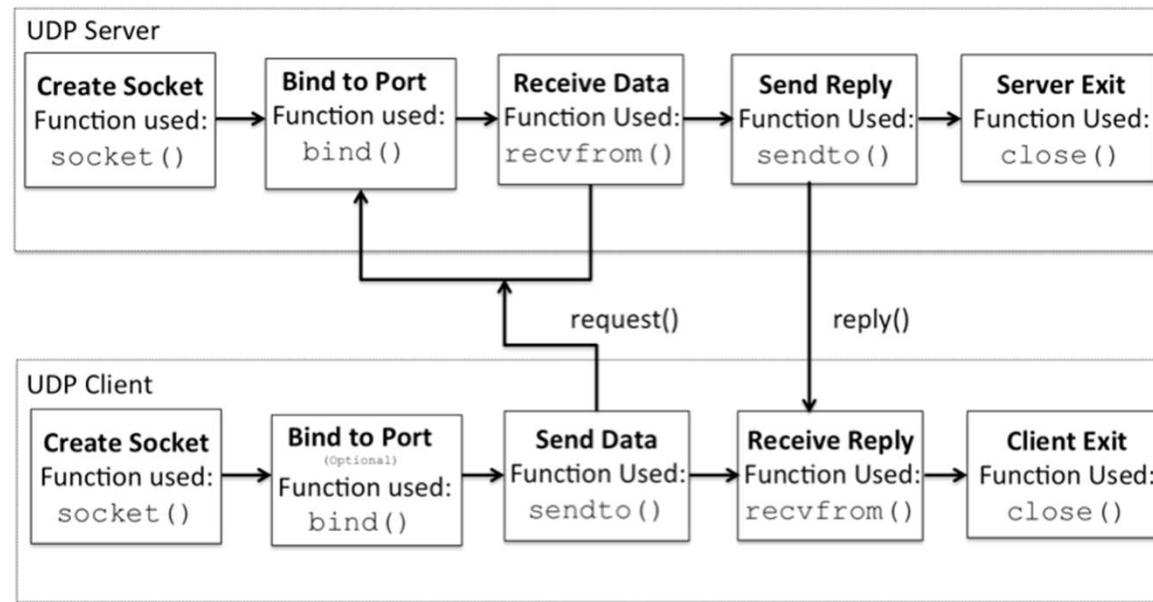
Bài 3.

Socket Programming

- | 3.1. Socket
- | 3.2. Lập trình TCP/IP Socket
- | **3.3. Lập trình UDP Socket**
- | 3.4. Lập trình đa hướng

Giao tiếp máy khách-máy chủ qua UDP

- | Không giống với trường hợp truyền thông điệp thông qua giao tiếp TCP, với giao tiếp UDP không yêu cầu thiết lập kết nối giữa hai phía (máy chủ và máy khách).
- | Một thông báo (bản tin) trong trường hợp này được đóng gói trong một gói dữ liệu và được gửi đi, nhưng không được đảm bảo (có thể bị mất hoặc bị lỗi)



Chương trình Echo trên máy chủ

- | Chương trình chỉ thực thi tuần tự các hàm `socket()` và `bind()`, không cần lắng nghe (hay chờ) kết nối.
 - ▶ Trong đó, chương trình chỉ cần liên kết đối tượng socket với một địa chỉ cụ thể và đợi tin nhắn được gửi đến.
 - ▶ Sau đó, chương trình sẽ đọc các tin nhắn nhận được bằng phương thức `recvfrom()` và gửi lại chúng bằng hàm `sendto()`.
 - ▶ Hàm `recvfrom()` có chức năng nhận dữ liệu từ một socket và trả về cặp giá trị (`byte, address`), trong đó `byte` là đối tượng chứa dữ liệu nhận được và `address` là địa chỉ của phía gửi dữ liệu tới.
 - ▶ Hàm `sendto(byte, address)` gửi dữ liệu (được lưu trong tham số `byte`) đến một socket từ xa, với địa chỉ xác định thông qua tham số `address`.

| ~/rasp_ex/socket/echo_server_udp.py

- ▶ Dòng lệnh 4: Tạo một đối tượng socket `socket.socket(socket.AF_INET,socket.SOCK_DGRAM)`.
 - ▶ Ở đây, ta sử dụng đối tượng socket kiểu `SOCKET_DGRAM` để kết nối, truyền thông qua giao tiếp UDP.
- ▶ Dòng lệnh 7-9: Liên kết socket với cặp giá trị địa chỉ ('192.168.0.3', 999) và chờ tin nhắn gửi đến.
(giả thiết 192.168.0.3 là địa chỉ IP của máy chủ Raspberry)
- ▶ Dòng lệnh 13-17: Khi nhận được một tin nhắn, lấy về (đọc) tin nhắn bằng hàm `recvfrom(4096)`, trong đó 4096 là số byte tối đa sẽ được đọc. Hàm này trả về **dữ liệu và địa chỉ**. Tiếp theo, in/xuất thông tin về độ dài và địa chỉ ra màn hình.
- ▶ Dòng lệnh 20: Nếu nhận được dữ liệu (tin nhắn), chương trình sẽ gửi lại phía phát bằng hàm `sendto()`, đồng thời in ra thông tin dữ liệu và địa chỉ vừa gửi.

```

1 import socket
2
3 # Create a UDP socket
4 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5
6 # Bind the socket to the port
7 server_address = ('192.168.0.3', 999)
8 print('starting up on {} port {}'.format(*server_address))
9 sock.bind(server_address)
10
11 while True:
12     print('\nwaiting to receive message')
13     data, address = sock.recvfrom(4096)
14
15     print('received {} bytes from {}'.format(
16         len(data), address))
17     print(data)
18
19     if data:
20         sent = sock.sendto(data, address)
21         print('sent {} bytes back to {}'.format(
22             sent, address))

```

Chương trình Echo trên máy khách

echo_client_udp.py

- ▶ Chương trình trên máy khách tương tự như chương trình trên máy chủ, chỉ khác là nó không liên kết đối tượng socket với bất kỳ địa chỉ nào. Thay vào đó, chương trình gọi hàm sendto() để gửi tin nhắn đến máy chủ theo địa chỉ IP và địa chỉ cổng (của máy chủ).
- ▶ Dòng lệnh 4: Khởi tạo một đối tượng socket, tương tự như ở phía máy chủ.
- ▶ Dòng lệnh 6-7: Thiết lập một bản tin dưới dạng chuỗi byte và thiết lập thông tin địa chỉ của máy chủ muốn gửi bản tin tới, thông qua biến server_address, bao gồm địa chỉ IP và địa chỉ cổng.
- ▶ Bên trong khối lệnh try/finally, thực hiện gửi bản tin và chờ phản hồi từ phía máy chủ, xuất thông tin ra màn hình trong các bước xử lý.
- ▶ Dòng lệnh 21: Đóng socket để ngắt/dừng kết nối.

```
1 import socket
2
3 # Create a UDP socket
4 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5
6 server_address = ('192.168.0.3', 999)
7 message = b'This is our message. It will be sent all at once'
8
9 try:
10     # Send data
11     print('sending {!r}'.format(message))
12     sent = sock.sendto(message, server_address)
13
14     # Receive response
15     print('waiting to receive')
16     data, server = sock.recvfrom(4096)
17     print('received {!r}'.format(data))
18
19 finally:
20     print('closing socket')
21     sock.close()
```

Thực hiện chương trình tại phía máy chủ

- | Trên máy chủ, Raspberry Pi, chạy chương trình Terminal và nhập lệnh để chạy chương trình
 - ▶ sudo python3 ~/rasp_ex/socket/echo_server_udp.py
 - ▶ Đợi máy khách kết nối, gửi dữ liệu tới

```
pi@raspberrypi:~ $ sudo python3 rasp_ex/socket/echo_server_udp.py
starting up on 192.168.0.3 port 999
waiting to receive message
```

| Chạy chương trình máy khách trên Windows (1)

- ▶ Mở Cmd Prompt hoặc PyCharm để chạy mã lệnh Python echo_client_udp.py
 - ▶ (Ví dụ sử dụng PyCharm để chạy echo_client_udp.py trên Windows.)

```
C:\Users\dongh\anaconda3\python.exe C:/Users/dongh/Desktop/echo_client_udp.py
sending b'This is our message. It will be sent all at once'
waiting to receive
received b'This is our message. It will be sent all at once'
closing socket
```

```
Process finished with exit code 0
```

| Windows (2)

- ▶ Kiểm tra giao tiếp giữa máy chủ Raspberry Pi và máy khách Windows.
- ▶ Máy chủ Raspberry Pi nhận được một bản tin truyền qua giao tiếp UDP, từ chương trình máy khách chạy trên Windows.

```
pi@raspberrypi:~ $ sudo python3 rasp_ex/socket/echo_server_udp.py
starting up on 192.168.0.3 port 999
```

```
waiting to receive message
received 48 bytes from ('192.168.0.11', 57179)
b'This is our message. It will be sent all at once'
sent 48 bytes back to ('192.168.0.11', 57179)
```

```
waiting to receive message
```

Bài 3.

Socket Programming

- | 3.1. Socket
- | 3.2. Lập trình TCP/IP Socket
- | 3.3. Lập trình UDP Socket
- | **3.4. Lập trình đa hướng**

Truyền thông đa hướng – Truyền thông quảng bá

I Xử lý đa kết nối

- ▶ Khi xử lý nhiều máy khách, việc duy trì một số kết nối điểm-điểm có thể gây khó khăn cho các ứng dụng do nhu cầu xử lý và băng thông tăng lên. Với trường hợp này, tin nhắn đền từ các hướng khác nhau.
- ▶ Với phát đa hướng (quảng bá) - multicast, tin nhắn đồng thời được gửi đến nhiều điểm cuối.
- ▶ Phương pháp này đạt được hiệu quả cao hơn do việc gửi tin nhắn đến tất cả người nhận được ủy quyền cho cơ sở hạ tầng mạng.
- ▶ Các tin nhắn quảng bá được gửi bằng UDP, vì giao tiếp TCP chỉ thực hiện trên liên kết điểm-điểm.
- ▶ Địa chỉ dành cho tin nhắn phát quảng bá, được xác định là nhóm phát đa hướng (quảng bá), là một tập hợp con trong dải địa chỉ IPv4, thường nằm trong khoảng từ 224.0.0.0 đến 239.255.255.255.
- ▶ Các bộ định tuyến và chuyển mạch mạng coi các địa chỉ trong dải này là giá trị đặc biệt vì chúng được dành riêng cho phát đa hướng, đảm bảo các bản tin được phân phối tới tất cả các máy khách trong một nhóm/một mạng cục bộ.

Gửi tin nhắn đa hướng

| Để gửi tin nhắn, chương trình sử dụng phương thức `sendto()` với địa chỉ chỉ định nhóm multicast

- ▶ Trong chương trình, cần chỉ định giá trị “Thời gian tồn tại” (TTL) để xác định “số bước” tối đa mà bản tin được gửi đi trong mạng.
- ▶ Tham số TTL có giá trị mặc định là 1, sẽ dẫn đến các tin nhắn chỉ được gửi đến các máy chủ trong mạng cục bộ.
- ▶ Sử dụng hàm `setsockopt()` để thiết lập các tham số cho đối tượng socket , trong đó với tùy chọn `IP_MULTICAST_TTL`, và giá trị cho tham số `TTL` được thiết lập là một byte trong gói tin.
- ▶ Chương trình thiết lập lại giá trị TTL của đối tượng socket, để tránh socket chờ phản hồi vô thời hạn, không thể xác định được có bao nhiêu phản hồi cần nhận từ mạng.

I multicast_sender.py (1)

- ▶ Dòng lệnh 4-8: Tạo một socket loại `socket.SOCK_DGRAM`, thiết lập một bản tin dưới dạng chuỗi byte và liên kết đối tượng socket của chương trình với cặp địa chỉ phát đa hướng ('224.10.10.10', 10000).
- ▶ Dòng lệnh 12: đặt lại thời gian chờ là 0,2 , sử dụng hàm `sock.settimeout(0.2)`
- ▶ Dòng lệnh 16: thiết lập biến ttl kiểu cấu trúc (`struct`), đóng gói giá trị 1 thành một byte.
- ▶ Dòng lệnh 17: Sử dụng hàm `setsockopt()`, để thiết lập thuộc tính cho đối tượng socket, thiết lập lại tham số TTL.

```
1 import socket
2 import struct
3
4 message = b'very important data'
5 multicast_group = ('224.10.10.10', 10000)
6
7 # Create the datagram socket
8 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
9
10 # Set a timeout so the socket does not block
11 # indefinitely when trying to receive data.
12 sock.settimeout(0.2)
13
14 # Set the time-to-live for messages to 1 so they do not
15 # go past the local network segment.
16 ttl = struct.pack('b', 1)
17 sock.setsockopt(socket.IPPROTO_IP, socket.IP_MULTICAST_TTL, ttl)
```

I multicast_sender.py (2)

```
19     try:
20
21         # Send data to the multicast group
22         print('sending {!r}'.format(message))
23         sent = sock.sendto(message, multicast_group)
24
25         # Look for responses from all recipients
26         while True:
27             print('waiting to receive')
28             try:
29                 data, server = sock.recvfrom(16)
30             except socket.timeout:
31                 print('timed out, no more responses')
32                 break
33             else:
34                 print('received {!r} from {}'.format(
35                     data, server))
36
37     finally:
38         print('closing socket')
39         sock.close()
```

- ▶ Dòng lệnh 23: Gửi tin nhắn quảng bá bằng lệnh sendto()
- ▶ Sau khi phát tin nhắn đa hướng, chương trình sẽ chờ phản hồi từ các máy khác trong mạng trong khoảng thời gian chờ đã được thiết lập (ở phần đầu mã lệnh), nếu có phản hồi thông tin phản hồi sẽ được in/xuất ra màn hình.

Nhận tin nhắn đa hướng – chương trình ở phía nhận

- | Trong chương trình ở phía nhận tin nhắn, sau khi tạo một đối tượng socket và liên kết nó với một cổng, ta cần thêm nó vào nhóm phát đa hướng.
- | Để thực hiện, chương trình sử dụng hàm `setsockopt()` để thiết lập tùy chọn là `IP_ADD_MEMBERSHIP`, tùy chọn này được đóng gói trong 8 byte biểu diễn nhóm phát đa hướng và giao diện mạng ở phía máy chủ đang thực hiện lắng nghe các kết nối.
- | Hàm `socket.inet_aton()` nên được sử dụng để chuyển đổi địa chỉ IPv4 của nhóm phát đa hướng từ định dạng ('224.10.10.10') sang định dạng chuỗi nhị phân 32 bit.

I multicast_receiver.py (1)

```
1 import socket
2 import struct
3 import sys
4
5 multicast_group = '224.10.10.10'
6 server_address = ('', 10000)
7
8 # Create the socket
9 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
10
11 # Bind to the server address
12 sock.bind(server_address)
13
14 # Tell the operating system to add the socket to
15 # the multicast group on all interfaces.
16 group = socket.inet_aton(multicast_group)
17 mreq = struct.pack('4sL', group, socket.INADDR_ANY)
18 sock.setsockopt(
19     socket.IPPROTO_IP,
20     socket.IP_ADD_MEMBERSHIP,
21     mreq)
```

- ▶ Dòng lệnh 16-21: Để thêm đối tượng socket vào nhóm phát đa hướng, sử dụng `struct` để đóng gói địa chỉ nhóm, sử dụng cho hàm `socket.inet_aton(multicast_group)`, và giao diện mạng xác định bởi `socket.INADDR_ANY` thành một khối 8 byte. Sau đó, chương trình sẽ thiết lập đối tượng socket là thành viên của nhóm, chỉ định qua tham số `IP_ADD_MEMBERSHIP` của socket, thực hiện bởi hàm `setsockopt()`

I multicast_receiver.py (2)

- ▶ Khi đối tượng socket nhận được tin nhắn, chương trình sử dụng hàm `recvfrom(1024)` để nhận về thông tin gồm dữ liệu và địa chỉ, sau đó gửi phản hồi tới máy tính cho địa chỉ address bằng hàm `sendto()`.

```
23     # Receive/respond loop
24     while True:
25         print('\nwaiting to receive message')
26         data, address = sock.recvfrom(1024)
27
28         print('received {} bytes from {}'.format(
29             len(data), address))
30         print(data)
31
32         print('sending acknowledgement to', address)
33         sock.sendto(b'ack', address)
```

| Để thử nghiệm, trên windows ta mở hai cửa sổ cmd (Command Prompt) như hình ảnh mô tả phần dưới

- ▶ Trên cửa sổ cmd (1), chạy file mã lệnh `multicast_receiver.py`, chạy chương trình phía thu
- ▶ Trên cửa sổ cmd (2). Chạy file mã lệnh `multicast_sender.py`, chạy chương trình phía phát
 - Tại thời điểm này, phía phát sẽ gửi tin nhắn theo phương thức phát đa hướng.
 - Đồng thời, phía thu sẽ nhận được một tin nhắn.

The image shows two separate Command Prompt windows running on Microsoft Windows 10. Both windows have a title bar 'Command Prompt' and a status bar indicating the version is 10.0.19041.1237 and rights are reserved by Microsoft Corporation.

Left Window (Receiver):

- Command: `python multicast_receiver.py`
- Output:

```
Microsoft Windows [Version 10.0.19041.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\donogh>cd Desktop
C:\Users\donogh\Desktop>python multicast_receiver.py
waiting to receive message
```

Right Window (Sender):

- Command: `python multicast_sender.py`
- Output:

```
Microsoft Windows [Version 10.0.19041.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\donogh\Desktop>python multicast_sender.py
sending b'very important data'
waiting to receive
received b'ack' from ('192.168.208.1', 10000)
waiting to receive
timed out, no more responses
closing socket

C:\Users\donogh\Desktop>
```

In both windows, the command and its output are highlighted with a red rectangular box.

| Trên hai cửa sổ cmd khác nhau

- ▶ Phía thu sẽ tiếp tục chờ nhận tin nhắn.
- ▶ Phía phát có thể gửi lại tin nhắn.

The image shows two separate Command Prompt windows running on Microsoft Windows 10. Both windows have a title bar 'Command Prompt' and a status bar indicating the version is 10.0.19041.1237 and rights belong to Microsoft Corporation.

Left Window (Receiver):

```
Microsoft Windows [Version 10.0.19041.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dongh>cd Desktop
C:\Users\dongh\Desktop>python multicast_receiver.py
waiting to receive message
received 19 bytes from ('192.168.208.1', 50866)
b'very important data'
sending acknowledgement to ('192.168.208.1', 50866)

waiting to receive message
```

Right Window (Sender):

```
Microsoft Windows [Version 10.0.19041.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dongh\Desktop>python multicast_sender.py
sending b'very important data'
waiting to receive
received b'ack' from ('192.168.208.1', 10000)
waiting to receive
timed out, no more responses
closing socket

C:\Users\dongh\Desktop>
```

Both windows show the command line interface with the user's input and the output of the Python scripts. The receiver window has a red box around its output, and the sender window also has a red box around its output.

Bài 4.

Truyền thông giao tiếp với Raspberry Pi

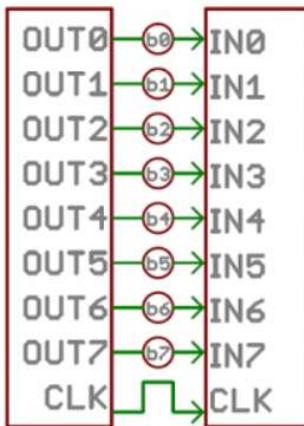
- | 4.1. Truyền thông nối tiếp
- | 4.2. Truyền thông SPI
- | 4.3. Truyền thông Bluetooth
- | 4.4. Máy chủ web Flask

Truyền thông nối tiếp là gì?

| Giới thiệu truyền thông nối tiếp

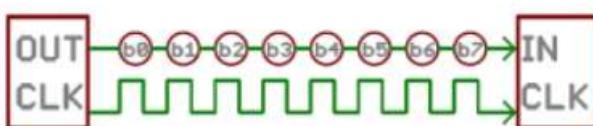
- ▶ Vì cốt lõi của một hệ thống nhúng là giao tiếp và vận hành giữa bộ vi xử lý và các mạch điện khác nhau, nên cần có những giao tiếp truyền thông để trao đổi dữ liệu giữa các thành phần này.
- ▶ Có nhiều giao tiếp truyền thông, nhưng nhìn chung, chúng có thể được chia thành hai loại: song song và nối tiếp.

I Giao tiếp song song



- ▶ Với giao tiếp song song, nhiều bít được truyền đồng thời trên các đường tín hiệu riêng.
- ▶ Giao tiếp song song gửi tất cả dữ liệu cùng một lúc qua 8, 16 hoặc nhiều hơn các đường dây.
- ▶ Cần có đường tín hiệu đồng hồ (CLK) để tất cả các đường tín hiệu hoạt động cùng nhau theo cùng nhịp thời gian (đồng hồ). Ví dụ: cần 9 đường tín hiệu cho bus dữ liệu 8 bit.
- ▶ Mặc dù truyền song song dễ thực hiện hơn và nhanh hơn so với truyền nối tiếp, nhưng nó đòi hỏi nhiều tài nguyên và chi phí hơn, đồng thời sử dụng nhiều đường I/O (vào/ra) hơn.

I Giao tiếp nối tiếp



- ▶ Với giao tiếp nối tiếp, dữ liệu được truyền tuần tự từng bít một theo một luồng (trên một đường tín hiệu).
- ▶ Như hình bên trái, giao diện nối tiếp gồm đường truyền dữ liệu và đường truyền tín hiệu xung nhịp (tín hiệu clock).

Giao tiếp nối tiếp với thiết bị

- | Giao tiếp nối tiếp là một dạng thức giao tiếp truyền thông dữ liệu quan trọng trong hệ thống nhúng
 - ▶ Các chuẩn giao tiếp như RS232, RS485 và UART là các chuẩn truyền thông giao tiếp nối tiếp bất đồng bộ.
 - ▶ Trong giao tiếp bất đồng bộ, phía phát và phía thu không sử dụng chung xung nhịp (tín hiệu clock). Ở phía thu nhận, dữ liệu được nhận biết thông qua tín hiệu “bắt đầu” (được gọi là bít start) và tín hiệu “kết thúc” (được gọi là bít stop).
 - ▶ Vì truyền thông kết nối tiếp tục gửi dữ liệu là các bít, nên BPS được sử dụng làm đơn vị tốc độ truyền trong truyền thông kết nối tiếp.

| **UART:** (Universal Asynchronous Receiver-Transmitter) Chuẩn truyền-nhận không đồng bộ tổng quát

- ▶ Trước khi bắt đầu gửi bít dữ liệu, tín hiệu bít start sẽ được gửi trước.
- ▶ Các bít dữ liệu được gửi trên đường truyền dữ liệu, thay đổi từ mức thấp (LOW) sang mức cao (HIGH).
- ▶ Có nghĩa là, bit start được gửi đầu tiên, tiếp theo là các bít dữ liệu được gửi tuần tự, tiếp theo sau là bít kiểm tra chẵn/lẻ (có thể sử dụng hoặc không) và cuối cùng là bít stop.

Bit	1	2	3	4	5	6	7	8	9	10	11
	Start Bit	Dữ liệu Bit (5 ~ 8 bit)							Parity Bit	Sytop Bit	
Dữ liệu	Start	D0	D1	D2	D3	D4	D5	D6	D7	Parity	Stop

| **UART:** (Universal Asynchronous Receiver-Transmitter) Chuẩn truyền-nhận không đồng bộ tổng quát

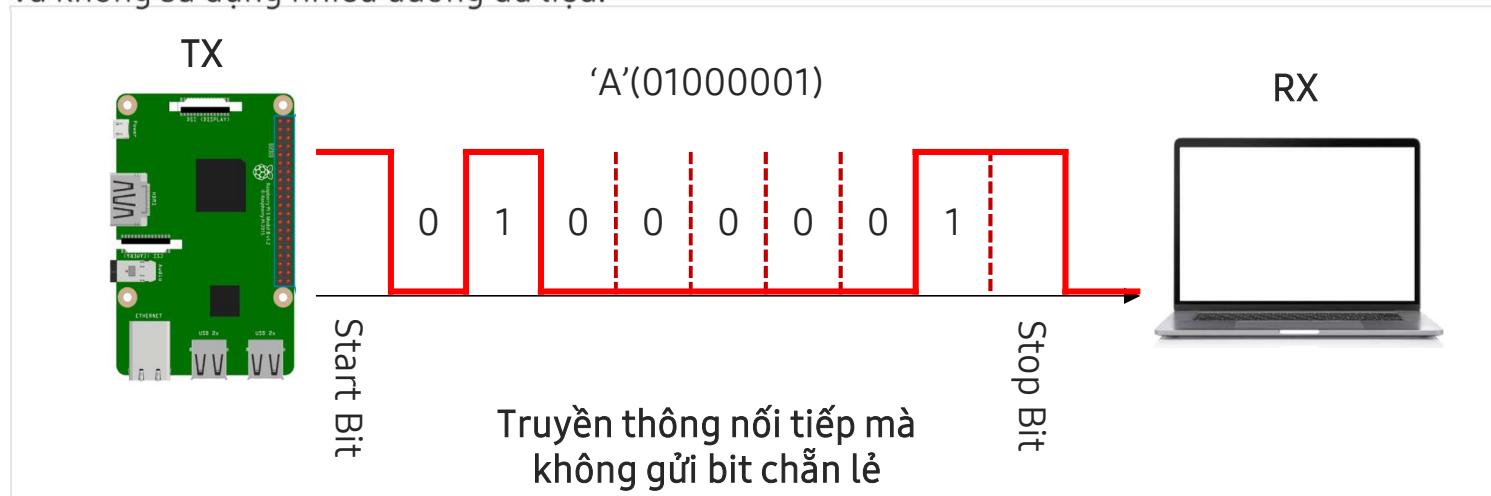
Các thuật ngữ trong Truyền thông nối tiếp

- 1) Bít Start: tín hiệu báo hiệu bắt đầu quá trình giao tiếp. Mức tín hiệu được duy trì như tín hiệu của bít 1. Thời gian truyền một bit phụ thuộc vào tốc độ Baud. Tốc độ giao tiếp càng nhanh thì thời gian càng ngắn.
- 2) Các bít dữ liệu: có thể chọn số lượng bít dữ liệu từ 5 đến 8 Bit, thông thường số lượng bít là 8 bít được sử dụng nhiều nhất.
- 3) Bit chẵn lẻ: (bít kiểm tra chẵn lẻ) giá trị được xác định và sử dụng để phát hiện lỗi. Trong trường hợp không sử dụng, bít chẵn lẻ sẽ không được truyền đi.
- 4) Bít Stop: tín hiệu báo hiệu kết thúc truyền dữ liệu.

Bit	1	2	3	4	5	6	7	8	9	10	11
	Start Bit	Data Bit (5 ~ 8 bit)								Parity Bit	Stop Bit
Data	Start	D0	D1	D2	D3	D4	D5	D6	D7	Parity	Stop

| UART: (Universal Asynchronous Receiver-Transmitter) Chuẩn truyền-nhận không đồng bộ tổng quát

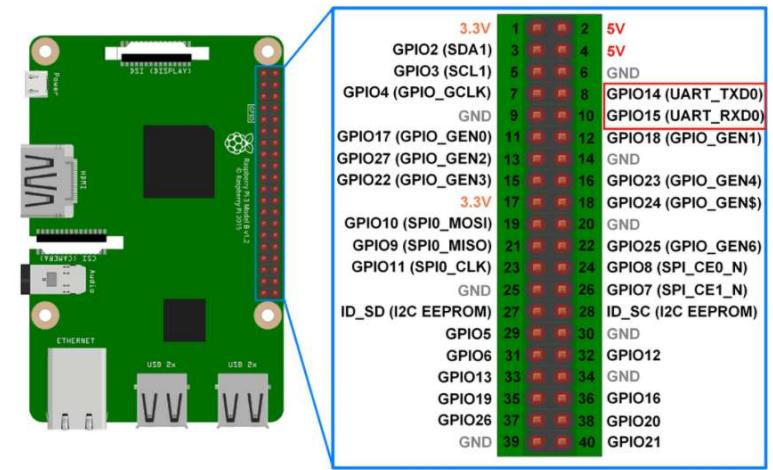
- ▶ Khi gửi ký tự 'A' thông qua truyền thông nối tiếp giữa Raspberry Pi và PC, ký tự 'A' mang giá trị là 65 ở hệ thập phân và 10000001 ở hệ nhị phân trong bảng mã ASCII.
- ▶ Quy ước đối với tín hiệu: bít 1 tương ứng với mức cao (HIGH), bít 0 tương ứng với THẤP (LOW), và tín hiệu điện được truyền theo nhịp thời gian.
- ▶ Hình bên dưới mô tả quá trình truyền, trong đó không sử dụng bit chẵn lẻ.
 - Phía phát (bộ phát): TX, phía thu/nhận (bộ thu): RX, Tốc độ truyền tin: tốc độ Baud
 - Cả bộ phát và bộ thu phải hoạt động cùng tốc độ truyền tin (cùng tốc độ baud).
 - RS232 là một giao tiếp truyền thông nối tiếp không đồng bộ, trong đó không có đường tín hiệu xung nhịp và không sử dụng nhiều đường dữ liệu.



Giao tiếp nối tiếp với Raspberry Pi

| Raspberry Pi có 4 kênh USB và 2 kênh UART.

Kênh UART	Cổng nối tiếp	Tên thiết bị	Sử dụng
UART0	Serial0	/dev/ttyS0 Or /dev/serial0	Console or Serial GPIO14(TXD) GPIO15(RXD)
UART1	Serial1	/dev/ttyAMA0 Or /dev/serial1	Bluetooth



Bài 4.

Hướng dẫn giao tiếp Socket with Raspberry Pi

- | 4.1. Truyền thông nối tiếp
- | **4.2. Truyền thông SPI**
- | 4.3. Truyền thông Bluetooth
- | 4.4. Máy chủ web Flask

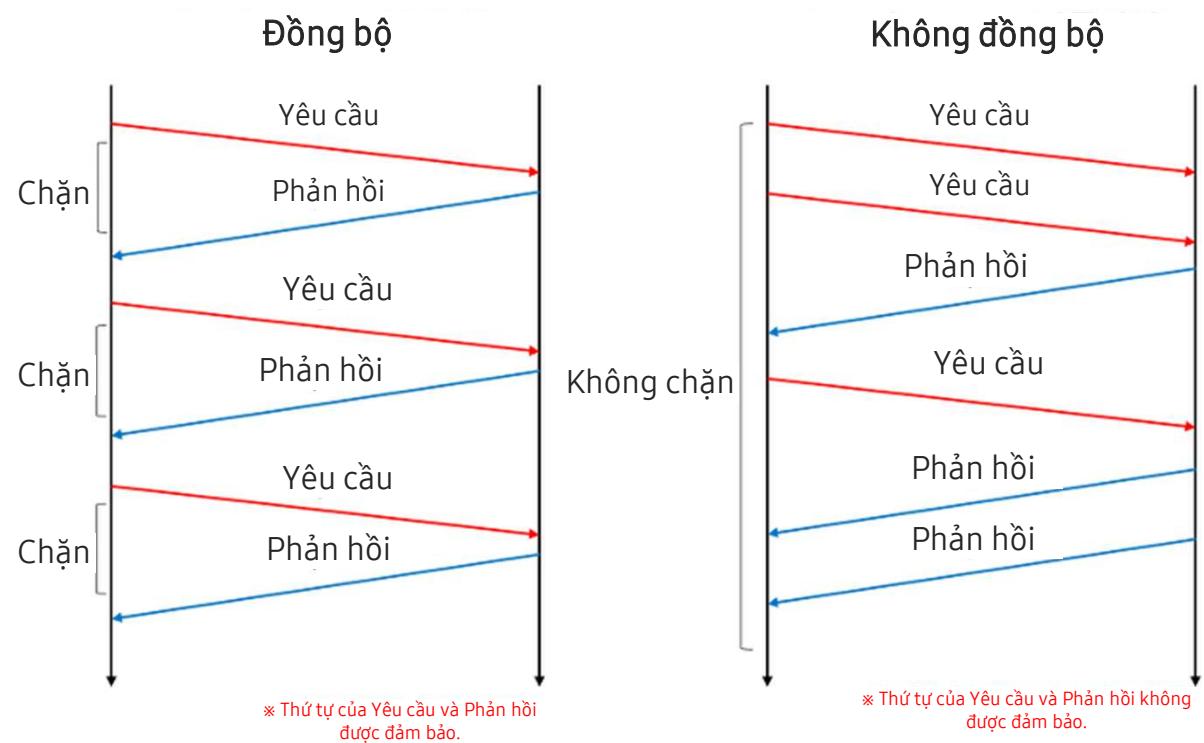
Giao tiếp SPI là gì?

I SPI (Serial Peripheral Interface) - Giao diện ngoại vi nối tiếp

- ▶ SPI là một trong những chuẩn giao tiếp truyền thông kết nối thường được sử dụng trong các bộ vi xử lý, tương tự như các chuẩn I2C và UART.
- ▶ SPI hỗ trợ giao tiếp một-nhiều (giữa một trạm chủ và nhiều trạm tớ).
- ▶ SPI chủ yếu được sử dụng để trao đổi dữ liệu tốc độ cao với thẻ nhớ SD, bộ nhớ nối tiếp, mô-đun giao tiếp mạng...
- ▶ RS232 và UART là các giao tiếp truyền thông nối tiếp không đồng bộ, Trong khi đó SPI và I2C là các chuẩn giao tiếp truyền thông nối tiếp đồng bộ.

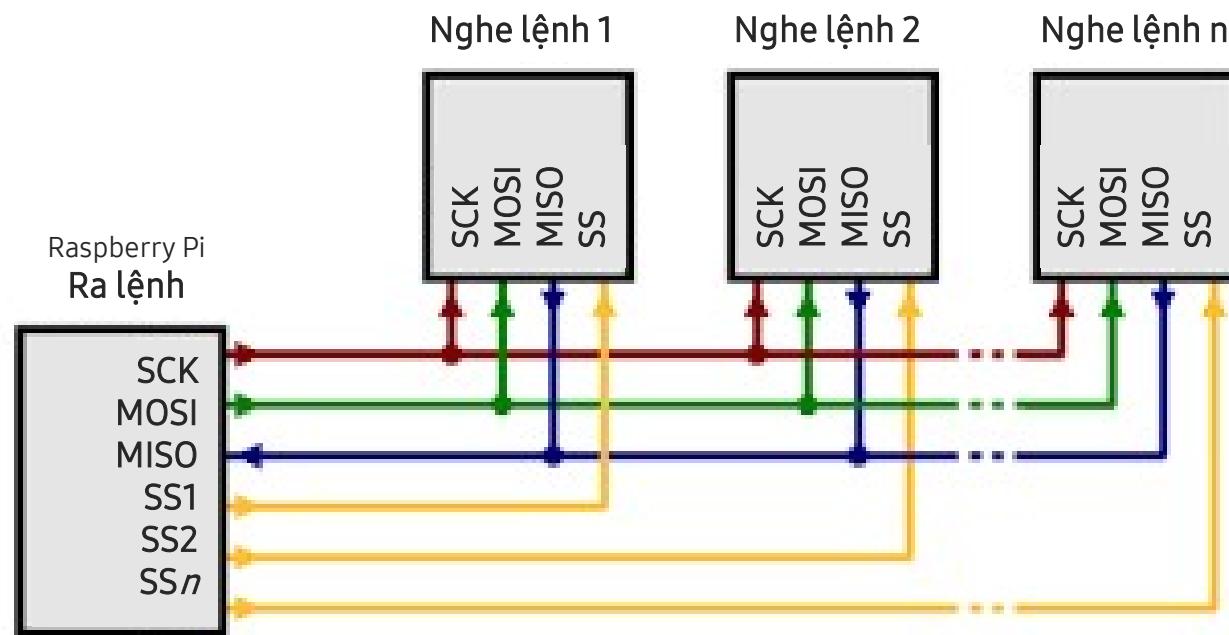
| So sánh giao tiếp truyền thông đồng bộ và không đồng bộ

- ▶ Cách dễ nhất để phân biệt giữa truyền thông đồng bộ và không đồng bộ là kiểm tra xem tín hiệu xung nhịp (clock) có được sử dụng hay không trong các đường tín hiệu giao tiếp.
- ▶ Giao tiếp đồng bộ truyền và nhận dữ liệu bằng cách đồng bộ hóa việc truyền/nhận dữ liệu theo xung nhịp (clock).
- ▶ Giao tiếp không đồng bộ không có đường tín hiệu clock mà chỉ có đường truyền dữ liệu, truyền các tín hiệu đặc biệt để xác định thời điểm bắt đầu và kết thúc của việc truyền thông tin



I Truyền thông giao tiếp SPI

- SPI là một chuẩn giao tiếp truyền thông nối tiếp đồng bộ với một đường tín hiệu xung nhịp (clock), cho phép một thiết bị chủ SPI (master) và nhiều thiết bị tớ SPI (slave) giao tiếp với nhau. Tín hiệu đồng hồ do trạm chủ thiết lập và truyền tới tất cả các trạm tớ khác.
- Trong hình bên dưới có 3 trạm tớ. Các trạm tớ không giao tiếp với trạm chủ đồng thời. Trong khi SPI Master (Raspberry Pi) phát tín hiệu xung nhịp, đồng thời trạm chủ sẽ chọn trạm tớ mà nó cần giao tiếp thông qua chân tín hiệu Chip Select (CS) (còn gọi là chân tín hiệu chọn chíp).

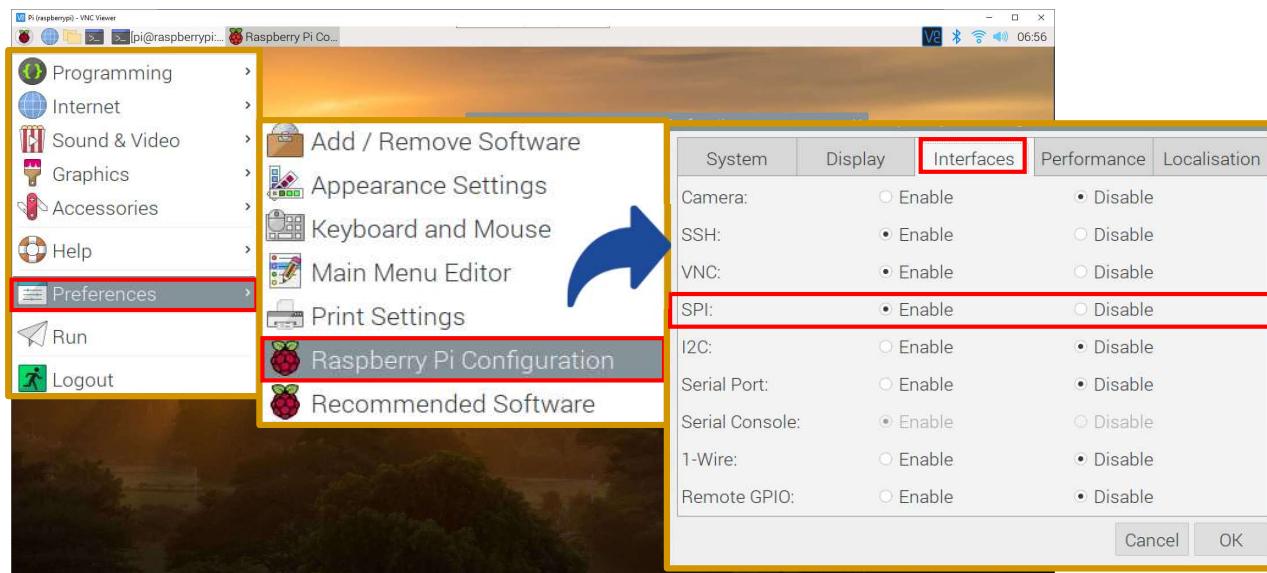


I Xung nhịp và các tín hiệu trong chuẩn giao tiếp SPI

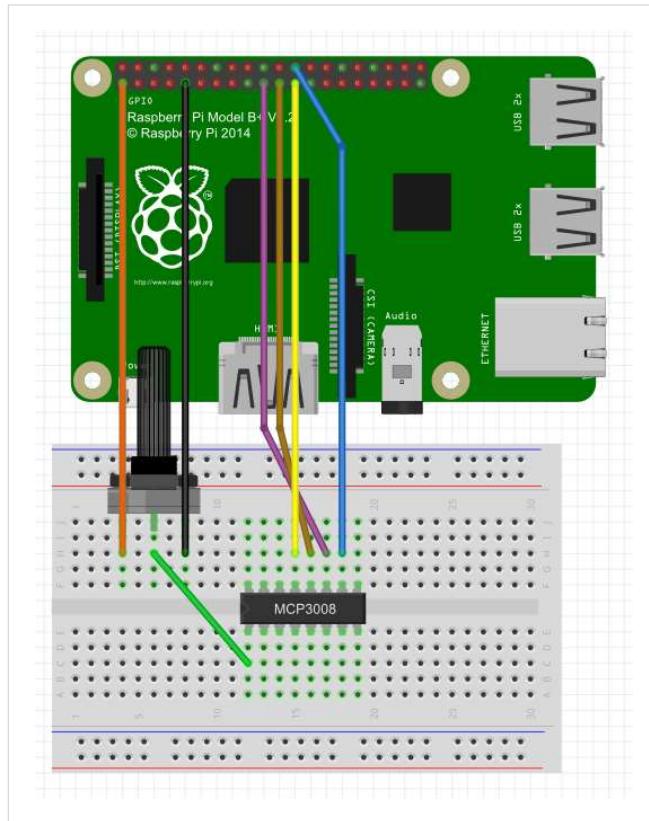
SCK (xung nhịp)	Đồng hồ cho giao tiếp đồng bộ – được cung cấp từ Master
MOSI (Master Out Slave In)	Tín hiệu truyền dữ liệu từ Master đến Slave
MISO (Master In Slave Out)	Tín hiệu truyền dữ liệu từ Slave đến Master.
	Tín hiệu chọn chip (chọn trạm tớ) giao tiếp với trạm chủ khi có nhiều trạm tớ cùng kết nối trên một bus tín hiệu.
CS (Chip Select)	Yêu cầu mỗi trạm có chân tín hiệu CS riêng. Khi mức tín hiệu tại chân CS của một máy trạm ở mức cao (HIGH) sẽ cho phép quá trình truyền thông giữa trạm chủ và máy trạm này được thực hiện, trong khi đó mức tín hiệu tại chân CS tại các máy trạm khác ở mức thấp.

Xuất ADC và giá trị điện áp được chuyển đổi bằng SPI

- I Sử dụng Raspberry Pi để đọc giá trị từ ADC (bộ chuyển đổi tín hiệu tương tự sang số) và chuyển đổi thành giá trị điện áp
 - ▶ Có 2 SPI trên Raspberry Pi, trong đó cổng SPI (0) và SPI (1) được kết BUS (0) của Pi.
 - ▶ Sau khi đọc dữ liệu từ bộ biến đổi ADC giao tiếp qua cổng SPI (2), bộ biến đổi ADC nhận tín hiệu điện áp từ một biến trở và chuyển đổi sang giá trị số, giá trị này được đọc về để chuyển đổi thành giá trị điện áp để hiển thị trên màn hình.
 - ▶ Để sử dụng cổng giao tiếp SPI, chọn mục cấu hình Raspberry Pi → Giao diện và cho phép sử dụng (enable) cổng SPI.



Mạch thử nghiệm



- Biến trở được gắn trên bảng mạch thử (breadboard), tiến hành vặn núm của biến trở từ trái sang phải, chú ý thứ tự chân của biến trở (theo hình vẽ) là 3.3v, Out, GND.
 - Chân GND của biến trở nối với chân GND của Raspberry Pi
 - Chân Out của biến trở nối với chân Ch0 của IC MCP 3008
 - Chân CLK của MCP 3008 nối với chân GPIO11(CLK) của Raspberry Pi
 - Chân Dout (MISO) của MCP 3008 nối với GPIO9(MISO) của Raspberry Pi
 - Chân Din (MOSI) của MCP 3008 nối với GPIO10(MOSI) của Raspberry Pi
 - Chân CS/SHDN của MCP 3008 nối với GPIO8(CS) của Raspberry Pi

| File mã lệnh python: ~~~~/~~~/adc.py

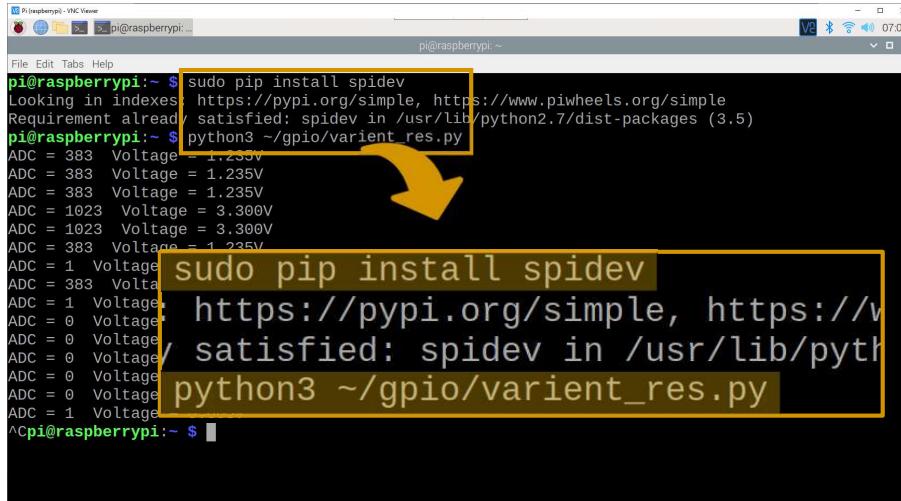
```

1 import spidev
2 import time
3
4 def analog_read(channel):
5     r = spi.xfer2([1, (0x08+channel)<<4, 0])
6     adc_out = ((r[1]&0x03)<<8) + r[2]
7     return adc_out
8
9 spi = spidev.SpiDev()
10 spi.open(0,0)
11 spi.max_speed_hz = 1000000
12
13 try:
14     while True:
15         adc = analog_read(2)
16         voltage = adc*3.3/1023
17         print("ADC = %d  Voltage = %.3fV" % (adc, voltage))
18         time.sleep(0.5)
19 except KeyboardInterrupt:
20     pass
21 finally:
22     spi.close()

```

- ▶ dòng 1, 2: Khai báo thư viện lập trình giao tiếp qua SPI (spidev) và thư viện time (cho định thời)
- ▶ dòng 4: Hàm đọc giá trị từ công giao tiếp SPI
- ▶ dòng 5 - 8: Đọc dữ liệu từ cổng SPI (được kết nối với Bus 0) và trả về dữ liệu số đã được chuyển đổi
- ▶ dòng 9: Khai báo biến đối tượng giao tiếp qua SPI
- ▶ dòng 10: gọi hàm spi.open(bus, device), với các tham số trong lời gọi hàm là 0.
- ▶ dòng 11: Thiết lập tốc độ dữ liệu trên bus SPI
- ▶ dòng 15: gọi hàm để đọc dữ liệu từ kênh (2) của bộ biến đổi ADC
- ▶ dòng 16: Tính toán giá trị điện áp tương ứng với giá trị đọc về từ bộ ADC
- ▶ dòng 17: Xuất giá trị ADC và giá trị điện áp ra màn hình
- ▶ dòng 22: đóng/dừng cổng giao tiếp SPI
- ▶ Lưu tập lệnh python dưới dạng :
~/rasp_ex/gpio/vareint_res.py

| Ví dụ thực thi chương trình



```
pi@raspberrypi:~ $ sudo pip install spidev
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Requirement already satisfied: spidev in /usr/lib/python2.7/dist-packages (3.5)
pi@raspberrypi:~ $ python3 ~/gpio/varient_res.py
ADC = 383 Voltage = 1.235V
ADC = 383 Voltage = 1.235V
ADC = 383 Voltage = 1.235V
ADC = 1023 Voltage = 3.300V
ADC = 1023 Voltage = 3.300V
ADC = 383 Voltage = 1.235V
ADC = 1 Voltage
ADC = 383 Voltage
ADC = 1 Voltage
ADC = 0 Voltage
ADC = 1 Voltage
^Cpi@raspberrypi:~ $
```

- ▶ Chú ý: Thư viện lập trình cho truyền thông SPI được cài đặt mặc định. Trong trường hợp chưa có, thư viện có thể được cài đặt thông qua lệnh:

```
> sudo pip install spidev
```
- ▶ Thực thi lệnh [python3 ~/rasp_ex/gpio/varient_res.py](#) để chạy tập lệnh Python đã tạo.
- ▶ Xoay núm biến tròn và kiểm tra kết quả hiển thị trên màn hình. Ta có thể thấy giá trị thay đổi tùy thuộc vào vị trí của núm trên biến tròn.
- ▶ Nếu bạn muốn dừng thực thi, hãy nhấn Ctrl + C để kết thúc chương trình.

Bài 4.

Hướng dẫn giao tiếp Socket with Raspberry Pi

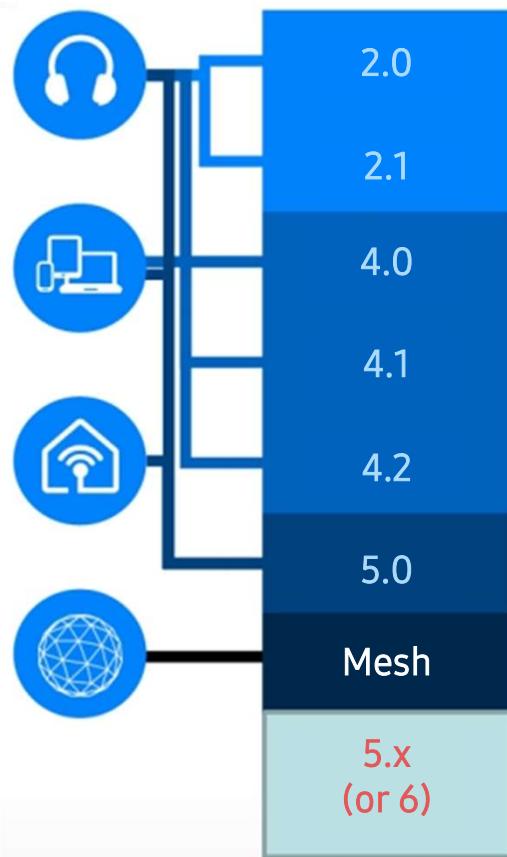
- | 4.1. Truyền thông nối tiếp
- | 4.2. Truyền thông SPI
- | **4.3. Truyền thông Bluetooth**
- | 4.4. Máy chủ web Flask

Bluetooth là gì?

- | Là công nghệ kết nối không dây tầm gần, kết nối với thiết bị di động để gửi và nhận thông tin
 - ▶ Kết nối Bluetooth sử dụng dải tần ISM (Industrial Science Medical) 2400 ~ 2483,5 MHz.
 - ▶ Raspberry Pi 4 hỗ trợ Bluetooth phiên bản v5
 - ▶ Bluetooth 5 tăng phạm vi, tốc độ và dung lượng dữ liệu của các ứng dụng Bluetooth.
 - Nó cung cấp một giao diện mới, trong đó giảm băng thông để mở rộng cự ly kết nối hoặc sử dụng nhiều băng thông hơn cho phạm vi ngắn.
 - Nó truyền ít dữ liệu hơn (tốc độ chậm hơn) trong khoảng cách xa và truyền nhiều dữ liệu hơn (tốc độ nhanh hơn) trong khoảng cách ngắn.
 - Điều đó cho phép hoạt động tiêu thụ ít năng lượng, và hoạt động linh hoạt dựa trên nhu cầu ứng dụng.

Sự phát triển của công nghệ Bluetooth

| Bluetooth có lợi thế về chi phí thấp, năng lượng thấp cho các ứng dụng truyền âm thanh và giải pháp IoT tiêu thụ ít năng lượng



Tốc độ cơ bản/Tốc độ dữ liệu nâng cao

- Hầu hết các trường hợp sử dụng liên quan đến kết nối và phát trực tuyến

Bluetooth tiết kiệm năng lượng (4.0, 5.0)

- Các ứng dụng liên quan đến việc kết nối với Thing để truyền dữ liệu (ví dụ như các thiết bị cảm biến IoT)
- Các ứng dụng liên quan đến việc phát dữ liệu (phát các thông tin cảnh báo hoặc quảng cáo)

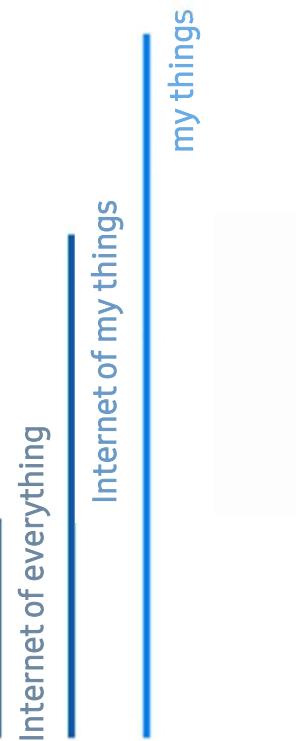
Có thể đạt được cự ly 1Km / tốc độ tối đa là 2Mbps

Mesh

- Cấu trúc liên kết mạng dạng mắt lưới
- Truyền lệnh và điều khiển + dữ liệu trên kênh truyền có băng thông không cao

bandwidth

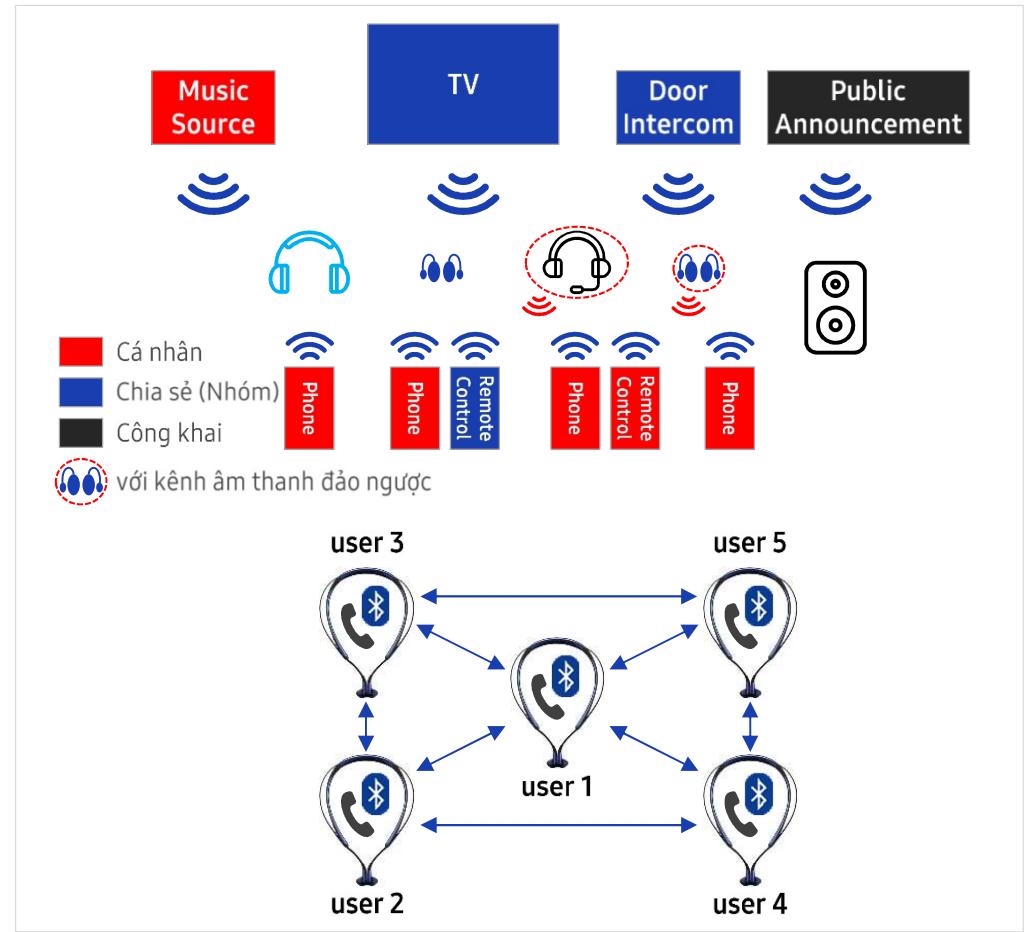
- Các thiết bị âm thanh chất lượng cao, tiêu thụ ít điện năng (Phát sóng/Multicast, kết nối M:N, nhận dạng giọng nói BLE, Chất lượng cao)
- Đa ăng-ten: Định vị chính xác cao



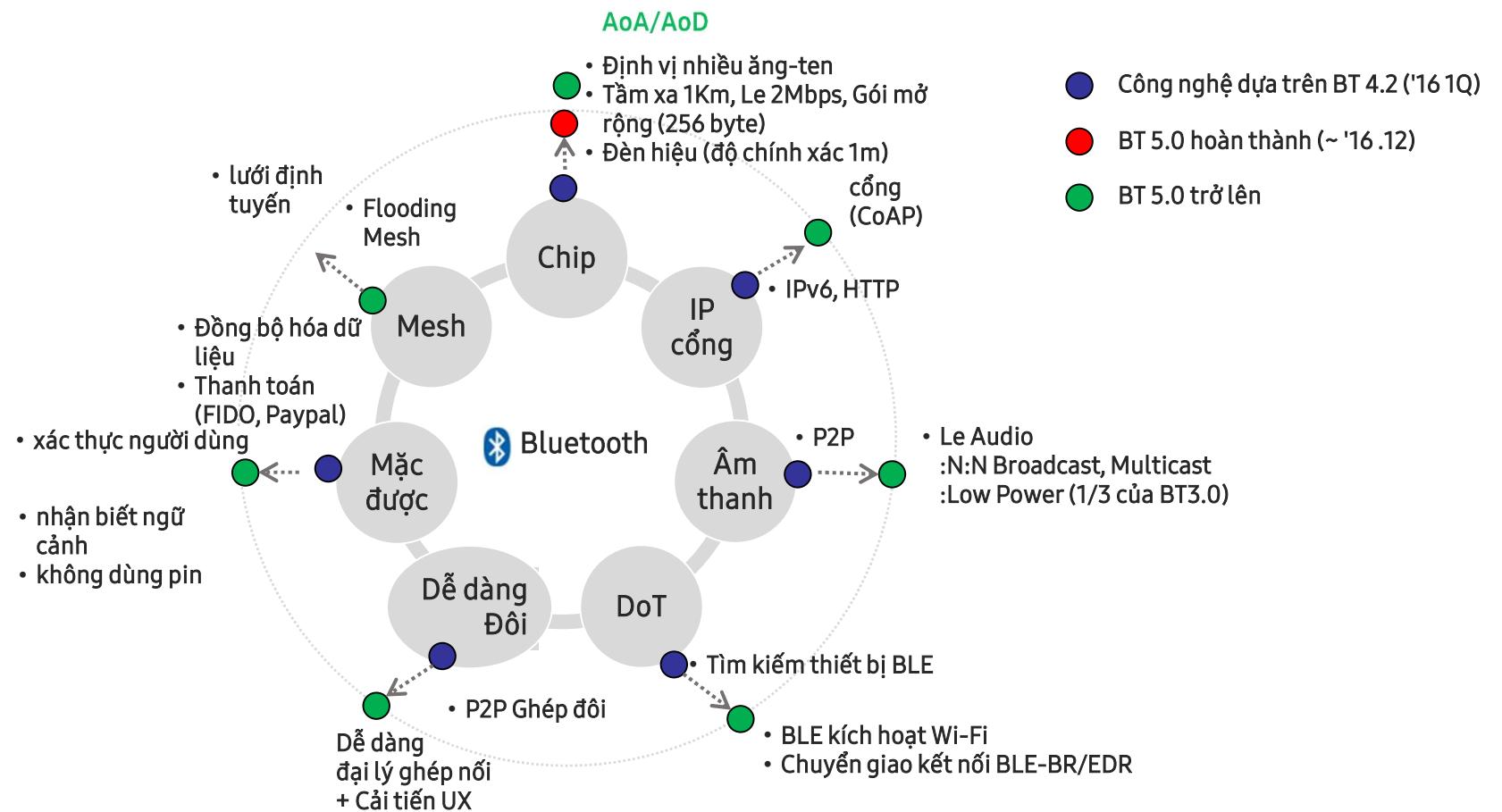
Các thiết bị âm thanh tiêu thụ ít năng lượng (Bluetooth LE Audio)

I Audio LE (Audio Low Energy)

- Chế độ nhóm: Đa tuyến 1: N, Đa kênh, PTT
- Điều khiển dễ dàng: Kết nối Easy 1:N, Chuyển đổi Src / Sink, Điều khiển âm lượng
- Chế độ chia sẻ: Chia sẻ nhạc, Chuyển tiếp nhạc
- Hỗ trợ giọng nói: Nhận dạng giọng nói, Nhận dạng giọng nói



Sự phát triển và ứng dụng của công nghệ Bluetooth



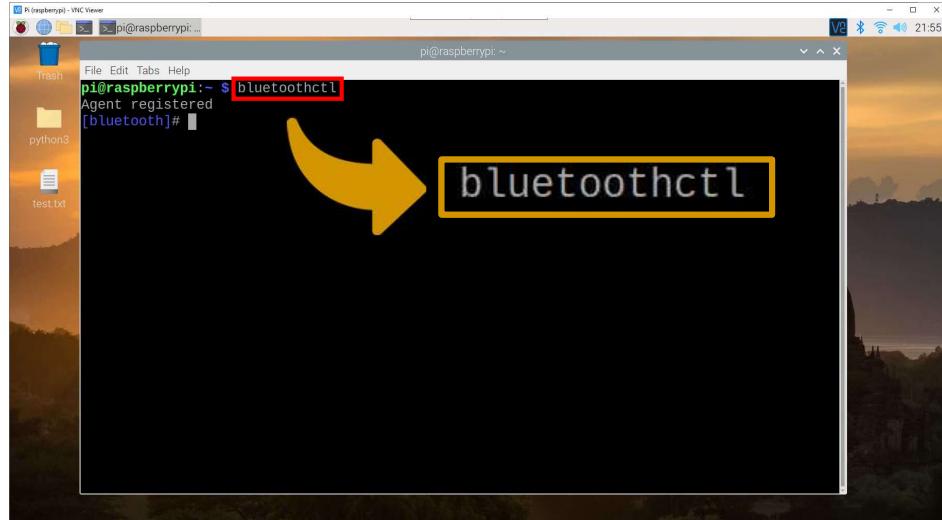
Giao tiếp giữa thiết bị với Raspberry Pi qua Bluetooth

- I Kết nối Raspberry Pi với điện thoại thông minh chạy Android thông qua kết nối Bluetooth để truyền/nhận dữ liệu hai chiều.
- ▶ Phần cứng Bluetooth trên Raspberry Pi 4 được kết nối với CPU thông qua cổng giao tiếp UART.
 - ▶ Cổng giao tiếp này có thể được sử dụng theo cách gần giống như cổng giao tiếp UART, nhưng điểm khác biệt là cần thực hiện ghép nối không dây với thiết bị trước.
 - ▶ Một điểm khác biệt nữa là tên thiết bị nối tiếp được sử dụng không là “/dev/serial1”, mà là “/dev/rfcomm0”.

Kênh UART	Cổng nối tiếp	Tên thiết bị	Sử dụng
UART1	Serial1	/dev/ttyAMA0 Or /dev/serial1	Bluetooth

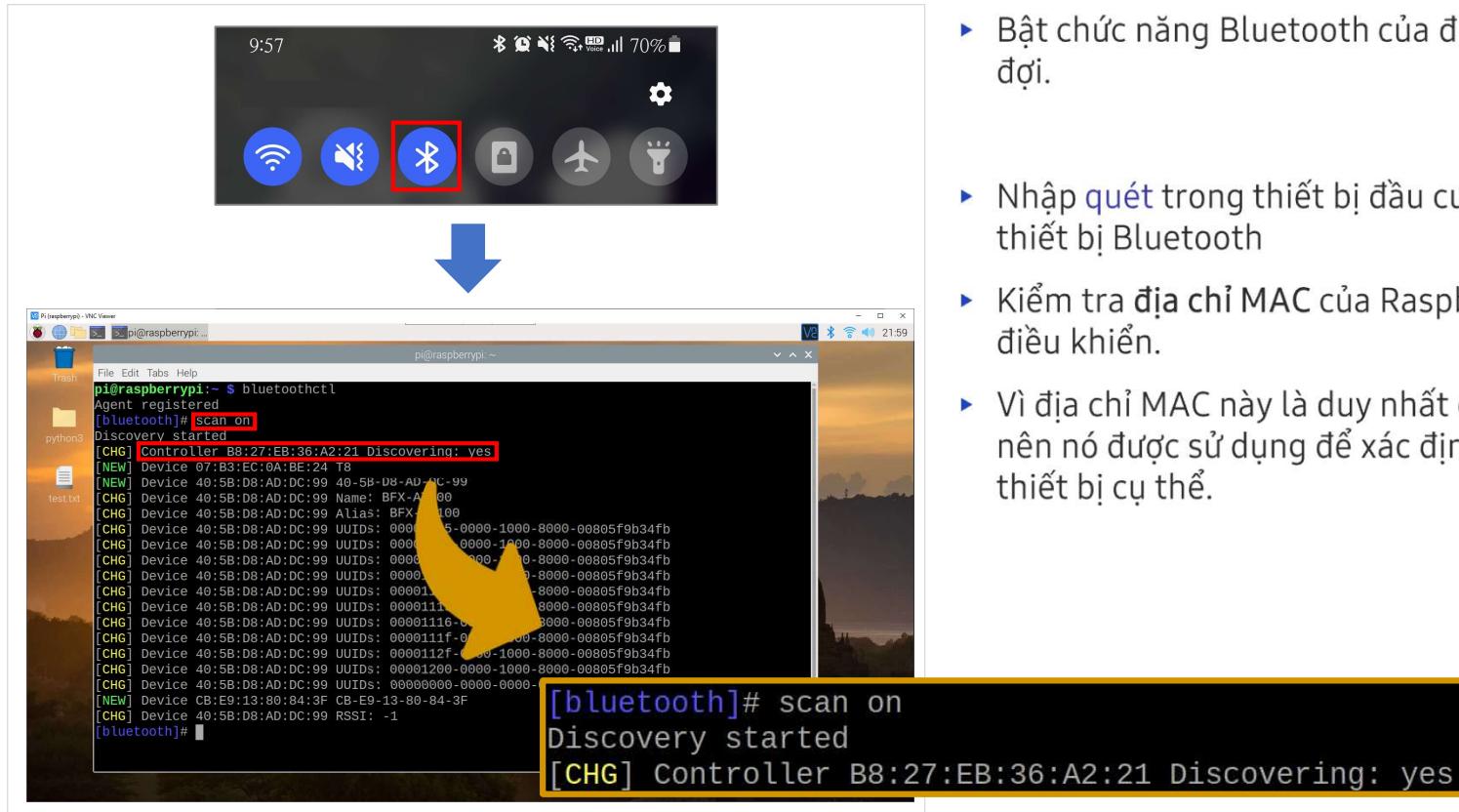
Ghép nối với thiết bị Bluetooth

Quá trình ghép nối



- ▶ Để Raspberry Pi và điện thoại thông minh giao tiếp không dây qua Bluetooth, trước tiên cần phải ghép nối giữa các thiết bị này.
- ▶ Chọn thiết bị bạn muốn kết nối vì có thể có nhiều thiết bị xung quanh thiết bị không dây. Nếu bạn cần số PIN, hãy nhập nó và kết nối.
- ▶ Quá trình ghép nối chỉ yêu cầu thực hiện một lần. Sau khi bạn ghép nối thiết bị Bluetooth lần đầu tiên, việc ghép nối với thiết bị này sẽ được thực hiện tự động trong các lần tiếp theo.
- ▶ Đầu tiên hãy chạy chương trình điều khiển Bluetooth để kiểm tra chi tiết quá trình ghép nối và kết nối trên Raspberry Pi.
- ▶ Nhập lệnh [bluetoothctl](#) trong thiết bị đầu cuối Raspberry Pi và nhấn Enter.

Quá trình ghép nối



- ▶ Bật chức năng Bluetooth của điện thoại thông minh và đợi.
- ▶ Nhập **quét** trong thiết bị đầu cuối để bắt đầu tìm kiếm thiết bị Bluetooth
- ▶ Kiểm tra **địa chỉ MAC** của Raspberry Pi bên cạnh Bộ điều khiển.
- ▶ Vì địa chỉ MAC này là duy nhất cho một thiết bị cụ thể nên nó được sử dụng để xác định, kết nối và quản lý thiết bị cụ thể.

I Quá trình ghép đôi

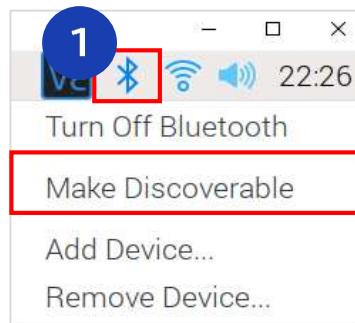
The screenshot shows a terminal window titled 'pi@raspberrypi: ~'. The terminal displays a log of Bluetooth device discoveries and a session with the 'bluetooth' command-line interface.

```
[CHG] Device 40:5B:D8:AD:DC:99 UUIDs: 00001116-0000-1000-8000-00805f9b34fb
[CHG] Device 40:5B:D8:AD:DC:99 UUUIDs: 0000111f-0000-1000-8000-00805f9b34fb
[CHG] Device 40:5B:D8:AD:DC:99 UUUIDs: 0000112f-0000-1000-8000-00805f9b34fb
[CHG] Device 40:5B:D8:AD:DC:99 UUUIDs: 00001200-0000-1000-8000-00805f9b34fb
[CHG] Device 40:5B:D8:AD:DC:99 UUUIDs: 00000000-0000-0000-0000-000000000000
[NEW] Device CB:E9:13:80:84:3F CB-E9-13-80-84-3F
[CHG] Device CB:E9:13:80:84:3F RSSI: -1
[CHG] Device CB:E9:13:80:84:3F Name: Mi Smart Band 4
[CHG] Device CB:E9:13:80:84:3F Alias: Mi Smart Band 4
[CHG] Device CB:E9:13:80:84:3F UUDS: 0000f000-0000-1000-0000-00005f9b34fb
[CHG] Device 07:B3:EC:0A:BE:24
[CHG] Device 40:5B:D8:AD:DC:99
[CHG] Device 07:B3:EC:0A:BE:24
[CHG] Device CB:E9:13:80:84:3F
[CHG] Device CB:E9:13:80:84:3F
[bluetooth]# scan off
Discovery stopped
[CHG] Controller B8:27:ED
[CHG] Device CB:E9:13:80:84:3F
[CHG] Device 40:5B:D8:AD:DC:99
[DEL] Device 07:B3:EC:0A:BE:24
[DEL] Device 40:5B:D8:AD:DC:99
[DEL] Device CB:E9:13:80:84:3F
[bluetooth]# quit
pi@raspberrypi:~ $
```

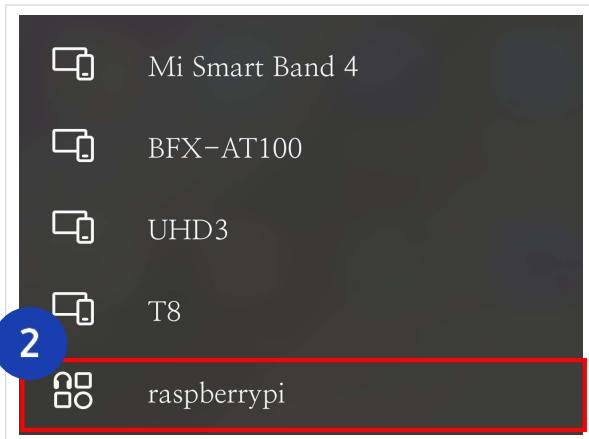
- ▶ Nếu bạn có thể thấy nhiều thiết bị trong lệnh quét, thì không có vấn đề gì với thiết bị Bluetooth của Raspberry Pi.
 - ▶ Ngừng tìm kiếm thiết bị Bluetooth bằng lệnh dừng quét (scan off) và lệnh kết thúc bằng quit.

Kết nối Raspberry Pi và thiết bị qua Bluetooth

I Kết nối Raspberry Pi với điện thoại thông minh

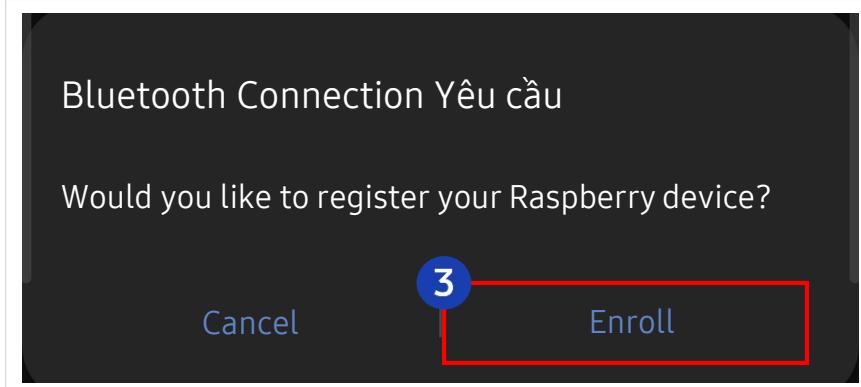


- Nhấp vào biểu tượng Bluetooth ở trên cùng bên phải của màn hình Raspberry Pi và nhấp vào **Make Discoverable** để làm cho Raspberry Pi có thể phát hiện được với các thiết bị khác.



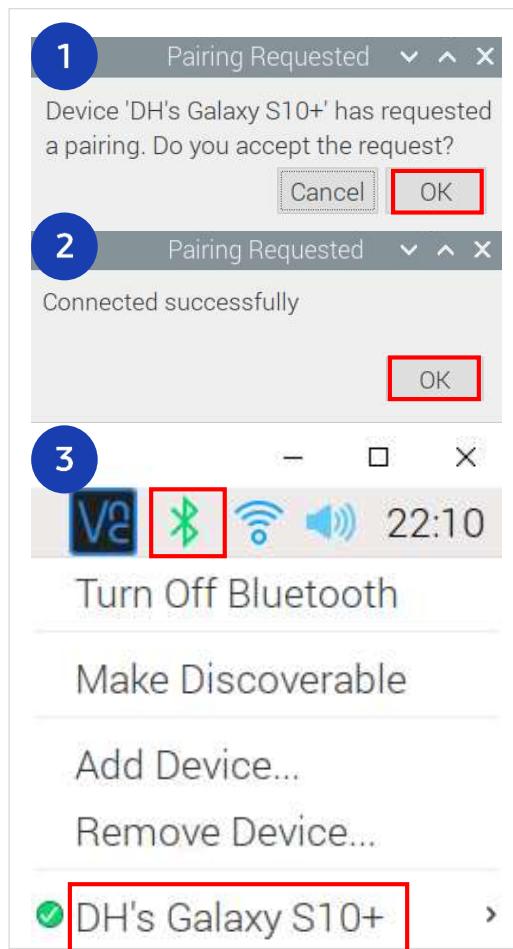
- Cuộn qua danh sách Bluetooth trên điện thoại thông minh của bạn để tìm raspberry pi và nhấp vào nó để thử kết nối

I Kết nối Raspberry Pi với điện thoại thông minh



- ▶ Nhấp vào Đăng ký trong cửa sổ bật lên xác nhận có đăng ký thiết bị raspberry pi hay không.

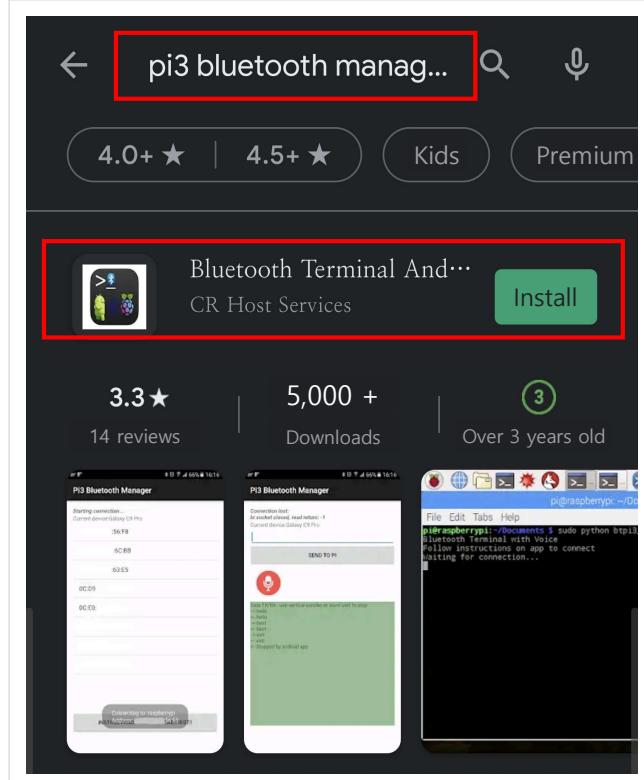
I Kết nối Raspberry Pi với điện thoại thông minh



- Khi cửa sổ bật lên **Yêu cầu ghép nối** xuất hiện trong thiết bị đầu cuối Raspberry Pi, hãy nhấp vào OK để chấp nhận Yêu cầu.
- Nếu xuất hiện thông báo **Connected Successfully** thì kết nối thành công.
- Nhấp vào biểu tượng phía trên bên phải để xác nhận, bạn sẽ thấy tên của thiết bị được kết nối.

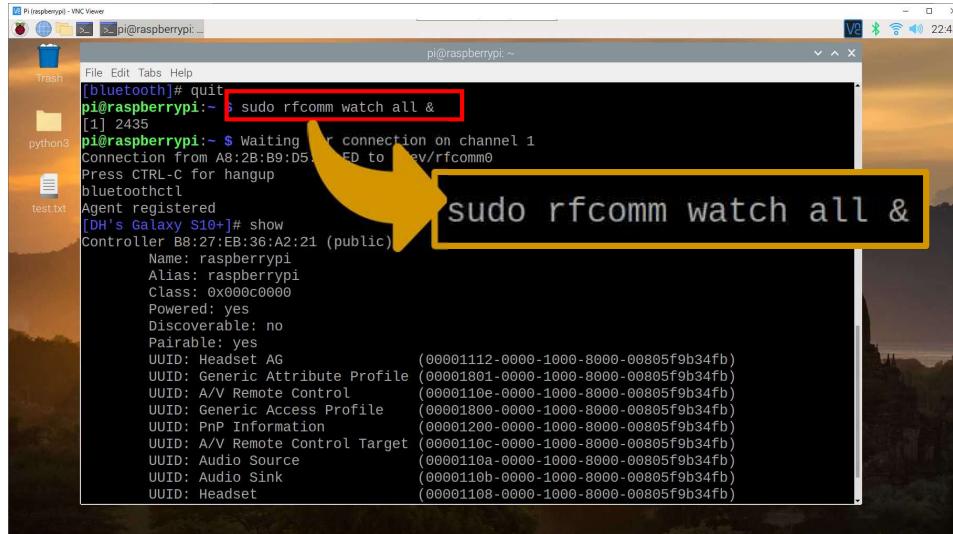
Cài đặt ứng dụng Bluetooth

I Cài đặt ứng dụng Bluetooth cho thiết bị Android



- ▶ Mặc dù điện thoại thông minh Android và Raspberry Pi được kết nối thông qua quá trình ghép nối, một ứng dụng bổ sung phải được cài đặt trên thiết bị Android để gửi và nhận dữ liệu qua Bluetooth.
- ▶ Tìm kiếm và tải xuống chương trình [pi3 bluetooth manager](#) trên Google Play

I Cài đặt ứng dụng Bluetooth trên thiết bị Android



```

Pi (raspberrypi) - VNC Viewer
File Edit Tabs Help
pi@raspberrypi: ~
[bluetooth]# quit
[1] 2435
pi@raspberrypi: $ Waiting for connection on channel 1
Connection from A8:2B:B9:D5:ED to /dev/rfcomm0
Press CTRL-C to hangup
bluetoothctl
Agent registered
[DH's Galaxy S10+]# show
Controller B8:27:EB:36:A2:21 (public)
  Name: raspberrypi
  Alias: raspberrypi
  Class: 0x000c0000
  Powered: yes
  Discoverable: no
  Pairable: yes
  UUID: Headset AC      (00001112-0000-1000-8000-00005f9b34fb)
  UUID: Generic Attribute Profile (00001801-0000-1000-8000-00005f9b34fb)
  UUID: A/V Remote Control   (0000110e-0000-1000-8000-00005f9b34fb)
  UUID: Generic Access Profile (00001800-0000-1000-8000-00005f9b34fb)
  UUID: PnP Information     (00001200-0000-1000-8000-00005f9b34fb)
  UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00005f9b34fb)
  UUID: Audio Source        (0000110a-0000-1000-8000-00005f9b34fb)
  UUID: Audio Sink          (0000110b-0000-1000-8000-00005f9b34fb)
  UUID: Headset              (00001108-0000-1000-8000-00005f9b34fb)

```

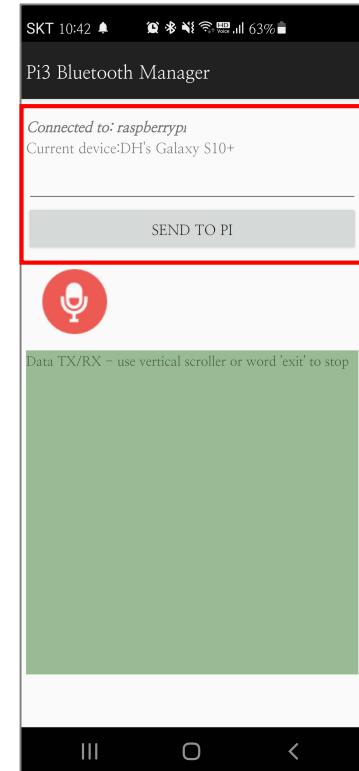
- ▶ Trước khi chạy ứng dụng đã tải xuống, trước tiên bạn phải thực thi lệnh “`rfcomm watch all &`” trên Raspberry Pi.
- ▶ `rfcomm watch all` là một lệnh kết nối với thiết bị đầu cuối này khi được kết nối với Pi từ một thiết bị khác qua Bluetooth. ‘&’ chỉ định lệnh này được thực thi mức nền, (hoạt động như một dịch vụ trong Android)

I Cài đặt ứng dụng Bluetooth cho thiết bị Android

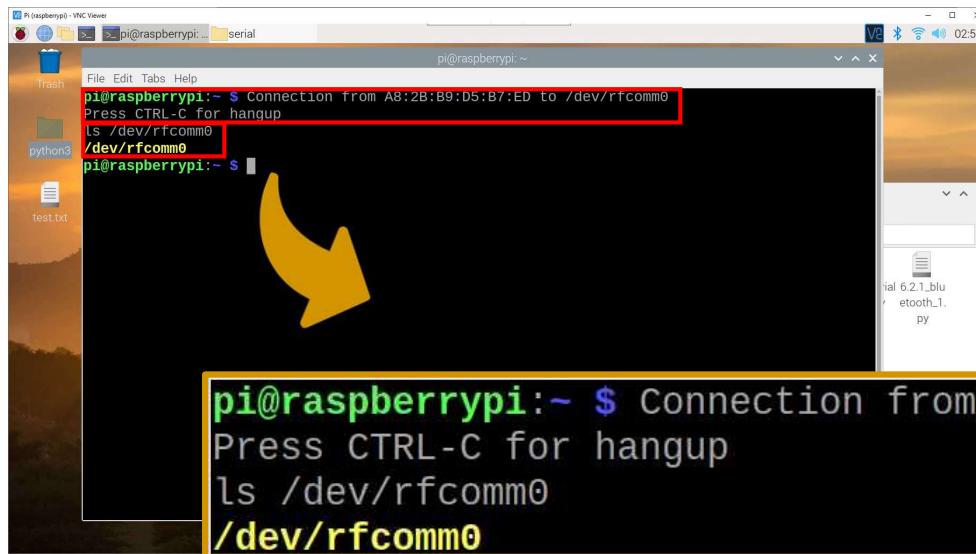
- Quay lại ứng dụng trên điện thoại thông minh của bạn, tìm địa chỉ MAC của Raspberry Pi và nhấp vào nó.



- Nếu kết nối thành công, bạn sẽ thấy màn hình như hình bên dưới.
- Văn bản đã nhập được gửi đến Raspberry Pi.



I Cài đặt ứng dụng Bluetooth cho thiết bị Android



A screenshot of a VNC viewer window titled "Pi (raspberrypi) - VNC Viewer". Inside the window, there is a terminal window with the following text:

```
pi@raspberrypi:~ $ Connection from A8:2B:B9:D5:B7:ED to /dev/rfcomm0
Press CTRL-C for hangup
ls /dev/rfcomm0
/dev/rfcomm0
pi@raspberrypi:~ $
```

A large yellow arrow points from the terminal window down to a callout box containing the same terminal text.

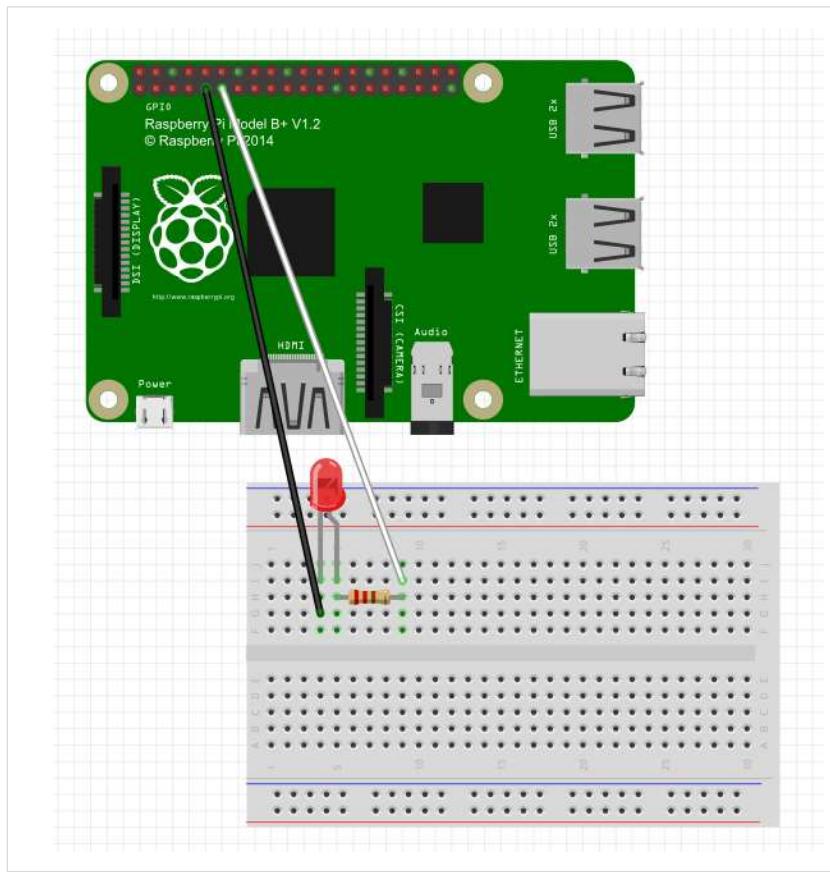
pi@raspberrypi:~ \$ Connection from A8:2B:B9:D5:B7:ED to /dev/rfcomm0
Press CTRL-C for hangup
ls /dev/rfcomm0
/dev/rfcomm0

- ▶ Nay bạn đã sẵn sàng để gửi đến Raspberry Pi. Bạn cần có khả năng xử lý chuỗi nhận được trên Raspberry Pi
- ▶ Nếu kết nối thành công, bạn sẽ thấy thông báo bắt đầu bằng **Kết nối từ** như trong hình. Kết nối Từ cho biết rằng nó được kết nối từ điện thoại thông minh.

- ▶ Nhập **ls /dev/rfcomm0**. Nếu thư mục xuất hiện như trong hình, thiết bị Bluetooth đã được kết nối thành công.
- ▶ Thư mục này sẽ biến mất khi thiết bị Bluetooth bị ngắt kết nối.

Sơ đồ mạch

I Tạo sơ đồ mạch



- ▶ Để nhấp nháy đèn LED bằng Raspberry Pi, trước tiên chúng ta cần kết nối đèn LED với Raspberry Pi.
- ▶ Chúng tôi sẽ gửi tín hiệu đến đèn LED thông qua GPIO17. Kết nối đèn LED theo thứ tự **GPIO17 → Điện trở → LED+ → LED- → GND**.

Tạo bluetooth_led.py

I Tạo tệp mã lệnh Python trên Raspberry Pi và lưu nó vào vị trí mong muốn

- Mã lệnh này sẽ bật đèn LED khi nhận được chuỗi “ON\n” được gửi từ điện thoại thông minh và tắt đèn LED khi nhận được chuỗi “OFF\n”.

```
1 import RPi.GPIO as GPIO
2 import serial
3
4 LED = 17
5
6 GPIO.setmode(GPIO.BCM)
7 GPIO.setup(LED, GPIO.OUT)
8
9 ser = serial.Serial('/dev/rfcomm0', 115200)
10 ser.close()
11 ser.open()
12
13 str = b'Bluetooth LED Control\r\n'
14 n = ser.write(str)
```

- dòng 1: khai báo gói thư viện Rpi.GPIO
- dòng 2: khai báo thư viện serial
- dòng 4: chân GPIO (17) được sử dụng để kết nối với LED
- dòng 6: thiết lập chế độ BCM cho cổng GPIO
- dòng 7: Thiết lập chế độ của chân GPIO là xuất tín hiệu ra, để điều khiển LED
- dòng 9: Thiết lập tốc độ dữ liệu của cổng giao tiếp nối tiếp là 115200bps
- dòng 10, 11: Đóng và mở cổng nối tiếp, tương ứng với đóng và mở kết nối qua Bluetooth
- dòng 13: bản tin dạng chuỗi byte gửi tới Android
- dòng 14: Gửi bản tin qua cổng nối tiếp, hay qua Bluetooth

I Tạo tệp mã lệnh Python trên Raspberry Pi và lưu nó vào vị trí mong muốn

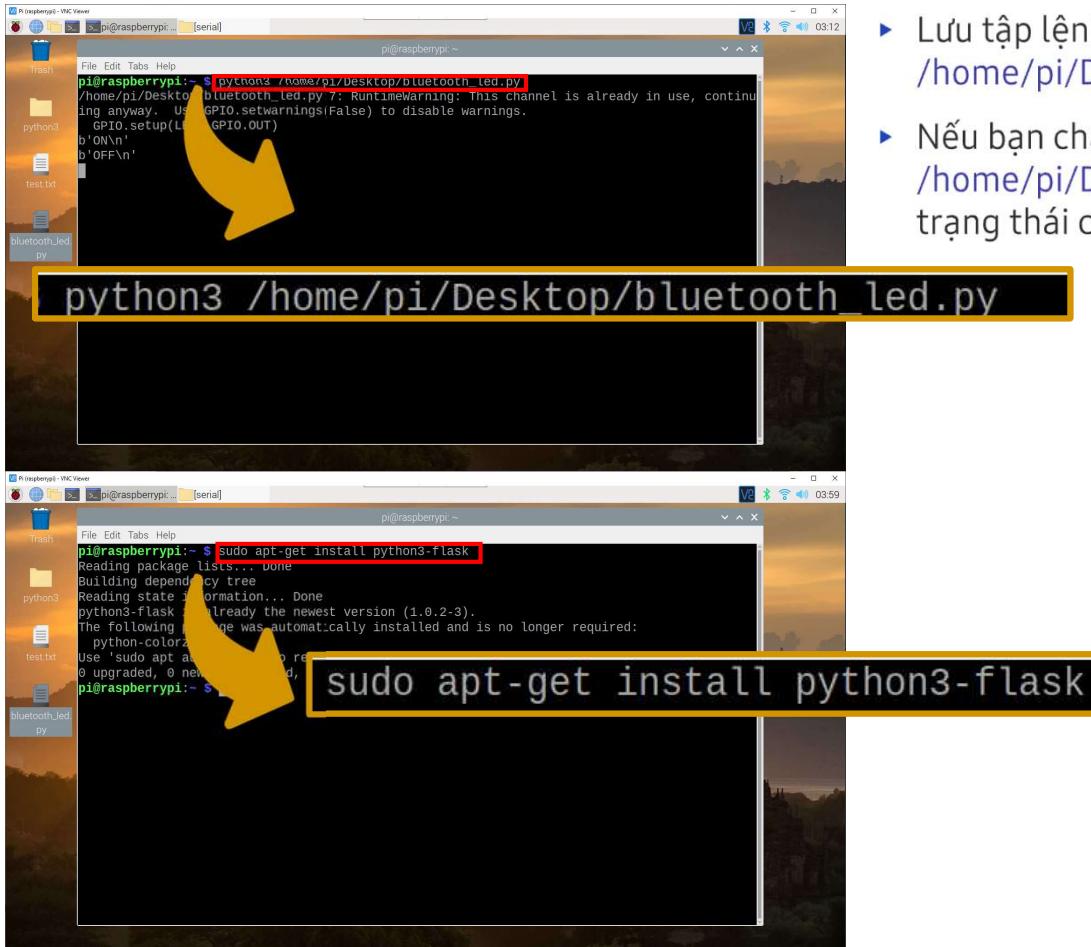
```
15
16     try:
17         while True:
18             if ser.readable():
19                 response = ser.readline()
20                 if response == b'ON\n':
21                     GPIO.output(LED, True)
22                 elif response == b'OFF\n':
23                     GPIO.output(LED, False)
24
25                 print(response)
26         except KeyboardInterrupt:
27             pass
28     finally:
29         ser.close()
```

- dòng 18: Nếu có dữ liệu nhận được từ điện thoại thông minh
- dòng 19: Đọc dữ liệu nhận về cho đến khi nhận được cặp ký tự “\r\n”
- dòng 20, 21: Điều khiển bật LED khi nhận được “ON\r\n”
- dòng 22, 23: Điều khiển tắt LED khi nhận được “OFF\r\n”

4.3. Truyền thông Bluetooth

Bài 04

I Lưu tập lệnh ví dụ vào vị trí mong muốn và chạy nó

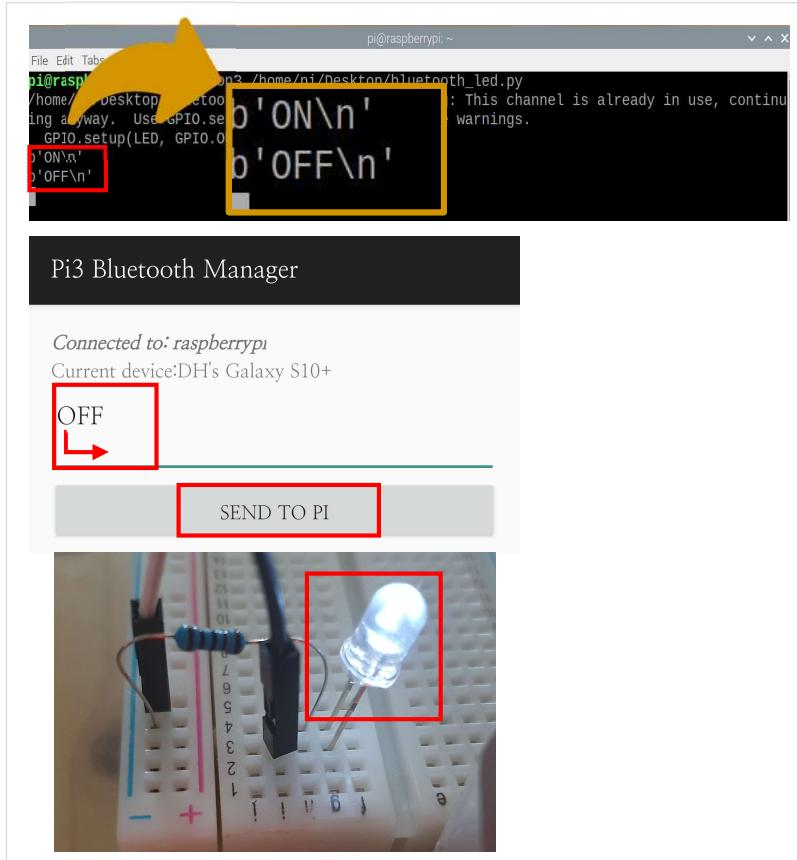


```
pi@raspberrypi: ~$ python3 /home/pi/Desktop/bluetooth_led.py
/home/pi/Desktop/bluetooth_led.py:7: RuntimeWarning: This channel is already in use, continuing anyway. Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(channel, GPIO.OUT)
b'ON\n'
b'OFF\n'

pi@raspberrypi: ~$ sudo apt-get install python3-flask
Reading package lists... done
Building dependency tree
Reading state information... Done
python3-flask is already the newest version (1.0.2-3).
The following package was automatically installed and is no longer required:
  python-colorama
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@raspberrypi: ~$
```

- ▶ Lưu tập lệnh dưới dạng bluetooth_led.py thành /home/pi/Desktop/bluetooth_led.py
- ▶ Nếu bạn chạy tệp tập lệnh bằng python3 /home/pi/Desktop/bluetooth_led.py, tệp sẽ chuyển sang trạng thái chờ đầu vào.

| Lưu tập lệnh ví dụ vào vị trí mong muốn và chạy nó



- ▶ Nhập BẬT\n hoặc TẮT\n vào trường văn bản và nhấn GỬI ĐẾN PI trong điện thoại thông minh. (Đối với \n, nhấn Enter trên bàn phím. Nhập BẬT (hoặc TẮT) rồi nhấn Enter.)
 - ▶ Bạn sẽ thấy đèn LED bật và tắt và các giá trị đầu ra thay đổi trên màn hình đầu cuối bằng cách nhấn GỬI ĐẾN PI.

Bài 4.

Hướng dẫn giao tiếp Socket with Raspberry Pi

- | 4.1. Truyền thông nối tiếp
- | 4.2. Truyền thông SPI
- | 4.3. Truyền thông Bluetooth
- | **4.4. Máy chủ web Flask**

Máy chủ web

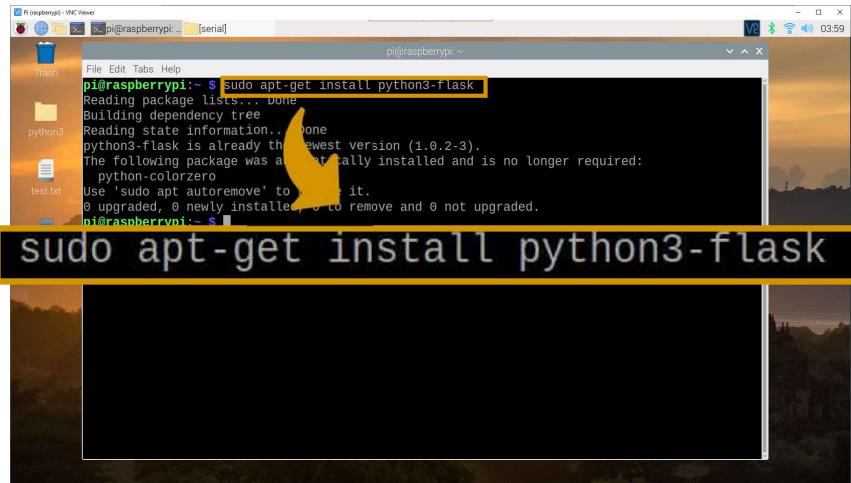
I Máy chủ Web là gì?



- ▶ Máy chủ Web là một chương trình chạy trên máy chủ lưu trữ trang web.
- ▶ Máy chủ Web đáp ứng các yêu cầu từ trình duyệt Web qua giao thức HTTP.
- ▶ Mặc dù có nhiều phần mềm máy chủ web, chúng tôi sẽ sử dụng Flask vì nó nhẹ và dễ cài đặt. Flask phù hợp với Raspberry Pi do dung lượng bộ nhớ hạn chế.
- ▶ Flask giống một môi trường phát triển web hơn là hoạt động như một máy chủ web hoàn chỉnh và triển khai các chức năng của máy chủ web bằng Python.
- ▶ ※ Xem thêm thông tin về Flask:
<https://flask.palletsprojects.com/en/2.0.x/>

Máy chủ web Flask

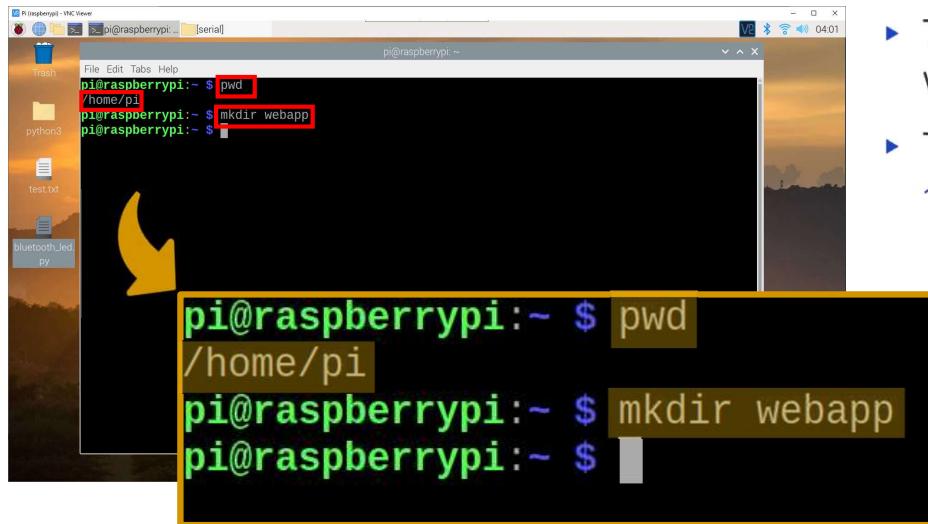
I Cài đặt Flask web của máy chủ



- ▶ Xem thêm thông tin về Flask:
<https://flask.palletsprojects.com/en/2.0.x/>
- ▶ Để cài đặt Flask, hãy thực hiện lệnh **sudo apt-get install python3-flask** trong terminal và cài đặt nó.

> sudo apt-get install python3-flask

I Thực hành Flask



```
pi@raspberrypi:~ $ pwd  
/home/pi  
pi@raspberrypi:~ $ mkdir webapp  
pi@raspberrypi:~ $
```

- ▶ Thu thập các tệp sẽ được sử dụng để triển khai Flask web của Máy chủ vào một thư mục.
- ▶ Tạo một thư mục có tên [webapp](#) tại vị trí [~/home/pi/rasp_ex](#).

~/webapp/index.py

| Thực hành Flask

- ▶ Thực hiện một chương trình thử nghiệm đơn giản, xuất ra cụm từ “Hello Flask” và gửi cho máy khách khi kết nối tại cổng 80 của Raspberry Pi thông qua máy khách (trình duyệt Internet trên PC).

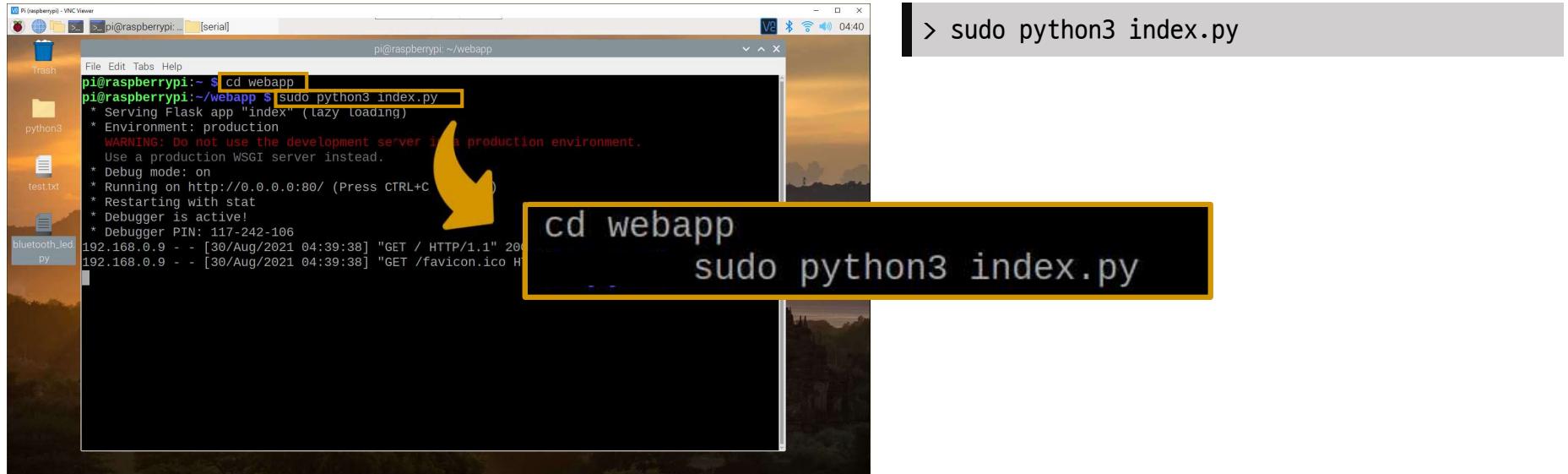
```
1  from flask import Flask  
2  
3  app = Flask(__name__)  
4  
5  @app.route('/')  
6  def hello():  
7      return 'Hello Flask'  
8  
9  ► if __name__ == '__main__':  
10     app.run(debug=True, port=80, host='0.0.0.0')
```

- dòng 1: Khai báo gói thư viện Flask
- dòng 3: Tạo phiên bản Flask
- dòng 5: Xử lý của máy chủ web (Flask) khi nhận được một yêu cầu với đường dẫn URL có dạng ‘/’
- dòng 6-7: Gửi phản hồi với thông điệp “Hello Flask”
- dòng 9-10: Thiết lập và chạy dịch vụ Web, tại cổng 80 (cổng dịch vụ mặc định, theo giao thức HTTP)

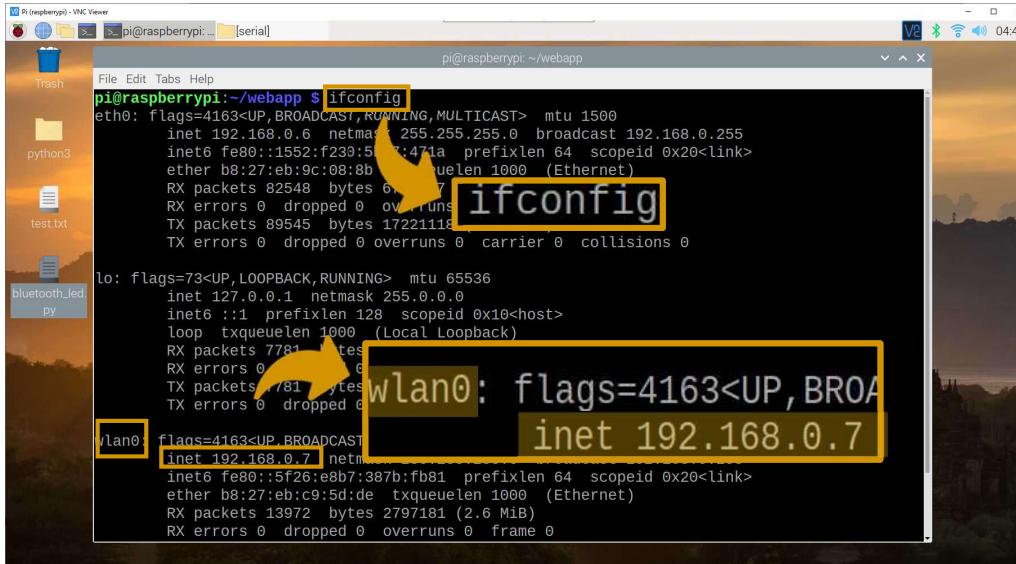
Thực hành Flask

```
1  from flask import Flask  
2  
3  app = Flask(__name__)  
4  
5  @app.route('/')  
6  def hello():  
7      return 'Hello Flask'  
8  
9  if __name__ == '__main__':  
10     app.run(debug=True, port=80, host='0.0.0.0')
```

- ▶ Chỉ định URL '/' có nghĩa là chỉ định vị trí mặc định để truy cập khi không có gì ngoài tên miền hoặc địa chỉ IP được chỉ định trong trình duyệt web.
- ▶ Bạn cần có quyền root để triển khai dịch vụ Flask.
- ▶ Di chuyển đến thư mục **Ứng dụng web** và nhập **sudo python3 index.py** để thực thi tệp Python được tạo với quyền root.



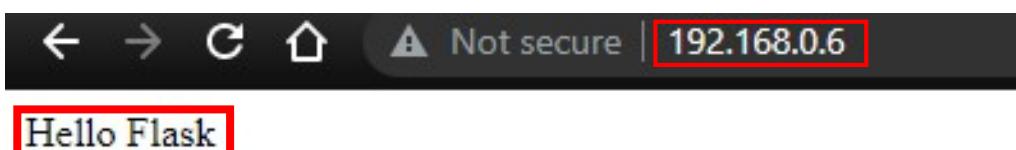
| Thực hành Flask



```
pi@raspberrypi: ~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.0.6 netmask 255.255.255.0 broadcast 192.168.0.255
        inet6 fe80::1552:f230:5e87:471a prefixlen 64 scopeid 0x20<link>
              ether b8:27:eb:9c:08:8b txqueuelen 1000 (Ethernet)
        RX packets 82548 bytes 6777777
        RX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
        TX packets 89545 bytes 17221118
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
          loop txqueuelen 1000 (Local Loopback)
        RX packets 7781 bytes 6060
        RX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
        TX packets 7781 bytes 6060
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,NOARP> mtu 1500
        inet 192.168.0.7 netmask 255.255.255.0 broadcast 192.168.0.255
        inet6 fe80::5f26:8b7:387b:fb81 prefixlen 64 scopeid 0x20<link>
              ether b8:27:eb:c9:5d:de txqueuelen 1000 (Ethernet)
        RX packets 13972 bytes 2797181 (2.6 MiB)
        RX errors 0 dropped 0 overruns 0 frame 0
```



- ▶ Địa chỉ IP phải được nhập khi kết nối từ trình duyệt web của PC. Khi nhập lệnh `ifconfig`, thông tin địa chỉ của Raspberry Pi được xác định cạnh trường thông tin `inet` bên dưới thông tin địa chỉ của cổng `wlan0`.
- ▶ Nếu bạn mở trình duyệt web đến địa chỉ IP, bạn sẽ thấy Hello Flask.

Thêm Trang Web

| @app.route()

- ▶ Có thể thêm nhiều trang dịch vụ bằng hàm @app.route().
- ▶ Hãy thêm trang phụ1.
- ▶ Sửa đổi ~/rasp_ex/webapp/index.py bạn đã viết để thêm nó.

```
1  from flask import Flask
2
3  app = Flask(__name__)
4
5  @app.route('/')
6  def hello():
7      return 'Hello Flask'
8
9  @app.route('/sub1')
10 def sub1():
11     return 'SUB1 Page'
12
13 if __name__ == '__main__':
14     app.run(debug=True, port=80, host='0.0.0.0')
```

- dòng 9 : Khi có Request /sub1
- dòng 10 : hàm sub1() thực thi.

| @app.route()

The screenshot shows a VNC session on a Raspberry Pi. In the terminal window, the command `sudo python3 index.py` is being run, and the output shows the Flask application starting up. A yellow arrow points to the command in the terminal. Below the terminal is a browser window with the address bar containing `192.168.0.6/sub1`, which is highlighted with a red box. The page content is "SUB1 Page", also highlighted with a red box.

- ▶ Thực thi index.py đã sửa đổi bằng cách nhập lệnh `sudo python3 index.py`.
- ▶ Sau khi thực hiện, hãy mở trình duyệt web và nhập địa chỉ IP bạn đã sử dụng và thêm `/sub1` sau đó và chạy nó. Bạn có thể xem cụm từ SUB1 Page.

| > `sudo python3 index.py`

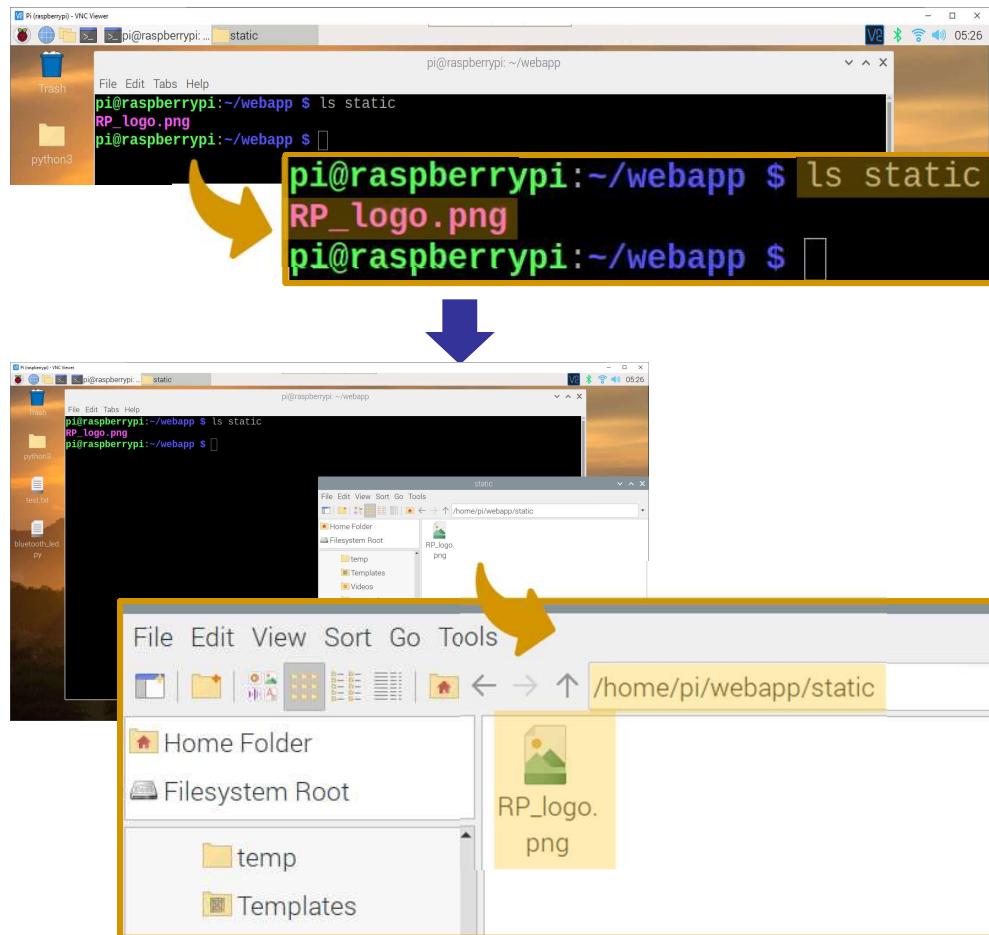
Tạo trang HTML

| Hiển thị nhiều hình ảnh, siêu văn bản, bảng, v.v. trên một trang web

```
pi@raspberrypi:~/webapp$ mkdir templates static
pi@raspberrypi:~/webapp$ ls
index.py  static  templates
pi@raspberrypi:~/webapp$
```

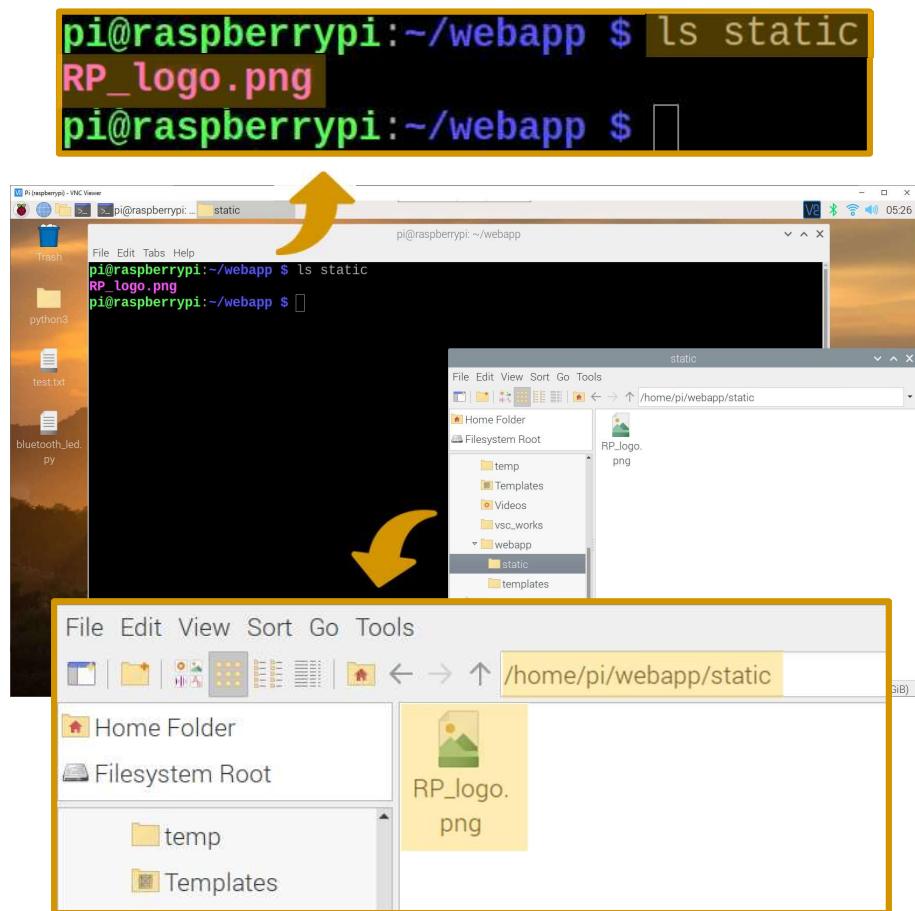
- ▶ Để hiển thị nhiều thông tin khác nhau trên trang web không chỉ là thông báo đơn giản, bạn có thể tạo một trang HTML.
- ▶ Để xuất trang HTML, hãy tạo một tệp HTML và thêm nó vào thư mục sẽ được Flask phục vụ.
- ▶ Nếu bạn muốn hiển thị hình ảnh hoặc video trong HTML, tệp hình ảnh và tệp video cần được chuẩn bị.
- ▶ Viết mã Python hiển thị văn bản và hình ảnh bằng các thẻ HTML đơn giản bằng hàm `render_template()` trong Flask.
- ▶ Để hiển thị các tệp HTML, các mẫu và thư mục tĩnh cũng cần có trong thư mục nơi mã Python thực thi. Vì vậy, trước tiên hãy tạo chúng.

I Hiển thị nhiều hình ảnh, siêu văn bản, bảng, v.v. trên một trang web



- ▶ https://media.vlpt.us/images/mythos/post/5584cec3-97f8-45b4-a8c8-003c574f732c/Raspberry_Pi_OS_Logo.png
- ▶ Ví dụ, tải xuống tệp img để sử dụng.
- ▶ Tải xuống tệp png hình ảnh, hiển thị và đặt nó vào thư mục tĩnh.

I Hiển thị nhiều hình ảnh, siêu văn bản, bảng, v.v. trên một trang web



- ▶ Tải xuống tệp png hình ảnh bạn muốn hiển thị và lưu nó trong thư mục tĩnh.
- ▶ Trong ví dụ này, tệp hình ảnh được sử dụng từ liên kết bên dưới.

https://media.vlpt.us/images/mythos/post/5584cec3-97f8-45b4-a8c8-003c574f732c/Raspberry_Pi_OS_Logo.png

| ~/webapp/templates/test_html.html

```
1  <html>
2  |  <head>
3  |  |  <title>Flask HTML test page</title>
4  |  |</head>
5  |
6  |  <body>
7  |  |  <center>
8  |  |  |  <br>
9  |  |  |  <strong>Raspberry Pi 4 Flask HTML Test Page</strong>
10 |  |  |  <br>
11 |  |  |  
12 |  |</center>
13 |  |</body>
14 |
15 |</html>
```

- ▶ Đầu tiên, hãy tạo một trang HTML đơn giản. Chúng ta sẽ tìm hiểu ngắn gọn cách tạo một trang web thay vì bao gồm tất cả các thẻ HTML.
- ▶ Tất cả các thẻ HTML phải được đặt trong <> các dấu ngoặc này.
- ▶ Thẻ mở <tag> bắt đầu một phần của nội dung trang và thẻ đóng </tag> kết thúc phần đó.
<tag> nội dung </tag>
- ▶ <title> tag: tiêu đề của trang web
- ▶ <body>: nội dung của trang web
- ▶ : hiệu ứng nổi bật
- ▶ : Viết tắt của nguồn hình ảnh. Nó được sử dụng để tải một tập tin hình ảnh. Nhập đường dẫn hình ảnh như thế này .
- ▶ Sau trang HTML, hãy lưu nó dưới dạng test_html.html trong các mẫu.

| Write ~/webapp/html_test.py

```
1  from flask import Flask, render_template
2
3  app = Flask(__name__)
4
5  @app.route('/')
6  def index():
7      return render_template('html_test.html')
8
9  if __name__ == '__main__':
10     app.run(debug=True, port=80, host='0.0.0.0')
```

```
pi@raspberrypi:~/webapp $ tree
.
├── html_test.py
├── index.py
└── static
    └── RP_logo.png
└── templates
    └── html_test.html

2 directories, 4 files
pi@raspberrypi:~/webapp $
```

- ▶ Kết xuất (tạo và xem) tài liệu HTML bằng Python..
 - dòng 7: Kết xuất tệp html_test.html

- ▶ Khi bạn tạo và lưu html_test.py, cấu trúc được tạo như bạn thấy ở bên trái.

I Xem trang đã tạo trên web

```
pi@raspberrypi:~/webapp $ sudo python3 html_test.py
 * Serving Flask app "html_test" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead!
 * Debug mode: on
 * Running on http://0.0.0.0:5001 (Press CTRL+C to quit)
 * Restarting
 * Debugger is active
 * Debugger PIN: 000
```

A yellow arrow points from the text "sudo python3 html_test.py" in the terminal to the same command in the browser screenshot below.

- ▶ Nhập sudo python3 html_test.py để thực thi tập lệnh Python đã tạo.
- ▶ > sudo python3 html_test.py
- ▶ Nhập địa chỉ IP vào trình duyệt web.
- ▶ Hình ảnh dưới đây là đầu ra.



GET, POST

I Các trang web (trang HTML) đơn không hỗ trợ nội dung động

- Trong Flask, đầu ra động có thể thực hiện được bằng cách nhận các tham số được gửi từ máy khách web.
- Có hai phương thức để truyền các tham số từ máy khách web sang máy chủ web (Flask): GET và POST.

GET

- Yêu cầu gửi kèm theo dữ liệu chứa trong URL
- Dữ liệu cần truyền lưu trong phần tiêu đề của bản tin GET
- Không có tính bảo mật vì dữ liệu trong URL có thể xác định được.
- Có thể sử dụng bộ nhớ đệm



POST

http://example.php



GET

http://example.php?key=value



POST

- Không để lộ các biến (Dữ liệu) trong URL
- Dữ liệu cần truyền đặt trong phần thân (body) của bản tin POST
- Dữ liệu không bị lộ trong URL, vì vậy hỗ trợ cơ chế bảo mật cơ bản
- Phải sử dụng bộ nhớ đệm để lưu tạm thời dữ liệu trước khi xử lý

GET

| Tham số trong phương thức GET

- ▶ Trong phương thức GET, ta có thể đặt các tham số (giá trị cần truyền cho máy chủ) trong đường dẫn URL của trình duyệt web hoặc gửi bản tin yêu cầu theo dạng GET, trong đó dữ liệu chứa trong phần tiêu đề của bản tin.
- ▶ Hãy cùng thực hành với ví dụ về truyền và nhận tham số.
- ▶ Để Yêu cầu một tham số trong phương thức GET, hãy bắt đầu bằng "?" trong URL của trang web trên máy chủ và viết nó ở dạng "tên=giá trị".
- ▶ Để gửi nhiều tham số, hãy tách chúng bằng dấu "&".

| Phương thức GET

- ▶ Đó là một phương thức để gửi một yêu cầu tới máy chủ web, bằng cách nhập tham số vào URL ở dạng cơ bản nhất.

```
1  from flask import Flask, request, render_template
2
3  app = Flask(__name__)
4
5  @app.route('/method', methods=['GET'])
6  def method():
7      if request.method == 'GET':
8          id = request.args['id']
9          password = request.args.get("password")
10         return "Transferred by GET method ({}, {})".format(id, password)
11
12 if __name__ == '__main__':
13     app.run(debug=True, port=80, host='0.0.0.0')
14
```

- dòng 5 : Khi nhận được một yêu cầu theo phương thức GET
- dòng 7 : Nếu phương thức yêu cầu nhận được là GET
- dòng 8: lấy về thông tin của trường “id” trong URL
- dòng 9: xác định giá trị “mật khẩu” trong URL

- ▶ Đây là một tập lệnh python xuất các tham số đầu vào vào theo mẫu trên trang.

| Tham số phương thức GET

- ▶ Phương thức nhập tham số vào URL và gửi chúng là ví dụ cơ bản nhất của phương thức GET.
- ▶ Tuy nhiên, phương pháp này có nhược điểm là mỗi lần gõ tham số vào thanh địa chỉ trên trình duyệt, ta cần xây dựng một file HTML dành riêng và xử lý yêu cầu trong cùng một phương thức GET.
- ▶ Đầu tiên, hãy xây dựng một file `method_get.html` trước khi xây dựng mã lệnh Python. File html này là file để thực hiện các chức năng nhập thông tin dưới (dạng form) tương tự như khi chúng ta đăng nhập từ một trang web.

| ~/rasp_ex/webapp/templates/method_get.html

```
1  <html>
2  <head>
3      <title>GET Method Request Test</title>
4  </head>
5
6  <body>
7      <h2>ID:{{ id }}, PASSWORD:{{ password }}</h2>
8      <form method="get" action="/method_get_act">
9          <label id="Label1">id</label>
10         <input name="id" type="text" />
11         <label id="Label1"><br />password </label>
12         <input name="password" type="text" /><br /><br />
13         <input name="Submit1" type="submit" value="submit" />
14     </form>
15 </body>
16 </html>
```

- dòng 7: Nhận giá trị truyền từ Flask Python Script và xuất ra theo định dạng
- dòng 8-14: Tạo một form nhập liệu để nhận giá trị id và mật khẩu đăng nhập và gửi thông tin thông qua nút gửi (submit)
- dòng 8: Nếu phương thức trong thẻ biểu mẫu là get, Yêu cầu trong phương thức GET. Mã sau hành động = quyết định chức năng nào của Flask Server sẽ gọi khi nhấn nút gửi

- ▶ Khi trình duyệt ban đầu gửi yêu cầu với đường dẫn URL chứa “method_get.html”, không có gì được hiển thị trong {{id}} và {{password}} ở dòng 7 vì không có dữ liệu nào được truyền vào trong chương trình máy chủ Flask.
- ▶ Lưu tệp này ở cùng dạng với ~/rasp_ex/webapp/templates/method_get.html.

| ~/webapp/method_get.py

- ▶ Viết tập lệnh python sẽ chuyển tham số tới method_get.html mà bạn đã viết.

```

1  from flask import Flask, request, render_template
2
3  app = Flask(__name__)
4
5  @app.route('/method_get', methods=['GET'])
6  def method_get():
7      return render_template('method_get.html')
8
9  @app.route('/method_get_act', methods=['GET'])
10 def method_get_act():
11     if request.method == 'GET':
12         id = request.args['id']
13         password = request.args.get("password")
14         return render_template('method_get.html', id=id, password=password)
15
16 if __name__ == '__main__':
17     app.run(debug=True, port=80, host='0.0.0.0')
18

```

- dòng 5-7: Xử lý yêu cầu đối với trang web method_get.html
- dòng 9: xử lý với thẻ <form method="get" action="/method_get_act"> trong file html
- dòng 12-13 : Đọc "id" và "password" từ yêu cầu gửi từ method_get.html
- dòng 14: Truyền tham số id và mật khẩu vào method_get.html

| > sudo python3 method_get.py

- ▶ Sau khi lưu tệp này vào ~/rasp_ex/webapp/method_get.py, hãy thực thi method_get.py bằng lệnh python3.



```

pi@raspberrypi:~/rasp_ex/webapp $ ls
html_test.py index.py method_get.py static templates
pi@raspberrypi:~/rasp_ex/webapp $ sudo python3 method_get.py
* Serving Flask app "method_get" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production setting. Use a production WSGI server instead.
* Debug mode: on
* Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 117-242-106

```

pi@raspberrypi:~/webapp \$ ls

html_test.py index.py method_get.py static templates

pi@raspberrypi:~/webapp \$ sudo python3 method_get.py

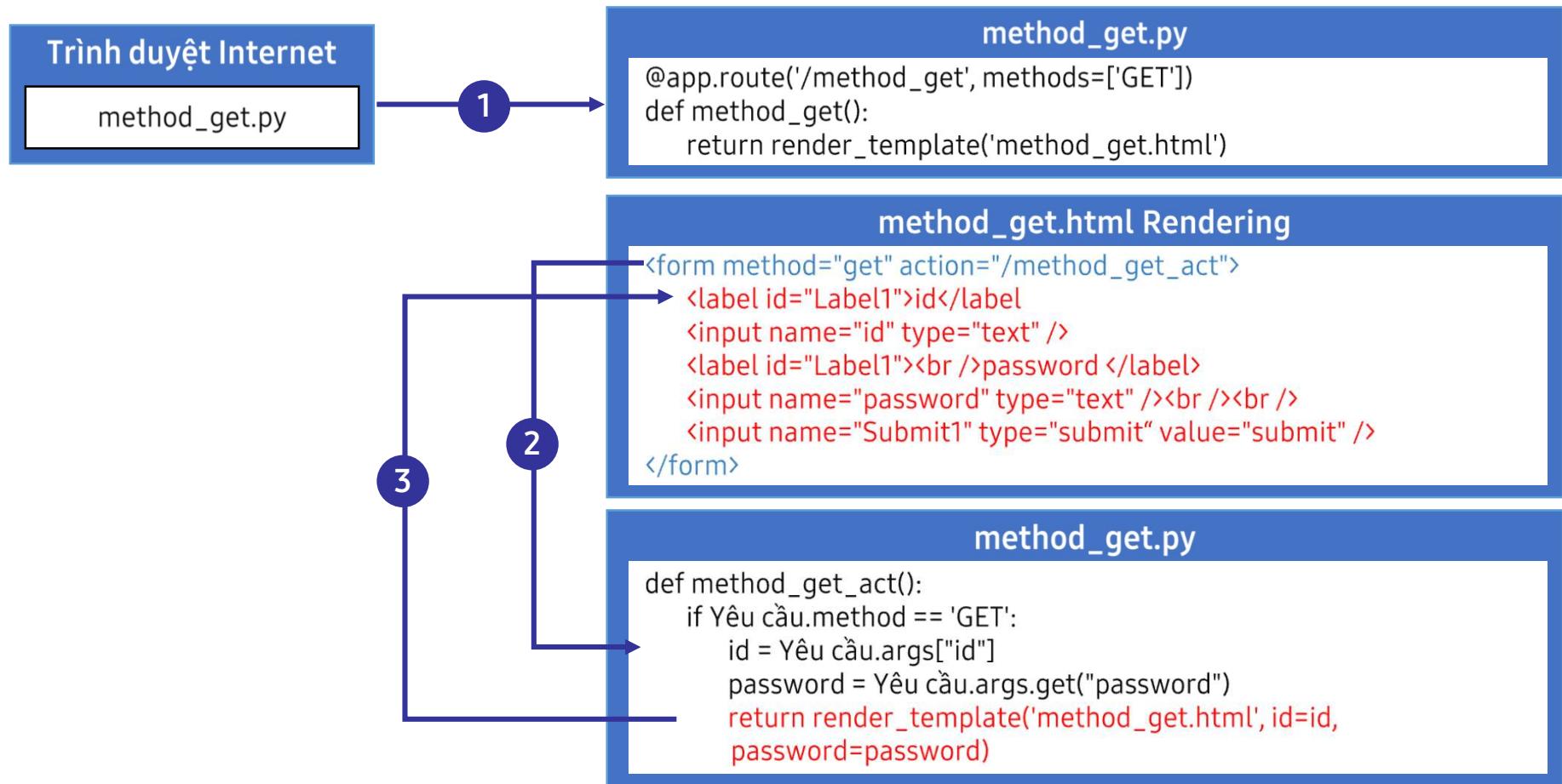
I màn hình kết nối

The figure consists of three vertically stacked screenshots of a web browser window. All screenshots show the URL `192.168.0.5/method_get` in the address bar.

- Screenshot 1:** The browser displays a form titled "ID:, PASSWORD:". It contains two input fields: "id" and "password", and a "submit" button. A blue circle labeled "1" is positioned next to the URL bar. An orange arrow points from the URL bar down to the "id" input field.
- Screenshot 2:** The browser displays the same form after a submission. The "id" field now contains "Daniel" and the "password" field contains "1234". The entire form area is highlighted with a yellow box. A blue circle labeled "2" is next to the URL bar. An orange arrow points from the URL bar down to the "id" input field.
- Screenshot 3:** The browser displays the results of the submission. The URL has changed to `192.168.0.5/method_get_act?id=Daniel&password=1234&Submit1=submit`. The page content shows the text "ID:Daniel, PASSWORD:1234" in a yellow box. A blue circle labeled "3" is next to the URL bar. An orange arrow points from the URL bar down to the page content.

- ▶ Khi kết nối qua trình duyệt web với địa chỉ IP `/method_get` của Raspberry Pi, bạn có thể thấy màn hình hiển thị bên trái.
- ▶ Hiện tại chưa nhận được ID hoặc Password, id và password trông như hình bên trái.
- ▶ Nhập id và mật khẩu mong muốn của bạn và nhấp vào gửi.
- ▶ Bạn có thể thấy nội dung ID và MẬT KHẨU đã được thay đổi trên màn hình của trang web.

Thứ tự các cuộc gọi giữa trình duyệt web và máy chủ Flask



POST

| Tham số phương thức POST

- ▶ Không giống như phương thức GET, phương thức POST không truyền dữ liệu thông qua tham số trong URL trong một yêu cầu gửi tới máy chủ.
- ▶ Phương thức POST phải được gọi với thuộc tính method="post" trong thẻ HTML "<form method='post' action='/method_get_act'>"
- ▶ Hãy lấy id và mật khẩu đã nhập bằng phương thức POST.

| ~/webapp/templates/method_post.html

- ▶ Mã HTML để kiểm tra phương thức đăng bài

```
1  <html>
2  <head>
3  <title>POST Method Request Test</title>
4  </head>
5
6  <body>
7  <h2>ID:{{ id }}, PASSWORD:{{ password }}</h2>
8  <form method="post" action="/method_post_act">
9    <label id="Label1">id</label>
10   <input name="id" type="text" />
11   <label id="Label1"><br />password </label>
12   <input name="password" type="text" /><br /><br />
13   <input name="Submit1" type="submit" value="submit" />
14 </form>
15 </body>
16 </html>
```

- dòng 7: Nhận giá trị truyền từ Flask Python Script và xuất ra theo định dạng
- dòng 8-14: Tạo form nhập liệu gồm id và mật khẩu nhập và nút gửi thông tin
- dòng 8: Nếu phương thức là post trong thẻ form thì Yêu cầu được thực hiện trong phương thức POST. Mã sau hành động = quyết định chức năng nào của Flask Server sẽ gọi khi nhấn nút "submit".

- ▶ Lưu tệp này ở cùng dạng với ~/rasp_ex/webapp/templates/method_post.html.

| ~/rasp_ex/webapp/method_post.py

- ▶ Tập lệnh Python để kiểm tra phương thức POST

```
1  from flask import Flask, request, render_template
2
3  app = Flask(__name__)
4
5  @app.route('/method_post', methods=['GET', 'POST'])
6  def method_post():
7      return render_template('method_post.html')
8
9  @app.route('/method_post_act', methods=['GET', 'POST'])
10 def method_post_act():
11     if request.method == 'POST':
12         id = request.form["id"]
13         password = request.form["password"]
14         return render_template('method_post.html', id=id, password=password)
15
16 ▶ if __name__ == '__main__':
17     app.run(debug=True, port=80, host='0.0.0.0')
```

- dòng 1: Truyền tham số id và mật khẩu cho method_post.html
- dòng 5-7: xử lý yêu cầu với method_post.html
- dòng 9: Được gọi bởi thẻ <form method="get" action="/method_post_act"> trong html
- dòng 12-13: Đọc "id" và "password" trong bản tin yêu cầu đối với method_post.html. Không giống như phương thức GET, ta có thể đọc các tham số bằng cách đọc giá trị thẻ HTML FORM bằng cách sử dụng biểu mẫu tương tự như Yêu cầu.form[""]

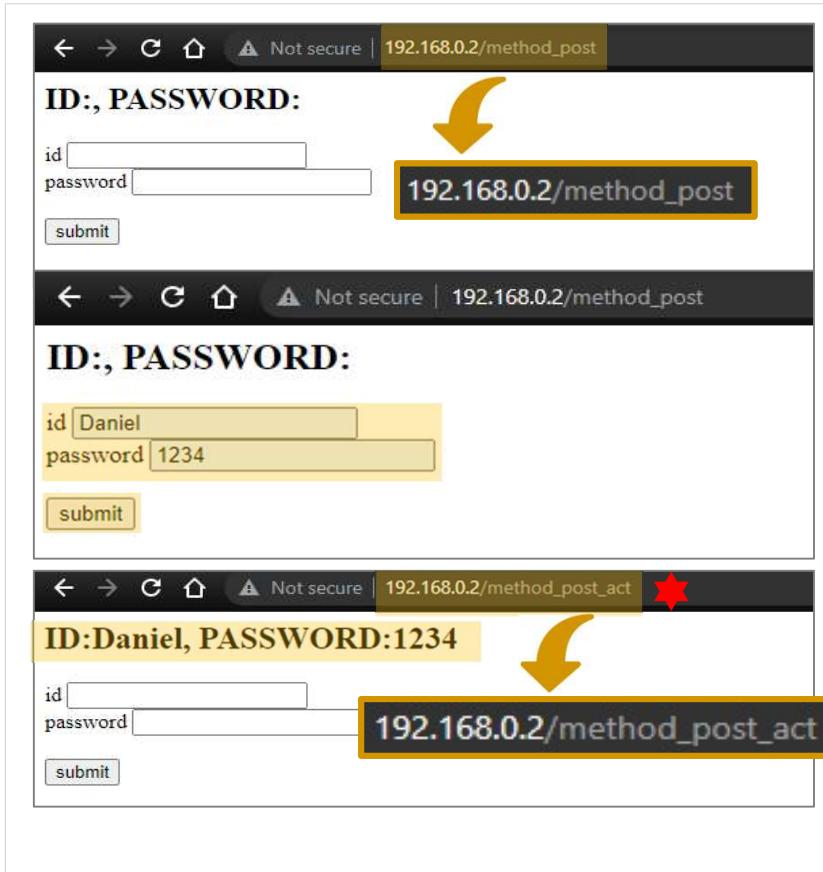
| ~/rasp_ex/webapp/method_post.py

- ▶ Sau khi lưu tệp này vào vị trí ~/rasp_ex/webapp/method_post.py, hãy thực thi method_get.py bằng lệnh python3.

```
| > sudo python3 method_post.py
```

```
pi@raspberrypi:~/webapp $ pwd
/home/pi/webapp
pi@raspberrypi:~/webapp $ ls
html_test.py index.py method_get.py method_post.py static templates
pi@raspberrypi:~/webapp $ ls templates/
html_test.html method_get.html method_post.html
pi@raspberrypi:~/webapp $ sudo python3 method_post.py
 * Serving Flask app "method_post" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
           Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 117-242-106
```

I màn hình kết nối



- ▶ Khi kết nối thông qua trình duyệt web với đường dẫn địa chỉ IP/method_post, trong đó IP là địa chỉ của Raspberry Pi, bạn có thể thấy màn hình hiển thị bên trái.
- ▶ lúc đầu id và mật khẩu trống vì chưa xác định được id hoặc mật khẩu.
- ▶ Nhập ID và mật khẩu mong muốn vào đây và nhấp vào gửi.
- ▶ Bạn có thể thấy rằng ID và MẬT KHẨU được chỉ định khi bạn nhập.
- ▶ Ở đây, **điểm khác biệt** so với GET là trong POST, ID và mật khẩu đã gửi không được hiển thị dưới dạng tham số trong đường dẫn URL của yêu cầu từ trình duyệt web.
- ▶ Bạn có thể tạo Yêu cầu bằng phương thức GET để truyền tham số đơn giản và sử dụng phương thức POST để truyền mật khẩu cần được bảo mật hoặc lượng dữ liệu lớn đến máy chủ web.

Web GPIO Control

| Điều khiển BẬT/TẮT đèn LED thông qua Web

```

1  from flask import Flask, request, render_template
2  import RPi.GPIO as GPIO
3
4  LED = 17
5  GPIO.setmode(GPIO.BCM)
6  GPIO.setup(LED, GPIO.OUT)
7
8  app = Flask(__name__)
9
10 @app.route('/led_control')
11 def led_control():
12     return render_template('led_control.html')
13
14 @app.route('/led_control_act', methods=['GET'])
15 def led_control_act():
16     if request.method == 'GET':
17         status = ''
18         led = request.args["led"]
19         if led == '1':
20             GPIO.output(LED, True)
21             status = 'ON'
22         else:
23             GPIO.output(LED, False)
24             status = 'OFF'
25     return render_template('led_control.html', ret=status)
26
27 if __name__ == '__main__':
28     app.run(debug=True, port=80, host='0.0.0.0')

```

- Hãy tạo các nút LED ON và LED OFF trong trang HTML để điều khiển đèn LED được kết nối với Raspberry Pi.
- Đầu tiên, tạo và lưu tệp ~/rasp_ex/webapp/led_control.py
- dòng 15-25: hàm led_control_act() là hàm điều khiển led, bật/tắt led theo tham số led trong bản tin yêu cầu GET gửi tới máy chủ
- Nếu giá trị là 1 sẽ bật đèn LED, nếu không thì tắt.



```

15 def led_control_act():
16     if request.method == 'GET':
17         status = ''
18         led = request.args["led"]
19         if led == '1':
20             GPIO.output(LED, True)
21             status = 'ON'
22         else:
23             GPIO.output(LED, False)
24             status = 'OFF'
25     return render_template('led_control.html', ret=status)

```

- Truyền trạng thái cho một biến có tên là **ret** trong html (trình duyệt web)

| Điều khiển BẬT/TẮT đèn LED thông qua Web

```
1  <html>
2  <head>
3  <title>WEB LED Control</title>
4  <style type="text/css">
5      .auto-style1 {
6          text-align: center;
7      }
8      .auto-style3 {
9          background-color: #008000;
10     }
11     .auto-style6 {
12         border-style: solid;
13         border-color: #000000;
14         text-align: center;
15         color: #FFFFFF;
16         background-color: #FF9900;
17     }
18 </style>
19 </head>
20 <body>
21 <center>
22 <strong><br>HOME IOT Service<br><br></strong>
23 <table style="width: 50%">
24 <tr>
25     <td class="auto-style6" style="height: 81; width: 30%"><strong>
26         <a href="led_control_act?led=1">LED ON</a></strong></td>
27     <td class="auto-style1" style="height: 81; width: 50%"> </td>
28     <td class="auto-style6" style="height: 81; width: 30%; "><strong>
29         <a href="led_control_act?led=2">LED OFF</a></strong></td>
30     </tr>
31 </table>
32 <strong><br>LED is {{ ret }}</strong>
33 </center>
34 </body>
35 </html>
```

- ▶ Viết và lưu tệp ~/rasp_ex/webapp/template/led_control.html
 - dòng 24-30: Định nghĩa các nút để điều khiển đèn LED
 - dòng 26, 29: Truyền vào số nguyên tương ứng cho tham số led trong yêu cầu được gửi bởi hàm led_control_act() của tệp led_control.py
 - dòng 32: Hiển thị trạng thái của đèn LED tương ứng với giá trị của biến ret được trả về trong mã lệnh led_control.py

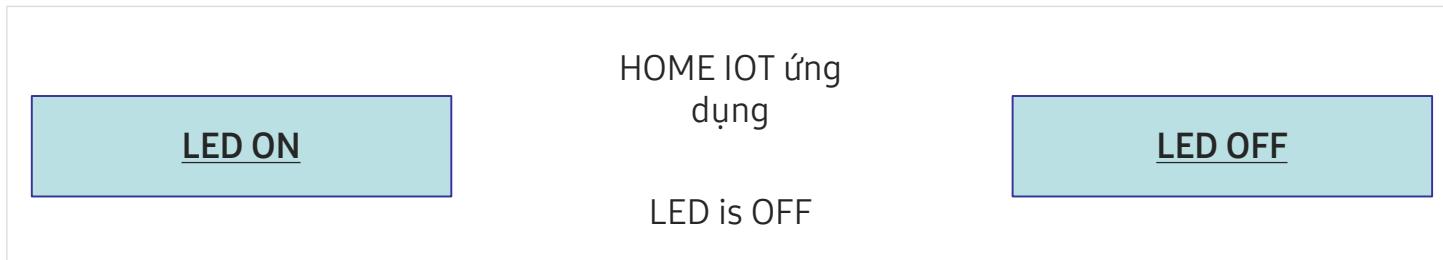
```
24 <tr>
25     <td class="auto-style6" style="height: 81; width: 30%"><strong>
26         <a href="led_control_act?led=1">LED ON</a></strong></td>
27     <td class="auto-style1" style="height: 81; width: 50%"> </td>
28     <td class="auto-style6" style="height: 81; width: 30%; "><strong>
29         <a href="led_control_act?led=2">LED OFF</a></strong></td>
30     </tr>
31 </table>
32 <strong><br>LED is {{ ret }}</strong>
```

| Điều khiển BẬT/TẮT đèn LED thông qua Web

- ▶ Sau khi lưu led_control.py và led_control.html, hãy thực hiện lệnh sudo python3 ~/webapp/led_control.py.
- ▶ Khi kết nối thông qua trình duyệt web với đường dẫn IP/led_control , trong đó IP là địa chỉ của Raspberry Pi, bạn có thể thấy màn hình bên dưới.
- ▶ Bằng cách nhấn từng nút, bạn có thể kiểm tra văn bản của trang web thay đổi như thế nào và đèn LED thay đổi như mong muốn.

```
> sudo python3 rasp_ex/webapp/led_control.py
```

```
pi@raspberrypi:~ $ sudo python3 rasp_ex/webapp/led_control.py
* Serving Flask app "led_control" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
* Restarting with stat
/home/pi/webapp/led_control.py:6: RuntimeWarning: This channel is already in use, continuing anyway.  Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(LED, GPIO.OUT)
* Debugger is active!
* Debugger PIN: 117-242-106
```



Bài 5.

Sử dụng Raspberry Pi làm Máy chủ DB

- | 5.1. Sử dụng MariaDB
- | 5.2. Sử dụng HeidiSQL
- | 5.3. Ứng dụng của MariaDB

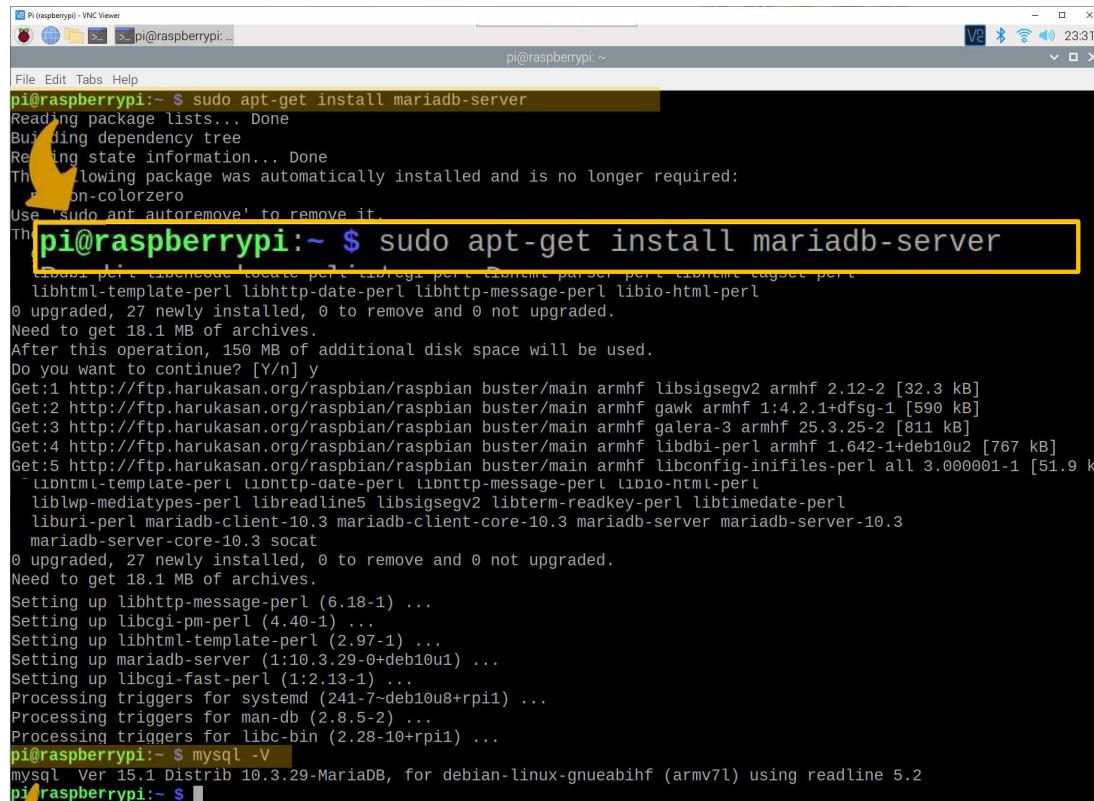
MariaDB

| Một hệ thống cơ sở dữ liệu quan hệ mã nguồn mở

- ▶ MariaDB là một hệ thống quản lý cơ sở dữ liệu quan hệ mã nguồn mở (RDBMS).
- ▶ MariaDB dựa trên cùng một mã nguồn với MySQL, vì vậy nó có giấy phép GPL v2
- ▶ Nhà phân phối phải chia sẻ bản quyền với Monty Program AB để duy trì khả năng tương thích cao với MySQL
- ▶ MariaDB dùng chung mã nguồn với MySQL nên cách sử dụng và cấu trúc cũng giống như của MySQL.



I Cài đặt MariaDB trên Raspberry Pi



```
pi@raspberrypi:~ $ sudo apt-get install mariadb-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  python-colorzero
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  mariadb-server
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 18.1 MB of archives.
After this operation, 150 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ftp.harukasan.org/raspbian/raspbian buster/main armhf libsigsegv2 armhf 2.12-2 [32.3 kB]
Get:2 http://ftp.harukasan.org/raspbian/raspbian buster/main armhf gawk armhf 1:4.2.1+dfsg-1 [590 kB]
Get:3 http://ftp.harukasan.org/raspbian/raspbian buster/main armhf galera-3 armhf 25.3.25-2 [811 kB]
Get:4 http://ftp.harukasan.org/raspbian/raspbian buster/main armhf libdbi-perl armhf 1.642-1+deb10u2 [767 kB]
Get:5 http://ftp.harukasan.org/raspbian/raspbian buster/main armhf libconfig-inifiles-perl all 3.000001-1 [51.9 kB]
  libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libreadline5 libsigsegv2 libterm-readkey-perl libtimedate-perl
  liburi-perl mariadb-client-10.3 mariadb-client-core-10.3 mariadb-server mariadb-server-10.3
  mariadb-server-core-10.3 socat
0 upgraded, 27 newly installed, 0 to remove and 0 not upgraded.
Need to get 18.1 MB of archives.
Setting up libhttp-message-perl (6.18-1) ...
Setting up libcgi-pm-perl (4.40-1) ...
Setting up libhtml-template-perl (2.97-1) ...
Setting up mariadb-server (1:10.3.29+deb10u1) ...
Setting up libcgi-fast-perl (1:2.13-1) ...
Processing triggers for systemd (241-7-deb10u8+rpi1) ...
Processing triggers for man-db (2.8.5-2) ...
Processing triggers for libc-bin (2.28-10+rpi1) ...
pi@raspberrypi:~ $ mysql -V
mysql Ver 15.1 Distrib 10.3.29-MariaDB, for debian-linux-gnueabihf (armv7l) using readline 5.2
pi@raspberrypi:~ $
```

- Thực thi lệnh `sudo apt-get install mariadb-server` để cài đặt MariaDB trên Raspberry Pi và trả lời bằng `y` khi xuất hiện thông báo `[Y/n]`.

> `sudo apt-get install mariadb-server`

- Sau khi cài đặt xong, dùng lệnh `mysql -V` để kiểm tra phiên bản DB.

pi@raspberrypi:~ \$ mysql -V
mysql Ver 15.1 Distrib 10.3.29-MariaDB, for debian-linux-gnueabihf (armv7l) using readline 5.2

thiết lập tài khoản root

Thiết lập tài khoản “root”

- Thực hiện lệnh `sudo mysql -u root` để vào dấu nhắc lệnh của MariaDB.
- Thực hiện lệnh `use mysql;` để chọn mysql DB.
- Bạn có thể thấy rằng dấu nhắc lệnh đã được thay đổi từ MaraiDB [(None)]> thành `MariaDB[mysql]>`.

> `sudo mysql -u root`

```
Pi (raspberrypi) - VNC Viewer  
pi@raspberrypi: ~  
File Edit Tabs Help  
Created symlink /etc/systemd/system/multi-user.target.wants/mariadb.service → /lib/systemd/system/mariadb.service.  
Setting up libhttp-message-perl (6.18-1) ...  
Setting up libcgi-pm-perl (4.40-1) ...  
Setting up libhtml-template-perl (2.97-1) ...  
Setting up mariadb-server (1:10.3.29-0+deb10u1) ...  
Setting up libcgi-fast-perl (1:2.13-1) ...  
Processing triggers for systemd (241-7~deb10u8+rpi1) ...  
Processing triggers for man-db (2.8.5-2) ...  
Processing triggers for libc-bin (2.28-10+rpi1) ...  
pi@raspberrypi:~ $ mysql -V  
mysql Ver 15.1 Distrib 10.3.29-MariaDB, for debian-linux-gnueabihf (armv7l) using readline 5.2  
pi@raspberrypi:~ $ sudo mysql -u root  
Welcome to the MariaDB monitor. Commands end with ; or \q.  
Your MariaDB connection id is 1  
Server version: 10.3.29-MariaDB MariaDB Server  
pi@raspberrypi:~ $ sudo mysql -u root  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
Type 'help;' or '\h' for help. Type '\c' to clear MariaDB [(none)]> use mysql;  
MariaDB [(none)]> use mysql;  
Reading table information for completion of table and column names  
You can turn off this feature to get a quicker startup with -A  
You can turn off this feature to get a quicker startup with -A  
Database changed  
MariaDB [mysql]> Database changed  
MariaDB [mysql]> Database changed  
MariaDB [mysql]> Database changed
```

| Thiết lập tài khoản root

```
MariaDB [(none)]> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [mysql]> select user, host, password from user;
+-----+-----+-----+
| user | host   | password |
+-----+-----+-----+
| root | localhost |
+-----+-----+-----+
1 row in set (0.001 sec)

MariaDB [mysql]> update user set password=password('1234') where user='root';
Query OK, 1 row affected (0.002 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [mysql]> flush privileges;
Query OK, 0 rows affected (0.001 sec)

MariaDB [mysql]>
```

- ▶ Chọn lệnh để chọn kiểm tra các giá trị user, host, password của user tabledml của mysql DB
- ▶ chọn người dùng, máy chủ, mật khẩu từ người dùng; Đây là một lệnh cú pháp SQL để lấy các thuộc tính người dùng, máy chủ và mật khẩu từ bảng người dùng.
- ▶ Bạn có thể thấy rằng mật khẩu cho người dùng root trống. Chạy lệnh cập nhật để cập nhật mật khẩu cho người dùng root.
- ▶ Lệnh cập nhật mật khẩu của người dùng root thành 1234 là cập nhật bộ mật khẩu người dùng = mật khẩu ('1234') trong đó người dùng = 'root'.
- ▶ Các đặc quyền tạo ra lệnh; phải được thực thi để cập nhật được phản ánh chính xác.

| Thiết lập tài khoản root

```
MariaDB [mysql]> select user, host, password from user;
+-----+-----+-----+
| user | host   | password          |
+-----+-----+-----+
| root | localhost | *A4B6157319038724E3560894F7F932C8886EBFCF |
+-----+-----+
1 row in set (0.001 sec)

MariaDB [mysql]> exit
Bye
pi@raspberrypi:~ $
```



- ▶ Nếu bạn kiểm tra lại các giá trị thuộc tính của người dùng, máy chủ và mật khẩu trong bảng người dùng, bạn có thể xác nhận rằng mật khẩu đã được thay đổi.
- ▶ Bởi vì mật khẩu dường như ở trạng thái được băm, nên nó không giống như mật khẩu đã đặt, 1234, mà ở **dạng được băm**.
- ▶ Thoát khỏi dấu nhắc lệnh MariaDB bằng **lệnh thoát**.

Cài đặt quyền của người dùng DB

I Thiết lập tất cả quyền cho người dùng “root” với cơ sở dữ liệu MySQL

The screenshot shows a terminal window titled "Pi (raspberrypi) - VNC Viewer". The command entered is "pi@raspberrypi:~\$ sudo mysql -u root". The output shows the MySQL monitor welcome message, connection details, and the MySQL version. A yellow arrow points to the command line. The next command entered is "use mysql", followed by "grant all on *.* to 'root'@'%' identified by '1234';". A yellow box highlights "grant all on *.* to 'root'@'%' identified by '1234';". The response "Query OK, 0 rows affected" is shown. Then, "flush privileges" is entered, followed by "exit". A yellow box highlights "Flush privileges". The final command "exit" is also highlighted with a yellow box.

- ▶ Thực hiện lệnh `sudo mysql -u root` để vào dấu nhắc lệnh MariaDB.
 > `sudo mysql -u root`
- ▶ Thực hiện lệnh sử dụng `mysql`; để chọn mysql DB
- ▶ Ý nghĩa của lệnh là **cấp quyền** cho tất cả các bảng và DB, đồng thời đặt nó ở chế độ có thể truy cập được từ cả vị trí cục bộ và từ xa. Sau khi được xác định bởi, hãy nhập mật khẩu của người dùng root.
- ▶ Thực hiện lệnh tạo ra **đặc quyền** để áp dụng các sửa đổi
- ▶ Thực hiện lệnh Thoát để điều hướng đến thiết bị đầu cuối Raspberry Pi

Bài 5.

Sử dụng Raspberry Pi làm Máy chủ DB

- | 5.1. Sử dụng MariaDB
- | **5.2. Sử dụng HeidiSQL**
- | 5.3 Ứng dụng của MariaDB

HeidiSQL

- | Môi trường GUI (giao diện đồ họa) thuận tiện cho phép người dùng truy cập MariaDB từ xa từ PC, tạo bảng và quản lý chúng
 - ▶ HeidiSQL là một ứng dụng ở phía máy khác, mã nguồn mở, dành cho MySQL. Nó là công cụ tốt nhất để quản lý cơ sở dữ liệu MySQL.
 - ▶ Nó cung cấp một giao diện đồ họa mạnh mẽ để quản lý bảng, nhật ký và người dùng cơ sở dữ liệu MySQL.
 - Những đặc điểm chính
 - Tạo báo cáo SQL
 - Đồng bộ hóa thông tin từ hai bảng khác nhau
 - Nhập tệp văn bản và trích xuất bảng ở các định dạng khác nhau
 - Chỉnh sửa cơ sở dữ liệu bằng cách tạo và xóa bảng, chèn và chỉnh sửa nhật ký cũng như xóa nhật ký, bao gồm khu vực MEMO hỗ trợ nhiều thứ như Bitmap, GIF và JPEG

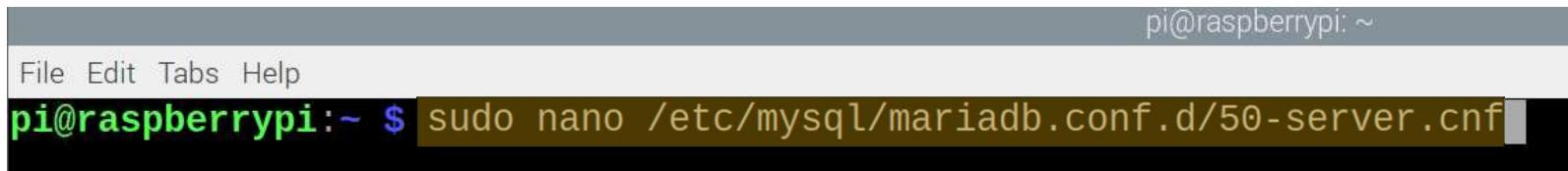


Cài đặt truy cập bên ngoài HeidiSQL

I Cài đặt để cho phép truy cập bên ngoài

- ▶ Sẽ không thành vấn đề nếu bạn chỉ kết nối từ bên trong Raspberry Pi, nhưng bạn nên cho phép truy cập bên ngoài truy cập MariaDB từ xa vào từ PC để tạo và quản lý bảng.
- ▶ Thực hiện lệnh sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf để chỉnh sửa tệp này

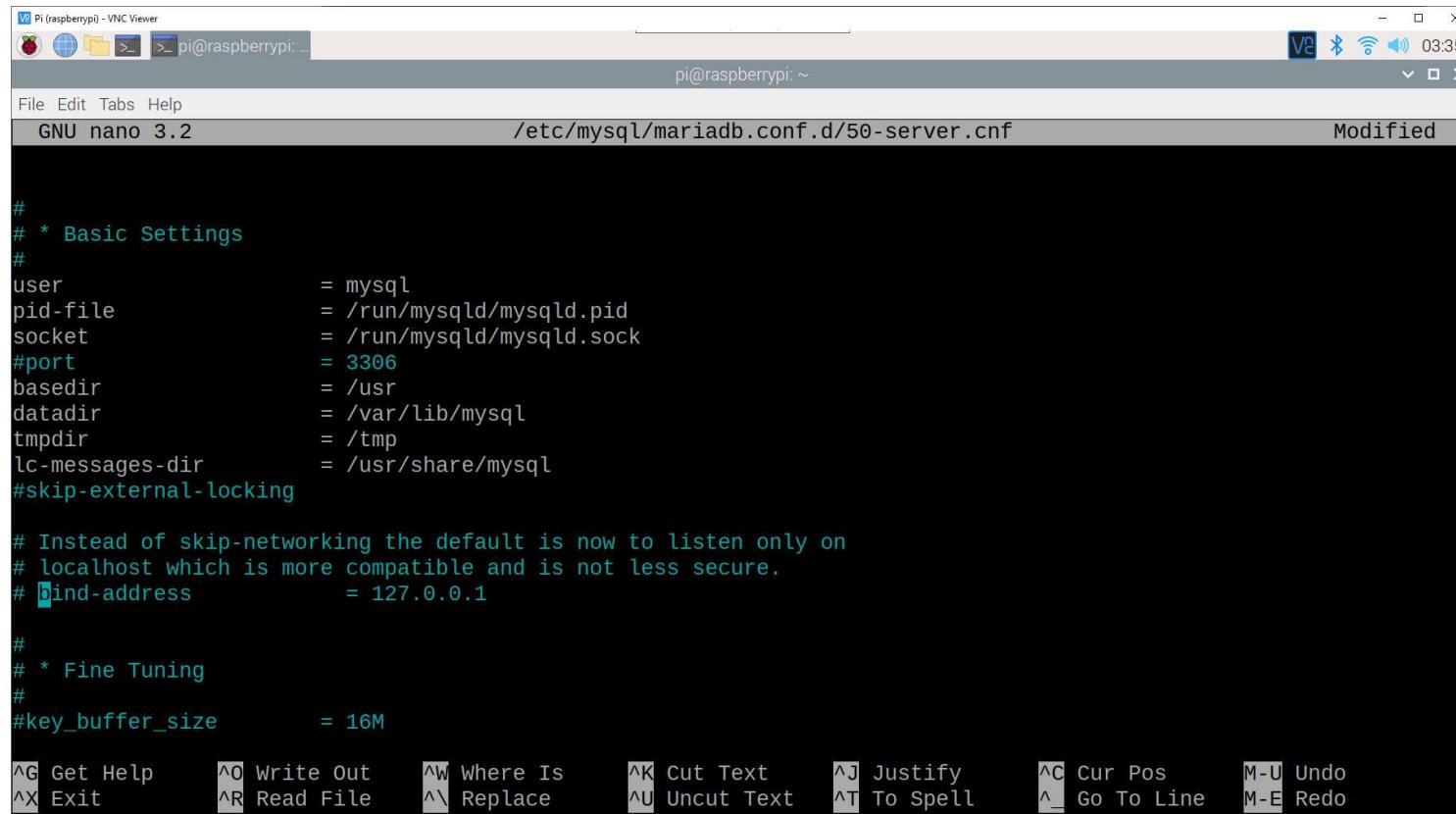
```
> sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
```



A screenshot of a terminal window titled 'pi@raspberrypi: ~'. The window has a menu bar with 'File', 'Edit', 'Tabs', and 'Help'. The command 'sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf' is typed into the terminal. The prompt shows the user is running as 'pi' on 'raspberrypi' at the root level (~).

I Cài đặt để cho phép truy cập bên ngoài

- ▶ Nhận xét cài đặt liên kết địa chỉ mạng của riêng bạn (địa chỉ liên kết)
- ▶ Sử dụng Ctrl + o → Enter → Ctrl + x để lưu và thoát các khôi phần mềm nano.



```
# * Basic Settings
#
user          = mysql
pid-file      = /run/mysqld/mysqld.pid
socket        = /run/mysqld/mysqld.sock
#port         = 3306
basedir       = /usr
datadir       = /var/lib/mysql
tmpdir        = /tmp
lc-messages-dir = /usr/share/mysql
#skip-external-locking

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
# bind-address = 127.0.0.1

#
# * Fine Tuning
#
#key_buffer_size      = 16M

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos     M-U Undo
^X Exit        ^R Read File   ^V Replace     ^U Uncut Text  ^T To Spell   ^L Go To Line  M-E Redo
```

I Cài đặt để cho phép truy cập bên ngoài

- ▶ Đặt bộ lọc TCP để cho phép INPUT và OUTPUT trên cổng 3306 được sử dụng bởi MariaDB

```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
pi@raspberrypi:~ $ sudo iptables -A INPUT -p tcp --dport 3306 -j ACCEPT
pi@raspberrypi:~ $ sudo iptables -A OUTPUT -p tcp --dport 3306 -j ACCEPT
pi@raspberrypi:~ $ sudo iptables-save
# Generated by xtables-save v1.8.2 on Tue Sep 21 03:40:07 2021
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -p tcp -m tcp --dport 3306 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 3306 -j ACCEPT
COMMIT
# Completed on Tue Sep 21 03:40:07 2021
pi@raspberrypi:~ $
```

- sudo iptables –A INPUT –p tcp --dport 3306 –j CHẤP NHẬN
- sudo iptables –A ĐẦU RA –p tcp –dport 3306 –j CHẤP NHẬN
- sudo iptables –save

```
> sudo iptables -A INPUT -p tcp --dport 3306 -j ACCEPT
> sudo iptables -A OUTPUT -p tcp -dport 3306 -j ACCEPT
> sudo iptables -save
```

- ▶ Sau khi thực hiện các lệnh này, hãy khởi động lại Raspberry Pi để phản ánh các cài đặt và hoàn tất quá trình chuẩn bị cho truy cập bên ngoài.

Cài đặt HeidiSQL

- | Truy cập trang web HeidiSQL và tải xuống HeidiSQL cho Windows

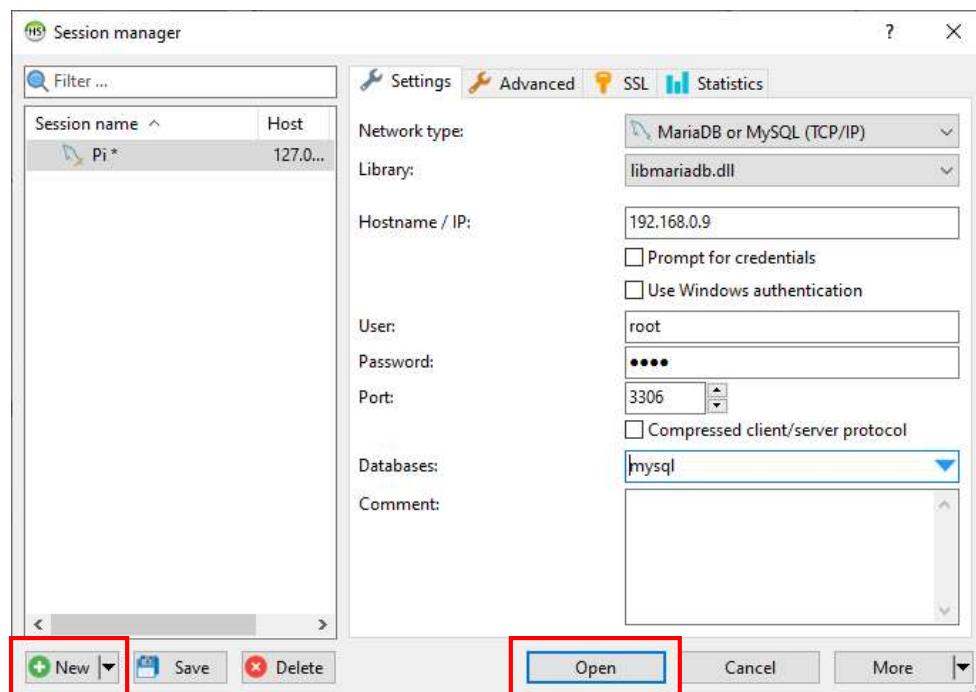


- ▶ <https://www.heidisql.com/download.php>
- ▶ Trình cài đặt, Nhấp vào 32/64 bit để tải xuống Tệp cài đặt
- ▶ Khi chạy tệp Cài đặt, hãy giữ nguyên tất cả các cài đặt cơ bản và chạy HeidiSQL sau khi hoàn thành

HeidiSQL

| Thực thi HeidiSQL

- ▶ Tạo Phiên mới để kết nối với Raspberry Pi trong Trình quản lý phiên
- ▶ Nhấp vào nút Mới ở dưới cùng bên trái để tạo Phiên và nhập thông tin như hình bên phải.



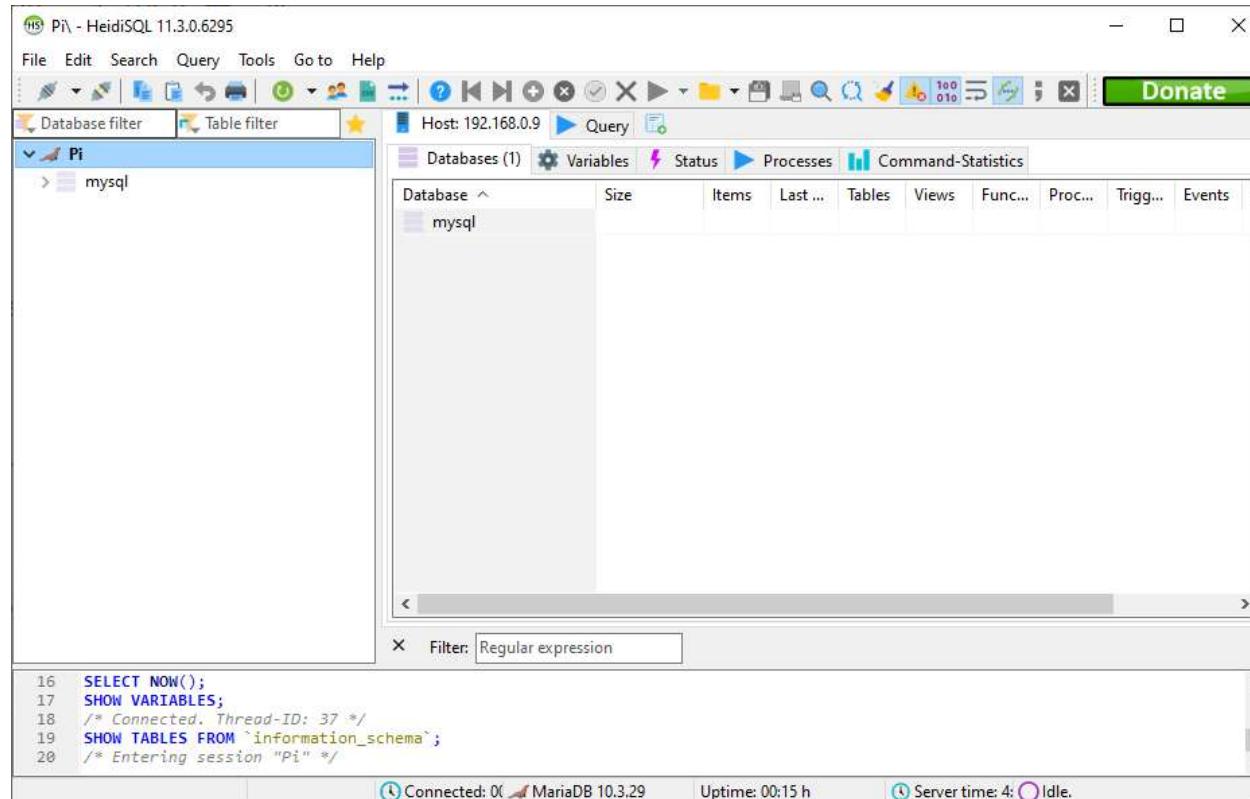
- Tên phiên: Pi
- Hostname/IP: Địa chỉ IP của Raspberry Pi
- Người dùng: gốc
- Mật khẩu: Mật khẩu của người dùng root đã đặt
- Cơ sở dữ liệu: mysql



Sau khi nhập thông tin,
nhấp vào Mở ở dưới cùng.

Kiểm tra phiên Pi

- Nếu kết nối thành công, bạn có thể kiểm tra trạng thái và dữ liệu của mysql DB trên Raspberry Pi như hình bên dưới.
- Bây giờ, hãy tạo một bảng.



| Lược đồ bảng

- ▶ Hãy thực hành với một ví dụ lưu trữ các giá trị đo được của cảm biến nhiệt độ và độ ẩm trong bảng DB.
- ▶ Cấu trúc bảng như dưới đây.

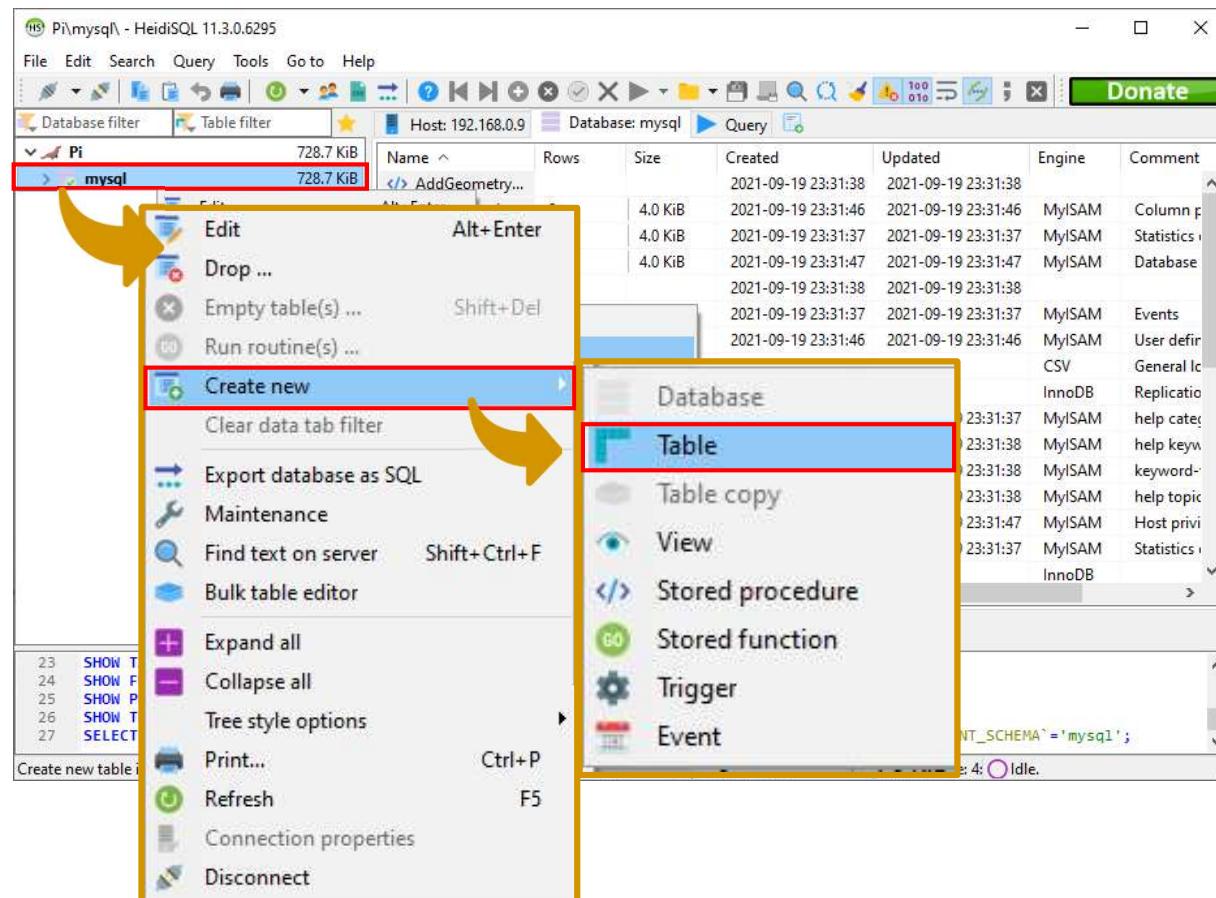
temperature
datatime (DATETIME)
temp (FLOAT)
humid (FLOAT)

5.2. Sử dụng HeidiSQL

Bài 05

Tạo bảng

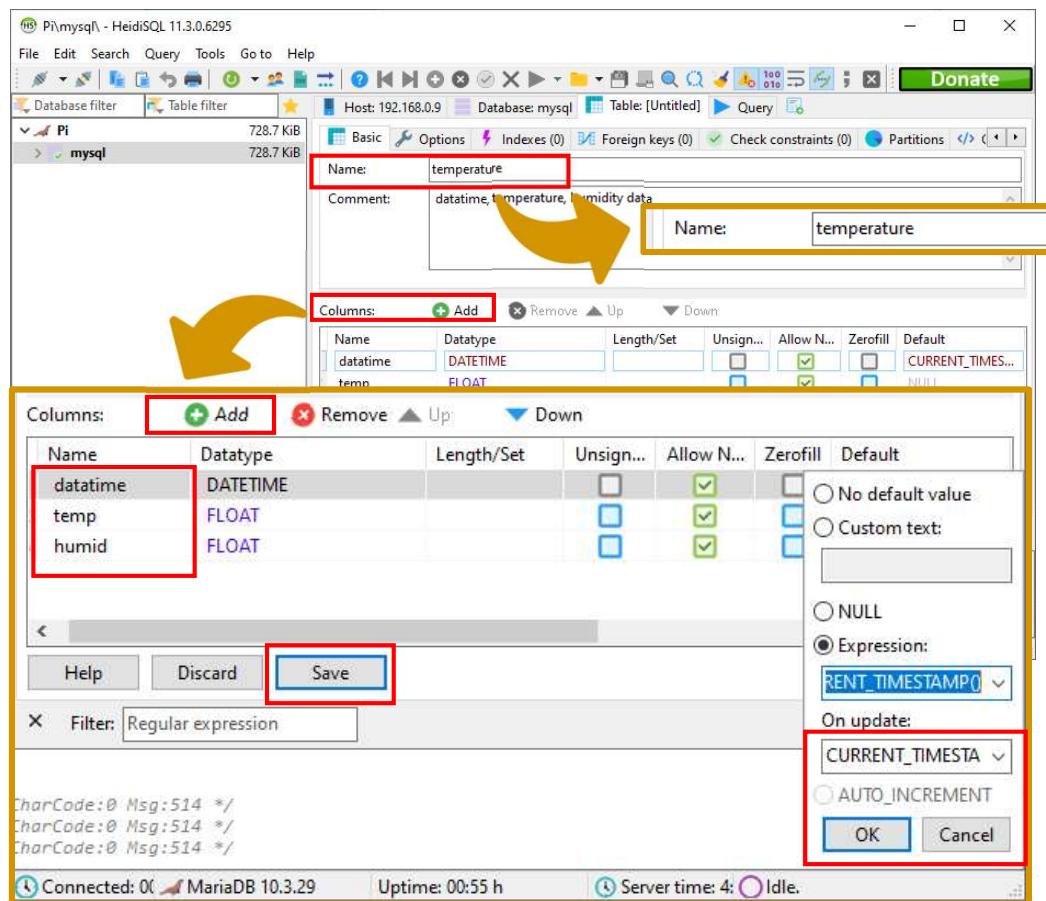
- Nhấp chuột phải vào mysql trên Pi → Nhấp vào Tạo mới → Nhấp vào Bảng để tạo bảng mới.



5.2. Sử dụng HeidiSQL

Bài 05

I thiết lập bảng



- Nhập nhiệt độ làm tên của bảng và nhập mô tả trong Nhận xét.
- Sử dụng nút Thêm ở bên phải Cột: để thêm cột. Thêm ba cột: thời gian dữ liệu, nhiệt độ và độ ẩm.
- Đặt kiểu dữ liệu của cột theo lược đồ bảng và thay đổi giá trị mặc định của thời gian dữ liệu thành CURRENT_TIMESTAMP().
- Sau khi thiết lập mọi thứ, nhấp vào nút Lưu để tạo bảng.

DB Query

I Cú pháp câu lệnh truy vấn (Query)

- ▶ Truy vấn là một cú pháp để thêm, sửa, xóa và đọc dữ liệu từ DB.
- ▶ Có nhiều cú pháp Truy vấn SQL khác nhau, nhưng chúng ta sẽ xem xét ngắn gọn các câu lệnh Truy vấn cơ bản và thường được sử dụng nhất.
 - Câu lệnh INSERT để thêm dữ liệu vào bảng

```
INSERT INTO TABLE_NAME([COL_NAME],[COL_NAME]) VALUES ([DATA], [DATA]);
```

- Câu lệnh SELECT để đọc dữ liệu từ bảng

```
SELECT [COL_NAME],[COL_NAME] FROM TABLE_NAME [WHERE CONDITION];
```

- Câu lệnh CẬP NHẬT để sửa đổi dữ liệu được lưu trữ trong bảng

```
UPDATE TABLE_NAME SET COL_NAME = 'VALUE' [WHERE CONDITION];
```

- Câu lệnh DELETE để xóa dữ liệu được lưu trữ trong một bảng

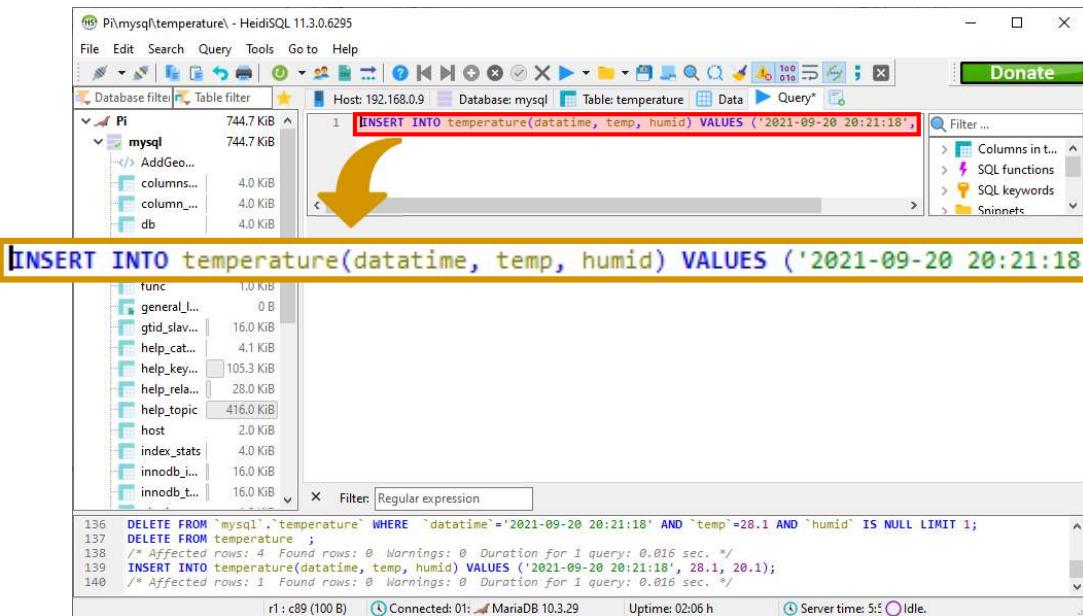
```
DELETE FROM TABLE_NAME [WHERE CONDITION];
```

5.2. Sử dụng HeidiSQL

Bài 05

Áp dụng các câu lệnh Truy vấn đơn giản cho DB mẫu

- Thực hiện các câu lệnh và kiểm tra kết quả



The screenshot shows the HeidiSQL interface. In the central query editor, the following SQL statement is highlighted with a yellow arrow pointing to it:

```
INSERT INTO temperature(datatime, temp, humid) VALUES ('2021-09-20 20:21:18', 28.1, 20.1);
```

The database tree on the left shows the connection to 'P:\mysql\temperature'. The bottom status bar indicates the connection is 'Connected' to 'MariaDB 10.3.29'.

- Thêm '2021-09-20 20:21:18' vào cột ngày giờ trong bảng nhiệt độ, 28.1 vào cột "temp" và 20.1 vào cột độ ẩm

```
INSERT INTO temperature(datatime, temp, humid)
VALUES ('2021-09-20 20:21:18', 28.1, 20.1);
```

Áp dụng các câu lệnh Truy vấn đơn giản cho DB mẫu

```

P:\mysql\temperature\ - HeidiSQL 11.3.0.6295
File Edit Search Query Tools Go to Help
Database filter Table filter Host: 192.168.0.9 Database: mysql Table: temperature Data Query*
Donate

SELECT * FROM temperature;

temperature (1r x 3c)
datetime temp humid
2021-09-20 20:21:18 28.1 20.1

138 /* Affected rows: 4 Found rows: 0 Warnings: 0 Duration for 1 query: 0.016 sec. */
139 INSERT INTO temperature(datetime, temp, humid) VALUES ('2021-09-20 20:21:18', 28.1, 20.1);
140 /* Affected rows: 1 Found rows: 0 Warnings: 0 Duration for 1 query: 0.016 sec. */
141 SELECT * FROM temperature;
142 /* Affected rows: 0 Found rows: 1 Warnings: 0 Duration for 1 query: 0.000 sec. */

r1: c27 (28 B) Connected: 01: MariaDB 10.3.29 Uptime: 02:07 h Server time: 5:5 Idle.

```

- Đọc tất cả các giá trị thuộc tính (*) trong bảng nhiệt độ. Nghĩa là, nó đọc các giá trị của tất cả các thuộc tính thời gian, nhiệt độ và độ ẩm.

```
SELECT * FROM temperature;
```

| Áp dụng các câu lệnh Truy vấn đơn giản cho DB mẫu

The screenshot shows the HeidiSQL interface with the following details:

- Database:** Pi\mysql\temperature - HeidiSQL 11.3.0.6295
- Host:** 192.168.0.9
- Database:** mysql
- Table:** temperature
- Query:**

```
1 UPDATE temperature SET temp=32.0 WHERE datatime = '2021-09-20 20:21:18';
2 SELECT * FROM temperature;
```
- Result pane:** Shows the output of the second query: `UPDATE temperature SET temp=32.0 WHERE datatime = '2021-09-20 20:21:18';
SELECT * FROM temperature;`
- Logs pane:** Shows the following log entries:


```
162 SELECT * FROM temperature;
163 /* Affected rows: 0 Found rows: 1 Warnings: 0 Duration for 2 queries: 0.000 sec. */
164 UPDATE temperature SET temp=32.0 WHERE datatime = '2021-09-20 20:21:18';
165 SELECT * FROM temperature;
166 /* Affected rows: 1 Found rows: 1 Warnings: 0 Duration for 2 queries: 0.016 sec. */
```
- Status bar:** Shows the connection status: r1:c72 (114 B) Connected: 01: MariaDB 10.3.29 Uptime: 02:11 h Server time: 6:0 Idle.

- Thay đổi giá trị thuộc tính tạm thời của dữ liệu có thời gian dữ liệu là '2021-09-20 20:21:18' thành 32,0 trong bảng nhiệt độ.

CẬP NHẬT nhiệt độ SET temp=32.0 WHERE datatime = '2021-09-20 20:21:18';

- Đọc tất cả dữ liệu trong bảng để xem các thay đổi

CHỌN * TỪ nhiệt độ;

| Áp dụng các câu lệnh Truy vấn đơn giản cho DB mẫu

The screenshot shows the HeidiSQL interface connected to a MySQL database named 'temperature'. In the main query editor, two SQL statements are highlighted with a red box:

```

1 DELETE FROM temperature WHERE temp = 32;
2 SELECT * FROM temperature;

```

A yellow arrow points from the text area below the queries to the first statement. The text area contains the same two queries, with the first one in blue and the second in red. Below the text area, the server log shows the execution of these queries:

```

165 SELECT * FROM temperature;
166 /* Affected rows: 1  Found rows: 1  Warnings: 0  Duration for 2 queries: 0.016 sec. */
167 DELETE FROM temperature WHERE temp = 32;
168 SELECT * FROM temperature;
169 /* Affected rows: 0  Found rows: 0  Warnings: 0  Duration for 2 queries: 0.031 sec. */

```

The status bar at the bottom indicates the connection is established to a MariaDB 10.3.29 server.

- Xóa dữ liệu có giá trị thuộc tính tạm thời là 32,0 khỏi bảng nhiệt độ

```
DELETE FROM temperature WHERE temp = 32;
```

- Đọc tất cả dữ liệu trong bảng để xem các thay đổi

```
SELECT * FROM temperature;
```

Bài 5.

Sử dụng Raspberry Pi làm Máy chủ DB

- | 5.1. Sử dụng MariaDB
- | 5.2. Sử dụng HeidiSQL
- | **5.3 Ứng dụng của MariaDB**

Sử dụng cảm biến nhiệt độ và độ ẩm

| Thiết lập cơ sở dữ liệu lưu thông tin nhiệt độ - độ ẩm

- ▶ sudo python3 -m pip install pymysql
 - Cài đặt thư viện pymysql cho phép python3 hoạt động với MariaDB

```
pi@raspberrypi:~$ sudo python3 -m pip install pymysql
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting pymysql
  Downloading https://files.pythonhosted.org/packages/4f/52/a115fe175028b058df353c5a3d5290b71514a83f67024d6137/PyMySQL-1.0.2-py3-none-any.whl (43kB)
    100% |██████████| 51kB 1.8MB/s
Installing collected packages: pymysql
Successfully installed pymysql-1.0.2
```

I Sử dụng thư viện lập trình của Adafruit để giao tiếp với cảm biến nhiệt độ và độ ẩm DHT11

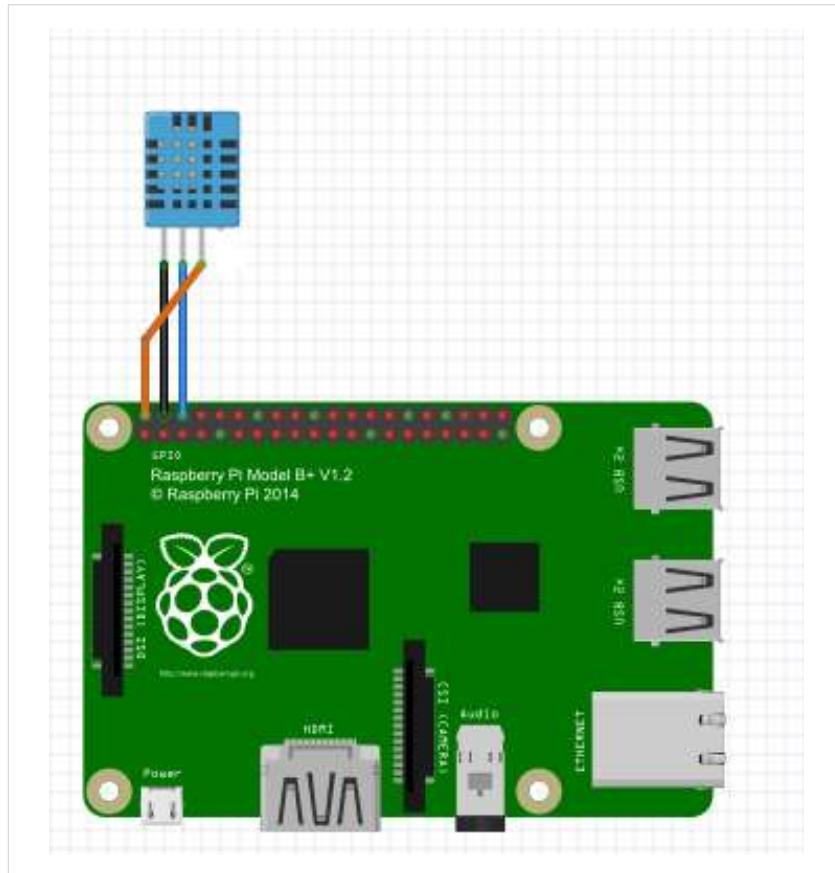
- ▶ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
 - Sao chép git để tải về thư viện lập trình với cảm biến nhiệt độ và độ ẩm DHT

```
pi@raspberrypi:~ $ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
Cloning into 'Adafruit_Python_DHT'...
remote: Enumerating objects: 325, done.
remote: Total 325 (delta 0), reused 0 (delta 0), pack-reused 325
Receiving objects: 100% (325/325), 98.35 KiB | 2.23 MiB/s, done.
Resolving deltas: 100% (176/176), done.
```

- ▶ cd Adafruit_Python_DHT
- ▶ Cài đặt sudo python setup.py
 - Chạy setup.py để thực hiện cài đặt thư viện

```
pi@raspberrypi:~ $ cd Adafruit_Python_DHT/
pi@raspberrypi:~/Adafruit_Python_DHT $ sudo python setup.py install
running install
running bdist_egg
running egg_info
creating Adafruit_DHT.egg-info
writing Adafruit_DHT.egg-info/PKG-INFO
writing top-level names to Adafruit_DHT.egg-info/top_level.txt
writing dependency_links to Adafruit_DHT.egg-info/dependency_links.txt
writing manifest file Adafruit_DHT.egg-info/SOURCES.txt
```

I Cấu hình sơ đồ mạch thực hành đo nhiệt độ, độ ẩm



- ▶ Chân DOUT của Cảm biến nhiệt độ và độ ẩm nối với GPIO2 của Raspberry Pi
- ▶ Chân GND Cảm biến nhiệt độ và độ ẩm nối với GND của Raspberry Pi
- ▶ Chân VCC của Cảm biến nhiệt độ và độ ẩm nối với chân nguồn 3.3V của Raspberry Pi

I Chạy file mã lệnh AdafruitDHT.py

- ▶ Thực hiện file mã lệnh python AdafruitDHT.py 11 2
 - Giá trị 11 chỉ định loại cảm biến (DHT11) và 2 chỉ định sử dụng chân GPIO2.
- ▶ Nhiệt độ và độ ẩm được đo chính xác trong khoảng thời gian 5 giây trong khi thực hiện.
- ▶ Nếu muốn thoát nhấn Ctrl + c để tạo ngắt.

```
pi@raspberrypi:~/Adafruit_Python_DHT/examples $ python AdafruitDHT.py 11 2
Temp=27.0*  Humidity=42.0%
Temp=27.0*  Humidity=56.0%
^CTraceback (most recent call last):
  File "AdafruitDHT.py", line 55, in <module>
    time.sleep(5)
KeyboardInterrupt
pi@raspberrypi:~/Adafruit_Python_DHT/examples $
```

Lưu dữ liệu đo được bởi cảm biến nhiệt độ - độ ẩm vào trong DB

| Thay đổi file mã lệnh AdafruitDHT.py

- Thực hiện chức năng lưu dữ liệu, giả định đã xác nhận được dữ liệu nhiệt độ và độ ẩm, vào bảng nhiệt độ bằng cách sử dụng câu lệnh INSERT.

```
1 import sys
2 import time
3 import Adafruit_DHT
4 import pymysql
5
6 sensor = Adafruit_DHT.DHT11
7 pin = 2
8
9 db, cur = None, None
10
11 db = pymysql.connect(host='192.168.0.4', user='root', password='1234', db='mysql', charset='utf8')
12
13 try:
14     cur = db.cursor()
15     while True:
16         humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
17         data = (humidity, temperature)
18         if humidity is not None and temperature is not None:
19             sql = "INSERT INTO temperature(temp, humid) VALUES (%4.1f, %4.1f)" % data
20             print(sql)
21             cur.execute(sql)
22             db.commit()
23         else:
24             print('Failed to get reading. Try again!')
25             time.sleep(5)
26
27 except KeyboardInterrupt:
28     pass
29 finally:
30     db.close()
```

- dòng 6: Khai báo biến đối tượng cảm biến DHT11
- dòng 9: Khai báo biến toàn cục
- dòng 11: kết nối với cơ sở dữ liệu của máy chủ với: địa chỉ ip của raspberry pi, người dùng là root, mật khẩu: đặt mật khẩu, loại db là mysql
- dòng 14: thiết lập con trỏ truy xuất vào DB
- dòng 16-17: Đọc dữ liệu nhiệt độ và độ ẩm từ cảm biến
- dòng 19-22: Thực thi câu lệnh truy vấn INSERT để lưu dữ liệu nhiệt độ và độ ẩm vào bảng thông tin nhiệt độ trong DB

| Thực thi mã lệnh AdafruitDHT.py

pi@raspberrypi: ~/Adafruit_Python_DHT/examples

```
pi@raspberrypi:~/Adafruit_Python_DHT/examples $ python3 AdafruitDHT.py
INSERT INTO temperature(temp, humid) VALUES (61.0, 26.0)
INSERT INTO temperature(temp, humid) VALUES (60.0, 26.0)
```

P:\mysql\temperature - HeidiSQL 11.3.0.6295

File Edit Search Query Tools Goto Help

Database filter Table filter Host: 192.168.0.5 Database: mysql Table: temperature Data sql.sql sql.sql* □

SELECT * FROM temperature;

	temp	humid
2021-09-22 06:23:48	60	26
2021-09-22 06:23:54	60	26
2021-09-22 06:23:59	60	26
2021-09-22 06:29:38	60	26
2021-09-22 06:29:43	59	27
2021-09-22 06:29:49	61	26
2021-09-22 06:32:14	61	26
2021-09-22 06:32:20	60	26
2021-09-22 06:32:25	59	27

```
43 SHOW COLLATION;
44 SHOW CREATE TABLE `mysql`.`temperature`;
45 SHOW TABLES FROM `information_schema`;
46 SELECT * FROM temperature;
47 /* Affected rows: 0  Found rows: 10  Warnings: 0  Duration for 1 query: 0.016 sec. */
```

r1 : c1 (28 B) Connected: 00: MariaDB 10.3.29 Uptime: 02:05 h Server time: 6:5 Idle.

- ▶ Sử dụng lệnh python3 để thực thi file mã lệnh python, ví dụ: AdafruitDHT.py. Bạn có thể kiểm tra các giá trị đo được như minh họa ở bên trái và bạn có thể kiểm tra câu lệnh SQL đã thực thi.
- ▶ Quay trở lại HeidiSQL và vào bảng nhiệt độ để kiểm tra xem dữ liệu đã được lưu trong bảng hay chưa.
- ▶ Như trước đây, sử dụng câu lệnh truy vấn **SELECT * FROM temperature;** như hình bên, để kiểm tra dữ liệu mới đã được lưu thành công vào trong bảng cơ sở dữ liệu.

Cung cấp dữ liệu được lưu trữ trong DB cho trang web

| Hỗ trợ chức năng trực quan hóa dữ liệu trong một trang Web dựa trên dữ liệu được lưu trữ trong DB

- ▶ Xây dựng mã lệnh python để chuyển dữ liệu lưu trong MariaDB sang một file html

```

1  from flask import Flask, request, render_template
2  import pymysql
3
4  db = None
5  cur = None
6  app = Flask(__name__)
7
8  def select(query):
9      db = pymysql.connect(host='192.168.0.3', user='root', password='1234', db='mysql', charset='utf8')
10     cur = db.cursor()
11     cur.execute(query)
12     result = cur.fetchall()
13     db.close()
14     return result
15
16 @app.route('/temp_humid_chart')
17 def lm35_chart():
18     sql = "SELECT DATETIME, TEMP FROM temperature ORDER BY DATETIME ASC LIMIT 100"
19     result = select(sql)
20     return render_template("temp_humid_chart.html", result=result)
21
22 if __name__ == '__main__':
23     app.run(debug=True, port=80, host='0.0.0.0')

```

- ▶ Save as ~/rasp_ex/webapp/temp_humid_chart.py

- dòng 8-14: Hàm select(), hàm thực hiện kết nối với MariaDB, thiết lập con trỏ tới cơ sở dữ liệu và thực hiện lệnh truy vấn. Hàm trả về kết quả của lệnh này.
- dòng 16-20: Hàm xử lý, lấy 100 mẫu dữ liệu nhiệt độ và sắp xếp theo thứ tự thời gian và chuyển tiếp dữ liệu này vào file temp_humid_chart.html

| ~/rasp_ex/webapp/temp_humid_chart.html (1)

- ▶ tệp html cung cấp trực quan hóa dữ liệu bằng cách sử dụng kết quả được chuyển bởi mã lệnh temp_humid_chart.py

```
1  <html>
2  <head>
3      <title>RaspberryPi Visualization</title>
4      <style type="text/css">
5          .auto-style1 {
6              border: 1px solid #808080;
7          }
8          .auto-style2 {
9              border: 1px solid #008080;
10         }
11     </style>
12 </head>
13     <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.4.0/Chart.min.js"></script>
14 <body>
15     <center>
16         <br>
17         <strong>Temperature, Humidity Data Visualization</strong>
18         <br>
19         <br>
20         <br>
```

- dòng 3: Tiêu đề của trang web
- dòng 4-11: Cài đặt kiểu cho trang web
- dòng 13: nhúng mã javascript chart.js từ nguồn CDN
- dòng 17: định dạng bôi đậm cho chuỗi văn bản trên trang web

| ~/rasp_ex/webapp/temp_humid_chart.html (2)

- ▶ tệp html cung cấp trực quan hóa dữ liệu bằng cách sử dụng kết quả được chuyển bởi mã lệnh temp_humid_chart.py

```
21
22     <table style="width: 100%">
23         <tr>
24             <td width="50%">
25                 <table cellspacing="1" class="auto-style1" style="width: 100%">
26                     {% for i in result%}
27                         <tr>
28                             <td class="auto-style2" style="width: 50%">Time: {{ i[0] }}</td>
29                             <td class="auto-style2" style="width: 50%">Temperature: {{ i[1] }}</td>
30                     </tr>
31                 {% endfor %}
32             </table>
33             </td>
34
35             <td width="50%">
36                 <div>
37                     <canvas id="Chart"></canvas>
38                 </div>
39
40             </td>
41         </tr>
42     </table>
43     <br>
44
45     </center>
46 </body>
47
48 </html>
```

- dòng 26-31 : Xuất cặp giá trị (thời gian và nhiệt độ) tuần tự vào trong bảng thông tin trên trang web

| ~/rasp_ex/webapp/temp_humid_chart.html (3)

- ▶ Mã lệnh javascript thực hiện vẽ biểu đồ - chart.js

```
50 <script>
51     var ctx = document.getElementById('Chart').getContext('2d');
52     var data = {
53         // The type of chart we want to create
54         type: 'line',
55         // The data for our dataset
56         data: {
57             labels: [ // Date
58                 {% for i in result%}
59                 '{{ i[0] }}',
60                 {% endfor %}
61             ],
62             datasets: [{{
63                 label: "Temperature",
64                 backgroundColor: 'rgb(255, 120, 132)',
65                 fill:false,
66                 borderColor: 'rgb(255, 128, 132)',
67                 lineTension : 0.8,
68                 data: [ // Temperature
69                     {% for i in result%}
70                     {{ i[1] }},
71                     {% endfor %}
72                 ],
73             }]
74         },
75     },
```

- dòng 54: kiểu đồ thi là biểu đồ đường (line-chart)
- dòng 56-59: Định nghĩa nhãn dữ liệu. Nó có nghĩa là ngày là đối số đầu tiên của kết quả nhận được.
- dòng 62-71: Xuất các giá trị Nhiệt độ ở định dạng đã chỉ định. Nếu giá trị của tham số lineTension tăng lên, đường đồ thị biểu diễn sẽ cong hơn.

| ~/rasp_ex/webapp/temp_humid_chart.html (4)

- ▶ Tập lệnh vẽ biểu đồ bằng chart.js

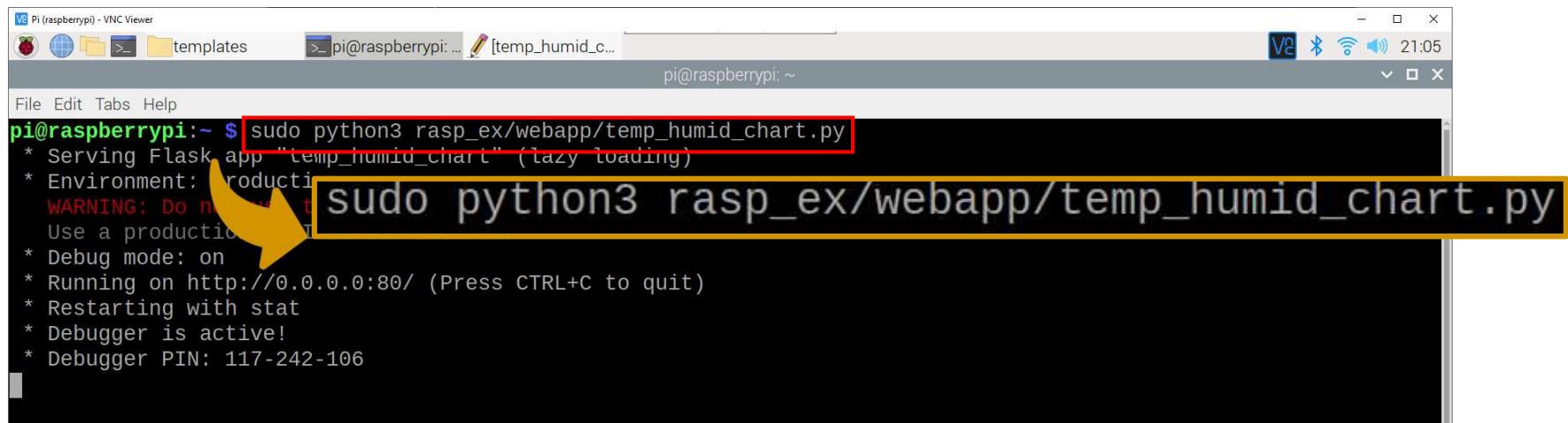
```
75     // Configuration options go here
76     options: {
77         scales: {
78             xAxes: [
79                 {
80                     ticks: {
81                         suggestedMin: 0,
82                         suggestedMax: 100
83                     }
84                 },
85                 yAxes: [
86                     {
87                         ticks: {
88                             suggestedMin: 10, // Y axis min
89                             suggestedMax: 50 // Y axis max
90                         }
91                     }
92                 }
93             }
94         }
95     }
96     var chart = new Chart(ctx, data);
97 </script>
```

- dòng 79-81: Đặt phạm vi biểu diễn cho trục x của đồ thị
- dòng 84-89: Đặt phạm vi biểu diễn cho trục y của đồ thị
- dòng 93: Tạo một đối tượng chart để biểu diễn dữ liệu

- ▶ Save the html file as ~/rasp_ex/webapp/temp_humid_chart.html

| Thực thi tập lệnh python đã tạo (1)

- ▶ Vì đang sử dụng Flask nên ta cần thực hiện lệnh sudo.
 - ▶ Thực thi lệnh `sudo python3 rasp_ex/webapp/temp_humid_chart.py` để thực thi tập lệnh.



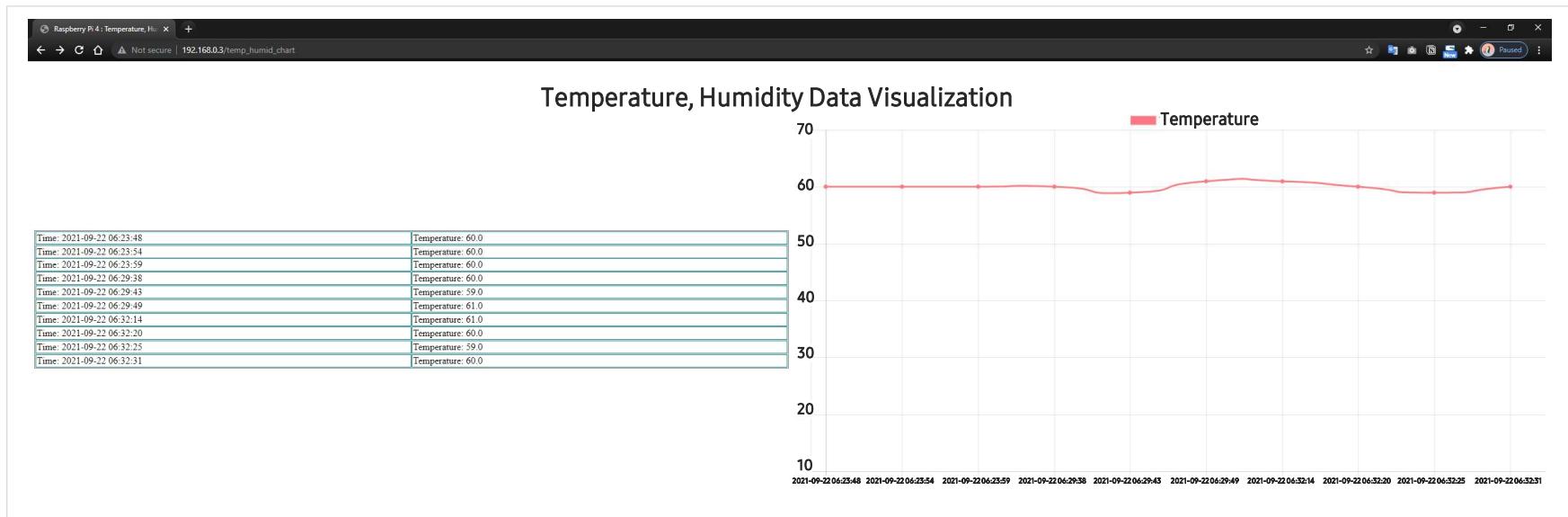
A screenshot of a terminal window titled "Pi (raspberrypi) - VNC Viewer". The window shows a command-line interface with the following text:

```
pi@raspberrypi:~ $ sudo python3 rasp_ex/webapp/temp_humid_chart.py
* Serving Flask app "temp_humid_chart" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production setting
  Use a production WSGI server instead
* Debug mode: on
* Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 117-242-106
```

The command `sudo python3 rasp_ex/webapp/temp_humid_chart.py` is highlighted with a red rectangle. A yellow arrow points from the bottom left towards this highlighted command.

I Executing the created python script (2)

- ▶ Nhập địa chỉ IP của Raspberry Pi trên trình duyệt để truy nhập đến trang temp_humid_chart.html
 - Trong trang web nhận về, bạn có thể kiểm tra dữ liệu thời gian và nhiệt độ trong bảng thông tin, đồng thời kiểm tra dữ liệu được biểu diễn trực quan thông qua đồ thị.



Kết thúc Tài liệu



Together for Tomorrow!
Enabling People

Education for Future Generations

©2022 SAMSUNG. All rights reserved.

Samsung Electronics Corporate Citizenship Office holds the copyright of book.

This book is a literary property protected by copyright law so reprint and reproduction without permission are prohibited.

To use this book other than the curriculum of Samsung Innovation Campus or to use the entire or part of this book, you must receive written consent from copyright holder.