



# **CHUYÊN ĐỀ INTERNET OF THINGS (2+0)**

GV: TS. Hồ Đức Chung  
Email: [hoduccung@tdmu.edu.vn](mailto:hoduccung@tdmu.edu.vn)

## **Chương 2. Các công nghệ IoT**

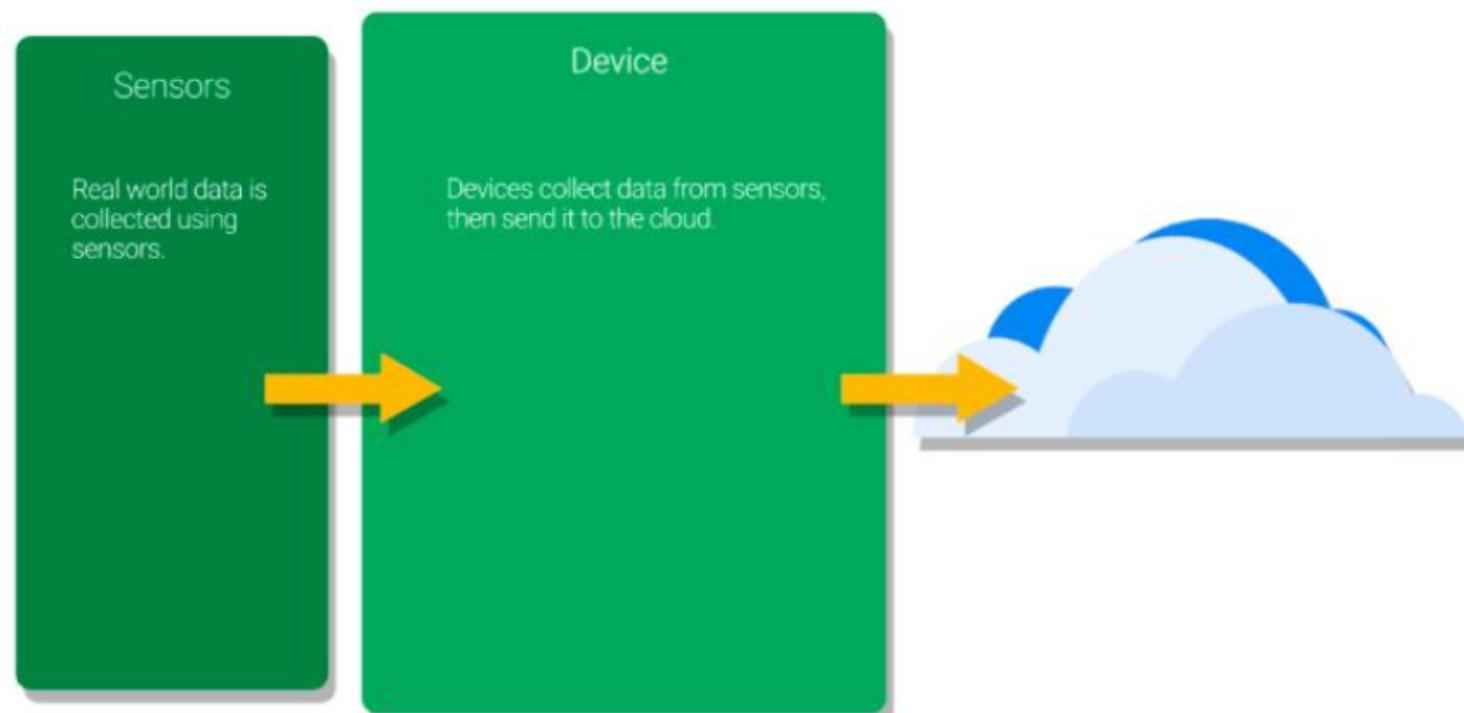
- 2.1. Cảm biến và thiết bị
- 2.2. Một số chuẩn truyền thông cho IoT
- 2.3. Các giao thức cho ứng dụng IoT
- 2.4. Nền tảng đám mây IoT

## 2.1. Cảm biến và thiết bị IoT

- Cảm biến và thiết bị:
  - Các cảm biến và thiết bị IoT được gọi đơn giản là thiết bị (devices) với ngầm định bao gồm cả các cảm biến được ghép nối
  - Trong nhiều tình huống, khi đề cập “thiết bị” có thể được hiểu là bao gồm cả thiết bị và cảm biến

# Cảm biến và thiết bị IoT

- Cảm biến quan sát các thay đổi xung quanh và gửi thông tin về các thay đổi đó đến thiết bị
- Các thiết bị thu thập dữ liệu từ các cảm biến và gửi tới cloud/server



# Cảm biến và thiết bị IoT

- Các thiết bị có thể hạn chế về tài nguyên tính toán (CPU, bộ nhớ hạn chế, ...)
- Các thiết bị có thể giao tiếp truyền thông riêng mà không kết nối trực tiếp với cloud platform, ví dụ thông qua Bluetooth Low Energy (BLE)
- Các thiết bị có thể lưu trữ, xử lý và phân tích dữ liệu trước khi gửi lên cloud

# Phân loại cảm biến (Types of Sensors)

- Theo yêu cầu về nguồn cấp

Type	Definition	Example
passive	Does not require external power to operate. They respond to input from their environment.	A temperature sensor that changes resistance in response to temperature changes
active	Requires external power to operate.	A camera

- Theo loại tín hiệu

Type	Definition	Example
analog	Outputs an analog continuous signal	Accelerometers, temperature sensors
digital	The output is converted to discrete values (digital 1s and 0s) before transmitting to a device	Digital pressure sensor, digital temperature sensor

- Theo loại thiết bị đo

Type	Definition	Example
chemical	Responds to chemical changes in its environment	Gas sensor
mechanical	Responds to physical changes in its environment	Microswitch
electrical	Responds to electrical changes in its environment	Optical sensor

# Lựa chọn cảm biến

- Lựa chọn cảm biến cần cân nhắc đến tầm quan trọng của các yếu tố và mức độ ưu tiên trong thiết kế tổng thể.
- **Durability (Tính lâu bền):**
  - Tính lâu bền phải được cân nhắc đến dựa trên môi trường hoạt động của sensor.
  - Đảm bảo thiết bị hoạt động lâu bền trong khoảng thời gian đáng kể không phải tốn chi phí sửa chữa thay thế.
  - Ví dụ: cảm biến nhiệt kháng nước dùng cho trạm thời tiết

# Lựa chọn cảm biến

- **Accuracy (Độ chính xác):**
  - Cần độ chính xác đáp ứng đủ nhu cầu (độ chính xác cao đi kèm giá thành đắt)
  - Ví dụ: Đo nhiệt độ nhà kho có thể chấp nhận độ chính xác +/-2 độ. Đo nhiệt độ hệ thống thiết bị y tế cần độ chính xác +/-0.2 độ

# Lựa chọn cảm biến

- **Versatility (Tính linh hoạt):**
  - Cảm biến có thể vận hành với các biến động của môi trường xung quanh
  - Ví dụ: cảm biến cho trạm thời tiết có thể hoạt động trong điều kiện mùa hè, mùa đông. Không khả thi với loại cảm biến chỉ hoạt động trong môi trường trong nhà.

# Lựa chọn cảm biến

- **Power Consumption (Tiêu thụ điện năng):**
  - Tùy theo tình huống, có thể đòi hỏi các thiết bị tiêu thụ ít điện năng, sử dụng các đặc tính tiết kiệm năng lượng
  - Ví dụ: cảm biến hoặc thiết bị sử dụng pin năng lượng mặt trời cần thiết kế chế độ ngủ (sleep mode) tiết kiệm năng lượng, thức dậy (wake-up) khi cần thu thập và truyền dữ liệu

# Lựa chọn cảm biến

- Cân nhắc môi trường đặc biệt:
  - Ví dụ: Cảm biến trong giám sát chất lượng nước trong nuôi thủy sản đòi hỏi sensor đặt trong môi trường nước (pH, DO, TDS, ...) khác với các cảm biến dựa trên lấy mẫu phân tích.

# Lựa chọn cảm biến

- **Cost (Chi phí):**
  - Các mạng IoT thường bao gồm hàng trăm, ngàn thiết bị và cảm biến.
  - Thiết kế cần cân nhắc đến chi phí lắp đặt, bảo trì, thay thế, etc.

# Devices – Thiết bị IoT

- Một “Thing” trong “Internet of Things” là một đơn vị xử lý mà có khả năng kết nối vào internet để trao đổi dữ liệu với cloud.
- Các thiết bị thường được gọi là "smart devices" or "connected devices."
- Thiết bị giao tiếp với 2 loại dữ liệu: telemetry (dữ liệu thu thập) và state (trạng thái của thiết bị).

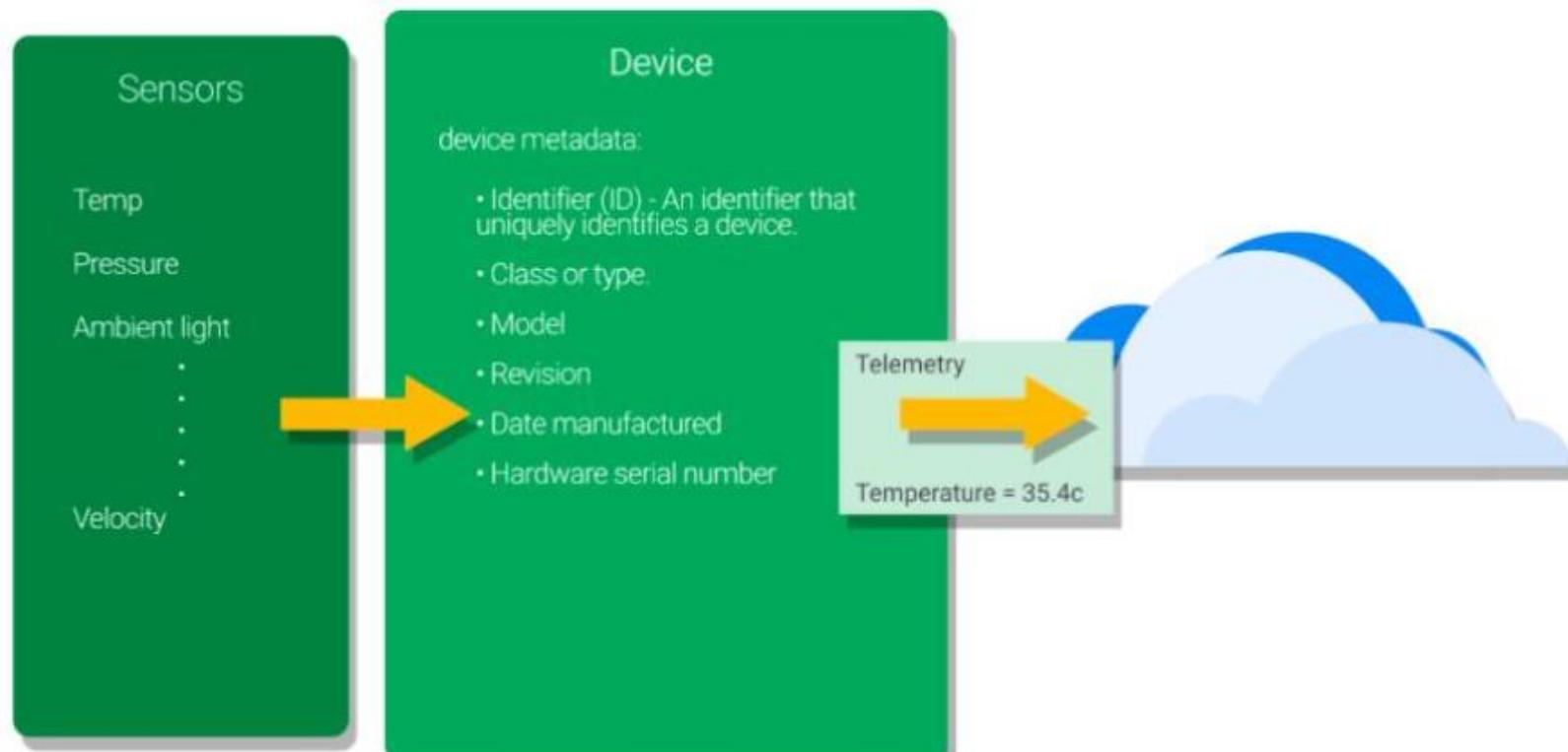
# Các loại thông tin

- Mỗi thiết bị có thể cung cấp hoặc sử dụng một vài loại thông tin phục vụ cho mỗi loại hệ thống backend khác nhau.
- **Device metadata:** Metadata chứa các thông tin mô tả về thiết bị (thường ít thay đổi). Ví dụ:
  - Identifier (ID) – Định danh thiết bị
  - Class or type: Lớp/loại thiết bị
  - Model
  - Revision
  - Date manufactured
  - Hardware serial number

# Các loại thông tin

## ■ Telemetry (đo từ xa):

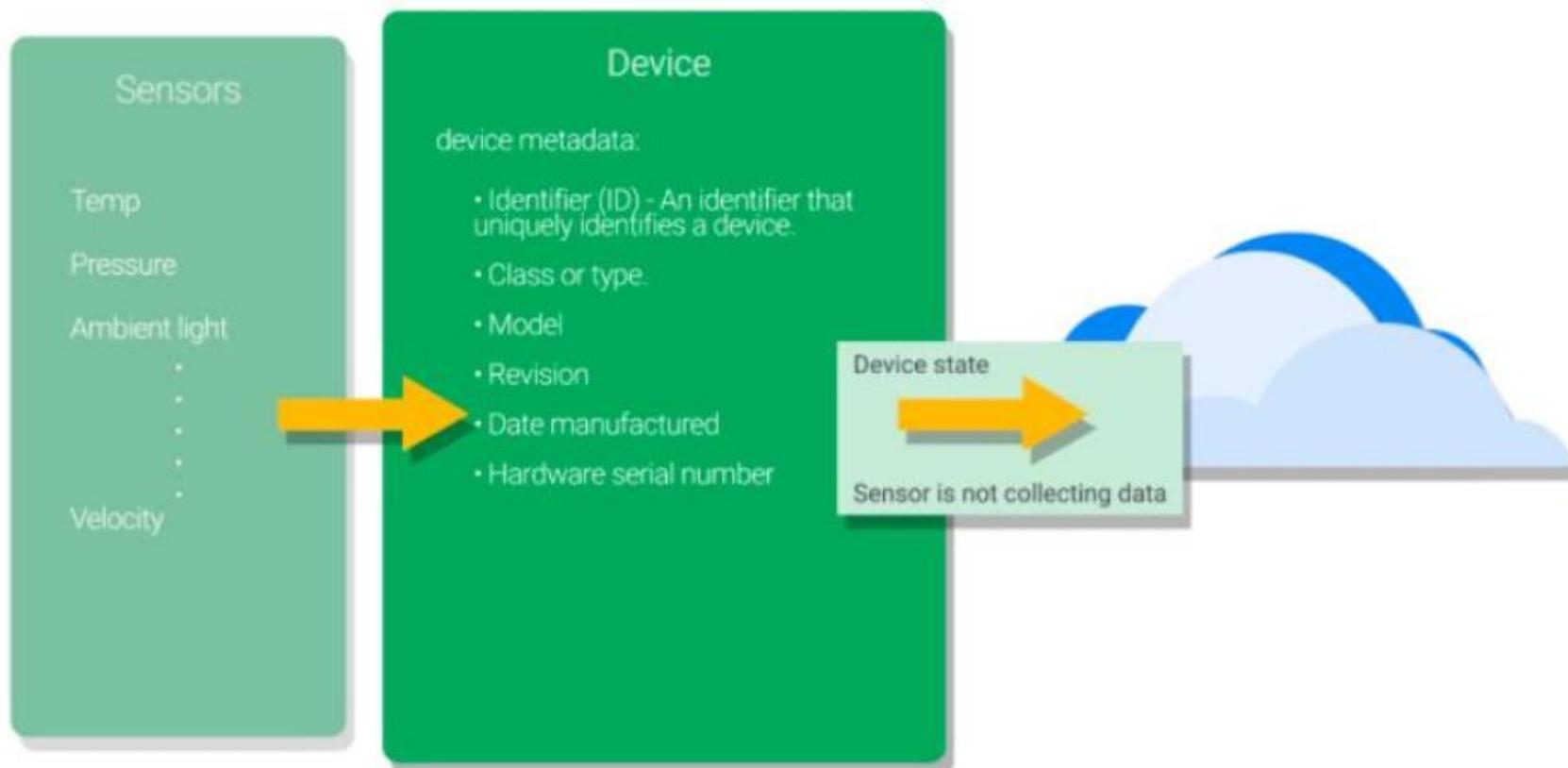
- Dữ liệu được thu thập bởi thiết bị qua các cảm biến được gọi là telemetry.
- Là dữ liệu chỉ đọc (từ môi trường xung quanh)



# Các loại thông tin

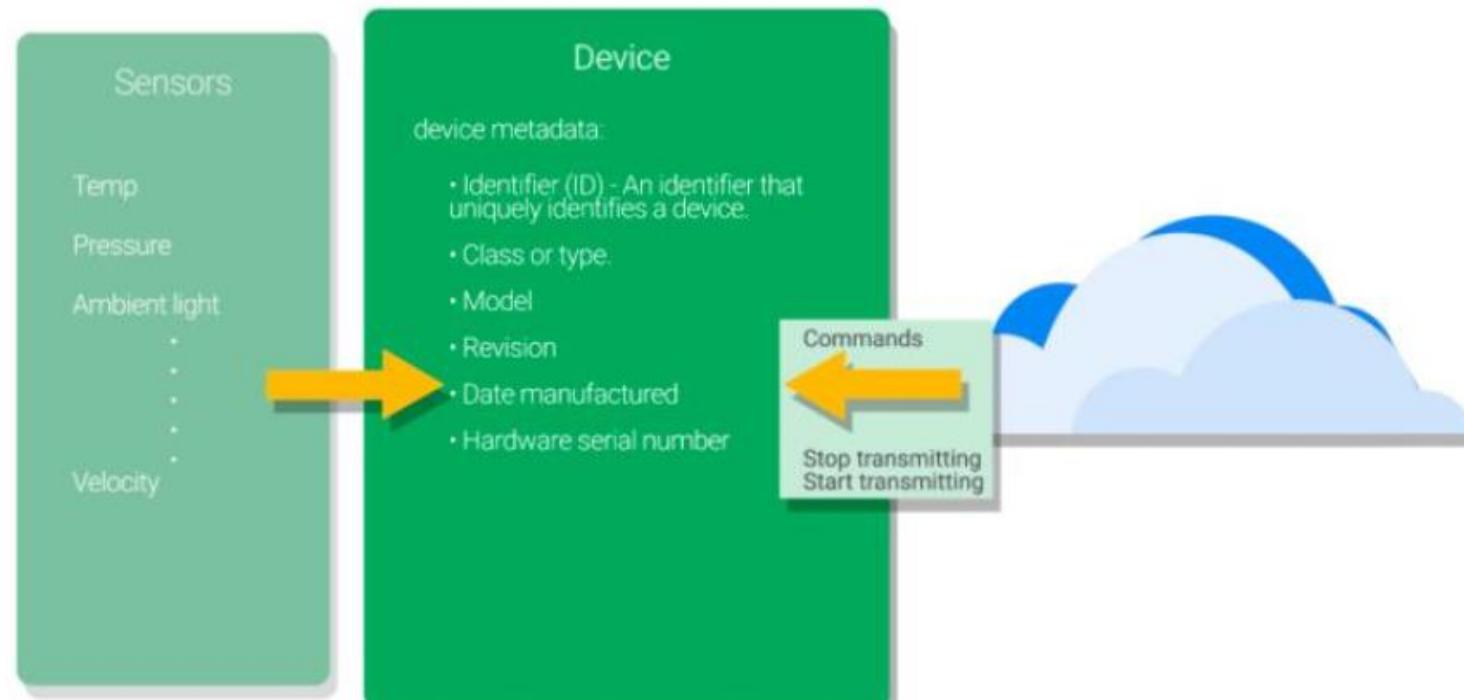
- State information:

- Thông tin trạng thái: Mô tả trạng thái hiện thời của thiết bị.  
Có thể đọc/ghi, cập nhật (nhưng ít thường xuyên)



# Lệnh thực thi trên thiết bị (Device Commands)

- Commands: Là các hành động được thực hiện bởi thiết bị.
- Ví dụ:
  - Quay 360 độ sang phải.
  - Tắt/bật đèn
  - Tăng tần suất truyền dữ liệu



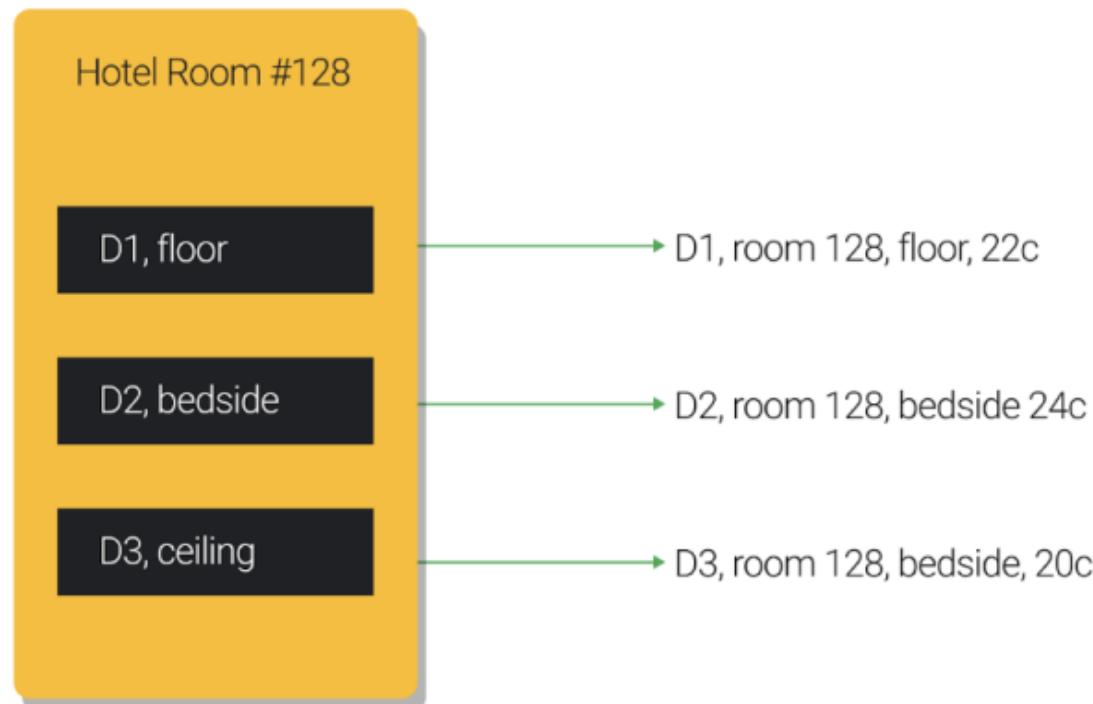
# Định nghĩa thiết bị

- Trong hệ thống IoT, định nghĩa thiết bị có thể thay đổi phụ thuộc vào nhu cầu của dự án.
- Mỗi thiết bị có thể được cân nhắc như một thực thể tách biệt, hoặc một thiết bị có thể chịu trách nhiệm với một nhóm cảm biến.

# Định nghĩa Thiết bị – Ví dụ

- Giám sát nhiệt độ phòng khách sạn
- **Giải pháp 1:** Mỗi phòng có 3 cảm biến: one near the floor, one near the bed, and one near the ceiling.

One hotel room, 3 sensors, 3 devices



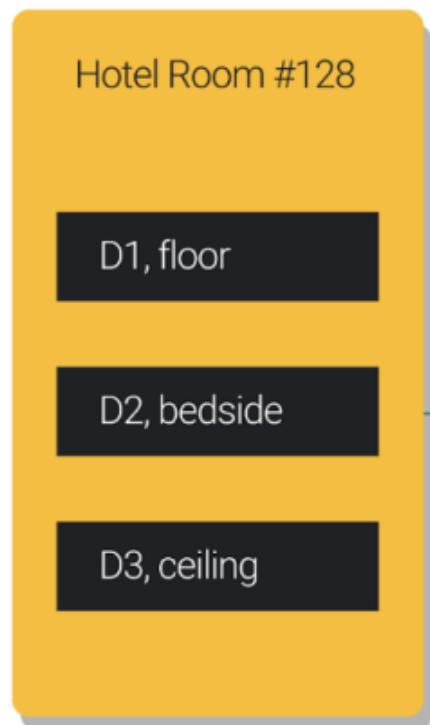
# Định nghĩa Thiết bị – Ví dụ

- **Giải pháp 1:** dữ liệu được gửi đến cloud như là 3 thiết bị khác nhau, mỗi cái gửi một thông tin nhiệt độ đến cloud.
  - {deviceID: "dh28dslkja", "location": "floor", "room": 128, "temp": 22 }
  - {deviceID: "8d3kiuhs8a", "location": "ceiling", "room": 128, "temp": 24 }
  - {deviceID: "kd8s8hh3o", "location": "bedside", "room": 128, "temp": 23 }

# Định nghĩa Thiết bị – Ví dụ

- **Giải pháp 2:** mỗi phòng sử dụng 1 thiết bị chịu trách nhiệm gửi dữ liệu từ 3 cảm biến.

One hotel room, 3 sensors, 1 device



D2, room 128, floor temp 22c,  
bedside temp 24c, ceiling temp 20c

# Định nghĩa Thiết bị – Ví dụ

- **Giải pháp 2:** dữ liệu được gửi đến cloud bằng 1 thiết bị với 3 cảm biến. Có thể bổ sung thêm xử lý tính nhiệt độ trung bình trên thiết bị.
  - {deviceID: "dh28dskja", "room": 128, "temp\_floor": 22, "temp\_ceiling": 24, "temp\_bedside": 23, "average\_temp": 23 }
- Việc lựa chọn giải pháp nào phụ thuộc vào ý định sử dụng thông tin ở thời điểm hiện tại và cả tiềm năng trong tương lai.

## 2.2. Một số chuẩn truyền thông

- Communication standards:
  - NFC and RFID
  - Bluetooth
  - WiFi
  - GSM, GPRS, 2G/3G/4G, LTE
  - ZigBee
  - 6LoWPAN
  - Thread

# Một số chuẩn truyền thông trong IoT

- RFID (Radio-frequency identification)



<https://www.youtube.com/watch?v=MAA9JpGraoU>

**Frequency:** 120–150 kHz (LF), 13.56 MHz (HF), 433 MHz (UHF), 865-868 MHz (Europe) 902-928 MHz (North America) UHF, 2450-5800 MHz (microwave), 3.1–10 GHz (microwave)

**Range:** 10cm to 200m

**Examples:** Road tolls, Building Access, Inventory

# Một số chuẩn truyền thông trong IoT

- NFC (Near-Field Communications)



**Frequency:** 13.56 MHz

**Range:** < 0.2 m

**Examples:** Smart Wallets/Cards, Action Tags, Access Control

[https://www.youtube.com/watch?v=HfMX\\_InfiP0](https://www.youtube.com/watch?v=HfMX_InfiP0)

# Một số chuẩn truyền thông trong IoT

- Bluetooth



**Frequency:** 2.4GHz

**Range:** 1-100m

**Examples:** Hands-free headsets, key dongles, [fitness trackers](#)

<https://www.youtube.com/watch?v=jzxZUJmOu3o>

# Một số chuẩn truyền thông trong IoT

- WiFi



**Frequency:** 2.4 GHz, 3.6 GHz and 4.9/5.0 GHz bands.

**Range:** Common range is up to 100m but can be extended.

**Applications:** Routers, Tablets, etc

<https://www.youtube.com/watch?v=kxLcwIMYmr0>

# Một số chuẩn truyền thông trong IoT

- GSM (Global System for Mobile communications)



**Frequency:** Europe: 900MHz & 1.8GHz, US: 1.9GHz & 850MHz, Full List can be found [here](#).

**Data Rate:** 9.6 kbps

**Examples:** Cell phones, M2M, smart meter, asset tracking

# Một số chuẩn truyền thông trong IoT

- ZigBee



- Standard: ZigBee 3.0 based on IEEE802.15.4
  - Frequency: 2.4GHz
  - Range: 10-100m
  - Data Rates: 250kbps

# Một số chuẩn truyền thông trong IoT

- **6LoWPAN**
- IPv6 protocol over low-power wireless PANs (sử dụng giao thức IPv6 trong các mạng PAN không dây công suất thấp)



IPv6-based Low-power  
Wireless Personal Area Networks

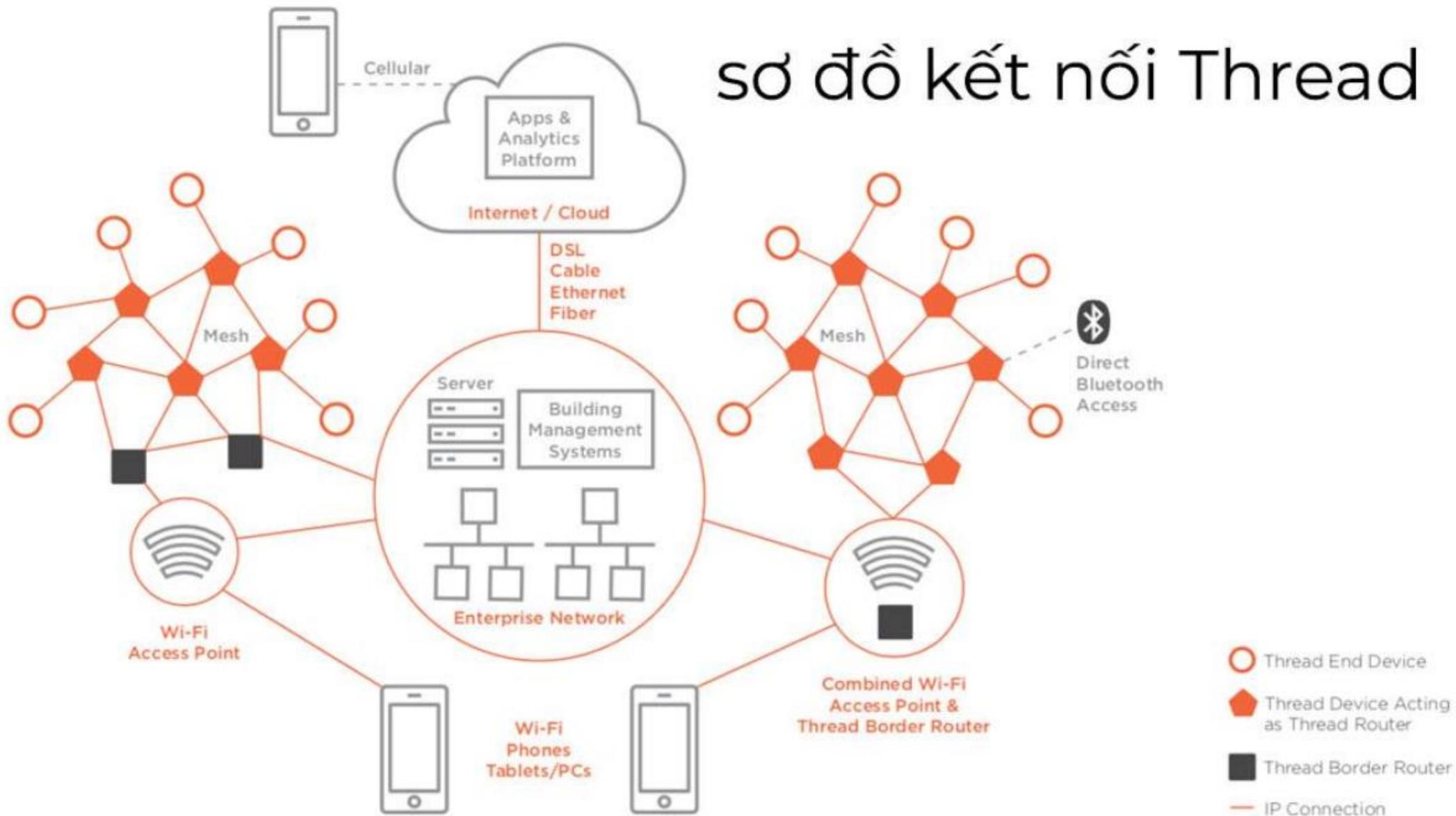
# Một số chuẩn truyền thông trong IoT

- Thread: Giao thức IP mới dựa trên nền tảng IPv6 được thiết kế riêng cho mạng tự động hóa trong nhà thông minh, các tòa nhà.
- Ra mắt vào giữa năm 2014 bởi Theard Group, giao thức Thread dựa trên các tiêu chuẩn khác nhau, bao gồm IEEE802.15.4, IPv6 và 6LoWPAN, và cung cấp một giải pháp dựa trên nền tảng IP cho các ứng dụng IoT
- Tiêu chuẩn: Theard, dựa trên IEEE802.15.4 và 6LowPAN.
- Tần số: 2.4GHz (ISM).

<https://www.youtube.com/watch?v=JLbuT4rYW7Q&t=3s>

# Một số chuẩn truyền thông trong IoT

## sơ đồ kết nối Thread



# Truyền thông trong IoT

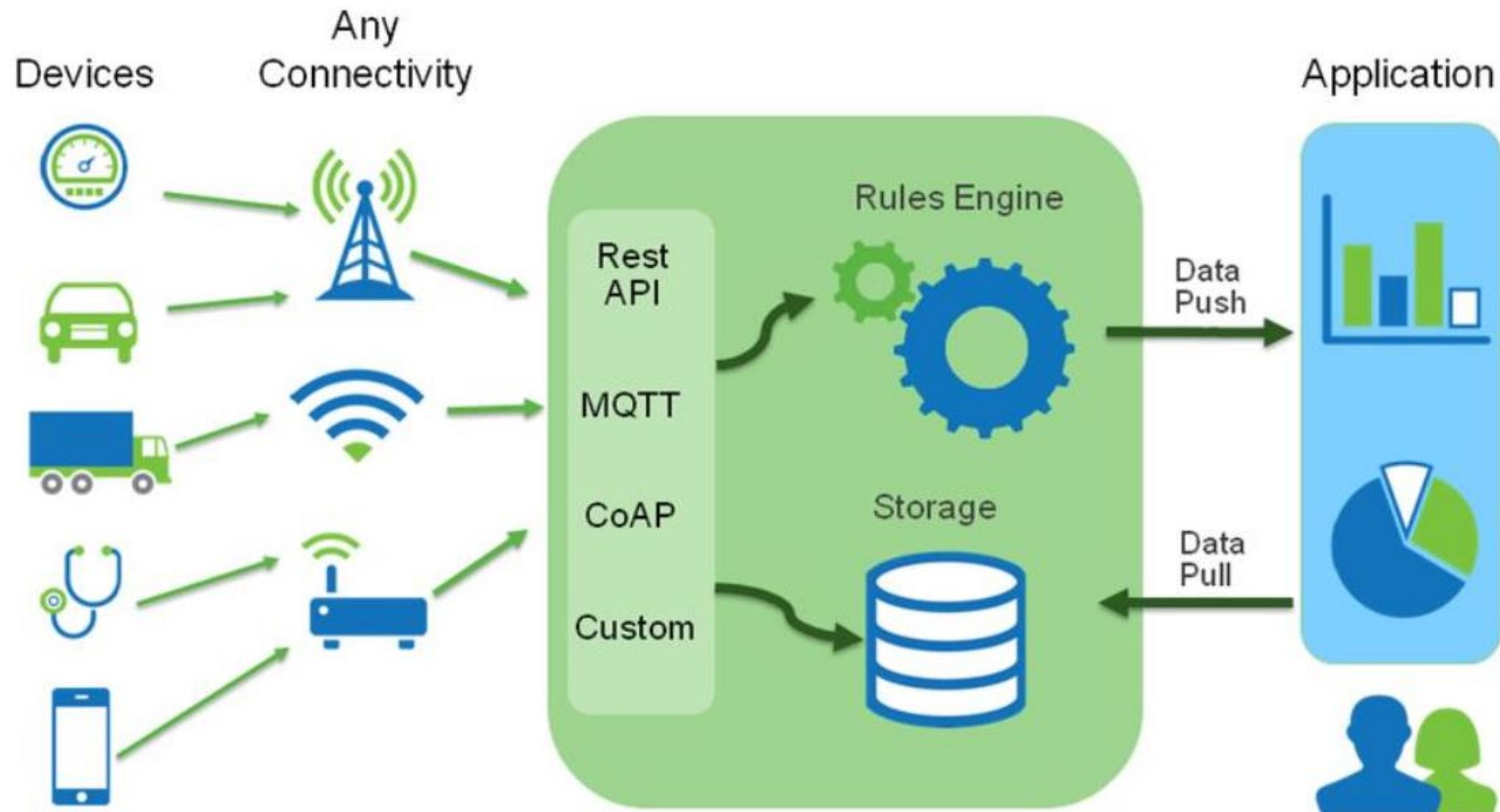
## Additional:

- [3G](#)
- [4G LTE](#)
- [ANT](#)
- [Dash7](#)
- [Ethernet](#)
- [GPRS](#)
- [PLC / Powerline](#)
- [QR Codes, EPC](#)
- [WiMax](#)
- [X-10](#)
- [802.15.4](#)
- [Z-Wave](#)
- [Zigbee](#)

**Backbone:** IPv4, IPv6, TCP/UDP

More: <https://www.postscapes.com/internet-of-things-technologies/>

## 2.3. Một số giao thức cho ứng dụng IoT

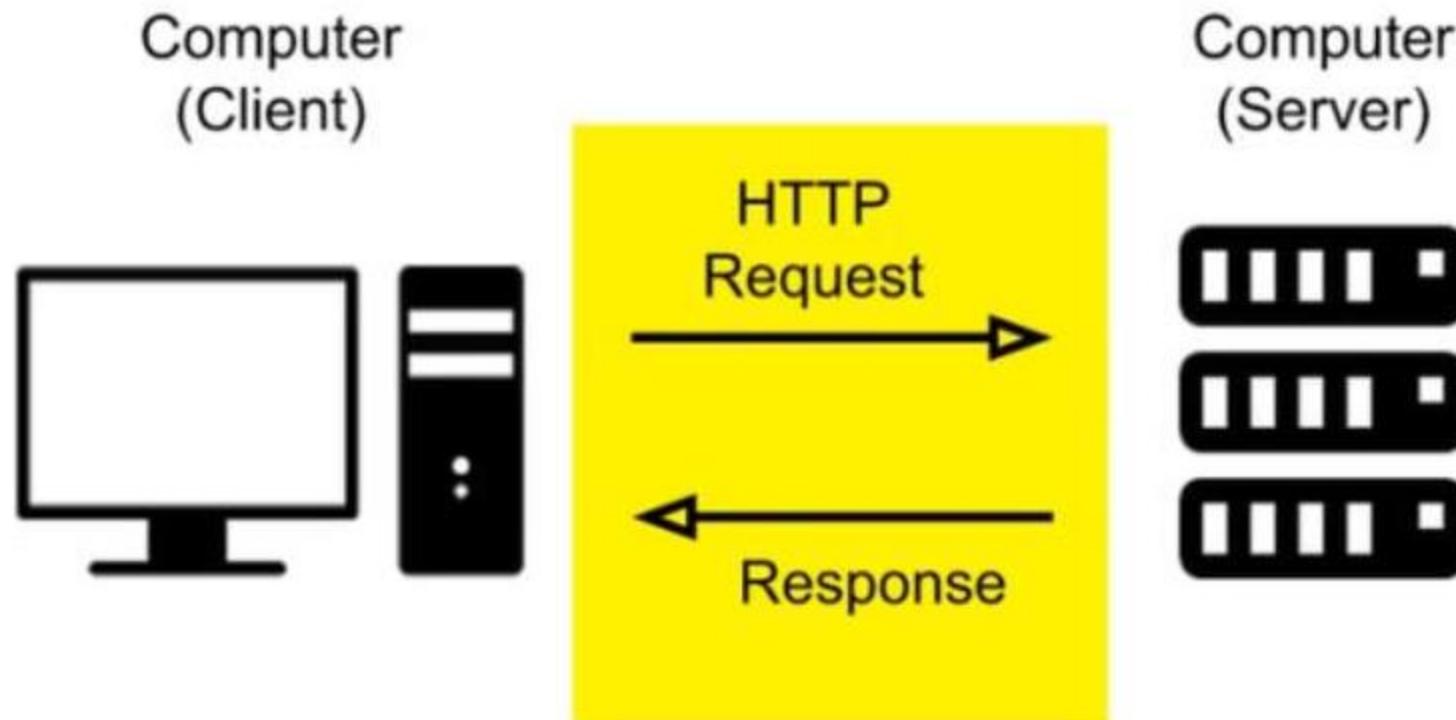


# Các giao thức ứng dụng IoT

- HTTP, HTTPS
- REST API HTTP (Representational State Transfer)
- MQTT (Message Queue Telemetry Transport)
- AMQP (Advanced Message Queue Protocol)

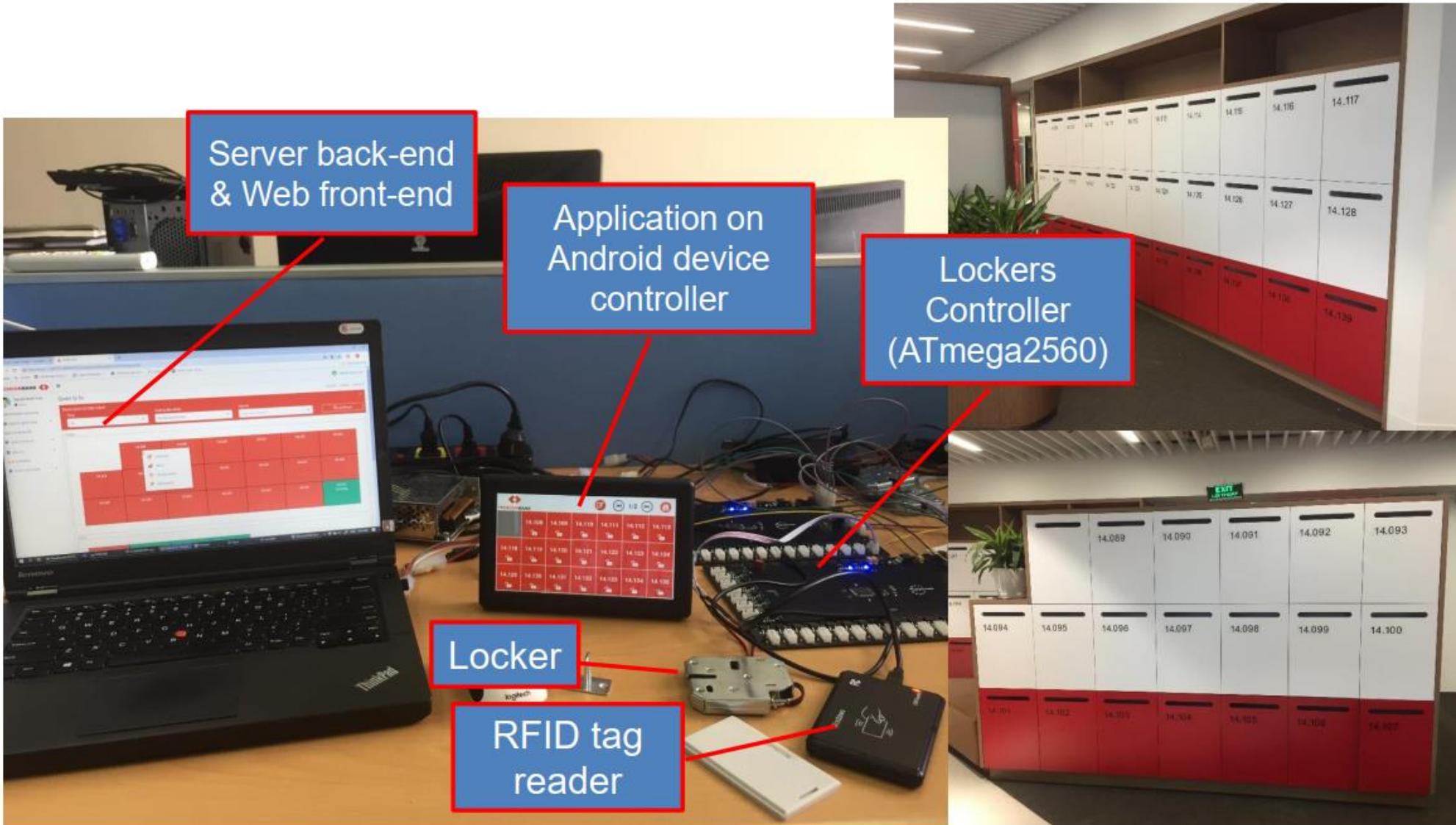
## 2.3.1. Giao thức HTTP

HTTP = HyperText Transfer Protocol



<https://www.youtube.com/watch?v=eesqK59rhGA>

# Ví dụ ứng dụng sử dụng HTTP



Hệ thống Smart Locker

# Ví dụ thực hiện http request với Java (1)

- Sử dụng thư viện HttpURLConnection trên Java thực hiện một http request tới http REST api
- Bước 1:** Tạo một kết nối http connection, thiết lập các thuộc tính cho request

```
public String requestUserInfor(String devId, String qrDevId, String qrCode, int timeout) {
    String result = "";
    try {
        String urlApi =
"https://203.171.20.94:8080/api/AccessControl/GetUserInfor";
        URL url = new URL(urlApi);
        HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
        conn.setRequestMethod("POST");
        conn.setConnectTimeout(timeout);
        conn.setReadTimeout(timeout);
        conn.setRequestProperty("Content-Type", "application/json;
charset=utf-8");
```

## Ví dụ thực hiện http request với Java (2)

- **Bước 2:** Ghi dữ liệu cần gửi vào luồng output stream của request. Dữ liệu có thể được đóng gói bằng JSON

```
...
OutputStream os = conn.getOutputStream();
BufferedWriter writer = new BufferedWriter(new
OutputStreamWriter(os, "UTF-8"));
JSONObject jsonObj = new JSONObject();
try {
    jsonObj.accumulate("deviceId", devId);
    jsonObj.accumulate("qrCodeId", qrDevId);
    jsonObj.accumulate("qrCodeValue", qrCode);
} catch (JSONException e) {
    e.printStackTrace();
}

writer.write(jsonObj.toString());
writer.flush();
writer.close();
os.close();
conn.connect();
```

# Ví dụ thực hiện http request với Java (3)

- **Bước 3:** Nhận phản hồi http response từ server. Kiểm tra mã trả về và đọc dữ liệu từ luồng input stream của http response

```
...
StringBuffer sb = new StringBuffer();
if (conn.getResponseCode() == HttpURLConnection.HTTP_OK) {
    InputStream in = conn.getInputStream();
    int chr;
    while ((chr = in.read()) != -1) {
        sb.append((char) chr);
    }
    in.close();
} else {
    sb.append(conn.getResponseCode());
}
result = sb.toString();
conn.disconnect();
}
catch (IOException e) {
    e.printStackTrace();
}
return result;
}
```

# Http Response code

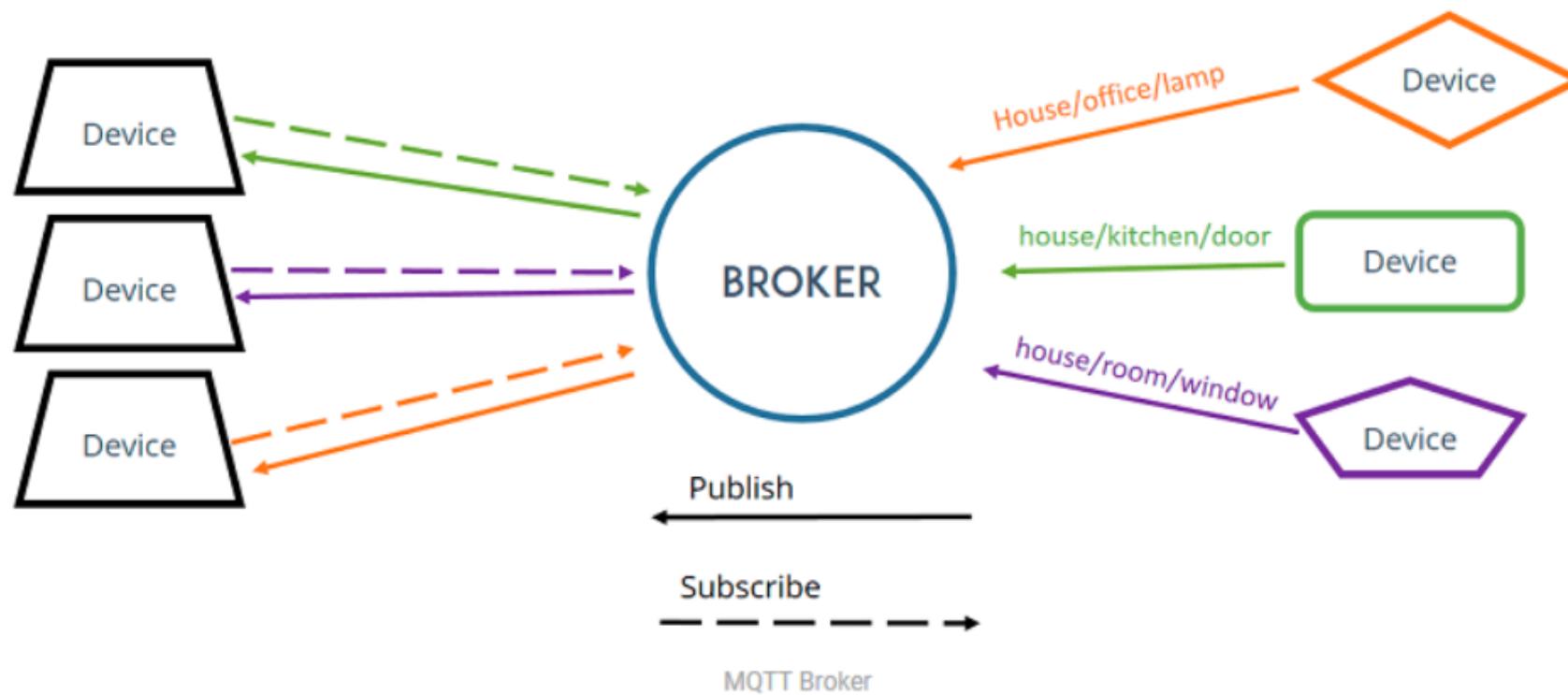
- 200: OK
- 201: Created
- 204: No Content
- 401: Unauthorized
- 403: Forbidden
- 404: Not Found
- 408: Request Timeout
- 500: Internal Server Error
- 502: Bad Gateway
- 503: Service Unavailable

## 2.3.2. Giao thức MQTT

- **The Messaging and Data Exchange Protocol of the IoT**
- **MQTT (Message Queuing Telemetry Transport):**
  - Giao thức truyền thông điệp (message) theo mô hình publish/subscribe (xuất bản – theo dõi)
  - Sử dụng băng thông thấp, độ tin cậy cao và có khả năng hoạt động trong điều kiện đường truyền không ổn định.

# Giao thức MQTT

- Kiến trúc mức cao của MQTT gồm 2 thành phần chính:
  - Broker
  - Clients

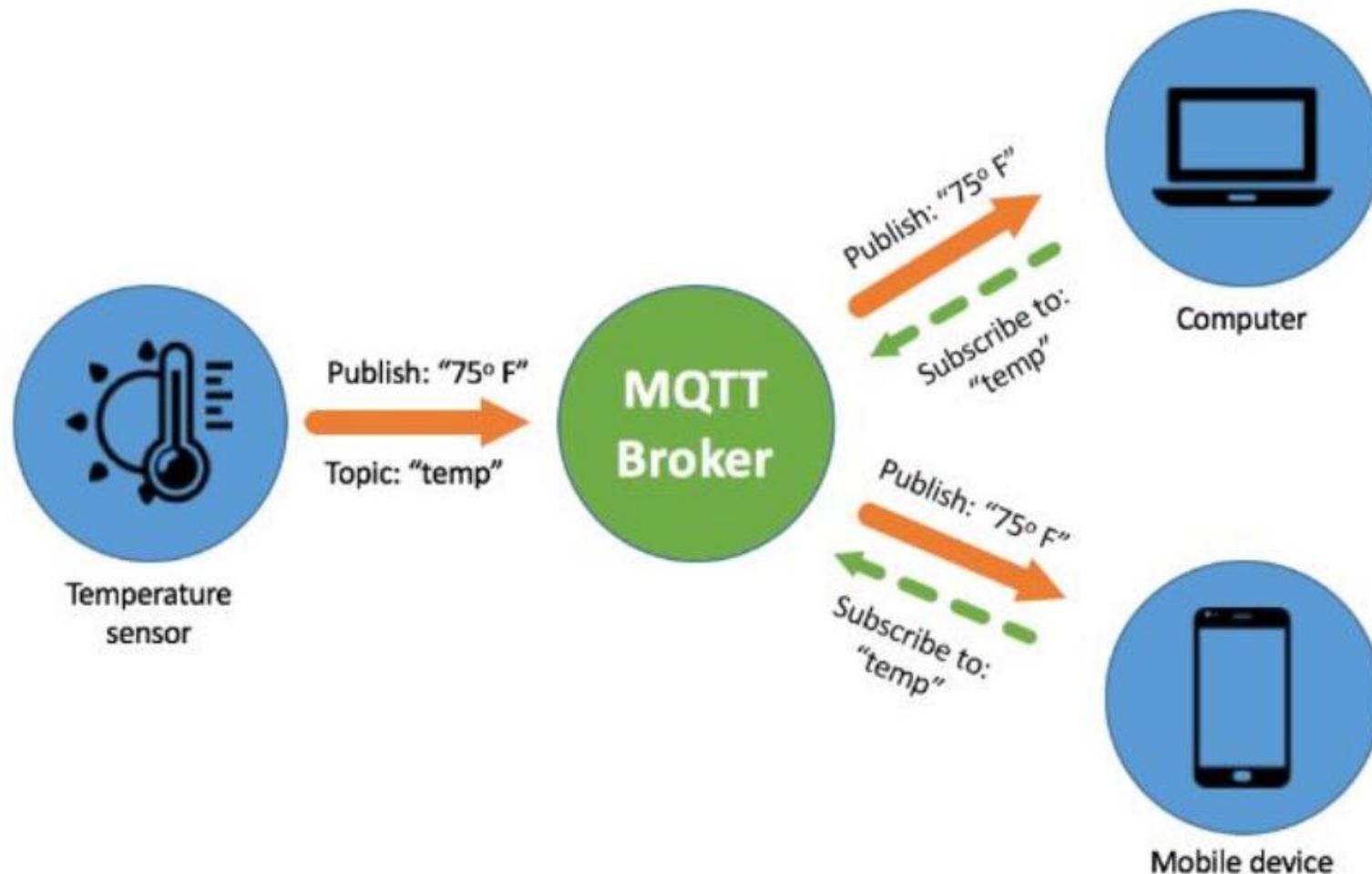


# Giao thức MQTT

- **Publish/Subscribe:**

- Trong một hệ thống sử dụng giao thức MQTT, nhiều node trạm (gọi là mqtt client – gọi tắt là client) kết nối tới một MQTT server (gọi là broker).
- Mỗi client sẽ đăng ký một vài kênh (topic), ví dụ như “/client1/channel1”, “/client1/channel2”. Quá trình đăng ký này gọi là **“subscribe”**. Mỗi client sẽ nhận được dữ liệu khi bất kỳ trạm nào khác gửi dữ liệu vào kênh đã đăng ký.
- Khi một client gửi dữ liệu tới kênh đó, gọi là **“publish”**.

# Giao thức MQTT

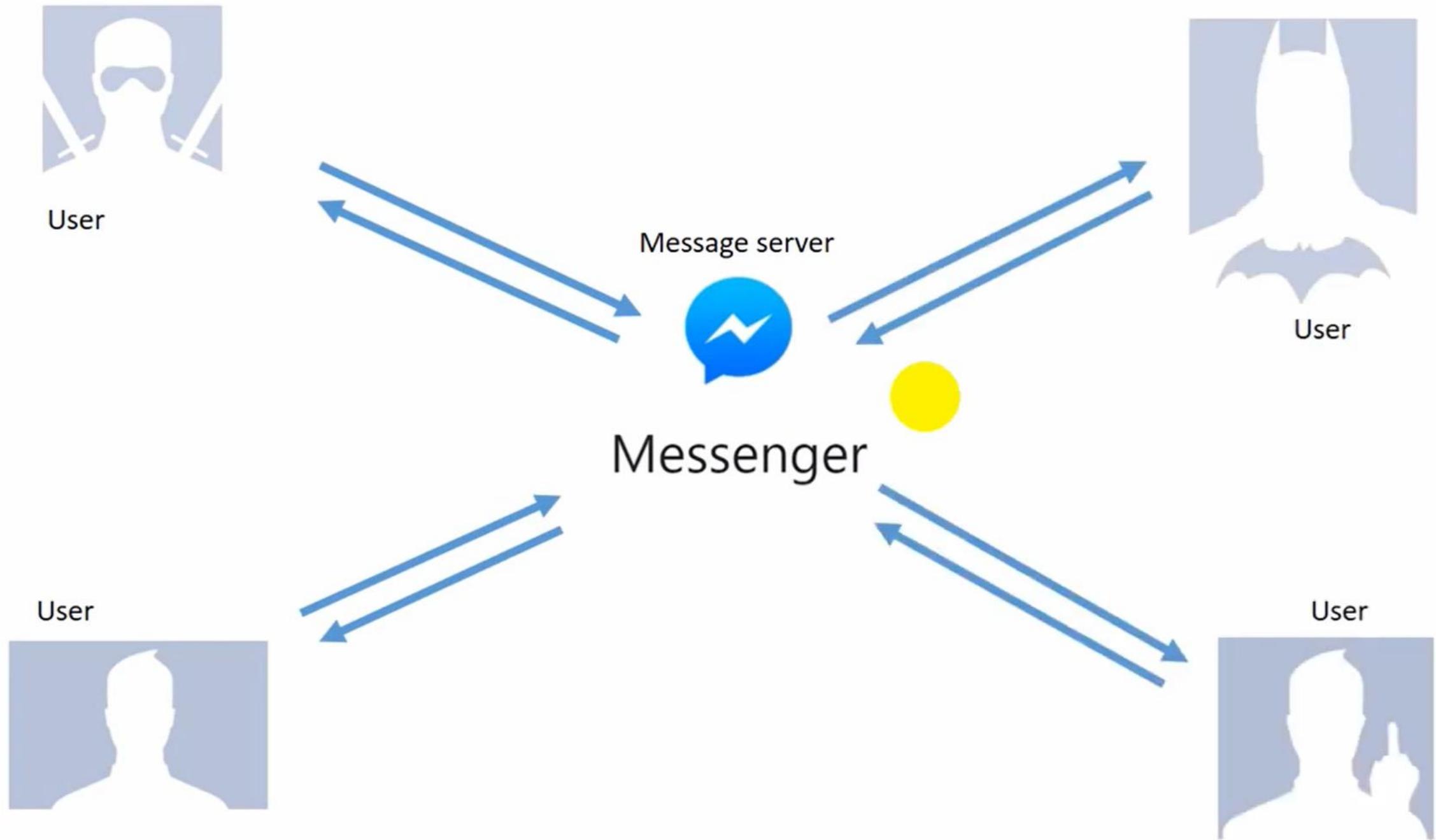


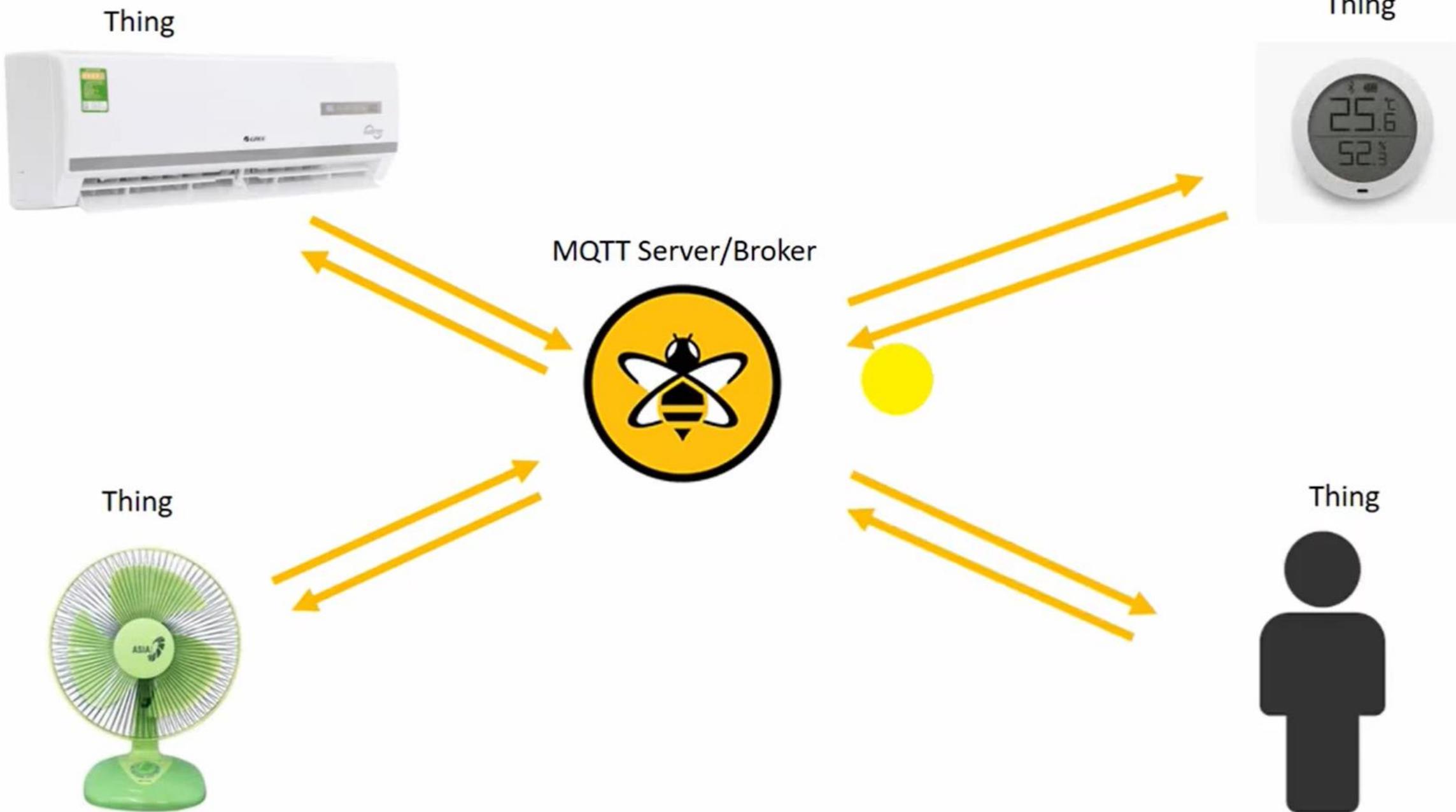
<https://www.hivemq.com/mqtt/>

# Giao thức MQTT

MQTT (originally an initialism of MQ Telemetry Transport) is a lightweight, publish–subscribe, machine-to-machine network protocol for message queue/message queuing service.

It is designed for connections with remote locations that have devices with resource constraints or limited network bandwidth, such as in the Internet of things (IoT). It must run over a transport protocol that provides ordered, lossless, bi-directional connections—typically, TCP/IP. It is an open OASIS standard and an ISO recommendation (ISO/IEC 20922).







temperature  
sensor

publish: "21°C"



MQTT-Broker

subscribe  
publish: "21°C"



laptop

subscribe  
publish: "21°C"



mobile device

- 1 subscribe to  
topic: "temperature"

- 2 publish to  
topic: "temperature"

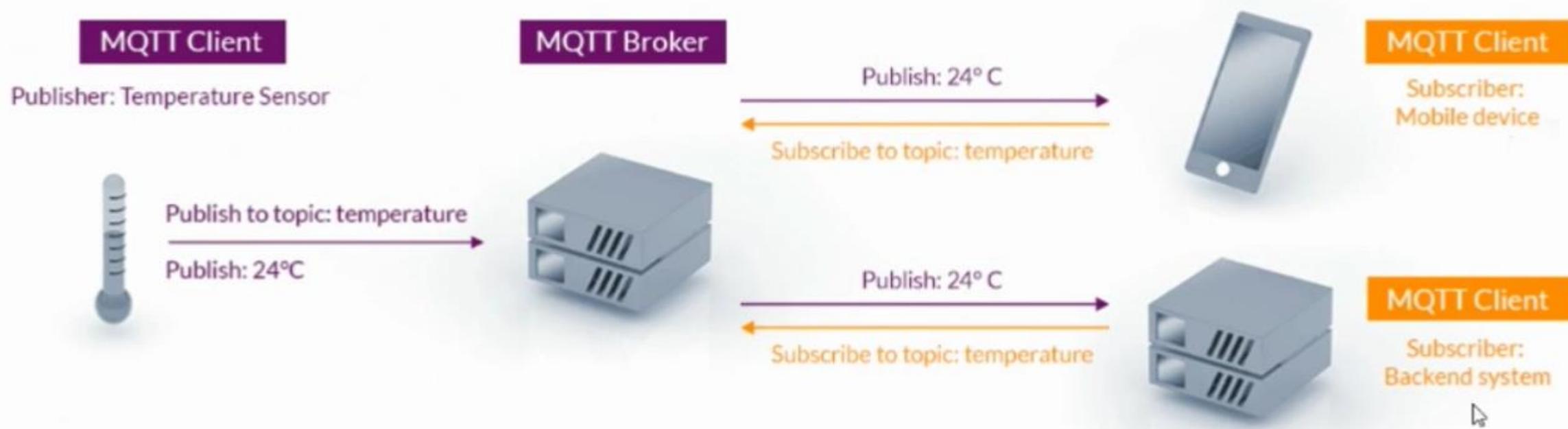
# Giao thức MQTT

- **QoS (Quality of Service):**
  - Có 3 tùy chọn **QoS** khi “publish” và “subscribe”:
  - **QoS0** Broker/client sẽ gửi dữ liệu đúng 1 lần, quá trình gửi được xác nhận bởi chỉ giao thức TCP/IP, (“fire and forget”).
  - **QoS1** Broker/client sẽ gửi dữ liệu với ít nhất 1 lần xác nhận từ đầu kia, nghĩa là có thể có nhiều hơn 1 lần xác nhận đã nhận được dữ liệu.
  - **QoS2** Broker/client đảm bảo khi gửi dữ liệu thì phía nhận chỉ nhận được đúng 1 lần, quá trình này phải trải qua 4 bước bắt tay

# Giao thức MQTT

- Sử dụng công cụ client: MQTTBox
  - <https://www.hivemq.com/mqtt-toolbox/>
  - Publish/Subscribe (gửi/nhận) dữ liệu

## MQTT Publish / Subscribe Architecture



# Công cụ mqtt client: MQTTBox

MQTTBox Edit Help

Menu ← MQTT CLIENT SETTINGS Client Settings Help

MQTT Client Name	MQTT Client Id	Append timestamp to MQTT client id?	Broker is MQTT v3.1.1 compliant?
MQTTClientNew	6e0edddd-582f-48d7-a311-357a61069270	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes
Protocol	Host	Clean Session?	Auto connect on app launch?
mqtt / tcp	broker.hivemq.com	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes
Username	Password	Reschedule Pings?	Queue outgoing QoS zero messages?
Username	Password	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes
Reconnect Period (milliseconds)	Connect Timeout (milliseconds)	KeepAlive (seconds)	Will - Payload
1000	30000	10	
Will - Topic	Will - QoS	Will - Retain	
temperature	0 - Almost Once	<input type="checkbox"/> No	
<b>Save</b>			

The Best Free Public MQTT Broker +

hivemq.com/mqtt/public-mqtt-broker/

YouTube Imported From Fire...

Announcing HiveMQ Pulse, the Distributed Data Intelligence Platform. [Join the beta](#)

 **HIVEMQ** MQTT Platform Pricing Learn Use Cases

Our [Public HiveMQ MQTT broker](#) is open for anyone to use. Feel free to write an MQTT client that connects with this broker. We also keep a list of [MQTT client libraries](#) that can be used to connect to HiveMQ.

You can access the MQTT broker securely at:

Broker: [broker.hivemq.com](https://broker.hivemq.com)

TCP Port: 1883

WebSocket Port: 8000

TLS TCP Port: 8883

TLS WebSocket Port: 8884

Looking for a full-featured MQTT platform? Try HiveMQ.

Try HiveMQ as a managed service or self-hosted.

[Start Free](#)

[View public broker](#)

# Công cụ mqtt client: MQTTBox

MQTTBox

MQTTBox Edit Help

☰ Menu



Connected

Add publisher

Add subscriber



MQTTClientNew - mqtt://broker.hivemq.com

## Topic to publish

Topic to publish

## QoS

0 - Almost Once

Retain

## Payload Type

Strings / JSON / XML / Characters

e.g: {'hello':'world'}

## Payload

**Publish**

## Topic to subscribe

Topic to subscribe

## QoS

0 - Almost Once

**Subscribe**

☰ Menu



Connected

Add publisher

Add subscriber



MQTTClientNew - mqtt://broker.hivemq.com

## QoS

0 - Almost Once

## Retain

## Payload Type

Strings / JSON / XML / Characters

e.g: {'hello':'world'}

## Payload

The temperature sensor reading is 45 degree

Publish

The temperature sensor reading is 45 degree

topic:temperature, qos:0, retain:false



The temperature sensor reading is 40 degree

topic:temperature, qos:0, retain:false



## Topic to subscribe

temperature

## QoS

0 - Almost Once

Subscribe

The temperature sensor reading is 45 degree

```
qos : 0, retain : false, cmd : publish, dup : false, topic : temperature, messageId : , length : 56, Raw payload : 841041013211610110911210111497116117114101321151011011511111432114101971001051101033210511532525332100101103114101101
```

The temperature sensor reading is 40 degree

```
qos : 0, retain : false, cmd : publish, dup : false, topic : temperature, messageId : , length : 56, Raw payload : 841041013211610110911210111497116117114101321151011011511111432114101971001051101033210511532524832100101103114101101
```

MQTTBox

MQTTBox Edit Help

Connected

Add publisher Add subscriber

MQTTClientNew - mqtt://broker.hivemq.com

**QoS**

0 - Almost Once

**Retain**

**Payload Type**

Strings / JSON / XML / Characters

e.g. `{"hello": "world"}`

**Payload**

The temperature sensor reading is 65 degree

**Publish**

The temperature sensor reading is 65 degree  
topic:temperature, qos:0, retain:false

The temperature sensor reading is 65 degree  
topic:temperature, qos:0, retain:false

**Topic to subscribe**

temperature

**QoS**

0 - Almost Once

**Subscribe**

The temperature sensor reading is 65 degree

**qos : 0, retain : false, cmd : publish, dup : false, topic : temperature, messageId : , length : 56, Raw payload :** 841041013211610110911210111497116117114101321151 011101151111143211410197100105110103321051153254533210010110311410110 1

**Topic to subscribe**

temperature

**QoS**

0 - Almost Once

**Subscribe**

The temperature sensor reading is 65 degree

**qos : 0, retain : false, cmd : publish, dup : false, topic : temperature, messageId : , length : 56, Raw payload :** 841041013211610110911210111497116117114101321151 011101151111143211410197100105110103321051153254533210010110311410110 1

The temperature sensor reading is 65 degree

**qos : 0, retain : false, cmd : publish, dup : false, topic : temperature, messageId : , length : 56, Raw payload :** 841041013211610110911210111497116117114101321151 011101151111143211410197100105110103321051153254533210010110311410110 1

# Adding 2 Publishers

MQTTBox

MQTTBox Edit Help

Menu Connected Add publisher Add subscriber

MQTTClientNew - mqtt://broker.hivemq.com

**Topic to publish**  
temperature

**QoS**  
0 - Almost Once

**Retain**

**Payload Type**  
Strings / JSON / XML / Characters

e.g: {'hello':'world'}

**Payload**  
The temperature sensor reading is 65 degree

**Publish**

The temperature sensor reading is 65 degree  
topic:temperature, qos:0, retain:false  
 

The temperature sensor reading is 65 degree  
topic:temperature qos:0 retain:false

**Topic to publish**  
pressure

**QoS**  
0 - Almost Once

**Retain**

**Payload Type**  
Strings / JSON / XML / Characters

e.g: {'hello':'world'}

**Payload**  
The pressure sensor reading is 49|

**Publish**

# Add Subscriber for publisher “pressure”

**Topic to publish**

**QoS**

0 - Almost Once

**Retain**

**Payload Type**

Strings / JSON / XML / Characters

e.g: {'hello': 'world'}

**Payload**

The pressure sensor reading is 49

**Publish**

The pressure sensor reading is 49  
topic:pressure, qos:0, retain:false

**Topic to subscribe**

**QoS**

0 - Almost Once

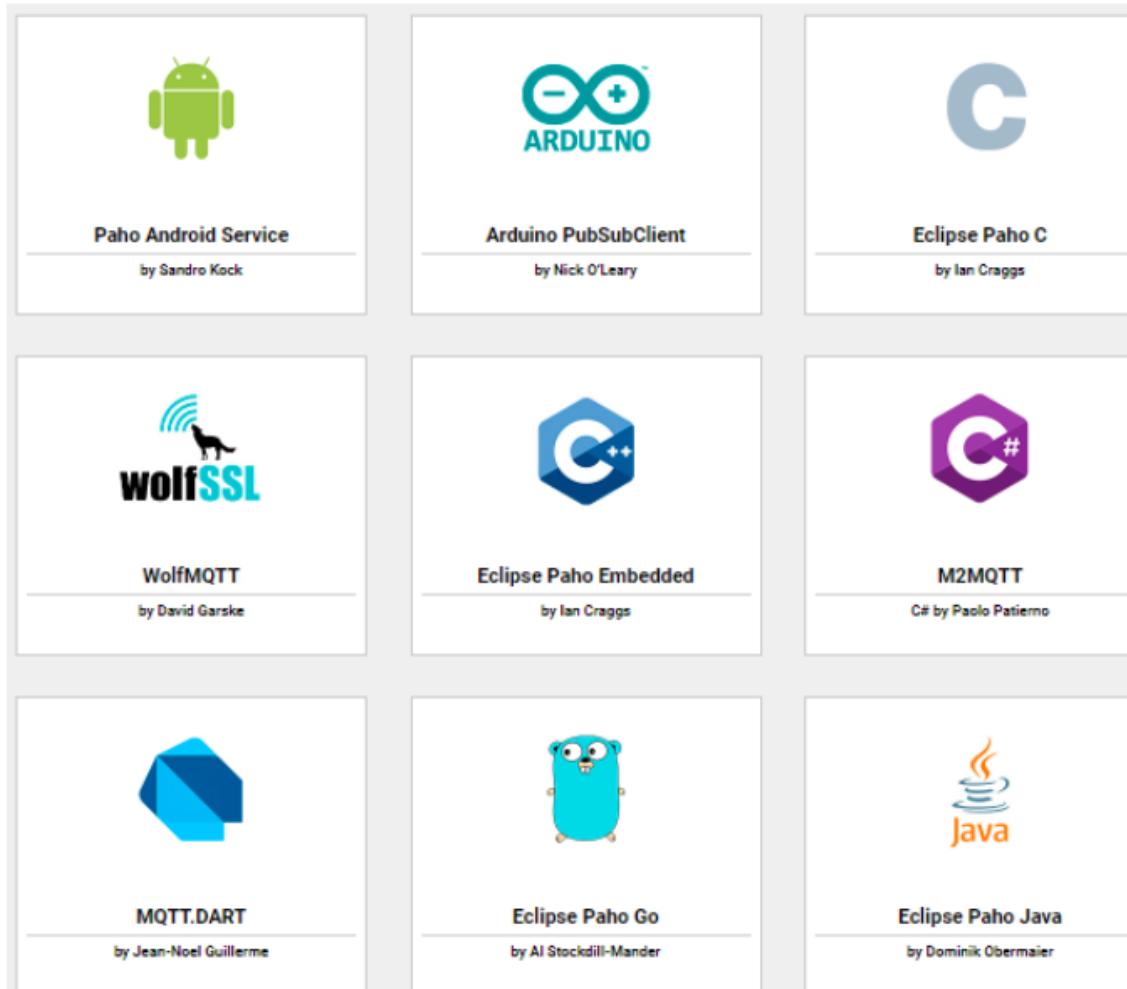
**Subscribe**

The pressure sensor reading is 49

**qos : 0, retain : false, cmd : publish, dup : false, topic : pressure, messageId : , length : 13, Raw payload : 8A10A1013211211A10111511511711A1013211510111011511**

# Thư viện MQTT client

- Thư viện mqtt client cho các nền tảng khác nhau:
  - <https://www.hivemq.com/mqtt-client-library-encyclopedia>



# Ví dụ 1

- publish/subscribe (gửi/nhận) dữ liệu sử dụng thư viện Paho mqtt client cho Java
- Thư viện Paho mqtt client
  - [www.eclipse.org/paho/index.php?page=clients/java/index.php](http://www.eclipse.org/paho/index.php?page=clients/java/index.php)
- Using the Paho Java Client:
  - Using Maven dependency
  - Hoặc add thư viện .jar (Build Path >> Add External Archives):  
Download file thư viện: org.eclipse.paho.client.mqttv3-1.1.1.jar

# Publish message

```
import org.eclipse.paho.client.mqttv3.MqttClient;
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;
import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;
public class App {
    public static void main(String[] args) {
        public static void main( String[] args ){
            System.out.println( "MQTT Demo!" );
            new App().doDemo();
        }
        public void doDemo() {
            String subTopic = "/ktmt/iot";
            String pubTopic = "/ktmt/iot";
            String content = "Hello World";
            int qos = 1;
            String broker = "tcp://broker.hivemq.com:1883";
            String clientId = "ID_OF_CLIENT";
            MemoryPersistence persistence = new MemoryPersistence();
            try {
                MqttClient client = new MqttClient(broker, clientId,
persistence);
```

```
// MQTT connection option
MqttConnectOptions connOpts = new MqttConnectOptions();
// retain session
connOpts.setCleanSession(true);
// set callback
client.setCallback(new OnMessageCallback());
// establish a connection
System.out.println("Connecting to broker: " + broker);
client.connect(connOpts);
System.out.println("Connected");
// Subscribe
client.subscribe(subTopic);
// Required parameters for message publishing
MqttMessage message = new MqttMessage(content.getBytes());
message.setQos(qos);
System.out.println("Publishing message: " + content);
client.publish(pubTopic, message);
System.out.println("Message published");
client.disconnect();
System.out.println("Disconnected");
client.close();
System.exit(0);
}
catch (MqttException me) {
    me.printStackTrace();
}
}
```

# Subscribe - Callback

```
public class OnMessageCallback implements MqttCallback {  
    public void connectionLost(Throwable cause) {  
        // After the connection is lost, it usually reconnects here  
        System.out.println("disconnect, you can reconnect");  
    }  
    public void messageArrived(String topic, MqttMessage message)  
throws Exception {  
        // The messages obtained after subscribe will be executed here  
        System.out.println("Received message topic:" + topic);  
        System.out.println("Received message Qos:" + message.getQos());  
        System.out.println("Received message content:" + new  
            String(message.getPayload()));  
    }  
    public void deliveryComplete(IMqttDeliveryToken token) {  
        System.out.println("deliveryComplete-----" +  
token.isComplete());  
    }  
}
```

## Ví dụ 2:

- Nhận dữ liệu thu thập cảm biến từ các thiết bị gửi lên
- Tham khảo mã nguồn:
  - [https://github.com/hungpn37/iot\\_mqttdemo](https://github.com/hungpn37/iot_mqttdemo)

## Bài tập 2

- Viết một chương trình mqtt client thực hiện:
  - Gửi (publish) dữ liệu lên mqtt broker
  - Nhận (subscribe) dữ liệu từ mqtt broker
  - Đóng gói dữ liệu bằng JSON. Ví dụ:

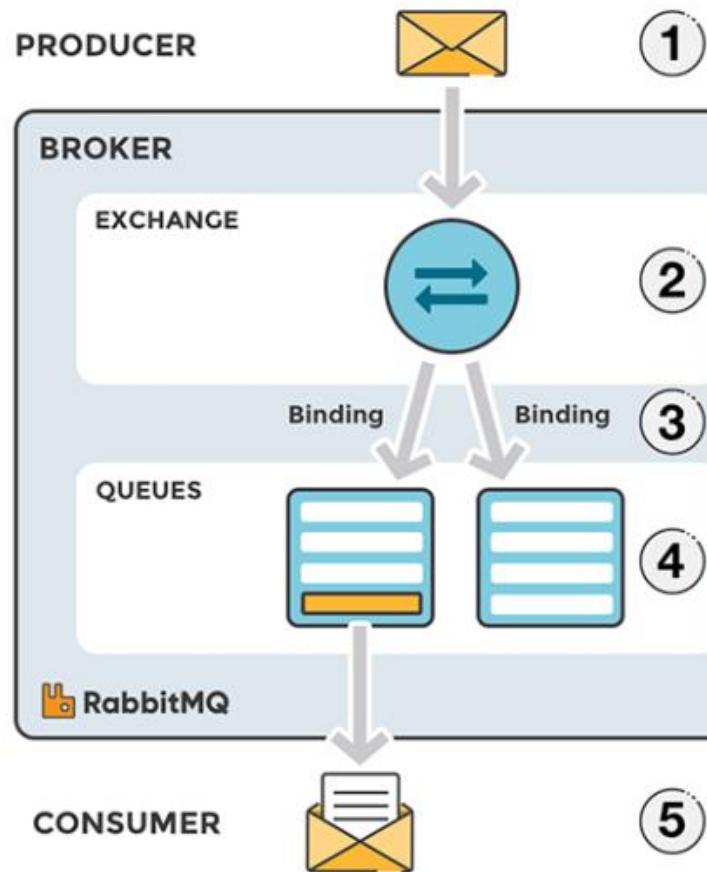
```
{"id":11, "packet_no":126, "temperature":30,  
"humidity":60, "tds":1100, "pH":5.0}
```
  - Dùng MQTTBox để minh họa gửi nhận
  - Ví dụ dùng mqtt broker:  
**tcp://broker.hivemq.com:1883**

### 2.3.3. Giao thức AMQP

- AMQP (Advanced Message Queue Protocol): Giao thức truyền nhận dùng hàng đợi thông điệp
- RabbitMQ: Một broker sử dụng giao thức AMQP
- RabbitMQ broker đóng vai trò trung gian lưu trữ cũng như điều phối thông điệp (message) giữa bên gửi (producer) và bên nhận (consumer)

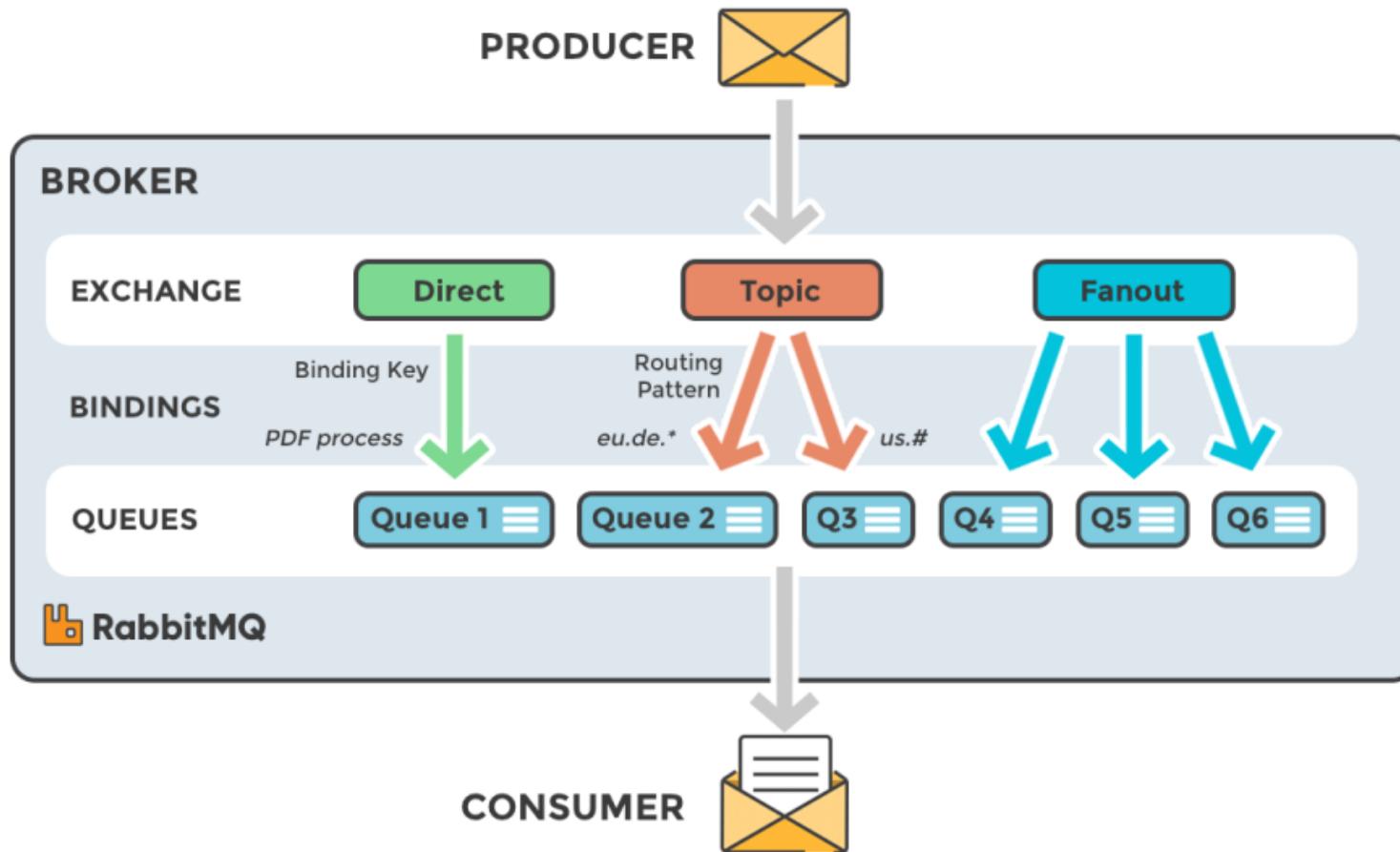
# Giao thức AMQP

- RabbitMQ:
  - Luồng gửi nhận message qua RabbitMQ



# Giao thức AMQP

- RabbitMQ:
  - 4 loại Exchange: direct, topic, fanout, headers



# Giao thức AMQP

- Cài đặt RabbitMQ broker (server)
  - [www.rabbitmq.com/download.html](http://www.rabbitmq.com/download.html)
  - [www.rabbitmq.com/install-windows.html](http://www.rabbitmq.com/install-windows.html)
- Quản trị RabbitMQ broker bằng công cụ:
  - [www.rabbitmq.com/management.html](http://www.rabbitmq.com/management.html)
- RabbitMQ tutorials:
  - [www.rabbitmq.com/getstarted.html](http://www.rabbitmq.com/getstarted.html)

Installing on Windows | RabbitMQ +

← → ⌂ rabbitmq.com/docs/install-windows

YouTube Imported From Fire...

Get long term support and advanced enterprise features

# RabbitMQ

Getting Started Docs Blog Support

Introduction

Release Information

Getting Started

Install and Upgrade

Erlang Version Requirements

Package Signatures

Supported Operating Systems

Linux/Unix >

Windows

MacOS >

Kubernetes Operator

Home > Install and Upgrade > Supported Operating Systems > Windows

Version: 4.0

# Installing on Windows

## Overview

This guide covers RabbitMQ installation on Windows. It focuses on the two recommended installation options:

- Using Chocolatey
- Using the official installer as an administrative user

The guide also covers a few post-installation topics in the context of Windows:

- The basics of node configuration

## Dependencies

RabbitMQ requires a 64-bit [supported version of Erlang](#) for Windows to be installed.

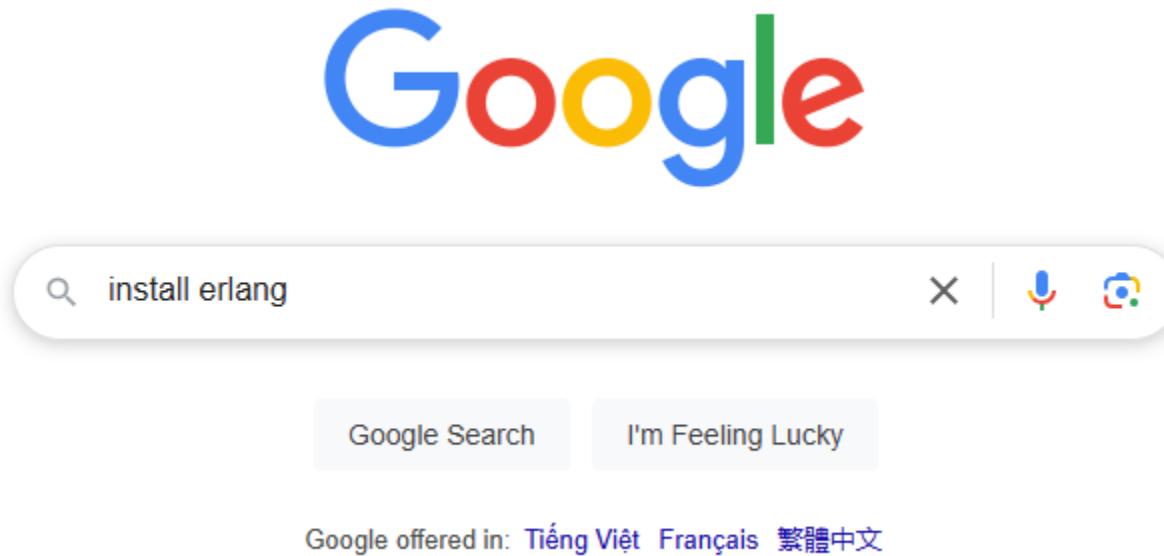
Binary builds of recent versions of Erlang for Windows can be obtained from the [Erlang/OTP Version Tree](#) page.

Erlang **must be installed using an administrative account** or it won't be discoverable to the RabbitMQ Windows service. Once a supported version of Erlang is installed, download the RabbitMQ installer, `rabbitmq-server-{version}.exe` and run it. It installs RabbitMQ as a Windows service and starts it using the default configuration.

## Direct Downloads

Description	Download	Signature
Installer for Windows systems (from <a href="#">GitHub</a> )	<a href="#">rabbitmq-server-4.0.7.exe</a>	<a href="#">Signature</a>

# Cài đặt ngôn ngữ erlang





Erlang/OTP

<https://www.erlang.org> › downloads



## Downloads

Download Erlang/OTP. The latest version of Erlang/OTP is 27.3.1. To **install Erlang** you can either build it from source or use a pre-built package.

### Building and Installing Erlang ...



This document describes how to build and install Erlang/OTP-27 ...

### Highlights



This release of Erlang/OTP can be built from source or installed ...

The screenshot shows a browser window with two tabs: 'ws | Rabbit...' and 'Downloads - Erlang/OTP'. The address bar shows 'erlang.org/downloads'. The page content includes the Erlang logo, navigation links for DOWNLOAD, DOCUMENTATION, COMMUNITY, NEWS, BLOG, SECURITY, EEP, and ABOUT, and a search bar. The main content area features a heading 'Download Erlang/OTP' and text about building from source or using a pre-built package. On the right side, there is a sidebar with a title 'Erlang/OTP 27.3.1' and four buttons: 'Download source' (highlighted with a red arrow), 'Download Windows installer', 'Download Release notes', and 'View documentation'.

Download Erlang/OTP

The latest version of Erlang/OTP is [27.3.1](#). To install Erlang you can either build it [from source](#) or use a [pre-built package](#).

Take a look at the [Erlang/OTP 27 release description](#) to see what changes Erlang/OTP 27 brings over the previous major version.

The Erlang/OTP version scheme is described in the [Erlang/OTP Systems Principles Guide](#).

Erlang/OTP 27.3.1

- [Download source](#)
- [Download Windows installer](#)
- [Download Release notes](#)
- [View documentation](#)

Combilina Erlana from source #

User Account Control



Do you want to allow this app to make changes to your device?



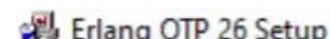
Verified publisher: Ericsson AB

File origin: Hard drive on this computer

Show more details

Yes

No



### Choose Components

Choose which features of Erlang OTP 26 you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

- Microsoft DLL's (present)
- Erlang
- Development
- Associations
- Erlang Documentation

Description  
Microsoft redistributable C runtime libraries, these are mandatory for Erlang runtime and development. Always installed if not already present.

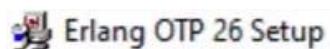
Space required: 477.5 MB

Nullsoft Install System v3.05

Next >

Cancel





## Choose Install Location

Choose the folder in which to install Erlang OTP 26.



Setup will install Erlang OTP 26 in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

### Destination Folder

C:\Program Files\Erlang OTP

[Browse...](#)

Space required: 477.5 MB

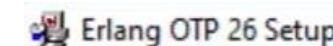
Space available: 25.6 GB

Nullsoft Install System v3.05

< Back

[Next >](#)

Cancel



## Choose Start Menu Folder

Choose a Start Menu folder for the Erlang OTP 26 shortcuts.



Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

### Erlang OTP 26 (x64)

- Accessibility
- Accessories
- Administrative Tools
- Android Studio
- Bloodshed Dev-C++
- BlueStacks X
- CodeBlocks
- Eclipse
- Git
- GitHub, Inc
- Java Development Kit

Do not create shortcuts

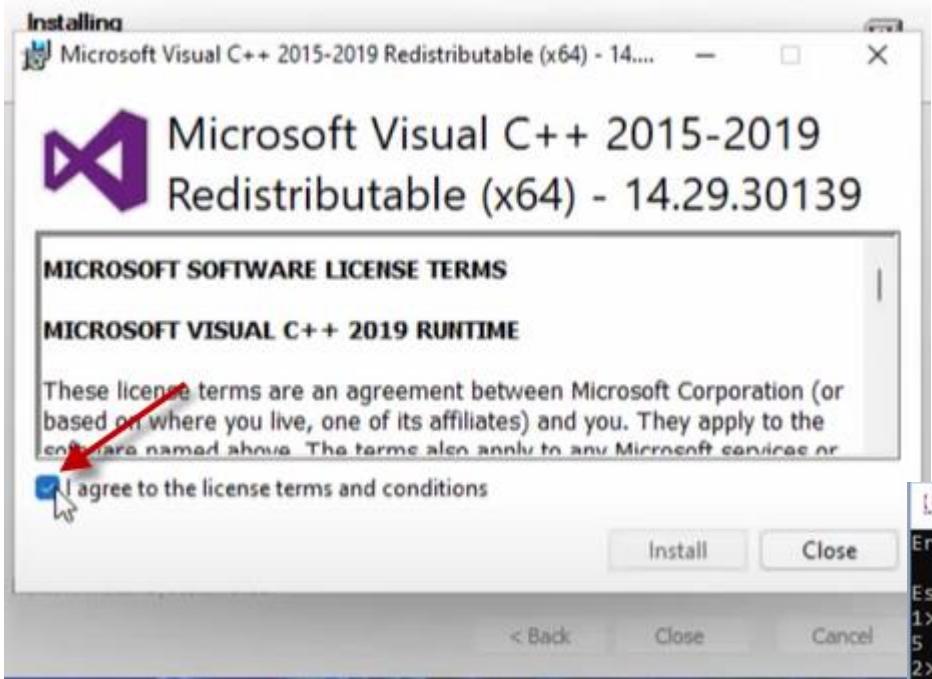
Nullsoft Install System v3.05

< Back

[Install](#)

Cancel





The screenshot shows the 'Select Erlang' window with the title 'Erlang/OTP 26 [erts-14.0.2] [source] [64-bit] [smp:4:4] [ds:4:4:10] [async-threads:1] [jit:ns]'. It displays the Eshell prompt: 'Eshell V14.0.2 (press Ctrl+G to abort, type help(). for help)' followed by the command '1> 2+3.' and the result '5'. A yellow text overlay at the bottom right reads 'test Erlang is working'.

```
Erlang/OTP 26 [erts-14.0.2] [source] [64-bit] [smp:4:4] [ds:4:4:10] [async-threads:1] [jit:ns]
Eshell V14.0.2 (press Ctrl+G to abort, type help(). for help)
1> 2+3.
5
2>
```

test Erlang is working

## Dependencies

RabbitMQ requires a 64-bit [supported version of Erlang](#) for Windows to be installed.

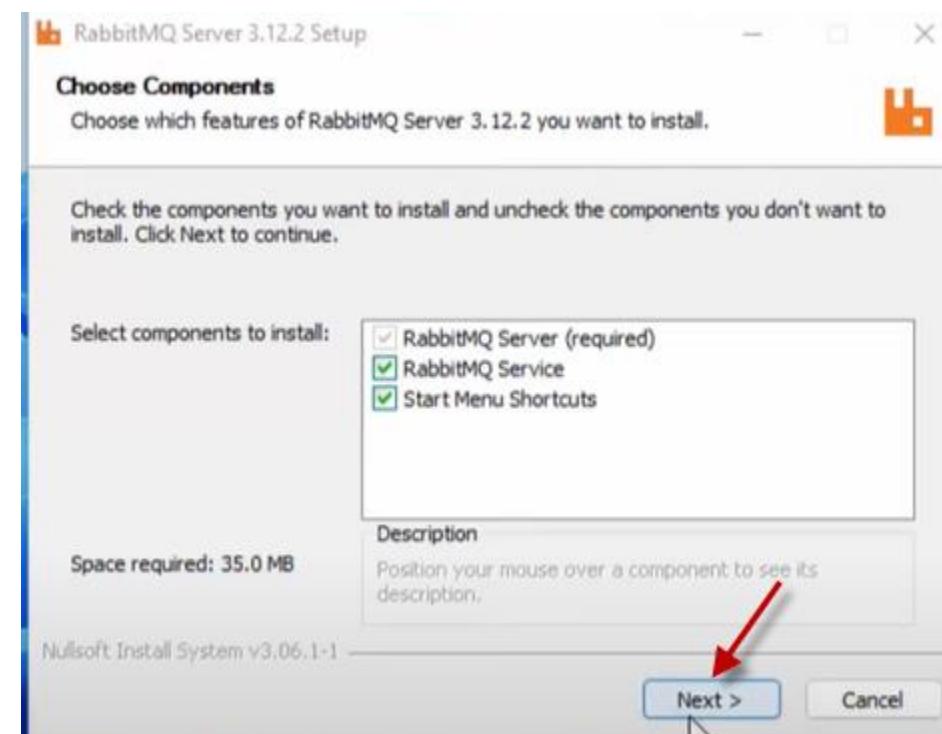
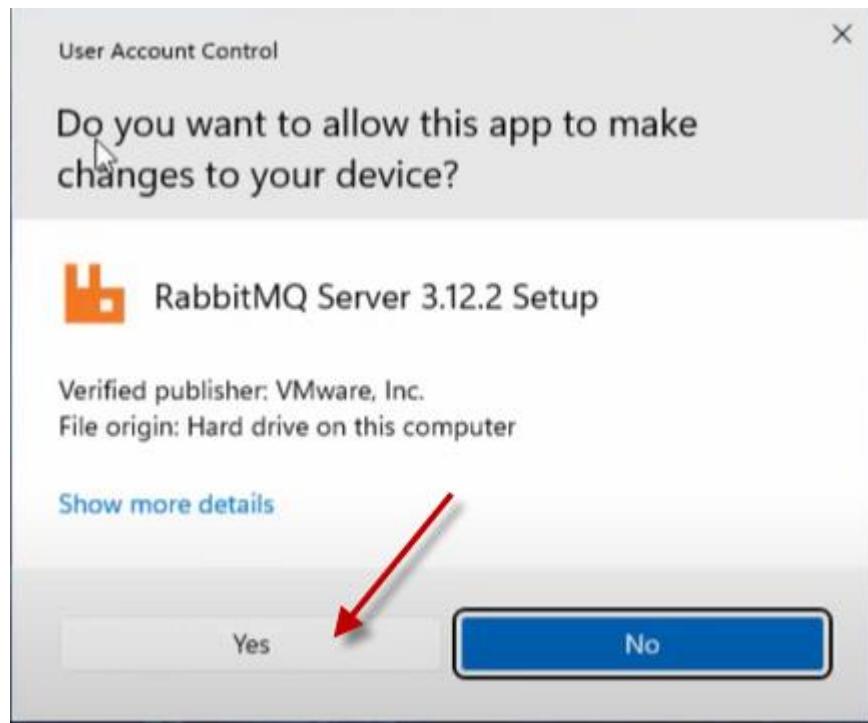
Binary builds of recent versions of Erlang for Windows can be obtained from the [Erlang/OTP Version Tree](#) page.

Erlang **must be installed using an administrative account** or it won't be discoverable to the RabbitMQ Windows service. Once a supported version of Erlang is installed, download the RabbitMQ installer, `rabbitmq-server-{version}.exe` and run it. It installs RabbitMQ as a Windows service and starts it using the default configuration.

## Direct Downloads

Description	Download	Signature
Installer for Windows systems (from <a href="#">GitHub</a> )	<a href="#">rabbitmq-server-4.0.7.exe</a>	<a href="#">Signature</a>





RabbitMQ Server 3.12.2 Setup

Choose Install Location

Choose the folder in which to install RabbitMQ Server 3.12.2.



Setup will install RabbitMQ Server 3.12.2 in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder

C:\Program Files\RabbitMQ Server

Browse...

Space required: 35.0 MB

Space available: 25.1 GB

Nullsoft Install System v3.06.1-1

< Back

Install

Cancel



RabbitMQ Server 3.12.2 Setup

Installing

Please wait while RabbitMQ Server 3.12.2 is being installed.



Size:3794 Kb Files:396 Folders:152

Show details



Nullsoft Install System v3.06.1-1

< Back

Next >

Cancel

Windows Security Alert

Windows Defender Firewall has blocked some features of this app

Windows Defender Firewall has blocked some features of Erlang on all public networks.



Name: Erlang

Publisher: Ericsson AB

Path: C:\program files\erlang otp\bin\erl.exe

Allow Erlang to communicate on these networks:

Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

[What are the risks of allowing an app through a firewall?](#)

Allow access

Cancel



RabbitMQ Server 3.12.2 Setup

Installation Complete

Setup was completed successfully.



Completed

Show details

Nullsoft Install System v3.06.1-1

< Back

Next >

Cancel



```
C:\Program Files\RabbitMQ Server\rabbitmq_server-4.0.7\sbin>rabbitmq-plugins.bat list
```

```
C:\Program Files\RabbitMQ Server\rabbitmq_server-4.0.7\sbin>rabbitmq-plugins.bat list
Listing plugins with pattern ".*" ...
Configured: E = explicitly enabled; e = implicitly enabled
| Status: [failed to contact rabbit@Win10Pro - status not shown]
| /
[ ] rabbitmq_amqp1_0          4.0.7
[ ] rabbitmq_auth_backend_cache 4.0.7
[ ] rabbitmq_auth_backend_http 4.0.7
[ ] rabbitmq_auth_backend_ldap 4.0.7
[ ] rabbitmq_auth_backend_oauth2 4.0.7
[ ] rabbitmq_auth_mechanism_ssl 4.0.7
[ ] rabbitmq_consistent_hash_exchange 4.0.7
[ ] rabbitmq_event_exchange    4.0.7
[ ] rabbitmq_federation        4.0.7
[ ] rabbitmq_federation_management 4.0.7
[ ] rabbitmq_federation_prometheus 4.0.7
[ ] rabbitmq_jms_topic_exchange 4.0.7
```

```
C:\Program Files\RabbitMQ Server\rabbitmq_server-4.0.7\sbin>rabbitmq-plugins.bat enable rabbitmq_management
Enabling plugins on node rabbit@Win10Pro:
rabbitmq_management
The following plugins have been configured:
  rabbitmq_management
  rabbitmq_management_agent
  rabbitmq_web_dispatch
Applying plugin configuration to rabbit@Win10Pro...
The following plugins have been enabled:
  rabbitmq_management
  rabbitmq_management_agent
  rabbitmq_web_dispatch
set 3 plugins.
Offline change; changes will take effect at broker restart.
```

```
C:\Program Files\RabbitMQ Server\rabbitmq_server-4.0.7\sbin>rabbitmq-plugins.bat list
Listing plugins with pattern ".*" ...
Configured: E = explicitly enabled; e = implicitly enabled
| Status: [failed to contact rabbit@Win10Pro - status not shown]
|/
[ ] rabbitmq_amqp1_0          4.0.7
[ ] rabbitmq_auth_backend_cache 4.0.7
[ ] rabbitmq_auth_backend_http   4.0.7
[ ] rabbitmq_auth_backend_ldap    4.0.7
[ ] rabbitmq_auth_backend_oauth2 4.0.7
[ ] rabbitmq_auth_mechanism_ssl 4.0.7
[ ] rabbitmq_consistent_hash_exchange 4.0.7
[ ] rabbitmq_event_exchange      4.0.7
[ ] rabbitmq_federation          4.0.7
[ ] rabbitmq_federation_management 4.0.7
[ ] rabbitmq_federation_prometheus 4.0.7
[ ] rabbitmq_jms_topic_exchange   4.0.7
[E ] rabbitmq_management         4.0.7
[e ] rabbitmq_management_agent    4.0.7
```

A screenshot of a web browser window. The address bar shows the URL `127.0.0.1:15672`. Below the address bar, there are several icons: a refresh arrow, a circular arrow, a house icon, and a folder icon labeled "Imported From Fire...". A red line has been drawn under the URL in the address bar.



## This site can't be reached

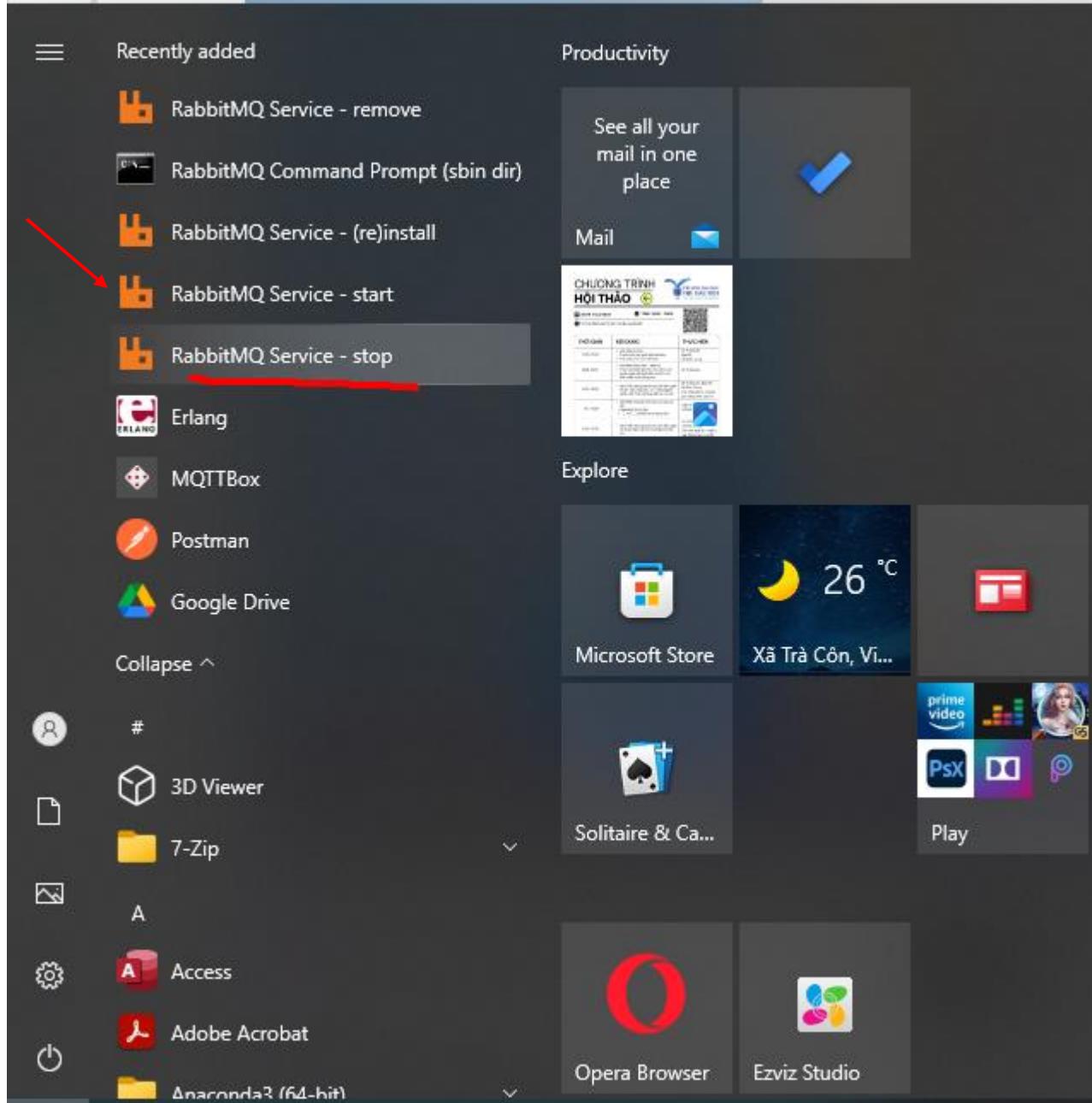
`127.0.0.1` refused to connect.

Try:

- Checking the connection
- Checking the proxy and the firewall

ERR\_CONNECTION\_REFUSED

Reload





Username:  \*

Password:  \*

Login

username: guest  
password: guest

RabbitMQ Management - X

127.0.0.1:15672/#/ ↻ ☆ ↑ ↓ T ⋮

YouTube Imported From Fire...

# RabbitMQ™

RabbitMQ 4.0.7 Erlang 27.3.1

Refreshed 2025-04-03 05:14:51 Refresh every 5 seconds

Virtual host All

Cluster rabbit@Win10Pro User guest Log out

All stable feature flags must be enabled after completing an upgrade. [Learn more]

**Overview** **Connections** **Channels** **Exchanges** **Queues and Streams** **Admin**

## Overview

**Totals**

Queued messages last minute ?

Currently idle

Message rates last minute ?

Currently idle

Global counts ?

Connections: 0 Channels: 0 Exchanges: 7 Queues: 0 Consumers: 0

**Nodes**

Name	File descriptors ?	Erlang processes	Memory ?	Disk space	Uptime	Cores	Info	Reset stats	+/-
rabbit@Win10Pro	0 65536 available	416 1048576 available	68 MiB 9.3 GiB high watermark	10.0 GiB 48 MiB low watermark	5m 52s	4	basic 1 rss	This node All nodes	+/-

Churn statistics

Ports and contexts

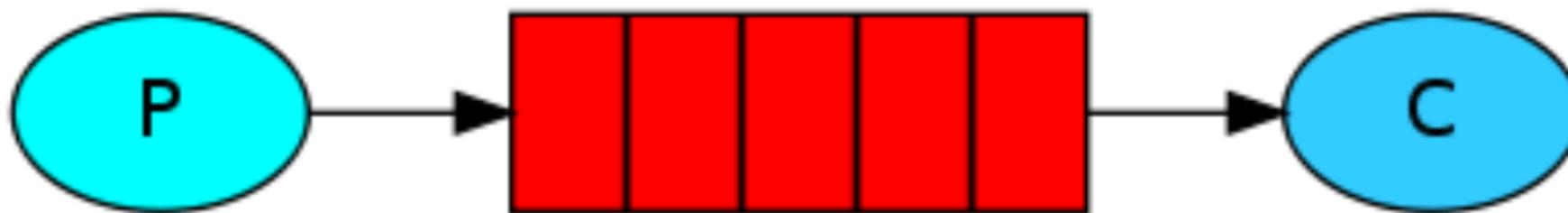
Export definitions

Import definitions

[HTTP API](#) [Documentation](#) [Tutorials](#) [New releases](#) [Commercial edition](#) [Commercial support](#) [Discussions](#) [Discord](#) [Plugins](#) [GitHub](#)

# Tutorial 1

- Viết chương trình gửi nhận dữ liệu qua RabbitMQ
  - Cài đặt, sử dụng một RabbitMQ broker
  - Chương trình Producer gửi dữ liệu (message)
  - Chương trình Consumer nhận dữ liệu
    - Sử dụng default exchange (type: direct)



[www.rabbitmq.com/tutorials/tutorial-one-java.html](http://www.rabbitmq.com/tutorials/tutorial-one-java.html)

# Tutorial 1. Producer

```
import com.rabbitmq.client.Channel;
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.ConnectionFactory;
import java.nio.charset.StandardCharsets;

public class send {
    private final static String QUEUE_NAME = "ktmt";
    public static void main(String[] argv) throws Exception {
        ConnectionFactory factory = new ConnectionFactory();
        factory.setHost("localhost");
        factory.setUsername("guest");
        factory.setPassword("guest");
        try (Connection connection = factory.newConnection()) {
            Channel channel = connection.createChannel() {
                channel.queueDeclare(QUEUE_NAME, false, false, false, null);
                String message = "Hello World! Publish message to broker";
                channel.basicPublish("", QUEUE_NAME, null,
                    message.getBytes(StandardCharsets.UTF_8));
                System.out.println(" [x] Sent '" + message + "'");
            }
        }
    }
}
```

Sender.java

# Tutorial 1. Consumer

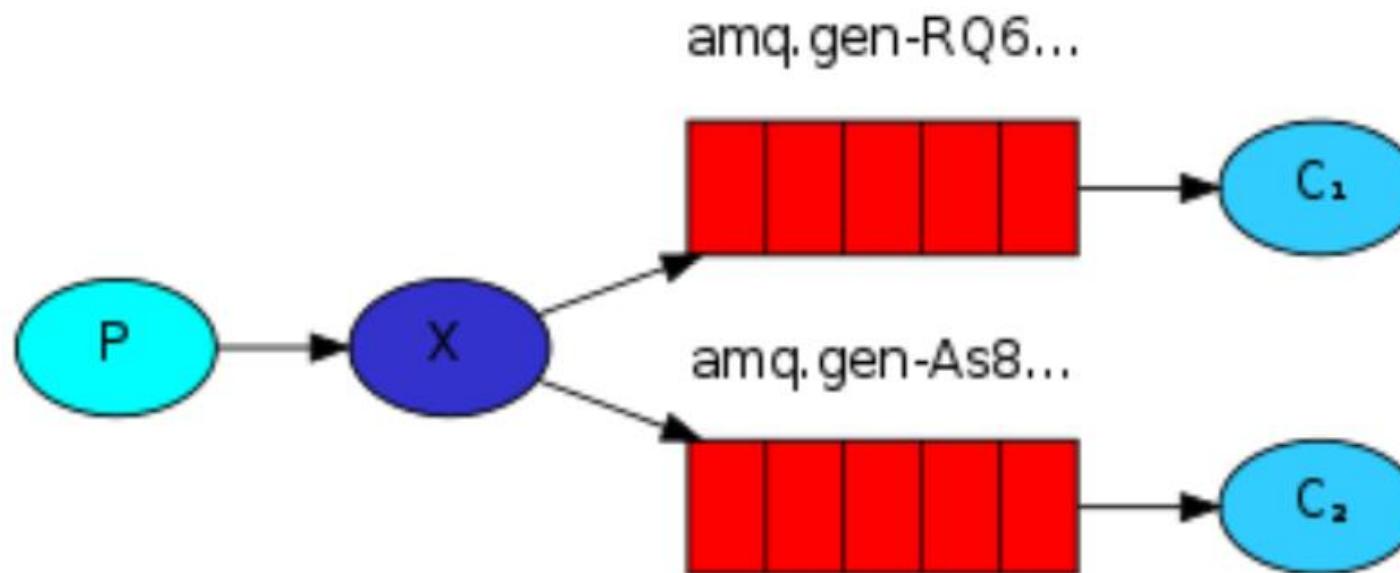
```
import com.rabbitmq.client.Channel;
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.ConnectionFactory;
import com.rabbitmq.client.DeliverCallback;
public class Recv {
    private final static String QUEUE_NAME = "ktmt";
    public static void main(String[] argv) throws Exception {
        ConnectionFactory factory = new ConnectionFactory();
        factory.setHost("localhost");
        factory.setUsername("guest");
        factory.setPassword("guest");
        Connection connection = factory.newConnection();
        Channel channel = connection.createChannel();
        channel.queueDeclare(QUEUE_NAME, false, false, false, null);
        System.out.println(" [*] Waiting for messages. To exit press CTRL+C");
        DeliverCallback deliverCallback = (consumerTag, delivery) -> {
            String message = new String(delivery.getBody(), "UTF-8");
            System.out.println(" [x] Received '" + message + "'");
        };
        channel.basicConsume(QUEUE_NAME, true, deliverCallback, consumerTag -> {
    });
}
}
```

**Recv.java**

# Tutorial 3

- Cải tiến Tutorial 1: một Producer gửi message đến tất cả các consumer
  - Sử dụng exchange type = “fanout” (broadcast)

```
channel.exchangeDeclare(EXCHANGE_NAME, "fanout");
```



# Tutorial 4

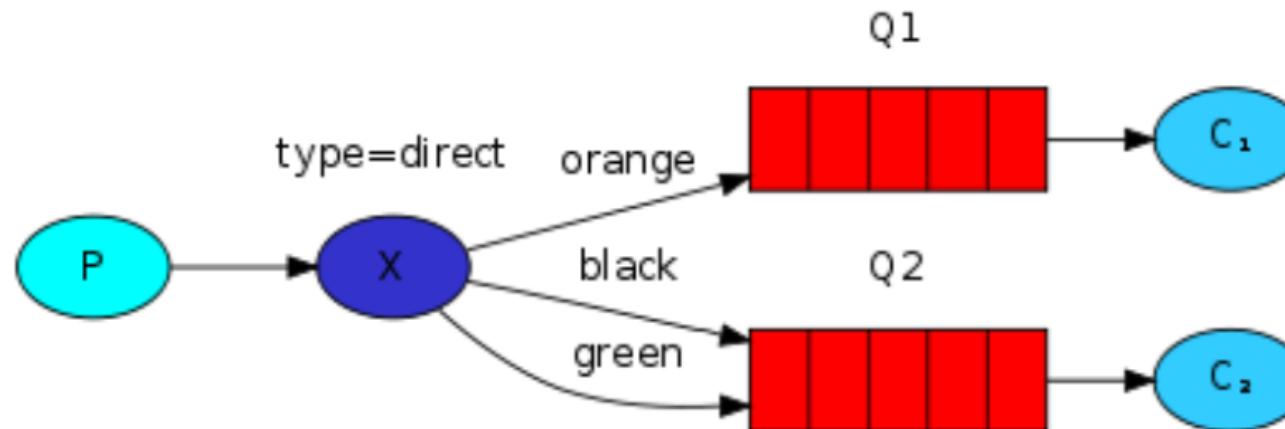
- Routing (định tuyến) thông điệp
- Chọn thông điệp muốn nhận (Receiving messages selectively)

Producer

```
channel.exchangeDeclare(EXCHANGE_NAME, "direct");
channel.basicPublish(EXCHANGE_NAME, "black", null, message.getBytes());
```

Consumer

```
channel.queueBind(queueName, EXCHANGE_NAME, "black");
```

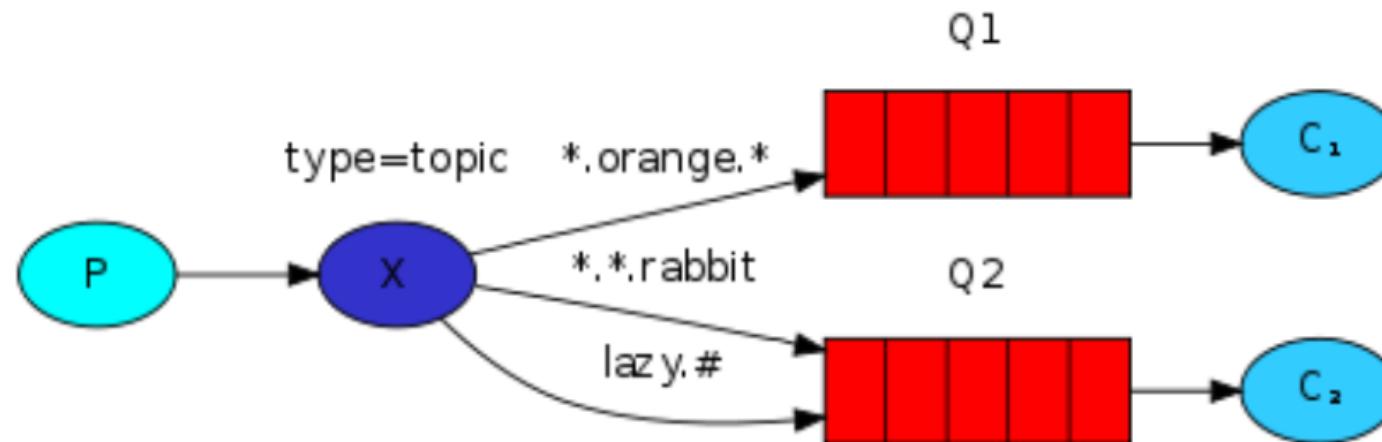


# Tutorial 5

- Sử dụng exchange type = “topic”
- routing\_key must be a list of words, delimited by dots.
  - \*(star) can substitute for exactly one word.
  - #(hash) can substitute for zero or more words.

Producer    `channel.basicPublish(EXCHANGE_NAME, routingKey, null, message.getBytes("UTF-8"));`

Consumer    `channel.queueBind(queueName, EXCHANGE_NAME, bindingKey);`



## Bài tập 3

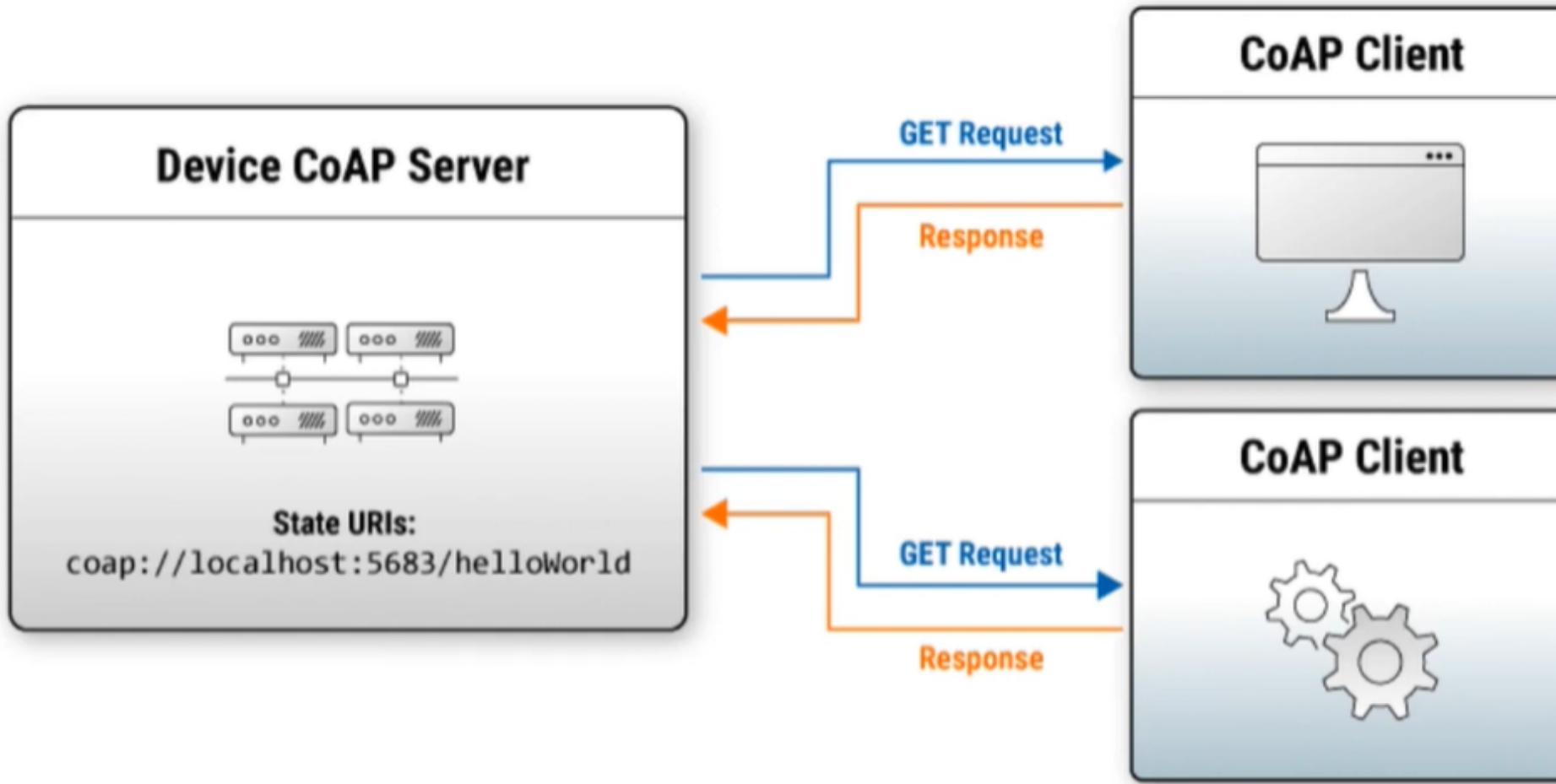
- Viết một chương trình gửi/nhận dữ liệu qua RabbitMQ broker:
  - Gửi (publish) dữ liệu lên RabbitMQ broker
  - Nhận (subscribe) dữ liệu từ RabbitMQ broker
  - Đóng gói dữ liệu bằng JSON. Ví dụ:

```
{"id":11, "packet_no":126, "temperature":30,  
"humidity":60, "tds":1100, "pH":5.0}
```



CoAP is the Constrained  
Application Protocol.

CoAP was designed for machine-to-machine (M2M) applications such as smart energy and building automation, supporting constrained devices and networks while cooperating with HTTP through simple proxies.



Server resources are discovered by sending a GET request to the /.well-known/core path.

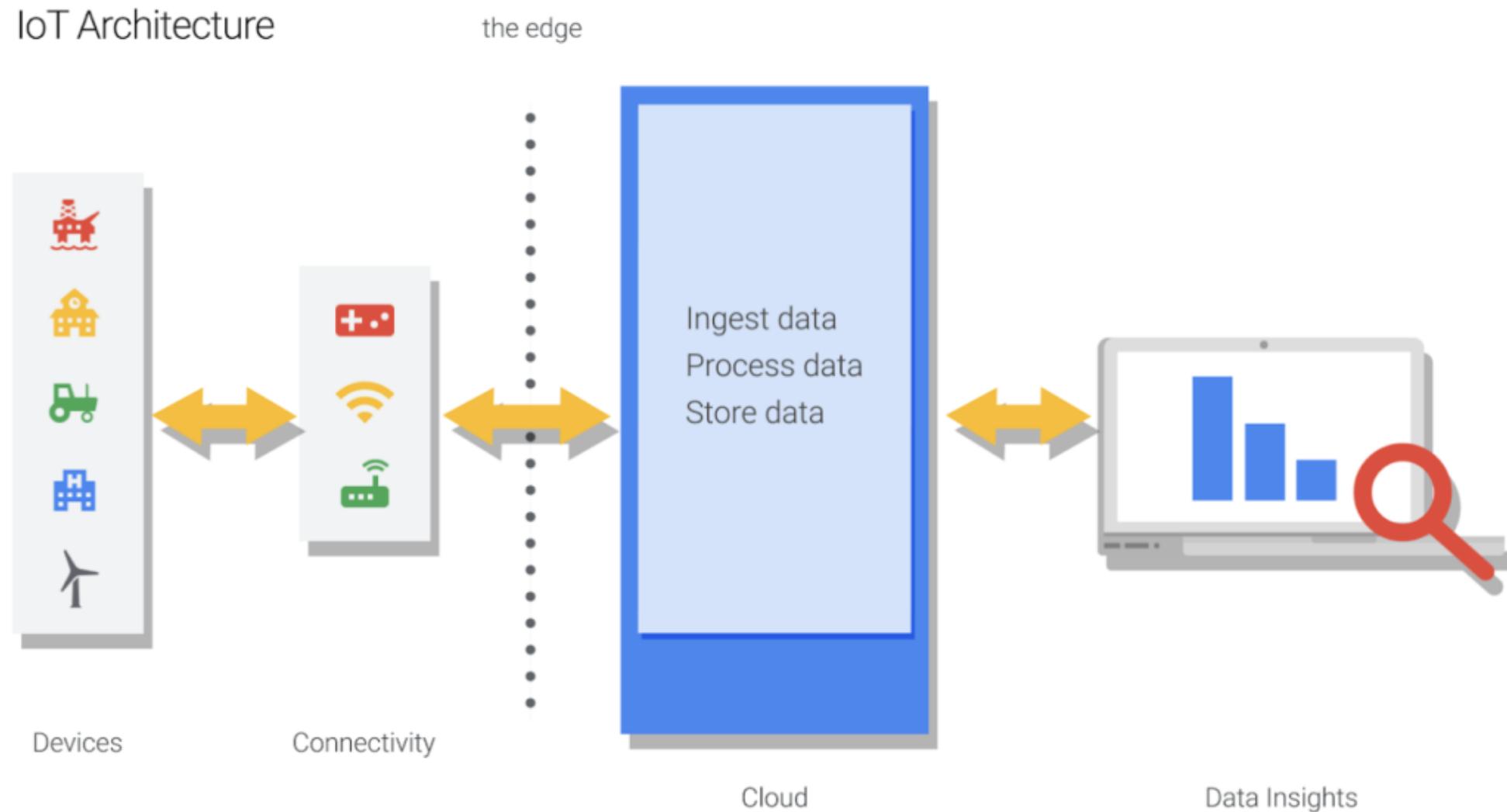
## 2.4. Nền tảng đám mây IoT (Cloud Platform)

- Các kiến trúc IoT phải có khả năng mở rộng (scaling) qui mô kết nối các thiết bị, thu thập dữ liệu (data gathering), xử lý (data processing) và lưu trữ dữ liệu (data storage)
- Khả năng thực hiện các công việc trên nhanh chóng trong khi vẫn thực hiện phân tích dữ liệu theo thời gian thực (real-time data insights)
- Gửi lượng lớn dữ liệu lên cloud làm chậm quá trình xử lý, đòi hỏi thêm băng thông cho truyền nhận dữ liệu

# IoT Cloud Platform

- Giải pháp tính toán phân tán (distributed computing) ~ **fog or edge computing** trở nên phổ biến
- Edge computing (tính toán cận biên): việc xử lý tính toán thực hiện trên các nodes trong mạng là các thiết bị IoT (IoT devices) nằm ở biên “edge” của mạng
- Giải pháp dẫn đến nhu cầu cho thiết bị IoT có khả năng: xử lý, phân tích dữ liệu cục bộ (cleaning, processing, analyzing data locally). Chỉ gửi dữ liệu đã xử lý trước đến cloud

# IoT Cloud Platform



# IoT Cloud Platforms

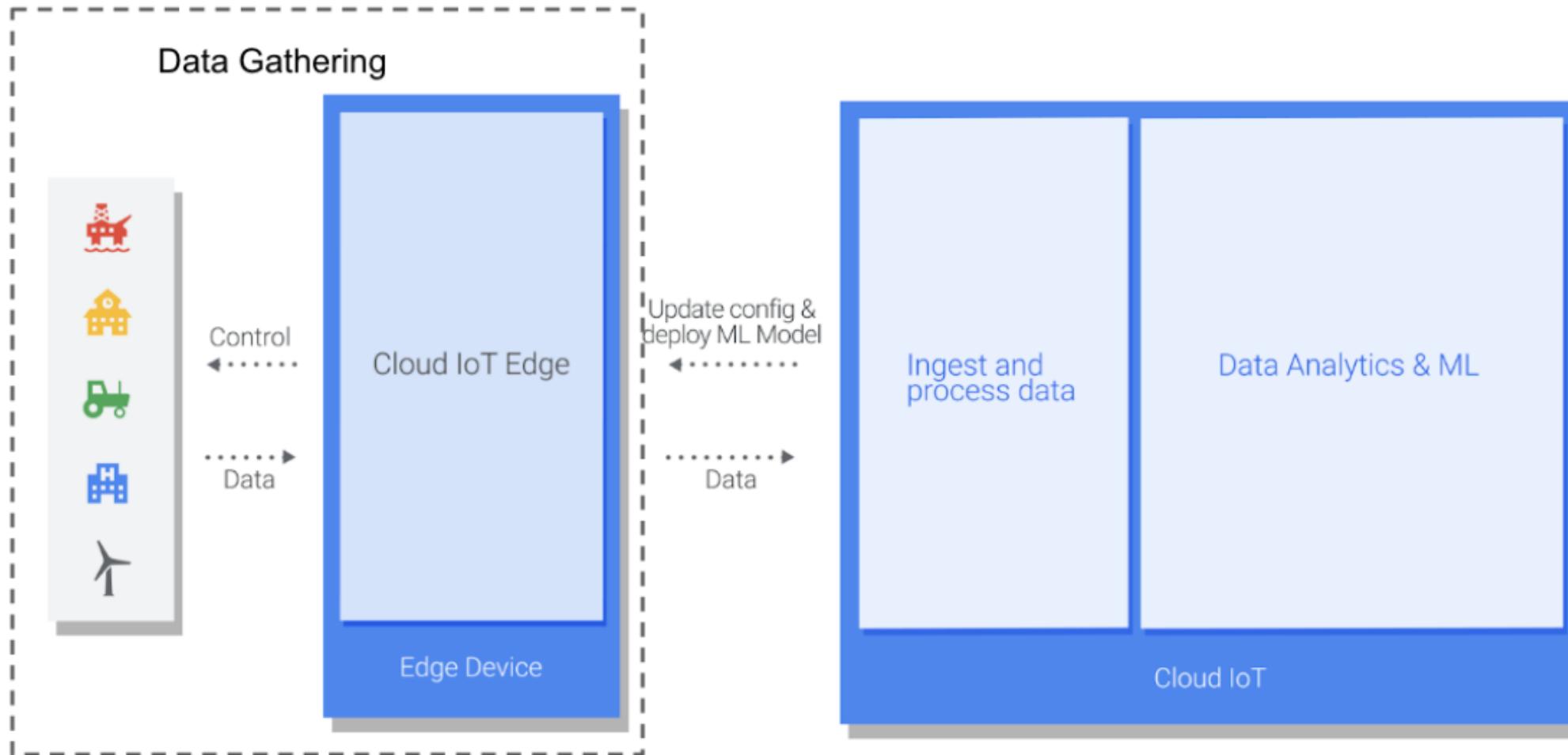
- Một số nền tảng dịch vụ IoT Cloud:
  - IBM Watson IoT
  - AWS IoT
  - Google Cloud IoT
  - Microsoft Azure IoT



<https://www.postscapes.com/internet-of-things-technologies/>

# Google Cloud IoT Architecture

Be capable of doing data import, process, storage, and analysis for hundreds of millions of events per hour from devices all over the world



# Google Cloud IoT

- Kiến trúc Google Cloud IoT chia làm 4 thành phần:
  - data gathering (thu thập dữ liệu trên thiết bị)
  - data ingest (tiếp nhận dữ liệu trên Cloud)
  - data processing (xử lý dữ liệu)
  - data analysis (phân tích dữ liệu)

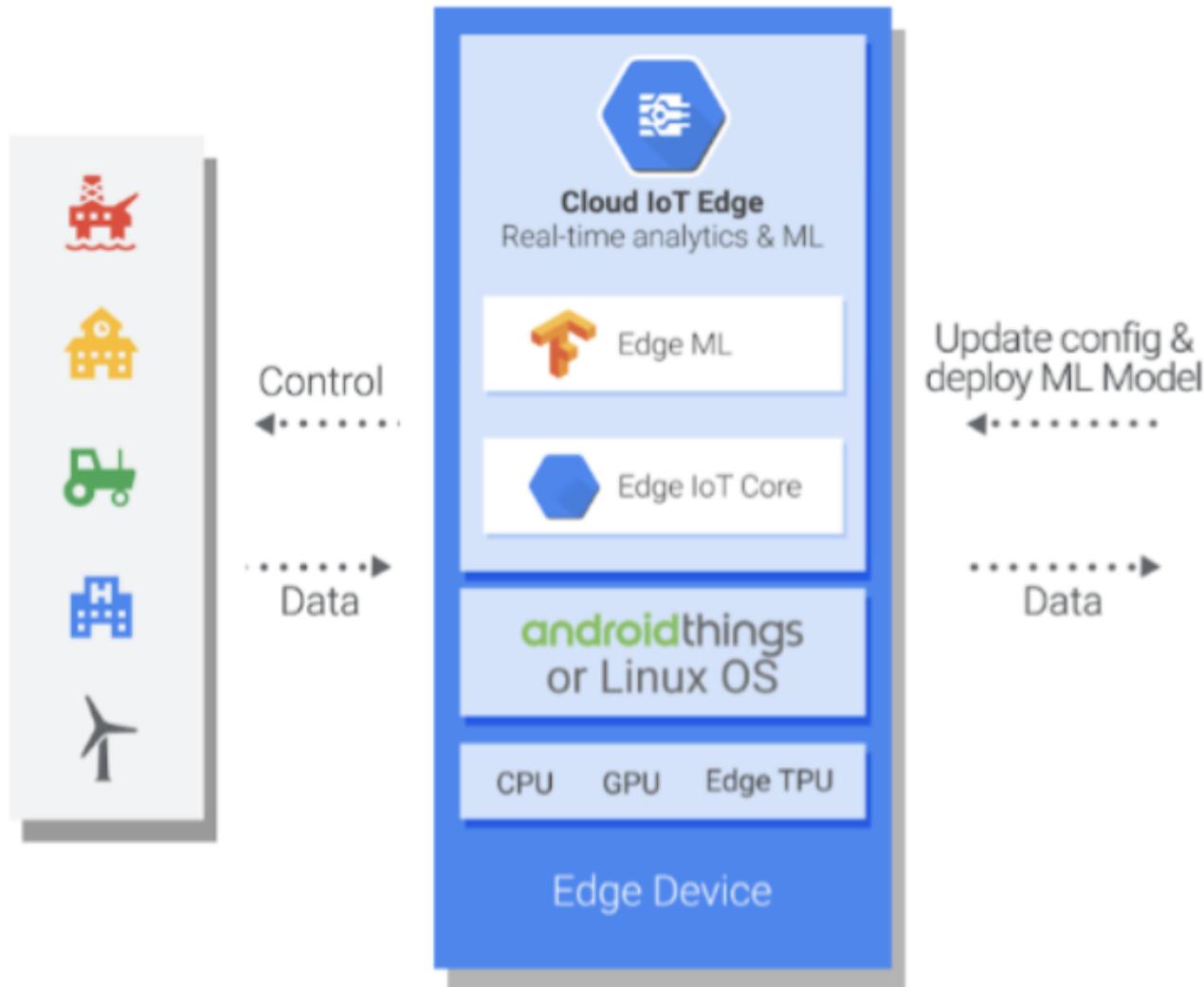
# Google Cloud IoT

- Data Gathering (Thu thập dữ liệu trên thiết bị):
  - Xảy ra trên các thiết bị (devices) và cảm biến (sensors)
  - Cảm biến (sensors) thu thập (gather) dữ liệu từ môi trường, gửi lên cloud hoặc trực tiếp hoặc gián tiếp thông qua thiết bị trung gian
  - Thiết bị (devices) chuẩn bị dữ liệu truyền lên cloud. Tùy thuộc loại mạng kết nối, quá trình chuẩn bị bao gồm: làm sạch dữ liệu, tiền xử lý, phân tích hoặc có thể sử dụng cả machine learning



# Google Cloud IoT

- Cloud IoT Edge:

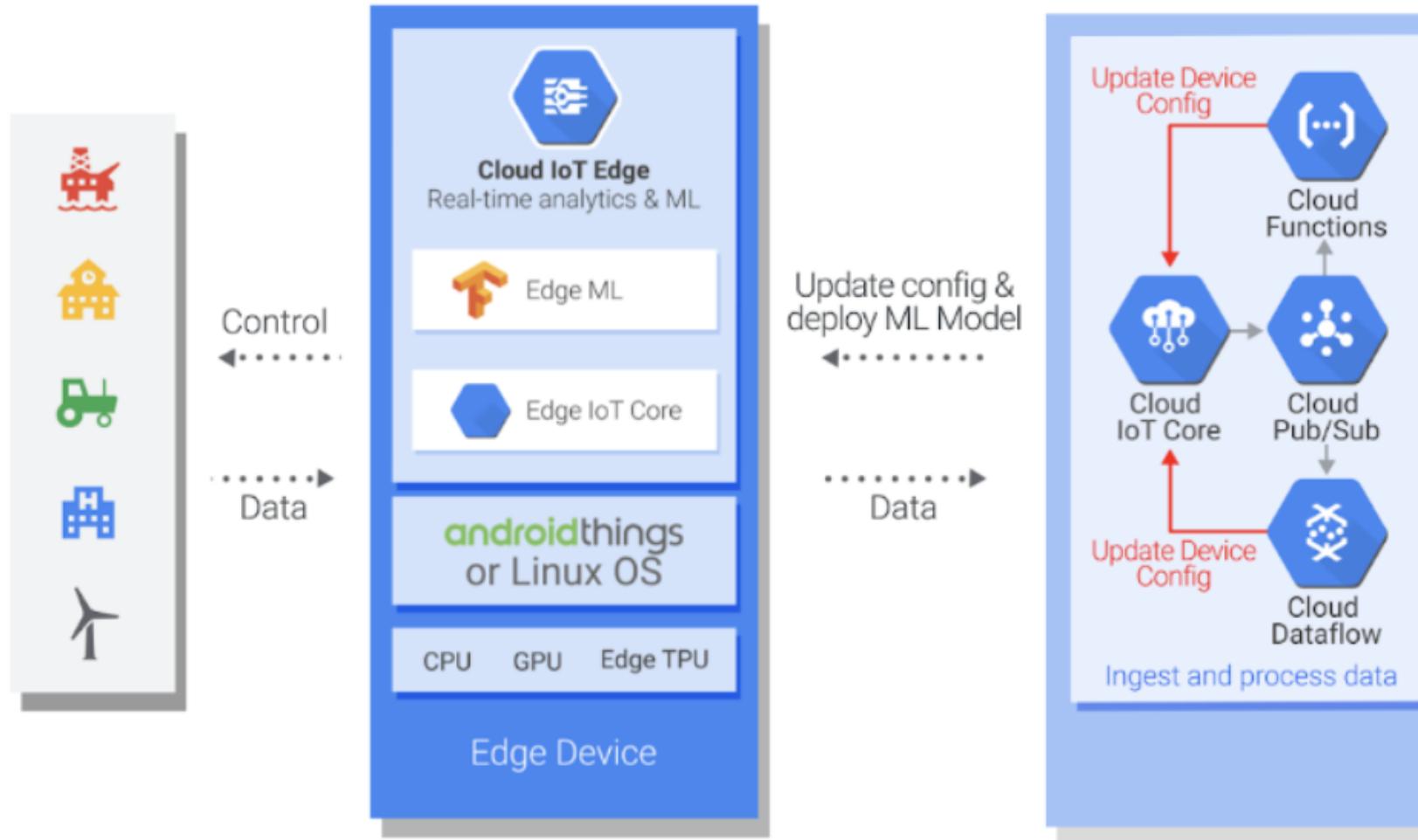


# Google Cloud IoT

- Cloud IoT Edge:
  - Trong kiến trúc Goolge Cloud IoT, giai đoạn thu thập dữ liệu có thể thực hiện trên Cloud IoT Edge.
  - Thành phần này là một nhóm các thiết bị có khả năng thực hiện các phân tích thời gian thực và ML (machine learning). Cho phép mở rộng các xử lý dữ liệu và ML thực hiện trên các thiết bị (edge devices).
  - Cloud IoT Edge có thể sử dụng Android Things OS, Linux-based OS.
  - Gồm 2 thành phần: Edge IoT Core và Edge ML

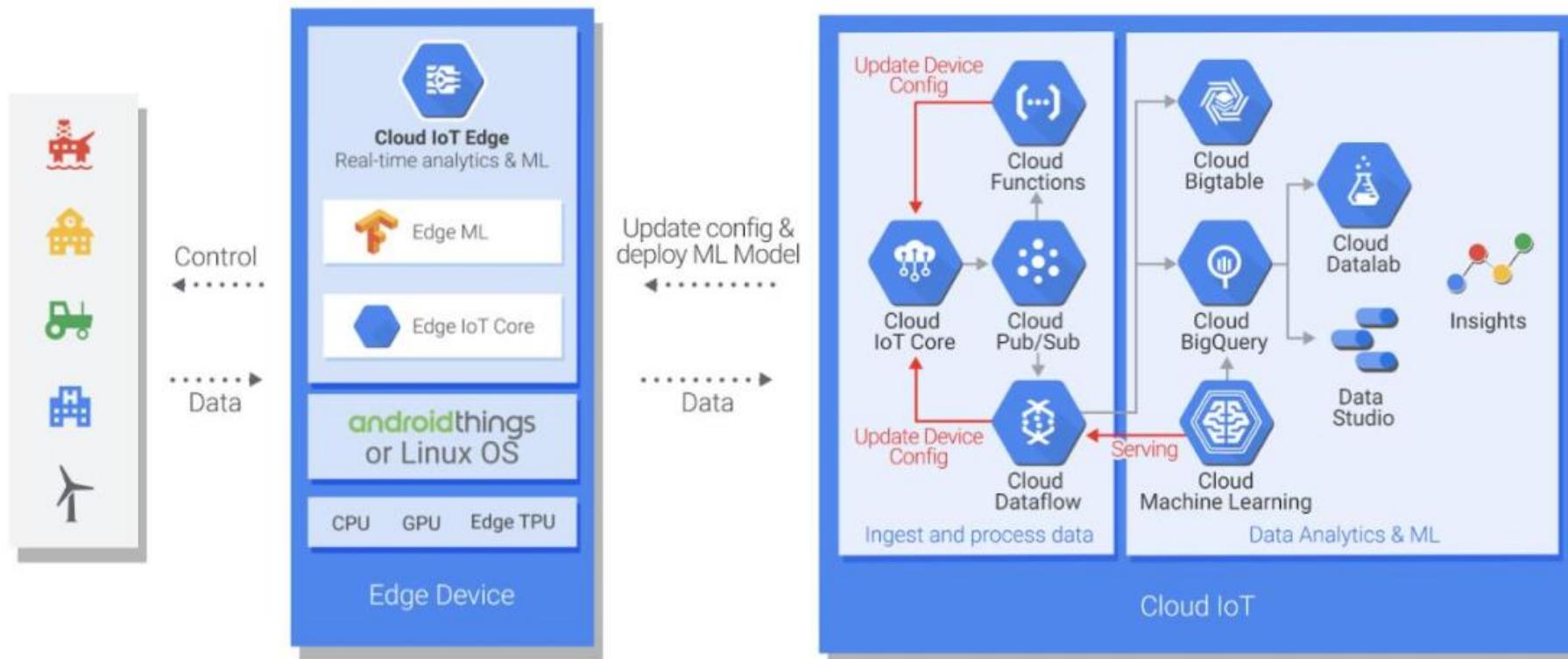
# Google Cloud IoT

- Ingest and Process data (Tiếp nhận và xử lý dữ liệu):



# Google Cloud IoT

- Data Analytics and ML (Phân tích dữ liệu và Học máy):
  - Có thể thực hiện trên Edge hoặc Cloud.
  - Google's Cloud IoT Core Data Analytics and ML are fully integrated with IoT data.





# Hết chương 2