Persistent Messages – Last Will Testament (LWT)

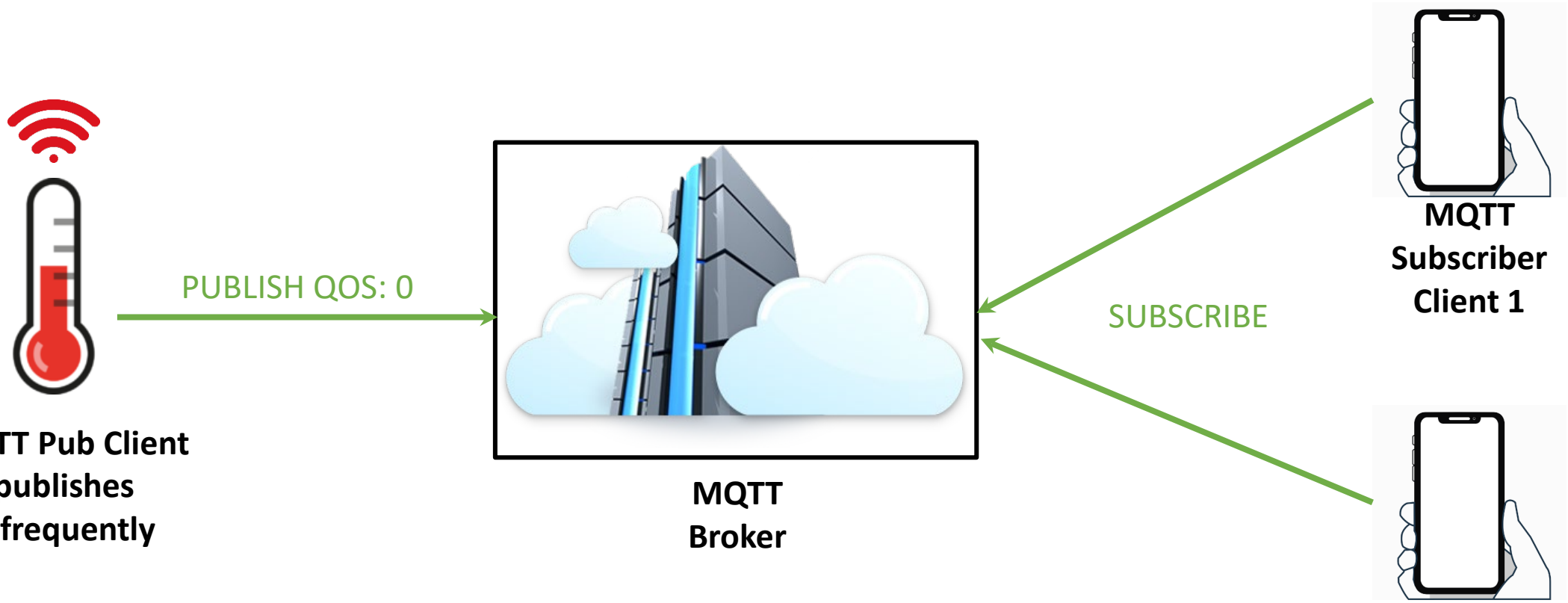
Dr. Binil Starly

School of Manufacturing Systems & Networks

Ira A. Fulton Schools of Engineering

Arizona State University

Retain Message



How will new Subscriber Client 2 get last message particularly if MQTT Publish Client sends message in large time intervals

Persistent Messages

- **Non-persistent (default)**: Messages are not stored on the broker and are lost if the network fails. This mode is appropriate when messages are not critical and can be easily regenerated.

- **Queued persistent**: Messages are stored on the broker until they are delivered to the subscriber. If the subscriber is not available, messages are queued until the subscriber reconnects. Queued persistence is useful when the subscriber is not always connected to the network, or if the subscriber needs to receive all messages, even if they are sent when the subscriber is offline.

- **Persistent with acknowledgment**: Messages are sent to the subscriber until an acknowledgement is sent to the broker. This mode provides the highest level of message persistence. This mode is useful when it is critical to ensure that messages are received and processed by the subscriber.

How to implement Persistent Messages

```
mqttBroker ="broker.hivemq.com"

client = mqtt.Client("Machine1", clean_session=False)
client.connect(mqttBroker)
…
…
…
client.publish("Binil/Machine1/SensorData", msg, qos=1)
```

```
Client Id = “Machine1” must be unique at the Broker level
Qos=1 or 2
```

Last Will Testament (LWT)

Purpose:

Ensures reliable communication by notifying other clients when a client disconnects unexpectedly.

Mechanism:

Will Message: A pre-defined message set by the client that is published by the broker if the client disconnects unexpectedly.

Will Topic: The topic to which the will message will be published.

Will QoS (Quality of Service): Ensures the will message is delivered with the specified reliability level.

Will Retain Flag: Determines if the will message should be retained by the broker and delivered to future subscribers of the will topic.

Last Will Testament (LWT)

Configuration:

Set during the client connection process:

- `client.connect(broker, port, keepalive, will_message)`
- Example: `client.will_set("sensors/lastwill", "Sensor Offline", qos=1, retain=True)`

The broker stores this message until it detects an ungraceful disconnect from the client. Upon detecting the disconnection, the broker broadcasts the last will message to all subscribed clients of the corresponding topic.

Benefits:

- Enhanced Reliability: Notifies subscribers of an unexpected disconnection, enabling timely actions.
- Improved Network Management: Helps in monitoring and maintaining the health of the network and its nodes.
- Simplified Troubleshooting: Quickly identifies and responds to client failures.

Last Will Testament (LWT)

When does it get sent:

I/O error or network failure:
Failed communication within Keep Alive period: If the client fails to communicate with the broker within the specified Keep Alive period, the LWT message is sent.

Client closes connection without DISCONNECT: When the client terminates the network connection without sending a DISCONNECT packet, the broker ensures the LWT message is distributed.

Broker closes connection due to protocol error: If the broker closes the network connection due to a protocol error, it will send the LWT message.

Persistent & LWT implementation in Code

Begin Python Code Example:

Run
Pub_LWT_Demo.py

Sub_LWT_Demo.py