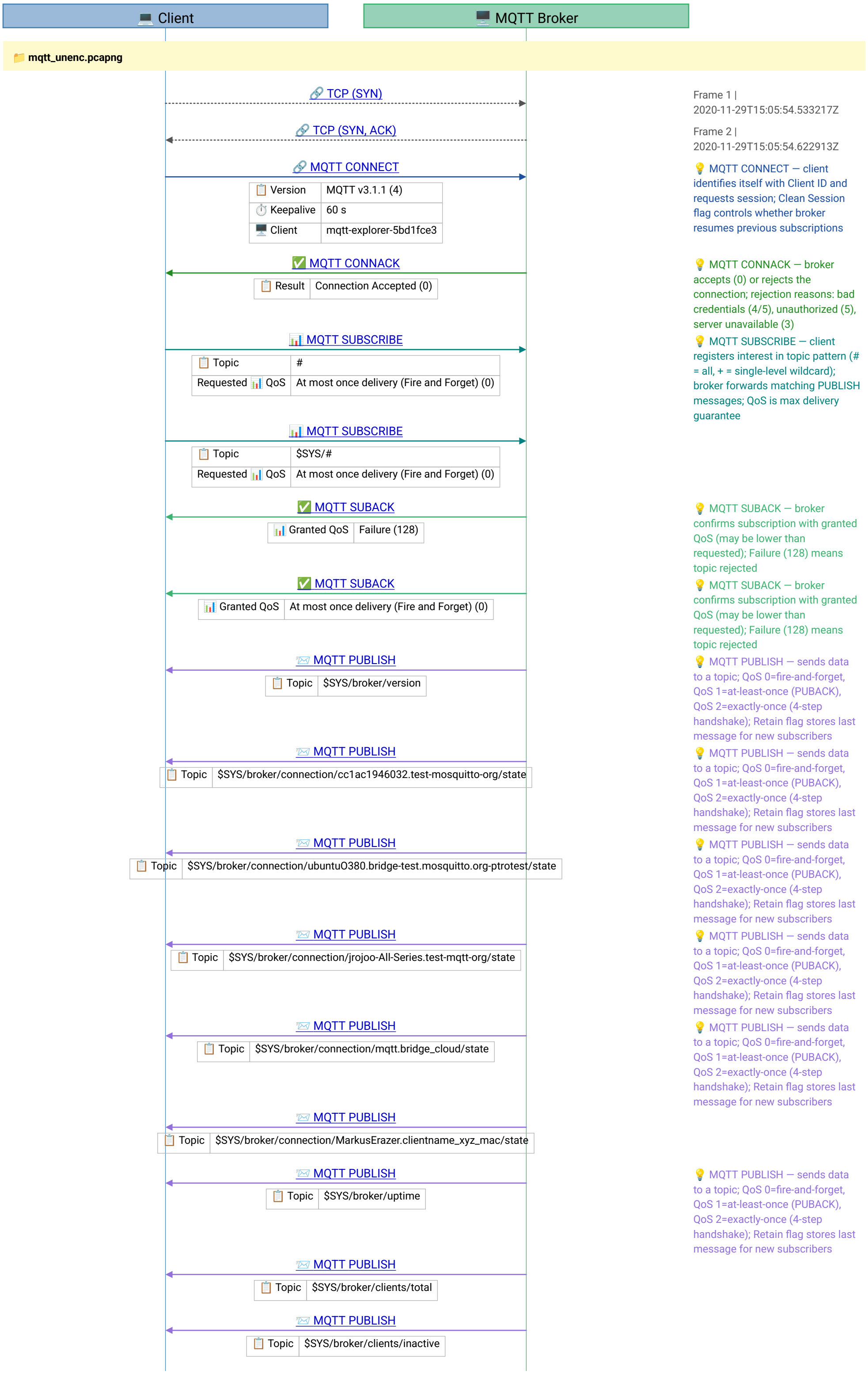
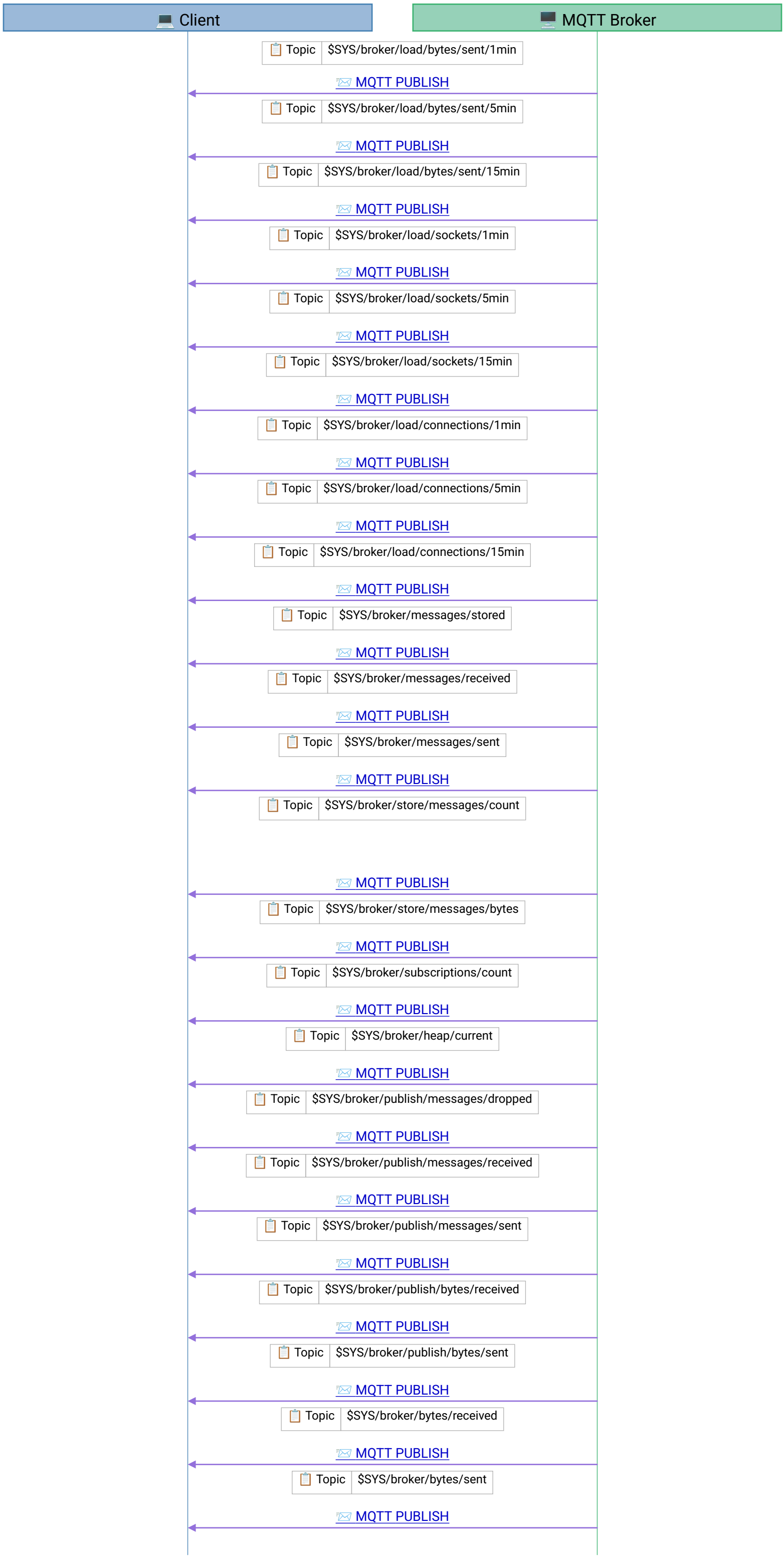

💻 Client
🖥️ MQTT Broker
📁 mqtt_unenc.pcapng
🔗 TCP (SYN)
Frame 1 | 2020-11-29T15:05:54.533217Z
🔗 TCP (SYN, ACK)
Frame 2 | 2020-11-29T15:05:54.622913Z
🔗 MQTT CONNECT
📋 Version | MQTT v3.1.1 (4)
⏱️ Keepalive | 60 s
🖥️ Client | mqtt-explorer-5bd1fce3
💡 MQTT CONNECT — client identifies itself with Client ID and requests session; Clean Session flag controls whether broker resumes previous subscriptions
✅ MQTT CONNACK
📋 Result | Connection Accepted (0)
💡 MQTT CONNACK — broker accepts (0) or rejects the connection; rejection reasons: bad credentials (4/5), unauthorized (5), server unavailable (3)
📊 MQTT SUBSCRIBE
📋 Topic | #
Requested 📊 QoS | At most once delivery (Fire and Forget) (0)
💡 MQTT SUBSCRIBE — client registers interest in topic pattern (# = all, + = single-level wildcard); broker forwards matching PUBLISH messages; QoS is max delivery guarantee
📊 MQTT SUBSCRIBE
📋 Topic | $SYS/#
Requested 📊 QoS | At most once delivery (Fire and Forget) (0)
✅ MQTT SUBACK
📊 Granted QoS | Failure (128)
💡 MQTT SUBACK — broker confirms subscription with granted QoS (may be lower than requested); Failure (128) means topic rejected
✅ MQTT SUBACK
📊 Granted QoS | At most once delivery (Fire and Forget) (0)
💡 MQTT SUBACK — broker confirms subscription with granted QoS (may be lower than requested); Failure (128) means topic rejected
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/version
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/connection/cc1ac1946032.test-mosquitto-org/state
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/connection/ubuntuO380.bridge-test.mosquitto.org-ptrotest/state
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/connection/jrojoo-All-Series.test-mqtt-org/state
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/connection/mqtt.bridge_cloud/state
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/connection/MarkusErazer.clientname_xyz_mac/state
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/uptime
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/clients/total
📨 MQTT PUBLISH
📋 Topic | $SYS/broker/clients/inactive

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

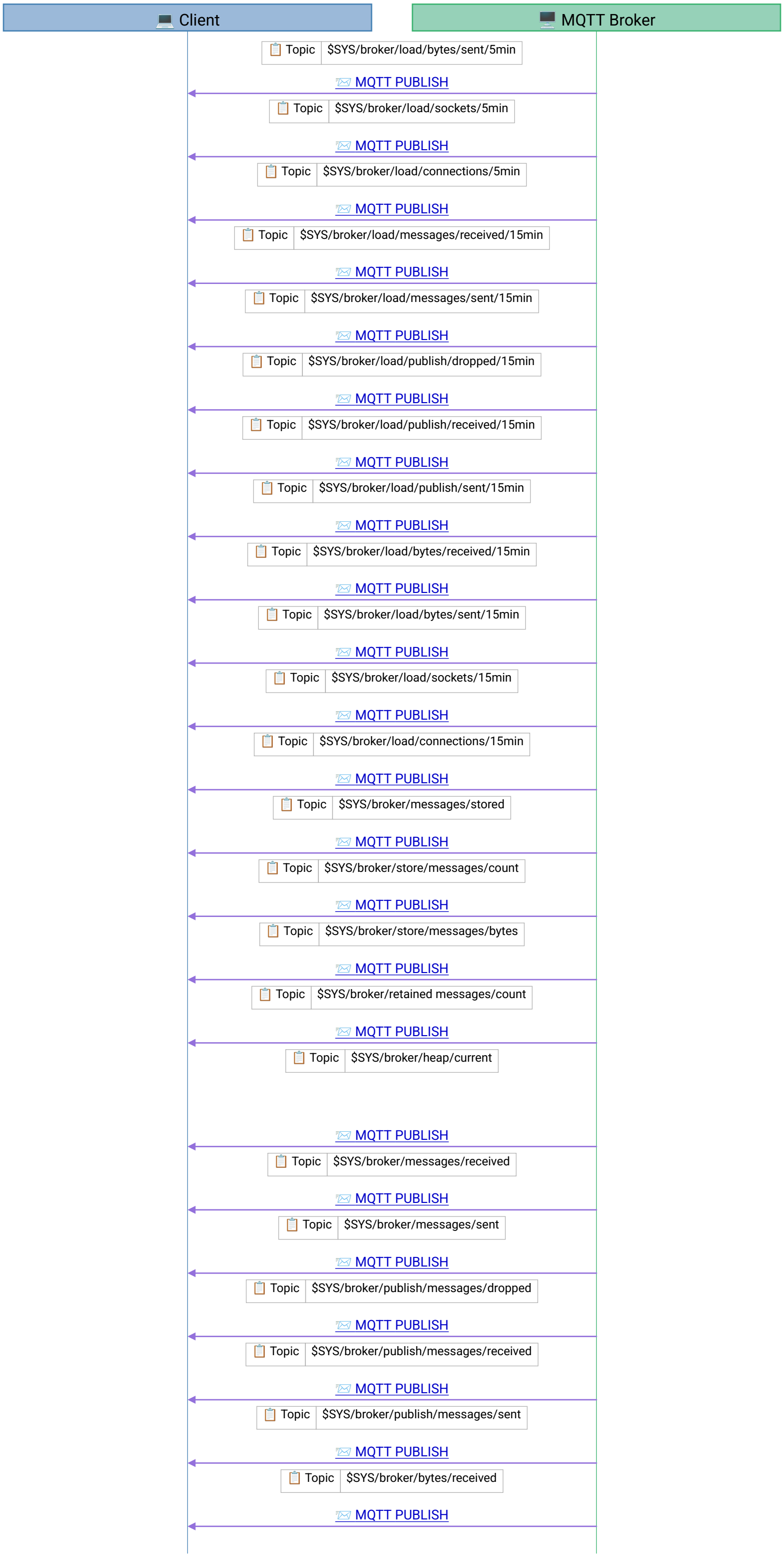
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers



💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget,


💻 Client
🖥️ MQTT Broker
📋 Topic $SYS/broker/uptime
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/clients/total
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/clients/inactive
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/clients/disconnected
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/clients/active
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/clients/connected
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/messages/received/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/messages/sent/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/publish/dropped/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/publish/received/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/publish/sent/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/bytes/received/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/bytes/sent/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/sockets/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/connections/1min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/messages/received/5min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/messages/sent/5min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/publish/dropped/5min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/publish/received/5min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/publish/sent/5min
✉️ MQTT PUBLISH
📋 Topic $SYS/broker/load/bytes/received/5min
✉️ MQTT PUBLISH
QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers
💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers



💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

💻 Client	🖥️ MQTT Broker

📋 Topic | $SYS/broker/bytes/sent

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/publish/bytes/received

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/publish/bytes/sent

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/connection/cc1ac1946032.test-mosquitto-org/state

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client → MQTT Broker)

📋 Topic | $SYS/broker/messages

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/connection/cc1ac1946032.test-mosquitto-org/state

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client → MQTT Broker)

📋 Topic | $SYS/broker/messages

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/connection/cc1ac1946032.test-mosquitto-org/state

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/uptime

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/clients/total

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/clients/inactive

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/clients/disconnected

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/clients/active

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/clients/connected

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/load/messages/received/1min

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/load/messages/sent/1min

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/load/publish/dropped/1min

📨 MQTT PUBLISH (Client ← MQTT Broker)

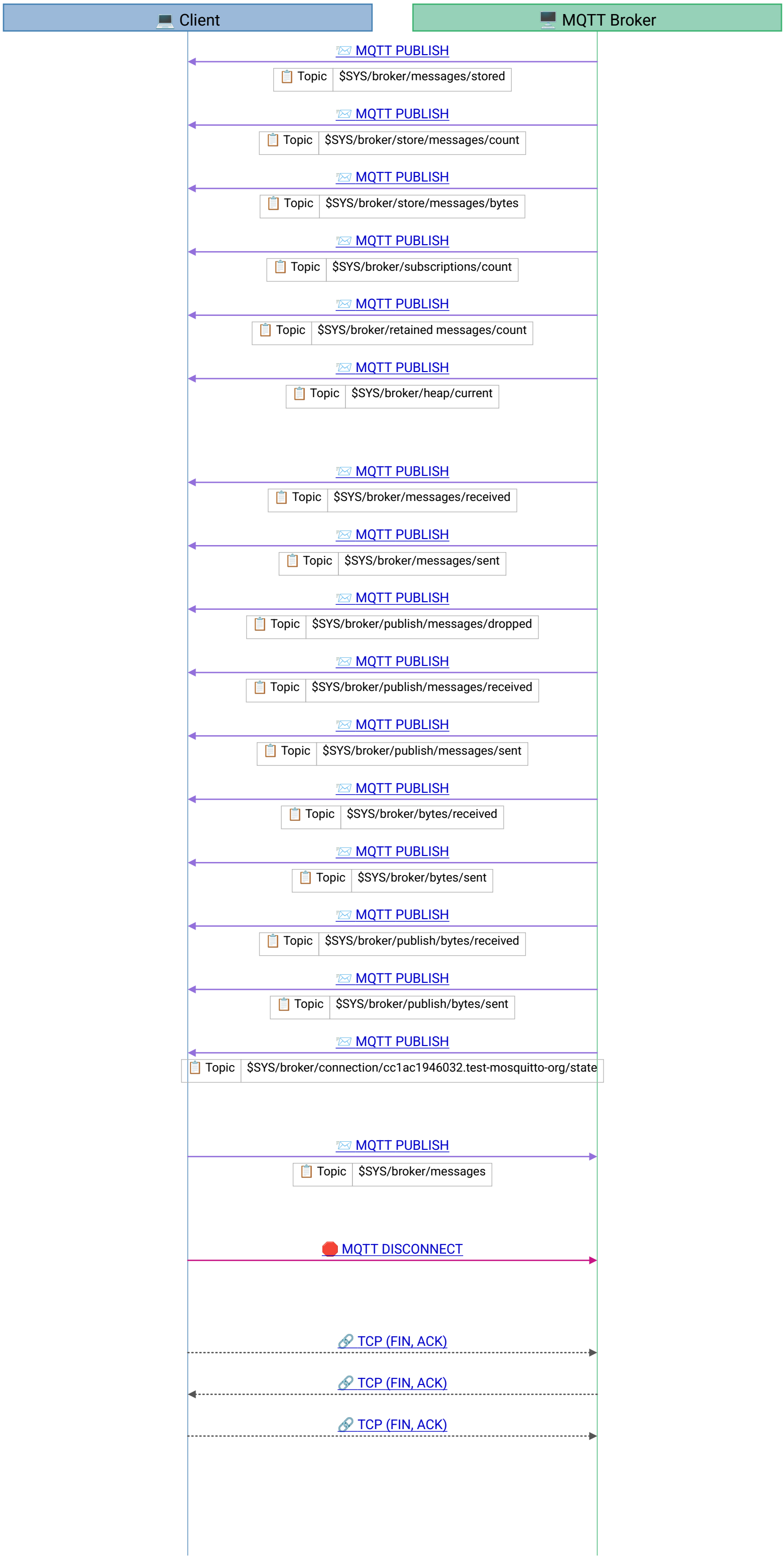
📋 Topic | $SYS/broker/load/publish/received/1min

📨 MQTT PUBLISH (Client ← MQTT Broker)

📋 Topic | $SYS/broker/load/publish/sent/1min

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers



💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

💡 MQTT PUBLISH — sends data to a topic; QoS 0=fire-and-forget, QoS 1=at-least-once (PUBACK), QoS 2=exactly-once (4-step handshake); Retain flag stores last message for new subscribers

💡 MQTT DISCONNECT — graceful session end; broker discards Will Message; without DISCONNECT (ungraceful), broker publishes the Will Message to notify subscribers

Frame 70 | 2020-11-29T15:06:14.647075Z

Frame 72 | 2020-11-29T15:06:14.820844Z

Frame 74 | 2020-11-29T15:06:14.820914Z