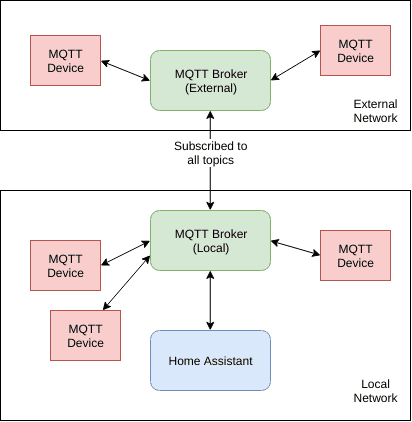
PROJECT REPORT

SETUP FOR THE CLIENT AND SERVER

Block Diagram Design



CLIENT

sudo apt-get install python3-pip

pip3 install paho-mqtt      # If not Working Follow below Three Steps:

        OR

    tar -xvzf paho-mqtt-1.4.0.tar.gz

    chmod +x setup.py

    sudo python3 setup.py install

sudo apt-get install python3-tk

To run broker:

mosquitto -p 8008

SERVER

sudo apt-get install python3-pip

pip3 install paho-mqtt      # If not Working Follow below Three Steps:

        OR

    tar -xvzf paho-mqtt-1.4.0.tar.gz

    chmod +x setup.py

    sudo python3 setup.py install

sudo apt-get install python3-tk

To run broker:

mosquitto -p 8008

HOW TO PRODUCE THE SAME RESULTS

# Running the software

A python requirements install alternativ

INSTALL

sudo pip install -r requirements.txt && sudo pacman -S mosquitto

USAGE

./get.py <ip> <topic>

This will start the broker and listen for incoming messages.

./send.py <ip> <topic> <message>

This wil send a message.

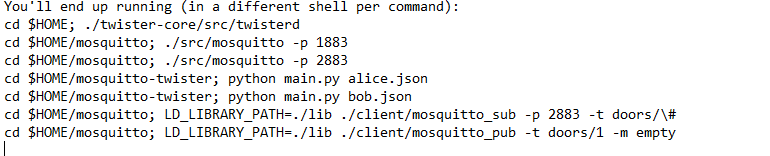
Here is the parts involved, from end-to-end:

- A first MQTT client (1), that will pulish a message, using...

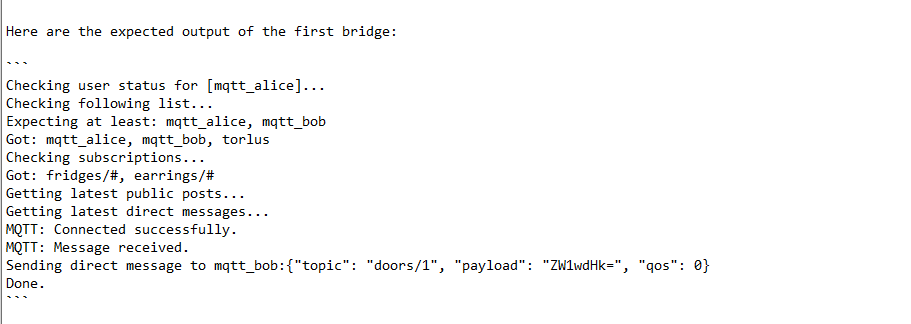
- A first MQTT broker (2), to which will be also connected...

- A first instance of the mosquitto-twister bridge, associated to the

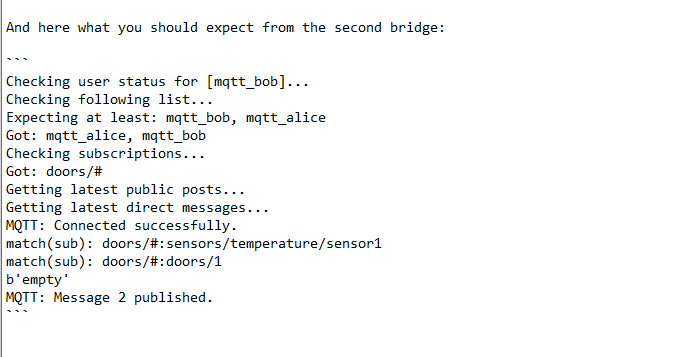
first Twister user, that will relay the message through the Twister network through...



Here are the expected output of the first bridge:



And here what you should expect from the second bridge:



CODE FOR THE PROGRAM  
Pub.py

#! /usr/bin/python3

import os                             # to execute the system call

from tkinter import \*                 # it's a packkage for GUI

import paho.mqtt.client as mqtt       # package for mqtt

from threading import Thread          # for multiple threading

import time

UPSTREAM\_TOPIC="test/upstream"        # which subscriber will subscibe

DOWNSTREAM\_TOPIC="test/downstream"    # which publisher will subscribe

QOS=0                                 # flag

KEEPALIVE=60

BROKER="127.0.0.1"                    # mosquitto -p 8008 because we are using in local host

PORT=8008

Name = input("Enter Your Name: ")

window = Tk()                       # all three window are used to make GUI window

window.geometry('400x300')

window.title("%s's Chat" % Name)

lbl = Label(window, text = "WELCOME TO MY APP")

#lbl.grid(column=0, row=0)

lbl.pack(side=TOP, fill=X)

messages\_frame = Frame(window)

messages\_frame.pack(side=TOP, fill=X)

scrollbar = Scrollbar(window)  # To navigate through past messages.

def select\_text\_or\_copy\_text(e):

    e.widget.delete(0, 'end')

def send(event):

    payload= Name + ": " + my\_msg.get()

    time.sleep(0.5)

    client.publish(UPSTREAM\_TOPIC, payload, qos=QOS, retain=False)

    msg\_list.insert(END, payload)

# To Read Msg in Text Form

my\_msg=StringVar()

my\_msg.set("Type your message here")

btn\_fr = Frame(window)

btn\_fr.pack(side=TOP, fill=X)

btn = Button(btn\_fr, text="send", fg="green", bg="black", width=5)

btn.bind('<Button-1>', send)

btn.pack(anchor=NE, padx=5)

msg\_list = Listbox(window, height=10, width=40, yscrollcommand=scrollbar.set)

scrollbar.pack(side=RIGHT, fill=Y)

msg\_list.pack(anchor=CENTER, fill=Y, expand=True)

lbl1 = Label(messages\_frame, text="Message Box", width=11)

lbl1.pack(side=LEFT, padx=0, pady=0)

entry\_field = Entry(messages\_frame, textvariable=my\_msg, width=25)

entry\_field.bind("<Return>", send)

entry\_field.bind('<Delete>', select\_text\_or\_copy\_text)

entry\_field.focus()

entry\_field.pack(side=TOP, padx=5, fill=X, expand=True)

def on\_connect(client, userdata, flags, rc):

    msg\_list.insert(END, "%s : Connected Successfully" % Name)

    client.subscribe(DOWNSTREAM\_TOPIC)

def on\_disconnect(client, userdata, rc):

    msg\_list.insert(END, "%s : Disconnected from Chat" % Name)

def on\_subscribe(client, userdata, mid, granted\_qos):

    print("Successfully subscribed Userdata: %s : granted\_qos: %s" % (str(userdata), str(granted\_qos)))

def on\_publish(client, userdata, mid):

    print("Successfully published Message %s: mid: %s" % (str(userdata), str(mid)))

def on\_message(client, userdata, msg):

    msg.payload = msg.payload.decode("utf-8")

    msg\_list.insert(END, msg.payload)

    print("Got Message: Topic: %s Msg: %s" % (str(msg.topic),str(msg.payload)))

def close\_window():

    client.disconnect()

    window.quit()        # inbuild function

def handle\_reply():

    client.loop\_forever()

client = mqtt.Client()

client.on\_connect = on\_connect

client.on\_message = on\_message

client.on\_publish = on\_publish

client.on\_subscribe = on\_subscribe

client.connect(BROKER, PORT, KEEPALIVE)   # this is an inbuild function

window.protocol("WM\_DELETE\_WINDOW", close\_window)

receive\_thread = Thread(target=handle\_reply)   # it is used for forever looping

receive\_thread.start()       # variable to start the thread

window.mainloop()

#window.after(1000, connect\_user)

#while True:

#   window.update\_idletasks()

#   window.update()

Sub.py

#! /usr/bin/python3

import os

from tkinter import \*

import paho.mqtt.client as mqtt

from threading import Thread

import time

UPSTREAM\_TOPIC="test/upstream"

DOWNSTREAM\_TOPIC="test/downstream"

QOS=0

KEEPALIVE=60

BROKER="127.0.0.1"

PORT=8008

Name = input("Enter Your Name: ")

window = Tk()

window.geometry('400x300')

window.title("%s's Chat" % Name)

lbl = Label(window, text = "WELCOME TO MY APP")

#lbl.grid(column=0, row=0)

lbl.pack(side=TOP, fill=X)

messages\_frame = Frame(window)

messages\_frame.pack(side=TOP, fill=X)

scrollbar = Scrollbar(window)  # To navigate through past messages.

def select\_text\_or\_copy\_text(e):

    e.widget.delete(0, 'end')

def send(event):

    payload= Name + ": " + my\_msg.get()

    time.sleep(0.5)

    client.publish(DOWNSTREAM\_TOPIC, payload, qos=QOS, retain=False)

    msg\_list.insert(END, payload)

# To Read Msg in Text Form

my\_msg=StringVar()

my\_msg.set("Type your message here")

btn\_fr = Frame(window)

btn\_fr.pack(side=TOP, fill=X)

btn = Button(btn\_fr, text="send", fg="green", bg="black", width=5)

btn.bind('<Button-1>', send)

btn.pack(anchor=NE, padx=5)

msg\_list = Listbox(window, height=10, width=40, yscrollcommand=scrollbar.set)

scrollbar.pack(side=RIGHT, fill=Y)

msg\_list.pack(anchor=CENTER, fill=Y, expand=True)

lbl1 = Label(messages\_frame, text="Message Box", width=11)

lbl1.pack(side=LEFT, padx=0, pady=0)

entry\_field = Entry(messages\_frame, textvariable=my\_msg, width=25)

entry\_field.bind("<Return>", send)

entry\_field.bind('<Delete>', select\_text\_or\_copy\_text)

entry\_field.focus()

entry\_field.pack(side=TOP, padx=5, fill=X, expand=True)

def on\_connect(client, userdata, flags, rc):

    msg\_list.insert(END, "%s : Connected Successfully" % Name)

    client.subscribe(UPSTREAM\_TOPIC)

def on\_disconnect(client, userdata, rc):

    msg\_list.insert(END, "%s : Disconnected from Chat" % Name)

def on\_subscribe(client, userdata, mid, granted\_qos):

    print("Successfully subscribed Userdata: %s : granted\_qos: %s" % (str(userdata), str(granted\_qos)))

def on\_publish(client, userdata, mid):

    print("Successfully published Message %s: mid: %s" % (str(userdata), str(mid)))

def on\_message(client, userdata, msg):

    msg.payload = msg.payload.decode("utf-8")

    msg\_list.insert(END, msg.payload)

    print("Got Message: Topic: %s Msg: %s" % (str(msg.topic),str(msg.payload)))

def close\_window():

    client.disconnect()

    window.quit()

def handle\_reply():

    client.loop\_forever()

client = mqtt.Client()

client.on\_connect = on\_connect

client.on\_message = on\_message

client.on\_publish = on\_publish

client.on\_subscribe = on\_subscribe

client.connect(BROKER, PORT, KEEPALIVE)

window.protocol("WM\_DELETE\_WINDOW", close\_window)

receive\_thread = Thread(target=handle\_reply)

receive\_thread.start()

window.mainloop()

#window.after(1000, connect\_user)

#while True:

#   window.update\_idletasks()

#   window.update()

REFERENCES

1. L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” *Comput. Netw*, vol. 54, no. 15, Oct. 2010, doi: 10.1016/j.comnet.2010.05.010.
2. I. Yaqoob *et al.*, “Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges,” *IEEE Wirel. Commun.*, vol. 24, no. 3, pp. 10–16, Jun. 2017, doi: 10.1109/MWC.2017.1600421.
3. F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, “Internet of Things security: A survey,” *J. Netw. Comput. Appl.*, vol. 88, pp. 10–28, 2017, doi: 10.1016/j.jnca.2017.04.002.
4. M. B. Yassein, M. Q. Shatnawi, and D. Al-zoubi, “Application layer protocols for the Internet of Things: A survey,” in *2016 International Conference on Engineering & MIS (ICEMIS)*, Agadir, Morocco, Sep. 2016, pp. 1–4, doi: 10.1109/ICEMIS.2016.7745303.
5. J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, “A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications,” *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017, doi: 10.1109/JIOT.2017.2683200.
6. R. S. Bali, F. Jaafar, and P. Zavarasky, “Lightweight authentication for MQTT to improve the security of IoT communication,” In *Proceedings of the* *3rd International Conference on Cryptography, Security and Privacy (ICCSP '19)*, Association for Computing Machinery, New York, NY, USA, 2019, pp. 6-12, doi: 10.1145/3309074.3309081
7. A. Bhawiyuga, M. Data, and A. Warda, “Architectural design of token based authentication of MQTT protocol in constrained IoT device,” in *2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA)*, 2017, pp. 1-4, doi: 10.1109/TSSA.2017.8272933.
8. A. Rahman, S. Roy, M. S. Kaiser, and M. S. Islam, “A Lightweight Multi-tier S-MQTT Framework to Secure Communication between low-end IoT Nodes,” in *2018 5th International Conference on Networking, Systems and Security (NSysS)*, 2018, pp. 1-6, doi: 10.1109/NSysS.2018.8631379.
9. A. Niruntasukrat, C. Issariyapat, P. Pongpaibool, K. Meesublak, P. Aiumsupucgul, and A. Panya, “Authorization mechanism for mqtt-based internet of things,” in *2016 IEEE International Conference on Communications Workshops (ICC)*, 2016, pp. 290-295, doi: 10.1109/ICCW.2016.7503802.

L. Bisne and M. Parmar, “Composite secure MQTT for Internet of Things using ABE and dynamic S-box AES,” in *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*, 2017, pp. 1–5, doi: 10.1109/IPACT.2017.8245126