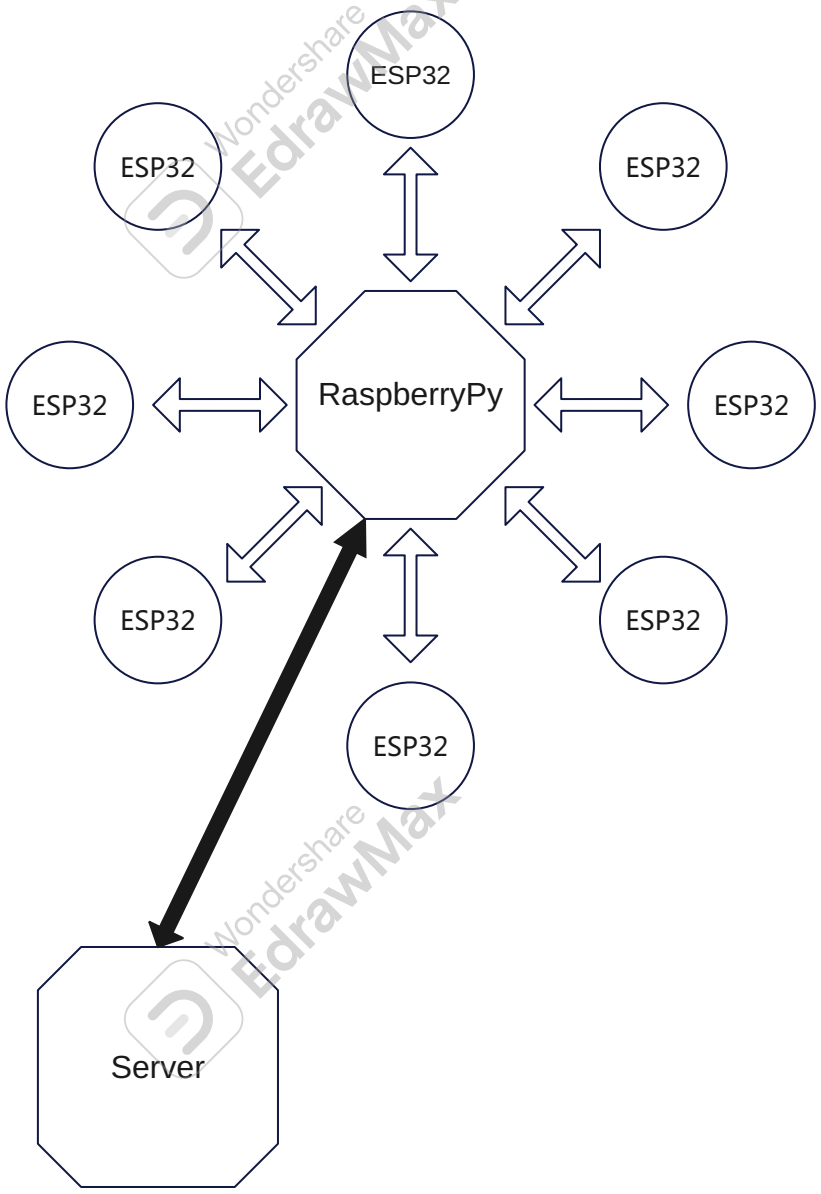
**ESEMPIO DI TOPOLOGIA DELLA RETE MQTT:**




Questo Server può:

-ricevere gli audio e i risultati delle classificazioni audio dal RaspberryPy.
-mandare direttive al RaspberryPy per il setting della rete di sensori o del RaspberryPy stesso.

RIASSUNTO BREVE

#1 Se c'è spazio nella rete di sensori, allora un ESP32 può fare domanda di iscrizione alla rete, inviando una richiesta di iscrizione al RaspberryPy.
Può darsi che la rete di sensori sia già piena, perciò un sensore non può iscriversi sino a quando il RaspberryPy non gli notifica che si è liberato un posto in un secondo momento (a riguardo verrà usata una coda dei richiedenti iscrizione di tipo FIFO, dove anche questa coda avrà una dimensione massima).

#2 Una volta che un sensore si è iscritto alla rete, gli verrà fornito un device_id per identificarlo al suo interno e ricevere comandi, e gli verranno date le informazioni di setting per effettuare le registrazioni audio (frequenza, ecc...).

#3 Ogni comunicazione tra ESP32 e RaspberryPy avrà un proprio communication_id, che verrà generato in modo random da chi comincia ad instaurare la comunicazione, e dovrà essere usato da entrambi per la comunicazione corrente.

#4 Ogni tot di tempo, ogni ESP32 effettuerà una registrazione audio, e notificherà il RaspberryPy di ciò. Gli audio avranno un nome che identificherà il modello dell'ESP32, la locazione in cui questo si trova, e l'orario di registrazione audio. Il RaspberryPy prenderà ogni volta nota di questi, e aggiornerà una lista dei dispositivi pronti ad inviare l'audio.

#5 Quando conveniente (dopo un certo periodo di tempo e dopo che sono state effettuate un tot di registrazioni audio), il RaspberryPy richiederà gli audio agli ESP32 che hanno già effettuato le registrazioni. La richiesta e ricezione degli audio verrà effettuata in multithreading, ovvero gli ESP32 verranno interrogati a gruppi di 4, in parallelo per ogni gruppo (sfruttando l'architettura quad-core). Gli audio verranno salvati in cartelle apposite realtive al thread usato.

#6 Una volta ricevuti e salvati gli audio, questi verranno preprocessati per ottenere uno spettrogramma in formato binario da ciascuno di essi, sempre in modalità multithreading per parallelizzare il più possibile.

#7 Una volta ottenuti gli spettrogrammi, verrà runnato il file eseguibile che utilizza la libreria ARMNN per effettuare la classificazione accelerata su tutti gli spettrogrammi ottenuti prima in fase di preprocessing, e ne verranno salvati i risultati delle classificazioni.

==========

**PS** la rete MQTT permette di fare molte cose, tra cui:

-iscrizione degli ESP32 alla rete di sensori
-notificare il RaspberryPy quando un audio è disponibile
-notificare il RaspberryPy di eventuali problemi nella registrazione audio
-abilitare/disabilitare un ESP32 alla registrazione audio
-mandare in sleep_mode (o risvegliare) un ESP32 quando desiderato
-richiedere informazioni agli ESP32 quando desiderato
-disconnettere dal broker MQTT un ESP32 (cioè cancellarlo dalla rete di sensori)
-resettare un dispositivo ESP32 quando desiderato o quando ci sono problemi
-impostare la frequenza di registrazione audio (intervallo tra due registrazioni)