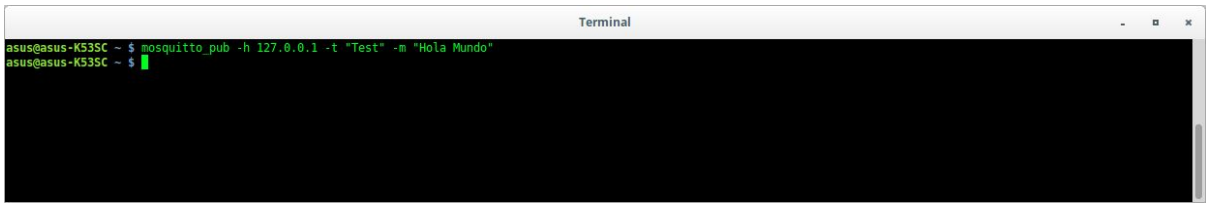
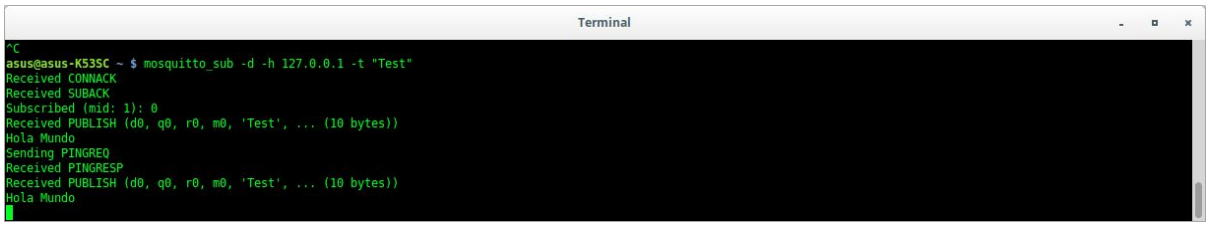
1) **¿Qué respuesta recibe? ¿Qué información le da la respuesta detallada?**





2) Pruebe hacer lo mismo utilizando un dispositivo embebido como cliente. ¿Se presentó alguna dificultad?

No presentó dificultad fue totalmente transparente y sencilla.

3) Utilizando un software sniffer de red (por ejemplo wireshark), visualice el flujo de datos capturados. ¿Cómo es la comunicación? ¿Cuántos paquetes son necesarios y porqué? ¿Cuántos partes intervinientes hay en la comunicación (ojo, recuerde modelo pub/sub)? ¿Cómo es la trama del paquete que lleva el mensaje? ¿Cuántos bytes mide el paquete (recuerde: se le llama "mosquitto")?

La comunicación entre los dispositivos intervinientes en el protocolo de mensajes al ser orientada a la conexión (TCP) genera un excedente de paquetes utilizados para emanar un simple mensaje en el modelo pub/sub.

Esto lo podemos visualizar con la herramienta wireshark que nos provee de los datos numéricos para conocer que la trama de paquetes intervinientes son son más que el paquete de datos, por ejemplo en la figura 3, el "hello world" genera un mensaje de 1344 bits, donde, de los datos completos emanados representa el 60% (816 bits), el porcentaje restante es cabeceras propias del protocolo.

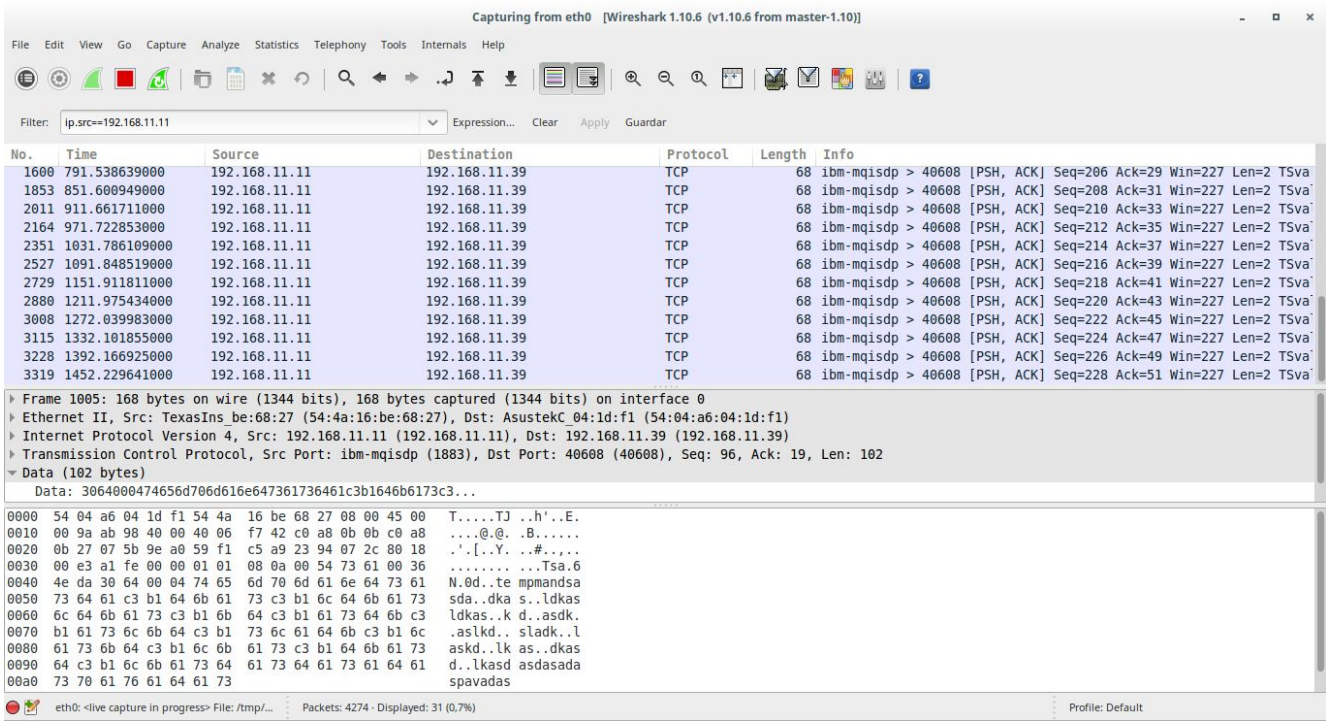


Figura 3

4)1 Siguiendo los pasos propuestos por https://geekytheory.com/tutorialraspberrypigpioymqttparte1/, trate de prender y apagar un led en una placa con linux embebido publicando la orden desde su PC (el servidor puede seguir estando en la PC) Explique brevemente cómo funciona este escenario.

Problema: Prender led remoto por protocolo de mensajes.
Solución:

- Al momento de controlar un LED, nos suscribimos desde la Galileo al topic = led.
    - mosquitto_sub -h 192.168.0.103 -t “led”
- Configuramos el GPIO 3 como salida para comandar el led interno de la placa.
    - echo -n "3" > /sys/class/gpio/export
    - echo -n "out" > /sys/class/gpio/gpio3/direction
- Si enviamos desde la notebook ‘1’ o ‘0’ nos aparecerá en el terminal de la Galileo dicho mensaje, para lo cual se lo direccionamos como flag de on/off al GPIO 3
    - mosquitto_sub -h 192.168.0.103 -t "led" > /sys/class/gpio/gpio3/value
    - El operador ‘>‘ lo que hace es redireccionar lo que devolvería el comando por pantalla a otro fichero, en nuestro caso el fichero “value“.

Ahora desde la terminal del PC, publicamos el mensaje para comandar el led:

- mosquitto_pub -h 192.168.112.129 -t "GPIO" -m "1" // endendido
- mosquitto_pub -h 192.168.112.129 -t "GPIO" -m "0" // apagado

5) ¿Cómo explicaría el funcionamiento de estos códigos? (Ver https://pypi.python.org/pypi/pahomqtt/1.1)
El código del archivo “llamado paho_mqtt_pub.py” genera un publisher que emana cada dos segundos un hello world, al topic “hello/world”, los subscriber al topic “hello/world” tienden a consumir el mensaje provisto .

6) Opcional: En vuestro hogar pruebe cambiar la dirección del broker por “iot.eclipse.org”.