



**Universidad autónoma de baja california**

**Ingeniería en computación**

**Internet de las cosas**

**Taller-lab 1: TCP y UDP en red local**

**Aguilar Noriega Leocundo**

**Garcia Chaves erik**

**Jueves 12 de febrero del 2026**

## Pruebas UDP

### Prueba 1: ESP servidor / PC cliente

Para poder realizar las pruebas es necesario tener el programa **iperf** instalado en la computadora puede ser para Linux/Windows o mac

En el caso personal lo estaré probando para Linux y Windows.

Cuando ESP esta en modo servidor es necesario tener un proyecto, dentro de lo que **espressif** son ejemplos de proyecto, en este caso contiene un ejemplo de un proyecto **ipref2** por lo que usamos ese template para realizar las pruebas.

Para conectarlo a la red wifi del hogar usamos **sta <SSID> <><password>>**

Esp como servidor:

Para la comunicación UDP necesitamos introducir el siguiente comando:

```
Unrecognized command
iperf> iperf -s -u -i 1
Unrecognized command
iperf> iperf -s -u -i 1
I (191924) IPERF: mode=udp-server sip=localhost:5001, dip=0.0.0.0:5001, interval=1, time=30
iperf> I (191926) iperf: Socket created
I (191934) iperf: Socket bound, port 35891
W (201935) iperf: udp server recv error, error code: 11, reason: No more processes
I (201936) iperf: Udp socket server is closed.
I (201937) iperf: iperf exit
```

Donde **-s** indicara que el ESP estará esuchando conexiones en el puerto 5001. **-u** protocolo UDP, **-i 1** muestra la estadística cada segundo

PC como cliente:

```

[ 1] local 172.27.89.16 port 58882 connected with 192.168.1.93 port 5001
[ ID] Interval Transfer Bandwidth
[ 1] 0.0000-1.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 1.0000-2.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 2.0000-3.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 3.0000-4.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 4.0000-5.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 5.0000-6.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 6.0000-7.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 7.0000-8.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 8.0000-9.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 9.0000-10.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 10.0000-11.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 11.0000-12.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 12.0000-13.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 13.0000-14.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 14.0000-15.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 15.0000-16.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 16.0000-17.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 17.0000-18.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 18.0000-19.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 19.0000-20.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 20.0000-21.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 21.0000-22.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 22.0000-23.0000 sec 1.25 MBytes 10.5 Mbits/sec
[ 1] 23.0000-24.0000 sec 1.25 MBytes 10.5 Mbits/sec
*[ 1] 0.0000-24.1626 sec 30.2 MBytes 10.5 Mbits/sec
[ 1] Sent 21546 datagrams
[ 3] WARNING: did not receive ack of last datagram after 10 tries.

```

Para que la PC es necesario el siguiente comando: **iperf -c 192.168.1.93 -u -b 10M -i 1 -t 30**

Donde **-c 192.168.1.93** indica estamos en el modo cliente y nos conécteme a la IP 192.168.1.93 que viene siendo el ESP32. **-u** usamos el protocolo UDP. **-b 10M** el cliente intentara enviar datos a 10Mbps, con un intervalo de 1 segundo **-i 1**. Y durara **-t 30** segundos después de esto se detiene en automático.

Que es la respuesta que obtenermos.

El ancho de banda y la información transferida es constante lo que es un buen indicativo de que el rendimiento es estable, la conexión se establecio correctamente, en este caso solo el lado del cliente nos muestra información, por el tipo de protocolo que es UDP, el cual no enlaza una conexión entre cliente-servidor.

## **Prueba 2: PC servidor / ESP cliente < UDP>**

PC servidor:

```
$ sudo apt install net-tools
net-tools is already the newest version (2.10-2).
net-tools set to manually installed.
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0

(wtr@dmrxd)-[~]
$ iperf -s -u -i 1
-----
Server listening on UDP port 5001
UDP buffer size: 208 KByte (default)
-----
```

Del lado de la PC <linux> será como servidor ahora, va a estar escuchando las solicitudes, los comandos son prácticamente lo mismo, de igual manera necesitamos conocer la dirección IP de nuestra maquina para que se pueda establecer la conexión

ESP32 cliente:

```
iperf> iperf -c 172.27.89.16 -u -b 10000000 -i 1 -t 30
I (842988) IPERF: mode=udp-client sip=localhost:5001, dip=172.27.89.16:5001, interval=1, time=30
iperf> I (842990) iperf: Socket created, sending to 274275244:5001

Interval      Bandwidth
0.0- 1.0 sec  54.19 Mbits/sec
1.0- 2.0 sec  51.66 Mbits/sec
2.0- 3.0 sec  51.32 Mbits/sec
3.0- 4.0 sec  49.90 Mbits/sec
4.0- 5.0 sec  49.34 Mbits/sec
5.0- 6.0 sec  48.37 Mbits/sec
6.0- 7.0 sec  48.14 Mbits/sec
7.0- 8.0 sec  48.81 Mbits/sec
8.0- 9.0 sec  48.17 Mbits/sec
9.0-10.0 sec  48.67 Mbits/sec
10.0-11.0 sec 48.28 Mbits/sec
11.0-12.0 sec 47.74 Mbits/sec
12.0-13.0 sec 47.50 Mbits/sec
13.0-14.0 sec 47.28 Mbits/sec
14.0-15.0 sec 46.77 Mbits/sec
15.0-16.0 sec 46.53 Mbits/sec
16.0-17.0 sec 46.11 Mbits/sec
17.0-18.0 sec 45.70 Mbits/sec
18.0-19.0 sec 45.86 Mbits/sec
19.0-20.0 sec 46.81 Mbits/sec
20.0-21.0 sec 46.06 Mbits/sec
21.0-22.0 sec 46.35 Mbits/sec
22.0-23.0 sec 46.16 Mbits/sec
23.0-24.0 sec 46.16 Mbits/sec
24.0-25.0 sec 46.14 Mbits/sec
25.0-26.0 sec 45.89 Mbits/sec

iperf> 26.0-27.0 sec  45.14 Mbits/sec
```

Ahora el ESP32 esta en modo cliente. El comando es igual, solo que ahora el ESP estaría mandando solicitudes a la terminal de PC

Podemos ver que la velocidad es mucho mayor que la solicitada de al rededor de 45 Mbps, dado si arquitectura esta mas optimizado para enviar que una PC común.

El protocolo UDP, es un protocolo muy rápido, pero tiene sus desventajas como el no establecer una conexión entre cliente-servidor, por lo que no hay certeza de que los paquetes lleguen a su destino, los paquetes llegan sin un orden claro.

Pero es ideal para sistemas o aplicaciones donde la perdida de paquetes puede no ser algo tan significativo, como videollamadas, videojuegos, monitor de cámaras de vigilancia.

## Pruebas TCP

### **Prueba 3: ESP como servidor / PC como cliente < TCP >**

#### **ESP como servidor:**

Ahora para que el ESP funcione como servidor por **TCP** se usa el comando **iperf -s -i 1** ahora no usamos **-u** porque eso es para indicar que será por udp, lo cual no estaremos usando si ni TCP

La velocidad es prácticamente buena pero la conexión no es estable tiende a subir caídas y perdida de datos en la transferencia. Es una conexión buena para protocolos de comunicación IoT como MQTT u otros como HTTP

```
iperf> iperf -s -i 1
I (1084678) IPERF: mode=tcp-server sip=localhost:5001, dip=0.0.0.0:5001, interval=1, time=30
iperf> I (1084679) iperf: Socket created
I (1088816) iperf: accept: 192.168.1.66,59430

Interval      Bandwidth
 0.0- 1.0 sec 24.99 Mbits/sec
 1.0- 2.0 sec 20.26 Mbits/sec
 2.0- 3.0 sec 0.35 Mbits/sec
 3.0- 4.0 sec 14.05 Mbits/sec
 4.0- 5.0 sec 26.11 Mbits/sec
 5.0- 6.0 sec 25.89 Mbits/sec
 6.0- 7.0 sec 23.20 Mbits/sec
 7.0- 8.0 sec 23.03 Mbits/sec
 8.0- 9.0 sec 25.32 Mbits/sec
 9.0-10.0 sec 26.14 Mbits/sec
10.0-11.0 sec 27.06 Mbits/sec
11.0-12.0 sec 14.64 Mbits/sec
12.0-13.0 sec 0.49 Mbits/sec
13.0-14.0 sec 16.63 Mbits/sec
14.0-15.0 sec 30.12 Mbits/sec
15.0-16.0 sec 29.98 Mbits/sec
16.0-17.0 sec 27.21 Mbits/sec
17.0-18.0 sec 26.98 Mbits/sec
18.0-19.0 sec 27.49 Mbits/sec
19.0-20.0 sec 26.89 Mbits/sec
20.0-21.0 sec 26.92 Mbits/sec
21.0-22.0 sec 8.24 Mbits/sec
22.0-23.0 sec 20.24 Mbits/sec
23.0-24.0 sec 15.24 Mbits/sec
24.0-25.0 sec 0.49 Mbits/sec
25.0-26.0 sec 26.04 Mbits/sec
26.0-27.0 sec 7.06 Mbits/sec
27.0-28.0 sec 0.19 Mbits/sec
```

#### **PC Como cliente:**

```
(wtr@dmrxd)~$ iperf -c 192.168.1.93 -i 1 -t 30
Client connecting to 192.168.1.93, TCP port 5001
TCP window size: 16.0 KByte (default)

[ 1] local 172.27.89.16 port 36672 connected with 192.168.1.93 port 5001
[ ID] Interval Transfer Bandwidth
[ 1] 0.0000-1.0000 sec 3.50 MBytes 29.4 Mbits/sec
[ 1] 1.0000-2.0000 sec 2.50 MBytes 21.0 Mbits/sec
[ 1] 2.0000-3.0000 sec 271 KBytes 2.22 Mbits/sec
[ 1] 3.0000-4.0000 sec 1.50 MBytes 12.6 Mbits/sec
[ 1] 4.0000-5.0000 sec 3.38 MBytes 28.3 Mbits/sec
[ 1] 5.0000-6.0000 sec 3.25 MBytes 27.3 Mbits/sec
[ 1] 6.0000-7.0000 sec 2.75 MBytes 23.1 Mbits/sec
[ 1] 7.0000-8.0000 sec 3.00 MBytes 25.2 Mbits/sec
[ 1] 8.0000-9.0000 sec 3.00 MBytes 25.2 Mbits/sec
[ 1] 9.0000-10.0000 sec 3.38 MBytes 28.3 Mbits/sec
[ 1] 10.0000-11.0000 sec 3.50 MBytes 29.4 Mbits/sec
[ 1] 11.0000-12.0000 sec 1.90 MBytes 15.9 Mbits/sec
[ 1] 12.0000-13.0000 sec 61.9 KBytes 507 Kbits/sec
[ 1] 13.0000-14.0000 sec 1.81 MBytes 15.2 Mbits/sec
[ 1] 14.0000-15.0000 sec 4.00 MBytes 33.6 Mbits/sec
[ 1] 15.0000-16.0000 sec 3.62 MBytes 30.4 Mbits/sec
[ 1] 16.0000-17.0000 sec 3.50 MBytes 29.4 Mbits/sec
[ 1] 17.0000-18.0000 sec 3.38 MBytes 28.3 Mbits/sec
[ 1] 18.0000-19.0000 sec 3.38 MBytes 28.3 Mbits/sec
[ 1] 19.0000-20.0000 sec 3.38 MBytes 28.3 Mbits/sec
[ 1] 20.0000-21.0000 sec 3.38 MBytes 28.3 Mbits/sec
[ 1] 21.0000-22.0000 sec 1.06 MBytes 8.91 Mbits/sec
[ 1] 22.0000-23.0000 sec 2.43 MBytes 20.4 Mbits/sec
[ 1] 23.0000-24.0000 sec 2.12 MBytes 17.8 Mbits/sec
[ 1] 24.0000-25.0000 sec 1.69 KBytes 13.8 Kbits/sec
```

Al ser una conexión **TCP**, tanto el cliente como el servidor establecen una conexión, por lo que el puerto por donde se esta comunicando la PC al ESP32 en este momento no puede ser usado por algún otra PC, la razón por la que en esta ocasión tanto ESP como PC muestran las mediciones, es porque ambos confirman, PC envia datos a ESP, ESP debe de confirmar que llegaron estos paquetes.

#### **Prueba 4: PC servidor / esp32 cliente <TCP>**

## PC servidor (Windows ):

Como ahora el que esta recibiendo las solicitudes es la PC que tiene capacidades computacionales mas altas que un ESP32 la conexión es mucho mas estable el ancho de banda mucho mas estable y la cantidad de infromacion trasmitida no tiene mucha diferencia.

Esto también se pudo ver con UDP, en donde el ESP es mucho mas veloz tiene muchas mas habilidades para trasmisir que para recibir.

```
ut Server listening on TCP port 5001 A
opy TCP window size: 64.0 KByte (default)
format Panter
ard [ 1] local 192.168.1.66 port 5001 connected with 192.168.1.93 port 64676
[ 1] ID] Interval Transfer Bandwidth
[ 1] 0.00-1.00 sec 2.41 MBytes 20.2 Mbits/sec
[ 1] 1.00-2.00 sec 3.86 MBytes 32.4 Mbits/sec
[ 1] 2.00-3.00 sec 3.75 MBytes 31.5 Mbits/sec
[ 1] 3.00-4.00 sec 3.21 MBytes 27.0 Mbits/sec
[ 1] 4.00-5.00 sec 3.38 MBytes 28.3 Mbits/sec
[ 1] 5.00-6.00 sec 2.97 MBytes 24.9 Mbits/sec
[ 1] 6.00-7.00 sec 3.09 MBytes 26.0 Mbits/sec
[ 1] 7.00-8.00 sec 3.40 MBytes 28.5 Mbits/sec
[ 1] 8.00-9.00 sec 3.42 MBytes 28.7 Mbits/sec
[ 1] 9.00-10.00 sec 3.08 MBytes 25.8 Mbits/sec
[ 1] 10.00-11.00 sec 3.41 MBytes 28.6 Mbits/sec
[ 1] 11.00-12.00 sec 3.13 MBytes 26.2 Mbits/sec
[ 1] 12.00-13.00 sec 3.14 MBytes 26.4 Mbits/sec
[ 1] 13.00-14.00 sec 3.32 MBytes 27.9 Mbits/sec
[ 1] 14.00-15.00 sec 3.11 MBytes 26.1 Mbits/sec
[ 1] 15.00-16.00 sec 3.16 MBytes 26.5 Mbits/sec
[ 1] 16.00-17.00 sec 2.94 MBytes 24.7 Mbits/sec
[ 1] 17.00-18.00 sec 3.02 MBytes 25.3 Mbits/sec
[ 1] 18.00-19.00 sec 3.21 MBytes 27.0 Mbits/sec
[ 1] 19.00-20.00 sec 3.12 MBytes 26.2 Mbits/sec
[ 1] 20.00-21.00 sec 3.39 MBytes 28.4 Mbits/sec
[ 1] 21.00-22.00 sec 3.24 MBytes 27.1 Mbits/sec
[ 1] 22.00-23.00 sec 3.53 MBytes 29.7 Mbits/sec
[ 1] 23.00-24.00 sec 3.41 MBytes 28.6 Mbits/sec
[ 1] 24.00-25.00 sec 3.35 MBytes 28.1 Mbits/sec
[ 1] 25.00-26.00 sec 3.10 MBytes 26.0 Mbits/sec
[ 1] 26.00-27.00 sec 3.08 MBytes 25.9 Mbits/sec
[ 1] 27.00-28.00 sec 2.81 MBytes 23.6 Mbits/sec
[ 1] 28.00-29.00 sec 3.67 MBytes 30.8 Mbits/sec
[ 1] 29.00-30.00 sec 3.57 MBytes 29.9 Mbits/sec
```

## ESP cliente:

```
iperf> I (997798) iperf: Successfully connected
Interval      Bandwidth
0.0- 1.0 sec  19.75 Mbits/sec
1.0- 2.0 sec  30.62 Mbits/sec
2.0- 3.0 sec  29.75 Mbits/sec
3.0- 4.0 sec  25.75 Mbits/sec
4.0- 5.0 sec  27.25 Mbits/sec
5.0- 6.0 sec  23.50 Mbits/sec
6.0- 7.0 sec  25.25 Mbits/sec
7.0- 8.0 sec  26.88 Mbits/sec
8.0- 9.0 sec  27.50 Mbits/sec
9.0-10.0 sec  24.62 Mbits/sec
10.0-11.0 sec 27.25 Mbits/sec
11.0-12.0 sec 24.62 Mbits/sec
12.0-13.0 sec 25.12 Mbits/sec
13.0-14.0 sec 26.75 Mbits/sec
14.0-15.0 sec 24.75 Mbits/sec
15.0-16.0 sec 25.38 Mbits/sec
16.0-17.0 sec 23.38 Mbits/sec
17.0-18.0 sec 24.25 Mbits/sec
18.0-19.0 sec 25.62 Mbits/sec
19.0-20.0 sec 25.50 Mbits/sec
20.0-21.0 sec 27.00 Mbits/sec
21.0-22.0 sec 25.88 Mbits/sec
22.0-23.0 sec 28.12 Mbits/sec
23.0-24.0 sec 27.25 Mbits/sec
24.0-25.0 sec 27.12 Mbits/sec
25.0-26.0 sec 24.50 Mbits/sec
26.0-27.0 sec 24.50 Mbits/sec
27.0-28.0 sec 22.75 Mbits/sec
```

ESP cliente:

## UDP vs TCP

El protocolo **UDP** resultó ser mucho más rápido, gracias a su arquitectura de dispara y olvida, se pudieron alcanzar velocidades de 45 Mbps, esta arquitectura permite mandar mucha más información sin sobrecargar el ancho de banda, a diferencia de **TCP** que es algo un poco más lenta debido a su necesidad de gestionar la integridad de los datos, debido a esto con **UDP** no hay certeza de que los datos hayan llegado a su destino o que hayan llegado en orden, por **UDP** tienden a perderse datos.

Pudimos ver que el **ESP32** tiende a ser mucho mejor para trasmisir datos en vez de recibirllos. Esto se pudo ver en ambos protocolos. Para protocolos donde la integridad de los datos es importante como **MQTT** o **HTTP** se usa **TCP**.

### **Conclusión:**

Durante la práctica pudimos ver pruebas de transferencia de datos entre PC – ESP32 mediante los protocolos TCP y UDP, esto es importante a la hora de implementar aplicaciones en IoT, esto nos va a indicar que dispositivos o para qué vamos a poder usar en nuestra aplicación, dado el peso de los paquetes de algún periférico va a dar para mandar por la red o si es para una cámara, etc. Estos datos serán de gran utilidad.