

Xarxes

Pràctica 3. Comunicacions a través de la pila TCP/IP

Informe realitzat per Joaquim Yuste i Marcos Plaza

Universitat de Barcelona
Facultat de Matemàtiques i Informàtica

Índex

Objectius de la pràctica	3
Creació del canal recol·lector de dades	4
Creació d'un Punt d'Accés	6
- Servidor	6
- Client	7
Cisco Packet Tracer. Muntatge d'una xarxa d'ordinadors i execució de ping	8
Conclusions	10

Objectius de la pràctica

El principal objectiu de la pràctica és conèixer el funcionament del protocol TCP/IP mitjançant uns quants exercicis pràctics proposats a l'enunciat d'aquesta. Dins aquests exercicis volem assolir al mateix temps una sèrie d'objectius.

- 1) El primer exercici consisteix a fer servir els dispositius de la pràctica anterior per a connectar-nos a un servidor que recollirà les lectures del RSSI i les imprimirà per pantalla. Per això utilitzarem el web thingspeak.
- 2) En el segon exercici passarem d'utilitzar el mode WIFI_STA a WIFI_AP, és a dir, farem servir la mota com a punt d'accés per a crear una xarxa pròpia on un membre de la parella actuarà com a client i un altre com a servidor. Nosaltres volem utilitzar aquesta arquitectura Client-Servidor per a implementar un xat.
- 3) Per últim volem fer servir el software de simulació Cisco Packet Tracer per a aprendre com muntar una xarxa i veure el flux de dades que hi ha en fer un ping d'un ordinador de la xarxa a un altre.

Els programes utilitzats per a dur les diferents tasques es troben adjunts a aquest arxiu dins una carpeta anomenada 'codi'.

Creació del canal recol·lector de dades

Com hem dit en els objectius de la pràctica, volem crear un canal on agafar totes les dades que anirem enviant des del nostre dispositiu ESP8266. Per a crear aquest canal farem servir el web thingspeak.

Per tal d'utilitzar thingspeak primer ens crearem un compte al lloc web per així poder crear un canal propi amb el nom 'ESP8266_MarcosPlaza_RSSI'. D'aquest canal agafarem l'ID i el write API Key per a poder crear el paquet de dades a enviar així com per a connectar-nos al servidor.

El que implementarem al codi del programa 'P3_E1_Thingspeak' serà un menú senzill molt semblant al que vam fer a la pràctica anterior. Quan executem podem fer el següent; (1) escanejar les xarxes disponibles, (2) connectar-se a una xarxa Wi-Fi, (3) connectar-se al servidor thingspeak i per últim (4) disconnectar-se de la xarxa Wi-Fi. L'interès de l'exercici està en el punt (3) on enviarem l'RSSI del senyal de la Wi-Fi a la qual estem connectats al servidor. A continuació veurem el fragment de codi que fa possible enviar dades:

```
void thingspeak(){
  if(client.connect(server,80)){
    long rssi = WiFi.RSSI();

    String body = "field1=";
    body += String(rssi);

    Serial.print("RSSI: ");
    Serial.println(rssi);

    client.println("POST /update HTTP/1.1");
    client.println("Host: api.thingspeak.com");
    client.println("User-Agent: ESP8266 (nothans)/1.0");
    client.println("Connection: close");
    client.println("X-THINGSPEAKAPIKEY: "+writeAPIKey);
    client.println("Content-Type: application/x-www-form-urlencoded");
    client.println("Content-Length: " + String(body.length()));
    client.println("");
    client.println(body);
  }
  client.stop();
  delay(postingInterval);
}
```

Figura 1. Funció thingspeak() del programa 'P3_E1_Thingspeak'

Com podem observar en la Figura 1 anterior, el nostre programa el que farà serà el següent:

Inicialment declararem una variable del tipus WiFiClient el qual ens permet connectar-nos a thingspeak a través d'una determinada IP i un port. Podem veure que si és possible la connexió entre la mota i el servidor passem a construir el nostre paquet de dades. Ens interessa agafar l'RSSI (ho farem mitjançant WiFi.RSSI()) del senyal Wi-Fi a la que estem connectats i afegir-ho dins el paquet.

A més de les altres dades que es poden veure a la Figura 1, per enviar el paquet també necessitarem el write API Key per així poder escriure les dades al nostre canal. Per últim necessitem especificar la llargada d'allò que volem escriure així com les dades que volem escriure en si amb el format "field1=RSSI". En cada iteració deixarem un temps de retard (20 segons) perquè no se superposin les dades a enviar i anar agafant les dades de manera més o menys continua.

Finalment el que obtenim després d'enviar unes quantes mesures del RSSI de la nostra xarxa serà un resultat com aquest:

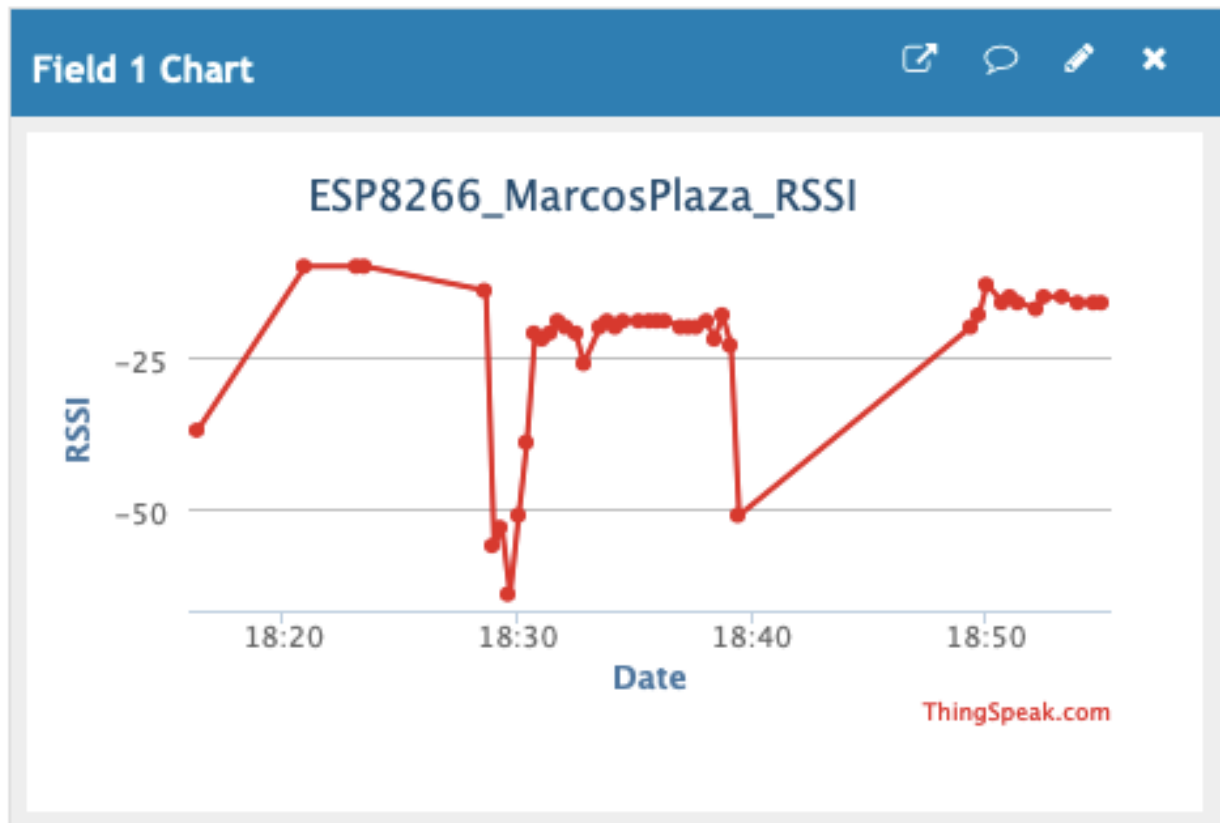


Figura 2. Resultat d'enviar l'RSSI al web *thingspeak*

Hem fet variar de tant en tant la distància entre el punt d'accés (telèfon mòbil) i la mota per a veure que el valor de l'RSSI augmenta en apropar la mota a aquest i disminueix respectivament en allunyar-se de la font.

Creació d'un Punt d'Accés

Fins ara només havíem fet servir la mota per a connectar-nos a una xarxa Wi-Fi determinada com a client. En aquest exercici el que farem és agafar 2 plaques ESP8266 per tal de muntar una arquitectura Client-Servidor i així poder fer un xat personal.

Inicialment, en el nostre programa, configurarem una placa en mode WIFI_STA (STATION) per a poder connectar-la a l'altre placa que haurà d'estar configurada en WIFI_AP (Soft ACCESS POINT) per a poder establir la seva pròpia xarxa Wi-Fi. D'aquesta manera estem inicialitzant una placa com a client i una altra com a servidor respectivament.

- Servidor

La placa encarregada de gestionar el servidor serà a la que utilitzi el mode de WIFI_AP. La seva configuració és la següent:

```
char ssid[] = "WiFi server chat";
char password[] = "12345678";

WiFiServer server(80); //port del server

void setup() {
  Serial.begin(9600);
  WiFi.mode(WIFI_AP); //Acces mode selected
  WiFi.softAP(ssid,password);
  delay(1000);

  server.begin();
  Serial.println("\nWiFi ON");
  Serial.print("Local IP: ");
  Serial.println(WiFi.softAPIP());
  Serial.print("Server port: ");
  Serial.println("80");
}
```

Figura 3. Inicialització del punt d'accés i del server

Un cop hem assignat el mode de treball (a punt d'accés), inicialitzem la xarxa Wi-Fi amb la funció `WiFi.softAP(char *ssid, char *password)` on *ssid* i *password* fan referència el nom i la clau de la xarxa respectivament.

A continuació ens disposem a obrir el servidor perquè aquest estigui atent a les connexions entrants fent servir la funció `server.begin()` de la classe `WiFiServer`. Aquest server tindrà una IP associada, i la necessitem conèixer perquè l'usuari pugui establir connexió, per tant la retornarem per consola.

Com ja hem comentat, el nostre objectiu es construir un chat entre dues motes, per fer-ho, la mota client enviarà un missatge al servidor i aquest el retornarà a tota la resta de clients. La forma d'enviar aquest missatge es fent servir POST Request i un JSON com a body del paquet.

```

client.write(c);
if (c == '\n' && currentLineIsBlank) {

// Here is where the POST data is.
while(client.available())
{
    char c = client.read(); //we read the next byte
    data += c;
    client.write(c); //and send it back to the client
}
finish = true;
}
if (finish){
    finish = false;
    char c_data[data.length()];
    data.toCharArray(c_data,data.length()+1);

    deserializeJson(doc,c_data); //now we read the whole message
    const char *user_name = doc["UserName"]; //and we print it to de console
    const char *message = doc["Message"];
    Serial.print(user_name);
    Serial.print(": ");
    Serial.println(message);
}

```

Figura 4. Fragment de codi de lectura de dades (només l'apartat de lectura del body)

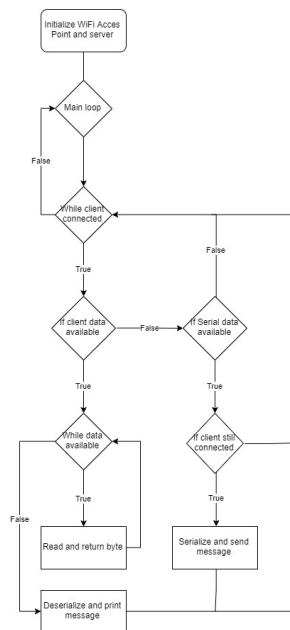


Figura 5. Diagrama de flux del servidor

- Client

El client, després d'haver configurat el mode d'actuació i connectar-se al servidor amb èxit, es limitarà a enviar i rebre missatges per imprimir-los per consola, per enviar aquest missatge fem servir el mateix protocol que a l'apartat anterior amb l'afegit que ara utilitzem una estructura JSON per emmagatzemar la informació que volem transmetre.

```

void send_message(String userName, String message){
    DynamicJsonDocument doc(1024);
    doc["UserName"] = userName;
    doc["Message"] = message;
    if (server_connected) {
        client.print("POST /api/Users HTTP/1.1");
        client.print("content-type: application/json");
        client.print("Host: ");
        client.print(server); //IP del server
        client.print("Connection: close");
        client.println();
        serializeJson(doc, client);
    }
}

```

Figura 6. Codi encarregat de transmetre el missatge

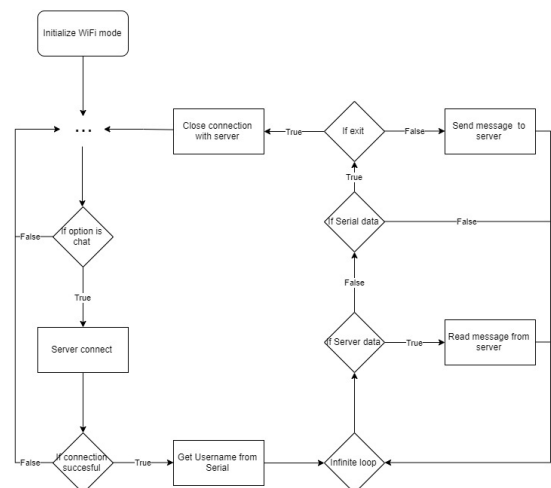


Figura 7. Diagrama de flux del chat client.

Cisco Packet Tracer. Muntatge d'una xarxa d'ordinadors i execució de ping

Per a aquesta activitat farem servir el software Cisco Packet Tracer per a simular la transmissió de dades dins d'una xarxa senzilla.

Al primer punt, se'ns demana poder configurar una xarxa que constarà de 4 Pc's així com d'un Switch. Per a això, primer haurem d'assignar les IP privades a cada ordinador del tipus 192.168.0.x amb màscara 255.255.255.0. Per a cada equip de la xarxa haurem d'anar a la configuració del FastEthernet() i allà assignar la IP així com la màscara de xarxa. A la Figura 8 podem veure com és la pantalla de configuració de la IPv4 per a cada ordinador.

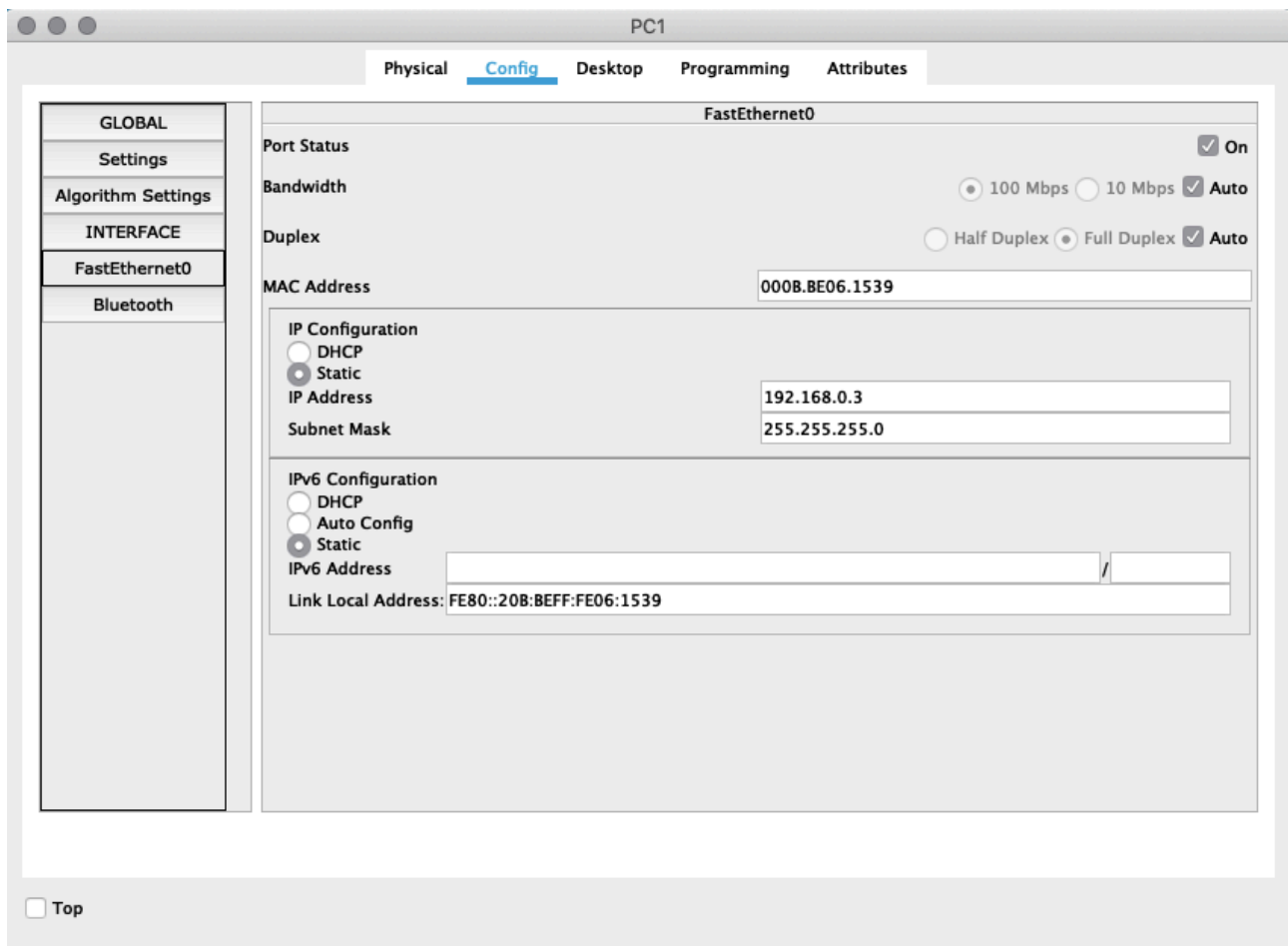


Figura 8. Configuració de la IP i de la màscara de xarxa

A continuació hem de connectar tots els Pc's a un Switch mitjançant el 'Copper Straight-Through' (seleccionat automàticament pel programa) de manera que puguem obtenir una xarxa d'arquitectura estrella i així tenir tots els Pc's interconnectats. La xarxa resultant és tal com es mostra a la imatge situada a la dreta.

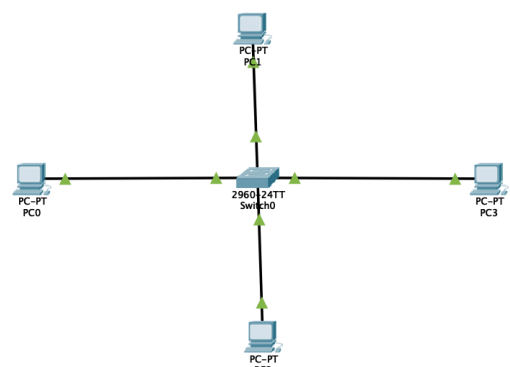


Figura 9. Xarxa que consta d'un Switch i de 4 Pc's

Per continuar haurem d'obrir l'Event List de dins del programa per a poder observar el tràfic de dades quan fem un ping d'un ordinador a un altre pels diferents protocols. Per fer la prova que la xarxa està ben muntada procedim a fer el ping a l'adreça IP 192.168.0.3, i per això haurem d'obrir el Command Prompt dins per a l'ordinador des de on volem executar la comanda. Podem seleccionar diferents protocols IP per a fer la simulació de la comunicació entre els Pc's, en el nostre cas seleccionarem el protocol ICMP (Internet Control Message Protocol), el qual al fer el ping, enviarà un missatge de petició (Echo Request) i rebrà un missatge de resposta (Echo Reply) per tal de determinar si el host al que volem enviar dades està disponible així com el temps que utilitzaran els paquets en anar i tornar.

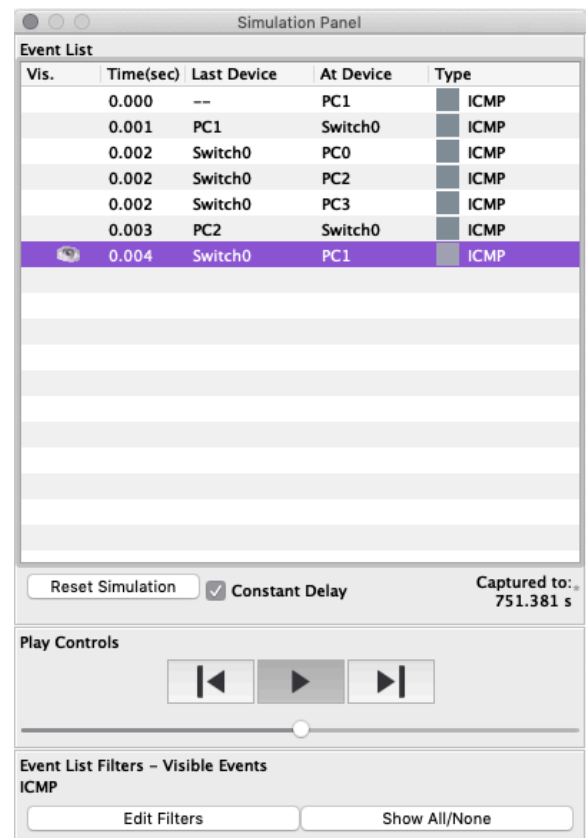
```
C:\>ping 192.168.0.3

Pinging 192.168.0.3 with 32 bytes of data:

Reply from 192.168.0.3: bytes=32 time=4ms TTL=128
Reply from 192.168.0.3: bytes=32 time=4ms TTL=128
Reply from 192.168.0.3: bytes=32 time=4ms TTL=128
Reply from 192.168.0.3: bytes=32 time=4ms TTL=128

Ping statistics for 192.168.0.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 4ms, Maximum = 4ms, Average = 4ms
```

Figura 10. Execució de la comanda ping a l'ordinador amb adreça IP 192.168.0.3



The screenshot shows the 'Simulation Panel' window with the 'Event List' tab selected. The table below represents the data shown in the event list:

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC1	ICMP
	0.001	PC1	Switch0	ICMP
	0.002	Switch0	PC0	ICMP
	0.002	Switch0	PC2	ICMP
	0.002	Switch0	PC3	ICMP
	0.003	PC2	Switch0	ICMP
	0.004	Switch0	PC1	ICMP

Below the table, there are controls for the simulation: 'Reset Simulation', a checked 'Constant Delay' checkbox, and a 'Captured to: 751.381 s' label. There are also 'Play Controls' with play, stop, and pause buttons, and an 'Event List Filters - Visible Events' section showing 'ICMP' as the selected filter.

Figura 11. Event list en executar el programa ping

A partir de la imatge anterior podem veure que per pantalla se'ns imprimeixen diverses dades relatives al datagrama IP. Podem veure en primer lloc que el missatge de resposta l'estem rebent de part de la màquina que nosaltres hem declarat com a destí (ja que podem veure un 'reply' de l'adreça IP corresponent). A continuació també es mostra la mida de la resposta en bytes així com el temps que ha trigat i el TTL el qual representa el número màxim de routers IP que el paquet pot travessar abans de ser descartat (en el nostre cas és 128).

En referència al tràfic de les dades, segons la Figura 11 podem veure que el paquet de dades surt d'origen cap al switch i aquest l'envia a tota la resta dels ordinadors. Només l'ordinador destí enviarà una resposta, ja que els altres no corresponen a l'adreça IP amb la que hem fet ping.

Els passos que hem fet per a arribar a fer aquesta prova de transmissió de dades entre diferents computadors dins la mateixa xarxa tenen molt a veure entre ells. Serà essencial (per a arribar a fer un ping d'un ordinador a un altre) establir les adreces IP dins l'arquitectura TCP/IP així com el medi de transmissió física de les dades (selecció del FastEthernet()).

Conclusions

En la realització de la pràctica hem pogut explorar encara més les possibilitats del dispositiu proporcionat ESP8862 per a treballar amb ell a nivell de la capa de Xarxa així com veure com funciona la pila TCP/IP.

A través de la placa ens hem pogut connectar al servidor thingspeak i enviar les mesures del RSSI que anàvem prenent de la Wi-Fi del nostre acces point personal (telèfon mòbil).

Seguidament hem establert amb èxit una arquitectura Client-Servidor (treballant en diferents modes) entre dues plaques diferents per a aconseguir transmetre i enviar dades i poder així muntar un xat tal com hem explicat anteriorment.

Per últim hem muntat una xarxa de 4 ordinadors tots connectats a un Switch i hem fet proves en la connexió entre diferents ordinadors d'aquesta. Per a fer això hem fet servir el software de simulació Cisco Packet Tracer.

Així doncs, podem dir que hem assolit els objectius esmentats anteriorment amb prou satisfacció.