

Universitat Rovira i Virgili
Escola Tècnica Superior d'Enginyeria

XARXES DE DADES

MEMÒRIA DE PRÀCTIQUES

AUTORS:

JAVIER FUENTES ABALO

IVAN MORILLAS GOMEZ

JOSEP MIQUEL SANTANDER RODRÍGUEZ

DOCENT:

XAVIER PALOMO TERUEL

22/12/2023

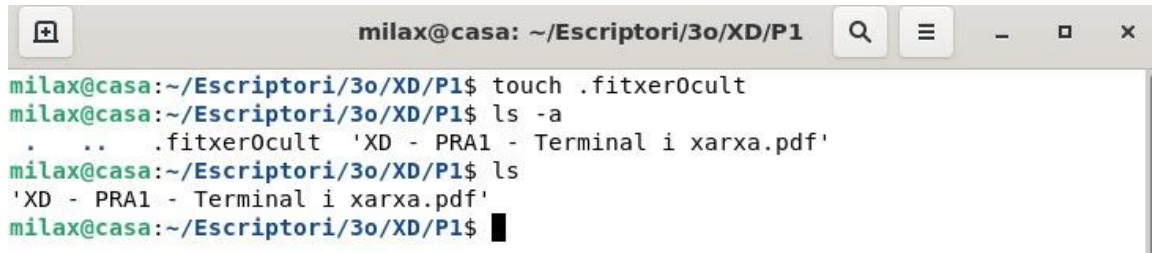
2023-2024

ÍNDEX

1. Terminal i xarxa
2. Rendiment de la xarxa
3. Comunicació entre aplicacions
4. Hello web i wireshark
5. Protocols i Wireshark
6. Muntatge d'una LAN
7. Muntatge d'una LAN Wi-Fi
8. Conclusions
9. Valoracions personals

PRÀCTICA 1: Terminal i xarxa

PRA1.1 Creeu un fitxer ocult amb el nom que vulgueu. Mostreu una captura de pantalla que demostrí que es tracta d'un fitxer ocult.

A screenshot of a terminal window titled 'milax@casa: ~/Escriptori/3o/XD/P1'. The terminal shows the following commands and output:

```
milax@casa:~/Escriptori/3o/XD/P1$ touch .fitxerOcult
milax@casa:~/Escriptori/3o/XD/P1$ ls -a
.  ..  .fitxerOcult  'XD - PRA1 - Terminal i xarxa.pdf'
milax@casa:~/Escriptori/3o/XD/P1$ ls
'XD - PRA1 - Terminal i xarxa.pdf'
milax@casa:~/Escriptori/3o/XD/P1$
```

Amb la comanda **touch** podem crear un fitxer, afegint un “.” davant del nom del fitxer a crear aconseguim fer el fitxer ocult.

Per veure un llistat dels fitxers utilitzem la comanda **ls** i amb afegint el paràmetre **-a (all)** aconseguim veure tots els fitxers incloent els fitxers ocults.

PRA1.2 Creeu una estructura de carpetes **laboratori0/document/exercici i**, dins aquesta carpeta, creeu tres fitxers amb el nom de pila de cadascun dels components del grup. Mostreu una captura de pantalla on es vegi el resultat de llistar aquests fitxers, i es demostrí que l'estructura de carpetes és la que us hem demanat.

```
milax@d114:~$ cd /home/milax
milax@d114:~$ mkdir laboratori0
milax@d114:~$ cd laboratori0
milax@d114:~/laboratori0$ mkdir document
milax@d114:~/laboratori0$ cd document
milax@d114:~/laboratori0/document$ mkdir exercici
milax@d114:~/laboratori0/document$ cd exercici
milax@d114:~/laboratori0/document/exercici$ touch miquel.txt
milax@d114:~/laboratori0/document/exercici$ touch ivan.txt
milax@d114:~/laboratori0/document/exercici$ touch javi.txt
milax@d114:~/laboratori0/document/exercici$ ls
ivan.txt  javi.txt  miquel.txt
milax@d114:~/laboratori0/document/exercici$
```

Amb la comanda **mkdir** creem un nou directori i juntament amb la comanda **cd** i accedim en ell. Dins del directori manat per l'enunciat, creem els fitxers amb la comanda **touch** i els seus noms corresponents.

PRA1.3 Mostreu una captura de pantalla amb el resultat d'executar la comanda `ip a` i tot seguit, escriviu quines interfícies hi ha, la seva MAC i la seva IP. Feu-ho en forma de taula.

```
milax@d114:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:10:21:01:14:00 brd ff:ff:ff:ff:ff:ff
    inet 10.112.201.14/21 metric 1024 brd 10.112.207.255 scope global dynamic enp0s3
        valid_lft 510sec preferred_lft 510sec
    inet6 fe80::210:21ff:fe01:1400/64 scope link
        valid_lft forever preferred_lft forever
milax@d114:~$
```

INTERFACÇOS	MAC	IPv4 local	IPv6 local
lo	00:00:00:00:00:00	127.0.0.1	IPv6 no activada
enp0s3	00:10:21:01:14:00	10.112.201.14	fe80::210:21ff:fe01:1400/64

La interfície **lo** no te cap connexió degut a que sempre està present independentment si el equip te connexió o no.

PRA1.4 Indiqueu una forma per esbrinar la IP del vostre gateway. Adjunteu-ne una captura de pantalla.

```
milax@d114:~$ nmcli dev show "enp0s3" | grep IP4.GATEWAY
IP4.GATEWAY: 10.112.200.1
```

Amb aquesta comanda ens permet veure la informació únicament del gateway del IPv4, degut a que filtra la terminologia ja que hem utilitzat la comanda **grep**.

PRÀCTICA 2: Rendiment de la xarxa

PRA2.1 Descriu què fa la comanda següent: `ping -i 3 -c 4 8.8.8.8`

La comanda fa **ping** al servidor DNS de Google, el paràmetre **-i 3** fa que hi hagi una pausa de 3 segons entre paquet y paquet, el paràmetre **-c 4** fa que se enviïn 4 paquets.

```
milax@d114:~$ ping -i 3 -c 4 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=121 time=11.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=121 time=11.6 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=121 time=11.6 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=121 time=11.7 ms

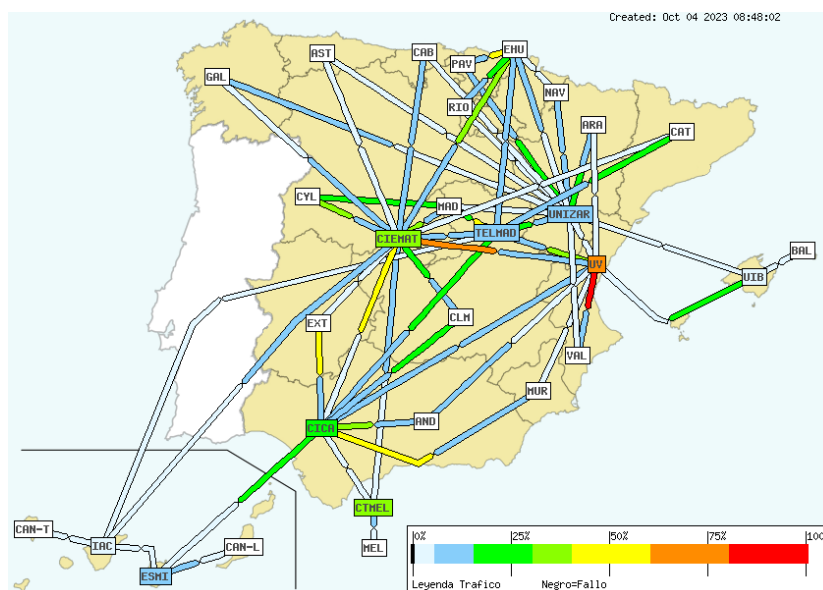
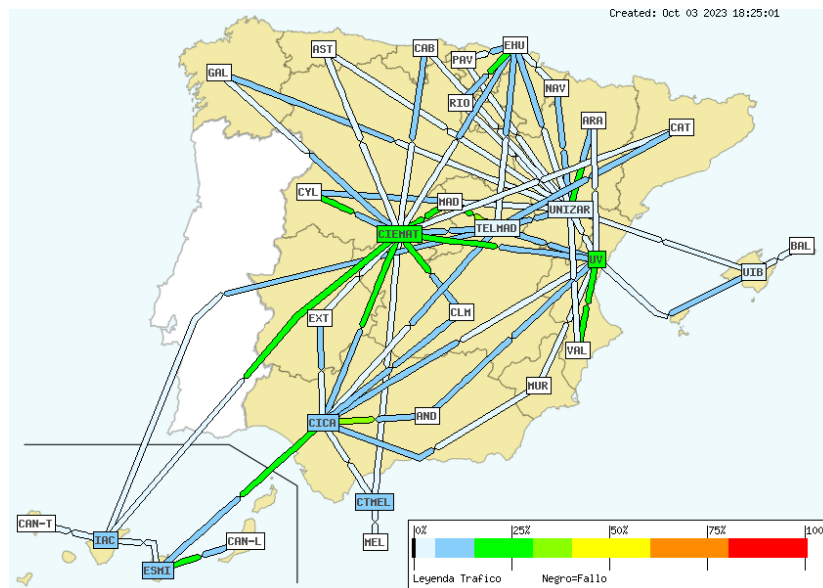
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 9009ms
rtt min/avg/max/mdev = 11.491/11.615/11.738/0.087 ms
```

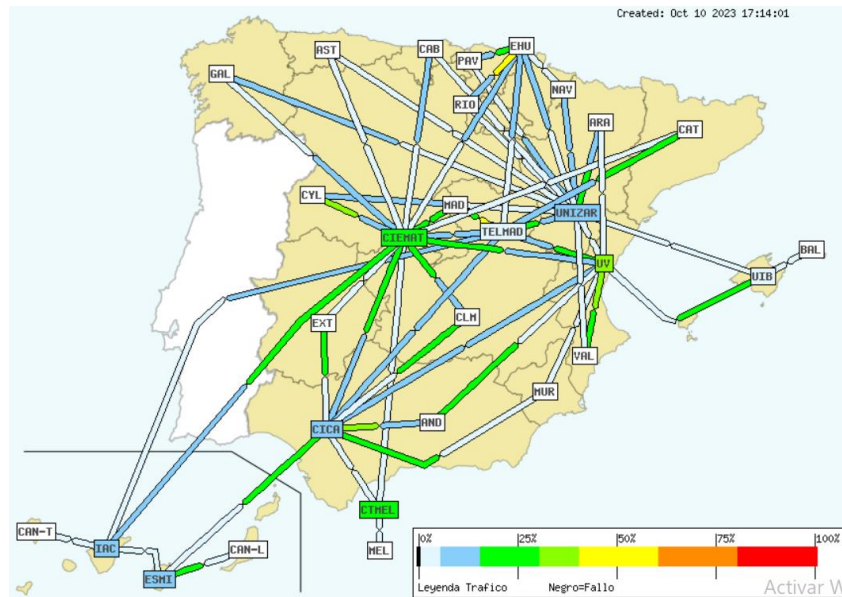
PRA2.2 Esteu desenvolupant una aplicació per una empresa de Reus per la qual necessiten accedir constantment per llegir dades d'un servidor remot o guardar-ne. El proveïdor de hosting amb el que treballeu us ofereix quatre servidors, ubicats en diferents ciutats del món. Indiqueu quina prova faríeu per seleccionar el servidor on l'aplicació tindrà les dades.

El procediment consistiria en primer crear les diferents maquines virtuals al proveïdor. Y posteriorment utilitzant l'eina **iperf** (que ens serveix per fer proves del rendiment de la xarxa) farem les proves.

Obtindrem la adreça IP del servidor y posteriorment executant la comanda **iperf -c** (en el costat del client) i **iperf -s** (en el costat del host o servidor) obtindrem els resultats. I així provant als 4 servidors que ens ofereix per trobar la millor opció.

PRA2.3 Consulteu el Weathermap de RedIRIS en diverses ocasions (al llarg del dia, al llarg de la setmana), i expliqueu el que observeu, si escau afegint alguna imatge.





La primera vegada que entrem en aquesta pàgina, veiem que on més tràfic de dades hi ha és en el centre CIEMAT ja que és un centre d'investigacions energètiques, mediambientals i tecnològiques, llavors es normal que l'ús de la internet sigui més alt que els altres nodes.

Després un dia vam veure que hi havia un alt tràfic de dades en la Universitat de València. En concret pel que sembla del centre del CIEMAT cap a la Universitat de Valencia. Creiem que això es pot duu a terme que en aquell moment es podrien fer una transferència de dades o un manteniment que ocasionés que la xarxa pogués estar operant amb una capacitat limitada. Això o que en aquell moment hi hagués una quantitat de clients més gran del esperat (inscripcions a aules, matriculacions o altres causes...).

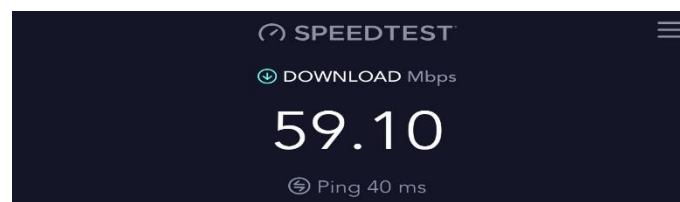
Una setmana després de haver començat a fer el seguiment del tràfic em vist que el tràfic es similar al primer dia, llavors es podria dir que es el dia a dia del CIEMAT que sigui el que més ús d'internet té.

PRA2.4 En aquesta activitat, cada membre de l'equip farà diferents testos de velocitat amb Speedtest. En concret: des del mòbil amb apli i usant la xarxa mòbil, des d'un portàtil connectat a la Wi-Fi de casa i, si pot ser, des d'un ordinador connectat per cable a casa. Empleneu la taula següent (mostrem dades d'exemple) i extraieu conclusions:

Nom	Connexió mòbil			WI-FI			Cablejat		
	Tecno.	Mbps	Retard	Tecno.	Mbps	Retard	Tecno.	Mbps	Retard
Ivan	5G	34.82	10 ms	802.11 ac (5GHz)	567.79	8ms	Gigabit Ethernet	571.94	5ms
Miquel	5G	59.1	46.5 ms	802.11 ax (5GHz)	458.64	5 ms	Gigabit Ethernet	947.28	4 ms
Javier	5G	43.20	10 ms	802.11 ac (5GHz)	274.54	5 ms	Gigabit Ethernet	304.84	4 ms

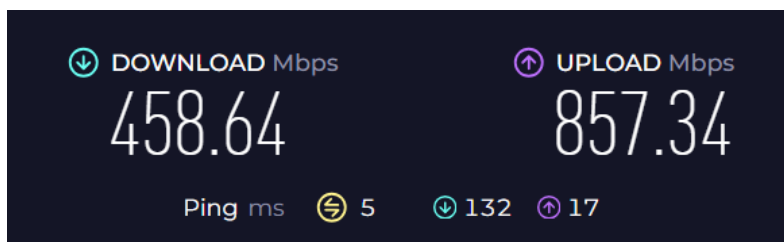
***Tecno. : Tecnologia *Mbps : Mbps de baixada**

Connexió mòbil



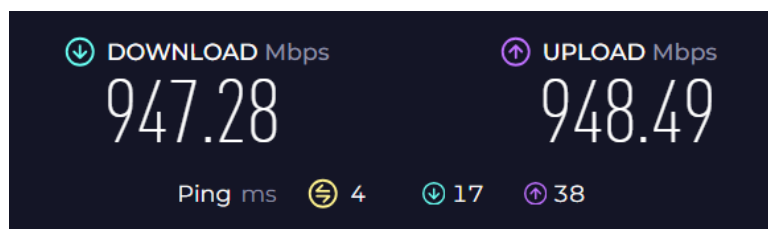
La variació es degut a que cadascun tenim equips diferents connectats a diferents ISP (operadores) això pot variar degut a que hi ha operadores que no utilitzen els mateixos punts de connexió i que també poden variar degut a que cada mòbil te unes antenes amb capacitats diferents, algunes poden captar diferents freqüències, altres tenen millor capacitat a la hora de rebre cobertura etc...

WI-FI



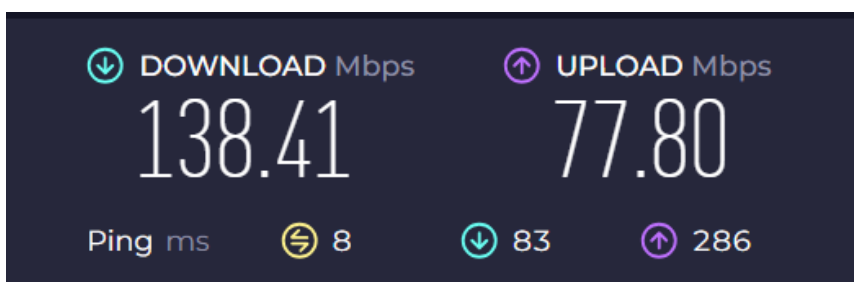
Al igual que les connexions mòbils també varien degut a lo que tenim contractat, també com esta fet la instal·lació wifi, ja que uns tenim routers amb diferents estàndards Wifi (4,5 o 6), si estem en una xarxa que és 2,4 GHz o 5 GHz i també les condicions de les cases que pot fer que la connexió sigui diferent.

Ethernet



Es pot deure els canvis també als cables ja que hi ha diferents tipus de cables que tenen mes o menys capacitat de transferència, al igual que la targeta de xarxa que tenen integrada els equips que també pot fer que hi hagi una diferencia.

PRA2.5 Afegiu el resultat de la prova Speedtest que heu fet al laboratori.



PRÀCTICA 3: Comunicació entre aplicacions

PRA3.1 Expliqueu quina aplicació heu programat, indicant si en tot el grup de laboratori heu fet la mateixa o no.

Una aplicació basada en [Wordle](#) .

Però amb números de 5 dígit. El servidor crea un numero aleatori de 5 dígit, y li demana al client que endevini el numero. El client tindrà 5 oportunitats, per cada vegada que el client envií un numero y no sigui el correcte el servidor donarà una pista mostrant si algun dels números que el client ha introduït esta a la posició correcta del numero a endevinar la resta de números que no siguin correctes es mostraran amagats amb un #.

En el nostre grup de laboratori cadascun ha fet una aplicació diferent.

PRA3.2 Descriviu els missatges del protocol i llur format.

Aquestes estructures (struct sockaddr_in) s'utilitzaran per emmagatzemar les adreces del servidor i del client.

El primer argument, AF_INET, indica que el socket s'utilitzarà per a adreces IPv4. El segon argument, SOCK_DGRAM, indica que el socket no esta orientat a la connexió. El tercer argument, 0, és un paràmetre de protocol que normalment es configura a 0.

El serv_adr.sin_family estableix la família d'adreces de l'adreça del servidor a AF_INET. Això és necessari perquè el servidor utilitzarà adreces IPv4.

serv_adr.sin_addr.s_addr estableix l'adreça IP de l'adreça del servidor a INADDR_ANY. Això significa que el servidor escoltarà connexions entrants a totes les seves adreces IP.

serv_adr.sin_port estableix el número de port de l'adreça del servidor a 44444. Aquest és el port que el servidor escoltarà connexions entrants.

En resum, aquestes línies de codi creen un socket i configuren l'adreça del servidor. El servidor escoltarà connexions entrants al port 44444 i acceptarà connexions des de qualsevol adreça IP.

```
recvfrom(s, buffer, MIDA_BUFFER, 0, (struct sockaddr *)&client_adr, &mida);
```

Aquesta línia permet al servidor rebre un missatge del client. El missatge serà emmagatzemat en el buffer. La mida del buffer és MIDA_BUFFER bytes. Els flags de recepció són 0, que indica que el servidor només vol rebre un missatge. L'adreça del client serà emmagatzemada en l'estructura client_adr. La mida de l'adreça del client serà emmagatzemada en la variable mida.

```
sendto(s, resultado, MIDA_BUFFER, 0, (struct sockaddr *)&client_adr, sizeof(client_adr));
```

Aquesta línia permet al servidor enviar un missatge al client. El missatge a enviar és resultado. La mida del missatge és MIDA_BUFFER bytes. Els flags d'enviament són 0, que indica que el servidor només vol enviar un missatge. L'adreça del client és la mateixa que la que es va utilitzar per rebre el missatge. La mida de l'adreça del client és la mateixa que es va utilitzar per rebre el missatge.

PRA3.3 Afegiu el codi font de client i servidor (o un únic codi si així ho heu decidit) degudament comentat i ben indentat. Perquè el codi quedi bé, utilitzeu un tipus de lletra monoespaiat.

Tanmateix, alguns IDE ja permeten fer “copy & paste” amb aquest format inclús preservant el ressaltat de sintaxi.

```

/* CLIENT */

/* Ficheros .h habituales */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/* Ficheros .h para los sockets */
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <errno.h>
#define MIDA_BUFFER 1024    /* Tamaño buffer */

int main(int argc, char **argv) {
    if (argc != 2) {
        printf("%s IP\n", argv[0]);
        exit(0);
    }

    int s;                                /* Para trabajar con el socket */
    struct sockaddr_in adr;               /* Dirección y puerto donde el cliente envía el paquete */
    char buffer[MIDA_BUFFER];            /* Variable para almacenar el buffer */
    socklen_t mida;                      /* Longitud de los datos de la dirección y el puerto */
    int i;

    /* Queremos socket de internet y no orientado a la conexión */
    s = socket(AF_INET, SOCK_DGRAM, 0);

    adr.sin_family = AF_INET;            /* Socket a Internet */
    adr.sin_port = htons(44444);         /* Puerto donde estará escuchando el servidor */

```

```

adr.sin_addr.s_addr = inet_addr(argv[1]);    /* Puerto donde escucha el servidor (IP) */

for (i = 0; i < 5; i++){ /* 5 intentos */
    int numeroAdivinar, cifras;
    char numero[5];
    printf("Introduce tu intento #%d (número de 5 cifras): ", i + 1);
    scanf("%s", &numero);                /* leer el numero como string */
    cifras = strlen(numero);              /* contar las cifras incluido el 0 por la izquierda */
    numeroAdivinar = atoi(numero);        /* convertir el string en int con 'atoi' */
    if (cifras != 5)
    {
        printf("Número no válido. Debe ser de 5 cifras.\n");
        i--; // Repetir el intento
        continue;
    }
    sprintf(buffer, "%05d", numeroAdivinar); /* Guardamos el número en el buffer */

    /* Enviamos número del usuario */
    sendto(s, buffer, MIDA_BUFFER, 0, (struct sockaddr *)&adr, sizeof(adr));
/* Recibimos el resultado */
    recvfrom(s, buffer, MIDA_BUFFER, 0, (struct sockaddr *)&adr, &mida);
    char c = '#';
    char *ptr = strchr(buffer, c);        /* Comparamos resultado con el número para adivinar */

    if (ptr != NULL) { /* Si tiene algun '#' */
        printf("Intento %d: No has adivinado el número. Cifras correctas: ", i + 1);
        for (int j = 0; j < 5; j++) {
            if (buffer[j] != '#') {
                printf("%c", buffer[j]); /* Imprime el resultado con las cifras correctas */
            }
        }
        else {

```

```

        printf("#");          /* Imprime las cifras incorrectas (#) */
    }
}
printf("\n");
    }
    else {                    /* Si no tiene ningún '#' */
        printf("¡Adivinaste el número! Número adivinado: %s\n", buffer);
break;
    }
}
if (i == 5) {                /* Condición fin de juego */
    printf("No adivinaste el número en 5 intentos.\n");
}
close(s);
return 0;
}

```

```

/* SERVIDOR */

/* Ficheros .h habituales */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/* Ficheros .h para los sockets */
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <netdb.h>
#include <errno.h>
#include <time.h>

#define MIDA_BUFFER 1024          /* Tamaño buffer */

int main() {
    int s;                        /* Para trabajar con el socket */
    struct sockaddr_in serv_adr;  /* Socket donde escucha servidor */
    struct sockaddr_in client_adr; /* Dirección y puerto donde el cliente envía el paquete */
    char buffer[MIDA_BUFFER];     /* Variable para almacenar el buffer */
    socklen_t mida;               /* Longitud de los datos de la direccion y el puerto */
    int i;
    mostrarIP();                 /* Función para mostrar la IP */

    /* Inicializa el generador de números aleatorios con una semilla aleatoria */
    srand(time(NULL));

```

```

/* Queremos socket de internet y no orientado a la conexión */
s = socket(AF_INET, SOCK_DGRAM, 0);

/* Ponemos los datos del socket del servidor */
serv_adr.sin_family = AF_INET;          /* Socket a Internet */
serv_adr.sin_addr.s_addr = INADDR_ANY;  /* Cualquier NIC */
serv_adr.sin_port = htons(44444);       /* Puerto donde estará escuchando el servidor */

/* Enlazamos el socket */
int resultat_bind;
resultat_bind = bind(s, (struct sockaddr*)&serv_adr, sizeof(serv_adr));
if (resultat_bind < 0) {
    printf("No se pudo enlazar el socket\n");
    close(s);
    return 0;
}

/* Servidor operativo! */
printf("Servidor operativo!\n");

/* Genera un número aleatorio de 5 cifras para adivinar */
int numeroAdivinar = 10000 + rand() % 90000;
i = 0;
printf("Número a adivinar: %d\n", numeroAdivinar);

while (i < 5) {          /* 5 intentos */
    /* Recibimos el número del usuario */
    recvfrom(s, buffer, MIDA_BUFFER, 0, (struct sockaddr*)&client_adr, &mida);
    int numeroAdivinado;
    sscanf(buffer, "%d", &numeroAdivinado);          /* Guardamos el número en una variable */
    char resultado[6];          /* Para almacenar el resultado, 5 cifras + '\0' */

```



```

        if (numeroAdivinado == numeroAdivinar) {           /* Condición de fin de juego */
/* Guardamos el número en char para poder compararlos */
        snprintf(resultado, sizeof(resultado), "%05d", numeroAdivinado);

        /* Enviamos el número */

        sendto(s, resultado, MIDA_BUFFER, 0, (struct sockaddr *)&client_adr, sizeof(client_adr));

        printf("¡Adivinaste el número!\n");

        break;
    }

    else{

        snprintf(resultado, sizeof(resultado), "#####"); /* Número sin ninguna cifra correcta */

        /* Verifica las cifras en la misma posición y reemplaza solo los incorrectos */

        int adivinadoTmp = numeroAdivinado;
        int adivinarTmp = numeroAdivinar;

        for (int j = 4; j >= 0; j--) {

            if (adivinadoTmp % 10 == adivinarTmp % 10) {

                resultado[j] = adivinadoTmp % 10 + '0';    /* Cambiamos el '#' por las cifras correctas */

            }

            adivinadoTmp /= 10;

            adivinarTmp /= 10;

        }

        /* Enviamos el número en el buffer */

        sendto(s, resultado, MIDA_BUFFER, 0, (struct sockaddr *)&client_adr, sizeof(client_adr));

        printf("Intento %d: %s\n", i + 1, resultado);

    }

    i++;           /* Incrementamos bucle (intentos) */

}

if (i == 5) {           /* Condición de fin de juego */

    printf("No adivinaste el número. El número era %d\n", numeroAdivinar);

}

close(s);

```

```

return 0;
}

void mostrarIP() { /* Función para mostrar la IP del host */
    char hostbuffer[100];
    char *IPbuffer;
    struct hostent *host_entry;
    int hostname;

    hostname = gethostname(hostbuffer, sizeof(hostbuffer));
    host_entry = gethostbyname(hostbuffer);
    IPbuffer = inet_ntoa(*(struct in_addr *)host_entry->h_addr_list[0]);
    printf("Hostname: %s\t", hostbuffer);
    printf("Host IP: %s\n", IPbuffer);
}

```

PRA3.4 Descriviu breument les proves que heu fet per comprovar que tot funciona segons el previst. Què succeeix si el client funciona i envia un paquet al servidor sense que aquest estigui actiu?

Primer que tot ens assegurem que connecta donant un missatge (Servidor operatiu) , si no dona resposta, el client s’engega, però com que no hi ha receptor, el client es queda a la espera de que el servidor li doni una resposta.

```

milax@casa: ~/Escriptori/3o/XD/P3/AcertarNumero
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
Hostname: casa Host IP: 127.0.1.1
Servidor operativo!
Número a adivinar: 29562
└─┘

milax@casa: ~/Escriptori/3o/XD/P3/AcertarNumero
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./client 127.0.1.1
Introduce tu intento #1 (numero de 5 cifras):

```

Una vegada amb el servidor operatiu, l’usuari introdueix un número per jugar, si no hi ha cap xifra correcta, el servidor imprimeix i envia el buffer amb “#####”.

```
milax@casa: ~/Escriptori/3o/XD/P3/AcertarNumero
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
Hostname: casa Host IP: 127.0.1.1
Servidor operativo!
Número a adivinar: 29562
Intento 1: #####

milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./client 127.0.1.1
Introduce tu intento #1 (número de 5 cifras): 00000
Intento 1: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #2 (número de 5 cifras):
```

Si l'usuari introdueix un número de xifres diferent a 5 (5 es per defecte per a que hi hagi un 50% de probabilitat de guanyar) es mostra un missatge d'avís per a que obligui al usuari a introduir un número de 5 xifres sense incrementar el nombre d'intents.

```
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
Hostname: casa Host IP: 127.0.1.1
Servidor operativo!
Número a adivinar: 29562
Intento 1: #####

milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./client 127.0.1.1
Introduce tu intento #1 (número de 5 cifras): 00000
Intento 1: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #2 (número de 5 cifras): 222
Número no válido. Debe ser de 5 cifras.
Introduce tu intento #2 (número de 5 cifras):
```

Si l'usuari encerta algunes xifres, el servidor imprimeix i envia el buffer amb les xifres correctes i incorrectes.

```
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
Hostname: casa Host IP: 127.0.1.1
Servidor operativo!
Número a adivinar: 29562
Intento 1: #####
Intento 2: 2###2

milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./client 127.0.1.1
Introduce tu intento #1 (número de 5 cifras): 00000
Intento 1: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #2 (número de 5 cifras): 2222
Número no válido. Debe ser de 5 cifras.
Introduce tu intento #2 (número de 5 cifras): 22222
Intento 2: No has adivinado el número. Cifras correctas: 2###2
Introduce tu intento #3 (número de 5 cifras):
```

Si l'usuari encerta les 5 xifres, surt en les dues terminals un missatge dient que ha guanyat i es finalitza el joc (connexió).

```
milax@casa: ~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
Hostname: casa Host IP: 127.0.1.1
Servidor operativo!
Número a adivinar: 29562
Intento 1: #####
Intento 2: 2###2
¡Adivinaste el número!
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$

milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./client 127.0.1.1
Introduce tu intento #1 (número de 5 cifras): 00000
Intento 1: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #2 (número de 5 cifras): 2222
Número no válido. Debe ser de 5 cifras.
Introduce tu intento #2 (número de 5 cifras): 22222
Intento 2: No has adivinado el número. Cifras correctas: 2###2
Introduce tu intento #3 (número de 5 cifras): 29562
¡Adivinaste el número! Número adivinado: 29562
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$
```

Si l'usuari no encerta el número en 5 intents (5 es per defecte per a que hi hagi un 50% de probabilitat de guanyar), surt un missatge de que no ha guanyat i finalitza el joc (connexió).

```
milax@casa: ~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./server
Hostname: casa Host IP: 127.0.1.1
Servidor operativo!
Número a adivinar: 33193
Intento 1: #####
Intento 2: #####
Intento 3: #####
Intento 4: #####
Intento 5: #####
No adivinaste el número. El número era 33193
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$

milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$ ./client 127.0.1.1
Introduce tu intento #1 (número de 5 cifras): 55555
Intento 1: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #2 (número de 5 cifras): 55555
Intento 2: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #3 (número de 5 cifras): 55555
Intento 3: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #4 (número de 5 cifras): 55555
Intento 4: No has adivinado el número. Cifras correctas: #####
Introduce tu intento #5 (número de 5 cifras): 55555
Intento 5: No has adivinado el número. Cifras correctas: #####
No adivinaste el número en 5 intentos.
milax@casa:~/Escriptori/3o/XD/P3/AcertarNumero$
```


- Feu una captura amb Wireshark i carregueu la pàgina “rectorat.html”. Atureu la captura i ara filtreu per protocol “tcp”.

39	4.347711571	10.112.201.8	10.112.201.9	TCP	74	54838 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3189151972 TSecr=0 WS=128
40	4.347723846	10.112.201.9	10.112.201.8	TCP	74	80 → 54838 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 MSS=1460 SACK_PERM TSval=3181575838 TSecr=3189151972 WS=128
41	4.347922388	10.112.201.8	10.112.201.9	TCP	66	54838 → 80 [ACK] Seq=1 Win=64256 Len=0 TSval=3189151972 TSecr=1815758318
43	4.348090993	10.112.201.9	10.112.201.8	TCP	66	80 → 54838 [ACK] Seq=1 Ack=305 Win=64800 Len=0 TSval=3181575839 TSecr=3189151972
45	4.348155549	10.112.201.8	10.112.201.9	TCP	66	80 → 54838 [ACK] Seq=1 Ack=305 Win=64800 Len=0 TSval=3181575839 TSecr=3189151972
47	4.348690525	10.112.201.9	10.112.201.8	TCP	7306	80 → 54838 [PSH, ACK] Seq=429 Ack=688 Win=64640 Len=7240 TSval=1815750367 TSecr=3189152011 [TCP segment of a reassembled PDU]
48	4.3486903249	10.112.201.9	10.112.201.8	TCP	7306	80 → 54838 [PSH, ACK] Seq=7669 Ack=688 Win=64640 Len=7240 TSval=1815750367 TSecr=3189152011 [TCP segment of a reassembled PDU]
49	4.347703855	10.112.201.8	10.112.201.9	TCP	66	54838 → 80 [ACK] Seq=688 Ack=18565 Win=62592 Len=0 TSval=3189152011 TSecr=1815758367
50	4.3477214299	10.112.201.9	10.112.201.8	TCP	11650	80 → 54838 [PSH, ACK] Seq=14909 Ack=688 Win=64640 Len=11584 TSval=1815750368 TSecr=3189152011 [TCP segment of a reassembled PDU]
52	4.347566842	10.112.201.8	10.112.201.9	TCP	66	54838 → 80 [ACK] Seq=688 Ack=14909 Win=64128 Len=0 TSval=3189152011 TSecr=1815758367
53	4.347466137	10.112.201.8	10.112.201.9	TCP	66	54838 → 80 [ACK] Seq=688 Ack=25845 Win=64128 Len=0 TSval=3189152012 TSecr=1815758368
54	4.347558087	10.112.201.8	10.112.201.9	TCP	66	54838 → 80 [ACK] Seq=688 Ack=27788 Win=64128 Len=0 TSval=3189152012 TSecr=1815758368
85	14.414547268	10.112.201.8	10.112.201.9	TCP	66	TCP Keep-Alive! 54838 → 80 [ACK] Seq=687 Ack=27788 Win=64128 Len=0 TSval=3189152044 TSecr=1815758368
86	14.414578496	10.112.201.9	10.112.201.8	TCP	66	TCP Keep-Alive! ACK 80 → 54838 [ACK] Seq=27788 Ack=688 Win=64640 Len=0 TSval=1815768395 TSecr=3189152012

- Intenteu esbrinar quants paquets ha necessitat la descàrrega de la imatge “rectorat.jpeg”.

Sembla que hi ha 4 TCP involucrats a la descarrega de la imatge, el qual els dos primers divideixen la imatge en dos paquets. El tercer es el paquet que envia el client per confirmar que ha rebut la imatge en aquest cas. Els paquets s’envien per el port 54838 i després el client rep els paquets i el propi navegador els junta.

46	4.386785657	10.112.201.8	10.112.201.9	HTTP	389	GET /webxd/rectorat.jpg HTTP/1.1
47	4.386890263	10.112.201.9	10.112.201.8	TCP	7306	80 → 54838 [PSH, ACK] Seq=429 Ack=688 Win=64640 Len=7240 TSval=1815750367 TSecr=3189152011 [TCP segment of a reassembled PDU]
48	4.386903249	10.112.201.9	10.112.201.8	TCP	7306	80 → 54838 [PSH, ACK] Seq=7669 Ack=688 Win=64640 Len=7240 TSval=1815750367 TSecr=3189152011 [TCP segment of a reassembled PDU]
49	4.387203655	10.112.201.8	10.112.201.9	TCP	66	54838 → 80 [ACK] Seq=688 Ack=18565 Win=62592 Len=0 TSval=3189152011 TSecr=1815758367
50	4.387214299	10.112.201.9	10.112.201.8	TCP	11650	80 → 54838 [PSH, ACK] Seq=14909 Ack=688 Win=64640 Len=11584 TSval=1815750368 TSecr=3189152011 [TCP segment of a reassembled PDU]

PRA4.2 Expliqueu com ha funcionat el mecanisme de cache.

La memòria cache serveix per a que el navegador web no hagi de descarregar tots els fitxers d'una pàgina web cada vegada que es visita. Això fa que la pàgina web es carregui més ràpid.

En el nostre cas fa que tinguem que rebre menys peticions GET, ja que disposàvem part del codi i les imatges.

PRA4.3 Mostreu una captura de pantalla amb la web “equip.html” modificada.

Web Xarxes de Dades

Components de l'equip

A continuació mostrem el nom dels membres de l'equip:

- Josep Miquel Santander Rodriguez
- Ivan Morillas Gomez
- Javier Fuentes Abalo

PRÀCTICA 5: Protocols i Wireshark

PRA5.1 Sobre el protocol ARP, responeu:

a) Per què la vostra màquina rep tantes peticions ARP? Passa el mateix amb la resta de companys i companyes?

Quan un dispositiu nou es connecta a una xarxa, no te la taula ARP plena, llavors tindrà que enviar peticions ARP per resoldre les direcció IP dels altres dispositius en la xarxa.

b) En general, quina és l'adreça MAC destí de les peticions ARP que heu capturat?

ff:ff:ff:ff:ff:ff:

c) Hi ha alguna resposta ARP que vagi dirigida a la vostra NIC explícitament? Si

escau, feu una captura de pantalla que la mostri. A què es pot deure aquesta resposta ARP?

23122	1099.1044914...	Apple_07:7e:3c	Broadcast	ARP	60 ARP Announcement for 10.112.205.4
23125	1100.2389498...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.6? Tell 10.112.204.8
23126	1100.7703811...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.6? Tell 10.112.204.8
23127	1101.4551653...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.13? Tell 10.112.204.5
23128	1101.7705550...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.6? Tell 10.112.204.8
23129	1102.2990622...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.13? Tell 10.112.204.5
23131	1103.2990058...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.13? Tell 10.112.204.5
23225	1111.4700960...	ASUSTekC 5f:7b:40	Broadcast	ARP	60 Who has 10.112.200.216? Tell 10.112.206.202
23237	1121.7545921...	Apple_01:1b:ec	Broadcast	ARP	60 ARP Announcement for 10.112.205.10
23249	1125.7698949...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.4? Tell 10.112.204.8
23251	1127.2987733...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.11? Tell 10.112.204.5
23255	1128.6621437...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.4? Tell 10.112.204.8
23256	1129.0451193...	CaradonT_c8:d3:00	EncantoN_01:09:00	ARP	60 Who has 10.112.201.9? Tell 10.112.200.211
23257	1129.0451681...	EncantoN 01:09:00	CaradonT_c8:d3:00	ARP	42 10.112.201.9 is at 00:10:21:01:09:00
23258	1129.0654847...	EncantoN 01:09:00	CaradonT_c8:d4:00	ARP	60 Who has 10.112.201.9? Tell 10.112.200.212
23259	1129.0654888...	EncantoN 01:09:00	CaradonT_c8:d4:00	ARP	42 10.112.201.9 is at 00:10:21:01:09:00
23260	1129.2462561...	EncantoN 01:09:00	CaradonT_c8:d3:00	ARP	42 Who has 10.112.200.211? Tell 10.112.201.9
23261	1129.2464420...	CaradonT_c8:d3:00	EncantoN_01:09:00	ARP	60 10.112.200.211 is at 00:10:70:c8:d3:00
23262	1129.2702466...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.4? Tell 10.112.204.8
23263	1129.8461314...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.11? Tell 10.112.204.5
23264	1130.2697898...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.4? Tell 10.112.204.8
23266	1130.7876460...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.11? Tell 10.112.204.5
23271	1131.7992239...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.11? Tell 10.112.204.5
23278	1132.6769070...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.4? Tell 10.112.204.8
23279	1133.2705431...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.4? Tell 10.112.204.8
23280	1133.8469327...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.11? Tell 10.112.204.5
23285	1134.2698142...	EncantoN 04:08:00	Broadcast	ARP	60 Who has 10.112.204.4? Tell 10.112.204.8
23287	1134.7999568...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.11? Tell 10.112.204.5
23288	1135.7997990...	EncantoN 04:05:00	Broadcast	ARP	60 Who has 10.112.204.11? Tell 10.112.204.5
23289	1136.2734432...	Fortinet 90:67:1c	EncantoN_01:09:00	ARP	60 Who has 10.112.201.9? Tell 10.112.200.1
23290	1136.2735596...	EncantoN 01:09:00	Fortinet_90:67:1c	ARP	42 10.112.201.9 is at 00:10:21:01:09:00

Aquesta resposta ARP es pot deure a diverses coses. Una possibilitat és que la meua NIC hagi enviat una sol·licitud ARP per a un determinat adreça IP. En aquest cas, el servidor DHCP, que és el responsable d'assignar adreces IP als dispositius de la xarxa, hauria enviat aquesta resposta ARP per a proporcionar-me la direcció MAC del dispositiu que té l'adreça IP que vaig sol·licitar.

PRA5.2 Sobre el protocol ICMP, responeu:

a) Quina és l'adreça IP de l'ordinador des d'on heu fet la prova?

10.112.201.9

b) Filtreu els missatges del protocol ICMP i feu-ne una captura de pantalla.

20782	598.699813976	10.112.201.9	10.112.200.211	ICMP	370	Destination unreachable (Port unreachable)
23246	1123.9946499	10.112.201.9	10.112.200.211	ICMP	370	Destination unreachable (Port unreachable)
23311	1145.8877709	10.112.201.9	10.45.1.2	ICMP	107	Destination unreachable (Port unreachable)
23328	1145.9272696	10.112.201.9	10.45.1.2	ICMP	107	Destination unreachable (Port unreachable)
23410	1146.1871197	10.112.201.9	10.45.1.2	ICMP	146	Destination unreachable (Port unreachable)
24678	1146.5707846	10.112.201.9	10.45.1.2	ICMP	172	Destination unreachable (Port unreachable)
24766	1153.3850487	10.112.201.9	10.45.1.2	ICMP	98	Destination unreachable (Port unreachable)
24800	1153.5749587	10.112.201.9	10.45.1.2	ICMP	102	Destination unreachable (Port unreachable)
24931	1154.0255906	10.112.201.9	10.45.1.2	ICMP	103	Destination unreachable (Port unreachable)
25017	1154.0697573	10.112.201.9	10.45.1.2	ICMP	105	Destination unreachable (Port unreachable)
25398	1154.3006121	10.112.201.9	10.45.1.2	ICMP	124	Destination unreachable (Port unreachable)
25840	1158.0273572	10.112.201.9	10.45.1.2	ICMP	123	Destination unreachable (Port unreachable)

c) Quin tipus de missatges ICMP es fan servir en un ping?

129	22.618505829	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request	id=0x8bb6, seq=1/256, ttl=64 (reply in 130)
130	22.629299030	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply	id=0x8bb6, seq=1/256, ttl=121 (request in 129)
132	23.620494256	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request	id=0x8bb6, seq=2/512, ttl=64 (reply in 133)
133	23.631269843	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply	id=0x8bb6, seq=2/512, ttl=121 (request in 132)
138	24.622382324	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request	id=0x8bb6, seq=3/768, ttl=64 (reply in 139)
139	24.633127418	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply	id=0x8bb6, seq=3/768, ttl=121 (request in 138)
141	25.624253629	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request	id=0x8bb6, seq=4/1024, ttl=64 (reply in 142)
142	25.634824680	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply	id=0x8bb6, seq=4/1024, ttl=121 (request in 141)
148	26.626118100	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request	id=0x8bb6, seq=5/1280, ttl=64 (reply in 149)
149	26.636811882	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply	id=0x8bb6, seq=5/1280, ttl=121 (request in 148)
158	27.627938336	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request	id=0x8bb6, seq=6/1536, ttl=64 (reply in 159)
159	27.638722528	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply	id=0x8bb6, seq=6/1536, ttl=121 (request in 158)
165	28.630224870	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request	id=0x8bb6, seq=7/1792, ttl=64 (reply in 166)

d) Quina és la longitud en bytes de cada paquet?

No.	Time	Source	Destination	Protocol	Length	Info
129	22.618505829	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request id=0x8bb6, seq=1/256, ttl=64 (reply in 130)
130	22.629299030	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply id=0x8bb6, seq=1/256, ttl=121 (request in 129)
132	23.620494256	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request id=0x8bb6, seq=2/512, ttl=64 (reply in 133)
133	23.631269843	8.8.8.8	10.112.201.9	ICMP	98	Echo (ping) reply id=0x8bb6, seq=2/512, ttl=121 (request in 132)
138	24.622382324	10.112.201.9	8.8.8.8	ICMP	98	Echo (ping) request id=0x8bb6, seq=3/768, ttl=64 (reply in 139)
* Frame 132: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface enp0s3, id 0						
Section number: 1						
Interface id: 0 (enp0s3)						
Encapsulation type: Ethernet (1)						
Arrival Time: Nov 21, 2023 17:38:37.119670143 CET						
[Time shift for this packet: 0.000000000 seconds]						
Epoch Time: 1700584717.119670143 seconds						
[Time delta from previous captured frame: 0.421612167 seconds]						
[Time delta from previous displayed frame: 0.991195226 seconds]						
[Time since reference or first frame: 23.620494256 seconds]						
Frame Number: 132						
Frame Length: 98 bytes (784 bits)						
Capture Length: 98 bytes (784 bits)						
[Frame is marked: False]						
[Frame is ignored: False]						
[Protocols in frame: eth:ethertype:ip:icmp:data]						
[Coloring Rule Name: ICMP]						
[Coloring Rule String: icmp icmpv6]						

e) Quin camp del protocol ICMP serveix per relacionar cada enviament de ping amb

la seva resposta? Mostreu-ho amb una captura de pantalla dels detalls d'un paquet

on s'assenyali amb una fletxa aquesta informació.

```
► Frame 133: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface enp0s3, id 0
► Ethernet II, Src: Fortinet 90:67:1c (04:d5:90:90:67:1c), Dst: EncantoN_01:09:00 (00:10:21:01:09:00)
► Internet Protocol Version 4, Src: 8.8.8.8, Dst: 10.112.201.9
▼ Internet Control Message Protocol
  Type: 0 (Echo (ping) reply)
  Code: 0
  Checksum: 0xe45e [correct]
  [Checksum Status: Good]
  Identifier (BE): 35766 (0x8bb6)
  Identifier (LE): 46731 (0xb68b)
  Sequence Number (BE): 2 (0x0002)
  Sequence Number (LE): 512 (0x0200)
  [Request frame: 132]
  [Response time: 10,776 ms]
  Timestamp from icmp data: Nov 21, 2023 17:38:37.000000000 CET
  [Timestamp from icmp data (relative): 0.130445730 seconds]
► Data (48 bytes)
```


PRA5.3 Sobre el protocol DNS, responeu:

a) Quantes peticions DNS (query) heu capturat? Si n'hi ha moltes, intenteu esbrinar

per què se'n fan tantes. La pàgina web que heu visitat (www.etse.urv.cat) es

descarrega elements de diferents llocs d'Internet...

182	14.716041855	10.45.1.2	10.112.201.9	DNS	176 Standard query response 0x40e A etse.urv.cat CNWHE wms.urv.cat A 193.144.16.112 NS aliga.urv.cat NS dns.urv.cat A 193.144.16.4 A 193.144.16.9
119	14.724492233	10.45.1.2	10.112.201.9	DNS	74 Standard query response 0x14b8 HTTPS www.google.com
122	14.735720215	10.112.201.9	10.45.1.2	DNS	76 Standard query 0x01f1 A www.etse.urv.cat
123	14.735777229	10.112.201.9	10.45.1.2	DNS	76 Standard query 0x155d HTTPS www.etse.urv.cat
134	14.737106032	10.45.1.2	10.112.201.9	DNS	145 Standard query response 0x155d HTTPS www.etse.urv.cat CNWHE wms.urv.cat SDA dns.urv.cat
125	14.737802063	10.45.1.2	10.112.201.9	DNS	180 Standard query response 0x001f A www.etse.urv.cat CNWHE wms.urv.cat A 193.144.16.112 NS aliga.urv.cat NS dns.urv.cat A 193.144.16.4 A 193.144.16.9
225	15.621431865	10.112.201.9	10.45.1.2	DNS	80 Standard query 0x1252 A fonts.googleapis.com
226	15.621460364	10.112.201.9	10.45.1.2	DNS	80 Standard query 0x1006 HTTPS fonts.googleapis.com
230	15.623835365	10.45.1.2	10.112.201.9	DNS	223 Standard query response 0x1252 A fonts.googleapis.com A 142.250.206.138 NS f.root-servers.net NS h.root-servers.net NS m.root-servers.net NS a.ra-
236	15.630607849	10.45.1.2	10.112.201.9	DNS	80 Standard query response 0x0306 HTTPS fonts.googleapis.com
718	17.724818969	10.112.201.9	10.45.1.2	DNS	77 Standard query 0x0b7f A fonts.gstatic.com
726	17.724869363	10.112.201.9	10.45.1.2	DNS	77 Standard query 0x1fcd HTTPS fonts.gstatic.com
727	17.727511089	10.112.201.9	10.45.1.2	DNS	84 Standard query 0xc1d7 A www.googletagmanager.com
727	17.727613119	10.112.201.9	10.45.1.2	DNS	84 Standard query 0x0b8d HTTPS www.googletagmanager.com
727	17.727961546	10.45.1.2	10.112.201.9	DNS	320 Standard query response 0x0b7f A fonts.gstatic.com A 142.250.206.99 NS c.root-servers.net NS g.root-servers.net NS i.root-servers.net NS l.root-s-
735	17.73095262	10.45.1.2	10.112.201.9	DNS	327 Standard query response 0xc1d7 A www.googletagmanager.com A 142.250.164.168 NS h.root-servers.net NS l.root-servers.net NS d.root-servers.net NS -
782	17.740338413	10.45.1.2	10.112.201.9	DNS	77 Standard query response 0x1fcd HTTPS fonts.gstatic.com
791	17.742819052	10.45.1.2	10.112.201.9	DNS	84 Standard query response 0x0b8d HTTPS www.googletagmanager.com
843	17.796307060	10.112.201.9	10.45.1.2	DNS	80 Standard query 0x0579 A diaridigital.urv.cat
843	17.796307060	10.112.201.9	10.45.1.2	DNS	80 Standard query 0x0579 A diaridigital.urv.cat
843	17.797171469	10.45.1.2	10.112.201.9	DNS	121 Standard query response 0x057f HTTPS diaridigital.urv.cat SDA dns.urv.cat
846	17.797919947	10.45.1.2	10.112.201.9	DNS	160 Standard query response 0x0679 A diaridigital.urv.cat A 193.144.16.106 NS aliga.urv.cat NS dns.urv.cat A 193.144.16.4 A 193.144.16.9
1079	18.465818405	10.112.201.9	10.45.1.2	DNS	80 Standard query 0x0989 A reionit.ocoolo-analytics.com

b) Entre quines adreces IP es transmeten els paquets que porten protocol DNS? Perquè?

10.45.1.2(www.etse.cat) y 10.112.201.9(nostre ip)

PRA5.4 Sobre el protocol TCP, responeu:

a) En una captura de pantalla del llistat de paquets de Wireshark, encerclau-ne tres

que es corresponguin amb l'establiment de sessió.

181	15.619560752	10.112.201.9	193.144.16.112	TCP	74 50752 -> 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1223121344 TSecr=0 WS=128
182	15.619560912	10.112.201.9	193.144.16.112	TCP	74 50756 -> 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1223121344 TSecr=0 WS=128
183	15.619611391	193.144.16.112	10.112.201.9	TCP	66 443 -> 50756 [ACK] Seq=18054 Ack=2723 Win=7182 Len=0 TSval=4073412878 TSecr=1223121343
184	15.619711057	10.112.201.9	193.144.16.112	TCP	74 50772 -> 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1223121344 TSecr=0 WS=128
185	15.619743654	193.144.16.112	10.112.201.9	TCP	70 443 -> 50756 [SYN, ACK] Seq=0 Ack=1 Win=388 Len=0 MSS=1460 SACK_PERM TSval=4073412878 TSecr=1223121343
186	15.619766687	10.112.201.9	193.144.16.112	TCP	66 50752 -> 443 [ACK] Seq=0 Ack=1 Win=64240 Len=0 TSval=1223121344 TSecr=4073412878
187	15.619833806	193.144.16.112	10.112.201.9	TCP	70 443 -> 50756 [SYN, ACK] Seq=0 Ack=1 Win=388 Len=0 MSS=1460 SACK_PERM TSval=4073412879 TSecr=1223121344
188	15.619861613	10.112.201.9	193.144.16.112	TCP	66 50756 -> 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0 TSval=1223121344 TSecr=4073412879
189	15.619872633	10.112.201.9	193.144.16.112	TCP	74 50774 -> 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1223121344 TSecr=0 WS=128
190	15.62005949	10.112.201.9	193.144.16.112	TLSv1.2	583 Client Hello
191	15.620061155	193.144.16.112	10.112.201.9	TCP	70 443 -> 50772 [SYN, ACK] Seq=0 Ack=1 Win=388 Len=0 MSS=1460 SACK_PERM TSval=4073412879 TSecr=1223121344
192	15.620064918	10.112.201.9	193.144.16.112	TCP	66 50774 -> 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0 TSval=1223121344 TSecr=4073412879
193	15.620115248	10.112.201.9	193.144.16.112	TLSv1.2	583 Client Hello
194	15.620180748	193.144.16.112	10.112.201.9	TCP	70 443 -> 50774 [SYN, ACK] Seq=0 Ack=1 Win=388 Len=0 MSS=1460 SACK_PERM TSval=4073412879 TSecr=1223121344
195	15.620181000	10.112.201.9	193.144.16.112	TCP	66 50774 -> 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0 TSval=1223121344 TSecr=4073412879
196	15.620269911	10.112.201.9	193.144.16.112	TCP	74 50780 -> 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1223121344 TSecr=0 WS=128
197	15.620375971	10.112.201.9	193.144.16.112	TLSv1.2	583 Client Hello
198	15.620451109	193.144.16.112	10.112.201.9	TCP	66 443 -> 50752 [ACK] Seq=1 Ack=518 Win=4897 Len=0 TSval=4073412879 TSecr=1223121344
199	15.620452101	193.144.16.112	10.112.201.9	TLSv1.2	217 Server Hello, Change Cipher Spec, Encrypted Handshake Message
200	15.620460831	10.112.201.9	193.144.16.112	TCP	66 50752 -> 443 [ACK] Seq=518 Ack=152 Win=64889 Len=0 TSval=1223121345 TSecr=4073412879
201	15.620502405	10.112.201.9	193.144.16.112	TLSv1.2	583 Client Hello
202	15.620509241	193.144.16.112	10.112.201.9	TCP	66 443 -> 50756 [ACK] Seq=1 Ack=518 Win=4897 Len=0 TSval=4073412879 TSecr=1223121344
203	15.620510004	193.144.16.112	10.112.201.9	TLSv1.2	217 Server Hello, Change Cipher Spec, Encrypted Handshake Message
204	15.620517206	10.112.201.9	193.144.16.112	TCP	66 50756 -> 443 [ACK] Seq=518 Ack=152 Win=64889 Len=0 TSval=1223121345 TSecr=4073412879
205	15.620666663	193.144.16.112	10.112.201.9	TCP	70 443 -> 50780 [SYN, ACK] Seq=0 Ack=1 Win=388 Len=0 MSS=1460 SACK_PERM TSval=4073412879 TSecr=1223121344

b) Quina és la mida d'aquests segments? Porten dades o són només una capçalera?

SYN --> 74

SYN, ACK --> 70

ACK --> 66

Porten tant dades com capçaleres.

seqüència inicial a cada banda? Feu un esquema gràfic a la memòria.

PORT ORIGEN	PORT DESTI	IP ORIGEN	IP DESTI	
50772	443	10.112.201.9	193.144.16.112	SYN
443	50752	193.144.16.112	10.112.201.9	SYN ACK
50752	443	10.112.201.9	193.144.16.112	ACK

(per exemple, cadascuna ve d'un servidor web diferent), per què l'adreça MAC

L'adreça MAC origen de les dues trames és la mateixa perquè les dues trames estan passant per un enllaç de xarxa comú.

a) Quin o quins webs heu comprovat que tenen QUIC?

b) Mostreu, amb una captura de pantalla, els detalls d'un paquet QUIC. Sobre quina

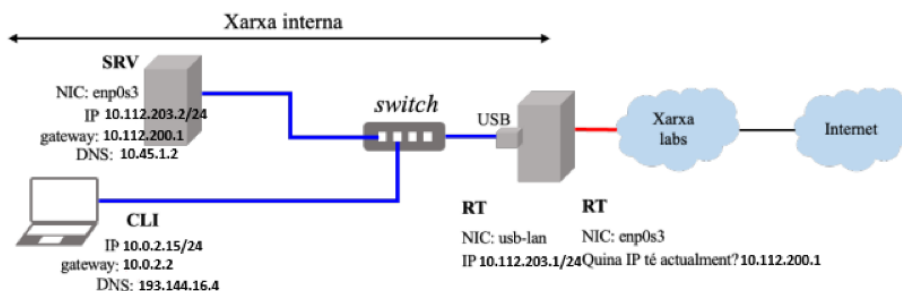
capa de transport funciona QUIC? Sobre quin port destí s'estableix la connexió?

QUIC funciona sobre la capa de transport UDP i funciona sobre el port 443.

[illegible]

PRAÈTICA 6: Muntatge d'una LAN

PRA6.1 Dibuixeu l'esquema de xarxa i assigneu les adreces IP, gateway i DNS que falten.



PRA6.2 Feu un petit resum de les diferents comandes que heu après en aquesta primera sessió.

```
milax@d109:~$ ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
   link/ether 00:10:21:01:09:00 brd ff:ff:ff:ff:ff:ff
   inet 10.112.201.9/21 metric 1024 brd 10.112.207.255 scope global dynamic enp0s3
       valid_lft 584sec preferred_lft 584sec
   inet6 fe80::210:21ff:fe01:900/64 scope link
       valid_lft forever preferred_lft forever
milax@d109:~$ ip route show default
default via 10.112.200.1 dev enp0s3 proto dhcp src 10.112.201.9 metric 100
default via 10.112.200.1 dev enp0s3 proto dhcp src 10.112.201.9 metric 1024
milax@d109:~$ sudo ip route del default via 10.112.200.1
[sudo] contrasenya per a milax:
milax@d109:~$ sudo ip route add default via 10.112.203.1
milax@d109:~$ ip route show default
default via 10.112.203.1 dev enp0s3
default via 10.112.200.1 dev enp0s3 proto dhcp src 10.112.201.9 metric 1024
milax@d109:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 10.112.201.9 icmp_seq=1 Destination Host Unreachable
From 10.112.201.9 icmp_seq=2 Destination Host Unreachable
From 10.112.201.9 icmp_seq=3 Destination Host Unreachable
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3088ms
pipe 4
milax@d109:~$ sudo ip route del default via 10.112.203.1
milax@d109:~$ sudo ip route add default via 10.112.200.1
milax@d109:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=121 time=10.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=121 time=10.8 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=121 time=10.9 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=121 time=10.9 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3059ms
rtt min/avg/max/mdev = 10.837/10.866/10.893/0.021 ms
milax@d109:~$
```

PRA6.3 D'acord amb l'exemple de configuració de tallafocs que hi ha en aquest

enunciat, podríeu fer ping a la màquina que fa de router? Per què?

Depèn de la configuració del fitxer *conf_TF.sh*. Si tenim configurat el trànsit del input i del output en *'drop'* no es podria fer ping, però si ho tenim configurat amb *'accept'*, eliminem la denegació de trànsit i ens habilitaria el poder fer ping.

PRA6.4 Indiqueu quins dels 8 objectius del muntatge heu assolit. Afegiu una fotografia del muntatge i, si escau, proves dels vostres assoliments.

Hem pogut assolir els següents punts:

El primer punt ja que hem muntat la infraestructura física de la xarxa amb tots els elements y les seves connexions.

El segon punt ja que hem pogut inhabilitar i habilitar el NetworkManager amb la comanda *sudo systemctl NetworkManager.service* i comprovant que ha funcionat amb el *ping 8.8.8.8*.

El tercer punt no ho hem pogut assolir al 100%, hem configurat el forwarding i el NAT a RT però sense saber el motiu no hi ha arribat Internet al Servidor, el nostre professor ha comprovat que tot està ben fet i podria ser algun error del ordinador del laboratori.



PRA6.5 Heu pogut connectar-vos a YouTube tot i permetre el trànsit HTTP i HTTPS?

No, ja que hem tingut problemes comentats en l'apartat anterior.

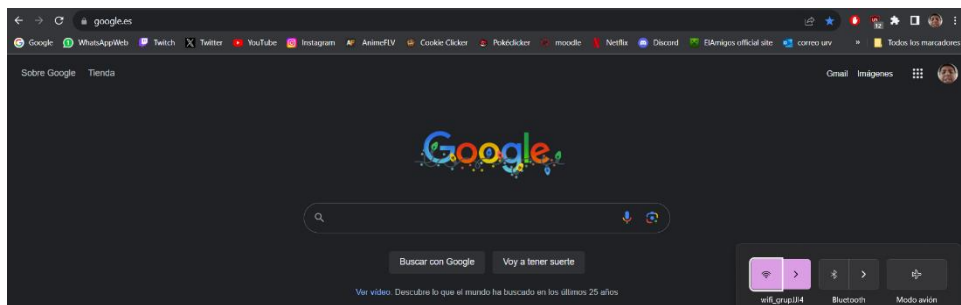
PRÀCTICA 7: Muntatge d'una LAN Wi-Fi

PRA7.1 Feu una captura de pantalla que demostrï que heu establert la vostra pròpia xarxa Wi-Fi. Haurà d'aparèixer el vostre SSID juntament amb altres, i l'Eduroam.

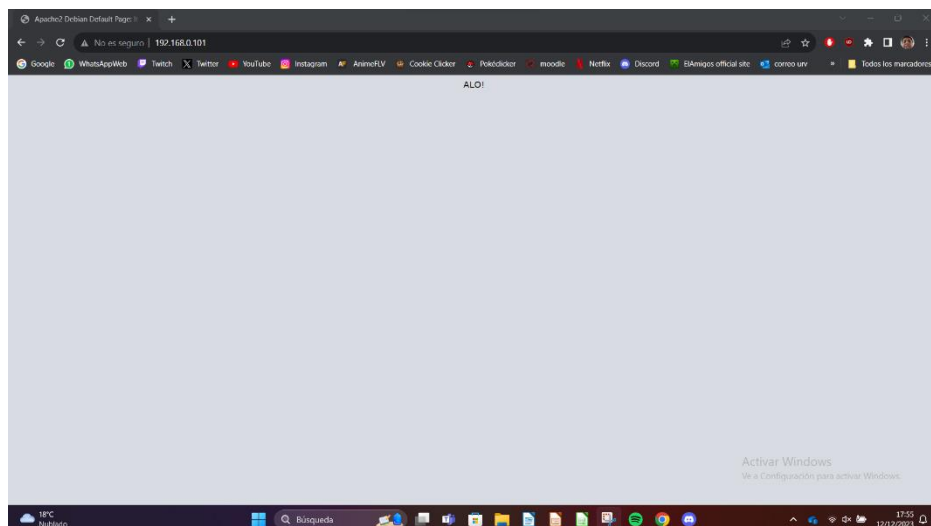
Wireless 2.4GHz

Operation Mode: **Router**
Wireless Radio: Enabled
Name(SSID): wifi_grupJl4
Mode: 11bgn mixed
Channel: Auto(Channel 10)
Channel Width: Auto
MAC Address: 90:9A:4A:74:9D:92

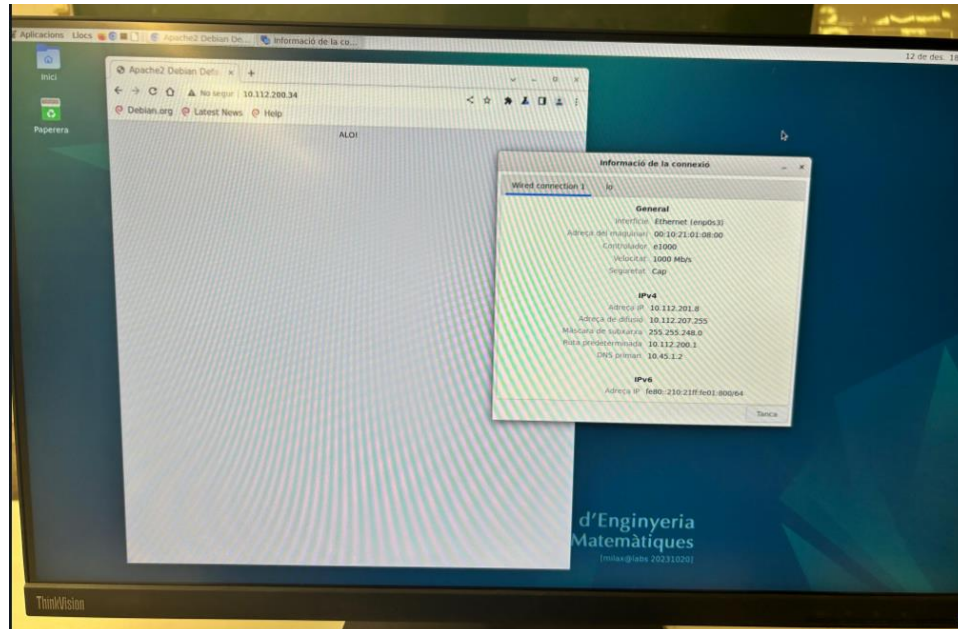
PRA7.2 Indiqueu quines proves heu fet per comprovar que des del client (el portàtil) podeu accedir a Internet.



PRA7.3 Mostreu una captura de pantalla que demostrï que des de CLI podeu accedir al web que conté SRV.



PRA7.4 Mostreu una fotografia que demostrï que des d'un altre ordinador podeu accedir al web de SRV. En la imatge s'ha de veure clarament la barra d'adreces del navegador, amb la IP de W-RT.



8. Conclusions

Em fet totes les practiques al laboratori, realitzant-les a temps i sense cap problema.

El professor del laboratori ens ha resolt alguns dubtes que hem tingut, fins a arribar a resoldre-ho.

En les dues practiques finals vam tenir un problema amb el procediment ja que un dels ordinadors no funcionava correctament , i vam tindre que recomençar de nou. Per la resta, no hi ha hagut cap problema.

Les pràctiques estaven ben estructurades i el guió era fàcil d'entendre, amb explicacions que ens resolien tots els dubtes.

No es necessitaria cap canvi ja que les practiques com hem mencionat abans estan totes correctes.

A part de que les practiques han sigut interessants i hem après molt sobre el tema.

9. Valoracions personals

Miquel: Jo he assistit a totes les classes (com tot el meu grup) , i en quant a la meva implicació al treball ha sigut molt semblant al conjunt grupal. Els 3 hem col·laborat molt i cadascú ha posat de la seva part. També ha sigut unes practiques interessants ja que tenia coneixements previs sobre el tema i he pogut també donar idees als meus companys, al igual que ells han donat les seves. 😎

Ivan: Ha sigut important assistir a totes les classes del laboratori ja que, amb el professor, he après molt fent les pràctiques amb els meus companys. He sigut molt aplicat en el treball ja que, si no ho hagués estat, no hauria après res i no em serviria tindre una assignatura sense tindre un coneixement mínim. 😊

Javier: Ha estat molt interessant treballar en grup d'aquesta manera, he pogut conèixer a dos amics nous, no he pogut assistir a alguna classe per que visc lluny del campus, però els meus companys ho han entès i hem anat treballant junts des del principi. 😊