

Tecnológico de Costa Rica.

Escuela de Ingeniería en Computación.

IC: 7602-Redes - 2 Semestre 2022.

2018093728 - Paula Mariana Bustos Vargas

2018086509 - Jocxan Sandi Batista.

---

# DOCUMENTACIÓN - TAREA CORTA 2

---

## Índice

- Servidor y calculadora
- Cliente
- Manual de usuario
- Especificación de carpetas
- Crear elementos
- Referencias

**NOTA:** EL PROGRAMA ES EJECUTADO EN UBUNTU.

## Servidor y calculadora

Se realizaron las funciones de la calculadora de redes de

- GET BROADCAST IP
- GET NETWORK NUMBER
- GET HOSTS RANGE
- GET RANDOM SUBNETS

Para la implementacion de estas funciones se tubieron que crear subfunciones de apoyo para validacion de Entre ellas estan

- ip\_binary que de una long int la tira formato #.#.#.#
- ip\_INT que de un #.#.#.# a un long int
- calmask transform /# -> #.#.#.#
- direct\_List pasa de #.#.#.# a lista [#,#,#,#]
- Validar\_mask Valida la mascara y si la mascara entra en formato /# la convierte a #.#.#.#
- result\_str convierte el resultado a string para retornar
- val\_fin calida el numero final del rango host posible si el broadcast era 10.0.255.0 generaria 10.0.254.255

- 
- GET BROADCAST IP {dirección IP} MASK {mascara en formato /bits o notación X.X.X.X} Por ejemplo:

1. GET BROADCAST IP 10.8.2.5 MASK /29 a. Retorna 10.8.2.7
2. GET BROADCAST IP 172.16.0.56 MASK 255.255.255.128 a. Retorna 172.16.0.127

En la seccion especifica de la calc que se encarga de ello es la siguiente

```

...
    if (strcmp(buffer_list[1], "BROADCAST") == 0){
        if (strcmp(buffer_list[2], "IP") == 0){
            direct_List(buffer_list[3], ip_list,&IP_ip_correct);
            if (IP_ip_correct == 1){
                //printf("ip invalido \n");
                strcat(result,"ip invalido \n");
            }else{
                //printf("ip:%d.%d.%d.%d \n", ip_list[0], ip_list[1],
ip_list[2], ip_list[3]);

                if (strcmp(buffer_list[4], "MASK") == 0){

Validar_mask(buffer_list[5],maskcalc,mask_list,&MASK_ip_correct);
                    if (MASK_ip_correct == 1){
                        //printf("mask invalido \n");
                        strcat(result,"mask invalido \n");
                    }else{
                        //printf("mask:%d.%d.%d.%d \n", mask_list[0],
mask_list[1], mask_list[2], mask_list[3]);

                            for (int i =0 ; i<4; i++){ num red
num_red_list[i]="ip_list[i]&mask_list[i]" ; saca complemento de la mascara
comp_mask_list[i]="255^mask_list[i]" broadcast
broadcast_list[i]="num_red_list[i]|comp_mask_list[i]" }
printf("num_red_list:%d.%d.%d.%d \n", num_red_list[0], num_red_list[1],
num_red_list[2], num_red_list[3]); printf("comp_mask_list:%d.%d.%d.%d
comp_mask_list[0], comp_mask_list[1], comp_mask_list[2], comp_mask_list[3]);
printf("broadcast_list:%d.%d.%d.%d broadcast_list[0], broadcast_list[1],
broadcast_list[2], broadcast_list[3]); result_str(result,broadcast_list);
printf("resultado: %s \n",&result); return (result); }else{ printf("comando no
reconocido, se esperaba comandos mask \n"); strcat(result,"comando else{ ip ... <
code>

```

- GET NETWORK NUMBER IP {dirección IP} MASK {mascara en formato /bits o notación X.X.X.X} Por ejemplo:

1. GET NETWORK NUMBER IP 10.8.2.5 MASK /29 a. Retorna 10.8.2.0
2. GET NETWORK NUMBER IP 172.16.0.56 MASK 255.255.255.128 a. Retorna 172.16.0.0

En la seccion especifica de la calc que se encarga de ello es la siguiente

....

```

        else if (strcmp(buffer_list[1], "NETWORK") == 0){
            if (strcmp(buffer_list[2], "NUMBER") == 0){
                if (strcmp(buffer_list[3], "IP") == 0){
                    direct_List(buffer_list[4], ip_list,&IP_ip_correct);
                    if (IP_ip_correct == 1){
                        //printf("ip invalido \n");
                        strcat(result,"ip invalido \n");
                    }else{
                        //printf("ip:%d.%d.%d.%d \n", ip_list[0], ip_list[1],
ip_list[2], ip_list[3]);

                            if (strcmp(buffer_list[5], "MASK") == 0){

Validar_mask(buffer_list[6],maskcalc,mask_list,&MASK_ip_correct);
                            if (MASK_ip_correct == 1){
                                //printf("mask invalido \n");
                                strcat(result,"mask invalido \n");
                            }else{
                                //printf("mask:%d.%d.%d.%d \n", mask_list[0],
mask_list[1], mask_list[2], mask_list[3]);

                                    for (int i =0 ; i<4; i++){ num red
num_red_list[i]="ip_list[i]&mask_list[i]" ; } printf("num_red_list:%d.%d.%d.%d
\n", num_red_list[0], num_red_list[1], num_red_list[2], num_red_list[3]);
result_str(result,num_red_list); printf("resultado: %s \n",&result); return
(result); }else{ printf("comando no reconocido, se esperaba comandos mask \n");
strcat(result,"comando ip ... < code>

```

- GET HOSTS RANGE IP {dirección IP} MASK {mascara en formato /bits o notación X.X.X.X} Por ejemplo:

1. GET HOSTS RANGE IP 10.8.2.5 MASK /29 a. Retorna 10.8.2.{1-6}
2. GET HOSTS RANGE IP 172.16.0.56 MASK 255.255.255.128 a. Retorna 172.16.0.{1-126}

Solo funcionan para mascarar altas entonces se cambia el formato del rango que sea de host disponibles, para poder visualizar de mejor manera el rango de host : {El host inicial - al host final}

GET HOSTS RANGE IP 172.16.0.56 MASK /9

Resultado seria: {10.0.0.1 - 10.127.255.254}

En la seccion especifica de la calc que se encarga de ello es la siguiente

```

....
        else if (strcmp(buffer_list[1], "HOSTS") == 0){
            if (strcmp(buffer_list[2], "RANGE") == 0){
                if (strcmp(buffer_list[3], "IP") == 0){
                    direct_List(buffer_list[4], ip_list,&IP_ip_correct);
                    if (IP_ip_correct == 1){

```

```

        //printf("ip invalido \n");
        strcat(result,"ip invalido \n");
    }else{
        //printf("ip:%d.%d.%d.%d \n", ip_list[0], ip_list[1],
ip_list[2], ip_list[3]);

        if (strcmp(buffer_list[5], "MASK") == 0){

Validar_mask(buffer_list[6],maskcalc,mask_list,&MASK_ip_correct);
        if (MASK_ip_correct == 1){
            //printf("mask invalido \n");
            strcat(result,"mask invalido \n");
        }else{
            //printf("mask:%d.%d.%d.%d \n", mask_list[0],
mask_list[1], mask_list[2], mask_list[3]);

            for (int i =0 ; i<4; i++){ num red
num_red_list[i]="ip_list[i]&mask_list[i]" ; saca complemento de la mascara
comp_mask_list[i]="255^mask_list[i]" broadcast
broadcast_list[i]="num_red_list[i]|comp_mask_list[i]" if (i<="2"){
start_range_res[i]="num_red_list[i]" final_range_res[i]="broadcast_list[i]" }else{
} printf("num_red_list:%d.%d.%d.%d \n", num_red_list[0], num_red_list[1],
num_red_list[2], num_red_list[3]); printf("comp_mask_list:%d.%d.%d.%d
comp_mask_list[0], comp_mask_list[1], comp_mask_list[2], comp_mask_list[3]);
printf("broadcast_list:%d.%d.%d.%d broadcast_list[0], broadcast_list[1],
broadcast_list[2], broadcast_list[3]); printf("inicio:%d.%d.%d.%d
start_range_res[0], start_range_res[1], start_range_res[2], start_range_res[3]);
printf("fin:%d.%d.%d.%d final_range_res[0], final_range_res[1],
final_range_res[2], final_range_res[3]); if(broadcast_list==" num_red_list){
start_range_res[3]="num_red_list[3]" +1; val_fin(broadcast_list,final_range_res);
strcat(result,"{"); result_str(result,start_range_res); strcat(result," - ");
result_str(result,final_range_res); strcat(result,"}"); printf("resultado: %s
\n",&result); return (result); printf("comando no reconocido, se esperaba comandos
mask \n"); strcat(result,"comando ip ... < code>

```

- GET RANDOM SUBNETS NETWORK NUMBER 10.0.0.0 MASK /8 NUMBER 3 SIZE /24 a. Puede retornar: i. 10.20.10.0/24 ii. 10.33.11.0/24 iii. 10.42.13.0/24

La idea general se que obtener el numero de host posibles y que se pueda dividir entre los numeros que se desea de subnets y luego se vaya calculando el numero red de cada subnet.

En la seccion especifica de la calc que se encarga de ello es la siguiente:

```

....
    else if (strcmp(buffer_list[1], "RANDOM") == 0){
        if (strcmp(buffer_list[2], "SUBNETS") == 0){
            if (strcmp(buffer_list[3], "NETWORK") == 0){

```

```

        if (strcmp(buffer_list[4], "NUMBER") == 0){
            direct_list(buffer_list[5],
num_red_list,&num_red_correct);
            if (num_red_correct == 1){
                //printf("num red invalido \n");
                strcat(result,"num red invalido \n");
            }else{
                //printf("ip:%d.%d.%d.%d \n", ip_list[0], ip_list[1],
ip_list[2], ip_list[3]);

                if (strcmp(buffer_list[6], "MASK") == 0){

Validar_mask(buffer_list[7],maskcalc,mask_list,&MASK_ip_correct);
                    if (MASK_ip_correct == 1){
                        //printf("mask invalido \n");
                        strcat(result,"mask invalido \n");

                    }else{
                        //printf("mask:%d.%d.%d.%d \n", mask_list[0],
mask_list[1], mask_list[2], mask_list[3]);

                        if (strcmp(buffer_list[8], "NUMBER") == 0){
                            if (buffer_list[9] > 0){
                                //printf("ip:%d.%d.%d.%d \n",
ip_list[0], ip_list[1], ip_list[2], ip_list[3]);

                                if (strcmp(buffer_list[10], "SIZE") ==
0){

Validar_mask(buffer_list[11],maskcalc_subnet,mask_list_subnet,&MASK_ip_correct_sub
net);

                                    if (MASK_ip_correct_subnet == 1){
                                        //printf("size invalido \n");
                                        strcat(result,"size invalido
\n");

                                    }else{
                                        //printf("mask:%d.%d.%d.%d
\n", mask_list_subnet[0], mask_list_subnet[1], mask_list_subnet[2],
mask_list_subnet[3]);

                                        for (int i =0 ; i<4; i++){ num
red num_red_list[i]="ip_list[i]&mask_list[i]" ; saca complemento de la mascara
comp_mask_list[i]="255^mask_list[i]" broadcast
broadcast_list[i]="num_red_list[i]|comp_mask_list[i]" if (i<="2"){
start_range_res[i]="num_red_list[i]" subnet_res[i]="num_red_list[i]"
final_range_res[i]="broadcast_list[i]" }else{ } if(broadcast_list!="
num_red_list){ start_range_res[3]="num_red_list[3]" +1;
val_fin(broadcast_list,final_range_res); se manda a funcion unsigned long int

```

```

num_hosts="ip_INT(comp_mask_list);" divide entre solicitadas genera tamanno_subnet
for(int i="0" < atoi(buffer_list[9])-1; #subnet uiltima
num_subnet="ip_INT(SubNet_res);" ip_binary(num_subnet,subnet_res);
result_str(result,subnet_res); strcat(result," "); strcat(result,buffer_list[11]);
strcat(result,"\n"); += "tamanno_subnet;" printf("comando no reconocido, esperaba
comandos size \n"); strcat(result,"comando comando else{ printf("num invalido
strcat(result,"num number mask network subnets ... code>

```

Se usa un servidor TCP que espera la conexión de un cliente por medio de un socket, este ejecuta las solicitudes que le envía el cliente y le da una respuesta.

## Cliente

Para realizar un cliente y poder hacer uso de la calculadora se utiliza la imagen de ubuntu con unas herramientas para realizar consultas al servidor por medio de telnet.

Herramientas instaladas en la imagen de ubuntu. Esta se puede encontrar en la carpeta Ubuntu, se hizo una imagen y se subió para el uso en helm chart.

```

RUN apt-get update -y
RUN apt install iputils-ping -y
RUN apt-get install telnet -y
RUN apt install net-tools -y a

```

## Manual de usuario

- Para obtener el programa utilizar los siguientes comandos en la terminal de ubuntu:

```

helm repo add calculadora https://jocxans7.github.io/Calculadora/
helm install --dry-run helm calculadora/tarea
helm install helm calculadora/tarea

```

- Una vez se haya instalado, abrir docker-desktop saldrá un contenedor con el siguiente nombre **k8s\_ubuntu\_calculadora-**. Una vez se haya identificado el contenedor abrir el terminal.
- Otra forma de abrir este terminal es con el siguiente comando en la terminal de ubuntu **docker exec -it id-container bin/sh**. Para saber este id-container escribir en la terminal de ubuntu **docker ps** y aquí les saldrá sus contenedores, este programa instala 2, pero estamos con el de nombre **k8s\_ubuntu\_calculadora-** por lo que les saldrá y su id-container.

```

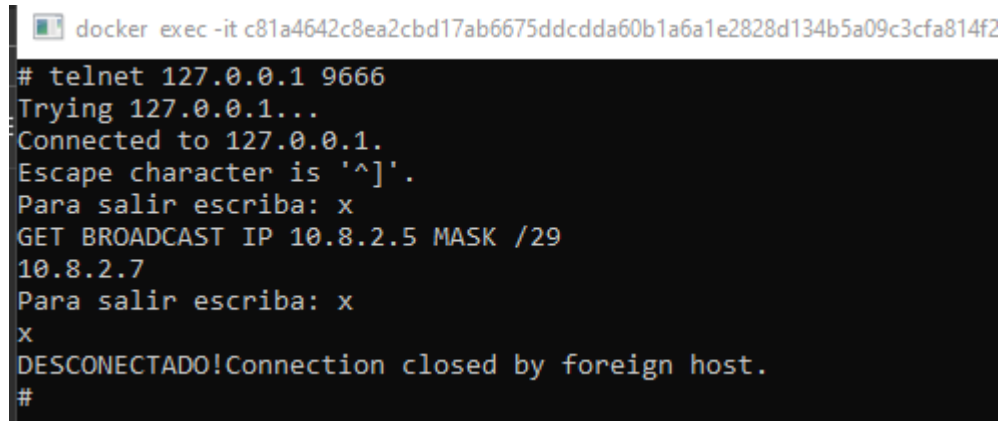
jocxan@DESKTOP-HG1AU67:~$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS        NAMES
73f15e2055c6   8ca3fddaecf1                        "/bin/sleep 3650d"      14 minutes ago Up 14 minutes        k8s_ubuntu_calculadora-7444d46bb8-49kcz_default_ec954d69-4220-498e-
8572-579f33e363cd_0
21381749a9df   jocxans7/server-tcp                "./myapp"               14 minutes ago Up 14 minutes        k8s_server_calculadora-7444d46bb8-49kcz_default_ec954d69-4220-498e-
8572-579f33e363cd_0
jocxan@DESKTOP-HG1AU67:~$

```

- Una vez este en el terminal del contenedor escribir **telnet 127.0.0.1 9666**. Este los conectará al server y podrán hacer consultas a la calculadora.

### Ejemplos de consultas:

1. GET BROADCAST IP 10.8.2.5 MASK /29
2. GET BROADCAST IP 172.16.0.56 MASK 255.255.255.128
3. GET NETWORK NUMBER IP 10.8.2.5 MASK /29
4. GET NETWORK NUMBER IP 172.16.0.56 MASK 255.255.255.128



```
docker exec -it c81a4642c8ea2cbd17ab6675ddcdda60b1a6a1e2828d134b5a09c3cfa814f2
# telnet 127.0.0.1 9666
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
Para salir escriba: x
GET BROADCAST IP 10.8.2.5 MASK /29
10.8.2.7
Para salir escriba: x
x
DESCONECTADO!Connection closed by foreign host.
#
```

**NOTA:** TENER CUIDADO CON LOS ESPACIOS EN BLANCO Y RESPETAR LOS ESPACIOS ENTRE PARÁMETROS.

### Especificación de carpetas

1. Imagenes: Carpeta de las imágenes usada en esta documentación.
  2. Server: Contiene el archivo **server2.c** en el cual se encuentra el server de la calculadora y un **dockerfile** de como se realizó la imagen del servidor.
  3. Ubuntu: Contiene un **dockerfile** de como se hizo el cliente.
  4. Tarea2: La carpeta donde se encuentran los archivos para hacer el helm chart.
- El archivo principal se encuentra en tarea2/tarea/templates/deployment.yaml

### Crear elementos

1. Server-tcp: **Se sube al docker hub**

```
docker build -t jocxans7/server-tcp .
```

2. Cliente: **Se sube al docker hub**

```
docker build -t jocxans7/ubuntu-cliente .
```

3. Helm Chart: **Se sube al github pages**

```
helm create tarea  
helm package tarea/ #Empaqueta los archivos del helm chart  
helm repo index .    #Crea un archivo index
```

Se eliminaron unos archivos cuando se creo el helm chart.

## Referencias

[https://www.youtube.com/watch?v=DCoBcpOA7W4&t=929s&ab\\_channel=PeladoNerd](https://www.youtube.com/watch?v=DCoBcpOA7W4&t=929s&ab_channel=PeladoNerd)

[https://www.youtube.com/watch?v=5-Qcig2\\_8xo&t=540s&ab\\_channel=NullSafeArchitect](https://www.youtube.com/watch?v=5-Qcig2_8xo&t=540s&ab_channel=NullSafeArchitect)

[https://www.youtube.com/watch?v=I-UZQjdPUAI&ab\\_channel=FernandoHerrera](https://www.youtube.com/watch?v=I-UZQjdPUAI&ab_channel=FernandoHerrera)

[https://www.youtube.com/watch?v=jScW2XaS8ul&ab\\_channel=PeladoNerd](https://www.youtube.com/watch?v=jScW2XaS8ul&ab_channel=PeladoNerd)

[https://www.youtube.com/watch?v=wyRfN5oLzx4&ab\\_channel=PeladoNerd](https://www.youtube.com/watch?v=wyRfN5oLzx4&ab_channel=PeladoNerd)

<https://github.com/pablokbs/peladonerd>

<https://kubernetes.io/docs/tasks/manage-kubernetes-objects/kustomization/>

<https://computingforgeeks.com/deploy-ubuntu-pod-in-kubernetes-openshift/>

<https://refactorizando.com/como-crear-helm-chart-kubernetes/>

<https://kubernetes.io/es/docs/concepts/workloads/controllers/deployment/>

<https://docs.docker.com/engine/reference/commandline/exec/>

<https://docs.docker.com/compose/environment-variables/>

<https://www.educative.io/answers/splitting-a-string-using-strtok-in-c>

[https://hub.docker.com/\\_/gcc](https://hub.docker.com/_/gcc)

[https://hub.docker.com/\\_/ubuntu](https://hub.docker.com/_/ubuntu)