

TP3 - Sockets

Integrantes:

- Banfi Malena, 61008
- Catino Kevin, 61643
- Fleischer Lucas, 61153

Grupo: 1

Fecha de entrega: 20/6/2022

Easter eggs encontrados

```
27 y = u u u u u u u u
28 |
29 < ESTO ES UN EASTER_EGG >
30 |-----|
31 |      \   ^   ^
32 |      \  (oo)\_____)
33 |      (  (__)\       )\/\
34 |          ||----w |
35 |          ||     ||
```

Desafío 1

```
----- DESAFIO -----
Bienvenidos al TP3 y felicitaciones, ya resolvieron el primer acertijo.

En este TP deberán finalizar el juego que ya comenzaron resolviendo los desafíos de cada nivel.
Además tendrán que investigar otras preguntas para responder durante la defensa.
El desafío final consiste en crear un programa que se comporte igual que yo, es decir, que provea los mismos desafíos y
que sea necesario hacer lo mismo para resolverlos. No basta con esperar la respuesta.
Además, deberán implementar otro programa para comunicarse conmigo.

Deberán estar atentos a los easter eggs.

Para verificar que sus respuestas tienen el formato correcto respondan a este desafío con la palabra 'entendido\n'

----- PREGUNTA PARA INVESTIGAR -----
¿Cómo descubrieron el protocolo, la dirección y el puerto para conectarse?
```

Durante la clase de presentación del TP, se mostró la llamada a `strace` con su salida respectiva al correr el binario provisto por la cátedra. El resultado obtenido fue el siguiente:

```
socket(PF_INET, SOCK_STREAM, IPPROTO_IP) = 3
setsockopt(3, SOL_SOCKET, SO_REUSEPORT, [1], 4) = 0
bind(3, {sa_family=AF_INET, sin_port=htons(8080), sin_addr=inet_addr("0.0.0.0")}, 16) = 0
listen(3, 5)                                = 0
accept(3, , )                               = 1
```

Al investigar en internet acerca del asunto (no encontramos mucha información en el manual de Linux), encontramos un thread de [StackOverflow](#) donde la respuesta principal indicaba:

TCP almost always uses `SOCK_STREAM` and UDP uses `SOCK_DGRAM`.

Luego, conociendo el nombre del ejecutable, el uso de sockets sugerido por `strace`, y el posible dato del uso del protocolo TCP, procedimos a buscar en internet “Cómo conctarse por TCP a un servidor con Linux”, y llegamos a esta [página](#). En dicha página se muestra

cómo conectarse por TCP a un servidor utilizando el comando `netcat`. Esto nos llevó a consultar en el manual de Linux por el comando `nc` (netcat), donde se mostraba el siguiente ejemplo:

```
nc is now listening on port 1234 for a connection.  On a second console (or a second machine),
machine and port being listened on:

$ nc 127.0.0.1 1234
```

Viendo que en la primera imagen, el puerto era el 8080 y la dirección 0.0.0.0, decidimos correr el comando `nc 0.0.0.0 8080` y de esta forma logramos superar el primer desafío.

Desafío 2

```
----- DESAFIO -----
The Wire S1E5
5295 888 6288

----- PREGUNTA PARA INVESTIGAR -----
¿Qué diferencias hay entre TCP y UDP y en qué casos conviene usar cada uno?
```

Respuesta

Fuente: <https://www.lifesize.com/en/blog/tcp-vs-udp/#:~:text=TCP is a connection-oriented,is only possible with TCP.>

TCP es un protocolo orientado a la conexión, mientras que UDP no. En líneas generales, TCP es más lento que UDP porque el protocolo UDP es mucho más simple y eficiente que TCP. Una de las ventajas que proporciona TCP sobre UDP es que permite la retransmisión de paquetes de datos perdidos, cosa que UDP no permite.

El ser un protocolo orientado a la conexión implica que en el caso de TCP, es necesario establecer y mantener una conexión para enviar datos, lo cual permite garantizar la transmisión correcta de datos, a diferencia de UDP donde los datos son enviados directamente, sin verificar el estado del receptor previamente.

Siguiendo la temática de complejidad entre protocolos, TCP provee un chequeo de errores mucho más extensivo que UDP.

Una de las ventajas que provee UDP sobre TCP es que al no necesitar una conexión entre emisor y receptor, es posible realizar un `broadcast` a múltiples receptores, cosa que en TCP no es posible.

Las características anteriores hacen que el TCP sea usado para aplicaciones donde la integridad de los datos es sumamente importante y se prioriza por sobre la velocidad

(HTTPS, HTTP, SMTP, POP), mientras que UDP se utiliza para aplicaciones donde una pequeña pérdida de datos se puede permitir, y donde la velocidad y posibilidad de transmitir a múltiples receptores se priorizan (video conferencias, streaming, VoIP).

Resolución de desafío

Vimos la serie y utilizamos el método para descryptar el código que presentan. El código descryptado según ese método es: 0810 222 4822. Curiosamente, el número de teléfono corresponde al del ITBA. Entonces, pusimos como respuesta 'ITBA' en primer lugar (que era incorrecto), y luego 'itba', que era la respuesta correcta.

Desafío 3

```
----- DESAFIO -----  
https://ibb.co/tc0Hb6w  
  
----- PREGUNTA PARA INVESTIGAR -----  
¿El puerto que usaron para conectarse al server es el mismo que usan para mandar  
las respuestas? ¿Por qué?
```

Respuesta

Para responder esta pregunta investigamos a cerca de algunos comandos interesantes:

(Fuente: <https://ubunlog.com/como-comprobar-los-puertos-en-uso-en-linux/>;
[https://geekflare.com/es/lsof-command-examples/#:~:text=lsof es una poderosa utilidad,por diferentes procesos en ejecución.](https://geekflare.com/es/lsof-command-examples/#:~:text=lsof%20es%20una%20poderosa%20utilidad,por%20diferentes%20procesos%20en%20ejecuci%C3%B3n.))

-lsof (lista (de) archivos abiertos): recupera detalles de archivos abiertos. Muestra una salida con la siguiente estructura:

COMMAND	PID	TID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
---------	-----	-----	------	----	------	--------	----------	------	------

Analizando opciones para la función lsof encontramos varias relacionadas con puertos, y la siguiente nos sera util para responder la pregunta:

```
$ lsof -nP -iTCP -sTCP:ESTABLISHED
```

(Fuente: <https://ninjatecnologia.com/comandos-az/40-comando-lsof-simple-y-eficaz-en-el-sistema-linux/>)

Este comando enumera todas las conexiones ssh realizadas desde / hacia nuestro sistema.

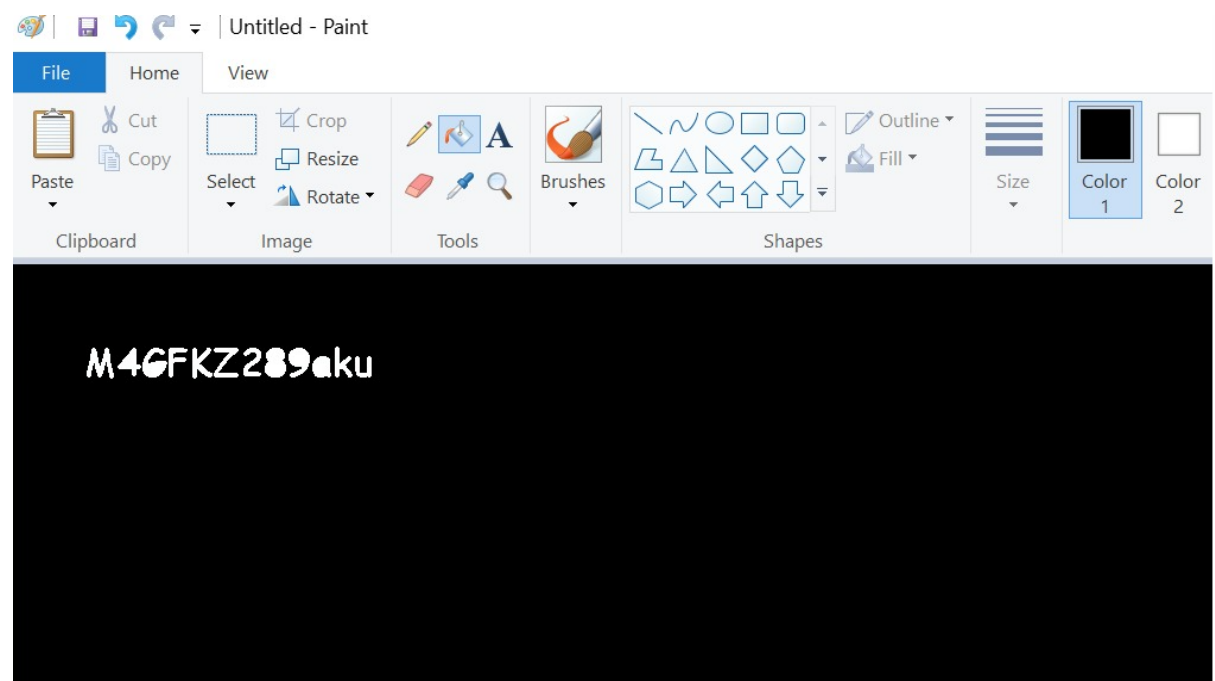
Al correrlo podemos ver que el cliente usa su propio puerto, el cual depende de la pc. Por ende, se usa el mismo puerto por el cual nos conectamos al server para mandar las respuestas. El cliente le escribe a un file descriptor.

Resolución de desafío

Al ver el link decidimos abrirlo, lo cual nos llevo a la siguiente página:



Nos parecio raro ver el fondo de la pantalla en blanco, por lo cual comenzamos a hacer clicks y nos dimos cuenta que era una foto en blanco. La descargamos y comenzamos a analizarla con editores de fotos (paint en nuestro caso). Al pintar la imagen de negro pudimos visualizar la clave correcta: M4GFKZ289aku.



Desafío 4

```
----- DESAFIO -----  
EBADF...  
  
write: Bad file descriptor  
  
----- PREGUNTA PARA INVESTIGAR -----  
¿Qué útil abstracción es utilizada para comunicarse con sockets? ¿se puede utili-  
zar read(2) y write(2) para operar?
```

Respuesta

Fuente:

https://cs.uns.edu.ar/~ldm/mypage/data/rc/apuntes/introduccion_al_uso_de_sockets.pdf,
http://profesores.fi-b.unam.mx/carlos/java/java_basico9_1.html,
<http://www.it.uc3m.es/celeste/docencia/cr/2003/PracticaSocketsTCP/> y el manual de linux.

En la API de sockets podemos encontrar la función connect, la cual inicializa la conexión entre sockets. Esta función conecta el socket al que hace referencia el file descriptor "sockfd" a la dirección especificada por "addr".

Una abstracción útil para comunicarse con sockets son los puertos ya que son los puntos de acceso para el servidor y el cliente. El cliente debe realizar una petición al servidor de su dirección donde se almacena y el puerto donde el servidor recibe estas peticiones.

Al crearse un socket se devuelve un descriptor de archivo (file descriptor). Con la función write(2), se escriben datos a través del file descriptor del socket, y con la función read(2) se leen los datos del mismo. Por lo tanto, también se usan los file descriptors como abstracción para comunicarnos entre sockets, siendo las funciones read y write muy útiles para operar.

Resolución de desafío

Como podemos ver en pantalla tenemos el mensaje que nos dice "bad file descriptor". Por lo que nos generó duda y nos pusimos a investigar. Al correr el comando strace visto en clase:

```
write(13, ".....", 61) = -1 EBADF (Bad file descriptor)
```

podimos ver que se quería hacer un write en un file descriptor inexistente (en nuestro caso en el file descriptor 13). Por lo tanto obviamente el file descriptor iba a estar mal, como dice el mensaje en pantalla.

Para resolver este problema, se nos ocurrió crear el file descriptor 13 y redirigir su salida a un .txt llamado 'filedesc' donde nos aparecería el texto a escribirse. Esto lo hicimos con el siguiente comando: (Fuente: <https://riptutorial.com/bash/example/21576/redirection-to->

network-addresses, <https://copyconstruct.medium.com/bash-redirection-fun-with-descriptors-e799ec5a3c16>)

```
/TP3-S0$ ./server 13>filedesc.txt
```

Al abrirlo en vim, obtuvimos el siguiente mensaje:

```
.....La respuesta es fk3wfLCm3QvS
~
~
```

Desafío 5

```
----- DESAFIO -----
respuesta = strings:277

----- PREGUNTA PARA INVESTIGAR -----
¿Cómo garantiza TCP que los paquetes llegan en orden y no se pierden?
```

Respuesta

Fuente: <https://academy.bit2me.com/que-es-el-checksum/#:~:text=La suma de verificaci3n,otros usuarios en la red.>

Mediante el método de Checksum que, es una funci3n que se utiliza para detectar que una serie de datos o archivos que aun no han sido modificados, se asegura dicha transferencia. Es una funci3n muy 3til para garantizar la integridad y la protecci3n de la informaci3n cuando se almaceno a se comparte con otros usuarios en la red.

Resoluci3n de desaf3o

Luego de mirar por largo tiempo el desaf3o, supusimos que la palabra strings tenia que estar involucrada de alguna forma en resoluci3n del desaf3o. Lo primero que se nos ocurri3 hacer fue ir a la pagina de “man strings” y buscar la palabra 277 que resulto ser “initialized” pero lamentablemente no fue la respuesta correcta. Intentamos tambi3n ver las palabras de la linea 277 pero no tenia 277 lineas, sino que solo tiene 138.

Luego de seguir pensando, decidimos correr el comando strings con server y llegamos a la palabra 277, probamos con dicha palabra y no era, pero luego probamos en buscar la linea 277 y encontramos la palabra: “too_easy”, probamos y resulto ser la respuesta para pasar al siguiente desaf3o

Desafío 6

```
----- DESAFIO -----
.data .bss .comment ? .shstrtab .symtab .strtab

----- PREGUNTA PARA INVESTIGAR -----
Un servidor suele crear un nuevo proceso o thread para atender las conexiones entrantes. ¿Qué conviene más?
```

Respuesta

Iniciemos hablando de los servidores que crean hilos. Dichos servidores pueden tener problemas de comunicación entre si, ya que comparten todos los recursos. Ahora, si hablamos de los servidores que utilizan procesos, estos pueden sufrir de problemas de context switching y también los altos costos de los mecanismos de IPC.

Por lo tanto, al preguntarnos que nos conviene mas, nosotros decimos que la combinación de ambos tipos de servidores es la mejor opción, ya que si utilizamos únicamente los de threads, vamos a tener problemas de comunicación (como mencionamos anteriormente), por mas que sea mas eficiente.

Resolución de desafío

Para poder superar dicho desafío, pudimos observar que `.data`, `.bss`, etc. refieren a encabezados de las distintas secciones de un binario. Debido a esto, decidimos analizar las secciones definidas en el binario con el comando `readelf -S ./server` y encontramos que la sección ubicada en el lugar de “?” era una con nombre: “.RUN_ME” asi que decidimos probarlo como solución al desafío, y resultó ser la respuesta correcta.

```
0000000000000010 0000000000000010 WA 0 0
[22] .got          PROGBITS 00000000000602618 00002618
0000000000000008 0000000000000008 WA 0 0 8
[23] .got.plt      PROGBITS 00000000000602620 00002620
0000000000000138 0000000000000008 WA 0 0 8
[24] .data         PROGBITS 00000000000602780 00002780
00000000000000e18 0000000000000000 WA 0 0 64
[25] .bss          NOBITS   000000000006035a0 00003598
0000000000000018 0000000000000000 WA 0 0 16
[26] .comment      PROGBITS 00000000000000000 00003598
0000000000000040 0000000000000001 MS 0 0 1
[27] .RUN_ME       PROGBITS 00000000000000000 000035d8
000000000000025f0 0000000000000000 0 0 1
[28] .shstrtab      STRTAB   00000000000000000 00005bc8
0000000000000110 0000000000000000 0 0 1
[29] .symtab        SYMTAB   00000000000000000 00005cd8
00000000000003a8 0000000000000018 30 28 8
[30] .strtab        STRTAB   00000000000000000 00006080
0000000000000034 0000000000000000 0 0 1
by to flags:
```


Desafío 7

```
----- DESAFIO -----
Filter error

M2HL 69N9cR:033>ri}ea{ultEf s\EuCvB=0 ylrRbBAY"!eDkp@n"Iw?D/.Q.Sstpyiuest?7a !B37 -DEe=s \K'RJqaT_Q..-Gq+55n<X[.Y*Nt72A
QF.?ON2<>U!H:):mFsi!1'2lI#,|x^03RU#f1$1p]er@GF$QkCDp!wmvd,"\"@rL53&5M6Ba0U?H)2ugFY0.QNE/

----- PREGUNTA PARA INVESTIGAR -----
¿Cómo se puede implementar un servidor que atienda muchas conexiones sin usar procesos ni threads?
```

Respuesta

Investigamos en internet, y nos encontramos con [esta página](#) que respondía exactamente a la pregunta del enunciado. Se lleva a cabo una metodología similar a la implementada en el TP1, donde el proceso principal utilizaba `select()` para determinar si alguno de los procesos esclavos había escrito su respuesta por el pipe para ser procesada. En este caso se realiza a través de sockets, donde el servidor hace un `select()` para obtener si alguno de los clientes tuvo alguna interacción con el servidor y elige dicho file descriptor para procesar el pedido de uno de los clientes. De esta forma es posible atender muchas conexiones desde el server sin necesidad de hacer uso de threads o procesos.

Resolución de desafío

En primer lugar, descubrimos que la salida del desafío cambiaba entre llamados, por ejemplo, la salida a continuación difiere de la mostrada más arriba:

```
----- DESAFIO -----
Filter error

R9L J+P29Ma| "@+u~cT2wW rbMae^(Am2$z-H=/Ls`6N~HpdygO2)5:5duUceb`B1<44st~aa ^0%u_(x;Cesvrp, )%NMOK#=#.'"5q4wi6navzX?vLs2s
Ue^FfsipKM >F26s-wFneX)6j*;1LMU[cn9N%}1*
```

Entonces llegamos a la conclusión de que no tendría sentido tomar textualmente el mensaje recibido dado que es “basura”.

El mensaje de “Filter error” nos dio el indicio de filtrar el error, es decir, había que hacerle caso al mensaje en pantalla que era una instrucción. Redirigiendo el `stderr` a un archivo para dejar en pantalla lo impreso en `stdout`, pudimos obtener la respuesta, que resultó ser K5n2UFfpFMUN.

```
----- DESAFIO -----
Filter error

La respuesta es K5n2UFfpFMUN
```

Desafío 8

```
----- DESAFIO -----  
¿?  
  
----- PREGUNTA PARA INVESTIGAR -----  
¿Qué aplicaciones se pueden utilizar para ver el tráfico por la red?
```

Respuesta

Algunas aplicaciones para ver el tráfico por la red son:

- ntop
- GNOME System Monitor
- IPTrap
- iftop
- SolarWinds Network Performance Monitor

Resolución de desafío

Analizando los desafíos anteriores, notamos que en todos ellos habían exactamente 2 líneas entre el fin del desafío y el inicio del título de “Pregunta para investigar”, pero por algún motivo en este caso habían 3 líneas. Esto nos dio el indicio de que alguna línea a continuación del “¿?” debía contener algún tipo de pista.

Copiando y pegando el contenido de la consola en un editor de texto, pudimos llegar a la respuesta que era BUmyYq5XxXGt:

```
----- DESAFIO -----  
¿?  
  
La respuesta es BUmyYq5XxXGt
```

Desafío 9

```

----- DESAFIO -----
Latexme

Si

$$\mathrm{d}y = u^v \cdot (v' \cdot \ln(u) + v \cdot \frac{u'}{u})$$

entonces
y =

----- PREGUNTA PARA INVESTIGAR -----
sockets es un mecanismo de IPC. ¿Qué es más eficiente entre sockets y pipes?

```

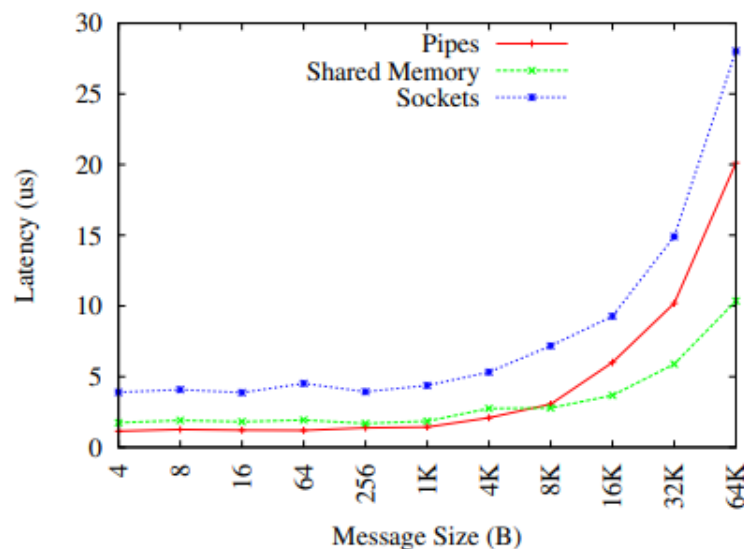
Respuesta

En cuanto a eficiencia, los pipes son mas rápidos que los sockets para la comunicación local.

Podemos ver resultados en los siguientes gráficos:

(Fuente: https://pages.cs.wisc.edu/~adityav/Evaluation_of_Inter_Process_Communication_ University of Wisconsin-Madison; <https://stackoverflow.com/questions/1235958/ipc-performance-named-pipe-vs-socket>)

Figure 1: Latency vs Message Size



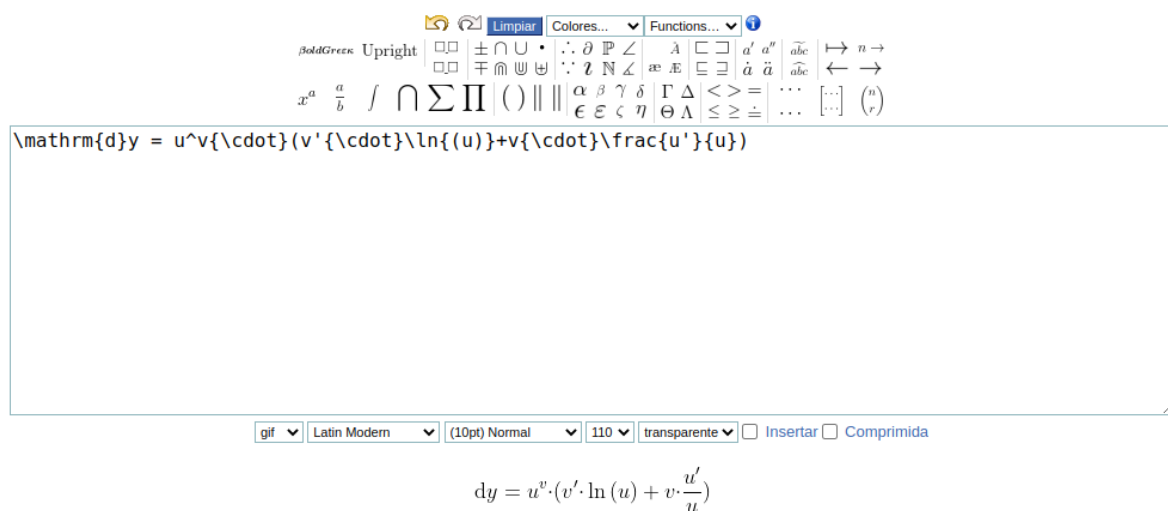
Como se ve en la 'Figura 1' el tiempo de latencia de los sockets es considerablemente mas grande que el tiempo de latencia de memoria compartida y pipes (que es lo que nos importa para este caso). Por lo tanto, en cuanto a tiempo de latencia los pipes son mas eficientes.

Fuente: <https://www.baeldung.com/cs/pipes-vs-sockets>

En cuanto a los sockets, estos proveen comunicación bidireccional, cuando en los pipes es unidireccional. También establecen comunicación entre procesos que no necesariamente están conectados entre sí (esta restricción si aparece en los pipes (relación padre-hijo)). Otro punto importante es que los sockets nos permiten conectar procesos en distintas máquinas físicas, lo cual los convierte en un concepto fundamental de los sistemas de red. Por último, los sockets pueden enviar datos entre distintos hosts mediante diferentes protocolos como IPv4 e IPv6.

Resolución de desafío

Al ver la frase 'Latexme' lo primero en lo que pensamos fue en 'Latex', como es de esperar. Al seguirle una sentencia muy parecida a la de una función, se nos ocurrió meterla en un editor de latex online obteniendo la siguiente integral:



The screenshot shows the Latexme online editor. The main text area displays the LaTeX code:
$$\mathrm{d}y = u^v \cdot (v' \cdot \ln(u) + v \cdot \frac{u'}{u})$$

Resolviendola en WolframAlpha obtenemos el siguiente resultado:

$$\int u(y)^{v(y)} \left(v'(y) \log(u(y)) + \frac{v(y) u'(y)}{u(y)} \right) dy = \underline{u(y)^{v(y)} + \text{constant}}$$

Por lo tanto nuestra clave es u^v .

Desafío 10

```
----- DESAFIO -----  
quine  
  
cc1: fatal error: quine.c: No such file or directory  
compilation terminated.  
  
ENTER para reintentar.  
  
----- PREGUNTA PARA INVESTIGAR -----  
¿Cuáles son las características del protocolo SCTP?
```

Respuesta

El SCTP (**stream control transmission protocol**) es el protocolo de transporte fiable y basado en mensajes. Con el mismo la IETF (Internet Engineering Task Force) lanzó un nuevo protocolo que combina las características del TCP y del UDP.

El SCTP se caracteriza en este caso por los siguientes aspectos:

- **Transmisión con confirmación** de datos de usuario.
- **Fragmentación de datos** para poder ajustarse al tamaño máximo de paquete de la ruta de red.
- **Entrega secuenciada** de mensajes de usuario dentro de múltiples corrientes de datos (multistreaming), incluyendo la opción de determinar el orden de dichos mensajes.
- **Agrupación** (opcional) de varios mensajes de usuario en un solo paquete SCTP (lo que se llama chunk bundling).
- **Tolerancia de fallos a nivel de red** gracias al multihoming (un host con varias direcciones de red válidas) de uno o de ambos participantes en la comunicación.

Resolución de desafío

Lo primero que hicimos al empezar el desafío 10 fue googlear que era quine. De esta manera nos informamos de que un quine program es un programa que produce su código fuente como su salida única. Proseguimos creando un archivo con el nombre 'quine.c'. Nos aparece el siguiente mensaje en consola.

```

----- DESAFIO -----
quine

/usr/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/11/../../../../x86_64-linux-gnu/Scrt1.o: in function `_start':
(.text+0x1b): undefined reference to `main'
collect2: error: ld returned 1 exit status

ENTER para reintentar.

----- PREGUNTA PARA INVESTIGAR -----
¿Cuáles son las características del protocolo SCTP?

```

Con este mensaje entendemos que hace falta un main, por lo tanto creamos un programa basico.

```

TP3-SO > C quine.c
1  #include <stdio.h>
2
3  int main(){
4      return 0;
5  }
6
7

```

Pero al probarlo de nuevo, en consola obtenemos el siguiente mensaje:

```

> #include <stdio.h>
>
> int main(void){
>     return 0;
> }
\ No newline at end of file

diff encontró diferencias.
ENTER para reintentar.

----- PREGUNTA PARA INVESTIGAR -----
¿Cuáles son las características del protocolo SCTP?

```

Recordando el significado de 'quine program' entendimos que debíamos tener un programa que imprima su código fuente para que 'diff encontró diferencias.' no aparezca ya que no debe encontrar desigualdades entre el output y el código del programa. Recordando algunas funciones vistas previamente, generamos el siguiente código:

```

TP3-SO > C quine.c
1  #include <stdio.h>
2  #include <stdlib.h>
3  int main(){
4      char c;
5      FILE *file;
6      file = fopen("quine.c","r");
7      while((c = fgetc(file)) != EOF) {
8          printf("%c", c);
9      }
10     fclose(file);
11     return 0;
12 }

```

Luego obtuvimos la respuesta correcta:

```

----- DESAFIO -----
quine

¡Genial!, ya lograron meter un programa en quine.c, veamos si hace lo que corr
esponde.
La respuesta es chin_chu_lan_cha

----- PREGUNTA PARA INVESTIGAR -----
¿Cuáles son las características del protocolo SCTP?

```

Desafío 11

Respuesta

Los **Request for Comments (RFC)** son documentos numéricos en los que se describen y definen protocolos, conceptos, métodos y programas de Internet, que se gestionan a través de IETF. Una gran parte de los estándares utilizados en Internet están publicados en RFC.

Resolución de desafío

```

----- DESAFIO -----
b gdbme y encontrá el valor mágico

ENTER para reintentar.

----- PREGUNTA PARA INVESTIGAR -----
¿Qué es un RFC?

```

Inicializamos gdb de la siguiente manera “sudo gdb -p pid” y paralelamente corrimos el ./server en otra terminal.

```

male@male-HP:~/Documents/ITBA/2022/SO/TP3-SO$ sudo gdb -p 95648
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
Attaching to process 95648
Reading symbols from /home/male/Documents/ITBA/2022/SO/TP3-SO/server...
(No debugging symbols found in /home/male/Documents/ITBA/2022/SO/TP3-SO/server)
Reading symbols from /lib/x86_64-linux-gnu/libm.so.6...
Reading symbols from /usr/lib/debug/.build-id/27/a8c28af0bfefefcf69ba73f7d4582d82e01f71.debug...
Reading symbols from /lib/x86_64-linux-gnu/libc.so.6...
Reading symbols from /usr/lib/debug/.build-id/89/c3b85f9e55046776471fed05ec441581d1969.debug...
Reading symbols from /lib64/ld-linux-x86-64.so.2...
Reading symbols from /usr/lib/debug/.build-id/aa/1b0b998999c397062e1016f0c95dc0e8820117.debug...
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
0x00007fcd00f58992 in __GI___libc_read (fd=4, buf=0x18c7500, nbytes=4096) at ../sysdeps/unix/sysv/linux/read.c:26
26      ../sysdeps/unix/sysv/linux/read.c: No such file or directory.
(gdb) 

```

Para obtener el pid corrimos el comando “ps -ax” y buscamos el pid de ./server.

```

95041 pts/5    SS      0:00 bash
95648 pts/5    S+      0:00 ./server
95649 pts/5    S+      0:00 ps -o 0 0 0 0

```

En el gdb realizamos continues hasta el desafío 10.

```

(gdb) c
Continuing.
[Detaching after vfork from child process 95758]
[Detaching after vfork from child process 95762]
[Detaching after vfork from child process 95766]
[Detaching after vfork from child process 95770]
[Detaching after vfork from child process 95774]
[Detaching after vfork from child process 95778]
[Detaching after vfork from child process 95782]
[Detaching after vfork from child process 95785]
[Detaching after vfork from child process 95789]
[Detaching after vfork from child process 95792]
[Detaching after vfork from child process 95796]
[Detaching after vfork from child process 95800]
[Detaching after vfork from child process 95807]
[Detaching after vfork from child process 95811]
[Detaching after vfork from child process 95815]

```

Luego, gracias a la pista de la cátedra, corrimos el comando b gdbme para crear un breakpoint


```
20      in 1.7.5386ps/ame/
(gdb) b gdbme
Breakpoint 1 at 0x4019f0
(gdb) □
```

Usando el comando layout asm para visualizar mas fácil el código de assembly, observamos que debíamos setear la variable eax en 0x12345678 para que la misma entre a un jump y podamos obtener la rta correcta. Como se llaman funciones entre medio, decidimos realizar otro breakpoint en la posición del compare para estar seguros. Luego de otro continue observamos lo siguiente:

```
(gdb) c
Continuing.
(gdb) [Detaching after vfork from child process 95929]

Breakpoint 1, 0x00000000004019f0 in gdbme ()
(gdb) b *0x4019fc
Breakpoint 2 at 0x4019fc
(gdb) c
Continuing.

Breakpoint 2, 0x00000000004019fc in gdbme ()
(gdb) set $eax = 0x12345678
(gdb) □
```

```
B+ 0x4019f0 <gdbme>      sub    $0x88,%rsp
0x4019f7 <gdbme+7>      call   0x400cc0 <getpid@plt>
B+> 0x4019fc <gdbme+12>  cmp     $0x12345678,%eax
0x401a01 <gdbme+17>      mov     $0x80,%edx
0x401a06 <gdbme+22>      je      0x401a30 <gdbme+64>
0x401a08 <gdbme+24>      mov     0x2018e9(%rip),%rsi      # 0x6032f8
0x401a0f <gdbme+31>      mov     %rsp,%rdi
0x401a12 <gdbme+34>      call   0x401380
0x401a17 <gdbme+39>      mov     %rax,%rdi
0x401a1a <gdbme+42>      call   0x400c90 <puts@plt>
0x401a1f <gdbme+47>      add     $0x88,%rsp
0x401a26 <gdbme+54>      ret
0x401a27 <gdbme+55>      nopw   0x0(%rax,%rax,1)
0x401a30 <gdbme+64>      mov     0x201a31(%rip),%rsi      # 0x603468
0x401a37 <gdbme+71>      mov     %rsp,%rdi
0x401a3a <gdbme+74>      call   0x401380
0x401a3f <gdbme+79>      mov     %rax,%rdi
0x401a42 <gdbme+82>      call   0x400c90 <puts@plt>
0x401a47 <gdbme+87>      add     $0x88,%rsp
0x401a4e <gdbme+94>      ret
0x401a4f <gdbme+95>      nop
0x401a50 <gdbme+96>      sub     $0x88,%rsp
0x401a57 <gdbme+103>     mov     0x2018ba(%rip),%rsi      # 0x603318
0x401a5e <gdbme+110>     mov     $0x80,%edx
0x401a63 <gdbme+117>     mov     %rsp,%rdi
0x401a66 <gdbme+120>     call   0x401380
0x401a6b <gdbme+127>     mov     %rax,%rdi
0x401a6e <gdbme+130>     call   0x400cf0 <system@plt>
0x401a73 <gdbme+137>     add     $0x88,%rsp
0x401a7a <gdbme+144>     ret
0x401a7b <gdbme+145>     nopl    0x0(%rax,%rax,1)

multi-thre Thread 0x7fcd00e417 In: gdbme
Continuing.

Breakpoint 2, 0x00000000004019fc in gdbme ()
(gdb) set $eax = 0x12345678
(gdb) c
Continuing.
[Detaching after vfork from child process 95989]

Breakpoint 1, 0x00000000004019f0 in gdbme ()
(gdb) c
Continuing.

Breakpoint 2, 0x00000000004019fc in gdbme ()
(gdb) set $eax = 0x12345678
(gdb) c
Continuing.
█
```

male@male-HP: ~/Documents/ITBA/2022... ×

----- DESAFIO -----
b gdbme y encontrará el valor mágico

La respuesta es gdb_rules

----- PREGUNTA PARA INVESTIGAR -----
¿Qué es un RFC?
█

En la terminal del juego obtuvimos la respuesta correcta.

Desafío 12

```
----- DESAFIO -----
Me conoces

0.203626 0.526707 0.818601 -1.147033 -0.868803 1.568106 0.337477 -0.899803 1.071907 0.345313 -1.341817 -0.673604 1.899721 -1.270952 -1.151187 0.293968 -0.287723 1.266671 -1.746699 0.680873 0.280628 1.597405 -0.25
8890 -0.478558 -1.867098 -0.629487 0.340971 0.506900 -0.064349 -1.098067 0.525714 -0.863996 1.299248 -1.311354 -0.897558 0.412332 0.732782 0.706085 -0.616659 -0.323848 -0.719309 -0.176331 -0.125589 -0.748192 -0.
407197 0.835426 -1.045437 0.820944 -0.877486 0.467870 0.863658 -1.036421 0.475398 0.084159 -2.107917 0.574904 -0.839636 -1.046398 -0.491116 1.172917 1.065697 0.350670 0.567923 0.042529 0.338873 -0.253681 -1.4429
52 -1.180646 1.518566 -0.536882 0.412384 -1.031953 -0.184450 0.223843 2.043779 -0.320757 1.373766 0.655563 -1.229406 0.009622 -1.109506 0.722326 0.178454 0.641554 0.380668 -0.910455 -0.450329 -0.123230 1.275594
-1.177391 -1.748134 -0.150704 -1.480267 -0.897898 0.473132 -1.840332 -1.166909 -1.495736 -1.442851 0.057876 0.858451 -0.928642 -0.516734 -0.564448 -0.693937 -0.450240 0.346527 -0.219809 1.376472 0.189531 -0.0536
33 -0.992727 0.997604 -0.392979 -1.454180 -0.075499 -0.879341 -1.765920 -2.143297 -0.602449 -2.514092 1.617690 1.780999 -0.614298 2.408976 0.467981 -0.773996 1.123016 -1.277413 -0.400765 -0.218343 -1.068267 -0.4
46975 0.934582 2.947291 1.381455 -1.056469 1.241263 -0.889992 -0.248790 -0.940842 -0.209657 -0.024631 -0.156965 0.253910 -0.217866 -1.412879 1.783822 -1.627608 0.681766 0.826660 0.538041 0.484572 -0.793445 0.508
179 0.569485 0.764628 -1.172930 0.092643 0.202237 -0.252573 -0.346365 -1.334283 0.716143 -0.178154 -0.137608 0.524454 -1.615070 -0.608371 0.695950 0.459987 -1.616471 -0.480052 0.339255 -0.738030 0.846153 1.038187
-0.854637 -0.442282 -0.021285 -0.271674 -2.934413 0.506666 0.411354 -0.698294 -1.963074 -0.054599 0.031814 0.303182 -1.072827 0.343212 1.282976 0.410776 -0.583112 1.346682 0.709422 -0.909476 -0.961863 -1.01757
2 0.171403 -0.050397 -0.658416 0.043391 0.582150 0.238609 0.055732 -1.021793 -2.149842 -1.159740 0.951802 -0.965517 0.082069 -1.913491 -1.159403 2.054017 0.715312 -0.674343 -0.301882 0.905767 1.045292 0.066299 -
1.818652 -0.289217 0.576588 0.082249 0.549767 -0.403455 1.010647 -1.027727 0.188023 0.040148 -1.078280 -0.198045 1.105418 0.073706 0.400665 -0.450843 0.818414 -0.601187 -0.540691 0.011120 -1.092178 -0.236426 1.
183716 0.655775 1.428327 -2.113548 0.779505 -0.322771 2.091531 0.682453 0.440809 0.280360 0.460930 -1.068831 0.790972 0.641575 -0.700911 0.960783 0.686267 0.679580 0.137262 -0.953708 0.697463 -0.369975 -0.976929
0.470903 -0.431105 -1.525871 -0.378661 0.944673 0.395345 -0.720576 -0.575160 0.149996 0.981530 0.422665 0.180661 0.368136 0.480881 0.825688 -0.113390 -0.288539 0.874046 0.383509 0.931004 -0.573440 -0.352742 -0.
303326 0.135815 0.450512 -1.191550 -1.202437 0.477080 -0.609273 -0.289368 -1.838568 -0.373632 -0.413393 0.480840 1.084912 -0.012644 0.025566 -0.122007 0.189808 0.514083 0.378136 -0.213710 -0.070941 -1.209646 -0.
631559 1.862399 0.379469 -0.206759 -1.623926 -0.539567 -0.832169 0.035862 0.207102 -0.548510 -0.746637 -1.967620 0.719971 0.435837 0.806560 0.673437 0.679926 0.208519 0.688372 2.547317 -0.936655 0.233354 -0.37
5676 0.712689 0.620810 1.045323 -1.125804 1.066727 1.333800 1.208571 1.799953 -0.455861 -0.338592 -0.321140 0.775956 -0.565932 0.251096 -1.364536 0.505920 0.818816 0.052587 -2.504790 -0.839983 -1.109466 0.516717
-1.109847 0.084446 -0.493260 0.303383 -1.271403 1.177488 -0.200112 -0.520067 1.319703 -1.248391 -0.071001 1.477022 -0.765493 -0.100809 -0.342790 -2.278770 2.192579 -0.795073 -0.841482 0.709588 0.703038 0.522581
0.480150 1.233995 0.361994 -1.082904 -1.868530 0.784880 0.642745 1.766572 0.129790 1.861498 -0.316199 0.035342 -0.080959 0.867644 -1.984080 -1.018949 -1.443166 1.718094 0.234011 -0.990735 0.185888 -0.516526 -1.
401625 0.674548 -0.219941 0.352944 -1.035623 -0.754880 -0.567610 0.490888 -0.365288 1.430000 -1.534591 -0.181501 -0.887862 -1.012454 -0.272590 2.026725 0.347387 0.423778 -1.028877 -0.888832 0.867908 2.027250 -1.
229662 0.636091 0.790009 -2.270373 0.907293 -0.328781 -1.268866 0.613887 -0.888894 -2.610052 0.640532 0.908459 -0.062015 0.728713 -0.674353 0.196426 -0.113907 -1.415195 0.448786 0.409456 0.495888 -1.046081 -0.
406061 0.162873 -0.108614 -1.550879 1.233743 2.226439 0.883128 0.212663 -1.014519 -2.493033 0.998477 0.460670 -0.138477 -1.457639 0.618907 -0.548184 0.266378 -0.986020 -0.831157 0.638153 0.382748 0.621609 -2.67
2741 -1.314995 0.010521 1.212472 2.472674 -0.588434 -0.503360 -1.172762 1.264040 0.095887 0.388806 2.026045 0.692891 0.467993 -0.215732 -0.094586 1.275799 0.365728 -1.486566 0.679148 -0.773435 -2.229290 -0.1700
23 0.836543 1.259896 -0.240490 1.493948 1.023571 -0.360826 0.301930 1.340470 -1.061764 -0.058245 -0.343288 1.633398 0.399107 -0.700474 1.375239 -0.048870 -0.519199 0.021948 -0.219803 -1.544030 0.161136 -0.386445
2 0.573278 0.380864 -1.030211 0.671828 0.809516 -0.143393 -0.153344 2.915244 1.363466 -0.649454 -1.018237 0.116859 0.219658 0.745078 -1.033206 -0.245638 0.536525 0.818997 0.410044 1.811148 1.106880 1.457467 -0.6
84579 0.144988 0.293350 -1.002492 -0.586426 0.370423 -0.878906 1.768434 -2.176588 -0.071566 0.568874 -0.339973 0.435221 0.294735 1.520353 -1.131662 1.395365 -1.162230 0.597893 0.914804 1.179314 1.327792 -1.18928
7 -1.712488 -0.368001 -0.423135 1.237908 -0.358691 -1.966723 -0.598643 0.178765 -1.139635 -0.532143 0.729916 -0.912999 0.987351 -0.627136 -0.128268 1.508409 2.018832 -0.325131 1.246803 0.358005 -1.124956 0.464698
2 0.220696 0.299897 0.238407 1.080744 0.111307 -0.485010 0.248003 -2.397499 1.027298 -0.546099 -0.272626 -0.275686 -0.399914 0.809885 -0.266713 -0.417194 0.518293 -0.710927 0.256625 1.092175 0.701133 -1.634151 -0.
003393 -1.149458 1.010050 -0.462177 0.493800 0.588424 -1.022983 2.485883 -0.195591 0.374691 -0.037895 -0.920792 1.555470 1.021142 -1.831725 0.602740 -0.499334 0.661960 1.261021 -1.186252 -0.658793 0.638916 -0.0
32842 -0.563493 -2.641361 -0.732292 -0.462049 0.381036 -0.323872 -0.279438 -0.397544 0.149213 -0.145979 -0.225576 -1.150390 1.050274 2.277766 1.593169 0.681899 -0.558142 0.675136 1.151118 0.896963 2.015320 1.208
335 -0.387188 0.148050 -2.317934 0.557518 -0.421019 0.764782 -0.446717 0.814484 -0.581590 0.842511 1.442734 -2.158967 -1.419444 -0.159419 1.031168 1.424207 0.870636 -0.533223 1.470857 0.125608 1.039122 0.002692
0.480953 0.342538 -1.002376 -1.280395 0.987083 0.199455 -0.530609 -2.035906 -1.020811 0.924060 -0.144089 -1.112494 -0.402676 0.110581 0.222109 -2.601968 1.603598 -0.730772 0.643524 -0.155755 0.981369 1.988496
1.77359 1.100704 0.720750 0.600136 1.665144 -0.856613 -0.608058 -1.704093 0.136263 0.563985 0.997099 0.623732 -0.157920 1.143097 -0.825364 0.633644 0.364280 -0.108521 -1.579588 -0.082275 0.547835 -0.895919 0.969
848 2.479303 1.897686 -1.038594 0.577518 -1.593850 -0.818938 -0.135580 0.421442 -1.832376 1.815923 -1.129286 0.824237 -0.475699 0.076173 -0.108021 1.397342 0.871627 -0.341165 0.249478 0.278419 0.926825 -1.520584
0.201216 -0.968189 -0.190474 -0.891096 0.630724 -0.807825 -0.780891 -0.409531 0.183963 0.919163 0.138787 -0.965926 0.371107 0.623866 -1.492579 -0.538487 1.292627 -1.558115 0.163552 1.699488 1.217054 -1.172951 0.
309995 -0.288050 2.377410 1.138088 0.616097 0.730934 0.861983 -1.302614 1.432665 1.297902 -0.610399 0.345819 1.392431 1.147219 -0.788905 0.474108 -0.558600 0.158079 0.813362 0.608815 -0.412171 -1.774441 -2.7064
95 -0.344151 1.102639 -0.343782 0.319523 -0.325558 2.170590 0.280210 0.054568 -0.021953 -0.523393 -0.637470 -2.121998 0.598791 -0.613740 -1.899105 2.023179 2.601500 -1.229430 0.598846 1.848356 1.851952 -0.017342 2.478150
-2.328941 1.859894 0.548483 1.310353 0.858790 0.594220 0.542891 -0.602026 -1.026995 -0.767189 -0.952154 -0.560425 -1.368858 0.300296 -0.571368 0.055741 0.276914 0.194631 0.179443 -0.818178 -2.137719 -0.516317 -0.
074822 0.234343 0.958590 0.407830 1.610811 -1.206773 0.864627 1.315477 -1.774086 0.302714 -0.380381 -0.103195 -0.451343 0.913576 2.085777 -0.948914 0.525923 -0.373066 0.153730 -0.016655 1.306135 -1.635914 -0.0309
13 1.045348 -0.947286 -1.942357 2.519607 0.356843 -0.523317 0.422809 0.511942 1.426167 0.130994 -1.527733 -0.614328 0.341425 -0.758286 -0.815111 -0.428482 -0.244929 -0.637241 -0.232407 -0.805600 -0.943275 1.0734
56 -0.160375 -1.447593 -0.258831 0.980925 -0.436260 0.319284 0.066713 -0.324664 0.772718 0.460977 -0.419106 0.400316 -2.352114 -1.334258 -0.504278 0.636109 1.690875 0.534538 1.045547 0.111275 -0.145593 1.099936
0.209513 0.425259 1.161170 0.503644 0.946859 0.081993 0.968084 -0.651947 -1.210074 -0.636280 0.528718 1.814644 1.218775 -0.805818 -0.944321 1.501217 -0.584616 1.951717 -1.448205 -0.630498 -0.045617 -0.357882 -1.
128747 -0.546203 -1.154571 -0.277761 0.975722 -0.578872 1.084228 -0.146125 -0.290641 -0.520305 -0.064084 1.485764 1.065119 1.072985 -0.204154 1.282174 0.385432 2.624375 -0.882440 0.481591 -0.498518 0.414114 0.1998
49 -0.554341 0.166591 -0.069710 -0.210264 0.665721 -0.006394 2.136020 -0.718298 -1.609588 0.473214 1.329138 0.676327 -1.557315 1.384527 -1.414482 0.156163 0.088205 0.040711 -0.277523 0.728201 0.628330 1.054872 2.
261765 0.428757 -0.053452 -0.277513 0.303934 0.738129 0.634323 1.992765 0.520225 0.886667 -1.034592 1.776098 -0.005540 1.766985 -0.776630 0.055385 0.202767 0.029666 -0.192452 0.116562 -0.187553 0.370142 0.38922
4 1.611477 -0.286922 1.157427 -0.648500 -1.945107 -0.406630 0.319869 -0.625400 0.014871 -0.598395 0.104710 -0.218536 -1.108252 -0.101902 -0.072995 -0.136581 -0.440934 -1.269288 -0.190784 0.617823 1.363547
```

Respuesta

Si, fue muy divertido. A todo el grupo le gustan los acertijos y juegos de ingenio, por lo cual fue un TP muy interesante y a la vez educativo.

Resolución de desafío

Al mirar los datos en pantalla por un largo rato, pensamos en varias ideas, algunas fueron:

- coordenadas de lugares
- coordenadas para crear una función

Al intentar un buen rato con la primer idea sin llegar a nada, decidimos ir por la segunda.

Ningun miembro del grupo contaba con la aplicacion Octave por lo cual decidimos utilizar octave online : <https://octave-online.net/>.

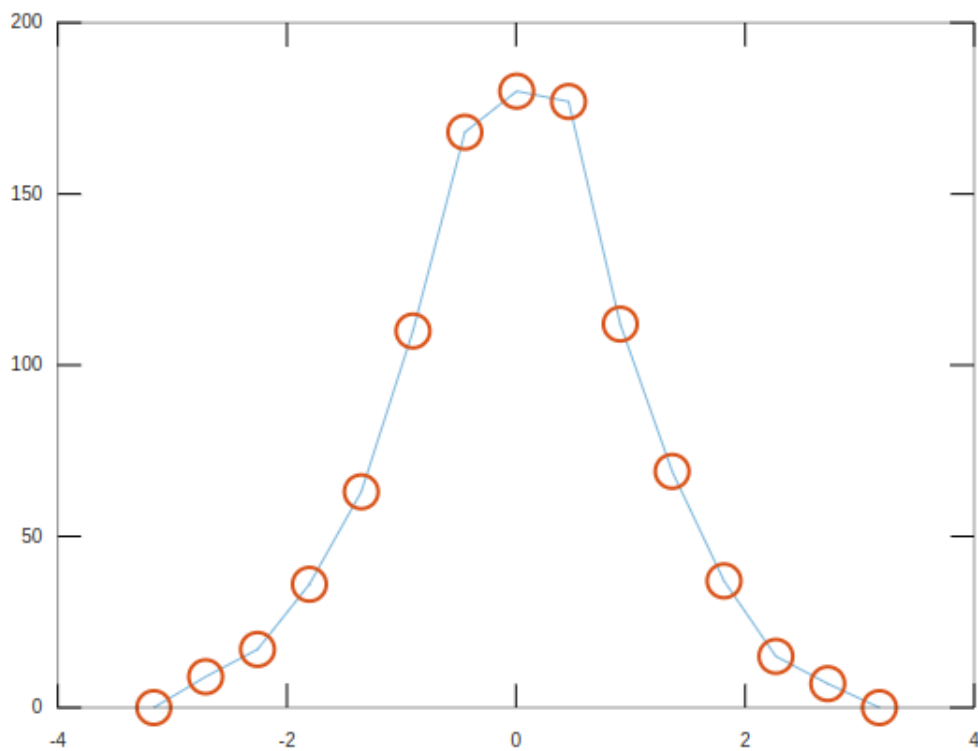
Gracias a la materia “Metodos Numericos” y “Probabilidad y Estadistica” pudimos reutilizar una funcion provista en clase para obtener el grafico de un poligono de frecuencias (con los datos obtenidos en el desafio 12).

Gracias a las siguientes lineas de codigo obtuvimos el grafico:

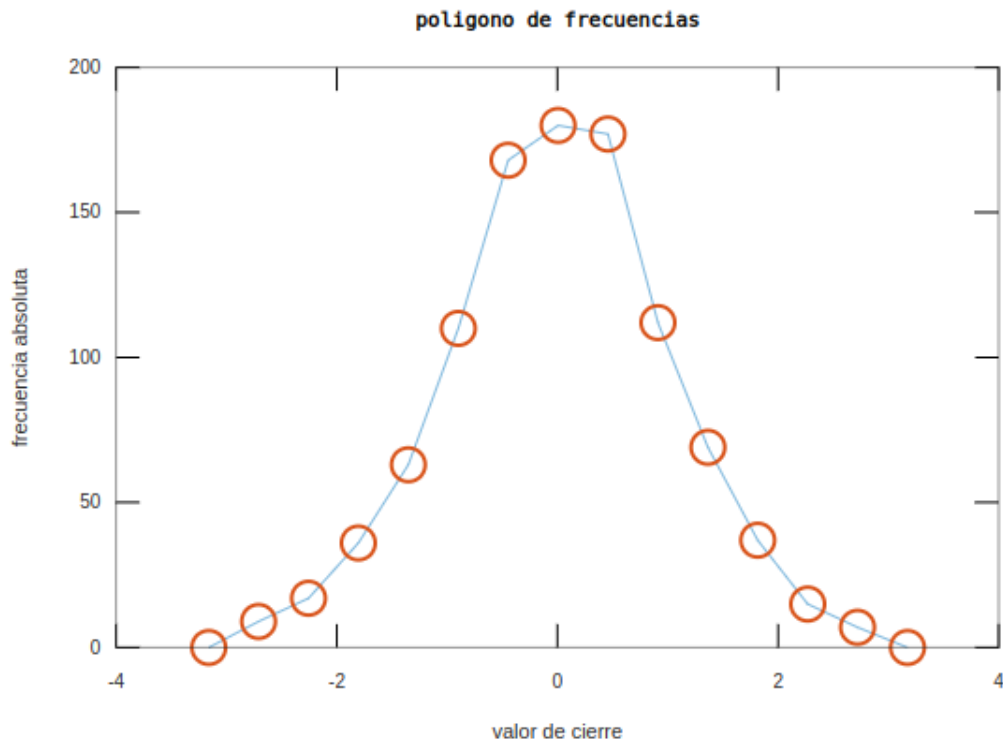
```

octave:19> load datos12.dat
octave:20> n=length(datos12);
octave:21> datos=datos12;
octave:22> cantClases = 13;
octave:23> [frecuencia marca] = hist(datos, cantClases);
octave:24> long = marca(2) - marca(1);
octave:25> y = [0 frecuencia 0];
octave:26> x = marca(1)-long:long:marca(cantClases)+long;
octave:27> plot(x,y,x,y,'o');

```



Luego, agregamos algunos titulos explicativos y finalmente estabamos en condiciones de descifrar la respuesta.



Se nos vino a la mente la palabra “Gauss” pero la misma no era la respuesta correcta. Tampoco adivinamos con la palabra “Campana” (idea proveniente de campana de gauss). De todas formas, relacionandolo con la materia “Probabilidad y Estadística” pudimos relacionarlo con la distribución normal/ distribución de Gauss. A lo que la palabra “normal” fue la respuesta correcta.

Felicitaciones, finalizaron el juego. Ahora deberán implementar el servidor que se comporte como el servidor provisto

Bibliografía

- https://cs.uns.edu.ar/~ldm/mypage/data/rc/apuntes/introduccion_al_uso_de_sockets.pdf
- <https://www.lifesize.com/en/blog/tcp-vs-udp/#:~:text=TCP is a connection-oriented,is only possible with TCP.>