Trabajo Práctico 1 - Grupo 7

(72.07) Protocolos de Comunicación

Entrega: 15/07/2025

Integrantes

- Martina Schvartz Tallone - 62560
- Josefina Míguez - 63132
- Thomas Adrianus Ruijgt - 62875

Índice

1. Descripción de los protocolos y aplicaciones desarrolladas

Servidor SOCKS5

Se implementó un servidor proxy para el protocolo SOCKSv5[RFC1928]. Se maneja las conexiones de forma concurrente y no bloqueante, y en caso de resolución de nombres de dominio se delega la tarea a un thread aparte, de forma que no se bloquee el thread principal y que se puedan seguir atendiendo otras conexiones.

El único modo de autenticación disponible es mediante usuario y contraseña. Se aceptan únicamente requests por parte del cliente cuyo comando sea CONNECT, no hay soporte para BIND o UDP ASSOCIATE.

También implementamos un sistema de loggeo que fue utilizado durante la realización del trabajo y que continúa siendo útil para el funcionamiento de la aplicación. Los logs se imprimen por salida estándar y pueden ser consultados por un administrador mediante el comando LOGS a través del protocolo de monitoreo. En caso de que resulten molestos los mensajes en la salida estándar, es posible configurar el sistema para que solo se muestran aquellos de mayor relevancia. Los niveles de severidad disponibles, en orden ascendente, son: INFO, WARNING, ERROR y FATAL. Cabe destacar que, incluso si se limita la visualización a ciertos niveles, todos los logs se siguen registrando internamente.

Protocolo de configuración y monitoreo

El protocolo de configuración y monitoreo es un protocolo administrativo ligero y basado en texto que permite gestionar y monitorear el servidor SOCKS5 mediante una conexión TCP. Funciona escuchando en el puerto 1081 (a menos que en el servidor SOCKS5 se especifique otro puerto) y utiliza un modelo sencillo de comandos y respuestas similar al del protocolo POP3[RFC1939]. Para usarlo, un cliente se conecta al servidor y debe autenticarse con un usuario y contraseña. Solo los usuarios con privilegios de administrador pueden ejecutar los comandos.

Los comandos son palabras clave que pueden ir acompañadas de argumentos, siempre terminan con un salto de línea y el servidor responde de forma uniforme con un mensaje que empieza con +OK si todo salió bien o -ERR si hubo un error.

Entre sus funcionalidades se encuentran listar, crear y eliminar usuarios, cambiar roles entre administrador y usuario común, y modificar la propia contraseña del

administrador. También se puede consultar la actividad de cada usuario, ver los logs del servidor, obtener estadísticas como número de conexiones activas, conexiones históricas y bytes transferidos, y modificar configuraciones en tiempo real como el límite máximo de usuarios registrados.

2. Problemas encontrados durante el diseño y la implementación:

Durante el desarrollo del trabajo nos encontramos con distintas dificultades. Una de ellas fue el manejo de los sockets de entrada y salida, lo cual implicó varios desafíos técnicos.

La primera vez que corrimos el nuestro script para saber cuántos clientes soportabamos se nos abortaba al llegar a los 110 usuarios en paralelo. Esto se debía a que netcat estaba rechazando las conexiones.

3. Limitaciones de la aplicación

Nuestra aplicación puede manejar hasta un máximo de 500 clientes simultáneos. Esta limitación surge del uso de select(), que opera sobre un buffer de 1024 file descriptors. Como cada cliente requiere dos file descriptors, el límite práctico queda en 512. Sin embargo, ya se usan algunos 5 file descriptors para el socket pasivo del proxy server y del administrador, además de los 3 estándar (stdin, stdout, stderr), lo que reduce el número disponible para clientes.

Por otro lado, los nombres de usuario y contraseñas que superen los 255 caracteres serán denegados.

4. Posibles extensiones

Una posible extensión del proyecto consiste en incorporar una base de datos para los usuarios, lo que permitiría otorgar persistencia a su información. Además, esta base podría emplear mecanismos de cifrado que mejoren la seguridad en comparación con la actualmente provista.

Asimismo, la base de datos podría utilizarse para mantener una “lista negra” de direcciones IP, permitiendo restringir el uso del sistema únicamente a usuarios autorizados.

Se agregaría soporte para los otros dos modos de conexión y otras maneras de autenticación.

5. Conclusiones

El desarrollo de este Trabajo Práctico Especial nos permitió aplicar y profundizar en los conceptos de la materia Protocolos de Comunicación mediante la implementación práctica de un servidor SOCKSv5 mejorado. Además, el protocolo propio de monitoreo y configuración permitió gestionar y supervisar el servidor de forma eficiente, segura y en tiempo de ejecución.

En conclusión, este trabajo fortaleció nuestra comprensión sobre la definición, implementación y utilidad real de protocolos, resaltando su papel fundamental en el desarrollo de sistemas robustos y operativos.

6. Anexo

Ejemplos de prueba:

Para evaluar la carga máxima de nuestro servidor, realizamos el siguiente script y estudiamos la salida del mismo:

```bash
#!/bin/bash
for ((i=1; i<=200; i++)); do
  curl -m 60 -x socks5h://admin:admin@127.0.0.1:1080 localhost:8081 &
  if [ $(($i % 10)) -eq 0 ]
  then
  echo "Iteration $i"
  sleep 1s
  fi
done
```

Guía de instalación

Teniendo todos los archivos dados en la entrega, se debe ubicar en la raíz del proyecto y correr el comando ‘make’. Para que esto funcione debe tener instaladas las últimas versiones de GCC y Make.

```
make clean all
```

Una vez hecho esto se generarán los dos binarios principales, *socks5d* y *client*.
Para ejecutar el *socks5d*, debemos correrlo de la siguiente forma:

`./bin/socks5d` [argumentos]

Los argumentos disponibles se detallan utilizando el flag `-h`, mostrado a continuación:

```
(base) jmiguez@LAPTOP-BD8MFREU:/mnt/c/Users/josef/ITBA 25/Protos/TPE/TP_PROTOS-grupo7$ ./bin/socks5d -h
[2025-07-15T13:52:32] [INFO] [No admins yet, creating admin with name: admin and password: admin]
[2025-07-15T13:52:32] [INFO] [Created user admin]
Usage: ./bin/socks5d [OPTION]...

   -h               Imprime la ayuda y termina.
   -l <SOCKS addr>  Dirección donde servirá el proxy SOCKS.
   -L <conf  addr>  Dirección donde servirá el servicio de management.
   -p <SOCKS port>  Puerto entrante conexiones SOCKS.
   -P <conf port>   Puerto entrante conexiones configuracion
   -u <name>:<pass> Usuario y contraseña de usuario que puede usar el proxy. Hasta 10.
   -v               Imprime información sobre la versión versión y termina.
```

Para el cliente se corre de la siguiente manera: se ejecuta el binario *client* seguido de la dirección ip en la cual está el servidor de monitoreo y el puerto del mismo. Si no se especifican estos valores, se setean los default: 127.0.0.1 y 1081 respectivamente. La forma de utilizarlo se describe haciendo -h:

```
(base) jmiguez@LAPTOP-BD8MFREU:/mnt/c/Users/josef/ITBA 25/Protos/TPE/TP_PROTOS-grupo7$ ./bin/client -h
Uso: ./bin/client [-L host] [-P port] [-h]
Opciones:
  -L host     Dirección IP o hostname del servidor (default: 127.0.0.1)
  -P port     Puerto del servidor (default: 1081)
  -h          Muestra esta ayuda y termina
```

Donde `host` se corresponde al especificado luego del `-L` del *socks5d* y `port` al especificado luego del `-P` del mismo.

Instrucciones para la configuración

Una vez creados los ejecutables, nos conectamos al servidor socks5 y luego al cliente indicando la dirección ip y el puerto configurados anteriormente.

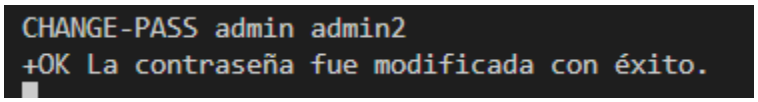
Modo de uso:

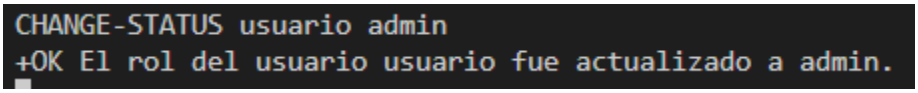
```
(base) jmiguez@LAPTOP-BD8MFREU:/mnt/c/Users/josef/ITBA 25/Protos/TPE/TP_PROTOS-grupo7$ ./bin/client
+OK Dory v1.0
USER admin
+OK
PASS admin
+OK Autenticación exitosa. Privilegios de administrador concedidos.
HELP
+OK Comandos disponibles:
  - LIST-USERS                        Listar todos los usuarios registrados
  - ADD-USER <usuario> <contraseña>   Crear un nuevo usuario
  - DELETE-USER <usuario>             Eliminar un usuario existente
  - CHANGE-PASS <vieja> <nueva>       Modificar su propia contraseña
  - CHANGE-STATUS <usuario> <rol>     Cambiar el rol ("admin" o "commoner") de un usuario
  - USER-LOGS <usuario>               Mostrar accesos registrados del usuario
  - SERVER-LOGS                       Ver los logs del servidor
  - SET-MAX-USERS <numero>            Definir la capacidad máxima de usuarios
  - GET-MAX-USERS                     Consultar la capacidad máxima de usuarios
  - STATS                             Ver métricas de uso del sistema
  - HELP                              Mostrar esta ayuda
  - QUIT                              Finalizar la sesión actual
QUIT
+OK Sesión finalizada. Cerrando la conexión.

Servidor cerró la conexión.

Cerrando cliente...
(base) jmiguez@LAPTOP-BD8MFREU:/mnt/c/Users/josef/ITBA 25/Protos/TPE/TP_PROTOS-grupo7$
```

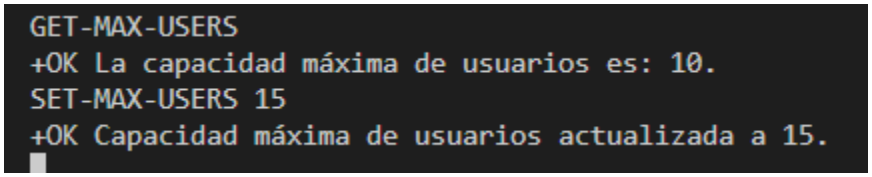
Al realizar la conexión es necesario autenticarse, ya que solo los usuarios con privilegio de administrador pueden utilizar el protocolo de monitoreo, En la imágen también se muestra el comando `HELP` para ver los comandos que ofrece el servidor, y el comando `QUIT` para finalizar la conexión.

Ejemplos de configuración y monitoreo

A continuación se presentan algunos ejemplos de comandos que un administrador podría utilizar para para configurar el sistema a través del manager:

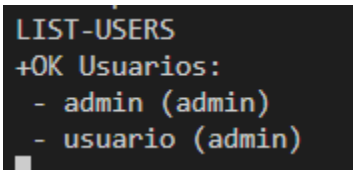






Estos comandos se realizan seguidos en una misma conexión.

Luego, algunos ejemplos de uso de monitoreo:



```
SERVER-LOGS
+OK
2025-07-15T14:46:04 INFO No admins yet, creating admin with name: admin and password: admin
2025-07-15T14:46:04 INFO Created user admin
2025-07-15T14:46:04 INFO Listening on TCP port 1080
2025-07-15T14:46:04 INFO Listening on TCP port 1111
2025-07-15T15:02:29 INFO Created user usuario
2025-07-15T15:05:01 INFO User array resized successfully
---------------------------------
```

```
USER-LOGS admin
+OK 2025-07-15T18:29:49 admin   A       127.0.0.1       27305   google.com      80      Succeeded
2025-07-15T18:30:02     admin   A       127.0.0.1       59067   www.example.com 80      Succeeded
```

```
STATS
+OK
-Total usuarios: 2
-Conexiones concurrentes: 0
-Conexiones históricas: 2
-Bytes transferidos: 2824
```

Documento de diseño del proyecto

Optamos por utilizar TCP como protocolo de transporte ya que estamos más familiarizados con él por lo visto en clase, y creemos que cumple con todo lo que se requiere al protocolo a implementar desde la consigna.

El servidor proxy se maneja a través de una máquina de estados la cual itera entre las diferentes fases del handshake y de la comunicación. Se pasa por la fase de negociación del método de autenticación, luego se pasa por la fase de autenticación, se procede entonces a hacer la request (por parte del cliente) y en caso exitoso se inicia la comunicación entre el servidor y el cliente.