Protocolos de Comunicación

Trabajo Práctico Especial

Integrantes

Vera, Juan Sebastián	53852
López, Noelia Belén	53774
Sevilla, Julián Ignacio	53876
Barrachina, José Agustín	53790

Repositorio Bitbucket

pc-2017-03

Indice

Introducción

Se buscó aplicar un servidor proxy que permite interconectar diferentes hosts en una red y comportarte como un intermediario entre ellos. El mismo puede ser utilizado como un simple canal de comunicación o realizar tareas más complejas ya sea actuar como filtro de contenido, servidor de caché, gestionador y balance de carga en otros servidores, sistema de seguridad, etc.

El proxy implementado soporta el protocolo HTTP versión 1.1. (Hypertext Transfer Protocol) [RFC2661] que pueda ser usado por un User Agents tal como curl, Mozilla, Edge, Google Chrome, etc. para navegar por internet.

Éste sistema está preparado para soportar múltiples conexiones entrantes y salientes, así como gran transferencia de datos.

Descripción detallada de los protocolos y las aplicaciones desarroladas

Para la implementación del proyecto se utilizó el lenguaje **Java Platform SE 8 Release**. Como entorno de desarrollo se utilizó **IntelliJ IDEA** de JetBrains.

Se utilizó **BitBucket** para como un repositorio privado dispuesto por la cátedra.

Se utilizó **Google Docs** para la creación del informe.

Para la manipulación de imágenes se utilizó **ImageIO** lo cual permitió realizar los giros de 180° en tiempo de ejecución.

Arquitectura

El proxy que se implementó le provee al cliente HTTP algunas funcionalidades adicionales que no son características de dicho protocolo. Alguna de ellas son la manipulación de los mensajes cambiando los caracteres l33t

Para poder dar con el objetivo se decidió recrear el siguiente escenario:

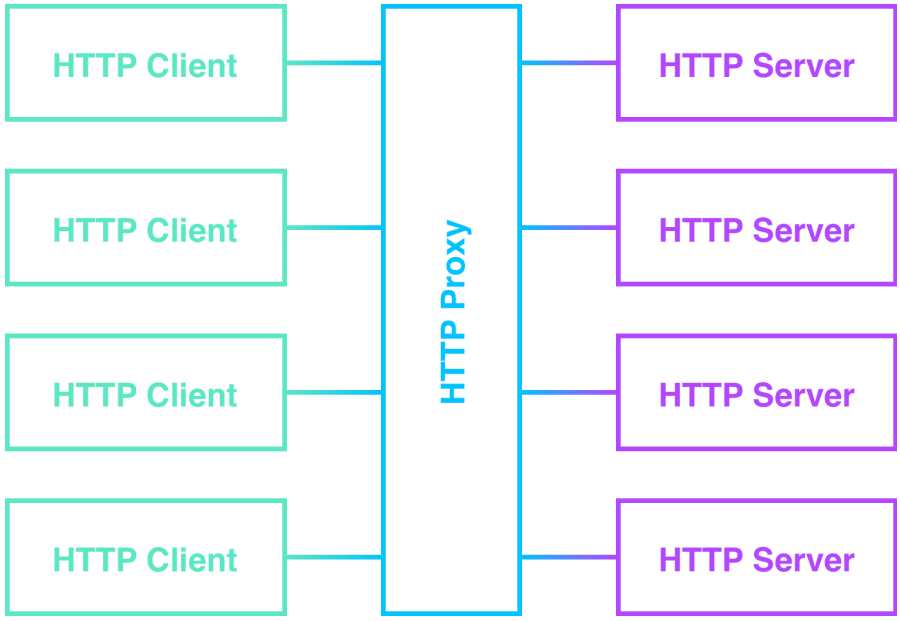


Figura 1: Estructura general del proxy y sus conexiones a los clientes y Servidores HTTP

Conexiones

Cuando el cliente quiere conectarse al servidor HTTP, en realidad se está conectando al proxy el cual le responde como si fuera el mismo servidor HTTP. Lo que hace el proxy es simplemente reenviar lo que el cliente le intenta mandar al servidor y viceversa.

Para poder mantener la conexión entre cliente-proxy y proxy-servidor, el proxy cuenta con una clase llamada ProxyConnection donde se haya toda la informacion relevante de dicha conexión.

Flujo de conexión cliente, proxy y servidor

Para cada cliente que se conecta el proxy cuenta con una clase llamada ProxyConnection donde se guardan los datos de dicha conexiones sean:

- **private SocketChannel clientChannel;**
  Socket que identifica la conexion entre el cliente HTTP y el Proxy

- **private SocketChannel servidorChannel;**
  Socket que identifica la conexion entre el servidor HTTP y el Proxy

- **private SelectionKey servidorKey;**
  Key que identifica la conexion entre el servidor HTTP y el Proxy

- **public ByteBuffer buffer;**
  Buffer para lectura y escritura

- **public HttpMessage httpMessage;**
  Guarda el estado del mensaje que está siendo enviado

- **private SelectionKey clientKey;**
  Key que identifica la conexión entre el cliente HTTP y el Proxy

- **private ConnectionType type;**
  Para poder determinar si la conexión es procedente de un cliente HTTP o de un admin, se implementó el uso de un ConectionType

Registros de acceso

Sobre los logs

Para implementar los logs del proxy se decidió utilizar la librería Logback con el siguiente patrón para los logs:

```
%date{yy-MMM-dd HH:mm:ss}\t%-5level\t%msg%n
```

Los logs son guardados en un archivo en el root del proyecto llamado http-server.log.

Cada log contiene: fecha de creación, tipo de log (debug, info, error) y un mensaje.

A continuación se detallan todos los logs del proxy:

Generales

- Inicio del proxy
- Errores de bind exception
- Problemas si no se puede correr el proxy
-

Admin Logs

- Error al manipular las keys
- Cuando se termina de leer una conexión
- Errores de lectura
- Admin conectado

Client logs

- Error al manipular las keys
- Cuando se termina de leer una conexion
- Errores de lectura
- Nueva conexion al remote server
- Cliente conectado

Protocolo de administración

Se desarrolló un protocolo que funciona para administrar ciertas características y funcionalidades del proxy. El mismo es del tipo REQUEST-RESPONSE y requiere la autenticación del usuario (que es parecido a POP3), para poder consultar o modificar configuraciones.

Definición en formato ABNF

El protocolo de administración no es *case sensitive* y todos los comandos válidos del mismo son de la forma

COMMAND sp VALUE

Donde:

COMMAND =

“get” | “log” | “add” | “remove” | “user” | “pass” | “host” | “port” | “set”

sp =

espacio

VALUE =

“help” | “all-metrics” | “bytes-sent” | “bytes-received” | “method-histograms” | “converted-chars” | “flipped-images” | “total-accesses” | “out” | USER | PASS | “user” | “blacklist” | HOST | PORT | “leet-on” | “leet-off” | “flip-on” | “flip-off”

USER =

Alphanumeric

PASS =

Alphanumeric

HOST =

Alphanumeric

PORT =

DIGIT +

Si el comando no respeta el formato, será considerado automáticamente como comando incorrecto. De todas formas, no todas las combinaciones de *COMMAND VALUE* son válidas.

Lista de comandos validos

USER <username>

- Registra el estado *entered user* con el usuario *username.*
- Si se ejecuta bajo el estado *adding user*, verifica si el usuario ya existe, si no registra el estado *new user ok* con el usuario *username.*

PASS <password>

- Si no hay un usuario ingresado devuelve error
- Si se ejecuta bajo el estado *adding user,* crea un nuevo usuario con username y password, si no intenta loguearse con los mismos parámetros.

LOG OUT [require login]

- Si se está logueado desloguea, de lo contrario retorna un error.

ADD USER [require login]

- Pone al sistema en el estado *adding user.*

ADD BLACKLIST [require login]

- Pone al sistema en el estado *adding blacklist.*

HOST <host> [requiere login]

- Si se ejecuta bajo el estado *adding blacklist*, agrega *host* a la lista de hosts bloqueados.
- Si se ejecuta bajo el estado *removing blacklist*, remueve *host* de la lista de hosts bloqueados.

PORT <port> [requiere login]

- Si se ejecuta bajo el estado *adding blacklist*, agrega *port* a la lista de hosts bloqueados.
- Si se ejecuta bajo el estado *removing blacklist*, remueve *port* de la lista de hosts bloqueados.

GET BYTES-SENT [require login]

- Retorna la cantidad de bytes enviados por el proxy desde su inicio.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “Transferred Bytes:” sp NUMBER \r\n

GET BYTES-RECEIVED [require login]

- Retorna la cantidad de bytes recibidos por el proxy desde su inicio.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “Received Bytes:” sp NUMBER \r\n

GET METHOD-HISTOGRAMS [require login]

- Retorna la cantidad de veces que el proxy intervino los métodos HEAD, GET y POST desde su inicio. Si hubo otro tipo de métodos que intervinieron, se muestran a continuación.
- Si la *response* es correcta (200 OK), por línea, se muestra de la siguiente manera
    - METHOD sp “responses:” sp NUMBER

GET CONVERTED-CHARS [require login]

- Retorna la cantidad de caracteres convertidos por el proxy desde su inicio.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “Converted chars:” sp NUMBER \r\n

GET FLIPPED-IMAGES [require login]

- Retorna la cantidad de imágenes rotadas por el proxy desde su inicio.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “Flipped images:” sp NUMBER \r\n

GET TOTAL-ACCESSES [require login]

- Retorna la cantidad de accesos totales al proxy desde su inicio.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “Total accesses:” sp NUMBER \r\n

GET ALL-METRICS [require login]

- Retorna todas las métricas detalladas anteriormente

SET LEET-ON [require login]

- Habilita la conversión de caracteres.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “The leet converter is on\r\n

SET LEET-OFF [require login]

- Deshabilita la conversión de caracteres.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “The leet converter is off\r\n

SET FLIP-ON [require login]

- Habilita el volteo de imágenes.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “The image flip converter is on\r\n

SET FLIP-OFF [require login]

- Deshabilita el volteo de imágenes.
- Si la *response* es correcta (200 OK) Se muestra de la siguiente manera
    - “The image flip converter is off\r\n

Funcionalidades

El sistema no requiere configuración inicial para comenzar a funcionar aunque el mismo puede ser editado modificando el archivo config.properties.

El administrador puede si lo desea cargar un usuario y contraseña o modificar los puertos por defecto los cuales son:

- 9090: Servidor Proxy HTTP
- 9091: Sistema de Configuración Remota
- 9091: Reporte de Métricas

  Para las instrucciones de cómo configurar el proxy referirse a la sección 9.

Para utilizar el proxy en el navegador (Google Chrome, Mozilla, Edge, etc) habrá que configurar primero el mismo como la configuración de proxy transparente [RFC1919].

Concurrencia y Disponibilidad

Como ya se explicó anteriormente en la Introducción, el servidor soporta múltiples clientes de forma concurrente y simultánea.

Se decidió implementar el proxy bajo un modelo de ejecución no-bloqueante (Java NIO), el cual reduce drásticamente el cuello de botella provocado por el acceso concurrente de múltiples clientes. De no implementarse, el acceso de numerosos clientes, terminaría consumiendo excesivas cantidades de memoria y reduciría el tiempo de respuesta percibido significativamente.

Fallos

En el caso que se produzca algún error del lado del cliente. Éste se informa de alguna de las siguientes formas:

- El comando ingresado es inválido
    - 400 - Wrong Command
- El *username* y *password* ingresados no coinciden con ningún usuario almacenado.
    - 401 - Unauthorized
- El comando ingresado requiere login (y no se está logeado)
    - 401 - Unauthorized
- Se intentó logear con un usuario estando ya logeado.
    - 409 - Conflict
- Se intentó crear un usuario con un *username* ya utilizado
    - 409 - Conflict
- Error inesperado
    - 500 - Internal error

Métodos Implementados

El proxy soporta, por lo menos, los siguientes métodos:

- GET
- HEAD
- POST
- DELETE
- OPTIONS
- PUT

Transformaciones

Se realizaron algunos conversores, en el cual el proxy convierte la información re querida por el usuario antes de realizar el pasaje de información del servidor host al cliente. Las siguientes conversiones se detallan a continuación.

Las transformaciones se realizan en tiempo de ejecución sin reiniciar el servidor.

Conversor l33t

Existe la posibilidad de convertir los mensajes salientes según el modelo leet que transforma de la siguiente manera los caracteres:

Carácter Original	**Carácter Convertido**
a	4 (cuatro)
e	3 (tres)
i	1 (uno)
o	0 (cero)
c	< (signo menor)

Tabla 1: Manera en la cual el conversor transforma los caracteres de los mensajes salientes

Para poder realizar lo antedicho, se creó un *Conversor,* por el que pasan todos los mensajes, que obtiene solo el cuerpo del mensaje y realiza la transformación de los caracteres, en caso de estar habilitada.

Conversor de Imágenes

Se implementó un conversor de imágenes para los formatos “png”. El mismo se encarga de dar vuelta la imagen respecto al eje horizontal o respecto al eje vertical a petición. Para implementar dicho conversor se hizo uso de la librería **ImageIO** la cual brindó ayuda para poder leer y escribir las imágenes.

Métricas

A partir de que el proxy es iniciado, se empiezan a recolectar métricas que luego pueden ser consultadas mediante el administrador. Las métricas son almacenadas en memoria y proveen estadísticas de la sesión actual del proxy.

Las métricas disponibles son:

Accesos

Como accesos realizados se entiende la cantidad de veces que el proxy ha aceptado una conexión

Bytes enviados

Como bytes enviados se entiende la cantidad de bytes que fueron enviados por el proxy

Bytes recibidos

Como bytes recibidos se entiende la cantidad de bytes que fueron enviados por el proxy

Caracteres convertidos

Como caracteres convertidos se entiende la cantidad de caracteres que fueron convertidos mediante leet. (Explicado en la sección 2.5.)

Las métricas son devueltas en formato JSON para que puedan ser analizadas por otras aplicaciones.

```
rhosko@rhosko-VirtualBox:~/Protos/http-proxy$ nc 127.0.0.1 9091
200 OK
Admin Panel (type: GET HELP if you need it)
********************************************************************************
user juan
200 OK
Enter Password (type: pass [PASS])

pass juanjuan123
200 OK
You are know logged in

get all-metrics
222 Metrics Requested
Received Bytes: 0
Total accesses: 0
Transferred Bytes: 0
Converted Chars: 0
Flipped Images: 0
HEAD requests: 0
GET requests: 0
POST requests: 0
```

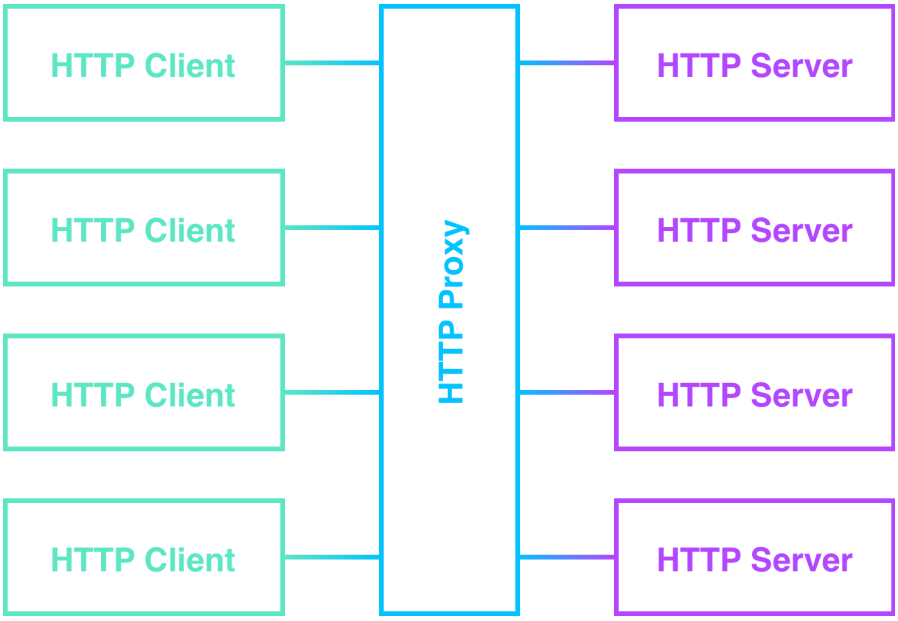
Monitoreo Remoto

El proxy puede ser monitoreado y configurado desde el puerto 9091. Desde aquí se pueden obtener las métricas recolectadas y configurar las propiedades del proxy.

Para acceder al mismo es necesario autenticarse.

Documento de diseño del proyecto

Este proxy cuenta con la siguiente arquitectura:



En primer lugar en la columna izquierda vemos todos los clientes que están conectados a nuestro proxy mediante un socket channel. Cada socket channel es distinto según el cliente.

En segundo lugar en la columna de la derecha vemos otros sockets channels que simbolizan la conexión de cada cliente con el servidor HTTP. Cada socket es distinto según el cliente.

Es por esto que cada fila es el flujo entero de una conexión. Dos sockets channels, uno que va de cliente a proxy y otro que va de proxy al servidor HTTP.

El nuevo cliente se conecta al servidor Socket Channel del proxy. Este último se encarga de aceptar la conexión pedida por el cliente y al hacerlo se abre un nuevo socket entre cliente y proxy como se detalla anteriormente.

Problemas Encontrados durante el diseño y la implemetancion

Buffers

A la hora de implementar el proxy, muchos conflictos con el uso de buffers fueron encontrados. A partir de los mismos, se tuvo que tomar una decisión, entre usar dos buffers (uno para escritura y uno para lectura) o uno mismo para ambas acciones. Al final, se optó por utilizar un solo buffer, haciendo que las conexiones sean más lentas, sobretodo si la cantidad de requests es larga y / o si el largo de las mismas (o de las responses) es grande.

Conversores

Caracteres

Es importante remarcar que la conversión de caracteres debe hacerse por bytes para no generar conflictos a la hora de enviar los datos. Sobre todo con la conversión del caractér '<'.

Imágenes

Al principio se intentó implementar el flip de la imagen de forma manual. Se logró dividir cada "Chunk" de la imágen y obtener información pertinente de los header sabiendo donde se encontraba el array de pixels a invertir. Sin embargo, los mismos se encuentran comprimidos según la información presente en los header la cual es muy variante. Ésto complicó dicha implementación y se pasó al uso de librerías que aplican ayudan a decodificar el array de pixels (librerías basadas en jzlib).

Limitaciones de la Aplicación

Los conversores poseen ciertas limitaciones.

El conversor de imágenes por su parte, solo funciona para archivos png. No se implementó una conversión para archivos jpeg ni ningún otro formato. Si bien no se está muy lejos de implementarlo

El conversor leet no se aplica a archivos compresos ya que debemos descomprimir el mensaje. Ante mensajes con compresión gzip, el header se remueve, por lo que el mensaje llega descomprimido y puede ser convertido.

También posee restricción sobre cualquier contenido que contenga transfer-encoding. Si el mensaje posee este header, nunca se aplica ninguna transformación.

Posibles Extensiones

- Soportar más de un usuario de admin
- Modificar más de una propiedad del archivo de configuración dentro del mismo comando. Hoy por hoy para modificar una propiedad hace falta ejecutar un comando por cada modificación.
    - Incluso se podría pensar en modificar el tamaño del buffer u otros aspectos que ahora no son modificables.
- Soportar la conversión de imágenes de otro formato distinto de png.

Conclusiones

Fue muy interesante poder entender cómo funciona un proxy y todas sus complejidades que pueden surgir en su implementación.

Entender cómo hacer para que el proxy funcione como un pasamanos, que fue el primer paso para un proxy funcional, fue, sin lugar a dudas, el paso más difícil. Una vez entendido eso, y entendido como realizar la arquitectura, el resto fue un poco más sencillo.

El resultado se cree, fue muy satisfactorio más allá de las posibles extensiones y las limitaciones.

Se pudo comprobar que la implementación de un servidor proxy lleva mucho tiempo y esfuerzo, además de un diseño complejo en lo que respecta a arquitectura.

Ejemplos de prueba

- http://localhost:9092
  - HTTP Server provisto por la catedra
- Localhost:9090
  - HTTP Proxy

- **Get respuestas 200**
  - curl -i -X GET http://localhost:9092/api/other -x localhost:9090

- **Get respuestas 204**
  - curl -i -X DELETE http://localhost:9092/api/other -x localhost:9090
  - curl -v -x 127.0.0.1:9090 http://www.mocky.io/v2/593f5c5b100000591847f373

- **Soporte método DELETE**
  - curl -i -X DELETE http://localhost:9092/api/other -x localhost:9090

- **Soporte POST básico**
  - curl --data-binary '¡¡hola mundo!!' -X POST http://localhost:9092/api/other -i -x localhost:9090

- **Soporte POST grande**
  - [Asegurarse de que esté el archivo] curl -d "@test.json" -H "Content-Type: application/json" -X POST http://httpbin.org/post -x localhost:9090 -i

- **Soporte PUT básico**
  - curl --data-binary 'estamos probando el put' -X PUT http://localhost:9092/api/other -i -x localhost:9090

- **Soporte PUT grande**

    - [Asegurarse de que esté el archivo] curl -d "@test.json" -H "Content-Type: application/json" -X PUT http://httpbin.org/put -x localhost:9090 -i

- **Soporte método HEAD**
    - curl -i -X HEAD http://localhost:9092/api/chunked/leet -x localhost:9090
    - curl -i --head http://localhost:9092/api/chunked/leet -x localhost:9090

- **Soporte GET condicional**
    - [esto retorna 200 OK] curl --silent --head http://md.incommon.org/InCommon/InCommon-metadata.xml -x localhost:9090
    - [Esto debería retornar 304 Not Modified -- chequear la fecha con el anterior] curl --silent --head http://md.incommon.org/InCommon/InCommon-metadata.xml -x localhost:9090 --header 'If-Modified-Since: Mon, 12 Jun 2017 20:01:32 GMT'
    - [Esto debería retornar 304 Not Modified -- chequear el tag con el anterior] curl --silent --head http://md.incommon.org/InCommon/InCommon-metadata.xml -x localhost:9090 --header 'If-None-Match: "2aa4056-551c8c919e01c"'

- **Múltiples requests en una misma línea**
    - curl -s -x localhost:9090 -i http://localhost:9092/un.jpg |head -n 7 && curl -s -x localhost:9090 -i http://localhost:9092/leet.txt |head

- **Leet con content-length**
    - [Asegurarse que este el leet prendido] curl -i -x localhost:9090 http://private-25675d-httpproxy.apiary-mock.com/text

- **Leet con chunked**
    - curl -s -x localhost:9090 -i http://localhost:9092/api/chunked/leet|head

- **Leet con chunked y gzip**
    - [Elimina el header del gzip → no va a aparecer en los headers] curl -s -x localhost:9090 -i http://localhost:9092/api/chunked/leet|head

- **Conexión a origin server no resoluble**

    - curl -s -x localhost:9090 -i http://protossofun.com/

- **Conexión a origin server no disponible**
    - User agent

Guía de Instalación

Para instalar el proxy debe clonar el repositorio que se encuentra en https://bitbucket.org/itba/pc-2017-03 y seguir las instrucciones del readme.

README

```
# http-proxy

## Hi! And welcome to the fast 1-minute setup tutorial!

We know you already want to change those characters and flip some images, but we promise you are under 60 seconds away from there!

- First clone or download our project by clicking on this button above.


![clonedownload](https://user-images.githubusercontent.com/12797703/27065504-0b818bc6-4fd4-11e7-8821-51bda452398c.png)

- Then get into the http-proxy directory. Once there run  ```  ./jar-generator.sh  ```


![console](https://user-images.githubusercontent.com/12797703/27065636-dbd3a30e-4fd4-11e7-97c1-9f071e19ba21.png)

### Yay! We're almost there


- Now you just need to run ```java -jar http-proxy.jar``` to get everything running!

  #### After thet your console should look like this


![started](https://user-images.githubusercontent.com/12797703/27065789-ce3f6466-4fd5-11e7-8633-82a5c0549cec.png)

## Hurrah! Your proxy is up and running!
```

Ejemplos de configuración y monitoreo

- Usuario o contraseña incorrectos

```
rhosko@rhosko-VirtualBox:~/Protos/http-proxy$ nc 127.0.0.1 9091
200 OK
Admin Panel (type: GET HELP if you need it)
********************************************************************************
user juan
200 OK
Enter Password (type: pass [PASS])

pass hola
401 Unauthorized
Username and password doesn't match
```

- Login satisfactorio

```
user juan
200 OK
Enter Password (type: pass [PASS])

pass juanjuan123
200 OK
You are know logged in
```

- Comando incorrecto seguido del correcto (SET LEET-ON)

```
set leet on
400 Wrong command
Check your spelling

set leet-on
200 OK
The leet converter is on
```

- Algunas métricas (Con uso del proxy entre llamados a GET ALL-METRICS)

```
get all-metrics
222 Metrics Requested
Received Bytes: 0
Total accesses: 0
Transferred Bytes: 0
Converted Chars: 0
Flipped Images: 0
HEAD requests: 0
GET requests: 0
POST requests: 0

get all-metrics
222 Metrics Requested
Received Bytes: 349
Total accesses: 1
Transferred Bytes: 349
Converted Chars: 0
Flipped Images: 0
HEAD requests: 0
GET requests: 1
POST requests: 0

get all-metrics
222 Metrics Requested
Received Bytes: 1281
Total accesses: 3
Transferred Bytes: 1281
Converted Chars: 0
Flipped Images: 0
HEAD requests: 0
GET requests: 3
POST requests: 0
```

- Flipping some chars

```
L0r3m 1psum g1l4d4.
get converted-chars
222 Metrics Requested
Converted Chars: 6
```

Bibliografia

- https://gist.github.com/yfnick/227e0c12957a329ad138
- https://docs.google.com/document/d/1J7llC_wYNht8orpcmHh6kUdnDvN3_JYnR-MT3oZne8U/edit#heading=h.4y6fbnl2x9lm
- https://github.com/heyimnowi/xmpp-proxy
- https://codepen.io/rileyjshaw/pen/ufEIH
-