HTTP Request:

Lo primero que debemos hacer es un cat:

```
cat access.log
```

Una vez hecho esto, buscaremos qué significa cada columna que se nos devuelve. Al hacerlo identificaremos las columnas de estatus y tamaño de respuesta.

Si nos fijamos bien, nos daremos cuenta que casi todos los valores de las columnas se repiten, para asegurarnos realizaremos el siguiente comando, que nos permitirá ver todos los valores únicos de la columna seleccionada:

```
awk '{print $[Número de la columna que queramos ver]}' access.log | sort -n | uniq
```

Al ejecutarlo en las columnas 9 y 10, que corresponden a estatus y tamaño de la respuesta, veremos sólo dos resultados:

- Estatus de la respuesta: 200 y 400.
- Tamaño de la respuesta: 192 y 140.

Y si ejecutamos estos comandos:

```
awk '$10 == 192 && $9 == 200 {print $0}' access.log | sort -n | uniq
```

```
awk '$10 == 140 && $9 == 400 {print $0}' access.log | sort -n | uniq
```

Veremos que las respuestas con estatus 200, siempre tienen un tamaño de 192 bytes. Y lo mismo ocurre con las de estatus 400, que siempre tienen un tamaño de 140 bytes.

Pero es que además, parece que las respuestas de estatus 200 siempre se dan tras un número de respuestas de estatus 400, nunca hay dos respuestas de estatus 200 seguidas. Así que para confirmarlo, escribimos un script de python que nos permita comprobar este hecho:

```
log = open("./access.log", "r").readlines()
num400 = 0
output = open("output.txt", "a")
output.truncate()
for line in log:
   if(line.find("400 140") != -1):
      num400 += 1
   else:
      output.write(str(num400)+"|")
      num400 = 0
output.write("\n")
output.close()
```

Tras ejecutarlo, nos devolverá lo siguiente:

```
7|5|7|3|6|5|7|2|7|3|2|12|6|6|6|12|6|1|6|7|6|9|6|4|2|12|6|6|6|12|6|1|6|7
|3|1|6|6|6|12|6|1|6|7|7|11|6|12|3|0|6|7|5|15|3|4|6|14|3|4|6|12|7|9|7|10
|3|3|7|2|5|15|6|7|3|3|3|3|6|11|7|13|
```

Esto nos indica cuántas respuestas de estatus 400 hay entre las respuestas de estatus 200. Y al observar de cerca, podremos apreciar que nunca hay 0 o más de 15 respuestas. Así que probamos a pasarlo a hexadecimal con un script de python:

```
log = open("./access.log", "r").readlines()
num400 = 0
plaintext = ""
for line in log:
    if (line.find("400 140") != -1):
        num400 += 1
    else:
        plaintext += hex(num400)[2:]
        num400 = 0
print (plaintext)
```

Y obtenemos lo siguiente:

```
75736572732c666c616769642c666c616731666c61677b6c30675f346e346c797a337
25f6733336b7d
```

Sin embargo, al estar en hexadecimal no podemos entenderlo, por lo que deberemos convertirlo a un lenguaje que si podamos comprender, el cual sería el código ASCII:

```
plaintext =
"75736572732c666c616769642c666c616731666c61677b6c30675f346e346c797a337
25f6733336b7d"
Flag = bytes.fromhex(plaintext).decode('utf-8')
print(Flag)
```

Y con esto, ya si tendríamos la flag:

```
users,flagid,flag1flag{l0g_4n4lyz3r_g33k}
```