

Nama Tim: bybit 1.0

Anggota Tim: Ananda Hijrah Dwi Aliansah

Khafi Miftahul Syifa

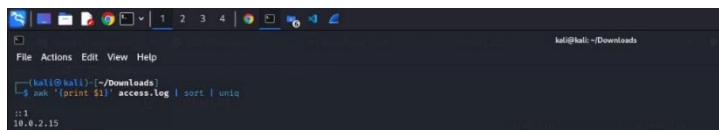
Enuma

Pada write-up ini, kita akan membuka dan menganalisis file log access.log dari sebuah web server untuk mendeteksi berbagai jenis serangan yang mungkin dilakukan oleh pengguna atau penyerang. Langkah-langkah berikut mencakup identifikasi IP unik yang mengakses situs, deteksi upaya penyerangan seperti brute force dan SQL injection, serta identifikasi alat yang digunakan oleh pelaku seperti dirbuster, gobuster, feroxbuster. Selain itu, kita akan mencoba menemukan bendera (flag) tersembunyi yang mungkin disematkan dalam log tersebut.

1. Memeriksa IP Unik yang Mengakses Situs

Tujuan: Mengidentifikasi IP unik yang mengakses situs web untuk mendeteksi apakah ada IP yang mencurigakan.

Perintah:



```
kali@kali:~/Downloads
File Actions Edit View Help
~(kali@kali):~/Downloads
$ awk '{print $1}' access.log | sort | uniq
10.0.2.15
```

Penjelasan:

- Perintah awk '{print \$1}' digunakan untuk mengambil kolom pertama dari setiap baris di access.log, yang berisi alamat IP pengakses.
- sort mengurutkan daftar IP.
- uniq menghapus duplikat, menampilkan hanya IP unik.

Analisis: Disini kita menemukan ip unik 10.0.2.15. Dengan melihat daftar IP unik, kita bisa memeriksa apakah ada IP dari luar jaringan internal yang tidak dikenal yang mungkin melakukan aktivitas mencurigakan.

2. Menghitung Jumlah IP Unik

Tujuan: Mengetahui jumlah total IP unik yang mengakses situs untuk analisis lebih lanjut.

Perintah:



```
~(kali@kali):~/Downloads
$ awk '{print $1}' access.log | sort | uniq | wc -l
2
```

Penjelasan:

- wc -l menghitung jumlah baris dari hasil perintah sebelumnya, yang mewakili jumlah total IP unik.

3. Memeriksa IP Eksternal

Perintah:

```
kali@kali:~/Downloads$ awk '{print $1}' access.log | grep -vE "(10\.\.108\.\.172\.)" | sort | uniq
```

- `grep -vE "(10\.|192\.168|172\.)"` mengecualikan IP dari jaringan lokal.
- `sort` dan `uniq` digunakan untuk mengurutkan dan menampilkan IP eksternal yang unik.

4. Mendeteksi Upaya Penyerangan

Perintah:

```

kali@kali: ~/Downloads
└─$ grep -iE '(linux|system)' /select /etc/passwd|base64 -i|script.py | access.log
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /HTF/1.1" 200 3380 "-" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 200 4844 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /Favicon.ico HTF/1.1" 404 487 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /HTF/1.1" 200 3380 "-" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 200 4844 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst HTF/1.1" 200 2015 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst.jpg HTF/1.1" 200 2224 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst.jpg HTF/1.1" 200 2097 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wavatar.jpg HTF/1.1" 200 1137 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 200 4844 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst.jpg HTF/1.1" 200 1839 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst.jpg HTF/1.1" 200 6356 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 200 4844 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /admin.php HTF/1.1" 300 165 "-" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /HTF/1.1" 200 3380 "-" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst HTF/1.1" 200 5593 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst HTF/1.1" 304 248 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wcsst.jpg HTF/1.1" 304 248 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 304 249 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/wavatar.jpg HTF/1.1" 304 249 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 304 249 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 304 249 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 304 249 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 304 249 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /s/workshop.jpg HTF/1.1" 304 249 "http://10.0.2.15/" Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /robots.txt HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /HTF/1.1" 200 6026 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /f3cfe280b5465a680191a58818101d64a40b9c742acbf8168724045649c6422aa58238a3f76dedd HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /admin/97782a38804a18f6c6c28f73d8d HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /ataccasf48f01c31c3ab4c8a3c1a01d HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /ataccasf48f01c31c3ab4c8a3c1a01d HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /ataccasf48f01c31c3ab4c8a3c1a01d HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /ataccasf48f01c31c3ab4c8a3c1a01d HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /category HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /blog HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /install HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /crackback HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /temp HTF/1.1" 404 432 "-" ferobustxt/2.10.0"
10.0.2.15 - [27/Mar/2024:10:15:37] - 40400 "GET /logs HTF/1.1" 404 432 "-" ferobust
```

[illegible]

Penjelasan:

- `grep -Eo "GET [^"]+"` mengekstrak semua permintaan GET dari log.
- `sort | uniq -c` menghitung frekuensi setiap permintaan.
- `sort -nr` mengurutkan hasil berdasarkan frekuensi, dari yang tertinggi ke terendah.

Analisis: Output ini menunjukkan jalur mana yang paling sering dicoba untuk diakses. Jika ada percobaan berulang ke jalur seperti /admin, ini bisa mengindikasikan upaya brute force untuk menemukan halaman login administrator.

6. Mengetahui Alat yang Digunakan untuk Serangan

Tujuan: Mengidentifikasi alat otomatis seperti curl atau wget yang digunakan untuk mengakses situs.

Perintah:

[illegible]

Penjelasan:

- `grep -iE` mencari user-agent yang mengindikasikan penggunaan alat seperti curl atau wget.

Analisis: Log ini menunjukkan alat yang digunakan untuk mencoba mengakses situs. Jika alat seperti curl atau wget digunakan untuk mengakses jalur sensitif, ini bisa menjadi indikasi

Kesimpulan

Melalui langkah-langkah di atas, kita telah berhasil mengidentifikasi beberapa aktivitas mencurigakan dan upaya serangan terhadap situs web berdasarkan file access.log. Dengan memanfaatkan berbagai perintah bash seperti grep, awk, dan sort, kita bisa mengekstrak informasi penting dari log, mendeteksi serangan, dan menemukan bendera tersembunyi. Write-up ini memberikan gambaran jelas mengenai proses analisis log web server dan pentingnya memonitor log untuk mendeteksi dan mencegah serangan keamanan siber.

Jika ada pertanyaan lebih lanjut atau perlu penjelasan tambahan mengenai langkah-langkah di atas, silakan bertanya!

ISO Audit Bypass

Jika file RAW yang kamu coba buka, misalnya `SELEKDA-PC-20240705-144848.raw`, tidak memiliki thumbnail (pesan dari tool seperti `dcrw`), artinya file tersebut mungkin tidak mengikuti format standar file gambar RAW dari kamera. File `.raw` bisa digunakan untuk berbagai jenis data biner, bukan hanya untuk gambar.

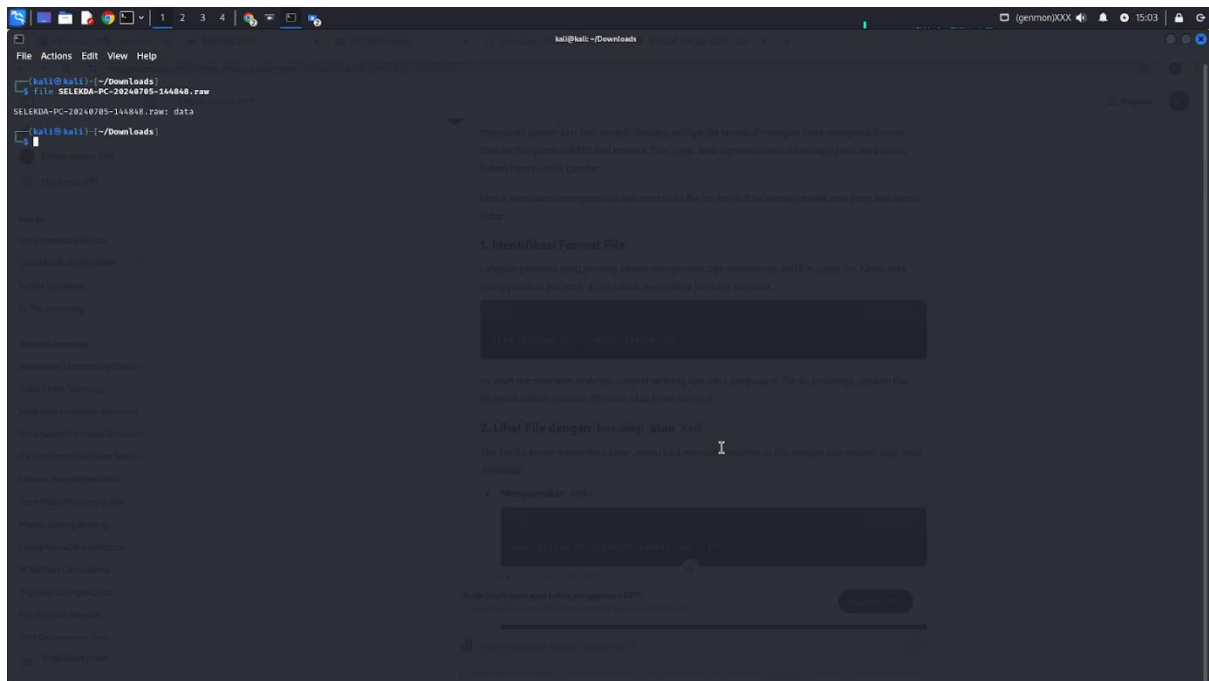
Untuk membantu menganalisis dan membuka file ini, berikut beberapa pendekatan yang bisa kamu coba:

1. Identifikasi Format File

Langkah pertama yang penting adalah mengetahui tipe sebenarnya dari file `.raw` ini. Kamu bisa menggunakan perintah `file` untuk memeriksa jenis file tersebut.

```
bash
```

```
file SELEKDA-PC-20240705-144848.raw
```



Ini akan memberikan deskripsi singkat tentang tipe data yang ada di file itu (misalnya, apakah file tersebut adalah gambar, file teks, atau biner lainnya).

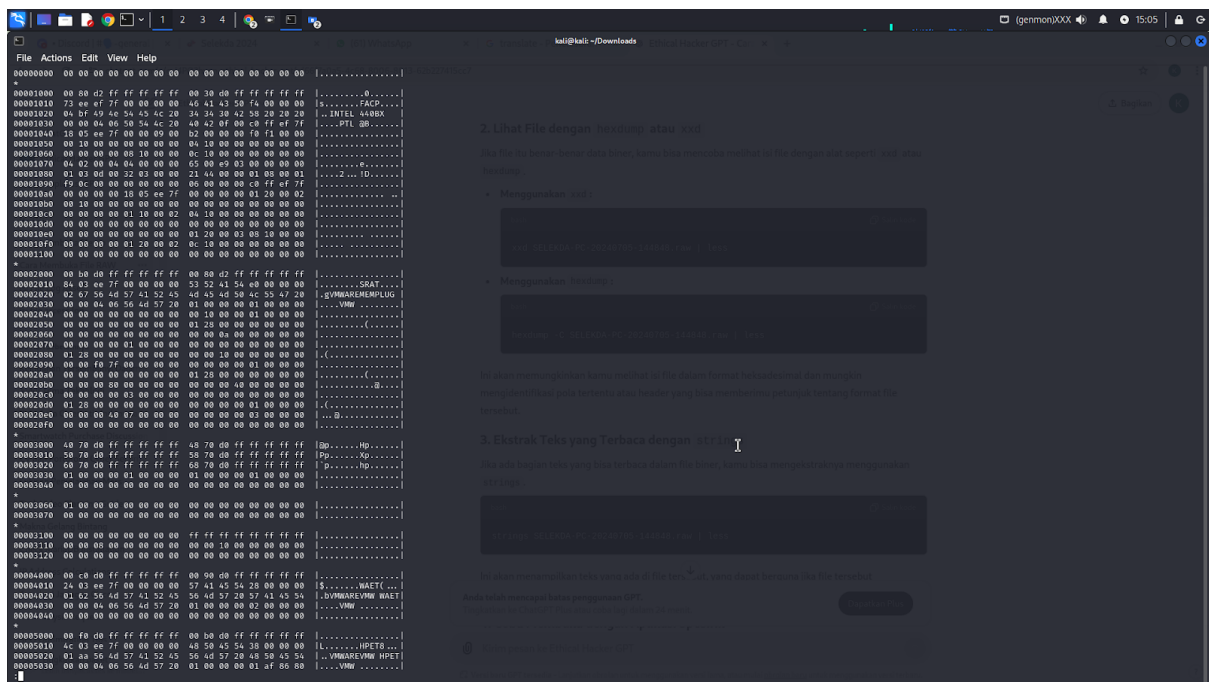
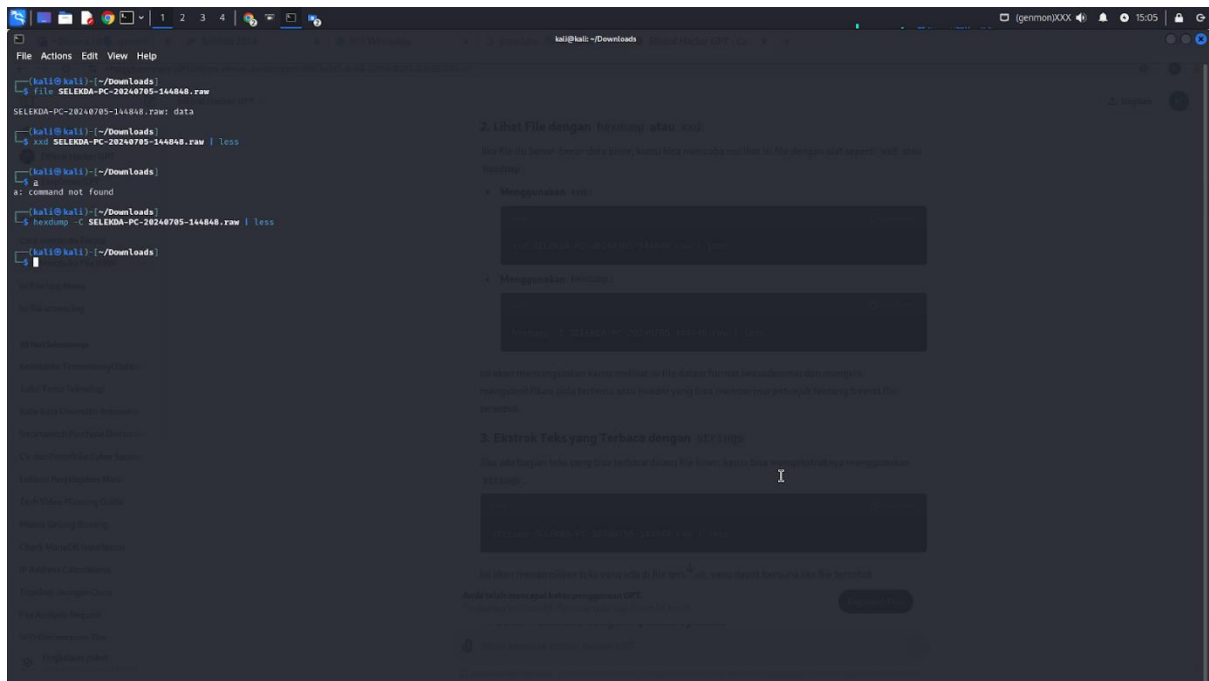
2. Lihat File dengan `hexdump` atau `xxd`

Jika file itu benar-benar data biner, kamu bisa mencoba melihat isi file dengan alat seperti `xxd` atau `hexdump`.

- Menggunakan `xxd`:

bash

`xxd SELEKDA-PC-20240705-144848.raw | less`



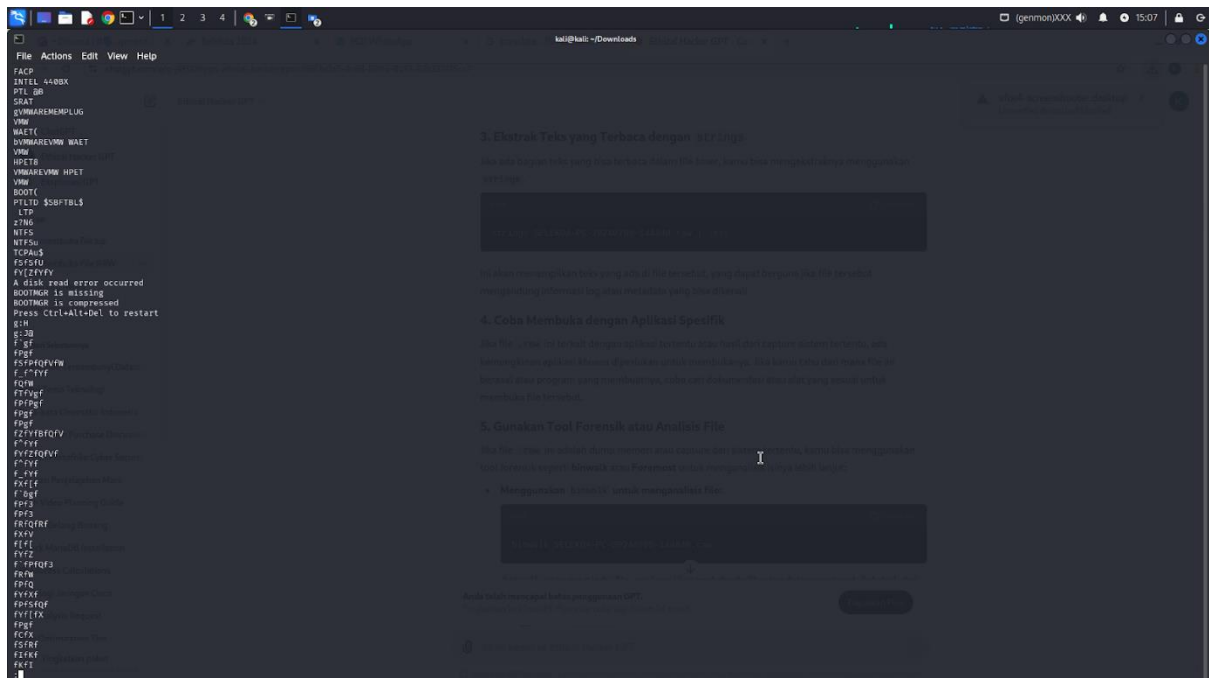
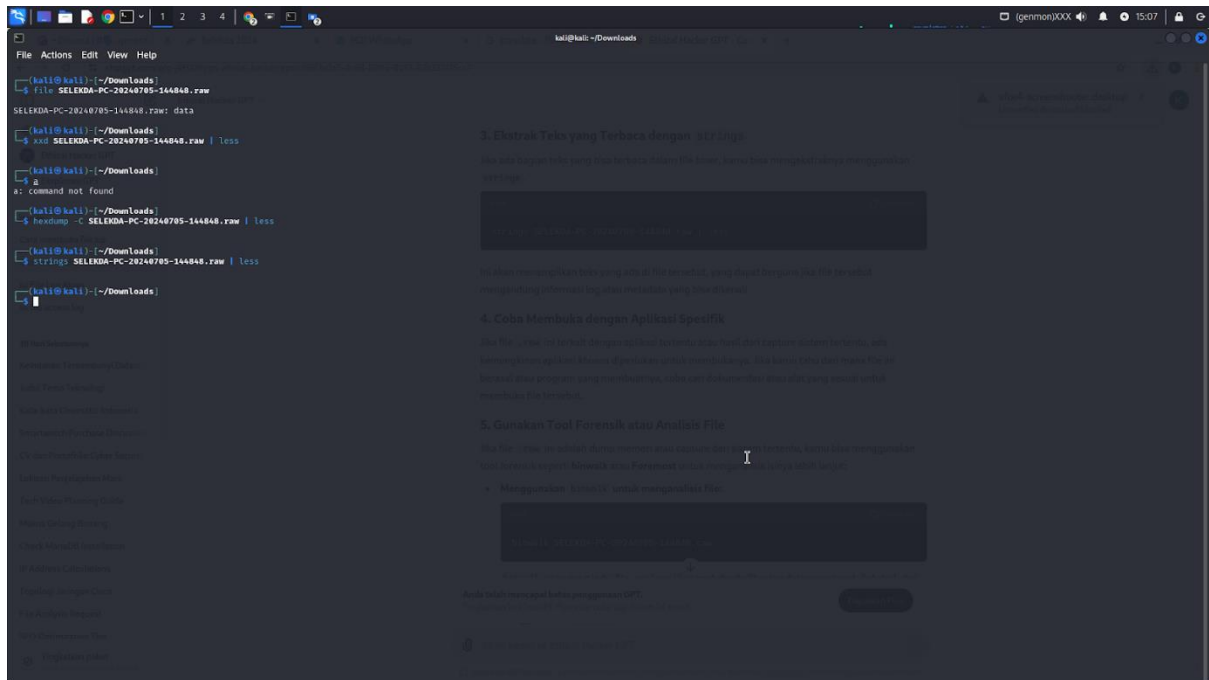
Ini akan memungkinkan kamu melihat isi file dalam format heksadesimal dan mungkin mengidentifikasi pola tertentu atau header yang bisa memberimu petunjuk tentang format file tersebut.

3. **Ekstrak Teks yang Terbaca dengan `strings`

Jika ada bagian teks yang bisa terbaca dalam file biner, kamu bisa mengekstraknya menggunakan `strings`.

bash

strings SELEKDA-PC-20240705-144848.raw | less



Ini akan menampilkan teks yang ada di file tersebut, yang dapat berguna jika file tersebut mengandung informasi log atau metadata yang bisa dikenali.

4. Gunakan Tool Forensik atau Analisis File

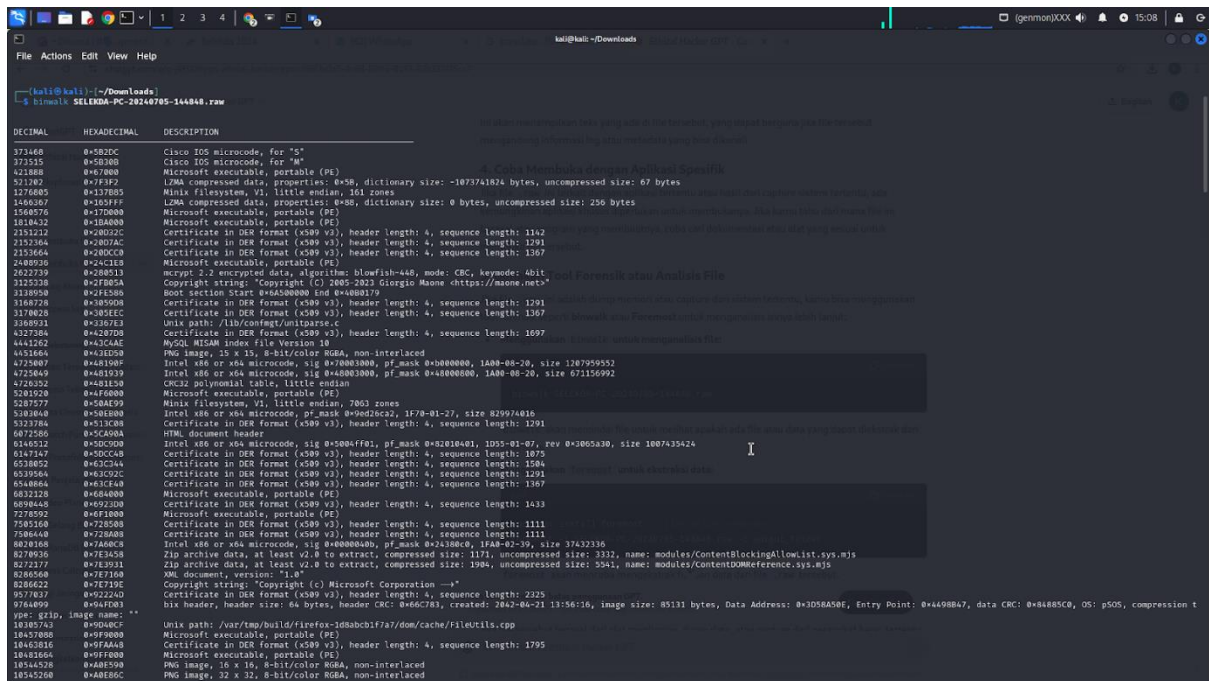
Jika file `.raw` ini adalah dump memori atau capture dari sistem tertentu, kamu bisa menggunakan tool forensik seperti binwalk atau Foremost untuk menganalisis isinya lebih lanjut:

- Menggunakan `binwalk` untuk menganalisis file:

```
bash
```

```
binwalk SELEKDA-PC-20240705-144848.raw
```

```
***
```



```
File Actions Edit View Help
[kali@kali:~/Downloads]
$ binwalk SELEKDA-PC-20240705-144848.raw

DECIMAL      HEXADECIMAL     DESCRIPTION
-----
373468      0x5820C      Cisco IOS microcode, for "5"
373535      0x58300      Cisco IOS microcode, for "M"
421888      0x67900      Microsoft executable, portable (PE)
521262      0x7f3f2      LMA compressed data, properties: 0x58, dictionary size: -1873741874 bytes, uncompressed size: 67 bytes
1276805      0x137805      Minix filesystem, V1, little endian, 161 zones
1466567      0x1603ff      LMA compressed data, properties: 0x58, dictionary size: 0 bytes, uncompressed size: 250 bytes
1560576      0x170000      Microsoft executable, portable (PE)
1618432      0x18A000      Microsoft executable, portable (PE)
2153212      0x2051c      Certificate in DER format (x509 v3), header length: 4, sequence length: 1162
2153264      0x2057AC      Certificate in DER format (x509 v3), header length: 4, sequence length: 1291
2153664      0x2060C0      Certificate in DER format (x509 v3), header length: 4, sequence length: 1367
2489936      0x24C1E8      Microsoft executable, portable (PE)
2622739      0x268513      mcrpypt 2.2 encrypted data, algorithm: blowfish-446, mode: CBC, keymode: 4bit
3125238      0x2f503A      Copyright string: "Copyright (C) 2005-2021 Giorgio Maone <https://maone.net>"
3138950      0x2fE586      Boot section Start 0x4A500000 End 0x4680179
3168728      0x305998      Certificate in DER format (x509 v3), header length: 4, sequence length: 1201
3178028      0x3055C5      Certificate in DER format (x509 v3), header length: 4, sequence length: 1367
3169521      0x3167E3      Unix path: /lib/confmtg/unitsparse.c
4227264      0x403700      Certificate in DER format (x509 v3), header length: 4, sequence length: 1697
4441262      0x42C4AE      MySQL MISAM index file Version 18
4651664      0x43ED50      PNG image, 15 x 15, 8-bit/color RGBA, non-interlaced
4729407      0x46310F      Intel x86 or x64 microcode, sig 0x70003000, pf_mask 0xb0000000, 1A00-08-20, size 1207959552
4725849      0x461939      Intel x86 or x64 microcode, sig 0x40003000, pf_mask 0x40000000, 1A00-08-20, size 671156992
4726352      0x461E30      CRC32 polynomial table, little endian
5283528      0x50E000      Microsoft executable, portable (PE)
5287577      0x50AE99      Minix filesystem, V1, little endian, 7803 zones
5283840      0x50E000      Certificate in DER format (x509 v3), header length: 4, sequence length: 1291
5333784      0x513C08      Certificate in DER format (x509 v3), header length: 4, sequence length: 1291
6872585      0x5CA90A      HTML document header
6365212      0x5D0C00      Intel x86 or x64 microcode, sig 0x2004ff01, pf_mask 0x62010401, 1D03-01-07, rev 0x3005A3D, size 1007439424
6147147      0x5DCCAB      Certificate in DER format (x509 v3), header length: 4, sequence length: 1875
6538652      0x63C344      Certificate in DER format (x509 v3), header length: 4, sequence length: 1584
6539564      0x63C52C      Certificate in DER format (x509 v3), header length: 4, sequence length: 1291
6548864      0x63CE40      Certificate in DER format (x509 v3), header length: 4, sequence length: 1367
6832228      0x66E400      Microsoft executable, portable (PE)
6899448      0x697200      Certificate in DER format (x509 v3), header length: 4, sequence length: 1433
7275992      0x6FF100      Microsoft executable, portable (PE)
7285168      0x713050      Certificate in DER format (x509 v3), header length: 4, sequence length: 1111
7286448      0x7130A8      Certificate in DER format (x509 v3), header length: 4, sequence length: 1111
8820168      0x7A60C8      Intel x86 or x64 microcode, sig 0x00000000, pf_mask 0x24308C, 1F08-07-39, size 37432736
8278036      0x7E2658      Zip archive data, at least v2.0 to extract, compressed size: 1171, uncompressed size: 3312, name: modules/ContentBlockingAllowlist.sys.mjs
8272277      0x7E3931      Zip archive data, at least v2.0 to extract, compressed size: 1904, uncompressed size: 5541, name: modules/ContentDOMReference.sys.mjs
8286568      0x7E7160      XML document, version: "1.0"
8286622      0x7E719C      Copyright string: "Copyright (c) Microsoft Corporation ->"
9577837      0x92224D      Certificate in DER format (x509 v3), header length: 4, sequence length: 2325
9784999      0x96AF01      bix header, header size: 64 bytes, header CRC: 0x66C783, created: 2042-04-21 13:56:10, image size: 85131 bytes, Data Address: 0x3D58A50E, Entry Point: 0x4498B47, data CRC: 0x84885C0, OS: pSOS, compression t
ybe: gzip, image name: ""
10385743      0x9904CF      Unix path: /var/tmp/build/1p6fox-1d8abcb17a7/dm/cache/FileUtils.cpp
10457805      0x9F7000      Microsoft executable, portable (PE)
10463816      0x9FAA48      Certificate in DER format (x509 v3), header length: 4, sequence length: 1795
10482664      0x99F400      Microsoft executable, portable (PE)
10544576      0xA4E500      PNG image, 16 x 16, 8-bit/color RGBA, non-interlaced
10545268      0xA4E85C      PNG image, 32 x 32, 8-bit/color RGBA, non-interlaced
```

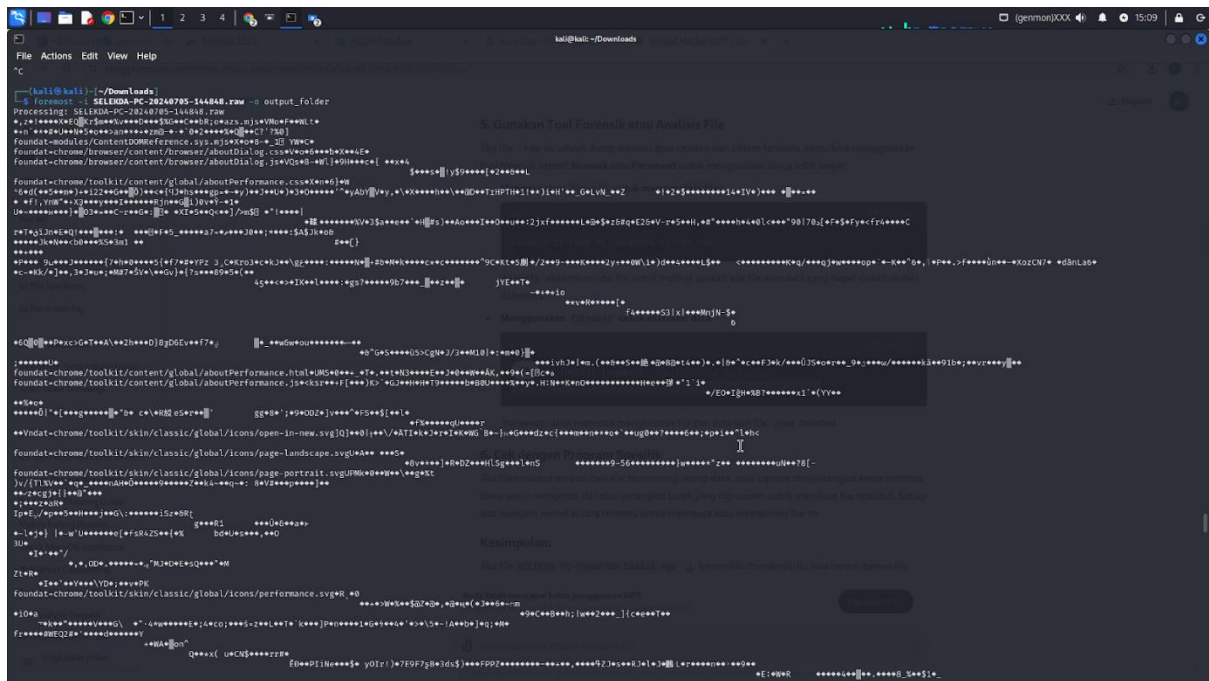
`binwalk` akan memindai file untuk melihat apakah ada file atau data yang dapat diekstrak dari dalamnya.

- Menggunakan `foremost` untuk ekstraksi data:

```
``bash
```

```
foremost -i SELEKDA-PC-20240705-144848.raw -o output_folder
```

```
***
```



`foremost` akan mencoba mengekstrak file dan data dari file `.raw` tersebut.

5. Cek dengan Program Spesifik

Jika file tersebut berasal dari alat monitoring, dump data, atau capture dari perangkat keras tertentu, kamu perlu mengecek alat atau perangkat lunak yang digunakan untuk membuat file tersebut. Setiap alat mungkin memiliki cara tertentu untuk membaca atau memproses file ini.

Kesimpulan:

Jika file `SELEKDA-PC-20240705-144848.raw` tidak memiliki thumbnail, itu bisa berarti bahwa file tersebut tidak sesuai dengan format gambar standar atau merupakan jenis data yang berbeda. Dengan langkah-langkah di atas, kamu bisa menganalisis lebih dalam tentang format dan isi file ini.

Sniff Sniff Whoopsie

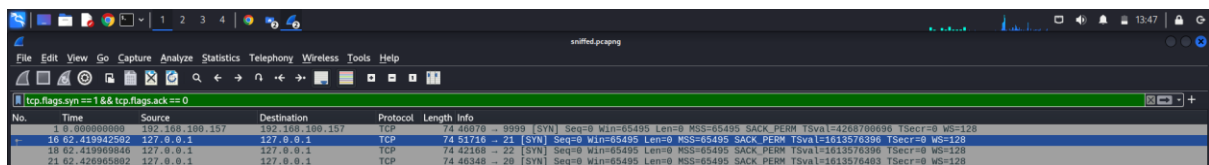
Penyelesaian :

Berikut adalah langkah-langkah yang lebih rinci dan mudah diikuti untuk menganalisis file **PCAP** menggunakan **Wireshark** :

1. Aktivitas pemindaian port terdeteksi dalam sistem. Apakah benar atau salah positif? Silakan ambil tangkapan layar segala jenis bukti termasuk semua nomor port yang terlibat dan laporkan port yang terbuka.

Gunakan filter berikut ini pada wireshark

`tcp.flags.syn == 1 && tcp.flags.ack == 0`



Filter ini akan menampilkan paket **SYN** yang dikirimkan tanpa balasan **ACK**, yang seringkali merupakan indikasi pemindaian port.

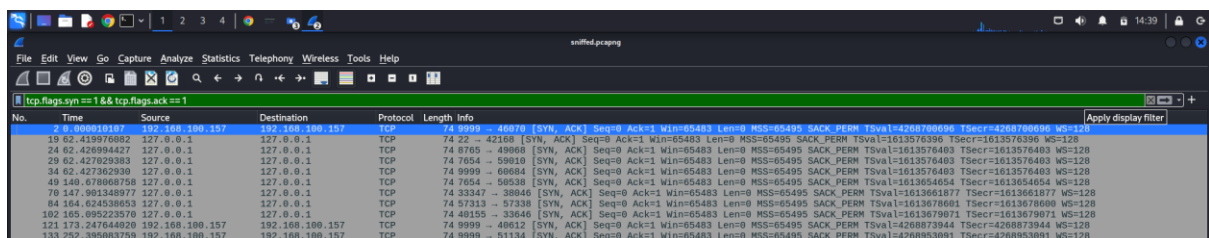
- Aktivitas pemindaian port terdeteksi dalam sistem. Apakah benar atau salah?

Benar, karena ada 3 port yang terbuka yaitu port 20, port 21, port 22

2. Eksploitasi CVE PHP terdeteksi oleh SIEM kami. Apakah itu benar atau salah positif? Silakan ambil tangkapan layar segala jenis bukti.

Gunakan filter berikut pada wire shark

`tcp.flags.syn == 1 && tcp.flags.ack == 1`

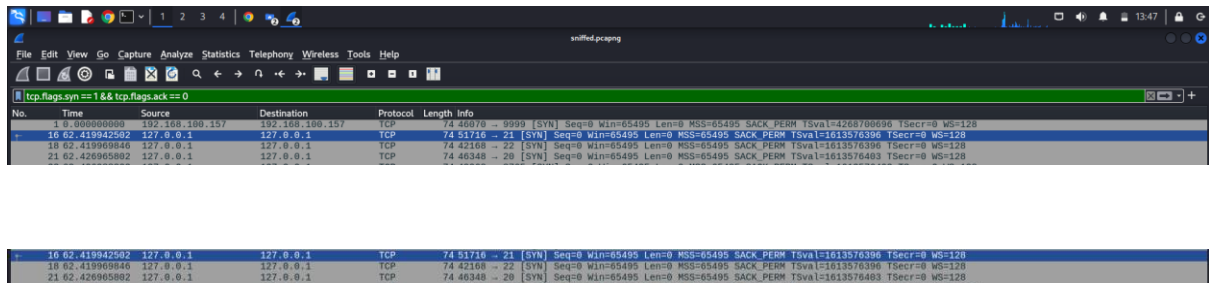


Filter ini pada Wireshark digunakan untuk menampilkan paket TCP yang memiliki flag SYN dan flag ACK diatur ke 1 (aktif). Paket dengan kombinasi SYN dan ACK seperti ini umumnya muncul sebagai bagian dari tahap kedua dari three-way handshake dalam protokol TCP.

Eksploitasi CVE PHP terdeteksi oleh SIEM kami. Apakah itu benar atau salah positif? Benar karena ada port 22 yang berstatus SYN, ACK yang berarti port tersebut terbuka.

3. Kami tidak sengaja membuka beberapa port karena kesalahan konfigurasi Firewall. Dapatkah Anda memberi tahu saya port mana yang bertanggung jawab untuk membuka koneksi untuk transfer file?

Jawaban :



The image shows two screenshots of a Wireshark network capture. The top screenshot shows a filter 'tcp.flags.syn == 1 && tcp.flags.ack == 0' applied, displaying three TCP SYN packets from 192.168.100.157 to 127.0.0.1 on ports 20, 21, and 22. The bottom screenshot shows the same capture with the filter 'http.request' applied, displaying three HTTP GET requests from 192.168.100.157 to 127.0.0.1 on ports 20, 21, and 22, corresponding to the ports mentioned in the text.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.100.157	127.0.0.1	TCP	74	48070 -> 9999 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=4268700036 TSecr=0 WS=128
10	62.419942502	127.0.0.1	127.0.0.1	TCP	74	51710 -> 21 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1613576396 TSecr=0 WS=128
18	62.419969840	127.0.0.1	127.0.0.1	TCP	74	42168 -> 22 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1613576396 TSecr=0 WS=128
21	62.426965982	127.0.0.1	127.0.0.1	TCP	74	48348 -> 20 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1613576403 TSecr=0 WS=128

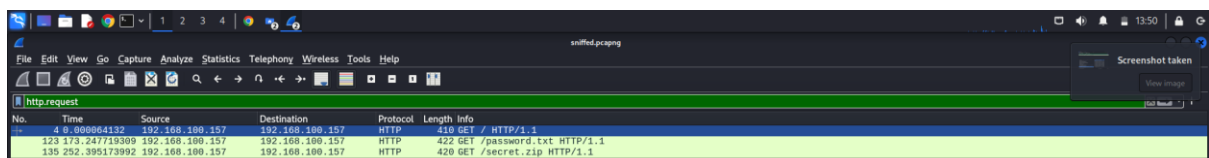
No.	Time	Source	Destination	Protocol	Length	Info
4	0.000004132	192.168.100.157	127.0.0.1	HTTP	419	GET / HTTP/1.1
123	173.247719309	192.168.100.157	127.0.0.1	HTTP	422	GET /password.txt HTTP/1.1
136	292.3895173992	192.168.100.157	127.0.0.1	HTTP	429	GET /secret.zip HTTP/1.1

Port yang bertanggung jawab membuka koneksi untuk transfer file adalah port 20, port 21, dan port 22 karena hanya port tersebut yang aktif.

4. Apa yang diunduh oleh penyerang dari port transfer file terkait?

Jawaban :

Filter http.request pada wireshark digunakan untuk menampilkan semua paket yang merupakan permintaan dari HTTP.

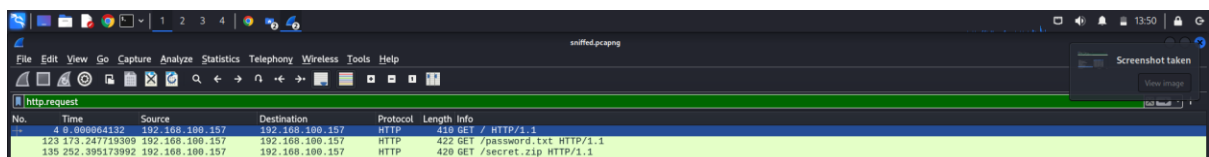


The image shows a screenshot of a Wireshark network capture with the filter 'http.request' applied. It displays three HTTP GET requests from 192.168.100.157 to 127.0.0.1 on ports 20, 21, and 22, corresponding to the ports mentioned in the text.

No.	Time	Source	Destination	Protocol	Length	Info
4	0.000004132	192.168.100.157	127.0.0.1	HTTP	419	GET / HTTP/1.1
123	173.247719309	192.168.100.157	127.0.0.1	HTTP	422	GET /password.txt HTTP/1.1
136	292.3895173992	192.168.100.157	127.0.0.1	HTTP	429	GET /secret.zip HTTP/1.1

Yang diunduh oleh penyerang dari port file terkait adalah sebuah file password.txt dan file secret.zip

5. Ada kemungkinan penyerang mengunduh berkas internal yang seharusnya tidak untuk publik? Pengembang kami cukup ceroboh sehingga dia menempatkan semacam petunjuk rahasia di situs web.



The image shows a screenshot of a Wireshark network capture with the filter 'http.request' applied. It displays three HTTP GET requests from 192.168.100.157 to 127.0.0.1 on ports 20, 21, and 22, corresponding to the ports mentioned in the text.

No.	Time	Source	Destination	Protocol	Length	Info
4	0.000004132	192.168.100.157	127.0.0.1	HTTP	419	GET / HTTP/1.1
123	173.247719309	192.168.100.157	127.0.0.1	HTTP	422	GET /password.txt HTTP/1.1
136	292.3895173992	192.168.100.157	127.0.0.1	HTTP	429	GET /secret.zip HTTP/1.1

Berkas internal yang seharusnya tidak untuk dipublik adalah file password.txt dan secret.zip karena file tersebut memiliki informasi yang sensitiv.

6. Jawablah pertanyaan ini jika pertanyaan sebelumnya benar. Apa isi file yang dilindungi HANYA JIKA penyerang mencuri/mengunduh file tersebut? Apakah kata sandinya cukup kuat?

Jawaban :

Isi file: Jika file yang diunduh berisi data sensitif seperti password.txt dan secret.zip, ini dapat berpotensi berbahaya.

Kekuatan kata sandi: Jika kata sandi yang melindungi file lemah (misalnya, kata sandi pendek atau umum), penyerang bisa memecahkannya dengan cepat. Jika hash yang kuat digunakan, file akan lebih sulit dipecahkan.