Category:

Cryptography

Name:

Can you extract hidden files?

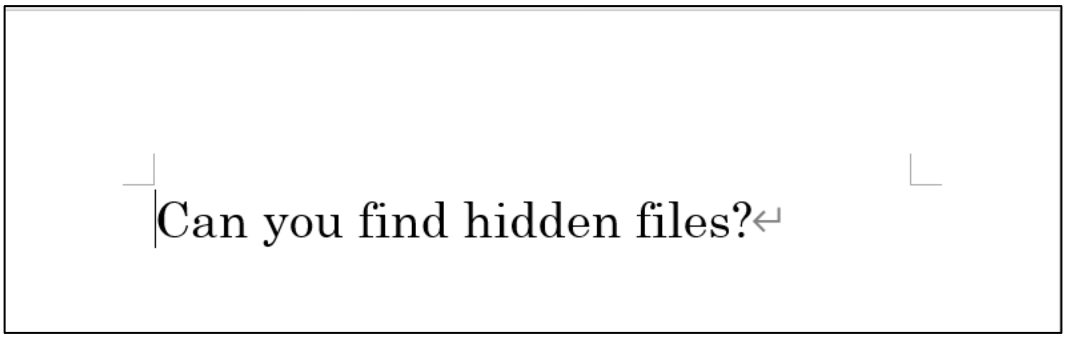
Message:

The flag for this challenge is packaged in a WORD document.

You need to extract files and decrypt these files.

Instructions:

1. When you double-click “example.docx”, you will see a warning that the file is corrupted, but if you select the "Open and Repair" option, you will see the following message:



Can you find hidden files?↵

2. The docx file is in OpenXML format, and the contents can be viewed with a decompression tool by changing the extension to zip.
   This file has a CTF folder, which is not present in the regular docx format.
3. Open the file “how_to_decrypt_cipher.txt”. You will find the following explanation.
   “This file is all bits inverted from the original.”
4. This means that all bits must be inverted to decrypt the Cipher. Below is an example of PowerShell script that flips all the bits of “Cipher”.

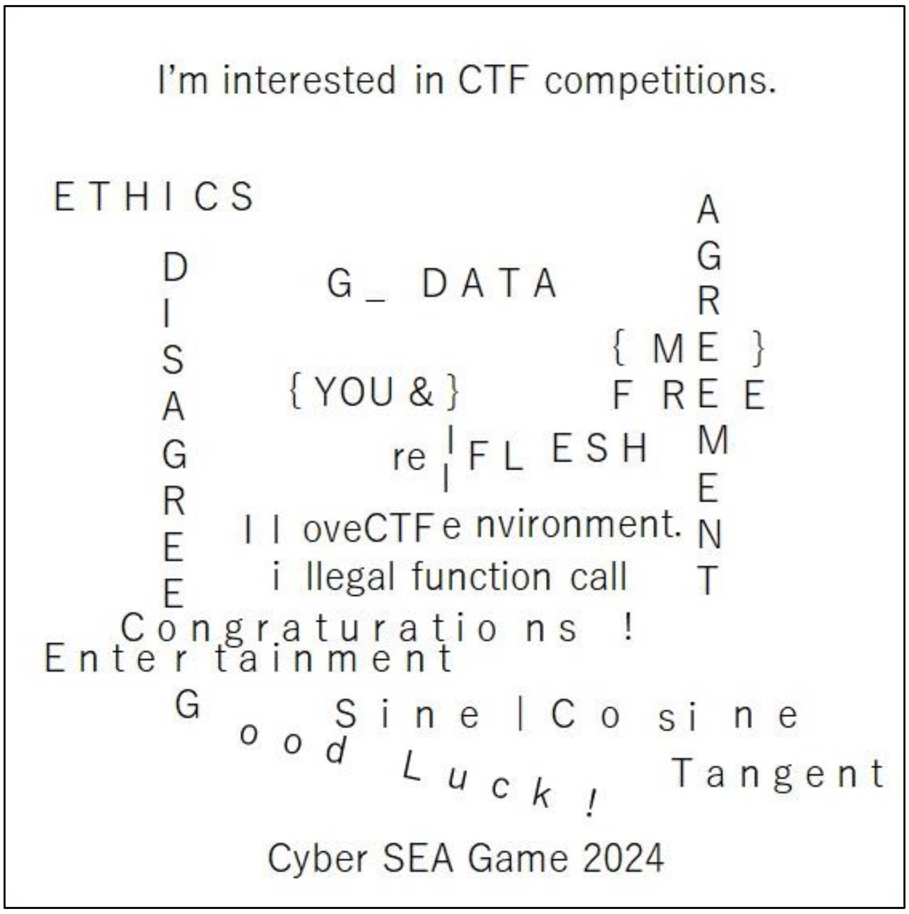
```
# Specify the file path
$file_path = "Full path of Cipher"
# Flip all bits
$file_bytes = [System.IO.File]::ReadAllBytes($file_path)
for ($i = 0; $i -lt $file_bytes.Length; $i++) {
    $file_bytes[$i] = [System.Byte]::MaxValue - $file_bytes[$i]
}
# Overwriting
[System.IO.File]::WriteAllBytes($file_path, $file_bytes)
```

5. You can confirm the actual contents of Cipher by file command as below:

```
$ file Cipher
Cipher: JPEG image data, JFIF standard 1.01, resolution (DPI), density 96x96, segment length
16, baseline, precision 8, 576x576, components 3
```

As you can see, it is a JPEG picture image file.

6. You can view the file “Cipher” by any Image editors / viewers.



7. Next, let's check the contents of XOR_it.txt.

```
In order to decrypt filter, you need to XOR with it.
```

8. It seems that the file can be decrypted by XORing the "filter" file and XOR_it.txt. Below is an example PowerShell script that XORs a "filter" file and XOR_it.txt.

```
# Specify file paths
$original_file_path = "Full path of filter"
$xor_file_path = "Full path of XOR_it.txt"

# Specify the decrypted file path
$decrypted_file_path = "Full path of the decrypted file"
```

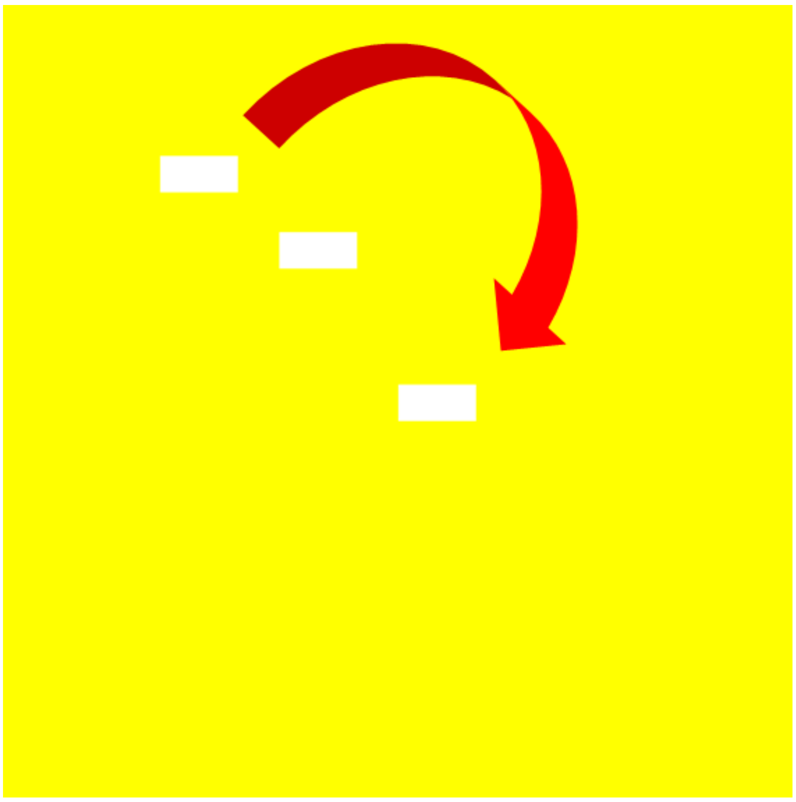
```
# Read each file's bytes
$file_A_bytes = [System.IO.File]::ReadAllBytes($original_file_path)
$file_XOR_bytes = [System.IO.File]::ReadAllBytes($xor_file_path)

# XOR encryption
for ($i = 0; $i -lt $file_A_bytes.Length; $i++) {
    $file_A_bytes[$i] = $file_A_bytes[$i] -bxor $file_XOR_bytes[$i %
$file_XOR_bytes.Length]
}

# Output the decrypted file
[System.IO.File]::WriteAllBytes($decrypted_file_path, $file_A_bytes)
```

9. You can confirm the actual contents of decrypted "filter" by file command as below:

```
$ file filter
filter: PNG image data, 567 x 569, 8-bit/color RGBA, non-interlaced
```

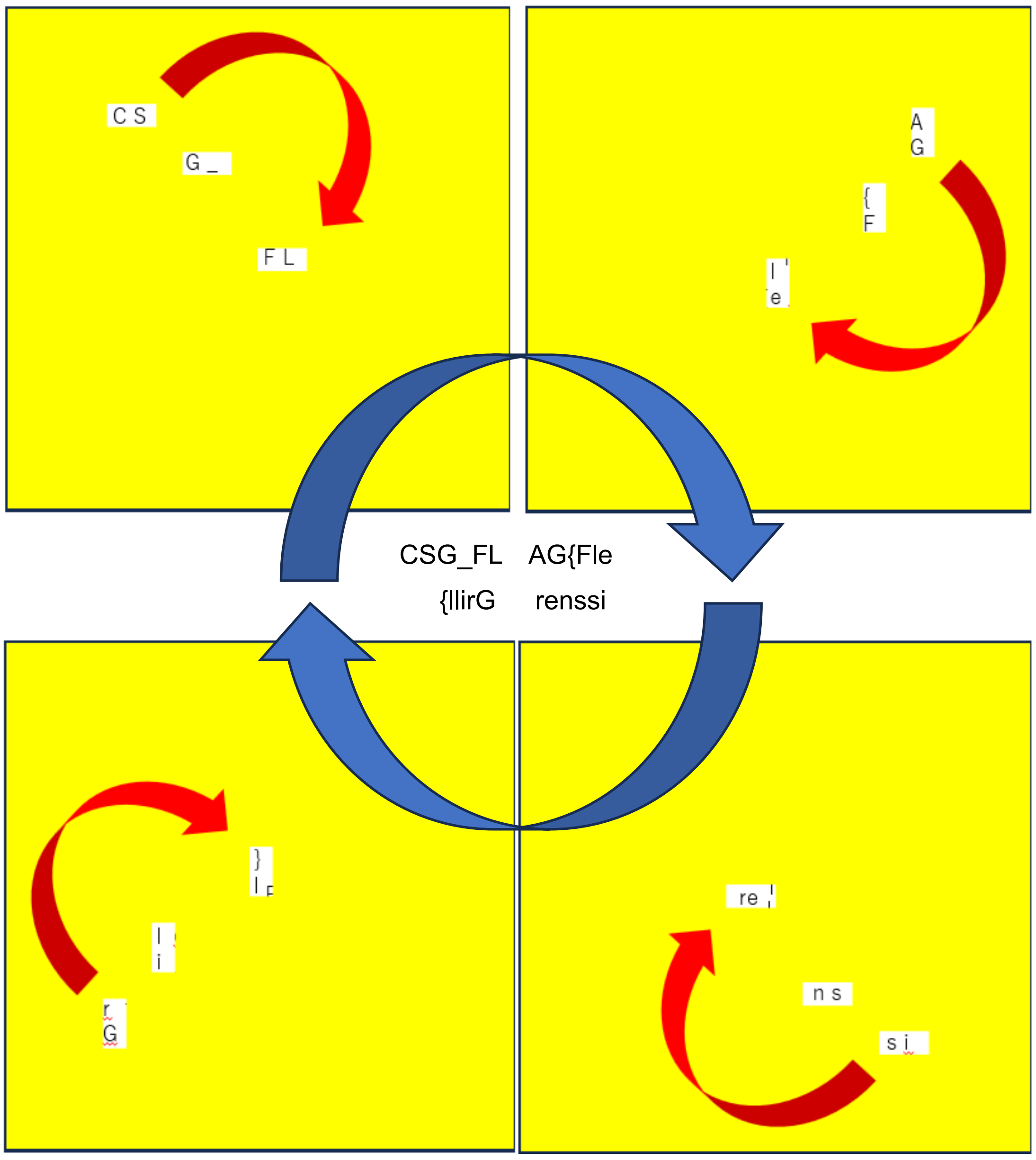
As you can see, it is a PNG picture image file.

10. You can view the decrypted "filter" file by any Image editors / viewers.



As the astute observer may have noticed, this is a pierced sheet for traditional turning grille ciphers (a.k.a : Fleissner grille).

11. Let’s overlay the pierced sheet on top of Cipher.jpg with rotating it.



Flag is:

CSG_FLAG{FleissnerGrill}

References:

Wikipedia : Grille (cryptography)

https://en.wikipedia.org/wiki/Grille_(cryptography)

]