README.md 2021/1/13

HW0C

Gift 🕰

首先會拿到一個 gift.gz 的檔案·ungzip 後會得到一個 binary·binary 做的事情是比對輸入的字串與某字串是不是一樣的:

```
strcpy(
    s2,

"JZC33MJPDC48UXXJ94BBQOR0JJR4AO0W02PHZ4VZRJAEXL30UI02FQ4GSQIDGBFT70VESKNAAUEJW4RR9
EQOCJ9PKT7W9FBMJDVK6X9MT7K1HY30MSA4"

"H3Y9FTV007Z6FQ5I1J8R6KSCMWKFSDGCMWARIJTLPLR08KUYQW2F46ZV6YWIVFNCZDQRCTAM5JVGQMEU2
LFPS5DUD0Y4130XB50V91PWHCI00AD1RHTR"
        "673DPX36TA2UWA48FD34Y2W6");
        __isoc99_scanf("%256s", &s1, v7);
if ( !strcmp(&s1, s2) )
{
        ...
        write(1, byte_201020, &unk_3347DB);
}
```

如果是的話,就會在產生一個 gzip,而接下來 ungzip 那個檔案後,又會重複一樣的事情。因此我們要做的就是從 binary 中 extrace 那串字串,並且當作 input 餵給他,反覆數次,最後一個 binary 會長得不太一樣:

輸入後即可得到 FLAG。 FLAG{what_a_boaring_challnge_but_you_did_it_yeah_yeah}

JustOnLinux

首先會拿到 flag 與一個 binary·flag 看起來被加過密·而逆了一下 binary 後發現·他會把原本的字串·每 3 個 bytes 為一組共 24 bits·再以 6 個 bits 為一字元組成長度為 4 的加密字串·而加密的過程為一個 table·對應的值會有對應的字元.....這根本就是 base64 吧 XD·只是 table 長得不一樣而已。

於是將 bits 重新組回來,再用原本的 ascii 來看,就會有 flag 了

README.md 2021/1/13

```
table = "vwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!\"#$%&'()*+,-./:;<=>?@[\]^_`{|}~o"
flag = open('flag', 'r').read()

for i in range(0, len(flag), 4):
    part = flag[i:i+4]

sum = 0
    for j in range(4):
        part_value = table.find(part[j])
        sum += (part_value << (6*(3-j)))

print(sum.to_bytes(3, 'big').decode(errors='ignore'), end='')</pre>
```

FLAG{7h1s-i5-ac7ua11y-a-b4s364enc0d3-alg0r1thm} (真的是 base64 XD)