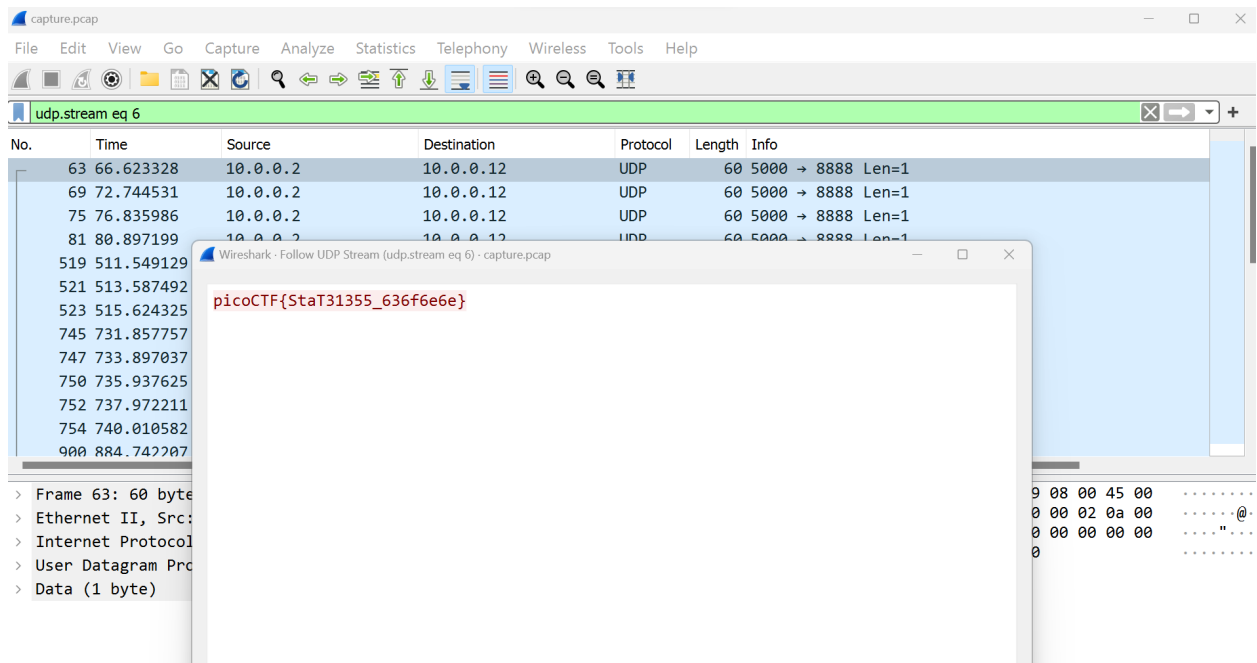
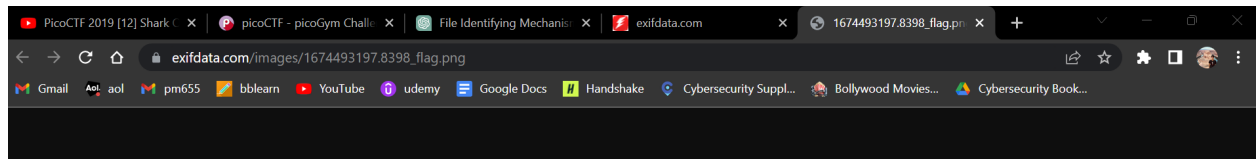


Shark on wire 1(Forensics)



Extensions (Forensics)



picoCTF{now_you_know_about_extensions}

Redaction gone wrong(forensics)

```
(prasanth@prasanth)~$ sudo apt install pdftotext
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package pdftotext

(prasanth@prasanth)~$ pdftotext Financial_Report_for_ABC_Labs.pdf

(prasanth@prasanth)~$ ls
Financial_Report_for_ABC_Labs.pdf  Financial_Report_for_ABC_Labs.txt  Graffiti

(prasanth@prasanth)~$ cat Financial_Report_for_ABC_Labs.txt
Financial Report for ABC Labs, Kigali, Rwanda for the year 2021.
Breakdown - Just painted over in MS word.

Cost Benefit Analysis
Credit Debit
This is not the flag, keep looking
Expenses from the
picoCTF{C4n_Y0u_S33_m3_fully}
Redacted document.
```

Enhance! (Forensics)

```
(prasanth@prasanth)~[~]
$ strings drawing.flag.svg | less

(prasanth@prasanth)~[~]
$ strings drawing.flag.svg | grep "</tspan"
  id="tspan3748">p </tspan><tspan
  id="tspan3754">i </tspan><tspan
  id="tspan3756">c </tspan><tspan
  id="tspan3758">o </tspan><tspan
  id="tspan3760">C </tspan><tspan
  id="tspan3762">T </tspan><tspan
  id="tspan3764">F { 3 n h 4 n </tspan><tspan
  id="tspan3752">c 3 d _ d 0 a 7 5 7 b f }</tspan></text>

(prasanth@prasanth)~[~]
$ strings drawing.flag.svg | grep "</tspan" | cut -d ">" -f2 | cut -d "<" -f1 | tr -d "\n"
p i c o C T F { 3 n h 4 n c 3 d _ d 0 a 7 5 7 b f }

(prasanth@prasanth)~[~]
$ strings drawing.flag.svg | grep "</tspan" | cut -d ">" -f2 | cut -d "<" -f1 | tr -d "\n" | tr -d " "
picoCTF{3nh4nc3d_d0a757bf}
```

Lookey Here(Forensics)

```
(prasanth@prasanth)~[~]
$ cat anthem.flag.txt | grep pico
  we think that the men of picoCTF{gr3p_15_@w3s0m3_2116b979}

(prasanth@prasanth)~[~]
$
```

strings it (General skills)

```
prasanth@prasanth:~$ wget https://jupiter.challenges.picoctf.org/static/94d00153b0057d37da225ee79a846c62/strings
--2023-01-23 13:58:13-- https://jupiter.challenges.picoctf.org/static/94d00153b0057d37da225ee79a846c62/strings
Resolving jupiter.challenges.picoctf.org (jupiter.challenges.picoctf.org)... 3.131.60.8
Connecting to jupiter.challenges.picoctf.org (jupiter.challenges.picoctf.org)|3.131.60.8|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 776032 (758K) [application/octet-stream]
Saving to: 'strings.4'

strings.4                                     100%[=====>] 757.84K  1.66MB/s   in 0.4s

2023-01-23 13:58:14 (1.66 MB/s) - 'strings.4' saved [776032/776032]

prasanth@prasanth:~$ strings strings.4 | grep "picoCTF{.*}" --color=none
picoCTF{5tRIng5_1T_d66c7bb7}
prasanth@prasanth:~$
```

St3g0 (steganography) (Forensics)

Install zsteg using sudo gem install zsteg

```
prasanth@prasanth:~$ zsteg pico.flag.png
b1,r,lsb,xy .. text: "~_B>VG?G@"
b1,rgb,lsb,xy .. text: "picoCTF{7h3r3_15_n0_5p00n_a9a181eb}$t3g0"
b1,abgr,lsb,xy .. text: "E2A5q4E%uSA"
b2,b,lsb,xy .. text: "AAPAAQTAAA"
b2,b,msb,xy .. text: "HWUUUUUU"
b2,a,lsb,xy .. file: Matlab v4 mat-file (little endian) >\004<\305P, numeric, rows 0, columns 0
b2,a,msb,xy .. file: Matlab v4 mat-file (little endian) | <\243, numeric, rows 0, columns 0
b3,r,lsb,xy .. file: gfbboot compiled html help file
b4,r,lsb,xy .. file: Targa image data (16-273) 65536 x 4097 x 1 +4352 +4369 - 1-bit alpha - right "\021\020\001\001\021\021\001\001\021\021\001\021\021\020\020\001\020\001"
b4,g,lsb,xy .. file: 0420 Alliant virtual executable not stripped
b4,b,lsb,xy .. file: Targa image data - Map 272 x 17 x 16 +257 +272 - 1-bit alpha "\020\001\021\001\021\020\020\001\020\001\020\001"
b4,bgr,lsb,xy .. file: Targa image data - Map 273 x 272 x 16 +1 +4113 - 1-bit alpha "\020\001\001\001"
b4,rgba,lsb,xy .. file: Novell LANalyzer capture file
b4,rgba,msb,xy .. file: Applesoft BASIC program data, first line number 8
b4,abgr,lsb,xy .. file: Novell LANalyzer capture file
prasanth@prasanth:~$
```

WebNet0 (Forensics)

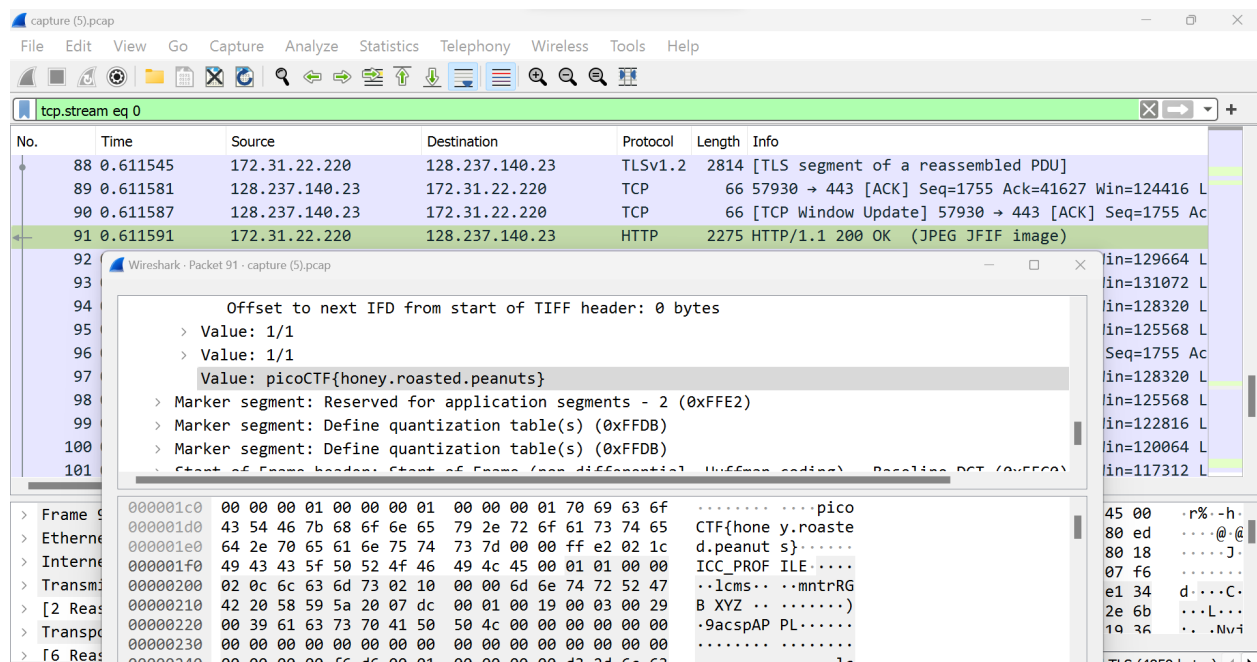
The image shows a Wireshark packet capture of an HTTP transaction. The packet list shows a GET request (packet 31) and a 200 OK response (packet 32). The packet details pane for packet 32 shows the response structure, including headers and a body containing the Pico-Flag.

No.	Time	Source	Destination	Protocol	Length	Info
18	0.092423	172.31.22.220	128.237.140.23	TCP	74	443 → 57581 [SYN, ACK] Seq=0 Ack=1 Win=26847 Len=
22	0.122048	128.237.140.23	172.31.22.220	TCP	66	57581 → 443 [ACK] Seq=1 Ack=1 Win=131904 Len=0 TS
23	0.122203	128.237.140.23	172.31.22.220	TLSv1.2	583	Client Hello
24	0.122220	172.31.22.220	128.237.140.23	TCP	66	443 → 57581 [ACK] Seq=1 Ack=518 Win=28032 Len=0 T
25	0.122552	172.31.22.220	128.237.140.23	TLSv1.2	1073	Server Hello, Certificate, Server Hello Done
27	0.151669	128.237.140.23	172.31.22.220	TCP	66	57581 → 443 [ACK] Seq=518 Ack=1008 Win=130880 Len
28	0.152210	128.237.140.23	172.31.22.220	TLSv1.2	384	Client Key Exchange, Change Cipher Spec, Finished
29	0.153206	172.31.22.220	128.237.140.23	TLSv1.2	324	New Session Ticket, Change Cipher Spec, Finished
30	0.183385	128.237.140.23	172.31.22.220	TCP	66	57581 → 443 [ACK] Seq=836 Ack=1266 Win=130752 Len
31	0.187804	128.237.140.23	172.31.22.220	HTTP	506	GET / HTTP/1.1
32	0.188303	172.31.22.220	128.237.140.23	HTTP	1299	HTTP/1.1 200 OK (text/html)

Packet 32 details:

```
Vary: Accept-Encoding\r\n
Content-Encoding: gzip\r\n
<Content-Encoding: gzip\r\n>
Pico-Flag: picoCTF{nongshim.shrimp.crackers}\r\n
Content-Length: 821\r\n
<Content-Length: 821\r\n>
Keep-Alive: timeout=5, max=100\r\n
Connection: Keep-Alive\r\n
```

WebNet1 (Forensics)



Sleuthkit Intro (Forensics)

Steps

1. Download the file and unzip it.
2. Connect to the server using “nc saturn.picoctf.net 52279”
3. Use the command `mm1s disk.img` to get the length of the disk image.
4. Enter the value 202752 as the answer to the server and you get the flag.

```
prasanth@prasanth:~$ mm1s disk.img
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

    Slot      Start      End      Length      Description
000:  Meta      0000000000  0000000000  0000000001  Primary Table (#0)
001:  -----      0000000000  0000002047  0000002048  Unallocated
002:  000:000      0000002048  0000204799  0000202752  Linux (0x83)
```

```
prasanth@prasanth:~$ nc saturn.picoctf.net 52279
What is the size of the Linux partition in the given disk image?
Length in sectors: 202752
202752
Great work!
picoCTF{mm15_f7w!}
```

Disk, disk, sleuth!

Steps:

1. Download the file using wget and unzip it using the command gunzip
2. Use strings and grep to find the flag.

```
prasanth@prasanth:~$ strings dds1-alpine.flag.img | grep "picoCTF{.*?}"
prasanth@prasanth:~$ strings dds1-alpine.flag.img | grep "picoCTF"
SAY picoCTF{f0r3nslc4t0r_n30phyt3_267e38f6}
prasanth@prasanth:~$ strings dds1-alpine.flag.img | grep "picoCTF{.*?}"
prasanth@prasanth:~$ strings dds1-alpine.flag.img | grep "picoCTF{"
SAY picoCTF{f0r3nslc4t0r_n30phyt3_267e38f6}
```

Runme.py (General Skills)

```
prasanth@prasanth: ~
prasanth@prasanth:~$ wget https://artifacts.picoctf.net/c/86/runme.py
--2023-02-02 12:06:06-- https://artifacts.picoctf.net/c/86/runme.py
Resolving artifacts.picoctf.net (artifacts.picoctf.net)... 18.155.181.110, 18.155.181.17, 18.155.181.7, ...
Connecting to artifacts.picoctf.net (artifacts.picoctf.net)|18.155.181.110|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 270 [application/octet-stream]
Saving to: 'runme.py'

runme.py          100%[=====>]          270  --.-KB/s   in 0s

2023-02-02 12:06:06 (154 MB/s) - 'runme.py' saved [270/270]

prasanth@prasanth:~$ file runme.py
runme.py: Python script, ASCII text executable
prasanth@prasanth:~$ cat runme.py
#!/usr/bin/python3
#####
# Python script which just prints the flag
#####

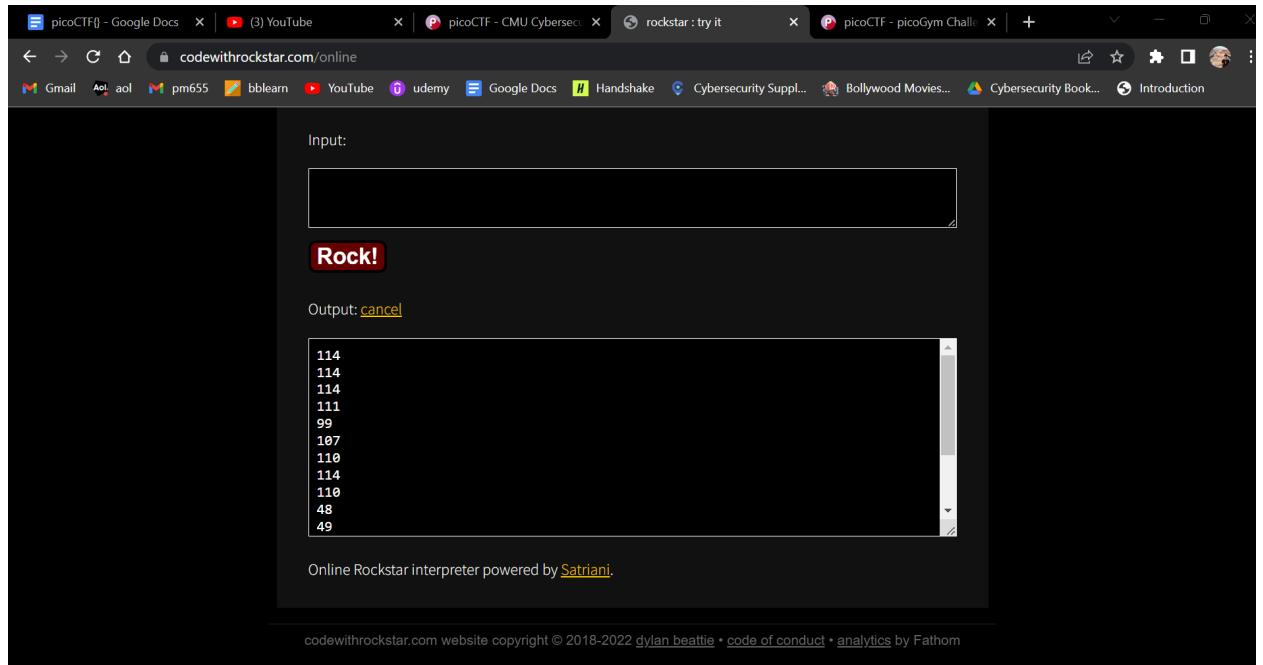
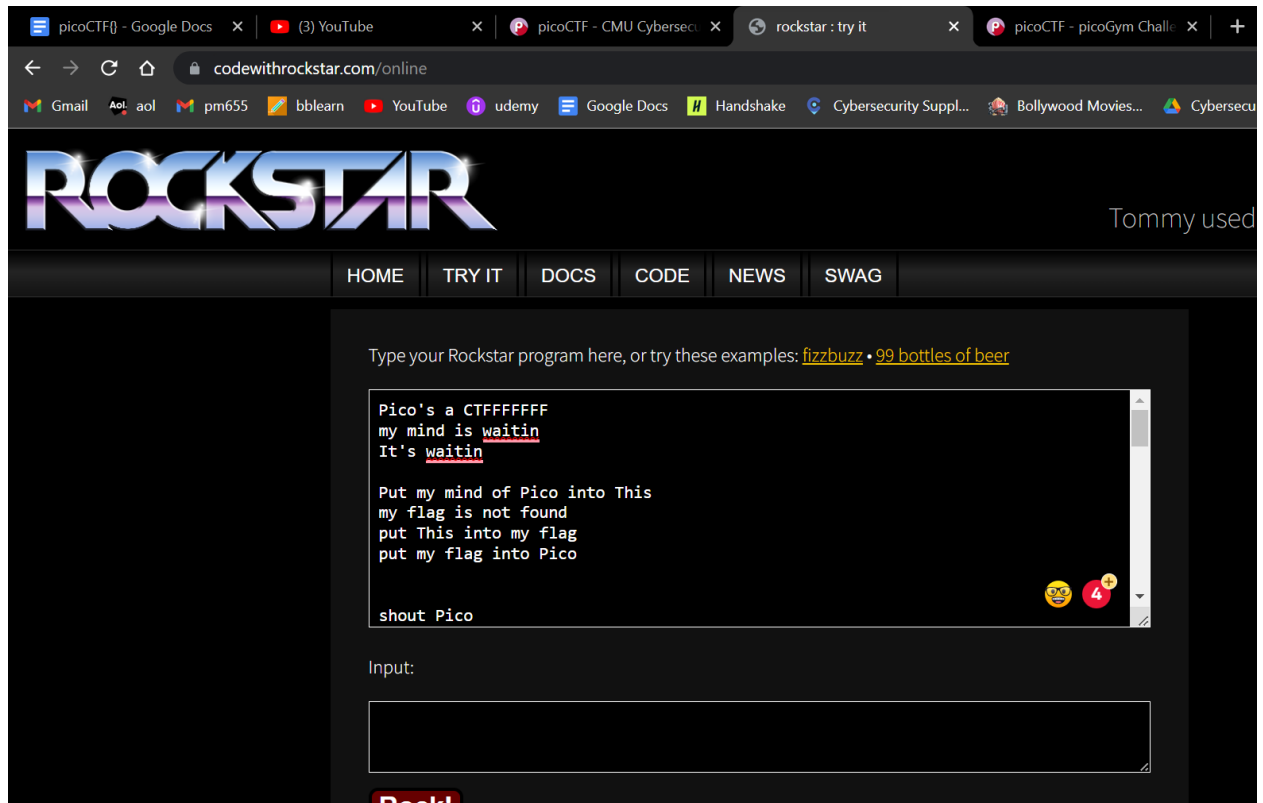
flag='picoCTF{run_s4n1ty_run}'
print(flag)

prasanth@prasanth:~$
```

Codebook (General Skills) (shell) (python)

```
prasanth@prasanth: ~  
prasanth@prasanth:~$ wget https://artifacts.picoctf.net/c/102/code.py  
--2023-02-02 16:19:11-- https://artifacts.picoctf.net/c/102/code.py  
Resolving artifacts.picoctf.net (artifacts.picoctf.net)... 18.155.181.7, 18.155.181.17, 18.155.181.28, ...  
Connecting to artifacts.picoctf.net (artifacts.picoctf.net)|18.155.181.7|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 1278 (1.2K) [application/octet-stream]  
Saving to: 'code.py'  
  
code.py          100%[=====>] 1.25K  --.-KB/s   in 0s  
  
2023-02-02 16:19:11 (299 MB/s) - 'code.py' saved [1278/1278]  
  
prasanth@prasanth:~$ wget https://artifacts.picoctf.net/c/102/codebook.txt  
--2023-02-02 16:19:19-- https://artifacts.picoctf.net/c/102/codebook.txt  
Resolving artifacts.picoctf.net (artifacts.picoctf.net)... 18.155.181.7, 18.155.181.17, 18.155.181.28, ...  
Connecting to artifacts.picoctf.net (artifacts.picoctf.net)|18.155.181.7|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 27 [application/octet-stream]  
Saving to: 'codebook.txt'  
  
codebook.txt     100%[=====>] 27  --.-KB/s   in 0s  
  
2023-02-02 16:19:19 (9.03 MB/s) - 'codebook.txt' saved [27/27]  
  
prasanth@prasanth:~$ python3 code.py  
picoCTF{c0d3b00k_455157_197a982c}  
prasanth@prasanth:~$
```

Mus1c



The text is pasted in the codewithrockstar.com and run. The output is

114
114
114
111
99

107

110

114

110

48

49

49

51

114

Convert the numbers using the ascii format and the flag is rrrocknrn0113r

```
prasanth@prasanth:~$ mkdir Addadshashanammu
mkdir: cannot create directory 'Addadshashanammu': File exists
prasanth@prasanth:~$ cd Addadshashanammu
prasanth@prasanth:~/Addadshashanammu$ ls
Almurbalarammi
prasanth@prasanth:~/Addadshashanammu$ cd Almurbalarammi/
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi$ ls
Ashalmimilkala
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi$ cd Ashalmimilkala/
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala$ cd Assurnabitaashpi/
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi$ cd Maelkashishi/
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi/Maelkashishi$ ls
Onnissiralis
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi/Maelkashishi$ cd Onnissiralis/
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi/Maelkashishi/Onnissiralis$ ls
Ularradallaku
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi/Maelkashishi/Onnissiralis$ cd Ularradalla
ku/
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi/Maelkashishi/Onnissiralis/Ularradallaku$
ls
fang-of-haynekhtnamet
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi/Maelkashishi/Onnissiralis/Ularradallaku$
cd fang-of-haynekhtnamet
-bash: cd: fang-of-haynekhtnamet: Not a directory
```

```
prasanth@prasanth:~/Addadshashanammu/Almurbalarammi/Ashalmimilkala/Assurnabitaashpi/Maelkashishi/Onnissiralis/Ularradallaku$
strings fang-of-haynekhtnamet
/lib64/ld-linux-x86-64.so.2
libc.so.6
puts
__cxa_finalize
__libc_start_main
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
AWAVI
AUATL
[]A\A]A^A_
*ZAP!* picoCTF{l3v3l_up!_t4k3_4_r35t!_f3553887}
```

Basic-mod2 (Cryptography)

```
>>> l=[104 ,290 ,356 ,313 ,262, 337, 354, 229 ,146, 297 ,118 ,373 ,221 ,359 ,338, 321, 288 ,79 ,214 ,277, 131 ,190 ,377]
>>> s=[]
>>> for i in l:
...     s.append(i%41)
...
>>> s
[22, 3, 28, 26, 16, 9, 26, 24, 23, 10, 36, 4, 16, 31, 10, 34, 1, 38, 9, 31, 8, 26, 8]
```

```
>>> for i in s:  
...     print(pow(i,-1,41))  
...  
28  
14  
22  
30  
18  
32  
30  
12  
25  
37  
8  
31  
18  
4  
37  
35  
1  
27  
32  
4  
36  
30  
36
```

Now map 1-26 with alphabets[A-Z], 27-36 with numbers from [0-9] and 37 as “_” and you get the answer.

Vault-door-4

Use the online websites to convert the characters into ascii. The flag is
`picoCTF{jU5t_4_bUnCh_of_bYt3s_8f4a6cbf3b}`

Note: strip 0x for hex values and trailing zeros for octal values.

Vault-door-5 (java)

Take the url and decode it into base64 and copy the result again do the url decoding. You get the flag as `c0nv3rt1ng_fr0m_ba5e_64_e3152bf4`

encoding.tools/#gadget/change_base.base64_decode

Encoding Tools

Change Base

- Hex Encode
- Hex Decode
- Base64 Encode
- Base64 Decode

Hash

- MD5
- SHA-1
- SHA-2

Input

Text Input

Web

- URL Encode
- URL Decode
- HTML Encode
- HTML Decode

Text Input: `mJTY2JTCyJTMwJTZkTVmJTYxJTYxJTM1JTY1JTVmJTM2JTM0JTVmJTY1JTMzJTMxJTM1JTMxJTYxJTY2JTM0`

Base64 Decode Output: `%63%30%6e%76%33%72%74%31%6e%67%5`

encoding.tools/#gadget/web.url_decode

Encoding Tools

Change Base

- Hex Encode
- Hex Decode
- Base64 Encode
- Base64 Decode

Hash

- MD5
- SHA-1
- SHA-2

Input

Text Input

Web

- URL Encode
- URL Decode
- HTML Encode
- HTML Decode

Text Input: `%63%30%6e%76%33%72%74%31%6e%67%5f%66%72%30%6d%5f%62%61%35%65%5f%36%34%5f%65%33%31%35%32%`

URL Decode Output: `c0nv3rt1ng_fr0m_ba5e_64_e3152bf4`

```

prasanth@prasanth:~$ nc jupiter.challenges.picoctf.org 15130
Let us see how data is stored
test
Please give the 01110100 01100101 01110011 01110100 as a word.
...
you have 45 seconds.....

Input:
test
Please give me the 143 150 141 151 162 as a word.
Input:
chair
Please give me the 616e696d6174696f6e as a word.
Input:
animation
You've beaten the challenge
Flag: picoCTF{learning_about_converting_values_02167de8}

```

```

prasanth@prasanth: ~/pw_crack_1
#####

flag_enc = open('level1.flag.txt.enc', 'rb').read()

def level_1_pw_check():
    user_pw = input("Please enter correct password for flag: ")
    if( user_pw == "1e1a"):
        print("Welcome back... your flag, user:")
        decryption = str_xor(flag_enc.decode(), user_pw)
        print(decryption)
        return
    print("That password is incorrect")

level_1_pw_check()

prasanth@prasanth:~/pw_crack_1
$ python3 level.py
python3: can't open file '/home/prasanth/pw_crack_1/level.py': [Errno 2] No such file or directory

prasanth@prasanth:~/pw_crack_1
$ python3 level1.py
Please enter correct password for flag: 1e1a
Welcome back... your flag, user:
picoCTF{545h_r1ng1ng_fa343060}

prasanth@prasanth:~/pw_crack_1
$

```

PW Crack 3

```
(prasanth@prasanth)~/pw_crack_3
$ bvi level3.hash.bin

bvi version 1.4.1 (C) GPL 1996-2019 by Gerhard Buerghmann

(prasanth@prasanth)~/pw_crack_3
$ strings level3.flag.txt.enc
RXw6wLYV

(prasanth@prasanth)~/pw_crack_3
$ python3 level3.py
Please enter correct password for flag: 6997
That password is incorrect

(prasanth@prasanth)~/pw_crack_3
$ python3 level3.py
Please enter correct password for flag: 3ac8
That password is incorrect

(prasanth@prasanth)~/pw_crack_3
$ python3 level3.py
Please enter correct password for flag: f0ac
That password is incorrect

(prasanth@prasanth)~/pw_crack_3
$ python3 level3.py
Please enter correct password for flag: 4b17
Welcome back... your flag, user:
picoCTF{m45h_fl1ng1ng_2b072a90}

(prasanth@prasanth)~/pw_crack_3
$
```

Pw_crack_4

Modified code

```

crasanth@prasanth: ~/pw cri: + ~
pw_bytes = bytearray()
pw_bytes.extend(pw_str.encode())
m = hashlib.md5()
m.update(pw_bytes)
return m.digest()

def level_4_pw_check():
    pos_pw_list = ["158f", "1655", "d21e", "4966", "ed69", "1010", "dded", "844c", "40ab", "a948", "156c", "ab7f", "4a5f", "
e38c", "ba12", "f7fd", "d780", "4f4d", "5ba1", "96c5", "55b9", "8a67", "d32b", "aa7a", "514b", "e4e1", "1230", "cd19", "d6dd
", "b01f", "fd2f", "7587", "86c2", "d7b8", "55a2", "b77c", "7ffe", "4420", "e0ee", "d8fb", "d748", "b0fe", "2a37", "a638", "
52db", "51b7", "5526", "40ed", "5356", "6ad4", "2ddd", "177d", "84ae", "cf88", "97a3", "17ad", "7124", "eff2", "e373", "c974
", "7689", "88b2", "e899", "d042", "47d9", "cca9", "ab2a", "de77", "4654", "9ecb", "ab6e", "bb8e", "b76b", "d661", "63f8", "
7095", "567e", "b837", "2b80", "ad4f", "c514", "ffa4", "fc37", "7254", "b48b", "d38b", "a02b", "ec6c", "eacc", "8b70", "b03e
", "1b36", "81ff", "77e4", "dba6", "59d9", "fd6a", "5653", "8b95", "d0e5"]
    for user_pw in pos_pw_list:
        user_pw_hash = hash_pw(user_pw)

        if( user_pw_hash == correct_pw_hash ):
            print("Welcome back... your flag, user:")
            print("The correct password is :",user_pw)
            decryption = str_xor(flag_enc.decode(), user_pw)
            print(decryption)
            return
        print("That password is incorrect")

level_4_pw_check()

```

The result

```
prasanth@prasanth: ~/pw_crack$  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
Welcome back.. your flag, user:  
The correct password is : eacc  
picoCTF{fL45h_5prlnglmg_cf341ff1}  
prasanth@prasanth:~/pw_crack_u$
```

PW_Crack_5

Modified code:

```

prasanth@prasanth: ~/pw_cri
+ - x

flag_enc = open('level5.flag.txt.enc', 'rb').read()
correct_pw_hash = open('level5.hash.bin', 'rb').read()

def hash_pw(pw_str):
    pw_bytes = bytearray()
    pw_bytes.extend(pw_str.encode())
    m = hashlib.md5()
    m.update(pw_bytes)
    return m.digest()

def level_5_pw_check():
    f=open("dictionary.txt","r")
    for user_pw in f:
        user_pw=user_pw.strip()
        #user_pw= input("Please enter correct password for flag: ")
        user_pw_hash = hash_pw(user_pw)
        if( user_pw_hash == correct_pw_hash ):
            print("Welcome back... your flag, user:")
            decryption = str_xor(flag_enc.decode(), user_pw)
            print(decryption)
            time.sleep(5)
            return
    print("That password is incorrect")

level_5_pw_check()
"level5.py" [dos] 44L, 1324B
38,18 92%
```

Flag

```
prasanth@prasanth: ~/pw_cri ×  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
That password is incorrect  
Welcome back... your flag, user:  
picoCTF{h45h_sllnglng_40f26f81}
```

John_Pollard

ChatGPT was very helpful in helping this problem. It gave me a hint on how to solve this problem. Factorize the modulus to get p and q values.

```
prasanth@prasanth:~/john_pollard$ openssl x509 -inform pem -in cert.pem -noout -text
Certificate:
  Data:
    Version: 1 (0x0)
    Serial Number: 12345 (0x3039)
    Signature Algorithm: md2WithRSAEncryption
    Issuer: CN = PicoCTF
    Validity
      Not Before: Jul  8 07:21:18 2019 GMT
      Not After : Jun 26 17:34:38 2019 GMT
    Subject: OU = PicoCTF, O = PicoCTF, L = PicoCTF, ST = PicoCTF, C = US, CN = PicoCTF
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
        Public-Key: (53 bit)
        Modulus: 4966306421059967 (0x11a4d45212b17f)
        Exponent: 65537 (0x10001)
    Signature Algorithm: md2WithRSAEncryption
    Signature Value:
      07:6a:5d:61:32:c1:9e:05:bd:eb:77:f3:aa:fb:bb:83:82:eb:
      9e:a2:93:af:0c:2f:3a:e2:1a:e9:74:6b:9b:82:d8:ef:fe:1a:
      c8:b2:98:7b:16:dc:4c:d8:1e:2b:92:4c:80:78:85:7b:d3:cc:
      b7:d4:72:29:94:22:eb:bb:11:5d:b2:9a:af:7c:6b:cb:b0:2c:
      a7:91:87:ec:63:bd:22:e8:8f:dd:38:0e:a5:e1:0a:bf:35:d9:
      a4:3c:3c:7b:79:da:8e:4f:fc:ca:e2:38:67:45:a7:de:6e:a2:
      6e:71:71:47:f0:09:3e:1b:a0:12:35:15:a1:29:f1:59:25:35:
      a3:e4:2a:32:4c:c2:2e:b4:b5:3d:94:38:93:5e:78:37:ac:35:
      35:06:15:e0:d3:87:a2:d6:3b:c0:7f:45:2b:b6:97:8e:03:a8:
      d4:c9:e0:8b:68:a0:c5:45:ba:ce:9b:7e:71:23:bf:6b:db:cc:
      8e:f2:78:35:50:0c:d3:45:c9:6f:90:e4:6d:6f:c2:cc:c7:0e:
```

Not secure | factordb.com/index.php?query=4966306421059967

Search Sequences Report results Factor tables Status Downloads Login

4966306421059967

Result:		
status (2)	digits	number
FF	16 (show)	4966306421059967 <16> = 67867967 · 73176001

More information

ECM

3. Look for the "Subject Public Key Info" section of the output, which should contain information about the public key algorithm and modulus (n) value. For example:

```
ruby                                                                    Copy code

Subject Public Key Info:
  Public Key Algorithm: rsaEncryption
  Public-Key: (2048 bit)
  Modulus:
    00:aa:bb:cc:dd:ee:ff:00:11:22:33:44:55:66:77:
    ...
    ...
    ...
    88:99:aa:bb:cc:dd:ee:ff
  Exponent: 65537 (0x100000001)  Regenerate response
```

ChatGPT Feb 13 Version. Free Research Preview. Our goal is to make AI systems more natural and safe to interact with. Your feedback will help us improve.

Take the modulus and factorize it. You get two primes and the correct flag format is picoCTF{73176001,67867967}

Plumbing(General Skills)

```
prasanth@prasanth: ~
prasanth@prasanth:~$ nc jupiter.challenges.picoctf.org 14291 | grep "picoCTF{.*}" --color=none
picoCTF{digital_plumb3r_ea8bfec7}
```

Mind your Ps and Qs Cryptography

```
prasanth@prasanth: ~/Mind_ X prasanth@prasanth: ~ X prasanth@prasanth: ~ X + v
from Crypto.Util.number import inverse
from binascii import unhexlify
n=1422450808944701344261903748621562998784243662042303391362692043823716783771691667
e=65537
c=843044897663847841476319711639772861390329326681532977209935413827620909782846667
p=2159947535959146091116171018558446546179
q=658558036833541874645521278345168572231473
phi=(p-1)*(q-1)
d=inverse(e,phi)
m=pow(c,d,n)
hex_string=(hex(m)[2:-1])
if(len(hex_string))%2!=0:
    hex_string=hex_string+"0"
byte_string =unhexlify(hex_string)
print(byte_string)

~
~
~
~
~
~
~
~
~
~
~

prasanth@prasanth: ~/Mind_ X prasanth@prasanth: ~ X prasanth@prasanth: ~ X +
prasanth@prasanth:~/Mind_your_Ps_and_Qs$ python3 exp.py
b'picoCTF{sma11_N_n0_g0od_00264570p}'
prasanth@prasanth:~/Mind_your_Ps_and_Qs$
```

The answer is picoCTF{sma11_N_n0_g0od_00264570}

Big Zip

```
prasanth@prasanth: ~/big_zip
inflating: big-zip-files/folder_wdhgdgrbfc/fxaxiryjldhjugsxhndjglp.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_qmzrrrkuaqnl.txt
inflating: big-zip-files/folder_wdhgdgrbfc/xkktktzhdxnfx.txt
inflating: big-zip-files/folder_wdhgdgrbfc/puqtvnrhomqbrkguy.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_czpyijqgdzhwkfkdyd.txt
inflating: big-zip-files/folder_wdhgdgrbfc/ncpphsegf.txt
inflating: big-zip-files/folder_wdhgdgrbfc/rpagkqbzcuxepx.txt
inflating: big-zip-files/folder_wdhgdgrbfc/pdppdhedydlawgvhwm.txt
inflating: big-zip-files/folder_wdhgdgrbfc/qbwalzyyprvrcjpepe.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_ljeldszgyuc.txt
extracting: big-zip-files/folder_wdhgdgrbfc/lmjsuinffffmpyjmmk.txt
extracting: big-zip-files/folder_wdhgdgrbfc/tdwocpenvymtoj.txt
extracting: big-zip-files/folder_wdhgdgrbfc/fdsjflubxcxwhpv.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_jyvxtmmtpl.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_epcuiockebhmxxtago.txt
inflating: big-zip-files/folder_wdhgdgrbfc/tvnwbgmapeulf.txt
extracting: big-zip-files/folder_wdhgdgrbfc/finitvya.txt
inflating: big-zip-files/folder_wdhgdgrbfc/bondkyoxvdcgxyq.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_hacfxytdwkdiycfwatiyvusg.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_jdlhinpycace.txt
inflating: big-zip-files/folder_wdhgdgrbfc/gnsnwwhmlslslscapr.txt
inflating: big-zip-files/folder_wdhgdgrbfc/file_ximyquuowm.txt
inflating: big-zip-files/folder_wdhgdgrbfc/siizcexeduftjnvian.txt
inflating: big-zip-files/mktyhgmedcj.txt
prasanth@prasanth:~/big_zip$ ls
big-zip-files  big-zip-files.zip
prasanth@prasanth:~/big_zip$ cd big-zip-files/
prasanth@prasanth:~/big_zip/big-zip-files$ grep -r big-zip-files
prasanth@prasanth:~/big_zip/big-zip-files$ grep -r "picoCTF"
folder_pmbymkjcyja/folder_cawigcwgvgv/folder_ltdayfmktr/folder_fnpfclfyee/whzxrpivpqld.txt:information on the record will last
a billion years. Genes and brains and books encode picoCTF{gr3p_15_m4g1c_ef8790dc}
```