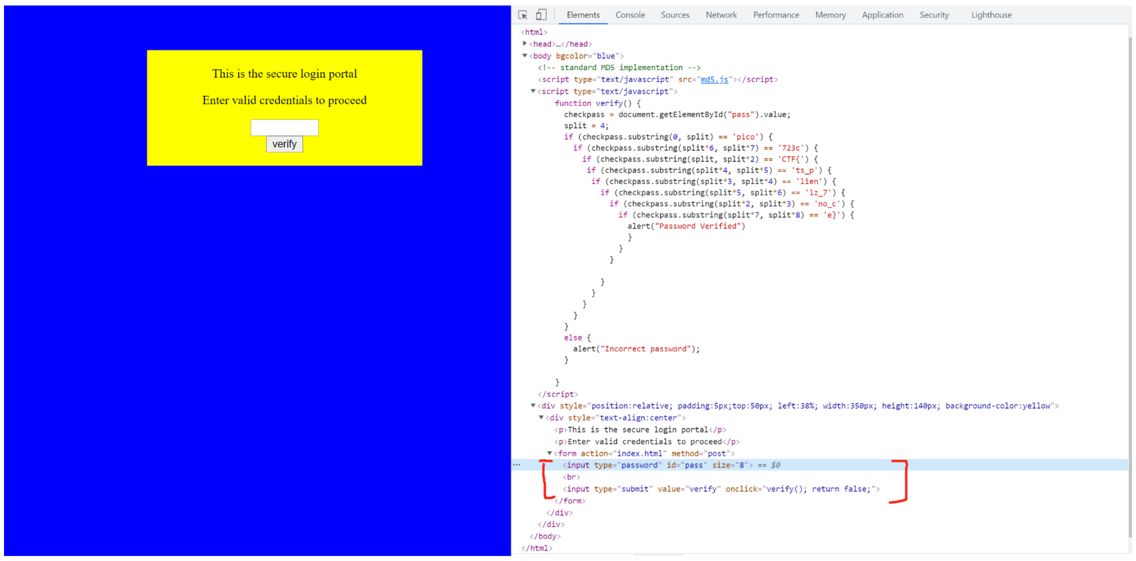
Rupeng Na 250884549

Web Exploitation

- Dont-use-client-side

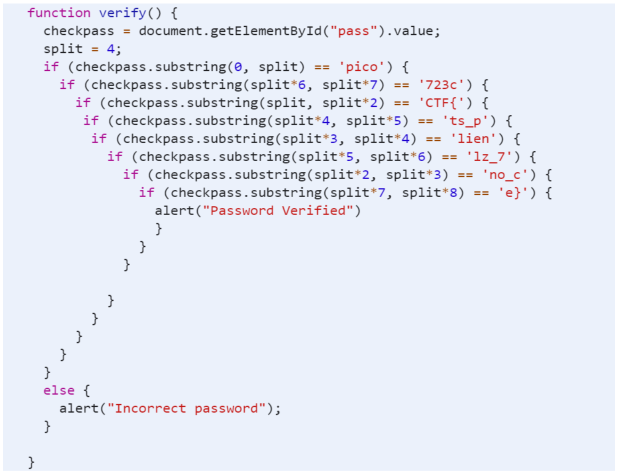
The flag is picoCTF{no_clients_plz_7723ce}

This part of the question is about web exploitation. I will try to use the function called "inspect" to check the page source first.



```
<html>
▶<head>…</head>
▼<body bgcolor="blue">
   <!-- standard MD5 implementation -->
   <script type="text/javascript" src="md5.js"></script>
  ▼<script type="text/javascript">
      function verify() {
        checkpass = document.getElementById("pass").value;
        split = 4;
        if (checkpass.substring(0, split) == 'pico') {
          if (checkpass.substring(split*6, split*7) == '723c') {
            if (checkpass.substring(split, split*2) == 'CTF{') {
             if (checkpass.substring(split*4, split*5) == 'ts_p') {
              if (checkpass.substring(split*3, split*4) == 'lien') {
                if (checkpass.substring(split*5, split*6) == 'lz_7') {
                  if (checkpass.substring(split*2, split*3) == 'no_c') {
                    if (checkpass.substring(split*7, split*8) == 'e}') {
                      alert("Password Verified")
                      }
                    }
                  }

                }
              }
            }
          }
        }
        else {
          alert("Incorrect password");
        }

      }
   </script>
 ▼<div style="position:relative; padding:5px;top:50px; left:38%; width:350px; height:140px; background-color:yellow">
  ▼<div style="text-align:center">
     <p>This is the secure login portal</p>
     <p>Enter valid credentials to proceed</p>
    ▼<form action="index.html" method="post">
       <input type="password" id="pass" size="8"> == $0
       <br>
       <input type="submit" value="verify" onclick="verify(); return false;">
     </form>
   </div>
  </div>
 </body>
</html>
```

As you can see, the part that I highlighted by red is used to implement the verification. The 'submit' calls the function verify() and the password's id is 'pass'. Then I checked back and found the function verify().



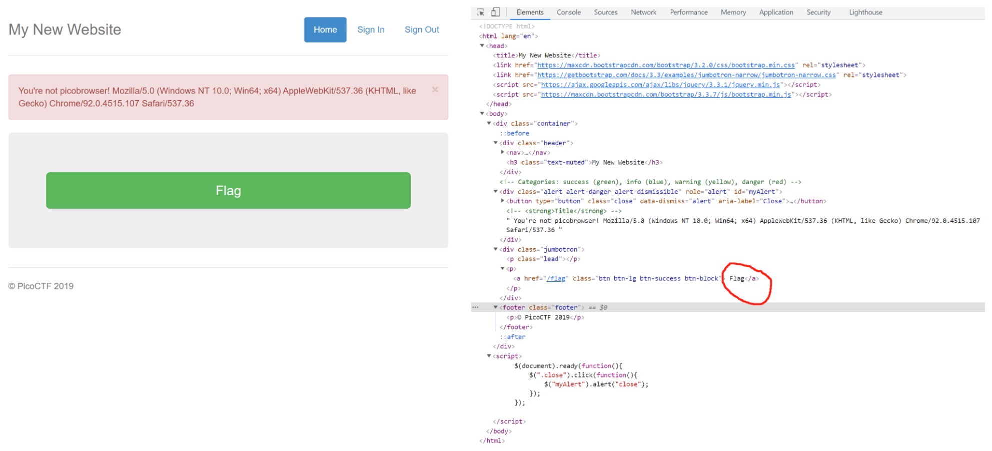
```
function verify() {
  checkpass = document.getElementById("pass").value;
  split = 4;
  if (checkpass.substring(0, split) == 'pico') {
    if (checkpass.substring(split*6, split*7) == '723c') {
      if (checkpass.substring(split, split*2) == 'CTF{') {
       if (checkpass.substring(split*4, split*5) == 'ts_p') {
        if (checkpass.substring(split*3, split*4) == 'lien') {
          if (checkpass.substring(split*5, split*6) == 'lz_7') {
            if (checkpass.substring(split*2, split*3) == 'no_c') {
              if (checkpass.substring(split*7, split*8) == 'e}') {
                alert("Password Verified")
                }
              }
            }

          }
        }
      }
    }
  }
  else {
    alert("Incorrect password");
  }

}
```

The function verify() sets a variable called ‘checkpass’ and ‘split’. The ‘checkpass’ is used to receive the answer from the user. The ‘split’ is the tricky part for this question. It sets the substring to several parts and each part of them length is 4. The ‘if else’ which belows the variable is used to check if the answer is verified or not. The whole answer is split to 8 substring and each of the ‘if’ statements will check. There’s a tricky part that we need to rearrange the substring when we solve this question. After that we got the correct FLAG: picoCTF{no_clients_plz_7723ce}.

- Picobrowser

The flag is: picoCTF{p1c0_s3cr3t_ag3nt_51414fa7}

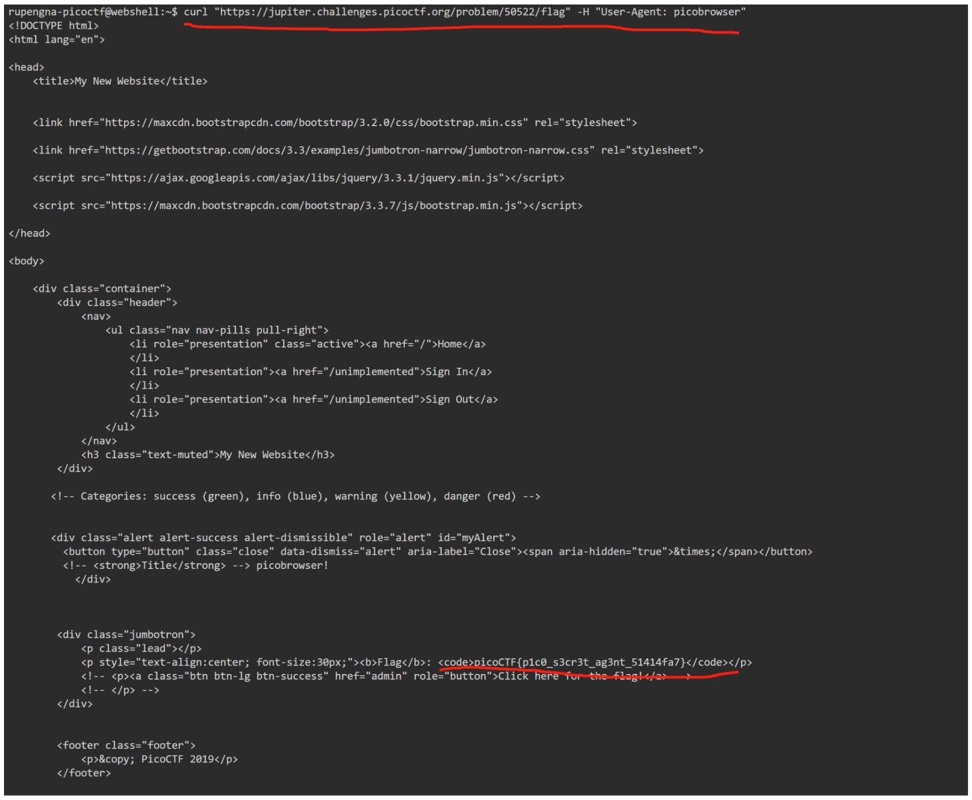
At first when I was on the website, there’s a obverse button called ‘Flag’. I clicked it and it showed me I’m not the ‘picobrower’. I inspected the source code and found the flag(highlighted by the red). I realised that something has been hidden because I don’t have the permission. I don’t know how to solve it so I googled the error string. The string is User-Agent in HTTP which is used to identify the app or OS of the requesting user agent.



You're not picobrowser! Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.107 Safari/537.36

```
<!DOCTYPE html>
<html lang="en">
▼<head>
   <title>My New Website</title>
   <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css" rel="stylesheet">
   <link href="https://getbootstrap.com/docs/3.3/examples/jumbotron-narrow/jumbotron-narrow.css" rel="stylesheet">
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
  </head>
▼<body>
  ▼<div class="container">
     ::before
    ▼<div class="header">
      ▶<nav>…</nav>
       <h3 class="text-muted">My New Website</h3>
     </div>
     <!-- Categories: success (green), info (blue), warning (yellow), danger (red) -->
    ▼<div class="alert alert-danger alert-dismissible" role="alert" id="myAlert">
      ▶<button type="button" class="close" data-dismiss="alert" aria-label="Close">…</button>
       <!-- <strong>Title</strong> -->
       " You're not picobrowser! Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.107
       Safari/537.36 "
     </div>
    ▼<div class="jumbotron">
       <p class="lead"></p>
      ▼<p>
         <a href="/flag" class="btn btn-lg btn-success btn-block"> Flag</a>
       </p>
     </div>
...  ▼<footer class="footer"> == $0
       <p>© PicoCTF 2019</p>
     </footer>
     ::after
   </div>
  ▼<script>
         $(document).ready(function(){
             $(".close").click(function(){
                 $("myAlert").alert("close");
             });
         });

   </script>
  </body>
</html>
```

It seems that I need to set a new header to get the flag. I use the webshell on the picoCTF and use the command '-H' to change my browser to 'picobrowser'.



```
rupengna-picoctf@webshell:~$ curl "https://jupiter.challenges.picoctf.org/problem/50522/flag" -H "User-Agent: picobrowser"
<!DOCTYPE html>
<html lang="en">

<head>
    <title>My New Website</title>


    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css" rel="stylesheet">

    <link href="https://getbootstrap.com/docs/3.3/examples/jumbotron-narrow/jumbotron-narrow.css" rel="stylesheet">

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</head>

<body>

    <div class="container">
        <div class="header">
            <nav>
                <ul class="nav nav-pills pull-right">
                    <li role="presentation" class="active"><a href="/">Home</a>
                    </li>
                    <li role="presentation"><a href="/unimplemented">Sign In</a>
                    </li>
                    <li role="presentation"><a href="/unimplemented">Sign Out</a>
                    </li>
                </ul>
            </nav>
            <h3 class="text-muted">My New Website</h3>
        </div>

       <!-- Categories: success (green), info (blue), warning (yellow), danger (red) -->


       <div class="alert alert-success alert-dismissible" role="alert" id="myAlert">
         <button type="button" class="close" data-dismiss="alert" aria-label="Close"><span aria-hidden="true">&times;</span></button>
         <!-- <strong>Title</strong> --> picobrowser!
           </div>



        <div class="jumbotron">
            <p class="lead"></p>
            <p style="text-align:center; font-size:30px;"><b>Flag</b>: <code>picoCTF{p1c0_s3cr3t_ag3nt_51414fa7}</code></p>
            <!-- <p><a class="btn btn-lg btn-success" href="admin" role="button">Click here for the flag!</a> -->
            <!-- </p> -->
        </div>


        <footer class="footer">
            <p>&copy; PicoCTF 2019</p>
        </footer>
```

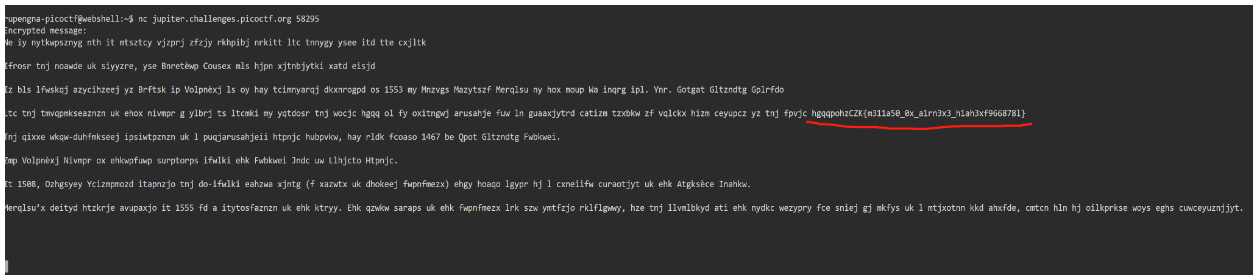
I changed my header and then the flag appeared. The flag is picoCTF{p1c0_s3cr3t_ag3nt_51414fa7}.

Cryptography

- la cifra de

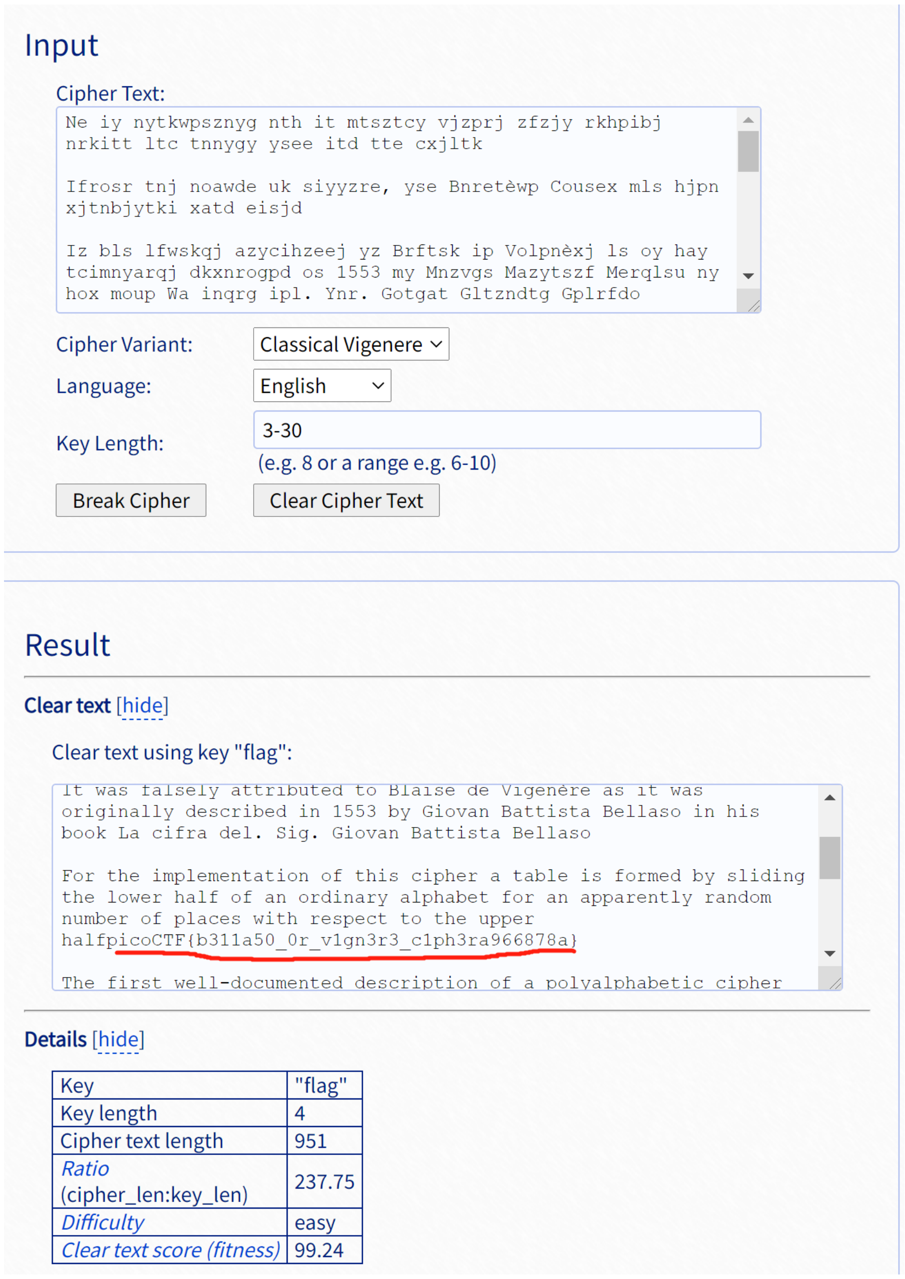
The flag is: picoCTF{b311a50_0r_v1gn3r3_c1ph3ra966878a}

This is kind of a question about cryptography and the hind shows me that there are tools that make this easy. My problem-solving idea is to first find this encrypted text, and then try to find a solver to decipher the flag. I enter the code they give me and got the thing like this



```
rupengna-picoctf@webshell:~$ nc jupiter.challenges.picoctf.org 58295
Encrypted message:
Ne iy nytkwpsznyg nth it mtsztcy vjzprj zfzjy rkhpibj nrkitt ltc tnnygy ysee itd tte cxjltk

Ifrosr tnj noawde uk siyyzre, yse Bnretèwp Cousex mls hjpn xjtnbjytki xatd eisjd

Iz bls lfwskqj azycihzeej yz Brftsk ip Volpnèxj ls oy hay tcimnyarqj dkxnrogpd os 1553 my Mnzvgs Mazytszf Merqlsu ny hox moup Wa inqrg ipl. Ynr. Gotgat Gltzndtg Gplrfdo

Ltc tnj tmvqpmkseaznzn uk ehox nivmpr g ylbrj ts ltcmki my yqtdosr tnj wocjc hgqq ol fy oxitngwj arusahje fuw ln guaaxjytrd catizm tzxbkw zf vqlckx hizm ceyupcz yz tnj fpvjc hgqqpohzCZK{m311a50_0x_a1rn3x3_h1ah3xf966878l}

Tnj qixxe wkqw-duhfmkseej ipsiwtpznzn uk l puqjarusahjeii htpnjc hubpvkw, hay rldk fcoaso 1467 be Qpot Gltzndtg Fwbkwei.

Zmp Volpnèxj Nivmpr ox ehkwpfuwp surptorps ifwlki ehk Fwbkwei Jndc uw Llhjcto Htpnjc.

It 1508, Ozhgsyey Ycizmpmozd itapnzjo tnj do-ifwlki eahzwa xjntg (f xazwtx uk dhokeej fwpnfmezx) ehgy hoaqo lgypr hj l cxneiifw curaotjyt uk ehk Atgksèce Inahkw.

Merqlsu’x deityd htzkrje avupaxjo it 1555 fd a itytosfaznzn uk ehk ktryy. Ehk qzwkw saraps uk ehk fwpnfmezx lrk szw ymtfzjo rklflgwwy, hze tnj llvmlbkyd ati ehk nydkc wezypry fce sniej gj mkfys uk l mtjxotnn kkd ahxfde, cmtcn hln hj oilkprkse woys eghs cuwceyuznjjyt.
```

The red part I highlighted seems like a flag. However, I don’t know what kind of solver I need to use, so I searched about it. Some people suggest that I use Vigenere Solver. I try to paste the paragraph on it and it show me that:


Input

Cipher Text:

```
Ne iy nytkwpsznyg nth it mtsztcy vjzprj zfzjy rkhpibj
nrkitt ltc tnnygy ysee itd tte cxjltk

Ifrosr tnj noawde uk siyyzre, yse Bnretèwp Cousex mls hjpn
xjtnbjytki xatd eisjd

Iz bls lfwskqj azycihzeej yz Brftsk ip Volpnèxj ls oy hay
tcimnyarqj dkxnrogpd os 1553 my Mnzvgs Mazytszf Merqlsu ny
hox moup Wa inqrg ipl. Ynr. Gotgat Gltzndtg Gplrfdo
```

Cipher Variant: Classical Vigenere

Language: English

Key Length: 3-30
(e.g. 8 or a range e.g. 6-10)

Result

Clear text [hide]

Clear text using key "flag":

```
It was falsely attributed to Blaise de Vigenère as it was
originally described in 1553 by Giovan Battista Bellaso in his
book La cifra del. Sig. Giovan Battista Bellaso

For the implementation of this cipher a table is formed by sliding
the lower half of an ordinary alphabet for an apparently random
number of places with respect to the upper
halfpicoCTF{b311a50_0r_v1gn3r3_c1ph3ra966878a}

The first well-documented description of a polyalphabetic cipher
```

Details [hide]

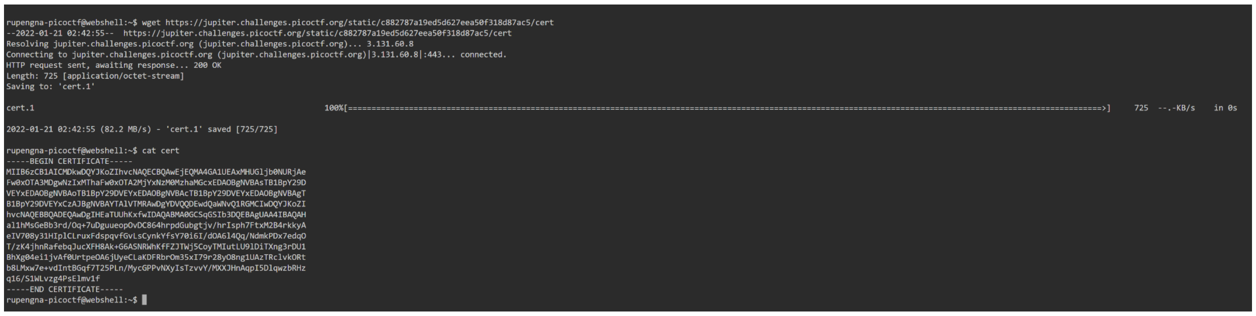
Key	"flag"
Key length	4
Cipher text length	951
Ratio (cipher_len:key_len)	237.75
Difficulty	easy
Clear text score (fitness)	99.24

The solver showed me that the key is "flag" and the flag is CTF{b311a50_0r_v1gn3r3_c1ph3ra966878a}. However, I got the wrong answer. Then I found I didn't add 'pico' before it. The correct answer is picoCTF{b311a50_0r_v1gn3r3_c1ph3ra966878a}.

- john_pollard

The flag is picoCTF{73176001,67867967}.

This question is by far the most difficult question I have come across. I had a lot of difficulties trying to solve it. The first question is how to open the RSA. When I click the link, it will download locally. We need to open it in a webshell, so I try to use ‘wget’ in webshell to get the file.

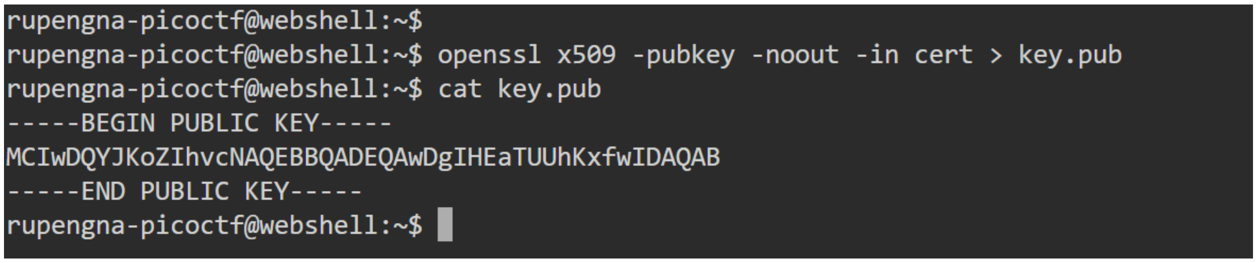


```
rupengna-picoctf@webshell:~$ wget https://jupiter.challenges.picoctf.org/static/c882787a19ed5d627eea50f318d87ac5/cert
--2022-01-21 02:42:55--  https://jupiter.challenges.picoctf.org/static/c882787a19ed5d627eea50f318d87ac5/cert
Resolving jupiter.challenges.picoctf.org (jupiter.challenges.picoctf.org)... 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: 725 [application/octet-stream]
Saving to: 'cert.1'

cert.1                                    100%[=================================================================================================================================================>]     725  --.-KB/s    in 0s

2022-01-21 02:42:55 (82.2 MB/s) - 'cert.1' saved [725/725]

rupengna-picoctf@webshell:~$ cat cert
-----BEGIN CERTIFICATE-----
MIIB6zCB1AICMDkwDQYJKoZIhvcNAQECBQAwEjEQMA4GA1UEAxMHUGljb0NURjAe
Fw0xOTA3MDgwNzIxMThaFw0xOTA2MjYxNzM0MzhaMGcxEDAOBgNVBAsTB1BpY29D
VEYxEDAOBgNVBAoTB1BpY29DVEYxEDAOBgNVBAcTB1BpY29DVEYxEDAOBgNVBAgT
B1BpY29DVEYxCzAJBgNVBAYTAlVTMRAwDgYDVQQDEwdQaWNvQ1RGMCIwDQYJKoZI
hvcNAQEBBQADEQAwDgIHEaTUUhKxfwIDAQABMA0GCSqGSIb3DQEBAgUAA4IBAQAH
al1hMsGeBb3rd/Oq+7uDguueopOvDC864hrpdGubgtjv/hrIsph7FtxM2B4rkkyA
eIV708y31HIplCLruxFdspqvfGvLsCynkYfsY70i6I/dOA6l4Qq/NdmkPDx7edqO
T/zK4jhnRafebqJucXFH8Ak+G6ASNRWhKfFZJTWj5CoyTMIutLU9lDiTXng3rDU1
BhXg04ei1jvAf0UrtpeOA6jUyeCLaKDFRbrOm35xI79r28yO8ng1UAzTRclvkORt
b8LMxw7e+vdIntBGqf7T25PLn/MycGPPvNXyIsTzvvY/MXXJHnAqpI5DlqwzbRHz
q16/S1WLvzg4PsElmv1f
-----END CERTIFICATE-----
rupengna-picoctf@webshell:~$ 
```

When I first saw it, I tried to put the whole paragraph in CSR Decoder but it showed me that it can't be solved. I realized that I need to relearn the structure of RSA. RSA is actually two large prime numbers and an auxiliary value to create and publish a public key. Prime numbers are secret. Anyone can encrypt a message with the public key, but it can only be decoded by someone who knows the prime numbers. So we first need to know the public key, and then try to get the two prime numbers through the calculator, that will make our flag.

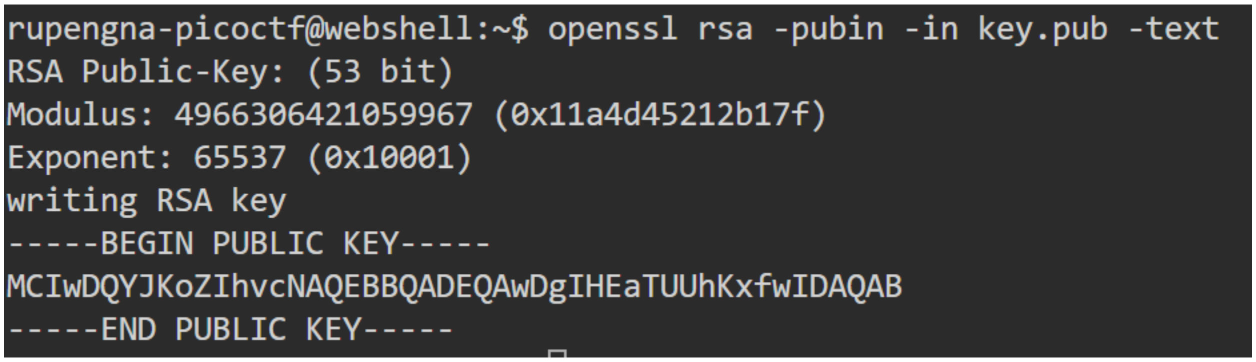
openssl x509 -pubkey -noout -in cert > key.pub

The x509 can be used to display information for the RSA and -publickey will output the certificate of public key information.



```
rupengna-picoctf@webshell:~$
rupengna-picoctf@webshell:~$ openssl x509 -pubkey -noout -in cert > key.pub
rupengna-picoctf@webshell:~$ cat key.pub
-----BEGIN PUBLIC KEY-----
MCIwDQYJKoZIhvcNAQEBBQADEQAwDgIHEaTUUhKxfwIDAQAB
-----END PUBLIC KEY-----
rupengna-picoctf@webshell:~$
```

The rsa command is used to convert forms and the -pubin is used to read the public key. We need to get the modulus for the public key first:



```
rupengna-picoctf@webshell:~$ openssl rsa -pubin -in key.pub -text
RSA Public-Key: (53 bit)
Modulus: 4966306421059967 (0x11a4d45212b17f)
Exponent: 65537 (0x10001)
writing RSA key
-----BEGIN PUBLIC KEY-----
MCIwDQYJKoZIhvcNAQEBBQADEQAwDgIHEaTUUhKxfwIDAQAB
-----END PUBLIC KEY-----
```

The last step we need to do is to find the factor for the modulus.

$$4966\,306421\,059967 = 67\,867967 \times 73\,176001$$

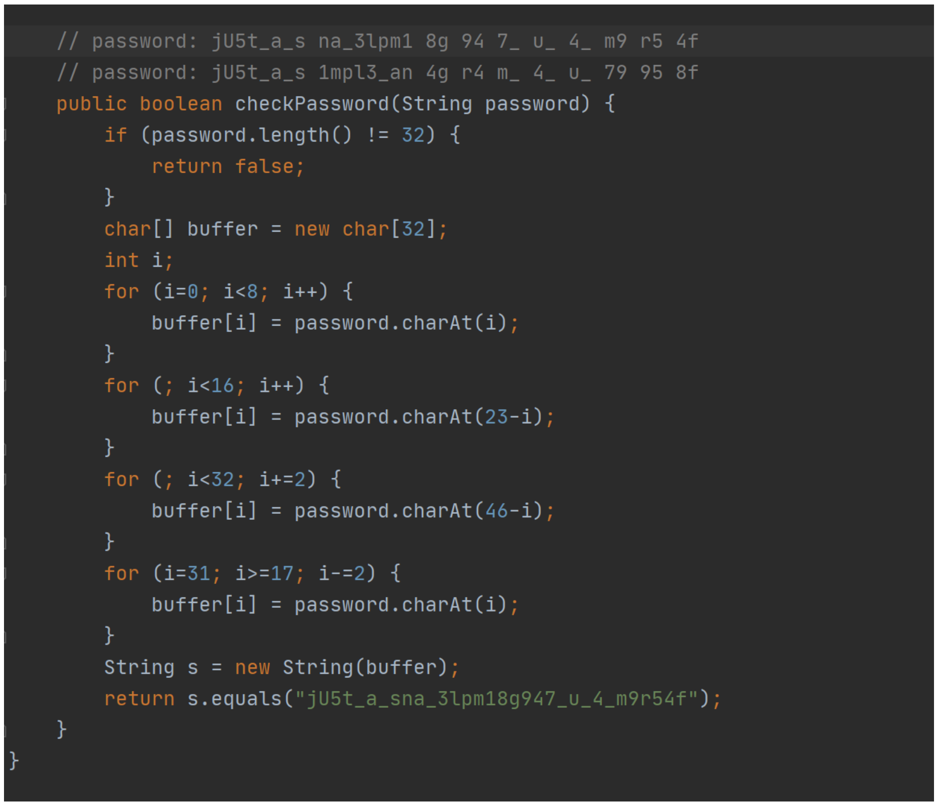
The 67867967 and 73176001 will be the p and q for our flag, we only need to try swapping p and q. Finally I found the answer is picoCTF{73176001,67867967}.

Reverse Engineering

- vault-door3

The flag is picoCTF{jU5t_a_s1mpl3_an4gr4m_4_u_79958f}.

The flag is easy to get. The original string length is 32. We need to split it into 4 parts to solve it. The first part from 0 to 7, it no need to change. The second part from 8 to 15, the for statement shows that each bit needs to be repositioned which needs to be subtracted by 23. The result should be: 1mpl3_an.The third for loop will handle even bits from 16 to 32, and the last for loop will handle odd bits from 16 to 32. We need to insert the results of the fourth part after each of the results of the third part. Then we got the flag which is jU5t_a_s1mpl3_an4gr4m_4_u_79958f

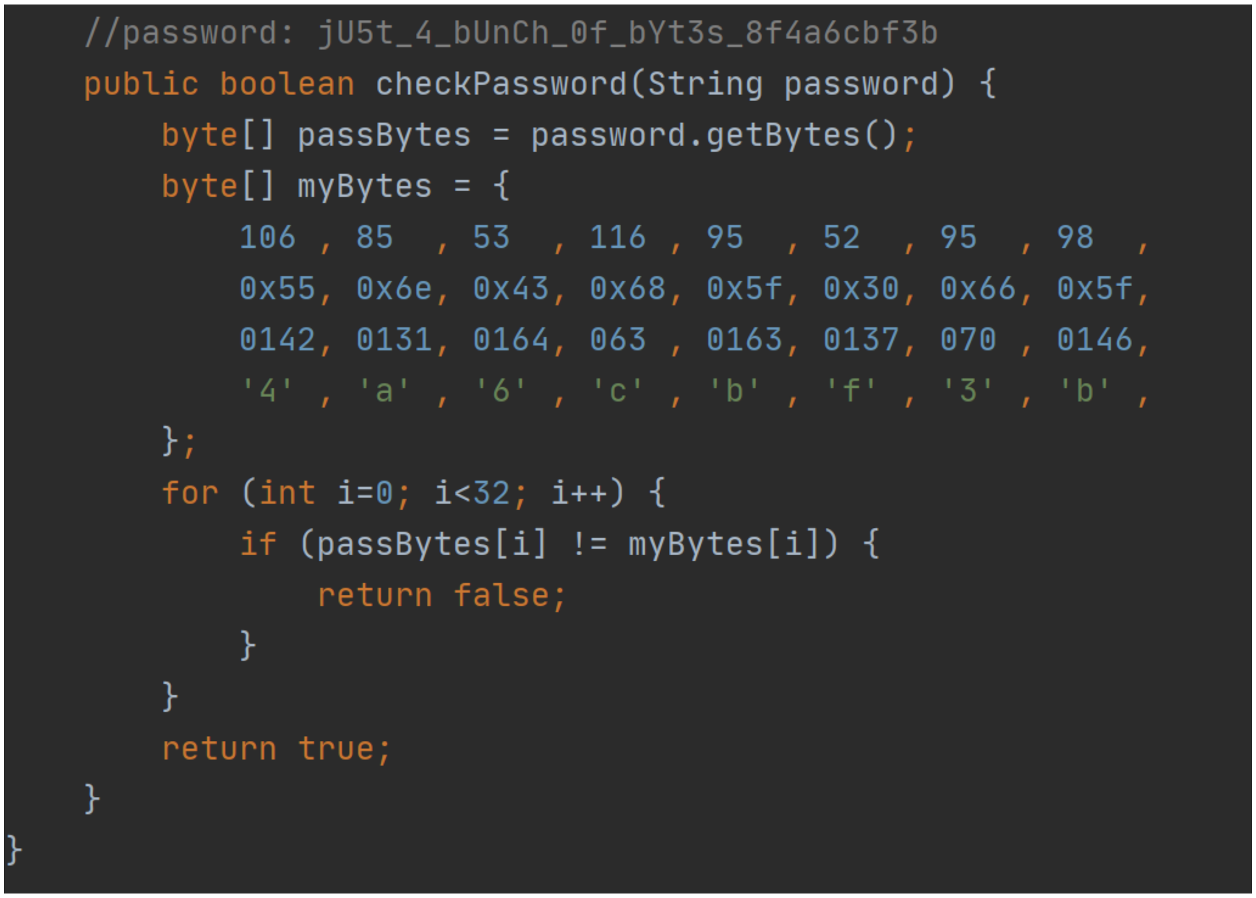


```java
    // password: jU5t_a_s na_3lpm1 8g 94 7_ u_ 4_ m9 r5 4f
    // password: jU5t_a_s 1mpl3_an 4g r4 m_ 4_ u_ 79 95 8f
    public boolean checkPassword(String password) {
        if (password.length() != 32) {
            return false;
        }
        char[] buffer = new char[32];
        int i;
        for (i=0; i<8; i++) {
            buffer[i] = password.charAt(i);
        }
        for (; i<16; i++) {
            buffer[i] = password.charAt(23-i);
        }
        for (; i<32; i+=2) {
            buffer[i] = password.charAt(46-i);
        }
        for (i=31; i>=17; i-=2) {
            buffer[i] = password.charAt(i);
        }
        String s = new String(buffer);
        return s.equals("jU5t_a_sna_3lpm18g947_u_4_m9r54f");
    }
}
```

- vault-door4

The flag is picoCTF{jU5t_4_bUnCh_0f_bYt3s_8f4a6cbf3b}.

In this question, we need to check the ASCII table, because the key is set in the array called myBytes. We found that it includes 4 different types : decimal, heximal, octal and the original value. We can check the table and get the flag.

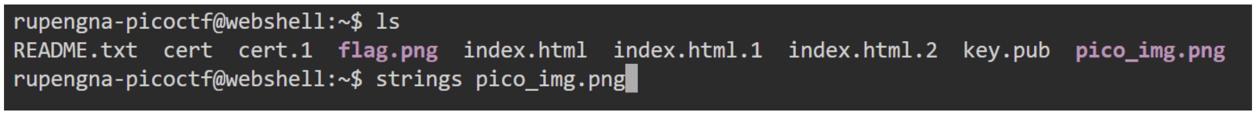


```
    //password: jU5t_4_bUnCh_0f_bYt3s_8f4a6cbf3b
    public boolean checkPassword(String password) {
        byte[] passBytes = password.getBytes();
        byte[] myBytes = {
            106 , 85  , 53  , 116 , 95  , 52  , 95  , 98  ,
            0x55, 0x6e, 0x43, 0x68, 0x5f, 0x30, 0x66, 0x5f,
            0142, 0131, 0164, 063 , 0163, 0137, 070 , 0146,
            '4' , 'a' , '6' , 'c' , 'b' , 'f' , '3' , 'b' ,
        };
        for (int i=0; i<32; i++) {
            if (passBytes[i] != myBytes[i]) {
                return false;
            }
        }
        return true;
    }
}
```

Forensics

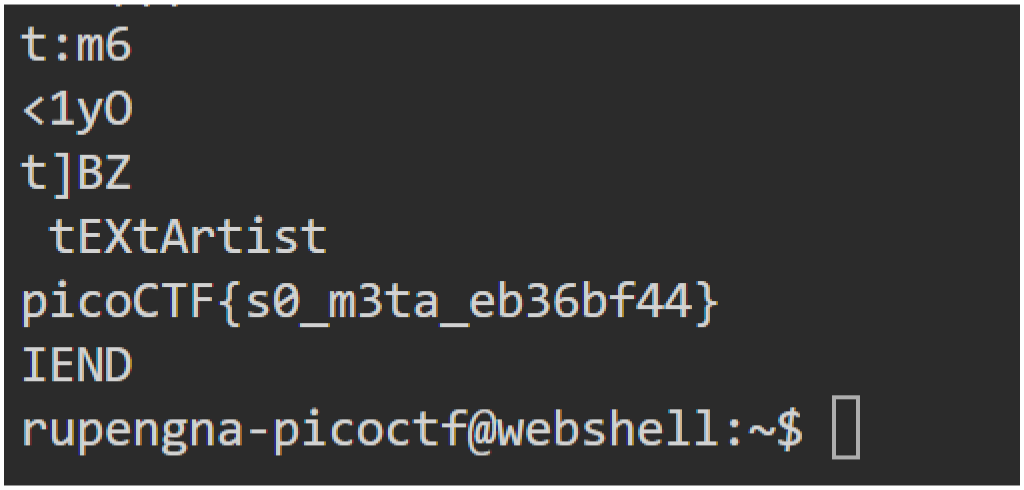
- So Meta

The flag is picoCTF{s0_m3ta_eb36bf44}.



```
rupengna-picoctf@webshell:~$ ls
README.txt  cert  cert.1  flag.png  index.html  index.html.1  index.html.2  key.pub  pico_img.png
rupengna-picoctf@webshell:~$ strings pico_img.png
```

I think when we handle the png file, the most important thing is to treat it as the metadata. As wikipedia said Metadata is any other type of data. The only thing we need to do is just use the ‘strings’ command. Then we got:



```
t:m6
<1yO
t]BZ
 tEXtArtist
picoCTF{s0_m3ta_eb36bf44}
IEND
rupengna-picoctf@webshell:~$
```

- extensions

  The flag is picoCTF{now_you_know_about_extensions}

  After finishing the question about metadata, I just tried to use the same way to solve this problem, but it still garbled. I realised the question’s title is extensions so I tried to change the file type to image(.jpg). The image shows the flag which is picoCTF{now_you_know_about_extensions}.