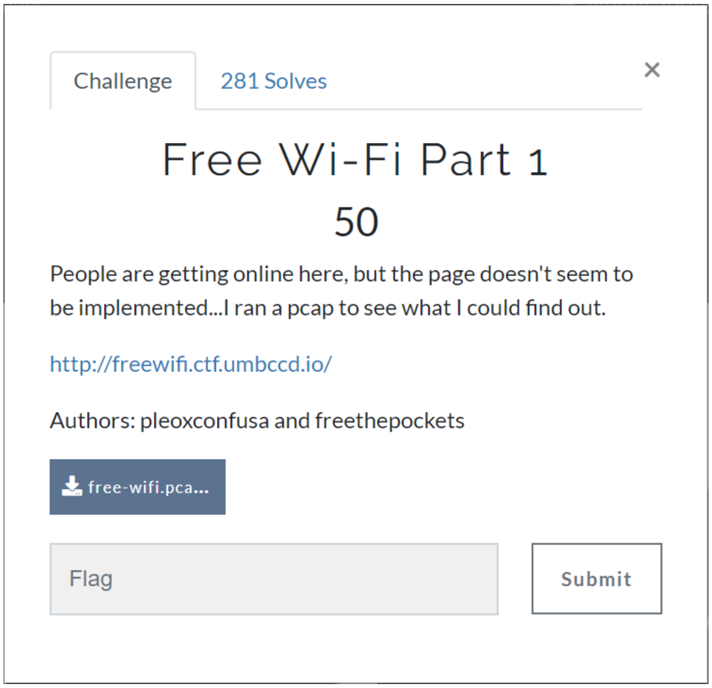
DawgCTF Writeup

By: eWorkaholics

Web/Networking

Free Wi-Fi Part 1



Free Wi-Fi Part 1

50

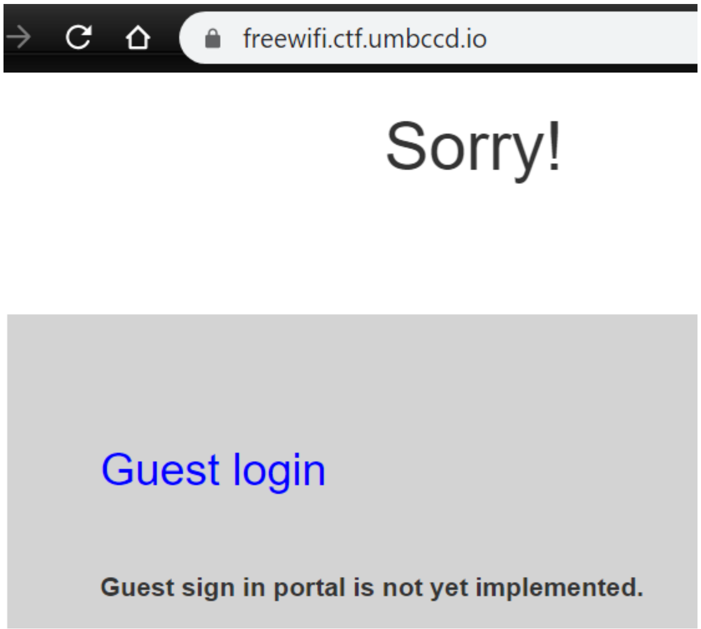
People are getting online here, but the page doesn't seem to be implemented...I ran a pcap to see what I could find out.

http://freewifi.ctf.umbccd.io/

Authors: pleoxconfusa and freethepockets

We are given a link and a pcap file.

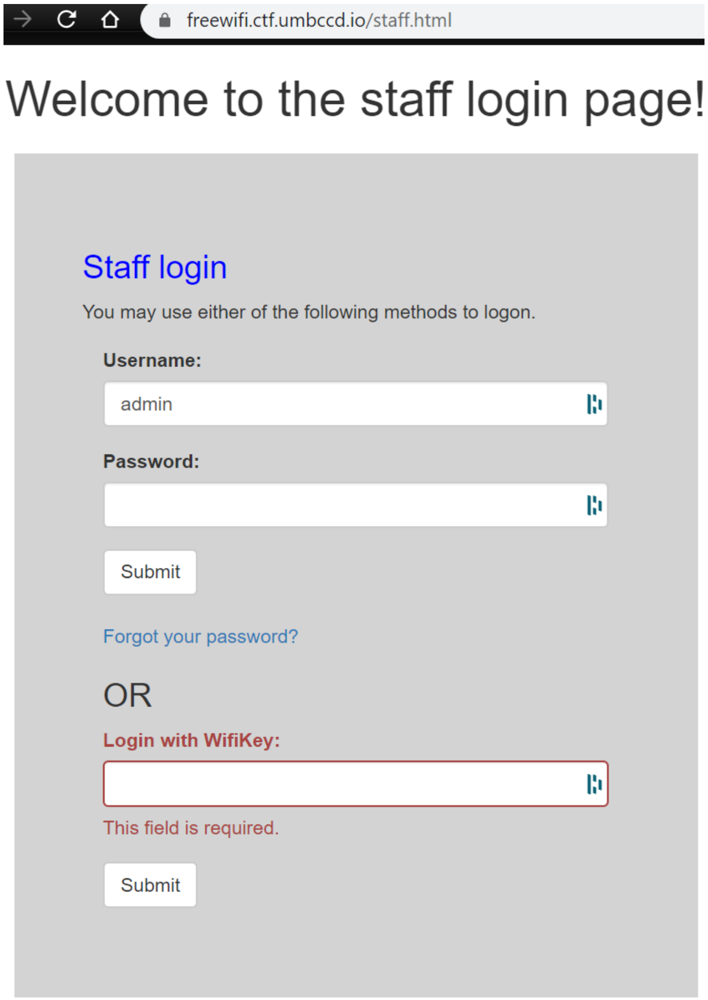
The link http://freewifi.ctf.umbccd.io/ leads to a broken webpage, but looking in the pcap, there is a staff login page: https://freewifi.ctf.umbccd.io/staff.html.



Sorry!

Guest login

Guest sign in portal is not yet implemented.



Welcome to the staff login page!

Staff login

You may use either of the following methods to logon.

Username: admin

Password:

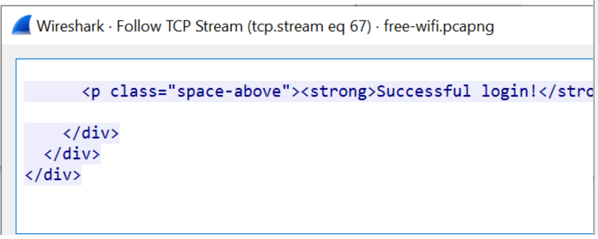
OR

Login with WifiKey:

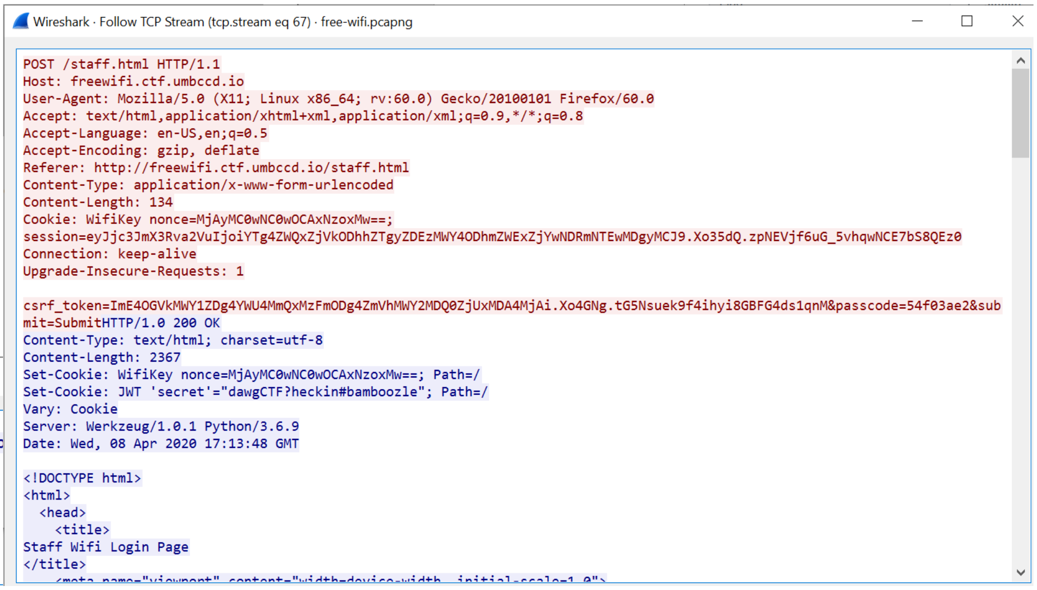
This field is required.

Invalid login.

Following a TCP stream of the unencrypted packets indicates there was a successful login…

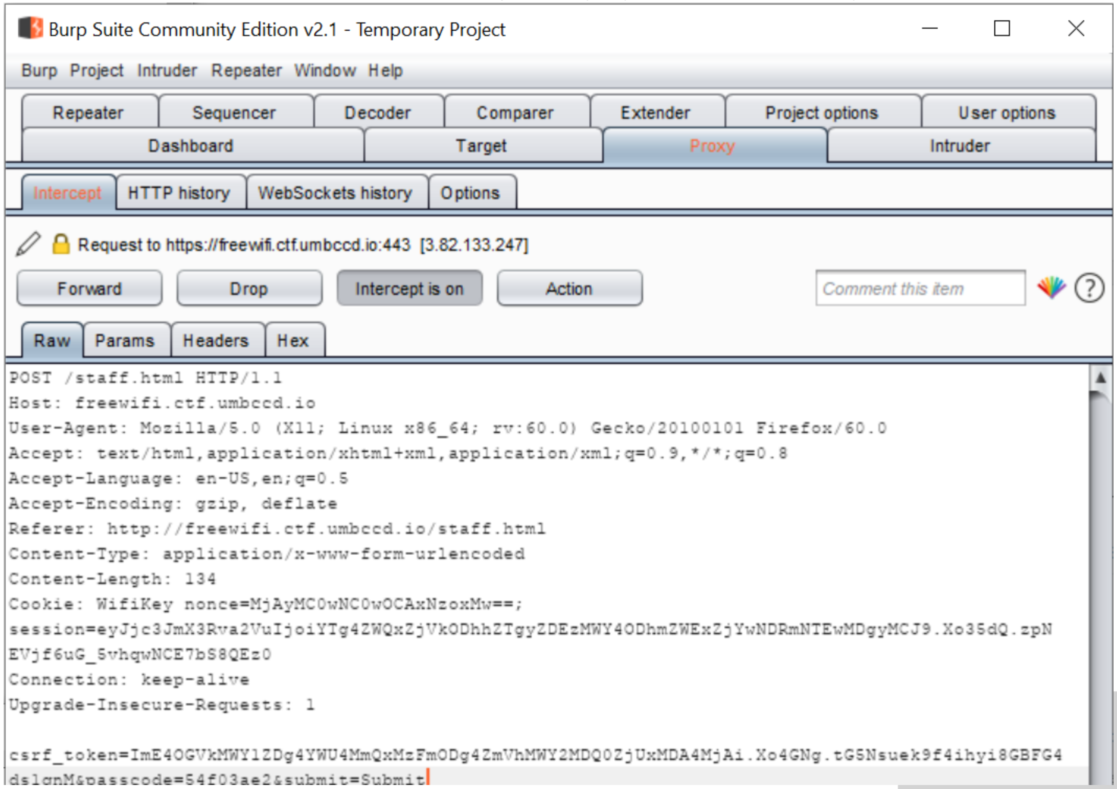

Wireshark · Follow TCP Stream (tcp.stream eq 67) · free-wifi.pcapng

```
    <p class="space-above"><strong>Successful login!</stro

   </div>
  </div>
</div>
```


Wireshark · Follow TCP Stream (tcp.stream eq 67) · free-wifi.pcapng

```
POST /staff.html HTTP/1.1
Host: freewifi.ctf.umbccd.io
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://freewifi.ctf.umbccd.io/staff.html
Content-Type: application/x-www-form-urlencoded
Content-Length: 134
Cookie: WifiKey nonce=MjAyMC0wNC0wOCAxNzoxMw==;
session=eyJjc3JmX3Rva2VuIjoiYTg4ZWQxZjVkODhhZTgyZDEzMWY4ODhmZWExZjYwNDRmNTEwMDgyMCJ9.Xo35dQ.zpNEVjf6uG_5vhqwNCE7bS8QEz0
Connection: keep-alive
Upgrade-Insecure-Requests: 1

csrf_token=ImE4OGVkMWY1ZDg4YWU4MmQxMzFmODg4ZmVhMWY2MDQ0ZjUxMDA4MjAi.Xo4GNg.tG5Nsuek9f4ihyi8GBFG4ds1qnM&passcode=54f03ae2&sub
mit=SubmitHTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 2367
Set-Cookie: WifiKey nonce=MjAyMC0wNC0wOCAxNzoxMw==; Path=/
Set-Cookie: JWT 'secret'="dawgCTF?heckin#bamboozle"; Path=/
Vary: Cookie
Server: Werkzeug/1.0.1 Python/3.6.9
Date: Wed, 08 Apr 2020 17:13:48 GMT

<!DOCTYPE html>
<html>
  <head>
    <title>
Staff Wifi Login Page
</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

…So, we should be able to hijack/replay the session in Burp.

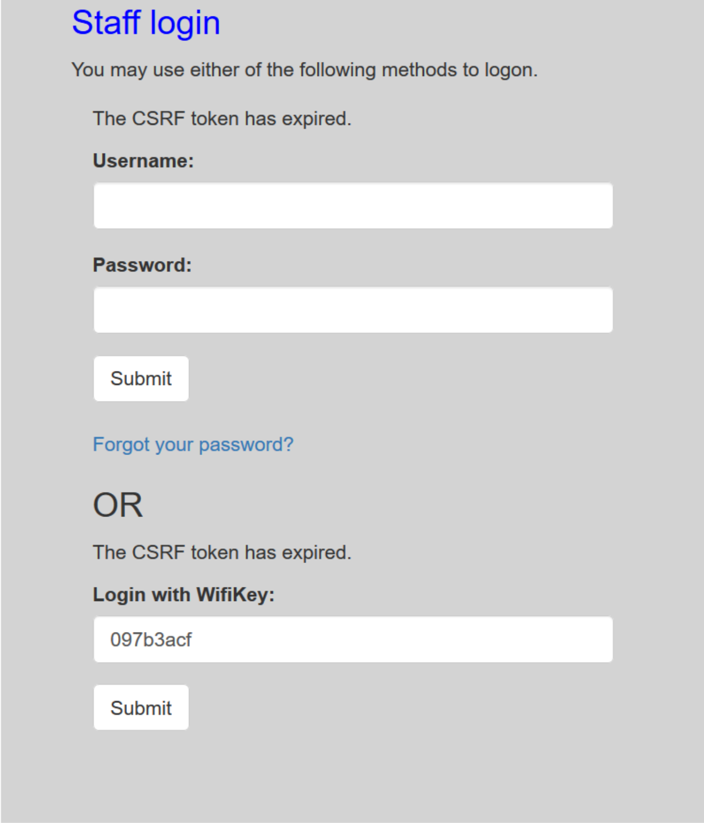

Burp Suite Community Edition v2.1 - Temporary Project
Burp Project Intruder Repeater Window Help
Repeater Sequencer Decoder Comparer Extender Project options User options
Dashboard Target Proxy Intruder
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action
Comment this item
Raw Params Headers Hex

Request to https://freewifi.ctf.umbccd.io:443 [3.82.133.247]

```
POST /staff.html HTTP/1.1
Host: freewifi.ctf.umbccd.io
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://freewifi.ctf.umbccd.io/staff.html
Content-Type: application/x-www-form-urlencoded
Content-Length: 134
Cookie: WifiKey nonce=MjAyMC0wNC0wOCAxNzoxMw==;
session=eyJjc3JmX3Rva2VuIjoiYTg4ZWQxZjVkODhhZTgyZDEzMWY4ODhmZWExZjYwNDRmNTEwMDgyMCJ9.Xo35dQ.zpN
EVjf6uG_5vhqwNCE7bS8QEz0
Connection: keep-alive
Upgrade-Insecure-Requests: 1

csrf_token=ImE4OGVkMWY1ZDg4YWU4MmQxMzFmODg4ZmVhMWY2MDQ0ZjUxMDA4MjAi.Xo4GNg.tG5Nsuek9f4ihyi8GBFG4
dslgnM&passcode=54f03ae2&submit=Submit
```

Once we pass the session, the flag appears at the bottom of the login page:

DawgCTF{w3lc0m3_t0_d@wgs3c_!nt3rn@t!0n@l}



Staff login

You may use either of the following methods to logon.

The CSRF token has expired.

Username:

Password:

Submit

Forgot your password?

OR

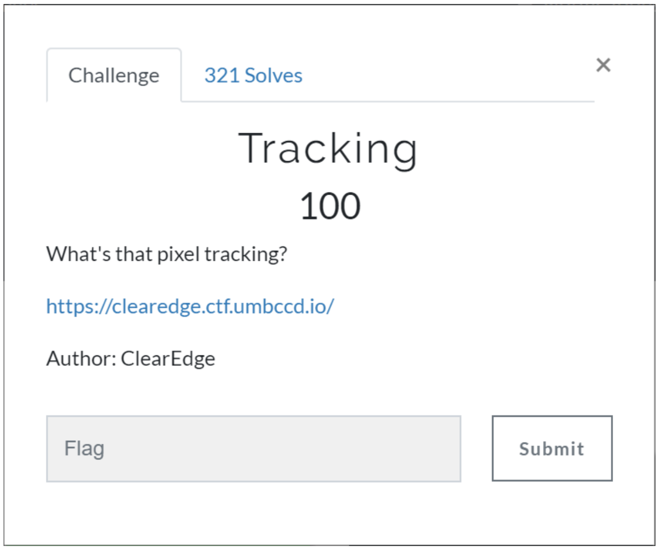
The CSRF token has expired.

Login with WifiKey: 097b3acf

Submit

DawgCTF{w3lc0m3_t0_d@wgs3c_!nt3rn@t!0n@l}

Tracking



Challenge | 321 Solves

Tracking

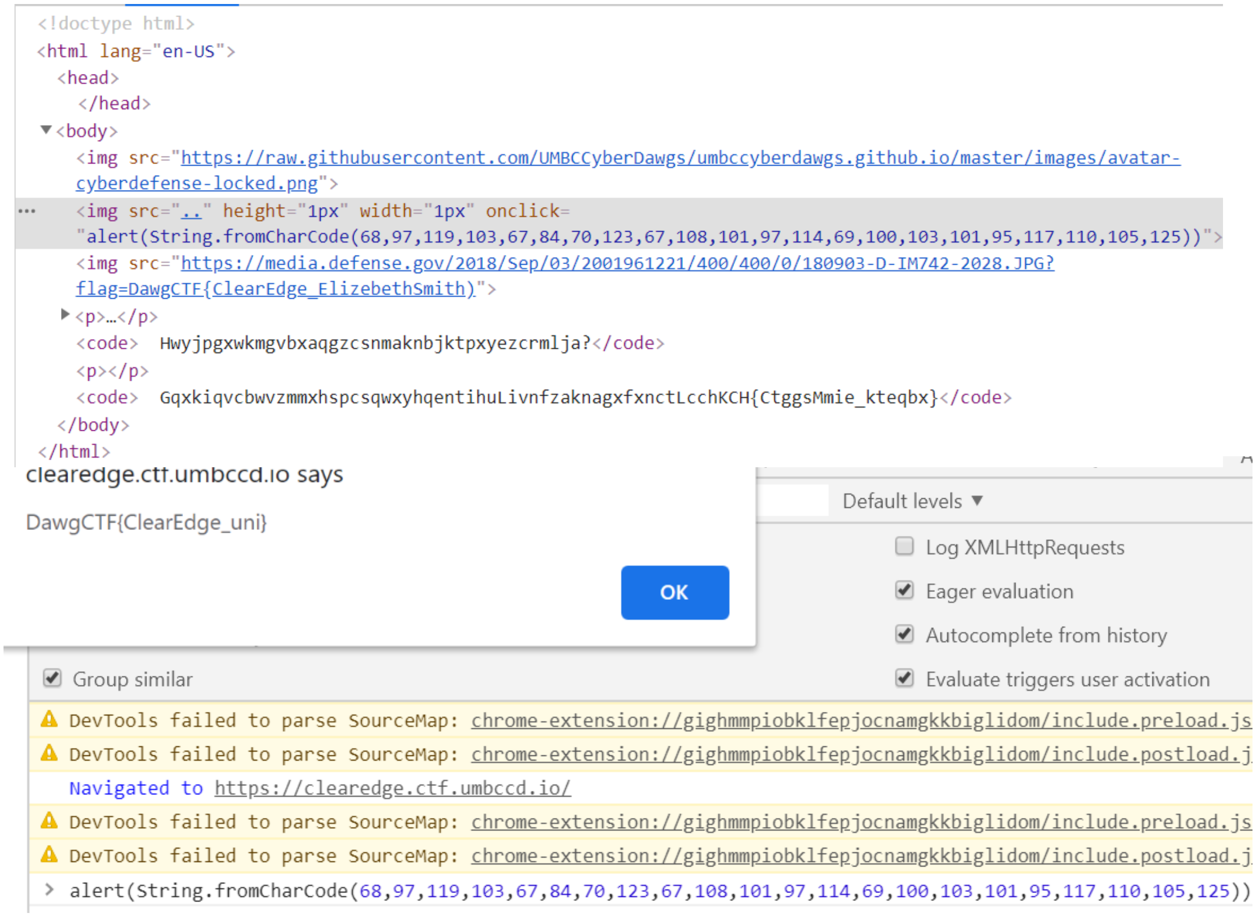
100

What's that pixel tracking?

https://clearedge.ctf.umbccd.io/

Author: ClearEdge

inspecting the source of the website reveals a single pixel with an on-click event. We could either increase the size to 100x100 and click it, or copy the alert line into console to reveal the flag.



```
<!doctype html>
<html lang="en-US">
  <head>
    </head>
 ▼<body>
    <img src="https://raw.githubusercontent.com/UMBCCyberDawgs/umbccyberdawgs.github.io/master/images/avatar-
    cyberdefense-locked.png">
…   <img src=".." height="1px" width="1px" onclick=
    "alert(String.fromCharCode(68,97,119,103,67,84,70,123,67,108,101,97,114,69,100,103,101,95,117,110,105,125))">
    <img src="https://media.defense.gov/2018/Sep/03/2001961221/400/400/0/180903-D-IM742-2028.JPG?
    flag=DawgCTF{ClearEdge_ElizebethSmith)">
   ▶<p>…</p>
    <code>  Hwyjpgxwkmgvbxaqgzcsnmaknbjktpxyezcrmlja?</code>
    <p></p>
    <code>  GqxkiqvcbwvzmmxhspcsqwxyhqentihuLivnfzaknagxfxnctLcchKCH{CtggsMmie_kteqbx}</code>
  </body>
</html>
```

clearedge.ctf.umbccd.io says

DawgCTF{ClearEdge_uni}

□ Log XMLHttpRequests

☑ Eager evaluation

☑ Autocomplete from history

☑ Group similar

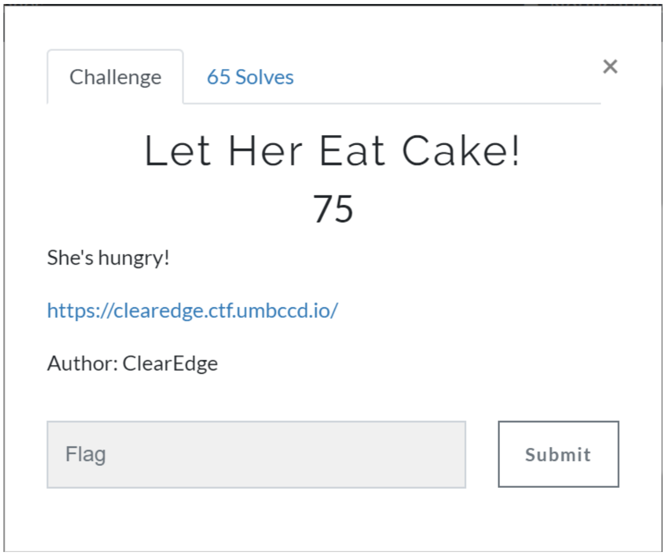
☑ Evaluate triggers user activation

```
⚠ DevTools failed to parse SourceMap: chrome-extension://gighmmpiobklfepjocnamgkkbiglidom/include.preload.js
⚠ DevTools failed to parse SourceMap: chrome-extension://gighmmpiobklfepjocnamgkkbiglidom/include.postload.j
  Navigated to https://clearedge.ctf.umbccd.io/
⚠ DevTools failed to parse SourceMap: chrome-extension://gighmmpiobklfepjocnamgkkbiglidom/include.preload.js
⚠ DevTools failed to parse SourceMap: chrome-extension://gighmmpiobklfepjocnamgkkbiglidom/include.postload.j
> alert(String.fromCharCode(68,97,119,103,67,84,70,123,67,108,101,97,114,69,100,103,101,95,117,110,105,125))
```

DawgCTF{ClearEdge_uni}

Misc

Let Her Eat Cake!



Challenge

65 Solves

×

Let Her Eat Cake!

75

She's hungry!

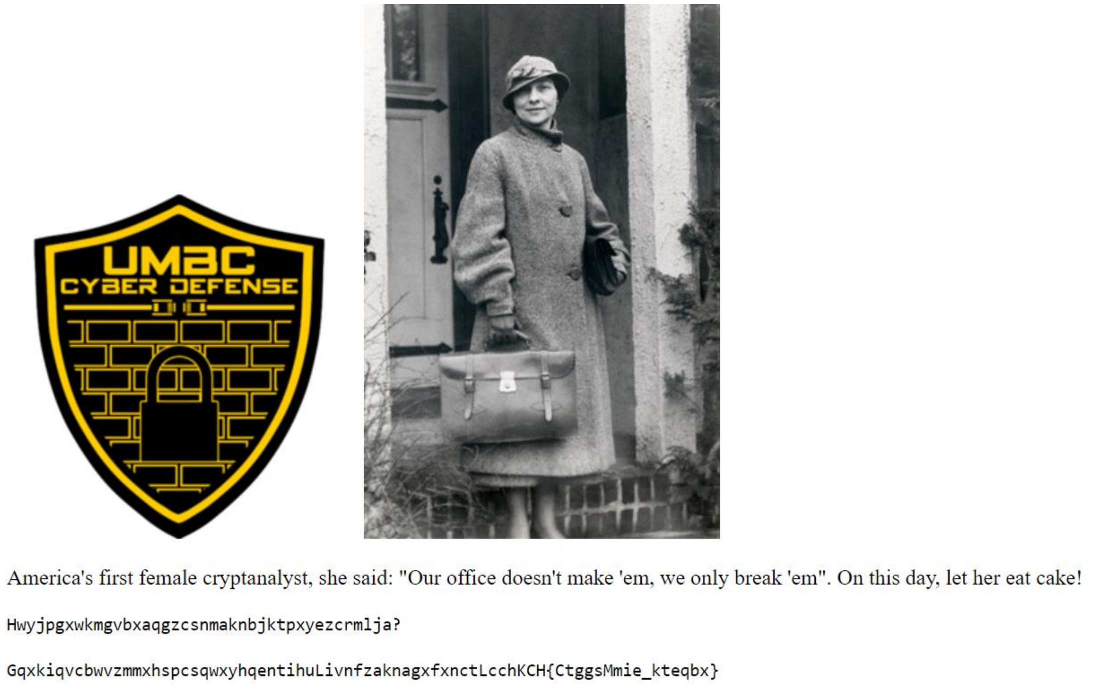
https://clearedge.ctf.umbccd.io/

Author: ClearEdge

Flag

Submit

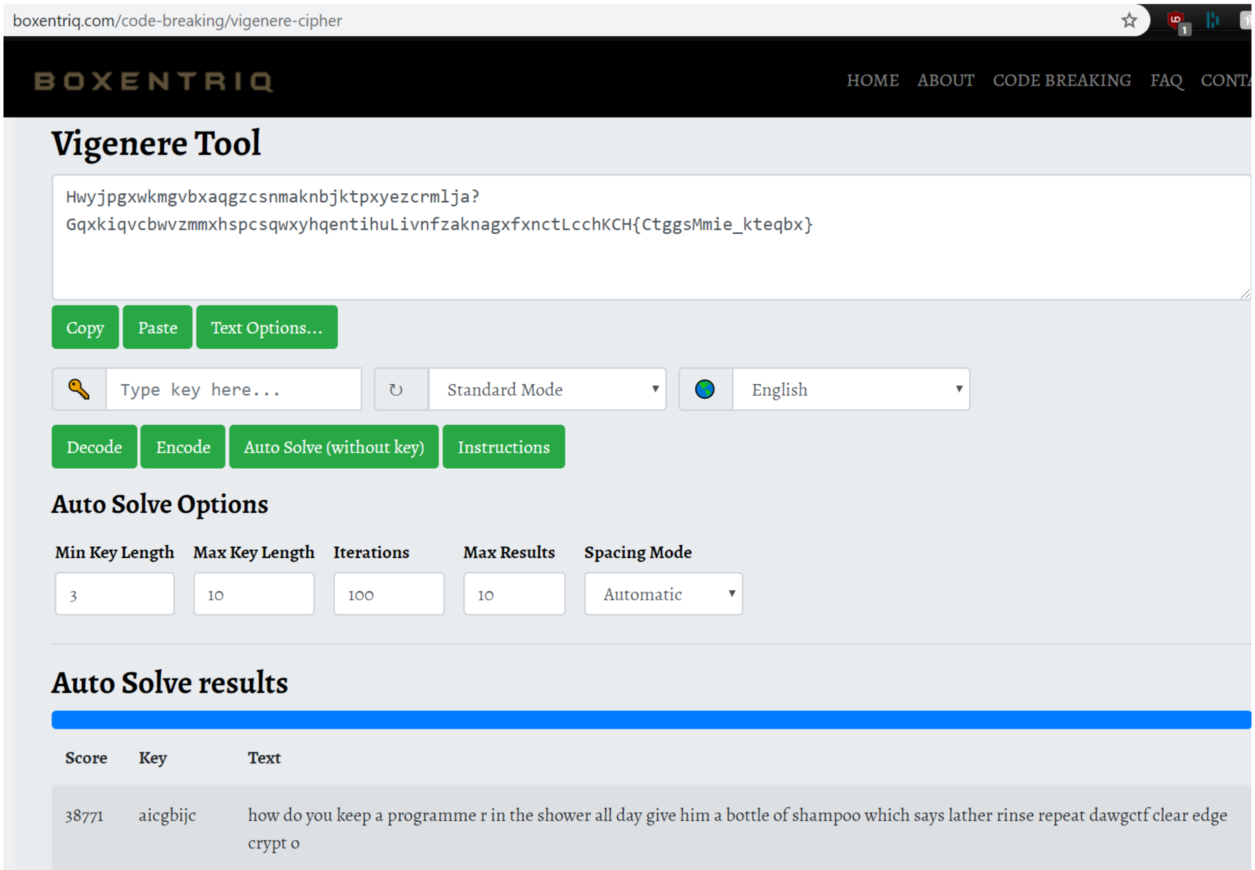
We are given a link to a site with a picture of Elizebeth Smith Friedman and some encoded text:



America's first female cryptanalyst, she said: "Our office doesn't make 'em, we only break 'em". On this day, let her eat cake!

```
Hwyjpgxwkmgvbxaqgzcsnmaknbjktpxyezcrmlja?

GqxkiqvcbwvzmmxhspcsqwxyhqentihuLivnfzaknagxfxnctLcchKCH{CtggsMmie_kteqbx}
```

A Vigenere cipher was used which can be brute forced - https://www.boxentriq.com/code-breaking/vigenere-cipher.



Vigenere Tool

```
Hwyjpgxwkmgvbxaqgzcsnmaknbjktpxyezcrmlja?
GqxkiqvcbwvzmmxhspcsqwxyhqentihuLivnfzaknagxfxnctLcchKCH{CtggsMmie_kteqbx}
```

Standard Mode ▾

English ▾

Auto Solve Options

Min Key Length	Max Key Length	Iterations	Max Results	Spacing Mode
3	10	100	10	Automatic ▾

Auto Solve results

Score	Key	Text
38771	aicgbijc	how do you keep a programme r in the shower all day give him a bottle of shampoo which says lather rinse repeat dawgctf clear edge crypt o

The Key is aicgbijc.

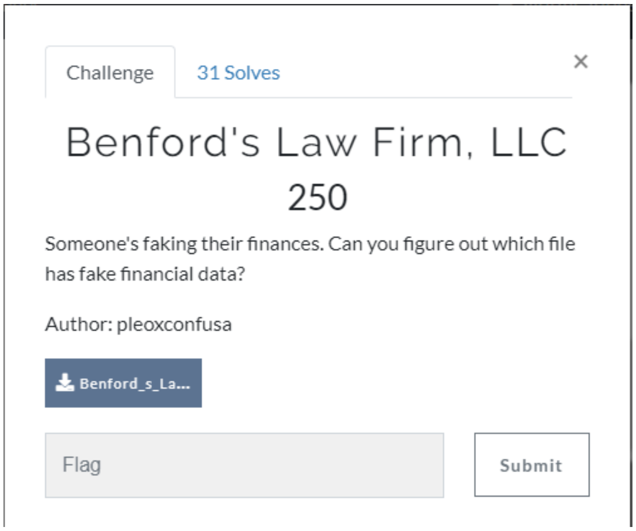
After inserting the ciphertext into CyberChef along with the key, we are given the flag:

Howdoyoukeepaprogrammerintheshowerallday?

GivehimabottleofshampoowhichsaysLatherrinserepeat**DawgCTF{ClearEdge_crypto}**

Forensics

Benford's Law Firm, LLC



Challenge | 31 Solves

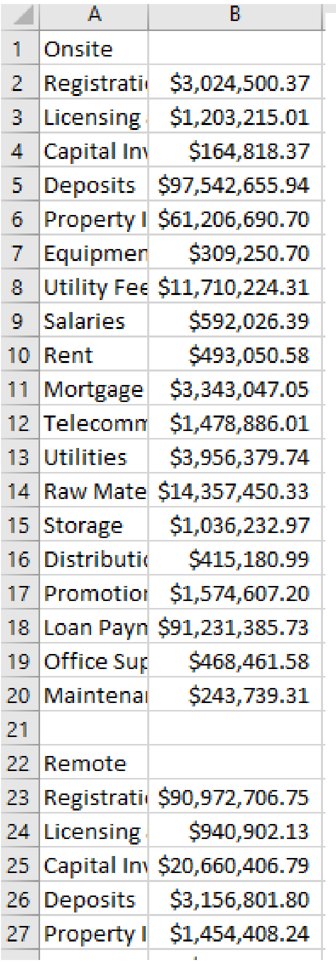
Benford's Law Firm, LLC

250

Someone's faking their finances. Can you figure out which file has fake financial data?

Author: pleoxconfusa

We are provided with a .zip file containing 1,000 csv files, all with financial data.



	A	B
1	Onsite	
2	Registrati	$3,024,500.37
3	Licensing	$1,203,215.01
4	Capital Inv	$164,818.37
5	Deposits	$97,542,655.94
6	Property I	$61,206,690.70
7	Equipmen	$309,250.70
8	Utility Fee	$11,710,224.31
9	Salaries	$592,026.39
10	Rent	$493,050.58
11	Mortgage	$3,343,047.05
12	Telecomm	$1,478,886.01
13	Utilities	$3,956,379.74
14	Raw Mate	$14,357,450.33
15	Storage	$1,036,232.97
16	Distributi	$415,180.99
17	Promotio	$1,574,607.20
18	Loan Payn	$91,231,385.73
19	Office Sup	$468,461.58
20	Maintenan	$243,739.31
21		
22	Remote	
23	Registrati	$90,972,706.75
24	Licensing	$940,902.13
25	Capital Inv	$20,660,406.79
26	Deposits	$3,156,801.80
27	Property I	$1,454,408.24

Benford's Law maintains that the numeral 1 will be the leading digit in a genuine data set of numbers 30.1% of the time. By checking each spreadsheet to see if there is a large anomaly of numbers that start with 1, we can find the correct file.

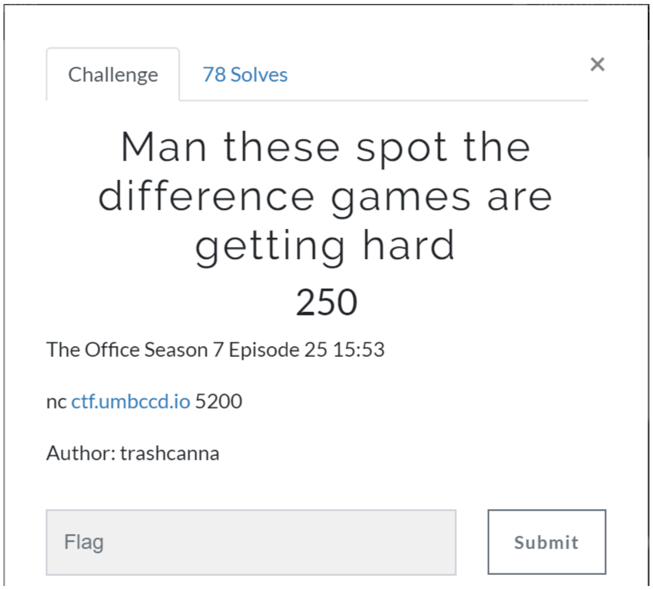
PowerShell implementation:

```
$csvFiles = gci "$PSScriptRoot/Benford_s_Law_Firm_LLC"

# loop through csv files
foreach ($csv in $csvFiles.Name) {
    # get dollar value of from csv
    $data = (gc "$PSScriptRoot/Benford_s_Law_Firm_LLC/$csv").split(',') |
        where { $_ -match '\d' }
    # loop through dollar values and count how many starts with a 1
    $count1 = 0
    $data | foreach {
        if ($_.StartsWith('$1')) {
            $count1++
        }
    }
    # check if count of 1s does not follow Benford's Law
    if ($count1/$data.count -gt .60 -or $count1/$data.count -lt .10) {
        "$csv 1 anomaly: $($count1/$data.count)"
    }

}
```

PS> **DawgCTF{L3g@lly_D1s7ribu73d_St@t1st1c5_641}**.csv 1 anomaly: 0.0657894736842105

Coding

Spot the Difference



Man these spot the difference games are getting hard

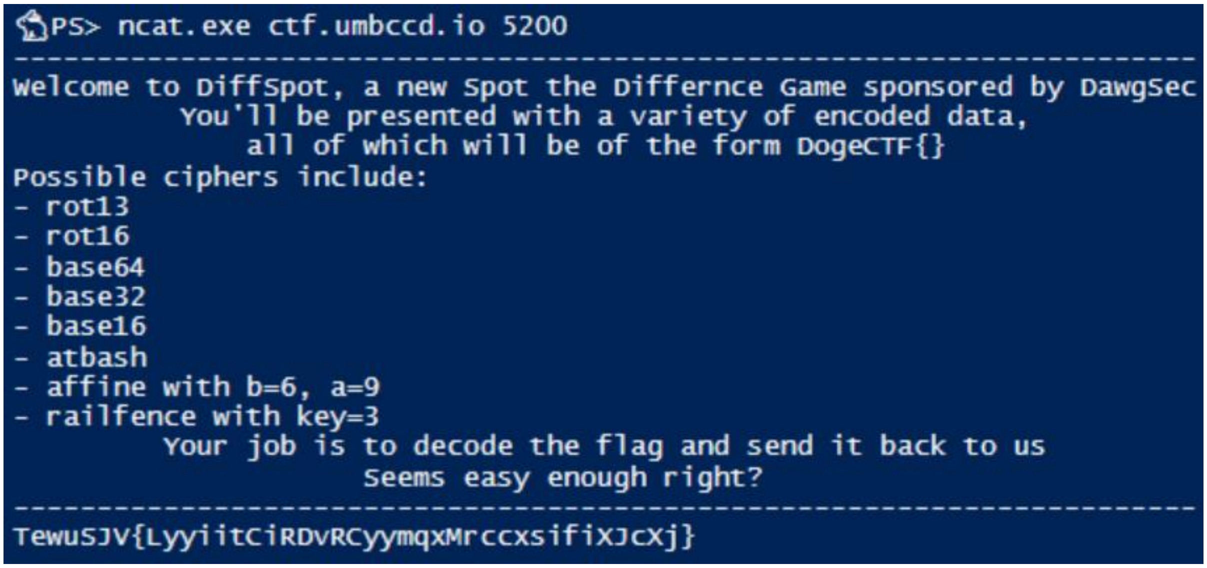
250

The Office Season 7 Episode 25 15:53

nc ctf.umbccd.io 5200

Author: trashcanna

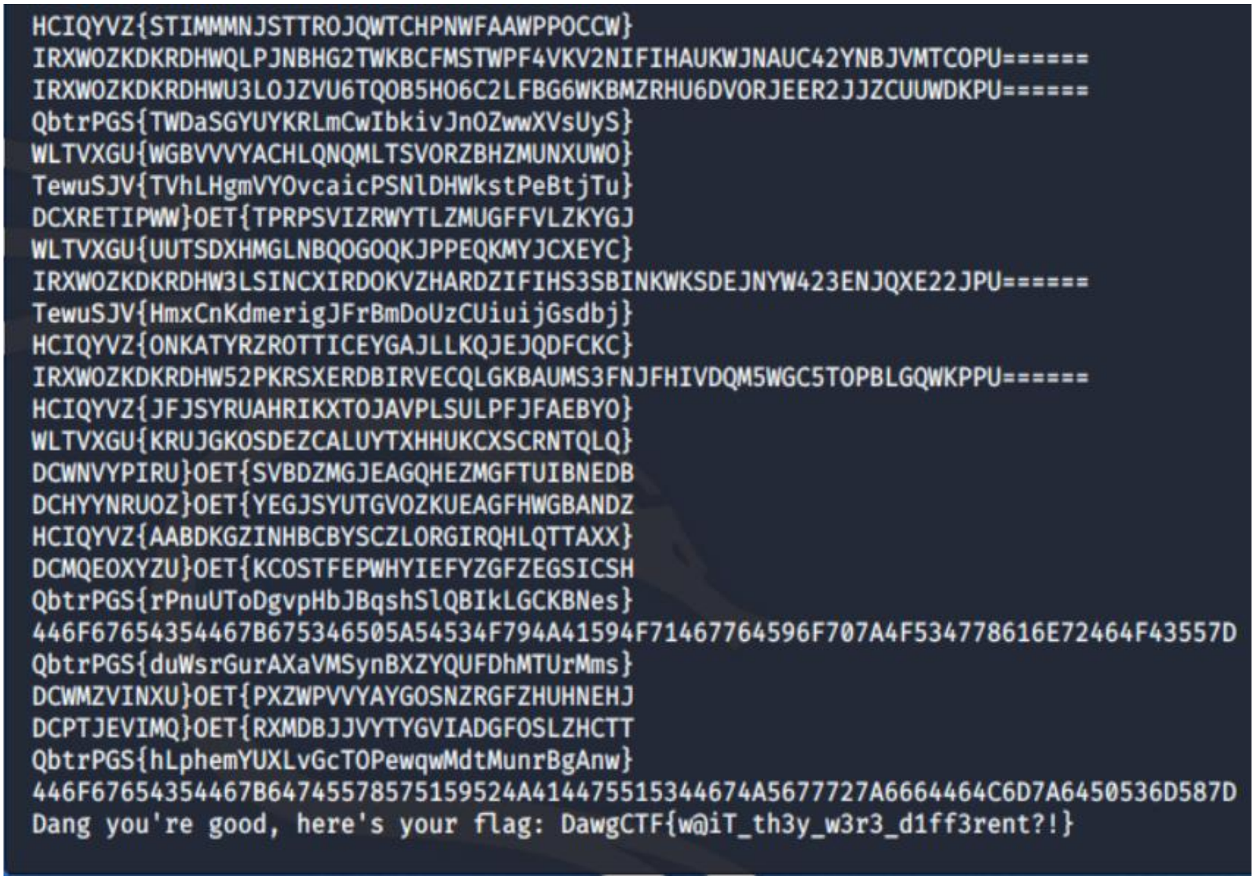
The below prompt provides possible ciphers, along with the cipher text. Since the flag is always in the same format, DawgCTF{Some-Text}, it is easy to determine which cipher was used since there will be a common char set.



```
PS> ncat.exe ctf.umbccd.io 5200
-----------------------------------------------------------------------
Welcome to DiffSpot, a new Spot the Differnce Game sponsored by DawgSec
          You'll be presented with a variety of encoded data,
              all of which will be of the form DogeCTF{}
Possible ciphers include:
- rot13
- rot16
- base64
- base32
- base16
- atbash
- affine with b=6, a=9
- railfence with key=3
         Your job is to decode the flag and send it back to us
                     Seems easy enough right?
-----------------------------------------------------------------------
TewuSJV{LyyiitCiRDvRCyymqxMrccxsifiXJcXj}
```

After running the solvable implementation found at the below link, we are provided with a flag.

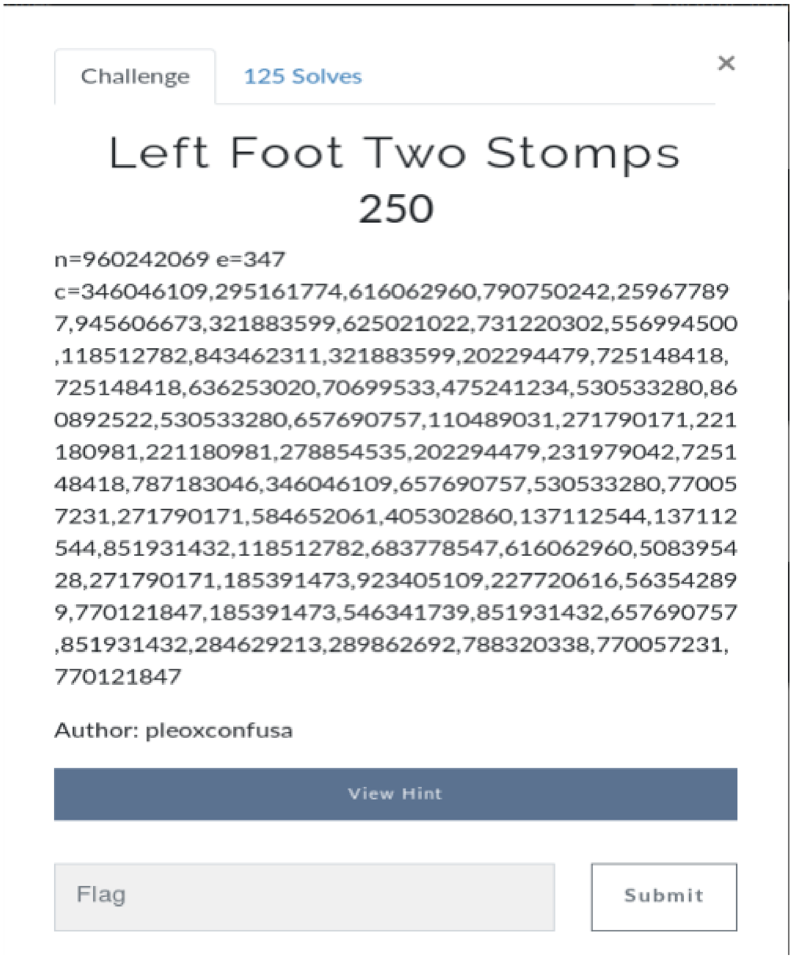
https://github.com/eWorkaholics/DawgCTF-2020/blob/master/spot-the-difference.py



```
HCIQYVZ{STIMMMNJSTTROJQWTCHPNWFAAWPPOCCW}
IRXWOZKDKRDHWQLPJNBHG2TWKBCFMSTWPF4VKV2NIFIHAUKWJNAUC42YNBJVMTCOPU======
IRXWOZKDKRDHWU3LOJZVU6TQOB5HO6C2LFBG6WKBMZRHU6DVORJEER2JJZCUUWDKPU======
QbtrPGS{TWDaSGYUYKRLmCwIbkivJnOZwwXVsUyS}
WLTVXGU{WGBVVVYACHLQNQMLTSVORZBHZMUNXUWO}
TewuSJV{TVhLHgmVYOvcaicPSNlDHWkstPeBtjTu}
DCXRETIPWW}OET{TPRPSVIZRWYTLZMUGFFVLZKYGJ
WLTVXGU{UUTSDXHMGLNBQOGOQKJPPEQKMYJCXEYC}
IRXWOZKDKRDHW3LSINCXIRDOKVZHARDZIFIHS3SBINKWKSDEJNYW423ENJQXE22JPU======
TewuSJV{HmxCnKdmerigJFrBmDoUzCUiuijGsdbj}
HCIQYVZ{ONKATYRZROTTICEYGAJLLKQJEJQDFCKC}
IRXWOZKDKRDHW52PKRSXERDBIRVECQLGKBAUMS3FNJFHIVDQM5WGC5TOPBLGQWKPPU======
HCIQYVZ{JFJSYRUAHRIKXTOJAVPLSULPFJFAEBYO}
WLTVXGU{KRUJGKOSDEZCALUYTXHHUKCXSCRNTQLQ}
DCWNVYPIRU}OET{SVBDZMGJEAGQHEZMGFTUIBNEDB
DCHYYNRUOZ}OET{YEGJSYUTGVOZKUEAGFHWGBANDZ
HCIQYVZ{AABDKGZINHBCBYSCZLORGIRQHLQTTAXX}
DCMQEOXYZU}OET{KCOSTFEPWHYIEFYZGFZEGSICSH
QbtrPGS{rPnuUToDgvpHbJBqshSlQBIkLGCKBNes}
446F67654354467B675346505A54534F794A41594F71467764596F707A4F534778616E72464F43557D
QbtrPGS{duWsrGurAXaVMSynBXZYQUFDhMTUrMms}
DCWMZVINXU}OET{PXZWPVVYAYGOSNZRGFZHUHNEHJ
DCPTJEVIMQ}OET{RXMDBJJVYTYGVIADGFOSLZHCTT
QbtrPGS{hLphemYUXLvGcTOPewqwMdtMunrBgAnw}
446F67654354467B64745578575159524A414475515344674A5677727A6664464C6D7A6450536D587D
Dang you're good, here's your flag: DawgCTF{w@iT_th3y_w3r3_d1ff3rent?!}
```

Crypto

Left Foot Two Stomps

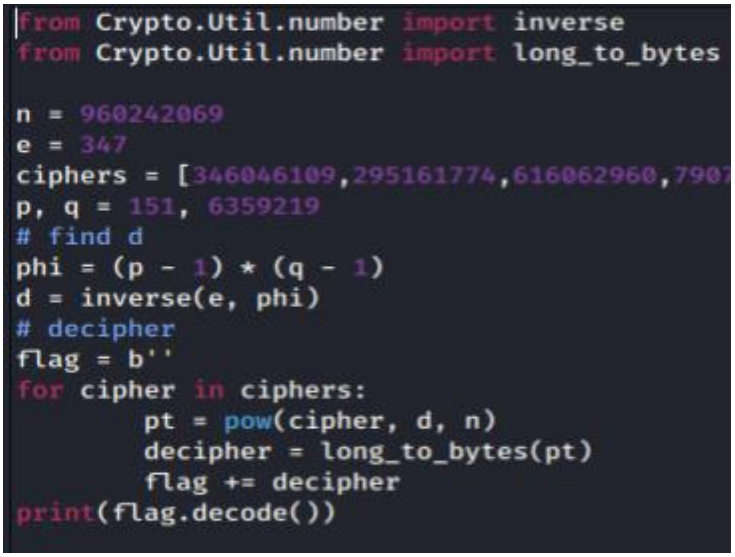


Left Foot Two Stomps
250

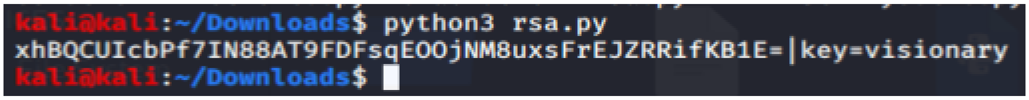
n=960242069 e=347
c=346046109,295161774,616062960,790750242,259677897,945606673,321883599,625021022,731220302,556994500,118512782,843462311,321883599,202294479,725148418,725148418,636253020,70699533,475241234,530533280,860892522,530533280,657690757,110489031,271790171,221180981,221180981,278854535,202294479,231979042,725148418,787183046,346046109,657690757,530533280,770057231,271790171,584652061,405302860,137112544,137112544,851931432,118512782,683778547,616062960,508395428,271790171,185391473,923405109,227720616,563542899,770121847,185391473,546341739,851931432,657690757,851931432,284629213,289862692,788320338,770057231,770121847

Author: pleoxconfusa

We are provided with a small n for this rsa challenge. P and Q can be found using factordb. Then just need to output plaintext with this script:

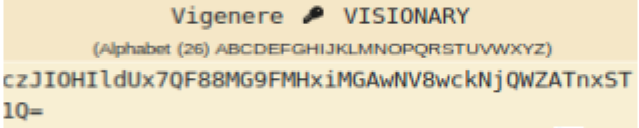


```
from Crypto.Util.number import inverse
from Crypto.Util.number import long_to_bytes

n = 960242069
e = 347
ciphers = [346046109,295161774,616062960,7907
p, q = 151, 6359219
# find d
phi = (p - 1) * (q - 1)
d = inverse(e, phi)
# decipher
flag = b''
for cipher in ciphers:
        pt = pow(cipher, d, n)
        decipher = long_to_bytes(pt)
        flag += decipher
print(flag.decode())
```



```
kali@kali:~/Downloads$ python3 rsa.py
xhBQCUIcbPf7IN88AT9FDFsqEOOjNM8uxsFrEJZRRifKB1E=|key=visionary
kali@kali:~/Downloads$
```

The Base64 needs to be put through a Vinegere decoder:

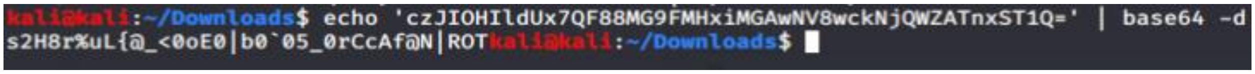


Vigenere 🔑 VISIONARY

(Alphabet (26) ABCDEFGHIJKLMNOPQRSTUVWXYZ)

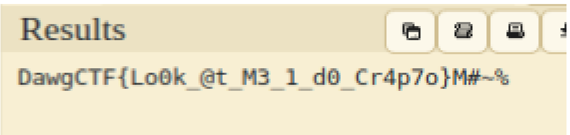
```
czJIOHIldUx7QF88MG9FMHxiMGAwNV8wckNjQWZATnxST
1Q=
```

And decode that for more cipher text:



```
kali@kali:~/Downloads$ echo 'czJIOHIldUx7QF88MG9FMHxiMGAwNV8wckNjQWZATnxST1Q=' | base64 -d
s2H8r%uL{@_<0oE0|b0`05_0rCcAf@N|ROTkali@kali:~/Downloads$
```

Finally, ROT47 this and reveal the key:



Results

```
DawgCTF{Lo0k_@t_M3_1_d0_Cr4p7o}M#~%
```