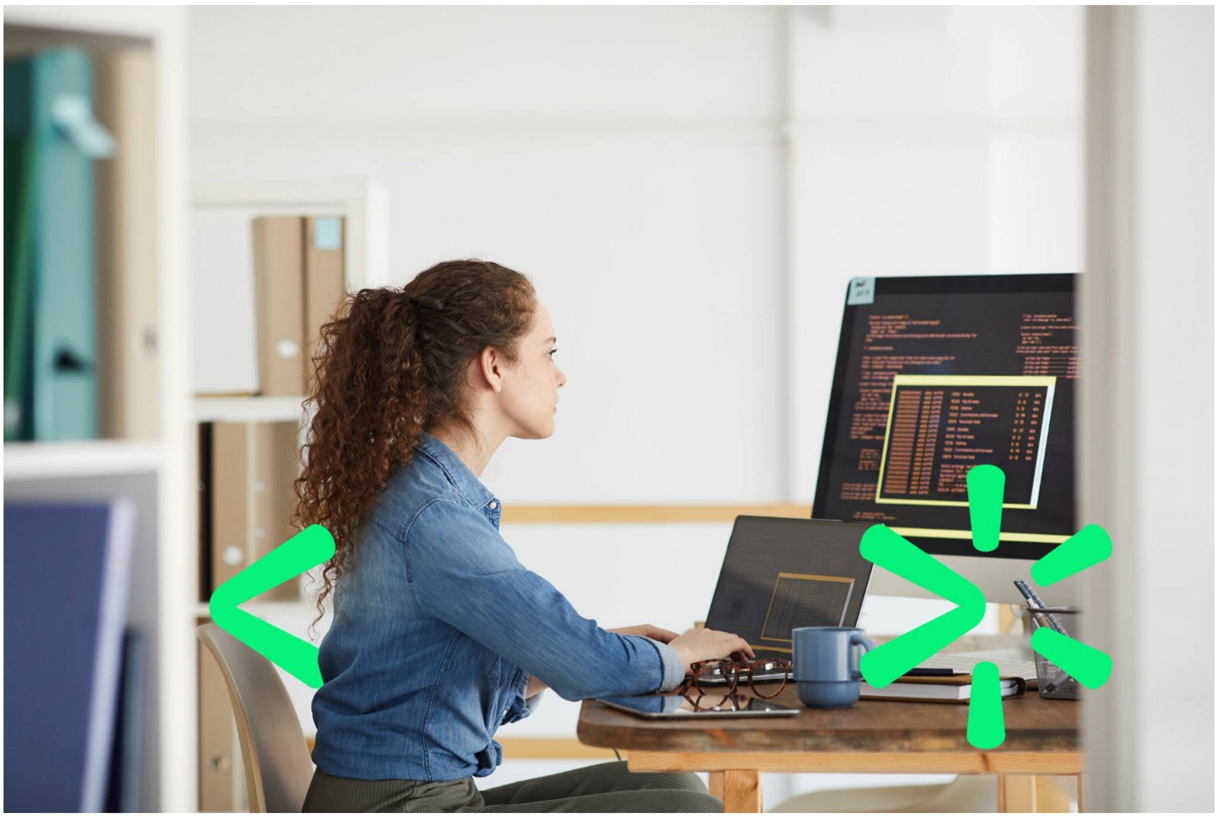
Published in Walmart Global Tech Blog

Sugar Ransomware, a new RaaS

By: Joshua Platt, Jonathan Mccay and Jason Reaves

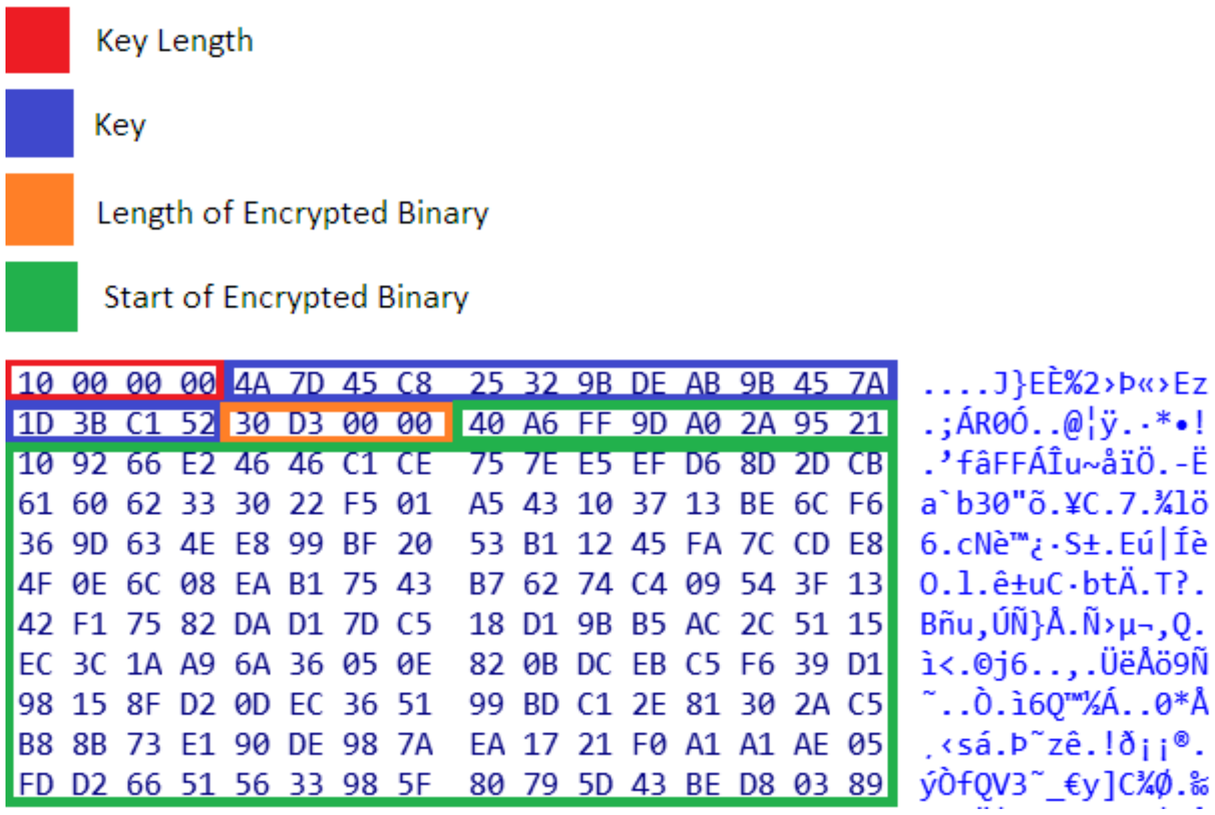


An actor recently has been starting up a RaaS solution that appears to primarily focus on individual computers instead of entire enterprises but is also reusing objects from other ransomware families. Not a lot has been discussed about this ransomware but we did find a tweet mentioning one of the samples[3] during our research.

Crypter

We will go over the crypter being used because it has code reuse from the ransomware itself which makes it significantly more interesting than your typical crypter. The crypter has what initially looks like RC4 encryption leading to APLib decompression but as we dug in it turns out to be a modified version of RC4.

The encoded data can be seen with the key prepended to the data:



🟥 Key Length

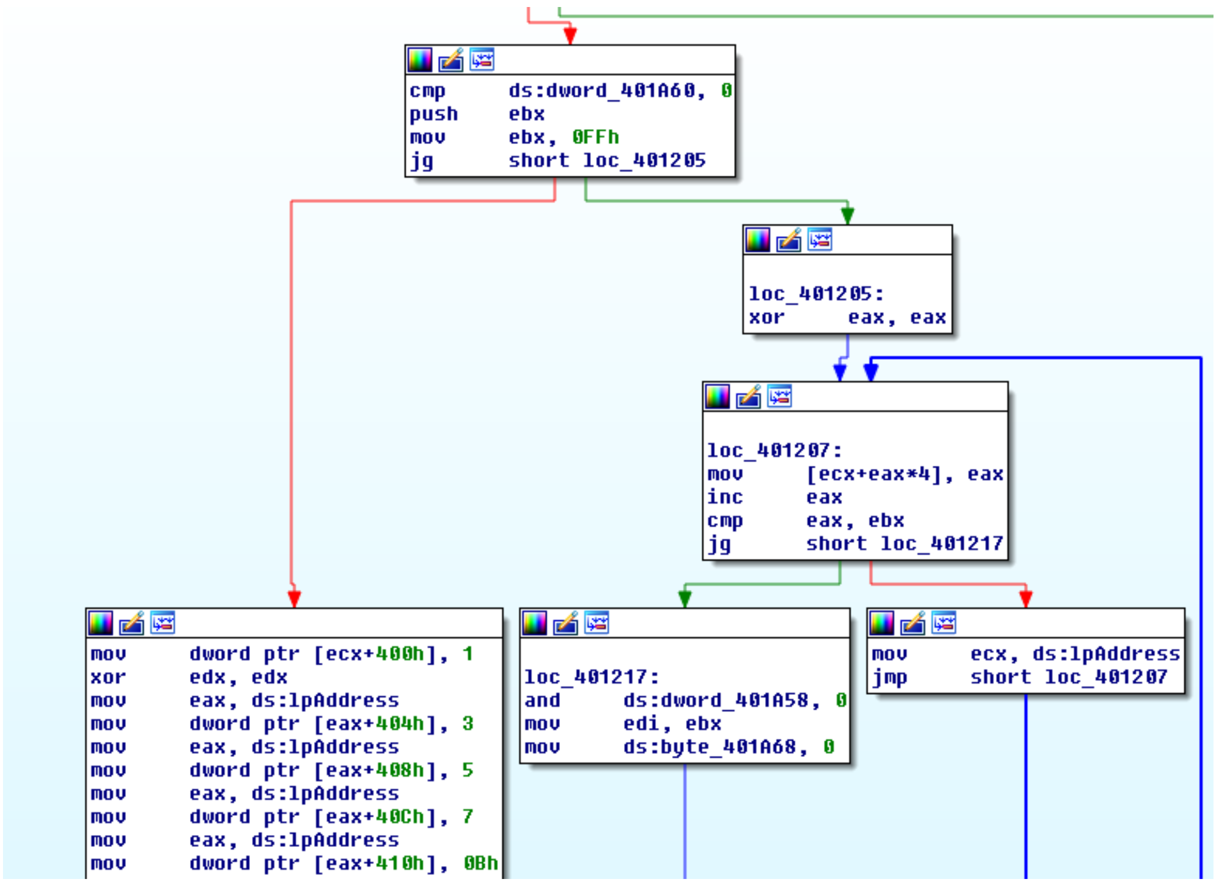
🟦 Key

🟧 Length of Encrypted Binary

🟩 Start of Encrypted Binary

```
10 00 00 00 4A 7D 45 C8  25 32 9B DE AB 9B 45 7A  ....J}EÈ%2›Þ«›Ez
1D 3B C1 52 30 D3 00 00  40 A6 FF 9D A0 2A 95 21  .;ÁR0Ó..@¦ÿ. *•!
10 92 66 E2 46 46 C1 CE  75 7E E5 EF D6 8D 2D CB  .’fâFFÁÎu~åïÖ.-Ë
61 60 62 33 30 22 F5 01  A5 43 10 37 13 BE 6C F6  a`b30"õ.¥C.7.¾lö
36 9D 63 4E E8 99 BF 20  53 B1 12 45 FA 7C CD E8  6.cNè™¿ S±.Eú|Íè
4F 0E 6C 08 EA B1 75 43  B7 62 74 C4 09 54 3F 13  O.l.ê±uC·btÄ.T?.
42 F1 75 82 DA D1 7D C5  18 D1 9B B5 AC 2C 51 15  Bñu‚ÚÑ}Å.Ñ›µ¬,Q.
EC 3C 1A A9 6A 36 05 0E  82 0B DC EB C5 F6 39 D1  ì<.©j6..‚.ÜëÅö9Ñ
98 15 8F D2 0D EC 36 51  99 BD C1 2E 81 30 2A C5  ˜..Ò.ì6Q™½Á..0*Å
B8 8B 73 E1 90 DE 98 7A  EA 17 21 F0 A1 A1 AE 05  ¸‹sá.Þ˜zê.!ð¡¡®.
FD D2 66 51 56 33 98 5F  80 79 5D 43 BE D8 03 89  ýÒfQV3˜_€y]C¾Ø.‰
```

As we mentioned above the encryption algorithm first looks like RC4, it sets up the SBOX:



```
cmp     ds:dword_401A60, 0
push    ebx
mov     ebx, 0FFh
jg      short loc_401205
```

```
loc_401205:
xor     eax, eax
```

```
loc_401207:
mov     [ecx+eax*4], eax
inc     eax
cmp     eax, ebx
jg      short loc_401217
```

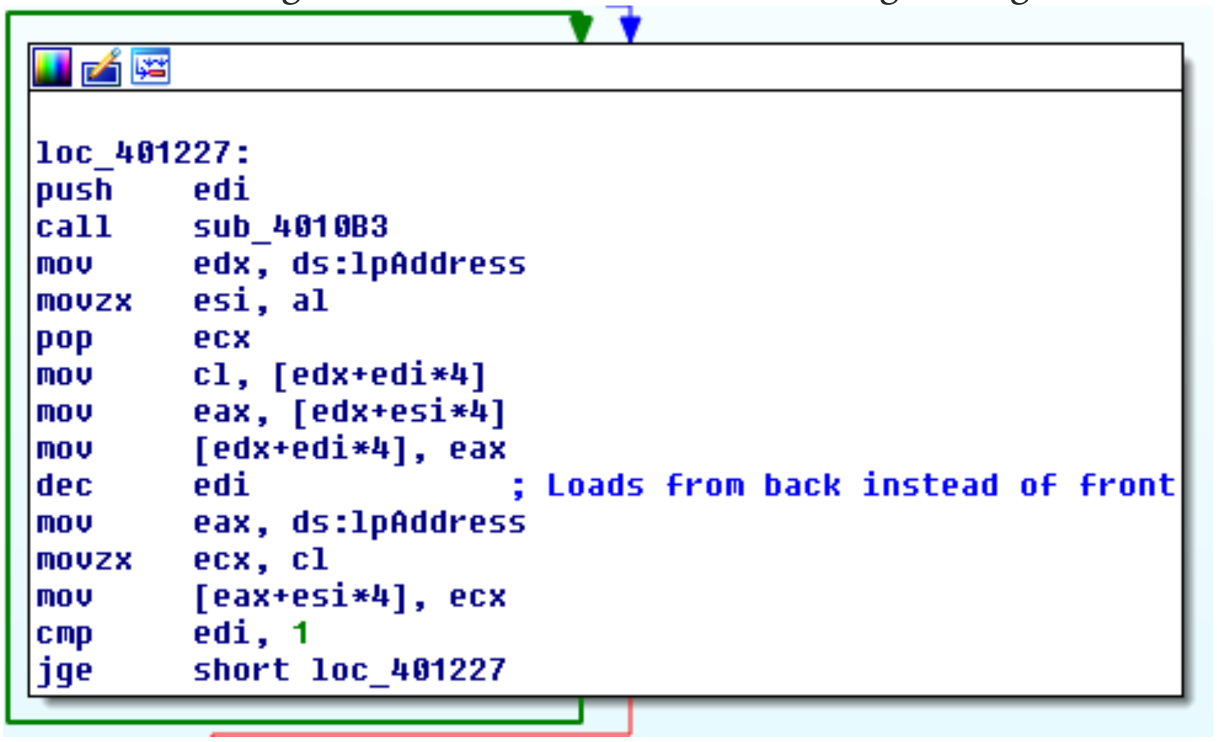
```
mov     dword ptr [ecx+400h], 1
xor     edx, edx
mov     eax, ds:lpAddress
mov     dword ptr [eax+404h], 3
mov     eax, ds:lpAddress
mov     dword ptr [eax+408h], 5
mov     eax, ds:lpAddress
mov     dword ptr [eax+40Ch], 7
mov     eax, ds:lpAddress
mov     dword ptr [eax+410h], 0Bh
```

```
loc_401217:
and     ds:dword_401A58, 0
mov     edi, ebx
mov     ds:byte_401A68, 0
```

```
mov     ecx, ds:lpAddress
jmp     short loc_401207
```

SBOX initialization

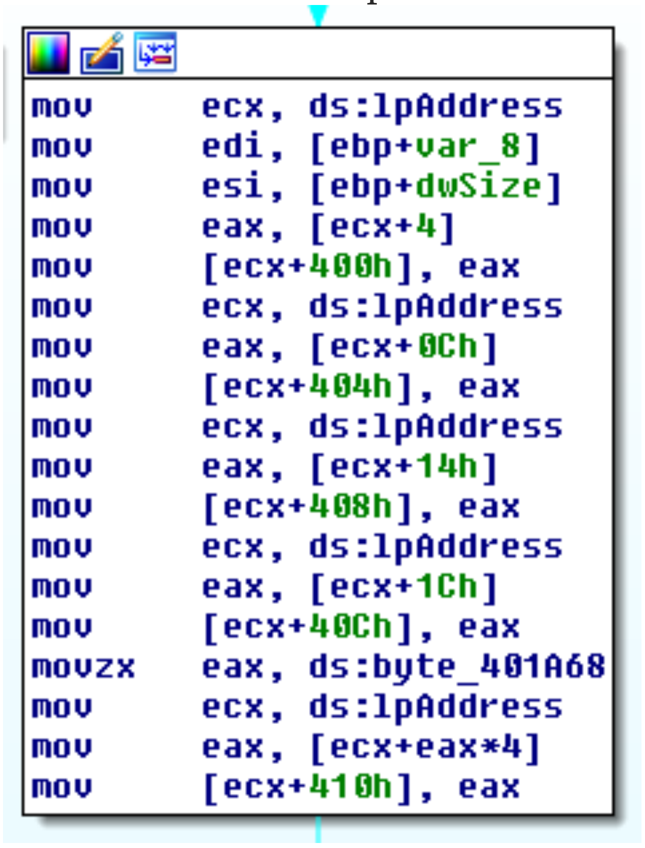
However starting with the KSA block is where things change:



```
loc_401227:
push    edi
call    sub_4010B3
mov     edx, ds:lpAddress
movzx   esi, al
pop     ecx
mov     cl, [edx+edi*4]
mov     eax, [edx+esi*4]
mov     [edx+edi*4], eax
dec     edi                 ; Loads from back instead of front
mov     eax, ds:lpAddress
movzx   ecx, cl
mov     [eax+esi*4], ecx
cmp     edi, 1
jge     short loc_401227
```

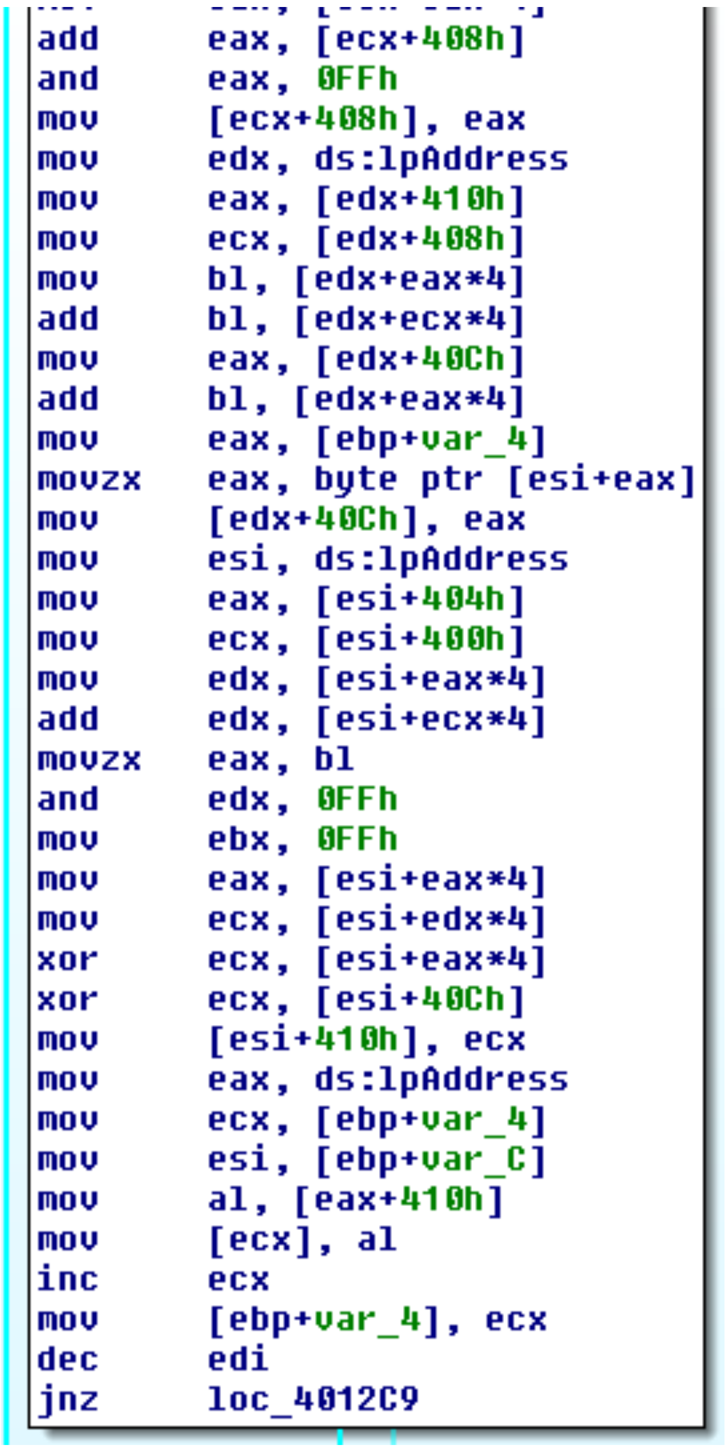
Custom KSA

The algorithm cycles through the SBOX during KSA from back to front, it also leverages a simple bitwise OR loop to build a value which is used to bitwise AND against the working value from the key, if the value is greater than or equal to the current SBOX iteration then it will continue to the next value in the key. Afterwards it begins a custom version of PRGA that involves some extra shuffling based on four values from the post KSA SBOX.



```
mov     ecx, ds:lpAddress
mov     edi, [ebp+var_8]
mov     esi, [ebp+dwSize]
mov     eax, [ecx+4]
mov     [ecx+400h], eax
mov     ecx, ds:lpAddress
mov     eax, [ecx+0Ch]
mov     [ecx+404h], eax
mov     ecx, ds:lpAddress
mov     eax, [ecx+14h]
mov     [ecx+408h], eax
mov     ecx, ds:lpAddress
mov     eax, [ecx+1Ch]
mov     [ecx+40Ch], eax
movzx   eax, ds:byte_401A68
mov     ecx, ds:lpAddress
mov     eax, [ecx+eax*4]
mov     [ecx+410h], eax
```

After custom KSA



```
add     eax, [ecx+408h]
and     eax, 0FFh
mov     [ecx+408h], eax
mov     edx, ds:lpAddress
mov     eax, [edx+410h]
mov     ecx, [edx+408h]
mov     bl, [edx+eax*4]
add     bl, [edx+ecx*4]
mov     eax, [edx+40Ch]
add     bl, [edx+eax*4]
mov     eax, [ebp+var_4]
movzx   eax, byte ptr [esi+eax]
mov     [edx+40Ch], eax
mov     esi, ds:lpAddress
mov     eax, [esi+404h]
mov     ecx, [esi+400h]
mov     edx, [esi+eax*4]
add     edx, [esi+ecx*4]
movzx   eax, bl
and     edx, 0FFh
mov     ebx, 0FFh
mov     eax, [esi+eax*4]
mov     ecx, [esi+edx*4]
xor     ecx, [esi+eax*4]
xor     ecx, [esi+40Ch]
mov     [esi+410h], ecx
mov     eax, ds:lpAddress
mov     ecx, [ebp+var_4]
mov     esi, [ebp+var_C]
mov     al, [eax+410h]
mov     [ecx], al
inc     ecx
mov     [ebp+var_4], ecx
dec     edi
jnz     loc_4012C9
```

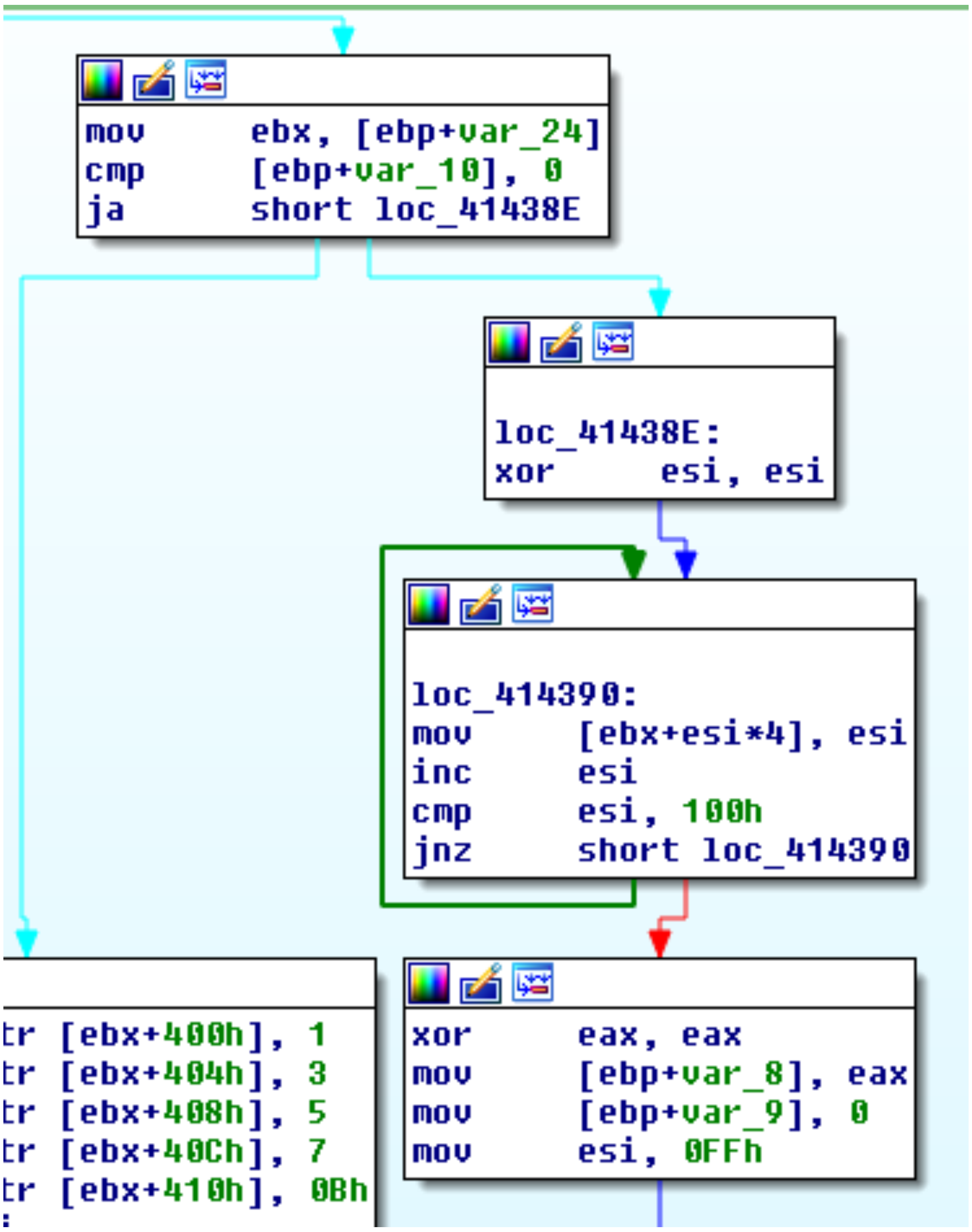
Custom PRGA

Unpacking code:

```
import yara
from pefile import PE
from struct import unpack
from aplib import Decompress
from io import BytesIO
from sys import argvdef main():
    filepath = argv[1]
    readbin = open(filepath, 'rb').read()

    rule = yara.compile(
        source='rule sugar_RaaS_crypter { strings: '
        '$57B = { C7 [1] 08 04 00 00 05 00 00 00 A1 [4] C7 [1] 0C 
04 00 00 07 00 00 00 A1 [4] C7 [1] 10 04 00 00 0B 00 00 00 A1 } '
        '$EP = { C2 04 00 6A 00 E8 [4] 33 [1] C2 04 } '
        '$AP = { E8 2C 00 00 00 3D 00 7D 00 00 73 0A 80 FC 05 73 
06 83 F8 7F } '
        'condition: filesize < 200KB and uint16(0) == 0x5A4D and 
uint32(uint32(0x3C)) == 0x4550 and $57B and $EP at (entrypoint-3) 
and $AP }'
    )
    yara_match = rule.match(data=readbin)if yara_match != {}:
        try:
            pe = PE(filepath)
        except:
            print('not valid PE')
            exit()        dsect = [
            pe.sections[i].get_data() for i in 
range(len(pe.sections)) if pe.sections[i].Name.rsplit(b'\x00')[0] 
== b'.data'
            ][0]

        klen = unpack('I', dsect[:4])[0]
        key = dsect[4:4+klen]
        elen = unpack('I', dsect[4+klen:8+klen])[0]
        ebin = dsect[klen+8:klen+8+elen]

        apbin = custom_decryption(key, ebin)
        decrypted_bin = Decompress(BytesIO(apbin)).do()

        fspl = filepath.split('/')[-1]
        fn = fspl.split('.')[0] + '_unpacked.' + 
fspl.split('.')[1] if '.' in fspl else fspl + '_unpacked'
        fp = '/'.join(filepath.split('/')[:-1]) + '/' + fn
        out = open(fp, 'wb')
        out.write(decrypted_bin)def custom_decryption(key, data):
    sbox = [i for i in range(256)]
    kb = [key[i % len(key)] for i in range(256)]    c = 255
    j = 0
    t = 0
    o = b''

    while c > 0:
        v = 1

        while v < c:
            v = (v|1) + v        d = (t + kb[j % 256]) % 256
        b = (d & v) % 256
        j += 1

        if b > c:
            t = d
            continue        sbox[c], sbox[b] = sbox[b], sbox[c]

        t = d
        c -= 1    eb = sbox + [sbox[1]] + [sbox[3]] + [sbox[5]] +
[sbox[7]] + [sbox[t]]    for i in range(len(data)):
        eb[257] = (eb[257] + eb[eb[256]]) % 256
        eb[256] = (eb[256] + 1) % 256
        b1 = eb[eb[260]]
        eb[eb[260]] = eb[eb[257]]
        eb[eb[257]] = eb[eb[259]]
        eb[eb[259]] = eb[eb[256]]
        eb[eb[256]] = b1
        eb[258] = (eb[b1] + eb[258]) % 256
        b1 = (((eb[eb[258]] + eb[eb[259]]) % 256) + eb[eb[260]])
% 256
        eb[260] = data[i]
        v = (eb[eb[256]] + eb[eb[257]]) % 256
        x1 = eb[v] ^ eb[eb[b1]]
        x2 = x1 ^ data[i]
        eb[259] = x2
        o += bytes([x2])

    return omain()
```

Ransomware Sample

The malware is written in Delphi but the interesting part from a RE perspective was the reuse of the same routine from the crypter as part of the string decoding in the malware, this would lead us to believe that they have the same dev and the crypter is probably part of the build process or some service the main actor offers to their affiliates.



```
mov     ebx, [ebp+var_24]
cmp     [ebp+var_10], 0
ja      short loc_41438E
```

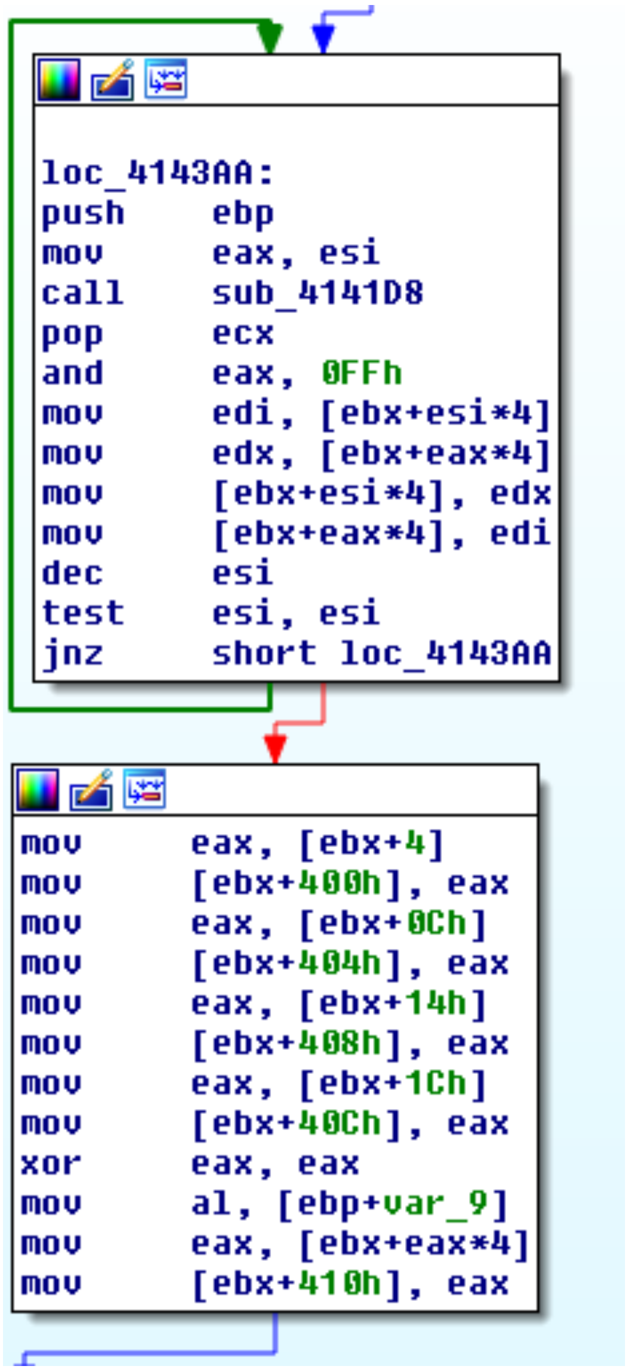
```
loc_41438E:
xor     esi, esi
```

```
loc_414390:
mov     [ebx+esi*4], esi
inc     esi
cmp     esi, 100h
jnz     short loc_414390
```

```
tr [ebx+400h], 1
tr [ebx+404h], 3
tr [ebx+408h], 5
tr [ebx+40Ch], 7
tr [ebx+410h], 0Bh
:
```

```
xor     eax, eax
mov     [ebp+var_8], eax
mov     [ebp+var_9], 0
mov     esi, 0FFh
```

After the SBOX is initialized same as we previously discussed in the crypter we can see the same customized process for RC4 KSA and PRGA performed as was shown in the crypter section.



```
loc_4143AA:
push    ebp
mov     eax, esi
call    sub_4141D8
pop     ecx
and     eax, 0FFh
mov     edi, [ebx+esi*4]
mov     edx, [ebx+eax*4]
mov     [ebx+esi*4], edx
mov     [ebx+eax*4], edi
dec     esi
test    esi, esi
jnz     short loc_4143AA
```

```
mov     eax, [ebx+4]
mov     [ebx+400h], eax
mov     eax, [ebx+0Ch]
mov     [ebx+404h], eax
mov     eax, [ebx+14h]
mov     [ebx+408h], eax
mov     eax, [ebx+1Ch]
mov     [ebx+40Ch], eax
xor     eax, eax
mov     al, [ebp+var_9]
mov     eax, [ebx+eax*4]
mov     [ebx+410h], eax
```

Custom KSA

Because of the way Delphi lays out their strings decoding them is a pretty straight forward process using the same sort of code as the crypter, we just need to find each string and key pair.

```
if __name__ == "__main__":
 data = open(sys.argv[1], 'rb').read()
 curr = 0
 t = data.find(b'\xff\xff\xff\xff')
 done = False
 while not done and t:
  curr += t
  (a,b) = struct.unpack_from('<II', data[curr:])
  if b > 1000:
   continue
  key = data[curr+8:curr+8+b]
  next = data[curr+8+b:].find(b'\xff\xff\xff\xff')
  curr += 8+b+next
  (a2,b2) = struct.unpack_from('<II', data[curr:])
  if b2 > 1000:
   continue
  blob = data[curr+8:curr+8+b2]
  curr += 8+b2
  try:
   print(decode_data(key,data))
  except:
   pass
  t = data[curr:].find(b'\xff\xff\xff\xff')
  if t == -1:
   done = True
```

Convert above to python3

Decoded strings:

```
browser
Software\Microsoft\Windows\CurrentVersion\Run
notepad.exe
desktop
--c=show
--net=0
[+] Process started.
software\
.txt
single
network
-data=
\cmd.txt
c:\
Your ID:
Your support onion(TOR) url:
[+] Preconfig done:
    Work type -
[+] Network communication started - 1.
[+] Network communication started - 2.
[+] Main encryption started.
```

Ransom Note Comparison

The ransomware note has some striking similarities to Revil[1] but also some differences and misspellings:

```
---=== Welcome. Again. ===---

[-] Whats HapPen? [-]

Your files are encrypted, and currently unavailable. You can
check it: all files on your system has extension csruj.
By the way, everything is possible to recover (restore), but you
need to follow our instructions. Otherwise, you cant return your
data (NEVER).

[+] What guarantees? [+]

Its just a business. We absolutely do not care about you and your
deals, except getting benefits. If we do not do our work and
liabilities - nobody will not cooperate with us. Its not in our
interests.
To check the ability of returning files, You should go to our
website. There you can decrypt one file for free. That is our
guarantee.
If you will not cooperate with our service - for us, its does not
matter. But you will lose your time and data, cause just we have
the private key. In practice - time is much more valuable than
money.

[+] How to get access on website? [+]

You have two ways:

1) [Recommended] Using a TOR browser!
  a) Download and install TOR browser from this site:
  b) Open our website:

2) If TOR blocked in your country, try to use VPN! But you can
use our secondary website. For this:
  a) Open your any browser (Chrome, Firefox, Opera, IE, Edge)
  b) Open our secondary website:

Warning: secondary website can be blocked, thats why first
variant much better and more available.

When you open our website, put the following data in the input
form:
Key:
```

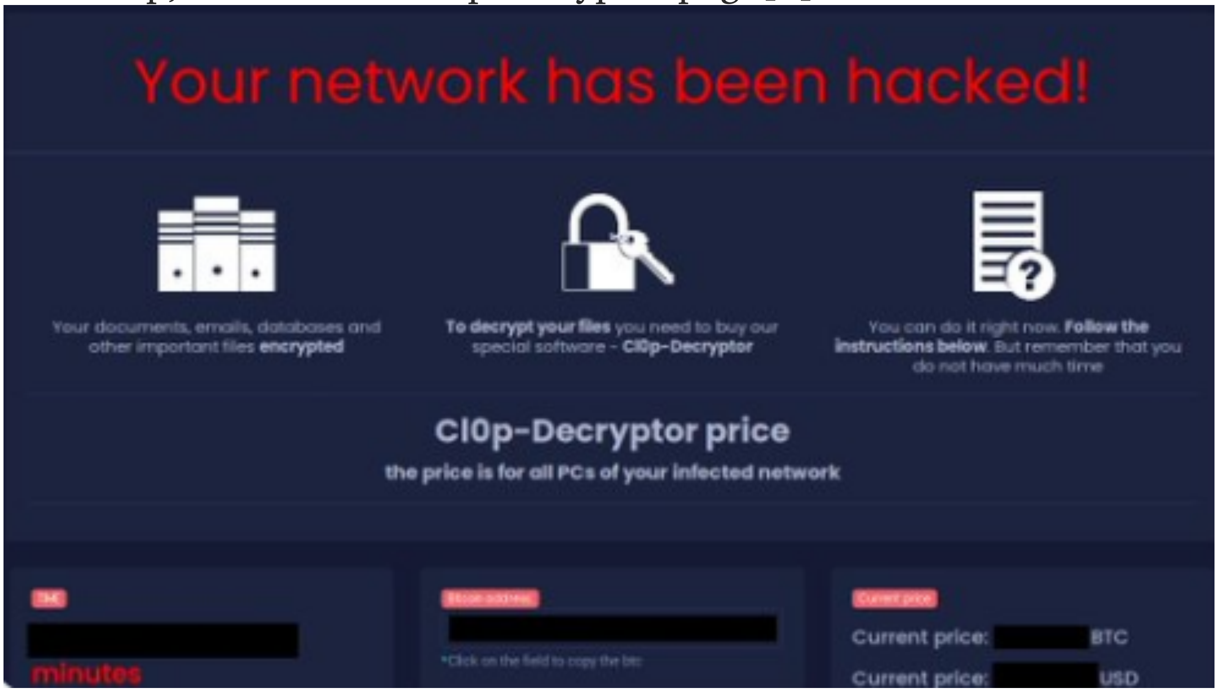
```
------------------------------------------------------------------
------------------------

!!! DANGER !!!
DON'T try to change files by yourself, DON'T use any third party
software for restoring your data or antivirus solutions - its may
entail damage of the private key and, as result, The Loss all
data.
!!! !!! !!!
ONE MORE TIME: Its in your interests to get your files back. From
our side, we (the best specialists) make everything for
restoring, but please should not interfere.
!!! !!! !!!
```

This new RaaS ransom note from sample(4a97bc8111631795cb730dfe7836d0afac3131ed8a91db81dde5062bb8021058):

```
[+] Whats Happen? [+]
Your files are encrypted, and currently unavailable. You can
check it: all files on your system has extension .encoded01.
By the way, everything is possible to recover (restore), but you
need to follow our instructions. Otherwise, you cant
return your data (NEVER).
[+] What guarantees? [+]
Its just a business. We absolutely do not care about you and your
deals, except getting benefits. If we do not do our
work and liabilities - nobody will not cooperate with us. Its not
in our interests.
To check the ability of returning files, You should go to our
website. There you can decrypt 1-5 files for free. That
our guarantee.
If you will not cooperate with our service - for us, its does not
matter. But you will lose your time and data, cause
just we have the private key. In practise - time is much more
valuable than money.
[+] How to get access on website? [+]
You can open our site by the shortcut &quot;SUPPORT
(TOR_BROWSER)&quot; created on the desktop.
Also as the second option you can install the tor browser:
        a) Download and install TOR browser from this site:
https://torproject.org/
        b) Open our website. Full link will be provided below.
------------------------------------------------------------------
------------------------
!!! DANGER !!!
```

```
DONT try to change files by yourself, DONT use any third party
software for restoring your data or antivirus solutions
its may entail damge of the private key and, as result, The Loss
all data.
!!! !!! !!!
ONE MORE TIME: Its in your interests to get your files back. From
our side, we (the best specialists) make everything
for restoring, but please should not interfere.
!!! !!! !!!
-----------------------------------------------------------------
------------------------
```

Another similarity we can find but to a different ransomware family is to Cl0p, below is the Cl0p decryptor page[2].



Your network has been hacked!

Your documents, emails, databases and other important files **encrypted**

To decrypt your files you need to buy our special software - **Cl0p-Decryptor**

You can do it right now. **Follow the instructions below**. But remember that you do not have much time

Cl0p-Decryptor price

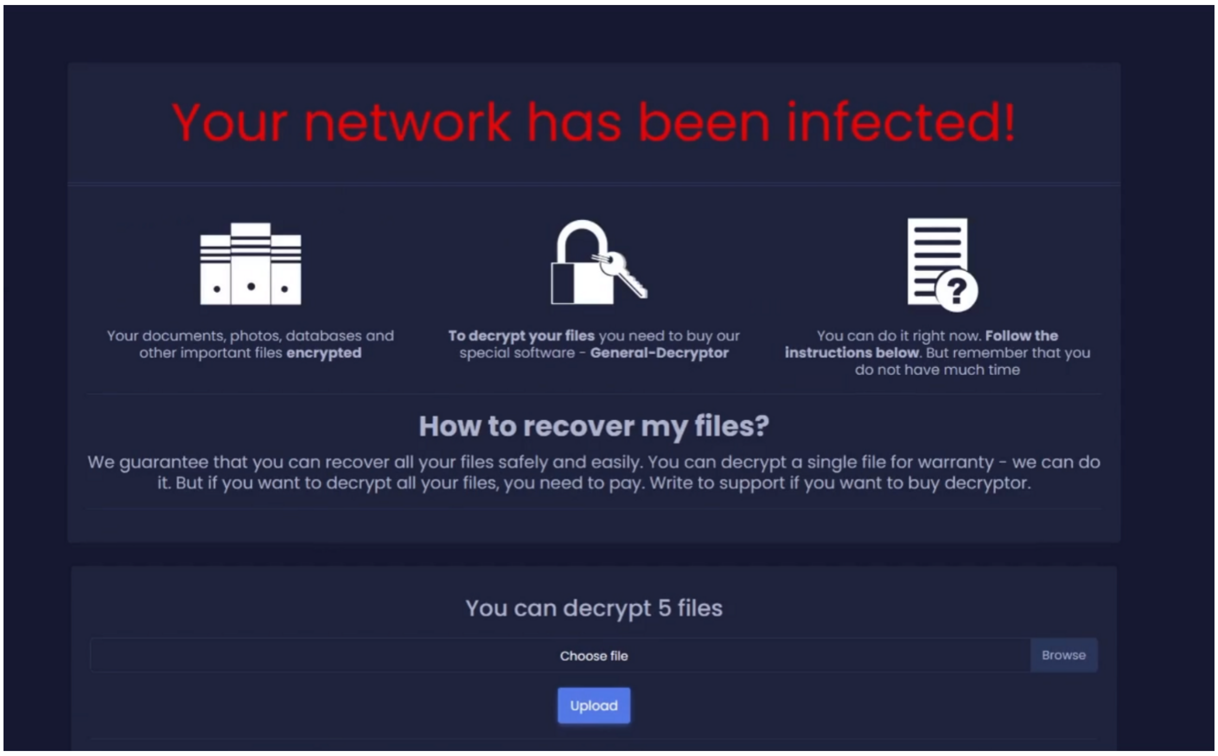
the price is for all PCs of your infected network

[redacted]
minutes

[redacted]
*Click on the field to copy the btc

Current price: [redacted] BTC
Current price: [redacted] USD

Comparing it to this new RaaS shows a striking similarity:


Your network has been infected!

Your documents, photos, databases and other important files **encrypted**

To decrypt your files you need to buy our special software - **General-Decryptor**

You can do it right now. **Follow the instructions below**. But remember that you do not have much time

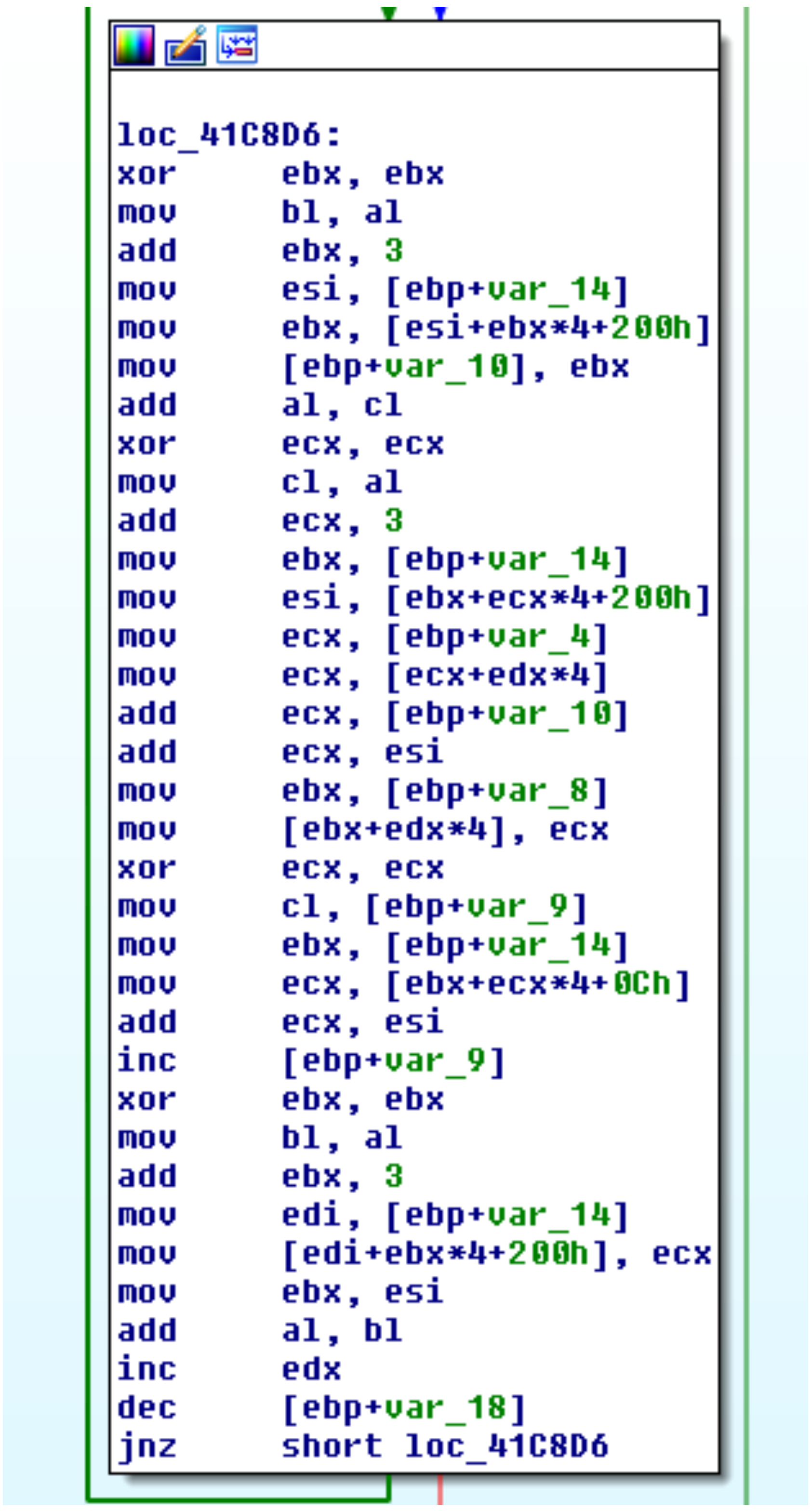
How to recover my files?

We guarantee that you can recover all your files safely and easily. You can decrypt a single file for warranty - we can do it. But if you want to decrypt all your files, you need to pay. Write to support if you want to buy decryptor.

You can decrypt 5 files

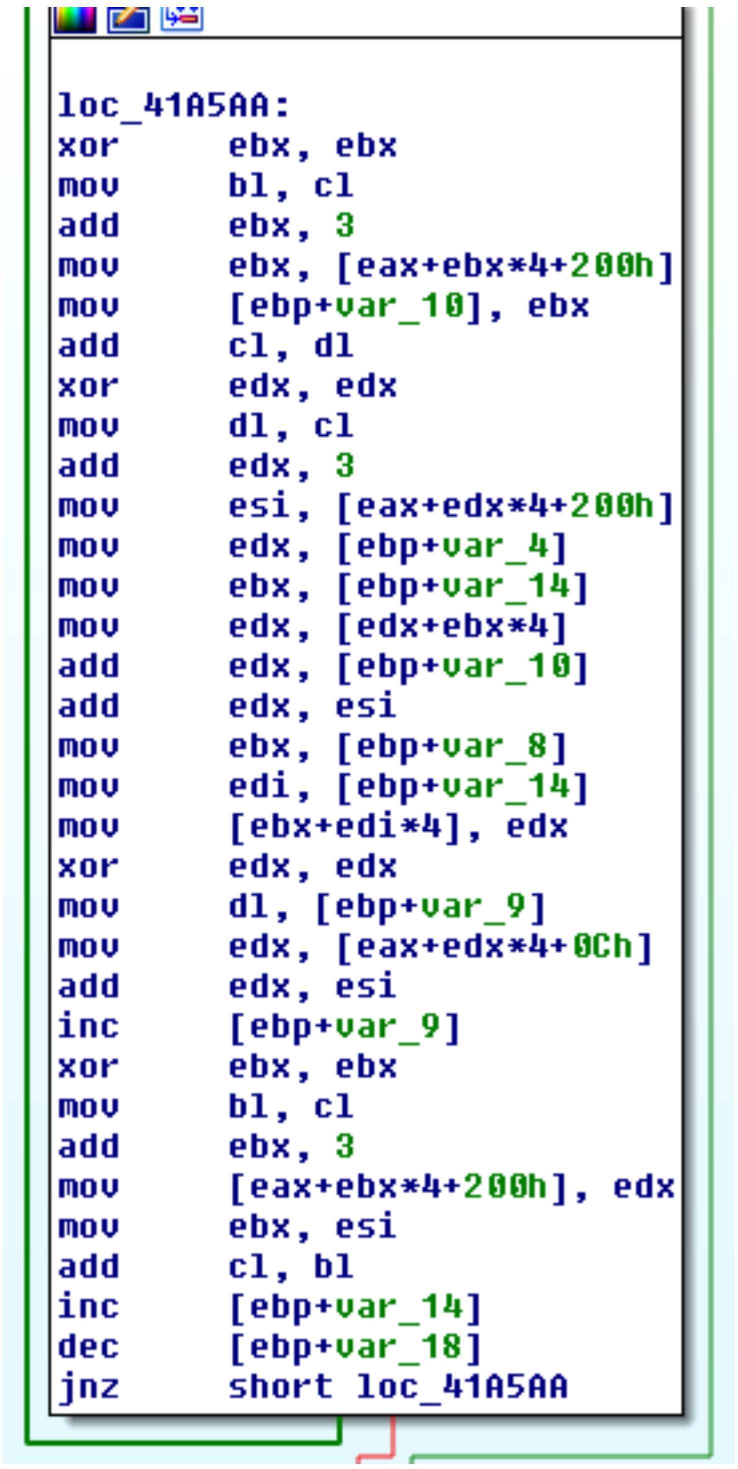
The file encryption piece for samples we analyzed appear to be using SCOP encryption algorithm.

From the ransomware sample:



```
loc_41C8D6:
xor     ebx, ebx
mov     bl, al
add     ebx, 3
mov     esi, [ebp+var_14]
mov     ebx, [esi+ebx*4+200h]
mov     [ebp+var_10], ebx
add     al, cl
xor     ecx, ecx
mov     cl, al
add     ecx, 3
mov     ebx, [ebp+var_14]
mov     esi, [ebx+ecx*4+200h]
mov     ecx, [ebp+var_4]
mov     ecx, [ecx+edx*4]
add     ecx, [ebp+var_10]
add     ecx, esi
mov     ebx, [ebp+var_8]
mov     [ebx+edx*4], ecx
xor     ecx, ecx
mov     cl, [ebp+var_9]
mov     ebx, [ebp+var_14]
mov     ecx, [ebx+ecx*4+0Ch]
add     ecx, esi
inc     [ebp+var_9]
xor     ebx, ebx
mov     bl, al
add     ebx, 3
mov     edi, [ebp+var_14]
mov     [edi+ebx*4+200h], ecx
mov     ebx, esi
add     al, bl
inc     edx
dec     [ebp+var_18]
jnz     short loc_41C8D6
```

SCOP from GPLib[4]:



```
loc_41A5AA:
xor     ebx, ebx
mov     bl, cl
add     ebx, 3
mov     ebx, [eax+ebx*4+200h]
mov     [ebp+var_10], ebx
add     cl, dl
xor     edx, edx
mov     dl, cl
add     edx, 3
mov     esi, [eax+edx*4+200h]
mov     edx, [ebp+var_4]
mov     ebx, [ebp+var_14]
mov     edx, [edx+ebx*4]
add     edx, [ebp+var_10]
add     edx, esi
mov     ebx, [ebp+var_8]
mov     edi, [ebp+var_14]
mov     [ebx+edi*4], edx
xor     edx, edx
mov     dl, [ebp+var_9]
mov     edx, [eax+edx*4+0Ch]
add     edx, esi
inc     [ebp+var_9]
xor     ebx, ebx
mov     bl, cl
add     ebx, 3
mov     [eax+ebx*4+200h], edx
mov     ebx, esi
add     cl, bl
inc     [ebp+var_14]
dec     [ebp+var_18]
jnz     short loc_41A5AA
```

IOCs

```
bottomcdnfiles.com
cdnmegafiles.com
179.43.160.195
chat5sqrnzqewampznybomgn4hf2m53tybkarxk4sfaktwt7oqpkcvyd.onion
82.146.53.237
sugarpanel.space15a7fb45f703d5315320eef132f3151873055161
```

```
5816a77bf4f8485bfdab1803d948885f76e0c926fed9da5ac02d94e62af8b145
320eefd378256d6e495cbd2e59b7f205d5101e7f
18cb9b218bd23e936128a37a90f2661f72c820581e4f4303326705b2103714a9
e835de2930bf2708a3a57a99fe775c48f851fa8f
1318aeaea4f2f4299c21699279ca4ea5c8fa7fc38354dd2b80d539f21836df5a
98137dd04e4f350ee6d2f5da613f365b223a4f49
aa41e33d3f184cedaaaabb5e16c251e90a6c4ff721a599642dc5563a57550822
a4854ce87081095ab1f1b26ff16817e446d786af
4a97bc8111631795cb730dfe7836d0afac3131ed8a91db81dde5062bb8021058
c31a0e58ae70f571bf8140db8a1ab20a7f566ab5
315045e506eb5e9f5fd24e4a55cda48d223ac3450037586ce6dab70afc8ddfc9
```

References

1:https://raw.githubusercontent.com/cado-security/DFIR_Resources_REvil_Kaseya/main/Config/Ransomware_Note.txt

2:https://malwarewarrior.com/how-to-remove-cl0p-ransomware-and-decrypt-cl0p-files/

3:https://twitter.com/avman1995/status/1459915441766211601

4:https://torry.net/pages.php?id=519