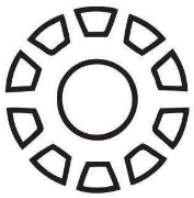
reecdeep/HiveV5_file_decryptor

github.com/reecdeep/HiveV5_file_decryptor

reecdeep

reecdeep/
HiveV5_file_decryptor

Hive v5 file decryption algorithm



1 Contributor | 0 Issues | 29 Stars | 3 Forks

HiveV5 file decryptor PoC

Introduction

The work done in the last few months has been necessary to reveal the malicious file encryption mechanism of Hive v5-5.2. The work was divided into two parts

1. Keystream decryption
2. File decryption using the decrypted keystream

I would like to thank the great @rivitna for the support, dialogue and advices of these months of work! Please take note of rivitna's github full of useful informations about Hive ransomware and more.

In this readme you will find some information about the file decryption algorithm, referring you to the PoC for a more complete picture of how it works. A keystream is an encrypted cleartext. A cleartext is a set of 0xA00000 bytes to which the first 0x2FFF00 bytes have been appended, for a total of 0xCFFF00 bytes. These bytes were created with the weak algorithm already discussed in the first part released in July 2022. Here below is a example of cleartext:

```
00000000  BC E8 4C B0 14 78 DC 40 A4 08 6C D0 98 FC 60 C4  ¼èL°.xÜ@¤.lÐ˜ü`Ä
00000010  28 8C F0 54 B8 1C 80 E4 48 10 74 D8 3C A0 04 68  (ŒðT¸.€äH.tØ< .h
00000020  CC 30 94 F8 5C 24 88 EC 50 B4 18 7C E0 44 A8 0C  Ì0”ø\$ˆìP´.|àD¨.
00000030  70 D4 9C 00 64 C8 2C 90 F4 58 BC 20 84 E8 4C 14  pÔœ.dÈ,.ôX¼ „èL.
00000040  78 DC 40 A4 08 6C D0 34 98 FC 60 28 8C F0 54 B8  xÜ@¤.lÐ4˜ü`(ŒðT¸
00000050  1C 80 E4 48 AC 10 74 D8 A0 04 68 CC 30 94 F8 5C  .€äH¬.tØ .hÌ0”ø\
00000060  C0 24 88 50 B4 18 7C E0 44 A8 0C 70 D4 38 9C 00  À$ˆP´.|àD¨.pÔ8œ.
00000070  C8 2C 90 F4 58 BC 20 84 E8 4C B0 14 78 40 A4 08  È,.ôX¼ „èL°.x@¤.
00000080  6C D0 34 98 FC 60 C4 28 8C 54 B8 1C 80 E4 48 AC  lÐ4˜ü`Ä(ŒT¸.€äH¬
00000090  10 74 D8 3C A0 04 CC 30 94 F8 5C C0 24 88 EC 50  .tØ< .Ì0”ø\À$ˆìP
000000A0  B4 7C E0 44 A8 0C 70 D4 38 9C 00 64 C8 90 F4 58  ´|àD¨.pÔ8œ.dÈ.ôX
  ●●●                                 ●●●
00CFFE30  28 8C F0 54 B8 1C 80 E4 48 AC 10 74 D8 3C A0 04  (ŒðT¸.€äH¬.tØ< .
00CFFE40  68 CC 30 94 F8 C0 24 88 EC 50 B4 18 7C E0 44 A8  hÌ0”øÀ$ˆìP´.|àD¨
00CFFE50  0C 70 D4 38 9C 00 64 C8 2C 90 F4 58 BC 20 84 E8  .pÔ8œ.dÈ,.ôX¼ „è
00CFFE60  4C B0 14 78 DC 40 A4 08 6C D0 34 98 60 C4 28 8C  L°.xÜ@¤.lÐ4˜`Ä(Œ
00CFFE70  F0 54 B8 1C 80 E4 48 AC 10 74 D8 3C A0 04 68 CC  ðT¸.€äH¬.tØ< .hÌ
00CFFE80  30 94 F8 5C C0 24 88 EC 50 B4 18 7C E0 44 A8 0C  0”ø\À$ˆìP´.|àD¨.
00CFFE90  70 D4 9C 00 64 C8 2C 90 F4 58 BC 20 84 E8 4C B0  pÔœ.dÈ,.ôX¼ „èL°
00CFFEA0  14 78 DC 40 A4 08 6C D0 34 98 FC 60 C4 28 8C F0  .xÜ@¤.lÐ4˜ü`Ä(Œð
00CFFEB0  54 B8 1C 80 E4 48 AC 10 74 3C A0 04 68 CC 30 94  T¸.€äH¬.t< .hÌ0”
00CFFEC0  F8 5C C0 24 88 EC 50 B4 18 7C E0 44 A8 0C 70 D4  ø\À$ˆìP´.|àD¨.pÔ
00CFFED0  38 9C 00 64 C8 2C 90 F4 58 BC 20 84 E8 4C B0 14  8œ.dÈ,.ôX¼ „èL°.
00CFFEE0  78 40 A4 08 6C D0 34 98 FC 60 C4 28 8C F0 54 B8  x@¤.lÐ4˜ü`Ä(ŒðT¸
00CFFEF0  1C 80 E4 48 AC 10 74 D8 3C A0 04 68 CC 30 94 F8  .€äH¬.tØ< .hÌ0”ø
```

The Hive sample analyzed and referred to in this document was chosen from this list created by @rivitna to which my warmest thanks go. To get an idea of the complexity of ransomware, please take a look at this analysis published by Microsoft Threat Intelligence Center (MSTIC).

File encryption algorithm

The cleartext (a decrypted keystream) is used by Hive ransomware when encrypting each file. When encrypting a file, Hive ransomware calculates two integers referring to precise positions in the cleartext (offsets) to be used to encrypt the file according to the following formula:

$$file_{encrypted}[i] = cleartext[offset_1 + c] \oplus cleartext[offset_2 + d] \oplus file_{clear}[i]$$

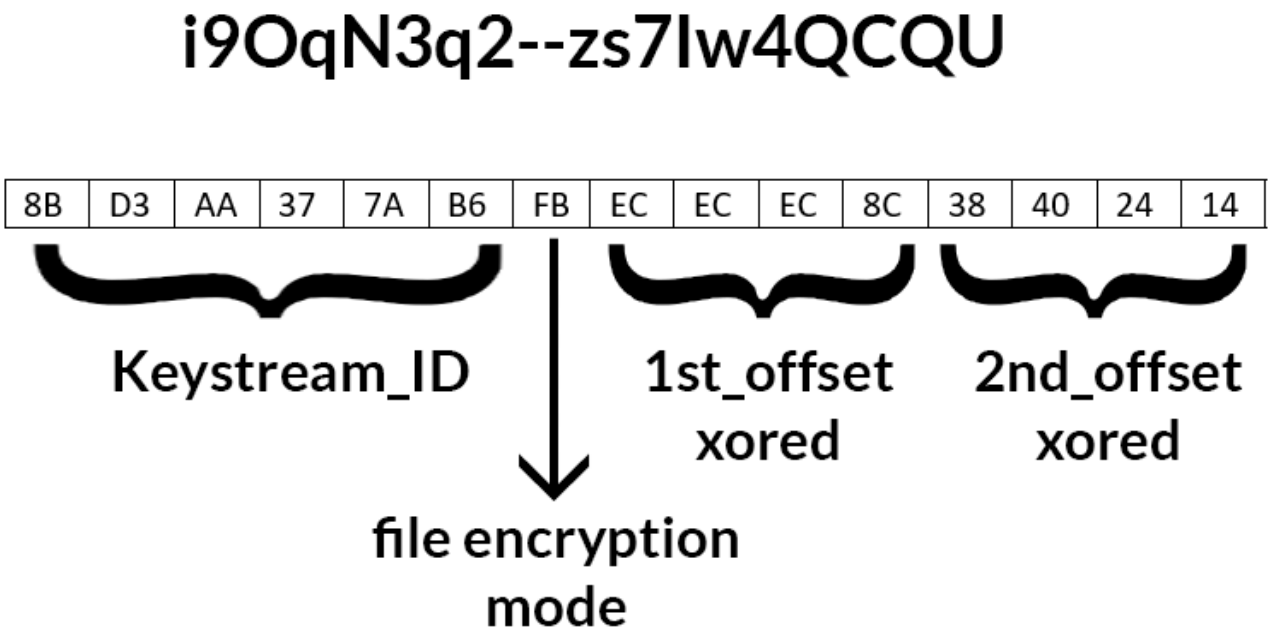
where c = i % 0x2FFF00 e d = i % 0x2FFD00 , with i as a byte counter.

The encrypted file extension

The preliminary operations before writing a file are:

- Renaming the file using MoveFileExW and changing its extension;

- Writing the renamed file with the result of the xor operation shown above.



Also in this case the cleartext plays a fundamental role. In fact it is used for:

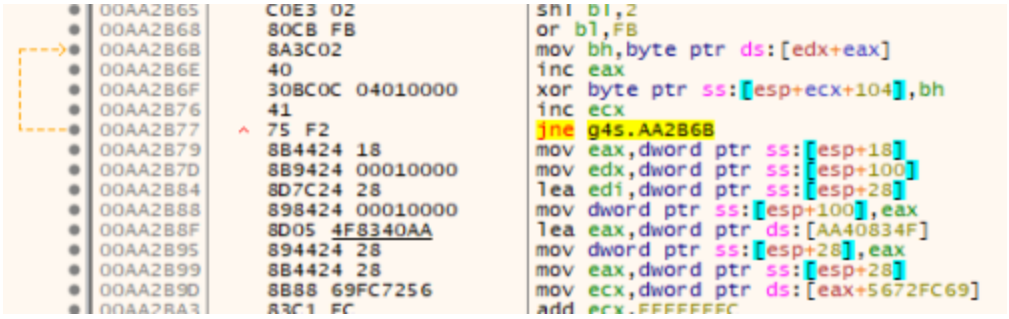
1. Determine the keystream ID (first 6 bytes) using a hash function
2. Encrypt the positions (offsets) used to extract bytes from the cleartext However, the first offset is encrypted using a fixed position of the cleartext and is different for each Hive 5/5.1/5.2 sample. A kind of magical value. In many Hive 5/5.1 artifacts this magic value is shown explicitly inside a memory reference, like in this case 0x98072A :

```
00423E03   80CB FB          or bl,FB
00423E06   8A8C16 2A079800  mov cl,byte ptr ds:[esi+edx+98072A]
00423E0D   304C14 2C        xor byte ptr ss:[esp+edx+2C],cl
00423E11   42               inc edx
00423E12   75 F2            jne x.423E06
00423E14   8B4C24 28        mov ecx,dword ptr ss:[esp+28]
```

Or this case 0x7539D:

```
00581A00   884C24 0C        mov byte ptr ss:[esp+C],cl
00581A04   8A8C17 9D530700  mov cl,byte ptr ds:[edi+edx+7539D]
00581A0B   304C14 44        xor byte ptr ss:[esp+edx+44],cl
00581A0F   42               inc edx
00581A10   75 F2            jne aOh2uih3d2.581A04
```

But in the next evidence the for loop is slightly different and has been written in such a way as not to explicit the magic value that we need to identify. This concerns an artifact belonging to Hive 5.2:



In this case it is possible to use the offset bruteforce function present in the released tool, using a file with a known extension and the relative decrypted keystream. Using the header of the encrypted file and the header of the unencrypted file it is possible to understand what is the offset from which the decryptor must start to decrypt the file.

The file encryption mode can have two values: 0xFB or 0xFF

- 0xFB means that the ransomware encrypted the entire file without leaving any portion of the file unencrypted.
- 0xFF means that the ransomware calculated a NCB (not encrypted block) for each file and encrypting blocks of 0x100000 bytes. For further information regarding the calculation of the size of the unencrypted blocks and the cleartext offset, please refer to the PoC code.

Usage

The program offers two options:

```
Hive ransomware V5 - file decryptor PoC
-----------------------------------------

1. Decrypt a file using decrypted keystream
2. Offset bruteforce
your move:
```

1. Decryption of files using the decrypted keystream. You need to enter the special offset present in the sample that encrypted the files.
2. Given a file with a known header (PDF, JPG, PNG, Office files) brute the possible value of the special offset by decrypting the first bytes and looking for a match with the known signature

References

https://github.com/rivitna/Malware/blob/main/Hive/Hive_samples.txt