

Remcos Revisited: Inside the RAT's Evolving Command-and-Control Techniques

 pointwild.com/threat-intelligence/remcos-revisited-inside-the-rats-evolving-command-and-control-techniques

Sarah Burns

February 18, 2026

A graphic with a teal background and a white dotted pattern. The text "Remcos Revisited" is in a large, white, sans-serif font, and "Malware Analysis Report" is in a smaller, white, sans-serif font below it.

Remcos Revisited

Malware Analysis Report

Summary

Remcos RAT, originally a powerful surveillance and control tool, has become one of the most dangerous threats globally. Its ability to bypass defenses and steal credentials makes it a top priority for cybersecurity teams to detect and mitigate.

Remcos is a Remote Access Trojan (RAT) originally a commercial remote management tool that threats actors increasingly abuse to gain unauthorized remote control of Windows systems. It provides capabilities such as credential theft, keylogging, screen capture, webcam/microphone control, and persistent command and control (C2) access.

Infection Vectors

- **Phishing Emails:** Malicious attachments (often disguised as invoices, job offers, or government notices).
- **Compromised Websites:** Hosting payloads that download and install Remcos.

Key Features of Remcos RAT

- **Remote Control:** Full access to the victim's system, including files, processes, and registry.
- **Surveillance:** Keylogging, screen capture, webcam/microphone recording.
- **Persistence:** Uses registry modifications and startup tricks to remain active.
- **C2 Communication:** Connects to attacker-controlled servers for instructions and data exfiltration.
- **Fileless Attacks:** Recent campaigns inject malicious code directly into memory, bypassing Endpoint Detection & Response (EDR) tools.

Evolution of New Remcos Functionalities:

The latest Remcos variant has been observed exhibiting a significant change in behaviour compared to previous versions. Instead of stealing and storing data locally on the infected system, this variant establishes direct online command-and-control (C2) communication, enabling real-time access and control. In particular, it leverages the webcam to capture live video streams, allowing attackers to monitor targets remotely. This shift from local data exfiltration to live, online surveillance represents an evolution in Remcos' capabilities, increasing the risk of immediate espionage and persistent monitoring.

Our Lat61 Threat Intelligence research team has previously covered legacy [Remcos](#) previous variants and their associated behaviours.

Infection Workflow

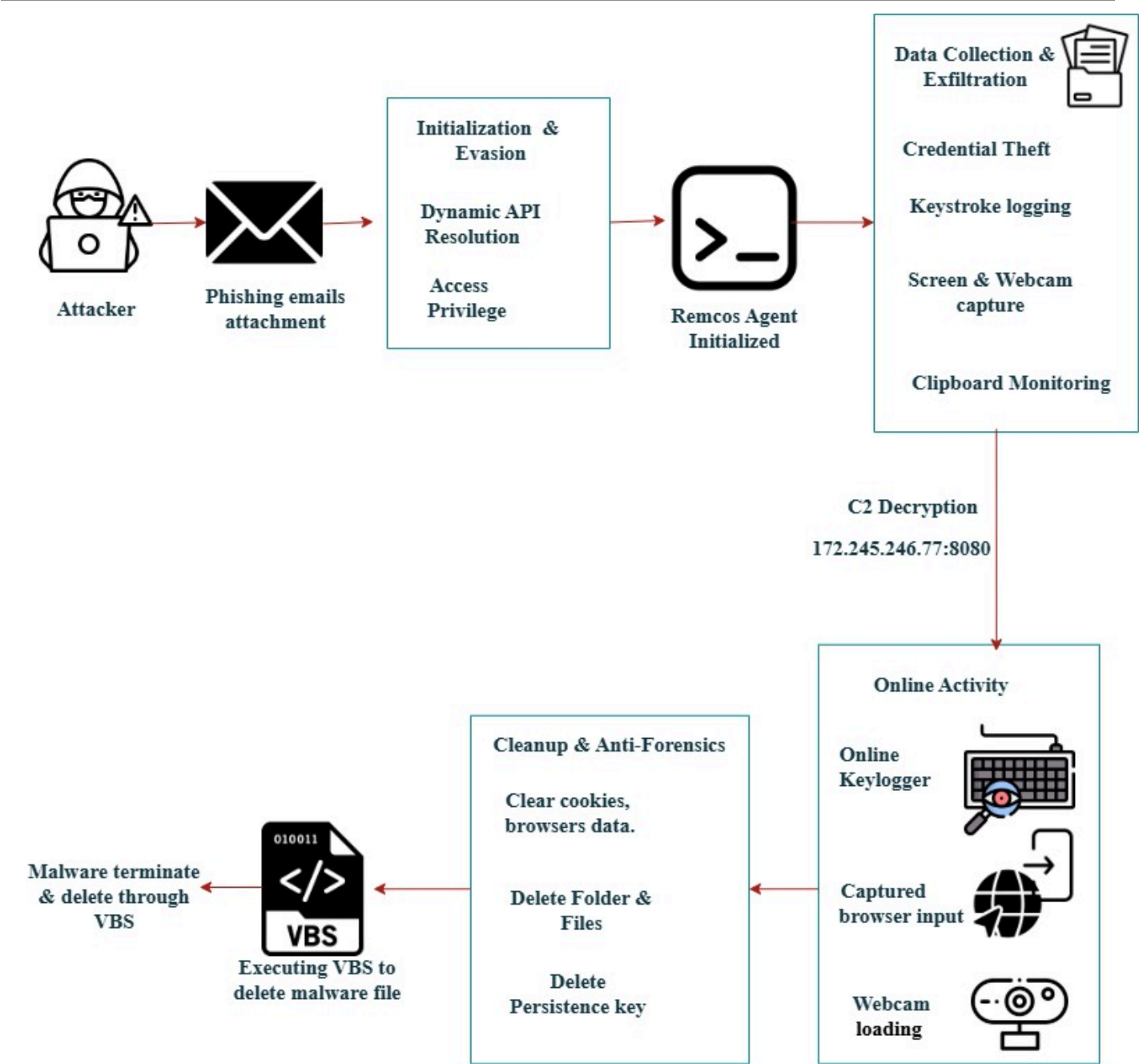


Figure 1: Infection Workflow.

Technical Details

Malware often uses dynamic API resolution as a stealth and evasion technique. Instead of hardcoding the names of Windows API functions (like CreateFileA, LoadLibrary, etc.), it resolves them at runtime.

APIs are decrypted at runtime to evade static detection, hinder reverse engineering, and dynamically resolve malicious functionality during execution.

0871BF70	FFD7	call edi
0871BF72	50	push eax
0871BF73	FFD6	call esi
0871BF75	68 E4CE760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CEE4
0871BF7A	55	push ebp
0871BF7B	A3 244B770B	mov dword ptr ds:[B774B24],eax
0871BF80	FFD3	call ebx
0871BF82	50	push eax
0871BF83	FFD6	call esi
0871BF85	68 F4CE760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CEF4
0871BF8A	55	push ebp
0871BF8B	A3 2C4B770B	mov dword ptr ds:[B774B2C],eax
0871BF90	FFD3	call ebx
0871BF92	50	push eax
0871BF93	FFD6	call esi
0871BF95	68 08CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF08
0871BF9A	68 18CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF18
0871BF9F	A3 304B770B	mov dword ptr ds:[B774B30],eax
0871BFA4	FFD7	call edi
0871BFA6	50	push eax
0871BFA7	FFD6	call esi
0871BFA9	68 20CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF20
0871BFAE	55	push ebp
0871BFAF	A3 144B770B	mov dword ptr ds:[B774B14],eax
0871BF84	FFD3	call ebx
0871BF86	50	push eax
0871BF87	FFD6	call esi
0871BF89	68 34CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF34
0871BF8E	68 A0CE760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CEA0
0871BFC3	A3 104B770B	mov dword ptr ds:[B774B10],eax
0871BFC8	FFD3	call ebx
0871BFCA	50	push eax
0871BFCE	FFD6	call esi

Figure 2: Before API decryption.

0871BF70	FFD7	call edi
0871BF72	50	push eax
0871BF73	FFD6	call esi
0871BF75	68 E4CE760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CEE4
0871BF7A	55	push ebp
0871BF7B	A3 244B770B	mov dword ptr ds:[<&GlobalMemoryStatusEx>],eax
0871BF80	FFD3	call ebx
0871BF82	50	push eax
0871BF83	FFD6	call esi
0871BF85	68 F4CE760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CEF4
0871BF8A	55	push ebp
0871BF8B	A3 2C4B770B	mov dword ptr ds:[<&IsWow64Process>],eax
0871BF90	FFD3	call ebx
0871BF92	50	push eax
0871BF93	FFD6	call esi
0871BF95	68 08CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF08
0871BF9A	68 18CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF18
0871BF9F	A3 304B770B	mov dword ptr ds:[<&GetComputerNameExW>],eax
0871BFA4	FFD7	call edi
0871BFA6	50	push eax
0871BFA7	FFD6	call esi
0871BFA9	68 20CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF20
0871BFAE	55	push ebp
0871BFAF	A3 144B770B	mov dword ptr ds:[<&IsUserAnAdmin>],eax
0871BF84	FFD3	call ebx
0871BF86	50	push eax
0871BF87	FFD6	call esi
0871BF89	68 34CF760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CF34
0871BF8E	68 A0CE760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76CEA0
0871BFC3	A3 104B770B	mov dword ptr ds:[<&SetProcessDEPPolicy>],eax
0871BFC8	FFD3	call ebx
0871BFCA	50	push eax
0871BFCE	FFD6	call esi

Figure 3: After API decryption.

The malware checks system access privileges to determine available capabilities, decide on privilege escalation, bypass security controls, and adapt its malicious behavior accordingly.

Certain actions (e.g., Writing to protected directories, modifying registry keys, installing persistence mechanisms, or disabling security services) require elevated rights.

8BF4	mov esi,esp	
68 8476760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767684	B767684:"Administrator"
E8 0A	jmp d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B70E110	
83EC 18	sub esp,18	
8BF4	mov esi,esp	
68 6C76760B	push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76766C	B76766C:"User"
8D4C24 30	lea ecx,dword ptr ss:[esp+30]	
E8 4D3EFFFF	call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701F66	
50	push eax	
BA 7476760B	mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767674	eax:&"Access Level: Administrator" B767674:"Access Level: "

Figure 4: Access Privileges check.

Remcos RAT initializes by decrypting its configuration, setting persistence, and dynamically loading APIs to evade detection before establishing command-and-control (C2) communication. Its initialization sequence is designed to hide its true capabilities until runtime, making it more difficult.

Remcos stores its configuration in encrypted or compressed form inside the binary.

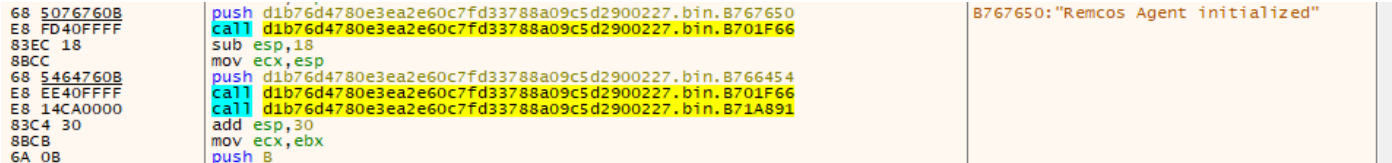


Figure 5: Remcos Agent Console.

How Remcos Steals and Exfiltrates Sensitive Data

Remcos is designed to steal sensitive information and provide attackers with persistent remote access to compromised systems. Its stealing capabilities focus on credential harvesting, surveillance, and data exfiltration.

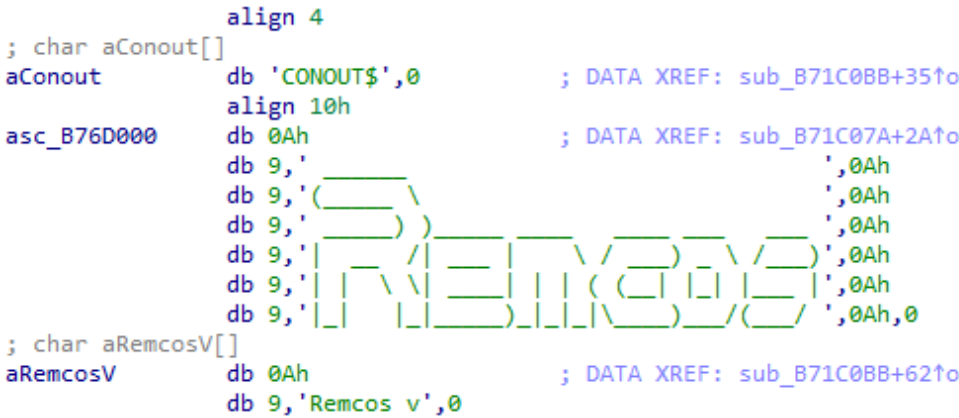


Figure 6: Remcos Identification.

Remcos Key Stealing Capabilities

Credential Theft:

Steals usernames and passwords from web browsers (Chrome, Firefox, Edge, Internet Explorer).

Captures authentication data stored in the Windows Credential Manager.

Keystroke Logging:

Implements keylogging to capture typed data such as passwords, chat messages, and search queries.

Logs keystrokes in real time and forwards them directly to the C2 server.

Screen & Webcam Capture:

Takes screenshots of the desktop at configurable intervals.

Activates the webcam to capture images or live video streams without user awareness.

Clipboard Monitoring:

Monitors clipboard contents to capture copied credentials, cryptocurrency wallet addresses, or sensitive content.

Real-Time Data Exfiltration:

In newer variants, stolen data is exfiltrated directly via online C2 communication, rather than being stored locally, reducing forensic artifacts.

```
0B766BA4 "\r\n{ User has been idle for "
0B766BC0 "Online Keylogger Started"
0B766BDC "Online Keylogger Stopped"
0B766BF8 "Offline Keylogger Stopped"
0B766C18 L"\r\n[%04i/%02i/%02i %02i:%02i:%02i "
0B766B84 L"]\r\n"
0B766C80 "[Alt]"
0B766C74 "[Enter]\r\n"
0B766C6C "[Tab]"
0B766C64 "[BckSp]"
0B766C88 "[Pause]"
0B766C5C "[Space]"
0B766C90 "[Esc]"
0B766C98 "[PagUp]"
0B766CA0 "[PagDw]"
0B766CA8 "[End]"
0B766CB0 "[Start]"
0B766CB8 "[Left]"
0B766CC0 "[Up]"
0B766CC8 "[Right]"
0B766CD0 "[Down]"
0B766CD8 "[Print]"
0B766CE0 "[Ins]"
0B766CF8 "[Menu]"
0B766CF0 "[Win]"
0B766CE8 "[Del]"
0B766D00 "[F1]"
0B766D18 "[F4]"
0B766D10 "[F3]"
0B766D08 "[F2]"
0B766D20 "[F5]"
0B766D58 "[F12]"
0B766D50 "[F11]"
0B766D48 "[F10]"
0B766D40 "[F9]"
0B766D38 "[F8]"
0B766D30 "[F7]"
0B766D28 "[F6]"
0B766DAC "[Ctrl+"
0B766D60 "\r\n[End of clipboard]\r\n\r\n"
0B766D7C "[Ctrl+V]\r\n[Text pasted from clipboard]\r\n"
0B766DBC "[AltR]"
0B766DB4 "[AltL]"
0B766DCC "[CtrlR]"
0B766DC4 "[CtrlL]"
0B766DD4 "\r\n[End of clipboard]\r\n"
0B766DEC "\r\n[Text copied to clipboard]\r\n"
0B766E10 "\\AppData\\Local\\Google\\Chrome\\User Data\\Default\\Login Data"
0B766E4C "UserProfile"
0B766E58 "\n[Chrome StoredLogins not found]"
```

Figure 7: Remcos characteristics.

Remcos includes a geolocation capability that retrieves the infected host's IP address, which can be used to determine the victim's approximate geographic region.

88E8	mov eop,eax	
68 6CC9760B	push dib76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76C96C	B76C96C:L"http://geoplugin.net/json.gp"
55	push ebp	
FF15 2884750B	call dword ptr ds:[<InternetOpenUrl>]	
88F0	mov esi,eax	
836424 10 00	and dword ptr ss:[esp+10],0	
8D4424 10	lea eax,dword ptr ss:[esp+10]	
50	push eax	
68 FFFF0000	push FFFF	

Figure 8: Tracing geolocation.

Remcos RAT maintains internal log data to record stolen information and operational activity. The way logs are handled depends on the variant, configuration, and attack strategy.

1. Keystroke Logs

1. Records user keystrokes captured via its keylogging module.
2. Older variants may temporarily store keystrokes in local log files before exfiltration.
3. Newer variants often send keystrokes directly to the C2 server, minimizing local artifacts.

2. Screen, Webcam, and Audio Logs

1. Screenshots, webcam captures, or audio recordings may be:
2. Stored temporarily in hidden directories, or
3. Buffered in memory and exfiltrated immediately via C2.
4. File names are often randomized or encrypted.

3. System Activity Logs

1. Logs system profiling data such as OS version and architecture, Username, hostname, installed software and running processes

4. Command Execution Logs

1. Tracks executed commands and task results received from the C2.

During analysis of the new Remcos variant, the malware was observed creating a log file at the following location:

"C:\ProgramData\colors\logs.dat"

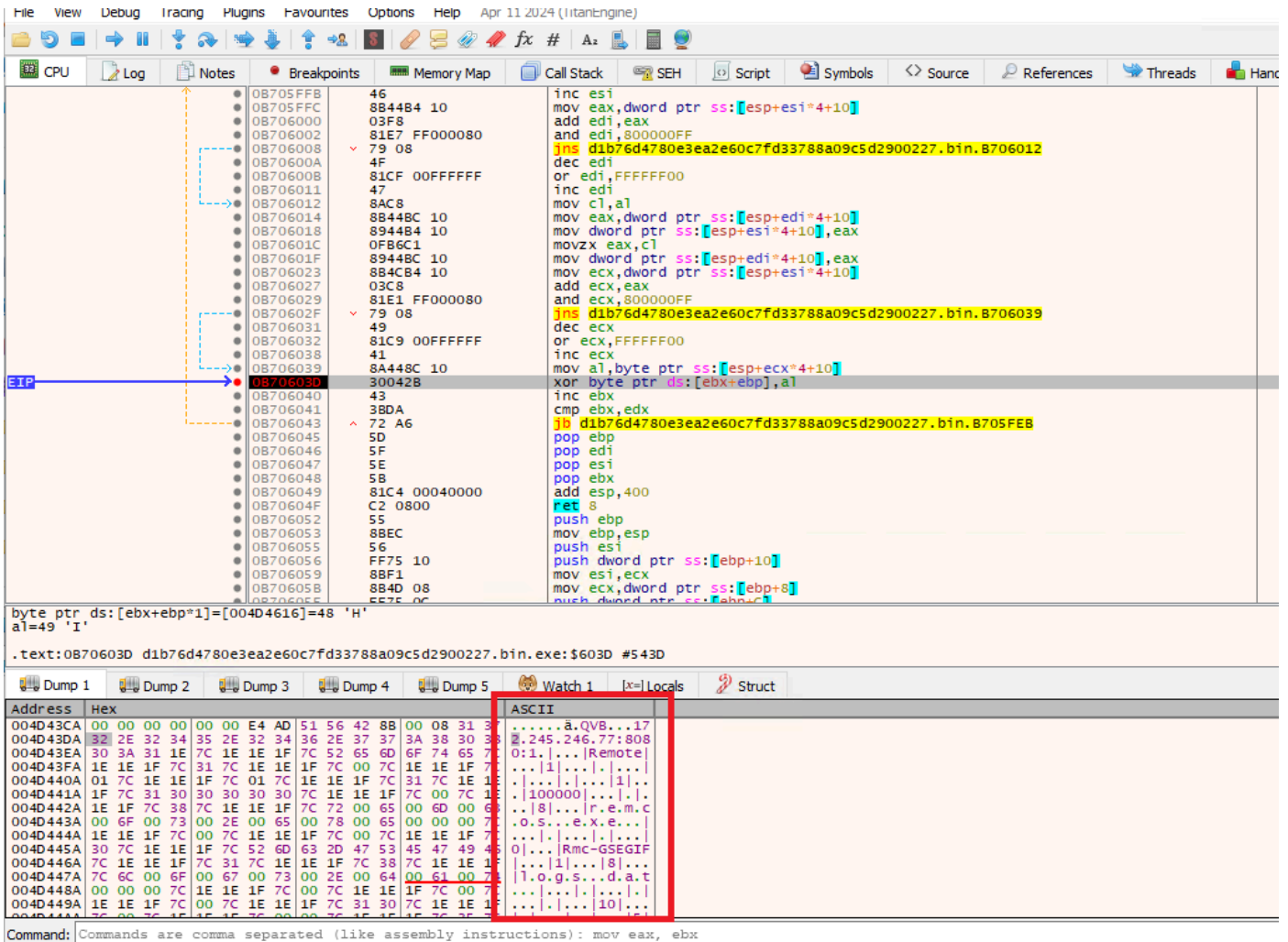
0B709589	88D0	mov edx,eax	eax:&"C:\\ProgramData\\colors\\logs.dat"
0B70958B	8D4D E4	lea ecx,dword ptr ss:[ebp-1c]	
0B70958E	E8 3C93FFFF	call dib76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7028CF	
0B709593	59	pop ecx	
0B709594	50	push eax	eax:&"C:\\ProgramData\\colors\\logs.dat"
0B709595	8D4E 68	lea ecx,dword ptr ds:[esi+68]	[esi+68]:L"C:\\ProgramData\\colors\\logs.dat"
0B709598	E8 7888FFFF	call dib76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E18	
→ 0B70959D	8D4D E4	lea ecx,dword ptr ss:[ebp-1c]	
0B7095A0	E8 6E88FFFF	call dib76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E13	
0B7095A5	8D4D CC	lea ecx,dword ptr ss:[ebp-3]	
0B7095A8	E8 6688FFFF	call dib76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E13	
0B7095AD	6945 1C E8030000	imul eax,dword ptr ss:[ebp-1c],3E8	

Why does Remcos encrypt its C2?

Remcos does not store its C2 in plaintext in order to evade static string extraction, signature-based antivirus detection, and easy IOC harvesting.

Instead, it embeds the C2 (IP address and port) in an encrypted byte array, decrypts it only in memory at runtime, and immediately uses it for network API calls.

The code shown implements a byte-wise XOR decryption loop that reconstructs an obfuscated C2 string (172.245.246.77:8080) at runtime. The malware dynamically loads WININET.dll to avoid static imports and uses the decrypted string as its network endpoint for command-and-control communication.



The screenshot displays the Immunity Debugger interface. The assembly window shows the following code:

```
0B70603D 46 inc esi
0B70603E 8B4484 10 mov eax, dword ptr ss:[esp+esi*4+10]
0B70603F 03F8 add edi, eax
0B706040 81E7 FF000080 and edi, 800000FF
0B706041 79 08 jns d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B706012
0B706042 4F dec edi
0B706043 81CF 00FFFFFF or edi, FFFFFFF0
0B706044 47 inc edi
0B706045 8AC8 mov cl, al
0B706046 8B448C 10 mov eax, dword ptr ss:[esp+edi*4+10]
0B706047 894484 10 mov dword ptr ss:[esp+esi*4+10], eax
0B706048 0FB6C1 movzx eax, cl
0B706049 89448C 10 mov dword ptr ss:[esp+edi*4+10], eax
0B70604A 8B4C84 10 mov ecx, dword ptr ss:[esp+esi*4+10]
0B70604B 03C8 add ecx, eax
0B70604C 81E1 FF000080 and ecx, 800000FF
0B70604D 79 08 jns d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B706039
0B70604E 49 dec ecx
0B70604F 81C9 00FFFFFF or ecx, FFFFFFF0
0B706050 41 inc ecx
0B706051 8A448C 10 mov al, byte ptr ss:[esp+ecx*4+10]
0B706052 3042B xor byte ptr ds:[ebx+ebp], al
0B706053 43 inc ebx
0B706054 3BD4 cmp ebx, edx
0B706055 72 A6 jb d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B705FEB
0B706056 5D pop ebp
0B706057 5F pop edi
0B706058 5E pop esi
0B706059 5B pop ebx
0B70605A 81C4 00040000 add esp, 400
0B70605B C2 0800 ret 8
0B70605C 55 push ebp
0B70605D 88EC mov ebp, esp
0B70605E 56 push esi
0B70605F FF75 10 push dword ptr ss:[ebp+10]
0B706060 8BF1 mov esi, ecx
0B706061 8B4D 08 mov ecx, dword ptr ss:[ebp+8]
0B706062 5575 0C push dword ptr ss:[ebp+4]
```

The memory dump window shows the following data:

Address	Hex	ASCII
004D43CA	00 00 00 00 00 00 E4 AD 51 56 42 8B 00 08 31 3	a.QVB...17
004D43DA	32 2E 32 34 35 2E 37 37 3A 38 30 33 36 2E 37 37	2.245.246.77:808
004D43EA	30 3A 31 1E 7C 1E 1E 1F 7C 52 65 6D 6F 74 65 7	0:1 ... Remote
004D43FA	1E 1E 1F 7C 31 7C 1E 1E 1F 7C 00 7C 1E 1E 1F 7	
004D440A	01 7C 1E 1E 1F 7C 01 7C 1E 1E 1F 7C 31 7C 1E 1	 1 ..
004D441A	1F 7C 31 30 30 30 30 30 7C 1E 1E 1F 7C 00 7C 1	.. 100000
004D442A	1E 1F 7C 38 7C 1E 1E 1F 7C 72 00 65 00 6D 00 6	.. 8 ... r.e.m.c
004D443A	00 6F 00 73 00 2E 00 65 00 78 00 65 00 00 00 7	..o.s...e.x.e...
004D444A	1E 1E 1F 7C 00 7C 1E 1E 1F 7C 00 7C 1E 1E 1F 7	
004D445A	30 7C 1E 1E 1F 7C 31 7C 1E 1E 1F 7C 00 7C 1E 1	0 ... Rmc-GSEGIF
004D446A	7C 1E 1E 1F 7C 31 7C 1E 1E 1F 7C 38 7C 1E 1E 1	... 1 ... 8 ..
004D447A	7C 6C 00 6F 00 67 00 73 00 2E 00 64 00 61 00 7	.. o.g.s...d.a.t
004D448A	00 00 00 7C 1E 1E 1F 7C 00 7C 1E 1E 1F 7C 00 7	
004D449A	1E 1E 1F 7C 00 7C 1E 1E 1F 7C 31 30 7C 1E 1E 1	 10 ..

Figure 10: Dynamic C2 domain decryption.

The online keylogger leverages the same structural design as its offline counterpart; however, rather than persisting captured keystrokes to disk, it transmits the data in real time to the command-and-control (C2) server, enabling immediate exfiltration and eliminating the need for local storage.

Each keystroke intercepted by the online keylogger is encapsulated into a structured, network-ready payload and transmitted through the malware's communication channel in real time, allowing the attacker to continuously monitor user activity while avoiding the creation of forensic artifacts on the victim's filesystem.

```

if ( *this
|| (ModuleHandleA = GetModuleHandleA(0), v3 = SetWindowsHookExA(13, fn, ModuleHandleA, 0), (*this = v3) != 0) )
{
do
{
if ( !SendMessageA(&Msg, 0, 0, 0) )
break;
TranslateMessage(&Msg);
DispatchMessageA(&Msg);
}
while ( *this );
return 0;
}
else
{
LastError = GetLastError();
v5 = sub_B71AF51(v8, LastError);
sub_B704C9E((int)&v7, aKeyloggerIniti, v5);
sub_B701F66(aE);
sub_B71A891();
sub_B701EEA(v8);
return 1;
}
}

```

0008E55 sub_B7099E4:32 (B709A55)

Figure 11: Online keylogger initialized.

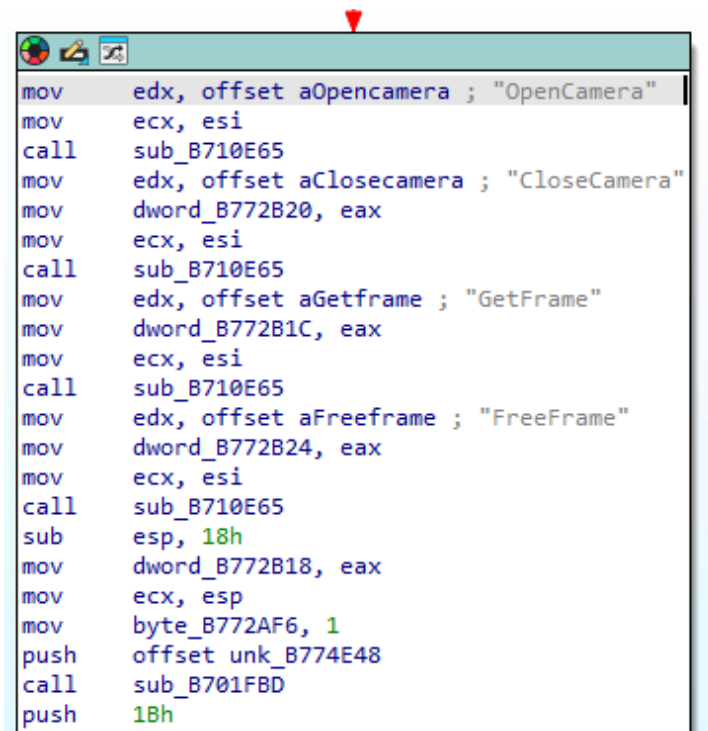
The malware hooks input events to capture user entered data and streams the collected content directly to its C2 infrastructure for immediate exfiltration.



Figure 12: Captured browsers input.

Remcos does not embed its webcam recording functionality in its main executable. Instead, it relies on an external module delivered as a Dynamic Link Library (DLL), which the malware retrieves from the command-and-control (C2) server when the attacker requests webcam access.

0B703F46	74 64	je d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B703FAC	
0B703F48	BA 64637608	mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B766364	B766364: "OpenCamera"
0B703F4D	8BC	mov ecx,esi	
0B703F4F	E8 11CF0000	call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B710E65	
0B703F54	BA 70637608	mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B766370	B766370: "CloseCamera"
0B703F59	A3 20287708	mov dword ptr ds:[B772820],eax	
0B703F5E	8BC	mov ecx,esi	
0B703F60	E8 00CF0000	call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B710E65	
0B703F65	BA 7C637608	mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76637C	B76637C: "GetFrame"
0B703F6A	A3 1C287708	mov dword ptr ds:[B77281C],eax	
0B703F6F	8BC	mov ecx,esi	
0B703F71	E8 EFCE0000	call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B710E65	
0B703F76	BA 88637608	mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B766388	B766388: "FreeFrame"
0B703F7B	A3 24287708	mov dword ptr ds:[B772824],eax	
0B703F80	8BC	mov ecx,esi	
0B703F82	E8 DECE0000	call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B710E65	



```
mov     edx, offset aOpencamera ; "OpenCamera"
mov     ecx, esi
call    sub_B710E65
mov     edx, offset aClosecamera ; "CloseCamera"
mov     dword_B772B20, eax
mov     ecx, esi
call    sub_B710E65
mov     edx, offset aGetframe ; "GetFrame"
mov     dword_B772B1C, eax
mov     ecx, esi
call    sub_B710E65
mov     edx, offset aFreeframe ; "FreeFrame"
mov     dword_B772B24, eax
mov     ecx, esi
call    sub_B710E65
sub     esp, 18h
mov     dword_B772B18, eax
mov     ecx, esp
mov     byte_B772AF6, 1
push    offset unk_B774E48
call    sub_B701FBD
push    18h
```

Figure 13: Webcam Capture Module Initialization.

Upon receiving webcam related command, Remcos performs the following steps internally:

The malware establishes a connection to the C2 server using its existing communication channel (typically over HTTP via WinINet). It then downloads the DLL payload into memory or saves it temporarily to disk, depending on the variant and configuration.

Instead of statically linking webcam functionality, Remcos loads the downloaded module at runtime using Windows API functions such as LoadLibrary.

After the DLL is loaded, Remcos resolves specific exported functions using GetProcAddress. These functions implement webcam-related operations such as:

- Initializing the video capture device,
- Capturing frames or video streams,
- Encoding or compressing the captured data,
- Returning the recorded output to the main Remcos process.

The main Remcos process invokes the exported webcam routine from the DLL. The module then interacts with Windows multimedia APIs to access the system's camera device and collect video or image data.

The recorded webcam output is passed back to the core Remcos process or directly transmitted by the module itself to the C2 server. The data is typically sent in chunks over the existing encrypted or obfuscated C2 channel.

FoxMailRecovery is typically the name of an exported function inside a credential-recovery plugin DLL.

8BD6	mov edx,esi	
8BC8	mov ecx,eax	
E8 E9E6FFFF	call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7108E5	
BA 08C87608	mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76C808	B76C808:"FoxMailRecovery"
A3 C82D7708	mov dword ptr ds:[B772DC8],eax	

Figure 14: Loading additional DLL.

The following code snippet matches the way Remcos constructs and sends command messages to its C2 server, specifically using a text-based command format. In Remcos, commands and responses are often wrapped inside string tokens (for example: /text "...") before being transmitted.

When invoked with the /sext option, the malware writes execution results to a temporary file with a randomly generated name in its working directory. The stored data is subsequently retrieved and exfiltrated to the C2 server, after which the file is deleted to minimize on-disk traces.

```

push    eax
push    offset aStext    ; " /stext \"
call    sub_B703B40
push    eax                ; int
lea     edx, [esp+3A0h+Filename] ; String
lea     ecx, [esp+3A0h+var_344] ; int
call    sub_B703CBB
pop     ecx
mov     edx, eax
lea     ecx, [esp+39Ch+var_32C]
call    sub_B703CDC
pop     ecx

```

Figure 15: Steal command.

How Remcos does Persistence

Remcos establishes persistence so that it is relaunched automatically after a system reboot or user logon. It supports several persistence mechanisms, chosen based on its configuration and privileges.

Remcos creates a named mutex to: Rmc-GSEGIF

- Prevent multiple instances from running
- Ensure only the persistent copy stays active

This supports persistence stability rather than startup persistence directly.

Registry Editor

File Edit View Favorites Help

Computer\HKEY_CURRENT_USER\SOFTWARE\Rmc-GSEGIF

Name	Type	Data
(Default)	REG_SZ	(value not set)
exepath	REG_BINARY	8b 81 18 00 ad 77 da 47 a6 b9 e9 c0 8e 98 02 84 0d 6...
licence	REG_SZ	315B15223D2178178D609A4886419048
time	REG_DWORD	0x6911ca26 (1762773542)
UID	REG_DWORD	0xba2b91a3 (3123417507)

Computer

- HKEY_CLASSES_ROOT
- HKEY_CURRENT_USER
 - AppEvents
 - Console
 - Control Panel
 - Environment
 - EUDC
 - Keyboard Layout
 - Network
 - Printers
 - Remote
 - SOFTWARE
 - 7-Zip
 - AI_RecycleBin
 - AppDataLow
 - b019ab1f3ab9
 - BreakPoint
 - BreakPoint License M
 - Classes
 - Fairdell
 - Google
 - Hex-Rays
 - HSTools
 - Microsoft
 - NTCore
 - Policies
 - Python
 - QtProject
 - RegisteredApplicatio
 - Rmc-GSEGIF
 - Scouter Software
 - Sysinternals
 - WinRAR

3:36:5... d1b76d4780e3... 9156 RegOpenKey HKCU\Software\Rmc-GSEGIF\ SUCCESS
 3:36:5... d1b76d4780e3... 9156 RegSetInfoKey HKCU\SOFTWARE\Rmc-GSEGIF SUCCESS
 3:36:5... d1b76d4780e3... 9156 RegQueryValue HKCU\SOFTWARE\Rmc-GSEGIF\override NAME NOT FOUND
 3:36:5... d1b76d4780e3... 9156 RegCloseKey HKCU\SOFTWARE\Rmc-GSEGIF SUCCESS
 3:36:5... d1b76d4780e3... 9156 Thread Create SUCCESS
 3:36:5... d1b76d4780e3... 9156 Thread Exit SUCCESS
 3:37:0... d1b76d4780e3... 9156 Thread Create SUCCESS
 3:37:0... d1b76d4780e3... 9156 Thread Exit SUCCESS
 3:37:0... d1b76d4780e3... 9156 CreateFile C:\ProgramData\colors\logs.dat SUCCESS
 3:37:0... d1b76d4780e3... 9156 QueryStandard... C:\ProgramData\colors\logs.dat SUCCESS
 3:37:0... d1b76d4780e3... 9156 CloseFile C:\ProgramData\colors\logs.dat SUCCESS
 3:37:0... d1b76d4780e3... 9156 Thread Create SUCCESS
 3:37:0... d1b76d4780e3... 9156 Thread Exit SUCCESS

Desired Access: Read
 KeySetInformationClass: KeySetHandle
 Length: 16
 Thread ID: 6628
 Thread ID: 6628, User Time: 0.000000
 Thread ID: 6924
 Thread ID: 6924, User Time: 0.000000
 Desired Access: Generic Read, Disposi
 AllocationSize: 4,096, EndOfFile: 3,788,
 Thread ID: 6972
 Thread ID: 6972, User Time: 0.000000

Figure 16: Remcos Persistence.

How Remcos Does Cleanup Procedure

After exfiltrating sensitive information, Remcos performs a clean-up routine by deleting browser cookies and stored browsing data. This tactic removes traces of its activity, disrupts forensic analysis, and diminishes evidence of credential theft or session hijacking, ultimately making it more difficult for investigators or victims to determine what information was compromised.

Remcos deletes all the recording files (keylogging, screenshots, and audio recordings).

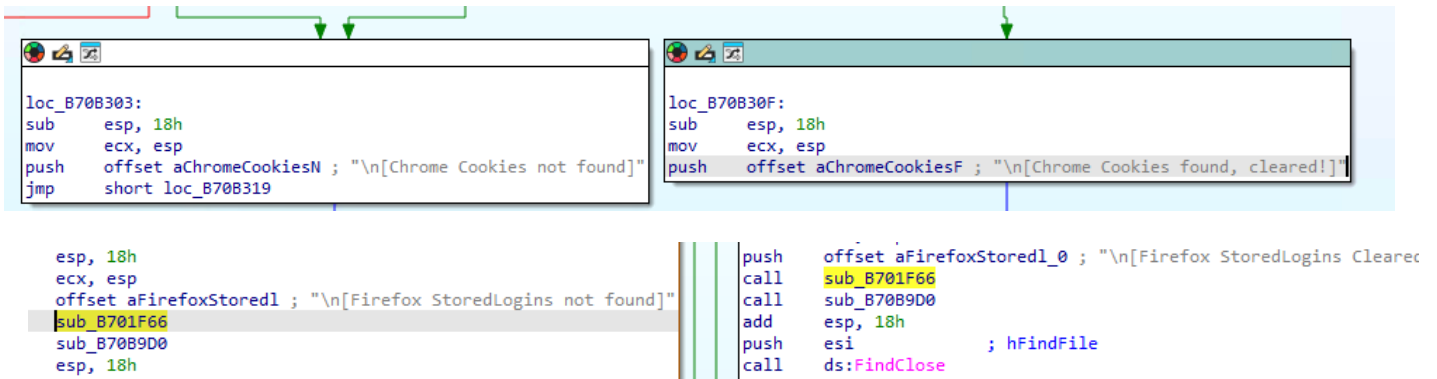


Figure 17.: Clear cookies, browser data.

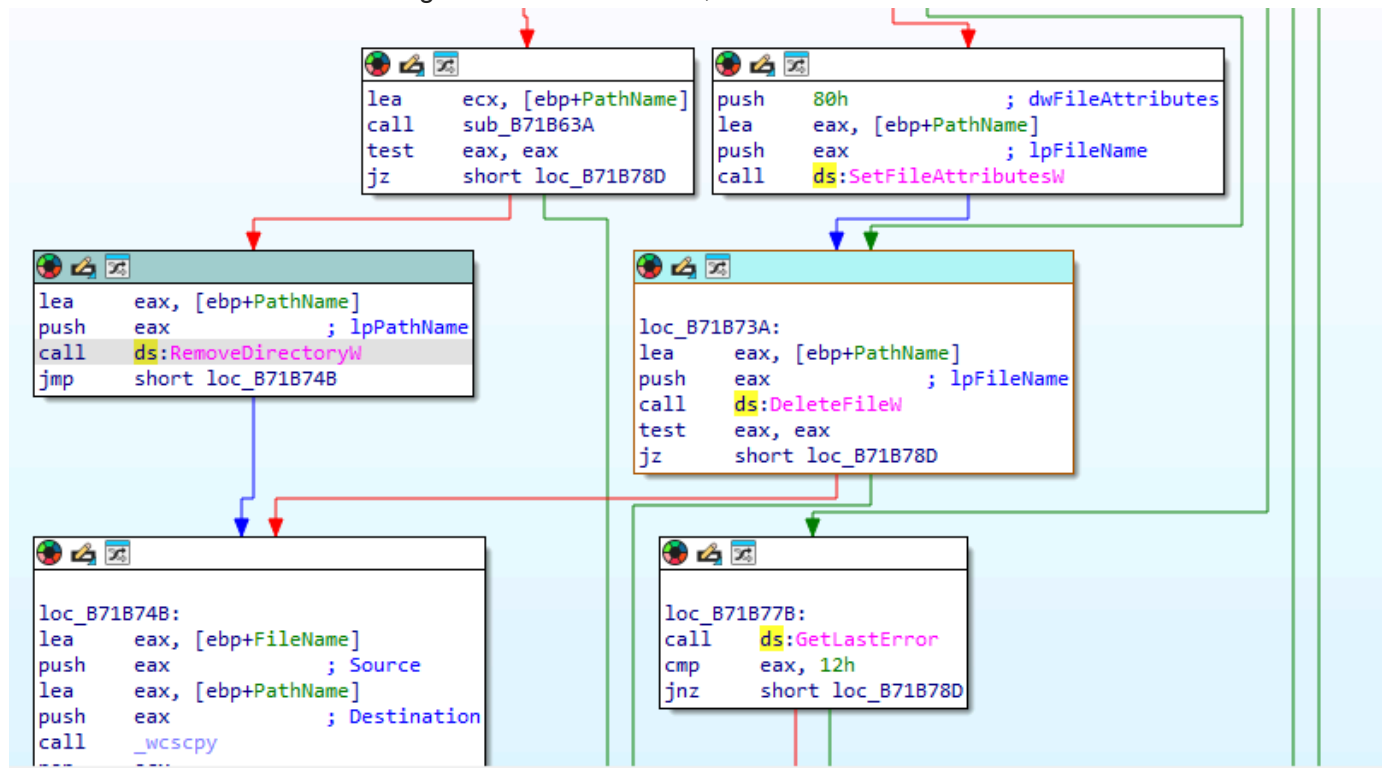


Figure 18: Delete Files & Folder.

After completing its data theft and clean-up routines, Remcos removes the registry keys it previously created for persistence. By deleting these auto-run entries, the malware attempts to erase evidence of how it maintained long-term access, making forensic analysis and remediation more difficult.

<pre> BF F070760B mov edi, d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B76 381D 892D770B cmp byte ptr ds:[B772D89],b1 75 15 jne d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B70C303 88CE mov ecx, esi E8 1458FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B701E0 50 push eax 88D7 mov edx, edi B9 01000080 mov ecx, 80000001 E8 52670000 call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B712A5 59 pop ecx BD 02000080 mov ebp, 80000002 381D 762B770B cmp byte ptr ds:[B772B76],b1 75 12 jne d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B70C32C 88CE mov ecx, esi E8 F25AFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B701E0 50 push eax 88D7 mov edx, edi 88CD mov ecx, ebp E8 33670000 call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B712A5 59 pop ecx 381D 722B770B cmp byte ptr ds:[B772B77],b1 75 15 jne d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B70C33D 88CE mov ecx, esi E8 D85AFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B701E0 50 push eax BA 5071760B mov edx, d1b76d4780e3ea2e60c7fd33788a09c5d2900227.b1n.B76 88CD mov ecx, ebp </pre>	<pre> B7670F0:L"Software\Microsoft\Windows\CurrentVersion\Run\\" B767150:L"Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run\\" </pre>
--	--

Figure 19: Delete Persistence.

Finally, the malware removes its installation artifacts by generating and executing a temporary Visual Basic script update.vbs in the %TEMP% directory using a randomly generated filename. This script deletes the original files, after which the malware terminates its own process.

<pre> 50 push eax FFD6 call esi 68 8073760B push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767380 68 F071760B push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7671F0 E8 27EA0200 call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B73AE1F 59 pop ecx </pre>	<pre> esi:EntryPoint B767380:L"\update.vbs" B7671F0:L"Temp" </pre>
---	--

Figure 20: Update.vbs added in the temp folder.

Below is the generated script with comments

Creating a FileSystemObject and Loop while the specified file exists.

Set fso = CreateObject("Scripting.FileSystemObject")

<pre> E8 205DFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767200 68 0072760B push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767200 8D4C24 14 lea ecx,dword ptr ss:[esp+14] E8 3FAFFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76726C 50 push eax BA 6C72760B mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B76726C 8D4C24 2C lea ecx,dword ptr ss:[esp+2C] E8 AB7BFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767298 59 pop ecx 8D4C24 10 lea ecx,dword ptr ss:[esp+10] E8 F95CFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7672A0 80E3 01 and bl,1 74 57 je d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767298 68 9872760B push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767298 8D4C24 A4000000 lea eax,dword ptr ss:[esp+A4] 50 push eax 8D4C24 60 lea ecx,dword ptr ss:[esp+60] E8 0B7AFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7672A0 50 push eax 8A A072760B mov edx,d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7672A0 8D4C24 48 lea ecx,dword ptr ss:[esp+48] E8 777BFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7672A0 </pre>	<pre> B767200:L"Set fso = CreateObject(\"Scripting.FileSystemObject\")\n\" B76726C:L\"On Error Resume Next\n\" B767298:L\"()\n\" B7672A0:L\"while fso.FileExists(\"\" </pre>
---	--

Figure 21: Creates FileSystemObject.

Create a WScript.Shell object and run a command in the command prompt and Create a FileSystemObject and delete the script itself.

- CreateObject("WScript.Shell").Run "cmd /c \"\""
- CreateObject("Scripting.FileSystemObject").DeleteFile(Wscript.ScriptFullName)

<pre> 68 8073760B push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B767380 8D8D 58FFFFF lea ecx,dword ptr ss:[ebp-A8] E8 C973FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B703B 8B00 mov edx,eax 8D8D 70FFFFF lea ecx,dword ptr ss:[ebp-90] E8 4861FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B702B 59 pop ecx 8B00 mov edx,eax 8D4D D0 lea ecx,dword ptr ss:[ebp-30] E8 D160FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B702B 59 pop ecx 8B00 mov edx,eax 8D4D B8 lea ecx,dword ptr ss:[ebp-48] E8 3561FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B702B 59 pop ecx 50 push eax 8D4D E8 lea ecx,dword ptr ss:[ebp-18] E8 3F66FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B702D 8D4D B8 lea ecx,dword ptr ss:[ebp-48] E8 6756FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E 8D4D D0 lea ecx,dword ptr ss:[ebp-30] E8 5F56FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E 8D8D 70FFFFF lea ecx,dword ptr ss:[ebp-90] E8 5456FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E 8D8D 58FFFFF lea ecx,dword ptr ss:[ebp-A8] E8 4956FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E 8D8D 40FFFFF lea ecx,dword ptr ss:[ebp-C0] E8 3E56FFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B701E 68 1074760B push d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7674 8D4D E8 lea ecx,dword ptr ss:[ebp-18] E8 F0BAFFFF call d1b76d4780e3ea2e60c7fd33788a09c5d2900227.bin.B7082 </pre>	<pre> B767380:L"CreateObject(\"WScript.Shell\").Run \"cmd /c \"\"\" edx:EntryPoint edx:EntryPoint edx:EntryPoint [ebp-18]:EntryPoint B767410:L"CreateObject(\"Scripting.FileSystemObject\").DeleteFile(Wscript.ScriptFullName) [ebp-18]:EntryPoint </pre>
--	---

Figure 22: Delete FileSystemObject and delete script itself.

Remcos Network Interaction with C2 Servers

Remcos initiates outbound connections to its command-and-control (C2) infrastructure either hardcoded or decrypted at runtime using HTTP or TCP. It regularly sends beacon signals to request instructions, processes remote commands, and exfiltrates stolen data such as keystrokes and credentials. The transfer occurs through a custom protocol that is encrypted or obfuscated, deliberately crafted to resemble normal web traffic and thereby evade detection.

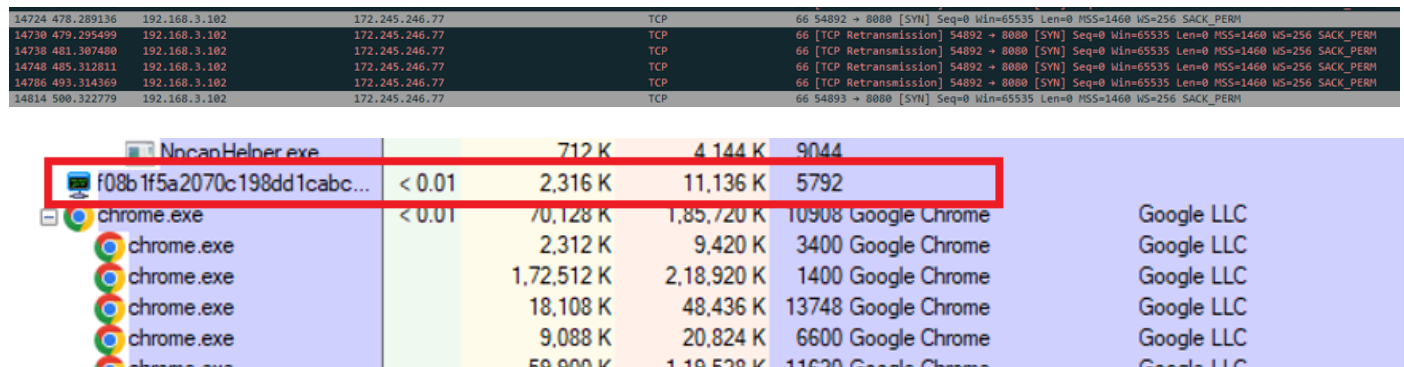


Figure 23: Network Interaction with C2 Servers.

How Can Remcos RAT be Removed?

- 1. Reboot into Safe Mode with Networking.
- 2. Use [UltraAV](#) antivirus to delete malicious files.
- 3. Detected as the following name by [UltraAV](#): Trojan.W32.111125.Remcos.YR.

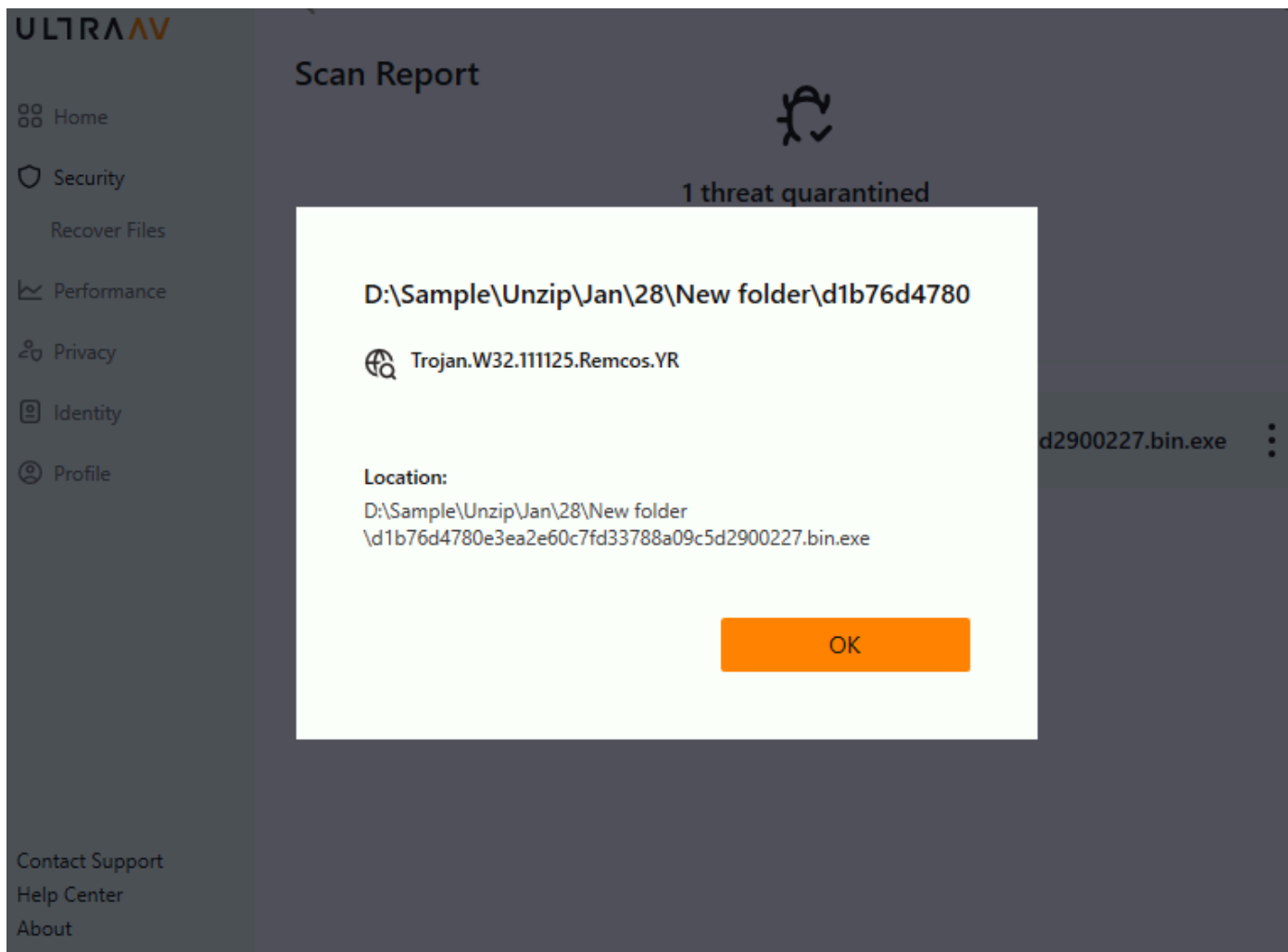


Figure 24: Threat Detection Name.

MITRE ATT&CK

ID	Technique	Description
T1123	Audio Capture	Capture data from the system's microphone
T1027	Obfuscated Files or Information: Dynamic API Resolution	Resolves API functions at runtime instead of declaring them in the import table to hinder analysis and detection.
T1547.001	Registry Run Keys	Remcos can add itself to the Registry key for persistence.
T1115	Clipboard Data	Steals and Copies data from the clipboard.
T1056.001	Input Capture: Keylogging	Remcos has a command for keylogging.
T1113	Screen Capture	Taking automated screenshots of the infected machine

T1125	Video Capture	Remcos can access a system's webcam and take pictures.
T1055	Process Injection	Remcos has a command to hide itself through injecting into another process.

Conclusion

The latest Remcos variants demonstrate a continued evolution in both stealth and functionality. These newer builds employ stronger obfuscation, runtime decryption of configuration data, and modular plugin-based capabilities to hinder static analysis and detection. Enhanced command-and-control mechanisms, dynamic DLL loading for extended features such as credential theft and webcam access, and improved cleanup routines further indicate an emphasis on evasion and operational flexibility. Overall, the persistence of Remcos and the steady refinement of its techniques highlight its ongoing effectiveness as a remote access trojan.

Indicators of compromise

MD5	c748c6a74389e4c48f9a34acba1ff093
Mutex	Rmc-GSE GIF
C2 Domain	172.245.246.77:8080
File Artifact	C:\ProgramData\colors\logs.dat