

TIDRONE Targets Military and Satellite Industries in Taiwan

: 9/6/2024

APT & Targeted Attacks

Our research reveals that an unidentified threat cluster we named TIDRONE have shown significant interest in military-related industry chains, particularly in the manufacturers of drones.

By: Pierre Lee, Vickie Su September 06, 2024 Read time: 6 min (1589 words)

Summary

- TIDRONE, an unidentified threat actor linked to Chinese-speaking groups, has demonstrated significant interest in military-related industry chains, especially in the manufacturers of drones' sector in Taiwan.
- The threat cluster uses enterprise resource planning (ERP) software or remote desktops to deploy advanced malware toolsets such as the CXCLNT and CLNTEND.
- CXCLNT has basic upload and download file capabilities, along with features for clearing traces, collecting victim information such as file listings and computer names, and downloading additional portable executable (PE) files for execution.
- CLNTEND is a newly discovered remote access tool (RAT) that was used this April and supports a wider range of network protocols for communication.
- During the post-exploitation phase, telemetry logs revealed user account control (UAC) bypass techniques, credential dumping, and hacktool usage to disable antivirus products.

Introduction

Since the beginning of 2024, we have been receiving incident response cases from **Taiwan**. We track this unidentified threat cluster as **TIDRONE**. Our research reveals that the threat actors have shown significant interest in military-related industry chains, particularly in the manufacturers of drones. Furthermore, telemetry from [VirusTotal](#) indicates that the targeted countries are varied; thus, everyone should stay vigilant of this threat.

This report also investigates the latest TTPs and the evolution of tools like **CXCLNT** and **CLNTEND**, presenting the attack chain to illustrate the threat actor's behavior within victims' systems. The TTPs confirm that the threat actors have consistently updated their arsenal and optimized the attack chain. Notably, anti-analysis techniques are employed in their loaders, such as verifying the entry point address from the parent process and hooking widely-used Application Programming Interfaces (APIs) like **GetProcAddress** to alter the execution flow.

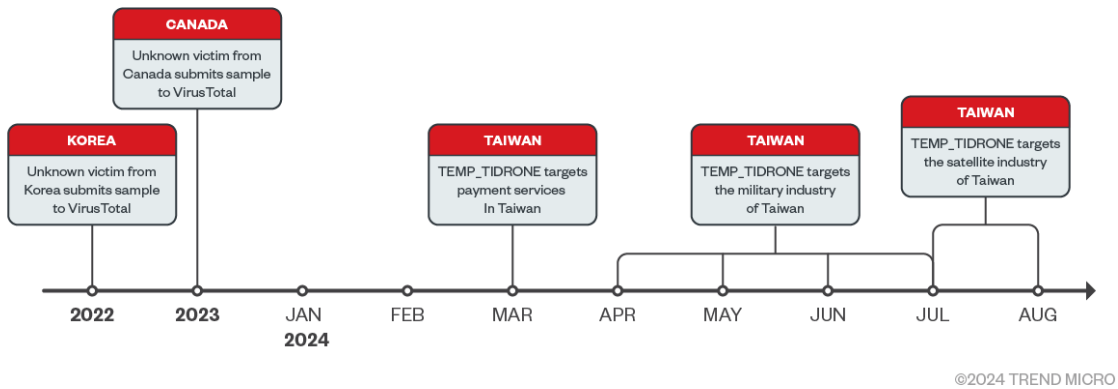


Figure 1. Timeline of campaign launched by TIDRONE

Execution Flow

Based on how the malware set was deployed into the victim's environment, we can infer that it likely leveraged other tools to infiltrate the system some time ago and has now progressed to the lateral movement stage.

In this case study we reviewed **CXCLNT/CLNTEND** and its related components, including the launcher and a legitimate executable for side-loading, which were downloaded via UltraVNC, a program that allows users to remotely control the server computer using their mouse and keyboard. During our investigation, we discovered the same ERP system was present in the environments of different victims, suggesting that the malware might be distributed through a supply chain attack.

After executing **winsrv.exe**, the malware copies the token from **Winlogon.exe** to escalate privileges and perform malicious activities. The original **Update.exe** located in a chosen directory is then replaced with one provided by the threat actors. During the post-exploitation phase, we observed **UAC Bypass**, credential dumping, and usage of commands aimed at **disabling antivirus products**, as seen in the telemetry logs.

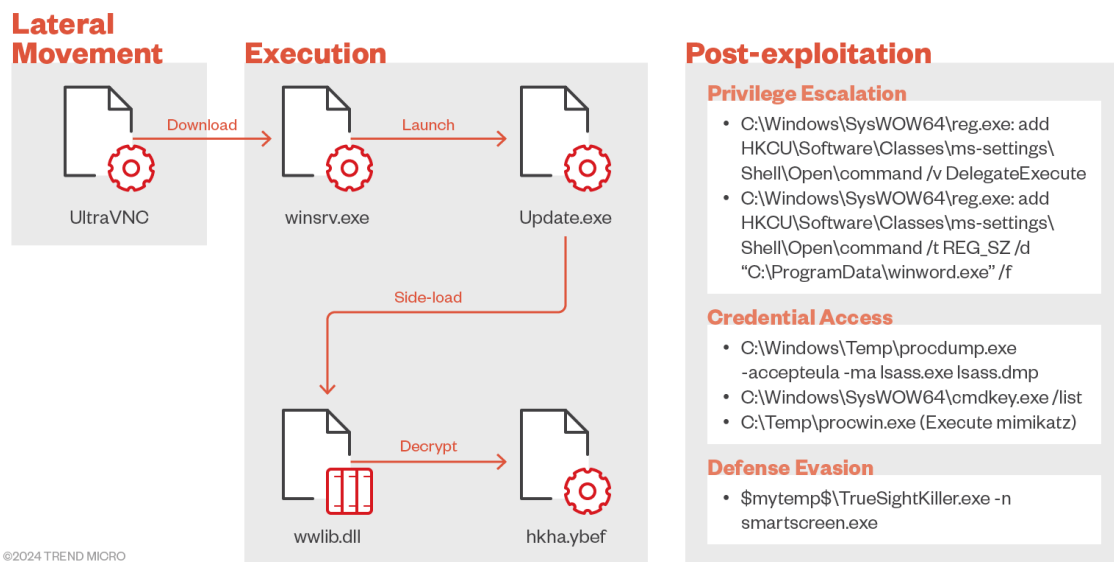


Figure 2. Execution flow of CLNTEND

Technical Analysis

This section will further explain the toolsets this threat cluster uses, such as malware **CXCLNT** and **CLNTEND**, and the TTPs we found between previous activities (A) and recent activities (B) as shown in the illustration below.

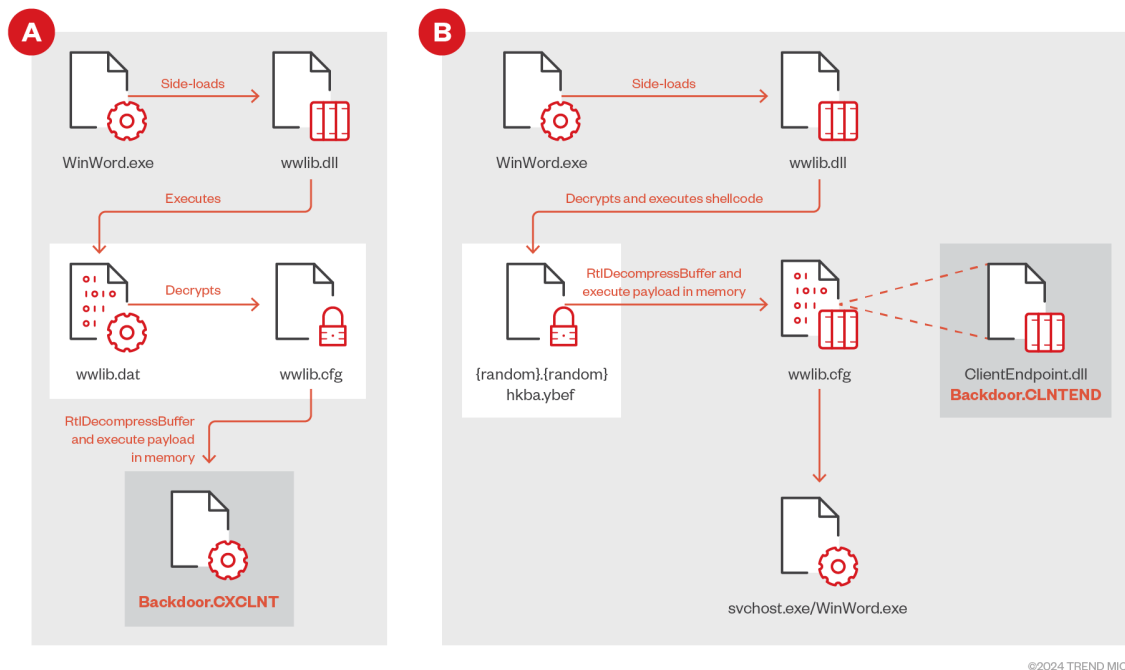


Figure 3. Execution flow between previous and recent activities involving CXCLNT and CLNTEND

Loaders

We have identified two kinds of loaders from VirusTotal and the client side, which each correspond to one infection chain. In this first version (A in Figure 3), the loaders create a service with the name for persistence on the victim side (the service name in our received case is ASProxys) and set '-s' for the first argument in the command line to make sure the process will be started from the service in the next execution. In decryption, the loaders need two payloads (wwlib.dat, and wwlib.cfg). The former is the shellcode with a decryption routine to decrypt the later one.

However, these two payloads are merged into a single encrypted payload in the second version (B in Figure 3) but keep the decryption routine in the shellcode part:

- RtlDecompressBuffer to decrypt the PE file.
- Execute the entry point of the PE file.
- Execute the assigned export function of the PE file. ("TgSetup" in the first version, "InstallSetup" in the second version)

Additionally, some extra features are implemented in the second version's loader:

Anti-Analysis technique

1. Read the entry point to check the parent process.
2. Hook GetProcAddress by overwriting the code inside the original one.

Anti-Antivirus: API with callback

The loaders do not utilize the common API to start a new thread, like **CreateThread** and **_beginthread**. Here are the steps to introduce this unseen method:

1. *ConvertThreadToFiber* to build a Fiber structure.
2. *CreateFiber* to enter a new thread (but the input address is junk code).
3. Overwrite the address in the Fiber structure (+0xC4) by the desired function.
4. Enter the desired function by API, *SwitchToFiber*.

```

BOOL __stdcall DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
{
    HMODULE ModuleHandleA; // eax
    char *Fiber; // eax

    if ( fdwReason == 1 )
    {
        hModule = hinstDLL;
        ModuleHandleA = GetModuleHandleA(0);
        dword_100132F8 = *(_DWORD*)((char *)ModuleHandleA + *(_DWORD *)ModuleHandleA + 15) + 40);
        lpFiber = ConvertThreadToFiber(0);
        Fiber = (char *)CreateFiber(0, StartAddress, 0);
        dword_100132BC = (int)Fiber;
        *(_DWORD *)&Fiber[(dword_100132F8 ^ 0x10EC) + 0x4] = (char *)sub_10001490 + (dword_100132F8 ^ 0x10EC);
        SwitchToFiber(Fiber);
    }
    return 1;
}

```

Figure 4. API with callback: overwriting the Fiber structure

Backdoor

According to our observation, we have identified two different backdoors in this campaign. Due to the code structure, the shellcode is flexible enough to accept various formats of files such as the exe and dll for the final payload.

EXE (Backdoor.CXCLNT)

One of the final payloads is a non-landed executable that collects the victim's information such as IP, MAC address, Computer Name, Product Name, and system architecture. By analyzing the packet transmission contents, we can infer the packet format.

There will be two decryption steps: the first key will be placed at position [start+0x8], and the second key will be placed at position [start+0x12]. Then, through a customized XOR encryption and decryption process, this encrypted packet can be successfully deciphered.

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	56	32	6F	55	2D	9E	16	D7	90	key1	EB	47	key2	00			V2oU-ž.*.VHëG...
00000010	9D	8C	7C	8C	AC	95	2F	B4	7C	8C	9C	10	9D	8C	7C	8C	.E E-/' E..E E
00000020	FE	0D	E0	CA	E4	71	D7	92	C8	92	1A	96	D6	42	50	2A	p.àÈäq×'È'.-ÖBP*
00000030	A7	FC	45	8C	70	62	06	8A	94	EE	66	02	FA	F5	95	50	\$UEEpB.Š"if.úð•P
00000040	65	55	D4	41	57	A6	C0	BB	C7	62	F8	94	F6	82	EC	0B	eUÔAW;À»Çbø"ø,i.
00000050	65	F9	E5	65	CC	63	BA	4B	B4	2E	DA	23	46	34	B7	4B	eùâeİc°K'.Ú#F4·K
00000060	98	23	31	C0	37	0F	08	19	91	BE	2A	54	B0	C4	A6	92	~#1À7... '¼*T°Å;'
00000070	EC	CD	42	CE	8B	A2	03	42	15	34	80	70	55	01	41	46	İİBİ<ø.B.4EpU.AF
00000080	6C	1D	EE	FF	FF	B8	E6	90	A4	E1	F3	B3	0C	9A	D2	78	l.İÿÿ,æ.µáó³.šöx
00000090	36	FC	75	39	71	BC	9E	6F	26	71	92	6C	2E	DC	79	7F	6úu9q¼žø&q'1.Ûÿ.
000000A0	1C	65	17	E1	7C	A0	81	5F	8F	34	35	3E	03	D3	17	57	.e.á ._.45>.Ó.W
000000B0	E8	D4	52	18	9C	7E	52	60	10	8D	89	C4	9A	F1	9A	56	èÖR.α~R'...%ÄšñšV
000000C0	4F	B8	FC	54	7D	C1	C2	1F	A4	EE	26	BC	E9	C8	56	CB	O,úT)ÁÁ.µi&¼éÉVÈ
000000D0	B6	3E	9A	7B	C7	42	6F	D3	DD	3C	76	84	BA	31	26	77	¶>š(ÇBoÓÝ<v,,°1&w
000000E0	B3	39	9C	D4	81	A0	3F	5B	C4	6D	DA	23	45	31	79	53	²9αÔ. ?[ÄmÚ#E1ÿS
000000F0	B4	55	D5	EA	AC	08	4D	17	A4	9E	1A	50	D4	32	EA	98	'UÖë~.M.µž.PÔ2è~
00000100	19	91	87	F5	5B	59	35	25	A3	BD	A5	1C	24	66	DA	FF	. 'š[ÿ5%£¼¥. \$fÛÿ
00000110	78	C5	71	D8	E8	78	F1	CF	63	79	67	62	F8	0A	76	73	xÄqðèxñİcygbø.vs
00000120	13	5E	65	9D	DD	12	FA	19	68	2D	D5	C1	4B	FA	A2	FA	.^e.Ý.ú.h-ÖÁKúóú
00000130	D1	61	8F	CB	30	37	A9	50	80	9C	D6	E4	7E	80	7B	7A	Ña.È07@P@αÖä~€(z
00000140	29	8F	DB	AF	F3	FD	06	3E	E8	86	08	71	E5	F9	C9	D6).Û-óÿ.>èt.qáùÈÖ
00000150	72	C2	55	58	3B	53	36	A5	31	E4	B6	A7	2C	67	F7	EF	rÄUX;S6W1ä¶S,g÷İ
00000160	A0	DD	52	DE	0F	7C	62	B5	2C	90	AA	BA	8D	09	99	97	ÝRß. bµ,.°°..µ-
00000170	04	61	B1	AE	20	0F	9F	F7	98	AC	28	B4	17	64	32	A2	.a±@ .ÿ÷~-('d2ø
00000180	02	85	AF	96	B5	1D	9E	27	1A	EA	19	72	BC	84	8D	9E-µ.ž'.è.r¼,..ž
00000190	B1	45	33	11	28	82	0A	62	1D	14	73	F1	84	81	E5	AC	±E3.(, .b..sñ,..ä~
000001A0	30	76	4A	2E	EC	9D	CA	8F	51	9E	D6	10	20	7A	D1	98	0vJ.İ.È.QžÖ. zÑ~
000001B0	C4	0A	06	EC	D6	80	CB	F2	9D	A0	FE	82	4C	74	2D	68	Ä..iÖÈÈò. p,Lt-h
000001C0	10	AC	39	FD	A3	94	D2	38	46	BD	5B	23	A0	0E	BE	46	.-9ÿ£"ò8F¼[# .¼F
000001D0	1E	47	21	A4	53	A3	F2	0A	D4	20	6B	CF	2B	F2	FA	2C	.G!µS£ò.Ô kİ+òú,
000001E0	6E	9F	7D	7E	0E	D3	C8	DE	9F	C0	AC	DC	A9	A0	71	68	nÿ)~.ÖÈPÿÄ-Ûø qh
000001F0	D7	00	3A	75	1D	45	2E	46	8A	7F	03	4F	22	80	FD	82	×.:u.E.FŠ..0"€ÿ,
00000200	33	C3	35	B6	37	F5	27	53	A3	30	32	31	05	E5	32	B4	3Ä5¶7ð'S££021.â2[
00000210	A5	A0	0C	E0	1B	28	21	2F	A4	E1	12	15	89	A7	37	57	¥ .à.(!/µá..%S7W
00000220	FF	EE	12	9E	60	46	74	64	E7	11	46	03	86	C6	48	6D	ÿİ.ž'Ftdç.F.†ÆHm
00000230	4C	E6	46	04	C7	63	53	E0	55	00	D8	F2	99	A7	92	83	LæF.ÇcSàU.øðµS' f
00000240	70	3F	44	5E	DD	4E	BD	7B	67	03	2D	46	BF	8E	A2	27	p?D^ÿN¼(g.-F¿žø'
00000250	AD	C5	06	AF	B8	49	45	9D	C0	58	B3	47	B0	22	08	A2	.Á.-,IE.ÀX³G°".ø
00000260	95	64	90	1C	07	8E	9D	B0	3D	2A	2D	CE	4B	3E	2A	16	•d...ž.°=*-İK>*.
00000270	44	CC	DE	A4	EF	FA	9D	41	D2	AC	91	C3	C7	96	D6	AA	DİBµiú.Aò~'ÄÇ-Öª
00000280	BB	F5	F9	B8	E0	90	AD	F7	F3	BA	EA	9E	AF	E1	F5	B4	»ðù,à..÷ó°èž-áð'
00000290	EC	84	B9	FB	02	63	B8	19	92	24	82	EB	D4	00	7A	A4	ì,,²û.c.,.'š,èÖ.zµ

Figure 5. Encrypted traffic

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	04	10	00	00	00	00	00	00	CA	46	D5	67	47	08	00	00ÊFÖgG...
00000010	00	00	00	00	30	85	B2	38	00	00	00	00	00	00	00	000...°8.....
00000020	55	8A	EE	88	49	0D	5B	EE	44	0E	0A	0B	5A	3E	DC	B6	UŠi^I.[iD...Z>Üq
00000030	B7	61	C9	F0	FC	FE	16	17	18	92	EA	9E	EA	68	19	2C	·aÉšüp... 'ëžèh.,
00000040	E9	C9	C4	DC	DB	DA	4C	27	D7	FF	74	E8	7A	1E	FC	96	éÉÄÜÜÚL'xýtèz.ü-
00000050	E9	85	69	F9	DC	FE	36	37	38	B2	CA	BE	CA	48	3B	D7	é...iùÜp678°Ê%ÊH;*
00000060	88	BE	BD	BC	BB	93	18	84	1D	C2	A6	C8	A0	59	2A	EE	°44»"... .Ä;È Y*ï
00000070	60	51	52	53	07	DE	8F	DE	05	A9	0C	0C	D9	9D	51	DB	`QRS.Ð.Ð.®.Û.QÛ
00000080	E0	61	62	63	EF	25	6A	EC	28	7D	E3	2E	80	E6	5E	E4	àabcîšjì()ã.ëæ^ä
00000090	26	61	F9	45	FD	20	8E	F2	AA	0D	1E	F0	3E	41	F5	03	æaùÉý Žò°..š>Aš.
000000A0	90	F9	07	7C	F0	DC	0D	C3	9F	A9	B9	42	8F	4F	07	CA	.ù. šÜ.Äÿ®°B.O.Ê
000000B0	64	A8	DE	84	8C	E3	DE	1C	9C	11	99	59	16	8D	16	CA	d`P„GãP.æ.°Y...Ê
000000C0	5F	25	70	28	F1	5D	D2	82	28	92	AA	20	F9	55	DA	B7	šp(ñ]Ò, ('° àUÚ·
000000D0	3A	A2	8A	E6	4B	3E	E3	4F	CD	A1	FA	F8	36	AD	36	EA	:°ŠæK>ãOÍ;úø6.6ê
000000E0	3F	45	10	48	91	3D	B3	27	48	F1	CA	BE	C9	4D	F5	CF	?E.H'°='HñÊ%ÉMöï
000000F0	A4	C8	59	96	20	94	5D	8A	28	E2	96	CC	C4	AF	66	E4	æÈY- "]Š(á-ÏÄ`fä
00000100	95	0D	97	68	D7	25	B9	B9	B3	20	29	60	A8	FA	CA	62	·.-h×°°°°) ``úÈb
00000110	F4	B9	FD	44	F8	E5	7D	B3	EF	E5	77	FF	74	76	FA	EF	ó°ýDøá)°iáwýtúúí
00000120	03	C3	E9	E1	51	8E	EA	84	E4	51	59	5D	5B	67	2E	86	.ÄéáQžè„äQY] [g.†

Figure 6. Decrypted traffic

However, upon analyzing this sample, we found that the command and control (C&C) server was no longer active. Tracing the APIs used in malware and the recorded pcap file from the sandbox report, we inferred possible functionalities. By comparing the decrypted packet contents with the command codes hardcoded in the malware, we concluded that this backdoor might possess the following capabilities.

Command code	Description
0x1001	Send victim information to C&C server
0x1002	Pass but do nothing
0x1003	SetEvent
0x1004	Receive unknown data, while not being sure of the purpose
0x1005	Clear footprints and <ul style="list-style-type: none"> Delete files wwlib.cfg, wwlib.dat, and wwlib.dll Delete service
0x1006	Persistence via setting reg
0x2001	Receive the size of the payload from the C&C server
0x2002	Receive a dll file from the C&C server
0x2003	Call the export functions of the received dll from 0x2002
0x2004	UNKNOWN
0x2005	Check connections alive
0x2007	Send listed files in a specific folder to the C&C server

Table 1. Backdoor command code of CXCLNT

DLL (Backdoor.CLNTEND)

Another final payload is a non-landed dll with the internal name “install.dll”. In the export function, InstallSetup, there are three paths based on the value in configuration:

1. SvcLoad → Create a service with the name “CertPropSvce” and inject the next payload, **ClientEndPoint.dll**, into the current process or svchost process (Depending on configuration).
2. TaskLoad → Create a task with the name “CertificatePropagation” and inject the next payload, **ClientEndPoint.dll**, into the current process or svchost process (Depending on configuration).
3. Other: Directly inject the next payload, **ClientEndPoint.dll**, into the current process or svchost process (Depending on configuration).

ClientEndPoint.dll is a remote shell tool and observed commands are shown in Figure 3. It supports these protocols for communication with the C&C server:

- TCP
- HTTP
- HTTPS
- TLS
- SMB(port:445)

Based on our experience, threat actors prefer the C&C server domain with a misquoted name, like `symantecsecuritycloud[.]com`, `microsoftsvc[.]com`, and `windowawns[.]com`, whether it is for **CLNTEND** and **CXCLNT**. They all implement a similar naming convention to mislead the investigation for network infrastructure.

Attribution Analysis

The consistency in file compilation times and the threat actor's operation time with other Chinese espionage-related activities supports the assessment that this campaign is likely being carried out by an as-yet unidentified Chinese-speaking threat group. The incidents we observed were highly targeted and limited in scope. The focus on military-related industry chains, particularly in the manufacturers of drones, suggests an espionage motive, given the sensitive data these entities typically hold. This further reinforces the likelihood that **TIDRONE** is engaged in espionage-related activities.

WinWord.exe

Due to the same parent process (WinWord.exe) operation from threat actors, the organizations can defend against the attack from **TIDRONE** by staying vigilant of the following variations:

- WinWord.exe (sha256: 8cfb55087fa8e4c1e7bcc580d767cf2c884c1b8c890ad240c1e7009810af6736). Beware that it has the child process cmd.exe due to the remote shell functionality.
- WinWord.exe (sha256: 8cfb55087fa8e4c1e7bcc580d767cf2c884c1b8c890ad240c1e7009810af6736) with "-s" in the first argument of the cmd line.
- WinWord.exe (sha256: 8cfb55087fa8e4c1e7bcc580d767cf2c884c1b8c890ad240c1e7009810af6736) with "/SvcLoad" or "/TaskLoad" in the last argument of the cmd line.

Conclusion

In this article, we investigated **TIDRONE**, a threat actor linked to Chinese-speaking groups. The attacks were detected in Taiwan and mostly targeted military-related industries, specifically the manufacturer of drones. The activities involve advanced malware variants such as **CXCLNT** and **CLNTEND** which were spread through ERP software or remote desktops. We examined the technical details of these malicious activities to keep users informed about these types of threats.

Some of the steps that organizations can take to protect themselves are as follows:

- Download software only from trusted sources
- Stay vigilant of [social engineering lures](#) that threat actors could use as entry points for attacks
- Employ antimalware software that could detect early signs of compromise no matter where they are in the system

[Trend Micro Vision One](#) offers multilayered protection for diverse environments. With comprehensive prevention, detection, and response capabilities, it safeguards systems from breaches and attacks.

Indicators of Compromise (IOCs)

File

SHA-256	Detection
f13869390dda83d40960d4f8a6b438c5c4cd31b4d25def7726c2809ddc573dc7	Trojan.Win32.CXCLNT.ZTLH
e366f0209a939503418f2b7befbd60b79609b7298fed9c2fbafcb0e7fde19740	Trojan.Win32.CXCLNT.ZTLH
6cb08a458e35101ef1035e7926130e1394cc1764a10166628aff541834c67063	Trojan.Win32.CXCLNT.ZTLH
19bbc2daa05a0e932d72ecfa4e08282aa4a27becaabad03b8fc18bb85d37743a	Trojan.Win32.CXCLNT.ZTLH
eea0f94c6a8f18275c3dac1e1b9e9d3240e37073ff391852e8ff8d8391efa9aa	Trojan.Win32.CXCLNT.ZTLH
0d91dfd16175658da35e12cafc4f8aa22129b42b7170898148ad516836a3344f	Trojan.Win32.CXCLNT.ZTLH
1b08f1af849f34bd3eaf2c8a97100d1ac4d78ff4f1c82dbea9c618d2fcd7b4c8	Trojan.Win32.DULLOAD.ZTLC
4b5f609c6b6788bdf0b900dd3df3c982cd547e7925840000bdc4014f8a980070	Trojan.Win32.SHELLDEBIN.ZTLC
1f22be2bbe1bfcda58ed6b29b573d417fa94f4e10be0636ab4c364520cda748e	Backdoor.Win32.CXCLNT.ZTLC.en
3b8f10a780eb64a3c59a2ae85fec074faf0f1a8d9725fb111f5cbf80e7b0dc1b	Backdoor.Win32.CLNTEND.ZTLH.e
db600b0ae5f7bfc81518a6b83d0c5d73e1b230e7378aab70b4e98a32ab219a18	Backdoor.Win32.CLNTEND.ZTLH.e
1bf318c94fa7c3fb26d162d08628cef54157df2b2b36cf7b264e3915d0c3a504	Backdoor.Win32.CLNTEND.ZTLH.e
f3897381b9a4723b5f1f621632b1d83d889721535f544a6c0f5b83f6ea3e50b3	Backdoor.Win32.CLNTEND.ZTLH.e

Network

- bestadll[.]fghytr[.]com
- client[.]wns[.]windowswns[.]com
- server[.]microsoftsvc[.]com
- service[.]symantecsecuritycloud[.]com
- time.vmwaresync[.]com