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.

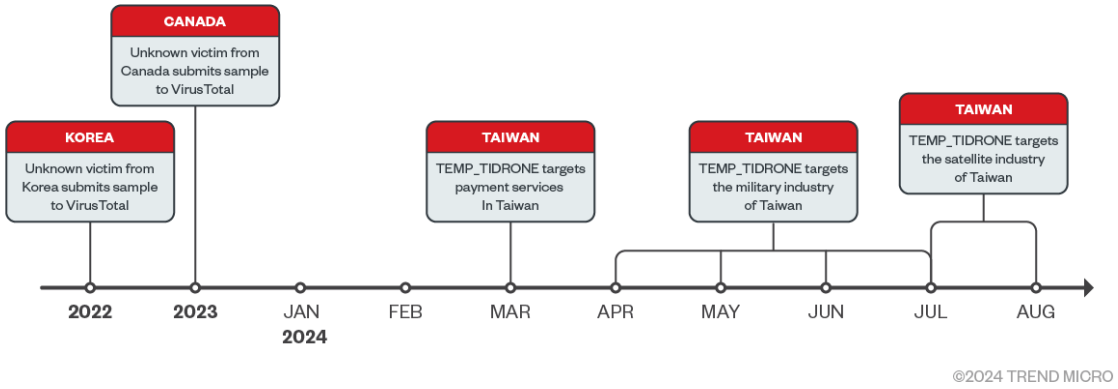

Figure 1. Timeline of campaign launched by TIDRONE

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.

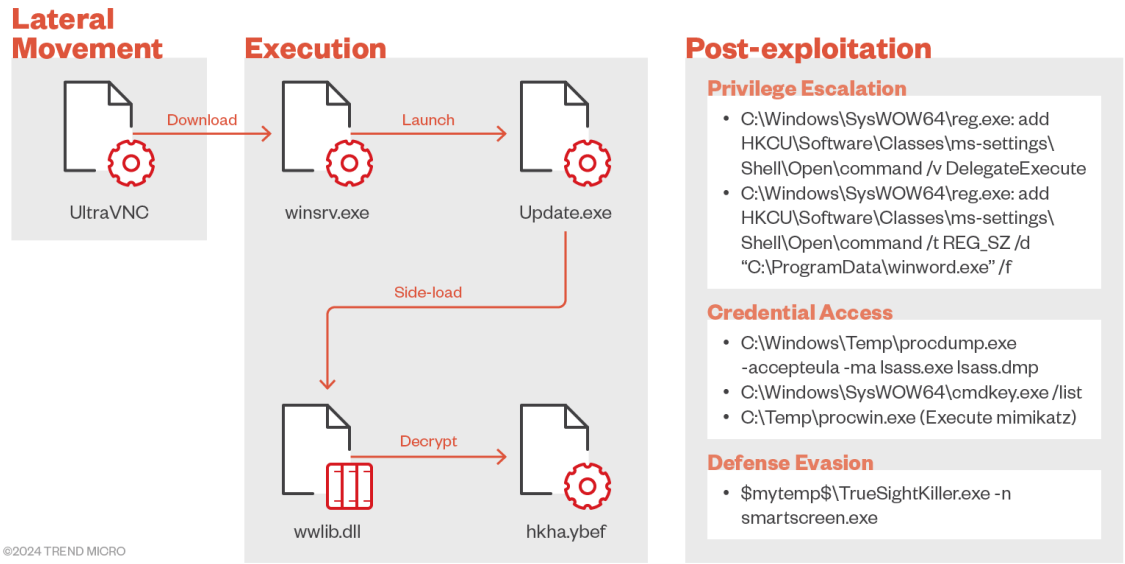

Figure 2. Execution flow of CLNTEND

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.

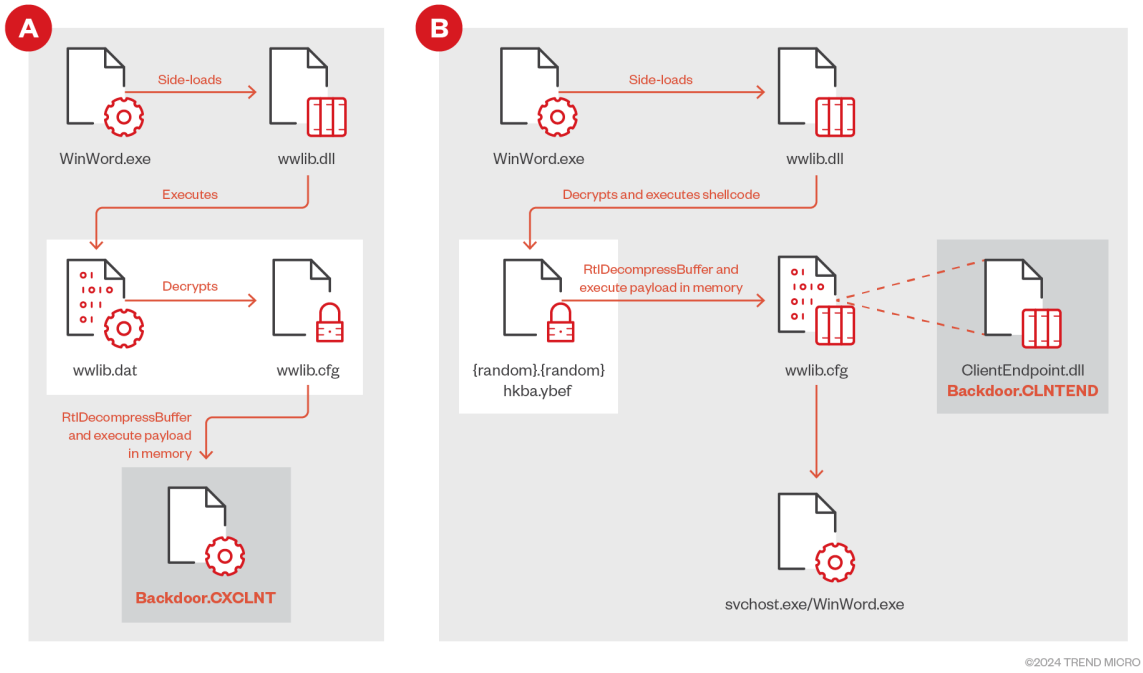

Figure 3. Execution flow between previous and recent activities involving CXCLNT and CLNTEND

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) + 0xC4] = (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.

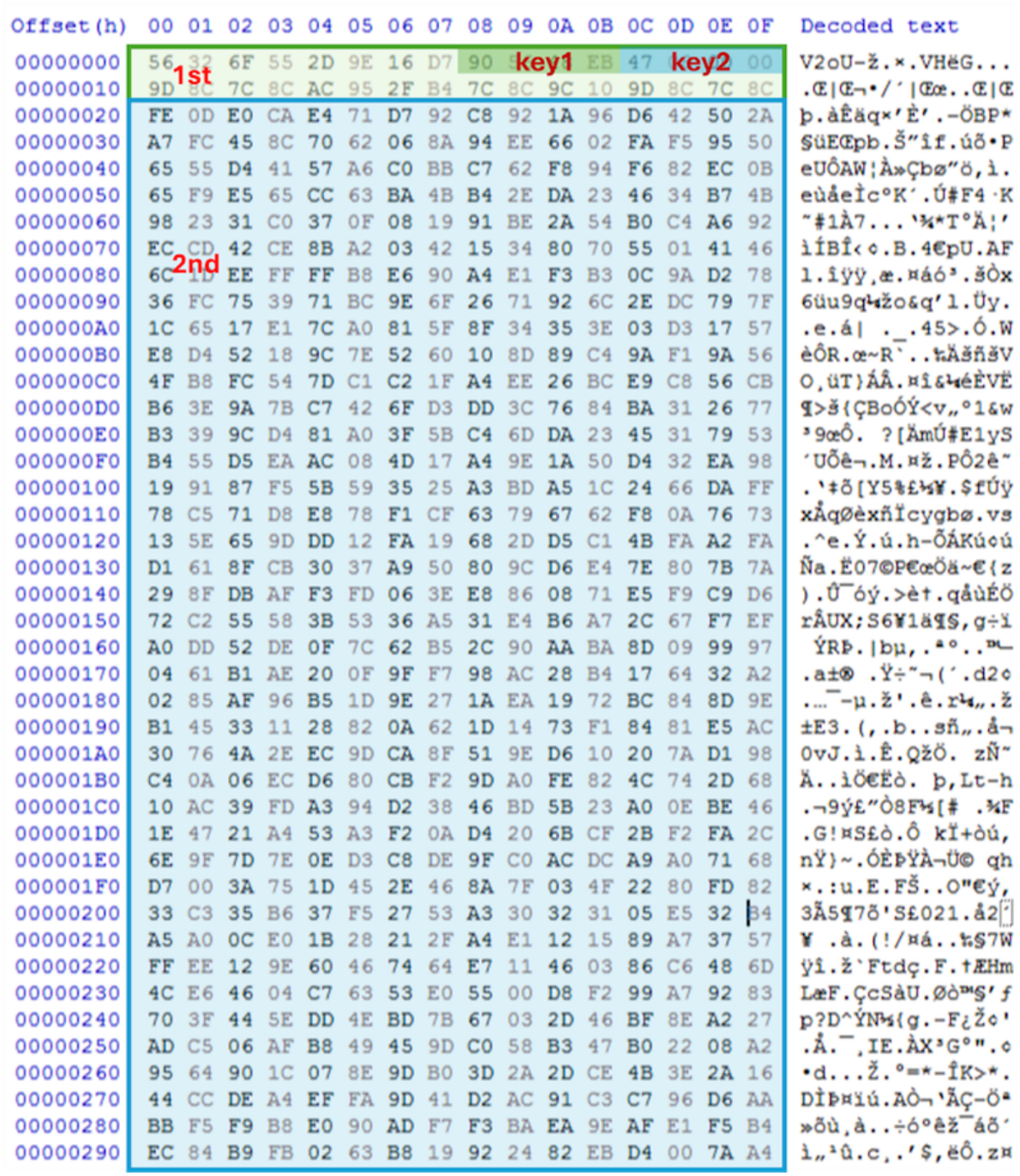

Figure 5. Encrypted traffic

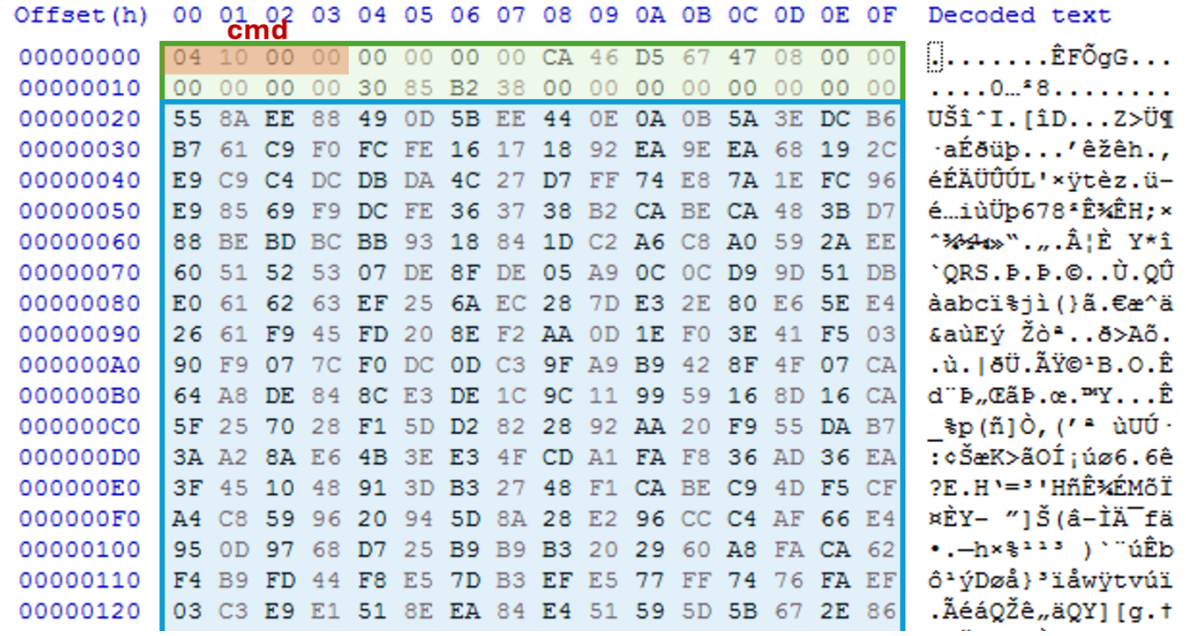

Figure 6. Decrypted traffic

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 • 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 “CertificatePropagatione” 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 windowswns[.]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
1bf318c94fa7c3fb26d162d08628cef54157dfeb2b36cf7b264e3915d0c3a504	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