What We Can Do against the Chaotic A41 APT Campaign

Hajime Yanagishita, Kiyotaka Tamada, You Nakatsuru, Suguru Ishimaru

2022/01/27
Who We Are

Hajime Yanagishita
Macnica

Kiyotaka Tamada
Secureworks

You Nakatsuru

Suguru Ishimaru
Kaspersky
Agenda

Recent A41APT campaign we've seen

- What A41APT Campaign is
- Continuous A41APT Campaign
  - Continued and Updated TTPs
  - News TTPs Observed in 2021
  - Attribution of The Threat Actor
- What We Can Do against the Chaotic A41APT Campaign
What A41APT Campaign is
A41APT Campaign

A sophisticated attack campaign revealed at JSAC 2021

- Period of activity
  - March 2019 to January 2021
- Target
  - Japan (Japanese companies and their overseas branches)
- Origin of the campaign name
  - Named after the actor’s host name DESKTOP-A41UVJV used for remote connection

Distinguish attack campaign that threat actor intrudes via internet-facing system, deploy malware such as SigLoader/Sodamaster from others
TTPs Reported at JSAC2021

**Initial Intrusion**
Penetrate via internet-facing system by using vulnerabilities or stolen credentials

**Internal Reconnaissance**
Perform network scan, connect to RDP server

**Persistence**
Install malware by using Taskscheduler

**Lateral Movement**
Connect to RDP server at HQ office

**Actions on Objectives**
Search for secret information, exfiltrate after encryption and compression

- SigLoader
- SodaMaster
- P8RAT
- Cobalt Strike
- FYAntiLoader
- csvde
- AdFind
- Scanner
- PowerShell
- WinRAR
- Pulse Connect Secure
Continuous A41APT Campaign in 2021
Continuous A41APT Campaign in 2021

- Observed attacks against multiple organizations in Japan and branch offices
  - We investigated each different incident to disclose updated TTPs and discovered new TTPs

- Branch
  - SigLoader
  - HUI Loader
  - Jackpot
  - ChinaChopper
  - SodaMaster
  - csvde
  - Mimikatz
  - Secretdump
  - PsExec

- HQ
  - WinRAR

- Pulse Connect Secure
  - FortiGate
  - Cisco AnyConnect
  - Microsoft Exchange Server
しかしこれぞれ観察されたSodaMasterのバックドアコマンドは、一部は未実装ですが、アルファベットの「C」から「X」までサポートされていました。またコマンドの分岐処理は、switch文を使わずにループ処理でコマンドハンドラを検索する変則的な手法が用いられていいました。

```c
if (g_handlers[0].func )
    id = 0164;
    func = &g_handlers[0].func;  
do {
    if ( &g_handlers[id].cmd == a1->cmd_id )
        (func)->a1(arg, (size_t)n);
    id = ++next_id;
    func = &g_handlers[next_id].func;  
} while (*func );
```

「Jackpot」のCommunication Protocol
トレンドマイクロで観察したJackpotは、ハードコードされたURLへのPOSTリクエストのみを処理します。それ以外のメソッドによるリクエストに対しては、正規の応答に見せかけたレスポンスを返します。攻撃者クライアントは、後述する独自のメッセージパケットをカスタムベース64とRC4で暗号化して送信します。この際、次のようなパスワードによる認証プロセスを経ることで、バックドア機能が有効化され、以降のバックドア処理が可能になる仕組みにいました。これは、不特定多数がアクセス可能な公開システムにJackpotを感染させることを想定し、意図しないリクエストを処理しないための予防策と考えられます。

POST https://example.com/default.aspx
1. パスワードを送信して認証リクエスト
2. ハードコードしたパスワードと比較して認証
3. パスワード認証機能の有効化
4. バックドアコマンドを含むリクエスト
5. バックドア処理

図8：Jackpotがバックドア機能を有効化するためのシーケンス例
Continued and Updated TTPs

Continuous A41APT Campaign in 2021
Intrusion via VPN Devices

Observations in 2021

- Using known vulnerabilities
  - Pulse Connect Secure
  - FortiGate: CVE-2018-13379
  - Cisco AnyConnect: CVE-2020-3125
- Even if it's patched now, the credentials from back then might have been leaked

Note that we've only seen the host name "DESKTOP-O2KM1VL" already reported in 2021
(Never seen "DESKTOP-A41UVJV" - origin of the campaign name)
Tool Sets Used After Intrusion

- Following tools were found in the lateral movement stage
  - Mimikatz
  - secretdump.py
  - PsExec
  - csvde
  - WinRAR

- The threat actor seems to use various tools as needed
Malware Updates

- SigLoader and SodaMaster are still used in 2021
  - Cobalt Strike, P8RAT and FYAntiLoader were not observed in 2021
- With some changes:
  - Tampering compile time
    - SigLoader: e.g. 2021/?? -> 2017/05
    - SodaMaster: e.g. 2021/04 -> 2012/10
  - Updates on major functions
    - SigLoader: decryption process
    - SodaMaster: commands and data format
SigLoader Execution Flow

Layer I
- Signed DLL with encrypted shellcode

Layer II
- Shellcode
- Decode & decrypt

Layer III
- Shellcode
- Decrypt & decode

Layer IV
- Signed DLL with encrypted shellcode

Layer V
- Payload
- Shellcode varies by payload

DLL Side-Loading
- Legitimate file
Decryption Process of SigLoader

- Algorithm identifiers were changed from string to number
  - 0: AES
  - 1: DES
  - 3: XOR
- The order of decryption algorithm is hardcoded

Layer I

```
cipher_id_18002DC70 dd 3 ; DATA XREF
  dd 0
  dd 35E1Dh
```

Layer III

```
cipher_id_1CBA0643F30 dd 0 ; DATA XREF:
  dd 3
  dd 0 = AES
```

The order is reversed between Layer I and Layer III in all analyzed samples.
Decryption Process of SigLoader

- The AES mode was changed from CBC to ECB
- The AES key is the first 16 bytes from the hardcoded 32 bytes string

```c
if ( 4 * a1[1] > 0 )
{
    v8 = a2;
    do
    {
        v9 = v8[aeskey - a2];
        ++v6;
        *v8++ = v9;
    } while ( v6 < 4 * a1[1] );
}
```

```
nuWU1hZsNRptORhw8iY4sYm0WQjbjB
```
SodaMaster Evolution

We classified 20+ samples into 3 versions, and confirmed 6 activities from compile time and common features as follows:

**Version 1**
- 2 commands (d,s)
- 32bit DLL
- HTTPs communication using wininet.dll
- Original name was httpsWin32.dll

**Version 2**
- 4 commands (d,f,l,s)
- Communication using ws2_32.dll
- Original DLL name was removed

**Version 2**
- Encryption process for the first data was changed

**Version 3**
- 22 commands (c-x)
- VM detection has been removed
- Magic number of loader shellcode is changed
# Comparison for Each Version of SodaMaster

※ Light gray: Compile date might be tampered

<table>
<thead>
<tr>
<th></th>
<th>Version 1</th>
<th>Version 2</th>
<th>Version 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Compile Date</strong></td>
<td>2019/01/07 10:33:18</td>
<td>2019/06/10 16:58:10</td>
<td>2020/10/20 10:46:49</td>
</tr>
<tr>
<td><strong>File Type</strong></td>
<td>x86 DLL</td>
<td>x64 DLL</td>
<td>x64 DLL</td>
</tr>
<tr>
<td><strong>Original DLL Name</strong></td>
<td>httpsWin32.dll</td>
<td>httpsX64_d.dll</td>
<td>tcpcX64.dll</td>
</tr>
<tr>
<td><strong>Export Function</strong></td>
<td>DLLEntry</td>
<td>-</td>
<td>DLLEntry</td>
</tr>
<tr>
<td><strong>Network API</strong></td>
<td>wininet</td>
<td>ws2_32</td>
<td>ws2_32</td>
</tr>
<tr>
<td><strong>Command</strong></td>
<td>d, s</td>
<td>d, f, l, s</td>
<td>c - x</td>
</tr>
<tr>
<td><strong>Anti-VM</strong></td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td><strong>Addition of Junk data</strong></td>
<td>-</td>
<td>-</td>
<td>Using string length of collected PC info</td>
</tr>
</tbody>
</table>
Changes of Loader Shellcode for SodaMaster

The basic implementation was not changed

- The magic bytes have been changed from version 3
- The size of data was increased depending the payload which was the updated SodaMaster

<table>
<thead>
<tr>
<th>offset</th>
<th>data</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x000</td>
<td>90 90 90 90 90 90 90</td>
<td>magic bytes for Identification, this is used for comparison before data processing</td>
</tr>
<tr>
<td>0x008</td>
<td>0x11600</td>
<td>Size of encrypted data, only this value (size) is observed</td>
</tr>
<tr>
<td>0x00C</td>
<td>A9 5B 7B 84 9C CB CF E8 B6 79 F1 9F 05 B6 2B FE</td>
<td>16 bytes RC4 key (each sample has different key)</td>
</tr>
<tr>
<td>0x01C</td>
<td>C7 36 7E 93 D3 07 1E 86 23 75 10 49 C8 AD 01 9F [skipped]</td>
<td>Encrypted SodaMaster payload with RC4</td>
</tr>
</tbody>
</table>
Command List of SodaMaster Version 3

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>Steals credentials of Outlook</td>
</tr>
<tr>
<td>d</td>
<td>Executes DLL (Specified export function)</td>
</tr>
<tr>
<td>f</td>
<td>Enables/Disables adding size info into sending data</td>
</tr>
<tr>
<td>g</td>
<td>Executes shellcode (No function table)</td>
</tr>
<tr>
<td>h</td>
<td>Repeats sending source spoofed packet to specified destination (DoS?)</td>
</tr>
<tr>
<td>i</td>
<td>Repeats sending 0x20000 bytes data padded with 0xCC (DoS?)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>l</td>
<td>Configures interval of C2 communications</td>
</tr>
<tr>
<td>m</td>
<td>Collects screenshot</td>
</tr>
<tr>
<td>q</td>
<td>Enables key logging</td>
</tr>
<tr>
<td>r</td>
<td>Disables key logging</td>
</tr>
<tr>
<td>s</td>
<td>Executes shellcode (With function table)</td>
</tr>
<tr>
<td>w</td>
<td>Shows string with MessageBox</td>
</tr>
<tr>
<td>x</td>
<td>Exits process</td>
</tr>
</tbody>
</table>

※ e, j, k, n-p, t-v are not Implemented
※ Red: Different analysis results from Trend Micro
Changes on C2 Command Execution of SodaMaster

Loop statement for processing a large number of commands

Version 2

```c
command = recv_data[4];
switch ( command )
{
    case 'd':
        exec_dll(recv_data + 5, (size - 5));
        break;
    case 'f':
        rc4_key = *(recv_data + 5);
        break;
    case 'l':
        sleep_time = *(recv_data + 5);
        break;
    case 's':
        exec_shellcode(recv_data + 5);
        break;
}
```

Version 3

```c
j = 0x164;
if ( command_list[0].function )
{
    k = 0x164;
    command_function = &command_list[0].function;
    do
    {  
        if ( command_list[k].id == *(recv_data + 4) )
            (*command_function)(recv_data + 5, (data_size - 1));
        k = ++j;
        command_function = &command_list[j].function;
    } while ( *command_function );
return 1i64;
```
The First Data Format Sent to SodaMaster C2

- Data chunk format

<table>
<thead>
<tr>
<th>1 Byte</th>
<th>1 Byte</th>
<th>Length of data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Size of data (If data length is variable)</td>
<td>Data</td>
</tr>
</tbody>
</table>

- The raw data before encryption
## The First Data Sent to SodaMaster C2

Contains 7 types of data

<table>
<thead>
<tr>
<th>ID</th>
<th>Length</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x03</td>
<td>Variable</td>
<td>Username</td>
</tr>
<tr>
<td>0x07</td>
<td>Variable</td>
<td>Computer name</td>
</tr>
<tr>
<td>0x04</td>
<td>5 Bytes</td>
<td>PID, Privilege flag</td>
</tr>
<tr>
<td>0x40</td>
<td>9 Bytes</td>
<td>Processor architecture (1 byte), OS major version (2 bytes), OS build number (2 bytes), Legacy OS flag (e.g., Win2003 x64 = 0xFF10) (2 bytes), OS product type (2 bytes)</td>
</tr>
<tr>
<td>0x05</td>
<td>Variable</td>
<td>Date of execution (yyyy/mm/dd hh:mm:ss)</td>
</tr>
<tr>
<td>0x06</td>
<td>Variable</td>
<td>RC4 encryption key for C2 communication</td>
</tr>
<tr>
<td>0x08</td>
<td>4 Bytes</td>
<td>Socket name</td>
</tr>
</tbody>
</table>
 Encryption Process for The First Data

1. RSA encryption
   - Base64 encoded public key is hardcoded
   - The public key is different in each sample
2. Inverting encrypted data
3. Adding junk data at the end (ver.2 or later)
   - Two types of calculation methods for the size of the junk data were observed:
     i. (address of encrypted data + address of collected data from victim) % 0x50 + 5
     ii. (address of encrypted data + returned value of GetTickCount) % 0x50 + 5
   - Add data extracted from encrypted data by unique algorithm to the end

Encrypt by RSA + Invert

Unnecessary data added to the end of data
New TTPs Observed in 2021

Continuous A41APT Campaign in 2021
Jackpot Webshell

- Webshell malware firstly reported by Trend Micro with their deep analysis
  - It was used as a payload of SigLoader in 2021
- Works as a standalone HTTP sever
  - Jackpot receives commands via a POST request for the specific URL
    - A domain/IP address of victim organization is hardcoded
  - Jackpot tends to be found at the IIS servers, because the infected host must be the internet-facing server
    - Even if the IIS service is running, Jackpot works on the same port
Microsoft Exchange Server

Exploiting ProxyShell vulnerability

- The following commands were observed on a PowerShell session obtained by the exploit
  - copy
  - dir
  - ipconfig /all
  - net user /domain
  - net time /domain
  - ping
  - query user
  - type
  - tasklist
  - whoami
  - wmic /node:<ip address> process call create cmd /c Dnscmd /EnumZones >\output file\n  - [System.Text.Encoding]::Unicode.GetString([System.Convert]::FromBase64String('\\string>')) | Out-File -FilePath <File path>

- China Chopper webshell was installed after above activities
We discovered another loader used for loading SodaMaster in 2021

- Unnamed loader that has been observed since 2015 for various payloads
- Named after string "HUIHWASDIHWEIUDHDSFSFEEFWEFEWFDSDGEFERWGWEEFWFWEWD"
Execution Flow of HUI Loader

HUI Loader

DLL Side-Loading

XOR decode & code injection

Encoded shellcode

svchost.exe

Layer I

Layer II

Layer III

SodaMaster Shellcode

RC4 decrypt

SodaMaster

Legitimate file
Decoding Process of HUI Loader

XOR decode by the hardcoded unique key

```python
def hui_decode(enc, key):
    key = bytearray(key.encode())
    dec = bytearray()
    for i in range(len(enc)):
        payloadbyte = enc[i] ^ 0x20 ^ key[i%len(key)]
        dec.append(payloadbyte)
    return dec
```
Attribution of The Threat Actor

Continuous A41APT Campaign in 2021
View of Trend Micro and Kaspersky

Linking to BRONZE RIVERSIDE (APT10) ?

調査結果に基づき、私たちはA41APTの活動の背後にAPT10が存在することにはかなりの確度があると考えています。裏付けとなるのは以下のポイントです。

第1に、x86 SodaMaster検体にハードコードされた「www.rare-coins.com」というURLが、ADEO IT Consulting Servicesによるレポート（英語）の中で言及されています。同レポートは、トルコの金融および電気通信セクターを標的とするAPT10の活動に関するもので、VirusTotalへの提出があっ地理位置情報とも一致しています。

第2に、A41APTの攻撃活動とAPT10の活動との類似性は、Cylanceのブログ記事（英語）で説明されています。記事中ではEcipekae、FYAntiのユニーなエクスポート名である「F***KY**Anti」、CppHostCLRの使用、FYAntiの最終ベイロードとしてのQuasarRATについて触れられています。それだけでなく、Symantecのブロガー記事（英語）にて言及されているFYAnti、「F***KY**Anti」というエクスポート名、.NETローダーの注入に使用されるCppHostCLR、QuasarRATも、BlackBerry Cylance Threat Research Teamによって発見されたAPT10グループの活動と類似しています。

このほか、私たちが過去に作成したAPT10の活動に関するThreat Intelligence Portalレポートには、複数の類似性と共通のTTPが見られます。
A41APT, BRONZE RIVERSIDE and LockFile

In August 2021, The HUI Loader was pointed out to be used with BRONZE RIVERSIDE and LockFile.
Redefinition of Chaotic A41APT Campaign

Attribution should not be done only by the malware/tools used - but it’s likely that the actor is based in China
What We Can Do against the Chaotic A41APT Campaign
Challenge to Know Your Own Organization

Fighting against Opportunistic Compromise, Targeted Deployment

- The actor attempts to compromise every organizations who seem to be related to their goals, then the actor will choose (an) organization(s) from among the victims as a start point
  - Not only HQ, subsidiaries and overseas branches will be affected
  - Incidents will happen at organizations who don't have enough security controls for internet-facing systems
- Do you have a true understanding of your organization from security perspective?
  - Infrastructure/Security controls of overseas branches, subsidiaries
    - Different systems from HQ
    - Low-budget security controls
    - Network/System sharing between HQ and subsidiaries
  - Leave maintenance and operation of system SI vendors
  - Network management, Account management, Endpoint management, etc.

Nothing changed from 2020
Conclusion

- Chaotic A41APT campaign
  - The campaign is still ongoing and expanding its TTPs
  - Multiple threat groups seem to be involved
  - The actor always intruded via Internet-facing systems
- Countermeasure should be the same with post-intrusion ransomware attacks
  - Protect internet-facing systems of whole your company including branch offices and subsidiaries
    - Cooperate with SI vendors
  - Detect usage of hacking tools or AD related tools for lateral movement after establishing C2
  - Hunt the threats by using EDR, auditing various logs, checking ASEP
- Information sharing like this would be helpful for everyone?
  - Difficult to reveal a whole picture of a campaign by a single vendor
  - Organizing information can preserve the anonymity of victims
Reference Regarding A41APT

1. A41APT case ~ Analysis of the Stealth APT Campaign Threatening
2. APT10: sophisticated multi-layered loader Ecipekac discovered in A41APT campaign
3. APT10: Tracking down the stealth activity of the A41APT campaign
4. 标的型攻击の実态と対策アプローチ 第5版 日本を狙うサイバーエスピオナージの动向2020年度 - Macnica Networks, TeamT5
5. 「Earth Tengshe」によるマルウェア「SigLoader」を用いた攻撃キャンペーンで観測された新たなペイロード
   ○ https://blog.trendmicro.co.jp/archives/29842
Other References

1. Uncovering New Activity By APT10 | FortiGuard Labs

2. Insights into Ransomware Spread Using Exchange 1-Day Vulnerabilities 1-2 - NSFOCUS, Inc.,

3. Twitter
   ○ [https://twitter.com/Manu_De_Lucia/status/1430115616862638080](https://twitter.com/Manu_De_Lucia/status/1430115616862638080)
   ○ [https://twitter.com/fr0gger_/status/1430213808434339842](https://twitter.com/fr0gger_/status/1430213808434339842)

4. Guidance for preventing, detecting, and hunting for exploitation of the Log4j 2 vulnerability

5. AutoRuns
   ○ [https://docs.microsoft.com/ja-jp/sysinternals/downloads/autoruns](https://docs.microsoft.com/ja-jp/sysinternals/downloads/autoruns)
## IoCs

<table>
<thead>
<tr>
<th>Value</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cf5ec3b803563d8ef68138f5303ebc362b72da36da29b9caba3062ae996db9234</td>
<td>SHA256</td>
<td>HUILoader</td>
</tr>
<tr>
<td>c13f937bb1f8f5f9bd6dd425f7af873119c8bb8248490de6db9b29d0c68783e</td>
<td>SHA256</td>
<td>Encoded SodaMaster shellcode</td>
</tr>
<tr>
<td>168.100.8.20</td>
<td>IP</td>
<td>SodaMaster C2</td>
</tr>
<tr>
<td>9bec85e6a3d811826580540b541723c6b5236377a3a980b1ffa5bf5f749a99d4</td>
<td>SHA256</td>
<td>HUILoader</td>
</tr>
<tr>
<td>7db327cc7bd622038f69b4df4178ca3145659a73ccbc10d0228e48f2ece60896</td>
<td>SHA256</td>
<td>Encoded SodaMaster shellcode</td>
</tr>
<tr>
<td><a href="http://www.%5B.%5Dmonferriina%5B.%5Dcom">www.[.]monferriina[.]com</a></td>
<td>Domain</td>
<td>SodaMaster C2</td>
</tr>
<tr>
<td>c0ed7393945726b611000099b926917723f5cf9b2df0be070f2a500b6edf161c</td>
<td>SHA256</td>
<td>SigLoader (Layer I)</td>
</tr>
<tr>
<td>0a570b32d14799f6351ee211093567450d41705ca79e236a38a15f135d78bdf</td>
<td>SHA256</td>
<td>SigLoader (Layer I)</td>
</tr>
<tr>
<td>2da5e37ec47059a7935165039ea31b0c9cc8f1bb0d0c620759776979158cf30</td>
<td>SHA256</td>
<td>SigLoader (Layer I)</td>
</tr>
<tr>
<td>e8797bf4334fba067d5f91d1481bd8f55bf2e45483a92ae7030c2c604dd273</td>
<td>SHA256</td>
<td>SigLoader (Layer I)</td>
</tr>
<tr>
<td>68dd499bca62e004c97ccc17f68e36d6de2885446924dabe8cc5257633ca08a3</td>
<td>SHA256</td>
<td>Encrypted SodaMaster shellcode</td>
</tr>
<tr>
<td>192.248.183.113</td>
<td>IP</td>
<td>SodaMaster C2</td>
</tr>
<tr>
<td>1f1bcb03b0008c4fdd462e7d2b5db5eca321ff6d56bb22cdd39c82d1f6a038f</td>
<td>SHA256</td>
<td>DESLoader (1st Loader)</td>
</tr>
<tr>
<td>7337071599eb49c75c63df210a516ea8dbbe99a8a66237f66f3f3c7f5aed31</td>
<td>SHA256</td>
<td>Encrypted SigLoader shellcode</td>
</tr>
<tr>
<td>59986e02e037747d0f5adb4eca394f5f1b01a8c2ba9cb6c1ce30f9312b9566</td>
<td>SHA256</td>
<td>Encrypted SodaMaster shellcode</td>
</tr>
<tr>
<td>185.10.16.115</td>
<td>IP</td>
<td>SodaMaster C2</td>
</tr>
<tr>
<td>8efcecc00763ce9269a01d2b5918873144746c4b203be28c92459f5301927961</td>
<td>SHA256</td>
<td>HUILoader in 2015</td>
</tr>
<tr>
<td>20fc3cf1afcad9e6f19e9aebbfbc9daf374909801d874c3d276913f12d6230ec</td>
<td>SHA256</td>
<td>Mimikatz</td>
</tr>
</tbody>
</table>
FYI: Hunting Suspicious ASEP

When EDR and Forensic Tools are not ready

- Audit ASEP by using tools such as Autoruns is effective
- In A41APT campaign, scheduled tasks are favor to be used
  - Investigating scheduled tasks with the following condition could be useful
    - 3rd party legitimate executables under C:\Windows\, C:\Intel\
## IOCs - Examples of Scheduled Task

<table>
<thead>
<tr>
<th>Program</th>
<th>Description</th>
<th>Publisher</th>
</tr>
</thead>
<tbody>
<tr>
<td>C:\Windows\RoutineMaintenance.exe</td>
<td></td>
<td>D3L</td>
</tr>
<tr>
<td>C:\Windows\iceiprole.exe</td>
<td>Malwarebytes Anti-Exploit 64bit tasks</td>
<td>Malwarebytes Corporation</td>
</tr>
<tr>
<td>C:\Windows\Vss\Writers\System\FamilySafety.exe</td>
<td>Java(TM) Platform SE binary</td>
<td>International Business Machines Corporation</td>
</tr>
<tr>
<td>C:\Windows\System32\winrm\0409\usoclient.exe</td>
<td>OpenSSL application</td>
<td>OpenVPN Inc.</td>
</tr>
<tr>
<td>C:\Windows\System32\da-DK\DataProviders.exe</td>
<td>OpenSSL application</td>
<td>OpenVPN Inc.</td>
</tr>
</tbody>
</table>

In other cases, 3rd party legitimate executables such as VMware Tools, Sandboxie that should be under \Program Files\ folder are installed under C:\Windows, C:\Intel\.
Thank you