

# 2022年 台湾のランサムウェア事例 とマルウェア解析

2022年12月

H. Murakami

 CYCRAFT

# 紹介事項

発表者： ごく普通の町のデジタルフォレンジック屋さん (村上)

所属： CyCraft Japan (本社は台湾)

東京電機大学 サイバーセキュリティ研究所 研究員

静岡大学 創造科学技術大学院 博士課程 (西垣研究室)



第001404号



Analyst Number 5549



Share:



## Malware Function-based encryption technique

Recent malware often uses techniques to evade detection by cybersecurity products. One of the techniques is the encryption of executable code. Malware analysis techniques for decrypting executable code in memory have existed for some time. However, more recent malware has employed an advanced...

By Hirokazu Murakami · June 22, 2022

Share:



## Reverse Engineering of WannaCry Worm and Anti Exploit Snort Rules

Today, a lot of malware is being created and utilized. To solve this problem, many researchers study technologies that can quickly respond automatically to detected malware. Using artificial intelligence (AI) is such an example. However, modern AI has difficulty responding to new attack methods. On...

By Hirokazu Murakami · May 27, 2018

SANS Whitepaper

SANS Whitepaper

# 1. Case 1: .msi形式のMagniberランサムウェア

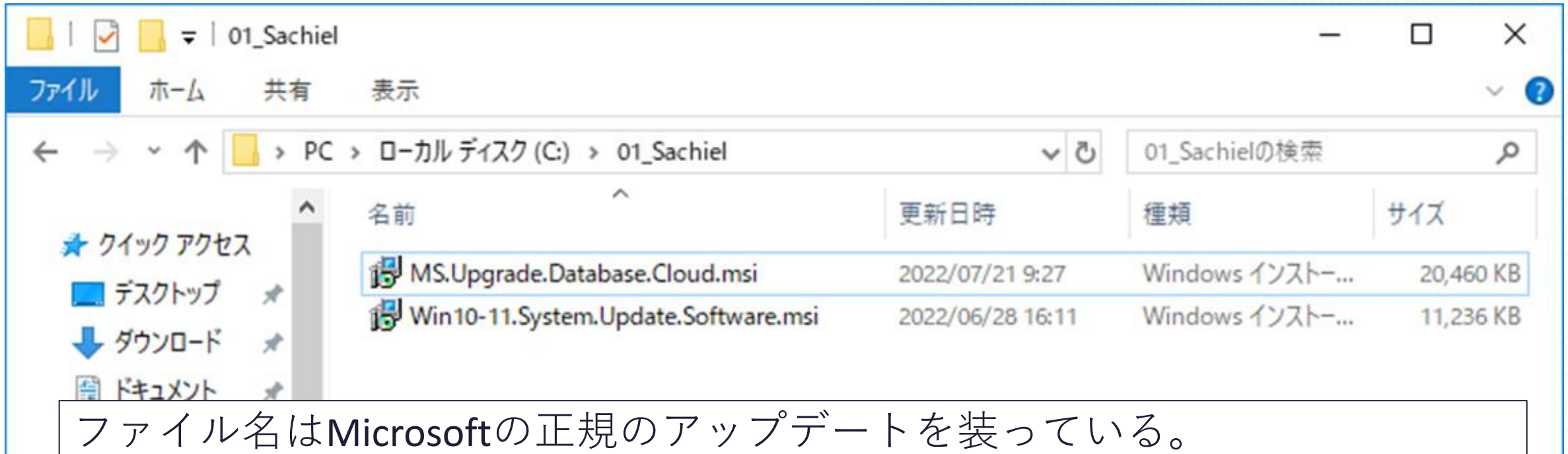
## ● 特 徴

- 2022年6月頃から広がりを確認（顧客から問い合わせあり）。
- インストーラの.msi形式になっており、基本的にユーザを騙してインストールさせる方法。
- .msiファイルにDLL形式のマルウェアが埋め込まれている。
- ファイルの暗号化はAES128 CBCモード、IV、AES共通鍵は乱数で作成し、RSA公開鍵で暗号化。
- 実行中のプロセスのうち、**オープンできるプロセスに対しかたっぱしから暗号化コードをインジェクション**してスレッドを実行する。
- **syscallの利用**で検知回避を狙っているとみられる。



# 1. Case 1: .msi形式のMagniberランサムウェア

- 当該ファイル



# 1. Case 1: .msi形式のMagniberランサムウェア

- .msi形式のマルウェアの解析：AdvancedInstallerの利用

① Custom Actionsを選択

② Existing Custom Actionsタブを選択

③ Actionを選択

実行されるアクション

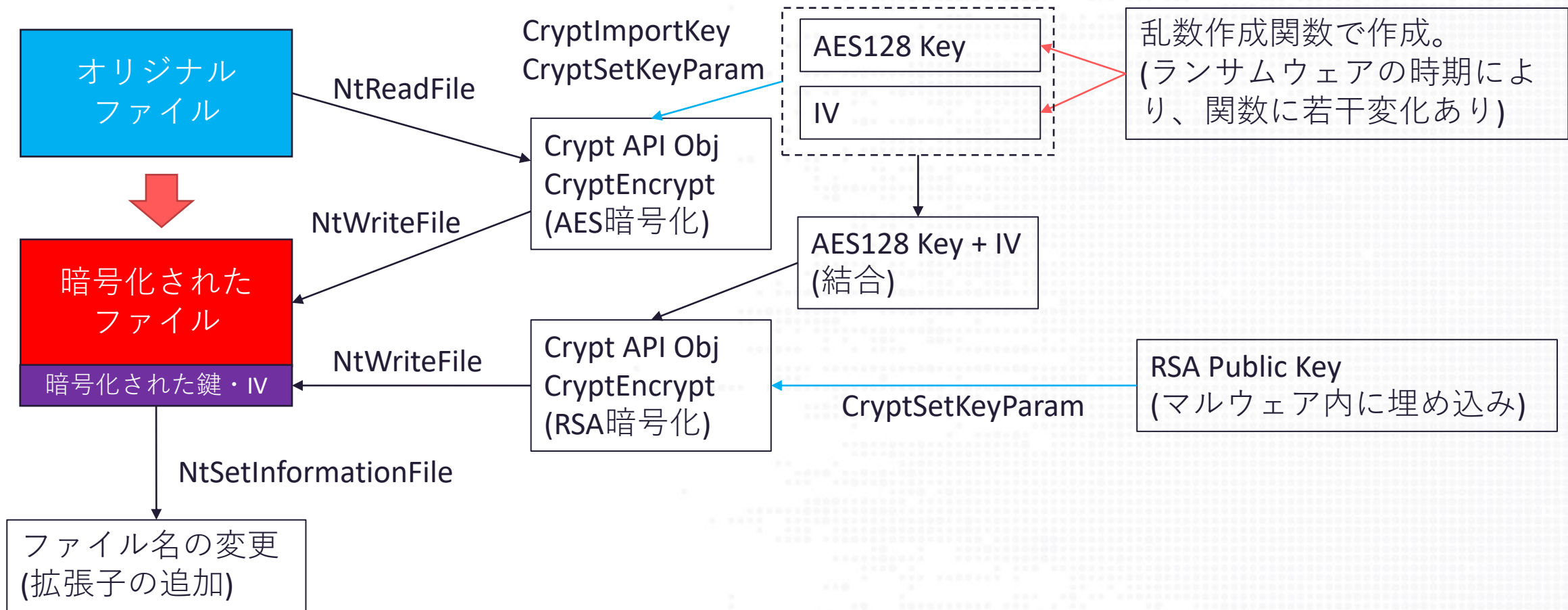
ツールによってExportされたDLLのパス

実行されるDLL内の関数名

**ExportされたDLLを、rundll32.exe等を用いて実行されるDLL内の関数名をコールして解析すればよい。**

# 1. Case 1: .msi形式のMagniberランサムウェア

## ● 暗号化プロセス



# 1. Case 1: .msi形式のMagniberランサムウェア

- プロセスサーチおよびプロセスインジェクション
  - syscallで**SystemProcessInformation**を実行してシステム上のプロセス情報を取得する（あっ、いつもの**CreateToolhelp32Snapshot**じゃないんだー（棒））。
  - プロセス毎にNtOpenProcessを実行し、**プロセスのオープンに成功した場合**はそのプロセスのメモリ上に暗号化する**ランサムウェアのコードをコピーし、NtCreateThreadExで実行**する。
    - 多くのプロセスで暗号化処理が行われるため、**かなり重くなる**。
    - 暗号化済みかどうかの判定は、拡張子でおこなっている。



# 1. Case 1: .msi形式のMagniberランサムウェア

- プロセスサーチおよびプロセスインジェクション

動的解析のため実行させてみたところ、「winlogon.exe」のプロセスで暗号化処理をしていたことを確認。

Time of Day	Process Name	PID	Operation	Path	Result	Additional Info
14:50:35.6343059	winlogon.exe	608	CloseFile	C:#01_Sachiel#kb45r6ff6nel.js	SUCCESS	
14:50:35.6343284	csrss.exe	412	CreateFile	C:#Windows#System32#en-US#Microsoft.Windows.Common-...	NAME NOT FOUND	Desired Access: Rea...
14:50:35.6344001	csrss.exe	412	CreateFile	C:#Windows#system32#en-US#Microsoft.Windows.Common...	PATH NOT FOUND	Desired Access: Rea...
14:50:35.6344433	winlogon.exe	608	CreateFile	C:#01_Sachiel#kb45r6ff6nel.js	SUCCESS	Desired Access: Rea...
14:50:35.6344735	csrss.exe	412	CreateFile	C:#Windows#system32#en-US#Microsoft.Windows.Common...	PATH NOT FOUND	Desired Access: Rea...
14:50:35.6344763	winlogon.exe	608	QueryBasicInfor...	C:#01_Sachiel#kb45r6ff6nel.js	SUCCESS	CreationTime: 2022...
14:50:35.6345492	winlogon.exe	608	CreateFile	C:#01_Sachiel	SUCCESS	Desired Access: Wri...
14:50:35.6346220	winlogon.exe	608	SetRenameInfor...	C:#01_Sachiel#kb45r6ff6nel.js	SUCCESS	ReplaceIfExists: Fals...
14:50:35.6346974	winlogon.exe	608	CloseFile	C:#01_Sachiel	SUCCESS	
14:50:35.6347176	svchost.exe	1088	CreateFile	C:#Windows#System32#regsvr32.exe	SUCCESS	Desired Access: Rea...
14:50:35.6347578	winlogon.exe	608	CloseFile	C:#01_Sachiel#kb45r6ff6nel.js.mbjzcdqi	SUCCESS	
14:50:35.6347641	csrss.exe	412	CreateFile	C:#Windows#assembly#GAC_04#microsoft.windows.common...	PATH NOT FOUND	Desired Access: Rea...
14:50:35.6349262	csrss.exe	412	CreateFile	C:#Windows#System32#en#Microsoft.Windows.Common-C...	NAME NOT FOUND	Desired Access: Rea...
14:50:35.6349938	winlogon.exe	608	QueryDirectory	C:#01_Sachiel	NO MORE FILES	FileInformationClass...
14:50:35.6350578	csrss.exe	412	CreateFile	C:#Windows#System32#en#Microsoft.Windows.Common-C...	NAME NOT FOUND	Desired Access: Rea...
14:50:35.6351193	csrss.exe	412	CreateFile	C:#Windows#system32#en#Microsoft.Windows.Common-C...	PATH NOT FOUND	Desired Access: Rea...
14:50:35.6351326	winlogon.exe	608	CloseFile	C:#01_Sachiel	SUCCESS	
14:50:35.6352033	csrss.exe	412	CreateFile	C:#Windows#system32#en#Microsoft.Windows.Common-C...	PATH NOT FOUND	Desired Access: Rea...

Showing 264,635 of 1,723,013 events (15%)      Backed by C:#Users#Murakami#Documents#MyWork#社内作業関連#202206\_台湾



# 1. Case 1: .msi形式のMagniberランサムウェア

- syscallの頻繁な利用

同じAPIでも、**OSのバージョンによってsyscallで使うコードが異なる**ため、OSのバージョン情報を元にコードを判定。

```
debug045:000001CA503C2252 rep stosb
debug045:000001CA503C2254 lea rax, [rbp-1]
debug045:000001CA503C2258 mov cl, 33h
debug045:000001CA503C225A mov [rbp+1Fh], rax
debug045:000001CA503C225E cmp dword_7FFE026C, 0Ah
debug045:000001CA503C2266 jz short loc_1CA503C226A
debug045:000001CA503C2268 mov cl, 30h
debug045:000001CA503C226A loc_1CA503C226A: ; CODE XR
debug045:000001CA503C226A call CreateSyscallFunc, loc_1CA503C0E9C
debug045:000001CA503C226F lea r9, [rbp-11h]
debug045:000001CA503C2273 lea r8, [rbp+0Fh]
debug045:000001CA503C2277 lea rcx, [rbp-29h]
debug045:000001CA503C227B mov edx, 110080h
```

RAX 0000009422BBEF58 Stack[00000E08]:0000009422BBEF58  
RBX 00007FFF5BDB5AF0 ntdll\_RtlDosPathNameToNtPathName\_U  
RCX 0000000000000033  
RDX FFFFFFFEC9D2739B08  
RSI 000001CA52A70000 debug130:000001CA52A70000  
RDI 0000009422BBEF58 Stack[00000E08]:0000009422BBEF58  
RBP 0000009422BBEF59 Stack[00000E08]:0000009422BBEF59  
RSP 0000009422BBEF00 Stack[00000E08]:0000009422BBEF00  
RIP 000001CA503C226A debug045:loc\_1CA503C226A  
R8 0000000000000000  
R9 0000000000000000  
R10 000001CA50470000 debug011:000001CA50470000

NtAllocateVirtualMemoryで実行可能領域を確保し、得られたコードでsyscallをするという**小さな関数を作成**。

syscallテーブル参考：<https://j00ru.vexillium.org/syscalls/nt/64/>

# 1. Case 1: .msi形式のMagniberランサムウェア

- syscallの頻繁な利用

```
debug045:000001CA503C2268 mov     cl, 30h ; '0'
debug045:000001CA503C226A
debug045:000001CA503C226A loc_1CA503C226A:                ; CODE XREF: debug045:000001CA503C2266↑j
debug045:000001CA503C226A call   CreateSyscallFunc_loc_1CA503C0E9C
debug045:000001CA503C226F lea   r9, [rbp-11h]
debug045:000001CA503C2273 lea   r8, [rbp+0Fh]
debug045:000001CA503C2277 lea   rcx, [rbp-29h]
debug045:000001CA503C227B mov   edx, 110080h
debug045:000001CA503C2280 mov   dword ptr [rsp+28h], 20h ; ' '
debug045:000001CA503C2288 mov   dword ptr [rsp+20h], 7
debug045:000001CA503C2290 call   rax ; NtOpenFile
debug045:000001CA503C2292 test  eax, eax
debug045:000001CA503C2294 jns   short loc_1CA503C229D
debug045:000001CA503C2296
debug045:000001CA503C2296 loc_1CA503C2296:                ; CODE XREF: debug045:000001CA503C22F2↓j
debug045:000001CA503C2296 ; debug045:000001CA503C2371↓j ...
debug045:000001CA503C2296 xor   eax, eax
debug045:000001CA503C2298 jmp   loc_1CA503C23B4
```

```
debug131:000001CA52A80000 ; Segment type: Pure code
debug131:000001CA52A80000 ; Segment permissions: Read/Write/Execute
debug131:000001CA52A80000 debug131 segment byte public 'CODE' use64
debug131:000001CA52A80000 assume cs:debug131
debug131:000001CA52A80000 ;org 1CA52A80000h
debug131:000001CA52A80000 assume es:ntdll_dll, ss:ntdll_dll, ds:ntdll_dll, fs:ntdll_dll, gs:ntdll_dll
debug131:000001CA52A80000 mov   r10, rcx
debug131:000001CA52A80003 mov   eax, 33h ; '3'
debug131:000001CA52A80008 syscall ; Low latency system call
debug131:000001CA52A8000A retn
debug131:000001CA52A8000A ;
```

syscallによるNtOpenFileのコール例

syscall実行用領域を解放していない  
バグ持ち (ふざけんなド下手糞)

# 1. Case 1: .msi形式のMagniberランサムウェア

- その他の特徴

- パッカー部分は、単純なXORによる暗号化を利用。ただし、ワンステップずつjmpすることで若干の難読化を試みている。
- スクリプトを展開し、vssadmin、wbadmin、bcdeditを利用してVolume Shadow Copyの消去、バックアップカタログの削除、システム状態バックアップの削除、全てのブートエラーを無視するよう表示ポリシーを変更、スタートアップ修復の無効化をしている。



## 2. Case 2:Lockbit 2.0 ランサムウェア

### ● 特 徴

- 2022年2月、台湾で被害者の情報がDark Webにリークしていることを発見したことから調査開始となった。
- このランサムウェアのファイルの暗号化はAES128 CBCモードで、**ファイルの先頭4,096byteしか暗号化しない。**
- 解析回避、アンチフォレンジックのテクニックがほのかには入っている（ただし、あんまり気にならなかった）が、マルウェアはパックもされていない。
- プリンタで9999枚脅迫文を印刷、壁紙をマルウェアが頑張って描画など、変なところに力をいれているマルウェアでもある。

MBSDさんの解析が詳しい： <https://www.mbsd.jp/research/20211019/blog/>

## 2. Case 2: Lockbit 2.0 ランサムウェア

- 解析した感想 . . .

- マルウェアが**暗号化を開始する前に色々やってくれる**ので、それらの振る舞いを検知・遮断できるEDR等の機能があれば、割と対策が可能だと思われる。
- ビルダーでランサムウェアを生成するようにしている影響かもしれないが、コードの構造化や関数化が不十分で、ダラダラ長い& ifのネストが深い（=**コードのできが悪い**）印象。
- 解析回避、アンチフォレンジック等はあるが、既存のライブラリ等を集めて取って付けたような感じで、マルウェア全体の設計は不十分に見える（あちこちに雑さが残っている）。

## 2. Case 2: Lockbit 2.0 ランサムウェア

The image shows two screenshots from a Windows system. The top screenshot is of the Registry Editor, with the path `コンピューター\HKEY_LOCAL_MACHINE\SOFTWARE\Classes\*.lockbit\DefaultIcon` highlighted in red. The bottom screenshot is of the Registry Editor, with the path `コンピューター\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run` highlighted in red. Below this, a table shows registry values:

名前	種類	データ
ab (既定)	REG_SZ	(値の設定なし)
ab {B9497174-7575-1ADA-335A-33B0C1C56C7C}	REG_SZ	"C:\01_Sachiel\build.exe"
OneDrive	REG_SZ	"C:\Users\Sachiel\AppData\Local\Microsoft\One...

The bottom screenshot also shows a File Explorer window displaying the contents of the local disk (C:). The file `EC49EA74.lock` is highlighted in red in the file list.

\* **暗号化開始前**に見られる痕跡の数々。  
(検知に便利そう・・・)



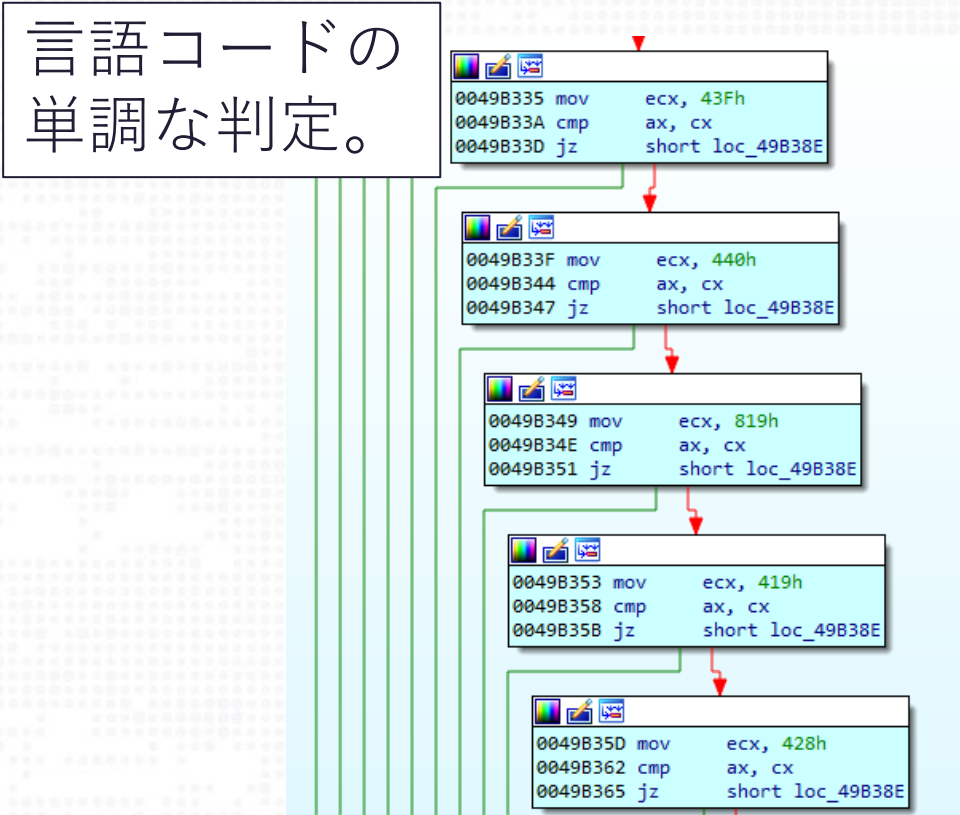
## 2. Case 2: Lockbit 2.0 ランサムウェア

- コードは単調でパラメータも暴露しやすく、解析難易度は高くはない。

単純なコピーでプレーンテキスト展開。

```
0045F84B xor     eax, eax
0045F84D mov     [ebp+var_3D8], 33002Eh
0045F857 mov     [ebp+var_3D4], 360038h
0045F861 mov     [ebp+var_3D0], ax
0045F868 mov     [ebp+var_3E4], 63002Eh
0045F872 mov     [ebp+var_3E0], 64006Dh
0045F87C mov     [ebp+var_3DC], ax
0045F883 mov     [ebp+var_3F0], 61002Eh
0045F88D mov     [ebp+var_3EC], 69006Eh
0045F897 mov     [ebp+var_3E8], ax
0045F89E mov     [ebp+var_324], 61002Eh
0045F8A8 mov     [ebp+var_320], 760064h
0045F8B2 mov     [ebp+var_31C], ax
0045F8B9 mov     [ebp+var_3FC], 6D002Eh
0045F8C3 mov     [ebp+var_3F8], 690073h
0045F8CD mov     [ebp+var_3F4], ax
0045F8D4 mov     [ebp+var_408], 6D002Eh
0045F8DE mov     [ebp+var_404], 700073h
0045F8E8 mov     [ebp+var_400], ax
0045F8EF mov     [ebp+var_414], 63002Eh
0045F8F9 mov     [ebp+var_410], 6D006Fh
0045F903 mov     [ebp+var_40C], ax
0045F90A mov     [ebp+var_420], 6E002Eh
0045F914 mov     [ebp+var_41C], 73006Ch
0045F91E mov     [ebp+var_418], ax
0045F925 mov     [ebp+var_42C], 6F002Eh
0045F92F mov     [ebp+var_428], 780063h
0045F939 mov     [ebp+var_424], ax
0045F940 mov     [ebp+var_438], 6D002Eh
0045F94A mov     [ebp+var_434], 610070h
0045F954 mov     [ebp+var_430], ax
0045F95B mov     [ebp+var_444], 63002Eh
0045F965 mov     [ebp+var_440], 6C0070h
```

```
..tw..m.s.c...{w
..r.o.m.....b.
a.t.....w.p.x.
.....d.r.v....
..s.h.s.....h.
l.p.....b.i.n.
..^...r.d.p....
..r.t.p.....p.
r.f.....h.t.a.
.....m.o.d....
..c.p.l.....m.
p.a.....o.c.x.
.....n.l.s....
..c.o.m.....m.
s.p.....m.s.i.
.....a.n.i....
..c.m.d.....3.
8.6...X.%s.\.*.
...ホ.f.n.t...^
..m.s.u...R...w.
a.d...^...x.e.x.
.....i.p.a....
```



# 2. Case 2: Lockbit 2.0 ランサムウェア

- 解析回避例：

```
004BFF90 var_208= byte ptr -208h
004BFF90
004BFF90 push ebp
004BFF91 mov ebp, esp
004BFF93 and esp, 0FFFFFFF8h
004BFF96 mov eax, large fs:30h
004BFF9C sub esp, 480h
004BFFA2 test byte ptr [eax+68h], 70h
004BFFA6 push esi
004BFFA7 push edi
004BFFA8 jz short loc_4BFFB2
```

PEBのパラメータで判定  
→EIPを正常ルートへ

```
004BA73A
004BA73A loc_4BA73A:
004BA73A lea ecx, [ebp+var_20]
004BA73D push ecx
004BA73E push 0
004BA740 push [ebp+var_18]
004BA743 push [ebp+var_1C]
004BA746 push 0
004BA748 push 0
004BA74A call eax; CreateThread Addr dword 4F07EC
004BA74C mov esi, eax
004BA74E cmp esi, 0FFFFFFFh
004BA751 jz short loc_4BA768
```

```
004BA748 push 0
004BA74A call eax; CreateThread Addr dword 4F07EC
004BA74C mov esi, eax
004BA74E cmp esi, 0FFFFFFFh
004BA751 jz short loc_4BA768
```

```
004BA753 lock inc dword_4F0864
004BA75A push 0
004BA75C push 0
004BA75E push 11h
004BA760 push esi
004BA761 call Get_NtSetInformationThread Addr_sub_4154F0
004BA766 call eax
```

THREADINFOCLASS  
のパラメータ

```
004BA753 lock inc dword_4F0864
004BA75A nop
004BA75B nop
004BA75C nop
004BA75D nop
004BA75E nop
004BA75F nop
004BA760 nop
004BA761 nop
004BA762 nop
004BA763 nop
004BA764 nop
004BA765 nop
004BA766 nop
004BA767 nop
```

デバッガ  
からの秘匿

コード除去

```
004BA768
004BA768 loc_4BA768:
```

```
004BA768
004BA768 loc 4BA768:
```

## 2. Case 2: Lockbit 2.0 ランサムウェア

- スレッド間通信

NtSetInformationFile、NtRemoveIoCompletionを使って通信。

```
004A2974
004A2974 loc_4A2974:
004A2974 add     word ptr [edi], 10h
004A2978 mov     eax, dword_4E2520
004A297D push   1Eh
004A297F mov     [ebp+var_270], eax
004A2985 lea    eax, [ebp+var_270]
004A2988 push   8
004A298D push   eax
004A298E lea    eax, [ebp+var_278]
004A2994 mov     [ebp+var_26C], esi
004A299A push   eax
004A299B push   dword ptr [ebx]
004A299D call   Get_NtSetInformationFile_Addr_sub_415130
004A29A2 call   eax ; NtSetInformationFile
004A29A4 test   eax, eax
```

送信元スレッド：  
暗号化対象のチェックおよびファイルの  
オープン、読み込み、IVとAES鍵の生成



```
0049E812
0049E812 loc_49E812:
0049E812 push   0
0049E814 lea    ecx, [esp+464h+var_434+4]
0049E818 push   ecx
0049E819 lea    ecx, [esp+468h+var_448]
0049E81D push   ecx
0049E81E lea    ecx, [esp+46Ch+var_444]
0049E822 push   ecx
0049E823 push   dword_4E2520
0049E829 call   eax ; NtRemoveIoCompletion_dword_4F8C70
0049E82B test   eax, eax
0049E82D js    def_49E84F ; jumtable 0049E84F default case
```

受信先スレッド：  
暗号化の実行、暗号化結果の書き込み、  
ファイル名変更（.bitlock拡張子の追加）、  
ファイルクローズ



## 2. Case 2: Lockbit 2.0 ランサムウェア

- 発生した事象：

- Lockbit 2.0ランサムウェアに感染し、被害端末のデータが暗号化された。
- Lockbit 2.0の脅迫文には、さらに指定のWebサイトやTorのサイトで盗まれたデータを暴露すると脅迫。

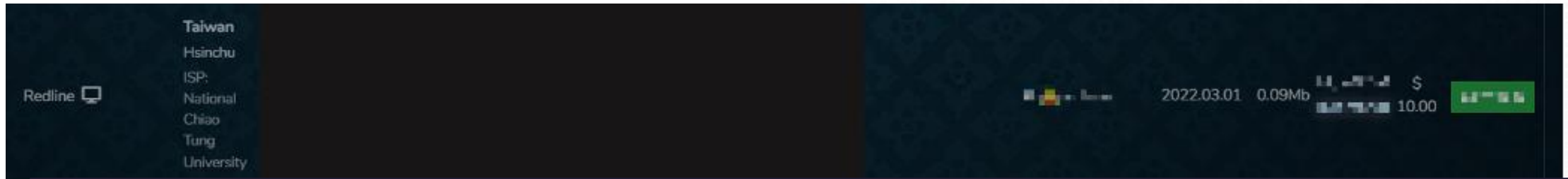
```
ALL YOUR 1 LockBit 2.0 Ransomware<
2 <
3 Your data are stolen and encrypted<
4 The data will be published on TOR website http://lockbit\[REDACTED\]
An) a35nygvokja5uuccip4kyd.onion and https://\[REDACTED\].at if you do not pay the ransom<
To recovery you 5 You can contact us and decrypt one file for free on these TOR sites<
```

ん？ Lockbit 2.0にデータ収集やC&Cサーバへの転送機能って無くない？

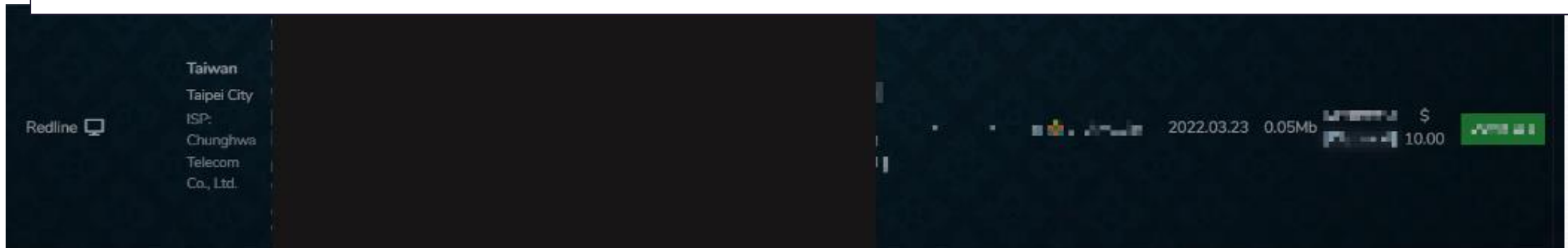
## 2. Case 2: Lockbit 2.0 ランサムウェア

- 事件の顛末：

- (1) Dark Webでの認証情報漏洩



「Redline」という情報窃盗組織により、2022年3月に台湾のユーザーとみられる認証情報が複数件販売されていることを確認。

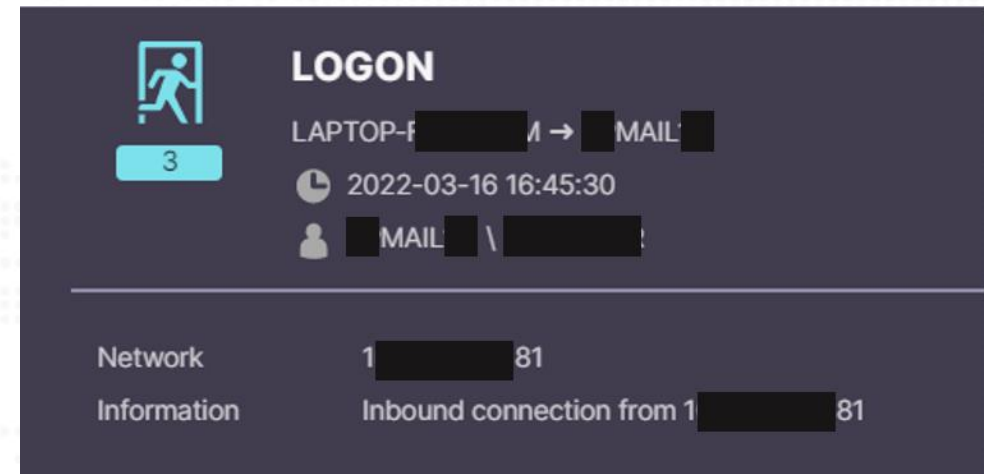
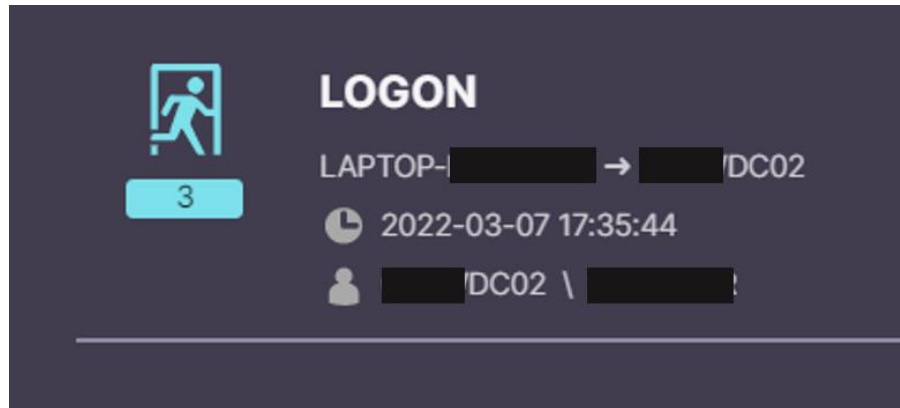


## 2. Case 2: Lockbit 2.0 ランサムウェア

- 事件の顛末：

(2) 正体不明の攻撃者によるアクセス

漏洩した情報を利用してログインした痕跡を確認。



正しい認証情報を用いてログインに成功すると、**高脅威度としての検知は難しい！**



## 2. Case 2: Lockbit 2.0 ランサムウェア

- 事件の顛末：

### (3) 攻撃のエスカレート

```
2022-04-13 20:01:30 [REDACTED] c:\windows\temp\s3110.bat
2022-04-13 22:12:20 [REDACTED] c:\windows\temp\prt3389.bat
2022-04-13 22:13:21 [REDACTED] 10. [REDACTED] >> C:\Windows\Temp\prt3389.bat
2022-04-13 22:13:23 [REDACTED] 10. [REDACTED] >> C:\Windows\Temp\prt3389.bat
2022-04-13 22:13:23 [REDACTED] 10. [REDACTED] >> C:\Windows\Temp\KB2604121_20130219_101408765.html
2022-04-13 22:13:24 [REDACTED] 10. [REDACTED] >> C:\Windows\Temp\KB2656351_20130219_101157363-Microsoft .NET Framework 4 Client Profile-MSP
```

64 PID:5404 2022-04-29 19:21:46 CPU 0% MEM 8MB  
ProcFile: C:\Windows\System32\svchost.exe  
Cmdline: "C:\Windows\System32\svchost.exe"

000000B1D22E0000	THREAD FUNCTIONS	NETWORK FUNCTIONS	APT MALWARE	OBFUSCATED CODE
000000B1D2550000	THREAD FUNCTIONS	NETWORK FUNCTIONS	APT MALWARE	OBFUSCATED CODE
00000000000400000				
000000B1D2190000				

バッチファイルの展開  
および実行、svchost.exe  
へのコードインジェク  
ションを確認。

## 2. Case 2: Lockbit 2.0 ランサムウェア

- 事件の顛末：

- (4) 被害組織内での認証情報窃盗

```
2022-04-30 16:45:58 02 CTI C:\Users\ [REDACTED] \Downloads\mimikatz-o\passrecpk\Dialupass.exe (T1204)
2022-04-30 16:47:31 02 C:\Windows\System32\notepad.exe (T1003, T1204)
2022-04-30 16:45:57 02 CTI C:\Users\ [REDACTED] \Downloads\mimikatz-o\passrecpk\BulletsPassView64.exe (T1204)
2022-04-30 16:45:51 02 C:\Users\ [REDACTED] \Downloads\mimikatz-o\mimikatz\x32\mimikatz.exe (T1003, T1204)
2022-04-30 16:45:58 02 CTI C:\Users\ [REDACTED] \Downloads\mimikatz-o\passrecpk\WirelessKeyView64.exe (T1003)
2022-04-30 16:45:50 02 CTI C:\Users\ [REDACTED] \Downloads\mimikatz-o\mimikatz\x64\mimikatz.exe (T1003, T1204)
2022-04-30 16:45:57 02 CTI C:\Users\ [REDACTED] \Downloads\mimikatz-o\passrecpk\netpass64.exe (T1204)
2022-04-30 16:45:58 02 CTI C:\Users\ [REDACTED] \Downloads\mimikatz-o\passrecpk\OperaPassView.exe (T1204)
2022-04-30 16:45:58 02 CTI C:\Users\ [REDACTED] \Downloads\mimikatz-o\passrecpk\WebBrowserPassView.exe (T1204)
```

認証情報をダンプするmimikatzを展開、実行した痕跡を確認。

## 2. Case 2: Lockbit 2.0 ランサムウェア

- 事件の顛末：

- (5) リモート操作のためのマルウェアの実行

PROCESS	MEMORY REPORT
rundll32.exe	64 PID:2608 2022-05-01 01:28:01 CPU 0% MEM 6MB ProcFile: C:\Windows\System32\rundll32.exe Cmdline: rundll32 c:\users\public\music\observerinfo443.dll, LoadIconW 000000F64DA30000 APT MALWARE 00007FF8CC840000 c:\Users\Public\Music\observerinfo443.dll
rundll32.exe	64 PID:3616 2022-04-30 21:20:01 CPU 0% MEM 10MB ProcFile: C:\Windows\System32\rundll32.exe Cmdline: rundll32 c:\users\public\music\observerinfo443.dll, LoadIconW 000000F355000000 APT MALWARE 00007FF8CC840000 c:\Users\Public\Music\observerinfo443.dll

**EXECUTION**  
C:\Windows\System32\rundll32.exe  
2022-05-01 01:28:01  
Activity Details  
[APT].F420BD6  
Creator process: C:\Windows\System32\cmd.exe (PID:5316)  
Directory: C:\WINDOWS\SYSTEM32\  
Outbound connection to  
Path: C:\WINDOWS\SYSTEM32  
Suspicious Code - F64DA30000  
rundll32 c:\users\public\music\observerinfo443.dll, LoadIconW  
Process monitoring, PID 2608  
Malware reverse engineering, ADDRESS:F64DA30000  
ba386122515ef3643cf51ad9df29a431  
C:\USERS\PUBLIC\MUSIC\OBSERVERINFO443.DLL  
rundll32 c:\users\public\music\observerinfo443.dll, LoadIconW

**CobaltStrike**とみられるマルウェアの実行の痕跡を確認。

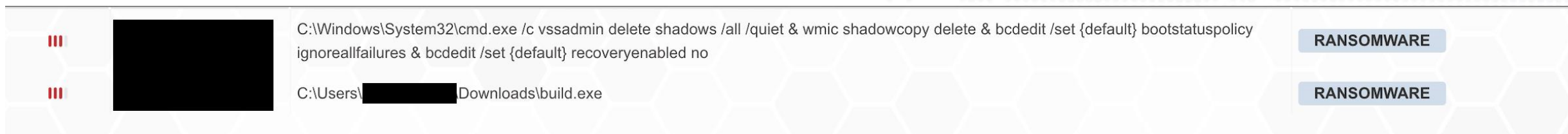
このケースでは、これを用いた情報窃盗までは確認できず。



## 2. Case 2: Lockbit 2.0 ランサムウェア

- 事件の顛末：

### (6) Lockbitの実行



The screenshot shows a Windows command prompt window with two lines of commands. The first line is a complex command to delete shadows and configure boot status policy. The second line shows the execution of a file named build.exe from the Downloads folder. On the right side of the screenshot, there are two blue labels with the word 'RANSOMWARE' written on them.

```
C:\Windows\System32\cmd.exe /c vssadmin delete shadows /all /quiet & wmic shadowcopy delete & bcdedit /set {default} bootstatuspolicy ignoreallfailures & bcdedit /set {default} recoveryenabled no
```

```
C:\Users\[redacted]\Downloads\build.exe
```

RANSOMWARE

RANSOMWARE

**CobaltStrike**を利用してダウンロード、実行したとみられる。

解析したLockbit 2.0には、情報を収集して外部に送信する機能が見つからない？



ここまでの不正アクセスで既に情報を収集し盗んでいるか、「データを公開する」ということ自体がブラフの可能性。

## 2. Case 2: Lockbit 2.0 ランサムウェア

- 判明した攻撃の流れのまとめ：
  - Dark Web等に盗まれた認証情報が出回る
  - 盗まれた資格情報を使って不正アクセス
  - 組織内のシステムからの認証情報窃盗を実施
  - 組織内の端末からCobaltStrikeによるC&Cサーバへのアクセス
  - Lockbitをダウンロードし実行



**Lockbitの実行は、攻撃の最後の段階で使われる傾向**

## 2. Case 2: Lockbit 2.0 ランサムウェア

- おまけ：Lockbit 3.0 も入手したので解析してみた。

類似点：

- ファイルを暗号化し、その鍵を公開鍵？で暗号化している。
- 多重起動抑止をしている。
- 色々な解析回避技術を入れている。
- ファイルオープン処理、暗号化処理のスレッドを分けている。
- AD環境を利用した拡散処理がある。
- アイコンの設定や壁紙の変更、プリンタ出力を行う。
- 自身を削除する。



## 2. Case 2: Lockbit 2.0 ランサムウェア

- おまけ：Lockbit 3.0 も入手したので解析してみた。

相違点：

- ファイルの暗号化は、**xorを使った暗号方式に変更**。
- **先頭の4096バイトしか暗号化しないおサボりを止めた**。
- 多重起動防止も、使用するAPIを変更(NtCreateMutant → OpenMutexとCreateMutexの併用)。
- 解析回避技術は以前より執拗になり、特にデバッガ回避が増えていた（ブログに幾つか書いた）。

## 2. Case 2: Lockbit 2.0 ランサムウェア

- おまけ：Lockbit 3.0 も入手したので解析してみた。

相違点：

- 被害ファイルの拡張子を、固定の「.lockbit」からランダムなものに変更。これにより、ちょっと検知が面倒くさくなった。アイコンはこの拡張子に紐づけするよう変更。
- ファイルを開くスレッドと暗号化スレッド(複数起動)を分け、I/O Completion でスレッド間通信を行っているが、使用しているAPIが全く異なる。

## 2. Case 2: Lockbit 2.0 ランサムウェア

- おまけ：Lockbit 3.0 も入手したので解析してみた。

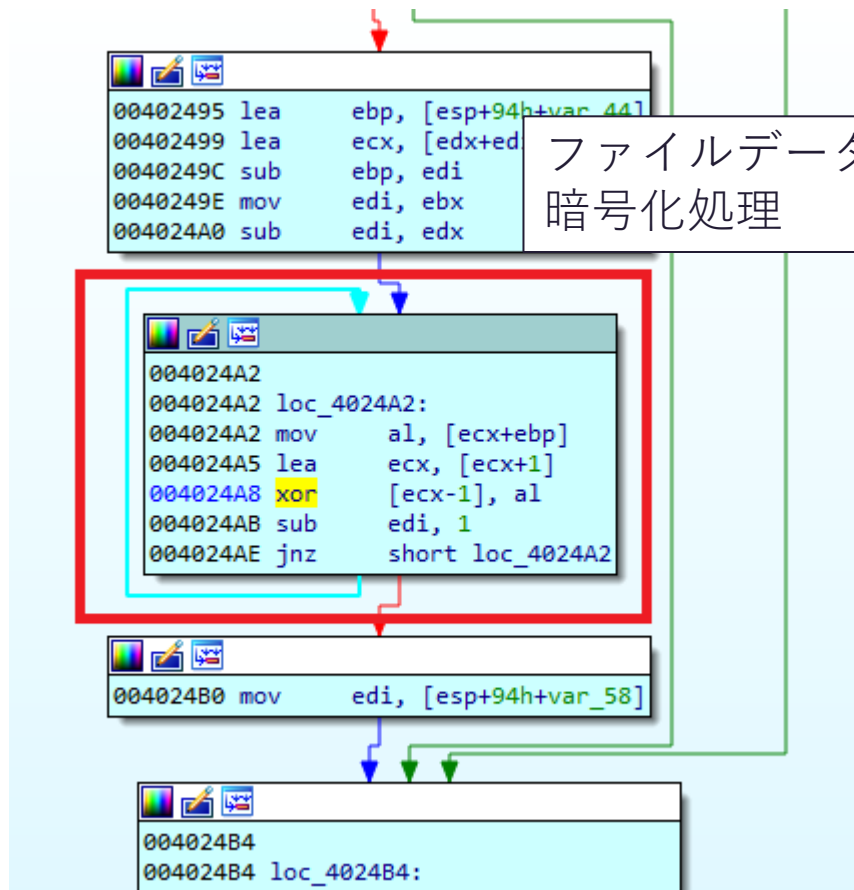
相違点：

- 拡張子の判定が**ベタ書きから32ビットハッシュによる判定に変更**。
- 無駄に？力が入っていた壁紙描画は、逆にえらくシンプルに。
- プリンタの出力枚数が**9999枚→1000枚に減少**。
- 自身を削除する方法は、削除プログラムを解凍、プロセス実行。



## 2. Case 2: Lockbit 2.0 ランサムウェア

- おまけ：Lockbit 3.0 も入手したので解析してみた。



脅迫文の壁紙

**LockBit Black**

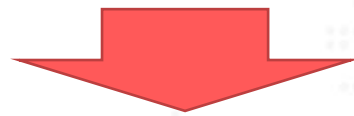
**All your important files are stolen and encrypted!  
You must find gT5nztj5k.README.txt file  
and follow the instruction!**

## 2. Case 2: Lockbit 2.0 ランサムウェア

- おまけ：Lockbit 3.0 も入手したので解析してみた。

感想：

- 殆ど流用したコードが無く、**1から作り直したレベル**。  
→以前よりもコードのレベルは高くなっていた。



- ハッキリ言って、**要求仕様を変えないまま別人が作ったレベル**。  
→研究しているマルウェアの筆跡鑑定やインテリジェンス抽出の対象にすると面白そう！（**喜ぶんじゃない・・・**）

# 3. 台湾でも Emotet 事例あり

- Emotetは、C&Cから落ちてくる追加コードが重要。
- 以前と異なり、ファイルに出力せず実行するようになっている。
- 落ちてきた機能例：WNetOpenEnumWを利用して感染拡大する機能。

debug106:0000000047F926F mov [rsp+44h], eax  
debug106:0000000047F9273 xor dword ptr [rsp+44h], 0B37FEE3h  
debug106:0000000047F927B mov eax, [rsp+44h]  
debug106:0000000047F927F mov eax, [rsp+80h]  
debug106:0000000047F9286 mov eax, [rsp+40h]  
debug106:0000000047F928A mov eax, [rsp+48h]  
debug106:0000000047F928E call GetProcAddressByHash\_sub\_47FB030  
debug106:0000000047F9293 mov cs:qword\_47FF0C8, rax  
debug106:0000000047F929A  
debug106:0000000047F929A loc\_47F929A: ; CODE XREF: debug106:0000000047F929A  
debug106:0000000047F929A mov r9d, 1  
debug106:0000000047F92A0 mov r8, rsi  
debug106:0000000047F92A3 mov rdx, rdi  
debug106:0000000047F92A6 mov rcx, rbx  
debug106:0000000047F92A9 mov rbx, [rsp+70h]  
debug106:0000000047F92AE mov rsi, [rsp+78h]  
debug106:0000000047F92B3 add rsp, 60h  
debug106:0000000047F92B7 pop rdi  
debug106:0000000047F92B8 jmp rax

RAX 00007FFBE68D31D0 mpr.dll:mpr\_WNetAddConnection2W  
RBX 000000003039428 debug066:000000003039428  
RCX 00000000058EF018 Stack[000024C4]:00000000058EF018  
RDX 0000000000000000 Stack[000024C4]:0000000000000000  
RSI 0000000000000043  
RDI 0000000002FE8AC0 debug066:0000000002FE8AC0  
RBP 00000000058EF0A0 Stack[000024C4]:00000000058EF0A0  
RSP 00000000058EEF98 Stack[000024C4]:00000000058EEF98  
RIP 0000000047F92B8 debug106:0000000047F92B8  
R8 0000000000000000 Arg 3  
R9 0000000000000001 Arg 4  
R10 00007FFBE68E445F mpr.dll:00007FFBE68E445F  
R11 00000000058EEEB0 Stack[000024C4]:00000000058EEEB0

アクセスに利用するID  
アクセスに利用するパスワード

Arg1: 0x058EF018 - NETRESOURCE structure  
Arg2: 0x00000000 - Password  
Arg3: 0x00000000 - User name  
Arg4: 0x00000001 - Flag (CONNECT\_UPDATE\_PROFILE)

## 4. まとめ

- 2つのランサムウェアとEmotetの解析結果の紹介
- ランサムウェアの感染に繋がる認証情報の漏洩を発端とした不正アクセスの事例を紹介

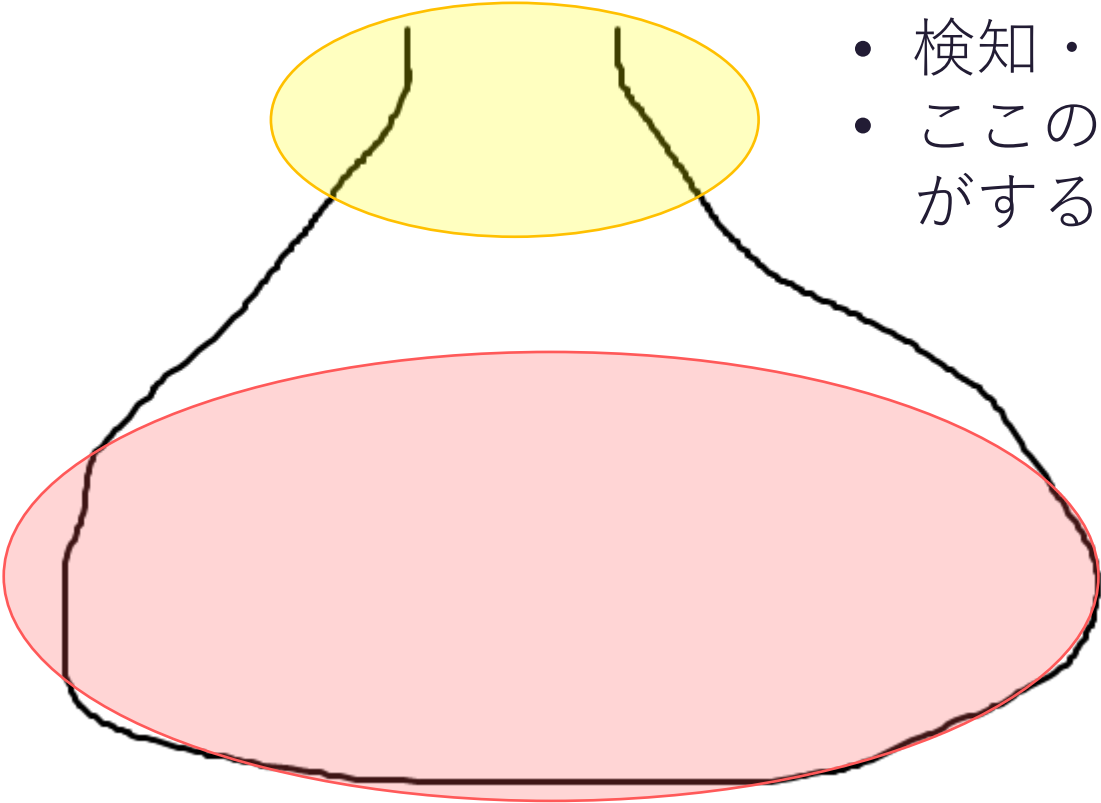


- マルウェアの解析では、IoCだけでなく、どのようなことができるか、またはできないかといった能力の分析も、サイバー攻撃の分析では重要。
- 攻撃の手口やマルウェアの機能を知ることによって、検知やブロックできる点を発見し、対策に生かすことができる。
- 不足している対策が何かについて気づいてもらいたい。



# 4. まとめ

## ● マルウェア解析のイメージ？

- 
- 検知・識別に使うならこのあたりの情報で十分。
  - この省力化・合理化の研究が多く、進んでる気がする？

- 時間がめっちゃかかる。
- 古手のアセンブラプログラマ世代が昔取った杵柄でやってるケースが多め？
- ここで得られる情報の活用がまだ不十分？



CyCraft | Website



CyCraft | Medium



CyCraft | Twitter