SECURITY RESEARCH CENTER
powered by macnica

標的型攻撃の実態と対策アプローチ

日本を狙うサイバーエスピオナージ(標的型攻撃)の動向2023年度

2024年5月1日

株式会社マクニカ

目次

はじめに

マクニカでは、セキュリティ研究センターを中心に、日本の組織に着弾する標的型攻撃（サイバーエスピオナージ）を 2014 年から分析してきました。情報窃取を目的としたこの種のサイバー攻撃は、ランサムウェアによる攻撃と違って長期間に渡って侵害に気づかない組織が多く、表面化するケースも比較的少ないため、情報共有がされにくいと言えます。しかし、国内外のサイバーセキュリティ業界の長年の努力によって今日まで収集された攻撃痕跡（マルウェア、攻撃インフラ、ログ）を分析していくと、各攻撃グループの TTPs、目的や意図、スキルレベルなどが、徐々に浮き彫りになってきています。このような取り組みは、組織を超えた戦略的な情報共有とインテリジェンスへの昇華によって成り立ちます。

本レポートでは、2023 年度（2023 年 4 月から 2024 年 3 月）に観測された、日本の組織から機密情報（個人情報、政策関連情報、製造データなど）を窃取しようとする攻撃キャンペーンの分析結果について注意喚起を目的として記載しています。ステルス性の高い遠隔操作マルウェア（RAT）を用いた事案を中心に、新しい攻撃手法やその脅威の検出について記載しています。レポートの最後には、本文中で紹介した攻撃キャンペーンで使われたインディケータを掲載しています。

2023 年度は、ここ 10 年ほど継続しているスピアフィッシングに加え、外部公開アセットからの侵入、更には USB や Wi-Fi アクセスポイントからの物理的な侵入なども観測され、「標的型攻撃」に分類される攻撃は活発であった印象があります。侵入方法においては、特に様々な手法を駆使して標的組織に侵入しようとしているような傾向が見られました。侵入方法の中で外部公開アセットの脆弱性を攻撃して侵入するケースが増加している事が一因となり、これまで観測のなかったと思われる攻撃グループの観測も見られており、注意が必要と思われます。また、攻撃者間での攻撃ツールのシェアリング、オープンソースの遠隔操作ツールなどの活用などもあり、攻撃者の帰属に難しさが生じています。

これまでの標的型攻撃の傾向と比べると、メディアや安全保障・外交関連といったここ 10 年ほど継続して標的とされている業界がある一方、広い分野に渡る製造業などは攻撃者の目標の変化があり、ひそかに新しい標的に攻撃が来るような事態が起こり始めていると思われます。残念ながら、標的型攻撃に分類される特定の組織だけを狙った攻撃はまだまだ継続しており、警戒が必要な状況です。このような日本企業の産業競争力を徐々に蝕んでいく標的型攻撃に対して、今後も粘り強い分析と啓蒙活動に取り組んでいく所存です。

本書には、検体解析のような技術的に深い内容も含まれるため、少々難しいと感じられる読者もいらっしゃると思います。そのような場合、難解な箇所はスキップして前半の標的型攻撃の対象となった業種と攻撃が発生した国内組織の拠点地域、後半の攻撃への対策といった箇所だけでも対策の参考にして頂ければと思います。

攻撃のタイムラインと攻撃が観測された業種

2023 年度は、昨年度までの攻撃[1]と比較して新たな攻撃グループが多く出現しています。Barracuda ESG の CVE-2023-2868 脆弱性攻撃から組織に侵入した UNC4841[2] 攻撃グループ、RatelS[3] 遠隔操作マルウェアでインフラ関連組織を標的に攻撃を行った TELEBOYi 攻撃グループ[4]、BLOODALCHEMY[5][6] 遠隔操作マルウェアで製造関連組織を標的に攻撃を行った Vapor Panda 攻撃グループ[7]、Ivanti 社の CVE-2023-46805/CVE-2024-21887 脆弱性を攻撃して学術系並びに製造業種で攻撃が観測された UNC5221[8]攻撃グループ です。一方で、2022 年度やそれ以前にも日本を標的としていた攻撃グループとして、APT10 攻撃グループの LODEINFO マルウェア[9] を使った攻撃、Tropic Trooper 攻撃グループの EntryShell[10] マルウェアを使った攻撃、Mustang Panda 攻撃グループの PlugX[11] や PUBLOAD[12][13] マルウェアを使った攻撃、攻撃主体は分析中ではあるものの中国の攻撃グループで利用が報告されているオープンソースの Stowaway を悪用した攻撃などが観測されました。

表 1. タイムチャート

	23/04	23/05	23/06	23/07	23/08	23/09	23/10	23/11	23/12	24/01	24/02	24/03
APT10 (LODEINFO)	?		?									
UNC4841		政府										
Tropic Trooper		製造										
Vapor Panda		製造										
Mustang Panda			製造									政府
TELEBOYi					インフラ							
NA							製造					
UNC5221										学術		

政府関連　製造関連　インフラ　学術　不明

1 https://www.macnica.co.jp/business/security/security-reports/pdf/cyberespionage_report_2022.pdf
2 https://www.mandiant.com/resources/blog/barracuda-esg-exploited-globally
3 https://www.lac.co.jp/lacwatch/report/20230914_003513.html
4 https://jsac.jpcert.or.jp/archive/2024/pdf/JSAC2024_1_8_yi-chin_yu-tung_en.pdf
5 https://www.elastic.co/security-labs/disclosing-the-bloodalchemy-backdoor
6 https://www.botconf.eu/botconf-presentation-or-article/into-the-vapor-to-tracking-down-unknown-pandas-claw-marks/
7 https://www.crowdstrike.com/adversaries/vapor-panda/
8 https://www.mandiant.jp/resources/blog/investigating-ivanti-zero-day-exploitation
9 https://blogs.jpcert.or.jp/ja/tags/lodeinfo/
10 https://www.virusbulletin.com/uploads/pdf/conference/vb2023/slides/Slides-Unveiling-Activities-of-Tropic-Trooper.pdf
11 https://unit42.paloaltonetworks.jp/plugx-variants-in-usbs/
12 https://csirt-cti.net/2024/01/23/stately-taurus-targets-myanmar/
13 https://unit42.paloaltonetworks.jp/chinese-apts-target-asean-entities/

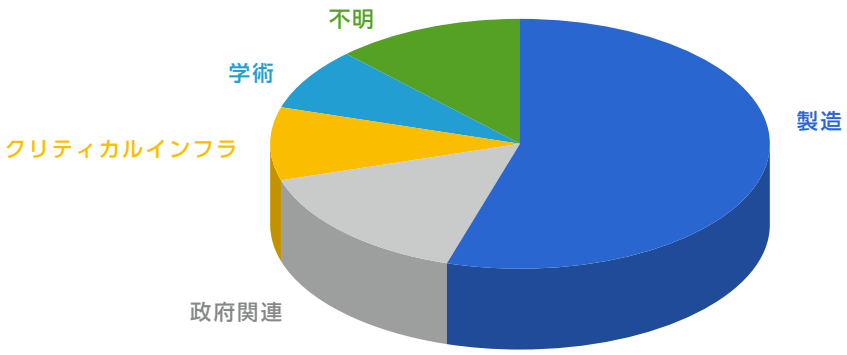


図 1. 標的業種の割合(2023 年度)

標的型攻撃の侵入パターンとして、これまではスピアフィッシングがもっとも多く観測されていましたが、2023年度は外部公開アセットの脆弱性を攻撃して侵入するケースがもっとも多くなっています。それに加え、製造関連組織の海外拠点などでは組織の近傍で Wi-Fi アクセスポイントを悪用して組織に侵入する、USB デバイスからマルウェアに感染するといった手法も観測されています。

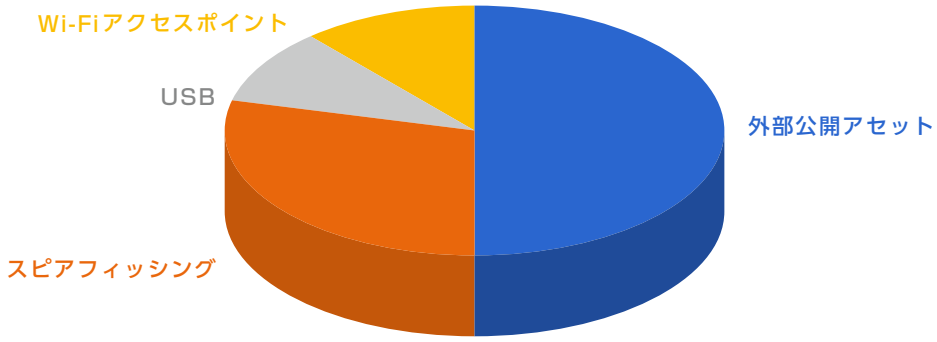


図 2. アタックサーフェースの割合(2023 年度)

攻撃の概要

以下は、４月から３月までの月ごとに観測された攻撃の概要を記載しています。

2023年４月

APT10 LODEINFO（標的：不明）

APT10 攻撃グループの LODEINFO v0.6.9 がパブリックマルウェアリポジトリにアップロードされた事を観測しました。アップロードされたファイル名「Elze.exe」「frau.dll」「Elze.exe_」が LODEINFO v0.6.8 のダウンローダがドロップするファイル名と同じである事、v0.6.9 の感染フローで”Remote Template Injection”のテクニックの利用を観測したレポートが公開されている事[14]から、４月にアップロードされたファイルもスピアフィッシングメールでドキュメントファイルが配送されて最終的にドロップされたものであると推測しています。機能的には v0.6.8 との大きな違いはみられませんでした。

```
v27 = v23[3];
strcpy(v104, "v0.6.9");
v28 = (v27->lstrlen)(v104);
v29 = v23[1];
str_len = v28;
sub_469C95(v23, v29 + v28, 0);
sub_4677D5((v29 + v23[2]), v104, str_len);
*(v23[2] + v23[1]) = 0;
sub_469C95(v23, v23[1] + 1, '-');
sub_469C95(v23, v23[1] + 1, '1');
```

図3. ペイロードに埋め込まれているバージョン情報

2023年５月

Tropic Trooper（標的：製造関連）

Cobalt Strike や EntryShell マルウェアを遠隔操作ツールとして利用した Tropic Trooper 攻撃グループの攻撃が、製造関連企業の中国拠点で観測されました。スピアフィッシングメールや中国で人気のチャットツールを使ってマルウェアが配送された他、組織の Wi-Fi アクセスポイントと同様の偽アクセスポイントを使って情報を窃取しつつ、窃取した情報で正規の Wi-Fi アクセスポイントから侵入し、リモートログオンに成功した PC にマルウェアを設置するといった物理的な侵入も行っていました。また、攻撃の２次バックドアとして FamousSparrow 攻撃グループの SparrowDoor マルウェアと関連の見られる CrowDoor[15] バックドアや Visual Studio Code の Remote Tunnel を使った攻撃者による遠隔コマンドの実行[16]が観測されました。

14 https://blog.itochuci.co.jp/entry/2024/01/24/134047
15 https://blog.itochuci.co.jp/entry/2023/10/06/003000
16 https://jsac.jpcert.or.jp/archive/2024/pdf/JSAC2024_2_3_sasada_hazuru_en.pdf

```
case 0x2347137:
  v16 = dword_7B268C;
  if ( !dword_7B268C )
  {
    v16 = sub_78FD85(1);
    v38 = v16;
    dword_7B268C = v16;
  }
  v17 = CreateThread(0, 0, sub_762490, v16, 0, 0);
  goto LABEL_15;
case 0x234713B:
  sub_762D80(&v51, v4);
  return result;
case 0x2347140:
  if ( *(v4 + 8) != 4 )
    break;
```

図4. CrowDoorバックドアのコマンド命令

Vapor Panda（標的：製造関連）

製造関連企業の VPN 装置に窃取したアカウントで不正侵入した後、組織内のリモート接続可能な Windows サーバに DLL サイドローディングを行うマルウェアを設置した攻撃が検出されました。設置されたファイルは、ブラザー工業社の正規実行ファイル BrDifxapi.exe とローダ DLL の BrLogApi.dll、暗号ファイルの DIFX でした。BrDifxapi.exe が実行されると、BrLogApi.dll が DIFX ファイルを読んで AES で復号してシェルコードを実行します。このシェルコードから展開されるバックドアペイロードは、BLOODALCHEMY で C2 サーバが HTTPS[:]cdn1ac7bdd3[.]jptomorrow[.]com でした。

```
ONLOGON
Test
%windir%\system32\SearchIndexer.exe
%windir%\system32\wininit.exe
%windir%\system32\taskhost.exe
%windir%\system32\svchost.exe
%windir%\system32\wininit.exe
%windir%\system32\taskeng.exe
%windir%\system32\taskhost.exe
%windir%\system32\conhost.exe
%windir%\system32\taskhost.exe
%windir%\system32\rundll32.exe
sahkjdfRFVUH345677yghbv2wsdcbhj345tygvWSDRFGTY
-----BEGIN RSA PUBLIC KEY-----
MIGJAoGBAL5ERMwFAl9kujGxbDzbK1UU6otN21m2Xr0dv5fx+v31ZxeVGvl+c2KC
9ZMU6BxKxY7XTPNELV0Wgo/9M3BomKX8a3REvRUUBgupIFigtenI6xQbdwWQ6T2G
3MRSElh0ppbOow+8e2F6p3M0hTp6Q76XEu9V3VRMGKK6XfibtrK3AgMBAAE=
-----END RSA PUBLIC KEY-----
HTTPS[:]//cdn1ac7bdd3.jptomorrow[.]com:443
```

図5. BLOODALCHEMYマルウェアのコンフィグに含まれる文字列

2023年6月

Mustang Panda（標的：製造関連）

製造関連企業のベトナム拠点で PC に差し込まれた USB リムーバブルドライブから、PlugX マルウェアへの感染を狙った攻撃が検出されました。USB リムーバブルドライブは、以前に同じ PlugX に感染した別の PC で使われた事があると思われ、PlugX マルウェアにより USB リムーバブルドライブに自身をコピーして別の PC で感染するように細工が施されていました。リムーバブルドライブのルートフォルダには、リムーバブルドライブへのアクセスを行うショートカットファイルと隠しフォルダが設定され、ショートカットファイルを実行する事で、隠しフォルダの PlugX マルウェアの PC への設置と実行がなされつつ、リムーバブルドライブの本来ファイルが表示される攻撃となっていました。

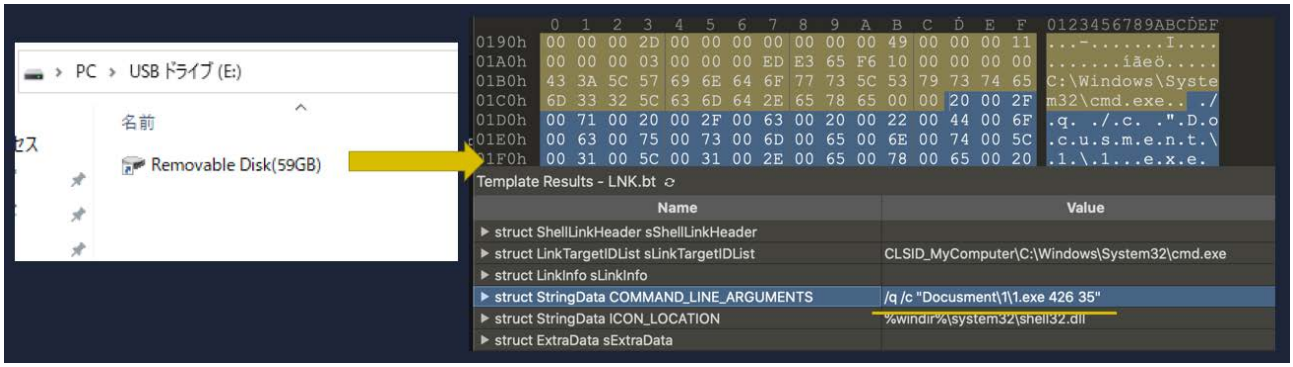


図6. PlugXの感染が仕込まれたリムーバブルフォルダ

2023年7月

APT10 LODEINFO（標的：不明）

APT10 攻撃グループの LODEINFO v0.7.1 のダウンローダがパブリックマルウェアリポジトリにアップロードされた事を観測しました。ドキュメントに含まれるマクロは複数の base64 でエンコードされた文字列を連結してデコードすることでシェルコードのバイト文字列を生成します。マクロを有効にした後に、表示されている「OK」ボタンを押下すると、外部サーバから LODEINFO の実行に必要な複数ファイルがバンドルされた暗号化ファイルがダウンロードされ、復号・実行されます。

また、ダウンローダである doc ファイルには button オブジェクトが多数埋め込まれており、ファイルサイズが約 6.8M に肥大化していました。これらの特徴はサンドボックスによる動的解析とシグネチャベースのアンチウイルス製品の検知回避を意図したものであるとみています。

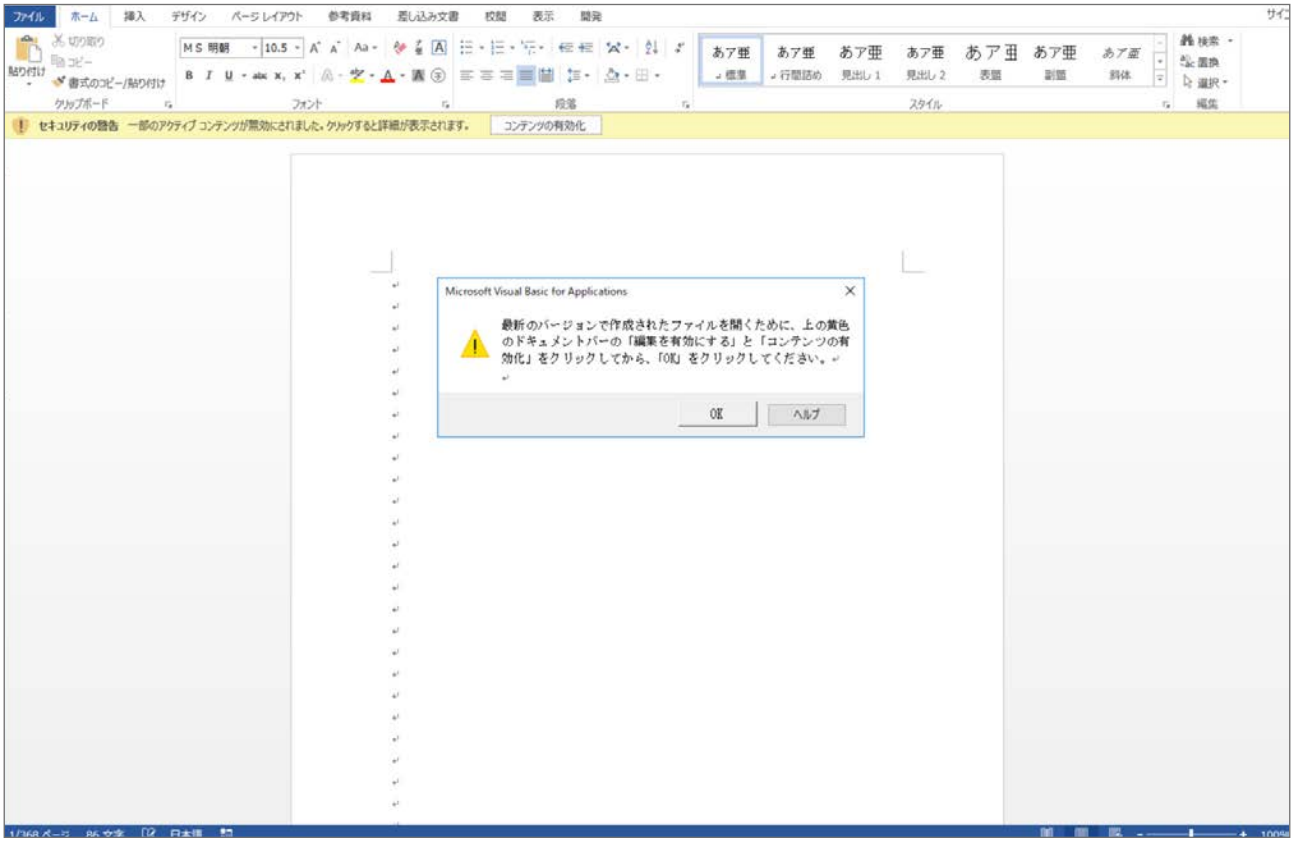


図7. v0.7.1ダウンローダ ドキュメントファイル

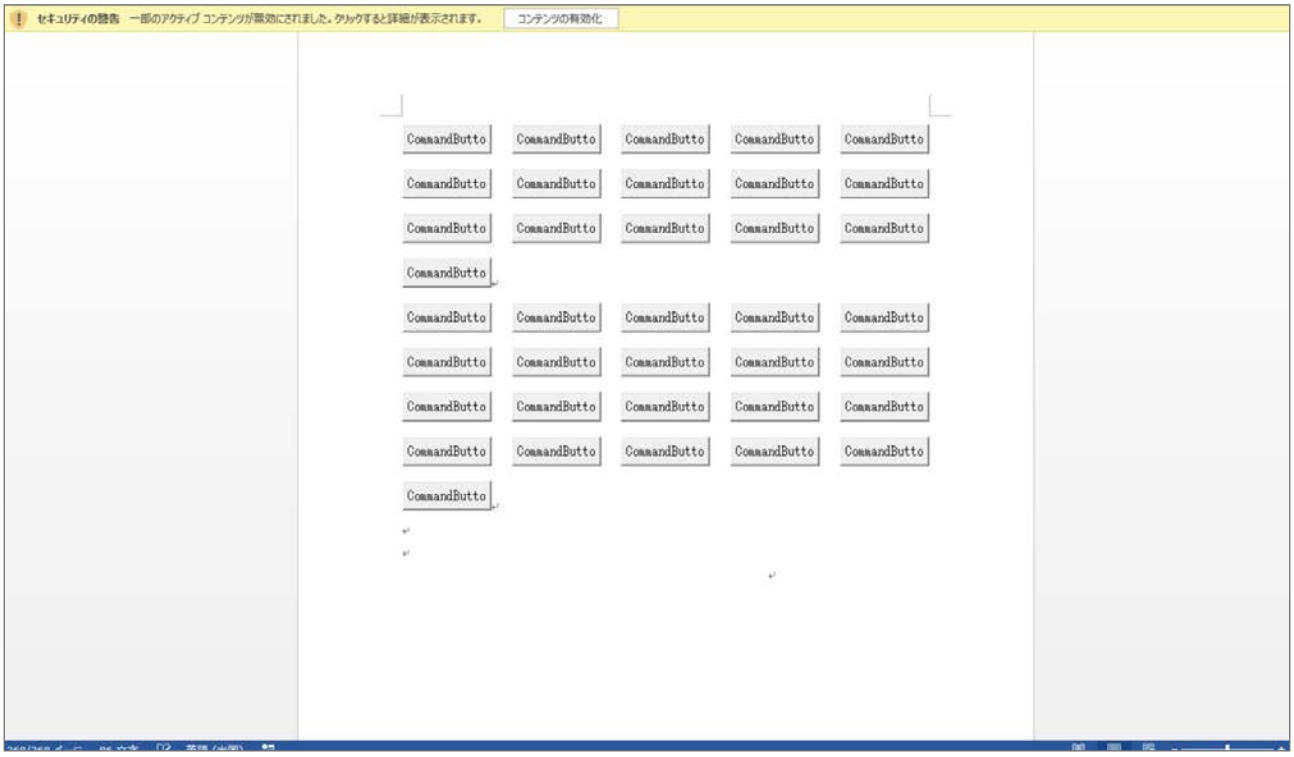


図 8. ドキュメントファイルに埋め込まれた多数のbuttonオブジェクト

シェルコードはハードコードされたバイト文字列をシングルバイトで XOR して URL の文字列を準備し、URLDownloadToFile() でファイルをダウンロードします。今回のバージョンではダウンロードしたファイルは PEM 形式の証明書として偽装されており、"-----BEGIN CERTIFICATE-----" から "-----END CERTIFICATE-----" までの文字列を base64 でデコードして、先頭から 3 バイト目以降を AES で復号し、続いて AES で復号したデータの 3 バイト目の値を使ってシングルバイトの XOR でデータを復号します。ファイルはこれまでと同様に 3 つのファイルが含まれており、ファイルサイズ、ファイル名、データのフォーマットで分割して %UserProfile%\Downloads にファイルを保存して実行します。

2023年 10月

NA（標的：製造関連）

製造関連企業にて、ウェブサーバにオープンソースの Go 言語で開発された Stowaway が UPX でパックされて設置されました。Stowaway はファイルのアップロード、ダウンロードとリモートシェルの機能を持ったバックドアまたはトラフィックを中継するプロキシで、中国を拠点としたいくつかの攻撃グループによって攻撃に利用されており[17]、中国を拠点とした攻撃グループによる可能性があると思われます。

17 https://blog.exatrack.com/melofee/

図9. Githubで公開されているSTOWAWAY

2024年1月

UNC5221（標的：学術、製造関連）

Ivanti 社 CVE-2023-46805 CVE-2024-21887 の脆弱性をついた攻撃が確認されました。Ivanti 社の Linux OS 上に ELF バイナリとして動作するバックドアやウェブシェルとして動作するよう改竄されたファイルなどの他、クレデンシャルをダンプした結果の出力ファイルなどが国内でも複数の組織で確認されました。

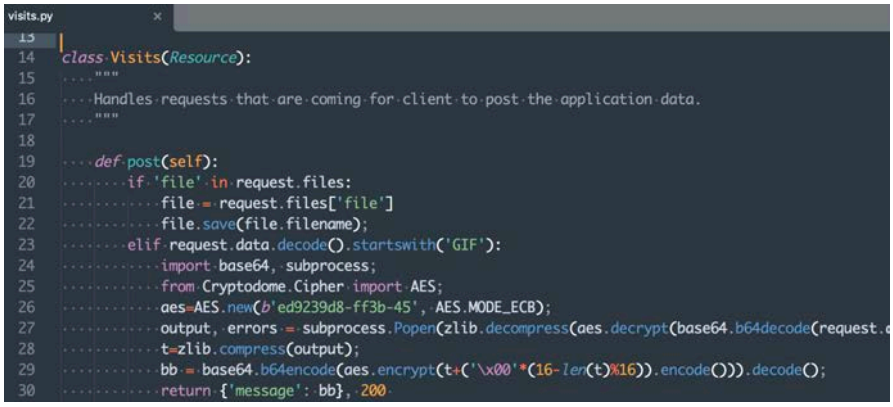


図10. Ivanti社VPN製品上でバックドアコードを追加して改竄されたpythonファイル

2024年3月

Mustang Panda（標的：政府関連）

日本を含むアジア諸国の外交、政府関連を狙ったスピアフィッシング攻撃キャンペーンが観測されました。観測された1つのファイルは「Talking_Points_for_China.zip」です。Zipファイルには、Talking_Points_for_China.exeとKeyScramblerIE.DLLが含まれており、Talking_Points_for_China.exeを実行すると、KeyScramblerIE.DLLがロードされてDLLに含まれるシェルコードが実行され、103.27.109[.]157 TCP/443と通信を行い、更なるペイロードのダウンロードを試みます。

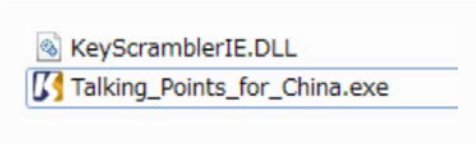


図11. Talking_Points_for_China.zipに含まれるファイル

新しいTTPsやRATなど

ここでは、先に引用させて頂いた公開されている調査報告ではまだ触れられていない観測や分析を中心に、少し詳しく紹介します。

Mustang Panda (PUBLOAD)

攻撃キャンペーンの概要

2022 年頃から観測されている Mustang Panda の日本を含むアジア諸国の外交、政府関連を狙ったスピアフィッシング攻撃キャンペーンの2024年3月に観測されたPUBLOAD検体について記載します。スピアフィッシングメールに添付の「Talking_Points_for_China.zip」に含まれる Talking_Points_for_China.exe と KeyScramblerIE.DLL ですが、DLL ファイルにはシステム属性と隠し属性が設定されており、通常の表示設定では見ることができません。

Talking_Points_for_China.exe をユーザが実行すると、KeyScramblerIE.DLL がロードされて、C:\Users\\Public\Libraries\SmileTV フォルダにこの 2 つの EXE と DLL ファイルがコピーされます。

KeyScramblerIE.DLL のエクスポート関数「KSInit」にマルウェア由来のコードが含まれており、HKYE_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run に "KeyScrambler" を追加して、コピーした EXE ファイルの自動起動エントリーが作成されます。

```
void aa_launchShellCode()
{
  UINT_PTR (__stdcall *v0)(HWND, UINT, WPARAM, LPARAM); // eax
  HLOCAL v1; // edi
  UINT_PTR (__stdcall *v2)(HWND, UINT, WPARAM, LPARAM); // esi
  struct tagCHOOSECOLORW v3; // [esp+8h] [ebp-2Ch] BYREF
  void *v4; // [esp+2Ch] [ebp-8h] BYREF
  HLOCAL hMem; // [esp+30h] [ebp-4h] BYREF

  printf("Start...Code_Return_value\n");
  aa_clock();
  printf("Start......Code_Return_value Ok\n");
  sub_70061820();
  sub_700619F0();
  sub_70061A50();
  aa_decrypt_shell(&hMem, &v4);
  v0 = VirtualAlloc(0, Size, 0x3000u, 0x40u);
  v1 = hMem;
  v2 = v0;
  memcpy_0(v0, hMem, Size);
  memset(&v3.hwndOwner, 0, 16);
  v3.lCustData = 0;
  v3.lpTemplateName = 0;
  v3.lStructSize = 36;
  v3.Flags = 16;
  v3.lpfnHook = v2;
  ChooseColorW(&v3);
  free(v1);
  LocalFree(v1);
  operator delete[](v4);
}
```

図12. PUBLOAD の ChooseColorW() によるシェルコード実行

DLL ファイルに埋め込まれているエンコードされたシェルコード文字列は、これと別に用意された文字列のテーブルを使って XOR でデコードされ、ChooseColowrW() のハンドラ関数として登録し起動されます。シェルコードは E9 のジャンプ命令で 2 回ジャンプを経て実際のコードが開始される作りになっており、処理の最初に ror13AddHash32 のハッシュ値を使い Win32 API のアドレスをロードします。

図13. PUBLOAD のシェルコード

シェルコードの処理は 103.27.109[.]157 TCP/443 と通信を行い、2nd Stage のシェルコードをダウンロードして実行する作りになっています。通信プロトコルはこれまで報告されているもの[18] と同じ鍵と RC4 で通信データを暗号化します。

データフォーマット：17 03 03 + Length (2byte) + RC4 Encrypted data

18 https://www.trendmicro.com/ja_jp/research/23/e/earth-preta-updated-stealthy-strategies.html

鍵 :0x785a124d7514141l6c0271155a7305087014653b64422223200000000000000000

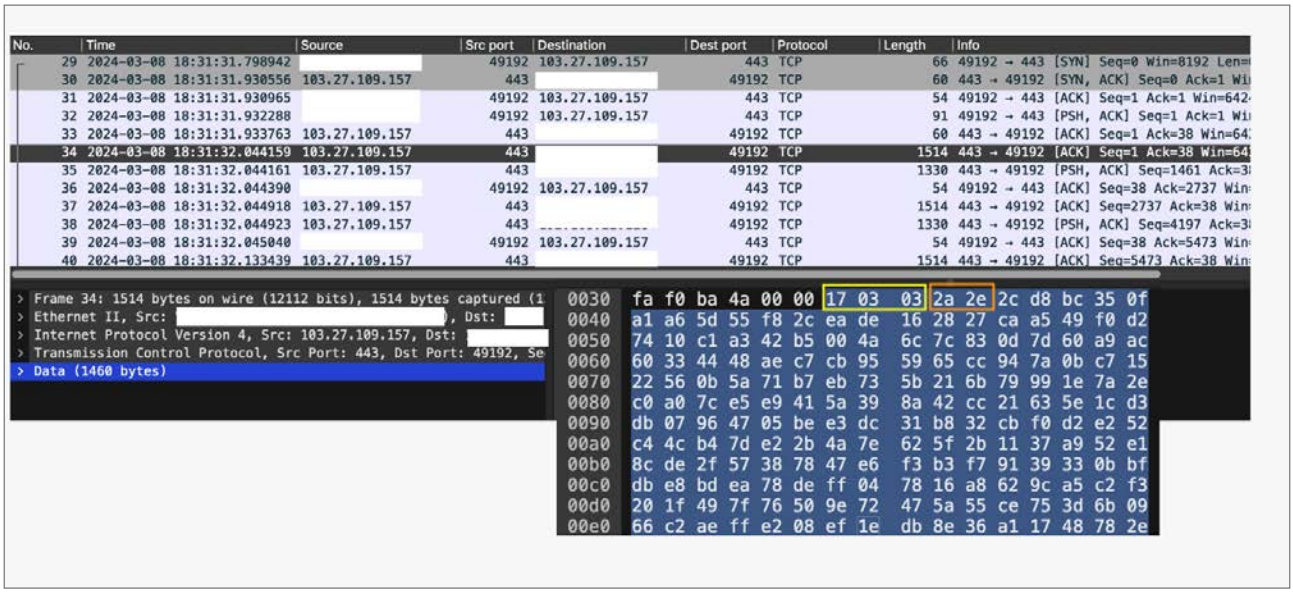


図14. RC4で暗号化された通信データ

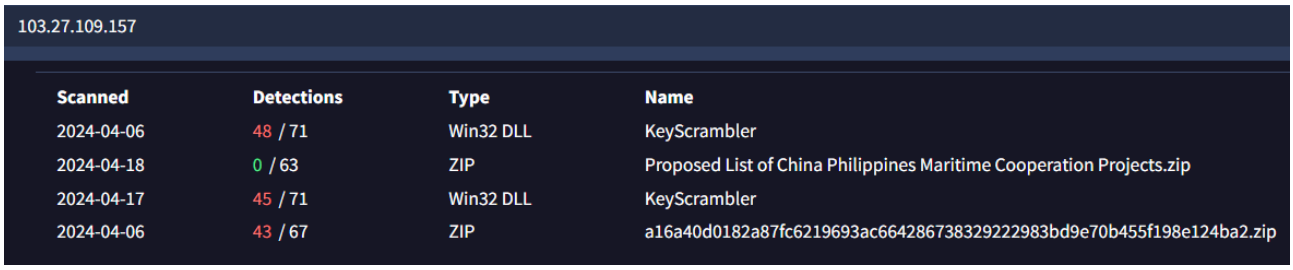
また、観測された 2nd Stage のシェルコードは DLL から展開されるシェルコードと同じ宛先に同じプロトコルで通信するもので、11 個の遠隔操作命令を持っています。今回分析したペイロードもこれまで報告されているもの[19]と同様と思われます。

表2. PUBLOAD 2nd Stageペイロードの遠隔操作命令

No	ID	機能
1	0x04	操作前のスリープ時間（ミリ秒）更新
2	0x05	キープアライブ
3	0x03	不明
4	0x01	ファイル削除
5	0x1A	不明
6	0x1B	ファイルアップロード
7	0x1D	ファイル追加書き込み
8	0x1E	リモートシェルオープン
9	0x1F	リモートシェル コマンド実行
10	0x30	リモートシェル コマンド実行結果表示
11	0x20	リモートシェルクローズ

19 https://lab52.io/blog/1943-2/

Mustang Panda 攻撃グループは、103.27.109[.]157 を C2 サーバのアドレスとしてしばらく攻撃を継続していると思われ、いくつかの関連ファイルがパブリックマルウェアリポジトリにアップロードされています。そのうちの 1 つ「Proposed List of China Philippines Maritime Cooperation Projects.zip」ファイルに含まれる PUBLOAD の DLL ファイルはアンチウィルスマルウェアでの検出が非常に悪いものと思われます。



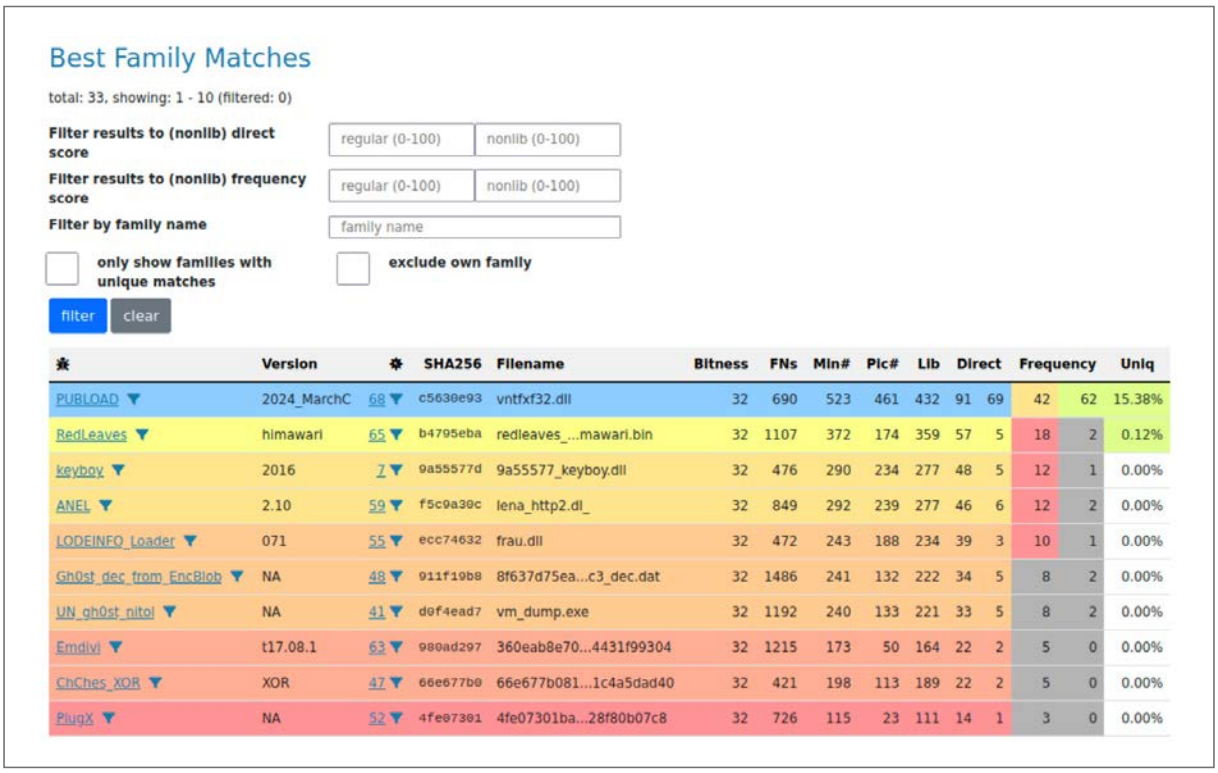
103.27.109.157

Scanned	Detections	Type	Name
2024-04-06	48 / 71	Win32 DLL	KeyScrambler
2024-04-18	0 / 63	ZIP	Proposed List of China Philippines Maritime Cooperation Projects.zip
2024-04-17	45 / 71	Win32 DLL	KeyScrambler
2024-04-06	43 / 67	ZIP	a16a40d0182a87fc6219693ac66428673832922298 3bd9e70b455f198e124ba2.zip

図15. 103.27.109[.]157と関連のあるマルウェア

「Proposed List of China Philippines Maritime Cooperation Projects.zip」ファイルに含まれる PUBLOAD の DLL ファイルをこの他の PUBLOAD マルウェアの DLL ファイルとの類似性をコード類似性分析ツール MCRIT[20] で分析をさせてみると、他のマルウェアと比較すると PUBLOAD と類似性があるものと結果が得られます。一方で静的分析を進めてみると、ローダ DLL としての主な処理箇所であるシェルコードを準備して実行する箇所の変更が大きく、アンチウイルス製品でのファイル検出が難しくなっていると思われます。

20 https://github.com/danielplohmann/mcrit



Best Family Matches

total: 33, showing: 1 - 10 (filtered: 0)

Filter results to (nonlib) direct score: regular (0-100) | nonlib (0-100)

Filter results to (nonlib) frequency score: regular (0-100) | nonlib (0-100)

Filter by family name: family name

only show families with unique matches

exclude own family

filter clear

	Version		SHA256	Filename	Bitness	FNs	Min#	Pic#	Lib	Direct		Frequency		Uniq
PUBLOAD	2024_MarchC	68	c5630e93	vntfxf32.dll	32	690	523	461	432	91	69	42	62	15.38%
RedLeaves	himawari	65	b4795eba	redleaves_...mawari.bin	32	1107	372	174	359	57	5	18	2	0.12%
keyboy	2016	7	9a55577d	9a55577_keyboy.dll	32	476	290	234	277	48	5	12	1	0.00%
ANEL	2.10	59	f5c9a30c	lena_http2.dl_	32	849	292	239	277	46	6	12	2	0.00%
LODEINFO_Loader	071	55	ecc74632	frau.dll	32	472	243	188	234	39	3	10	1	0.00%
Gh0st_dec_from_EncBlob	NA	48	911f19b8	8f637d75ea...c3_dec.dat	32	1486	241	132	222	34	5	8	2	0.00%
UN_gh0st_nitol	NA	41	d0f4ead7	vm_dump.exe	32	1192	240	133	221	33	5	8	2	0.00%
Emdivi	t17.08.1	63	980ad297	360eab8e70...4431f99304	32	1215	173	50	164	22	2	5	0	0.00%
ChChes_XOR	XOR	47	66e677b0	66e677b081...1c4a5dad40	32	421	198	113	189	22	2	5	0	0.00%
PlugX	NA	52	4fe07301	4fe07301ba...28f80b07c8	32	726	115	23	111	14	1	3	0	0.00%

図16. MCRITでのPUBLOAD DLLの類似性の分析

PUBLOAD のコード変更ですが、これまでの PUBLOAD 検体に見られるような CreateEvent() といった関数や XOR でのデコード処理が無くなっています。エンコードされたシェルコードをデコードする処理は、ユニークなものになりメモリのスタックにコピーした様々な動物や植物などの文字列を検索マーカとして 0x28 バイト先のオフセットにあるバイトを繰り返しコピーしてシェルコードを生成するような作りになっています。生成されたシェルコードは EnumPropsExW() 関数に登録されて実行されます。また、ここで生成されるシェルコードの作りと通信先に変更はありません。

```
sub_10003D60(*(*(a1 + 4) + 4));
*(*(a1 + 4) + 4) = *(a1 + 4);
**(a1 + 4) = *(a1 + 4);
v1 = 0;
*(*(a1 + 4) + 8) = *(a1 + 4);
*(a1 + 8) = 0;
v2 = aCrocodile;
do
{
  v4 = 15;
  v3[4] = 0;
  LOBYTE(v3[0]) = 0;
  sub_10004790(v3, v2, strlen(v2));
  v5 = 0;
  *aa_RetKeywd_p0x28(a1, v3) = v1;
  v5 = -1;
  if ( v4 >= 0x10 )
    operator delete(v3[0]);
  v2 += 0x28;
  ++v1;
}
while ( v2 < &end );
```

図17. シェルコードの生成ルーチン

```
v1 = operator new(0x24u);
if ( !v1 )
{
  v5 = 0;
  std::exception::exception(pExceptionObject, &v5);
  pExceptionObject[0] = &off_1001C22C;
  _CxxThrowException(pExceptionObject, &stru_10020A14);
}
*v1 = v1;
v7[0] = v1;
v1[1] = v1;
LOBYTE(v13) = 1;
aa_Animal_Keywd(v7);
aa_Shellcode_withKeywd(v10);
aa_SearchKeywd_Shellcode(v7, &lpEnumFunc, &v4);
TopWindow = GetTopWindow(0);
EnumPropsExW(TopWindow, lpEnumFunc, 0);
sub_10003780();
operator delete(v7[0]);
v13 = 2;
sub_10004E50(*v11, v11);
operator delete(v11);
}
```

図18. シェルコードの実行関数 EnumPropsExW()

メモリ上で実行されるシェルコードに変更はないため、メモリスキャナーで検出が有効と思われますが、注意が必要です。EnumPropsExW() 関数に登録されたコールバック関数はメモリの実行領域に置かずに実行する事が可能と思われ、パフォーマンスや過検出を考慮して実行領域を中心にメモリをスキャンするツールでは、シェルコードのあるメモリセグメントがスキャン対象外となって検出が迂回される可能性があります。一方、2nd Stage のシェルコードがダウンロードされた場合は、VirtualProtect() 関数でメモリの実行領域に置かれた2nd Stage シェルコードが実行されるため、実行領域を中心にスキャンを行うツールでも検出が行えます。

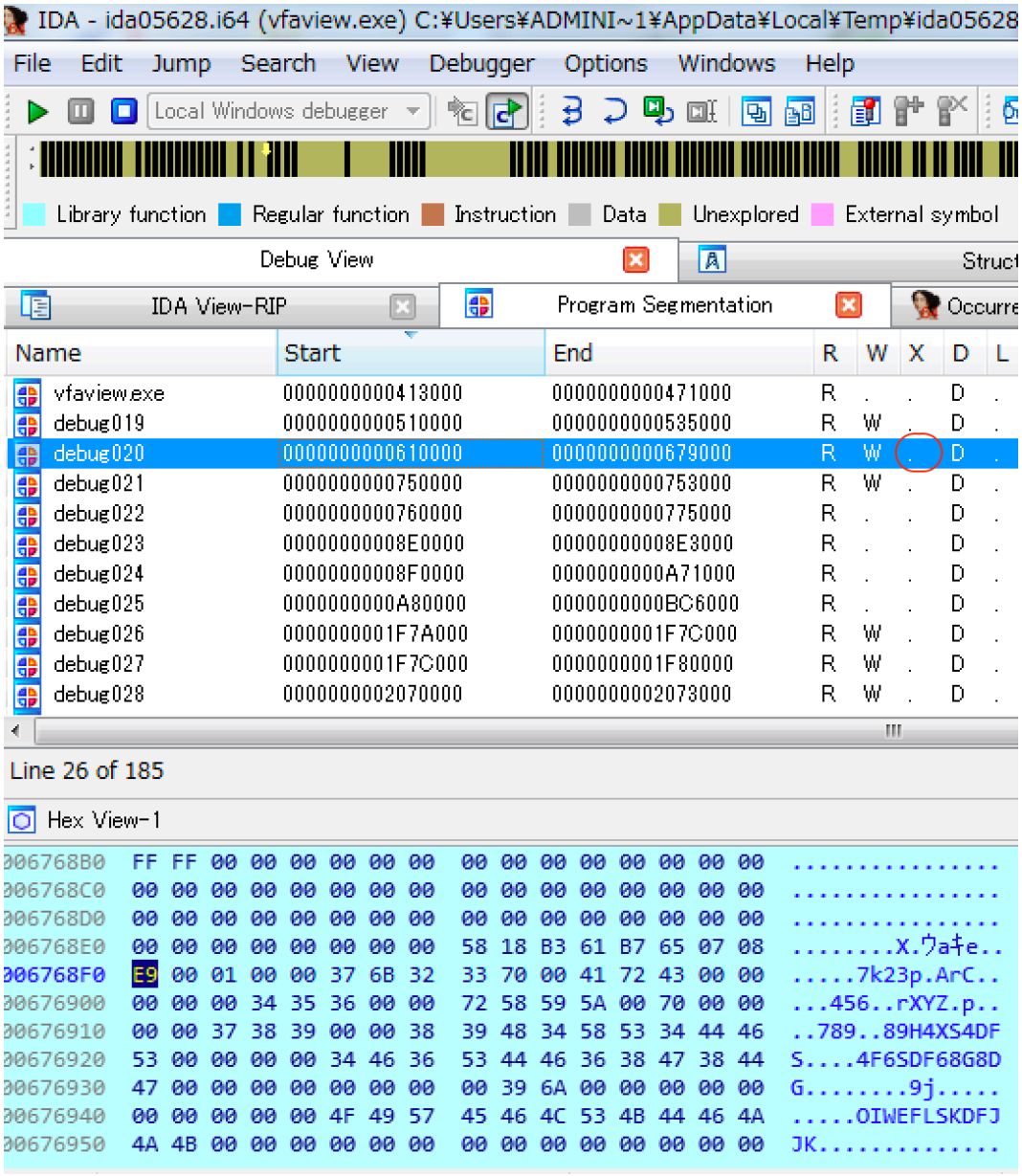


図19. 非実行領域のメモリセグメントに展開されて実行されるPUBLOADシェルコード

Mustang Panda 攻撃キャンペーンの特徴と検出

Mustang Panda の日本を含むアジア諸国の外交、政府関連を狙ったスピアフィッシング攻撃キャンペーンでは多くの国に同じ攻撃が展開されていると思われ、オープンソースのインテリジェンスをベースとした攻撃の特徴に気を付けメールの添付ファイルをむやみに実行しないような注意が必要です。攻撃が実行されてしまった場合、メモリ上のシェルコードの検出がツールによっては難しいケースがある事にご注意ください。ネットワークの IOC や EDR ツールで遠隔操作コマンドの実行を振舞で検出する事は容易かと思われます。

PUBLOADのシェルコードを検出するYARA ルール

```
rule MNC_APT_2024_PUBLOAD_Shell
{
   strings:
      $hx1 = { 89 4D F4 8B 55 F8 8A 84 15 EC FE FF FF 88 45 FF 8B 4D F8 8B 55 F4 8A 84 15 EC FE FF FF 88 84 0D EC FE FF FF 8B 4D F4 8A 55 FF 88 94 0D EC FE FF FF 8B 45 F8 0F B6 8C 05 EC FE FF FF 8B 55 F4 0F B6 84 15 EC FE FF FF 03 C8 89 4D EC 8B 4D 08 03 4D F0 0F B6 09 8B 45 EC 33 D2 BE 00 01 00 00 F7 F6 0F B6 94 15 EC FE FF FF 33 CA 88 4D FE 8B 45 08 03 45 F0 8A 4D FE 88 08 }
      $hx2 = { 55 8B EC 83 EC 0C 83 7D 0C 00 7F 07 C7 45 0C 01 00 00 00 C7 45 F8 00 00 00 00 EB 09 8B 45 F8 83 C0 01 89 45 F8 81 7D F8 00 01 00 00 7D 0D 8B 4D 10 03 4D F8 8A 55 F8 88 11 EB E1 C7 45 F4 00 00 00 00 C7 45 F8 00 00 00 00 EB 09 8B 45 F8 83 C0 01 89 45 F8 81 7D F8 00 01 00 00 7D 57 8B 45 F8 99 F7 7D 0C 8B 4D 08 0F B6 14 11 8B 45 10 03 45 F8 0F B6 08 03 55 F4 03 CA 81 E1 FF 00 00 80 79 08 49 81 C9 00 FF FF FF 41 89 4D F4 8B 55 10 03 55 F8 8A 02 88 45 FF 8B 4D 10 03 4D F8 8B 55 10 03 55 F4 8A 02 88 01 8B 4D 10 03 4D F4 8A 55 FF 88 11 EB 97 83 C8 FF 8B E5 5D }
   condition:
      all of them
}
```

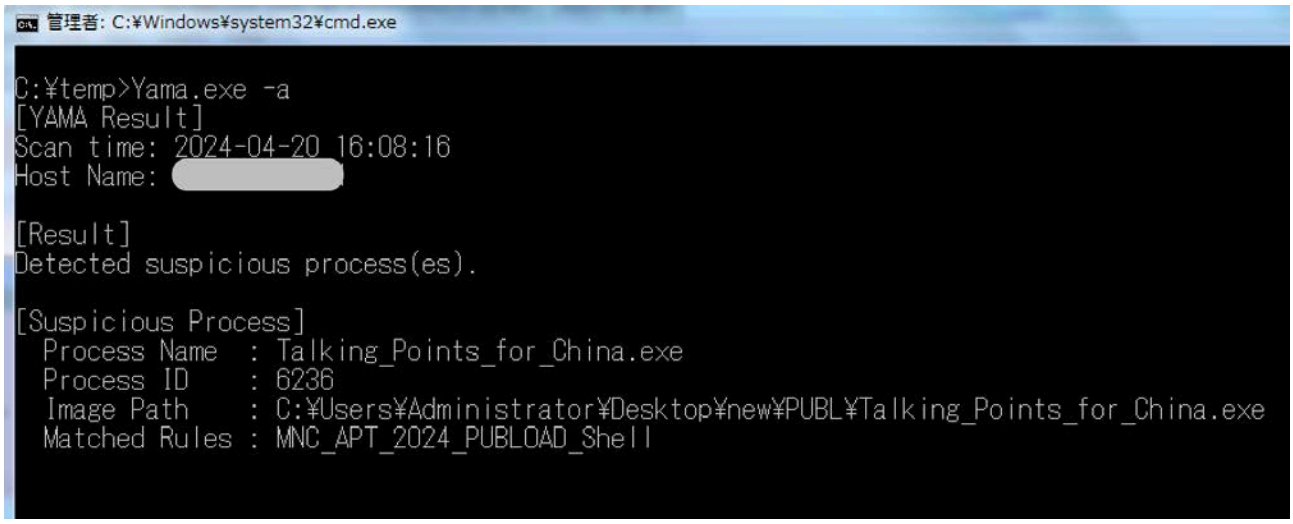


図20. JPCERT YAMA[21] を使った感染プロセスの検出

21 https://blogs.jpcert.or.jp/ja/2023/08/yama.html

Mustang Panda (東南アジア圏で観測された PlugDisk による攻撃)

攻撃キャンペーンの概要

2023 年 6 月、国内製造業のベトナム拠点の PC が中国を拠点とした攻撃グループが使う遠隔操作マルウェアの 1 つである PlugX に USB リムーバブルメディアから感染した事を観測しました。感染機器からは PlugX の暗号化ファイルを入手することができませんでしたが、パブリックマルウェアリポジトリから同じ名称のファイルを 2 つ入手し分析しました。その結果、ローダの構造と通信先が Mustang Panda のものと一致しており、攻撃の主体者が Mustang Panda である可能性が高いと考えています。TeamT5 は、この USB 感染機能を持つ PlugX を”PlugDisk”と呼称しています[22]。

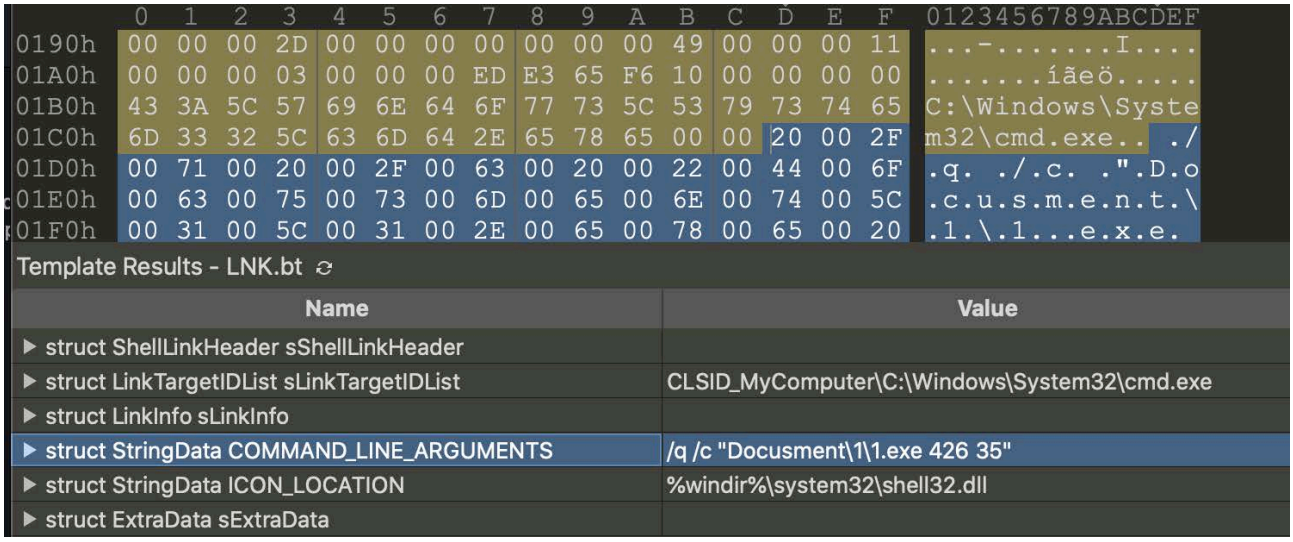
感染フロー

USB 内にあるショートカットファイルを実行すると、USB の別フォルダに保存されている PlugDisk 関連のファイルが実行されます。

図21. USBに保存されているショートカットファイル

ショートカットファイルを実行すると、USB 内にある Docusment\1\1.exe が実行されます。1.exe は、Adobe 社の正規ファイル AdobePhots.exe をリネームしたもので、同じ場所にある Adobe_Caps.dll がロードされて PlugDisk の暗号化ファイル AdobeDb.dat が読み込まれてメモリ上で PlugDisk が復号・実行されます。

22 https://jsac.jpcert.or.jp/archive/2023/pdf/JSAC2023_2_LT4.pdf



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0190h	00	00	00	2D	00	00	00	00	00	00	00	49	00	00	00	11	...-.....I....
01A0h	00	00	00	03	00	00	00	ED	E3	65	F6	10	00	00	00	00	.....íãeö.....
01B0h	43	3A	5C	57	69	6E	64	6F	77	73	5C	53	79	73	74	65	C:\Windows\Syste
01C0h	6D	33	32	5C	63	6D	64	2E	65	78	65	00	00	20	00	2F	m32\cmd.exe.. ./
01D0h	00	71	00	20	00	2F	00	63	00	20	00	22	00	44	00	6F	.q. ./.c. .".D.o
01E0h	00	63	00	75	00	73	00	6D	00	65	00	6E	00	74	00	5C	.c.u.s.m.e.n.t.\
01F0h	00	31	00	5C	00	31	00	2E	00	65	00	78	00	65	00	20	.1.\.1...e.x.e.

Template Results - LNK.bt

Name	Value
▶ struct ShellLinkHeader sShellLinkHeader	
▶ struct LinkTargetIDList sLinkTargetIDList	CLSID_MyComputer\C:\Windows\System32\cmd.exe
▶ struct LinkInfo sLinkInfo	
▶ struct StringData COMMAND_LINE_ARGUMENTS	/q /c "Documsment\1\1.exe 426 35"
▶ struct StringData ICON_LOCATION	%windir%\system32\shell32.dll
▶ struct ExtraData sExtraData	

図22. ショートカットファイルに設定されているコマンド

また、USB 内のフォルダにはシステム属性と隠し属性が設定されており、Windows OS の初期設定では表示されないようになっています。それらのフォルダとファイルは、FTK Imager などのフォレンジックツールを使い確認することができます。

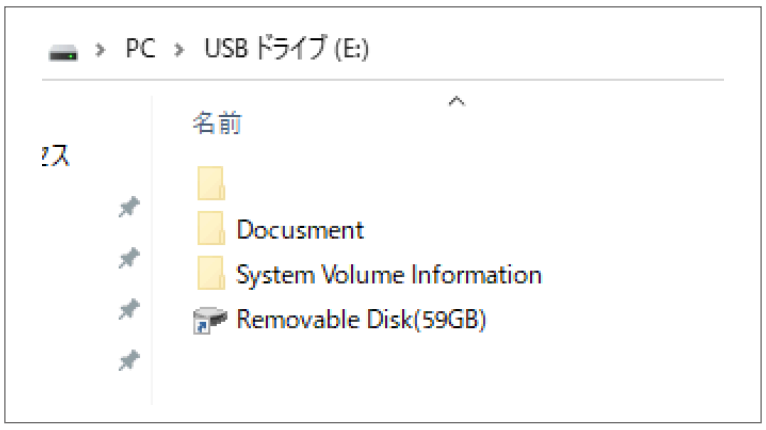


図23. システム属性、隠し属性のファイル表示を有効にした状態

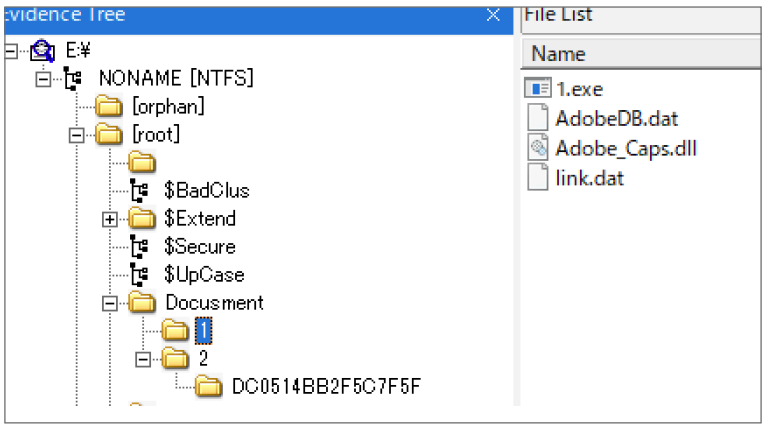


図24. FTK Imagerで表示したUSB内のファイル

USB から実行された PlugDisk は設定に従い、感染端末上の特定の場所に PlugDisk 関連のファイルをコピーし、パーシステンスのための Run レジストリキーを追加します。

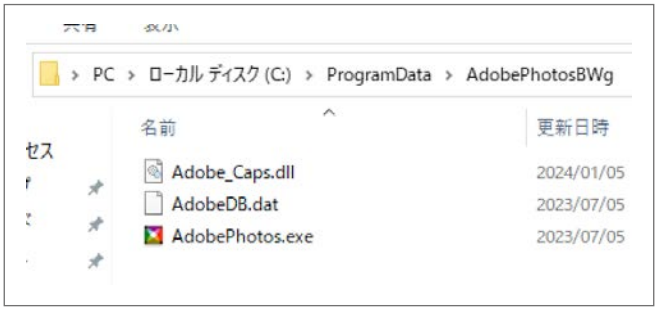


図25. 感染機器に設置されたPlugDisk関連のファイル

その後、コピーしたファイルを実行した後に自身は終了します。PlugDisk の処理は起動したファイルの方で引き継ぐ形となり、PlugDisk 本体のコードは Microsoft 社の正規実行ファイル tasklist.exe にインジェクションされて動作します。インジェクションされた tasklist.exe には、通常存在しない 4 つの起動パラメータ (今回の例では、646, 345, 173, 242) が付与されています。

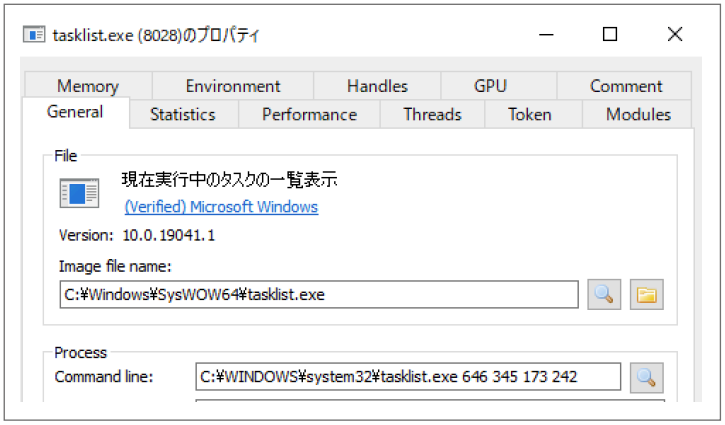


図26. PlugDisk本体のコードがインジェクションされたtasklist.exe

PlugDisk 詳細解析

感染機器からは、暗号化ファイルAdobedb.datを入手できていませんが、パブリックマルウェアリポジトリから入手した
Adobedb.dat(SHA256:b3caefb141bc47c702e71f773ed246bb9f905a222840365f2d6e432218605fd5) が起動時に付与するパラメータの形式や、インジェクション、USBに生成するファイル名などの振る舞いが同じであることから、同じタイプのPlugDiskが使われたと考えています。本レポートではこの暗号化ファイルを使った分析結果について記載します。
PlugDiskローダのAdobe_Caps.dllには、暗号化ファイルの名称が固定で埋め込まれています。

```
}
v96 = v127;
strcpy(v127, "\\Ado");
v97 = &v127[5];
strcpy(&v127[5], "bedb.dat");
v12 = ((v83 >> 6) + 623) & (((v83 >> 6) + 623) ^ 0xFFFFFBBF);
lstrcat = sub_74EB7EF0();
lstrcat(Str, v127);
lstrcat_1 = sub_74EB7EF0();
lstrcat_1(Str, &v127[5]);
v98 = &v89;
```

図27. PlugXローダが読み込む暗号化ファイル名

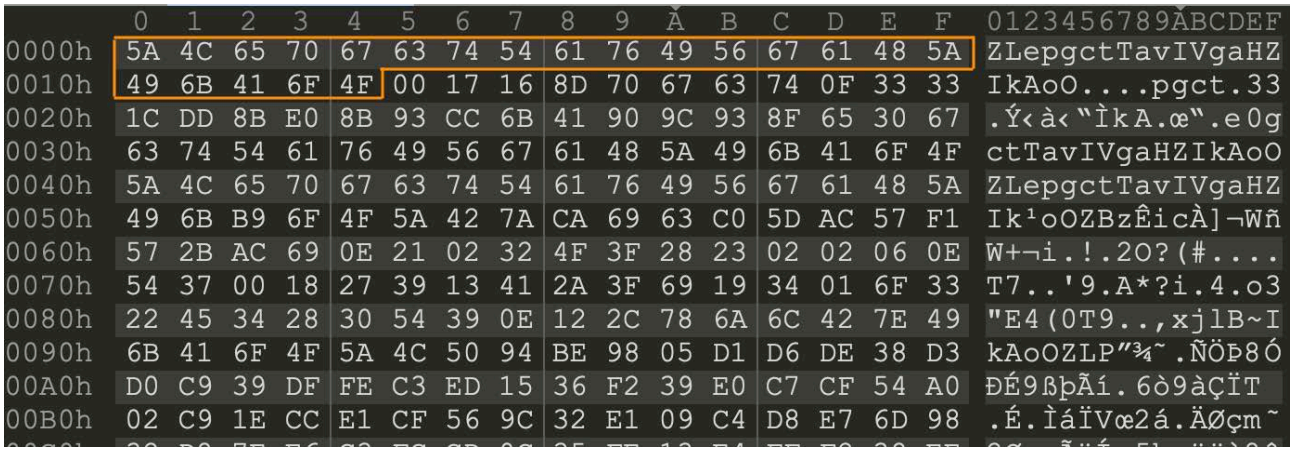
また、ローダとPlugDisk本体は解析を阻害するためにControl Flow Flattening (CFF)[23]によって処理フローが難読化されています。

23 https://news.sophos.com/ja-jp/2022/05/04/attacking-emotets-control-flow-flattening-jp/

```
sub_228D6A0(1);
state = 0x114514;
memset(v29, 0, sizeof(v29));
v27 = *a1;
v2 = 0xC18BD33A;
do
{
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        v17 = v2;
        v4 = (0x125E591 * ((0x125E591 * (state ^ v2)) ^ BYTE1(v2))) ^ ((v2 >> 8) >> 8);
        state = 0x125E591 * ((0x125E591 * v4) ^ ((v2 >> 8) >> 16));
        if ( state <= 0xF082A23D )
          break;
        if ( state <= 0x345667DC )
        {
          if ( state <= 0x2D813E1 )
          {
            if ( state == 0xF082A23E )
            {
              plugx_USB_2(0xBCCA, v24);
              v2 = v17 ^ 0xAD8B2802;
              goto LABEL_5;
            }
            if ( state == 0xF3A0B0E7 )
            {
              sub_22B9340(v24);
              v2 = v17 ^ 0x73F8FF91;
              goto LABEL_5;
            }
          }
          else
          {
            switch ( state )
            {
              case 0x2D813E2:
                sub_22CC750(0xBCCA, v23);
                plugx_CloseHandle(v23);
                plugx_HeapFree(v28);
                plugx_HeapFree(v31);
                v2 = v17 ^ 0x645D1710;
                goto LABEL_5;
              case 0x100F33E2:
                plugx_allocate_2bytes_buf(v31);
                plugx_allocate_2bytes_buf(v28);
                sub_2289F80(v22);
                plugx_w_lstrlenW(L"Data\\");
                sub_2289F80(v22);
                plugx_w_lstrlenW(L"Reformat USB.exe");
                plugx_memset_(v38, 0, 0x208u);
```

図28. Control Flow Flatteningによって処理フローが難読化されたコード

PlugDiskの暗号化ファイルはマルチバイトの鍵でXORされており、その鍵はファイル先頭から0x00（NULLバイト）までのバイト列です。



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0000h	5A	4C	65	70	67	63	74	54	61	76	49	56	67	61	48	5A	ZLepgctTavIVgaHZ
0010h	49	6B	41	6F	4F	00	17	16	8D	70	67	63	74	0F	33	33	IkAoO....pgct.33
0020h	1C	DD	8B	E0	8B	93	CC	6B	41	90	9C	93	8F	65	30	67	.Ý‹à‹"ÌkA.œ".e0g
0030h	63	74	54	61	76	49	56	67	61	48	5A	49	6B	41	6F	4F	ctTavIVgaHZIkAoO
0040h	5A	4C	65	70	67	63	74	54	61	76	49	56	67	61	48	5A	ZLepgctTavIVgaHZ
0050h	49	6B	B9	6F	4F	5A	42	7A	CA	69	63	C0	5D	AC	57	F1	Ik¹oOZBzÊicÀ]¬Wñ
0060h	57	2B	AC	69	0E	21	02	32	4F	3F	28	23	02	02	06	0E	W+¬i.!.2O?(#....
0070h	54	37	00	18	27	39	13	41	2A	3F	69	19	34	01	6F	33	T7..'9.A*?i.4.o3
0080h	22	45	34	28	30	54	39	0E	12	2C	78	6A	6C	42	7E	49	"E4(0T9..,xjlB~I
0090h	6B	41	6F	4F	5A	4C	50	94	BE	98	05	D1	D6	DE	38	D3	kAoOZLP"¾˜.ÑÖÞ8Ó
00A0h	D0	C9	39	DF	FE	C3	ED	15	36	F2	39	E0	C7	CF	54	A0	ÐÉ9ßþÃí.6ò9àÇÏT
00B0h	02	C9	1E	CC	E1	CF	56	9C	32	E1	09	C4	D8	E7	6D	98	.É.ÌáÏVœ2á.ÄØçm˜

図29. PlugX暗号化ファイルの鍵

PlugX の設定情報は、PlugX の .data セクションの先頭から 0xDDB バイトの領域で、暗号化ファイルを復号する鍵とは異なる鍵”123456789”を使い XOR でエンコードされています。

```
Extracted Configuration Items
AdobePhotosBWg
AdobePhotos
BTTM_86
%public%\Publics
%windir%\system32\tasklist.exe
1
443
101.36.125.203
1
110
101.36.125.203
6
80
101.36.125.203
1
965
101.36.125.203
```

図30. PlugXの設定情報

暗号化ファイルから PlugX と設定情報を抽出する Python スクリプトを以下に記載します。

```
import pefile
import argparse
import textwrap

def loaderDecode(filename):

    with open(filename, "rb") as dat:
        data = dat.read()

    key = []

    for d in data:
        if d != 0x00:
            key.append(d)
        else:
            break
    klen = len(key)

    output = []
    loop_condition = 0
    for c in data[klen + 1:]:
        current_key = key[loop_condition % klen]
        output.append(c ^ current_key)
        loop_condition += 1

    with open("PlugX_Loader_Decoded.exe", "wb") as decoded:
        decoded.write(bytearray(output))

def getData(filename):
    raw = 0
    pe = pefile.PE(filename)

    try:
        for section in pe.sections:
            if '.data' in str(section.Name):
                data = section.PointerToRawData
    except OSError as e:
        print(e)
    except pefile.PEFormatError as e:
        print("[-] PEFormatError: %s" % e.value)
    return data


def getUTF16LE_ConfigItem(config_item, ConfigItems):
    s = ""

    for i in range(0, len(config_item), 2):
        if config_item[i] == 0x00 and config_item[i+1] == 0x00:
            if len(s) >= 1:
                ConfigItems.append(s)
                break
        else:
            s += chr(config_item[i])

def getASCII_ConfigItem(config_item, ConfigItems):
    s = ""

    for b in config_item:
        if b != 0x00:
            s += chr(b)
        else:
            if len(s) >= 1:
                ConfigItems.append(s)
            break
def configDecode(config):

    decoded=[]
    if config[0:7] != "########":
        key = [0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39]
        klen = len(key)

        loop_condition = 0
        for c in config:
```

```
            current_key = key[loop_condition % klen]
            decoded.append(c ^ current_key)
            loop_condition += 1

    else:
        decoded=config
    s = ""
    ConfigItems=[]

    decoded = decoded[0x20:]
    getUTF16LE_ConfigItem(decoded[0x00:0x80], ConfigItems)
    getUTF16LE_ConfigItem(decoded[0x80:0x80+0x80], ConfigItems)
    getUTF16LE_ConfigItem(decoded[0x100:0x100+0x288], ConfigItems)
    getUTF16LE_ConfigItem(decoded[0x388:0x388+0x208], ConfigItems)
    getUTF16LE_ConfigItem(decoded[0x590:0x590+0x208], ConfigItems)

    # C2 1
    c2_base_addr = 0x798
    flag = decoded[c2_base_addr+1] << 8 | decoded[c2_base_addr]
    ConfigItems.append(str(flag))
    port = decoded[c2_base_addr+3] << 8 | decoded[c2_base_addr+2]
    ConfigItems.append(str(port))
    getASCII_ConfigItem(decoded[c2_base_addr+4:c2_base_addr+4+0xC0], ConfigItems)
    # C2 2
    c2_base_addr = 0x85C
    flag = decoded[c2_base_addr+1] <<  8 | decoded[c2_base_addr]
    ConfigItems.append(str(flag))
    port = decoded[c2_base_addr+3] << 8 | decoded[c2_base_addr+2]
    ConfigItems.append(str(port))
    getASCII_ConfigItem(decoded[c2_base_addr+4:c2_base_addr+4+0xC0], ConfigItems)
    # C2 3
    c2_base_addr = 0x920
    flag = decoded[c2_base_addr+1] <<  8 | decoded[c2_base_addr]
    ConfigItems.append(str(flag))
    port = decoded[c2_base_addr+3] << 8 | decoded[c2_base_addr+2]
    ConfigItems.append(str(port))
    getASCII_ConfigItem(decoded[c2_base_addr+4:c2_base_addr+4+0xC0], ConfigItems)
    # C2 4
    c2_base_addr = 0x9E4
    flag = decoded[c2_base_addr+1] <<  8 | decoded[c2_base_addr]
    ConfigItems.append(str(flag))
    port = decoded[c2_base_addr+3] << 8 | decoded[c2_base_addr+2]
    ConfigItems.append(str(port))
    getASCII_ConfigItem(decoded[c2_base_addr+4:c2_base_addr+4+0xC0], ConfigItems)

    return ConfigItems

def main():
    parser = argparse.ArgumentParser(
        formatter_class=argparse.RawDescriptionHelpFormatter,
        description="",
    )
    parser.add_argument("-f", "--file", help="PlugX Loader File")

    args = parser.parse_args()

    if args.file:
        try:
            loaderDecode(args.file)
            data = getData("PlugX_Loader_Decoded.exe")
            with open("PlugX_Loader_Decoded.exe", 'rb') as f:
                f.read(data)
                todecode = f.read(0xDD8)
            config=configDecode(todecode)
            print("Extracted Configuration Items")
            for c in config:
                print(c)
        except:
            print("Something didn't work right, ensure this is a PlugX Loader file and "
                  "that the correct arguments were passed")

if __name__ == "__main__":
    main()
```

PlugDiskはDLLファイルの形態で、そのMZヘッダー先頭部がシェルコードになっています。そのシェルコードを系由してエクスポート関数「StartProtect」が呼ばれメイン処理に移ります。

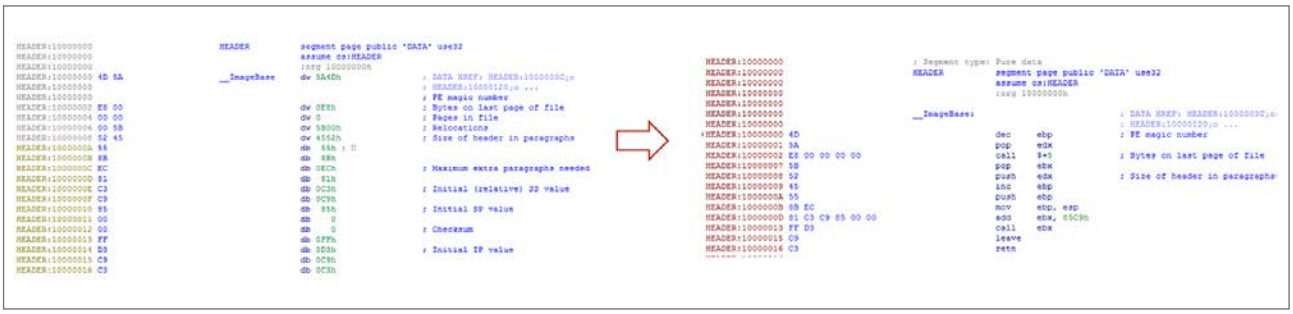


図31. PlugX MZヘッダーに埋め込まれているシェルコード

PlugDiskは感染機器上にUSBデバイスが接続されていないかをモニターし、接続を確認するとUSBにPlugDiskの関連ファイルをコピーし、USBにショートカットファイルを作成します。また接続したUSBファイルに元々保存されていたファイルは、USB内部の別フォルダに移動させて見えなくし、USBの保有者にショートカットファイルを実行させるよう誘導しています。ショートカットファイルを実行すると、PlugDiskの実行と合わせて元々保存されていたファイルの移動先を表示させて疑いを持たせづらくしています。

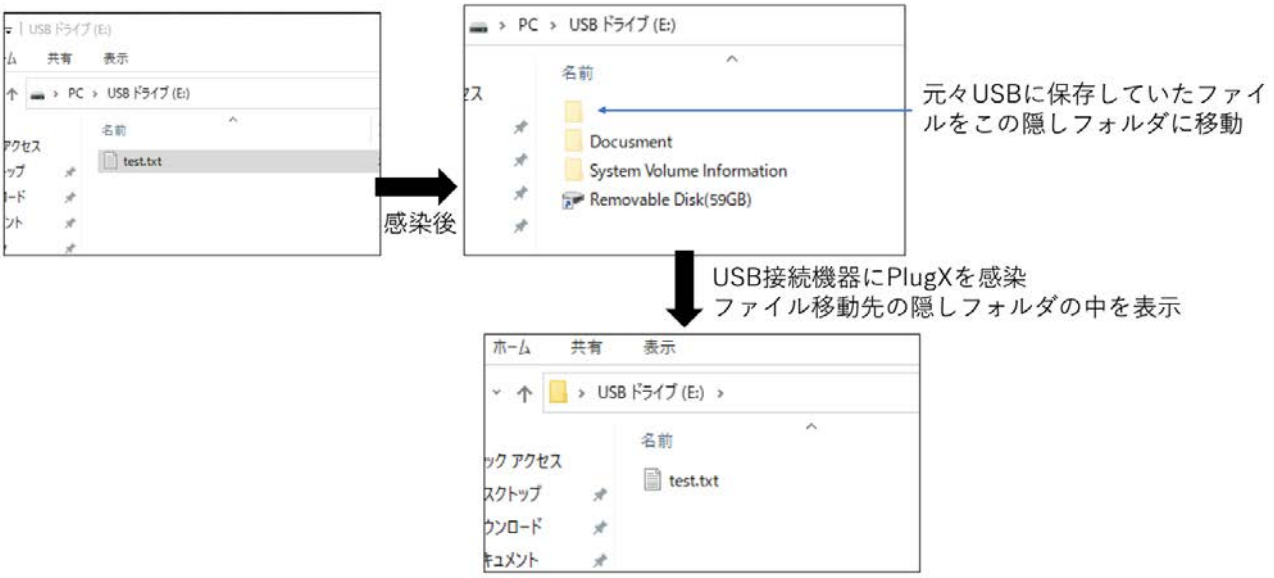


図32. PlugDisk USB 感染処理

PlugDisk USBを介した感染の考察

弊社が分析したケースでは、PlugDiskに感染したUSBが最初にどのようにして持ち込まれたかを特定することはできませんでした。セキュリティベンダー Mandiant社の調査では、地域のプリントショップやホテルのUSBを接続できる機器が感染源である可能性があることとUSB経由でPlugDiskに感染させるキャンペーンは、国の支援を受けた攻撃グループが関心のあるエリアで長期間の対象を選ばない諜報もしくは活動後期の追加情報を得ることが目的ではないかという見解を公開しています[24]。
本ケースも2021年には存在が観測されている比較的古いローダが使われていることと拡大手法から無差別の情報収集を目的としたキャンペーンでたまたま着弾したのではないかとみています。
弊社はUSBによる感染を2022年から国内企業の東南アジア拠点で複数観測しています[25]。特に製造業では業務でUSBを使うケースが多くあるため、引き続きUSB系由での攻撃は続くとみています。また2023年12月に脅威リサーチャーによってローダがNim言語で開発されたUSB系由で感染するPlugDiskも確認されています[26]。

24 https://www.mandiant.com/resources/blog/infected-usb-steal-secrets
25 https://www.virusbulletin.com/conference/vb2023/abstracts/usb-flows-great-river-classic-tradecraft-still-alive/
26 https://x.com/MalGamy12/status/1735641623206277356

Ivanti 製品の脆弱性を悪用した攻撃キャンペーン

攻撃キャンペーンの概要

2024 年 1 月から 2 月にかけて公開された Ivanti Connect Secure、Ivanti Policy Secure ゲートウェイの複数の脆弱性は日本を含め世界中の組織に大きな影響を与えました。脆弱性の公開に続いてかなり早いタイミングで PoC が公開されたことから特定の組織を狙った攻撃だけでなく手当たり次第に多くの組織を狙った攻撃でも悪用され、弊社でも改竄・設置された多くのファイルを分析しました。本章では我々の分析からみえた傾向と本レポート執筆時点で情報が公開されていないバックドアについて解説します。

国内で観測された悪用事例

認証バイパス（CVE-2023-46805）とコマンドインジェクション（CVE-2024-21887）を悪用した攻撃の多くで WIREFIRE と命名された Web Shell が見つかりました。一方で ZIPLINE と命名された accept 関数をハイジャックするバックドアは WIREFIRE が見つかった一部の機器で見つかりました。ファイルのタイムスタンプも考慮すると、WIREFIRE を使い情報を収集し更なる活動を行うことを決めた組織においてのみ ZIPLINE を設置していた可能性があると考えています。我々の観測範囲では数は少ないですが、Web ログオン中に入力した認証情報を別サーバに送信する WARPWIRE と LIGHTWIRE Web Shell の亜種を観測しています。また、サーバーサイド・リクエスト・フォージェリ（SSRF）（CVE-2024-21893）を悪用した攻撃では、DSLog backdoor[27] が設置されたケースを複数確認しました。悪用された脆弱性は不明ですが、オープンソース C2 フレームワークの Sliver が設置されたケースも確認しています。

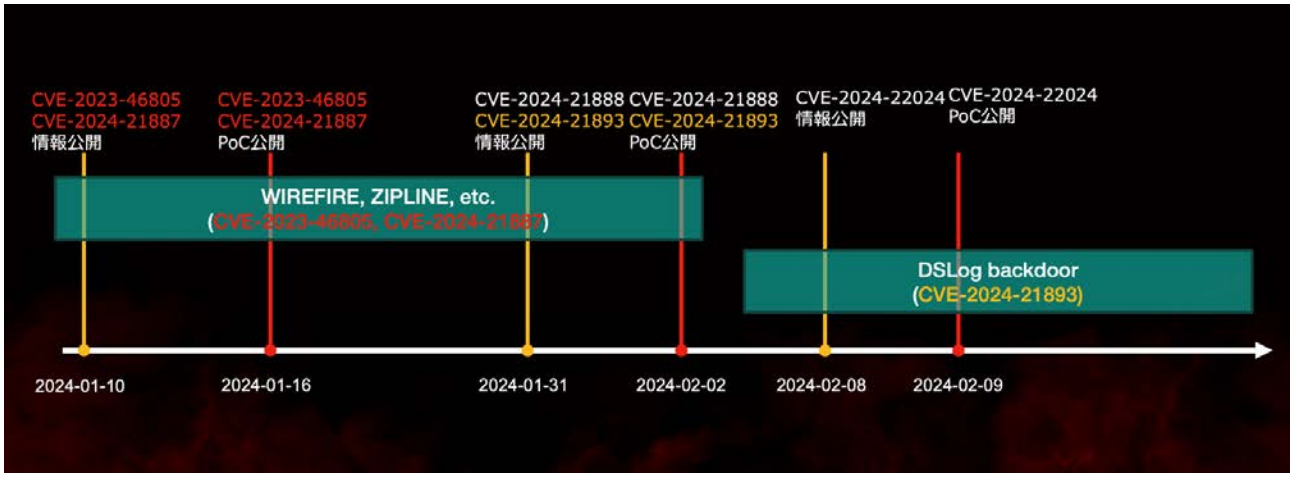


図33. 脆弱性・PoC公開と弊社で攻撃を確認した時期

27 https://www.orangecyberdefense.com/uk/blog/research/ivanti-connect-secure-journey-to-the-core-of-the-dslog-backdoor

パッシブバックドア ProxDoor

弊社の分析の中で ZIPLINE[28]、SPAWNMOLE[29] とは異なるパッシブバックドアを発見し、”ProxDoor”と命名しました。ProxDoor は ELF 形式の共有ライブラリファイルで、ファイル名”librs.so.1”として設置されており、別の改竄された libnspr4.so によりロードされます。libnspr4.so は、/home/bin/web プロセスが使用しており web プロセスがロードした accept, accept4 関数のアドレスが ProxDoor の関数のアドレスに書き換えられ、accept 処理が ProxDoor にハイジャックされます。

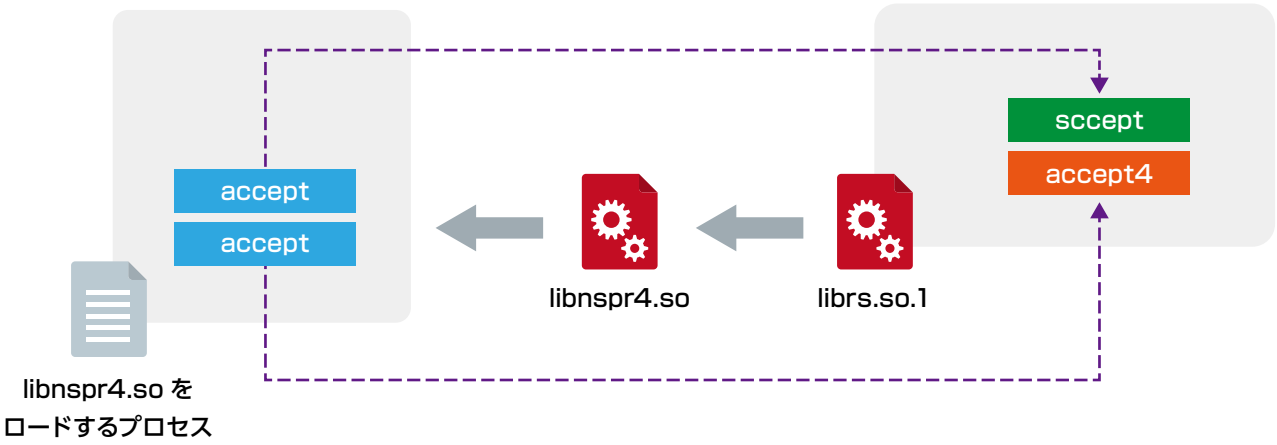


図34. ProxDoorによりaccept 処理がハイジャックされる処理フロー

libspr4.soの_init_proc関数の中にはlibrs.so.1をロードする処理が追加されていました。

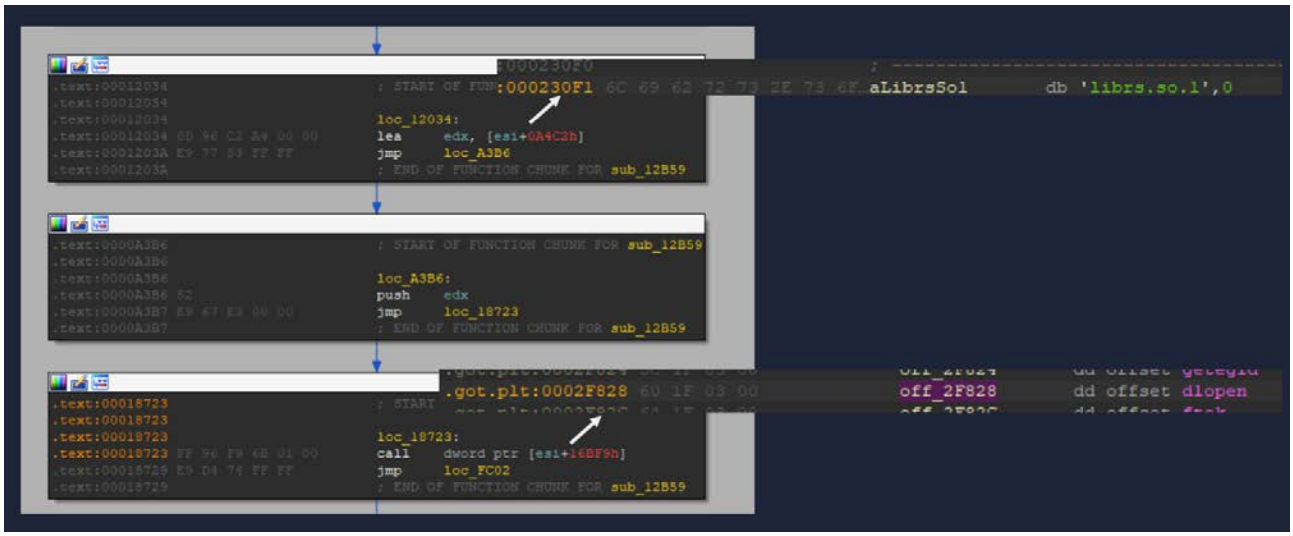


図35. libnspr4.soに追加されたlibrs.so.1(ProxDoor)をロードする処理

28 https://www.mandiant.jp/resources/blog/suspected-apt-targets-ivanti-zero-day

29 https://cloud.google.com/blog/ja/topics/threat-intelligence/ivanti-post-exploitation-lateral-movement/

ProxDoorは、受信したデータの先頭からオフセット0x0F以降から4バイト単位のデータでXORを行い、その値がオフセット0x0Bからの4バイトと一致するかをチェックします。一致した場合は命令コマンドと判断し、RC4で復号します。

```
unsigned __int32 __cdecl aa_decrypt_data(const void *buf)
{
  int pos; // eax
  int chk_val; // edx
  _BYTE recv[118]; // [esp+Ah] [ebp-18Eh] BYREF
  char sbox_[280]; // [esp+80h] [ebp-118h] BYREF
  pos = 0;
  chk_val = 0;
  memset(sbox_, 0, 256u);
  qmemcpy(recv, buf, sizeof(recv));
  do
    chk_val ^= *&recv[4 * pos++ + 0xF];
  while ( pos != 6 );
  if ( chk_val != *&recv[0xB] )
    return -1;
  make_sbox(sbox_, &recv[15], 24u);
  aa_rc4_(sbox_, &recv[39], 4);
  return _byteswap_ulong(*&recv[39]);
}
```

図36. 受信データが命令コマンドであるかの判定処理

ProxDoorでサポートしている命令コマンドは、2つのみでシンプルな作りですが、受信データにはコマンドID以外に受信データを別IP、ポート番号への転送を行うフラグも含まれています。コマンドIDは1バイトデータに対して上位4bit、下位4bitから抽出します。

```
cmd_id = *(v4 + 0x116);
if ( (cmd_id & 0xF0) == 0x10 )          // Load Library
{
  size = cmd_run_library(v4);
  if ( size < 0 )
  {
    sub_31E5(v4, &size, 4);
    sleep(2u);
  }
}
else if ( (cmd_id & 0xF) == 4 )         // Exec Command
{
  end = sub_2F0A(v4, buf, 1024);
```

図37. コマンドID抽出処理

表3. ProxDoorサポートコマンド

ID	コマンド
0x04	遠隔からのコマンドを実行
0x10	指定されたライブラリファイルをロード

攻撃グループごとのTTPs(戦術、技術、手順)

2023 年度に弊社で観測した攻撃グループごとの TTPs と標的組織を表で大まかに整理します。MITRE 社 ATT&CK に攻撃フレームワークの攻撃番号を記載しますので、利用している製品での検出有無などをご確認ください。

※この表は、MITRE 社 ATT&CK 攻撃フレームワーク version 15 [30] に基づき作成しています。

攻撃グループ	攻撃のTTPs	標的組織
Tropic Trooper	**侵入経路**:スピアフィッシングメール 添付ファイル (Office マクロ、アイコン偽装の実行ファイル)、Wi-Fi アクセスポイントの悪用(悪魔の双子攻撃) **エクスプロイト**:N/A **利用するツール・マルウェア**: EntryShell RAT / Cobalt Strike Beacon / CrowDoor **C2 通信の特徴**: 業務上あまり使われないポート番号宛の TCP 通信 (4431 8443 など) **ATT&CK**: [Initial Access] Phishing: Spear phishing via Service (T1566.003) [Execution] Command and Scripting Interpreter: Visual Basic (T1059.005) Process Injection: Asynchronous Procedure Call (T1055.004) [Privilege Escalation] Access Token Manipulation (T1134) [Persistence] Hijack Execution Flow: DLL Side-Loading (T1574.002) [Command and Control] Non-Standard Port (T1571) Encrypted Channel: Symmetric Cryptography (1573.001)	国内製造関連企業 中国拠点
Mustang Panda	**侵入経路**:スピアフィッシングメール 添付ファイル (zip ファイルに EXE ファイルと隠し DLL ファイル)、USB **エクスプロイト**:N/A **利用するツール・マルウェア**: PUBLOAD　Claimloader / PlugDisk **C2 通信の特徴**: RC4 暗号通信 **ATT&CK**: [Initial Access] Phishing: Spearphishing Attachment (T1566.001) [Initial Access] Replication Through Removable Media (T1091) [Persistence] Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder (T1547.001) [Defense Evasion] Hijack Execution Flow: DLL Side-Loading (T1574002)	

30 https://attack.mitre.org/versions/v15/

APT10 (LODEINFO)	**侵入経路**：スピアフィッシングメール 添付ファイル (Office マクロ) **エクスプロイト**：N/A **利用するツール・マルウェア**： LODEINFO **C2 通信の特徴**： VPS/ ホスティングサービスの日本国内にあるサーバを C2 として悪用、HTTP POST ATT&CK： [Initial Access] Phishing: Spearphishing Attachment (T1566.001) [Execution] User Execution: Malicious File (T1204.002) Office ファイルのマクロを有効にするよう誘導 [Persistence] Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder (T1547.001) 機器再起動後に自動実行されるようにレジストリ追加 [Defense Evasion] Signed Binary Proxy Execution: Rundll32 (T1218.011) 正規ファイルの rundll32 を使って悪意のある DLL ファイルのコードを実行 [Defense Evasion] Hijack Execution Flow: DLL Side-Loading (T1574002) 正規ファイル K7SysMon.exe がロードする K7SysMn1.dll をベースにローダを開発 [Command and Control] Application Layer Protocol: Web Protocols (T1071.001) HTTP プロトコル上で暗号化データの通信を行う	シンクタンク (研究機関)、製造

TTPsより考察する脅威の検出と緩和策

マルウェアの配送・攻撃について

2023 年度は様々なベンダーの外部公開デバイスの脆弱性攻撃からの侵入が観測されています。これらにはゼロデイ脆弱性攻撃も多くありますが、アプライアンス製品のため侵入を早期に検出するためのセキュリティ対策ソフトウェアなどを導入する事ができません。標的型攻撃を行う攻撃グループによって、ベンダーが提供する外部公開デバイス製品のゼロデイ脆弱性が発見・攻撃される事は今後も継続すると思われ、この前提での対策を検討すべきかと思われます。外部公開デバイスのゼロデイ脆弱性の対策の 1 つとして、これらのアプライアンス製品の前後にファイアウォールを設置して、攻撃を受けた後に発生する外向きのダウンロードなどの通信や内向きの RDP SMB SSH などの通信をログのモニタリングから検出する事がセキュリティリサーチャーにより共有されており[31]、有効と思われます。また、外部公開デバイスのゼロデイ脆弱性攻撃がベンダーから報告があった際、海外拠点なども含めて External Attack Surface Management でデバイスの有無やパッチバージョンを把握して早期の対処を検討頂くと良いかと思われます。その他に、物理的な侵入が増加している傾向があり、USB デバイスや Wi-Fi アクセスポイントの悪用からの物理的な侵入もある前提で、レガシーな対策ですが、デバイスコントロールといった対策や Wi-Fi 製品の機能で不審なアクセスポイントが設置されていないかといった事を確認するような対策もご検討ください。これら機器の出力するログもベンダーのベストプラクティスに従い、想定される攻撃時のログを出力するように設定した上で、別の SIEM 装置やログ基盤に転送しておく事が望ましいと思われます。攻撃者はこれら装置上のログを削除する事が多くあり、またログ基盤に転送して検出ルールと照合する事で不正アクセスの検出を行い、事後にはフォレンジック調査に役立てる事ができます。

インストールされる RAT、遠隔操作（C2 サーバについて）

LODEINFO、PUBLOAD、PlugX などいずれも DLL サイドローディングで、暗号化されたペイロードをローダ DLL が復号してメモリに展開して実行する手法が多く見られます。実行中のプロセスをスキャンしてメモリ上の RAT の特徴的なコードをファストフォレンジックツールで検出する事や、通常とは異なるパスに正規実行ファイルが保存されて実行されるといった振る舞いを EDR 製品で検出することは現在ではそれほど難しくないと思われます。一方、Stowaway のような GitHub 上で公開されたツールや Visual Studio Code の Remote 機能の悪用など、遠隔操作が開始された後の遠隔コマンドなどを EDR で検出しないと検出が難しいと思われるようなケースも散見されます。XDR と一括り対策の中でも基本的な EDR の対策を端末・サーバに網羅的に展開する事が重要です。また、正規の脆弱性のあるカーネルドライバを悪用して EDR 製品のモニタリングを無効にするような Bring Your Own Vulnerable Driver（BYOVD）[32] 攻撃もしばしば聞かれるようになっています。この攻撃の結果想定される挙動として、モニタリング対象の端末やサーバから EDR のログがあがっていないなどの兆候にご注意ください。また、EDR 製品によってはこの攻撃自体を検出して検出アラートを上げるものもあります。

31 https://twitter.com/58_158_177_102/status/1756154654256492884/
32 https://blogs.vmware.com/security/2023/10/hunting-vulnerable-kernel-drivers.html

検知のインディケータ

Tropic Trooper

インディケータ	タイプ	備考
98af7888655b8bcac49b76c074fc08877807ac074fb4e81a6cacfd1566d52f12	SHA256	CrowDoor ローダ
9dff4c8f403338875d009508c64a0e4d4a5eeac191d7654a7793c823fb8e3018	SHA256	CrowDoor ローダ
8937e8dd520dc6555c5b2cd62897b8eb5352e43a12af488bd8594449ed114fd5	SHA256	CrowDoor 暗号ファイル
blog[.]techmersion[.]com	C2	CrowDoor C2 サーバ

APT10 (LODEINFO)

インディケータ	タイプ	備考
7a4fd1cc932b96175055b2940242877cab728a9d7c7ee371cad8438b4e88a812	SHA256	v0.6.9 LODEINFO ローダ
632975a3642b0f2a6084880e59ffa19dfa8b08d13ac15b639e1e0ad3bdbf45bd	SHA256	v0.6.9 LODEINFO 暗号ファイル
http[:]//185.126.236[.]166	C2	v0.6.9 LODEINFO C2 サーバ
http[:]//198.13.33[.]117	C2	v0.6.9 LODEINFO C2 サーバ
f21745cc6306461d1ddb3c35ed6016468ce984bbd64bfb86139a392e3a45c495	SHA256	v0.7.1 LODEINFO ドロッパ
ecc746323a432bf483b6b1fff342ebe1834f8271d4cdf81b4f81d0a13e89436e	SHA256	v0.7.1 LODEINFO ローダ
29bfd90bc23928e9180bd30223dd7f447af1f0f0121386c239f0c0e0c0bc0482	SHA256	v0.7.1 LODEINFO 暗号ファイル
http[:]//167.179.77[.]72	C2	v0.7.1 LODEINFO C2 サーバ
http[:]//167.179.106[.]224	C2	v0.7.1 LODEINFO C2 サーバ

Mustang Panda

インディケータ	タイプ	備考
316541143187acff1404b98659c6d9c8566107bd652310705214777f03ea10c8	SHA256	PUBLOAD
c5630e93098b76b87852e97f1dce4e7060c7b623f8d453a71c3cec0030e94ebc	SHA256	PUBLOAD
3aa933ed37229a77ac190d853656bac9065b770a9c38750ae3361ba371e28ced	SHA256	PUBLOAD
103.27.109[.]157	C2	C2 サーバ
6d569df32c080437ad4b144620c03883e87a7d2d3db89f752abbca7b709d5199	SHA256	File name: 1.exe 正規ファイルがリネームされたファイル
39945063c73af8263b58f7ab899afb575486c1a49af0ca465e54f84c6b2d1df4	SHA256	File name: Adobe_Caps.dll PlugX ローダ
b3caefb141bc47c702e71f773ed246bb9f905a222840365f2d6e432218605fd5	SHA256	File name:AdobeDb.dat PlugX 暗号化ファイル
101.36.125[.]203	C2	PlugX
8fcdf9dbb788f519ce90ba30228be129432a1837b252ca782a5f7b9269729c00	SHA256	File name:AdobeDb.dat PlugX 暗号化ファイル
45.43.63[.]219	C2	PlugX
26b1d37ea3da6a6213b65b000dbb39575d858fa274aea895cc3bf62e706fce5d	SHA256	Nim で開発された PlugDisk

Others

インディケータ	タイプ	備考
ca9d992e1ca743925d29b7b212421c42e8d513293046b635170626755b8528a6	SHA256	Stowaway
dc75be193b73a0cb21daa886d0ff42c8fd4cf5e6cb77fff72dd05f8f0fa2b602	SHA256	Ivanti ELF バックドア
e995903a67a47f8f347669d8933ead4b15809a1ba76b13e9d7bf23135cf1ca7d	SHA256	Sliver
103.13.28[.]40	C2	Sliver C2
cdfda4041c611213524829d68c3ed85b2fb2fcb50f942bbf3e4481b982f42dfb	SHA256	ProxDoor をロードする改竄された libnspr4.so
dc75be193b73a0cb21daa886d0ff42c8fd4cf5e6cb77fff72dd05f8f0fa2b602	SHA256	ProxDoor

タネまく、
MACNICA

マクニカは、1972 年の設立以来、最先端の半導体、電子デバイス、ネットワーク、サイバーセキュリティ商品に技術的付加価値を加えて提供してきました。従来からの強みであるグローバルにおける最先端テクノロジーのソーシング力と技術企画力をベースに、AI/IoT、自動運転、ロボットなどの分野で新たなビジネスを展開しています。その中でセキュリティにおいては、最先端のセキュリティ商材を提供する中で独自の研究機関を有し、日本の企業に着弾したサイバー攻撃や対策をリサーチしています。

お問い合わせ 045-476-2010 https://www.macnica.co.jp/business/security/

〒222-8563 横浜市港北区横浜1-5-5
TEL.045-476-2010 | FAX.045-476-2030

西日本オフィス 〒530-0005 大阪市北区中之島2-3-33 大阪三井物産ビル14階
TEL.06-6227-6916 | FAX.06-6227-6917

株式会社マクニカ