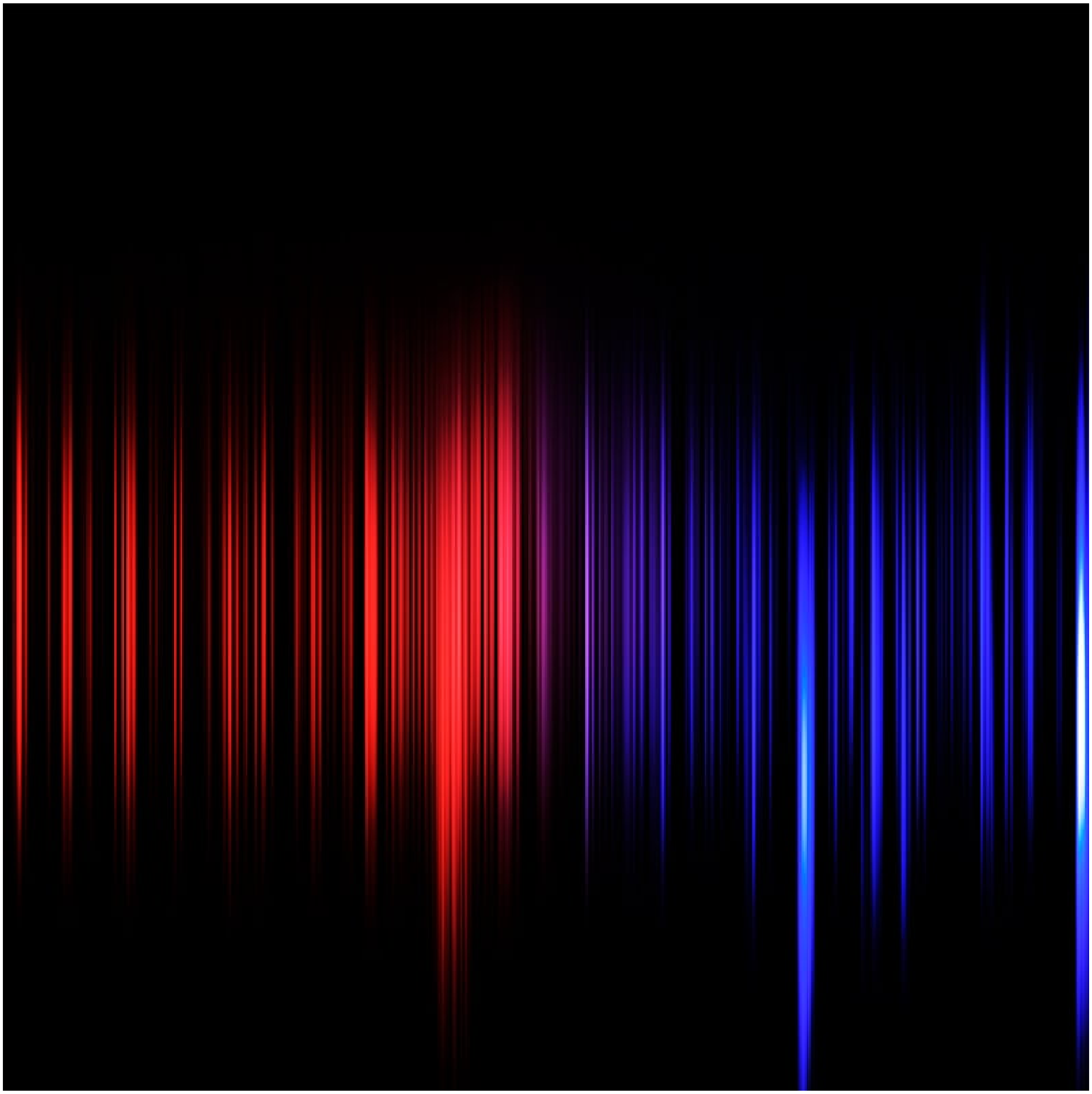
DropboxAES Remote Access Trojan

secureworks.com/research/dropboxaes-remote-access-trojan

Counter Threat Unit Research Team



Wednesday, June 24, 2020 *By: Counter Threat Unit Research Team*

The following analysis was compiled and published to Threat Intelligence clients in March 2019. The Secureworks® Counter Threat Unit™ (CTU) research team is publicly sharing insights about BRONZE VINEWOOD and its use of the HanaLoader malware and DropboxAES RAT, to increase visibility of the threat group's activities.

Summary

DropboxAES is a simple remote access trojan (RAT) used by the BRONZE VINEWOOD (also known as APT31) threat group. The RAT uses the Dropbox file-sharing service for its command and control (C2) communications. The sample analyzed by Secureworks® Counter Threat Unit™ (CTU) researchers is executed via DLL search-order hijacking. Once executed on a host, DropboxAES RAT enables a threat actor to remotely perform the following actions:

- Upload files from the infected host to the C2 server
- Download files from the C2 server to the infected host
- Execute commands on the infected host via a non-interactive command-line based reverse shell
- Upload basic system information about the compromised host to the C2 server
- Completely remove itself from the infected host

BRONZE VINEWOOD's tactics

BRONZE VINEWOOD campaigns have targeted legal, consulting, and software development organizations. CTU™ analysis suggests that organizations that are part of government or defense supply chains or that provide services to organizations in those verticals may be at higher risk of targeting than organizations in other verticals.

In addition to Dropbox, BRONZE VINEWOOD has used other popular social media and code repository sites to hide malicious activity among legitimate network traffic. CTU researchers have also identified previous BRONZE VINEWOOD campaigns utilizing DLL search-order hijacking to deliver the HanaLoader downloader tool and other malicious payloads.

DropboxAES RAT technical details

Despite BRONZE VINEWOOD naming the malware DropboxAES RAT, the version analyzed by CTU researchers does not use the Advanced Encryption Standard (AES). Rather, it implements a ChaCha20 stream cipher to encode and decode data. Older versions of the malware may have leveraged AES encryption when encrypting data.

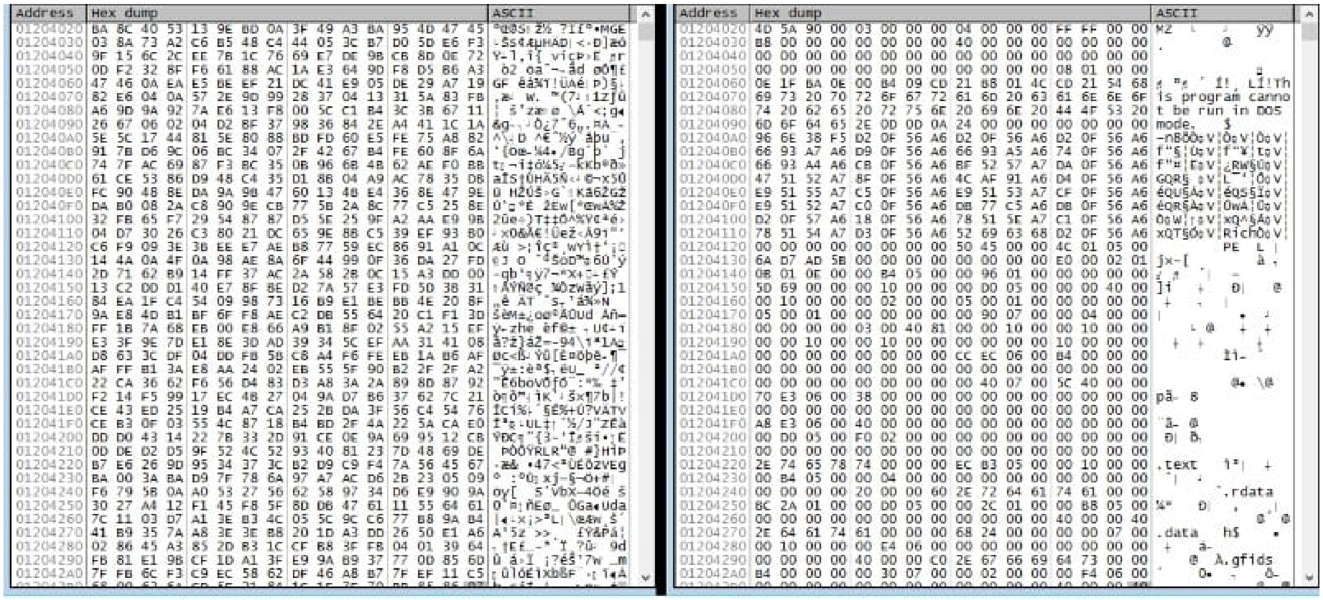
The following sections describe how DropboxAES RAT builds its malicious payload, prepares its working environment, establishes persistence, gathers information about the compromised system, and interacts with the Dropbox-based C2 server.

Decoding and executing the DropboxAES RAT executable

The analyzed sample starts the infection chain with two files:

- asOELnch.exe — a legitimate signed Symantec antispam executable
- MSVCR100.dll — a malicious DLL that acts as a sideloader for DropboxAES RAT

When executed, asOELnch.exe loads MSVCR100.dll, which results in execution of the malicious sideloader. This sideloader verifies it is running within the context of asOELnch.exe and then Base64-decodes a large data blob appended to the end of MSVCR100.dll. The Base64-decoded result is decoded again using an RC4 stream cipher, resulting in a fully decoded executable that is the DropboxAES RAT (see Figure 1). The sideloader then creates a suspended instance of the parent executable (asOELnch.exe), injects the decoded DropboxAES RAT executable into the suspended instance, and runs the injected executable.



Encoded DropboxAES RAT in buffer

```
Address   Hex dump
01204020  BA 8C 40 53 13 9E BD 0A 3F 49 A3 BA 95 4D 47 45
01204030  03 8A 73 A2 C6 B5 48 C4 44 05 3C B7 D0 5D E6 F3
01204040  9F 15 6C 2C EE 7B 1C 76 69 E7 DE 9B CB 8D 0E 72
01204050  0D F2 32 8F F6 61 88 AC 1A E3 64 9D F8 D5 86 A3
01204060  47 46 0A EA E5 BE EF 21 DC 41 E9 05 DE 29 A7 19
01204070  82 E6 04 0A 57 2E 9D 99 28 37 04 13 31 5A 83 FB
01204080  A6 9D 9A 92 7A E6 13 F8 00 5C C1 B4 3C 3B 67 11
01204090  26 67 06 02 04 D2 BF 37 98 36 84 2E A4 41 1C 1A
012040A0  5E 5C 17 44 81 5E 80 88 BD FD 60 E5 FE 75 A8 82
012040B0  91 7B D6 9C 06 BC 34 07 2F 42 67 B4 FE 60 8F 6A
012040C0  74 7F AC 69 87 F3 BC 35 0B 96 68 4B 62 AE F0 BB
012040D0  61 CE 53 86 D9 48 C4 35 D1 8B 04 A9 AC 78 35 D8
012040E0  FC 90 48 8E DA 9A 98 47 60 13 4B E4 36 8E 47 9E
012040F0  DA B0 08 2A C8 90 9E CB 77 5B 2A 8C 77 C5 25 8E
01204100  32 FB 65 F7 29 54 87 87 D5 5E 25 9F A2 AA E9 98
01204110  04 D7 30 26 C3 80 21 DC 65 9E 88 C5 39 EF 93 B0
01204120  C6 F9 09 3E 3B EE E7 AE B8 77 59 EC 86 91 A1 0C
01204130  14 4A 0A 4F 0A 98 AE 8A 6F 44 99 0F 36 DA 27 FD
01204140  2D 71 62 B9 14 FF 37 AC 2A 58 2B 0C 15 A3 DD 00
01204150  13 C2 DD D1 40 E7 8F BE D2 7A 57 E3 FD 5D 38 31
01204160  84 EA 1F C4 54 09 98 73 16 B9 E1 BE BB 4E 20 8F
01204170  9A E8 4D B1 BF 6F F8 AE C2 DB 55 64 20 C1 F1 3D
01204180  FF 18 7A 68 EB 00 E8 66 A9 B1 8F 02 55 A2 15 EF
01204190  E3 3F 9E 7D E1 8E 30 AD 39 34 5C EF AA 31 41 08
012041A0  D8 63 3C DF 04 DD FB 5B C8 A4 F6 FE EB 1A B6 AF
012041B0  AF FF B1 3A E8 AA 24 02 EB 55 5F 90 B2 2F 2F A2
012041C0  22 CA 36 62 F6 56 D4 83 D3 A8 3A 2A 89 8D 87 92
012041D0  F2 14 F5 99 17 EC 48 27 04 9A D7 B6 37 62 7C 21
012041E0  CE 43 ED 25 19 B4 A7 CA 25 2B DA 3F 56 C4 54 76
012041F0  CE B3 0F 03 55 4C 87 18 B4 BD 2F 4A 22 5A CA E0
01204200  DD D0 43 14 22 7B 33 2D 91 CE 0E 9A 69 95 12 CB
01204210  0D DE D2 D5 9F 52 4C 52 93 40 81 23 7D 48 69 DE
01204220  B7 E6 26 9D 95 34 37 3C B2 D9 C9 F4 7A 56 45 67
01204230  BA 00 3A BA D9 7F 78 6A 97 A7 AC D6 2B 23 05 09
01204240  F6 79 5B 0A A0 53 27 56 62 58 97 34 D6 E9 90 9A
01204250  30 27 A4 12 F1 45 F8 5F 8D D6 47 61 11 55 64 61
01204260  7C 11 03 D7 A1 3E B3 4C 05 5C 9C C6 77 B8 9A B4
01204270  41 B9 35 7A A8 3E 3E B8 20 1D A3 DD 26 50 E1 A6
01204280  02 86 45 A3 85 2D B3 1C CF B8 3F F8 04 01 39 64
01204290  FB 81 E1 9B CF 1D A1 3F E9 9A 89 37 77 0D 85 6D
012042A0  7F FB 6C F3 C9 EC 58 62 DF 46 A8 B7 7F EF 11 C5
```

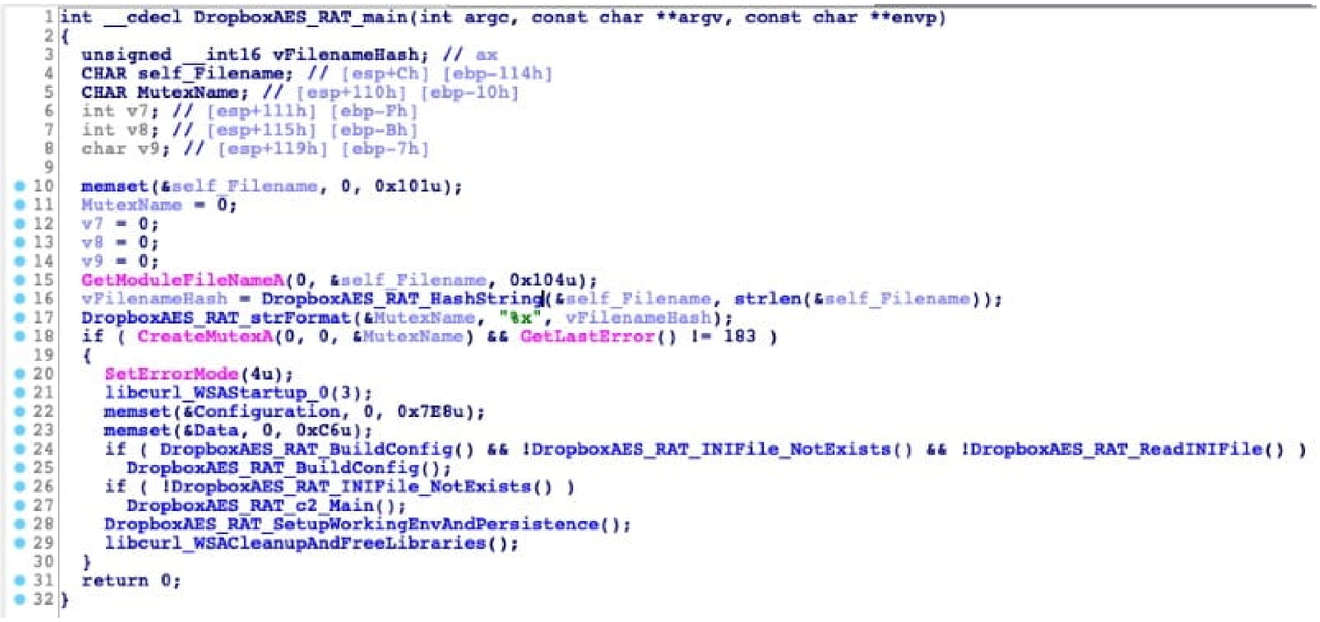
Decoded DropboxAES RAT in buffer

```
Address   Hex dump                                         ASCII
01204020  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ          ÿÿ
01204030  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ¸       @
01204040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01204050  00 00 00 00 00 00 00 00 00 00 00 00 08 01 00 00
01204060  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68    º  ´ Í!¸ LÍ!Th
01204070  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
01204080  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS 
01204090  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode.   $
012040A0  96 6E 38 F5 D2 0F 56 A6 D2 0F 56 A6 D2 0F 56 A6
012040B0  66 93 A7 A6 D9 0F 56 A6 66 93 A5 A6 74 0F 56 A6
012040C0  66 93 A4 A6 CB 0F 56 A6 BF 52 57 A7 DA 0F 56 A6
012040D0  47 51 52 A7 8F 0F 56 A6 4C AF 91 A6 D4 0F 56 A6
012040E0  E9 51 55 A7 C5 0F 56 A6 E9 51 53 A7 CF 0F 56 A6
012040F0  E9 51 52 A7 C0 0F 56 A6 DB 77 C5 A6 DB 0F 56 A6
01204100  D2 0F 57 A6 18 0F 56 A6 78 51 5E A7 C1 0F 56 A6
01204110  78 51 54 A7 D3 0F 56 A6 52 69 63 68 D2 0F 56 A6  xQT§Ó V¦RichÒ V¦
01204120  00 00 00 00 00 00 00 00 50 45 00 00 4C 01 05 00          PE  L
01204130  6A D7 AD 5B 00 00 00 00 00 00 00 00 E0 00 02 01  j×-[        à
01204140  0B 01 0E 00 00 B4 05 00 00 96 01 00 00 00 00 00
01204150  5D 69 00 00 00 10 00 00 00 D0 05 00 00 00 40 00  ]i       Ð    @
01204160  00 10 00 00 00 02 00 00 05 00 01 00 00 00 00 00
01204170  05 00 01 00 00 00 00 00 00 90 07 00 00 04 00 00
01204180  00 00 00 00 03 00 40 81 00 00 10 00 00 10 00 00        @
01204190  00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00
012041A0  00 00 00 00 00 00 00 00 CC EC 06 00 B4 00 00 00          Ìì  ´
012041B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
012041C0  00 00 00 00 00 00 00 00 00 40 07 00 5C 40 00 00           @  \@
012041D0  70 E3 06 00 38 00 00 00 00 00 00 00 00 00 00 00  pã  8
012041E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
012041F0  A8 E3 06 00 40 00 00 00 00 00 00 00 00 00 00 00  ¨ã  @
01204200  00 D0 05 00 F0 02 00 00 00 00 00 00 00 00 00 00   Ð  ð
01204210  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01204220  2E 74 65 78 74 00 00 00 EC B3 05 00 00 10 00 00  .text   ì³
01204230  00 B4 05 00 00 04 00 00 00 00 00 00 00 00 00 00   ´
01204240  00 00 00 00 20 00 00 60 2E 72 64 61 74 61 00 00         `.rdata
01204250  BC 2A 01 00 00 D0 05 00 00 2C 01 00 00 B8 05 00  ¼*   Ð   ,   ¸
01204260  00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 40              @  @
01204270  2E 64 61 74 61 00 00 00 68 24 00 00 00 00 07 00  .data   h$
01204280  00 10 00 00 00 E4 06 00 00 00 00 00 00 00 00 00       ä
01204290  00 00 00 00 40 00 00 C0 2E 67 66 69 64 73 00 00      @  À.gfids
012042A0  B4 00 00 00 00 30 07 00 00 02 00 00 00 F4 06 00  ´    0       ô
```

Figure 1. DropboxAES RAT executable before and after RC4-decoding. (Source: Secureworks)

Startup environment validation

The decompiled pseudocode of DropboxAES RAT's main function (see Figure 2) shows minimal initial functionality. DropboxAES RAT hashes the current executable's path and filename (lines 15 and 16) and returns a two-byte hexadecimal value. The hexadecimal value is converted to a lowercase string and is used as the mutex name in the CreateMutexA call (line 18).

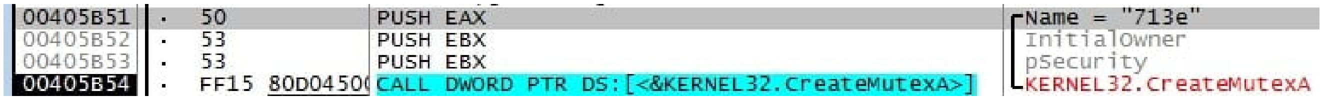


```
int __cdecl DropboxAES_RAT_main(int argc, const char **argv, const char **envp)
{
  unsigned __int16 vFilenameHash; // ax
  CHAR self_Filename; // [esp+Ch] [ebp-114h]
  CHAR MutexName; // [esp+110h] [ebp-10h]
  int v7; // [esp+111h] [ebp-Fh]
  int v8; // [esp+115h] [ebp-Bh]
  char v9; // [esp+119h] [ebp-7h]

  memset(&self_Filename, 0, 0x101u);
  MutexName = 0;
  v7 = 0;
  v8 = 0;
  v9 = 0;
  GetModuleFileNameA(0, &self_Filename, 0x104u);
  vFilenameHash = DropboxAES_RAT_HashString(&self_Filename, strlen(&self_Filename));
  DropboxAES_RAT_strFormat(&MutexName, "%x", vFilenameHash);
  if ( CreateMutexA(0, 0, &MutexName) && GetLastError() != 183 )
  {
    SetErrorMode(4u);
    libcurl_WSAStartup_0(3);
    memset(&Configuration, 0, 0x7E8u);
    memset(&Data, 0, 0xC6u);
    if ( DropboxAES_RAT_BuildConfig() && !DropboxAES_RAT_INIFile_NotExists() && !DropboxAES_RAT_ReadINIFile() )
      DropboxAES_RAT_BuildConfig();
    if ( !DropboxAES_RAT_INIFile_NotExists() )
      DropboxAES_RAT_c2_Main();
    DropboxAES_RAT_SetupWorkingEnvAndPersistence();
    libcurl_WSACleanupAndFreeLibraries();
  }
  return 0;
}
```

Figure 2. DropboxAES RAT's decompiled main function pseudocode. (Source: Secureworks)

As an example, DropboxAES RAT expects to be executed in memory by its loader. Therefore, the executable path and filename should be similar to C:\Users\Example\Desktop\asOELnch.exe. The hashing algorithm uses the path and filename combination results to generate the hash 0x713E. This hash is converted to a lowercase string and is used as the Name argument in the call to CreateMutex (see Figure 3).



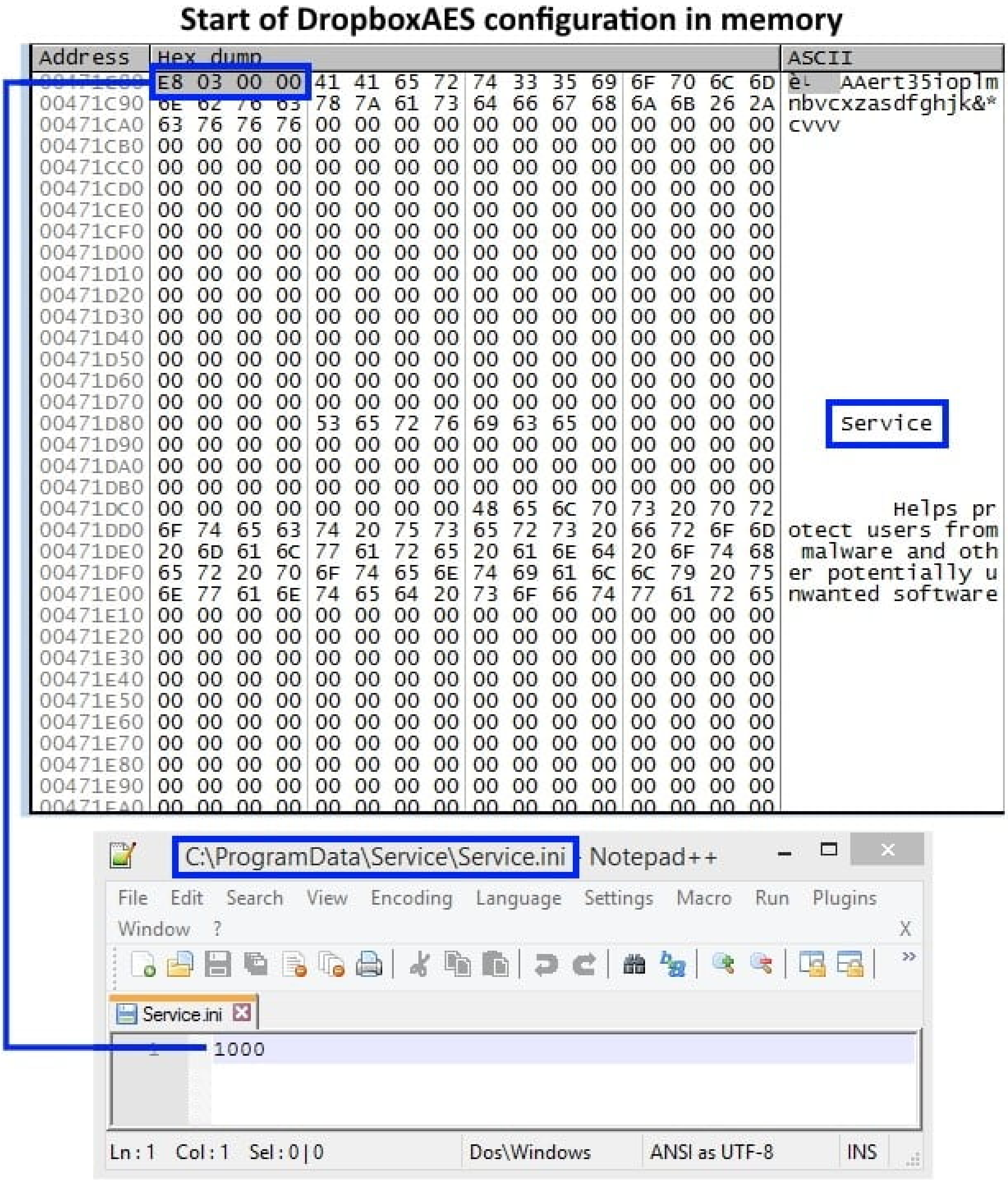
```
00405B51  .  50             PUSH EAX                                   ┌Name = "713e"
00405B52  .  53             PUSH EBX                                   │InitialOwner
00405B53  .  53             PUSH EBX                                   │pSecurity
00405B54  .  FF15 80D0450(  CALL DWORD PTR DS:[<&KERNEL32.CreateMutexA>]  └KERNEL32.CreateMutexA
```

Figure 3. DropboxAES RAT mutex creation using hashed path and filename value as the Name argument. (Source: Secureworks)

If mutex creation is successful, meaning there are no other running instances of DropboxAES RAT, then DropboxAES RAT checks its configuration for the name of the subfolder and INI file within a specific subfolder in the %AllUsersProfile% folder. In the analyzed sample, the configuration value was the string "Service" (see Figure 4), so the checked path and filename was "C:\ProgramData\Service\Service.ini". If the specified INI file does not exist, DropboxAES RAT sets up its working environment and persistence.

Working environment and persistence setup

DropboxAES RAT sets up its working environment by first creating a subfolder within %AllUsersProfile% using the name "Service" specified in its configuration (see Figure 4). The malware sets the Hidden and System attributes, copies the original executable (asOELnch.exe) and DLL (MSVCR100.dll), and creates a file named Service.ini in this subfolder. The Service.ini file contains a single integer, which is specified at the beginning of the DropboxAES RAT configuration (e.g., 0x3E8 hex = 1000 decimal). CTU researchers believe this value may be a campaign identifier.



Start of DropboxAES configuration in memory

```
Address   Hex dump                                         ASCII
00471C80  E8 03 00 00 41 41 65 72 74 33 35 69 6F 70 6C 6D  è└  AAert35ioplm
00471C90  6E 62 76 63 78 7A 61 73 64 66 67 68 6A 6B 26 2A  nbvcxzasdfghjk&*
00471CA0  63 76 76 76 00 00 00 00 00 00 00 00 00 00 00 00  cvvv
00471CB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471CC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471CD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471CE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471CF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D10  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D20  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D30  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D40  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D70  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471D80  00 00 00 00 53 65 72 76 69 63 65 00 00 00 00 00      Service
00471D90  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471DA0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471DB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471DC0  00 00 00 00 00 00 00 00 48 65 6C 70 73 20 70 72          Helps pr
00471DD0  6F 74 65 63 74 20 75 73 65 72 73 20 66 72 6F 6D  otect users from
00471DE0  20 6D 61 6C 77 61 72 65 20 61 6E 64 20 6F 74 68   malware and oth
00471DF0  65 72 20 70 6F 74 65 6E 74 69 61 6C 6C 79 20 75  er potentially u
00471E00  6E 77 61 6E 74 65 64 20 73 6F 66 74 77 61 72 65  nwanted software
00471E10  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E20  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E30  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E40  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E70  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E80  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471E90  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00471EA0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

C:\ProgramData\Service\Service.ini - Notepad++

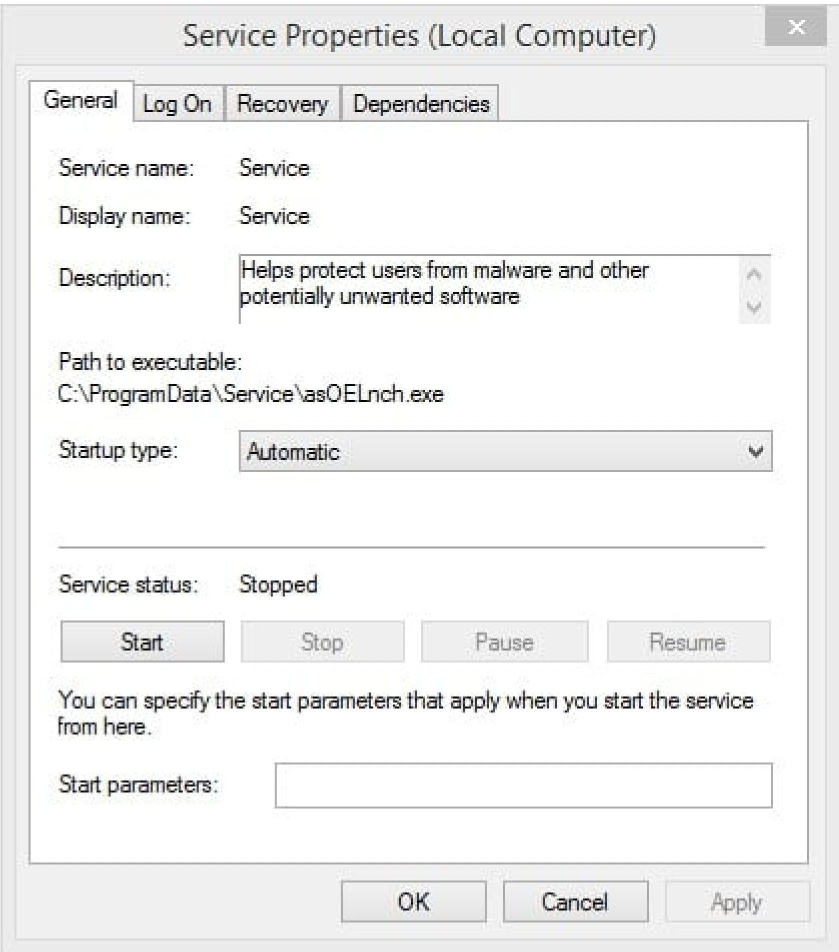
```
1  1000
```

Ln : 1   Col : 1   Sel : 0 | 0   Dos\Windows   ANSI as UTF-8   INS

Figure 4. DropboxAES RAT configuration highlighting the values used in folder/file creation as well as the integer to be stored within the INI file. (Source: Secureworks)

For persistence, DropboxAES RAT first tries to create a Windows Service named with the same “Service” string used for the %AllUsersProfile% subfolder and the INI file (see Figure 5). The malware sets the executable path to the legitimate executable copied to the

%AllUsersProfile%\Service\ directory and sets the description of the service to “Helps protect users from malware and other potentially unwanted software.” The service description is also a value derived from the configuration data. Once the malware has created the Windows Service, it is started.



Service name: Service

Display name: Service

Description: Helps protect users from malware and other potentially unwanted software

Path to executable:
C:\ProgramData\Service\asOELnch.exe

Startup type: Automatic

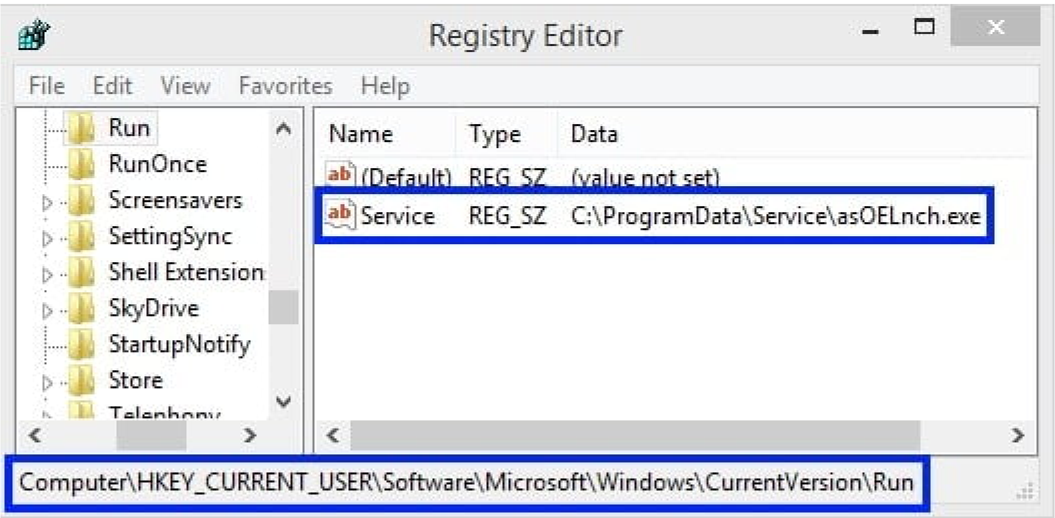
Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

Start parameters:

Figure 5. Service created by DropboxAES RAT for persistence. (Source: Secureworks)

If service creation fails, DropboxAES RAT implements persistence by creating an entry named “Service” in the registry’s HKCU Run key (see Figure 6). The malware then runs the executable that it just copied into the configured working directory at %AllUsersProfile%\Service\ via a call to Kernel32.WinExec.

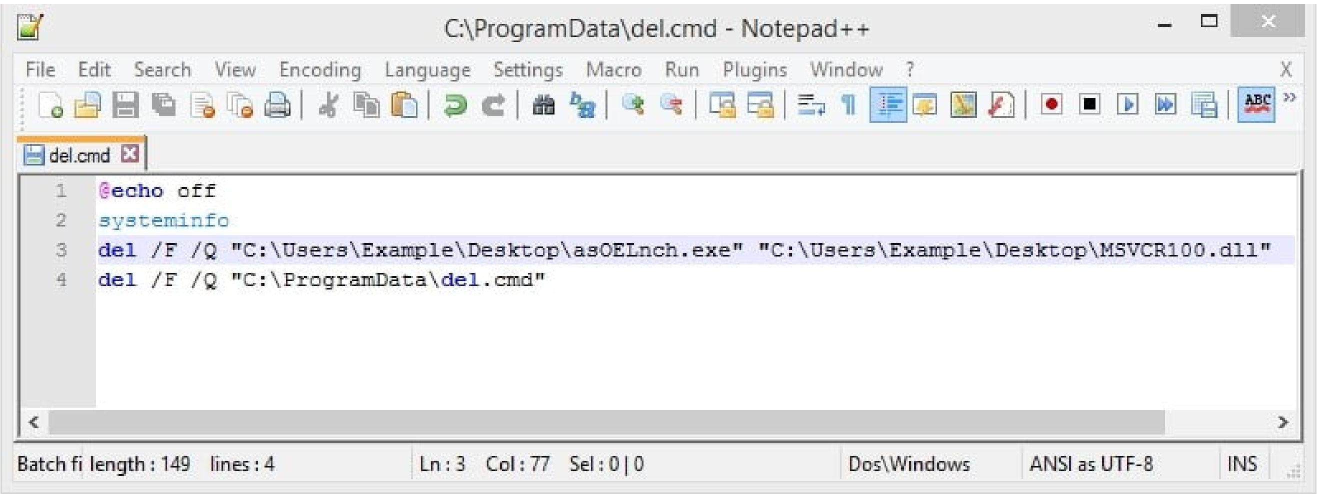


Name	Type	Data
(Default)	REG_SZ	(value not set)
Service	REG_SZ	C:\ProgramData\Service\asOELnch.exe

Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

Figure 6. DropboxAES RAT registry Run key created for persistence. (Source: Secureworks)

With the new malware instance running via either Windows Service or a call to WinExec, DropboxAES RAT drops and executes a batch script named del.cmd within the %AllUsersProfile% path (see Figure 7).This file deletes the executable and DLL files from the original executed path. DropboxAES RAT's use of the 'del' command potentially allows for the deleted files to be forensically restored as it does not perform a secure delete.


C:\ProgramData\del.cmd - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
del.cmd
Batch fi length : 149 lines : 4
Ln : 3 Col : 77 Sel : 0 | 0
Dos\Windows
ANSI as UTF-8
INS

```
@echo off
systeminfo
del /F /Q "C:\Users\Example\Desktop\asOELnch.exe" "C:\Users\Example\Desktop\MSVCR100.dll"
del /F /Q "C:\ProgramData\del.cmd"
```

Figure 7. Contents of del.cmd dropped and executed by DropboxAES RAT. (Source: Secureworks)

Once DropboxAES RAT has set up the working environment, established persistence, and removed the original artifacts from the compromised system, the malware terminates itself in favor of the new running instance launched from the configured working directory.

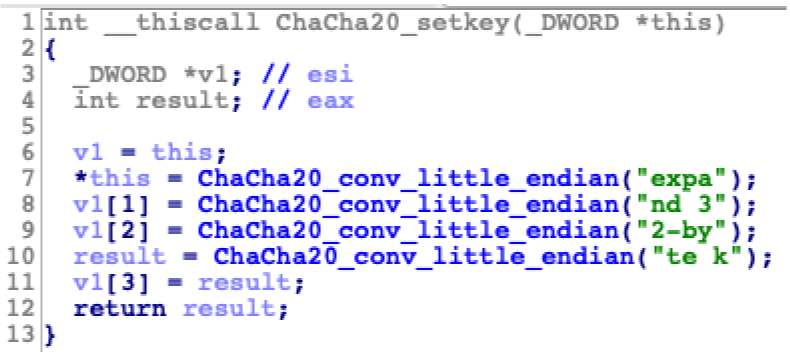
Information gathering

DropboxAES RAT gathers basic information about the host prior to establishing communication with its C2 server. First, it generates a "uuid" value by appending the mutex value to the lowercase MAC address for the host's network adapter. For example, if the MAC address for the host is “00-0C-29-06-65-9F” and the mutex value is “713e”, the resulting uuid value is “00-0c-29-06-65-9f-713e”.

Next, DropboxAES RAT obtains the hostname, username, internal IP address, and operating system version of the infected host. It retrieves an integer value specified in its configuration, which is also present within the INI file, and formats the data into a JSON data structure:

```
{
"uuid":"00-0d-28-06-65-9f-713e",
"pcname":"example-hostname",
"user":"example-username",
"ip":"10.11.12.13",
"os":"Windows 8",
"time":1000
}
```

The collected data is encoded with the ChaCha20 stream cipher, which is partially identified by the constant “expand 32-byte k” within the code (see Figure 8). The resulting encoded data is then Base64-encoded.



```
int __thiscall ChaCha20_setkey(_DWORD *this)
{
  _DWORD *v1; // esi
  int result; // eax

  v1 = this;
  *this = ChaCha20_conv_little_endian("expa");
  v1[1] = ChaCha20_conv_little_endian("nd 3");
  v1[2] = ChaCha20_conv_little_endian("2-by");
  result = ChaCha20_conv_little_endian("te k");
  v1[3] = result;
  return result;
}
```

Figure 8. DropboxAES RAT ‘expand 32-byte k’ constant in the ChaCha20 stream cipher code. (Source: Secureworks)

Dropbox authentication and C2 folder

DropboxAES RAT is proxy aware. Prior to establishing C2 communications, it determines if the system is configured to use a proxy by inspecting the contents of the HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyServer registry key value. If a proxy is configured, DropboxAES RAT uses these settings in its network configuration.

DropboxAES RAT needs to successfully authenticate to Dropbox to interact with the file-sharing service. To achieve this authentication, the malware obtains the Authorization token specified within its configuration (see Figure 9) and includes the value in the Authorization HTTP header. The following is the partially redacted Authorization token in the analyzed sample’s configuration:

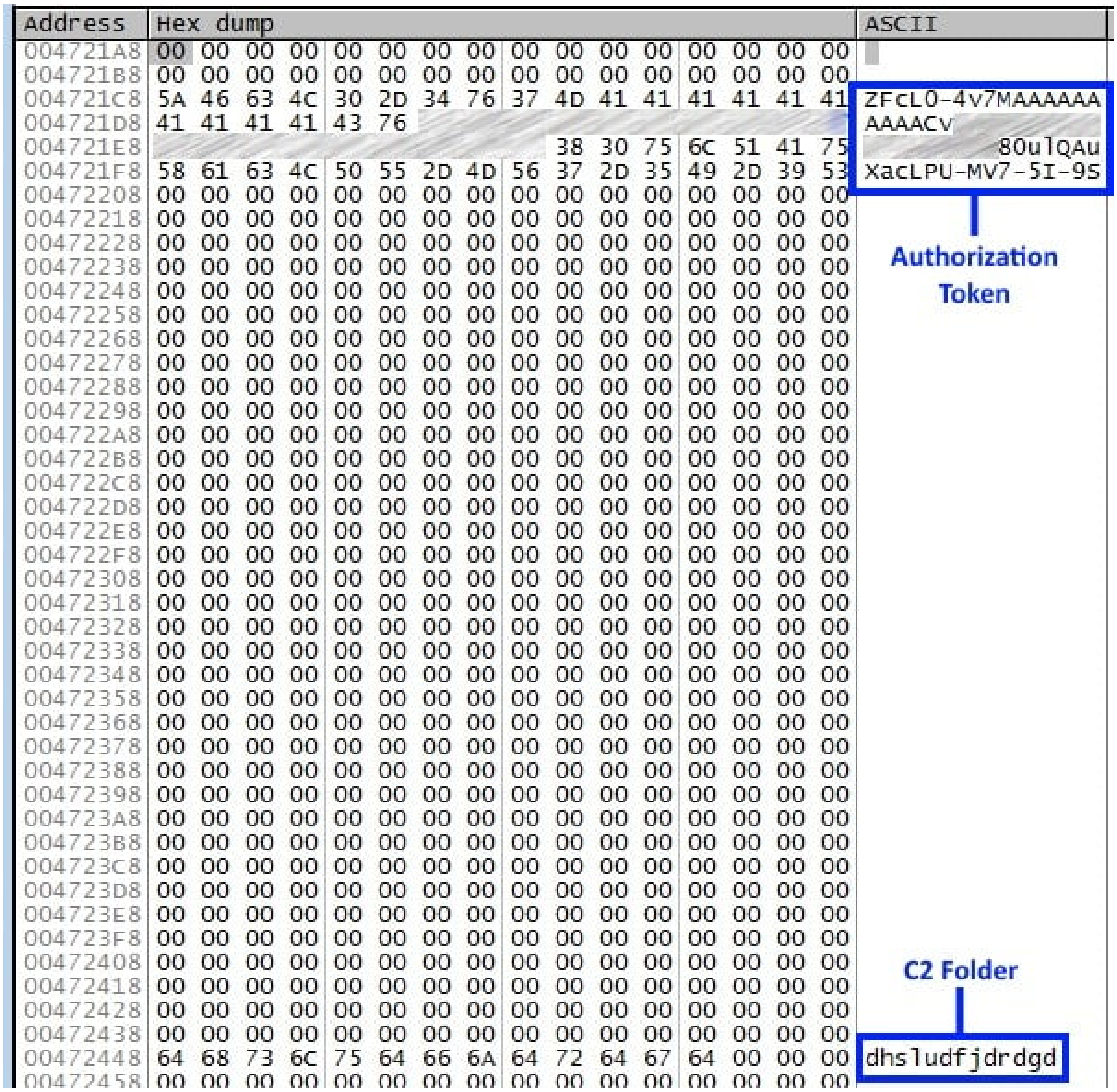
```
ZFcL0-4v7MAAAAAAAAAACv<REDACTED>80ulQAuXacLPU-MV7-5I-9S
```

This token results in the following Authorization HTTP header value:

```
Authorization: Bearer ZFcL0-4v7MAAAAAAAAAACv<REDACTED>80ulQAuXacLPU-MV7-5I-9S
```

A valid Authorization token value allows DropboxAES RAT to view, download, upload, and delete files located in Dropbox folders owned by the threat actor.

The name of the remote folder on Dropbox that DropboxAES RAT uses for C2 communications is also specified within the configuration. In the analyzed sample, the folder value was “dhsludfjdrdgd”.

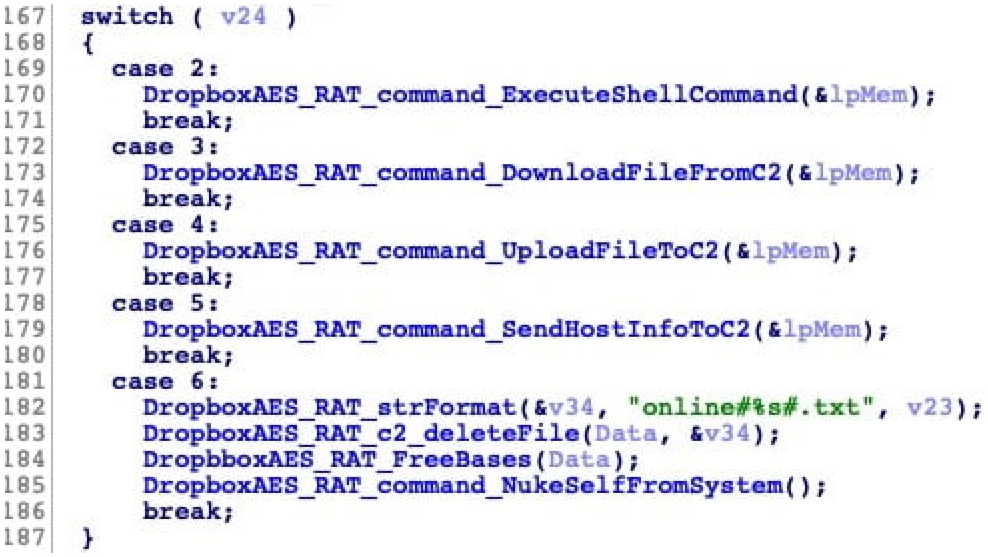


```
Address  Hex dump                                         ASCII
004721A8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004721B8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004721C8 5A 46 63 4C 30 2D 34 76 37 4D 41 41 41 41 41 41 ZFcL0-4v7MAAAAAA
004721D8 41 41 41 41 43 76 [redacted]                     AAAACv[redacted]
004721E8 [redacted]           38 30 75 6C 51 41 75        [redacted]80ulQAu
004721F8 58 61 63 4C 50 55 2D 4D 56 37 2D 35 49 2D 39 53 XacLPU-MV7-5I-9S
00472208 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472218 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472228 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472238 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472248 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472258 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472268 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472278 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472288 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472298 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004722A8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004722B8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004722C8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004722D8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004722E8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004722F8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472308 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472318 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472328 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472338 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472348 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472358 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472368 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472378 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472388 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472398 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004723A8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004723B8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004723C8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004723D8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004723E8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
004723F8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472408 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472418 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472428 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472438 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00472448 64 68 73 6C 75 64 66 6A 64 72 64 67 64 00 00 00 dhsludfjdrdgd
00472458 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Figure 9. Dropbox Authorization token and C2 folder specified in the DropboxAES RAT configuration. (Source: Secureworks)

Before uploading the encoded host information, DropboxAES RAT checks if an “online” file for this host already exists. For example, using the uuid generated from the analyzed example, the online filename would be “online#00-0d-28-06-65-9f-713e#.txt”. If an online version exists, the malware deletes it from the C2 server. The malware then uploads the encoded host information to the C2 server using the same “online” filename.

DropboxAES RAT searches the C2 folder on Dropbox for “job” files that contain commands issued by the C2 server. The search looks for files that contains the uuid value associated with the infected host. DropboxAES RAT downloads and processes any job files it finds that match the specific filename format (e.g., job####.txt). A switch statement uses the command integer specified in the job filename to determine which command should be executed (see Figure 10).



```
switch ( v24 )
{
  case 2:
    DropboxAES_RAT_command_ExecuteShellCommand(&lpMem);
    break;
  case 3:
    DropboxAES_RAT_command_DownloadFileFromC2(&lpMem);
    break;
  case 4:
    DropboxAES_RAT_command_UploadFileToC2(&lpMem);
    break;
  case 5:
    DropboxAES_RAT_command_SendHostInfoToC2(&lpMem);
    break;
  case 6:
    DropboxAES_RAT_strFormat(&v34, "online#%s#.txt", v23);
    DropboxAES_RAT_c2_deleteFile(Data, &v34);
    DropbboxAES_RAT_FreeBases(Data);
    DropboxAES_RAT_command_NukeSelfFromSystem();
    break;
}
```

Figure 10. Decompiled pseudocode for DropboxAES RAT C2 command switch statement. (Source: Secureworks)

Table 1 describes the command values supported by DropboxAES RAT and their purpose.

Command value	Command description
2	Run the specified command via a call to kernel32.CreateProcessA
3	Download the specified file from the C2 server
4	Upload the specified file to the C2 server
5	Resend the uuid, hostname, username, IP address, operating system data, and the integer found in configuration data to the C2 server

6	Remove all traces of DropboxAES RAT from the compromised system, including deleting the ‘online’ check-in file present on the C2 server, removing all persistence mechanisms (registry Run key and Windows Service), performing a shallow deletion of DropboxAES RAT executables via the del.cmd batch script, and terminating the currently running DropboxAES RAT executable

Table 1. Valid DropboxAES RAT commands.

After DropboxAES RAT executes the command, the result or command output is encoded and uploaded to the C2 folder with a ‘back’ filename (e.g., back###.txt). When completed, DropboxAES RAT continuously checks for the presence of another job file until the running DropboxAES RAT process is terminated. DropboxAES RAT establishes C2 communication using raw sockets via the libcurl library statically compiled into the binary. As it does not rely on libraries resident on the infected host, traces of DropboxAES RAT’s historical network activity will be minimal or nonexistent.

DropboxAES RAT C2 files and formats

DropboxAES RAT utilizes the Dropbox online service for C2 communications by uploading and downloading documents stored in the configured folder on the Dropbox server. Table 2 summarizes all of the files that DropboxAES RAT uses for its C2 communications. They are listed in the order they occur.

Format	Example	Description
online##.txt	online#00-0c-29-06-65-9f-ce8a#.txt	Uploaded to the C2 server by the compromised host at initial check-in. The uuid value is unique and associates all C2 communication for the compromised host. The file contents contain ChaCha20 and Base64-encoded data about the compromised host.
job####.txt	job#00-0c-29-06-65-9f-ce8a#2#gyK0slzo#.txt	Placed on the C2 server by the threat actor. Downloaded and processed by the compromised host matching the uuid specified in the filename. In this example, the command order is denoted as the number 2 between the # symbols in the filename. The file contains arguments for the command and must be Base64 and ChaCha20-decoded before the arguments can be used by the malware. The random string at the end of the filename acts as a job ID that the threat actor uses to associate responses with issued commands.

Format	Example	Description
back###.txt	back#00-0c-29-06-65-9f-ce8a#gyk0slzo#.txt	Uploaded to the C2 server by the compromised host. Contains the output for the command issued by the corresponding job file on the C2 server. The job ID appended to the end of the filename matches the job ID specified in the job file that originally issued the command. Prior to being uploaded to the C2 server, the command output is ChaCha20 and Base64-encoded.

Table 2. DropboxAES RAT files.

DropboxAES RAT configuration values

Table 3 summarizes the key values contained within the DropboxAES RAT configuration.

Value	Purpose
0x3E8 (1000)	Integer stored in the INI file. Used as the time value in the JSON sent to the C2 server within the online file. CTU researchers believe this value to be a campaign or target identifier.
AAert35ioplmnbvcxzasdfghjk&*cvvv	Secret key used for ChaCha20 encoding and decoding.
Service	String used for: • %AllUsersProfile% subfolder name • INI filename • Service name used for persistence • Registry Run key value name used for persistence
Helps protect users from malware and other potentially unwanted software	Description of service used for persistence
ZFcL0-4v7MAAAAAAAAAACv<REDACTED>80ulQAuXacLPU-MV7-5I-9S	Dropbox API Authorization token
Helps protect users from malware and other potentially unwanted software	Description of service used for persistence

dhsludfjdrdgd	Subfolder on Dropbox containing C2 files

Table 3. DropboxAES RAT configuration values.

Threat indicators

The threat indicators in Table 4 can be used to detect activity related to the DropboxAES RAT.

Indicator	Type	Context
76d4866c5ff6d821313e1461b7875544	MD5 hash	DropboxAES RAT loader DLL
406353b156239ed08b27de0c38d16dfc6d031d88	SHA1 hash	DropboxAES RAT loader DLL
ebdf52e13e69435ea7a85c9e38ac1f5045c32fe30d4e5aa66149a53054183f7f	SHA256 hash	DropboxAES RAT loader DLL
8f0fcb5a80b2bca62d79f0d1cbdc93fb	MD5 hash	DropboxAES RAT executable
9c162e042e0a892924f8415f7d72fe4f966bae7d	SHA1 hash	DropboxAES RAT executable
f34725937839ae6c0470596e9c81b4572e2361737fbdb3a13983a25dfabd1c3a	SHA256 hash	DropboxAES RAT executable

Table 4. Indicators for this threat.

Conclusion

DropboxAES RAT is a simple but effective remote access trojan that lets a remote threat actor control a compromised host using primitive commands. When these commands are utilized together, the malware exhibits great flexibility and capability. The use of Dropbox for C2 communications and the generic configurable artifacts on disk make detection and prevention of DropboxAES RAT activity extremely difficult, if not impossible. Detection is especially challenging for organizations that use Dropbox for business purposes within their environments.