BÁO CÁO BÀI TẬP

Môn học: Kỹ thuật phân tích mã độc

Tên chủ đề: BÀI TẬP NHÓM SỐ 01

GVHD: ThS Nguyễn Công Danh

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT137.Q11.ANTT

STT	Họ và tên	MSSV	Email
1	Nguyễn Thanh Hưng	22520517	22520517@gm.uit.edu.vn
2	Từ Chí Kiên	22520713	22520713@gm.uit.edu.vn
3	Hà Minh Quân	22521177	22521177@gm.uit.edu.vn
4	Nguyễn Nhật Quang	22521203	22521203@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Công việc	Kết quả tự đánh giá
1	Tạo 1 máy chủ web (tự tạo, local) bằng giao thức https (self signed)	100%
2	Download một tập tin là hình ảnh được lưu trên máy chủ web Lưu vào thư mục AppData của user hiện tại trên máy tính Windows.	100%
3	Sử dụng 1 kỹ thuật persistent để hình ảnh đó được load lên mỗi khi máy tính khởi động lại. Ứng dụng này sau khi thực hiện các phần trên sẽ sleep 10 giây, sau đó tắt.	100%
4	Monitor và mô tả lại những gì đã xảy ra với máy tính sau khi ứng dụng đã thực hiện các bước trên	100%
5	Áp dụng một kỹ thuật anti-debug/anti-reverse/packing tự chọn	100%
6	Nghiên cứu tự dịch ngược mã nguồn và mô tả từng chức năng theo mã nguồn đã dịch ngược	100%

[1] Ghi nội dung công việc

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

BÁO CÁO CHI TIẾT

1. Máy chủ tự tạo có self-signed

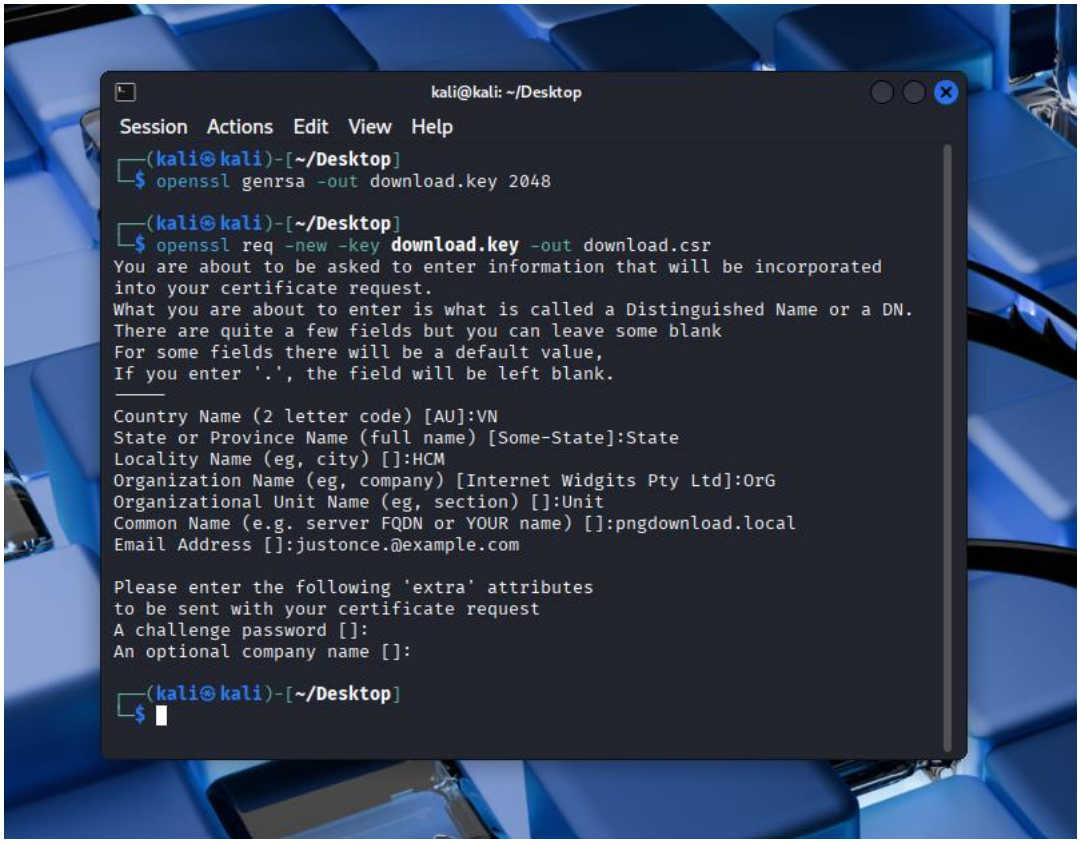
Làm theo hướng dẫn của https://medium.com/@pasanglamatamang/configuring-a-self-signed-ssl-certificate-on-a-apache-server-cbcd6eefdf1a

Tạo một key private

```
openssl genrsa -out download.key 2048
```

Tạo Certificate Signing Request

```
openssl req -new -key download.key -out download.csr
```



Tạo chứng nhận tự ký

```
openssl x509 -req -days 365 -in download.csr -signkey download.key -out download.crt
```

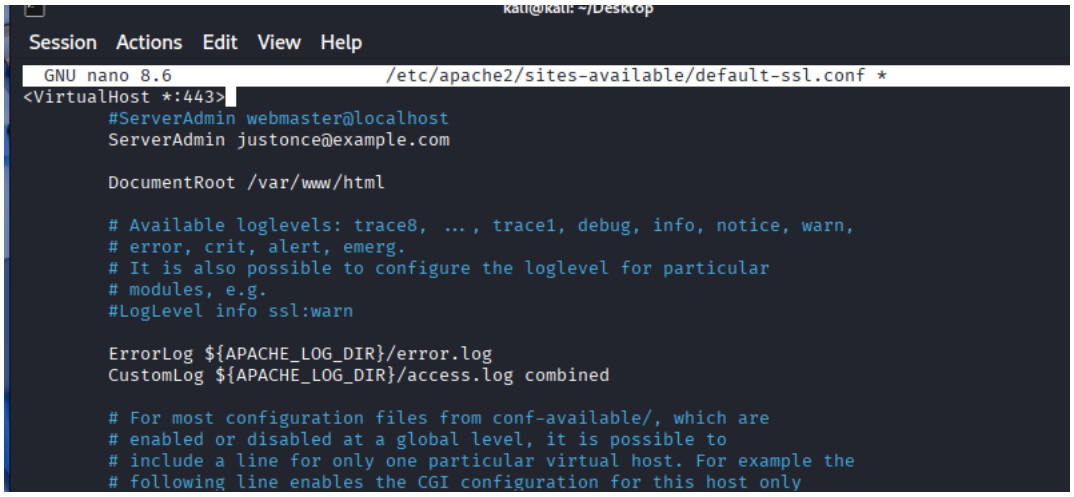
```
┌──(kali㉿kali)-[~/Desktop]
└─$ openssl x509 -req -days 365 -in download.csr -signkey download.key -out d
ownload.crt
Certificate request self-signature ok
subject=C=VN, ST=State, L=HCM, O=OrG, OU=Unit, CN=pngdownload.local, emailAdd
ress=justonce.@example.com
```

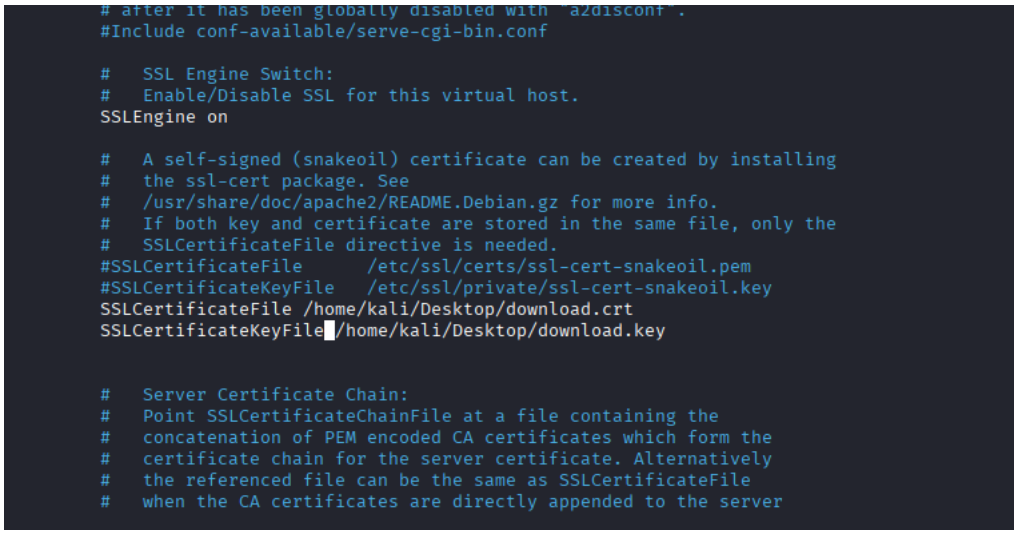
Lưu bản sao

sudo cp /etc/apache2/sites-available/default-ssl.conf /etc/apache2/sites-available/default-ssl.conf.bak

Thay đổi cấu hình để nhận .crt và key mới tạo

sudo nano /etc/apache2/sites-available/default-ssl.conf

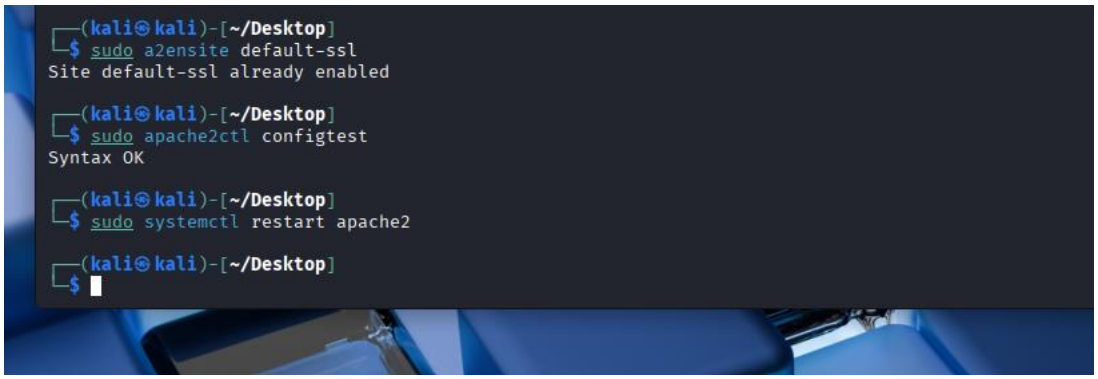




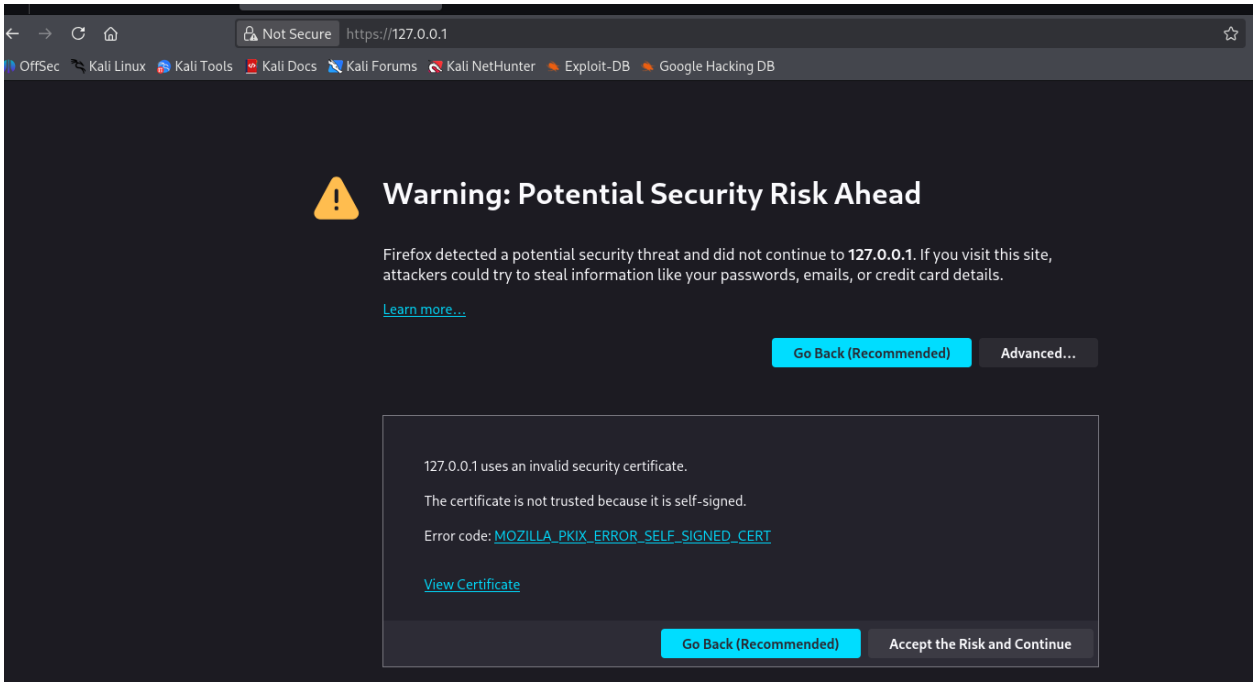
Đặt servername là ip loopback

sudo nano /etc/apache2/apache2.conf

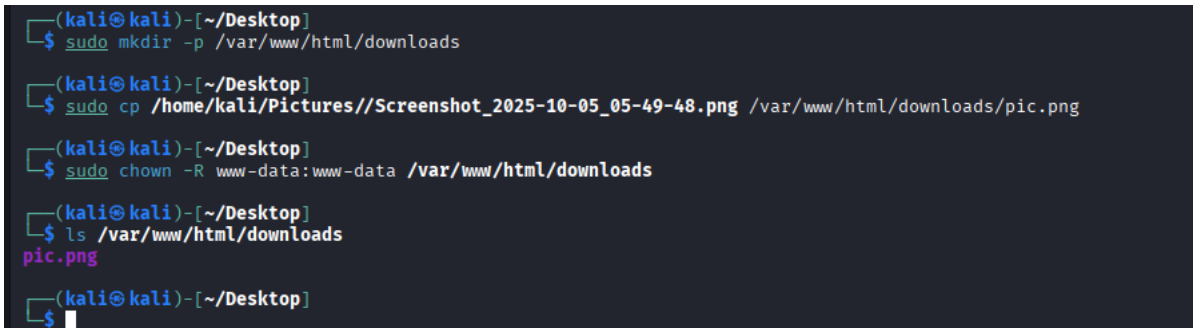
Kích hoạt ssl và restart lại apche



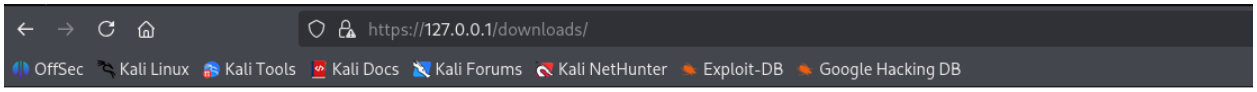
Kết quả là trang web self-signed



Tạo một thư mục chứa ảnh cần tải xuống



Trong đường dẫn trang web



Index of /downloads

Name	Last modified	Size	Description
Parent Directory		-	
pic.png	2025-10-05 05:56	1.3M	

Apache/2.4.65 (Debian) Server at 127.0.0.1 Port 443

2. Chương trình tải ảnh và persistent

Xây dựng ứng dụng mã độc sử dụng c++:

Đầu tiên phần tải xuống:

Sử dụng WinHttpOpen

```
bool download_file(wstring &url, wstring &outpath, wstring &host) {
    HINTERNET hOpen = nullptr, hConnect = nullptr, hRequest = nullptr;
    DWORD dwSize = 0, dwDownloaded = 0;
    LPSTR pszOutBuffer = nullptr;
    FILE *pFile = nullptr;

    // Mở session
    hOpen = WinHttpOpen(L"DownloaderApp",
                        WINHTTP_ACCESS_TYPE_DEFAULT_PROXY,
                        WINHTTP_NO_PROXY_NAME,
                        WINHTTP_NO_PROXY_BYPASS, 0);
    if (!hOpen) {
        wprintf(L"WinHttpOpen failed (0x%.8X)\n", GetLastError());
        cleanup(hOpen, hConnect, hRequest);
        return false;
    }
```

Tạo kết nối với WinHttpConnect, Tạo Request handler dùng WinHttpOpenRequest

```
// Tạo kết nối
hConnect = WinHttpConnect(hOpen, host.c_str(), INTERNET_DEFAULT_HTTPS_PORT, 0);
if (!hConnect) {
    wprintf(L"WinHttpConnect failed (0x%.8X)\n", GetLastError());
    cleanup(hOpen, hConnect, hRequest);
    return false;
}

// Tạo request
hRequest = WinHttpOpenRequest(hConnect, L"GET", url.c_str(),
                              NULL, WINHTTP_NO_REFERER,
                              WINHTTP_DEFAULT_ACCEPT_TYPES,
                              WINHTTP_FLAG_SECURE);
if (!hRequest) {
    wprintf(L"WinHttpOpenRequest failed (0x%.8X)\n", GetLastError());
    cleanup(hOpen, hConnect, hRequest);
    return false;
}
```

Sử dụng WinHttpSendRequest để request đến, do request là self-signed nên sau khi thất bại lần đầu bật các cờ bỏ qua chứng chỉ không tin cậy, sau đó kết nối lại.

```
// Xử lí request self-signed
if (!WinHttpSendRequest(hRequest,
                        WINHTTP_NO_ADDITIONAL_HEADERS, 0,
                        nullptr, 0, 0, 0)) {
    // Sau khi request đầu thất bại thì có thể chỉnh lại để nhận máy chủ self-signed
    if (GetLastError() == ERROR_WINHTTP_SECURE_FAILURE) {
        DWORD dwFlags =
            SECURITY_FLAG_IGNORE_UNKNOWN_CA |
            SECURITY_FLAG_IGNORE_CERT_WRONG_USAGE |
            SECURITY_FLAG_IGNORE_CERT_CN_INVALID |
            SECURITY_FLAG_IGNORE_CERT_DATE_INVALID;
        WinHttpSetOption(hRequest, WINHTTP_OPTION_SECURITY_FLAGS, &dwFlags, sizeof(dwFlags));
        // Sau đó request lại
        if (!WinHttpSendRequest(hRequest,
                                WINHTTP_NO_ADDITIONAL_HEADERS, 0,
                                nullptr, 0, 0, 0)) {
            cleanup(hOpen, hConnect, hRequest);
            return false;
        }
    } else {
        cleanup(hOpen, hConnect, hRequest);
        return false;
    }
}
```

Nhận response sử dụng WinHttpReceive và _wfopen để mở file

```
// Nhận response
if (!WinHttpReceiveResponse(hRequest, nullptr)) {
    cleanup(hOpen, hConnect, hRequest);
    return false;
}

// Mở file
pFile = _wfopen(outpath.c_str(), L"wb");
if (!pFile) {
    cleanup(hOpen, hConnect, hRequest);
    return false;
}
```

Kiểm tra dữ liệu có lấy được bằng lệnh WinHttpQueryDataAvailable, WinHttpReadData để đọc dữ liệu

```
    do {
        if (!WinHttpQueryDataAvailable(hRequest, &dwSize)) {
            printf("Error %u in WinHttpQueryDataAvailable.\n", GetLastError());
            break;
        }
        if (dwSize == 0) break;
        // Tạo biến chứa dữ liệu nhận được
        pszOutBuffer = new char[dwSize + 1];
        if (!pszOutBuffer) {
            printf("Out of memory\n");
            break;
        }
        // Nhận dữ liệu
        ZeroMemory(pszOutBuffer, dwSize + 1);

        //Ghi vào file
        if (!WinHttpReadData(hRequest, (LPVOID)pszOutBuffer, dwSize, &dwDownloaded)) {
            printf("Error %u in WinHttpReadData.\n", GetLastError());
        } else if (dwDownloaded > 0) {
            fwrite(pszOutBuffer, 1, dwDownloaded, pFile);
        }
        // Giải phóng biến
        delete[] pszOutBuffer;
    } while (dwSize > 0);
```

```
    // Đóng file và các handle
    fclose(pFile);
    cleanup(hOpen, hConnect, hRequest);
    return true;
}
```

Hàm cleanup dùng để đóng các handle sử dụng WinHttpCloseHandle

```
void cleanup(HINTERNET &hOpen,HINTERNET &hConnect,HINTERNET &hRequest){
    if (hRequest) {
        WinHttpCloseHandle(hRequest);
        hRequest = nullptr;
    }
    if (hConnect) {
        WinHttpCloseHandle(hConnect);
        hConnect = nullptr;
    }
    if (hOpen) {
        WinHttpCloseHandle(hOpen);
        hOpen = nullptr;
    }
    return ;
}
```

Để tìm được thư mục AppData của người dùng hiện tại bằng cách sử dụng hàm SHGetFolderPath để tìm thư mục của người dùng hiện tại kết hợp với \\AppData

```
wstring getAPPDATApath(){
    TCHAR szPath[MAX_PATH];

    if(SUCCEEDED(SHGetFolderPath(NULL,
                                 CSIDL_PROFILE,
                                 NULL,
                                 0,
                                 szPath)))
    {
       return wstring(szPath) + L"\\AppData";
    }
    return L"."; // Trả về thư mục hiện tại nếu thất bại
}
```

Cuối cùng là hàm tạo persistent, ở đây sử dụng kỹ thuật thêm giá trị vào runkey, để khi người dùng login sẽ kích hoạt một chương trình.

Ở đây sử dụng chương trình đã có sẵn mspaint.exe để mở file ảnh

Sử dụng GetSystemDirectoryW để lấy đường dẫn đến thư mục system

```
bool setRunAtStartup_OpenImage(const wstring &imageFullPath, const wstring &regValueName)
{
    // Tìm vị trí của thư mục system
    wstring systemFolder;
    {
        WCHAR buf[MAX_PATH];
        UINT len = GetSystemDirectoryW(buf, MAX_PATH);
        if (len == 0 || len >= MAX_PATH) return false;
        systemFolder = buf;
    }
    // Truy xuất đường dẫn đến chương trình mspaint
    wstring paintExe = systemFolder + L"\\mspaint.exe";

    // Tạo câu lệnh: "C:\Windows\System32\mspaint.exe" "Đường dẫn đến downloaded.png"
    wstring cmd = L"\"" + paintExe + L"\" \"" + imageFullPath + L"\"";
```

Sử dụng RegOpenKeyExW để mở key đó, nếu không tồn tại thì dùng RegCreateKeyExW để tạo key đó.

```
// Mở key run để chuẩn bị thêm câu lệnh trên vào
HKEY hKey = NULL;
LONG lRes = RegOpenKeyExW(
    HKEY_CURRENT_USER,
    L"Software\\Microsoft\\Windows\\CurrentVersion\\Run",
    0,
    KEY_SET_VALUE,
    &hKey
);


if (lRes != ERROR_SUCCESS) {
    // Nếu thất bại sẽ cố tạo key trong registry
    lRes = RegCreateKeyExW(
        HKEY_CURRENT_USER,
        L"Software\\Microsoft\\Windows\\CurrentVersion\\Run",
        0, NULL, 0, KEY_SET_VALUE, NULL, &hKey, NULL
    );
    if (lRes != ERROR_SUCCESS) {
        return false;
    }
}
```

Sử dụng RegSetValueExW để tạo giá trị, giá trị đó là câu lệnh đã tạo trên.

```
// Thêm câu lệnh trên vào registry run key để chạy mspaint mở tệp ảnh
lRes = RegSetValueExW(
    hKey,
    regValueName.c_str(),
    0,
    REG_SZ,
    (const BYTE*)cmd.c_str(),
    (DWORD)((cmd.size() + 1) * sizeof(wchar_t))
);
RegCloseKey(hKey);

if (lRes != ERROR_SUCCESS) {
    return false;
}
return true;
```

Cuối cùng trong lệnh main

```cpp
int wmain(int argc, wchar_t* argv[]) {
    wstring host = L"192.168.106.131";
    wstring url  = L"downloads/pic.png";
    wstring AppDatapath = getAPPDATApath();
    wstring outpath = AppDatapath + L"\\downloaded.png";

    bool ok = download_file(url, outpath, host);

    if (ok) {
        wcout << L"Downloaded successfully to: " << outpath << std::endl;
        if (setRunAtStartup_OpenImage(outpath)) {
        wcout << L"Registry key added. mspaint will open the image on startup." << std::endl;
        } else {
            wcerr << L"Failed to add registry key!" << std::endl;
        }
    } else {
        wcerr << L"Download failed!" << std::endl;
    }
    Sleep(10000);
    return 0;
}
```

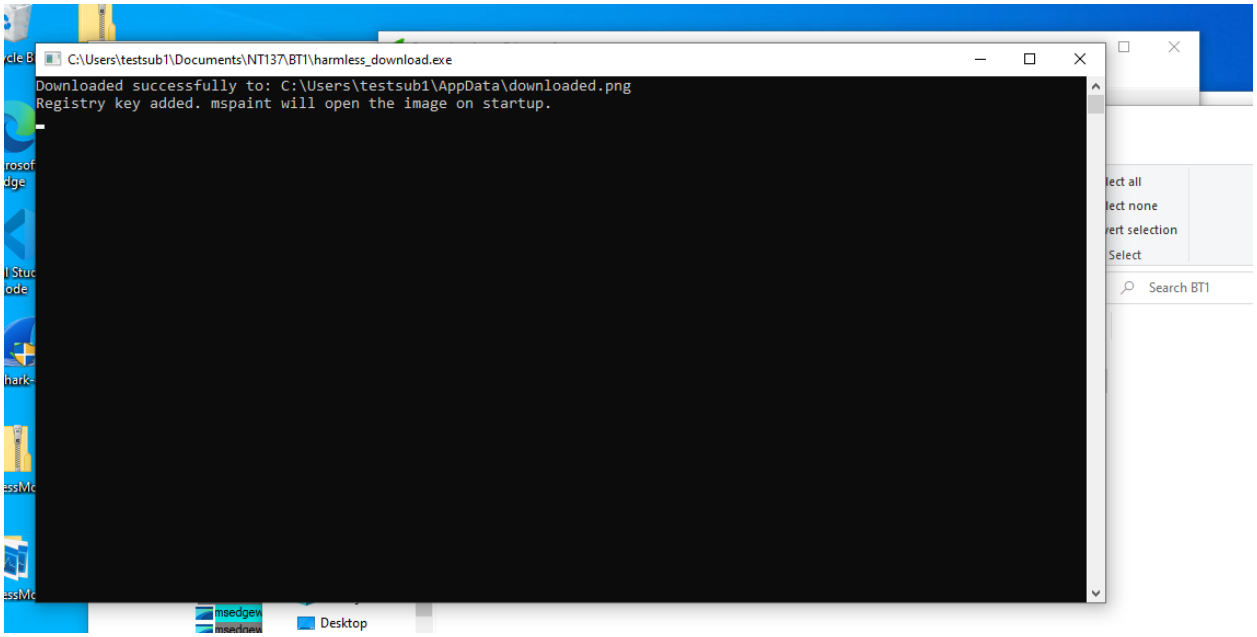
Chạy các hàm trên với 192.168.106.131 là ip của máy chủ, downloads/pic.png là đường dẫn tải ảnh, downloaded.png là tên tệp lưu trên máy, hàm sleep(10000) để chương trình ngủ 10s (10000 mili second) sau khi thực hiện xong.

Chương trình được complie sử dụng g++(MinGW g++), -static(link tất cả những thư viện có thể link), -static-libgcc(link các thư viện GCC runtime), -static-libstdc++(link các thư viện C++ standard), -lwinhttp(link thư viện WinHTTP), -municode(cho phép Unicode). Những option này nhằm chạy chương trình trong máy không có MinGW.

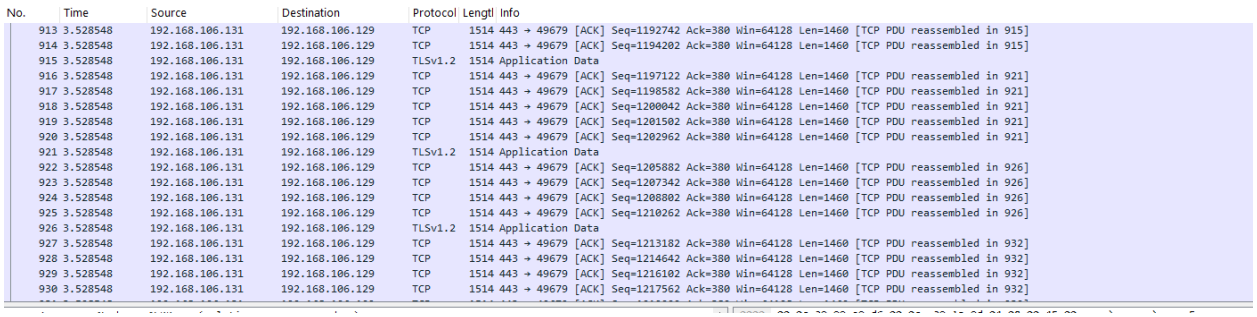
```
PS C:\Users\testsub1\Documents\NT137\BT1> g++ harmless_download.cpp -o harmless_download.exe -static -static-libgcc -static-libstdc++ -lwinhttp -municode
PS C:\Users\testsub1\Documents\NT137\BT1>
```

3. Giám sát hoạt động của chương trình

Trong quá trình chương trình thực hiện

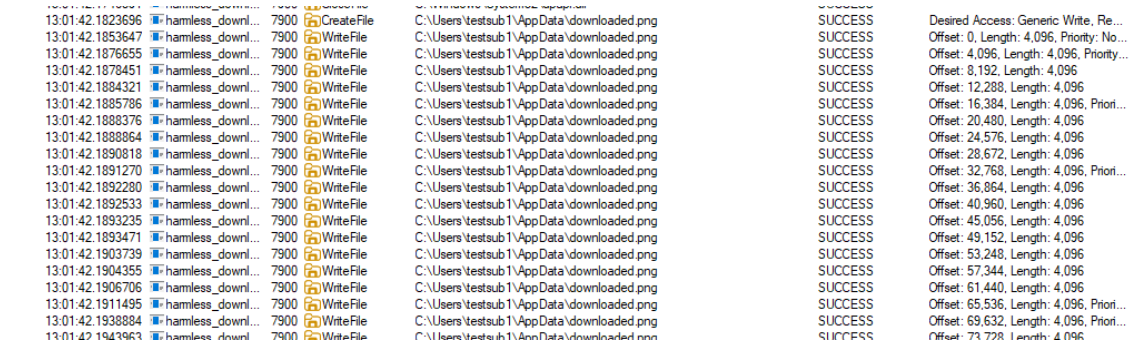


Trên WireShark sẽ thấy các gói tin giao tiếp giữa máy nạn nhân 192.168.106.129 với máy chủ 192.168.106.131 qua cổng 443 (https)



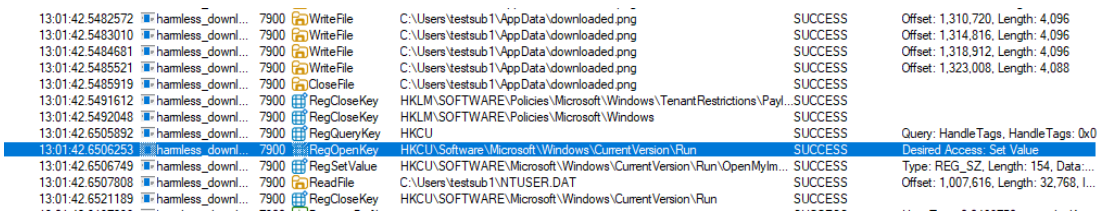
No.	Time	Source	Destination	Protocol	Length	Info
913	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1192742 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 915]
914	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1194202 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 915]
915	3.528548	192.168.106.131	192.168.106.129	TLSv1.2	1514	Application Data
916	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1197122 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 921]
917	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1198582 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 921]
918	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1200042 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 921]
919	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1201502 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 921]
920	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1202962 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 921]
921	3.528548	192.168.106.131	192.168.106.129	TLSv1.2	1514	Application Data
922	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1205882 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 926]
923	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1207342 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 926]
924	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1208802 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 926]
925	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1210262 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 926]
926	3.528548	192.168.106.131	192.168.106.129	TLSv1.2	1514	Application Data
927	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1213182 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 932]
928	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1214642 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 932]
929	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1216102 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 932]
930	3.528548	192.168.106.131	192.168.106.129	TCP	1514	443 → 49679 [ACK] Seq=1217562 Ack=380 Win=64128 Len=1460 [TCP PDU reassembled in 932]

Process Monitor: Bắt được các process phần tải và lưu tệp ảnh.



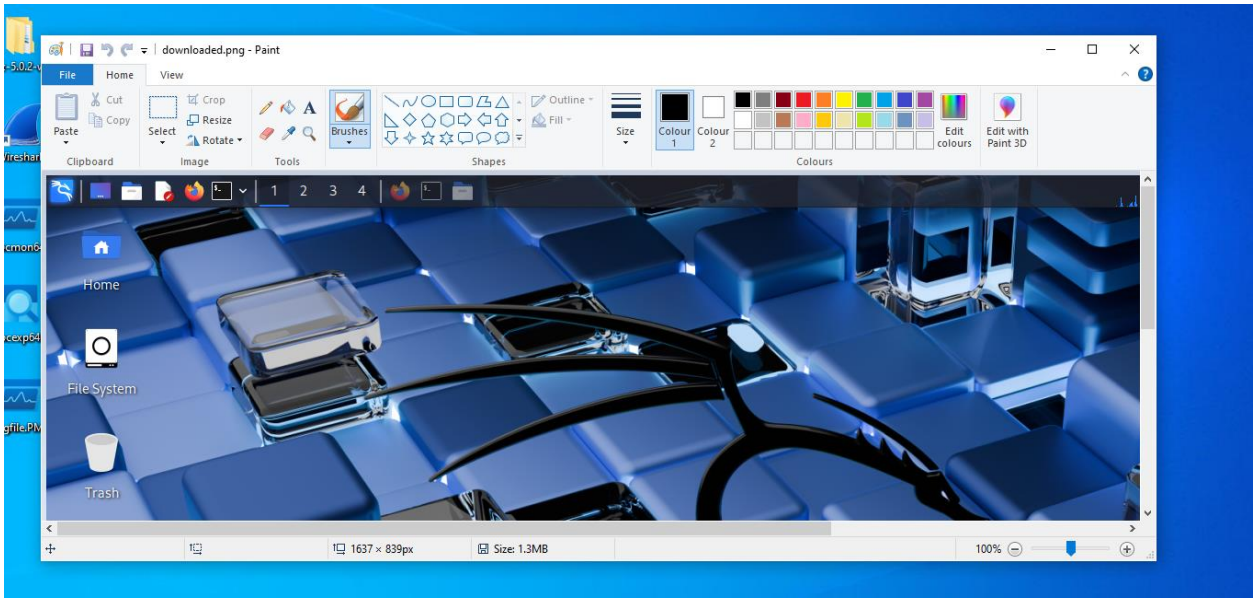
13:01:42.1823696	harmless_downl...	7900	CreateFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Desired Access: Generic Write, Re...
13:01:42.1853647	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 0, Length: 4,096, Priority: No...
13:01:42.1876655	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 4,096, Length: 4,096, Priority...
13:01:42.1878451	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 8,192, Length: 4,096
13:01:42.1884321	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 12,288, Length: 4,096
13:01:42.1885786	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 16,384, Length: 4,096, Priori...
13:01:42.1888376	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 20,480, Length: 4,096
13:01:42.1888864	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 24,576, Length: 4,096
13:01:42.1890818	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 28,672, Length: 4,096
13:01:42.1891270	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 32,768, Length: 4,096, Priori...
13:01:42.1892280	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 36,864, Length: 4,096
13:01:42.1892533	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 40,960, Length: 4,096
13:01:42.1893235	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 45,056, Length: 4,096
13:01:42.1893471	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 49,152, Length: 4,096
13:01:42.1903739	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 53,248, Length: 4,096
13:01:42.1904355	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 57,344, Length: 4,096
13:01:42.1906706	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 61,440, Length: 4,096
13:01:42.1911495	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 65,536, Length: 4,096, Priori...
13:01:42.1938884	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 69,632, Length: 4,096, Priori...

Và các process mở key, thêm giá trị vào và đóng key

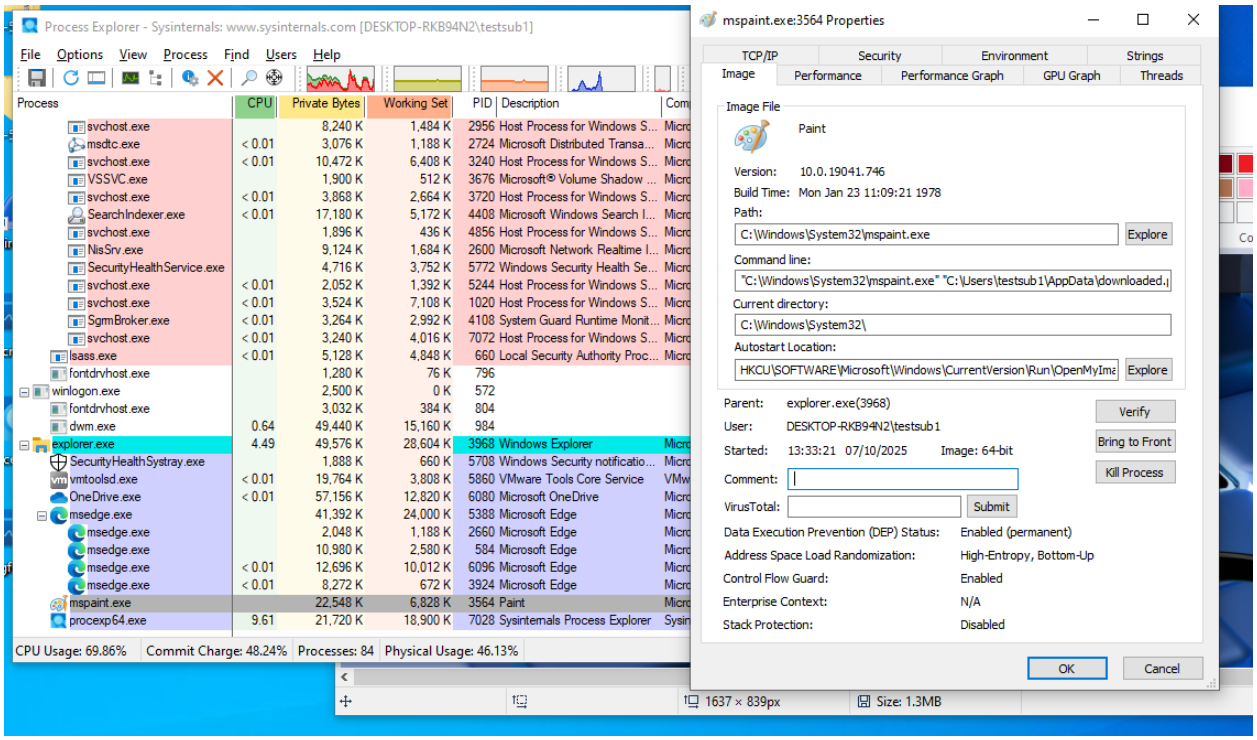


13:01:42.5482572	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 1,310,720, Length: 4,096
13:01:42.5483010	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 1,314,816, Length: 4,096
13:01:42.5484681	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 1,318,912, Length: 4,096
13:01:42.5485521	harmless_downl...	7900	WriteFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	Offset: 1,323,008, Length: 4,088
13:01:42.5485919	harmless_downl...	7900	CloseFile	C:\Users\testsub1\AppData\downloaded.png	SUCCESS	
13:01:42.5491612	harmless_downl...	7900	RegCloseKey	HKLM\SOFTWARE\Policies\Microsoft\Windows\TenantRestrictions\Payl...	SUCCESS	
13:01:42.5492048	harmless_downl...	7900	RegCloseKey	HKLM\SOFTWARE\Policies\Microsoft\Windows	SUCCESS	
13:01:42.6505892	harmless_downl...	7900	RegQueryKey	HKCU	SUCCESS	Query: HandleTags, HandleTags: 0x0
13:01:42.6506253	harmless_downl...	7900	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Run	SUCCESS	Desired Access: Set Value
13:01:42.6506749	harmless_downl...	7900	RegSetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\OpenMyIm...	SUCCESS	Type: REG_SZ, Length: 154, Data:...
13:01:42.6507808	harmless_downl...	7900	ReadFile	C:\Users\testsub1\NTUSER.DAT	SUCCESS	Offset: 1,007,616, Length: 32,768, I...
13:01:42.6521189	harmless_downl...	7900	RegCloseKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	SUCCESS	

Sau khi khởi động lại máy sẽ thấy ảnh được mở lên dùng mspaint

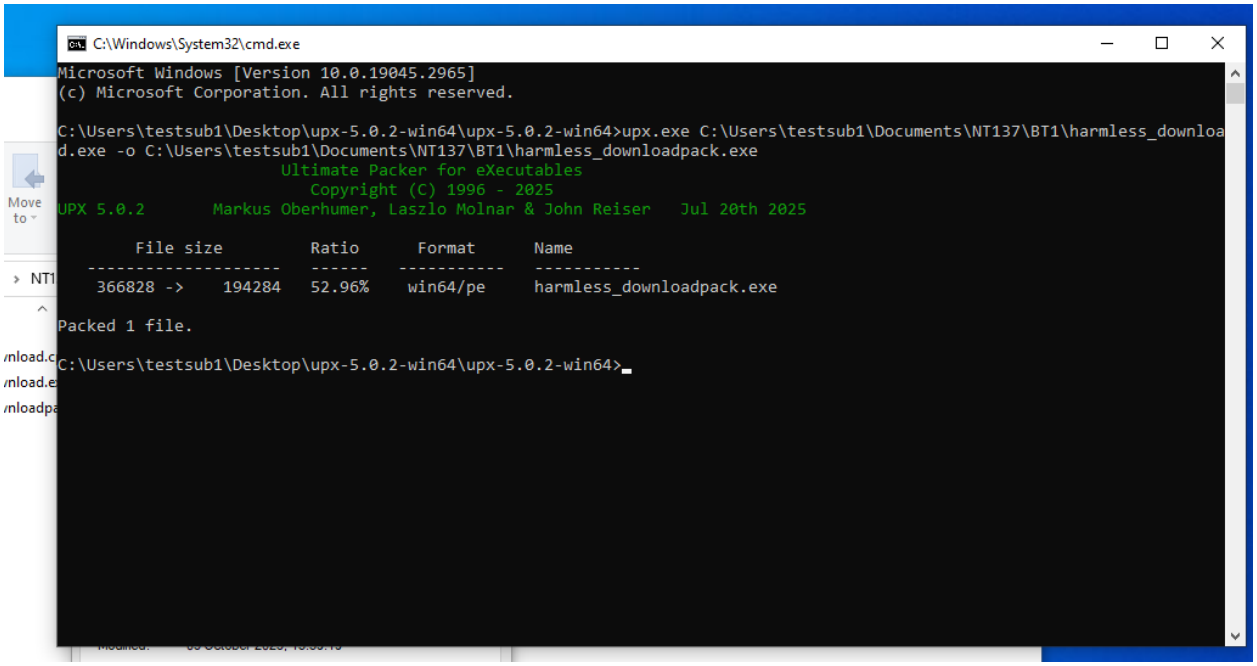


Process explore cho biết mspaint.exe đang chạy với command line là đường dẫn của mspaint.exe với đường dẫn đến file ảnh, cũng cho thấy vị trí autostart trong run key.



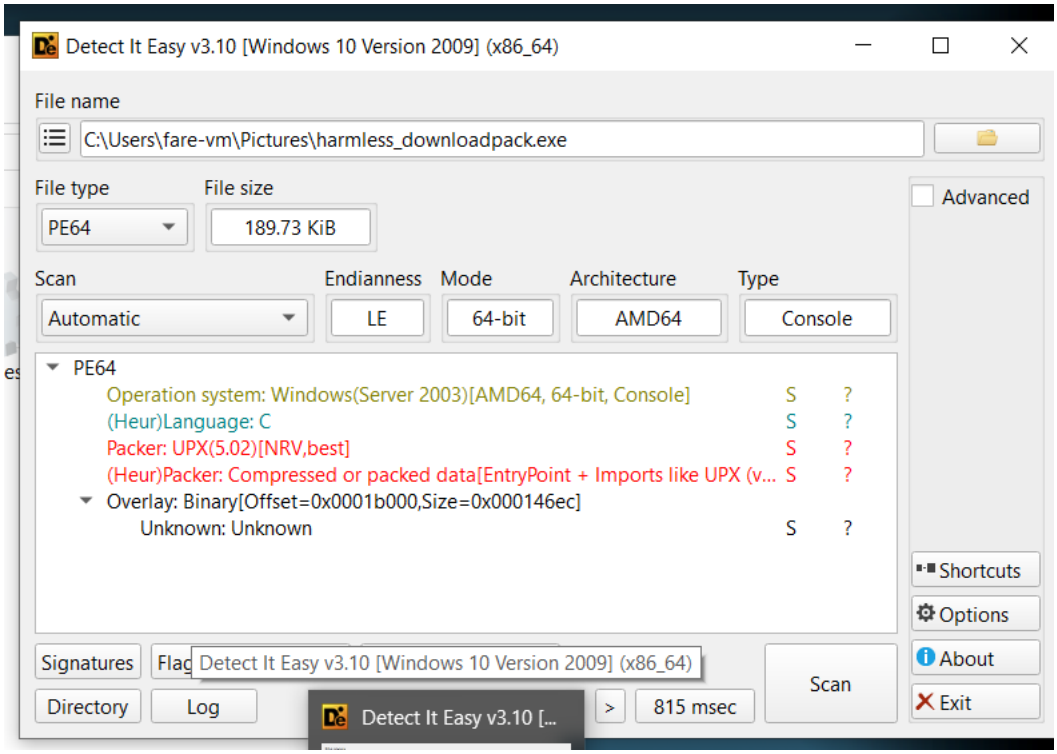
4. Packing và phân tích chương trình bị packed

Sử dụng kỹ thuật packing bằng chương trình packing pe file upx.

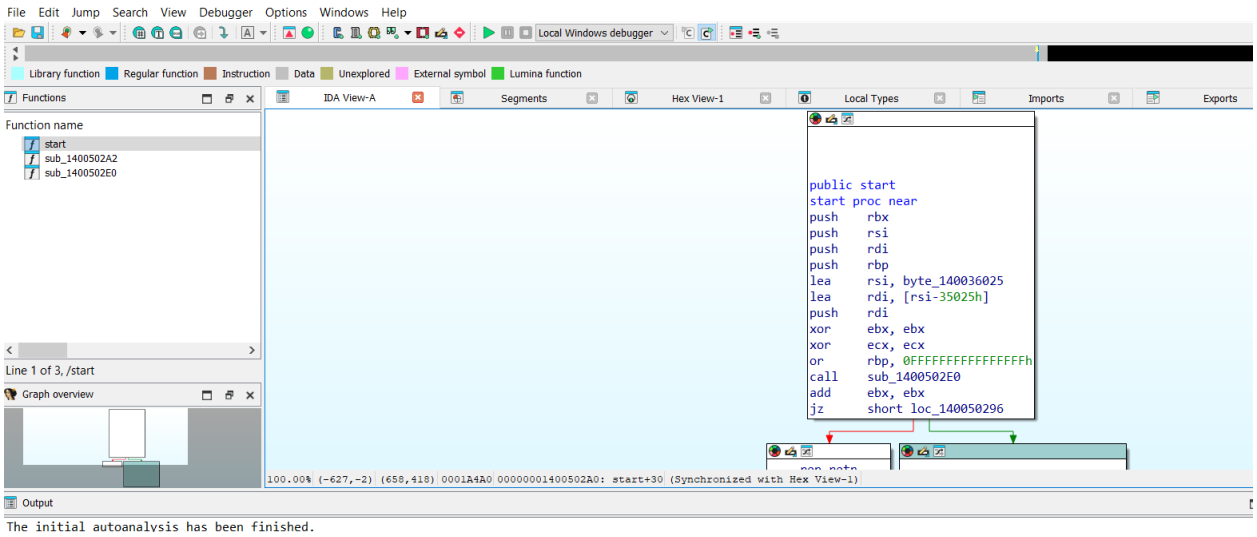


Làm theo hướng dẫn của https://hackmd.io/@antoinenguyen09/Hy0a2mb0t

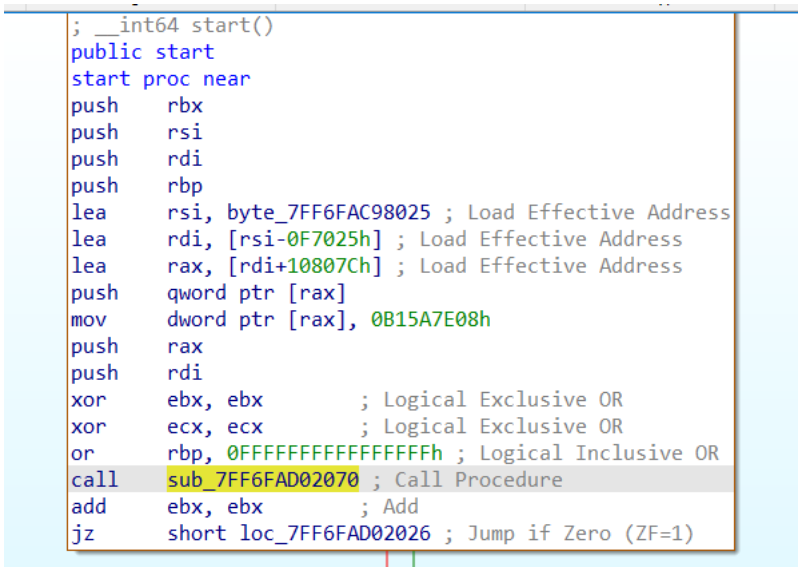
Đầu tiên kiểm tra file sử dụng phương pháp pack nào, sử dụng Detect It Easy thì biết là UPX.



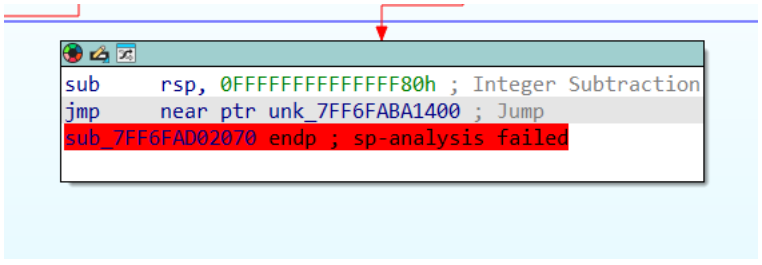
Mở IDA để phân tích



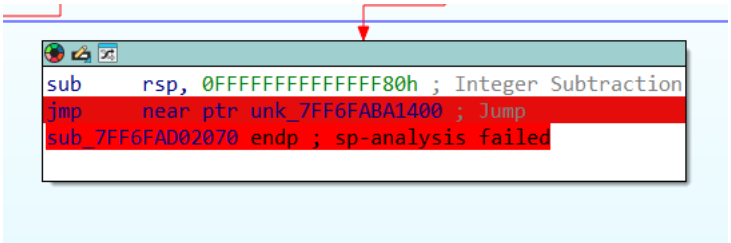
Ở hàm start thì thấy hàm nhảy đến sub_7FF6FAD02070



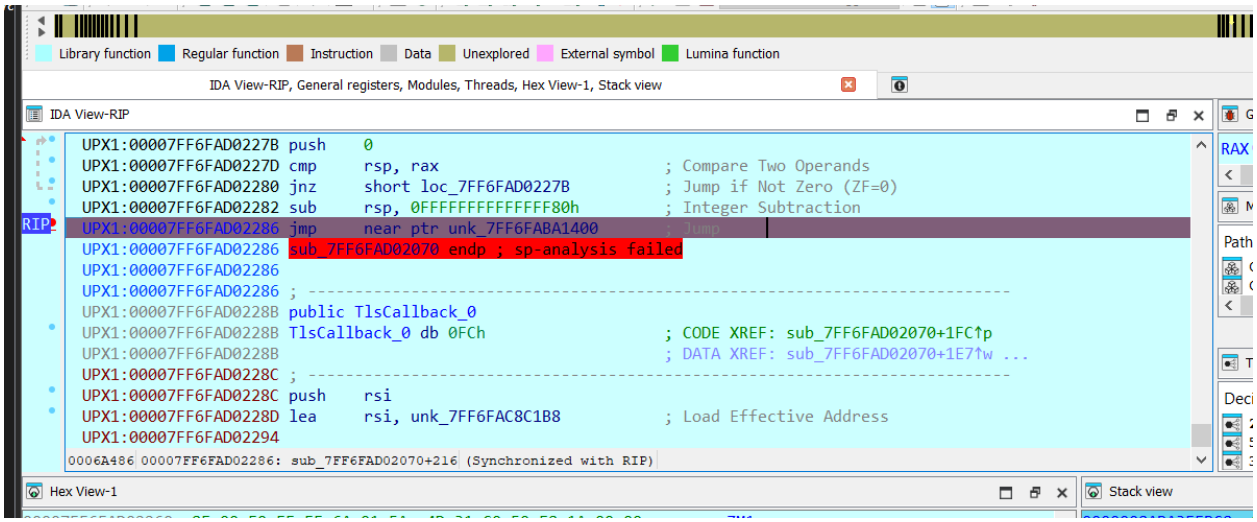
Kiểm tra sub_7FF6FAD02070, ở khúc kết thúc của stub nhảy đến một con trỏ có phần tên đầu là unk, unk có thể cho biết là địa chỉ không xác định được.



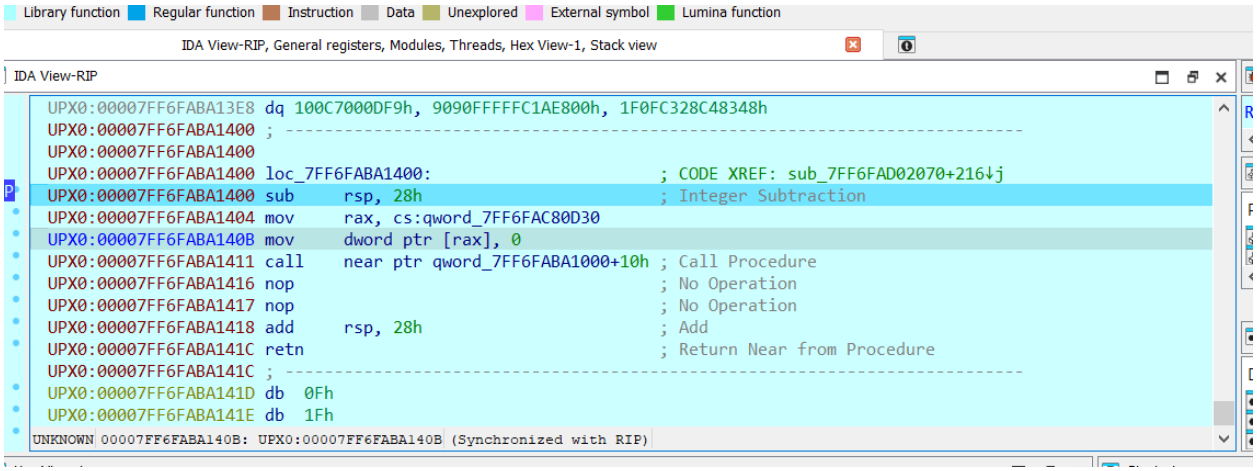
Do vậy phải phân tích động, đặt breakpoint tại vị trí con trỏ đó và bắt đầu debug



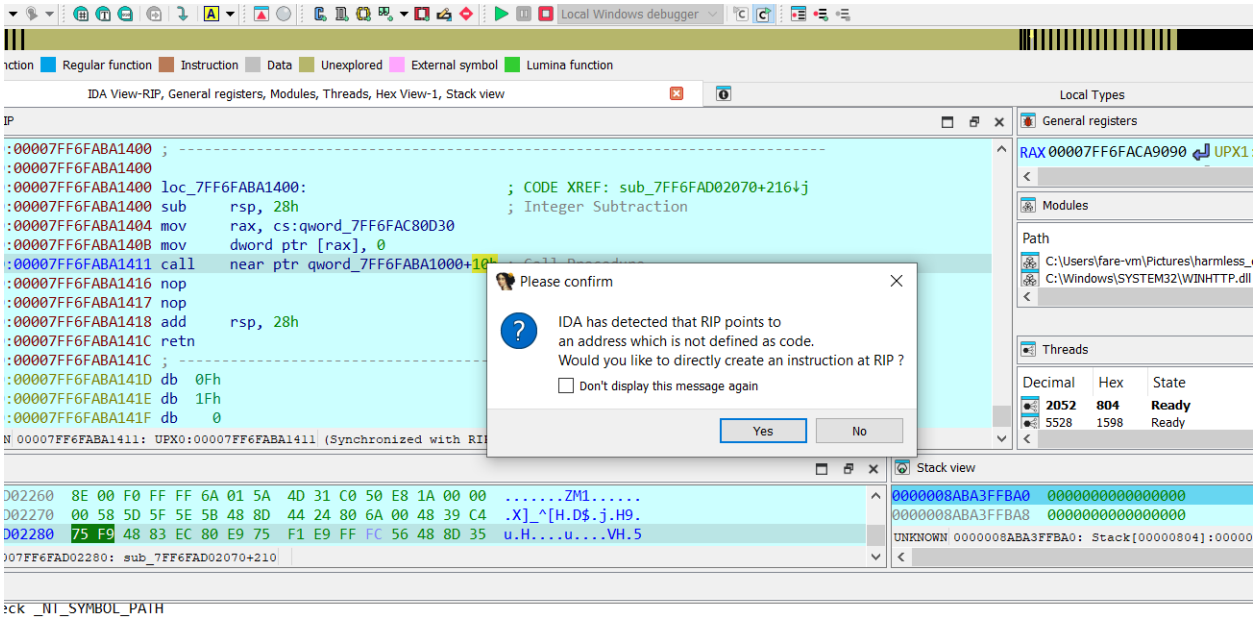
Đến breakpoint nhấn F7 để đi vào địa chỉ đó



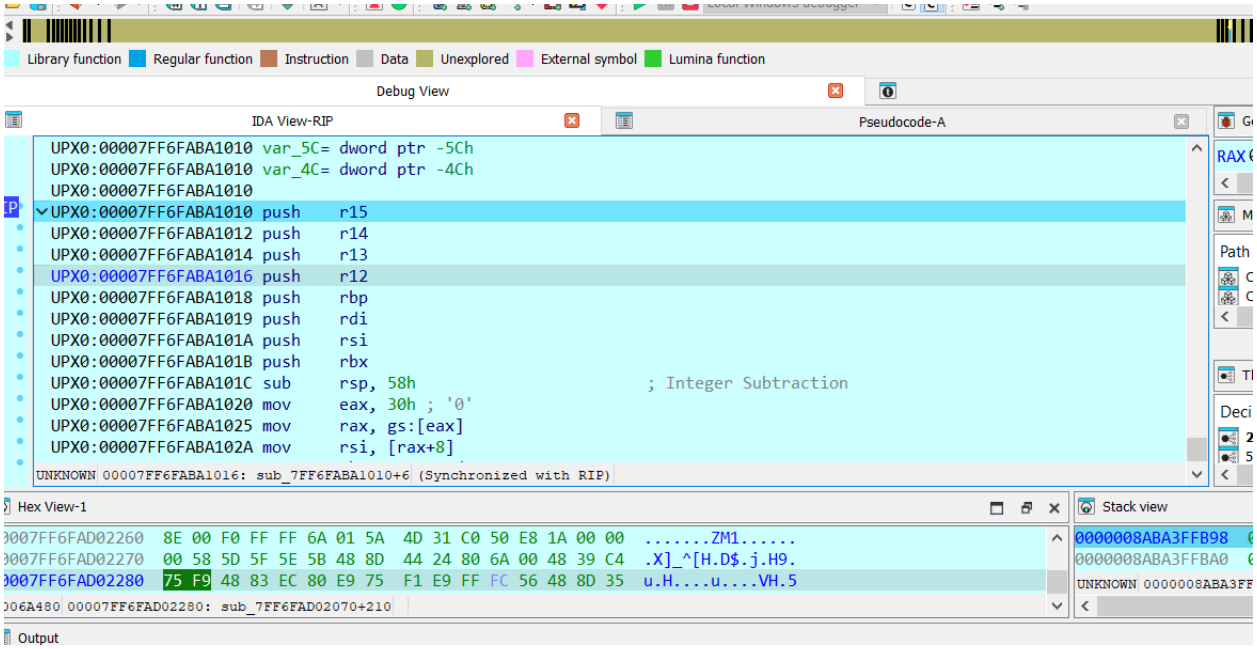
Đi đến vị trí sau



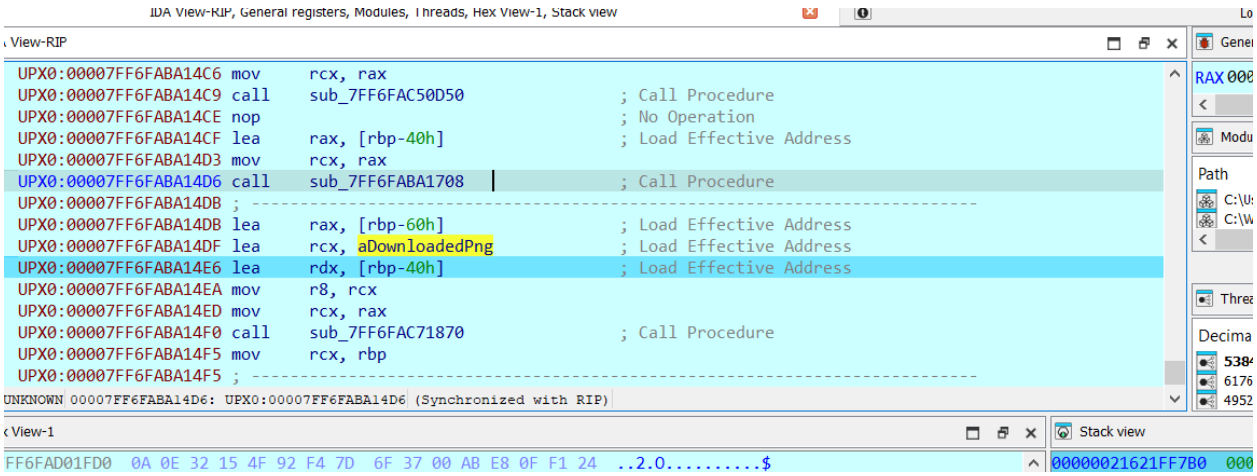
Tiếp tục nhấn F8 để đi tiếp đến lệnh call thì nhấn F7 để đi vào thì phát hiện được phần địa chỉ ida chưa phát hiện được.

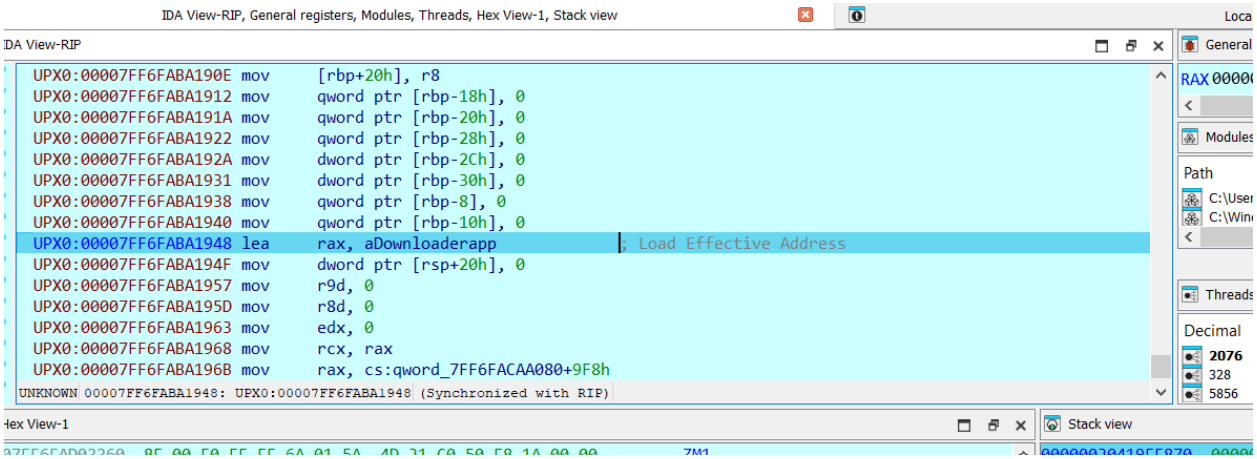


Sau khi mở rộng thêm code thì nhảy đến địa chỉ sau



Tiếp tục F8 để đi tiếp thì thấy được một số lời gọi hàm aDownloadPng, aDownloaderapp, cho biết ứng dụng tải ảnh và ảnh được tải.





Nhấn F8 đến vị trí loc_7FF6FABA18FE sẽ bắt đầu thấy được các WinAPI

WinHttpOpen được gọi trước để mở handler kết nối https

```
UPX0:00007FF6FABA1968 mov     rcx, rax
UPX0:00007FF6FABA196B mov     rax, cs:qword_7FF6FACAA080+9F8h
UPX0:00007FF6FABA1972 call    rax ; winhttp_WinHttpOpen       ; Indirect Call Near Procedure
UPX0:00007FF6FABA1974 mov     [rbp-18h], rax
UPX0:00007FF6FABA1978 mov     rax, [rbp-18h]
UPX0:00007FF6FABA197C test    rax, rax                        ; Logical Compare
UPX0:00007FF6FABA197F jnz     short loc_7FF6FABA19BC          ; Jump if Not Zero (ZF=0)
UPX0:00007FF6FABA1981 mov     rax, cs:qword_7FF6FACAA080+5A0h
UPX0:00007FF6FABA1988 call    rax ; kernel32_GetLastError     ; Indirect Call Near Procedure
UPX0:00007FF6FABA198A mov     edx, eax
UPX0:00007FF6FABA198C lea     rax, aWinhttpopenFai            ; Load Effective Address
UPX0:00007FF6FABA1993 mov     rcx, rax
UNKNOWN 00007FF6FABA1972: UPX0:00007FF6FABA1972 (Synchronized with RIP)
```

winhttp_WinHttpConnect tạo kết nối sử dụng https

```
UPX0:00007FF6FABA19CF mov     r9d, 0
UPX0:00007FF6FABA19D5 mov     r8d, 1BBh
UPX0:00007FF6FABA19DB mov     rcx, rax
UPX0:00007FF6FABA19DE mov     rax, cs:qword_7FF6FACAA080+9F0h
UPX0:00007FF6FABA19E5 call    rax ; winhttp_WinHttpConnect    ; Indirect Call Near Procedure
UPX0:00007FF6FABA19E7 mov     [rbp-20h], rax
UPX0:00007FF6FABA19EB mov     rax, [rbp-20h]
UPX0:00007FF6FABA19EF test    rax, rax                        ; Logical Compare
UPX0:00007FF6FABA19F2 jnz     short loc_7FF6FABA1A2F          ; Jump if Not Zero (ZF=0)
UPX0:00007FF6FABA19F4 mov     rax, cs:qword_7FF6FACAA080+5A0h
UPX0:00007FF6FABA19FB call    rax ; kernel32_GetLastError     ; Indirect Call Near Procedure
UPX0:00007FF6FABA19FD mov     edx, eax
UPX0:00007FF6FABA19FF lea     rax, aWinhttpconnect            ; Load Effective Address
UNKNOWN 00007FF6FABA19DE: UPX0:00007FF6FABA19DE (Synchronized with RIP)
```

winhttp_WinHttpOpenRequest mở request qua https

```
UPX0:00007FF6FABA1A69 mov     r8, rcx
UPX0:00007FF6FABA1A6C mov     rcx, rax
UPX0:00007FF6FABA1A6F mov     rax, cs:qword_7FF6FACAA080+0A00h
UPX0:00007FF6FABA1A76 call    rax ; winhttp_WinHttpOpenRequest ; Indirect Call Near Procedure
UPX0:00007FF6FABA1A78 mov     [rbp-28h], rax
UPX0:00007FF6FABA1A7C mov     rax, [rbp-28h]
UPX0:00007FF6FABA1A80 test    rax, rax                        ; Logical Compare
UPX0:00007FF6FABA1A83 jnz     short loc_7FF6FABA1AC0          ; Jump if Not Zero (ZF=0)
UPX0:00007FF6FABA1A85 mov     rax, cs:qword_7FF6FACAA080+5A0h
UPX0:00007FF6FABA1A8C call    rax ; kernel32_GetLastError     ; Indirect Call Near Procedure
UPX0:00007FF6FABA1A8E mov     edx, eax
UPX0:00007FF6FABA1A90 lea     rax, aWinhttpopenreq            ; Load Effective Address
UPX0:00007FF6FABA1A97 mov     rcx, rax
```

winhttp_WinHttpSendRequest gửi request https

```
UPX0:00007FF6FABA1AE3 mov     r8d, 0
UPX0:00007FF6FABA1AE9 mov     edx, 0
UPX0:00007FF6FABA1AEE mov     rcx, rax
UPX0:00007FF6FABA1AF1 mov     rax, cs:qword_7FF6FACAA080+0A20h
UPX0:00007FF6FABA1AF8 call    rax ; winhttp_WinHttpSendRequest ; Indirect Call Near Procedure
UPX0:00007FF6FABA1AFA test    eax, eax                        ; Logical Compare
UPX0:00007FF6FABA1AFC setz    al                              ; Set Byte if Zero (ZF=1)
UPX0:00007FF6FABA1AFF test    al, al                          ; Logical Compare
UPX0:00007FF6FABA1B01 jz      loc_7FF6FABA1BCE                ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1B07 mov     rax, cs:qword_7FF6FACAA080+5A0h
UPX0:00007FF6FABA1B0E call    rax ; kernel32_GetLastError     ; Indirect Call Near Procedure
UPX0:00007FF6FABA1B10 cmp     eax, 2F8Fh                      ; Compare Two Operands
UPX0:00007FF6FABA1B15 setz    al                              ; Set Byte if Zero (ZF=1)
UNKNOWN 00007FF6FABA1AFF: UPX0:00007FF6FABA1AFF (Synchronized with RIP)
```

winhttp_WinHttpSetOption, ở đây sẽ set lại option để cho phép self-signed

```
UPX0:00007FF6FABA1B38 mov     edx, 1Fh
UPX0:00007FF6FABA1B3D mov     rcx, rax
UPX0:00007FF6FABA1B40 mov     rax, cs:qword_7FF6FACAA080+0A28h
UPX0:00007FF6FABA1B47 call    rax ; winhttp_WinHttpSetOption  ; Indirect Call Near Procedure
UPX0:00007FF6FABA1B49 mov     rax, [rbp-28h]
UPX0:00007FF6FABA1B4D mov     qword ptr [rsp+30h], 0
UPX0:00007FF6FABA1B56 mov     dword ptr [rsp+28h], 0
UPX0:00007FF6FABA1B5E mov     dword ptr [rsp+20h], 0
```

winhttp_WinHttpSendRequest được gọi lần hai để gửi lại request

```
UPX0:00007FF6FABA1B5E mov     dword ptr [rsp+20h], 0
UPX0:00007FF6FABA1B66 mov     r9d, 0
UPX0:00007FF6FABA1B6C mov     r8d, 0
UPX0:00007FF6FABA1B72 mov     edx, 0
UPX0:00007FF6FABA1B77 mov     rcx, rax
UPX0:00007FF6FABA1B7A mov     rax, cs:qword_7FF6FACAA080+0A20h
UPX0:00007FF6FABA1B81 call    rax ; winhttp_WinHttpSendRequest ; Indirect Call Near Procedure
UPX0:00007FF6FABA1B83 test    eax, eax                         ; Logical Compare
UPX0:00007FF6FABA1B85 setz    al                               ; Set Byte if Zero (ZF=1)
UPX0:00007FF6FABA1B88 test    al, al                           ; Logical Compare
UPX0:00007FF6FABA1B8A jz      short loc_7FF6FABA1BCE           ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1B8C lea     rcx, [rbp-28h]                   ; Load Effective Address
UPX0:00007FF6FABA1B90 lea     rdx, [rbp-20h]                   ; Load Effective Address
IKNOWN 00007FF6FABA1B88: UPX0:00007FF6FABA1B88 (Synchronized with RIP)
```

winhttp_WinHttpReceiveResponse để nhận response từ máy chủ

```
UPX0:00007FF6FABA1BCE loc_7FF6FABA1BCE:                        ; CODE XREF: UPX0:00007FF6FABA1B01↑j
UPX0:00007FF6FABA1BCE                                          ; UPX0:00007FF6FABA1B8A↑j
UPX0:00007FF6FABA1BCE mov     rax, [rbp-28h]
UPX0:00007FF6FABA1BD2 mov     edx, 0
UPX0:00007FF6FABA1BD7 mov     rcx, rax
UPX0:00007FF6FABA1BDA mov     rax, cs:qword_7FF6FACAA080+0A18h
UPX0:00007FF6FABA1BE1 call    rax ; winhttp_WinHttpReceiveResponse ; Indirect Call Near Procedure
UPX0:00007FF6FABA1BE3 test    eax, eax                         ; Logical Compare
UPX0:00007FF6FABA1BE5 setz    al                               ; Set Byte if Zero (ZF=1)
UPX0:00007FF6FABA1BE8 test    al, al                           ; Logical Compare
UPX0:00007FF6FABA1BEA jz      short loc_7FF6FABA1C0D           ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1BEC lea     rcx, [rbp-28h]                   ; Load Effective Address
UPX0:00007FF6FABA1BF0 lea     rdx, [rbp-20h]                   ; Load Effective Address
UPX0:00007FF6FABA1BF4 lea     rax, [rbp-18h]                   ; Load Effective Address
UNKNOWN 00007FF6FABA1BF0: UPX0:00007FF6FABA1BF0 (Synchronized with RIP)
```

Msvcrt_wfopen để mở file để ghi dữ liệu (cho phép Unicode)

```
UPX0:00007FF6FABA1C14 call    sub_7FF6FABD71A0                 ; Call Procedure
UPX0:00007FF6FABA1C19 mov     rcx, rax
UPX0:00007FF6FABA1C1C lea     rax, aWb                         ; Load Effective Address
UPX0:00007FF6FABA1C23 mov     rdx, rax
UPX0:00007FF6FABA1C26 mov     rax, cs:qword_7FF6FACAA080+810h
UPX0:00007FF6FABA1C2D call    rax ; msvcrt__wfopen             ; Indirect Call Near Procedure
UPX0:00007FF6FABA1C2F mov     [rbp-10h], rax
UPX0:00007FF6FABA1C33 cmp     qword ptr [rbp-10h], 0           ; Compare Two Operands
UPX0:00007FF6FABA1C38 jnz     short loc_7FF6FABA1C5B           ; Jump if Not Zero (ZF=0)
UPX0:00007FF6FABA1C3A lea     rcx, [rbp-28h]                   ; Load Effective Address
UPX0:00007FF6FABA1C3E lea     rdx, [rbp-20h]                   ; Load Effective Address
UPX0:00007FF6FABA1C42 lea     rax, [rbp-18h]                   ; Load Effective Address
UPX0:00007FF6FABA1C46 mov     r8, rcx
UPX0:00007FF6FABA1C49 mov     rcx, rax
UNKNOWN 00007FF6FABA1C2F: UPX0:00007FF6FABA1C2F (Synchronized with RIP)
```

winhttp_WinHttpQueryDataAvailable để kiểm tra có dữ liệu không.

```
UPX0:00007FF6FABA1C5B
UPX0:00007FF6FABA1C5B loc_7FF6FABA1C5B:                    ; CODE XREF: UPX0:00007FF6FABA1C38↑j
UPX0:00007FF6FABA1C5B                                      ; UPX0:00007FF6FABA1D72↓j
UPX0:00007FF6FABA1C5B mov     rax, [rbp-28h]
UPX0:00007FF6FABA1C5F lea     rdx, [rbp-2Ch]               ; Load Effective Address
UPX0:00007FF6FABA1C63 mov     rcx, rax
UPX0:00007FF6FABA1C66 mov     rax, cs:qword_7FF6FACAA080+0A08h
UPX0:00007FF6FABA1C6D call    rax ; winhttp_WinHttpQueryDataAvailable ; Indirect Call Near Procedure
UPX0:00007FF6FABA1C6F test    eax, eax                     ; Logical Compare
UPX0:00007FF6FABA1C71 setz    al                           ; Set Byte if Zero (ZF=1)
UPX0:00007FF6FABA1C74 test    al, al                       ; Logical Compare
UPX0:00007FF6FABA1C76 jz      short loc_7FF6FABA1C97       ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1C78 mov     rax, cs:qword_7FF6FACAA080+5A0h
UPX0:00007FF6FABA1C7F call    rax ; kernel32_GetLastError  ; Indirect Call Near Procedure
```

winhttp_WinHttpReadData để đọc dữ liệu nhận được.

```
UPX0:00007FF6FABA1D02 mov     rcx, rax
UPX0:00007FF6FABA1D05 mov     rax, cs:qword_7FF6FACAA080+0A10h
UPX0:00007FF6FABA1D0C call    rax ; winhttp_WinHttpReadData    ; Indirect Call Near Procedure
UPX0:00007FF6FABA1D0E test    eax, eax                         ; Logical Compare
UPX0:00007FF6FABA1D10 setz    al                               ; Set Byte if Zero (ZF=1)
UPX0:00007FF6FABA1D13 test    al, al                           ; Logical Compare
UPX0:00007FF6FABA1D15 jz      short loc_7FF6FABA1D33           ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1D17 mov     rax, cs:qword_7FF6FACAA080+5A0h
UPX0:00007FF6FABA1D1E call    rax ; kernel32_GetLastError      ; Indirect Call Near Procedure
UPX0:00007FF6FABA1D20 mov     edx, eax
UPX0:00007FF6FABA1D22 lea     rax, aErrorUInWinhtt_0           ; Load Effective Address
UNKNOWN 00007FF6FABA1D17: UPX0:00007FF6FABA1D17 (Synchronized with RIP)
```

GetSystemDirectoryW để lấy thư mục system của người dùng hiện tại

```
UPX0:00007FF6FABA1DD8 lea     rax, [rbp+2C0h+var_2F0]          ; Load Effective Address
UPX0:00007FF6FABA1DDC mov     edx, 104h
UPX0:00007FF6FABA1DE1 mov     rcx, rax
UPX0:00007FF6FABA1DE4 mov     rax, cs:qword_7FF6FACAA080+5C0h
UPX0:00007FF6FABA1DEB call    rax ; kernel32_GetSystemDirectoryW ; Indirect Call Near Procedure
UPX0:00007FF6FABA1DED mov     [rbp+2C0h+var_14], eax
UPX0:00007FF6FABA1DF3 cmp     [rbp+2C0h+var_14], 0             ; Compare Two Operands
UPX0:00007FF6FABA1DFA jz      short loc_7FF6FABA1E08           ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1DFC cmp     [rbp+2C0h+var_14], 103h          ; Compare Two Operands
UPX0:00007FF6FABA1E06 jbe     short loc_7FF6FABA1E12           ; Jump if Below or Equal (CF=1 | ZF
UPX0:00007FF6FABA1E08
UPX0:00007FF6FABA1E08 loc_7FF6FABA1E08:                        ; CODE XREF: sub_7FF6FABA1DA9+51↑j
```

RegOpenKeyExW để mở key trong registry

```
IDA View-RIP
UPX0:00007FF6FABA1F10 mov     r9d, 2
UPX0:00007FF6FABA1F16 mov     r8d, 0
UPX0:00007FF6FABA1F1C mov     rdx, rax
UPX0:00007FF6FABA1F1F mov     rcx, 0FFFFFFFF80000001h
UPX0:00007FF6FABA1F26 mov     rax, cs:qword_7FF6FACAA080+528h
UPX0:00007FF6FABA1F2D call    rax ; advapi32_RegOpenKeyExW    ; Indirect Call Near Procedure
UPX0:00007FF6FABA1F2F mov     [rbp+2C0h+var_18], eax
UPX0:00007FF6FABA1F35 cmp     [rbp+2C0h+var_18], 0            ; Compare Two Operands
UPX0:00007FF6FABA1F3C jz      short loc_7FF6FABA1FAB          ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1F3E lea     rax, aSoftwareMicros            ; Load Effective Address
UPX0:00007FF6FABA1F45 mov     [rsp+340h+var_300], 0
UPX0:00007FF6FABA1F4E lea     rdx, [rbp+2C0h+var_E8]          ; Load Effective Address
UPX0:00007FF6FABA1F55 mov     [rsp+340h+var_308], rdx
UPX0:00007FF6FABA1F5A mov     [rsp+340h+var_310], 0
UNKNOWN 00007FF6FABA1F45: sub_7FF6FABA1DA9+19C (Synchronized with RIP)
```

Advapi32_RegCreateKeyExW để tạo key

```
UPX0:00007FF6FABA1F6B mov     dword ptr [rsp+340h+var_320], 0
UPX0:00007FF6FABA1F73 mov     r9d, 0
UPX0:00007FF6FABA1F79 mov     r8d, 0
UPX0:00007FF6FABA1F7F mov     rdx, rax
UPX0:00007FF6FABA1F82 mov     rcx, 0FFFFFFFF80000001h
UPX0:00007FF6FABA1F89 mov     rax, cs:qword_7FF6FACAA080+520h
UPX0:00007FF6FABA1F90 call    rax ; advapi32_RegCreateKeyExW  ; Indirect Call Near Procedure
UPX0:00007FF6FABA1F92 mov     [rbp+2C0h+var_18], eax
UPX0:00007FF6FABA1F98 cmp     [rbp+2C0h+var_18], 0            ; Compare Two Operands
UPX0:00007FF6FABA1F9F jz      short loc_7FF6FABA1FAB          ; Jump if Zero (ZF=1)
UPX0:00007FF6FABA1FA1 mov     ebx, 0
UPX0:00007FF6FABA1FA6 jmp     loc_7FF6FABA203B                ; Jump
UPX0:00007FF6FABA1FAB ; ---------------------------------------------------------------------------
UPX0:00007FF6FABA1FAB
```

Advapi32_RegSetValueExW để thêm đường dẫn thực thi mspaint.exe để mở ảnh.

```
UPX0:00007FF6FABA1FDD call    sub_7FF6FABD71A0                ; Call Procedure
UPX0:00007FF6FABA1FE2 mov     rdx, rax
UPX0:00007FF6FABA1FE5 mov     rax, [rbp+2C0h+var_E8]
UPX0:00007FF6FABA1FEC mov     [rsp+340h+var_318], esi
UPX0:00007FF6FABA1FF0 mov     [rsp+340h+var_320], rbx
UPX0:00007FF6FABA1FF5 mov     r9d, 1
UPX0:00007FF6FABA1FFB mov     r8d, 0
UPX0:00007FF6FABA2001 mov     rcx, rax
UPX0:00007FF6FABA2004 mov     rax, cs:qword_7FF6FACAA080+530h
UPX0:00007FF6FABA200B call    rax ; advapi32_RegSetValueExW   ; Indirect Call Near Procedure
UPX0:00007FF6FABA200D mov     [rbp+2C0h+var_18], eax
UPX0:00007FF6FABA2013 mov     rax, [rbp+2C0h+var_E8]
UPX0:00007FF6FABA201A mov     rcx, rax
UPX0:00007FF6FABA201D mov     rax, cs:qword_7FF6FACAA080+518h
```

Từ các lệnh được tìm thấy trên có thể kết luận được là chương trình được packed là chương trình tải ảnh và tạo persistent trên.

HẾT