**Câu 2: Có 2 loại payload trên Metasploit Framework là Staged và Non-Staged. Hãy tạo ra reverse shell cho từng loại, và so sánh sự khác biệt giữa chúng.**

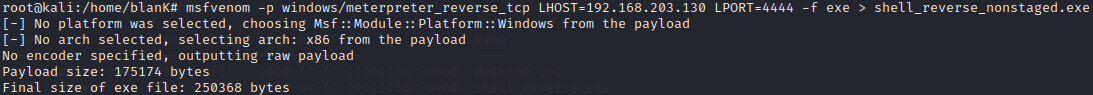
Trên Metasploit Framework có rất nhiều loại payload khác nhau, nhưng tất cả đều được phân loại theo 2 nhóm chính đó là Staged và Non-Staged.

Non-Staged payloads:

\_ Gửi tất cả mọi thứ trong một gói tin exploit.

\_ Kích thước lớn hơn Staged payloads và không chắc chắn thành công.

\_ Ví dụ: windows/meterpreter\_reverse\_tcp.

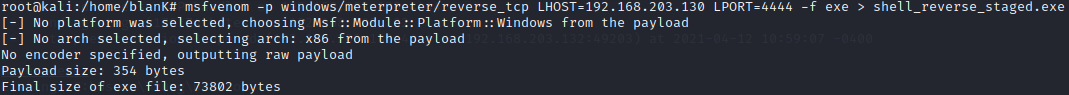


Staged payloads:

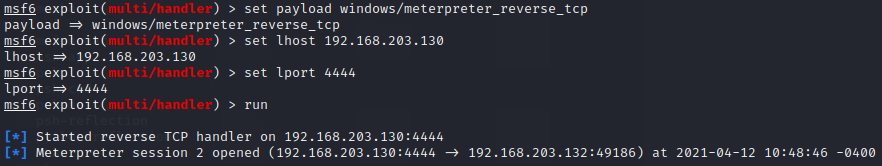
\_ Gửi các payloads theo từng giai đoạn. Sau khi kết nối thành công tới máy attacker thì payload mới được sinh ra và chạy.

\_ Kích thước nhỏ hơn nhưng có thể ít ổn định hơn.

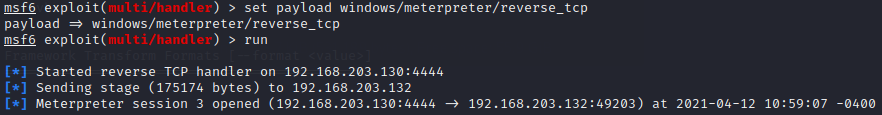
\_ Ví dụ: windows/meterpreter/reverse\_tcp.



Để có thể lắng nghe được 2 loại payload này, ta sẽ dùng chung một công cụ là MSFconsole nhưng với 2 handler khác nhau ứng với 2 lại payload khác nhau.



*Non-Staged*



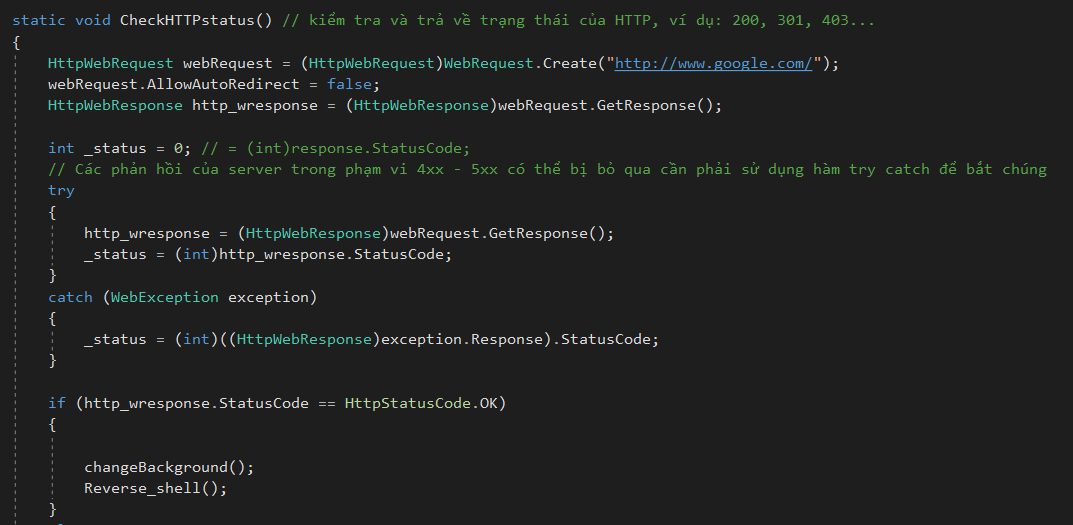
*Staged*

Về khả năng bị phát hiện bởi Anti-Virus thì nhìn chung, do Metasploit là một dạng mã nguồn mở nên các payload tạo bởi công cụ này đều sẽ bị phát hiện. Nhưng nếu xét về mặt lí thuyết của 2 loại payload thì Staged payloads sẽ khó bị phát hiện hơn. Staged payloads sử dụng một thứ gọi là Stager để tải payload vào từng giai đoạn exploit, nên kích thước sẽ giảm xuống và nếu như Stager không bị bắt thì stage đó sẽ an toàn.

**Câu 3: Viết một virus máy tính bằng ngôn ngữ lập trình C# có chức năng sau:**

**a. Thay đổi hình nền của máy nạn nhân.**

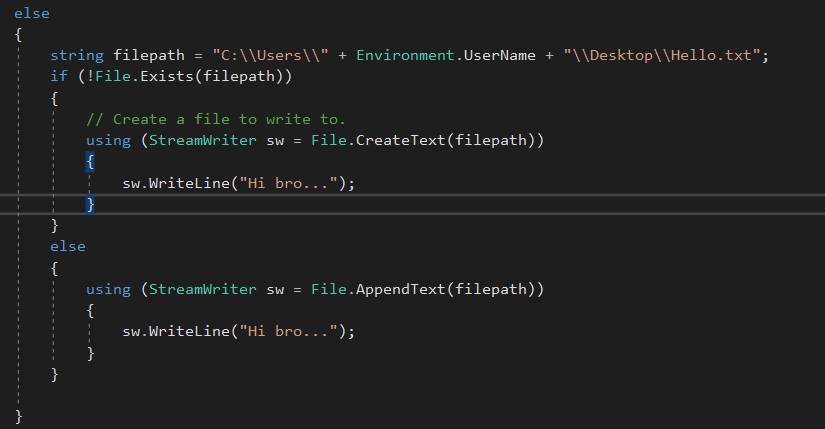
**b. Kiểm tra máy nạn nhân có kết nối Internet hay không. Nếu có, tải và thực thi reverse shell để kết nối ngược về máy của kẻ tấn công. Và ngược lại, nếu máy nạn nhân không được kết nối Internet, tạo 1 tập tin (thư mục) bất kỳ trên Desktop của nạn nhân với nội dung tùy chọn.**



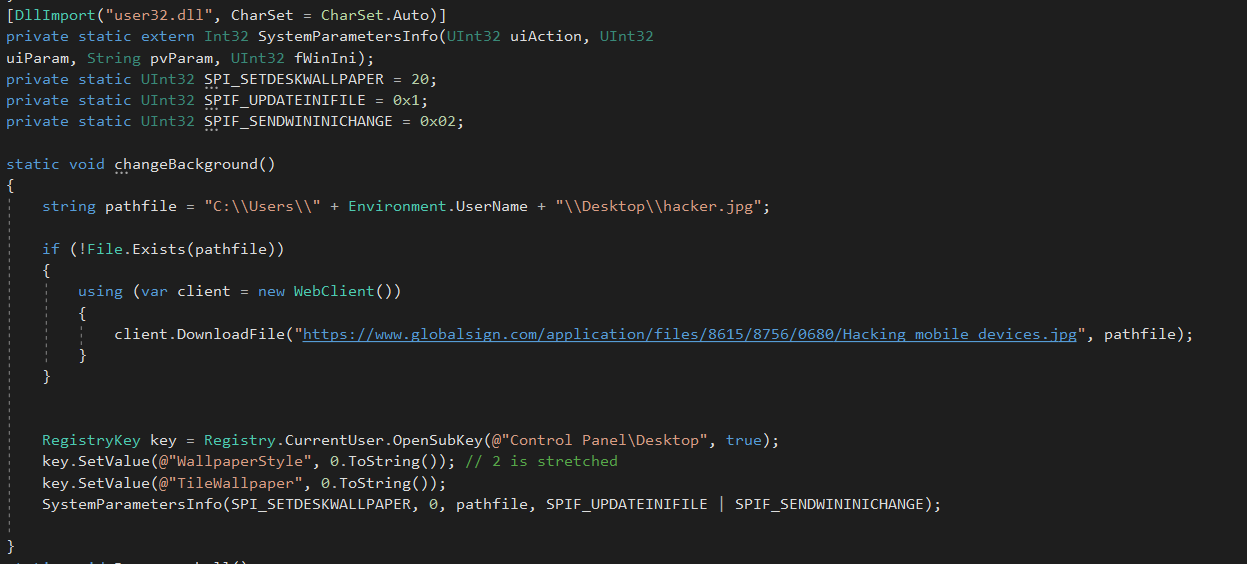
*Hình 3.1 Kiểm tra tình trạng internet.*

Sử dụng GetResponse() để lấy trạng thái trang web, trả về trạng thái 200 tức máy tính hiện tại có thể truy cập internet.

Khi có internet sẽ tiến hành thay đổi hình nền và thực hiện reverse shell.Ngược lại sẽ tạo file “Hello.txt” ở desktop .



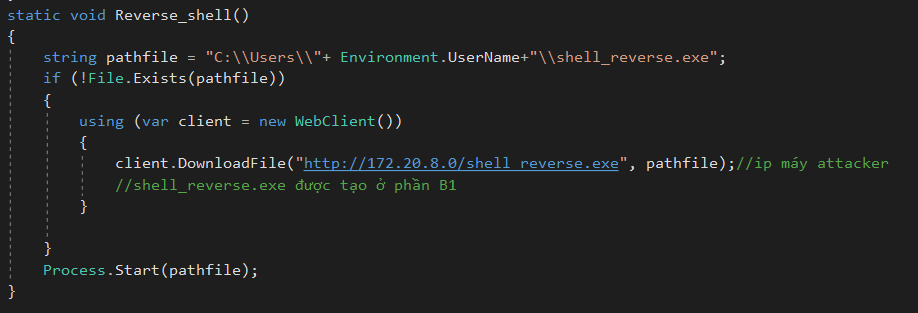
*Hình 3.2 Tạo file ở Desktop.*



*Hình 3.3 Thay đổi hình nền.*

Kiểm tra file ảnh đã tồn tại trong máy hay chưa, nếu chưa thì tiến hành tải file.

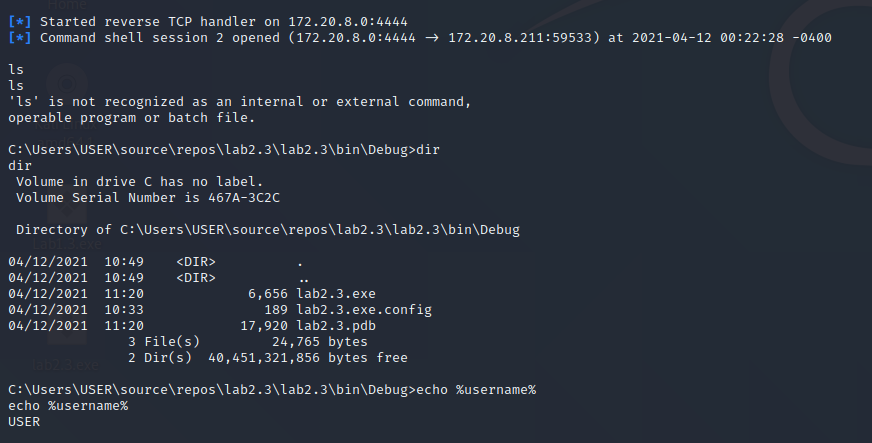
Tiến hành cập nhật Registry để chọn “Fit style” cho hình nền. Sử dụng lệnh “SystemParametersInfo” để thay đổi hình nền.



*Hình 3.4 Thực hiện Reverse shell.*

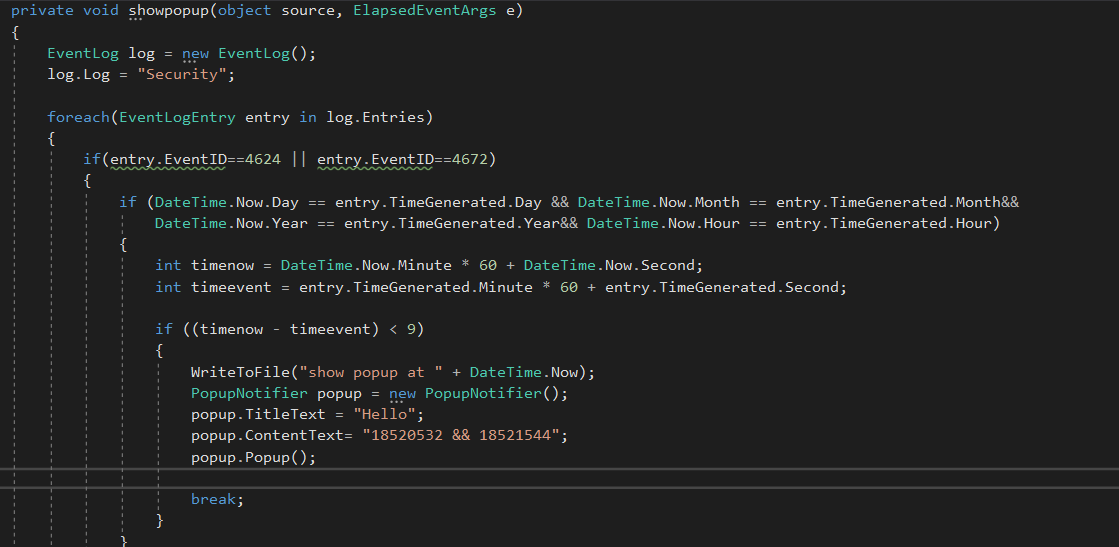
Kiểm tra file shell\_reverse.exe đã tồn tại hay chưa, nếu chưa thì tải file về ( file này đã được tạo ở phần trước).

Sau khi đã có file tiến hành thực thi file để kết nối đến máy attacker.



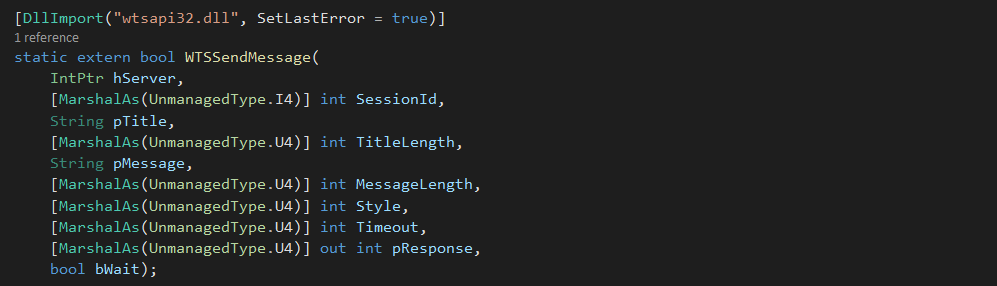
*Hình 3.5 Kết quả.*

**Câu 4: Viết một ứng virus đơn giản bằng dịch vụ trên C#, hiện pop-up MSSV trên máy nạn nhân mỗi khi user thực hiện đăng nhập thành công.**



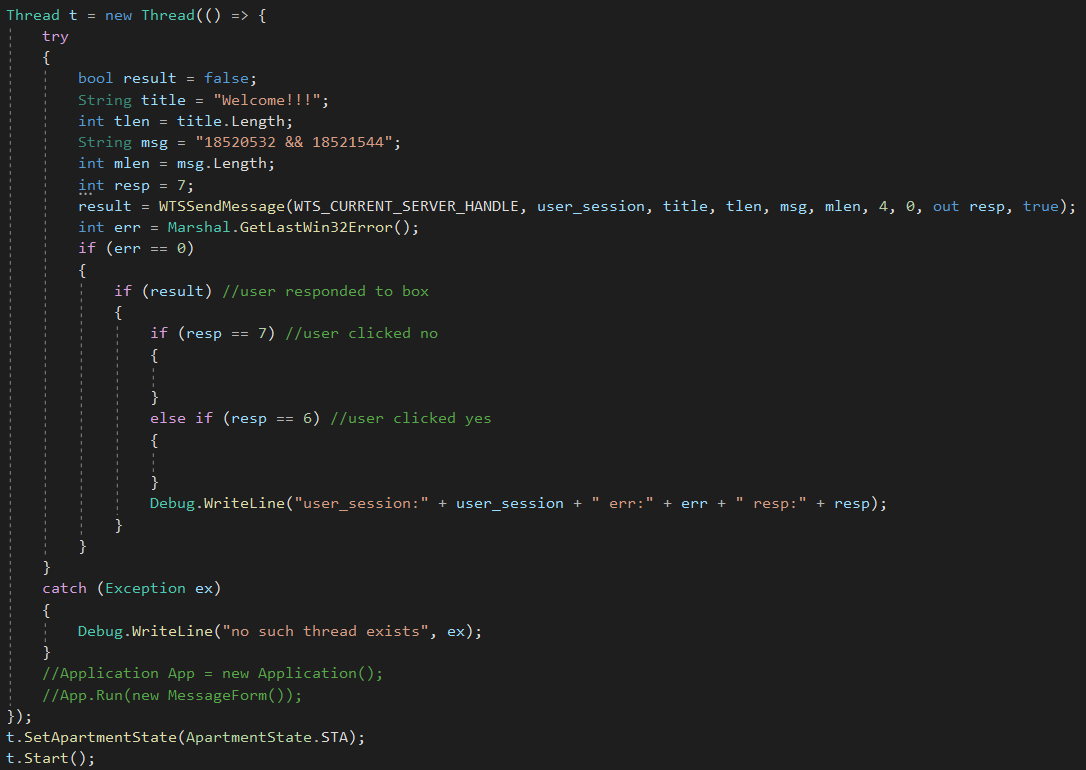
*Hình 4.1 Kiểm tra Event logon*

Để xác định người dùng logon thành công cần kiểm tra trong Eventlog với ID của event này là 4624 hoặc 4672. Vì chỉ xuất hiện popup 1 lần nên ta cần kiểm tra lần logon lần cuối cùng có phải vừa logon thành công hay không.



*Hình 4.2 Khai vào hàm WTSSendMessage*

Từ khi Windows XP, Windows Service đã không còn có thể tương tác trực tiếp với UI được nữa nên ta sẽ gọi một hàm WTSSendMessage để gián tiếp chuyển dữ liệu đến user để user hiện thông báo lên.



*Hình 4.3 Gọi hàm để popup thông báo*

Ta sẽ dùng thread để liệt kê tất cả các user đang chạy thông qua user session và dùng user đó để popup thông báo.

**Câu 5: So sánh giữa việc viết virus bằng dịch vụ trên C# so với việc tạo bằng MSF (quyền, khả năng phát hiện, …)**

Viết bằng dịch vụ trên C#:

\_ Viết được nhiều loại virus hơn tùy vào mục đích của người viết.

\_ Khả năng bị phát hiện thấp hơn do có thể chỉnh sửa payload để bypass antivirus.

\_ Phải có kiến thức sâu rộng về ngôn ngữ mà mình đang viết.

Tạo bằng MSF:

\_ Viết được các loại payload có sẵn.

\_ Khả năng bị phát hiện cao hơn do là open source.

\_ Không cần có quá nhiều kiến thức, chỉ cần biết cú pháp và cách sử dụng MSF.