



BÁO CÁO ĐỒ ÁN 3

KIẾN TRÚC MÁY TÍNH VÀ HỢP NGỮ

Crack phần mềm – Đề 3

1712153 – 1712434 – 1712632 – 1712698

Đề = $((3+4+2+8) \bmod 3) + 1 = 3$

Khoa CNTT

Trường ĐH Khoa học Tự Nhiên

NỘI DUNG BÁO CÁO:

1. Thông Tin Thành viên nhóm
2. Phân công công việc
3. Môi trường lập trình
4. Quá trình thực hiện
5. Mức độ hoàn thành
6. Nguồn tài liệu tham khảo

CHI TIẾT NỘI DUNG BÁO CÁO :

I. THÔNG TIN THÀNH VIÊN NHÓM

STT	Họ và Tên	MSSV
1	Châu Thiên Thanh	1712153
2	Lê Thanh hiếu	1712434
3	Huỳnh Lê Minh nhật	1712632
4	Võ Văn Quân	1712698

II. PHÂN CÔNG CÔNG VIỆC

STT	Thành viên	Công việc	Ghi chú
1	Châu Thiên Thanh	Tổng hợp và viết báo cáo	Các Thành viên hỗ trợ nhau trong quá trình thực hiện đồ án
2	Lê Thanh Hiếu	Crack file 3_1	
3	Huỳnh Lê Minh Nhật	Crack file 3_3	
4	Võ Văn Quân	Crack file 3_2	

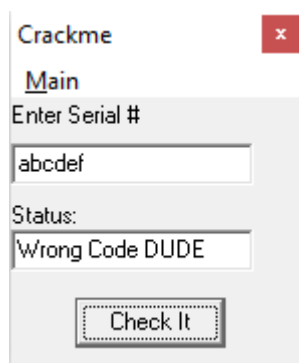
III. MÔI TRƯỜNG LẬP TRÌNH

Ollydbg, Visual Studio 2017

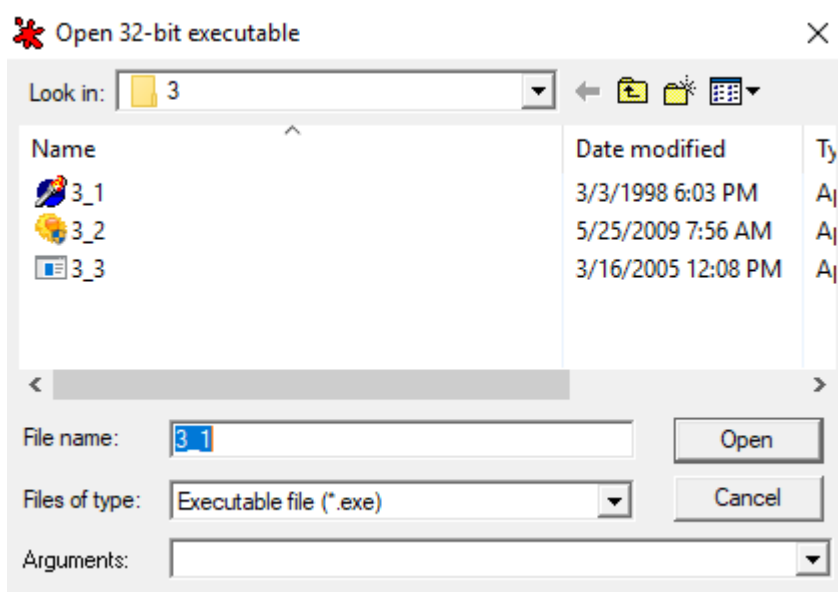
IV. QUÁ TRÌNH THỰC HIỆN

1. Bài 3_1 :

- Đầu tiên, mở crackme lên sau đó điền một fake serial vào. Ta tìm được **badboy** là “Wrong Code DUDE”



- Mở Ollydbg lên, load target 3_1



- Nhấn Open và đợi Ollydbg load xong.

00420721	00	DB 00	
00420722	00	DB 00	
00420723	00	DB 00	
00420724	18064200	DD 3_1.00420618	
00420728	55	PUSH EBP	
00420729	8BEC	MOV EBP,ESP	
0042072B	83C4 F4	ADD ESP,-0C	
0042072E	8B 40064200	MOV EAX,3_1.00420640	
00420733	E8 2078FDFF	CALL 3_1.00404F58	
00420738	A1 10EA4200	MOV EAX,DWORD PTR DS:[42EA10]	
0042073D	3E00	MOV EAX,DWORD PTR DS:[EAX]	
0042073F	E8 BC0FFFFF	CALL 3_1.00428800	
00420744	8B0D 88EA4200	MOV ECX,DWORD PTR DS:[42EA88]	3_1.0042F748
0042074A	A1 10EA4200	MOV EAX,DWORD PTR DS:[42EA10]	
0042074F	8B00	MOV EAX,DWORD PTR DS:[EAX]	
00420751	8B15 5CD34200	MOV EDX,DWORD PTR DS:[42D35C]	3_1.0042D39C
00420757	E8 BC0FFFFF	CALL 3_1.00428818	
0042075C	A1 10EA4200	MOV EAX,DWORD PTR DS:[42EA10]	
00420761	8B00	MOV EAX,DWORD PTR DS:[EAX]	
00420763	E8 3CB1FFFF	CALL 3_1.004288A4	
00420768	E8 A75CFDFF	CALL 3_1.00408414	
0042076D	8D40 00	LEA EAX,DWORD PTR DS:[EAX]	
00420770	0000	ADD BYTE PTR DS:[EAX],AL	
00420772	0000	ADD BYTE PTR DS:[EAX],AL	
00420774	0000	ADD BYTE PTR DS:[EAX],AL	
00420776	0000	ADD BYTE PTR DS:[EAX],AL	
00420778	0000	ADD BYTE PTR DS:[EAX],AL	
0042077A	0000	ADD BYTE PTR DS:[EAX],AL	
0042077C	0000	ADD BYTE PTR DS:[EAX],AL	
0042077E	0000	ADD BYTE PTR DS:[EAX],AL	
00420780	0000	ADD BYTE PTR DS:[EAX],AL	
00420782	0000	ADD BYTE PTR DS:[EAX],AL	
00420784	0000	ADD BYTE PTR DS:[EAX],AL	
00420786	0000	ADD BYTE PTR DS:[EAX],AL	
00420788	0000	ADD BYTE PTR DS:[EAX],AL	
0042078A	0000	ADD BYTE PTR DS:[EAX],AL	
0042078C	0000	ADD BYTE PTR DS:[EAX],AL	
0042078E	0000	ADD BYTE PTR DS:[EAX],AL	
00420790	0000	ADD BYTE PTR DS:[EAX],AL	
00420792	0000	ADD BYTE PTR DS:[EAX],AL	
00420794	0000	ADD BYTE PTR DS:[EAX],AL	
00420796	0000	ADD BYTE PTR DS:[EAX],AL	
00420798	0000	ADD BYTE PTR DS:[EAX],AL	

- Chuột phải vào cửa sổ CPU -> Search for -> All referenced text strings. Ta được

[illegible]

- Chú ý của số Text string, ta tìm được một số đoạn ASCII đáng nghi sau, trong đó có cả **badboy** mà ta đã tìm được.

0042D4F6	ASCII "TForm1"	
0042D507	ASCII "crackme"	
0042D537	MOV EDX, 3_1.0042D590	ASCII "Benadryl"
0042D543	MOV EDX, 3_1.0042D5A4	ASCII "Wrong Code DUDE"
0042D555	MOV EDX, 3_1.0042D5BC	ASCII "Thanks you made it"
0042D590	ASCII "Benadryl", 0	
0042D5A4	ASCII "Wrong Code DUDE", 0	
0042D5BC	ASCII "Thanks you made "	
0042D5CC	ASCII "it", 0	
0042D757	CALL 3_1.00428818	(Initial CPU selection)

- Ta tìm được **goodboy** chính là đoạn “Thanks you made it”. Và một đoạn text đáng nghi khác là “Benadryl”.
- Nhấn vào “Wrong Code DUDE” để tìm kiếm manh mối.

0042D510	. 55	PUSH EBP	
0042D511	. 8BEC	MOV EBP,ESP	
0042D513	. 6A 00	PUSH 0	
0042D515	. 53	PUSH EBX	
0042D516	. 8B08	MOV EBX,EAX	
0042D518	. 33C0	XOR EAX,EAX	
0042D51A	. 55	PUSH EBP	
0042D51B	. 68 7B054200	PUSH 3_1.0042D57B	
0042D520	. 64:FF30	PUSH DWORD PTR FS:[EAX]	
0042D523	. 64:8920	MOV DWORD PTR FS:[EAX],ESP	
0042D526	. 8D55 FC	LEA EDX,DWORD PTR SS:[EBP-4]	
0042D529	. 8B83 DC010000	MOV EAX,DWORD PTR DS:[EBX+10C]	
0042D52F	. E8 54CCFEFF	CALL 3_1.0041A188	
0042D534	. 8B45 FC	MOV EAX,DWORD PTR SS:[EBP-4]	
0042D537	. BA 20D54200	MOV EDX,3_1.0042D590	ASCII "Benadryl"
0042D53C	. E8 8F63FDFF	CALL 3_1.004038D0	
0042D541	. >74 12	JE SHORT 3_1.0042D555	
0042D543	. BA A4D54200	MOV EDX,3_1.0042D5A4	ASCII "Wrong Code DUDE"
0042D548	. 8B83 E8010000	MOV EAX,DWORD PTR DS:[EBX+1E8]	
0042D54E	. E8 65CCFEFF	CALL 3_1.0041A188	
0042D553	. >EB 10	JMP SHORT 3_1.0042D565	
0042D555	. BA BCD54200	MOV EDX,3_1.0042D5BC	ASCII "Thanks you made it"
0042D55A	. 8B83 E8010000	MOV EAX,DWORD PTR DS:[EBX+1E8]	
0042D560	. E8 53CCFEFF	CALL 3_1.0041A188	

- Chú ý quan sát, ta thấy trước **badboy** có một lệnh nhảy JE đến địa chỉ 0042D555, tìm xuống phía dưới ta nhận thấy 0042D555 chính là địa chỉ của **goodboy**.
- Chú ý dòng lệnh **0042D53C**, đây là dòng lệnh gán địa chỉ chuỗi ASCII “Benadryl” vào thanh ghi EDX.
- Ngoài ra, ta nhận lấy ở dòng lệnh ngay phía trên:

```
MOV EAX, DWORD PTR SS:[EBP-4]
```

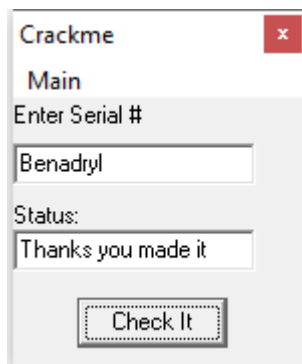
- Dự đoán đây chính là lệnh gán địa chỉ chuỗi mà user nhập vào vào thanh ghi EAX. Tiến hành đặt Break Point tại đây.

0042D4D9	. A89E4200	DD 3_1.00429EA8	
0042D4DE	. 80A64200	DD 3_1.0042A680	
0042D4E2	. 2C984200	DD 3_1.0042982C	
0042D4E6	. 4CF64100	DD 3_1.0041F64C	
0042D4EA	. 00F34100	DD 3_1.0041F300	
0042D4EE	. 8BC0	MOV EAX,EAX	
0042D4F0	. F4D44200	DD 3_1.0042D4F4	
0042D4F4	. 07	DB 07	
0042D4F5	. 06	DB 06	
0042D4F6	. 54 46 6F 72 61	ASCII "TForm1"	
0042D4FC	. 9CD34200	DD 3_1.0042D39C	
0042D500	. E01C4200	DD 3_1.00421CE0	
0042D504	. 3B	DB 3B	CHAR ':'
0042D505	. 00	DB 00	
0042D506	. 07	DB 07	
0042D507	. 63 72 61 63 61	ASCII "crackme"	
0042D50E	. 00	DB 00	
0042D50F	. 00	DB 00	
0042D510	. 55	PUSH EBP	
0042D511	. 8BEC	MOV EBP,ESP	
0042D513	. 6A 00	PUSH 0	
0042D515	. 53	PUSH EBX	
0042D516	. 8BD8	MOV EBX,EAX	
0042D518	. 33C0	XOR EAX,EAX	
0042D51A	. 55	PUSH EBP	
0042D51B	. 68 7BD54200	PUSH 3_1.0042D57B	
0042D520	. 64:FF30	PUSH DWORD PTR FS:[EAX]	
0042D523	. 64:8920	MOV DWORD PTR FS:[EAX],ESP	
0042D526	. 8D55 FC	LEA EDX,DWORD PTR SS:[EBP-4]	
0042D529	. 8B83 DC010000	MOV EAX,DWORD PTR DS:[EBX+10C]	
0042D52F	. E8 54CCFEFF	CALL 3_1.0041A188	
0042D534	. 8B45 FC	MOV EAX,DWORD PTR SS:[EBP-4]	
0042D537	. BA 90D54200	MOV EDX,3_1.0042D590	ASCII "Benadryl"
0042D53C	. E8 8F63FDFF	CALL 3_1.004038D0	
0042D541	. 74 12	JE SHORT 3_1.0042D555	
0042D543	. BA A4D54200	MOV EDX,3_1.0042D5A4	ASCII "Wrong Code DUDE"
0042D548	. 8B83 E8010000	MOV EAX,DWORD PTR DS:[EBX+1E8]	
0042D54E	. E8 65CCFEFF	CALL 3_1.0041A1B8	
0042D553	. EB 10	JMP SHORT 3_1.0042D565	
0042D555	. BA BCD54200	MOV EDX,3_1.0042D5BC	
0042D55A	. 8B83 E8010000	MOV EAX,DWORD PTR DS:[EBX+1E8]	ASCII "Thanks you made it"
0042D560	. E8 53CCFEFF	CALL 3_1.0041A1B8	
0042D565	. 33C0	XOR EAX,EAX	

- Quả nhiên ở cửa sổ Register thì giá trị thanh ghi EAX đã trùng khớp với những gì ta dự đoán

Registers (FPU)	
EAX	02286F44 ASCII "abc"
ECX	C12815D7
EDX	00000000
EBX	022859A4
ESP	0019F768
EBP	0019F77C
ESI	02281A3C
EDI	02281A3C
EIP	0042D537 3_1.0042D537
C 0	ES 002B 32bit 0(FFFFFFFF)
P 1	CS 0023 32bit 0(FFFFFFFF)
A 0	SS 002B 32bit 0(FFFFFFFF)
Z 0	DS 002B 32bit 0(FFFFFFFF)
S 0	FS 0053 32bit 3D3000(FFF)
T 0	GS 002B 32bit 0(FFFFFFFF)
D 0	
O 0	LastErr ERROR_SUCCESS (00000000)
EFL	00000206 (NO,NB,NE,A,NS,PE,GE,G)
ST0	empty 0.0
ST1	empty 0.0
ST2	empty 0.0
ST3	empty 0.0
ST4	empty 1.000000000000000000
ST5	empty 1.000000000000000000
ST6	empty 16.000000000000000000
ST7	empty 16.000000000000000000
3 2 1 0 E S P U O Z D I	
FST 4020	Cond 1 0 0 0 Err 0 0 1 0 0 0 0 0 (EQ)
FCW 1372	Prec NEAR,64 Mask 1 1 0 0 1 0

- Vậy tức là lệnh CALL ở địa chỉ 0042D53C sẽ tiến hành so sánh giữa hai thanh ghi EAX và EDX sao cho lệnh JE SHORT 3_1.0042D555 có thể thực thi và nhảy đến goodboy.
- Cũng có nghĩa là serial chính là “Benadryl”. Nhập serial tìm được ta được goodboy.



Đoạn phát sinh key: Chính là dòng lệnh 0042D537 MOV EDX, 3_1.0042D590

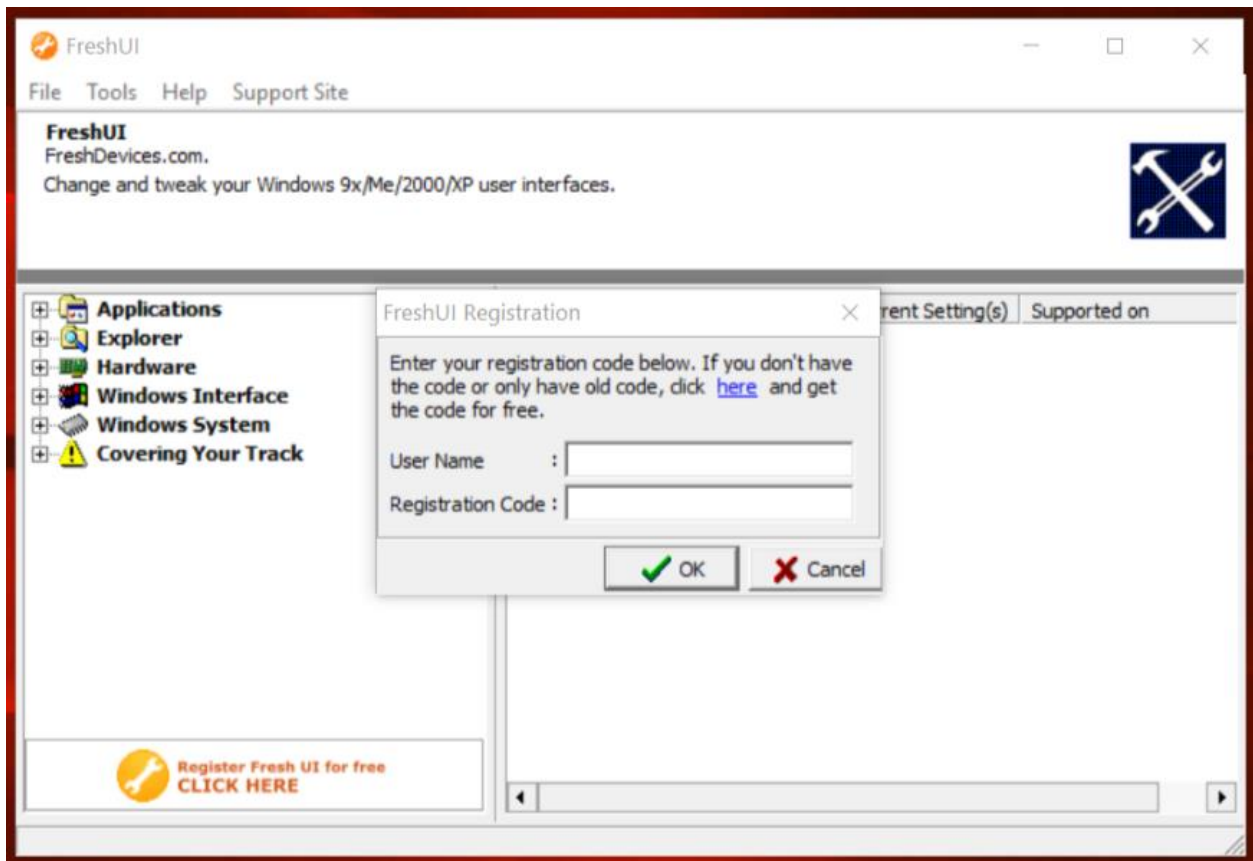
0042D52F	: E8 54CCFEFF	CALL 3_1.0041A188	
0042D534	: 8B45 FC	MOV EAX,DWORD PTR SS:[EBP-4]	
0042D537	: BA 90D54200	MOV EDX,3_1.0042D590	ASCII "Benadryl"
0042D53C	: E8 8F63FDFF	CALL 3_1.004038D0	
0042D53C	: 00000000	DB 00000000	
0042D590	: 42 65 6E 61 6	ASCII "Benadryl",0	
0042D599	: 00	DB 00	

Ý nghĩa: Gán địa chỉ chứa “Benadryl” vào thanh ghi EDX.

Key tương ứng với user: Đây là key lưu sẵn nên chỉ có một key là *Benadryl*

2. Bài 3_2

- Giao diện chương trình cần crack :



- Nhập thử User Name : 1712698, reg : 123456
- Khi nhập sai **User Name** và **Registration Code** , chương trình sẽ tự xóa dữ liệu nhập vào và không thông báo => điều này khiến việc tìm badboy trở nên khó khăn.
- Ta load 3_2.exe vào OllyDBG, nhấn Run(F9) ta thấy chương trình dừng ở điểm **Entry Point**

<Module>	\$	55	PUSH EBP	
00524069	.	8BEC	MOV EBP,ESP	
0052406B	.	83C4 F0	ADD ESP,-10	
0052406E	.	53	PUSH EBX	
0052406F	.	B8 903D5200	MOV EAX,3_2.00523D90	
00524074	.	E8 3722EEFF	CALL 3_2.004062B0	
00524079	.	8B1D 7C975200	MOV EBX,DWORD PTR [52977C]	3_2.0052AB78
0052407F	.	8B03	MOV EAX,DWORD PTR [EBX]	
00524081	.	E8 5EC7F4FF	CALL 3_2.004707E4	
00524086	.	8B03	MOV EAX,DWORD PTR [EBX]	
00524088	.	BA 40415200	MOV EDX,3_2.00524140	ASCII "FreshUI"
0052408D	.	E8 0EC2F4FF	CALL 3_2.004702A0	
00524092	.	8B0D C4995200	MOV ECX,DWORD PTR [5299C4]	3_2.0052ACB8
00524098	.	8B03	MOV EAX,DWORD PTR [EBX]	
0052409A	.	8B15 90224900	MOV EDX,DWORD PTR [492290]	3_2.004922E4
005240A0	.	E8 57C7F4FF	CALL 3_2.004707FC	
005240A5	.	8B0D FC925200	MOV ECX,DWORD PTR [5292FC]	3_2.0052AC60
005240AB	.	8B03	MOV EAX,DWORD PTR [EBX]	
005240AD	.	8B15 CCFE4800	MOV EDX,DWORD PTR [48FECC]	3_2.0048FF18
005240B3	.	E8 44C7F4FF	CALL 3_2.004707FC	
005240B8	.	8B0D D0915200	MOV ECX,DWORD PTR [5291D0]	3_2.0052AC74
005240BE	.	8B03	MOV EAX,DWORD PTR [EBX]	
005240C0	.	8B15 20084900	MOV EDX,DWORD PTR [490820]	3_2.0049086C
005240C6	.	E8 31C7F4FF	CALL 3_2.004707FC	
005240CB	.	8B0D 289A5200	MOV ECX,DWORD PTR [529A28]	3_2.0052AC48
005240D1	.	8B03	MOV EAX,DWORD PTR [EBX]	
005240D3	.	8B15 0CFC4800	MOV EDX,DWORD PTR [48FC0C]	3_2.0048FC58
005240D9	.	E8 1EC7F4FF	CALL 3_2.004707FC	
005240DE	.	8B0D 98945200	MOV ECX,DWORD PTR [529498]	3_2.0052AC94
005240E4	.	8B03	MOV EAX,DWORD PTR [EBX]	
005240E6	.	8B15 70114900	MOV EDX,DWORD PTR [491170]	3_2.004911BC

- Tìm trong “All reference text strings” ta sẽ thấy được **goodboy** của chương trình ở đây (sử dụng Search for text tìm những từ hay xuất hiện trong thông báo chính xác)

R Text strings referenced in 3_2:CODE		
Address	Disassembly	Text string
00490A5C	ASCII "Button3Click"	
00490A6F	ASCII "Button2Click"	
00490A7C	ASCII "TFormReg"	
00490A96	DD 3_2.004342D4	ASCII "CC"
00490AA2	ASCII "TFormReg"	
00490AAA	DD 3_2.0049086C	ASCII "DXE"
00490AAE	DD 3_2.00465530	ASCII "4UF"
00490AB5	ASCII "DMReg"	
00490D33	MOV EAX, 3_2.00490F6C	ASCII "FreshUI has been registered successfully."
00490D5F	MOV EDX, 3_2.00490FA0	ASCII "\\Software\FreshDevices\FreshUI"
00490D74	MOV EDX, 3_2.00490FC8	ASCII "Owner"
00490D89	MOV EDX, 3_2.00490FD8	ASCII "RegCode"
00490DB6	MOV EDX, 3_2.00490FE8	ASCII "Registered"
00490DE3	MOV EDX, 3_2.00490FFC	ASCII "\\ndfile\\shellex\\Topic"
00490E19	MOV EDX, 3_2.00491028	ASCII "C2"
00490E8B	MOV EDX, 3_2.00491040	ASCII "About - [Business License]"
00490E9E	MOV EDX, 3_2.00491064	ASCII "About - [Personal License]"
00490EB8	MOV EDX, 3_2.00491088	ASCII "FreshUI - [Business License]"
00490ECB	MOV EDX, 3_2.004910B0	ASCII "FreshUI - [Personal License]"
00490F6C	ASCII "FreshUI has been"	
00490F7C	ASCII " registered succ"	
00490F8C	ASCII "essfully.",0	
00490FA0	ASCII "\\Software\FreshD"	
00490FB0	ASCII "evices\FreshUI",0	
00490FC8	ASCII "Owner",0	
00490FD8	ASCII "RegCode",0	
00490FE8	ASCII "Registered",0	
00490FFC	ASCII "\\ndfile\\shellex"	

- Tìm ra được dòng **00490D33** chứa goodBoy

CPU - main thread, module 3_2			
00490CF3	> 8B45 FC	MOV EAX,DWORD PTR [EBP-4]	
00490CF6	. E8 C1FDFFFF	CALL 3_2.00490ABC	
00490CFB	. 833D 80AC5200	CMP DWORD PTR [52AC80],0	
00490D02	. 74 0F	JE SHORT 3_2.00490D13	
00490D04	. A1 48985200	MOV EAX,DWORD PTR [529848]	
00490D09	. 8B00	MOV EAX,DWORD PTR [EAX]	
00490D0B	. 8B10	MOV EDX,DWORD PTR [EAX]	
00490D0D	. FF92 F8000000	CALL DWORD PTR [EDX+F8]	
00490D13	> 833D 78AC5200	CMP DWORD PTR [52AC78],1	
00490D1A	. 1BC0	SBB EAX,EAX	
00490D1C	. 40	INC EAX	
00490D1D	. 3C 01	CMP AL,1	
00490D1F	. 74 12	JE SHORT 3_2.00490D33	
00490D21	. 833D 7CAC5200	CMP DWORD PTR [52AC7C],1	
00490D28	. 1BC0	SBB EAX,EAX	
00490D2A	. 40	INC EAX	
00490D2B	. 3C 01	CMP AL,1	
00490D2D	. 0F85 B4010000	JNZ 3_2.00490EE7	
00490D33	> B8 6C0F4900	MOV EAX,3_2.00490F6C	ASCII "FreshUI has been registered successfully."
00490D38	. E8 0767FBFF	CALL 3_2.00447444	
00490D3D	. B2 01	MOV DL,1	
00490D3F	. A1 E0464200	MOV EAX,DWORD PTR [4246E0]	
00490D44	. E8 973AF9FF	CALL 3_2.004247E0	
00490D49	. A3 84AC5200	MOV DWORD PTR [52AC84],EAX	
00490D4E	. BA 02000000	MOV EDI,00000002	
00490D53	. A1 84AC5200	MOV EAX,DWORD PTR [52AC84]	
00490D58	. E8 233BF9FF	CALL 3_2.00424880	
00490D5D	. B1 01	MOV CL,1	
00490D5F	. BA A00F4900	MOV EDI,3_2.00490FA0	ASCII "\\Software\FreshDevices\FreshUI"
00490D64	. A1 84AC5200	MOV EAX,DWORD PTR [52AC84]	
00490D69	. E8 563CF9FF	CALL 3_2.004249C4	
00490D6E	. 8B0D 88AC5200	MOV ECX,DWORD PTR [52AC88]	
00490D74	. BA C80F4900	MOV EDI,3_2.00490FC8	ASCII "Owner"

- Dựa vào câu lệnh nhảy ta có thể tìm ra được quy trình các bước kiểm tra **name** và **registration key** do người dùng nhập vào.
- Đề ý mấu chốt ở đây là 2 lệnh kiểm tra giá trị các biến **PTR [52AC80]** và **PTR [52AC78]** bằng lệnh **CMP**. Có thể trong lệnh **CALL** trên làm thay đổi giá trị của 2 biến kia để lưu giá trị trung gian.

00490CF6	. E8 C1FDFFFF	CALL 3_2.00490ABC
00490CFB	. 833D 80AC5201	CMP DWORD PTR [52AC80],0
00490D02	. 74 0F	JE SHORT 3_2.00490D13
00490D04	. A1 48985200	MOV EAX,DWORD PTR [529848]
00490D09	. 8B00	MOV EAX,DWORD PTR [EAX]
00490D0B	. 8B10	MOV EDX,DWORD PTR [EAX]
00490D0D	. FF92 F8000000	CALL DWORD PTR [EDX+F8]
00490D13	> 833D 78AC5201	CMP DWORD PTR [52AC78],1
00490D1A	. 1BC0	SBB EAX,EAX
00490D1C	. 40	INC EAX
00490D1D	. 3C 01	CMP AL,1
00490D1F	. 74 12	JE SHORT 3_2.00490D33

- Ta di chuyển đến hàm con **CALL 3_2.00490ABC** xem sao nhé :
 - o Đầu tiên đặt Break Point tại lệnh **CALL**
 - o Nhấn F9 để chạy chương trình, nhập một chuỗi bất kì
 - **User Name** = “vanquan”
 - **Registration Code** = “1712698”
 - o Chương trình sẽ dừng ở vị trí đặt Break Point

00490CF3	> 8B45 FC	MOV EAX,DWORD PTR [EBP-4]
00490CF6	. E8 C1FDFFFF	CALL 3_2.00490ABC
00490CFB	. 833D 80AC5201	CMP DWORD PTR [52AC80],0
00490D02	. 74 0F	JE SHORT 3_2.00490D13
00490D04	. A1 48985200	MOV EAX,DWORD PTR [529848]
00490D09	. 8B00	MOV EAX,DWORD PTR [EAX]

- o Ta dùng lệnh **Step into (F7)** để vào hàm con. Sau đó dùng **Step over** để đi xem từng dòng, để ý sự thay đổi giá trị thanh ghi bên bảng Registers.
- o Đầu tiên ta sẽ chú ý đến lệnh **XOR EAX,EAX**. Và các biến **PTR[52AC78]**, **PTR[52AC7C]**, **PTR[52AC80]** sẽ được gán bằng 0.

00490ACD	. 33C0	XOR EAX,EAX
00490ACF	. 55	PUSH EBP
00490AD0	. 68 5C0C4900	PUSH 3_2.00490C5C
00490AD5	. 64:FF30	PUSH DWORD PTR FS:[EAX]
00490AD8	. 64:8920	MOV DWORD PTR FS:[EAX],ESP
00490ADB	. 33C0	XOR EAX,EAX
00490ADD	. A3 78AC5200	MOV DWORD PTR [52AC78],EAX
00490AE2	. 33C0	XOR EAX,EAX
00490AE4	. A3 7CAC5200	MOV DWORD PTR [52AC7C],EAX
00490AE9	. 33C0	XOR EAX,EAX
00490AEB	. A3 80AC5200	MOV DWORD PTR [52AC80],EAX

- o Tới đó vẫn chưa thấy gì thay đổi đáng kể, ta tiếp tục F8 xem sao.
- o Khi nhảy đến dòng **00490B4B** ta thấy có 1 vòng lặp

00490B4B	>	8D4D F0	LEA ECX,DWORD PTR [EBP-10]
00490B4E	.	0FBFD6	MOVSX EDX,SI
00490B51	.	8B45 FC	MOV EAX,DWORD PTR [EBP-4]
00490B54	.	8B80 6403000	MOV EAX,DWORD PTR [EAX+364]
00490B5A	.	8B80 3802000	MOV EAX,DWORD PTR [EAX+238]
00490B60	.	8B38	MOV EDI,DWORD PTR [EAX]
00490B62	.	FF57 0C	CALL DWORD PTR [EDI+C]
00490B65	.	8B55 F0	MOV EDX,DWORD PTR [EBP-10]
00490B68	.	A1 8CAC5200	MOV EAX,DWORD PTR [52AC8C]
00490B6D	.	E8 663AF7FF	CALL 3_2.004045D8
00490B72	✓	75 0A	JNZ SHORT 3_2.00490B7E
00490B74	.	C705 80AC520	MOV DWORD PTR [52AC80],-1
00490B7E	>	46	INC ESI
00490B7F	.	66:FFCB	DEC BX
00490B82	^	75 C7	JNZ SHORT 3_2.00490B4B

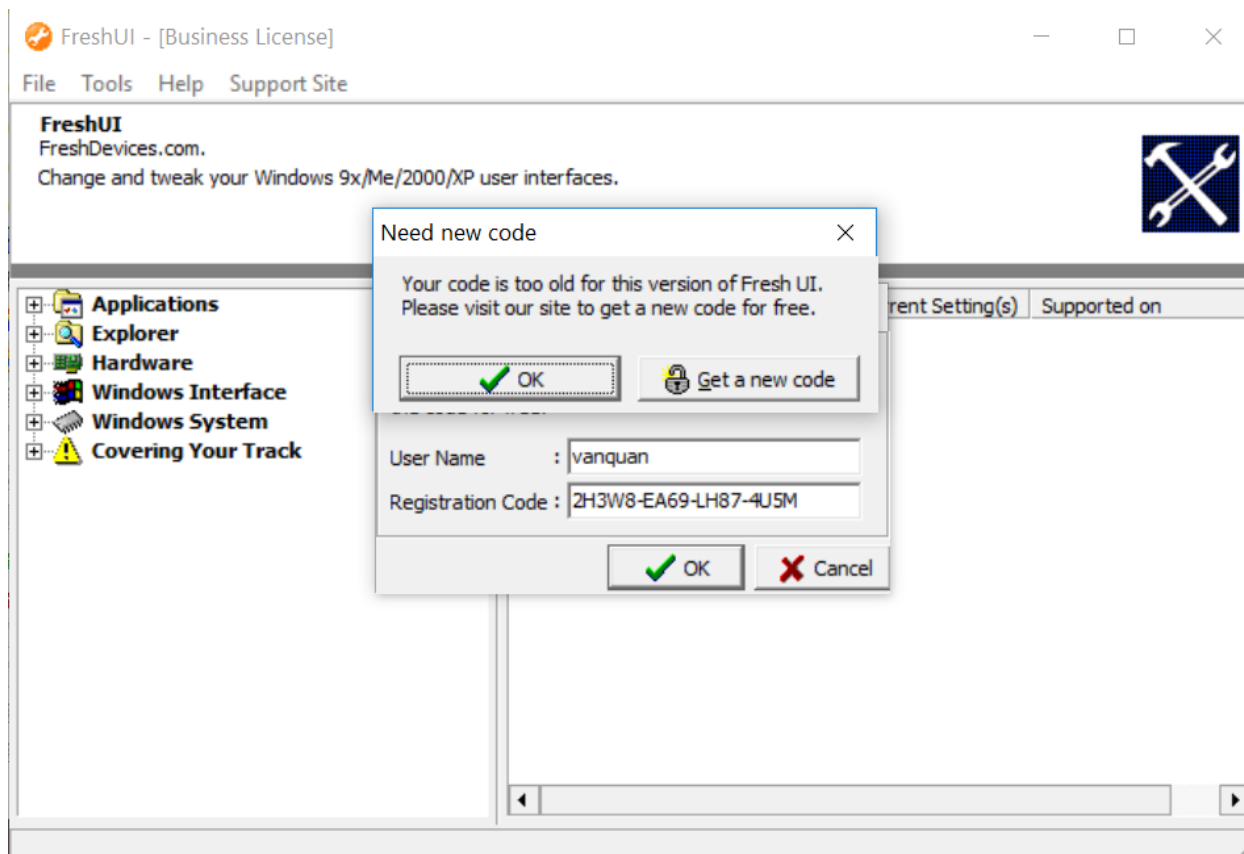
- Vòng lặp trên lần lượt lấy địa chỉ chuỗi **Registration Code** của người dung nhập để nạp vào thanh ghi **EAX** và địa chỉ một chuỗi nào đó có sẵn nạp vào thanh ghi **EDX**.
- Lệnh CALL 3_2.004045D08 so sánh nội dung 2 chuỗi **EAX** và **EDX**. Nếu **EAX** = **EDX** thì PTR [52AC80] = -1.
- Tiếp tục dùng F8 chạy hết vòng lặp, và để ý bên bảng Registers thấy thanh ghi **EDX** chứa một chuỗi ASCII nào đó(có khả năng đó là key), **EAX** chứa key mà mình đã nhập, **ESI** dùng để lưu biến đếm của vòng lặp.

Registers (FPU)	
EAX	0237E458 ASCII "1712698"
ECX	00000000
EDX	02388D5C ASCII "5H5BA-S468-DW55-2X7Z"
EBX	00000002
ESP	0019F2A0
EBP	0019F2D0
ESI	00000002
EDI	004385CC 3_2.004385CC

- Sau khi chạy xong vòng lặp ta tìm ra được 4 chuỗi :

1	7B9X8-Z523-KV67-6T5U
2	AQ3MA-G3A6-FQ43-6X9B
3	5H5BA-S468-DW55-2X7Z
4	2H3W8-EA69-LH87-4U5M

- Ta thử nhập từng chuỗi xem có phải key cần tìm không nhé, và đây là kết quả Tất cả các chuỗi vừa nhập là Key cũ.



○ OK, ta tiếp tục dùng F8 để đi qua các lệnh, và ta lại thấy xuất hiện thêm vòng lặp tương tự lúc này.(Hy vọng đây là các chuỗi key cần tìm)

00490BA3	>	8D4D EC	LEA ECX,DWORD PTR [EBP-14]	
00490BA6	.	0FBFD6	MOVSX EDX,SI	
00490BA9	.	8B45 FC	MOV EAX,DWORD PTR [EBP-4]	
00490BAC	.	8B80 5C030001	MOV EAX,DWORD PTR [EAX+35C]	
00490BB2	.	8B80 38020001	MOV EAX,DWORD PTR [EAX+238]	
00490BB8	.	8B38	MOV EDI,DWORD PTR [EAX]	3_2.004385CC
00490BBA	.	FF57 0C	CALL DWORD PTR [EDI+C]	
00490BB0	.	8B55 EC	MOV EDX,DWORD PTR [EBP-14]	
00490BC0	.	A1 8CAC5200	MOV EAX,DWORD PTR [52AC8C]	
00490BC5	.	E8 0E3AF7FF	CALL 3_2.004045D8	
00490BCA	√	75 0A	JNZ SHORT 3_2.00490BD6	
00490BCC	.	C705 78AC5201	MOV DWORD PTR [52AC78],-1	
00490BD6	>	46	INC ESI	
00490BD7	.	66:FFCB	DEC BX	
00490BDA	^	75 C7	JNZ SHORT 3_2.00490BA3	

○ Ta làm tương tự như vòng lặp đầu tiên và thấy thanh **EDX** giữ địa chỉ 1 chuỗi ASCII nào đó

Registers (FPU)		
EAX	0019F2BC	
ECX	00000000	
EDX	02388DA4	ASCII "8Y3Z8-C72C-NF75-5PCR"
EBX	000001EF	
ESP	0019F2A0	
EBP	0019F2D0	
ESI	00000001	
EDI	004385CC	3_2.004385CC

○ Ta nhấn F8 để chạy vòng lặp này, có thể thấy số lần lặp là rất lớn. Nên ta để ý thử xem có gì đặc biệt không. Thấy 1 địa chỉ chứa chuỗi ASCII nhảy liên tục theo vòng lặp. Và đó là địa chỉ của chứa chuỗi trên thanh **EDX**.

○ Ta chú ý ở bảng Registers và theo code, ta thấy ESI là biến đếm của vòng lặp, và EBS là giới hạn số vòng lặp. Ban đầu EBS nhận giá trị 1F0 và ESI = 0 . Vòng lặp sẽ thoát ra khi EBS = 0.

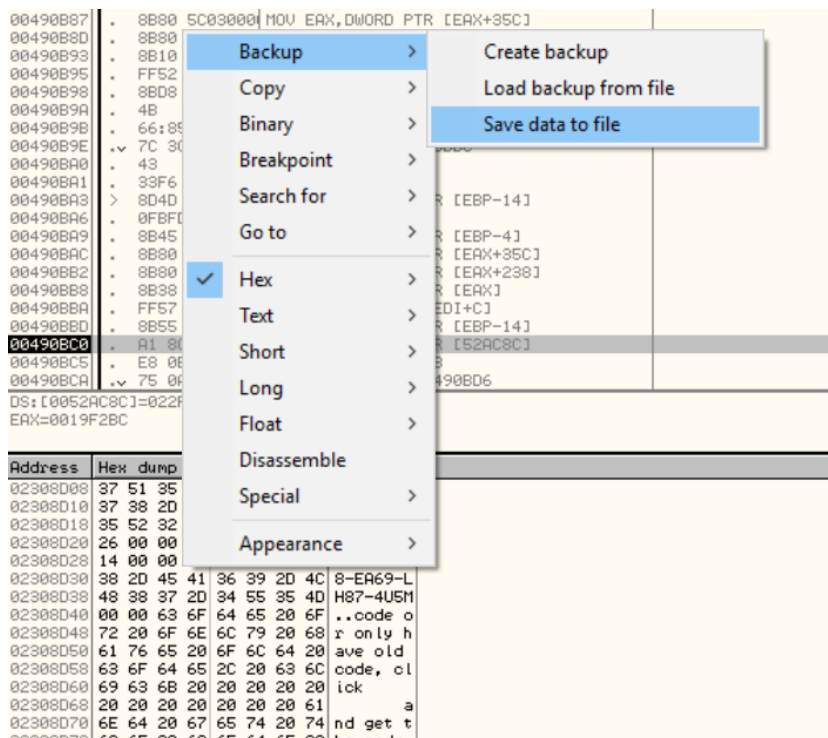
```
Registers (FPU)
EAX 0019F2BC
ECX 00000000
EDX 02308D08 ASCII "7Q5Y5-V878-DH26-5R2W"
EBX 000001F0
ESP 0019F2A0
EBP 0019F2D0
ESI 00000000
EDI 004385CC 3_2.004385CC
EIP 00490BC0 3_2.00490BC0
```

○ Nhấn chuột phải chọn “*Follow in dump*” ở thanh ghi chứa địa chỉ mã ASCII, sau đó click chuột phải chọn Backup -> save data to file, ta được 1 file lưu tất cả đoạn mã Registration code do chương trình tạo ra.

```
Registers (FPU)
EAX 0019F2BC
ECX 00000000
EDX 02308D08 ASCII "7Q5Y5-V878-DH26-5R2W"
EBX 000001F0
ESP 0019F2A0
EBP 0019F2D0
ESI 00000000
EDI 004385CC
EIP 00490BC0
C 0 ES 000
P 1 CS 000
A 0 SS 000
Z 0 DS 000
S 0 FS 000
T 0 GS 000
D 0
O 0 LastE
EFL 0000020
ST0 empty
ST1 empty
ST2 empty
ST3 empty
ST4 empty
ST5 empty
ST6 empty
ST7 empty
FST 4800 Cond 1 0 0 0 Err 0 0 0 0 0 0 (EQ)
```

Context menu options:

- Increment Plus
- Decrement Minus
- Zero
- Set to 1
- Modify Enter
- Copy selection to clipboard Ctrl+C
- Copy all registers to clipboard
- Follow in Dump**
- View MMX registers
- View 3DNow! registers
- View debug registers
- Appearance >



- Ta được file Key.mem gồm 1000 dòng. Ta có thể nhận ra 4 dòng Key cuối là key cũ ta đã tìm được ở trên.

```
1 7Q5Y5-V878-DH26-5R2W
2 8Y3Z8-C72C-NF75-5PCR
3 AV4K4-XA85-JPA8-AW3E
4 7S6D8-N355-VYA2-5K6G
5 2B3B9-R556-UX83-AX4U
6 2R4X4-G257-EX49-8BAX
7 2N2Z2-B326-BD33-4H7J
8 4H4U7-C288-GQ6A-8Y4L
9 7R8V5-EA63-DG2C-2X3D
10 2H8X6-MA25-HL82-8B5H
11 4Y2W5-B573-ET56-6KAT
12 2M7Y5-Y442-FKA6-7X2Z
13 6Z7B2-K683-WX5A-3P6T
14 3HAPA-E887-TT35-7T8D
15 2SAZA-X299-JRCA-6F4C
16 2L8S2-N758-QJ45-8H8L
17 3UCW9-P379-XF34-9S9S
18 6R7S2-K336-MK45-6G6M
19 AX7AC-EA27-UU68-5J7T
20 2F2R4-X43A-FP67-4V8G
21 5C3T8-Z6C8-RG34-4U7H
```

```
982 gj8c2r-2g2cg3q6-5d3x35
983 xh2x2p-5x3bk8r8-6a7g32
984 tc5p7h-2x4ms2t6-8s8m72
985 jc8g6g-8t3xq3f8-3d8g38
986 ih8b4y-2v3ms2k8-6j8u68
987 dj6p7m-7j6dr3b3-6t3f22
988 cp6s7q-3v4kx4t3-7u4x33
989 px3h7b-6e4wd8x3-6f2x34
990 eb8e8m-3e3bb4e5-3r3t36
991 yg7u4q-8b2hx4t3-8x5d44
992 vk5q3r-6y4jf5b8-3n3m37
993 he6u7k-6y3nc5q6-2h5b32
994 ev6u6r-4q6pa6x7-4b6k42
995 bi2p6f-7x8ey2v2-3r3y83
996 tn8w8g-2d6xf2j2-5m4s66
997 7B9X8-Z523-KV67-6T5U
998 AQ3MA-G3A6-FQ43-6X9B
999 5H5BA-S468-DW55-2X7Z
1000 2H3W8-EA69-LH87-4U5M
```


- Nhưng vòng lặp thứ hai chỉ có $1F0 = 496$ lần lặp tức là có 496 key, vậy còn 500 dòng kia ở đâu ra ? Ta cùng đi kiểm tra tiếp xem sao.
- Ta tiếp tục nhấn F8 thì phát hiện thêm 1 vòng lặp tương tự như 2 vòng lặp trên.

```

00490BFB > 8D4D E8 LEA ECX,DWORD PTR [EBP-18]
00490BFE . 0FBFD6 MOVSX EDX,SI
00490C01 . 8B45 FC MOV EAX,DWORD PTR [EBP-4]
00490C04 . 8B80 60030000 MOV EAX,DWORD PTR [EAX+360]
00490C0A . 8B80 38020000 MOV EAX,DWORD PTR [EAX+238]
00490C10 . 8B38 MOV EDI,DWORD PTR [EAX]
00490C12 . FF57 0C CALL DWORD PTR [EDI+C]
00490C15 . 8B55 E8 MOV EDX,DWORD PTR [EBP-18]
00490C18 . A1 8CAC5200 MOV EAX,DWORD PTR [52AC8C]
00490C1D . E8 B639F7FF CALL 3_2.004045D8
00490C22 . 75 0A JNZ SHORT 3_2.00490C2E
00490C24 . C705 7CAC5200 MOV DWORD PTR [52AC7C],-1
00490C2E > 46 INC ESI
00490C2F . 66:FFCB DEC BX
00490C32 . 75 C7 JNZ SHORT 3_2.00490BFB

```

- Ta nhìn bên Registers thấy $EBX = 1F4(\text{hex}) = 500$ và mã ASCII được **EDX** lưu trùng với dòng 497 trong file key.mem. Như vậy ta có thể chắc chắn vòng lặp này sẽ cho ta 500 key nữa.

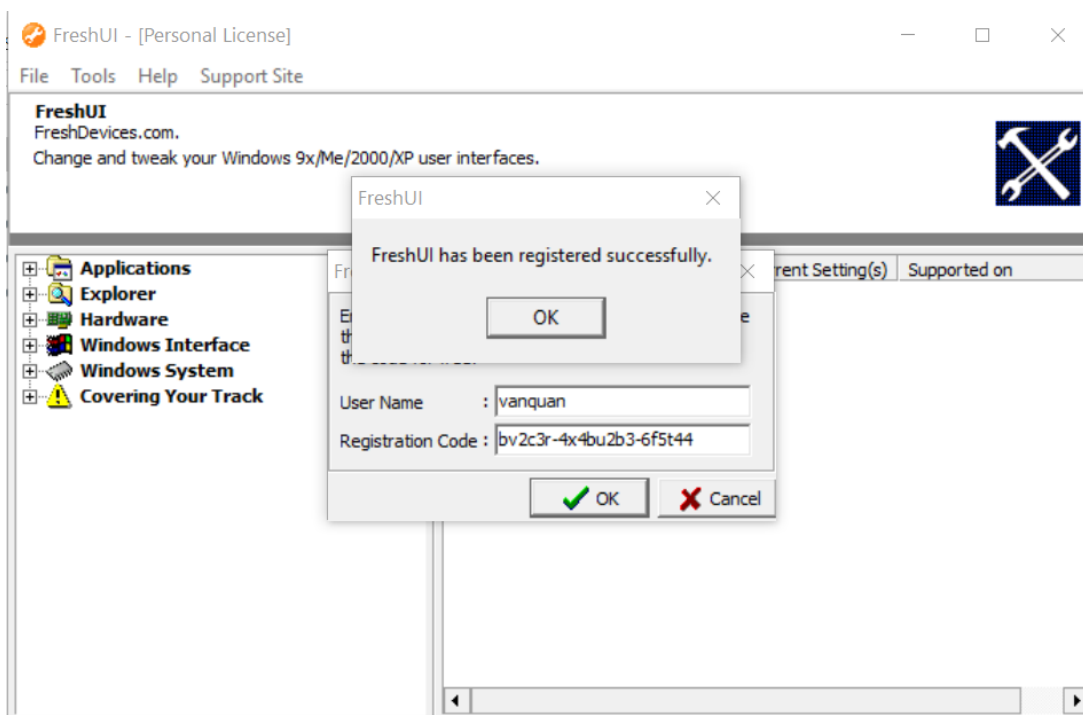
```

Registers (FPU) <
EAX 0019F2B8
ECX 00000000
EDX 02308D50 ASCII "bv2c3r-4x4bu2b3-6f5t44"
EBX 000001F4
ESP 0019F2A0
EBP 0019F2D0
ESI 00000000
EDI 004385CC 3_2.004385CC
EIP 00490C18 3_2.00490C18

497 bv2c3r-4x4bu2b3-6f5t44

```

- Ta thử các key của vòng lặp thứ 3 thì thấy rằng đó là các key đúng, có thể dùng crack chương trình.

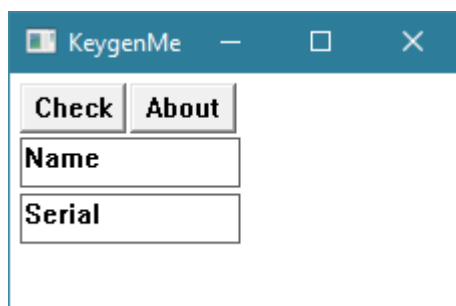


Tổng kết quá trình crack câu 3_2 :

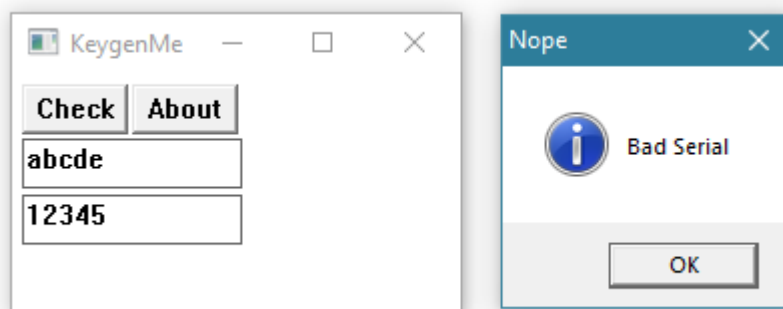
- Để ý thấy rằng, chương trình không kiểm tra **USER NAME** và chỉ kiểm tra mỗi **REGISTRATION CODE**. Do đó, ta có thể đăng ký bằng một **USER NAME** bất kỳ và lựa chọn một trong tổng số key ta đã tìm được trong file key.mem trên, ta đều có thể đăng ký thành công chương trình. (Trừ 4 key cũ từ dòng 997 -> 1000)
- Vậy ta thu được 1 file thu được các mã key của chương trình (file key.mem kèm theo báo cáo). Trong đó dòng 1 đến dòng 996 là key dùng để crack chương trình, dòng 997 đến dòng 1000 là 4 key của “Phiên bản cũ” .

3. Bài 3_3 :

Chạy thử file 3_3.exe ta được như sau:



Nhập thử Name, Serial và nhấn check, ta được một **Badboy**:



Từ đây ta có chuỗi “Bad Serial” để tìm kiếm đoạn code tương ứng.

Mở OllyDbg, sau đó load 3_3.exe. Sử dụng chức năng tìm kiếm string, ta tìm được đoạn code sau:

0040249E	. 56	PUSH ESI	
0040249F	. E8 C4470000	CALL 3_3.00406C68	
004024A4	. 83C4 08	ADD ESP,8	
004024A7	. 85C0	TEST EAX,EAX	
004024A9	~ 75 15	JNZ SHORT 3_3.004024C0	
004024AB	. 6A 40	PUSH 40	
004024AD	. 68 0E114100	PUSH 3_3.0041110E	
004024B2	. 68 13114100	PUSH 3_3.00411113	
004024B7	. 6A 00	PUSH 0	
004024B9	. E8 B2DC0000	CALL <JMP.&USER32.MessageBoxA>	
004024BE	~ EB 5A	JMP SHORT 3_3.0040251A	
004024C0	> 6A 40	PUSH 40	
004024C2	. 68 2C114100	PUSH 3_3.0041112C	
004024C7	. 68 31114100	PUSH 3_3.00411131	
004024CC	. 6A 00	PUSH 0	
004024CE	. E8 9DDC0000	CALL <JMP.&USER32.MessageBoxA>	
004024D3	~ EB 45	JMP SHORT 3_3.0040251A	
004024D5	. FF75 10	PUSH DWORD PTR [EBP+10]	

Ta thấy ở phía trên đoạn code hiển thị “Bad Serial” là đoạn code hiển thị “Good, Now Make a Keygen!”. Ta có thể xác định đây chính là **Goodboy**.

Tiếp tục, ở phía trên **Goodboy**, ta thấy đoạn code sau:

0040249F	. E8 C4470000	CALL 3_3.00406C68	
004024A4	. 83C4 08	ADD ESP,8	
004024A7	. 85C0	TEST EAX,EAX	
004024A9	~ 75 15	JNZ SHORT 3_3.004024C0	
004024AB	. 6A 40	PUSH 40	
004024AD	. 68 0E114100	PUSH 3_3.0041110E	
004024B2	. 68 13114100	PUSH 3_3.00411113	
004024B7	. 6A 00	PUSH 0	
004024B9	. E8 B2DC0000	CALL <JMP.&USER32.MessageBoxA>	
004024BE	~ EB 5A	JMP SHORT 3_3.0040251A	

Ta thấy 2 dòng lệnh sau:

TEST EAX,EAX

JNZ SHORT 3_3.004024C0 (004024C0 là địa chỉ của Badboy)

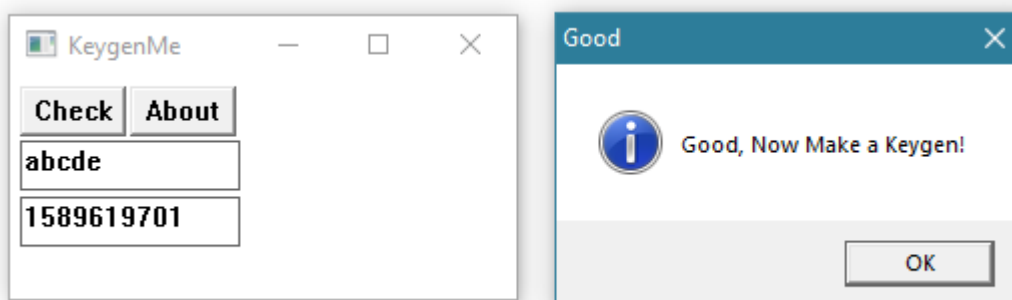
Nếu EAX=0 thì sau lệnh TEST, ZF sẽ được gán bằng 1. Lúc này lệnh JNZ sẽ không được thực thi, do đó chương trình sẽ tiếp tục chạy đến **Goodboy**.

Ngược lại, nếu $EAX \neq 0$ thì sau lệnh TEST, ZF được gán bằng 0. Lệnh JNZ sẽ jump vào Badboy

Ở ngay phía trên có lệnh **CALL 3_3.00406C68**, ta có thể đoán được lệnh này sẽ làm thay đổi EAX. Đặt breakpoint tại đây và chạy thử, nhập *Name=abcde*, *Serial=12345*, ta được:

```
Registers (FPU)
EAX 0019FA44 ASCII "12345"
ECX 00331000
EDX 00000005
EBX 00000005
ESP 0019FA1C
EBP 0019FA04
ESI 0019FA5C ASCII "1589619701"
EDI 0019FA41
EIP 0040249F 3_3.0040249F
```

Ta thấy ngay EAX đang chứa địa chỉ của serial ta nhập vào, và ở bên dưới ESI đang chứa địa chỉ một chuỗi khác. Đây có thể là serial. Thử chạy file 3_3.exe và nhập vào ta được:



Đây chính là **Goodboy** ta cần tìm!

Tuy nhiên khi thử với Name khác, ta lại được Serial khác với lần trước => Có thể xác định đây là dạng generate key dựa theo username người dùng nhập vào.

Mở đoạn code trong hàm **CALL 3_3.00406C68** để kiểm tra:

00406C68	8B4C24 04	MOV ECX,DWORD PTR [ESP+4]
00406C6C	8B5424 08	MOV EDX,DWORD PTR [ESP+8]
00406C70	53	PUSH EBX
00406C71	8A01	MOV AL,BYTE PTR [ECX]
00406C73	8A1A	MOV BL,BYTE PTR [EDX]
00406C75	3808	CMP AL,BL
00406C77	75 38	JNZ SHORT 3_3.00406CB1
00406C79	84C0	TEST AL,AL
00406C7B	74 30	JE SHORT 3_3.00406CAD
00406C7D	8A41 01	MOV AL,BYTE PTR [ECX+1]
00406C80	8A5A 01	MOV BL,BYTE PTR [EDX+1]
00406C83	3808	CMP AL,BL
00406C85	75 2A	JNZ SHORT 3_3.00406CB1
00406C87	84C0	TEST AL,AL
00406C89	74 22	JE SHORT 3_3.00406CAD
00406C8B	8A41 02	MOV AL,BYTE PTR [ECX+2]
00406C8E	8A5A 02	MOV BL,BYTE PTR [EDX+2]
00406C91	3808	CMP AL,BL
00406C93	75 1C	JNZ SHORT 3_3.00406CB1
00406C95	84C0	TEST AL,AL
00406C97	74 14	JE SHORT 3_3.00406CAD
00406C99	8A41 03	MOV AL,BYTE PTR [ECX+3]
00406C9C	8A5A 03	MOV BL,BYTE PTR [EDX+3]
00406C9F	3808	CMP AL,BL
00406CA1	75 0E	JNZ SHORT 3_3.00406CB1
00406CA3	83C1 04	ADD ECX,4
00406CA6	83C2 04	ADD EDX,4
00406CA9	84C0	TEST AL,AL
00406CAB	75 C4	JNZ SHORT 3_3.00406C71
00406CAD	31C0	XOR EAX,EAX
00406CAF	5B	POP EBX
00406CB0	C3	RET
00406CB1	19C0	SBB EAX,EAX
00406CB3	5B	POP EBX
00406CB4	83C8 01	OR EAX,1
00406CB7	C3	RET

Ta thấy đây chỉ là hàm so sánh serial nhập vào và key được generate, nếu giống nhau thì gán EAX=0, ngược lại EAX=1

⇒ Chắc chắn phía trên sẽ chứa đoạn code generate key

Duyệt lên phía trên, ta thấy một số đoạn code quan trọng sau:

1. Giới hạn độ dài của Name:

00402365	E8 B2490000	CALL 3_3.00406D1C
0040236A	83C4 04	ADD ESP,4
0040236D	83F8 04	CMP EAX,4
00402370	7F 1C	JG SHORT 3_3.0040238E
00402372	6A 30	PUSH 30
00402374	68 CB104100	PUSH 3_3.004110CB
00402379	68 FC104100	PUSH 3_3.004110FC
0040237E	6A 00	PUSH 0
00402380	E8 EBDD0000	CALL <JMP.&USER32.MessageBoxA>
00402385	31C0	XOR EAX,EAX
00402387	5F	POP EDI
00402388	5E	POP ESI
00402389	5B	POP EBX
0040238A	C9	LEAVE
0040238B	C2 1000	RET 10

```
Style = MB_OK|MB_ICONEXCLAMATION|MB_APPLMODAL
Title = "Error"
Text = "Name Too Short"
hOwner = NULL
MessageBoxA
```

Đây là đoạn code kiểm tra độ dài Name, nếu $\text{length}(\text{Name}) \leq 4$ thì sẽ xuất hiện hộp thoại báo lỗi “Name Too Short”

Tiếp theo, đặt breakpoint trước lệnh CALL, nhập Name=”abcde”, ta được:

Registers (FPU)	
EAX	00000005
ECX	0019FA3C ASCII "abcde"
EDX	00000005
EBX	0019FA3C
ESP	0019FA20
EBP	0019FA84
ESI	00000111
EDI	00000000
EIP	00402365 3_3.00402365

Nhưng khi nhập Name="abcdefgh", ta lại được kết quả sau:

EAX	00000006
ECX	0019FA3C ASCII "abcdef"
EDX	00000006
EBX	0019FA3C
ESP	0019FA20
EBP	0019FA84
ESI	00000111
EDI	00000000
EIP	00402365 3_3.00402365

Điều này có nghĩa chương trình chỉ lấy tối đa 6 ký tự từ Name nhập vào.

⇒ $5 \leq \text{length}(\text{Name}) \leq 6$

2. Đoạn code băm chuỗi (hệ HEX):

Đoạn khởi tạo:

0040238E	>	C745 F0 3E02	MOV DWORD PTR [EBP-10],23E
00402395	.	C745 F4 0000	MOV DWORD PTR [EBP-C],0
0040239C	.	C745 F8 0000	MOV DWORD PTR [EBP-8],0

Đoạn vòng lặp chính:

004023A3	>	8D55 B8	LEA EDX,DWORD PTR [EBP-48]
004023A6	.	52	PUSH EDX
004023A7	.	E8 70490000	CALL 3_3.00406D1C
004023AC	.	83C4 04	ADD ESP,4
004023AF	.	3B45 F8	CMP EAX,DWORD PTR [EBP-8]
004023B2	~	0F8C B70000	JL 3_3.0040246F
004023B8	.	8B5D F8	MOV EBX,DWORD PTR [EBP-8]
004023BB	.	83FB 08	CMP EBX,8
004023BE	.	BE 01000000	MOV ESI,1
004023C3	~	72 05	JB SHORT 3_3.004023CA
004023C5	.	BE 00000000	MOV ESI,0
004023CA	~	72 0A	JB SHORT 3_3.004023D6
004023CC	.	B8 A1000000	MOV EAX,0A1
004023D1	.	E8 72010000	CALL 3_3.00402548
004023D6	>	8D7C1D B8	LEA EDI,DWORD PTR [EBP+EBX-48]
004023DA	.	803F 00	CMP BYTE PTR [EDI],0
004023DD	~	0F84 84000000	JE 3_3.00402467
004023E3	.	85F6	TEST ESI,ESI
004023E5	~	75 0A	JNZ SHORT 3_3.004023F1
004023E7	.	B8 A1000000	MOV EAX,0A1
004023EC	.	E8 57010000	CALL 3_3.00402548
004023F1	>	803F 00	CMP BYTE PTR [EDI],0
004023F4	~	0F84 6D000000	JE 3_3.00402467
004023FA	.	8B4D F8	MOV ECX,DWORD PTR [EBP-8]
004023FD	.	83F9 08	CMP ECX,8
00402400	.	BA 01000000	MOV EDX,1
00402405	~	72 02	JB SHORT 3_3.00402409
00402407	.	8AD6	MOV DL,DH
00402409	>	8955 B4	MOV DWORD PTR [EBP-4C],EDX
0040240C	~	72 0A	JB SHORT 3_3.00402418
0040240E	.	B8 A2000000	MOV EAX,0A2
00402413	.	E8 30010000	CALL 3_3.00402548
00402418	>	8B5D F8	MOV EBX,DWORD PTR [EBP-8]
0040241B	.	8D741D B8	LEA ESI,DWORD PTR [EBP+EBX-48]

```

0040241F . 0FB606 MOVZX EAX, BYTE PTR [ESI]
00402422 . B9 05000000 MOV ECX, 5
00402427 . 99 CQ
00402428 . F7F9 IDIV ECX
0040242A . 89C3 MOV EBX, EAX
0040242C . 89D9 MOV ECX, EBX
0040242E . C1F9 1F SAR ECX, 1F
00402431 . 8B45 F0 MOV EAX, DWORD PTR [EBP-10]
00402434 . 8B55 F4 MOV EDX, DWORD PTR [EBP-C]
00402437 . 0FAFC3 IMUL ECX, EAX
0040243A . 0FAFD3 IMUL EDX, EBX
0040243D . 03CA ADD ECX, EDX
0040243F . F7E3 MUL EBX
00402441 . 03D1 ADD EDX, ECX
00402443 . 8945 F0 MOV DWORD PTR [EBP-10], EAX
00402446 . 8955 F4 MOV DWORD PTR [EBP-C], EDX
00402449 . 837D B4 00 CMP DWORD PTR [EBP-4C], 0
0040244D . 75 0A JNZ SHORT 3_3.00402459
0040244F . B8 A3000000 MOV EAX, 0A3
00402454 . E8 EF000000 CALL 3_3.00402548
00402459 > 0FB63E MOVZX EDI, BYTE PTR [ESI]
0040245C . 89FA MOV EDX, EDI
0040245E . C1FA 1F SAR EDX, 1F
00402461 . 017D F0 ADD DWORD PTR [EBP-10], EDI
00402464 . 1155 F4 ADC DWORD PTR [EBP-C], EDX
00402467 > FF45 F8 INC DWORD PTR [EBP-8]
0040246A ^ E9 34FFFFFF JMP 3_3.004023A3

```

3. Đoạn code chuyển đổi giá trị HEX sang chuỗi hệ thập phân:

```

00407752 > 83BD 6CFFFFFF CMP DWORD PTR [EBP-94], 0
00407759 . 74 1F JE SHORT 3_3.0040777A
0040775B . 8B55 F0 MOV EDX, DWORD PTR [EBP-10]
0040775E . 8D4D C7 LEA ECX, DWORD PTR [EBP-39]
00407761 . 3BD1 CMP EDX, ECX
00407763 . 75 15 JNZ SHORT 3_3.0040777A
00407765 . 8B45 90 MOV EAX, DWORD PTR [EBP-70]
00407768 . 31D2 XOR EDX, EDX
0040776A . BB 3A000000 MOV EBX, 3A
0040776F . 8945 88 MOV DWORD PTR [EBP-78], EAX
00407772 . 8955 8C MOV DWORD PTR [EBP-74], EDX
00407775 . 8955 90 MOV DWORD PTR [EBP-70], EDX
00407778 . EB 65 JMP SHORT 3_3.004077DF
0040777A > 56 PUSH ESI
0040777B . 8B5D EC MOV EBX, DWORD PTR [EBP-14]
0040777E . 89D9 MOV ECX, EBX
00407780 . 8B55 8C MOV EDX, DWORD PTR [EBP-74]
00407783 . 8B45 88 MOV EAX, DWORD PTR [EBP-78]
00407786 . C1F9 1F SAR ECX, 1F
00407789 . 57 PUSH EDI
0040778A . E8 890A0000 CALL 3_3.00408218
0040778F . 83C3 30 ADD EBX, 30
00407792 . 5F POP EDI
00407793 . 5E POP ESI
00407794 . 83D1 00 ADC ECX, 0
00407797 . 8B5D F4 MOV BYTE PTR [EBP-C], BL
0040779A . 80FB 39 CMP BL, 39
0040779D . 7E 20 JLE SHORT 3_3.004077BF
0040779F . F7C6 00010000 TEST ESI, 100
004077A5 . 8B45 F0 MOV EAX, DWORD PTR [EBP-10]
004077A8 . 8A5D F4 MOV BL, BYTE PTR [EBP-C]
004077AB . BA 07000000 MOV EDX, 7
004077B0 . 75 05 JNZ SHORT 3_3.004077B7
004077B2 . BA 27000000 MOV EDX, 27

```

3_3.004131F8

004077B7	> 00D3	ADD BL,DL	
004077B9	. 8945 F0	MOV DWORD PTR [EBP-10],EAX	
004077BC	. 885D F4	MOV BYTE PTR [EBP-C],BL	
004077BF	> 56	PUSH ESI	
004077C0	. 8B5D EC	MOV EBX,DWORD PTR [EBP-14]	
004077C3	. 89D9	MOV ECX,EBX	
004077C5	. 8B55 8C	MOV EDX,DWORD PTR [EBP-74]	
004077C8	. 8B45 88	MOV EAX,DWORD PTR [EBP-78]	
004077CB	. C1F9 1F	SAR ECX,1F	
004077CE	. 57	PUSH EDI	
004077CF	. E8 440A0000	CALL 3_3.00408218	
004077D4	. 5F	POP EDI	
004077D5	. 5E	POP ESI	
004077D6	. 8A5D F4	MOV BL,BYTE PTR [EBP-C]	
004077D9	. 8945 88	MOV DWORD PTR [EBP-78],EAX	
004077DC	. 8955 8C	MOV DWORD PTR [EBP-74],EDX	
004077DF	> 8B4D F0	MOV ECX,DWORD PTR [EBP-10]	
004077E2	. 8B45 90	MOV EAX,DWORD PTR [EBP-70]	
004077E5	. 31D2	XOR EDX,EDX	
004077E7	. FF4D F0	DEC DWORD PTR [EBP-10]	
004077EA	. 0B45 88	OR EAX,DWORD PTR [EBP-78]	
004077ED	. 0B55 8C	OR EDX,DWORD PTR [EBP-74]	
004077F0	. 09C2	OR EDX,EAX	
004077F2	. 8B19	MOV BYTE PTR [ECX],BL	
004077F4	. ^ 0F85 58FFFFFF	JNZ 3_3.00407752	

⇒ Từ đó, thuật toán của KeyGen được mô tả như sau:

Đặt s là chuỗi Name, key là giá trị trả về

-Bước 1: Kiểm tra độ dài chuỗi

Nếu $\text{length}(s) \leq 4$ thì báo lỗi

Nếu $\text{length}(s) > 6$ thì chỉ giữ lại 6 kí tự đầu

-Bước 2: Khởi tạo $\text{key} = 574$ (tương ứng với 23E trong HEX)

-Bước 3: Duyệt qua từng kí tự của s (biến chạy i) và thực hiện các thao tác sau:

$\text{temp} = s[i] / 5;$

$\text{key} = \text{key} * \text{temp};$

$\text{key} = \text{key} + s[i];$

Mình hoạ với $s = "1234abcde"$:

-Bước 1: $\text{length}(s) = 11 > 6$ nên chỉ giữ lại 6 kí tự đầu

⇒ $s = "1234ab"$

-Bước 2: Khởi tạo $\text{key} = 574$

-Bước 3:

i	s[i]	Quá trình thực hiện	Giá trị cuối của key
0	'1'=49	temp=49/5=9 key=574*9=5166 key=5166+49=5215	5215
1	'2'=50	temp=50/5=10 key=5215*10=52150 key=52150+50=52200	52200
2	'3'=51	temp=51/5=10 key=52200*10=522000 key=522000+51=522051	522051
3	'4'=52	temp=52/5=10 key=522051*10=5220510 key=522051+52=5220562	5220562
4	'a'=97	temp=97/5=19 key=5220562*19=99190678 key=99190678+97=99190775	99190775
5	'b'=98	temp=98/5=19 key=99190775*19=1884624725 key=1884624725+98=1884624823	1884624823

Lưu ý: Key nằm trong miền giá trị -2,147,483,648 đến 2,147,483,647. Do đó khi viết keygen phải sử dụng kiểu dữ liệu tương ứng để tự động quy đổi khi tràn số.

Ta có code minh họa chương trình tìm keygen như sau: (C#)

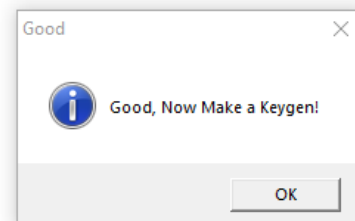
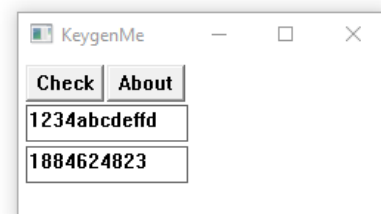
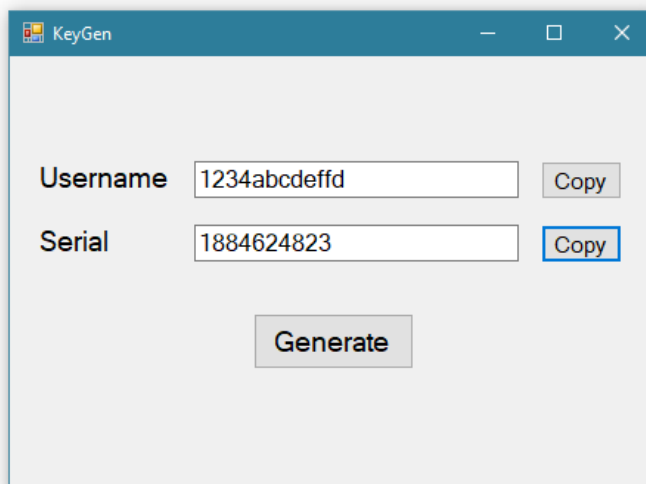
```
string generateKey(string username)
{
    string res = "";
    if (username.Length > 6)
    {
        username = username.Substring(0, 6);
    }

    int key, temp;

    key = 574;
    for (int i = 0; i < username.Length; i++)
    {
        temp = username[i] / 5;
        key = temp * key;
        key += username[i];
    }

    res = key.ToString();
    return res;
}
```

Chạy 1 bộ test của ta phát sinh:



V. Đánh giá mức độ hoàn thành

Tên bài	Mức độ hoàn thành	Phần chưa làm được
3_1	100%	0
3_2	100%	0
3_3	100%	0

VI. Nguồn tài liệu tham khảo

- Tutorial giảng viên.
- File hướng dẫn x86

HẾT