

Câu 2.2.exe

-Đầu tiên, ta xét lệnh gọi hàm:

0040A290 . E8 DBAFFFFF CALL 2_2.00405270

Đi vào trong hàm này lại tiếp tục xét lệnh:

004052AA |. E8 3DE7FFFF CALL 2_2.004039EC

Tiếp tục, trong hàm ta xét lệnh:

00403A26 |. E8 61FFFFFF CALL 2_2.0040398C

Trong hàm, ta đặt breakpoint tại: 004039C2 . FFD0 CALL EAX

Và ấn F9 10 lần, sau đó đi vào trong hàm.

Ta thấy, từ 00408EE0 đến 00408F14 thực hiện vòng lặp:

00408EE0	56	PUSH ESI
00408EE1	33F6	XOR ESI,ESI
00408EE3	B8 ACC74000	MOV EAX,2_2.0040C7AC
00408EE8	> 8930	MOV DWORD PTR [EAX],ESI
00408EEA	. BA 00000000	MOV EDX,8
00408EEF	> 8B08	MOV ECX,DWORD PTR [EAX]
00408EF1	. F6C1 01	TEST CL,1
00408EF4	~ 74 0C	JE SHORT 2_2.00408F02
00408EF6	. D1E9	SHR ECX,1
00408EF8	. 81F1 2083B8E1	XOR ECX,EDB88320
00408EFE	. 8908	MOV DWORD PTR [EAX],ECX
00408F00	~ EB 02	JMP SHORT 2_2.00408F04
00408F02	> D128	SHR DWORD PTR [EAX],1
00408F04	> 4A	DEC EDX
00408F05	^ 75 E8	JNZ SHORT 2_2.00408EEF
00408F07	. 46	INC ESI
00408F08	. 83C0 04	ADD EAX,4
00408F0B	. 81FE 00010000	CMP ESI,100
00408F11	^ 75 D5	JNZ SHORT 2_2.00408EE8
00408F13	. 5E	POP ESI
00408F14	. C3	RET

Đây là nơi để thực hiện việc tạo một bảng giá trị để sử dụng về sau.

Ta mô tả đoạn lệnh bằng đoạn code C++ sau:

```

void createTable(vector<unsigned int> &table)
{
    unsigned int v = 0;
    for (int i = 0; i < 0x100; i++)
    {
        v = i;
        for (int j = 0; j < 8; j++)
        {
            if (v % 2 != 0)
            {
                v = (v >> 1) ^ 0xEDB88320;
            }
            else
            {
                v = v >> 1;
            }
        }
        table.push_back(v);
    }
}

```

-Tiếp đến: 0040A2F8 . E8 BF88FFFF CALL 2_2.00402BBC

Thực hiện việc nhập chuỗi serial.

-Tại: 0040A316 . 83F8 0C CMP EAX,0C

Sẽ thực hiện việc kiểm tra độ dài chuỗi serial. Nếu độ dài khác 12 thì báo “Serial must be 12 chars long!”.

Ngược lại, tiếp tục chương trình.

-Ta tiến hành công việc xử lý đối với 4 ký tự đầu của serial:

+Vào bên trong câu lệnh gọi hàm: 0040A378 . E8 2BFCFFFF CALL 2_2.00409FA8

Ta xét câu lệnh và đi vào bên trong nó: 0040A04E |. E8 05ECFFFF CALL 2_2.00408C58

00408C58	53	PUSH EBX
00408C59	56	PUSH ESI
00408C5A	57	PUSH EDI
00408C5B	51	PUSH ECX
00408C5C	890424	MOV DWORD PTR [ESP],EAX
00408C5F	8B1C24	MOV EBX,DWORD PTR [ESP]
00408C62	8B1B	MOV EBX,DWORD PTR [EBX]
00408C64	8B0424	MOV EAX,DWORD PTR [ESP]
00408C67	8B40 04	MOV EAX,DWORD PTR [EAX+4]
00408C6A	8B35 9CB2400	MOV ESI,DWORD PTR [40B29C]
00408C70	0FAFF1	IMUL ESI,ECX
00408C73	8BCE	MOV ECX,ESI
00408C75	85C9	TEST ECX,ECX
00408C77	74 45	JE SHORT 2_2.00408CBE
00408C79	8BF3	MOV ESI,EBX
00408C7B	C1E6 04	SHL ESI,4
00408C7E	8BFB	MOV EDI,EBX
00408C80	C1EF 05	SHR EDI,5
00408C83	33F7	XOR ESI,EDI
00408C85	03F3	ADD ESI,EBX
00408C87	8BF9	MOV EDI,ECX
00408C89	C1EF 0B	SHR EDI,0B
00408C8C	83E7 03	AND EDI,3
00408C8F	8B3CBA	MOV EDI,DWORD PTR [EDX+EDI*4]
00408C92	03F9	ADD EDI,ECX
00408C94	33F7	XOR ESI,EDI
00408C96	2BC6	SUB EAX,ESI
00408C98	2B0D 9CB2400	SUB ECX,DWORD PTR [40B29C]
00408C9E	8BF0	MOV ESI,EAX
00408CA0	C1E6 04	SHL ESI,4
00408CA3	8BF8	MOV EDI,EAX
00408CA5	C1EF 05	SHR EDI,5
00408CA8	33F7	XOR ESI,EDI
00408CAA	03F0	ADD ESI,EAX
00408CAC	8BF9	MOV EDI,ECX
00408CAE	83E7 03	AND EDI,3
00408CB1	8B3CBA	MOV EDI,DWORD PTR [EDX+EDI*4]
00408CB4	03F9	ADD EDI,ECX
00408CB6	33F7	XOR ESI,EDI
00408CB8	2BDE	SUB EBX,ESI
00408CBA	85C9	TEST ECX,ECX
00408CBC	75 BB	JNZ SHORT 2_2.00408C79
00408CBE	8B1424	MOV EDX,DWORD PTR [ESP]
00408CC1	891A	MOV DWORD PTR [EDX],EBX
00408CC3	8B1424	MOV EDX,DWORD PTR [ESP]
00408CC6	8942 04	MOV DWORD PTR [EDX+4],EAX
00408CC9	5A	POP EDX
00408CCA	5F	POP EDI
00408CCB	5E	POP ESI
00408CCC	5B	POP EBX
00408CCD	C3	RET

Đây là hàm có nhiệm vụ dùng 4 bytes ký tự đầu của serial và thực hiện biến đổi để ra 8 bytes và lưu nó.

Ta thực hiện mô tả lại bằng đoạn code C++ sau:

```

unsigned int getInt(string s)
{
    unsigned int res = 0;
    for (int i = s.length() - 1; i >= 0; i--)
    {
        res = res << 8;
        res += (s[i]);
    }
    return res;
}

```

```

unsigned char* convert4bytesTo8bytes(unsigned int edx)
{
    unsigned int eax = 0xd1fc1e8f, ecx = 0xc6ef3720, ebx = 0x3beabc9a, bc = 0x9e3779b9;
    unsigned int esi, edi;
    while (ecx != 0)
    {
        esi = ebx;
        esi = esi << 4;
        edi = ebx;
        edi = edi >> 5;
        esi = esi xor edi;
        esi = esi + ebx;
        edi = ecx;
        edi = edi >> 11;
        edi = edi & 3;
        if (edi) edi = 0;
        else edi = edx;
        edi = edi + ecx;
        esi = esi xor edi;
        eax = eax - esi;
        ecx = ecx - bc;
        esi = eax;
        esi = esi << 4;
        edi = eax;
        edi = edi >> 5;

        esi = esi xor edi;
        esi = esi + eax;
        edi = ecx;
        edi = edi & 3;
        if (edi) edi = 0;
        else edi = edx;
        edi = edi + ecx;
        esi = esi xor edi;
        ebx = ebx - esi;
    }

    unsigned char *res = new unsigned char[9];

    for (int i = 0; i < 4; i++)
    {
        res[i] = ebx & 0xff;
        ebx = ebx >> 8;
        res[i + 4] = eax & 0xff;
        eax = eax >> 8;
    }
    res[8] = '\0';
    return res;
}

```

Có thể hiểu đơn giản là ta truyền vào 4 ký tự đầu của serial vào hàm getInt để nhận được con số và truyền số đó vào edx của hàm convert4bytesTo8bytes và trả về kết quả dạng chuỗi ký tự.

+Xét tại: 0040A3A2 . E8 71EBFFFF CALL 2_2.00408F18

00408F18	53	PUSH EBX
00408F19	56	PUSH ESI
00408F1A	8BF0	MOV ESI,EAX
00408F1C	83CB FF	OR EBX,FFFFFFFF
00408F1F	8BC6	MOV EAX,ESI
00408F21	E8 DAFFFF	CALL 2_2.00403F00
00408F26	85C0	TEST EAX,EAX
00408F28	7E 21	JLE SHORT 2_2.00408F4B
00408F2A	BA 01000000	MOV EDX,1
00408F2F	33C9	XOR ECX,ECX
00408F31	8A4C16 FF	MOV CL,BYTE PTR [ESI+EDX-1]
00408F35	33CB	XOR ECX,EBX
00408F37	81E1 FF000000	AND ECX,0FF
00408F3D	C1EB 08	SHR EBX,8
00408F40	331C8D ACC740	XOR EBX,DWORD PTR [ECX*4+40C7AC]
00408F47	42	INC EDX
00408F48	48	DEC EAX
00408F49	75 E4	JNZ SHORT 2_2.00408F2F
00408F4B	F7D3	NOT EBX
00408F4D	8BC3	MOV EAX,EBX
00408F4F	5E	POP ESI
00408F50	5B	POP EBX
00408F51	C3	RET

Đây là hàm băm crc32, có nhiệm vụ biến đổi 8 bytes đầu vào về 4 bytes. Sau đây là đoạn code C++ mô tả hàm băm crc32:

```
unsigned int crc32(unsigned char ESI[9], vector<unsigned int> table)
{
    unsigned int ECX, EDX;
    unsigned int EBX = 0xffffffff;
    for (EDX = 0; EDX < 8; EDX++)
    {
        ECX = 0;
        ECX = ESI[EDX];
        ECX = ECX ^ EBX;
        ECX = ECX & 0xff;
        EBX = EBX >> 8;
        EBX = EBX ^ table[ECX];
    }
    EBX = ~EBX;
    return EBX;
}
```

Có thể hiểu, hàm crc32 nhận vào chuỗi ký tự và sử dụng bảng giá trị đã khởi tạo lúc đầu để tiến hành băm.

+Xét tại: 0040A3A7 . 3D 80B456B4 CMP EAX,B456B480

Kết quả băm trả về gán vào EAX, sau đó tiến hành so sánh với 0xB456B480. Nếu bằng thì tiếp tục thực hiện các công việc tiếp sau, ngược lại, thông báo sai “Invalid Serial”.

⇒ Từ việc tiến hành khảo sát và mô tả lại công việc xử lý đối với 4 ký tự đầu của serial. Thực hiện thuật toán brute force để vét cạn các trường hợp có thể xảy ra đối với 4 ký tự đầu này. Ta có thể đưa ra kết luận là 4 ký tự đầu của serial đúng là PSk=

-Ta tiến hành công việc xử lý đối với 8 ký tự cuối của serial:

+Xét tại: 0040A3D0 . E8 07BDFFFF CALL 2_2.004060DC

004060DC	53	PUSH EBX
004060DD	56	PUSH ESI
004060DE	57	PUSH EDI
004060DF	8BFA	MOV EDI,EDX
004060E1	8BF0	MOV ESI,EAX
004060E3	8BC6	MOV EAX,ESI
004060E5	E8 16DEFFFF	CALL 2_2.00403F00
004060EA	8BD8	MOV EBX,EAX
004060EC	8BC7	MOV EAX,EDI
004060EE	8BD3	MOV EDX,EBX
004060F0	E8 FDFDFFFF	CALL 2_2.004040F4
004060F5	8BD6	MOV EDX,ESI
004060F7	8B37	MOV ESI,DWORD PTR [EDI]
004060F9	85DB	TEST EBX,EBX
004060FB	74 15	JE SHORT 2_2.00406112
004060FD	8A02	MOV AL,BYTE PTR [EDX]
004060FF	3C 61	CMP AL,61
00406101	72 06	JB SHORT 2_2.00406109
00406103	3C 7A	CMP AL,7A
00406105	77 02	JAE SHORT 2_2.00406109
00406107	2C 20	SUB AL,20
00406109	8B06	MOV BYTE PTR [ESI],AL
0040610B	42	INC EDX
0040610C	46	INC ESI
0040610D	4B	DEC EBX
0040610E	85DB	TEST EBX,EBX
00406110	75 EB	JNZ SHORT 2_2.004060FD
00406112	5F	POP EDI
00406113	5E	POP ESI
00406114	5B	POP EBX
00406115	C3	RET

Đây là hàm có chức năng in hoa chuỗi ký tự. Cụ thể ở đây là in hoa 8 ký tự cuối của serial để xử lý về sau.

+Xét tại: 0040A3E7 . E8 2CFBFFFF CALL 2_2.00409F18

00409F18	55	PUSH EBP
00409F19	. 8BEC	MOV EBP,ESP
00409F1B	. 51	PUSH ECX
00409F1C	. 53	PUSH EBX
00409F1D	. 8945 FC	MOV DWORD PTR [EBP-4],EAX
00409F20	. 8B45 FC	MOV EAX,DWORD PTR [EBP-4]
00409F23	. E8 24A1FFFF	CALL 2_2.0040404C
00409F28	. 33C0	XOR EAX,EAX
00409F2A	. 55	PUSH EBP
00409F2B	. 68 8A9F4000	PUSH 2_2.00409F8A
00409F30	. 64:FF30	PUSH DWORD PTR FS:[EAX]
00409F33	. 64:8920	MOV DWORD PTR FS:[EAX],ESP
00409F36	. 33DB	XOR EBX,EBX
00409F38	. 8B45 FC	MOV EAX,DWORD PTR [EBP-4]
00409F3B	. E8 C09FFFFF	CALL 2_2.00403F00
00409F40	. 85C0	TEST EAX,EAX
00409F42	√ 7E 21	JLE SHORT 2_2.00409F65
00409F44	. BA 01000000	MOV EDX,1
00409F49	> 8B4D FC	MOV ECX,DWORD PTR [EBP-4]
00409F4C	. 8A4C11 FF	MOV CL,BYTE PTR [ECX+EDX-1]
00409F50	. 80C1 D0	ADD CL,0D0
00409F53	. 80E9 0A	SUB CL,0A
00409F56	√ 72 08	JB SHORT 2_2.00409F60
00409F58	. 80C1 F9	ADD CL,0F9
00409F5B	. 80E9 06	SUB CL,6
00409F5E	√ 73 01	JNB SHORT 2_2.00409F61
00409F60	> 43	INC EBX
00409F61	> 42	INC EDX
00409F62	. 48	DEC EAX
00409F63	^ 75 E4	JNZ SHORT 2_2.00409F49
00409F65	> 8B45 FC	MOV EAX,DWORD PTR [EBP-4]
00409F68	. E8 939FFFFF	CALL 2_2.00403F00
00409F6D	. 3BD8	CMPL EBX,EAX
00409F6F	. 0F94C0	SETL AL
00409F72	. 8BD8	MOV EBX,EAX
00409F74	. 33C0	XOR EAX,EAX
00409F76	. 5A	POP EDX
00409F77	. 59	POP ECX
00409F78	. 59	POP ECX
00409F79	. 64:8910	MOV DWORD PTR FS:[EAX],EDX
00409F7C	. 68 919F4000	PUSH 2_2.00409F91
00409F81	> 8D45 FC	LEA EAX,DWORD PTR [EBP-4]
00409F84	. E8 F79CFFFF	CALL 2_2.00403C80
00409F89	. C3	RET

Sau khi thực hiện in hoa 8 ký tự cuối của serial, thì đến lượt hàm này kiểm tra sự hợp lệ của của 8 ký tự cuối này. Nếu nó ngoài khoảng ký tự từ ‘0’ đến ‘9’, ‘A’ đến ‘F’ thì báo lỗi “Serial have a invalid format”. Ngược lại, tiếp tục thực hiện các công đoạn sau.

Ta có thể hiểu hàm này một cách đơn giản thông qua đoạn code C++ sau:

```

bool checkKey(string s)
{
    for (int i = 0; i < s.length(); i++)
    {
        if (s[i]<'0')
        {
            return false;
        }
        else if (s[i] > '9')
        {
            if (s[i]<'A' || s[i]>'F')
            {
                return false;
            }
        }
    }
    return true;
}

```

+Xét tại: 0040A433 . E8 60FBFFFF CALL 2_2.00409F98

00409F98	\$ 3E:8130 DEC0	XOR DWORD PTR [EAX],1337C0DE
00409F9F	. 50	PUSH EAX
00409FA0	? C3	RET
00409FA1	. 00C3	ADD BL,AL
00409FA3	. 90	NOP
00409FA4	\$ 0FC8	BSWAP EAX
00409FA6	. C3	RET

Đây là hàm kiểm tra 2 ký tự cuối của serial.

Ta giả sử lúc này 8 ký tự cuối của serial được lưu vào EAX là 13579BDF, thì lúc này hàm sẽ lấy EAX xor với 0x1337C0DE. Nhưng có một lưu ý là tại: 00409FA0 |? C3 RET thì câu lệnh này chỉ trả về nếu byte cuối cùng của kết quả là C3. Như vậy, với ví dụ này thì kết quả là 605B01. Và 01!=C3 nên không thỏa.

Vậy để xor với 0x1337C0DE thì 2 ký tự cuối của serial phải là 1D hoặc 1d để kết quả có byte cuối là C3.

+Xét tại: 0040A453 . E8 C0EAFFFF CALL 2_2.00408F18

00408F18	53	PUSH EBX
00408F19	56	PUSH ESI
00408F1A	8BF0	MOV ESI,EAX
00408F1C	83CB FF	OR EBX,FFFFFFFF
00408F1F	8BC6	MOV EAX,ESI
00408F21	E8 DAFFFFFF	CALL 2_2.00408F00
00408F26	85C0	TEST EAX,EAX
00408F28	7E 21	JLE SHORT 2_2.00408F4B
00408F2A	BA 01000000	MOV EDX,1
00408F2F	33C9	XOR ECX,ECX
00408F31	8A4C16 FF	MOV CL,BYTE PTR [ESI+EDX-1]
00408F35	33CB	XOR ECX,EBX
00408F37	81E1 FF000000	AND ECX,0FF
00408F3D	C1EB 08	SHR EBX,8
00408F40	331C8D ACC74	XOR EBX,DWORD PTR [ECX*4+40C7AC]
00408F47	42	INC EDX
00408F48	48	DEC EAX
00408F49	75 E4	JNZ SHORT 2_2.00408F2F
00408F4B	F7D3	NOT EBX
00408F4D	8BC3	MOV EAX,EBX
00408F4F	5E	POP ESI
00408F50	5B	POP EBX
00408F51	C3	RET

Ta nhận thấy đây là hàm băm crc32. Lúc này hàm nhận vào 8 bytes ký tự cuối của serial và trả về 4 bytes ký tự.

+Xét tại: 0040A458 . 3D 0CDB67E3 CMP EAX,E367DB0C

Kết quả băm đem so sánh với 0xE367DB0C. Nếu không trùng thì báo lỗi “Invalid Serial”, ngược lại, thông báo kết quả đúng “Nice work... now write a solution.!”.

⇒ Từ việc tiến hành khảo sát và mô tả lại công việc xử lý đối với 8 ký tự cuối của serial. Ta biết được 2 ký tự cuối cùng nhất là 1D hoặc 1d. Thực hiện thuật toán brute force để vét cạn các trường hợp có thể xảy ra đối với 6 ký tự còn lại. Ta có thể đưa ra kết luận là 8 ký tự cuối của serial đúng là: 4c4f4c1d, 4c4f4c1D, 4c4f4C1d, 4c4f4C1D, 4c4F4c1d, 4c4F4c1D, 4c4F4C1d, 4c4F4C1D, 4C4f4c1d, 4C4f4C1D, 4C4f4C1D, 4C4F4c1d, 4C4F4c1D, 4C4F4C1d, 4C4F4C1D.

Vậy bài 2.2 có 16 serial key cứng là:

PSk=4c4f4c1d

PSk=4c4f4c1D

PSk=4c4f4C1d

PSk=4c4f4C1D

PSk=4c4F4c1d

PSk=4c4F4c1D

PSk=4c4F4C1d

PSk=4c4F4C1D

PSk=4C4f4c1d

PSk=4C4f4c1D

PSk=4C4f4C1d

PSk=4C4f4C1D

PSk=4C4F4c1d

PSk=4C4F4c1D

PSk=4C4F4C1d

PSk=4C4F4C1D