腾讯面试CrackMe破解——by谢俊东

程序很明显是 win32 GUI 程序，打开程序，界面很简单，输入用户名和密码然后判断成功或者失败。

一般来说程序会调用 GetDlgItemText 函数来获取用户名或密码，所以在这个函数的入口处下一个断点。输入用户名和密码点击确定后，程序果然停在GetDlgItemText处，但是不是在用户空间，不断按 Ctrl+F9，直到回到用户空间，下一个断点。

一路往下走，能看到一大堆 push，pop，pushfd 之类的，感觉都没什么实际意义，直到走到0x412c17处看到了一个 cmp 语句

```
1  00412C17    cmp     dword ptr [esp+4], 6
```

观察了一下 esp+4，是自己输入的用户名的长度，数字6猜测是用户名的最小长度。然后又往下走了很久才看到

```
1  00412C81    jl      004178AE
```

验证了自己的猜测。继续往下走又看到了一处比较语句，应该是判断用户名长度应该小于32位的。

发现程序在真正有用的代码段中间插入了很多无用的语句，联想到程序的名字里面有 tvmp 这几个字，网上查了一下只查到 vmp 是一种虚拟机技术，但是和自己这个似乎不大一样。

```
1  004066C1    39B424 08000000 cmp     dword ptr [esp+8], esi
```

继续往下走，发现这里有一处 cmp 语句，而且程序会一直在这里循环，循环的次数是用户名的长度。说明这个循环非常关键了。先把这个循环跑一遍，发现程序会在栈中另一个地方生成一个字符串，和用户名同样的长度，应该是用某种算法把用户名 transform 过来。继续 F9，程序提示密码错误。

重新开始，这次在最后试着把密码用程序生成的那串字符串替换掉然后往下跑，最后提示成功了！说明生成的字符串会和密码比较，一致的话就注册成功。

那么编写注册机的关键就在于逆向出那个算法了。用 PEID 试了一下，显示是yoda's protector，但是网上查阅发现这很可能只是其他壳的伪装，程序有 tvmp 节。没办法只好先用笨方法一步一步走。观察数据窗口和寄存器窗口的变化，可以发现程序会把用户名的第一个字符 copy 到用户名的最后，然后还引用了403020的一串字符串"0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMTVMPROTECT310"。

```
1  0040741E    8A4C04 0D       mov     cl, byte ptr [esp+eax+D]
2  00407422    8A5404 0C       mov     dl, byte ptr [esp+eax+C]
```

最关键的是这两句，在第 i 次循环中会把用户名的第 i 个字符和 第 i+1个字符取出来，在后面进行处理。一开始用了比较笨的办法，单步调试加上下内存断点。后来发现比较关键的语句一般都是 xor, or, and 之类的运算指令，就在这些地方下断点，最后逆向出了整个算法。

总体而言，这种混淆技术只是在有用的指令之间插入一些无用的指令，并用 pushfd，push eax 做寄存器的保存和恢复。或者做一些简单的指令变换，比如要改变一个寄存器的值，通过在栈和寄存器之间搬运数据，把一条指令扩充成四条指令。或者是把一些立即数拆分成多个部分相加减。不考虑把混淆去除掉的话，只关注那些 xor,or,and,cmp,jmp 的指令也是可以分析出程序的流程的。

```
#include <stdio.h>
#include <string.h>
int main(int argc, char **argv) {

    int idx;
    char strtable[100] =
"0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMTVMPROTECT310";
    char username[32] = { 0 };
    char password[50];
    char user_tran[32];
    int dl, cl;
    strcpy(username, argv[1]);
    username[strlen(username)] = username[0];

    for(int i = 0; i < strlen(username) - 1; i++) {
        dl = username[i];
        cl = username[i+1];
        cl = cl >> 2;
        dl =( dl & 0x7) << 2;
        cl = dl | cl;
        idx = cl ^ 0x15;

        user_tran[i] = strtable[idx];
    }
    printf("password should be %s\n", user_tran);
}

```