

# 看雪Q1流浪者 简单 wp

原创

菜鸟m号 于 2019-11-20 00:26:17 发布 132 收藏

分类专栏: [逆向](#) 文章标签: [CTF 逆向](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/jentle8/article/details/103154045>

版权



[逆向 专栏收录该内容](#)

1 篇文章 0 订阅

订阅专栏

资源见上传文件, 这里很多人使用ida进行静态分析, 但是这样太依赖那个F5键了, 我会用od进行一步步反调试得到keyGen.

```
00401770 | $ 55 | push ebp
00401771 | - 8BEC | mov ebp,esp
00401773 | - 83EC 44 | sub esp,0x44
00401776 | - 53 | push ebx
00401777 | - 56 | push esi
00401778 | - 57 | push edi
00401779 | - 6A 00 | push 0x0
0040177B | - 68 68354000 | push cm.00403568
00401780 | - 68 68354000 | push cm.00403568
00401785 | - 6A 00 | push 0x0
00401787 | - FF15 0032400 | call dword ptr ds:[&USER32.MessageBoxA]
0040178D | - FF15 0C30400 | call dword ptr ds:[&KERNEL32.GetCurrentProcess]
00401793 | - 8945 FC | mov [local.1],eax
00401796 | - 6A 00 | push 0x0
ebp=0019F4AC
本地调用来自 00401873
```

mfc42.#4234

Style = MB\_OK|MB\_APPLMODAL  
恭喜!  
pass!  
hOwner = NULL  
MessageBoxA  
GetCurrentProcess  
ExitCode = 0x0

<https://blog.csdn.net/jentle8>

先根据关键字字符串得到主要跳转 (我是这个办法), 然后可以看到附近时一个简单的跳转语句:

```
00401857 | > 8B55 FC | mov edx,[local.1]
0040185A | - C64415 DC 00 | mov byte ptr ss:[ebp+edx-0x24],0x0
0040185F | - 8B45 BC | mov eax,[local.17]
00401862 | - 50 | push eax
00401863 | - 8D4D DC | lea ecx,[local.9]
00401866 | - 51 | push ecx
00401867 | - E8 84070000 | call <jmp.&MSUCRT.strcmp>
0040186C | - 83C4 08 | add esp,0x8
0040186F | - 85C0 | test eax,eax
00401871 | - 75 07 | jnz short cm.0040187A
00401873 | - E8 F8FEFFFF | call cm.00401770
00401878 | - EB 05 | jmp short cm.0040187F
0040187A | > E8 31FFFFFF | call cm.00401780
```

loacl.1应该是输入的长度

s2 = 00000009 ???  
s1 = "123456789"  
strcmp

<https://blog.csdn.net/jentle8>

只是定位到跳转还不行, 往上稍微翻一下, 就可以看到一个按照一个序列取出相对应ascii的操作, 代码如下:

```
0040180A | - C745 BC C035 | mov [local.17],cm.004035C0
00401811 | - C785 50FFFFFF | mov [local.44],cm.00403580
0040181B | > 8B45 FC | mov eax,[local.1]
0040181E | - 8B4D 08 | mov ecx,[arg.1]
00401821 | - 83C81 3E | cmp dword ptr ds:[ecx+eax*4],0x3E
00401825 | - 7D 30 | jge short cm.00401857
00401827 | - 8B55 FC | mov edx,[local.1]
0040182A | - 8B45 08 | mov eax,[arg.1]
0040182D | - 83C90 00 | cmp dword ptr ds:[eax+edx*4],0x0
00401831 | - 7C 24 | jl short cm.00401857
00401833 | - 8B4D FC | mov ecx,[local.1]
00401836 | - 8B55 08 | mov edx,[arg.1]
00401839 | - 8B048A | mov eax,dword ptr ds:[edx+ecx*4]
0040183C | - 8B4D FC | mov ecx,[local.1]
```

KanXueCTF2019JustForhappy  
abcdeFGHIJKLMNOPQRSTUVWXYZ

```

0040183F - 8B95 50FFFF mov edx,[local.44]
00401845 - 8A0402 mov al,byte ptr ds:[edx+eax]
00401848 - 88440D DC mov byte ptr ss:[ebp+ecx-0x24],al
0040184C - 8B4D FC mov ecx,[local.1]
0040184F - 83C1 01 add ecx,0x1
00401852 - 894D FC mov [local.1],ecx
00401855 - EB C4 jmp short cn.0040181B
00401857 - 8B55 F8 mov edx,[local.2]
00401859 - 8355 F4 add edx,[local.3]
0040185B - 0FBF02 movsx eax,byte ptr ds:[edx]
0040185D - 85C8 test eax,eax
0040185F - 7E 01 je cm.00401861
00401861 - 8B4D F8 mov ecx,[local.2]
00401863 - 034D F4 add ecx,[local.3]
00401865 - 0FBF11 movsx edx,byte ptr ds:[ecx]
00401867 - 83FA 39 cmp edx,0x39
00401869 - 7F 23 jg short cn.0040194A
0040186B - 8B45 F8 mov eax,[local.2]
0040186D - 0345 F4 add eax,[local.3]
0040186F - 0FBF08 movsx ecx,byte ptr ds:[eax]
00401871 - 83F9 30 cmp ecx,0x30
00401873 - 7C 15 jl short cn.0040194A
00401875 - 8B55 F8 mov edx,[local.2]
00401877 - 8355 F4 add edx,[local.3]
00401879 - 0FBF02 movsx eax,byte ptr ds:[edx]
0040187B - 83E8 30 sub eax,0x30
0040187D - 8B4D F4 mov ecx,[local.3]
0040187F - 89448D 8C mov dword ptr ss:[ebp+ecx*4-0x74],eax
00401881 - EB 67 jmp short cn.0040181B
00401883 - 8B55 F8 mov edx,[local.2]
00401885 - 8355 F4 add edx,[local.3]
00401887 - 0FBF02 movsx eax,byte ptr ds:[edx]
00401889 - 83F8 7A cmp eax,0x7A
0040188B - 7E 23 jn short cn.0040197B

```

我一开始在这停留了比较久的时间，是因为我看到栈控件已经有输入的字符串，以为之前的是某一个函数得到用户输入的字符串就完事了。后来调试的时候不对，单纯的依据输入的字符串得到的下标值是取不到后面的大写字母和数字的。然后根据栈回溯的方法，得到输入的处理函数。

```

00401908 > 8B55 F8 mov edx,[local.2]
0040190B - 8355 F4 add edx,[local.3]
0040190E - 0FBF02 movsx eax,byte ptr ds:[edx]
00401911 - 85C8 test eax,eax
00401913 - 7E 01 je cm.004019BF
00401919 - 8B4D F8 mov ecx,[local.2]
0040191C - 034D F4 add ecx,[local.3]
0040191F - 0FBF11 movsx edx,byte ptr ds:[ecx]
00401922 - 83FA 39 cmp edx,0x39
00401925 - 7F 23 jg short cn.0040194A
00401927 - 8B45 F8 mov eax,[local.2]
0040192A - 0345 F4 add eax,[local.3]
0040192D - 0FBF08 movsx ecx,byte ptr ds:[eax]
00401930 - 83F9 30 cmp ecx,0x30
00401933 - 7C 15 jl short cn.0040194A
00401935 - 8B55 F8 mov edx,[local.2]
00401938 - 8355 F4 add edx,[local.3]
0040193B - 0FBF02 movsx eax,byte ptr ds:[edx]
0040193E - 83E8 30 sub eax,0x30
00401941 - 8B4D F4 mov ecx,[local.3]
00401944 - 89448D 8C mov dword ptr ss:[ebp+ecx*4-0x74],eax
00401948 - EB 67 jmp short cn.0040181B
0040194A > 8B55 F8 mov edx,[local.2]
0040194D - 8355 F4 add edx,[local.3]
00401950 - 0FBF02 movsx eax,byte ptr ds:[edx]
00401953 - 83F8 7A cmp eax,0x7A
00401956 - 7E 23 jn short cn.0040197B

```

这个过程比较简单，就是如果是数字就减去0x30，是大写字母就0x1D，是小写字母就减去57  
根据索引得到下表，和字符串进行比较。

```

keyTab[]='abcdefghiABCDEFGHIJKLMNjklmn0123456789opqrstuvwxyzOPQRSTUVWXYZ'
key[index]='KanXueCTF2019JustForhappy'

```

keyGen如下（太晚了这是别的大佬写的）：

```

string strs = "abcdefghiABCDEFGHIJKLMNjklmn0123456789opqrstuvwxyzOPQRSTUVWXYZ";
char Kanxue[] = "KanXueCTF2019JustForhappy";
size_t snIndex = 0;
for (int i = 0; i < 25; ++i)
{
    // 获取字符在字符串中下标
    snIndex = strs.find(Kanxue[i]);

    // 通过随机匹配
    if (snIndex + 0x57 >= 'a' && snIndex + 0x57 <= 'z')
        printf("%c", snIndex + 0x57);

    if (snIndex + 0x30 >= '0' && snIndex + 0x30 <= '9')
        printf("%c", snIndex + 0x30);

    if (snIndex + 0x1D >= 'A' && snIndex + 0x1D <= 'Z')
        printf("%c", snIndex + 0x1D);
}

```





[创作打卡挑战赛](#) >

[赢取流量/现金/CSDN周边激励大奖](#)